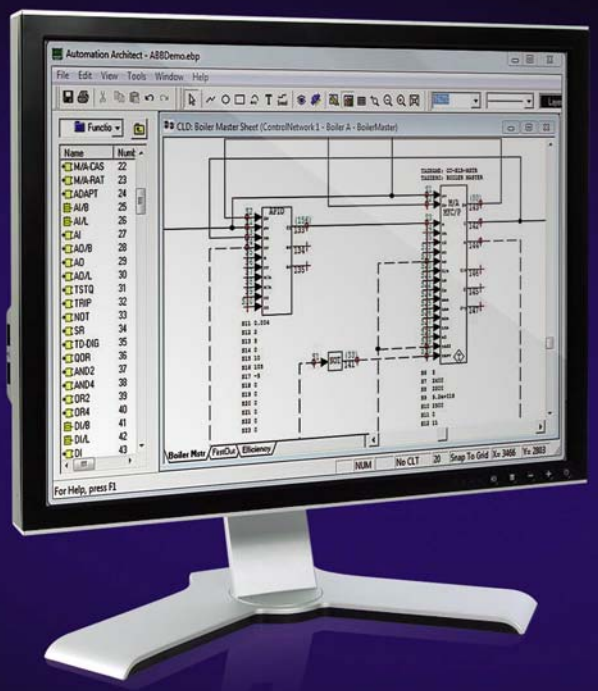


Symphony Plus - Harmony

S+ Engineering: Composer Harmony
Function Code Application Manual
Volume 1 - Function Codes 1-174

Symphony Plus

Power and productivity
for a better world™

NOTICE

This document contains information about one or more ABB products and may include a description of or a reference to one or more standards that may be generally relevant to the ABB products. The presence of any such description of a standard or reference to a standard is not a representation that all of the ABB products referenced in this document support all of the features of the described or referenced standard. In order to determine the specific features supported by a particular ABB product, the reader should consult the product specifications for the particular ABB product.
ABB may have one or more patents or pending patent applications protecting the intellectual property in the ABB products described in this document.
The information in this document is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this document.
In no event shall ABB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.
This document and parts thereof must not be reproduced or copied without written permission from ABB, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.
The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license. This product meets the requirements specified in EMC Directive 2004/108/EC and in Low Voltage Directive 2006/95/EC.

TRADEMARKS

Symphony is a registered or pending trademark of ABB S.p.A.
All rights to copyrights, registered trademarks, and trademarks reside with their respective owners.
Copyright © 2011-2013 ABB.
All rights reserved.

Release: July 2013
Document number: 2VAA000844R0001 Vol. 1 Rev. D

TABLE OF CONTENTS

1. Function Generator 1-1

1.1 Explanation 1-2

1.1.1 High and Low Limits 1-3

1.2 Applications 1-3

2. Manual Set Constant (Signal Generator)2-1

2.1 Applications 2-1

3. Lead/Lag 3-1

3.1 Explanation 3-1

3.1.1 Lag Function 3-1

3.1.2 Lead Function 3-2

3.2 Applications 3-2

4. Pulse Positioner 4-1

4.1 Explanation 4-1

4.1.1 Cycle Time 4-2

4.1.2 Stroke Rate 4-2

4.2 Applications 4-3

5. Pulse Rate 5-1

5.1 Explanation 5-1

5.2 Applications 5-2

6. High/LowLimiter 6-1

6.1 Applications 6-1

7. Square Root 7-1

7.1 Applications 7-1

8. Rate Limiter 8-1

8.1 Explanation 8-1

8.2 Applications 8-1

9. Analog Transfer 9-1

9.1 Explanation 9-1

9.2 Applications 9-2

10. High Select 10-1

10.1 Applications 10-1

11. Low Select 11-1

11.1 Applications 11-1

12. High/Low Compare 12-1
12.1 Applications 12-1

13. Integer Transfer 13-1
13.1 Applications 13-1

14. Summer (4-Input) 14-1

15. Summer (2-Input) 15-1

16. Multiply 16-1

17. Divide 17-1

18. PID Error Input 18-1
18.1 Explanation 18-1
18.2 Applications 18-3

19. PID (PV and SP) 19-1
19.1 Explanation 19-1
19.2 Examples 19-2

24. Adapt 24-1
24.1 Applications 24-1

25. Analog Input (Periodic Sample) 25-1
25.1 Applications 25-2

26. Analog Input/Loop 26-1
26.1 Applications 26-2

30. Analog Exception Report 30-1
30.1 Explanation 30-1
30.1.1 Exception Reports 30-1
30.1.2 Alarm Reports 30-2

31. Test Quality 31-1
31.1 Applications 31-1

32. Trip 32-1

33. Not 33-1

34. Memory 34-1
34.1 Applications 34-1

35. Timer 35-1
35.1 Explanation 35-1

36. Qualified OR (8-Input) 36-1
36.1 Applications 36-1

37. AND (2-Input) 37-1

38. AND (4-Input) 38-1

39. OR (2-Input) 39-1

40. OR (4-Input) 40-1

41. Digital Input (Periodic Sample) 41-1
41.1 Examples 41-2

42. Digital Input/Loop 42-1
42.1 Examples 42-2

45. Digital Exception Report 45-1
45.1 Explanation 45-1
45.2 Examples 45-2
48.3 Explanation 48-2
48.3.1 Exception Reports 48-2
48.3.2 Alarm Reports 48-2

50. Manual Set Switch 50-1

51. Manual Set Constant 51-1

52. Manual Set Integer 52-1
52.1 Examples 52-1

55. Hydraulic Servo 55-1
55.1 Explanation 55-3
55.1.1 Outputs 55-3
55.1.2 Specifications 55-6

57. Node Statistics Block 57-1
57.1 Explanations 57-3
57.1.1 Specifications 57-3

58. Time Delay (Analog) 58-1
58.1 Explanation 58-1
58.1.1 Specifications 58-1

58.2 Applications **58-2**
58.2.1 Fixed Time Delay 58-2
58.2.2 Variable Time Delay 58-2
58.2.3 System Modeling 58-3

59. Digital Transfer **59-1**
59.1 Applications **59-1**

61. Blink **61-1**
61.1 Applications **61-1**

62. Remote Control Memory **62-1**
62.1 Explanations **62-1**
62.1.1 Specifications 62-1
62.1.2 Control Station Control 62-2
62.2 Applications **62-3**

63. Analog Input List (Periodic Sample)63-1
63.1 Explanation **63-2**
63.1.1 Specifications 63-2

64. Digital Input List (Periodic Sample)64-1
64.1 Explanation **64-2**
64.1.1 Specifications 64-2

65. Digital Sum With Gain **65-1**
65.1 Applications **65-1**

66. Analog Trend **66-1**
66.1 Explanations **66-1**
66.1.1 Specifications 66-1
66.2 Applications **66-2**

67. Digital Exception Report with Alarm Deadband **67-1**
67.1 Explanation **67-1**
67.1.1 Exception Reports 67-1
67.2 Examples **67-2**

68. Remote Manual Set Constant **68-1**
68.1 Explanation **68-1**

69. Test Alarm **69-1**

79. Control Interface Slave 79-1
79.1 Explanation 79-2
79.1.1 Outputs 79-2
79.1.2 Specifications 79-2

80. Control Station 80-1
80.1 Explanation 80-3
80.1.1 Outputs 80-3
80.1.2 Specifications 80-4
80.2 Applications 80-8

81. Executive 81-1
81.1 Explanation 81-1
81.1.1 Outputs 81-1

82. Segment Control 82-1
82.1 Explanation 82-2
82.1.1 Specifications 82-2
82.1.2 Outputs 82-5

83. Digital Output Group 83-1
83.1 Explanation 83-2
83.1.1 Specifications 83-2

84. Digital Input Group 84-1
84.1 Explanation 84-1
84.1.1 Specifications 84-1

85. Up/Down Counter 85-1
85.1 Explanation 85-1
85.1.1 Specifications 85-1
85.2 Reset 85-2
85.3 Normal 85-2
85.4 Alarm 85-3
85.5 Applications 85-3

86. Elapsed Timer 86-1
86.1 Explanation 86-1
86.1.1 Specifications 86-1
86.1.2 Outputs 86-2
86.2 Applications 86-3

87. Digital Logic Station Interface 87-1

88. Digital Logic Station 88-1
88.1 Explanation 88-2
88.1.1 Specifications 88-2
88.1.2 Outputs 88-2
88.2 Applications 88-3

89. Last Block 89-1

90. Extended Executive 90-1
90.1 Explanation 90-2
90.1.1 Specifications 90-2
90.1.2 Outputs 90-3
90.2 Example 90-4
90.2.1 Function Block Configuration Required for Time-Synchronization 90-4

91. BASIC Configuration (BRC-100/200)91-1
91.1 Explanation 91-1
91.1.1 Specifications 91-1
91.1.2 Outputs 91-2
91.2 Application 91-2

92. Invoke BASIC 92-1
92.1 Explanation 92-1
92.1.1 Example 1 92-1
92.1.2 Example 2 92-2

93. BASIC Real Output 93-1
93.1 Application 93-1

94. BASIC Boolean Output 94-1

95. Module Status Monitor 95-1
95.1 Explanation 95-2
95.1.1 Specifications 95-2

96. Redundant Analog Input 96-1
96.1 Explanation 96-1
96.1.1 Specifications 96-1
96.2 Logic Flow 96-2
96.3 Applications 96-2

97. Redundant Digital Input 97-1
97.1 Applications 97-1

98. Slave Select 98-1
98.1 Specifications 98-1
98.2 Applications 98-3

99. Sequence of Events Log 99-1
99.1 Explanation 99-1
99.1.1 Specifications 99-1
99.1.2 Outputs 99-2

100.Digital Output Readback Check 100-1
100.1Explanation 100-2

101.Exclusive OR 101-1

102.Pulse Input/Period 102-1
102.1Explanation 102-1
102.1.1Specifications 102-2
102.1.2Outputs 102-2

103.Pulse Input/Frequency 103-1
103.1Explanation 103-1
103.1.1Specifications 103-1
103.1.2Outputs 103-2

104.Pulse Input/Totalization 104-1
104.1Explanation 104-1
104.1.1Specifications 104-2
104.1.2Outputs 104-2

109.Pulse Input/Duration 109-1
109.1Explanation 109-1
109.1.1Specifications 109-1
109.1.2Outputs 109-2

110.Rung (5-Input) 110-1
110.1Explanation 110-1
110.1.1Specifications 110-2
110.1.2Outputs 110-2
110.2Applications 110-2
110.2.1Specifications 110-3

111.Rung (10-Input) 111-1
111.1Explanation 111-2
111.1.1Specifications 111-2

111.1.2Outputs 111-3
111.2Applications 111-3

112.Rung (20-Input) 112-1
112.1Explanation 112-2
112.1.1Specifications 112-3
112.1.2Outputs 112-3
112.2Applications 112-3

114.BCD Input 114-1
114.1Explanation 114-1
114.1.1Specifications 114-2
114.1.2Outputs 114-2
114.2Example 114-2

115.BCD Output 115-1
115.1Explanation 115-1
115.1.1Specifications 115-2
115.1.2Output 115-3
115.2Example 115-3

116.Jump/Master Control Relay 116-1
116.1Explanation 116-1
116.1.1Specifications 116-1
116.1.2Output 116-2
116.2Applications 116-2

117.Boolean Recipe Table 117-1
117.1Explanation 117-2
117.1.1Specifications 117-2
117.1.2Output 117-2
117.2Applications 117-2

118.Real Recipe Table 118-1
118.1Explanation 118-2
118.1.1Specifications 118-2
118.1.2Output 118-2
118.2Applications 118-2

119.Boolean Signal Multiplexer 119-1
119.1Explanation 119-1
119.1.1Specifications 119-1
119.1.2Output 119-2
119.2Applications 119-2

120.Real Signal Multiplexer 120-1
120.1Explanation 120-1
120.1.1Specifications 120-1
120.1.2Output 120-2

121.Analog Input/Cnet 121-1

122.Digital Input/Cnet 122-1

123.Device Driver 123-1
123.1Explanation 123-1
123.1.1Specifications 123-2
123.1.2Outputs 123-4
123.2Applications 123-4
123.3Distributed Recipe Handling 123-13
123.3.1Storing Recipe Data 123-13
123.4Recipe Handling for a Fixed Batch Sequence 123-14

124.Sequence Monitor 124-1
124.1Explanation 124-2
124.1.1Specifications 124-3
124.1.2Outputs 124-5
124.2Applications 124-5

125.Device Monitor 125-1
125.1Explanation 125-1
125.1.1Specifications 125-2
125.1.2Output 125-2
125.2Applications 125-3

126.Real Signal Demultiplexer 126-1
126.1Explanation 126-1
126.1.1Select Mode 126-1
126.1.2Integer Mode 126-1
126.1.3BCD Mode 126-2
126.2Applications 126-3

128.Slave Default Definition 128-1
128.1Explanation 128-2

129.Multistate Device Driver 129-1
129.1Explanation 129-2
129.1.1Specifications 129-4
129.1.2Outputs 129-7

129.2Applications....129-8

132.Analog Input/Slave 132-1
132.1Explanation....132-2
132.1.1Specifications....132-2
132.2Application....132-4

133.Smart Field Device Definition 133-1
133.1Explanation....133-1
133.1.1Specifications....133-1
133.2Application....133-6
133.3Type AVS Positioner Application....133-7
133.4Flowmeter Application....133-8

134.Multi-Sequence Monitor 134-1
134.1Explanation....134-3
134.1.1Specifications....134-4
134.1.2Outputs....134-6
134.2Applications....134-6

135.Sequence Manager 135-1
135.1Explanation....135-2
135.1.1Specifications....135-3
135.1.2Outputs....135-4
135.2Applications....135-4

136.Remote Motor Control 136-1
136.1Explanation....136-2
136.1.1Normal Operation of the Remote Motor Control....136-2
136.1.2Abnormal Operation....136-2
136.1.3Pulsed Outputs....136-2
136.1.4Specifications....136-3
136.1.5Outputs....136-3

137.C and BASIC Program Real Output With Quality 137-1

138.C or BASIC Program Boolean Output With Quality.... 138-1

139.Passive Station Interface 139-1
139.1Explanation....139-2
139.1.1Specifications....139-2
139.1.2Outputs....139-3
139.2Application....139-3

140.Restore 140-1

140.1Specifications **140-2**
140.2Module Memory Utilization **140-3**
140.3Memory Usage Equations **140-6**
140.4Applications **140-6**

141.Sequence Master **141-1**
141.1Explanation **141-2**
141.1.1Specifications 141-2

142.Sequence Slave **142-1**

143.Invoke C **143-1**
143.1Applications **143-2**

144.C Allocation **144-1**

145.Frequency Counter/Slave **145-1**

146.Remote I/O Interface **146-1**
146.1Explanation **146-4**
146.2RMP/RSP Memory Usage Calculation (Expander Bus) **146-5**
146.3Applications **146-6**

147.Remote I/O Definition **147-1**
147.1Explanation **147-2**
147.1.1Specifications 147-2
147.2RMP/RSP Memory Usage Calculation (Expander Bus) **147-3**
147.3Applications **147-4**

148.Batch Sequence **148-1**
148.1Explanation **148-2**
148.1.1Specifications 148-2
148.1.2Outputs 148-3
148.1.3Run-Time Fault Code Explanation 148-3
148.2Application **148-6**

149.Analog Output/Slave **149-1**
149.1Explanation **149-2**
149.1.1Specifications 149-2
149.2Applications **149-3**

150.Hydraulic Servo Slave **150-1**
150.1Explanation **150-2**

151.Text Selector **151-1**

151.1Specifications **151-2**

152.Model Parameter Estimator **152-1**
152.1Explanation **152-2**
152.2Specifications **152-2**
152.3Applications **152-2**

153.ISC Parameter Converter **153-1**
153.1Explanation **153-2**
153.2Applications **153-4**

154.Adaptive Parameter Scheduler **154-1**
154.1Explanation **154-1**
154.1.1Specifications 154-2
154.2Applications **154-2**

155.Regression **155-1**
155.1Explanation **155-2**
155.1.1Specifications 155-4
155.1.2Outputs 155-6
155.2Applications **155-7**
155.2.1Regression Block Application Considerations 155-8

156.Advanced PID Controller **156-1**
156.1Explanation **156-3**
156.1.1Specifications 156-3
156.1.2Outputs 156-5
156.2Classical PID Controller **156-7**
156.3Noninteracting PID Controller **156-9**
156.4Manual Reset PID Controller **156-10**
156.5Applications **156-12**

157.General Digital Controller **157-1**
157.1Explanation **157-2**
157.1.1Specifications 157-2
157.1.2Output 157-3
157.2Application **157-3**

160.Inferential Smith Controller **160-1**
160.1Explanation **160-1**
160.1.1Specifications 160-3
160.2ISC Structure **160-4**
160.3Applications **160-5**

161.Sequence Generator 161-1
161.1Explanation 161-2
161.1.1Specifications 161-5
161.1.2Outputs 161-7
161.2Applications 161-7

162.Digital Segment Buffer 162-1
162.1Explanation 162-1

163.Analog Segment Buffer 163-1
163.1Explanation 163-1

165.Moving Average 165-1
165.1Explanation 165-1
165.1.1Specifications 165-2
165.1.2Output 165-2

166.Integrator 166-1
166.1Explanation 166-1
166.1.1Specifications 166-2
166.1.2Outputs 166-2

167.Polynomial 167-1

168.Interpolator 168-1
168.1Explanation 168-2
168.2Applications 168-3

169.Matrix Addition 169-1
169.1Explanation 169-2

170.Matrix Multiplication 170-1
170.1Explanation 170-2

171.Trigonometric 171-1

172.Exponential 172-1

173.Power 173-1

174.Logarithm 174-1

A. List of Function Codes A-1
A.1 Introduction A-1
A.2 Cross Reference - Numerical A-2
A.3 Cross Reference - Alphabetical A-5

A.4 Cross Reference - Categorization A-8

B. Harmony Bridge Controller (BRC-300/400/410) and HPG800 ..... B-1
B.1 Memory Utilization and Execution Times B-1
B.2 Memory Usage Equations B-7
B.3 Function Blocks - BRC-300/400/410 and HPG800 ..... B-9
B.4 Module Status Information - BRC-300/400/410 and HPG800 ..... B-10

C. Harmony Bridge Controller (BRC-100/200) C-1
C.1 Memory Utilization and Execution Times C-1
C.2 Memory Usage Equations C-7
C.3 Function Blocks - BRC-100/200 ..... C-8
C.4 Module Status Information - BRC-100/200 ..... C-9

D. Harmony Area Controller (HAC) D-1
D.1 Memory Utilization and Execution Times D-1
D.2 Memory Usage Equations D-7
D.3 Function Blocks - HAC D-8
D.4 Module Status Information - HAC D-9

E. Multi-Function Processors (IMMFP11/12) E-1
E.1 Memory Utilization and Execution Times E-1
E.2 Memory Usage Equations E-6
E.3 Function Blocks - IMMFP11/12 ..... E-7
E.4 Module Status Information - IMMFP11/12 ..... E-8

F. Sequence of Events Monitor (INSEM01 and INSEM11/SPSEM11) F-1
F.1 Memory Utilization F-1
F.2 Module Status Information F-1

G. Communication Modules G-1
G.1 Introduction G-1
G.2 IINICT03/03A/13 and INICT12 Cnet to Computer Transfer Modules G-1
G.3 INIET800 and SPIET800 Cnet to Computer Transfer Modules G-3
G.4 INIIT03 and INIIT13/SPIIT13 Local Transfer Module G-5
G.5 INNPM11/22, INNPM12 and SPNPM22 Network Processing Modules G-6
G.5.1 Cnet Mode G-7
G.5.2 Plant Loop Mode G-8
G.6 PNI800 Plant Network Interface 800 Module G-9
G.7 CP800 Communications Processor Module (HPC800) G-11
G.8 SPIPT800 INFI-NET to PN800 Transfer Module G-13

H. Point Quality Definition H-1
H.1 General Description H-1

H.2 Individual Analog Inputs H-1
H.3 Group Analog Inputs H-1
H.4 Group Analog Outputs H-1
H.5 Digital I/O H-1
H.6 Peer-to-Peer and Module Bus I/O H-1

I. Console Engineering Unit Descriptions I-1
I.1 Engineering Unit Descriptions Tables I-1

J. Symphony Plus HC800 Control Processor J-1
J.1 Memory Utilization and Execution Times J-1
J.2 Memory Usage Equations J-7
J.3 Function Blocks - HC800 ..... J-8
J.4 Module Status Information - HC800 ..... J-9

LIST OF TABLES

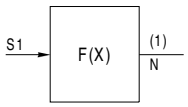
Table 24-1 Conversions Performed by the Adapt Block 24-1
Table 34-1 Power Up or Controller Reset Truth Table 34-1
Table 34-2 Normal Operation Truth Table 34-1
Table 37-3 2-Input AND Truth Table 37-1
Table 38-4 4-Input AND Truth Table 38-1
Table 39-5 2-Input OR Truth Table 39-1
Table 40-6 4-Input OR Truth Table 40-1
Table 55-7 Module Status Bit Map 55-4
Table 55-8 LVDT Status 55-5
Table 55-9 Position Feedback Options 55-7
Table 61-10 Truth Table for Output Values 61-1
Table 62-1 RCM Input to Output Relationship 62-1
Table 69-2 Outputs from Test Alarm Block 69-1
Table 80-3 Track Behavior of the Station Block 80-4
Table 82-4 Total Segment Checkpoint Utilization 82-3
Table 90-1 Available PC View Tag Names 90-5
Table 101-1 Exclusive OR Truth Table 101-1
Table 114-1 BCD to Real Conversion Format 114-1
Table 115-1 Module Output Capacity 115-1
Table 123-1 Override Permissive/State Specifications 123-2
Table 123-2 Recipe Contents 123-14
Table 126-1 Integer Mode Input to Output Relationship 126-2
Table 126-2 Each Group of Outputs Represents Two Real Digits 126-2
Table 126-3 Sample Outputs of a Real Signal Demultiplexer Block 126-3
Table 129-1 Truth Table for Selection of Output Masks in Auto Mode 129-3
Table 133-2 Specification S3 Engineering Units 133-3
Table 140-3 Save Action 140-2
Table 140-4 BRC-100/200/300/400 and IMMFP11/12 Additional NVRAM and Checkpoint Utilization Byte Size 140-3
Table 140-5 HAC Additional NVRAM and Checkpoint Utilization Byte Size 140-4
Table 141-6 Output Descriptions 141-3
Table 146-1 Block Inputs to be Reserved 146-2
Table 146-2 Example S4 Calculation 146-3
Table 147-1 Function Blocks Supported by Remote I/O Definition Block (Expander Bus) .. 147-3
Table 147-2 Function Blocks Supported by Remote I/O Definition Block (HNET) 147-3
Table 147-3 RMP Shared Memory Usage 147-3
Table 148-1 BSEQ Run-Time Fault Codes 148-3
Table 151-1 HSI Color Codes 151-2
Table 161-2 Definition of Step Inputs for Sequence Generator Blocks 161-5

Table 168-3 Interpolator Block Determines Steam Properties 168-3
Table A-1 Numerical Listing A-2
Table A-2 Alphabetical Listing A-5
Table A-3 Adapt A-8
Table A-4 Advanced Functions A-8
Table A-5 BASIC Language A-8
Table A-6 Batch Functions A-8
Table A-7 Batch Language A-9
Table A-8 C Language A-9
Table A-9 Communications A-9
Table A-10 Computing A-9
Table A-11 Controlway/Module Bus/CW800 and Peer-to-Peer Network I/O A-10
Table A-12 Control Function Blocks A-10
Table A-13 Exception Report A-10
Table A-14 Executive A-10
Table A-15 Factory Instrumentation Protocol A-11
Table A-16 Field I/O A-11
Table A-17 Harmony I/O and Foreign Devices (HART & PROFIBUS) A-11
Table A-18 Harmony Bridge Controller Executive A-12
Table A-19 Ladder Logic A-12
Table A-20 Logic A-12
Table A-21 I/O Expander Bus A-12
Table A-22 Restore A-13
Table A-23 Sequence Command A-13
Table A-24 Sequence of Events A-13
Table A-25 Signal Select A-13
Table A-26 Signal Status A-14
Table A-27 Station A-14
Table A-28 Text Selector A-14
Table A-29 Trend A-14
Table A-30 Trip A-14
Table A-31 User Defined Function A-14
Table B-1 BRC-300/400/410 and HPG800 Memory Utilization and Execution Times B-1
Table B-2 FC221 Execution Times B-7
Table B-3 BRC-300/400/410 and HPG800 Modules B-9
Table B-4 Bit Description - BRC-300/400/410 and HPG800 ..... B-10
Table B-5 Byte Description - BRC-300/400/410 and HPG800 ..... B-10
Table C-1 BRC-100/200 Memory Utilization and Execution Times C-1
Table C-2 FC221 Execution Times C-7
Table C-3 BRC-100/200 Module C-8
Table C-4 Bit Description - BRC-100/200 ..... C-9
Table C-5 Byte Description - BRC-100/200 ..... C-10

Table F-1 Available Module Configuration Memory F-1
Table F-2 Module Memory Utilization for Function Codes F-1
Table F-3 Bit Description - INSEM01 and SEM11 ..... F-1
Table F-4 Byte Description - INSEM01 ..... F-2
Table J-1 HC800 Memory Utilization and Execution Times J-1
Table J-2 HC800 Module J-8
Table J-3 Byte and Bit Description - HC800 ..... J-9
Table J-4 Byte Description - HC800 ..... J-10

1. Function Generator

This function approximates a nonlinear output to input relationship. The input range is divided into five sections and a linear input to output relationship is set up for each of the five sections. This function then computes an output that is related to the input according to the five linear relationships.



NOTES:

1. When function code 1 is utilized as a shaping algorithm for function code 222 (analog in/channel), its tunable specifications are not adaptable.

2. When function code 1 is used as a shaping algorithm, it can not at the same time also be used as a logic function because the block output will not respond to the specification S1 input. Function code 1 should not be referenced by function blocks other than function code 177 or function code 222 blocks utilizing it as a shaping algorithm.

3. Multiple instances and combinations of function code 177 and 222 function blocks can utilize the same function code 1 function block as a shaping algorithm. The function code 1 shaping algorithm function block is not required to be in the same segment as the function code 177 or function code 222 blocks.

Outputs

Blk	Type	Description
N	R	Output value of function

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input
S2	Y	9.2 E18	R	Full	Input coordinate
S3	Y	0.000	R	Full	Output coordinate for S2
S4	Y	9.2 E18	R	Full	Input coordinate
S5	Y	0.000	R	Full	Output coordinate for S4
S6	Y	9.2 E18	R	Full	Input coordinate
S7	Y	0.000	R	Full	Output coordinate for S6
S8	Y	9.2 E18	R	Full	Input coordinate
S9	Y	0.000	R	Full	Output coordinate for S8
S10	Y	9.2 E18	R	Full	Input coordinate
S11	Y	0.000	R	Full	Output coordinate for S10
S12	Y	9.2 E18	R	Full	Input coordinate
S13	Y	0.000	R	Full	Output coordinate for S12

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

1.1 Explanation

To set up this function, first determine what the output should be for a given range of input and draw a graph to show this relationship. Divide the graphed relationship into five sections, preferably into sections where straight lines can closely approximate the graph as shown in Figure 1-1.

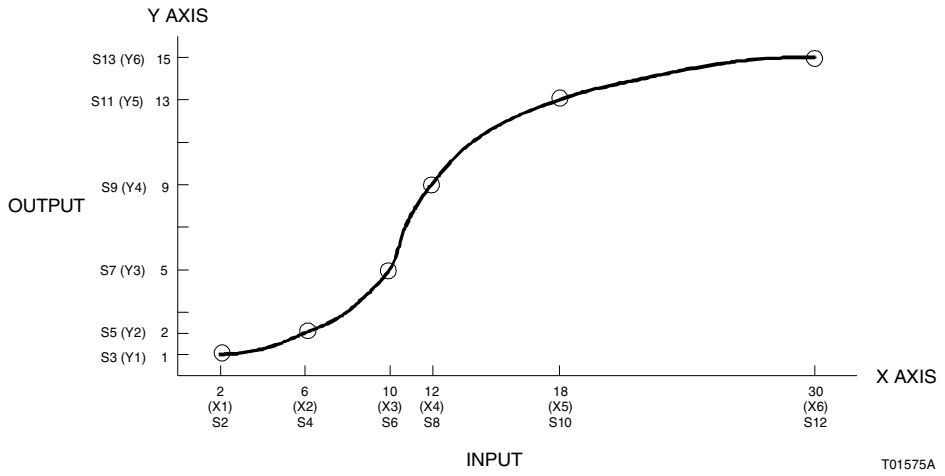


Figure 1-1 Graph of Input Versus Output

The coordinates of the end points of the sections are used as entries for S2 through S13. The even-numbered specifications are the X-axis coordinates and the odd-numbered are the Y-axis coordinates. Consequently, when the X-axis input value is at S2, the output will be the value of S3 as shown in the graph. This divides the graph into five linear (straight-line) sections, with each section having its own particular slope as shown in Figure 1-1.

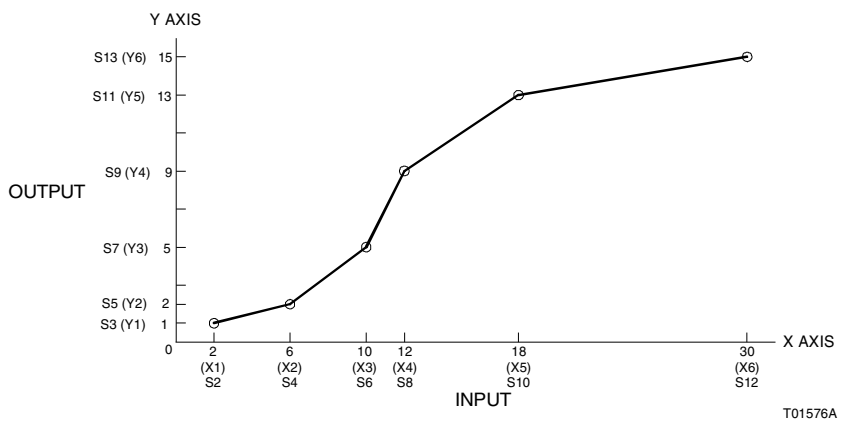


Figure 1-1 Sectioned Input-Output Graph

If the input value is between two X-axis points, the output will be determined by the equation:

$$\textit{Block Output} = (Y_{n-1}) + \frac{(Y_n - Y_{n-1})}{(X_n - X_{n-1})} \times (X - X_{n-1})$$

where:

X	=	Present input value.
X_n	=	X-axis specification point just to the right of the present input value.
X_{n-1}	=	X-axis specification point just to the left of the present input value.
Y_n	=	Y-axis coordinate that corresponds to X_n.
Y_{n-1}	=	Y-axis coordinate that corresponds to X_{n-1}.

$$\frac{(Y_n \equiv Y_{n-1})}{(X_n - X_{n-1})}$$ Slope of the particular graph segment between (X_n,Y_n) and (X_{n-1},Y_{n-1}). This is the unit output change per unit input change.

$X - X_{n-1}$ = Amount that the input is above the next lower specification point.

For example, suppose the graph shown in Figure 1-1 is a graph of desired output values for input values. These values may represent any engineering units.

First, the graph is divided into five sections as shown in Figure 1-1. The coordinates of the end points of these segments are then entered into the module.

Suppose the input <S1> to the function block represented by Figure 1-1 is six units. This corresponds to point S4. Therefore, the output will be two units (S5). If the input is ten units (which corresponds to S6), the output will be five units and so on. If the input is between six units and ten units (for example, seven units), the output is determined according to the function equation. The values for the equation become:

X_{input} = 7

X_n = S6 = 10

X_{n-1} = S4 = 6

Y_n = S7 = 5

Y_{n-1} = S5 = 2

The equation becomes:

$$\begin{aligned} Output &= S5 + \frac{(S7 - S5)}{(S6 - S4)} \times (7 - S4) \\ &= 2 + \frac{5-2}{10-6} \times (7-6) \\ &= 2.75 \end{aligned}$$

1.1.1 High and Low Limits

If the input <S1> goes higher than the S12 value, the output will remain at the S13 value for the high limit. If the input goes below the S2 value, the input will remain at the S3 value for the low limit.

1.2 Applications

Five possible applications of function generators are illustrated in Figures 1-1, 1-1, 1-1, 1-1 and 1-1. Figures 1-1 and 1-1 illustrate the use of multiple function generators to achieve good resolution when representing a complex function.

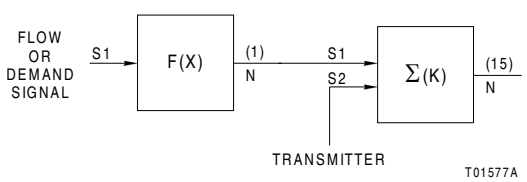


Figure 1-1 Programming Set Point or Bias

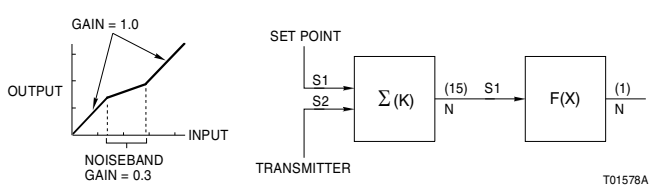


Figure 1-1 Noise Filter

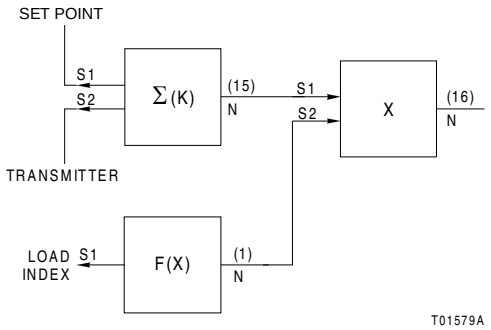
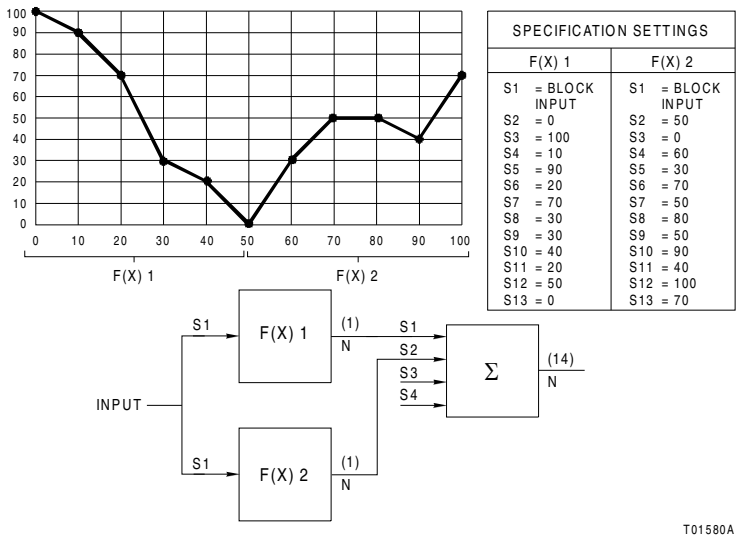
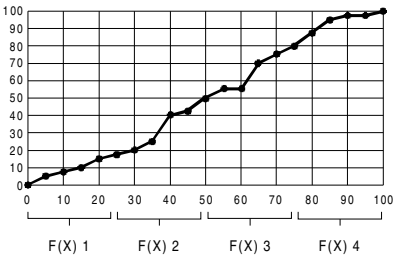


Figure 1-1 Obtain an Adaptive Gain for a Nonlinear Process



SPECIFICATION SETTINGS	
F(X) 1	F(X) 2
S1 = BLOCK INPUT	S1 = BLOCK INPUT
S2 = 0	S2 = 50
S3 = 100	S3 = 0
S4 = 10	S4 = 60
S5 = 90	S5 = 30
S6 = 20	S6 = 70
S7 = 70	S7 = 50
S8 = 30	S8 = 80
S9 = 30	S9 = 50
S10 = 40	S10 = 90
S11 = 20	S11 = 40
S12 = 50	S12 = 100
S13 = 0	S13 = 70

Figure 1-1 Greater Curve Resolution - Two Function Generators



SPECIFICATION SETTINGS			
F(X) 1	F(X) 2	F(X) 3	F(X) 4
S1 = BLOCK INPUT	S1 = BLOCK INPUT	S1 = BLOCK INPUT	S1 = BLOCK INPUT
S2 = 0	S2 = 25	S2 = 50	S2 = 75
S3 = 0	S3 = 18	S3 = 48	S3 = 80
S4 = 5	S4 = 30	S4 = 55	S4 = 80
S5 = 5	S5 = 20	S5 = 55	S5 = 87
S6 = 10	S6 = 35	S6 = 60	S6 = 85
S7 = 7	S7 = 25	S7 = 55	S7 = 95
S8 = 15	S8 = 40	S8 = 65	S8 = 90
S9 = 10	S9 = 40	S9 = 70	S9 = 97
S10 = 20	S10 = 45	S10 = 70	S10 = 95
S11 = 15	S11 = 42	S11 = 75	S11 = 98
S12 = 25	S12 = 50	S12 = 75	S12 = 100
S13 = 18	S13 = 48	S13 = 80	S13 = 100

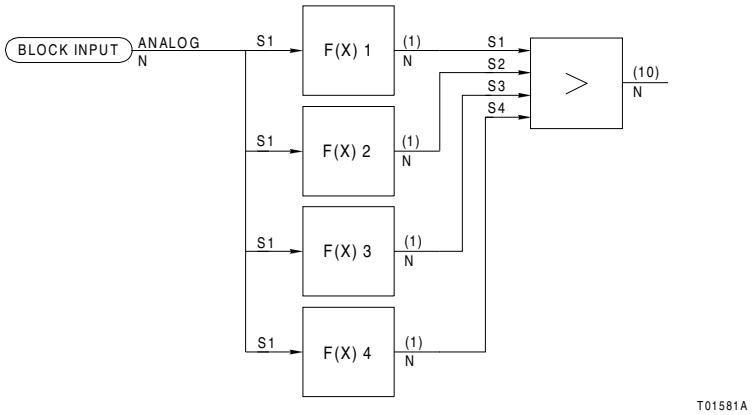
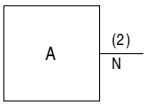


Figure 1-1 Greater Curve Resolution - Four Function Generators

2. Manual Set Constant (Signal Generator)

The output of the manual set constant is an analog signal developed within the function that is equal to <S1>. This function provides a tunable output value in engineering units.



Outputs

Blk	Type	Description
N	R	User selected constant

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0.000	R	Full	Output value in engineering units

2.1 Applications

Scaler

Figure 2-1 illustrates how to use the manual set constant as a scaler. In the example, the transmitter has a range of 200 to 700 pounds per square inch. The range is scaled up 200 pounds per square inch by setting the manual set constant to 200. The summer adds <S1> and <S2> to provide an output range of 400 to 900 pounds per square inch.

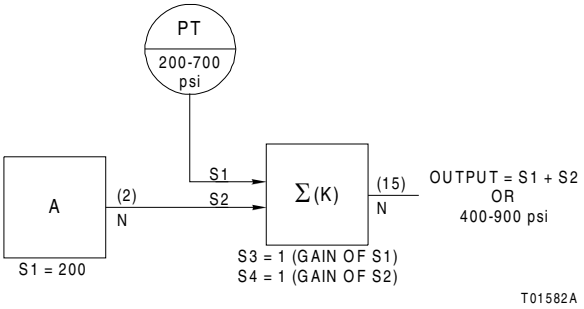


Figure 2-1 Scaler Configuration

Set Point

Figure 2-1 illustrates how to use the manual set constant for a set point configuration. The transmitter range is ten to 20 inches of water. The desired set point (output of the summer block) is 15 inches of water. By setting the manual set constant to 15, the summer subtracts <S2> from <S1>. Thus, when <S1> equals 15 inches of water, output N of the summer block equals zero inches of water which shows the level has reached set point.

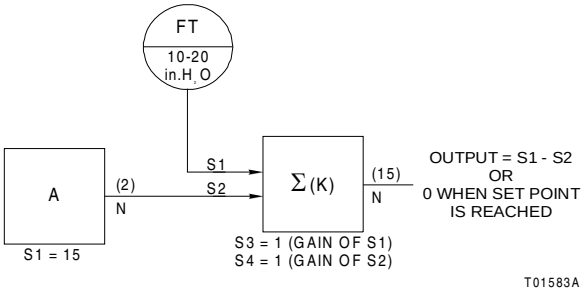
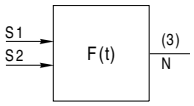


Figure 2-1 Set Point Configuration

3. Lead/Lag

The output of a lead/lag function code equals the product of the time function and the input value. Specifications S3 and S4 provide lead (S3) or lag (S4) functions. Function code 3 also serves as a lead/lag filter.



Outputs

Blk	Type	Description
N	R	Output value with lead/lag function applied

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input
S2	N	0	I	Note 1	Block address of track switch signal: 0 = track 1 = release
S3	Y	0.000	R	Full	Time constant T1 (lead) sec
S4	Y	0.000	R	Full	Time constant T2 (lag) sec

NOTES:

1.Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

3.1 Explanation

Function code 3 causes the output of the function block to lead or lag changes in the input signal <S1>. The following equation describes the operation:

$$Y = Y_L + \frac{S3(\langle S1\rangle - \langle S1_L\rangle)}{S4 + dt} + \frac{dt(\langle S1\rangle - Y_L)}{S4 + dt}$$

where:

<S1>	=	Present input value.
<S1$_L$>	=	Value of the input on the previous cycle.
S3	=	Value of time constant T1 (lead) in seconds.
S4	=	Value of time constant T2 (lag) in seconds.
Y	=	Present output value.
Y_L	=	Value of the output on the previous cycle.
dt	=	Module cycle time (seconds).

The S2 term enables or disables this function. If <S2> is a logic 0, then the output equals the input <S1>. If <S2> is a logic 1, the lead or lag function is implemented.

3.1.1 Lag Function

To select the lag function, leave S3 at its initial value (0) and enter a number for S4. The equation then becomes:

$$Y = Y_L + \frac{dt(\langle S1\rangle - Y_L)}{S4 + dt}$$

S4 is the time constant term. This is the time required for the output of this function to reach 63.2 percent of the input value. The output will not reach approximately 99 percent of the input value until the end of five time constants. In this application, it will be five times S4 before the output reaches the input value. To calculate the S4 term needed for the output to equal the input in a certain number of seconds (t), use the following equation:

$$S4 = \frac{t}{5}$$

where:

S4	=	Time constant term for function code 3.
t	=	Number of seconds for the output to reach about 99 percent of the input value.
5	=	Number of time constants required for the output to reach about 99 percent of the input value.

For example, for the output to reach the input level in 30 seconds, the S4 term needed would be:

$$S4 = \frac{30}{5} = 6$$

3.1.2 Lead Function

To select only a lead function, leave S4 at its initial value of zero and enter a number for S3. The equation then becomes:

$$Y = Y_L + \frac{[S3(\langle S1\rangle - \langle S1_L\rangle)] + [dt(\langle S1\rangle - Y_L)]}{dt}$$

where:

<S1>	=	Present input value.
<S1$_L$>	=	Value of the input on the previous cycle.
S3	=	Time constant T1 (lead) in seconds.
S4	=	Time constant T2 (lag) in seconds.
Y	=	Present output value.
Y_L	=	Value of the output on the previous cycle.
dt	=	Module cycle time (seconds).

The output is set to the value that the input will be in (S3) seconds if it continues to change at the same rate as it did during the last cycle. The lead function is essentially equal to the derivative function except that the block output eventually equals the input if the input remains constant long enough. The output of a derivative function is zero when the input is not changing.

3.2 Applications

Figures 3-3 and 3-4 illustrate some general input and output signal shapes for a function code 3 used as a lag filter and as a lead filter respectively. The input signals shown in Figures 3-1 and 3-1 are ideal waveforms for electronic circuits. Actual outputs and inputs vary because Symphony function codes are preprogrammed algorithms.

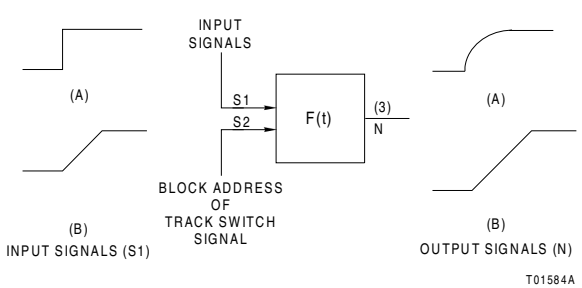


Figure 3-1 Lag Filter

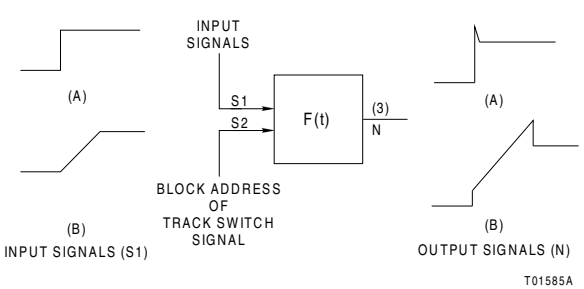


Figure 3-1 Lead Filter

Figures 3-5 and 3-6 are simplified examples of using function code 3 in boiler applications. Figure 3-5 shows function code 3 used as a lag to delay decreases in air flow for a load decrease. Figure 3-6 shows function code 3 used as a lead/lag to compensate for drum level shrink and swell due to changes in steam flow.

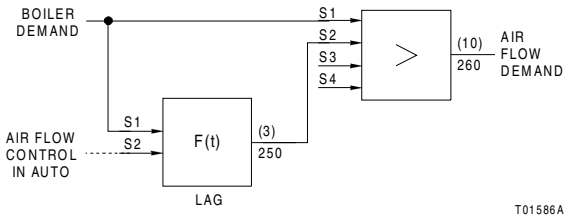


Figure 3-1 Lag to Delay Decreases in Air Flow on a Load Decrease

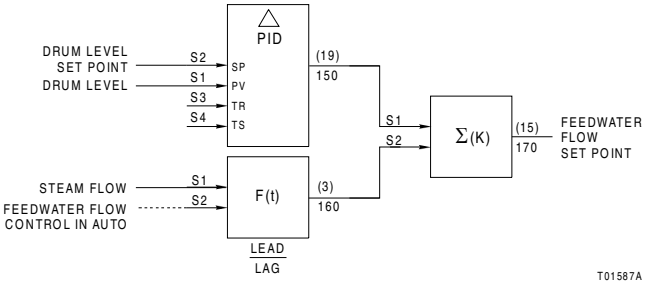
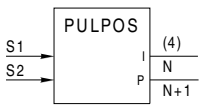


Figure 3-1 Lead/Lag to Compensate for Drum Level Shrink and Swell

4. Pulse Positioner

The pulse positioner (PULPOS) function code compares two analog input signals and produces output pulses that are proportional in time duration to the difference between these two analog signals. Both inputs are expressed as a percentage of the total range or span of the process parameter. Any difference is converted to a timed forward or reverse boolean output. The time duration of the boolean signal is proportional to the percent error and the specified stroke time. There are provisions for specifying the error dead band and the cycle time.
The PULPOS function code uses two consecutive block addresses. For correct operation, block addresses N and N+1 should be sent directly to two digital output blocks in the same I/O module. This is accomplished by using any digital output (such as function code 79, 83, 225, etc.) for the Harmony controllers.
Outputs N and N+1 of function code 4 must be in the same I/O group of function code 83 (digital output group).



Outputs

Blk	Type	Description
N	B	Output value of timed raise (forward pulse)
N+1	B	Output value of timed lower (reverse pulse)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of set point signal
S2	N	5	I	Note 1	Block address of feedback signal
S3	Y	0.000	R	Full	Forward stroke rate (%/sec)
S4	Y	0.000	R	Full	Reverse stroke rate (%/sec)
S5	Y	0.000	R	Full	Deadband (%) – absolute deadband
S6	Y	0.000	R	Full	Cycle time (secs)

NOTES:
1.Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

This function code is not supported for use with S800 I/O. Attempting to connect this function code to S800 I/O will cause the IOR800 to enter error mode.

NOTE:

4.1 Explanation

The output of the pulse position function is a series of pulses having pulse durations proportional to the difference between the desired set point value input signal <S1> and the actual value feedback signal <S2>. These two signals are expressed in percent of total range or span of the process parameter. <S1> and <S2> are typically scaled to the same engineering unit to obtain proper operation.
This function block produces two outputs. Output N generates a signal when the measured process value is less than the desired output. Output N+1 generates a signal when the measured process value is greater than the desired value. The assigned block number (N) references the first output which is a forward or increase output. The next consecutive block number (N+1) references the second output which is a reverse or decrease output. If this block drives a digital I/O module, both outputs must be directed to the same group on the I/O module and must use consecutive I/O module outputs.
The difference between the desired value signal <S1> and the actual value (feedback) signal <S2> is called the error signal. Specification S5 is the deadband, i.e., the amount of error that is allowed before a correction is necessary. If <S2> is less than <S1> by an amount greater than the deadband, there will be a forward output. To determine the forward output pulse widths use the calculation:

$$\textit{Forward output pulse duration} = \frac{\langle S1\rangle - \langle S2\rangle}{S3} \textit{ seconds}$$

If $\langle S1\rangle - \langle S2\rangle > S5$
Forward output pulse OFF time = (S6) – (forward output pulse duration)

Reverse output pulse duration $= \frac{\langle S2\rangle - \langle S1\rangle}{S4}$

If $\langle S2\rangle - \langle S1\rangle > S5$
Reverse output pulse OFF time = (S6) – (reverse output pulse duration)

where:

<S1>	Value of set point signal
<S2>	Value of feedback signal
S3	Value of forward stroke rate (percent per second)
S4	Value of reverse stroke rate (percent per second)
S5	Value of deadband (percent)
S6	Cycle time (seconds)

NOTES:

1. Forward and reverse output pulse durations are computed to the nearest ten-millisecond minimum for Harmony controllers.

2. If output blocks N and N+1 are not directly connected to a digital output, the forward and reverse output pulse durations are set to module segment time.

4.1.1 Cycle Time

Cycle time (S6) sets the time between calculations, or how often this function is calculated. It delays the processing of the block.

4.1.2 Stroke Rate

The stroke rates (S3 and S4) are entered in units of percent per one second. The stroke rate sets the length of time that the forward or reverse signal remains high for each percent of error (when error is greater than deadband). If S3 is set to ten percent per second and the deadband is set to two percent, then the forward output will be held high one second for every ten percent error, or until the next cycle, whichever comes first. If there is 90 percent error above the deadband when S3 is ten percent per second, then the forward output remains high for nine seconds or until the cycle ends. The minimum pulse duration is ten milliseconds and the smallest incremental pulse length possible is ten milliseconds.

Forward output pulse duration $= \frac{90\%}{10\% \text{ per sec}} = 9 \text{ sec}$

If 90% > 2

4.2 Applications

Figure 4-1 shows how the pulse position function code can control a pulse type valve positioner. The PULPOS function

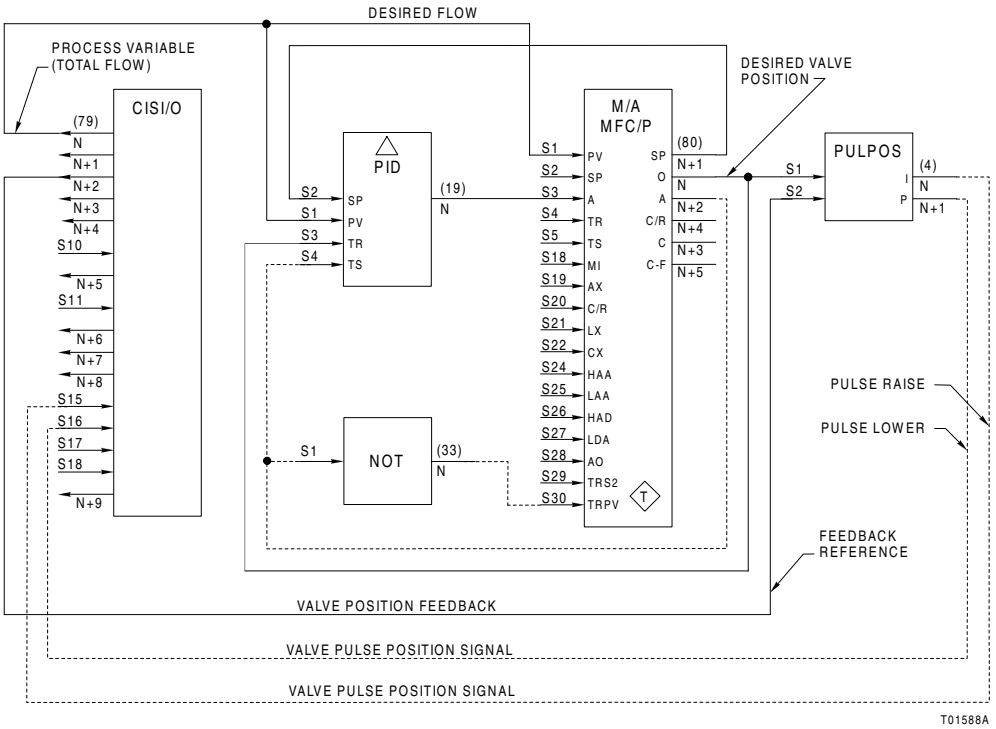


Figure 4-1 PULPOS Controlling a Pulse Valve Positioner

block is internally automatic, but there are no provisions for operator intervention. Using the PID (function code 19) and the M/A (function code 80) control loop allows selecting a desired set point for operation in the auto mode. The control loop then regulates the set point signal to the PULPOS function block to maintain a constant valve position based on the valve position feedback.

In manual mode, the PULPOS function block set point is directly selected via manipulation of the control output value. In either manual or auto mode, the PULPOS function block controls the field device based on the specification settings and the relationship exhibited between <S1> and <S2>. The PULPOS function block pulses a raise or lower output signal to adjust for errors.

NOTES:

1. The outputs of the PULPOS must go to the same device definition function code when used in BRC-100 or HAC controllers.
2. The control stations should be configured to display the position feedback on the output bar graph.

3. For BRC-100 and HAC controllers with firmware earlier than D0, a maximum of two pulse positioner function blocks can be configured in each Harmony I/O block (i.e., DIO-400, CIO-100)Figure 4-1 demonstrates how the pulse positioner function code can control a pulse type valve positioner utilizing a Harmony CIO-100 block. The following conditions must be met for this configu-

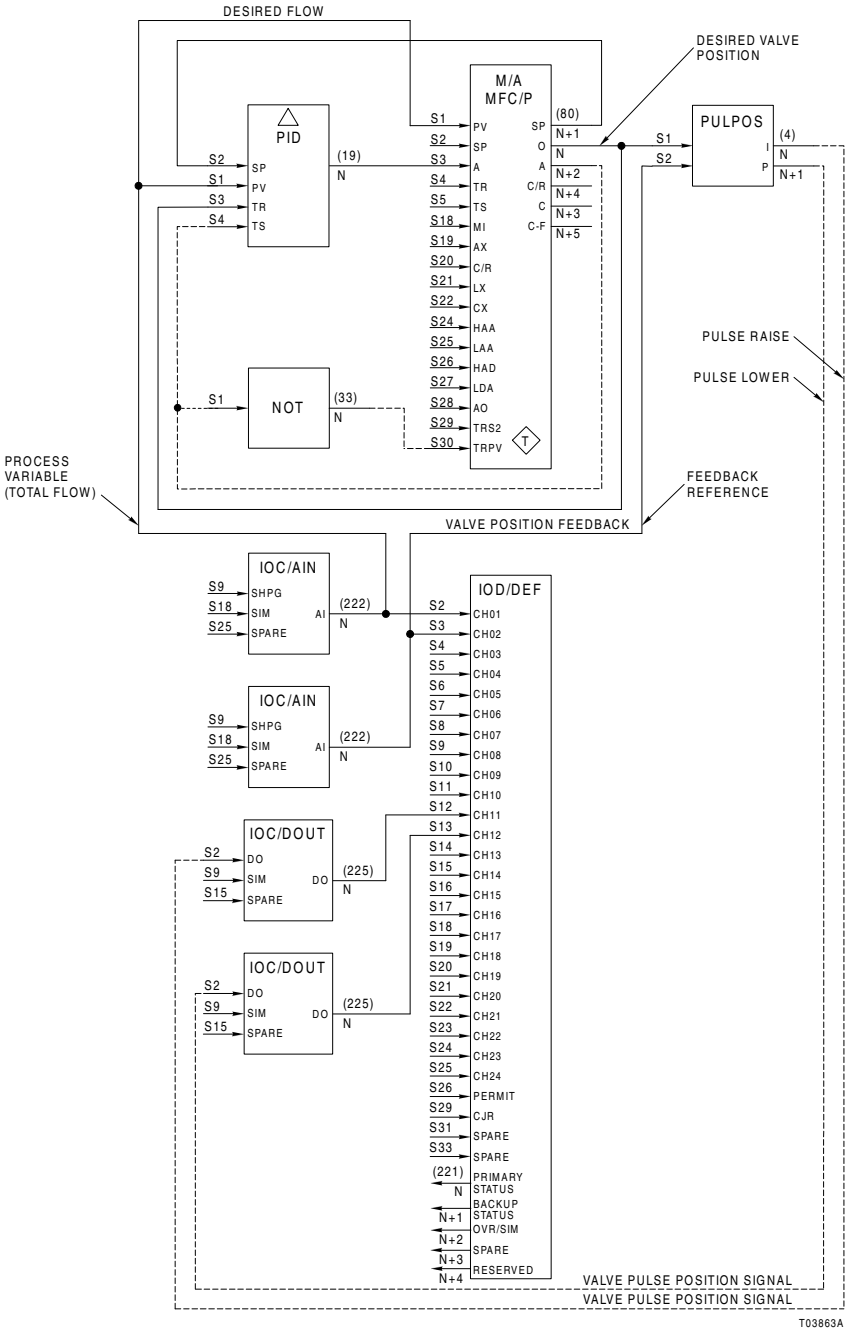


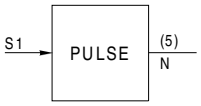
Figure 4-1 CIO-100 Configuration

ration:

- The pulse positioner function block, its associated digital output function blocks (FC225), and the device definition function block (FC221) must all reside in the same segment control block.
- The pulse positioner function block must be connected directly to a pair of digital output function blocks (FC225).
- The pulse positioner function block and its associated pair of digital output function blocks must be assigned to the same device definition function block (FC221).

5. Pulse Rate

This function accepts an analog input in engineering units/ time and produces a pulsed output signal where the pulse rate is proportional to the analog input.
The output of this function block must be directly connected to a digital output. Use function code 79 or 83 for the Harmony controllers.



Outputs

Blk	Type	Description
N	B	Repetitive pulse output having a duration proportional to the analog input

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input signal
S2	Y	0.000	R	Full	Scaling parameter (units/pulse)
S3	Y	0.000	R	Full	Low cutoff (no output below value)

NOTES:
1.Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

This function code is not supported for use with S800 I/O. Attempting to connect this function code to S800 I/O will cause the IOR800 to enter error mode.

NOTE:

5.1 Explanation

The <S1> input is an analog signal representing rate in terms of engineering units per unit of time. The S2 term sets the number of input engineering units that produce a 50 millisecond output pulse. The number of output pulses is according to the equation:

$$\textit{Number of output pulses per second} = \frac{X}{S2}$$

where:

X = Maximum value of input signal <S1>.

$S2$ = Value of scaling parameter (units per pulse).

Suppose the input signal represents zero to 100 gallons per second. It is necessary to obtain one output pulse for every 100 gallons. To accomplish this, set S2 to 100.00. If the input signal is 100 gallons per second, the output is one pulse per 100 gallons or one pulse per second. If the input signal decreases to 50 gallons per second, the output would be one pulse every two seconds, and so on. If the input flow rate is in units per minute or units per hour, then S2 must be scaled accordingly. The application section gives the procedure for determining S2.
The output pulses are always 50 milliseconds in duration, and the minimum time between pulses is 50 milliseconds so there is a limit of ten pulses per second.

5.2 Applications

The output of this function may be used to drive a counter via a digital output. To implement this function to drive a counter, follow these steps:

1. Determine the maximum flow rate for the input. Although this function always calculates the number of output pulses by units per second, flows in units per hour and units per minute may be used in the equation given in Step 5 because a factor can be inserted to adjust the scaling.
2. Determine the maximum input value to the pulse rate function at maximum flow.
3. Determine the counter capacity as follows:

$$\textit{Maximum counts per hour} = \frac{10^n}{\textit{minimum reset time in hours}}$$

where:

n = Number of digits of the counter.

Divide the results of this equation by 60 to obtain counts per minute or by 3600 to obtain counts per second.
The minimum reset time should generally be more than 24 hours.

4. Determine the desired output in terms of counts (or pulses) per hour (assuming the flow rate remains at maximum). Choose the desired counts per time to be less than what was determined in Step 3. It is generally best to make the output differ from the input by a factor of some power of ten (10, 100, 1000 etc.).
5. Calculate the S2 scaling factor using the following equation:

$$S2 = \frac{\langle S1 \rangle \textit{ maximum flow}}{\textit{desired output counts per hour at max. flow}} \times \frac{3600 \textit{ sec}}{hour}$$

where:

<S1> = Value of input signal (units per second).

This equation is to be used when the flow rate is units per hour. When the flow rate is in terms of units per minute, use 60 seconds per minute in place of the 3600 seconds per hour and substitute the units of minutes for units of hours in the equation. If the input flow rate is in seconds, omit the conversion factor entirely and use units of seconds for the terms.
Figure 5-1 shows one example of function code 5 used to obtain a count of total pounds of flow. In this example, the range of the flow is zero to 500,000 pounds per hour:

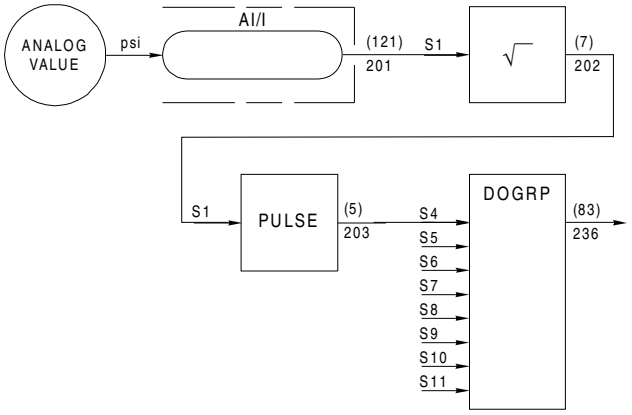
1. The maximum flow rate is 500,000 pounds per hour.
2. The input range is zero to 500, so the maximum input is 500.
3. The counter to be used has six digits and the counter should not reset in less than 24 hours. So the maximum count per hour allowable is:

$$\textit{Max. counts per hour} = \frac{10^n}{\textit{min. reset time in hours}}$$

$$= \frac{10^6}{24}$$

$$= \frac{1,000,000}{24} = 41,667 \textit{ counts per hour}$$

4. The desired counts for maximum flow is 500. This means each count will represent 1,000 pounds. This is considerably less than the counter capacity for 24 hours determined in Step 3.



SPECIFICATIONS			
FUNCTION CODE 27	FUNCTION CODE 7	FUNCTION CODE 5	FUNCTION CODE 83
S1 = 0 S2 = 500,000	S1 = 201 S2 = 707.107	S1 = 202 S2 = 3600 S3 = 0	S4 = 206

T02028A

Figure 5-1 Count Total Pounds of Flow

5. The scaling factor is:

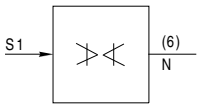
$$S2 = \frac{\textit{value of <S1> at max. flow}}{\textit{desired output counts/hr at max. flow}}$$

$$= \frac{500}{500\ \textit{counts/hr}} \times \frac{3600\ \textit{sec}}{\textit{hour}}$$

$$= 3600$$

6. High/LowLimiter

This block limits the output signal to a range that lies between a specified high and low limit. The output equals the input <S1> when the input is between the limits. Output N equals the high limit when the input is higher than the high limit and equals the low limit when the input is lower than the low limit.



Outputs

Blk	Type	Description
N	R	Output value equals input value unless the input reaches the low or high limit. Then, the output value assumes the value of the limit.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input
S2	Y	9.2 E18	R	Full	Value of high limit
S3	Y	-9.2 E18	R	Full	Value of low limit

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

6.1 Applications

Figure 6-1 shows a limiter used to limit the bias range. The flow transmitter input range is zero to 300 gallons per minute. The bias value for this example is +20 gallons per minute but the high limit should not exceed 320 gallons per minute. By using the configuration shown, the 20 gallons per minute bias is achieved without exceeding the maximum limit.

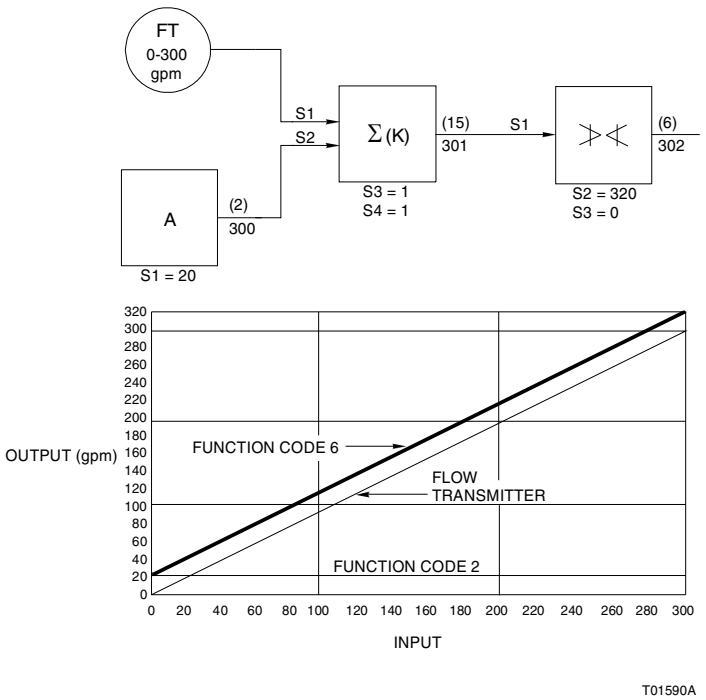


Figure 6-1 Limit a Bias Range

Figure 6-2 shows function code 6 used to limit the input to a divider function. This is necessary in many applications to prevent unprocessable quotients (i.e., divide by zero), especially when <S2> is very small.

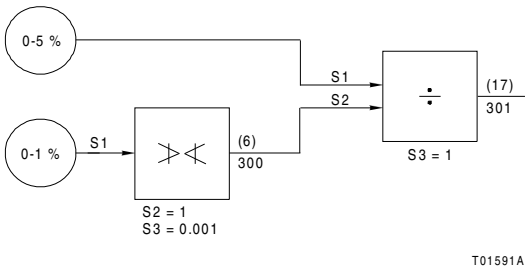


Figure 6-1 Limiting Input to a Divider

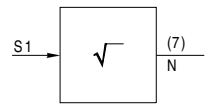
7. Square Root

This function computes the square root of the input signal in engineering units. The output equals a factor (k) times the square root of the input. The equation for this function is:

$$Y = S2\sqrt{\langle S1\rangle}$$

where:

<S1>	=	Input value.
S2	=	Gain value (k) in engineering units.
Y	=	Output value (Y = 0 if <S1> ≤ 0).



NOTES:

1. When function code 7 is utilized as a shaping algorithm for function code 222 (analog in/channel), its tunable specifications are not adaptable.

2. When function code 7 is used as a shaping algorithm, it can not at the same time also be used as a logic function because the block output will not respond to the specification S1 input. Function code 7 should not be referenced by function blocks other than function code 222 utilizing it as a shaping algorithm.

3. Multiple instances of function code 222 function blocks may utilize the same function code 7 function block as a shaping algorithm. The function code 7 shaping algorithm function block is not required to be in the same segment as the function code 222 blocks.

Outputs

Blk	Type	Description
N	R	Output value equals square root of input value multiplied by the gain value (k)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	6	I	Note 1	Block address of input
S2	Y	1.000	R	Full	Gain value (k) in engineering units (EU)

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

7.1 Applications

Specification S2 is the gain (k) applied to the value $\sqrt{\langle S1\rangle}$ and can be any real number. It is used to scale an input signal to a meaningful or easy to work with output signal. Figure 7-1 shows an example of how function code 7 can be used. In the example, a flow rate of zero to 50,000 pounds per hour is being measured by a differential pressure transducer whose output range is zero to 200 inches of water. The flow is a function of the square root of the differential pressure multiplied by some constant (k). The equation for this example is:

$$Flow = k\sqrt{\textit{diff. pressure}}$$

If it is known that the flow is 50,000 pounds per hour at a transmitter output indicating 200 inches of water differential pressure, the required constant (k) can be calculated as follows:

$$\textit{50,000 pounds per hour} = k\sqrt{200}$$

50,000 pounds per hour = k (14.142)

$$\frac{50,000}{14,142} = k$$

k = 3,535.534

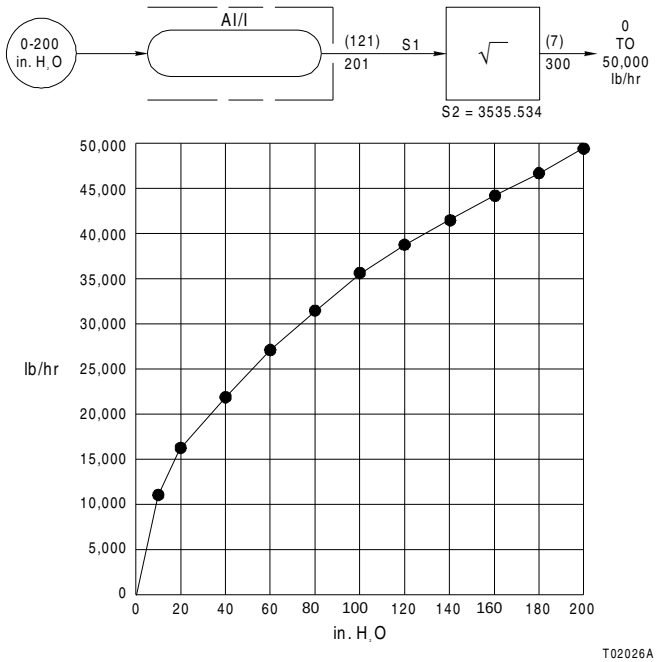


Figure 7-1 Converting a Pressure Signal to a Flow Rate

Many nonlinear inputs need to be converted to linear outputs. Figure 7-1 illustrates converting a nonlinear pressure signal to a linear flow signal using function code 7.

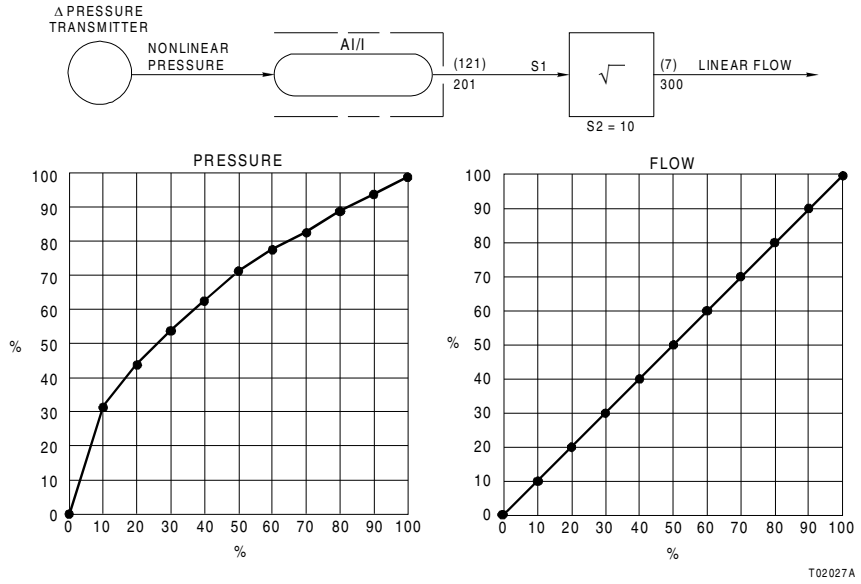
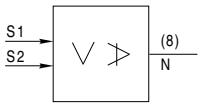


Figure 7-1 Converting a Nonlinear Pressure Input to a Linear Flow Output

8. Rate Limiter

The output of this block equals the input until the input rate of change exceeds the limit value (S3 and S4). When the rate of change of the input is greater than the limit, the output changes at the rate established by the limit until the output equals the input.



Outputs

Blk	Type	Description
N	R	Output rate equals input rate until the input rate exceeds the rate limit. Then, the output rate equals the limit.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input
S2	N	0	I	Note 1	Block address of track switch signal: 0 = track 1 = release
S3[2]	Y	0.000	R	Full	Increase rate limit (1/sec)
S4[2]	Y	0.000	R	Full	Decrease rate limit (1/sec)

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12 and 31,998 for the HC800, BRC-400/410, HPG800 and HAC.
2. S3 and S4 must be positive numbers.

8.1 Explanation

Function code 8 limits the rate of change of the output according to preset limits. To activate this function set <S2> to a logic 1. With <S2> set to a logic 0, the output is the same as the input.
Specification S3 is the rate limit of an increasing input signal in engineering units per second. As long as the input rate of increase is less than S3, the output equals the input. When the rate at which the input increases exceeds the setting of S3, the output changes at the rate set by S3 as long as the input rate of increase remains greater. Specification S4 limits the output rate of decrease when the input rate of decrease is greater than S4.

8.2 Applications

Figure 8-1 illustrates how to use the rate limiter for bumpless transfer. In the example, the M/A station sends an automatic (logic 1) or manual (logic 0) signal to <S2> of the rate limiter. When in automatic, function code 8 limits the rate of change to

the set point. For example, placing the station in automatic may cause a drastic change in set point demand. The rate limiter slows the increase or decrease set point demand to the station providing a bumpless transfer from manual to automatic.

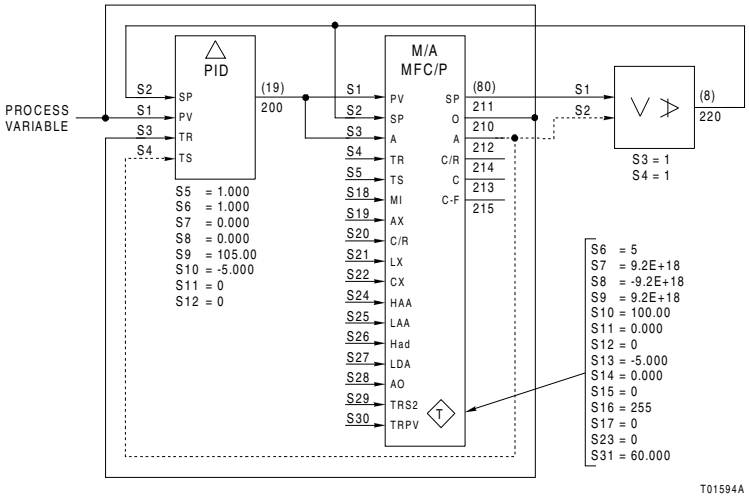
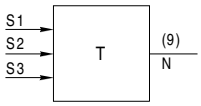


Figure 8-1 Limit the Rate of Change of the Set Point

9. Analog Transfer

This function selects one of two inputs depending on boolean input <S3>. The output of function code 9 equals the input determined by the state of input <S3>. There are two time constants to provide smooth transfer in both directions.



Outputs

Blk	Type	Description
N	R	Output equals one of two possible inputs

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of first input
S2	N	5	I	Note 1	Block address of second input
S3	N	0	I	Note 1	Block address of transfer signal: 0 = output equals <S1> 1 = output equals <S2>
S4	Y	0.000	R	Full	Transfer to input 1 time constant/sec
S5	Y	0.000	R	Full	Transfer to input 2 time constant/sec

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

9.1 Explanation

Specification S3 is the block address of the transfer signal that selects which input (<S1> or <S2>) transfers to the output. If <S3> is a logic 0, then <S1> will be transferred to the output. If <S3> is a logic 1, <S2> will be transferred to the output. When the transfer block changes the input selected, the output level changes to the new input level exponentially over a period of five time constants when the transfer time constant (S4 and S5) is set to a value other than zero. After five time constants, the output tracks the selected input.
Specifications S4 and S5 are time constant terms. They specify the time required for the previous output value to reach 63.2 percent of the present input value. The output will essentially match the new input value after five time constants have passed. To calculate S4 or S5 so that the output equals the input in a certain number of seconds (t), use the following equation:

$$S4 \text{ or } S5 = \frac{t}{5}$$

where:

S4 or S5	=	Time constant term for function code 9.
t	=	Transfer time. The number of seconds for the output to match the input value. A common transfer time is ten seconds.
5	=	The number of time constants required for the output to match the input value.

For example, if the required output must match the <S1> level 30 seconds after a transfer, and match the <S2> level in 15 seconds after a transfer:

$$S4 = \frac{t}{5} = \frac{30}{5} = 6.0$$

$$S5 = \frac{t}{5} = \frac{15}{5} = 3.0$$

9.2 Applications

Figure 9-1 shows how function code 9 can be used as an analog memory. In this example the output tracks <S1> when the <S3> digital input is a logic 0. The output value holds at its last level when the <S3> input is a logic 1.

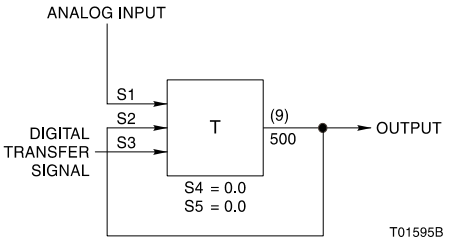


Figure 9-1 Analog Transfer Function Used as a Memory Function

Figure 9-2 shows how function code 9 can be used as a manual to automatic transfer switch. When <S3> of function code 9 equals 0, the auto signal for the M/A block equals the analog input (<S1>) of function code 9. When <S3> of function code 9 equals 1, the auto input signal tracks the output of the M/A station.

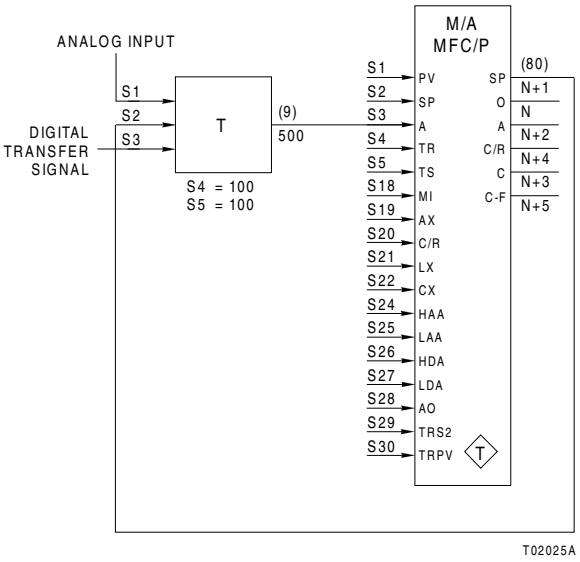
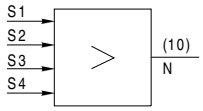


Figure 9-1 Manual to Auto Transfer Switch

10. High Select

This function selects and outputs the input with the highest algebraic value.



Outputs

Blk	Type	Description
N	R	Output equals the highest of the four inputs

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	8	I	Note 1	Block address of first input
S2	N	8	I	Note 1	Block address of second input
S3	N	8	I	Note 1	Block address of third input
S4	N	8	I	Note 1	Block address of fourth input

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

10.1 Applications

The most common use of function code 10 is to select the highest value. Function code 10 can also be used to memorize the highest value over a period of time.
To memorize the highest value over a period of time (time is set with S3 of function code 9), use function code 10 with function code 9 as shown in Figure 10-1. Create a loop with the output of function code 10 as an input for function code 9, and the output of function code 9 as one input to function code 10. Function code 9 tracks the output of function code 10. The output of function code 9 feeds back to function code 10, thus, memorizing the value of the input with the highest algebraic value. This continues until S3 of function code 9 switches the input signal from <S2> to <S1>.

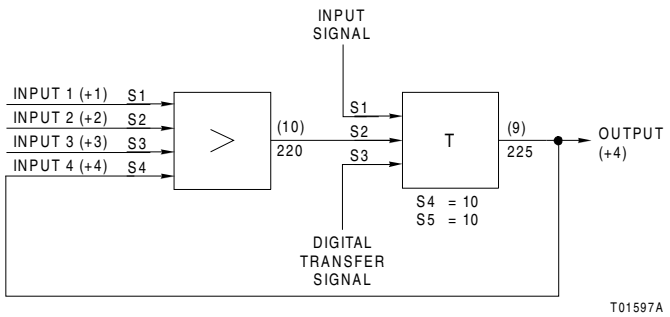
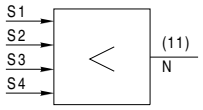


Figure 10-1 Memorize a High Value Over a Period of Time

11. Low Select

This function selects and outputs the input with the lowest algebraic value.

Outputs



Blk	Type	Description
N	R	Output equals the lowest of the four inputs

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	9	I	Note 1	Block address of first input
S2	N	9	I	Note 1	Block address of second input
S3	N	9	I	Note 1	Block address of third input
S4	N	9	I	Note 1	Block address of fourth input

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

11.1 Applications

As well as selecting the lowest input value (common application), function code 11 can be used to memorize the lowest value over a period of time.
To memorize the lowest value over a period of time (set with S3 of function code 9), use function code 11 with function code 9 as shown in Figure 11-1. Create a loop with the output of function code 11 as an input to a function code 9 block and the output of function code 9 as an input to the function code 11 block. By selecting the output of the block executing function code 11 as the value that function code 9 tracks, the output of function code 11 feeds back into function code 11. As a result, this yields the same value as the output of function code 11 for the period of time that it is the input with the smallest algebraic value.

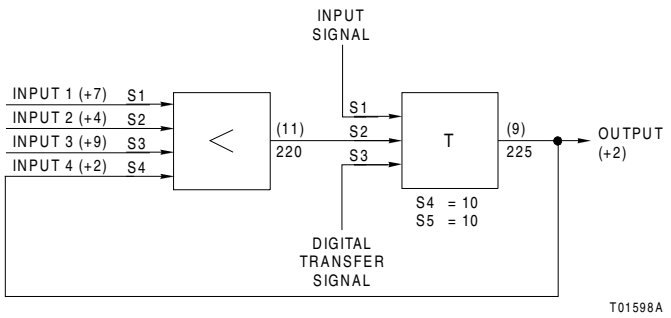
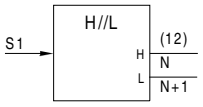


Figure 11-1 Memorize a Low Value Over a Period of Time

12. High/Low Compare

This function has two outputs. When the input is equal to or exceeds the high limit, output N equals logic 1. When the input is equal to or less than the low limit, output N+1 equals logic 1. If the value of the input is between the assigned limits, both outputs are a logic 0.

NOTE: This block uses two consecutive addresses for the outputs.



Outputs

Blk	Type	Description
N	B	High alarm output: 0 = high limit not reached 1 = high limit reached
N+1	B	Low alarm output: 0 = low limit not reached 1 = low limit reached

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input
S2	Y	0.000	R	Full	Value of alarm point/high limit
S3	Y	0.000	R	Full	Value of alarm point/low limit

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

12.1 Applications

Figure 12-1 shows function code 12 used as a signal monitor. In this example the signal shows tank level and function code 12 activates high (eight feet) and low (two feet) level alarms.

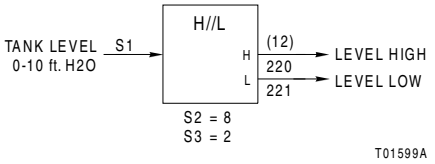


Figure 12-1 Signal Monitor

Function code 12 may also be used to indicate when the difference between two related signals is more or less than desired as shown in Figure 12-2. In this example air flow is subtracted from fuel flow by using function code 15, and function code 12 monitors the difference. If the difference exceeds a preset value, it will cause the appropriate alarm to be activated.

NOTE: It is not necessary for the high alarm value in function code 12 (S2) to be greater than the low alarm value (S3). In the example illustrated in Figure 12-2, if S2 of function code 12 = -1, and S3 = +1, the high and low outputs will both be logic 1 when fuel and air flows are within ±1 of each other.

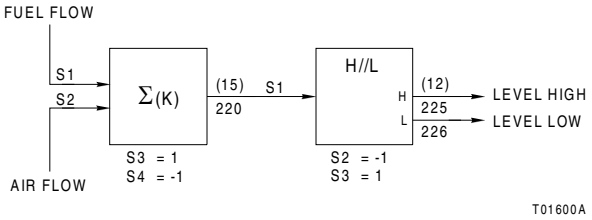
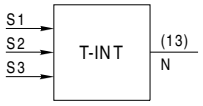


Figure 12-1 Monitor the Relationship Between Two Signals

13. Integer Transfer

This function provides a means for switching integer values. When <S3> equals zero, the output equals <S1>. When <S3> equals one, the output equals <S2>.

NOTE: This is different from the analog transfer (function code 9), which has an optional transfer time feature.



Outputs

Blk	Type	Description
N	I	Output equals one of two possible inputs

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	2	I	Note 1	Block address of first input
S2	N	2	I	Note 1	Block address of second input
S3	N	0	I	Note 1	Block address of transfer signal: 0 = S1 1 = S2

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

13.1 Applications

A common use for function code 13 is to dynamically modify integer parameters. Figure 13-1 illustrates using function code 13 with function codes 52 and 24 to change the set point tracking option in a manual/auto station. An output from a remote control memory (RCM) block (function code 62) to <S3> of function code 13 determines which input (zero for <S1> or one for <S2>) is sent to the adapt block. Function code 24 adapts <S30> of the (function code 80) to the value received from the function code 13 block.

For example, when the output of the RCM equals zero, the output of function code 13 equals <S1> (one) causing <S30> of the control station to be adapted to set point track option one (track the process variable).

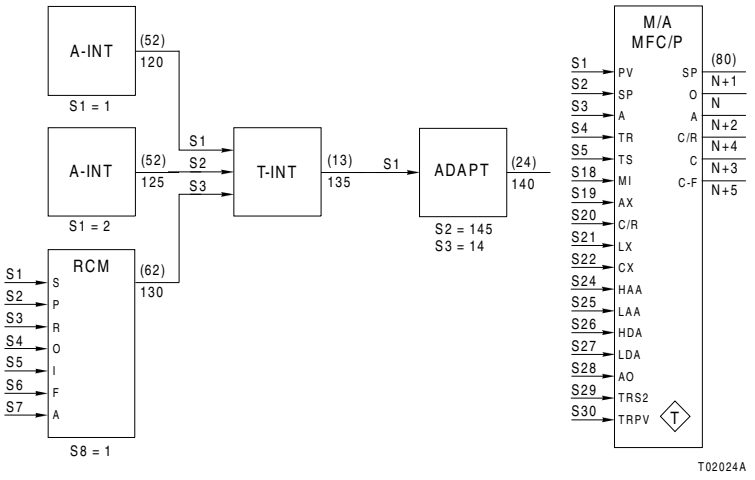
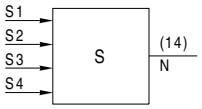


Figure 13-1 Dynamic Adaptive Control Strategy

14. Summer (4-Input)

This function computes the algebraic sum of up to four inputs with unity gain.
The output equation is:

Output (EU) = *<S1>* + *<S2>* + *<S3>* + *<S4>*



Outputs

Blk	Type	Description
N	R	Output value is the algebraic sum of the four input signals

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of first input
S2	N	5	I	Note 1	Block address of second input
S3	N	5	I	Note 1	Block address of third input
S4	N	5	I	Note 1	Block address of fourth input

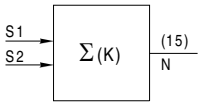
NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

15. Summer (2-Input)

This function performs a weighted sum of two inputs. By choosing the proper gains and inputs this block can perform proportional, bias or difference functions. It also can be used as a scaler for non-zero based signals by referencing the second input to a constant block.
The following equation describes the operation of this function:

$$Output = (<S1> \times S3) + (<S2> \times S4)$$



Outputs

Blk	Type	Description
N	R	Output value is the weighted algebraic sum of the two input signals

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of first input
S2	N	5	I	Note 1	Block address of second input
S3	Y	1.000	R	Full	Gain parameter of first input
S4	Y	1.000	R	Full	Gain parameter of second input

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

Applications

Besides performing proportional, bias or difference functions, this code also can be used for scaling. By referencing the second input to a constant block or to a manual set constant block (function code 2), a non-zero based signal can be scaled.
The example in Figure 15-1 shows how to scale an input with a range of 200 to 500 engineering units to give an output of ten to 110 engineering units.

The S3 constant is calculated using the equation:

$$S = \frac{Output\ Span}{<S1>\ Span}$$

$$= \frac{110 - 10}{500 - 200}$$

$$= 0.333$$

Fixed block four connects to S2 to give it a constant value of -1.0. Specification S2 could be set to any fixed value by using function code 2, but this approach requires more memory than using a fixed block. Since <S2> and S4 are both constants in this example, they can be considered as a unit. The following equation determines the value for the product of <S2> and S4:

$$<S2> \times S4 = Output\ min. - (<S1>\ min. \times S3\ min.)$$

In this example then:

$$<S2> \times S4 = 10 - [(200)(0.333)] = -56.667$$

<S2> and S4 could then be set to any allowable value that will give the product of -56.667. In our example, <S2> is set to -1.000 so S4 is set to 56.667.

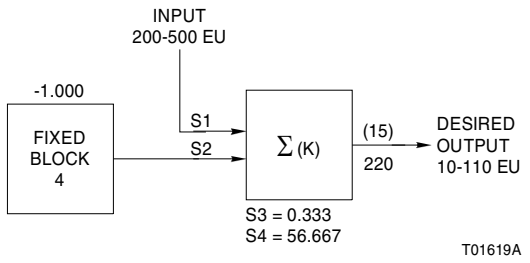


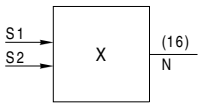
Figure 15-1 Scaler

16. Multiply

This function performs a multiplication of two input signals (<S1> by <S2>) with the result multiplied by a constant gain parameter (S3).

Output (EU) = $S3 \times (<S1> \times <S2>)$

Outputs



Blk	Type	Description
N	R	Output value is the weighted product of the two input signals

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	6	I	Note 1	Block address of first input
S2	N	6	I	Note 1	Block address of second input
S3	Y	1.000	R	Full	Gain parameter

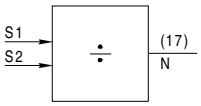
NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

17. Divide

This function causes one input <S1> to be divided by a second input <S2> and the quotient to be multiplied by a constant (S3).

$$\textit{Output (EU)} = S3 \times \frac{\langle S1 \rangle}{\langle S2 \rangle}$$



Outputs

Blk	Type	Description
N[1]	R	Output value is the weighted quotient of the two input signals

NOTES:

1. If S2 is set to 0, the output (N) is the largest value possible within the controller (e.g., 4.0 E06).

Specifications

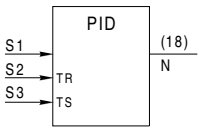
Spec	Tune	Default	Type	Range	Description
S1	N	6	I	Note 1	Block address of first input
S2	N	6	I	Note 1	Block address of second input
S3	Y	1.000	R	Full	Output gain parameter

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

18. PID Error Input

This function provides proportional, integral and derivative actions on an error signal developed from the process variable (PV) and set point (SP) inputs. The block has three inputs and one output. Besides the error input, there are track reference and track switch input signals. If the track switch <S4> is a zero, the output follows the track reference signal <S3>. This provides for smooth control transfers from manual to automatic mode. The parameters for this function block include an overall gain constant (S5), a proportional constant (S6), an integral constant (S7) and a derivative gain constant (S8).



Outputs

Blk	Type	Description
N	R	Output is PID signal in engineering units (EU)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of error signal
S2	N	1	I	0 or 1	Reserved
S3	N	5	I	Note 1	Block address of track reference signal
S4	N	1	I	Note 1	Block address of track switch signal: 0 = track 1 = release
S5	Y	1.000	R	Full	(K) gain multiplier
S6	Y	1.000	R	Full	(K_P) proportional constant
S7	Y	0.000	R	0 - 9.2 E18	(K_I) integral constant (1/min)
S8	Y	0.000	R	Full	(K_D) derivative constant (min)
S9	Y	105.000	R	Full	High output limit
S10	Y	-5.000	R	Full	Low output limit

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

18.1 Explanation

This function operates on the input error signal according to the equation:

$$Output\ (\%) = S5 \times [<S1> \times (S6 + S7I + S8D)]$$

where:

$I = dt$

$D = \frac{d}{dt}$

The purpose of the gain multiplier (S5) is to convert or scale the output. Typically it is used to convert the output to percent for input to a station or output to a field device. Figure 18-1 shows an example.

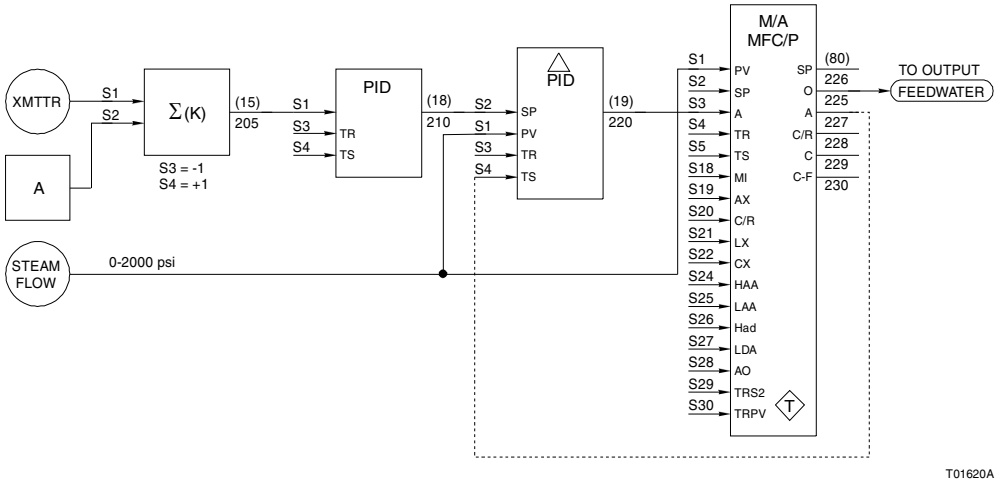


Figure 18-1 Scaling the Output

To calculate S5 in engineering units, use the equation:

$$S5 = \frac{\textit{DESIRED span}}{\textit{CURRENT span}}$$

To calculate S5 in percent, use the equation:

$$S5 = \frac{\textit{DESIRED max.} - \textit{DESIRED min.}}{\textit{S1 max.} - \textit{S1 min.}}$$

or

$$S5 = \frac{100 - 0}{\textit{S1 max.} - \textit{S1 min.}}$$

If the signal range is zero to 100 percent, the possible range of the error signal is -100 to +100 percent. Use the following equation to determine the gain multiplier S5:

$$S5 = \frac{100}{\langle S1_{\text{span}} \rangle}$$

For example, if the range of an error signal is zero to 50 cubic feet per second, then determine S5 as follows:

$$S5 = \frac{100}{50 - 0} = 2$$

NOTE: When a negative output is anticipated, the low limit (S10) must be adjusted in a negative direction to encompass the limits of the output signal range.

18.2 Applications

Figure 18-1 illustrates a PID error input block used to calibrate a demand value with an error value. By using function code 18 versus function code 19, the error input to S1 can be reported to a console.

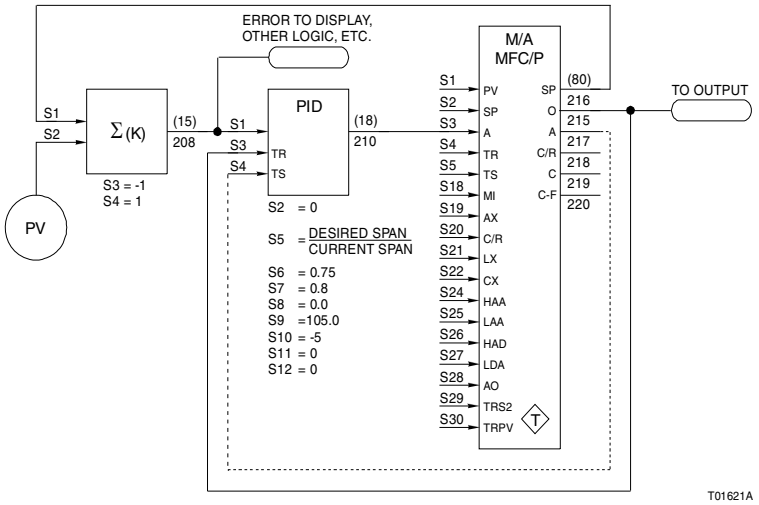
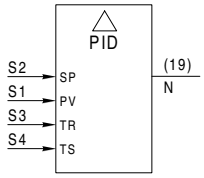


Figure 18-1 Application

19. PID (PV and SP)

This function provides proportional, integral and derivative action on an error signal developed from the process variable (PV) and set point (SP) inputs. The block has four inputs and one output. Besides the PV and the SP inputs, there are track reference and track switch input signals. If the track switch <S4> is a zero, the output will follow the track reference signal <S3>. This provides smooth control transfers from manual to automatic mode. The parameters for the function block include an overall gain multiplier (S5), a proportional constant (S6), an integral constant (S7) and a derivative gain constant (S8).



Outputs

Blk	Type	Description
N	R	Output is PID signal in engineering units (EU)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of process variable input
S2	N	5	I	Note 1	Block address of set point
S3	N	5	I	Note 1	Block address of track reference signal
S4	N	1	I	Note 1	Block address of track switch signal: 0 = track 1 = release
S5	Y	1.000	R	Full	(K) gain multiplier
S6	Y	1.000	R	Full	(K_P) proportional constant
S7	Y	0.000	R	0 - 9.2 E18	(K_I) integral constant (1/min)
S8	Y	0.000	R	Full	(K_D) derivative constant (min)
S9	Y	105.000	R	Full	High output limit
S10	Y	-5.000	R	Full	Low output limit
S11	Y	0	B	0 or 1	Set point change: 0 = normal 1 = integral only ($K_I \neq 0$)
S12	Y	0	B	0 or 1	Controller action on error: 0 = reverse acting on error 1 = direct acting on error

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12

   31,998 for the HC800, BRC-400/410, HPG800 and HAC

19.1 Explanation

This function supports two modes: direct and reverse. The function is in the direct mode when S12 is a logic 1 and in the reverse mode when S12 is a logic 0.

Direct Mode
(S4 = 1 and S12 = 1)

The set point <S2> subtracts from the process variable input <S1>.

Reverse Mode
(S4 = 1 and S12 = 0)

The process variable input <S1> is subtracted from the set point <S2>.

Refer to Figure 19-1 for an illustration of the PID algorithm for function code 19.

S5

The purpose of the gain multiplier (S5) is to convert or scale the output. Typically it is used to convert the output to percent for input to a station or output to a field device. Figure 19-2 shows an example.
To calculate S5 in engineering units, use the equation:

$$S5 = \frac{DESIRED\ SPAN}{CURRENT\ SPAN}$$

To calculate S5 in percent, use the equation:

$$S5 = \frac{100}{(\langle S1 \rangle_{max}) - (\langle S1 \rangle_{min})}$$

For example, with the function in the direct mode, the range of the process variable signal is zero to 200 cubic feet per second and the maximum control output is 100 percent.

The gain multiplier (S5) is determined as follows:

$$S5 = \frac{100}{(200 - 0)}$$

$$S5 = 0.5$$

Controller Start-Up
(Block 10 – start-up in progress = 1)

PID_{Output} = <S3>

Manual Mode
(S4 = 0)

PID_{Output} = <S3>

High and Low Output Limits (S9 and S10)

Specifications S9 and S10 set the limits of the output block value (N). The default values of S9 and S10 provide an output range of -5.000 to +105.000. When a negative output is anticipated, the low output limit (S10) must be adjusted in a negative direction to encompass the limits of the output signal range.

Set Point Change (S11)

Set point modifier. This specification defines the action taken on a set point change. A normal setting results in a jump in the control output due to the proportional contribution created by a set point change. When set to integral only on set point change, the proportional and derivative contributions of the error are not applied with set point changes. This action eliminates the jump in the control output and results in an integral only action on a change in set point.

0 = normal
1 = integral only on set point change

19.2 Examples

Figure 19-1 illustrates the PID algorithm for function code 19. Figure 19-1 shows how the PID block is typically used with a station in a control loop.

Notes for Figure 19-1

Note 1 - Bias

The bias term is either equal to the value of the combined proportional plus integral term calculated when K_i was last set to a value greater than zero (normal reset), or it is equal to the value of the track reference (external reset or tracking).

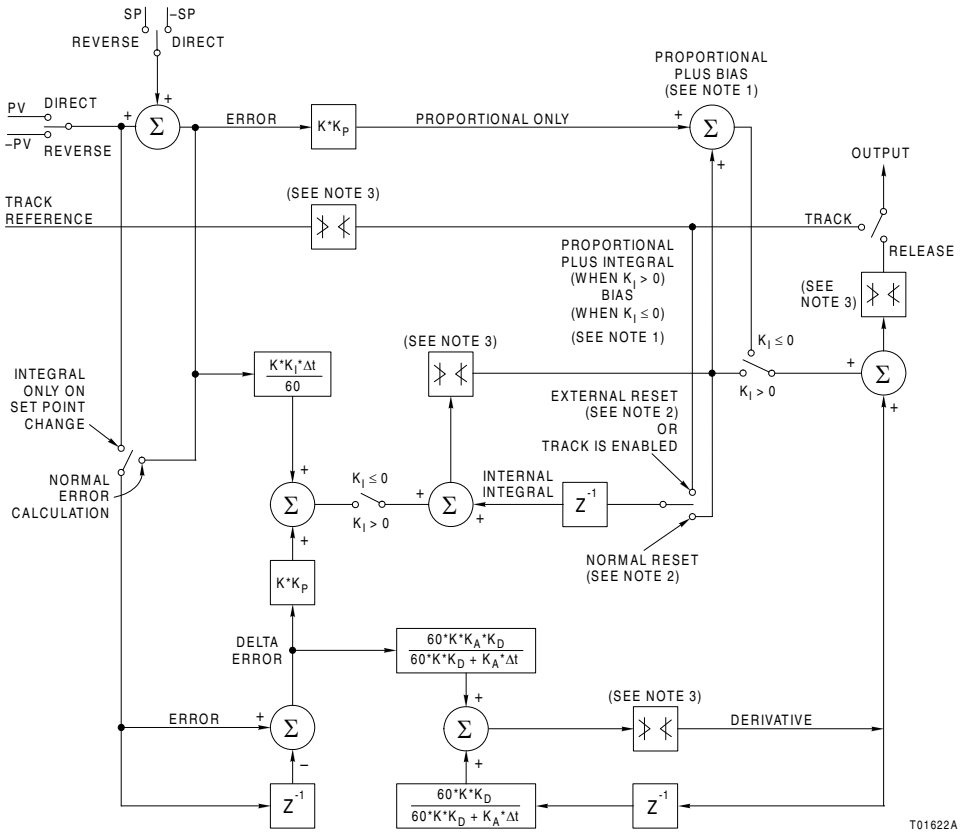


Figure 19-1 PID (PV and SP)

Note 2 - Normal Reset, Auto Selected External Reset, and External Reset

When K_i is less than zero (proportional plus integral), the internal value of the integral term is determined based on the PID reset mode specified in S5 of function code 82.
When S5 equals zero (normal reset), the PID calculates the value of the internal integral by summing the proportional term with the previous value of the internal integral.

When S5 equals two (external reset), the PID sets the value of the internal integral to the value of the current track reference.

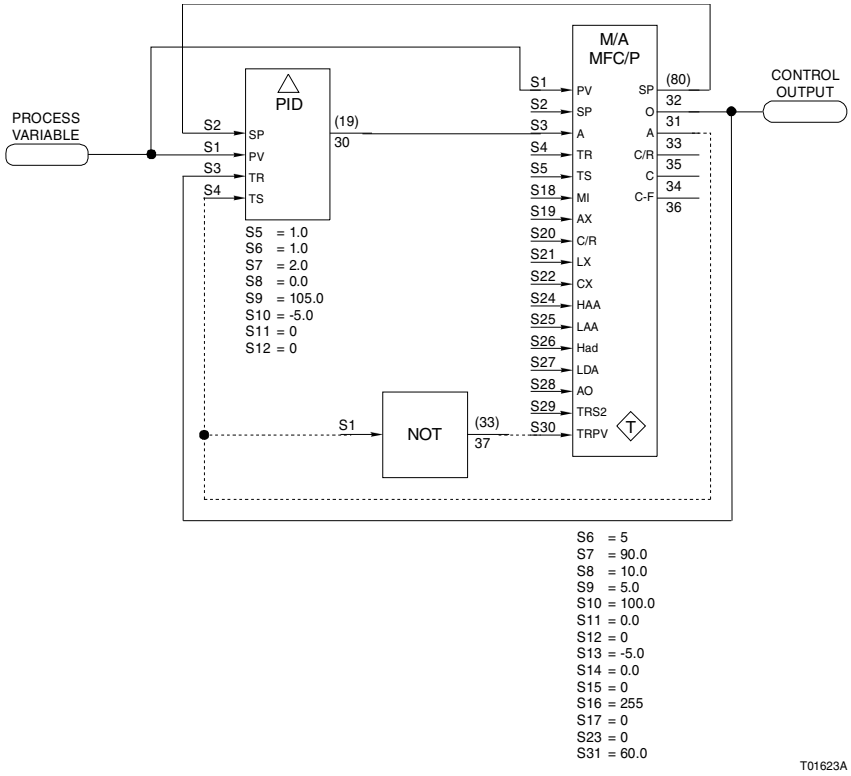


Figure 19-1 PID Control with Deviation

When S5 equals one (auto selected external reset), the PID sets the value of the internal integral to the value of the current track reference only if the value of the PID output from the previous execution period does not match the current track reference value. Otherwise, the PID calculates the internal integral value as if it were set for normal reset. The auto select external reset mode only applies when K_i is less than zero. When K_i is greater than zero (proportional only), the auto selection is disabled.

Note 3 - Range Limiting

The track reference, the proportional plus integral term, and the output are all range limited based on the high and low limits specified in S9 and S10 respectively. The high and low range limits for the derivative term are:

Derivative high limit = output high limit – output low limit

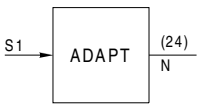
Derivative low limit = output low limit – output high limit

24. Adapt

This function allows the adaptation of a tunable parameter in the system (most tunable parameters may be modified during execution). It permits configuration of dynamic versus static loop gains in control schemes. All gains and time constants are tunable parameters. Therefore, using this function, you can set gains and time constants to fit current process operating states.

NOTE: Tunable alarm specifications in exception report function blocks are not adaptable (e.g., function code 30, S5 and S6).

The adapt function block output value has no significance. The adapted specification is modified only during execution, and the original parameter that is stored in nonvolatile random access memory (NVRAM) is not modified. Thus, the revised specification is not accessible via any of the operator interface devices unless the adapt input is read. Adapted block inputs are read by reading the output of the block preceding the block of interest.



Outputs

Blk	Type	Description
N	N/A	No significance

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of input
S2	N	0	I	Note 1 Note 2	Address of block containing specification to be adapted
S3	N	0	I	0 - 64	Specification number of specification to be adapted

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12, and 31,998 for the HC800, BRC-400/410, HPG800 and HAC.
2. The block address of the block to be adapted must be greater than the block address of the Adapt block.

The adapt function code has been enhanced to support cross type conversions for modules. The block internally converts the input value to the type of value required by the adapted block specification type. Table 24-1 shows the type of conversions performed. If the input type matches the output type, no conversion is performed. The specification value is changed to match the input value exactly when no type conversion is performed.

Table 24-1 Conversions Performed by the Adapt Block

Input Type of S1	Adapted Specification Type			
	Boolean	**Real**	**Integer (0 to 255)**	**Integer (-32768 to +32767)**
Boolean	Direct conversion	0 → 0.0, 1 → 1.0	0 → 0, 1 → 1	0 → 0, 1 → 1
Real	< 1.0 → 0, ≥ 1.0 → 1	Direct conversion	≤ 0.0 → 0, 0.0 to 255.0 → 0 to 255 ≥ 255.0 → 255	≤ -32768.0 → -32768, -32768.0 to +32767.0 → -32768 to +32767, ≥ +32767.0 → +32767
Integer	Not allowed	Not allowed	Not allowed	Direct conversion

24.1 Applications

Figures 24-1, 24-2 and 24-3 illustrate some uses of the adapt function code.
Figures 24-1 and 24-2 show the input to an adapt block as the result of a function defined by function code 1. If the function varies with time, the adapted parameter also varies with time. The same is true for functions of pressure, temperature, tank level, etc. This arrangement makes variable control of tunable parameters possible, allowing compensation for gains inherent in a process.

In Figure 24-1, the high output limit S9 of a function code 19 block varies as a function of *x* as defined in a function code 1 block.

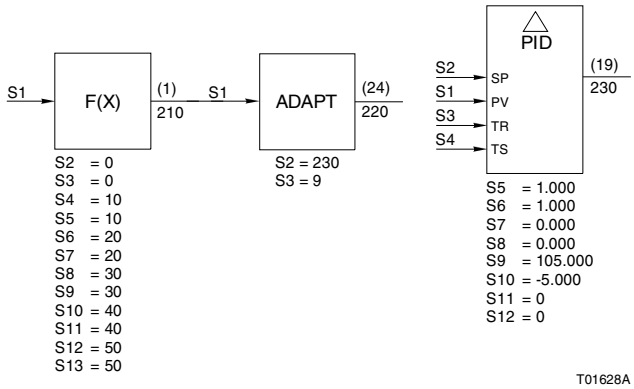


Figure 24-1 Creation of a Sliding Limiter or Index

In Figure 24-2, an adapt block adapts with proportional constant S6 of a function code 19 block to the value received from a function code 1 block, allowing the modification of the proportional constant with changes in a specified parameter, *x*.

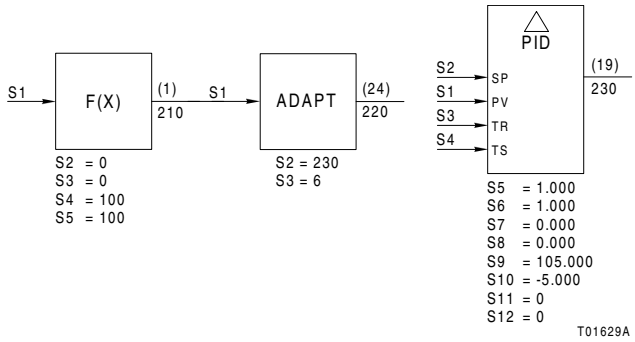


Figure 24-2 Achieve Variable Controller Proportional Constant

Figure 24-3 shows the use of an adapt block in conjunction with a function code 9 block to set a value to one of two constants, depending on an external condition. The input to the adapt block can be a linear signal or a selected fixed signal.

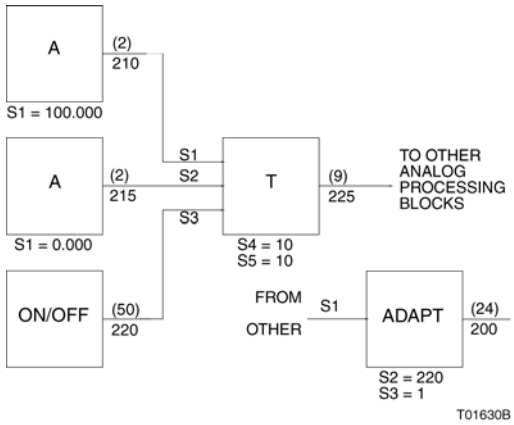
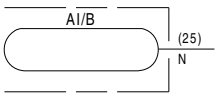


Figure 24-3 signal*Select One of Two Input Values*

25. Analog Input (Periodic Sample)

In the BRC-100 controller the analog input function code acquires an analog input signal from another module in the same PCU node via the Controlway/module bus. This analog input signal is updated at periodic intervals. The update time is specified by the periodic I/O sampling period (S13) of the segment control block located in the module containing function code 25.
The HAC controller uses FC25 to acquire an analog input signal from another HAC controller via the peer-to-peer network. Specification S8 in FC 57 node statistics block is used to specify the maximum number of destination nodes the HAC controller may communicate with. This analog input signal is updated at periodic intervals. The update time is specified by the periodic I/O sampling period (S13) of the segment control block located in the module containing function code 25.
To insure that the signal is successfully acquired from the source node/module, the analog signal generates a point quality flag. To test the quality of the signal, include a function code 31 (test quality) block in the configuration. The output of the test quality block can be used as an input to other digital processing blocks. Refer to Appendix H, for a definition of point quality.

NOTE: A block defined as an analog input reads a value for any existing analog output from a function block configured in the source node/module. No additional configuration in the source node/module is required.



Outputs

Blk	Type	Description
N	R	Output value and quality. Quality: 0 = good 1 = bad

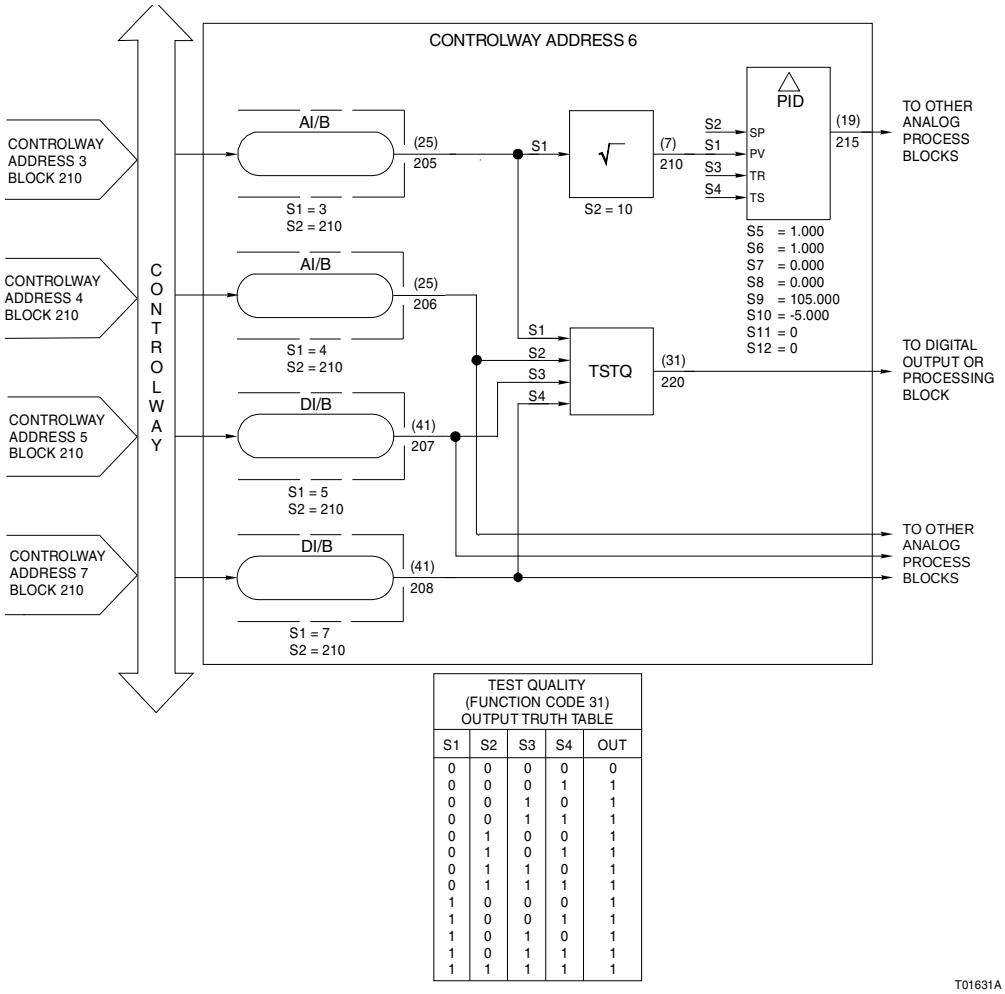
Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Requested module address
S2	N	0	I	Note 2	Requested block address

NOTES:
1. Range values are:0-31 for the BRC-100, IMMFP11/12
   1-250 for the HAC
2. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

25.1 Applications

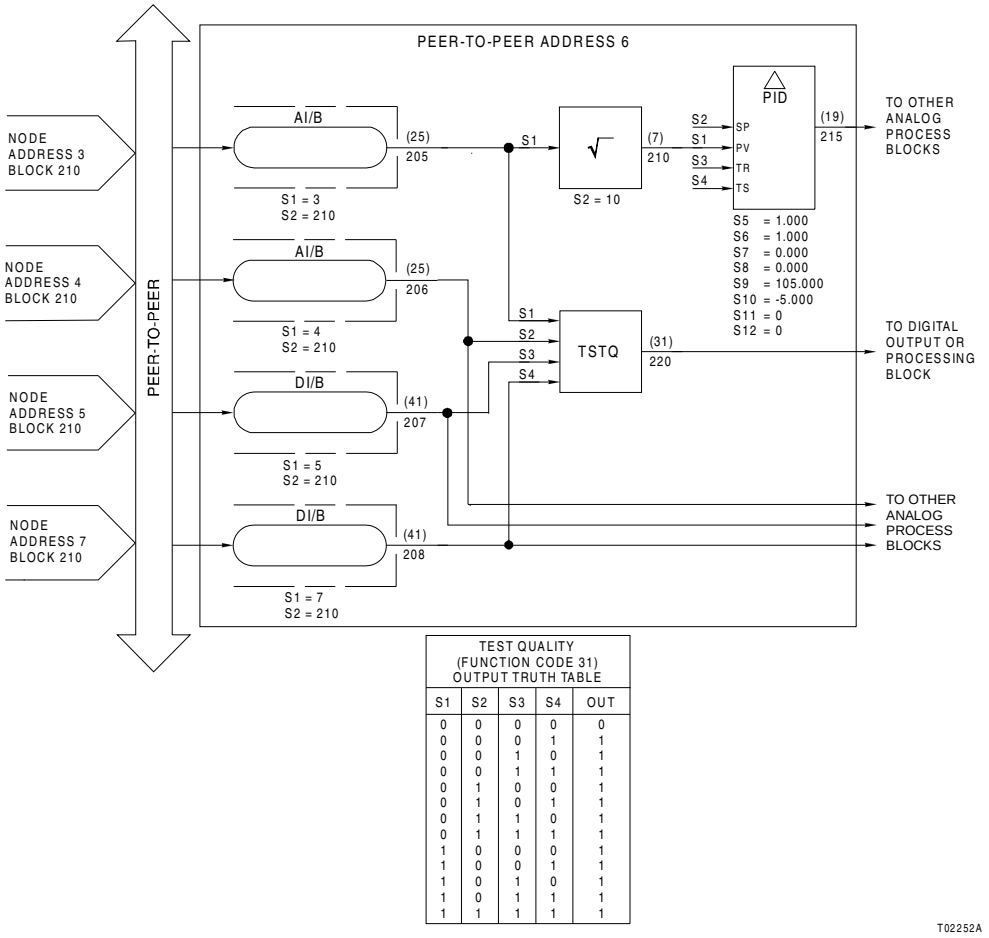
Figure 25-1 and 25-1 shows how to use function code 25 to transport an analog value from one module to another module via the Controlway or module bus. The function code 31 (test quality) block optionally monitors point quality.



TEST QUALITY (FUNCTION CODE 31) OUTPUT TRUTH TABLE

S1	S2	S3	S4	OUT
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	1	1	1

Figure 25-1 Acquire an Analog Signal from Another Module (BRC-100 only)



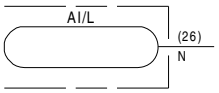
TEST QUALITY (FUNCTION CODE 31) OUTPUT TRUTH TABLE

S1	S2	S3	S4	OUT
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	1	1	1

Figure 25-1 Acquire an Analog Signal from Another Module (HAC only)

26. Analog Input/Loop

The analog input/loop function code acquires analog signals via the communication highway in the same control network. The input points are located in other Harmony control units (HCUs) and must have an exception report defined (e.g., analog exception report, function code 30). Updates are on an exception report basis. Exception report intervals are specified in the executive block or the segment control block. Use function code 121 for communication to other control networks.
To insure that the signal is successfully acquired from the Controlway/module bus, the analog signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The quality of the point cannot be used as an input to any other type of block. However, the output of the test quality block, representing the quality, can be used as an input to other analog processing blocks. Refer to Appendix H, for a definition of point quality.
Use function code 69 to test the alarm associated with the analog input/loop function block.



Outputs

Blk	Type	Description
N	R	Analog output value

NOTES:

1. Function code 26 cannot connect to a function code 30 in a different Cnet control network. Use function code 121 to connect with another loop.
2. If a controller utilizes an imported analog value from the loop in several instances, the function blocks that utilize this analog value must be connected to one analog input/loop function block that handles the importation of this point. It is not possible to import exception reports from a particular address to more than one destination analog input/loop function block within a single controller configuration.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 31	Requested module address
S2	N	0	I	Note 1	Requested block address
S3	N	0	I	0 - 249	Requested node address

NOTE:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC.

26.1 Applications

Figure 26-1 illustrates using function code 26 to acquire an analog value from another node. The Cnet connects the analog output signal in node one to the analog input in node two.

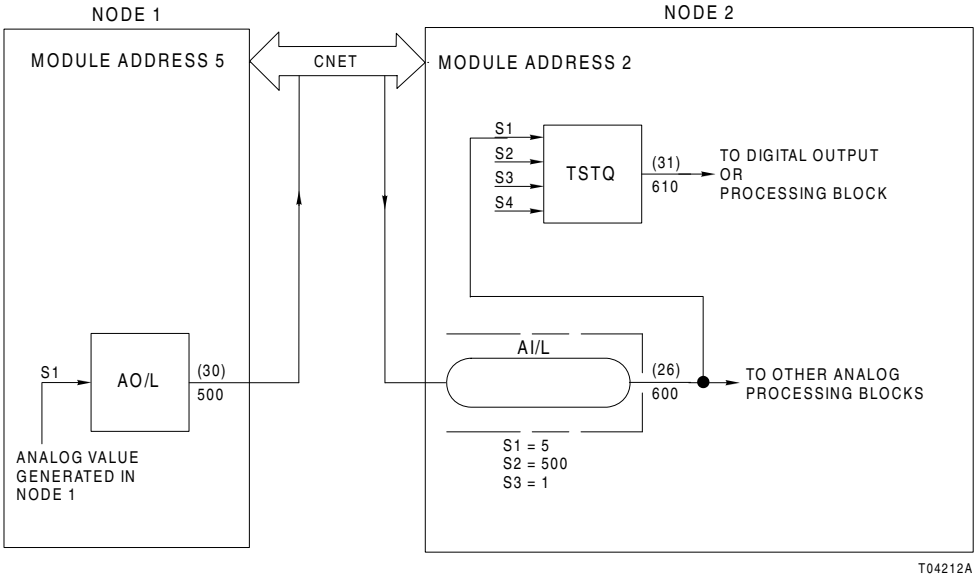
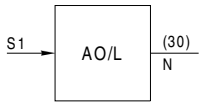


Figure 26-1 Acquire an Analog Value from Another Node

30. Analog Exception Report

The analog exception report function code allows an analog value to be sent on the communication highway if the value changes outside a configured deadband. This function also generates an alarm if the high or low limit values are reached. The analog exception report is transmitted after a time limit that is configured in the modules executive or segment control block.
This function does not perform any conversion of its input. Specifications S3 and S4 (zero and span of input) are used internally, and significant change (S7) is used to report to other Harmony control units (HCUs) or operator interface devices. Specification S2 (engineering units) is used for reporting to these higher level devices.
To insure that the signal is successfully transferred, the analog signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The quality of the point cannot be used as an input to any other type of block. The analog output value can be used as an input to any analog processing block. Refer to Appendix H, for a definition of point quality.



Outputs

Blk	Type	Description
N	R	Analog output value and quality

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input
S2	N	0	I	0 - 255	Engineering unit identifier
S3	N	0.000	R	Full	Zero of <S1> in engineering units
S4	N	100.000	R	Full	Span of <S1> in engineering units
S5	Y[2]	100.000	R	Full	High alarm point limit value
S6	Y[2]	0.000	R	Full	Low alarm point limit value
S7	N	1.000	R	Full	Significant change (% of span)

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC
2. All alarm limits are tunable, but not adaptable. Function code 24 cannot be used to adapt the values of alarm limits.

30.1 Explanation

30.1.1 Exception Reports

An exception report is returned to the bus interface module or network processing module following a report poll message if a report enable message has been received for the block number. This requires that an analog input block (function code 26 or 121) be configured in some other HCU, or as a point from a network interface unit, or as a tag defined in a console, referencing the output of the analog exception report block.
An exception report will occur when:

$$t_r \geq t_{max}$$

where:

t_r = Time since last report.

t_{max} = Maximum report time for this point as specified by the executive block or the segment control block.

or

$$\left|\langle S1\rangle - \langle S1_1\rangle\right| \geq \frac{S7 \times S4}{100} \quad \textit{and} \quad t_r \geq t_{min}$$

where:

<S1>	=	Current value of input.
$<S1_1>$	=	Last reported value of input.
S4	=	Span of input in engineering units.
S7	=	Significant change in percent.
t_r	=	Time since last report.
t_{min}	=	Minimum report time for this block as specified by the executive or segment control block.

30.1.2 Alarm Reports

An alarm report is returned to the bus interface or network processing module following a report poll message if:

1. A report enable message has been received for the block number.

and

2. The low or high alarm set point is exceeded.

An alarm report occurs when:

1. <S1> ≥ S5, and current status ≠ high alarm.
2. <S1> ≤ S6, and current status ≠ low alarm.
3. (S6 + DB) ≤ <S1> ≤ (S5 – DB), and current status ≠ normal.

where:

<S1>	=	Current value of input.
S5	=	Value of high alarm limit.
S6	=	Value of low alarm limit.
DB	=	$\frac{\textit{Alarm deadband} \times S4}{100}$
S4	=	Span of input in engineering units.

Alarm deadband is defined in the executive segment control block for the harmony controllers. The executive or segment control block defines the alarm deadband for all high/low alarm reports on the module specified. Alarm deadbands prevent an excessive number of alarm reports when values are hovering around the alarm limit.

4. Time limit (T_{max}) generates an exception report after a time interval configured in the executive or segment control block.

A sample input, with alarm and exception reports identified, is plotted in Figure 30-1.

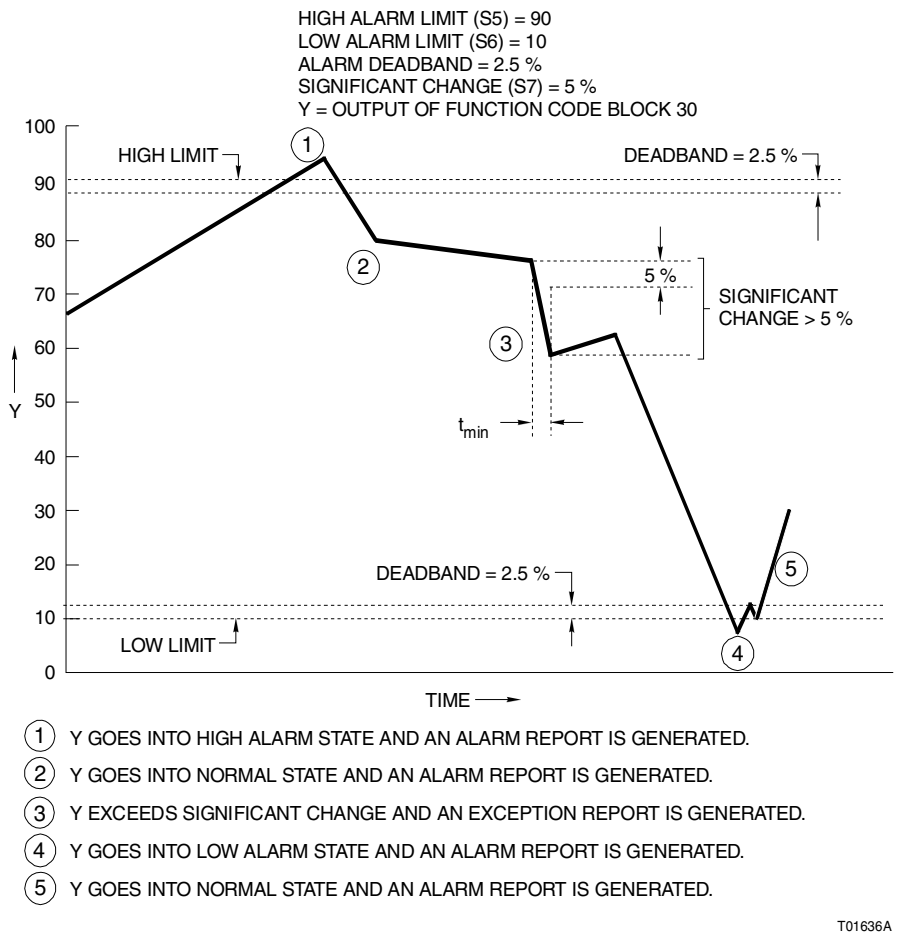
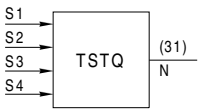


Figure 30-1 Analog Exception Report Input with Alarm and Exception Reports

31. Test Quality

The test quality function code checks the point quality of up to four inputs. It is a four input logical OR function that sets the output to a logic 0 if all tested points are good, and to a logic 1 if one or more tested inputs are bad.
Analog and digital I/O and Cnet inputs can be tested for quality. Quality is not propagated through module function blocks. All internal points will have good quality.



Outputs

Blk	Type	Description
N	B	Point quality: 0 = all inputs good 1 = at least 1 input bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of first input
S2	N	0	I	Note 1	Block address of second input
S3	N	0	I	Note 1	Block address of third input
S4	N	0	I	Note 1	Block address of fourth input

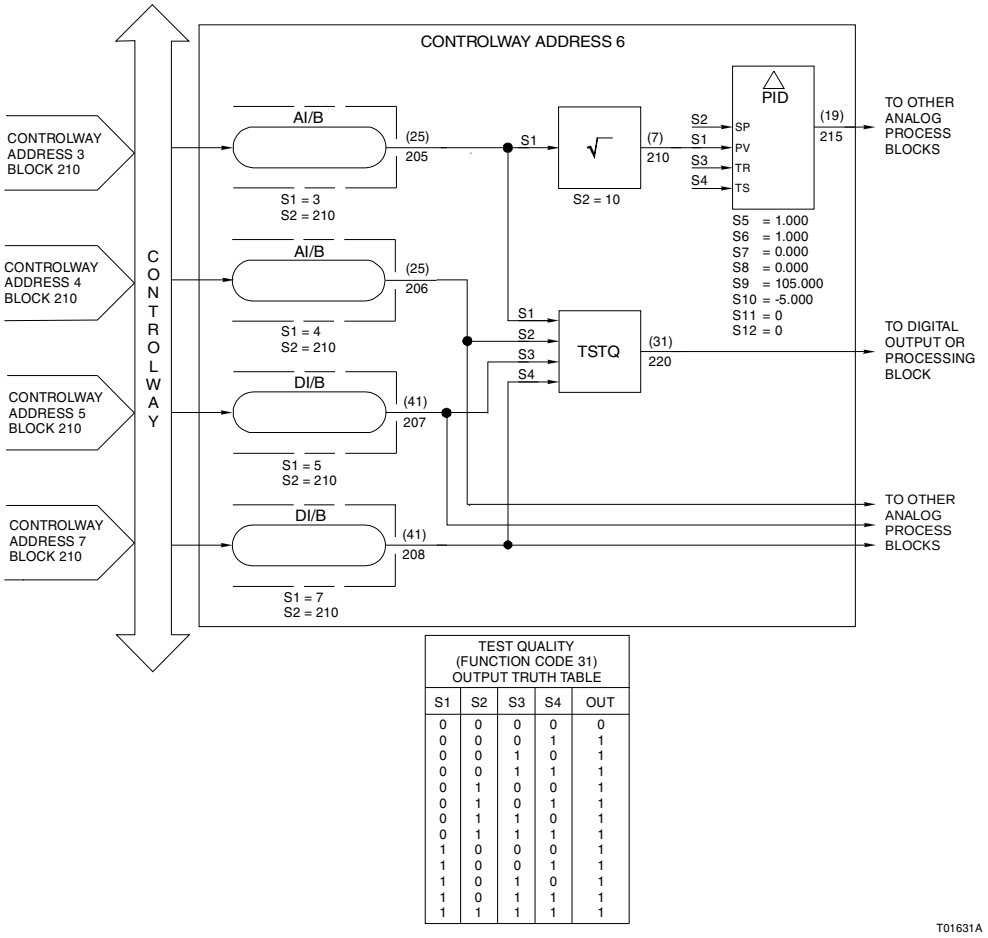
NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

Function code 31 may be used with any Harmony function code with a quality output.

31.1 Applications

The example shown in Figure 31-1 shows four analog points transferred from Controlway addresses three, four, five and seven into Controlway address six. The test quality block checks the point quality of all four analog points. If one or more of

the points are bad quality, the output of the test quality block (block 220) is a logic 1. When all points are good, the output is a logic 0.



TEST QUALITY (FUNCTION CODE 31) OUTPUT TRUTH TABLE

S1	S2	S3	S4	OUT
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	1	1	1

Figure 31-1 Test the Quality of Analog and Digital Inputs

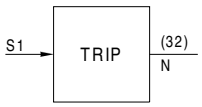
32. Trip

The trip function code disables a controller by internally forcing it into the error mode when the input is a logic 1. When the controller goes to error mode, the machine fault timer times out, stopping all communication with the controller. All outputs go to their default value when the timer times out. Analog values go to zero percent, 100 percent, or hold, depending on the output's hardware configuration. All logic outputs for controllers go to logic 0. Harmony rack I/O modules have software configured logic states (function code 128).

NOTE: This function code is used to support Harmony rack I/O modules only.

When function code 32 trips the controller, reset the controller by pressing the reset pushbutton on the front of the controller. After the controller is reset, it will be in error mode. A Composer workstation or Conductor console will display the number of the block that tripped the controller, allowing the control strategy to be checked to determine the reason for the trip. To place the controller in execute mode after an error, place it in configure mode and then in execute mode.

NOTE: The cause of the trip must be corrected or the controller will not remain in execute mode.



Outputs

Blk	Type	Description
N	B	Module status: 0 = normal operation 1 = module disabled

Specifications

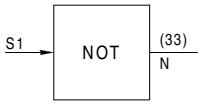
Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of input

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

33. Not

Function code 33 performs a logical negation of the input (the output is the opposite of the input).



Outputs

Blk	Type	Description
N	B	When input equals 1, output equals 0 When input equals 0, output equals 1

Specifications

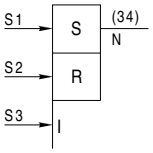
Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of input

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

34. Memory

This function memorizes its previous output when both inputs are logic 0. Specification S1 is the set (S) input, and S2 is the reset (R) input. When both inputs have the value logic 1, the output assumes the override state specified by S4. Specification S3 is the initial state flag. The value specified in <S3> will be the output after power up or a controller reset. Table 34-1 shows that the initial state depends solely on the value of <S3>. The values of <S1>, <S2> and S4 have no effect on initial output.
Table 34-2 shows the output for all other input combinations.



Outputs

Blk	Type	Description
N	B	Refer to Tables 34-1 and 34-2

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of set input
S2	N	0	I	Note 1	Block address of reset input
S3	N	0	I	Note 1	Block address of initial state
S4	N	0	B	0 or 1	Override value: 0 = reset 1 = set

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

Table 34-1Power Up or Controller Reset Truth Table

Inputs		Override S4	Initial <S3>	Output N
<S1>	<S2>			
0	0	X	0	0

Table 34-2 Normal Operation Truth Table

Inputs		Override S4	Initial <S3>	Output N
<S1>	<S2>			
0	0	X	X	No change from previous output
1	0	X	X	1
0	1	X	X	0
1	1	0	X	0
1	1	1	X	1

34.1 Applications

This function code creates deadbands and retains signals. Figure 34-1 shows how to control a fan in the following manner:

1. When air temperature reaches 21°C, the fan will turn on.
2. When air temperature goes below 15°C, the fan will shut off.
3. The fan will turn on when it receives a logic 1 signal, and it will turn off when it receives a logic 0 signal.

4. Use a function code 12 (high/low compare) block to activate the <S1> (S) input when the air temperature reaches or exceeds 21°C. Use the same function code 12 block to activate the <S2> (R) input when air temperature is 15°C or lower. Figure 34-1 illustrates this configuration.

If T ≥ 21°C, <S1>=1, <S2>=0, output=1, fan turns on.
If T is between 15° and 21°C (15°C < T < 21°C), <S1>=0, <S2>=0, output = no change from previous output (1), fan remains on.
If T ≤ 15°C, <S1>=0, <S2>=1, output=0, fan shuts off.

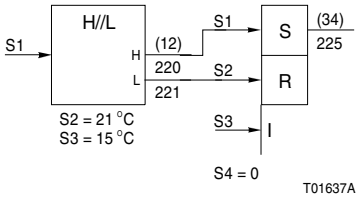
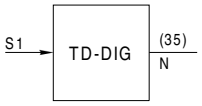


Figure 34-1 Regulatory Control

35. Timer

The timer function code performs timing, pulsed timing, or timed out delay functions. The timing mode is specified by S2 and the duration of time delay is specified by S3. Figure 35-1 shows output shapes for each mode of operation.



Outputs

Blk	Type	Description
N	B	Logic state defined for pulse, timing or timed out delay functions

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of input
S2	N	0	I	0 - 2	0 = pulse output 1 = timed out 2 = timing 00X = normal 10X = time during start up period **NOTE:** The tens digit is not used and must be set to 0
S3	Y	0.000	R	Full	Time delay in secs

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

35.1 Explanation

Pulse Output Mode
In the pulse output mode, the output becomes logic 1 whenever the input becomes logic 1. If the input returns to logic 0 before the time delay (S3) ends, the output will remain logic 1 for the entire interval. If the input remains a logic 1 after the time delay expires, the output will return to a logic 0 at the end of the time delay, and remain a logic 0 until there is another 0 to 1 transition of the input. Despite how long the input value remains in the logic 1 state, the output remains in the logic 1 state for the duration of the time delay specified in S3. The characteristic of the pulse output mode is often called one shot.

Timed Out Mode
In the timed out mode, the input must remain logic 1 for longer than the time delay before the output will track it. The output will remain logic 0 if the input pulse duration is shorter than the time delay, and will become logic 1 only if the input remains logic 1 for a period of time exceeding the time delay. It will then track the input.

Timing Mode
In the timing mode, the output tracks the input for the length of the time delay, but transitions to logic 0 at the end of the time delay, despite the input value. The output becomes a logic 1 whenever the input becomes a logic 1. If the input returns to

logic 0 before the specified time delay ends, then the output also returns to logic 0. If the input remains a logic 1 after the specified time delay, the output will return to logic 0 at the end of the time delay.

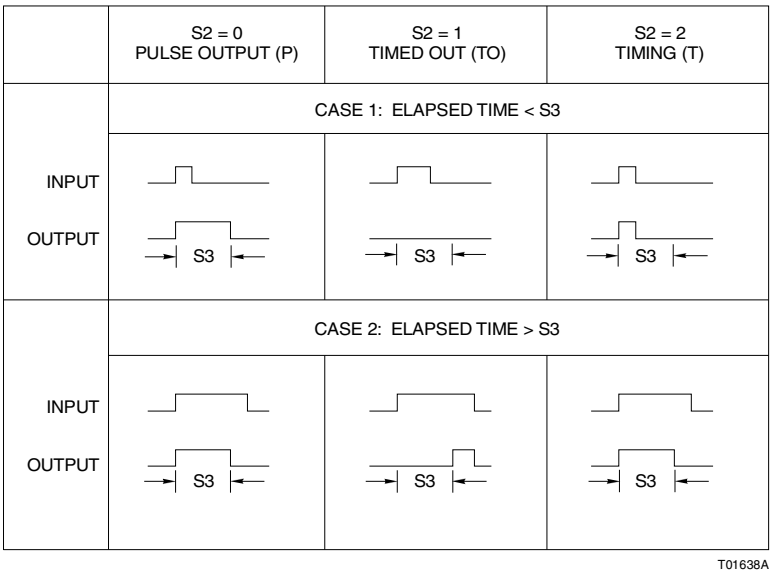
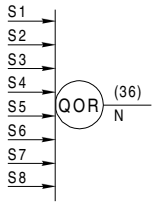


Figure 35-1 Output Shapes Obtained in the Three Timing Modes

36. Qualified OR (8-Input)

The qualified OR function code monitors the status of up to eight digital inputs and produces an output signal based upon conditions set by S9 and S10. The output is a logic 1 or 0 and is based upon the number of inputs being less than, equal to, or greater than the number specified in S9 and the condition set by S10.

Outputs



Blk	Type	Description
N	B	Output equals: Logic 0 when: No. of logic 1 inputs < S9 Logic 1 when: S10 = 0 and no. of logic 1 inputs ≥ S9 ***or*** S10 = 1 and no. of logic 1 inputs = S9

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of first input
S2	N	0	I	Note 1	Block address of second input
S3	N	0	I	Note 1	Block address of third input
S4	N	0	I	Note 1	Block address of fourth input
S5	N	0	I	Note 1	Block address of fifth input
S6	N	0	I	Note 1	Block address of sixth input
S7	N	0	I	Note 1	Block address of seventh input
S8	N	0	I	Note 1	Block address of eighth input
S9	N	0	I	0 - 8	Number of inputs that must equal logic 1
S10	N	0	B	0 or 1	Select inputs: 0 = no. of logic 1 inputs ≥ S9 1 = no. of logic 1 inputs = S9

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

36.1 Applications

This function can monitor a group of devices to determine if a certain number of the devices are operational at any given time. Figure 36-1 illustrates the use of function code 36 to monitor the number of pumps running and limit demand when less than two are operational. The output of the qualified OR block serves as a transfer signal for an analog transfer block.

If less than two pumps are running, the output of the analog transfer block will be the constant identified in the manual set constant block, and if two or more pumps are running, the output equals the input from the process.

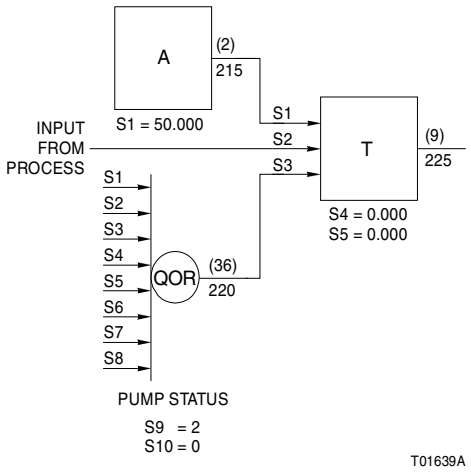
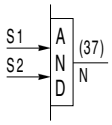


Figure 36-1 Signal Regulation Based on the Number of Pumps Operational

37. AND (2-Input)

The 2-input AND function code performs the logical AND function. The output is logic 1 when both inputs are logic 1.



Outputs

Blk	Type	Description
N	B	Refer to truth table (Table 37-1)

Table 37-3 2-Input AND Truth Table

Inputs		Output N
S1	S2	
0	0	0
0	1	0
1	0	0
1	1	1

Specifications

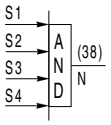
Spec	Tune	Default	Type	Range	Description
S1	N	1	I	Note 1	Block address of first input
S2	N	1	I	Note 1	Block address of second input

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

38. AND (4-Input)

The 4-input AND function code performs the logical AND function. The output is logic 1 when all the inputs are logic 1.



Outputs

Blk	Type	Description
N	B	Refer to truth table (Table 38-4)

Table 38-4 4-Input AND Truth Table

Inputs				Output N	Inputs				Output N
<S1>	<S2>	<S3>	<S4>		<S1>	<S2>	<S3>	<S4>	
0	0	0	0	0	1	0	0	0	0
0	0	0	1	0	1	0	0	1	0
0	0	1	0	0	1	0	1	0	0
0	0	1	1	0	1	0	1	1	0
0	1	0	0	0	1	1	0	0	0
0	1	0	1	0	1	1	0	1	0
0	1	1	0	0	1	1	1	0	0
0	1	1	1	0	1	1	1	1	1

Specifications

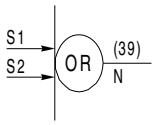
Spec	Tune	Default	Type	Range	Description
S1	N	1	I	Note 1	Block address of first input
S2	N	1	I	Note 1	Block address of second input
S3	N	1	I	Note 1	Block address of third input
S4	N	1	I	Note 1	Block address of fourth input

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

39. OR (2-Input)

The 2-input OR function code performs the logical OR function. The output is logic 1 if either or both of the inputs (<S1> and <S2>) are logic 1. The output is logic 0 when both inputs are logic 0.



Outputs

Blk	Type	Description
N	B	Refer to truth table (Table 39-5)

Table 39-5 2-Input OR Truth Table

Inputs		Output N
<S1>	<S2>	
0	0	0
0	1	1
1	0	1
1	1	1

Specifications

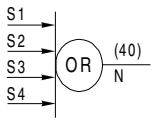
Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of first input
S2	N	0	I	Note 1	Block address of second input

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

40. OR (4-Input)

The 4-input OR function code is used to perform the logical OR function. The output is logic 1 when one or more inputs equal logic 1. When no inputs equal logic 1, the output is logic 0.



Outputs

Blk	Type	Description
N	B	Refer to truth table (Table 40-6)

Table 40-6 4-Input OR Truth Table

Inputs				Output N	Inputs				Output N
<S1>	<S2>	<S3>	<S4>		<S1>	<S2>	<S3>	<S4>	
0	0	0	0	0	1	0	0	0	1
0	0	0	1	1	1	0	0	1	1
0	0	1	0	1	1	0	1	0	1
0	0	1	1	1	1	0	1	1	1
0	1	0	0	1	1	1	0	0	1
0	1	0	1	1	1	1	0	1	1
0	1	1	0	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of first input
S2	N	0	I	Note 1	Block address of second input
S3	N	0	I	Note 1	Block address of third input
S4	N	0	I	Note 1	Block address of fourth input

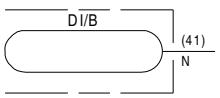
NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

41. Digital Input (Periodic Sample)

In the HAC controller, the digital input (periodic sample) function code acquires digital signals via the Controlway/module bus from other master modules. The input points must be in the same process control unit (PCU) node. The signal is updated at periodic intervals depending on the periodic I/O sampling period that is specified in the segment control block for the Harmony controllers.
The HAC controller uses FC 45 to acquire a digital input signal from another HAC controller via the peer-to-peer network. Specification S8 in FC 57 is used to specify the maximum number of destination nodes the HAC controller may communicate with. The signal is updated at periodic intervals depending on the periodic I/O sampling period that is specified in the segment control block for the Harmony controllers.
To insure that the signal is successfully acquired from the source node/module, the digital signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The output of the test quality block can be used as an input to other digital processing blocks to provide signal quality information. Refer to Appendix H, for a definition of point quality.

NOTE: A block defined as an analog input reads a value for any existing analog output from a function block configured in the source node/module. No additional configuration in the source node/module is required.



Outputs

Blk	Type	Description
N	B	Output value and quality. Quality: 0 = good 1 = bad

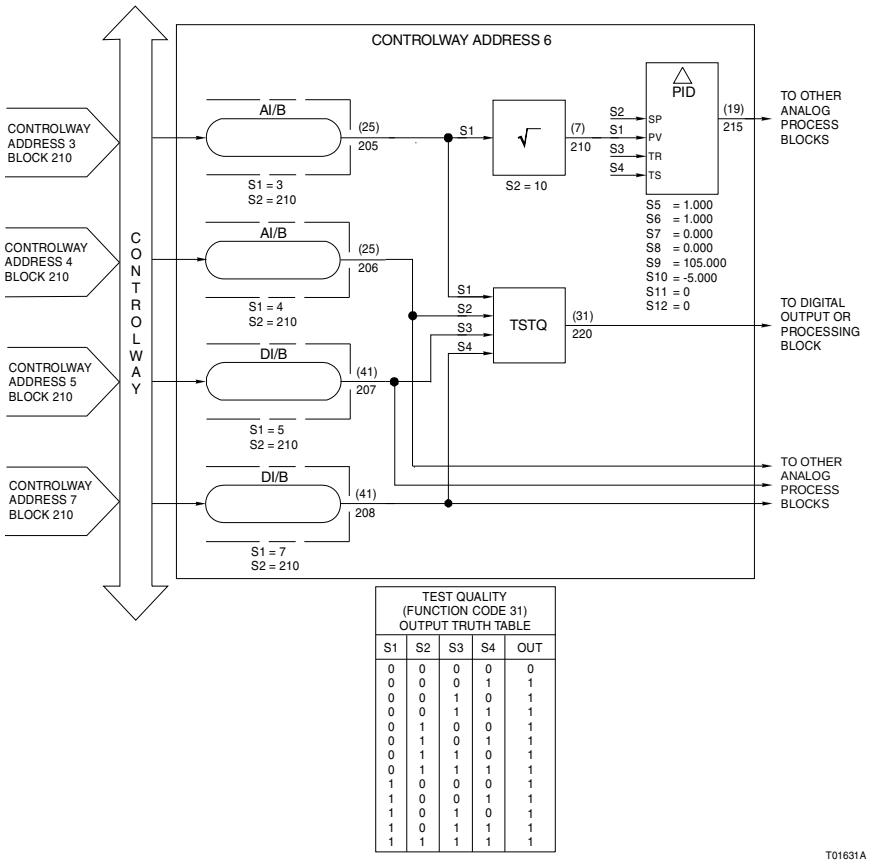
Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 31	Source module address
S2	N	0	I	Note 1	Source block address

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

41.1 Examples

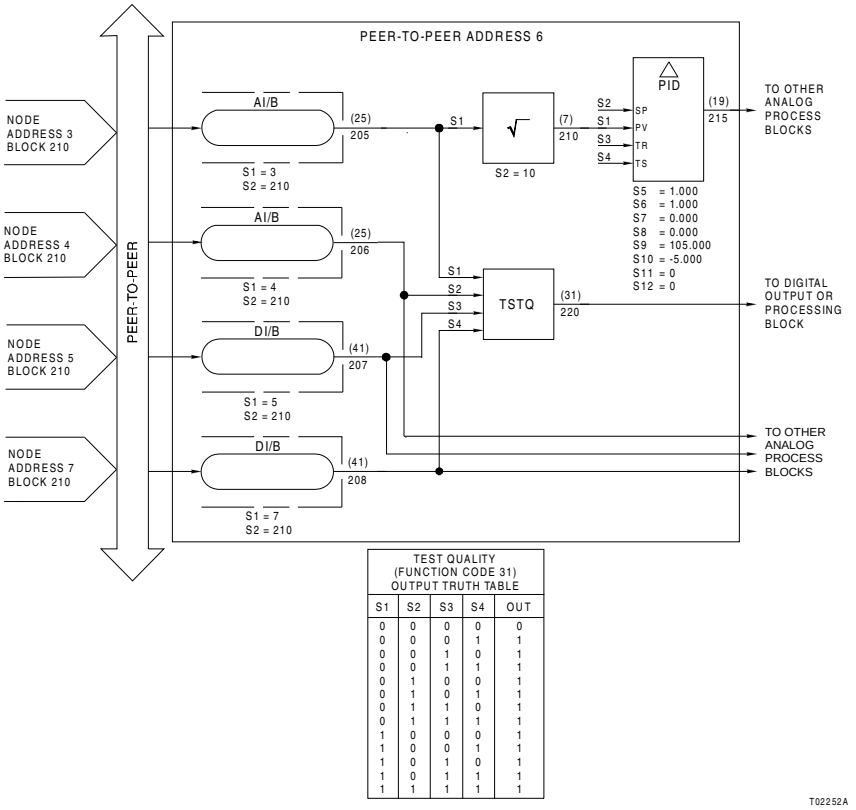
Figure 41-1 and 41-1 illustrates the configuration required to test the point quality from a digital input (periodic sample) function code.



TEST QUALITY (FUNCTION CODE 31) OUTPUT TRUTH TABLE

S1	S2	S3	S4	OUT
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	1	1	1

Figure 41-1 Acquire a Digital Signal from Another Module (BRC-100 only)



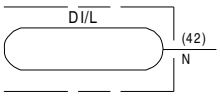
TEST QUALITY (FUNCTION CODE 31) OUTPUT TRUTH TABLE

S1	S2	S3	S4	OUT
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	1	1	1

Figure 41-1 Acquire a Digital Signal from Another Module (HAC only)

42. Digital Input/Loop

The digital input/loop function code acquires digital signals via the communication highway in the same loop. The input points are located in other Harmony control units (HCUs) and must have an exception report defined (e.g., digital exception report, function code 45). Updates are on an exception report basis. Exception report intervals are specified in the executive block or the segment control block. Use function code 122 for communication to other loops.
To insure that the module successfully acquires a signal from another node, the digital signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The quality of the point cannot be determined by any other type of block. The output of the test quality block can be used as an input to other digital processing blocks to provide signal quality information. Refer to Appendix H, for a definition of point quality.
Use function code 69 to test the alarm associated with the analog input/loop function block.



Outputs

Blk	Type	Description
N	B	Value of the function block output and quality

NOTES:

1. Function code 42 cannot connect to a function code 45 in a different Cnet loop. Use function code 122 to connect with another loop.
2. If a module utilizes an imported digital value from the loop in several instances, the function blocks that utilize this digital value must be connected to one digital input/loop function block that handles the importation of this point. It is not possible to import exception reports from a particular address to more than one destination digital input/loop function block within a single module configuration.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 31	Requested module address
S2	N	0	I	Note 1	Requested block address
S3	N	0	I	1-250	Requested node address

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

42.1 Examples

Figure 42-1 illustrates the configuration required to test point quality. Input to a function code 42 block must come from a function code 45 block.

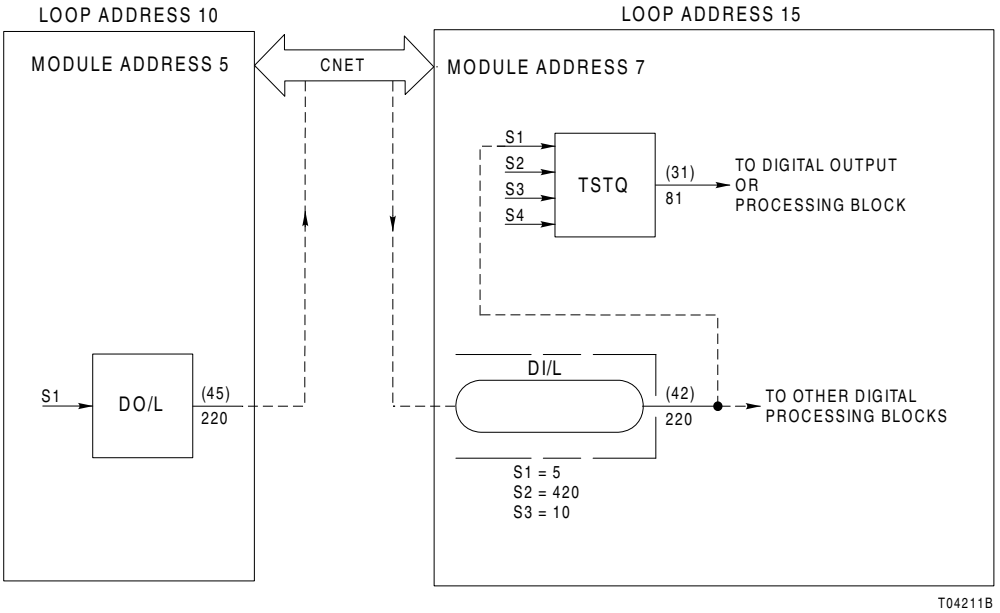
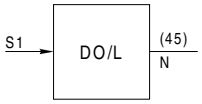


Figure 42-1 Configuration Required to Test Point Quality

45. Digital Exception Report

The digital exception report function code exception reports <S1>. Specification S2 defines the alarm state.
Exception reports are activated by a report enable signal from the module bus. The maximum number of exception reports allowed depends on the controller configuration. The number of exception reports is defined in the executive block of the controller.
Use function code 69 to test the alarm associated with the analog input/loop function block.



Outputs

Blk	Type	Description
N	B	Value of exception report

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of input
S2	N	2	I	0 - 2	Alarm state: 0 = logic 0 alarm state 1 = logic 1 alarm state 2 = no alarm state defined

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

45.1 Explanation

Exception Reports

An exception report returns to the module bus following a report poll message if a report enable message has been received for the block number. To import this data to another PCU, this requires programming a digital input/loop block (function code 42) or digital input/Cnet (function code 122) in another node referencing the output of the function code 45 block. To import this data to a console the point must be added to the tag list.
An exception report occurs when:

1. *<S1> = <S1>L* and *tr > tmin*

where:

<S1>	=	Current value of input.
$<S1>_L$	=	Last reported value of input.
t_r	=	Time since last report.
t_{min}	=	Minimum report time for the block as specified by the executive block or the segment control block.

or

2. *tr > tmax*

where:

t_r	=	Time since last report.
t_{max}	=	Maximum report time for the block as specified by the executive block or the segment control block.

An alarm report returns to the bus interface module following a report poll message if a report enable message has been received for the block number.
An alarm report occurs when:

3. <S1> = S2, current status = normal, and S2 ≠ 2

where:

<S1>	=	Current value of input.

S2 = Alarm state:
0 = alarm when <S1> = 0
1 = alarm when <S1> = 1
2 = no alarm

45.2 Examples

Figure 45-1 illustrates an example configuration for a digital exception report function code.

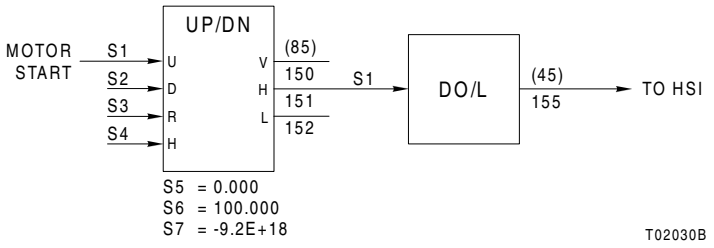
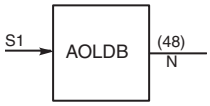


Figure 45-1 Example Digital Exception Report Configuration

a. Analog Exception Report with High/Low Alarm Deadband

The analog exception report with high/low alarm deadband function code allows an analog value to be sent on the communication highway if the value changes outside a configured deadband. This function also generates an alarm if the high or low limit values are reached. The analog exception report is transmitted after a time limit that is configured in the module's executive or segment control block.
This function does not perform any conversion of its input. Specifications S3 and S4 (zero and span of input) are used internally, and significant change (S7) is used to report to other Harmony control units (HCUs) or operator interface devices. Specification S2 (engineering units) is used for reporting to these higher level devices.
To insure that the signal is successfully transferred, the analog signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The quality of the point cannot be used as an input to any other type of block. The analog output value can be used as an input to any analog processing block. Refer to Appendix H, for a definition of point quality.
Use function code 69 to test the alarm associated with the analog exception report with high/low alarm deadband function block.



Outputs

Blk	Type	Description
N	R	Analog output value and quality

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input
S2	N	0	I	0 - 255	Engineering unit identifier
S3	N	0.000	R	Full	Zero of <S1> in engineering units
S4	N	100.000	R	Full	Span of <S1> in engineering units
S5	Y[2]	100.000	R	Full	High alarm point limit value in engineering units
S6	Y[2]	0.000	R	Full	Low alarm point limit value in engineering units
S7	N	1.000	R	Full	Significant change in engineering units
S8	Y	1.000	R	Full	High alarm deadband in engineering units
S9	Y	1.000	R	Full	Low alarm deadband in engineering units

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC
2. All alarm limits are tunable, but not adaptable. Function code 24 cannot be used to adapt the values of alarm limits.

48.3 Explanation

48.3.1 Exception Reports

An exception report is returned to the bus interface module or network processing module following a report poll message if a report enable message has been received for the block number. This requires that an analog input block (function code 26 or 121) be configured in some other HCU, or as a point from a network interface unit, or as a tag defined in a console, referencing the output of the analog exception report block.
An exception report will occur when:

$$t_r \geq t_{max}$$

where:

t_r	=	Time since last report.
t_{max}	=	Maximum report time for this point as specified by the executive block or the segment control block.

or

$$\left|\langle S1\rangle - \langle S1_1\rangle\right| \geq S7 \quad \textit{and} \quad t_r \geq t_{min}$$

where:

<S1>	=	Current value of input.
<S1$_1$>	=	Last reported value of input.
S7	=	Significant change in engineering units.
t_r	=	Time since last report.
t_{min}	=	Minimum report time for this block as specified by the executive or segment control block.

48.3.2 Alarm Reports

An alarm report is returned to the bus interface or network processing module following a report poll message if:

1. A report enable message has been received for the block number.

and

2. The low or high alarm set point is exceeded.

An alarm report occurs when:

1. <S1> ≥ S5, and current status ≠ high alarm.
2. <S1> ≤ S6, and current status ≠ low alarm.
3. (S6 + S9) ≤ <S1> ≤ (S5 – S8), and current status ≠ normal.

where:

<S1>	=	Current value of input.
S5	=	Value of high alarm limit.
S6	=	Value of low alarm limit.
S8	=	High alarm deadband in engineering units
S9	=	Low alarm deadband in engineering units

4. Time limit (T_{max}) generates an exception report after a time interval configured in the executive or segment control block.

A sample input, with alarm and exception reports identified, is plotted in Figure 48-1.

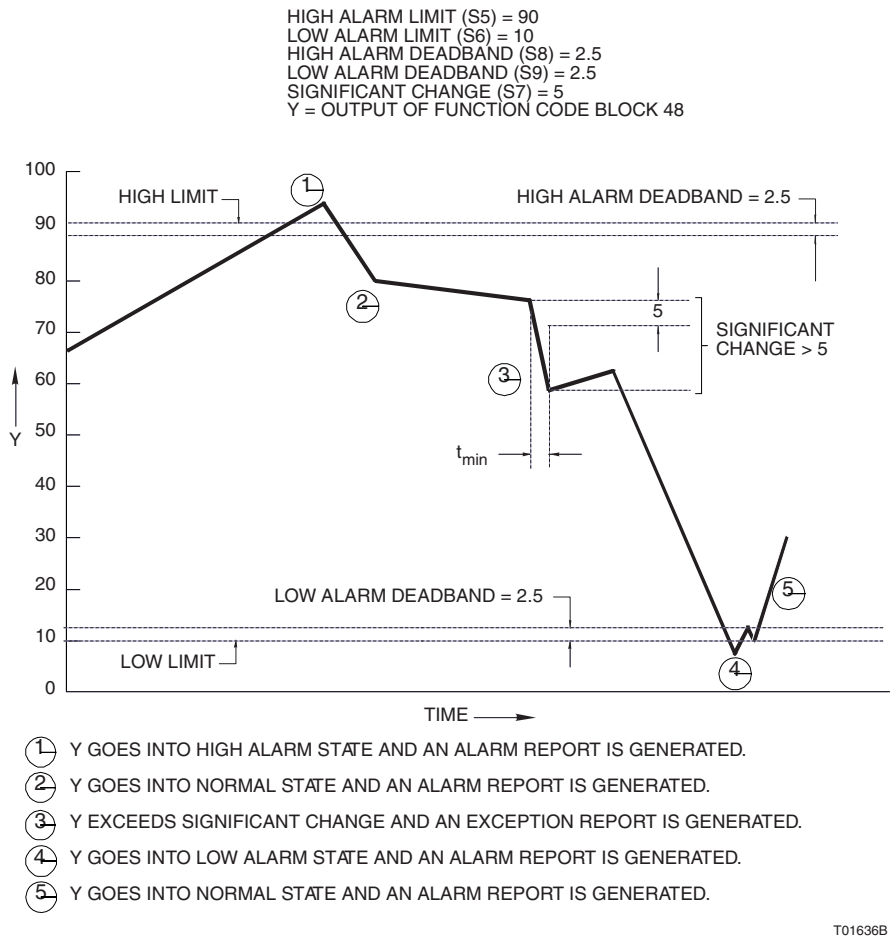
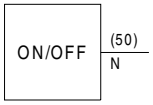


Figure 48-1 Analog Exception Report with High/Low Alarm Deadband and Exception Reports

50. Manual Set Switch

This function provides an adjustable boolean output, either logic 1 or logic 0. This function enables the engineer or technician to turn devices ON or OFF through an operator interface device (e.g., Conductor NT HSI). Specification S1 is a tunable parameter that can be changed while the module is executing.



Outputs

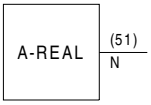
Blk	Type	Description
N	B	Output value equals S1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0	B	0 or 1	Desired output value

51. Manual Set Constant

The manual set constant function code provides an adjustable real value that is not tunable. It performs the same function as function code 2, except that function code 51 is not tunable.



Outputs

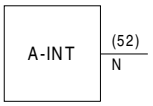
Blk	Type	Description
N	R	Output value equals S1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0.000	R	Full	Desired output value

52. Manual Set Integer

The manual set integer function code provides a tunable integer constant.



Outputs

Blk	Type	Description
N	I	Output value equals S1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0	I	Full	Integer constant specified

52.1 Examples

Function code 52 is commonly used with function codes 13 and 24 to dynamically modify integer parameters. Figure 52-1 is an example of how to use the manual set integer to change the set point tracking option in a station. The value in the remote control memory block (function code 62) determines which input the function code 13 block reads (<S3> equals zero for <S1>, and <S3> equals one for <S2>). The adapt block (function code 24) adapts S30 of the control station (function code 80) to the value received from the integer transfer block (function code 13). When the S3 input (output from the remote control memory) equals zero, the output of the integer transfer block equals its S1 input (one). Thus, S30 of the station adapts to set point track option one (track the process variable).

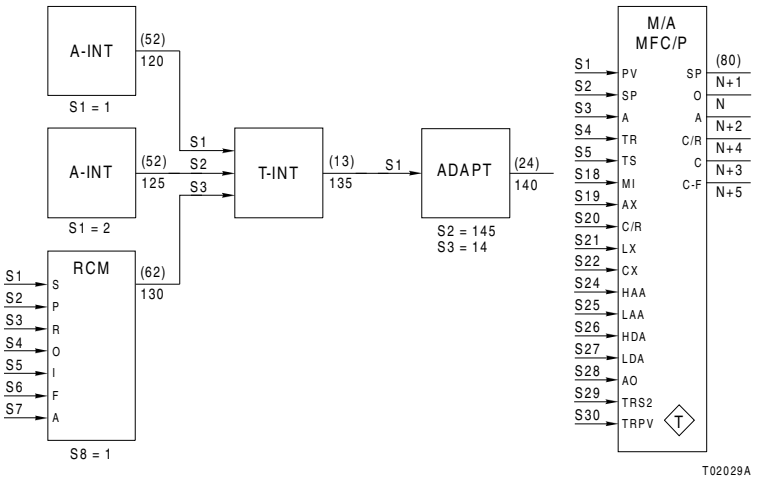
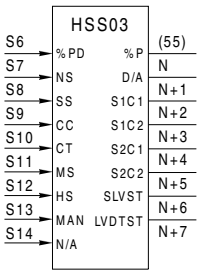


Figure 52-1 Dynamically Modify an Integer Parameter

55. Hydraulic Servo

The hydraulic servo function code defines startup, run-time, calibration, mode select, and failure mode specifications for the IMHSS03 Hydraulic Servo Module. This function code is used as an interface to the BRC-100 and HAC controllers. Individual block addresses are specified by this function code to provide position demand, calibration function and mode select parameters.
The function block has ten outputs. The outputs display valve position, module status, process control status, as well as the status of the components within the turbine control loop such as servo coil status and linear variable differential transformer (LVDT) status.



Outputs

Blk	Type	Description
N	R	Percent or volts actuator position with quality
N+1	R	IMHSS03 D/A converter output with quality (expressed as % of span)
N+2	R	Servo 1 coil 1 output current in % or volts, with quality
N+3	R	Servo 1 coil 2 output current in % or volts, with quality
N+4	R	Servo 2 coil 1 output current in % or volts, with quality
N+5	R	Servo 2 coil 2 output current in % or volts, with quality
N+6	R	Module status
N+7	R	LVDT status
N+8	B	Module hardware status: 0 = good 1 = bad
N+9	B	Module communication and watchdog timer status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Module bus address of module
S2	N	0	I	0 - 5	Output device select: 0 = output 1 active servo output, output 2 unused 1 = output 1 unused, output 2 active servo output 2 = output 1 active servo, output 2 redundant servo output 3 = output 1 active servo, output 2 redundant servo at 20% standby 4 = output 1 active 4-20 mA I/H converter output 5 = output 1 active 20-160 mA I/H converter output
S3	N	0	I	0 - 4	LVDT select: 0 = input 1 LVDT, input 2 unused 1 = input 1 unused, input 2 LVDT 2 = input 1 primary LVDT, input 2 backup LVDT 3 = input 1 backup LVDT, input 2 primary LVDT 4 = input 1 LVDT I/H display only, input 2 4-20 mA contingency error monitor

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S4	N	0	I	0 - 3	Output action on LVDT error: 0 = valve closes to 0% position: 0 V in servo mode 4 mA in 4-20 mA mode 20 mA in 20-160 mA mode 1 = valve opens to 100% position: max V in servo mode 20 mA in 4-20 mA mode 160 mA in 20-160 mA mode 2 = valve remains at current position (I/H mode only) 3 = valve moves to null position (I/H mode only)
S5	N	1.0	R	0.0 or 0.4-15.0	LVDT frequency select in KHz: 0.0 = DC LVDT mode
S6	N	5	I	0 - 9998	Block address of the percent position demand
S7	N	5	I	0 - 9998	Block address of LVDT null check mode select: 0 = do not stop calibration at null point 1 = stop at null for primary LVDT 2 = stop at null for secondary LVDT
S8	N	6	I	0 - 9998	Block address of calibration stroke time select (in secs)
S9	N	6	I	0 - 9998	Block address of calibration cycles count (1.0 to 8.0)
S10	N	5	I	0 - 9998	Block address of calibration type select: 0 = full calibration. Record both 0% and 100% calibration values
S10 *(cont.)*	N	5	I	0 - 9998	1 = 0% calibration. Record only 0% LVDT value 2 = 100% calibration. Record only 100% LVDT value
S11	N	0	I	0 - 9998	Block address of calibrate mode enable: 0 to 1 transition = calibrate
S12	N	0	I	0 - 9998	Block address of calibrate GO/HOLD select: 0 = hold 1 = go
S13	N	0	I	0 - 9998	Block address of hard manual mode select: 0 = auto mode 1 = hard manual mode
S14	N	0	I	0 - 9998	Spare boolean block address parameter
S15	Y	0	I	0 - 3	Block output display units: 0 = output N in %, outputs N+2 to N+5 in % 1 = output N in volts, outputs N+2 to N+5 in % 2 = output N in %, outputs N+2 to N+5 in volts 3 = output N in volts, outputs N+2 to N+5 in volts

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S16	Y	1	I	0 - 31	Demodulator gain constant: 0 = auto-tuning enabled 1 = 2.0 12 = 36.3 23 = 87.7 2 = 4.0 13 = 40.0 24 = 93.6 3 = 5.9 14 = 44.0 25 = 100.9 4 = 7.8 15 = 48.8 26 = 108.0 5 = 10.0 16 = 52.0 27 = 118.0 6 = 11.8 17 = 54.9 28 = 126.2 7 = 16.0 18 = 61.5 29 = 132.0 8 = 21.1 19 = 64.2 30 = 141.3 9 = 24.7 20 = 69.8 31 = 152.0 10 = 28.3 21 = 72.0 11 = 32.0 22 = 78.6
S17	Y	1	I	0 - 31	Control gain proportional constant: 0 or 1 = 1.0 12 = 18.1 23 = 43.8 2 = 2.0 13 = 20.0 24 = 43.8 3 = 2.9 14 = 22.0 25 = 50.4 4 = 3.9 15 = 24.4 26 = 54.0 5 = 5.0 16 = 26.0 27 = 59.4 6 = 5.9 17 = 27.4 28 = 63.1 7 = 8.0 18 = 30.7 29 = 66.0 8 = 10.5 19 = 32.1 30 = 70.6 9 = 12.3 20 = 34.9 31 = 76.0 10 = 14.1 21 = 36.0 11 = 16.0 22 = 39.3
S18	Y	-10.0	R	Full	LVDT 1 differential voltage at 0% actuator position
S19	Y	+10.0	R	Full	LVDT 1 differential voltage at 100% actuator position
S20	Y	-10.0	R	Full	LVDT 2 differential voltage at 0% actuator position
S21	Y	+10.0	R	Full	LVDT 2 differential voltage at 100% actuator position
S22	Y	5.0	R	0 - 100	Contingency deadband in %
S23	Y	0.0	R	Full	Spare real parameter

55.1 Explanation

55.1.1 Outputs

N
Percent actuator position with quality. This output displays actuator position read from the LVDT or position feedback device. Quality will be displayed as bad in the event of an A/D error, a bus transmission error, or an LVDT error (i.e., a primary or secondary error if using one LVDT, or error on both LVDTs when operating with redundant LVDTs). Otherwise, the quality will indicate good.

N+1
IMHSS03 D/A converter output. This output displays the D/A converter output value as a percent of span. This is the position demand signal feedback to generate an error signal. A hardware controller manipulates the error signal to produce the control output signal. This output value can be used as a guide to match to the BRC-100 position demand and bring the IMHSS03 module out of hard manual mode. Quality will be displayed as bad in the event of a D/A, A/D or bus transmission error. Otherwise, the quality will indicate good.

N+2
Servo 1, coil 1 control output. This output displays the IMHSS03 analog control output signal to servo valve 1, coil 1, expressed as a percent of span. Quality will be displayed as bad in the event of an A/D error, bus transmission error, shorted output, or opened output. The output is good quality otherwise.

NOTE: This output block will display a value of zero with good quality when defined as unused by S2.

N+3
Servo 1, coil 2 control output. This output displays the IMHSS03 analog control output signal to servo valve 1, coil 2, expressed as a percent of span. Quality will be displayed as bad in the event of an A/D error, bus transmission error, shorted output, or opened output. The output is good quality otherwise.

NOTE: This output block will display a value of zero with good quality when defined as unused by S2.

N+4
Servo 2, coil 1 control output. This output displays the IMHSS03 analog control output signal to servo valve 2, coil 1, expressed as a percent of span. Quality will be displayed as bad in the event of an A/D error, bus transmission error, shorted output, or opened output. The output is good quality otherwise.

NOTE: This output block will display a value of zero with good quality when defined as unused by S2.

N+5
Servo 2, coil 2 control output. This output displays the IMHSS03 analog control output signal to servo valve 2, coil 2, expressed as a percent of span. Quality will be displayed as bad in the event of an A/D error, bus transmission error, shorted output, or opened output. The output is good quality otherwise.

NOTE: This output block will display a value of zero with good quality when defined as unused by S2.

N+6
Module status is converted into a real output as an integer with the bit map shown in Table 55-7.

Table 55-7 Module Status Bit Map

Bit	Binary Weight	Description
0-1	1	Calibration/automatic tuning command status: 0 = CAL/ATOP command in progress 1 = CAL/ATOP command complete. No command issued 2 = Calibration failure 3 = Demodulator gain auto-tune failure
2	4	Servo outputs in bypass mode
3	8	Module mode: 0 = auto 1 = hard manual
4	16	Contingency/actuator positioning error: 0 = no error 1 = error
5	32	A/D error: 0 = no error 1 = error
6	64	D/A error: 0 = no error 1 = error
7	128	DPRAM error: 0 = no error 1 = error
8-23	—	Reserved

Bits 0-1
Status of calibration or automatic tuning command.

0 = A calibration or automatic tuning is in process.

1 = The command has been correctly completed or the command has never been issued.

2 = An error has occurred during a calibration operation. It indicates one of three causes. The 100 percent voltage value was less than the zero percent voltage value for the LVDTs. An LVDT failure

occurred during calibration. Refer to S17, S18, S19 and S20 for additional information on calibration failures.

3 = A demodulator gain auto-tune failure. If using redundant LVDTs, the LVDTs do not match and cannot be configured for redundant operation. The auto-tuning procedure has picked the best values for the primary LVDT as selected in S3. The backup LVDT is considered failed. Redundant LVDT operation is only available if the backup LVDT is replaced and the auto-tune procedure is performed again.

Bit 2
The servo outputs are bypassed (grounded) due to an I/O error that interferes with positioning. These errors include opened and shorted LVDTs and DA errors.
Bit 3
Module mode. This output indicates the mode of the IMHSS03 module. A value of zero at this bit indicates automatic mode; a one value indicates hard manual mode.
Bit 4
Contingency actuator positioning error. This bit is one if the measured actuator position deviates beyond an established deadband from the position set point established in the IMHSS03 module. Otherwise, this bit value is zero.
Bit 5
A/D error. This bit value is one if the error is associated with the A/D converter on the IMHSS03 module. Otherwise, this bit value is zero.
Bit 6
D/A error. This bit value is one if the error is associated with the D/A converter on the IMHSS03 module. Otherwise, this bit value is zero.
Bit 7
DPRAM error. This bit value is one if a DPRAM error is detected on the IMHSS03 module. Otherwise, this bit value is zero.

N+7
LVDT status. The LVDT status is converted into a real output as an integer with the bit map shown in Table 55-8.

Table 55-8 LVDT Status

Bit	Binary Weight	Description
0	1	Active LVDT or position feedback device: 0 = LVDT 1 1 = LVDT 2
1	2	LVDT 1 primary: 0 = good 1 = bad
2	4	LVDT 1 secondary 1: 0 = good 1 = bad
3	8	LVDT 1 secondary 2: 0 = good 1 = bad
4	16	LVDT 2 primary: 0 = good 1 = bad
5	32	LVDT 2 secondary 1: 0 = good 1 = bad
6	64	LVDT 2 secondary 2: 0 = good 1 = bad

Table 55-8 LVDT Status (Continued)

Bit	Binary Weight	Description
7	128	LVDT at null point: 0 = not at null 1 = at null
8-23	—	Reserved

Bit 0
Active LVDT or position feedback device. This bit indicates which LVDT is being used to determine the percent actuator position. A zero bit value indicates that input one is currently selected; a bit value of one indicates that input two is currently selected. In the event of an error to both LVDTs in a redundant LVDT situation, this output will display the most recently functioning LVDT.
Bit 1
LVDT 1 primary 1 status. This bit value will be one in the event of a primary failure of LVDT 1 (i.e., no signal present on either of the LVDT 1 secondary outputs). Otherwise, this bit value will be zero.
Bit 2
LVDT 1 secondary 1 status. This bit value will be one in the event of an LVDT 1 secondary 1 failure. Otherwise, this bit value will be zero.
Bit 3
LVDT 1 secondary 2 status. This bit value will be one in the event of an LVDT 1 secondary 2 failure. Otherwise, this bit value will be zero.
Bit 4
LVDT 2 primary status. This bit value will be one in the event of a primary failure of LVDT 2 (i.e., no signal present on either of the LVDT 2 secondary outputs). Otherwise, this bit value will be zero.
Bit 5
LVDT 2 secondary 1 status. This bit value will be one in the event of an LVDT 2 secondary 1 failure. Otherwise, this bit value will be zero.
Bit 6
LVDT 2 secondary 2 status. This bit value will be one in the event of an LVDT 2 secondary 2 failure. Otherwise, this bit value will be zero.
Bit 7
LVDT at null point. During calibration, this bit value is one when the 50 percent (null) valve position is reached for the selected LVDT or position feedback device. Otherwise, this bit value is zero. This bit is always a zero value during both turbine control and hard manual modes of operation.

N+8
Module hardware status. This output will display a one if the module has encountered a fatal error and stopped operation. Otherwise, a value of zero is displayed.

N+9
Module communication and watchdog timer status. This output will display a one if the communications between the module and the BRC-100 controller are lost. Otherwise, a value of zero is displayed.

55.1.2 Specifications

S1
Module bus address of the IMHSS03 Hydraulic Servo Module.

S2
Defines the type and configuration of turbine control valve which will be driven by the IMHSS03 module. There are six available options:

0 = One hydraulic servo valve connected to output 1. Servo 2 output currents (N+4 and N+5) will display a value of zero with good quality.

1 = One hydraulic servo valve connected to output 2. Servo 1 output currents (N+2 and N+3) will display a value of zero with good quality.

2 = Redundant hydraulic servo valves used, both with active control signals.

3 = Redundant hydraulic servo valves used, output 1 as primary and output 2 as secondary (20 percent standby signal).

4 = I/H converter mode (four to 20 milliamps) connected to output 1. Servo 2 output currents (N+4 and N+5) and servo 1 coil 2 output current (N+3) will display a value of zero with good quality.

5 = I/H converter mode (20 to 160 milliamps) connected to output 1. Servo 2 output currents (N+4 and N+5) and servo 1 coil 2 output current (N+3) will display a value of zero with good quality.

S3
Selects the position feedback input configuration to be used. Table 55-9 describes the available options.

Table 55-9 Position Feedback Options

Spec Value	Input 1	Input 2
0	LVDT	No connection
1	No connection	LVDT
2	Primary LVDT	Backup LVDT
3	Backup LVDT	Primary LVDT
4	LVDT (display only in I/H mode)	4-20 mA transmitter (contingency error only in I/H mode)

S4
Defines the action to be taken in the event of total LVDT failure. Total LVDT failure is defined as a failure of one LVDT in a single LVDT configuration or both LVDTs in a redundant LVDT configuration.
If operating in the servo valve mode, this specification can be set to ground (outputs disabled) or output the maximum signal (100 percent) to the outputs in the event of a total LVDT failure.

NOTE: Control of the valve position is impossible without position feedback information.

The outputs are either grounded or driven to 100 percent to avoid large swings in the valve position due to control loss. The adjustment of the servo valve spool determines the speed in which the servo valve closes or opens. If 100 percent is selected, the outputs disabled light does not illuminate in the event of an error.
If a total failure of the LVDT occurs, the output is driven based upon selection made in this specification. The options are as follows:

0 = Valve closes to a zero percent position. Grounded outputs in the servo valve mode are four milliamps in the four to 20 milliamp mode, or 20 milliamps in the 20 to 160 milliamp mode.

1 = Valve opens to 100 percent position. Outputs to maximum in the servo valve mode are 20 milliamps in the four to 20 milliamp mode, or 160 milliamps in the 20 to 160 milliamp mode.

2 = Valve remains at its current position. Control maintained. I/H mode only.

3 = Valve moves to a null position. I/H mode only.

In either mode, if the LVDT error is a failure of one secondary, the module can be calibrated to return to normal operation.

S5
LVDT frequency selected (in kilohertz) from 0.4 kilohertz to 15.0 kilohertz. If a DC LVDT is used, a value of zero must be entered for this specification.

S6
Block address of the percent position demand requested. Quality of the position demand block is ignored.

S7
Block address of the LVDT null mode selected. When this specification is set to one or two and calibration is enabled, the valve ramps to and holds at the LVDT null point for the primary or backup LVDT, respectively.

0 = Do not stop calibration at LVDT null point.

1 = Stop calibration operation at the null point of the primary LVDT.

2 = Stop calibration operation at the null point of the backup LVDT.

In servo valve mode, the LVDT null point is defined to be the valve position where the LVDT secondaries have equal voltage, usually close to a 50 percent valve position. In the I/H converter mode, the LVDT null point is the 50 percent output value (i.e., 12 milliamps for the four to 20 milliamp mode and 90 milliamps for the 20 to 160 milliamp mode).

S8
Block address of the calibration stroke time. This specification indicates the time in seconds for the valve to be driven from a zero percent actuator position to a 100 percent actuator position. Stroke times normally used for calibration are 30 seconds, 60 seconds, 35 minutes (2,100 seconds) and 70 minutes (4,200 seconds). Minimum stroke time is 30 seconds.

S9
Block address of the calibration cycles count. This specification determines the number of calibration cycles to be performed during the calibration operation. The calibration operation can perform from one to eight cycles (1.0 to 8.0).

S10
Block address of the calibration type select. This specification selects whether a full calibration will be performed or only the 100 percent LVDT voltage or zero percent LVDT voltage will be recorded.

0 = Full calibration. Record both zero and 100 percent values.

1 = Zero percent calibration. Record only the zero percent LVDT value.

2 = 100 percent calibration. Record only the 100 percent LVDT value.

S11
Block address of the calibrate mode enable. A transition of this input block value from a zero to a one begins the calibration process.

0 = Do not calibrate or terminate calibration.
1 = Begin calibration (when tuned from a zero to a one).

S12
Block address of the calibrate GO/HOLD select.

0 = Hold. Pause calibration execution.
1 = Go. Execute calibration procedure.

S13
Block address of the hard manual mode select. Hard manual mode can be forced by setting this specification to a one. When this specification is reset to a zero, the module will return to automatic operation if the 0 - 9998 position demand (S6) equals IMHSS03 D/A output value (N+1).

S14
Spare boolean block address parameter.

S15
Selects whether the LVDT feedback (output block N) is displayed as a voltage or a percentage.

0 = Output N in percent, outputs N+2 to N+5 in percent
1 = Output N in volts, outputs N+2 to N+5 in percent
2 = Output N in percent, outputs N+2 to N+5 in volts
3 = Output N in volts, outputs N+2 to N+5 in volts

S16
Defines the demodulator gain constant used by the IMHSS03 module. This value can be zero or an integer from one to 31. If zero is selected, the IMHSS03 module automatically selects the optimal demodulator gain value based on the highest non-saturated demodulator gain value measured on the IMHSS03 module. If this option is selected, the optimal demodulator gain value is automatically written into S16.
A non-zero number in this specification directs the IMHSS03 module to use the corresponding table value as the demodulator gain constant and the automatic tuning operation is not performed. Refer to the specifications table for the list of available gain constant values.

NOTE: If a change is made to the demodulator gain value, the proportional gain values must be adjusted (i.e., the user-selectable proportional gain values). Afterwards, an LVDT calibration must be performed.

S17
Defines the proportional gain constant used by the IMHSS03 module. The value shown in the specifications table is used as the proportional gain constant.

S18
LVDT differential voltage at zero percent actuator position for LVDT 1. This value is set by the operator or can be overwritten for both LVDTs automatically by performing a calibration operation.

S19
LVDT differential voltage at 100 percent actuator position for LVDT 1. This value is set by the operator or can be overwritten for both LVDTs automatically by performing a calibration operation.

S20
LVDT differential voltage at zero percent actuator position for LVDT 2. This value is set by the operator or can be overwritten for both LVDTs automatically by performing a calibration operation.

S21
LVDT differential voltage at 100 percent actuator position for LVDT 2. This value is set by the operator or can be overwritten for both LVDTs automatically by performing a calibration operation.

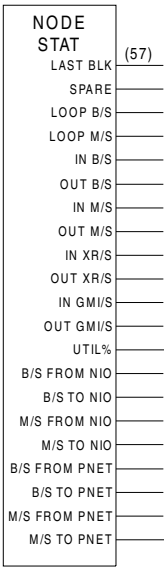
S22
Contingency deadband in percent. The contingency error bit indicates an error when the valve cannot be driven to the position demand plus or minus this contingency deadband.

S23
Spare real parameter.

57. Node Statistics Block

The node statistics block combines the functionality of node performance statistic outputs and last block along with extended executive block functions. The function code is part of the HAC controller default block configuration at blocks 31,999 through 32,025. The terminating executive function code replaces the last block function code 89 in the default configuration. The output indicates the options configuration including the module network type, time-stamping, I/O Hnet cable length, peer-to-peer Hnet cable length, and revision checking. The node performance statistics are included as the block outputs of this function code.

NOTE: This function code is supported only on the HAC controller.



Outputs

Blk	Type	Description
31,999	R	Options configuration Network type XXX3 = Cnet with time-stamping I/O Hnet cable length XX0X = 4000m XX1X = 9000m XX2X = 7000m XX3X = 5000m XX4X = 2600m XX5X = 1600m XX6X= 1300m XX7X= 1100m Peer-to-peer Hnet cable length X0XX = 4000m X1XX = 9000m X2XX = 7000m X3XX = 5000m X4XX = 2600m X5XX = 1600m X6XX = 1300m X7XX = 1100m Revision check 0XXX = Revision check disabled 1XXX = Revision check enabled
32,000	R	Reserved
32,001	R	Total number of bytes transmitted by the node per second
32,002	R	Total number of messages transmitted by the node per second.
32,003	R	Total number of bytes received by the node per second.
32,004	R	Total number of bytes sent by the node per second.
32,005	R	Total number of messages received by the node per second.
32,006	R	Total number of messages sent by the node per second.
32,007	R	Total number of exception reports received by the node per second.
32,008	R	Total number of exception reports sent by the node per second.
32,009	R	Total number of GMI messages received by the node per second.
32,010	R	Total number of GMI messages sent by the node per second.

Outputs (Continued)

Blk	Type	Description
32,011	R	Percentage of node's module processing power currently being used.
32,012	R	Number of bytes transferred per second on the expander bus from the NIO to the communication module.
32,013	R	Number of bytes transferred per second on the expander bus from the communication module to the NIO.
32,014	R	Number of messages transferred per second on the expander bus from the NIO to the communication module.
32,015	R	Number of messages transferred per second on the expander bus from the communication module to the NIO.
32,016	R	Reserved
32,017	R	Reserved
32,018	R	Reserved
32,019	R	Reserved
32,020	R	Reserved
32,021	R	Reserved
32,022	R	Reserved
32,023	R	Reserved
32,024	R	Reserved
32,025	R	Reserved

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	NULL	E90 String	0 - 32 chars	Reserved for future use
S2[1]	N	20000	R	10,008 - 400,000	Amount of memory (in bytes) allocated for XR routes connected to this module and the number of HCUs on the system.
S3	Y	0.0	R	FULL	Reserved for future use
S4	Y	0.0	R	FULL	Reserved for future use
S5	Y	0.0	R	FULL	Reserved for future use
S6	Y	0.0	R	FULL	Reserved for future use
S7	Y	0	INT	FULL	Reserved for future use
S8[2]	N	0	INT	0 - 4	Maximum number of destination nodes connected through the peer-to-peer network. A value of zero disables the peer-to-peer channel status reporting for stand-alone HAC units. A value of zero has no effect on the peer-to-peer channel status reporting of redundant HAC units.
S9	N	0.0	R	FULL	Reserved for future use

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S10	N	0.0	R	FULL	Reserved for future use

NOTE:

1. The amount of memory needed = (18 x number of export routes) + (18 x number of HCUs on the system).
2. Setting this value too low results in the module entering error mode with a configuration error (type 0x6 - data type conflict).

57.1 Explanations

57.1.1 Specifications

S1
Reserved for future use.

S2
The amount of memory (in bytes) allocated for XR routes connected to this module and the number of HCUs in the system. Failure to allocate sufficient memory will prevent the establishment of some XR routes.

S3-S7
Reserved for future use.

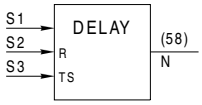
S8
Defines the maximum number of HAC nodes that can be referenced by peer-to-peer network function blocks (FC 25, FC 41, FC 63, FC 64, FC 95) in the HAC function block configuration. Setting this value too low results in the module entering error mode with a configuration error (type 0x6 - data type conflict).

NOTE: When utilizing FC 95 to monitor the module status of the backup HAC (FC 95, S4 = 1X), S8 of FC 57 must be set to account for the backup as a separate node.

S9-S10
Reserved for future use.

58. Time Delay (Analog)

The time delay function code provides a pure delay on an analog signal. It can be used to create fixed or variable time delays, or model systems that represent dynamic time delays.



Outputs

Blk	Type	Description
N	R	Time delayed function of input

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input (X)
S2	N	6	I	Note 1	Block address of rate input (R, in units/sec)
S3	N	1	I	Note 1	Block address of track switch signal: 0 = track 1 = release
S4	N	0.000	R	Full	Length of queue (L, in units)
S5	N	1	I	1 - 32767 Note 2	Number of intervals (N)

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC.
2. Excludes COMxx modules. The range for these modules is 1-255.

58.1 Explanation

58.1.1 Specifications

S1 – <X>
Block address of the input.

S2 – <R>
Block address of rate input in units per second.

S3 – <T>
Block address of track switch signal.

0 = track input
1 = release

When <S3> = 0, it initializes all *n* elements of the memory to the input value:

Then $M_1 = M_2 = ...M_n$ = <S1>, and elapsed time since the last sample = 0.

There is no time delay.

If <S3> = 1, then:

$$\textit{Time Delay} = \frac{S4}{\langle S2 \rangle}$$

$$\textit{Time Interval (TI)} = \frac{\textit{Time Delay}}{S5}$$

$$\textit{Elapsed Time} = ET + t$$

where:

ET	Time since last sample in seconds

t	Time since last algorithm execution in seconds
TI	Internal input sample time in seconds
Mi	Memory locations where input values are stored during the time delay

If ET ≥ TI:

$$M_i = M_i + 1, \text{ for } i = 0 \text{ to } n$$

$$M_n = \text{output value, and } M_1 = \langle S1 \rangle$$

$$\text{Elapsed time} = ET - TI$$

S4 – L
(Length of queue) Length of the queue in units. The queue is the number of units over which the time delay is effective.

S5 – N
(Number of intervals) Number of times, from one to 190, that the input is to be sampled. Determine N by dividing the time delay (TD) by the desired sampling frequency.

58.2 Applications

58.2.1 Fixed Time Delay

For a fixed time delay, the rate input, <S2>, is constant. The time delay between output and input varies only with S4. It is directly proportional to S4. For example, simulate the time delay for flow through a pipe. Assume a required time delay of two minutes with input sampling desired every five seconds. Select the default value of 1.0 (found in fixed block six) for <S2> since rate is constant for fixed delays.

<S2>	=	Rate in units per second = 1.0
S4	=	Length of the queue in units
S5	=	Number of intervals
Time Delay	=	2 minutes = 120 seconds
TD	=	120 seconds
TD	=	$\frac{S4}{\langle S2 \rangle}$
120	=	$\frac{S4}{1.0}$
S4	=	120 units = Length of queue

For input sampling every five seconds:

$$N = \frac{TD}{5\ sec}$$

$$= \frac{120\ sec}{5\ sec}$$

$$= 24\ intervals$$

Figure 58-1 is an illustration of this example.

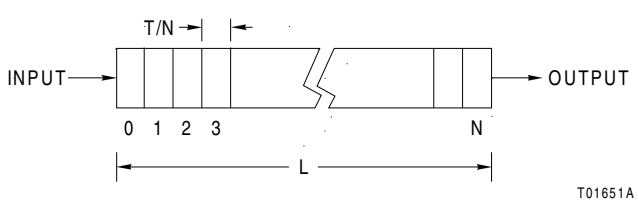


Figure 58-1 Fixed Time Delay Example

58.2.2 Variable Time Delay

Variable time delays may be dynamically adjusted by changing the value of <S2>. Using a function code 9 block, as illustrated in Figure 58-1, the two fixed input rates can be switched. In the fixed time delay example, when <S2> equals 1.0, the time delay, S4/<S2> equals 120 seconds. By changing <S2> to 2.0, the time delay becomes 60 seconds, and the timing interval, TD/N equals 2.5 seconds. Changing the rate input <S2> while holding all other parameters constant changes the

timing interval. Faster rates produce more frequent input sampling, and slower rates produce less frequent input sampling for the same number of intervals.

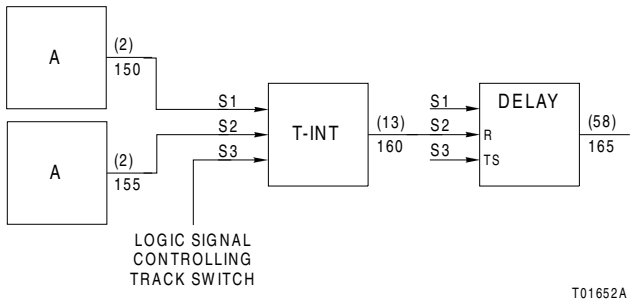


Figure 58-1 Variable Time Delay

58.2.3 System Modeling

The analog time delay block may be used to model a physical system that represents a dynamic time delay. For example, an oil pipeline may have a measurement device at a different location than the indicator/controller. With this function code, a measurement can be taken. This function code delays sending the value to the controller until the element of oil reaches the controller. Specification 4 may be specified in feet, <S2> in feet per second, and N to establish the needed resolution. If S4 = 100 feet ***and*** <S2> varies from ten feet per second to 20 feet per second, then TD will vary between ten seconds and five seconds.
If sampling is required every 0.5 seconds to achieve the needed resolution, then:

$$N = \frac{\textit{longest time delay}}{0.5\ \textit{sec}}$$

$$= \frac{10\ \textit{sec}}{0.5\ \textit{sec}}$$

$$= 20\ \textit{intervals}$$

The longest time delay can present a worst case scenario. No matter what the time delay, the input will be sampled 20 times over that period to insure adequate resolution.

For the shortest time delay, the sampling intervals will be:

$$\frac{5\ \textit{sec}}{20\ \textit{intervals}} = 0.25\ \textit{second}$$

In most cases, the delay in a process consists of more than a pure time delay (deadtime). There is usually an additional time lag that may be a first, second, or higher order lag. In general, the process responds to a second order lag response. This can be simulated accurately by using a time delay and a first order lag. If necessary, another first order lag function block can be added. Figure 58-1 shows a graphic representation of a function and a simulated response. Figure 58-1 illustrates the configuration required to simulate the response shown in Figure 58-1.

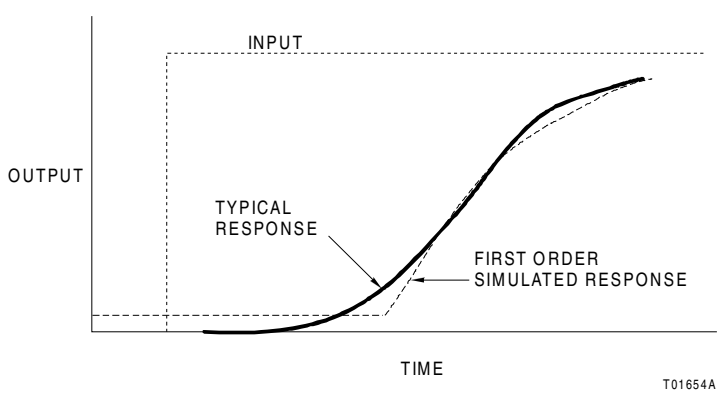


Figure 58-1 First Order Simulated Response to Typical Time Delay

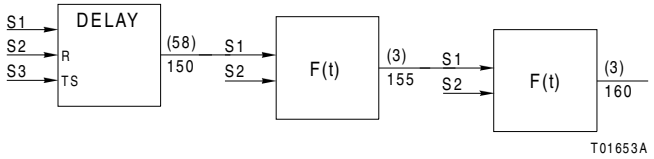
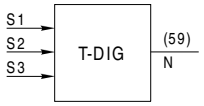


Figure 58-1 Response Simulation Configuration

59. Digital Transfer

This function code selects one of two digital inputs, depending on a transfer switch signal. The state of the transfer switch <S3> is determined externally, either by automatic control or by an operator.



Outputs

Blk	Type	Description
N	B	Value of selected input

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of first input
S2	N	0	I	Note 1	Block address of second input
S3	N	0	I	Note 1	Block address of track switch signal: 0 = <S1> 1 = <S2>

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

59.1 Applications

Function code 59 can control equipment based on the status of other equipment or digital inputs. Figure 59-1 illustrates a safety feature.

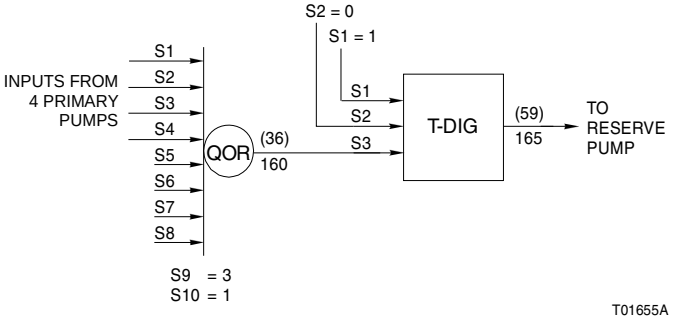


Figure 59-1 Safety Configuration

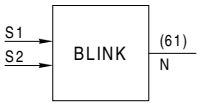
The controlled reserve pump must activate if less than three of four primary pumps are functioning. The function code 59 block controls the status of the reserve pump.

<S1> = 1 (on), and <S2> = 0 (off)

The transfer switch is the output of a qualified OR block. If less than three pumps are operating, then the output of the qualified OR block will be at zero. That output is the <S3> input to the digital transfer block. A zero input for <S3> causes <S1> to be the output from the block. When <S1> equals one (on), the pump activates. When three or more of the pumps are operational, the pump will be turned off.

61. Blink

The blink function code generates a pulsating output signal. When <S1> and <S2> are logic 1, the output toggles between logic 1 and logic 0. The duration of either the logic 0 or logic 1 state is dependent on the cycle time of the block. The duration of either logic state is limited to be no less than 0.2 second. Output N equals <S1> until <S1> and <S2> equal one, then output N blinks. Refer to Table 61-10 for all possible output values.



Outputs

Blk	Type	Description
N	B	Refer to Table 61-10

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of first input
S2	N	0	I	Note 1	Block address of second input

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

Table 61-10 Truth Table for Output Values

Inputs		Output N
S1	S2	
0	0	0
0	1	0
1	0	1
1	1	Blink

61.1 Applications

Figure 61-1 shows how to use function code 61 for alarm indication. The source receives the alarm signal and sends the signal to a blink block and a timer block. The timer block outputs a boolean signal. In this example, the timer is the timed out option. If the length of the input pulse is greater than or equal to the timing interval (S3 of function code 35), a logic 1 will be output for the length of the pulse once it has exceeded the timing interval. If the length of the pulse is shorter than the timing interval, a logic 0 will be the output. The output of the timer block is the <S1> input for a memory block. An external alarm acknowledgment signal is the <S2> input to the memory block. The acknowledge signal is logic 0 if the alarm has not been acknowledged and logic 1 if it has.

If the point is in alarm, a logic 1 signal will be sent to both the blink and timer blocks. The duration of the logic 1 signal is longer than the timing interval, so the output of the timer block is a logic 1. If the alarm has not been acknowledged, then the S2 input to the memory block will be a logic 0. When <S1> equals logic 1 and <S2> equals logic 0, the output of the memory block is a logic 1. Thus, both inputs to the blink block are logic 1 and the output will toggle between logic 1 and logic 0. Once the alarm is acknowledged, the output of the memory block goes to logic 0, causing the output of the blink block to track the <S1> input. When the point comes out of alarm, both inputs to the blink block will be logic 0, and the output will be logic 0.

The OR (function code 39) forces the blink action to remain active and in alarm until the alarm is acknowledged whether or not the alarm signal is still present. If the signal is still in alarm when acknowledged, the value remains alarmed but the blinking action stops.

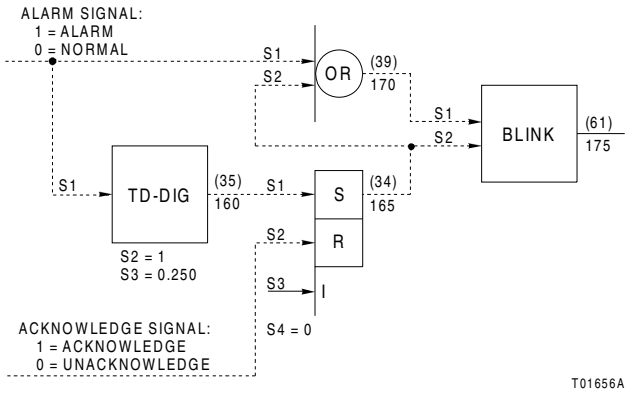
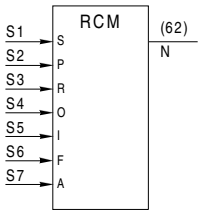


Figure 61-1 Alarm Indication

62. Remote Control Memory

The remote control memory (RCM) function code is a set/reset flip-flop memory accessible by a console, control station, Batch 90 or computer via a network interface unit (NIU).
The output is determined from the inputs S, R and P as shown in Table 62-1 and Figure 62-1.

Table 62-1 RCM Input to Output Relationship



S	R	P	Output N
0	0	X	Last
1	0	1	1
0	1	X	0
1	1	1	Override <S4>

NOTE:
S = Local set <S1>, or remote set (console or network interface unit)
R = Local reset <S3>, or remote reset (console or network interface unit)
P = Set permissive <S2>
X = Either logic 0 or logic 1 (does not care)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of local set signal
S2	N	1	I	Note 1	Block address of set permissive signal: 0 = no permissive 1 = permissive
S3	N	0	I	Note 1	Block address of local reset signal
S4	N	0	I	Note 1	Block address of override signal: 0 = reset override 1 = set override
S5	N	0	I	Note 1	Block address of initialize signal: 0 = reset 1 = set
S6	N	0	I	Note 1	Block address of feedback signal
S7	N	0	I	Note 1	Block address of alarm signal
S8	N	0	I	0 - 255	Type parameter

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

62.1 Explanations

62.1.1 Specifications

S1 – SET
Block address of the local set input.

S2 – PERM
Block address of the set permissive input. Specification <S2> must equal logic 1 for the remote control memory block to act on any set, local or remote input.

S3 – RES
Block address of local reset input.

S4 – OVR
Block address of override input. If <S1>, <S2> and <S3> equal logic 1, the output tracks <S4>.

S5 – INIT
Block address of the input referenced upon power up or resetting of the module.

If <S5> = 0, N = 0
If <S5> = 1, N = 1

S6 – FB
Block address of the feedback signal. It is affected by the output of the RCM block and transmits a status signal to the console or network interface unit. It can be feedback from an internal or external logic input.

S7 – ALRM
Block address of the alarm input. This transmits a status signal to the console or network interface unit. Logic 1 is an alarm state.

S8 – TYPE
Switch type parameter. With S6 and S7, it defines the configuration of pushbutton displays shown on display screens.

0 = output indicator □ ■
1 = no indicator
2 = output and feedback indicators ◎ ◀ ▭ ◁
3 = feedback indicator only ◁ ◀

62.1.2 Control Station Control

The operator can initiate two types of pushbutton commands from a control station: ***pulse commands*** and ***sustained commands***. The commands are either ***pulse set*** and ***pulse reset***, or ***sustain set*** and ***sustain reset***. The type of commands selected depends on the configuration of the control station.
There are two types of inputs to the block: logic set/reset inputs and the operator initiated remote set/reset inputs from the console. Commands from a control station are logic commands because they enter the RCM block from other blocks in the module. Commands from a console are not logic commands because they enter the RCM block directly from the console. Logic inputs from control stations or configuration override operator initiated inputs from a console. The module acts on the remote (from operator interface station, etc.) and logic (<S1>, <S3>) commands as shown by the equivalent circuit in Figure

62-1, and outputs the proper value. Logic commands always override remote commands, and sustain commands override pulse commands.

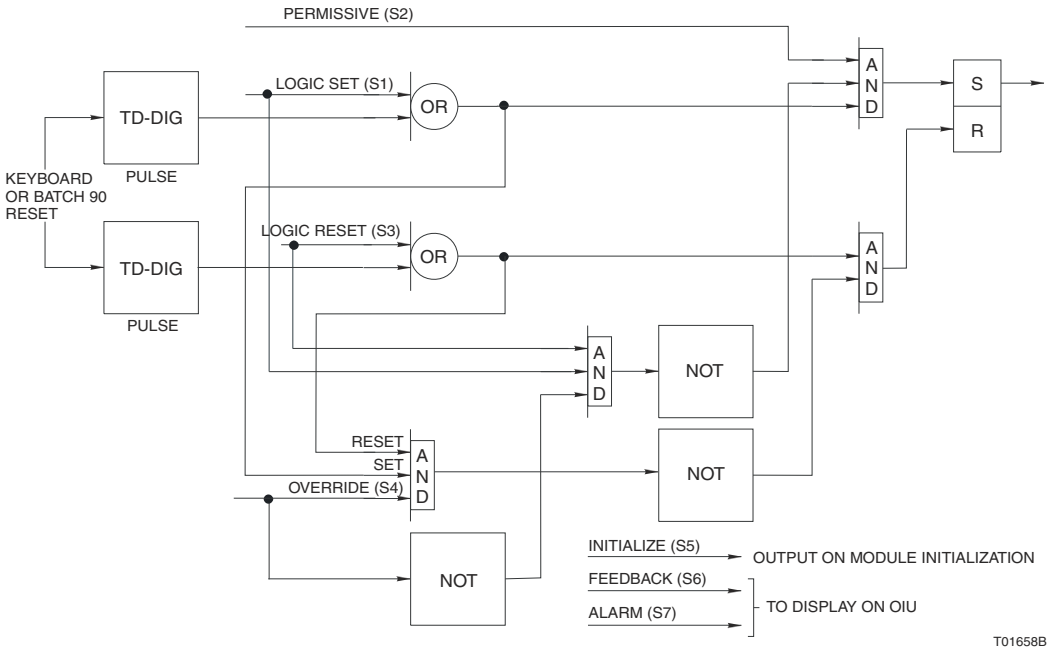


Figure 62-1 Internal Logic of RCM Block

Pulse Set
Causes the internal set signal to go to logic 1 for one cycle, provided there is no contradictory logic or sustain command issued.

Pulse Reset
Causes the internal reset signal to go to logic 1 for one cycle, provided there is no contradictory logic or sustain command issued.

Sustain Set
Causes the internal set signal to go to logic 1 and remain there as long as the sustain set is in effect or until a contradictory logic command is issued.

Sustain Reset
Causes the internal reset signal to go to logic 1 and remain there as long as the sustain reset is in effect or until a contradictory logic command is issued.

Red Tag
Commands flag controls that are under maintenance. A red tag, remove red tag, or get red tag status can be requested. All red tag commands have a 16 bit non-zero key as an identifier. The module maintains up to three keys for every pushbutton block, allowing the establishment of up to three red tags. The red tag functions act only as labels, and do not interfere with module operations. The red tag function does not provide positive lock-out of equipment operation. The red tag function only inhibits control at the console when it inhibits operator commands.

62.2 Applications

Figure 62-1 illustrates the module logic and circuits required to control a motor with a function code 62 block. When a start signal is received (either logic or remote), the circuit is completed, energizing the motor start relay and closing the normally

open seal switch. When the start signal returns to logic 0 after one cycle, the closed switch completes the circuit keeping the motor turned on until a stop signal breaks the circuit.

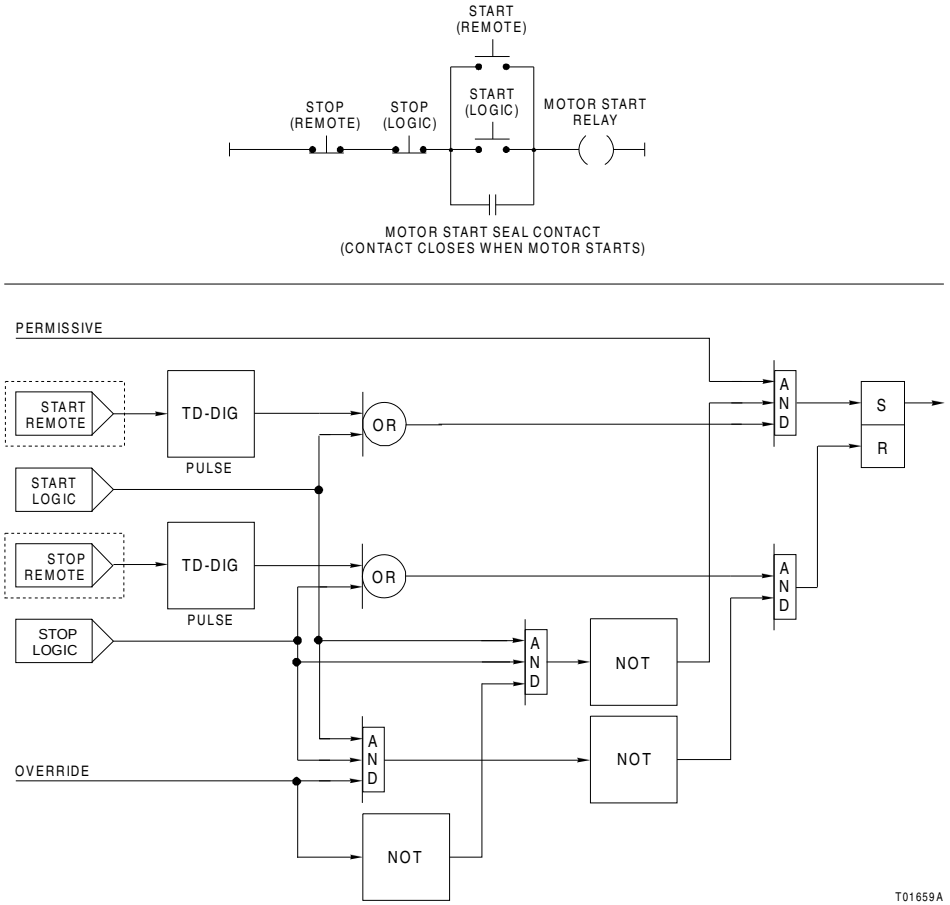


Figure 62-1 Logic and Circuitry of RCM Block Used for Motor Control

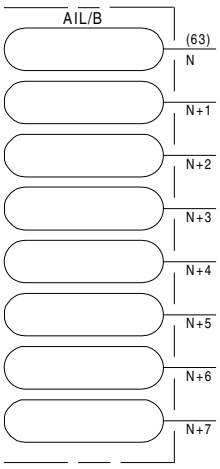
63. Analog Input List (Periodic Sample)

In the BRC-100 controller, the analog input list (periodic sample) function code acquires the data values of eight analog points in a single Controlway/module bus message. The source module and destination module must be located in the same process control unit. Input data consists of analog signals and qualities. Data transfer between the source module and destination module is via the module bus. The address and block number of the source module and the update time are configurable parameters.

The HAC controller uses function code 63 to acquire the data values of eight analog points in a single peer-to-peer network message. Input data consists of analog signals and qualities. Data transfer between the source module and destination module is via the peer-to-peer network. The address and block number of the source node and the update time are configurable parameters.

Output data consists of real numbers corresponding to the analog data received, and point quality indicators. To insure that the signal is successfully transferred, the analog signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The quality of the point cannot be used as an input to any other type of block. The analog output value can be used as an input to any analog processing block. Refer to Appendix H, for a definition of point quality.

NOTE: A block defined as an analog input reads a value for any existing analog output from a function block configured in the source node/module. No additional configuration in the source node/module is required.



Outputs

Blk	Type	Description
N	R	<S3>
N+1	R	<S4>
N+2	R	<S5>
N+3	R	<S6>
N+4	R	<S7>
N+5	R	<S8>
N+6	R	<S9>
N+7	R	<S10>

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	1.0	R	Full	Sample period (in seconds)
S2	N	0	I	Note 1	Source node/module address
S3	N	Note 2	I	Note 3	Source block address (N)
S4	N	Note 2	I	Note 3	Source block address (N+1)
S5	N	Note 2	I	Note 3	Source block address (N+2)
S6	N	Note 2	I	Note 3	Source block address (N+3)
S7	N	Note 2	I	Note 3	Source block address (N+4)
S8	N	Note 2	I	Note 3	Source block address (N+5)
S9	N	Note 2	I	Note 3	Source block address (N+6)

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S10	N	Note 2	I	Note 3	Source block address (N+7)

NOTES:
1. Range values are: 0-31 for the BRC-100, IMMFP11/12
   1-250 for the HAC
2. Default values are:9,999 for the BRC-100, IMMFP11/12
   2 for the HAC
3. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

63.1 Explanation

63.1.1 Specifications

S1 – MBUPD
(Sample period) Defines the update rate for the Controlway/module bus or peer-to-peer network message inputs. Sample period is specified in seconds. Specification S1 is tunable. The system allows tuning the value shown for the sample period, however, the original sample period will be retained. To change it the module must be placed in configure mode.

S2 – SMAD
(Source node/module address) In the BRC-100 controller, the address of the node/module containing the eight values desired. The address of the source module must be between zero and 31 inclusive.
In the HAC controller, S2 is the Cnet address of the HAC controller node that this function code is sampling data from via the peer-to-peer network. This node address must be between 1-250.

S3
Block address for output block N.

S4
Block address for output block N+1.

S5
Block address for output block N+2.

S6
Block address for output block N+3.

S7
Block address for output block N+4.

S8
Block address for output block N+5.

S9
Block address for output block N+6.

S10
Block address for output block N+7.

64. Digital Input List (Periodic Sample)

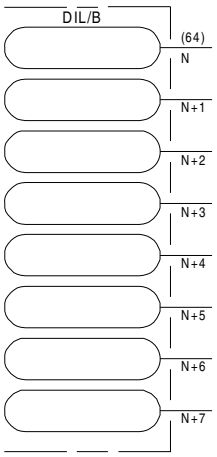
The digital input list (periodic sample) function code acquires data describing eight digital points in a single message. The source module and destination module must be located in the same process control unit. Input data consists of digital signals and qualities. Data transfer between the source and destination module is via the Controlway/module bus. The address and block number of the source module, and the update time are configurable parameters.

The HAC controller uses function code 64 to acquire the data values of eight digital points in a single peer-to-peer network message. Input data consists of analog signals and quantities. Data transfer between the source module and destination module is via the peer-to-peer network. The address block number of the source node and the sample time are configurable parameters.

Output data consists of boolean numbers corresponding to the digital data received and input point quality indicators. To insure that the signal is successfully transferred, the digital signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The quality of the point cannot be used as an input to any other type of block. The digital output value can be used as an input to any digital processing block. Refer to Appendix H, for a definition of point quality.

NOTE: A block defined as an analog input reads a value for any existing analog output from a function block configured in the source node/module. No additional configuration in the source node/module is required.

Outputs



Blk	Type	Description
N	B	<S3>
N+1	B	<S4>
N+2	B	<S5>
N+3	B	<S6>
N+4	B	<S7>
N+5	B	<S8>
N+6	B	<S9>
N+7	B	<S10>

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	1.0	R	Full	Sample period (in seconds)
S2	N	0	I	Note 1	Source node/module address
S3	N	Note 2	I	Note 3	Source block address (N)
S4	N	Note 2	I	Note 3	Source block address (N+1)
S5	N	Note 2	I	Note 3	Source block address (N+2)
S6	N	Note 2	I	Note 3	Source block address (N+3)
S7	N	Note 2	I	Note 3	Source block address (N+4)
S8	N	Note 2	I	Note 3	Source block address (N+5)
S9	N	Note 2	I	Note 3	Source block address (N+6)

Specifications

Spec	Tune	Default	Type	Range	Description
S10	N	Note 2	I	Note 3	Source block address (N+7)

NOTES:
1. Range values are: 0-31 for the BRC-100, IMMFP11/12
   1-250 for the HAC
2. The default values are: 9,999 for the BRC-100, IMMFP11/12
   2 for the HAC
3. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

64.1 Explanation

64.1.1 Specifications

S1 – MBUPD
(Sample period) Defines the update rate for the Controlway/module bus or peer-to-peer network message inputs. The sample period is specified in seconds. The system allows tuning of the value shown for the sample period. However, the original sample period will be retained. To change it, the module must be placed in configure mode.

S2 – SMAD
(Source node/module address) In the BRC-100 controller**,** the address of the module in the same PCU containing eight values desired. The address of the source module must be between zero and 31.
In the HAC controller, S2 is the Cnet node address of the HACO1 controller node that this function code is sampling data from via the peer-to-peer network. This node address must be between one and 250.

S3
Block address for output block N.

S4
Block address for output block N+1.

S5
Block address for output block N+2.

S6
Block address for output block N+3.

S7
Block address for output block N+4.

S8
Block address for output block N+5.

S9
Block address for output block N+6.

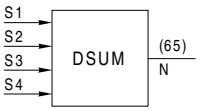
S10
Block address for output block N+7.

65. Digital Sum With Gain

This function code computes a weighted sum of four boolean inputs using the following equation:

Output = *S5<S1>* + *S6<S2>* + *S7<S3>* + *S8<S4>*

It can be used to initiate a control action based on the number of boolean inputs that have a status of logic 1. These inputs could represent the status of pumps, valves, motors, etc.



Outputs

Blk	Type	Description
N	R	S5<S1> + S6<S2> + S7<S3> + S8<S4>

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of <S1>
S2	N	0	I	Note 1	Block address of <S2>
S3	N	0	I	Note 1	Block address of <S3>
S4	N	0	I	Note 1	Block address of <S4>
S5	Y	1.000	R	Full	Gain parameter for <S1>
S6	Y	1.000	R	Full	Gain parameter for <S2>
S7	Y	1.000	R	Full	Gain parameter for <S3>
S8	Y	1.000	R	Full	Gain parameter for <S4>

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

65.1 Applications

Figure 65-1 shows how to use function code 65 to determine flow rates from a digital indication of pump status. In the example, each operating pump provides a constant flow rate of 20 gallons per minute. An operating pump provides an output of logic 1. Specifications <S1> through <S4> provide the pump status inputs while S5 through S8 are pump flow rates. When pumps one, three and four are operating, the output is as follows.

Output = *S5 <S1>* + *S6 <S2>* + *S7 <S3>* + *S8 <S4>*

$$= \frac{20\ gal\ (1)}{min} + \frac{20\ gal\ (0)}{min} + \frac{20\ gal\ (1)}{min} + \frac{20\ gal\ (1)}{min}$$

$$= \frac{60\ gal}{min}$$

Function code 65 can also be used for binary to real conversion. Binary to real conversion changes digital signals to analog signals (i.e., counters). Specifications <S1> through <S4> provide the binary inputs. Specifications S5 through S8 weight the inputs to achieve the desired real output. For example:

<S1> and <S4> = logic 1

<S2> and <S3> = logic 0

S5 = 1.0

S6 = 2.0

S7 = 4.0

S8 = 8.0

Output= S5 <S1> + S6 <S2> + S7 <S3> + S8 <S4>

= 1.0(1) + 2.0(0) + 4.0(0) + 8.0(1)

= 9.0

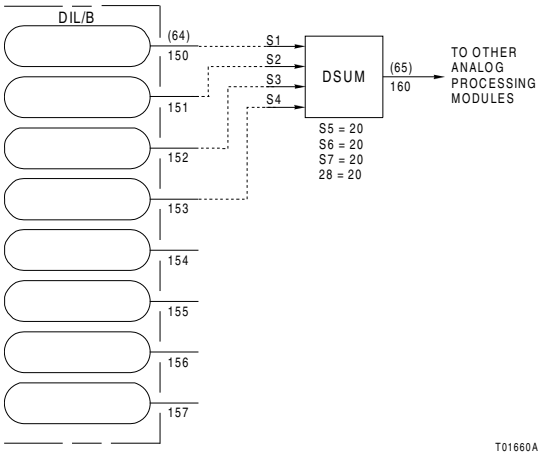
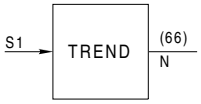


Figure 65-1 Determine Flow Rates From a Digital Indication of Pump Status

66. Analog Trend

The analog trend function code is part of the Symphony distributed trending package. It performs initial data compression for operator consoles and open access systems at the module level.
Trend data can be gathered in two ways, normal or fast, depending on the desired resolution. Specification S3 selects the trend resolution. Normal trending mode gathers trend data every 60 seconds. Fast trending is every 15 seconds.
Specification S2 selects the trend mode. Over the collection period each value collected is the sample, mean, minimum, maximum or sum of the input value <S1>.
Output data consists of the last value saved.



Outputs

Blk	Type	Description
N	R	Last value saved

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of analog input
S2	N	1	I	0 - 4	Trend mode: 0 = sample 1 = mean 2 = minimum 3 = maximum 4 = sum
S3	N	0	I	0 or 1	Trend resolution[2]: 0 = normal 1 = fast

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC
2. Fast trending requires a higher amount of module utilization than normal trending. Specification S3 cannot be changed during the on-line configuration mode.

66.1 Explanations

66.1.1 Specifications

S1 – BLKADR
Block address of the analog input to be trended.

S2
(Trend mode) Sets the trend mode during the collection period (S3). Each value collected is the sample, mean, minimum, maximum, or sum for the point.

0 = sample; current point saved every 15 or 60 seconds

1 = mean; arithmetic mean value of all points over the trending interval

2 = minimum; minimum point value over trending interval

3 = maximum; maximum point value over trending interval

4 = sum; sum of all values collected over trending interval

S3
(Trend resolution) Sets the trend interval.

0 = normal; every 60 seconds
1 = fast; every 15 seconds

NOTE: Fast trending requires a higher amount of module utilization than normal trending.

66.2 Applications

To trend values, configure both an analog exception report and a trend block for each point trended. Figure 66-1 shows a sample analog trending configuration. Figure 66-2 shows a sample digital trending configuration. The block outputs go to the communication highway and have assigned tag names in the operator interface station. Both outputs must be configured for each point for display purposes.
To create a trend database and a tag database, refer to the human system interface configuration instruction. These instructions have sections titled ***Trends*** and ***Tag Database*** that will provide the necessary instructions to create the trend and tag databases.

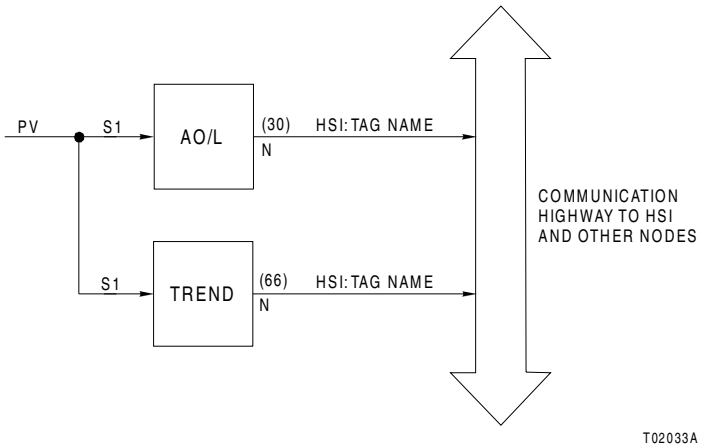


Figure 66-1 Configuration for Trending an Analog Point

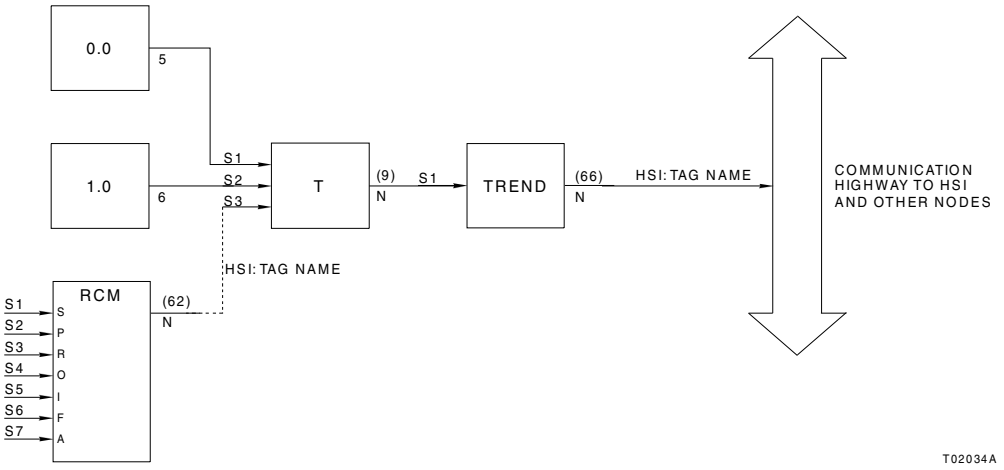
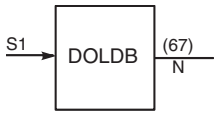


Figure 66-1 Configuration for Trending a Digital Point

67. Digital Exception Report with Alarm Deadband

The digital exception report function code exception reports <S1>. Specification S2 defines the alarm state. Specification S3 defines the alarm deadband time in seconds. After the alarm state is set, the <S1> value must exit the defined alarm state for the time period specified by S3 for the alarm state to be cleared. The S3 timer restarts on each entry into the alarm state. Exception reports are activated by a report enable signal from the module bus. The maximum number of exception reports allowed depends on the controller configuration.
Use function code 69 to test the alarm associated with the digital exception report with alarm deadband function block.



Outputs

Blk	Type	Description
N	B	Value of exception report

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of input
S2	N	2	I	0 - 2	Alarm state: 0 = logic 0 alarm state 1 = logic 1 alarm state 2 = no alarm state defined
S3	Y	0.000	R	Full	Alarm deadband time period in seconds
S4	Y	0.000	R	Full	Spare

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

67.1 Explanation

67.1.1 Exception Reports

An exception report returns to the module bus following a report poll message if a report enable message has been received for the block number. To import this data to another PCU requires programming a digital input/loop block (function code 42) or digital input/Cnet (function code 122) in another node referencing the output of the function code 67 block. To import this data to a console the point must be added to the tag list.
An exception report occurs when:

1. $<S1> = <S1>_L$ and $t_r > t_{min}$ and S2 = 2

where:

<S1>	=	Current value of input.
$<S1>_L$	=	Last reported value of input.
t_r	=	Time since last report.
t_{min}	=	Minimum report time for the block as specified by the executive block or the segment control block.

or

1. $t_r > t_{max}$ *and S2 = 2*

where:

t_r	=	Time since last report.
t_{max}	=	Maximum report time for the block as specified by the executive block or the segment control block.

An alarm report occurs when:

2. *<S1> = S2, current status = normal, and S2 ≠ 2 and S3 = 0.000*

If S3 is not 0.000, then the alarm deadband time holds the alarm state active. <S1> must exit the alarm state specified by S2 for the time specified in S3 for the alarm state to be cleared. The S3 timer restarts on each entry into the alarm state.

where:

<S1>	=	Current value of input.
S2	=	Alarm state: 0 = alarm when <S1> = 0 1 = alarm when <S1> = 1 2 = no alarm

67.2 Examples

Figure 67-1 illustrates an example configuration for a digital exception report function code

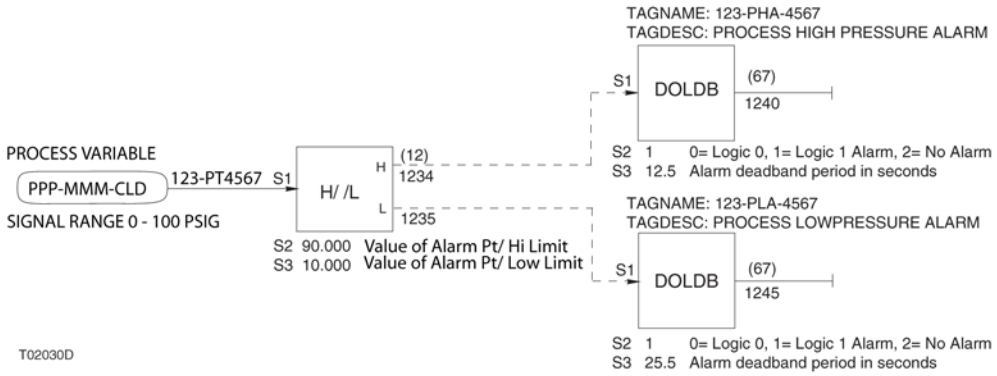
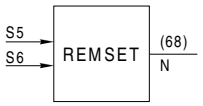


Figure 67-1 Example Digital Exception Report With Alarm Deadband Configuration

68. Remote Manual Set Constant

The remote manual set constant (RMSC) function code allows the value of a constant to be entered to the control scheme via a console (i.e., operator interface station) or other device such as the plant computer via a network interface unit. When the function block receives this command, an exception report is generated to notify the operator or computer that the value has changed. An exception report is also sent on the loop at the maximum exception report time; or if in the track mode, the value of S6 exceeds the significant change parameter of the segment or module, whichever applies.
High and low limits may be set to guard against unreasonable values. Specifications S2 and S3 select the high and low limits and also set the span for significant change. Specification S4 specifies the output value of the controller in normal mode
(S5 = 0) on initial startup. The S4 value is updated via a console and is output from the block after a mode change to execute or a reset of the module.
Specifications <S5> and <S6> are the block addresses of the track switch and track reference values. The RMSC block can be forced to output the value from <S6> when the track switch <S5> goes to one. An exception report is generated each time the track switch activates or deactivates.



Outputs

Blk	Type	Description
N	R	Last value entered for S4 when <S5> = 0, or value of <S6> when <S5> = 1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Full	Engineering unit identifier
S2	N	100.000	R	Full	High limit
S3	N	0.000	R	Full	Low limit
S4	N	0.000	R	Full	Initial output value after addition, modification, power up of block
S5	N	0	I	Note 1	Block address of track switch: 0 = normal 1 = track
S6	N	5	I	Note 1	Block address of track value

NOTES:
1. Maximum values are: 9,998 for the BRC-100
31,998 for the HC800, BRC-400/410, HPG800 and HAC

68.1 Explanation

Figure 68-1 shows an example of a configuration to enable operator intervention in case of temperature transmitter failure. The test quality block (function code 31) outputs a zero as long as the transmitter input is good. The NOT block (function code 33) inverts this to a one and the REMSET output tracks the transmitter until the transmitter fails. At that time, the REMSET is released to the operator.

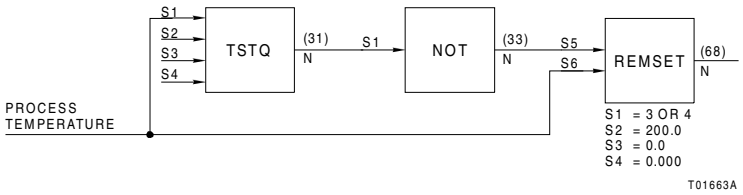
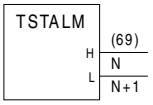


Figure 68-1 Configuration for Operator Intervention

69. Test Alarm

The test alarm function code tests the alarm status of a designated input signal. It can test the alarm state on any block output that has a defined quality or alarm, including exception reports, station blocks, and device drivers.
There are two separate boolean outputs. Table 69-2 shows the different output descriptions for the types of blocks monitored. Note that for control stations, S2 selects the type of alarm monitored (absolute or deviation). Alarm status is zero equals no alarm, one equals alarm.



Outputs

Blk	Type	Description
N	B	High alarm indicator: 0 = no alarm 1 = alarm
N+1	B	Low alarm indicator: 0 = no alarm 1 = alarm

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block number to be tested for alarms
S2	N	0	I	0 - 2	Alarm condition tested for: 0 = absolute alarms 1 = deviation alarms (stations only) 2 = device drivers

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

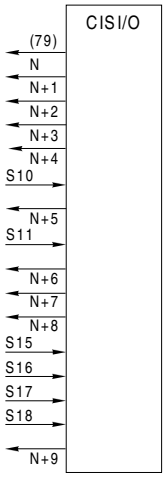
Table 69-2 Outputs from Test Alarm Block

Input Type	Outputs	
	N	N+1
Boolean	Alarm	0
Real	High alarm	Low alarm
FC 80	High alarm (type selected with S2)	Low alarm (type selected with S2)
FC 123, 129	Alarm	Mode: 0 = manual 1 = auto/remote

79. Control Interface Slave

Use the control interface slave function code to interface a Harmony controller to a control I/O module. Control I/O modules have four analog inputs, two analog outputs, three digital inputs, and four digital outputs. This function code defines the characteristics of that I/O module and how the system responds to an I/O module failure.

Outputs



Blk	Type	Description
N	R	Analog input 1
N+1	R	Analog input 2
N+2	R	Analog input 3
N+3	R	Analog input 4
N+4	R	Analog output 1 feedback
N+5	R	Analog output 2 feedback
N+6	B	Digital input 1
N+7	B	Digital input 2
N+8	B	Digital input 3
N+9	B	I/O module status flag: 0 = good, 1 = bad

The control system must be carefully evaluated to establish default values that will prevent personal injury and/or property damage in case of module failure.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Control interface I/O module address
S2	N	0.000	R	Full	Analog input 1 zero in engineering units
S3	N	0.000	R	Full	Analog input 1 span in engineering units
S4	N	0.000	R	Full	Analog input 2 zero in engineering units
S5	N	0.000	R	Full	Analog input 2 span in engineering units
S6	N	0.000	R	Full	Analog input 3 zero in engineering units
S7	N	0.000	R	Full	Analog input 3 span in engineering units
S8	N	0.000	R	Full	Analog input 4 zero in engineering units
S9	N	0.000	R	Full	Analog input 4 span in engineering units
S10	N	2	I	0 - 2046	Block address of analog output 1
S11	N	2	I	0 - 2046	Block address of analog output 2
S12	N	2	I	0 or 1	Digital input 1 logic sense
S13	N	2	I	0 or 1	Digital input 2 logic sense
S14	N	2	I	0 or 1	Digital input 3 logic sense

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S15	N	2	I	0 - 2046	Block address of digital output 1
S16	N	2	I	0 - 2046	Block address of digital output 2
S17	N	2	I	0 - 2046	Block address of digital output 3
S18	N	2	I	0 - 2046	Block address of digital output 4
S19	N	0	I	0 or 1	Failure action (lack of I/O module response or bad analog input reference voltage): XX0 = trip module XX1 = continue to operate I/O module status flag (bad status indication): 0XX = set I/O module status flag output N+9 to bad when the CIS module fails or is in error mode 1XX = set I/O module status flag output N+9 to bad when the I/O module fails or there is a bad quality on one of the module's I/O points

79.1 Explanation

79.1.1 Outputs

This block provides analog and digital I/O on the same board. The final output is an I/O module status flag, which shows the status of the control I/O module.

0 = good
1 = bad

Analog output feedbacks (N+4 and N+5) provide feedback to M/A stations (function code 80) that have hardware stations with bypass capabilities. This feedback signal provides an alignment reference for the software logic, before establishing control of the process when exiting from bypass or configure mode (bumpless transfer).

NOTE: If S19 equals zero, and if there is a bad or unwired input, then the control module is tripped.

79.1.2 Specifications

S1 – IOSLV
(Control I/O module address) I/O module address set by the address dipswitch on the I/O module. Available addresses are zero through 63.

S2 to S9
(AIZn and AISPn) Set the zero and span values for four analog inputs. If an analog value goes below the zero value or exceeds the span by more than five percent, it will produce a bad quality output. A specified span other than zero (default) for an unused analog input produces a bad quality signal for that input.

S10 – AO1
Block address of input for analog output one.

S11 – AO2
Block address of input for analog output two.

S12 to S14 - DIn
Set the logic sense for three digital inputs. Digital inputs can be defined for direct or reverse acting logic operations. These are defined as follows:

0 = reverse acting (logic 0 = closed, logic 1 = open)
1 = direct acting (logic 0 = open, logic 1 = closed)
2 = unused input

S15 to S18 - DOn
Block addresses of the inputs for the four digital outputs.

S19 – FAIL
Value defines the module response in case of control I/O module failure. Lack of I/O module response or a bad analog input reference voltage will cause a transfer to *error* mode.
Specification S19 also determines which failure conditions cause the I/O module status flag output N+9 to go to logic 1 (bad).

X X X

Failure Action
0 = trip control module
1 = continue operation

Reserved for Future Use

I/O Module Status Flag
0 = selects the I/O module status flag output N+9 to equal logic 1 (bad quality) when the I/O module is failed or in error mode.
1 = selects the I/O module status flag output N+9 to equal logic 1 (bad quality) when the control I/O module is in error mode or a defined I/O point (AI, AO, DI or DO) is bad quality.

80. Control Station

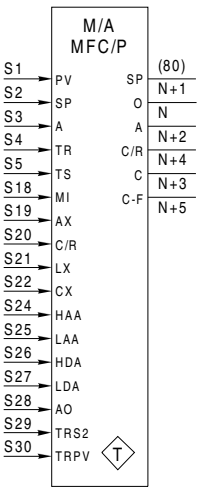
There are three types of stations, each controllable through a digital (NDCS03) or analog (IISAC01) control station and/or a human system interface (HSI). The types include basic, cascade and ratio stations.
A ***basic*** station generates a set point (SP) and provides manual/automatic transfers, control output adjustments in manual control mode, and set point adjustments in automatic control mode.
A ***cascade*** station provides the same functions as a basic station plus an additional mode that allows the set point to be controlled by external input signal <S2>.
A ***ratio*** station provides the same functions as a basic station, but differs from the cascade station in its method of set point generation while in the ratio mode. The wild variable <S2> multiplied by a ratio adjustment factor (ratio index) determines the set point output while in ratio mode. The initial ratio index value is calculated by the station to maintain the current set point output value when the station is placed into the ratio mode. While in the ratio mode, the ratio index value is displayed in place of the set point value and can be adjusted (ramped up or down) by the operator to obtain the desired set point output.
Station control allows changing the mode, set point, ratio index and control output of a control station by manipulating a control station element.
The control output (CO) value during startup is configurable. If S16 defines a DCS station (S16 equals zero through seven) or SAC station (S16 equals zero through 63), then the control output during startup tracks the displayed control output on the DCS station or SAC station. If a communication failure exists for the station (or S16 equals 255), then the control output tracks the CO track signal <S4> during startup.

NOTE: Valid station addresses are zero through 63 and 100 through 163 for Harmony controllers at 40 kilobaud.

The associated analog output (AO) <S28> generates auto-bypass when the AO has bad quality and communicates this state to the control interface module analog output.

NOTES:
1. The maximum ratio for the wild variable is ten when using a DCS station and 100 when using a SAC station. The minimum practical ratio is 0.05.

2. The local SAC/DCS link communication baud rate is set by the hundreds digit S3 of the extended executive block (function code 90, block 20) for Harmony controllers. The remote SAC link located on CIO-100 Harmony I/O blocks is not affected by the S3 specification setting.



Outputs

Blk	Type	Description
N	R	Control output (0% to 100%)
N+1	R	Set point
N+2	B	Automatic mode flag: 0 = manual 1 = automatic
N+3	B	Level flag: 0 = local 1 = computer
N+4	B	Cascade/ratio mode flag: 0 = basic 1 = cascade/ratio
N+5	B	Computer status flag: 0 = computer OK 1 = computer failed

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input to be displayed on the PV scale of a station (can be used for SP track S30)
S2	N	5	I	Note 1	Block address of SP track signal (S29 SP track switch)
S3	N	5	I	Note 1	Block address of auto signal
S4	N	5	I	Note 1	Block address of control output track signal (TR)
S5	N	0	I	Note 1	Block address of control output track switch (TS): 0 = no track 1 = track
S6	N	5	I	1, 2, 3, 5, 6, 7 or 8	Initial mode of station after startup: 1 = computer, manual 2 = computer, auto 3 = computer, cascade/ratio 5 = local, manual 6 = local, auto 7 = local, cascade/ratio 8 = previous mode
S7	Y[2]	9.2 E 18	R	Full	PV high alarm point in engineering units
S8	Y[2]	-9.2 E 18	R	Full	PV low alarm point in engineering units
S9	Y[2]	9.2 E 18	R	Full	PV-SP deviation alarm point in engineering units (for console only)
S10	N	100.000	R	Full	Signal span of PV in engineering units
S11	N	0.000	R	Full	Zero value of PV in engineering units
S12	N	0	I	Full	PV engineering units identifier (for console only)
S13	N	-5.000	R	Full	Signal span of SP in engineering units
S14	N	0.000	R	Full	Zero value of SP in engineering units
S15	N	0	I	Full	SP engineering units identifier (for OIS console only)
S16	N	255	I	Full	Control station address: 254 = passive station 255 = no station
S17	N	0	I	0 - 7	Mode of system default if the computer fails while the loop is under computer control: 0 = computer (auto/manual mode unchanged) 1 = computer, manual 2 = computer, auto 3 = computer, cascade/ratio 4 = local (auto/manual mode unchanged) 5 = local, manual 6 = local, auto 7 = local, cascade/ratio
S18	N	0	I	Note 1	Block address of the transfer to manual signal: 0 = no transfer 1 = transfer to manual and hold
S19	N	0	I	Note 1	Block address of the transfer to auto signal: 0 = no transfer 1 = transfer to auto and hold

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S20	N	0	I	Note 1	Block address of the transfer to cascade/ratio signal: 0 = no transfer 1 = transfer to cascade/ratio and hold
S21	N	0	I	Note 1	Block address of the transfer to local signal: 0 = no transfer 1 = transfer to local and hold
S22	N	0	I	Note 1	Block address of the transfer to computer signal: 0 = no transfer 1 = transfer to computer and hold
S23	N	0	I	0 - 4	Station type: 0 = basic with set point 1 = basic without set point 2 = basic with bias 3 = ratio 4 = cascade
S24	N	0	I	Note 1	Block address of external high absolute alarm flag: 0 = no alarm 1 = high absolute alarm
S25	N	0	I	Note 1	Block address of external low absolute alarm flag: 0 = no alarm 1 = low absolute alarm
S26	N	0	I	Note 1	Block address of external high deviation alarm flag: 0 = no alarm 1 = high deviation alarm
S27	N	0	I	Note 1	Block address of external low deviation alarm flag: 0 = no alarm 1 = low deviation alarm
S28	N	2	I	Note 1	Block address of analog output associated with this station
S29	N	0	I	Note 1	Block address of switch to have SP track <S2>: 0 = no track 1 = track <S2>
S30	N	0	I	Note 1	Block address of switch to have SP track <S1> (PV): 0 = no track 1 = track <S1>
S31	N	60.000	R	Full	Computer watchdog time period (in secs)

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC
2. Specification is tunable, but not adaptable.

80.1 Explanation

80.1.1 Outputs

N

Control output expressed in percent. The station mode and the control output auto <S3> input determine the control output.

N+1

Set point expressed in engineering units. The input to <S2> and the station mode determine the set point output.

N+2
Mode indicator.

0 = manual
1 = automatic

N+3
Level indicator.

0 = local
1 = computer

N+4
Station mode indicator.

0 = basic
1 = cascade/ratio

N+5
Computer status flag.

0 = computer OK
1 = computer failed, station mode dependent on S17

80.1.2 Specifications

S1 – PV
Block address of the input to be displayed on the PV scale of a station (can be used for SP track S30). This input drives the control station process variable indicator (in engineering units).

S2 – SPT
Set point track signal. For stations in basic mode, S2 is an external set point track signal. For stations in the cascade mode, <S2> is the cascade input. For stations in ratio mode, <S2> is the uncontrolled or wild variable. Table 80-1 shows the track behavior of the station block.

NOTE: For stations in ratio mode, the wild variable must be limited to positive numbers, and the set point range must be a positive number. Low limits on either signal must not be less than zero.

Table 80-3 Track Behavior of the Station Block

Station Mode	Manual	Auto	Cascade	Ratio
Basic	A	A	—	—
Cascade	A	A	B	—
Ratio	A	A	—	C

NOTE:
A = When the station tracks and what it tracks depends on <S29> and <S30>. Specification S29 indicates when <S2> should be tracked, and <S30> indicates when <S1> should be tracked. If both indicate tracking, <S29> takes precedence.
B = Specification S2 is tracked unconditionally and the station displays <S2>.
C = Specification S2 multiplied by the ratio index (displayed in digital set point display), is tracked unconditionally and the product of <S2> and the ratio index displays on the station bar graph.

S3 – CO AUTO
Block number whose output value is the control output when the station is in automatic mode (usually the output of a PID block).

S4 – CO TRACK
Block number whose output is the control output when the station is tracking (<S5> equals one). Specification S4 also provides a reference for the station control output when the module completes its startup mode, if an active hard station is not present.

S5 – CO SWITCH
Block number whose output value determines whether the control output is to track <S4>.

0 = no tracking
1 = track

NOTE: Both the *MAN* and *AUTO* lights on the digital control station (NDCS01) light when the module is in the control output override mode (tracking). For the NDCS03 station, just the track light is displayed. The manual/auto status flag does not change when in override state. The actual operating state is saved and will be restored when the track flag goes to zero (normal). This note is not applicable to the analog control station (IISAC01).

S6 – SMODE
Initial mode of the station after startup. For the configurable startup period after power up/reset/switch to execute mode, the station will be in local manual mode. After the startup period expires, the mode is as indicated by S6 unless overridden by <S18> to <S22>. Specifications <S18> through <S22> change the station mode and control level through logic. Any of these specifications override S6. When adding a station block with S6 set to previous mode (eight), the actual mode will be local manual until modified by the operator or logic. When selecting computer or local cascade ratio, cascade or ratio control will be implemented depending on S23 (station type).

1 = computer manual
2 = computer auto
3 = computer cascade/ratio
5 = local manual
6 = local auto
7 = local cascade/ratio
8 = previous mode

S7 – PVH
Sets the engineering units value of the process variable that a high alarm generates and displays on the Human System Interface (HSI).

S8 – PVL
Sets the engineering units value of the process variable that a low alarm generates and displays on the control station or HSI.

S9 – PVDEV
Engineering units value of allowed deviation between the process variable and set point. A high deviation alarm generates when the process variable is greater than the set point and the value of the difference between the two is greater than or equal to S9. A low deviation alarm generates when the process variable is less than the set point and the value of the difference between the two is greater than or equal to S9. These alarm conditions report to the HSI.

S10 – PVSPAN
Sets the signal span of the process variable in engineering units.

S11 – PVZERO
Zero value of the process variable in engineering units.

S12 – PVEU
Process variable engineering units identifier, used by the HSI.

S13 – SPSPAN
Sets the signal span of set point in engineering units. When left at default, set point span equals span defined by S10. The default setting for S13 is -5.000. It should be noted that HSIs will display the PV span (S10) for the set point span (S13). The Harmony controllers use the PVSPAN (S10) and PVEU (S12) for both the process variable and set point as long as the set point span is set to the default value.

NOTE: Some DCS stations will not function correctly using the -5.000 default value. If a problem exists, set S13 and S14 the same as S10 and S11.

S14 – SPZERO
Zero value of set point in engineering units.

S15 – SPEU
Set point engineering units identifier, used by the HSI.

S16 – DCSADR
Set S16 to the control station address (254 equals passive station and 255 equals no station). Valid station addresses are zero through seven and 100 through 107 for DCS stations, and zero through 63 and 100 through 163 for SAC stations. Addresses 100 through 163 represent actual station addresses of zero to 63. If selecting addresses 100 through 163, the Harmony controllers will not report the status of the control station in the module status or module problem reports. In all other respects, station operation remains unchanged. This specification must be set to 254 when a passive station interface (function code 139) S1 points to this function code.

NOTE: Valid station addresses are zero through 63 and 100 through 163 for the Harmony I/O CIO-100 and for Harmony controllers at 40 kilobaud.

S17 – CFAIL
Defines the mode the station will default to in the event of a computer failure while it is under computer control. When selecting computer or local cascade/ratio, the station assumes cascade or ratio control depending on S23.

0 = computer manual/auto mode unchanged
1 = computer manual
2 = computer auto
3 = computer cascade/ratio
4 = local manual/auto mode unchanged
5 = local manual
6 = local auto
7 = local cascade/ratio

S18 through S22
Block addresses that change the station mode and control level through logic. If more than one (<S18> through <S22>) are set, <S18> takes precedence over <S19>, which takes precedence over <S20>, etc. If the station is in manual mode, <S20> places the station in automatic and cascade/ratio mode simultaneously. When both <S21> and <S22> are set, <S21> overrides <S22>.

NOTE: When one or more inputs to <S18> through <S22> equal one, the station displays the interlock state. The station is locked in a particular mode. The output cannot be changed by the operator unless S18 equals one. Inputs <S18> through <S22> are typically driven by fault logic that places the control loop in a known condition when a failure is detected. Removing an interlock state leaves the current mode unchanged, but allows the operator to change the mode.

S18 – MANXFR
Block address of the transfer to manual signal.

0 = no transfer
1 = transfer to manual

S19 – AUTOXFR
Block address of the transfer to auto signal.

0 = no transfer
1 = transfer to auto

S20 – CSRXFR
Block address of the transfer to cascade/ratio signal. This specification transfers to cascade or ratio control depending on the type of station selected with S23.

0 = no transfer
1 = transfer to cascade/ratio

S21 – LOCLXFR
Block address of the transfer to local signal.

0 = no transfer
1 = transfer to local

S22 – CMPXFR
Block address of the transfer to computer signal.

0 = no transfer
1 = transfer to computer

S23 – STNTYP
Provides a choice between several station types for normal operation. The definition of <S2> depends on S23. The value of S23 also determines whether cascade or ratio control is adopted when S6, S17 and <S20> are set to cascade/ratio (refer to S6). This specification will not be fully operational until the HSI display strategies are modified. Until that time, only zero, three and four are valid type specifications.

0 = basic with set point
1 = basic without set point (presently not implemented)
2 = basic with bias (presently not implemented)
3 = ratio
4 = cascade

S24 through S27
Block addresses that provide a mechanism for external absolute and deviation alarming. Specifications S24 and S25 can be used for end of travel alarms. Specifications S24 through S27 are logical ORed with their respective internal alarms, defined by S7 through S9, to determine the alarm states presented to the operator.

S24 – EHALRM
Sets the external high absolute alarm flag.

0 = no alarm
1 = high absolute alarm

S25 – ELALRM
Sets the external low absolute alarm flag.

0 = no alarm
1 = low absolute alarm

S26 – EHDALRM
Sets the external high deviation alarm flag.

0 = no alarm
1 = high deviation alarm

S27 – ELDALRM
Sets the external low deviation alarm flag.

0 = no alarm
1 = low deviation alarm

S28 – AOBLK
Analog output block number associated with the station. Use this specification for proper operation of bypass logic. When S28 equals zero or two, it is not used. Any setting other than zero or two must be a block number of the control interface module (CIS) function code 79, the analog output/slave (ASO) function code 149, the analog output/channel (AO/CH) function code 223, or the device definition (DD) function code 221.

NOTES:
1. Only the physical AO from function code 79 is referenced to check the quality of the AO for auto bypass when the quality is bad. When using function code 79, S28 should reference output block N+4 or N+5, not output block N.

2. When specification S28 is connected to block output number four of a device definition (function code 221) function block, the bypass logic of the control station (function code 80) function block operates in the same manner as when specification S28 is set to zero or two. The operational difference between referencing the device definition function block and setting specification S28 to zero or two is that when specification S28 references the device definition function block, the control station function block interfaces to a SAC station attached to a control I/O (CIO) block.

3. When specification S28 is connected to the block output of an analog output/channel (function code 223) function block and specification S16 specifies a valid SAC address, the control station function block interfaces to a SAC station attached to a control I/O block.

4. When an analog out/channel (function code 223) function block is placed into simulation (specification S15 of function code 223 set to 1) its associated control station function block will place its SAC station (auto-bypass enabled) into bypass operation with its demand output set to the last non-simulated control output value and function code 223 will not set the suspect bit. The auto-bypass SAC station will remain in bypass operation until the simulation option is disabled and the current control output value is set to match the control output value that was in effect prior to the simulation.

5. The IISAC01 analog control station bypass function takes a higher precedence in the control of the analog output field element it shares in common with the CIO 100 block. This means that the bypass functionality takes precedence over function code 223 undefined, override, simulation, and normal modes of operation. It also takes precedence over function code 79 undefined and normal modes of operation.

6. Specification S28 may be connected to an analog output channel (function code 223) that is associated with an AOT analog output channel. In this application an IISAC01 is not used and S16 is set to 254 or 255. The station (function code 80) function block will transfer to manual only if both redundant AOT Harmony I/O blocks experience failures on that particular channel.

Specification S28 provides a way to automatically monitor the CIO block/CIS module or AOT block/ASO module output channel. Should the CIO block/CIS module or AOT block/ASO module detect a fault on the current output, the station transfers to manual. If using a control station with a CIO block/CIS module, selecting auto-bypass on the station causes it to transfer to bypass when the output faults.

NOTE: The HSI does not indicate that the station is in bypass or locked in manual (as it does if S18 equals one).

S29 – SPTRCK
Block address of the analog output set point track signal. This determines the track behavior of the set point in conjunction with <S30>. Specification S29 indicates when <S2> should be tracked, and <S30> indicates when <S1> should be tracked. If both indicate tracking, <S29> overrides <S30>. If this specification equals one, it will cause the set point to track <S2> whether the station is in manual or automatic mode. This specification is not applicable when the station is in cascade mode because the cascade input uses none of the internal station logic for control.

0 = no track
1 = track <S2>

S30 – PVTRCK
Block address of the process variable track signal. It determines the track behavior of the set point in conjunction with <S29>. Specification <S29> indicates <S2> should be tracked, and <S30> indicates <S1> should be tracked. If both indicate tracking, <S29> overrides <S30>. When <S30> equals one, it causes the set point to track <S1> whether the station is in automatic or manual mode. This specification is not applicable when the station is in cascade mode because the cascade input uses none of the internal station logic for set point control.

0 = no track
1 = track <S1>

S31 – CMPWDG
Computer watchdog time-out interval times computer communications when a station is under computer control. Timing starts when a computer OK message goes from the network interface to the module, signifying that the computer received all information transmitted from the module. The timer is reset by each subsequent OK message from the network interface and station variable settings.
For example, if the station is switched from manual to computer control, a message will be sent to the network interface, which will generate an OK message and initiate timing. If the elapsed time between OK messages exceeds the value of S31, the timer times out. The control mode is then determined by the value of S17 (computer watchdog time-out option) in the station. If the computer replies to a module message before the time interval is over, the timer resets itself and begins timing again with the next communication. The interval is selected in seconds, with a default value of 60 seconds. An interval value of 0.0 disables the computer watchdog time-out feature.

80.2 Applications

Figure 80-1 illustrates a single input, single output control loop run by a control station in basic mode (function code 79). This configuration uses a PID block for error correction. The process variable and the control output interface with the field devices through a control interface module block. If station parameters such as process variable, set point, and control output are to be trended, an AO/L block is not necessary because the current values are obtained from the station exception report. The exception reports are on the loop without an AO/L block. Only a trend block (function code 66 or 179) is necessary.
Figure 80-1 shows how an auto bypass capable hardstation is configured to operate on the remote SAC link of a CIO block using function code 221, function code 223, and function code 80 function blocks interfaced to CIO/SAC01 hardware.

Figure 80-1 shows how an indicating only hardstation is configured to operate on the remote SAC link of a CIO block using function code 221, function code 223, and function code 80 function blocks interfaced to CIO/SAC01 hardware.

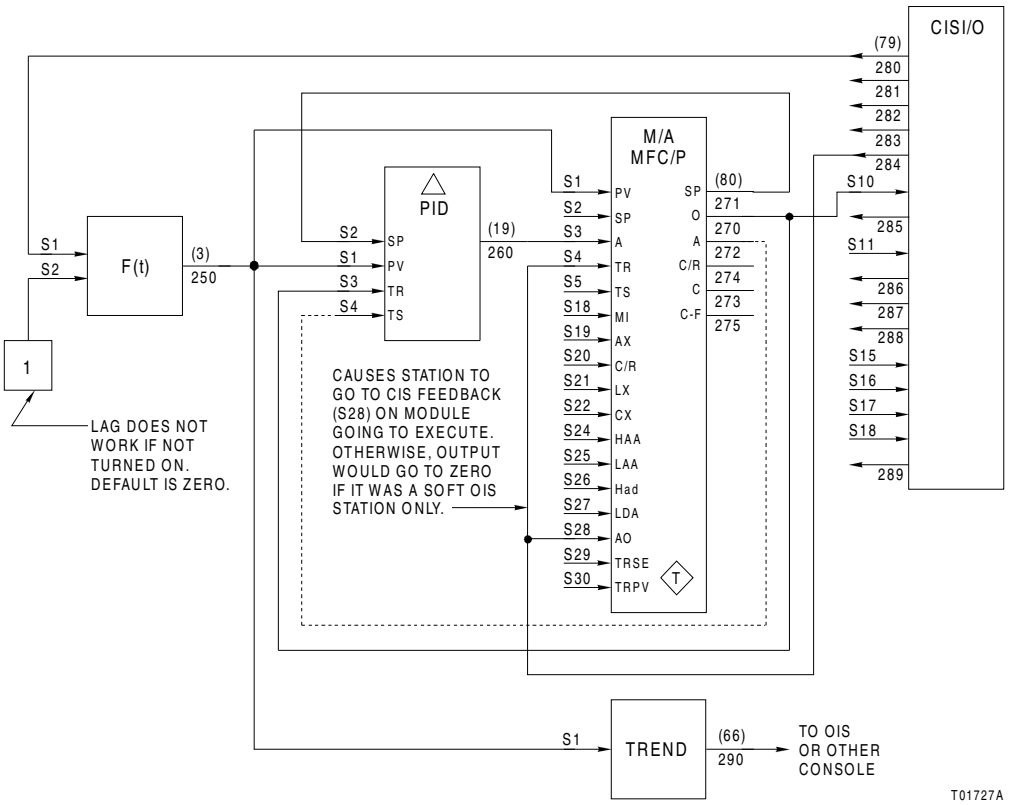


Figure 80-1 Single Input, Single Output Control Loop with Auto Bypass

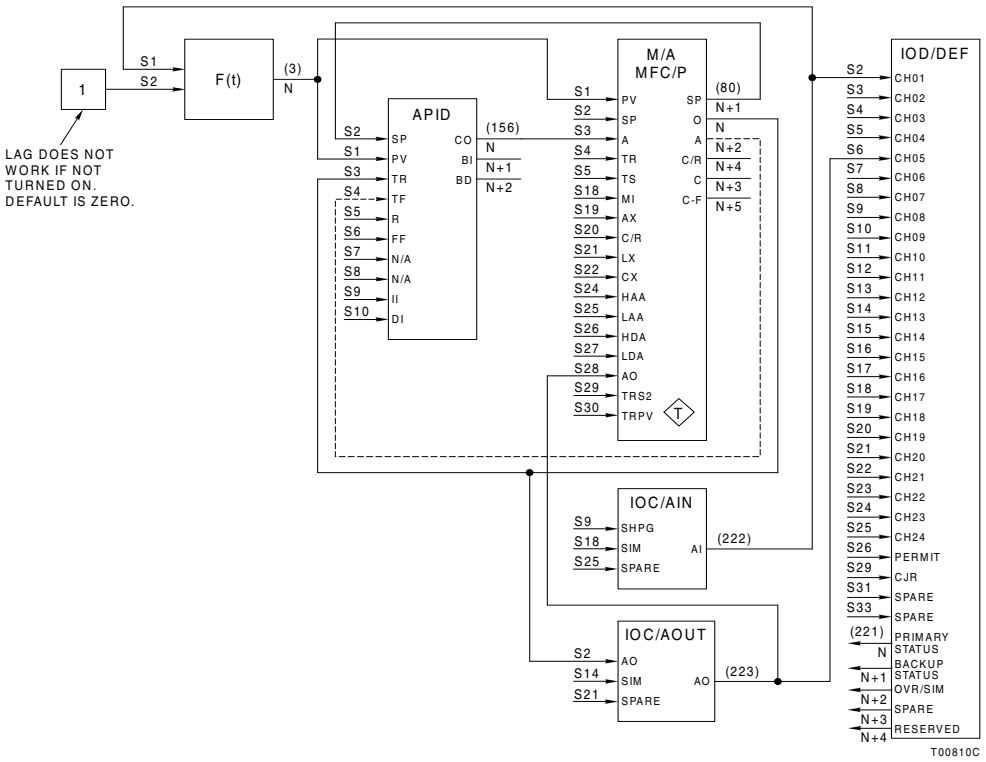


Figure 80-1 Single Input, Single Output Control Loop with Auto Bypass

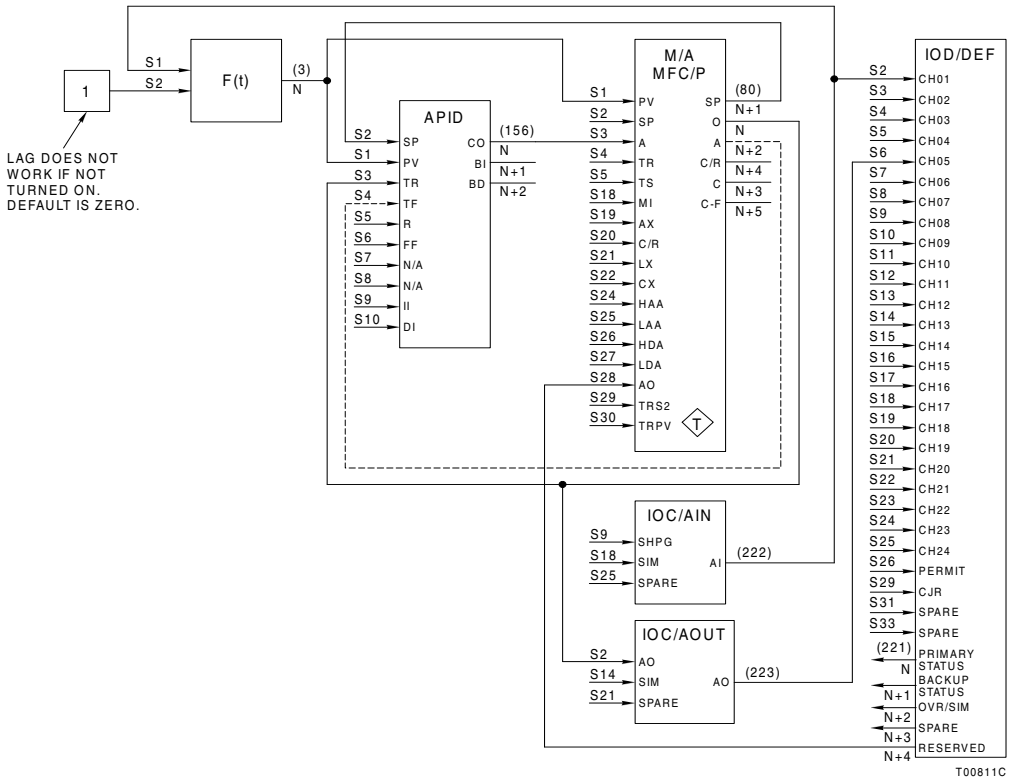
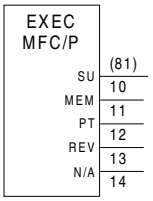


Figure 80-1 Single Input, Single Output Control Loop without Auto Bypass

81. Executive

The executive function code resides in fixed block zero of a control module. It selects the output mode of the light emitting diodes (LED) on the front panel of the control module. The LEDs display the status of the control module or an internal memory location. The executive function code has 15 output blocks numbered zero through 14. Fixed blocks zero through nine are fixed values.



Outputs

Blk	Type	Description
0	B	Logic 0
1	B	Logic 1
2	B	0 or 0.000
3	R	-100.000
4	R	-1.000
5	R	0.000
6	R	1.000
7	R	100.000
8	R	-9.2 E18
9	R	9.2 E18
10	B	Startup in progress flag: 0 = no 1 = yes
11	R	Memory display value
12	R	System free time (%)
13	R	Revision level
14	R	Reserved

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0	I	0 or 1	Front plate LED display mode: 0 = normal (i.e., module status) 1 = display memory (for diagnostic purposes only)
S2	Y	0	I	0 - 255	Memory display address (most significant byte)
S3	Y	0	I	0 - 255	Memory display address (middle byte)
S4	Y	0	I	0 - 255	Memory display address (least significant byte)

81.1 Explanation

The front panel display of the controller provides diagnostic information that describes the CPU operating condition, additional memory operating information, and additional operating information on the entire module. The product instruction for this controller provides a full description of diagnostic codes.

81.1.1 Outputs

Output blocks zero through nine are various system constants and are described in the output table. Output blocks ten through 13 provide module status information.

10
(Startup in progress flag) Logic 1 for the startup period specified by S4 of function code 90 when the module is in execute mode. When startup is successful, this signal reverts to logic 0, and remains at logic 0 as long as the module is in execute mode.

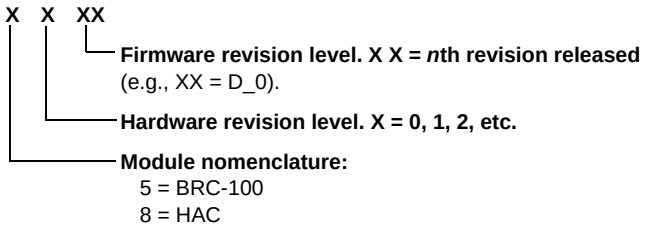
0 = no
1 = yes

11
(Memory display value) Either the memory address selected with S2 through S4 or the current module status, depending on which option was selected with S1.

12
(System free time in percent) Percentage of free time left in the control module.

Free time = 1 – total module utilization

13
(Revision level) Four digit number identifying the module nomenclature, hardware revision level, and firmware revision level as illustrated below.

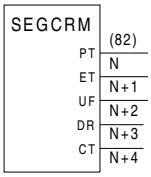


14
Reserved for future use.

82. Segment Control

The segment control function code groups subsequent blocks into a scan cycle executed at a specified rate and priority. Each module can support up to eight segment control blocks. Each segment control block provides five outputs. Block address 15 contains one permanently configured segment control block. The other seven are configurable.

NOTE: Online configuration allows changing the function block configuration during controller execution without interrupting the control process. For the HAC only, function code 82 can not be added or deleted via online configuration.



Outputs

Blk	Type	Description
N	R	Elapsed time of previous cycle in units set by S1
N+1	R	Elapsed time of current cycle in units set by S1
N+2	R	Processor utilization (%)
N+3	R	Checkpoint overrun count number (number of cycles over that are specified in S4)
N+4	R	Cycle time overrun in units set by S1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	1	I	01, 02, 11, 12, 21, 22, 31 or 32	Segment attributes; tune and modify lock (1X, 2X and 3X can be unlocked with switch combinations via special operations) **Tune**: 0X = tune and modify allowed 1X = tuning not allowed 2X = modify lock 3X = tune and modify lock **Target period time units**: X1 = seconds X2 = minutes
S2	Y	0.250	R	Full	Target period (seconds or minutes as specified by S1)
S3	N	0	I	0 - 255	Segment priority (0 = lowest). Can not be modified through online configuration.
S4	N	1	I	0 - 32,767	Checkpoint period (number of cycles per checkpoint)
S5	N	0	I	0, 1, or 2	PID reset mode: 0 = normal reset 1 = auto selected external reset 2 = external reset
S6	N	10.000	R	Full	PID maximum derivative gain
S7	N	1.000	R	0.0 - 9.2 E18	Minimum report time for all exception reports in this segment (seconds or minutes as specified by S1)
S8	N	60.000	R	0.0 - 9.2 E18	Maximum report time for all exception reports in this segment (seconds or minutes as specified by S1)
S9	N	2.000	R	0.0 - 9.2 E18	Significant change parameter for all control loop (i.e., station) exception reports in this segment (in % of span)
S10	N	1.000	R	0.0 - 9.2 E18	Alarm deadband for all high and low alarm reports in this segment (in % of span)

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S11	N	1.000	R	0.0 - 9.2 E18	Alarm deadband for all deviation alarm reports in this segment (in % of span)
S12	N	0.000	R	Full	Reserved
S13	N	1.000	R	0.0 - 9.2 E18	Periodic I/O sampling period for this segment (in seconds). This is a multiple of the extended executive block (function code 90, block 20, S2)
S14	Y	9.2 E18	R	0.0 - 9.2 E18	Segment cycle time alarm limit (seconds or minutes as specified by S1)
S15	N	0	I	0 or 1	Auto sequencing: 0 = off 1 = on

82.1 Explanation

The segment control block divides the set of function blocks configured in a module into subsets (segments), and specifies the operating parameters for each segment individually. A segment starts with the block number of a segment control block and ends at the next higher numbered segment control block or last block. For example, if there is a segment control block configured in block 1000, the block numbers would be divided into two segments. The first segment would contain blocks 15 through 999, and the second segment would contain blocks 1000 through the last configurable block. Fixed block 15 contains one permanently configured segment control block, which also occupies blocks 16, 17, 18 and 19. Blocks 15 through 19 cannot be used for any other purpose nor can they be deleted. Up to seven additional segment control blocks can be placed at any configurable block number location greater than 30.

82.1.1 Specifications

S1 – TUNIT
(Segment attributes) Defines the tuning and modification option and execution cycle time units for the segment. If the tens digit is a zero, then all tunable parameters in the segment may be tuned or modified.

NOTE: When multiple segments are used, leave enough free time to run all of the segments.

X X

Ones digit
X1 = seconds
X2 = minutes

Tens digit
0X = tuning allowed
1X = tuning not allowed
2X = modified lock
3X = tune and modify lock

If the tens digit is a one, change is not permitted to tunable parameters in the segment. This software lock insures that tunable parameters in a critical segment cannot be changed while the module is online to the process.

NOTE: The lock option does not affect adaptable functions. Outside segments adapt into segments that are locked.

If the tens digit is a two, modification of the block numbers within this segment is prohibited. This includes the addition of new functionality. Tuning can be performed while in this state. To access logic in this segment, it is necessary to initialize the NVRAM or perform the segment lock special operation.
If the tens digit is a three, tuning and modification to logic in this segment are prohibited. To gain access to logic in this segment, it is necessary to initialize NVRAM or perform the segment lock special operation.
The time units parameter defines the measurement of time units of the segment execution cycle. Specification S2 defines the desired length of the execution cycle.

S2 – CYCTIM
Sets the target segment execution cycle in time units selected with the ones digit of S1. In each segment, blocks execute in a predefined order, selected with S15. A cycle consists of one execution of the blocks plus any idle time (cycle time

remaining after the cycle has been executed). Cycle time is the length of time from the start of one cycle to the start of the next cycle.

NOTE: S2 can be set less than 20 msec, but the checkpoint period (S4) must be adjusted upward such that the following condition is true:
S2 x S4 >= 20 msec
When S2 is less than the segment execution time (e.g., S2 is set to zero), the rule is:
segment execution time x S4 >= 20 msec

S3 – SPRI
(Segment priority) Assigns execution priorities to up to eight active segments. An active segment is one that is ready to run or is running. If two or more segments are active, the processor will run the highest priority segment. Segment priorities should be selected from zero to seven with zero being the lowest priority segment.

NOTE: The segment priority can not be modified through online configuration.

S4 – CHKPER
Applies to redundant module configurations. Checkpointing is the mechanism which keeps the backup module state current with that of the primary module. Checkpointing is the action of initially copying the configuration (once at startup) and after that all significant dynamic data (block outputs, partial results of chained calculations, integration counts, etc.) to the backup module as a block of data. This is essential for the bumpless takeover by the backup should the primary module fail.
The smooth transfer from primary to backup control is the result of the execution rate (time) of the segment, the frequency of the checkpointing operation, and the process dynamics. The actual failover from the primary to the backup occurs in ten milliseconds or less. The checkpointing operation governs the offset or data age between the primary and backup module. The frequency of checkpoint is a multiple of the segment execution time. The default setting for S4 is one. This specifies a checkpoint operation each segment cycle (250 milliseconds). With S4 set to four, the checkpoint operation occurs every fourth segment cycle (one second).
Large configurations have the potential of the backup being many cycles behind the primary when the transfer time exceeds the segment execution rate. This is especially true when using multiple segments.
To compute the time required to checkpoint the dynamic data of a given segment in an Harmony controller, divide the sum of the individual function blocks checkpoint utilization by 1,000,000 (IMMFP03 and BRC controllers) or 22,000 (IMMFP11/12 controllers) bytes per second. The resultant time is the minimum checkpoint time (in seconds). This time must be divided by the selected execution rate of the segment rounded upward to the nearest whole number and configured as the checkpoint period (S4).

NOTE: Refer to function code 140 for checkpoint utilization information.

For example, suppose a given configuration contains the function blocks shown in Table 82-4

Table 82-4 Total Segment Checkpoint Utilization

Function Code	Quantity	Checkpoint Utilization	Total Utilization
82	1	36	36
90	1	52	52
156	40	40	1,600
80	40	68	2,720
221	20	46	920
222	80	54	4,320
223	40	54	2,160
224	80	54	4,320
225	80	54	4,320
179	20	1400	28,000
9	40	14	560
33	40	2	80
37	80	2	160
39	80	2	160
30	40	14	560
		Total Utilization	= 49,968

With a segment cycle time of 0.25 second, calculate the checkpoint period (S4):

$$t = \frac{\textit{total utilization}}{(\text{bytes per sec})} = \frac{49,968}{1,000,000}$$

t = 0.049968 sec. (time for one set of data)

$$\textit{checkpoint period} = \frac{0.049968}{0.25} = 0.199872$$

round up to next integer S4 = 1

NOTES:

1. Programs such as C language or BASIC must be accounted for when calculating S4.

2. When S2 is set to less than 20 msec, then S4 must be adjusted upward such that the following condition is true:
S2 x S4 >= 20 msec
When S2 is less than the segment execution time (e.g., S2 is set to zero), the rule is: segment execution time x S4 >= 20 msec

S5 – XRES
(PID reset mode: 0 = normal reset, 1 = auto selected external reset, 2=external reset) Affects all function code 18 and 19 blocks in the segment. When S5 equals one, the internal memory of function code 18 and 19 blocks in the segment follows the track input, despite the status of the track/release flag. A change in input is modified by proportional and derivative action, and added to the track signal. This combined signal is the output. This prevents reset windup, which may occur in batch systems where controllers may be monitoring control variables but not performing any control action during the current step. In other words, the output of the PID block is not used in the current process step. The controller receives the signal, takes action to correct the error, sees no result, and takes action to correct the error again. As long as the controller receives no results from its control action, it continues to try to correct the error. When the controller goes into service on some other step of the process, it winds up so far beyond the value of the controlled variable that it cannot control it. The external reset option allows controller alignment while it is not being used for control functions.
Refer to ***Examples*** in the section describing function code 19 for a more detailed explanation of the effect of each value.

S6 – PID GAIN
(PID maximum derivative gain) Limits the derivative gain value in all PID blocks in the segment.

S7 – MINXTM
Defines the minimum report time for all exception reports in the segment. Minimum exception reporting time prevents loading on the communication highway. Exception reports will not be sent on the communication highway at each minimum exception report interval unless a value has changed by more than the operator defined significant change (S9) since the last exception report. The default value is one second.

S8 – MAXXTM
Defines the maximum interval between updates of information sent on the communication highway. If the value of a point has not exceeded the significant change (S9) over this time period, a report of its value will automatically be sent on the communication highway. The default value is 60 seconds.

S9 – SIGCHG
Significant change parameter for all communication highway exception reports in this segment except those with their own significant change parameters (i.e., function code 30), expressed as percent of span. It defines the percent of span a point value must change to cause an exception report to be generated.

S10 – HLALMDB
Alarm deadband for all high/low alarm reports in this segment, expressed as percent of span. Alarm deadbands prevent excessive alarm reports when values are hovering around the alarm limit.

S11 – DVALMDB
Alarm deadband for all deviation alarm reports in this segment expressed as a percent of span. Alarm deadbands prevent excessive alarm reports when values are hovering around the alarm limit. Deviation alarm deadbands are for stations only, since only stations have deviation alarms.

S12
Reserved.

S13 – MBUS
(Periodic I/O sampling period for this segment: expressed in seconds) A multiple of the base periodic I/O sampling period, which the extended executive function code 90 (S2) defines. This specification defines the rate at which this segment samples/updates data across the peer-to-peer network or the Controlway/module bus.

S14 – CYCALM
Segment cycle time alarm limit, expressed in seconds. If segment cycle time exceeds this number, block N+4 will output the cycle time overrun in units set by S1.

S15 – SEQUEN
(Auto sequencing signal: 0 = off, 1 = on) If this specification equals one, the module finds and saves the most logical execution order of the function blocks and will execute them in that order, despite block numbers. Auto sequencing helps prevent loopbacks. Loopbacks occur when a block requires the output of a higher numbered block to complete its execution. The segment must then go through two or more execution cycles before the output of the first block is correct. If the auto sequencing function is off (zero), blocks are executed in ascending numerical order.

82.1.2 Outputs

N
Elapsed time of the previous execution cycle in S1 units. The elapsed time includes any segment idle time. If the time required to execute the blocks within the segment is less than the requested cycle time, the remainder is idle time spent waiting before starting the next cycle. Any idle time is available for lower priority segments. This output verifies that the cycle time specified by S2 is met.

N+1
Elapsed time of the current execution cycle in S1 units. This elapsed time does not include any segment idle time. It is a measure of the actual runtime of the blocks within the segment, plus the block runtime of all higher priority segments. This output verifies that the segment is running. A continual upward ramp indicates that the segment is not running. This occurs when higher priority segments consume all the processor time, or when a basic program is waiting for operator input, in an infinite loop, or aborted because of some error condition.

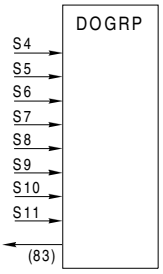
N+2
Processor utilization in percent. This output represents the proportional amount of total module utilization time (100 percent - system free time at block 12) that is used by this segment. This amount of time should be less than 100 percent by a nominal percentage (i.e., ten to 15 percent) dependent on the configuration.

N+3
Checkpoint overrun count number. The number of cycles executed over that are specified by S4. This output verifies that the checkpoint period is met. A continual upward ramp indicates that the segment is never getting the link for dynamic data transfer. A cyclic ramp indicates that dynamic data transfer is occurring, but not at the requested rate. Depending on the overrun, this may be an acceptable situation. If not, then the checkpoint period of the segment or the next highest priority segment must be increased until no overrun occurs.

N+4
Cycle time overrun in units specified by S1. If cycle time exceeds that set by S14, the overrun will be output from this block. This output enables the program logic to take specific action based on a given cycle time alarm limit being exceeded.
Two items must be considered when configuring multiple segments. First, each segment should run within the requested cycle time. Second, insure that the dynamic data of each segment is sent to the backup module within the requested checkpoint period. Achieving this usually involves fine tuning the segment, because the cycle time and checkpoint period of each segment are affected by all segments above it in priority.
To determine if a segment is running within its requested cycle time, observe outputs N and N+1. If output N+1 is continually ramping up, then the segment is not running. This occurs when higher priority segments consume all the processor time, or when a C or basic program is in an infinite loop, waiting for operator input, or aborted because of some error condition. If output N is greater than the requested cycle time set by S2 (when more than one segment exists), the segment is consuming more processor time than the requested period. In this case, cycle time must be increased until the segment output N is equal to the time it really takes to execute the segment.
To determine if a segment checkpoint period is occurring within the requested time period, observe output N+3. If N+3 is continually ramping up, then the checkpointing (refer to S4) of dynamic data is not occurring. This happens when higher priority segments demand all the link time, or if the segment is not running. If N+3 has a cyclic ramp, then the dynamic data is being sent to the backup module, but not at the requested rate. Depending on the amount of overrun, this may be an acceptable situation. If it is not acceptable, then the checkpoint period of this or the next highest priority segment must be increased until no overrun occurs.

83. Digital Output Group

This block writes a group of eight outputs to an IMDSO14 or IMDSO15 digital module.



Outputs

Blk	Type	Description
N	B	Status of output group: 0 = good 1 = bad (I/O module failed to respond)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus I/O module address of digital I/O module
S2	N	0	I	0 - 120	I/O module definition = hold + type + group **Hold:** (defaults set with function code 128): 0XX = go to default values on loss of control module 1XX = hold I/O module output on loss of control module **Type:** X0X = IMDSM05 X1X = IMDSO04, IMDSO14, IMDSO15 X2X = IMDSO01, IMDSO02, IMDSO03 **CSC01 group settings:** XX0 = outputs 0 - 7 XX1 = outputs 8 - 15
S3	N	0	I	0, 1, 2, 4	Action on I/O module failure: 0 = trip control module 1 = continue to operate 2 = continue to operate for 1 cycle after I/O module fails (LIO status set) 4 = continue to operate after I/O module fails (LIO status disabled)
S4	N	0	I	Note 1	Block address of value for first output
S5	N	0	I	Note 1	Block address of value for second output
S6	N	0	I	Note 1	Block address of value for third output
S7	N	0	I	Note 1	Block address of value for fourth output
S8	N	0	I	Note 1	Block address of value for fifth output
S9	N	0	I	Note 1	Block address of value for sixth output
S10	N	0	I	Note 1	Block address of value for seventh output
S11	N	0	I	Note 1	Block address of value for eighth output

NOTE:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

The control system must be carefully evaluated to establish default values that will prevent personal injury and/or property damage in the case of module failure.

83.1 Explanation

83.1.1 Specifications

S1 – SLVADR
Address of the digital I/O module (zero to 63).

S2 – SLVDEF
I/O module definition = hold + type + group

NOTE: The hundreds digit must be the same for both output groups on IMDSM05 and IMDSO14 modules.

X X X

Group = Defines which group of outputs from the I/O module is being handled by this block. IMDSO01/02/03/15 modules can only have a group value of zero because they have only one group of outputs. IMDSM05 and IMDSO14 modules can accommodate 16 outputs.
- XX0 = outputs zero to seven
- XX1 = outputs eight to 15

Type = Type of I/O module
- X0X = IMDSM05 (default setting)
- X1X = IMDSO14, IMDSO15, IMDSO04
- X2X = IMDSO01, IMDSO02, IMDSO03

Hold = Defines action taken by block on loss of control module. Defaults set with function code 128, except for the IMDSM05 whose defaults are set with hardware switches. For the IMDSM05, refer to the installation section of the ***Digital I/O Slave Module (IMDSM05)*** instruction.
- 0XX = go to default values on loss of control module
- 1XX = hold I/O module outputs on loss of control module

S3 – TRIP
Defines the action of the control module in the event of an I/O module failure.

0 = trip control module
1 = continue operation
2 = continue to operate for 1 cycle after I/O module fails (LIO status set)
4 = continue to operate after I/O module fails (LIO status disabled)

NOTE: Only a 0 or 1 may be specified for firmware revisions prior to K_0. The full range is supported on BRC300/400/410 or HPG800 modules for firmware revisions K_0 or later.

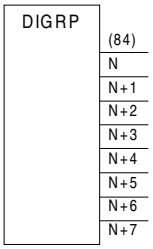
S4 – S11
Block addresses of the values of the eight I/O module outputs. Specification S4 is the block address of the value for the first output, S5 is the block address of the value for the second output, etc.

84. Digital Input Group

The digital input group function code reads a group of eight inputs from an IMDSI12, IMDSI13, IMDSI14, IMDSI15, or IMDSI22 digital module.
To insure that the signals successfully transfer across the controlway/module bus, the signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The quality of the signal cannot be determined by any other type of block. However, the output of the test quality block can be used as an input to other digital processing blocks to provide signal quality information. Refer to Appendix H, for a definition of point quality.

The control system must be carefully evaluated to establish default values that will prevent personal injury and/or property damage in case of module failure.

Outputs



Blk	Type	Description
N	B	Output 1 with quality
N+1	B	Output 2 with quality
N+2	B	Output 3 with quality
N+3	B	Output 4 with quality
N+4	B	Output 5 with quality
N+5	B	Output 6 with quality
N+6	B	Output 7 with quality
N+7	B	Output 8 with quality

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus I/O module address
S2	N	0	I	0 or 1	Input group: 0 = inputs 1 - 8, group A 1 = inputs 9 - 16, group B
S3	N	0	I	0 or 1	Action on I/O module failure: 0 = trip control module 1 = continue to operate

84.1 Explanation

84.1.1 Specifications

S1 – SLVADR
I/O module associated with this block (zero through 63).

S2 – GROUP
Input group. Digital I/O modules can handle two groups of eight inputs. Input group is identified by the following:

0 = inputs 1 through 8
1 = inputs 9 through 16

S3 – TRIP

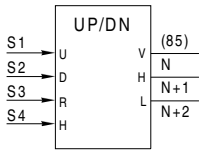
Defines the action of the control module in the event of I/O module failure.

0 = trip control module
1 = continue operation

85. Up/Down Counter

The up/down counter function code is a digital counter with separate inputs for incrementing and decrementing the counter. The count updates on a logic 0 to logic 1 (positive) transition on either input.
External logic can be used to hold or reset the count. The counter can be set to an initial value upon start-up or following reset.
This function has high and low alarm features. Each drives a separate output when the input exceeds the alarm value.

Outputs



Blk	Type	Description
N	R	Contents of counter
N+1	B	High alarm status: 0 = no alarm 1 = count has reached high limit
N+2	B	Low alarm status: 0 = no alarm 1 = count has reached low limit

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of up trigger
S2	N	0	I	Note 1	Block address of down trigger
S3	N	0	I	Note 1	Block address of reset flag: 0 = run 1 = reset
S4	N	1	I	Note 1	Block address of hold flag: 0 = hold 1 = release
S5	Y	0.000	R	Full	Value of count on reset
S6	Y	9.2 E18	R	Full	High alarm value
S7	Y	-9.2 E18	R	Full	Low alarm value

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

85.1 Explanation

The contents of the counter output N is dependent on the reset input, hold input, and the up and down triggers. The reset input overrides all other inputs. Three types of operations are possible: normal, reset and alarm.

85.1.1 Specifications

S1 – UPTRIG
Block address of up trigger. A zero to one transition of <S1> increments the counter by one.

S2 – DNTRIG
Block address of down trigger. A zero to one transition of <S2> decrements the counter by one.

S3 – RESET
Block address of reset input.

0 = run
1 = reset

S4 – HOLD
Block address of the hold input.

0 = hold
1 = release

When <S4> equals zero, the counter remains fixed at the current value despite <S1> and <S2>. When <S4> equals one, the counter releases to respond to <S1> or <S2>.

S5 – COUNT
Value of count on reset. When <S3> equals one, the count goes to the number specified by S5.

S6 – HIALRM
High alarm value. If the value of N equals or exceeds <S6>, output N+1 equals one, signifying a high alarm state. If the reset input <S3> is 1 indicating reset, output N+1 equals zero indicating high alarm cleared.

S7 – LOALRM
Low alarm value. If the value of N equals or falls below <S7>, output N+2 equals one, signifying a low alarm state. If the reset input <S3> is 1 indicating reset, output N+2 equals zero indicating low alarm cleared.

85.2 Reset

If <S3> = 1, then Y = S5 and N+1 = 0
If <S3> = 0, then High and Low alarm status (N+1/N+2) are set as described in ***Alarm***.
where:

<S3>	=	Value of reset input: 0 = run 1 = reset
S5	=	Value of count on reset.
Y	=	Current value of count.

85.3 Normal

If previous <S1> = 0 and current <S1> = 1, then

Y = previous Y+1

If previous <S1> = current <S1>, then

Y = previous Y

If previous <S2> = 0 and current <S2> = 1, then

Y = previous Y - 1

If previous <S2> = current <S2>, then

Y = previous Y

If <S4> = 0, then

Y = previous Y, despite <S1> and <S2>.

where:

<S1>	=	Value of up input.
<S2>	=	Value of down input.
<S4>	=	Value of hold input: 0 = hold 1 = release
Y	=	Current value of count.

85.4 Alarm

If Y ≥ S6 and <S3> = 0, then:

N+1 = 1
else N+1 = 0

If Y ≤ S7 and <S3> = 0, then

N+2 = 1
else N+2 = 0

where:

Y	=	Current value of count.
N+1	=	High alarm indicator: 0 = normal 1 = high alarm
N+2	=	Low alarm indicator: 0 = normal 1 = low alarm

85.5 Applications

Up/down counters count events and enable alarms or trigger events when alarm values are reached.
Figure 85-1 shows an example of the counter used to alert the operator of service requirements. The motor must be removed from service and rebuilt after performing 100 starts. This configuration increments the counter each time the motor starts. The high alarm limit is 100 (starts). When N equals the high alarm limit <S6>, output N+1 equals one. Output N+1 goes to a digital output over the loop function block causing an exception report to be sent to the communication highway. This triggers a *service required* display on the console.

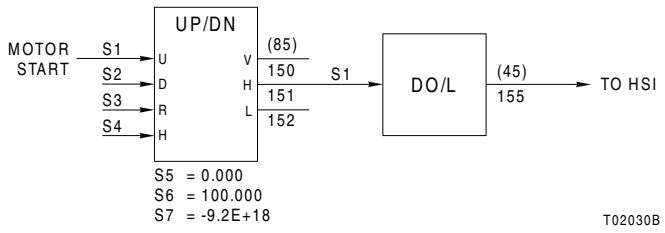
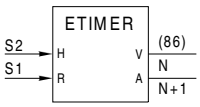


Figure 85-1 Up/Down Counter Maintenance Requirement Alert

86. Elapsed Timer

The elapsed timer function code measures elapsed time. This is especially useful for batch control applications, and may be used for maintenance purposes such as operating time measurements. Time units are configurable to seconds, minutes, hours or days with S3. The timer performs either up or down timing functions, and can be held to a particular value or reset to a specified initial value by external logic.



Outputs

Blk	Type	Description
N	R	Current value of timer (always positive)
N+1	B	Time alarm indicator: 0 = alarm time not reached 1 = alarm time reached

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	1	I	Note 1	Block address of reset flag: 0 = run 1 = reset
S2	N	0	I	Note 1	Block address of hold flag: 0 = hold 1 = release
S3	N	0	I	0 - 3	Units of time: 0 = seconds 1 = minutes 2 = hours 3 = days
S4	Y	0.000	R	0.0 - 9.2 E18	Value of time alarm
S5	Y	0.000	R	0.0 - 9.2 E18	Value of timer on reset
S6 *(cont.)*	N	0	I	000, 001, 100 or 101	Up/down indicator: XX0 = up XX1 = down
S6	N	0	I	000, 001, 100 or 101	Tune/adapt reset timer: 0XX = reset timer upon tune or adapt operation 1XX = do not reset timer upon tune or adapt operation **NOTE:** Tens digit is not important

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

86.1 Explanation

The elapsed timer function code provides a timer with up and down timing functions, automatic reset and hold functions based on external logic functions, and alarm indication if the count reaches a preset alarm value. The reset function overrides all other inputs.

86.1.1 Specifications

S1 – RESET

Block address of the reset input. When <S1> equals one, the timer resets to the value specified by S5.

0 = run
1 = reset

S2 – HOLD
Block address of the hold input. When <S2> equals zero, output N remains at the current value, despite the value of the up/down indicator (S6).

0 = hold
1 = release

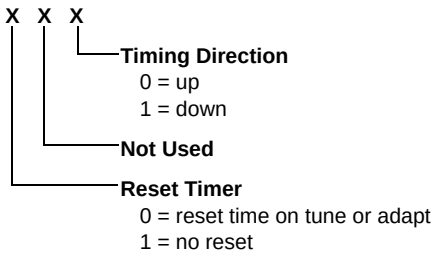
S3 – TIME
Sets the units of time.

0 = seconds
1 = minutes
2 = hours
3 = days

S4 – TALRM
Value of the time alarm. If this value is reached, output N+1 equals one. Specification S3 sets the time alarm time units.

S5 – TIMRES
Sets the value of the timer on a reset. If <S1> goes to one, the timer goes to the value selected with S5. This value is in the units of time selected with S3.

S6 – IND
Up/down indicator. The ones digit of this specification defines the direction of timing. The hundreds digit resets the timer on a tune or adapt operation.



86.1.2 Outputs

N
Current value of the timer. The timer output is based on an internal ten millisecond resolution timer. However, the actual output updates once every segment cycle. Output N is calculated in reset or normal operation.
Reset:
If <S1> = 1, then

N = S5 despite <S2>

where:

N = Current value of timer.

Normal Operation:
Hold count:
If <S1> = 0 and <S2> = 0, then

N = previous N

Increase count:
If <S1> = 0, <S2> = 1, and S6 = XX0, then

N = previous N + elapsed time since last run

Decrease count:
If <S1> = 0, <S2> = 1, and S6 = XX1, then

N = previous N - elapsed time since last run

Tune/Adapt:
If S6 = 0XX and S5 is tuned, then

N = S5, regardless of <S2>

If S6 = 1XX and S5 is tuned, then

N = previous N ± elapsed time since last run

where:
N = Current value of timer.
or
N = S5 when <S1> = 1.

N+1
Alarm indicator. This output indicates when the timer value reaches or exceeds the alarm value specified in S4.
Alarm
If S6 = XX0 and Y ≥ S4, then

N+1 = 1

If S6 = XX0 and Y < S4, then

N+1 = 0

If S6 = XX1, and Y ≤ S4, then

N+1 = 1

If S6 = XX1, and Y > S4, then

N+1 = 0

where:
Y = Current value of timer.

86.2 Applications

Figure 86-1 shows an example of the elapsed timer being used to calculate the average power consumption over a demand period of 15 minutes. The elapsed timer counts up from zero minutes to 15 minutes, and resets itself to zero minutes when the timer reaches 15. Power is integrated and divided by elapsed time to determine average power consumption per unit of time.

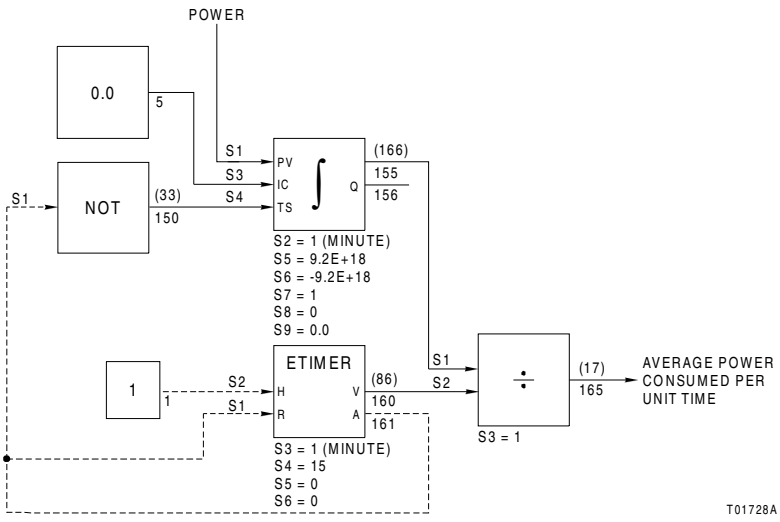
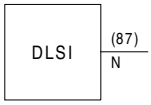


Figure 86-1 Determining Average Power Consumption Over Time

87. Digital Logic Station Interface

The digital logic station (DLS) interface function code specifies the IMDSM05 Digital Slave Module as an interface device between the digital logic station and a module. Each IMDSM05 module can interface with up to eight digital logic stations.



Outputs

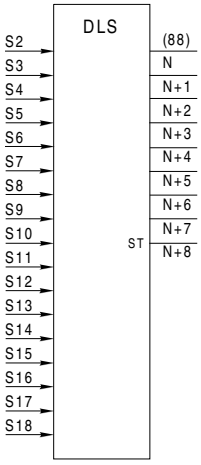
Blk	Type	Description
N	B	I/O module status flag: 0 = good 1 = bad (I/O module failed to respond)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus I/O module address of IMDSM05

88. Digital Logic Station

The digital logic station (DLS) function code configures a digital logic station for use with a module. Each DLS station provides one group of eight inputs and two groups of eight outputs to a module.



Outputs

Blk	Type	Description
N	B	Value of pushbutton 1: 0 = open, 1 = closed
N+1	B	Value of pushbutton 2: 0 = open, 1 = closed
N+2	B	Value of pushbutton 3: 0 = open, 1 = closed
N+3	B	Value of pushbutton 4: 0 = open, 1 = closed
N+4	B	Value of pushbutton 5: 0 = open, 1 = closed
N+5	B	Value of pushbutton 6: 0 = open, 1 = closed
N+6	B	Value of pushbutton 7: 0 = open, 1 = closed
N+7	B	Value of pushbutton 8: 0 = open, 1 = closed
N+8	B	DLS status flag: 0 = good, 1 = bad

NOTE: The pushbutton outputs are pulsed or maintained depending on the digital logic station address specified with S1. An address of 0 to 7 provides a pulsed output. An address of 100 to 107 provides a maintained output.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 107	Logic station address: 0 - 7 = pulse 100 - 107 = maintained
S2	N	0	I	Note 1	Block address of logic station interface.
S3	N	0	I	Note 1	Block address, output 0,group A
S4	N	0	I	Note 1	Block address, output 1,group A
S5	N	0	I	Note 1	Block address, output 2,group A
S6	N	0	I	Note 1	Block address, output 3,group A
S7	N	0	I	Note 1	Block address, output 4,group A
S8	N	0	I	Note 1	Block address, output 5,group A
S9	N	0	I	Note 1	Block address, output 6,group A
S10	N	0	I	Note 1	Block address, output 7,group A
S11	N	0	I	Note 1	Block address, output 0,group B
S12	N	0	I	Note 1	Block address, output 1,group B
S13	N	0	I	Note 1	Block address, output 2,group B
S14	N	0	I	Note 1	Block address, output 3,group B
S15	N	0	I	Note 1	Block address, output 4,group B
S16	N	0	I	Note 1	Block address, output 5,group B

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S17	N	0	I	Note 1	Block address, output 6,group B
S18	N	0	I	Note 1	Block address, output 7,group B

NOTES:
1. Maximum values are: 1023 for IMLMM02
2046 for NMFC01/02, IMMFC05, and IMMPC01
9998 for IMMFC03/04, IMMFP01/02/03/04, and BRC-100/200/300
31998 for BRC-400

88.1 Explanation

The internal logic of the digital logic station provides either a one cycle pulse or a maintained input (as long as the pushbutton is depressed) for all eight pushbuttons. Specification S1 selects the input type for the entire group.

Each pushbutton has two LEDs associated with it, but independent from it. Thus, the combination can be used for either related events or independently.

If parallel control is desired from a console, the digital logic station can be configured in combination with a remote control memory (function code 62) or device driver (function code 123) block. Control is gained from the station or the console.

88.1.1 Specifications

S1 – LSADR

Specification S1 is the digital logic station address. Each DLS station has an address which is set with a three position dipswitch on the station. Valid hardware addresses are zero through seven. With this specification, specify the address as zero to seven or 100 to 107, depending on the type of pushbutton outputs desired.

If the station is addressed from zero to seven (true address), the pushbutton outputs are a pulse output. When a pushbutton is pressed, the associated output goes to a one for one cycle, then returns to zero.

If the station is addressed from 100 to 107 (true address plus 100), the pushbutton outputs are maintained. When a pushbutton is pressed, the associated output goes to a logic 1 and remains there until the pushbutton is released.

S2 – LSI
Specification S2 is the block address of logic station interface. This specification identifies the function code 87 block providing the interface between the digital logic station and its IMDSM05 module.

S3 to S18 - A0 to B7
Specifications S3 through S18 are the block addresses of the output values for output groups A and B. These values are displayed on the LEDs on the front plate of the digital logic station. A logic 0 output turns the LED off, and a logic 1 output lights the LED.

88.1.2 Outputs

N through N+7
Outputs N through N+7 are the values of pushbutton inputs one through eight.

0 = open
1 = closed

N+8

Output N+8 is the digital logic station status flag.

0 = good
1 = bad

88.2 Applications

Figure 88-1 shows a typical configuration of function code 88 used in conjunction with function codes 87 and 35.

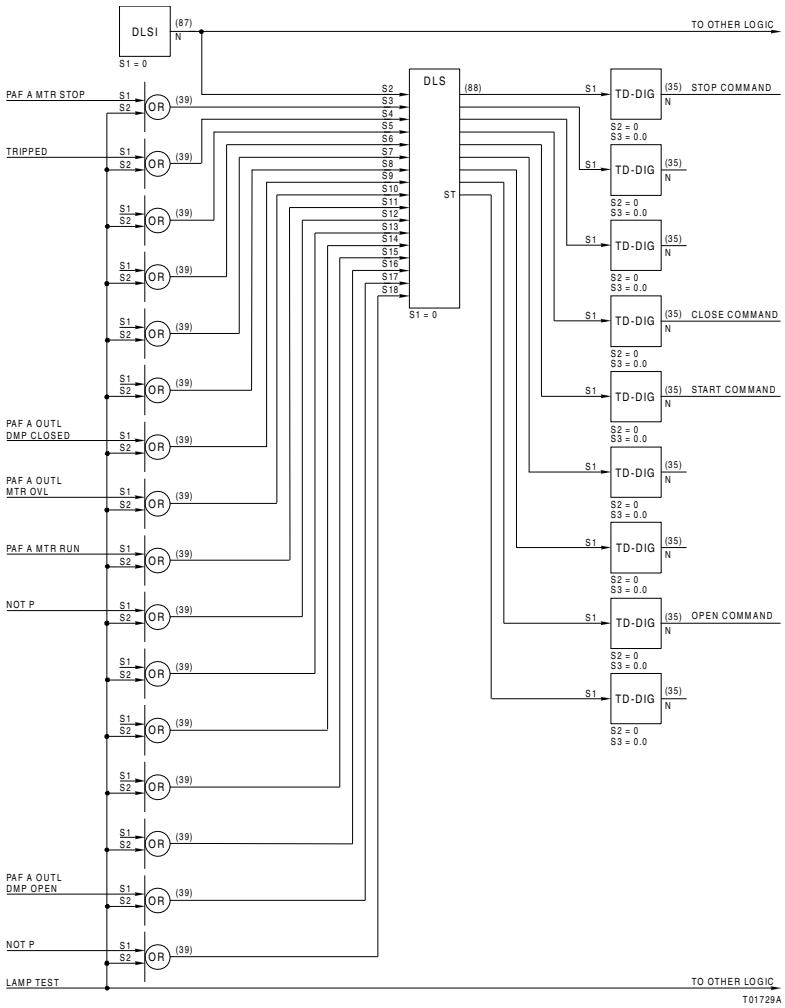


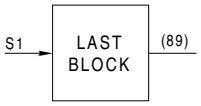
Figure 88-1 Typical Digital Logic Station Configuration

89. Last Block

The last block function code marks the end of the function block configuration space. The last block function code is a fixed block located in the highest available block number of the module. The last block cannot be deleted. Additional last blocks cannot be added to the configuration.

NOTE: This function code is supported only by the BRC-100/200 and the IMMFP11/12 controllers.

In Symphony systems, the last block function code has one output. The output indicates the special operations options configuration including the module network type, time-stamping, Hnet cable length, and revision checking. Special operations are documented in the installation section of the individual controller's product instruction.



Outputs

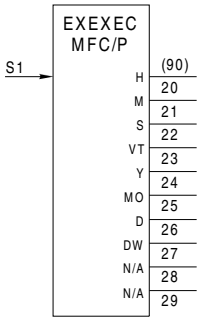
Blk	Type	Description
0 - 9999	R	Special operations options configuration Network type X0X0 = Plant loop X0X1 = Cnet X0X3 = Cnet with time-stamping Hnet cable length (BRC-100/200 only) X00X = 1200m X02X = 3000m X03X = 2000m X04X = 800m Revision check (BRC-100/200 only) 00XX = Revision check disabled 10XX = Revision check enabled

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	-32,768 - 32,767	Reserved integer

90. Extended Executive

The extended executive block function code defines variables that affect module operations. Function code 90 sets system operation characteristics for modules. Use function code 90 with function codes 81 (executive) and 82 (segment control) to define module operations. Function code 90 resides in fixed block 20 and occupies ten function blocks (20 through 29).



Outputs

Blk	Type	Description
20	R	Time of day, hours
21	R	Time of day, minutes
22	R	Time of day, seconds
23	B	Time/date/synchronization flag: 0 = time/date invalid 1 = time/date valid
24	R	Calendar, year (0 - 99)
25	R	Calendar, month (1 - 12)
26	R	Calendar, day (1 - 31)
27	R	Calendar, day of week (1 - 7, Sunday = 1)
28	R	Reserved
29	R	Reserved

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of configure mode lockout flag: 0 = configure mode allowed 1 = configure mode locked out
S2	N	0.250	R	0 - 9.2 E18	Base periodic I/O sampling period for module (in seconds)
S3	Y[2]	0 (BRC300/400/410) 20000 (HC800)	I	00000 - 22111	Redundancy configuration flag and DCS/SAC interface link: XXXX0 = no redundancy XXXX1 = redundancy[6] XXX0X = module goes to error mode on I/O module trip[3] XXX1X = module halts on I/O module trip XX0XX = 5 kbaud DCS/SAC link XX1XX = 40 kbaud SAC link X0XXX = RS-232 mode[4] X1XXX = RS-485 mode X2XXX = both RS-232 and RS-485 modes[4] 0XXXX = expander bus only[5] 2XXXX = both Harmony Net and expander bus, or HN800
S4	Y	15.000	R	0 - 9.2 E18	Module startup time (in secs)
S5	N	0.250	R	0 - 9.2 E18	Logic station poll rate (in secs)
S6	N	0	I	0 or 1	SOE monitor time-synchronization flag: 0 = inhibit sync of SOE monitor to time of day 1 = sync of SOE monitor to time of day

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12 31,998 for the HC800, BRC-400/410, HPG800 and HAC.
2. Only the ***redundancy configuration flag*** and the ***module mode on I/O module trip*** selections are tunable.
3. I/O trip (function code 32) is applicable only to Harmony rack I/O modules (expander bus).
4. Applies to the serial port on the controllers, not the SAC/DCS station link.
5. Modification of the I/O interface type out of expander bus only mode is not permitted unless the Harmony net hardware is installed properly. This setting is not valid for HN800 installations.
6. Redundant BRC-300 controllers on Hnet must be configured for redundancy when used in systems containing NPM22 modules.

90.1 Explanation

90.1.1 Specifications

S1 – LOCK
Block address of configure mode lockout flag.

0 = configure mode allowed
1 = configuration locked

The default value of S1 is block address zero, which provides unlocked configuration. A logic 1 locks the configuration. Unless S1 is a block address of a tunable block, once locked, a configuration cannot be unlocked. To change a locked configuration, the module must be initialized and reconfigured.

NOTE: An engineering lock action may be implemented by using a tunable block as the input to S1. This way the configuration may be unlocked by authorized persons through tuning the appropriate block.

S2 – IOTIME
Base peer-to-peer network or Controlway/module bus I/O period in seconds. This specification defines the rate at which the module samples/updates data across the peer-to-peer network or Controlway/module bus. Specification S13 in the segment control block (function code 82) specifies the I/O period for each segment. If S2 equals two seconds, and S13 (for a particular segment) equals four seconds, the I/O for that segment is read every four seconds. Specification S13 for each segment control block should be a multiple of S2, because S2 defines the fastest peer-to-peer network or Controlway/module bus update time. For example, if S2 equals two and S13 equals 0.5, sample I/O will only be read every two seconds.

S3 – BACKUP

Provides a redundant module configuration flag, sets the module mode on an I/O module trip, sets the DCS interface link data rate, and the I/O interface control.

X X X X X

Redundancy configuration flag
XX0 = no redundant module
XX1 = module is part of redundant arrangement
NOTE: The redundancy configuration flag must be properly configured to insure the correct operation of the backup status (BAC) and redundancy link channel failure (R1F and R2F) indications in the BRC-100 and HAC controller module status. The setting of this flag is reflected in the redundancy expected (RDEXP) indication in the controller status. It is also reflected in the backup configured (BKCFG) indication in the HAC's network processor module status.

Module mode on I/O module trip
X0X = error mode on I/O module trip
X1X = halt module on I/O module trip

Local DCS interface link
0XX = 5 kbaud DCS/SAC link
1XX = 40 kbaud SAC link Harmony controller.

Serial port mode
0XXX = RS-232 mode (both ports)
1XXX = RS-485 mode (terminal port, printer port disabled)
2XXX = RS-485 on terminal port, RS-232 on printer port

I/O Interface Control
0XXXX = expander bus only
2XXXX = both Harmony Net and expander bus, or HN800

NOTES:

1. I/O interface control must be properly configured in order to insure the correct operation of the Harmony Net channel failure and relay fault indications in the controller module status.

2. I/O interface control is not permitted to be configured as (0XXXX) expander bus only mode if Harmony I/O and Foreign Devices (HART & PROFIBUS) function blocks (Table A-17) exist in the configuration, or if HN800 is connected.

3. I/O interface control must be properly configured in order to insure the correct operation of the Harmony Net channel failure fault indications in the controller module status.

S4 – START

Module startup time in seconds. Upon startup of the module, fixed block ten in the module has an output of logic 1 for the length of time specified here. This shows that the startup has been initiated and is proceeding correctly.

S5 – LPOLL

Defines the rate (in seconds) at which logic stations are polled for information.

S6 – SYNC

Sets the sequence of events monitor time-synchronization indicator.

0 = inhibit synchronization of sequence of events monitor to time of day

1 = synchronize sequence of events monitor to time of day

If S6 equals one, then the module time output in blocks 20, 21 and 22 equals the system time sent to the module over the INFI-NET or PN800 network.
If S6 equals zero, the time shown in blocks 20, 21 and 22 is relative to the time the device was powered up, with power up time being 00:00:00.

90.1.2 Outputs

20

Displays the hour of the time of day.

21
Displays the minutes of the time of day.

22
Displays the seconds of the time of day.

23
Displays the time and date synchronization flag. The synchronization flag shows if the module is time-synchronized with the Symphony system. The module receives system time from the network processing module or bus interface module on start-up. Output 23 is a logic 0 until the module receives the current time.

0 = time/date invalid
1 = time/date valid

24
Displays the calendar year (zero to 99).

25
Displays the calendar month (one to 12).

26
Displays the calendar day (one to 31).

27
Displays the day of the week:

1 = Sunday
2 = Monday
3 = Tuesday
4 = Wednesday
5 = Thursday
6 = Friday
7 = Saturday

28 and 29
Reserved.

90.2 Example

90.2.1 Function Block Configuration Required for Time-Synchronization

To collect continuous data correctly in a batch configuration, all elements of the system (Batch Historian, PC View, Human System Interface (HSI), Harmony controllers and Cnet/PN800) must share a system-wide time system. The control system must have a time master, such as Conductor NT or S+ Operations HSI, to generate a system-wide time base. The PC View synchronizes with the control system through six tags configured within the Harmony controller. Those six tags (within the controller) are analog exception report function block (function code 30) outputs from an extended executive block that are

also defined within the PC View tag data base. Figure 90-1 shows an example of the function block configuration required for time-synchronization. The tag names are shown in Table 90-1.

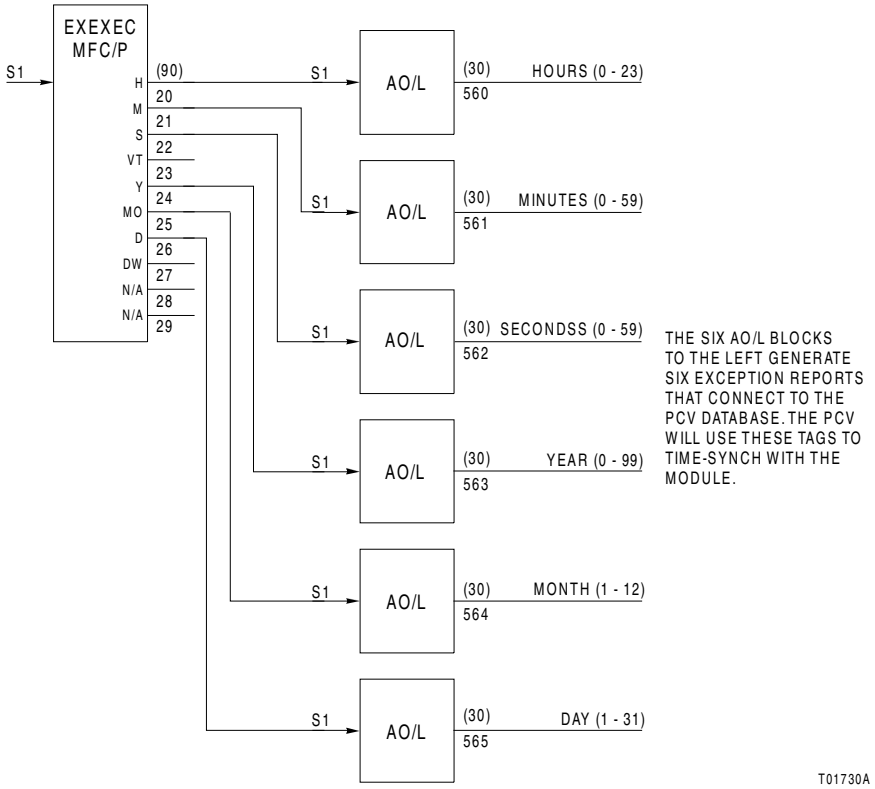


Figure 90-1 Configuration Required for PC View Time-Synchronization

Table 90-1 Available PC View Tag Names

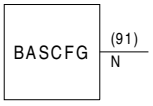
Tag Name	Description
N90HOUR	Current system hours (output 20)
N90MIN	Current system minutes (output 21)
N90SEC	Current system seconds (output 22)
N90YR	Current system year (output 24)
N90MN	Current system month (output 25)
N90DAY	Current system day (output 26)

NOTE: Only one module, in any process control unit on the communication highway, needs to be configured with time-synchronization tags for Batch Historian. The address of the function blocks and their meaning must match the correct tag name.

91. BASIC Configuration (BRC-100/200)

The BASIC configuration function code defines the amount of NVRAM and RAM memory allocated to the BASIC program. It also defines BASIC operating modes. This function code provides one output that the BASIC program sets using the BOUT command. The ***MFC BASIC Programming Language Reference*** provides complete instructions for programming the module with BASIC. Only one BASIC configuration function block can be configured per module.

NOTE: This function code is supported only on the BRC-100/200 and the IMMFP11/12 controllers.



Outputs

Blk	Type	Description
N	R	The BOUT command in the BASIC program sets output N

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	1	I	0 or 1	Action on BASIC error: 0 = trip module 1 = write error data to terminal and suspend BASIC
S2	N	1	I	0 or 1	Load/run flag: 0 = auto startup on restart 1 = manual startup on restart
S3	N	1	I	Note 1	BASIC string space allocation in 1-kbyte increments (NVRAM)
S4	N	1	I	Note 2	BASIC data space allocation in 1-kbyte increments (RAM)
S5	N	1	I	0-63	BASIC program space allocation in 1-kbyte increments (NVRAM)

NOTES:

1. Maximum value is S3 + S5 ≤ 414.
2. Maximum values is S3 + S4 + S5 ≤ 1530.

91.1 Explanation

91.1.1 Specifications

S1 – ERROR

Sets the action taken on a BASIC error.

0 = trip module
1 = write error to data terminal and suspend BASIC

The error writes to the printer or display screen designated in the BASIC program as the data terminal, and the BASIC program stops.

S2 – LRFLAG

Load/run flag. It defines the action to be taken by the BASIC interpreter when an invoke BASIC function block is executed. If zero, then the BASIC program contained in EEROM automatically loads into RAM and executes. If one, then the BASIC interpreter prompts for further action.

0 = auto startup of the invoke BASIC block
1 = manual startup at the invoke BASIC block

S3 – STRSPC

BASIC string space allocation defined in one kilobyte increments. BASIC string space is allocated in RAM memory. Estimate the amount of required string space from the program's declarations section.

S4 – DATSPC
Sets the BASIC data space allocation defined in one kilobyte increments. Data consists of independent variables and constants. BASIC data space is allocated in RAM memory. Estimate the amount of required data space from the program's declarations section.

S5 – PGMSPC
Sets the BASIC program space allocation defined in one kilobyte increments. The program consists of a group of functions that perform specified actions and return values depending on the value of one or more independent variables. BASIC program space is allocated in both NVRAM and RAM memory. The amount of BASIC program space is the BASIC program file size.

91.1.2 Outputs

N
Any value from the BASIC program. The BASIC program specifies output N using the BOUT command.

91.2 Application

Figure 91-1 shows a sample of a BASIC program in a module.

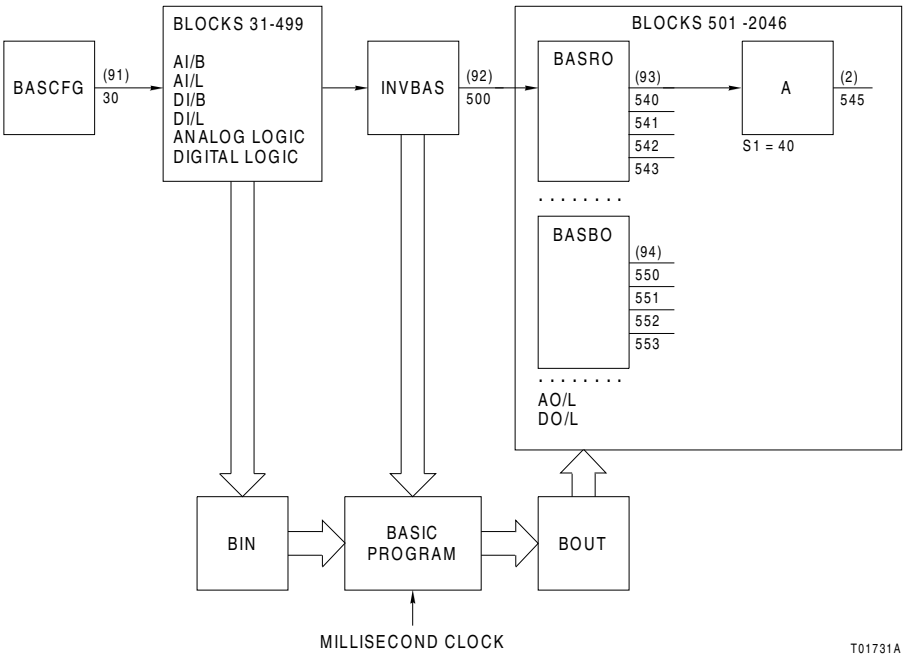


Figure 91-1 Typical BASIC Configuration

Step 1
Place function code 91 (configure BASIC) in block 30. Placement of this function code is user selected.
S1 = 1
Write any errors to the data terminal and suspend BASIC.
S2 = 1
BASIC interpreter prompts for action before running the program. BASIC provides a *ready* reply when addressed by a dumb terminal or personal computer. Specification S2 equals one while it is being programmed. After programming is complete, change S2 to zero. When S2 equals zero, the program runs automatically on module power up.

S3 = 1 = 1k of string space allotted.
S4 = 5 = 5k of data space allotted.
S5 = 10 = 10k of program space allotted.

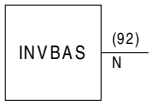
Step 2
Place function code 92 (invoke BASIC) in block 500. Placement of this function code is user selected.
S1 = 1
This block invokes BASIC when executed. Control returns to the next numbered block when BASIC exits.

Step 3
Place the module in execute mode. The ready reply shows on the display. Enter the BASIC statements. Remember to save the BASIC program before changing S2 in block 30 to zero.

92. Invoke BASIC

The invoke BASIC function code causes the controller BASIC interpreter to execute. If automatic load/run has been selected in the configure BASIC function block, the interpreter runs the program. Otherwise it prompts for further action. This block must be in the same segment as the configure BASIC block. A function block segment can contain multiple invoke BASIC blocks, each of which will initiate execution of the BASIC program. This function code provides one output that is selected in the BASIC program with the BOUT command. Figure 92-1 shows a sample BASIC configuration. Reference the ***MFC BASIC Programming Language Reference*** for instructions on programming with BASIC.

NOTE: This function code is supported only on the BRC-100 and the IMMFP11/12 controllers.



Outputs

Blk	Type	Description
N	R	The BOUT command in the BASIC program sets output N

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0.000	R	Full	Initiates execution of the BASIC program by providing a value for SYSVAR

92.1 Explanation

This block initiates execution of the BASIC program by assigning a value to the BASIC variable SYSVAR. SYSVAR is then either an input to the BASIC program itself or selects the entry point of a BASIC program as shown by examples one and two. Figure 92-1 shows a typical controller BASIC configuration.

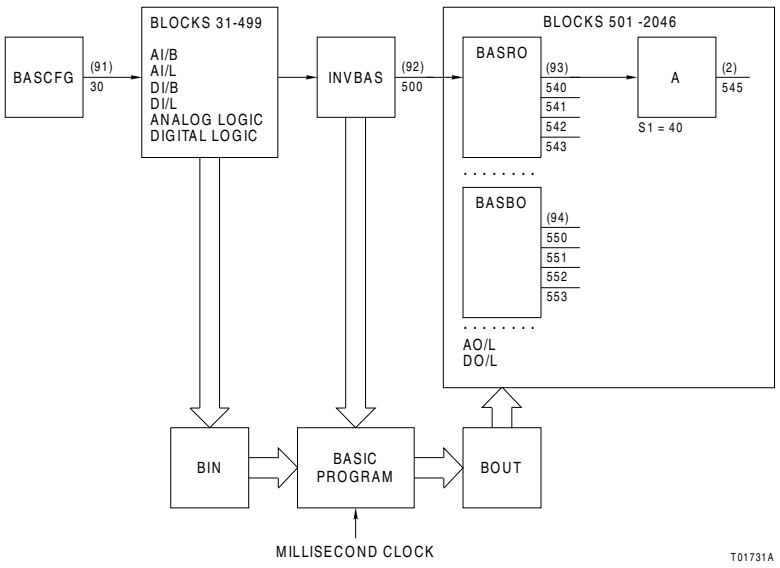


Figure 92-1 Typical BASIC Configuration

92.1.1 Example 1

When SYSVAR is an input, assume:

S1 equals 2 and the first function containing SYSVAR is:

100 A = SYSVAR * 10,

the program begins executing at line 100 and A equals 20.
If there are more than one invoke BASIC block configured, the value of S1 can remain the same or be different in each one. For example, if the second invoke BASIC block has S1 equals three, then A equals 30 for that particular iteration of the program, etc.

92.1.2 Example 2

SYSVAR selects the entry point of a BASIC program:

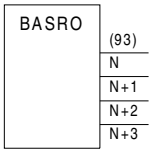
```
10 ON SYSVAR GOTO 100...
```

Every time the program receives a value for SYSVAR, it goes to line 100 and begins execution. When using SYSVAR in this manner (as a trigger), its value is unimportant.

93. BASIC Real Output

The BASIC real output function code provides a means of transmitting real data values from the BASIC, Batch 90 and C Language programs to other function blocks. There are no specifications for this block. The BOUT command in the program defines the four outputs. Figure 0-1 shows a sample BASIC configuration. Refer to the ***MFC BASIC Programming Language Reference*** for instructions on programming with BASIC.

NOTE: This function code is supported only on the BRC-100 and the IMMFP11/12 controllers.



Outputs

Blk	Type	Description
N	R	Program command BOUT sets the output value
N+1	R	
N+2	R	
N+3	R	

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Full	Not used

93.1 Application

Syntax:

BOUT *<block number>, <value>* **BOUT 40, 4.75**

where:

<block number>	=	Valid integer block number configured in the target module.
<value>	=	Arithmetic expression stored as the block output.

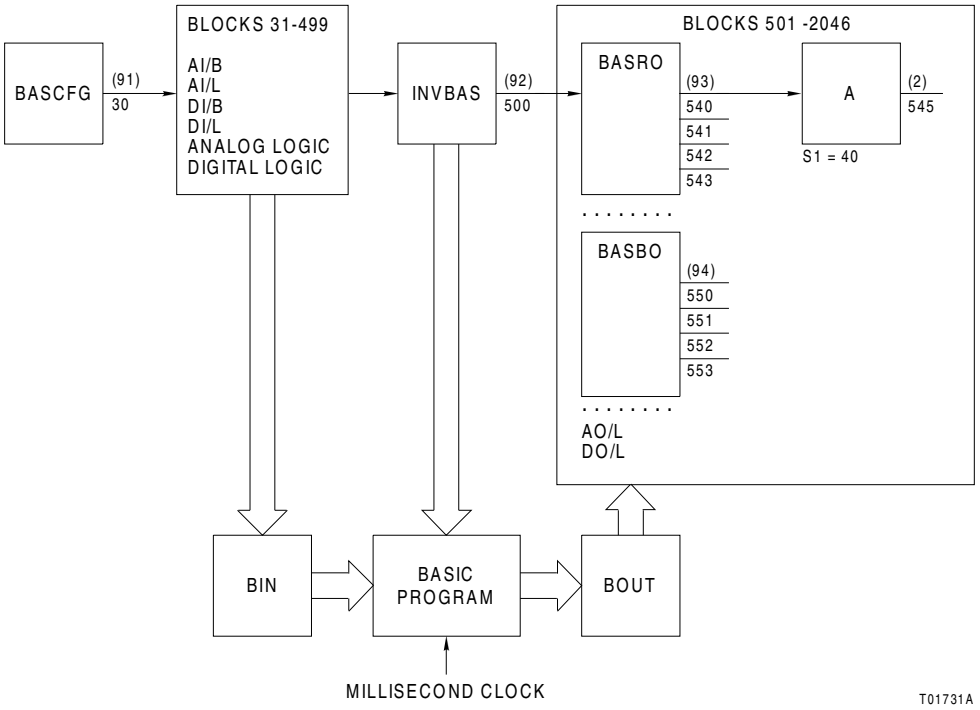


Figure 0-1. Typical BASIC Configuration

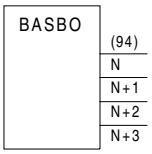
94. BASIC Boolean Output

The BASIC boolean output (BASBO) function code provides a means of transmitting boolean data values from BASIC, Batch 90, or C Language to other function blocks. There are no specifications for this block. The BOUT command in the program defines the four outputs.

NOTE: This function code is supported only on the BRC-100 and the IMMFP11/12 controllers.

For instructions on programming the modules, refer to:

- ***MFC BASIC Programming Language Reference***.
- ***Batch 90 and User Defined Function (UDF) Programming Language Reference***.
- ***C Utility Program***.



Outputs

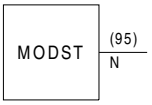
Blk	Type	Description
N	B	Program command BOUT sets the output value
N+1	B	
N+2	B	
N+3	B	

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Full	Not used

95. Module Status Monitor

This function code monitors the status of a designated node or module. A module either monitors the status of itself or another module in the same HCU, or another HAC controller on the same peer-to-peer network (HAC only). Function code 95 monitors selected bits from one of the five or 16 module status bytes. Specification S3 identifies the status byte, and S5 through S12 identify the bits to be monitored. Module mode can be monitored by selecting S4 equals one. If the communication status of the target module is good, it sets the output quality to good, and outputs a logical ORed value. The values of the selected bits in the status byte determine the logical ORed output. If the communication status of the target module is bad, it sets the output quality to bad and the output value is a logic 1. This block generates a problem report when a communication failure exists.



Outputs

Blk	Type	Description
N	B	Selected quality and value: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0.000	R	Note 1	Sample period (in seconds)
S2	N	0	I	Note 2	Target node/module address
S3	Y	0	I	0 - 4 or 0 - 15	Module status byte number (refer to Appendix G and note following S3 explanation)
S4	Y[3]	0	I	00,01,10 or 11	Configuration flags: Target offline detection flag: X0 = no X1 = yes Target rank (HAC only) 0X = monitor primary module status 1X = monitor backup module status[4]
S5	Y	0	I	0 or 1	Bit 0 of selected monitored byte: 0 = no 1 = yes
S6	Y	0	I	0 or 1	Bit 1 of selected monitored byte: 0 = no 1 = yes
S7	Y	0	I	0 or 1	Bit 2 of selected monitored byte: 0 = no 1 = yes
S8	Y	0	I	0 or 1	Bit 3 of selected monitored byte: 0 = no 1 = yes
S9	Y	0	I	0 or 1	Bit 4 of selected monitored byte: 0 = no 1 = yes
S10	Y	0	I	0 or 1	Bit 5 of selected monitored byte: 0 = no 1 = yes
S11	Y	0	I	0 or 1	Bit 6 of selected monitored byte: 0 = no 1 = yes

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S12	Y	0	I	0 or 1	Bit 7 of selected monitored byte: 0 = no 1 = yes

NOTES:
1. The range is the positive real value range, however the practical lower limit is determined by the Base periodic I/O sampling period defined in the extended executive (FC 90, block 20, S2).
2. Range values are: 0-31 for the BRC-100, IMMFP11/12
   1-250 for the HAC
3. The tens digit of S4 is non tunable.
4. Valid only when S2 is configured to monitor the same node address as the controller in which the function code 95 resides.

95.1 Explanation

95.1.1 Specifications

S1 – MBRD
Peer-to-peer network or the Controlway/module bus read time in seconds. This defines the frequency of reading the module status byte. The system allows tuning of the value shown for the update period. However, the original peer-to-peer network or Controlway/module bus sample period will be retained. To change the update period, the module must be placed in configure mode.

S2 – MADR
Address of the target module.

NOTE: Setting S2 to the same node/module address as the node/module address of the controller in which the FC 95 resides causes FC 95 to monitor that controllers module status directly without generating any peer-to-peer network or Controlway/module bus message traffic.

S3 – SBYTE
Module status byte number. This defines which of the five or 16 module status bytes (numbered from zero to four or zero to 15) will be monitored.

NOTE: The correct value for S3 can be obtained by subtracting one from the byte numbers found in the appendix for the controller being used.

Refer to proper controller appendix for more information on module status bytes.

S4 – OFFDET
Identifies if module mode status is part of the information monitored.

0 = no
1 = yes

When the one's digit of S4 equals one, a signal identifying module mode is part of the input to the logical OR to produce the output. The signal is a zero when the module is in execute mode, and a one when the module is in configure or error mode. In the HAC controller, the ten's digit of S4 allows the user to specify which controller of a redundant pair of HACs to be monitored.

0X = Monitor primary module status.
1X = Monitor backup module status. Valid only when S2 is configured to monitor the same node address as the controller in which the function code 95 resides.

NOTE: When utilizing FC95 to monitor the module status of the backup HAC (FC95, S4 = 1X), S8 of FC57 must be set to account for the backup as a separate node.

S5 through S12
BIT0 to BIT7
Bit selection inputs for the eight bits in the module status monitored byte.

If: BIT*n* = 0, the bit is not monitored
If: BIT*n* = 1, the bit is monitored

This function code logically ANDs S5 through S12 with the eight bits of the module status byte. The values resulting from that action are then logically ORed to produce the output value. The equivalent circuit in Figure 95-1 illustrates this action. If the values are:

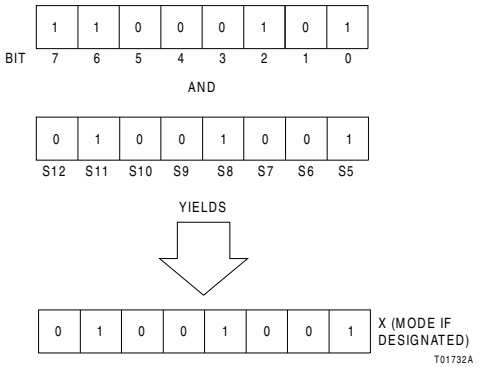


Figure 95-1 Example Bit Values

Then, the values are logically ORed, producing an output of logic 1 since at least one of the inputs equals logic 1.

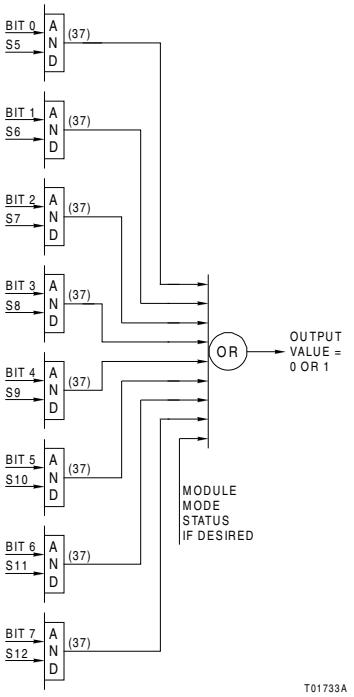
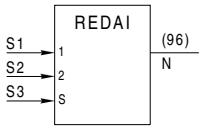


Figure 95-1 Equivalent Circuit

96. Redundant Analog Input

The redundant analog input (REDAI) function code is a transfer between a pair of redundant analog signals. An external logic select input <S3> controls the transfer. The output value is equal to the input value selected with <S3>.
Function code 96 checks the difference between the two inputs and the rate of change of the selected input. Output quality is bad if the quality of the selected input is bad, if the rate of change exceeds the rate limit specified by S5, or if the difference between the two inputs is larger than S4. To test the quality of the signal, include a function code 31 (test quality) block in the configuration. The quality of the point cannot be used as an input to any other type of block. However, the output value can be used as an input to other blocks. Refer to Appendix H, for a definition of point quality.



Outputs

Blk	Type	Description
N	R	Output N = <S1> if <S3> = 0 Output N = <S2> if <S3> = 1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of first input
S2	N	5	I	Note 1	Block address of second input
S3	N	0	I	Note 1	Block address of select input; 0 = <S1>, 1 = <S2>
S4	Y	9.2 E18	R	Full	Deviation limit (same units as <S1> and <S2>)
S5	Y	9.2 E18	R	Full	Rate limit (same units as <S1> and <S2> per sec)
S6	Y	9.2 E18	R	Full	Rate error deadband (same units as <S1> and <S2>)

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

96.1 Explanation

96.1.1 Specifications

S1 – X1
Block address of first input.

S2 – X2
Block address of second input.

S3 – SELCT
Block address of select input. The value in this block determines which input (<S1> or <S2>) is the output.

0 = output <S1>
1 = output <S2>

S4 – DEV
Maximum deviation permitted between <S1> and <S2>. This applies when both <S1> and <S2> are good quality.

S5 – RATE
Maximum allowable rate of change per second for the selected input (same units as <S1> and <S2> per second). If the rate of change minus S6 exceeds S5, output quality is bad.

S6 – DBAND
Defines the deadband for the rate limit. When the rate exceeds the rate limit plus S6, output quality is bad. The deadband prevents excessive bad quality readings when the rate is hovering around the limit.

96.2 Logic Flow

If <S3> = 0; output value and quality = <S1> with quality
If <S3> = 1; output value and quality = <S2> with quality
If <S1> and <S2> quality = good, ***and*** <S1> – <S2> > S4;
output quality = 1 (bad)
If rate of change of output > S5 + S6; output quality = 1 (bad)
where:

<S1>	= Value of first input.
<S2>	= Value of second input.
<S3>	= Value of select input.
0	= <S1>.
1	= <S2>.
S4	= Deviation limit.
S5	= Rate limit.
S6	= Rate deadband.

96.3 Applications

Function code 96 relies on external logic to select and output one of two redundant analog inputs. Figure 96-1 shows the configuration for the output of a function code 98 controlling the selection. When the active module in the slave select block changes from module one to module two, the active input to the redundant analog input block switches from the first input <S1> to the second input <S2>. Configuring the blocks so that inputs one and two are received from modules one and two, respectively, insures that if one module goes bad, the redundant analog input automatically switches to the good module.

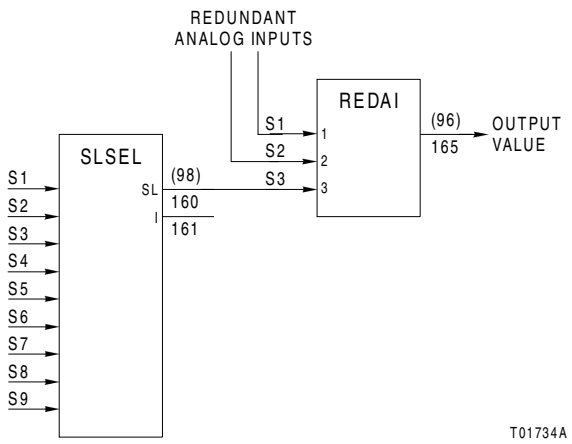
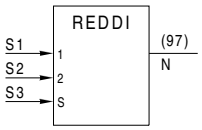


Figure 96-1 Choosing Between Redundant Analog Inputs

97. Redundant Digital Input

The redundant digital input function code selects and transfers a pair of redundant boolean signals. An external logic select input <S3> controls the transfer. The output value is equal to the input value selected with <S3>.
Function code 97 monitors the action of the two inputs <S1> and <S2>. Output quality is bad if the quality of the selected input is bad or if there is a difference between the two inputs when both have good quality. To test the quality of the signal, include a function code 31 in the configuration. The quality of the point cannot be used as an input to any other type of block. However, the output value can be used as an input to other blocks. Refer to Appendix H, for a definition of point quality.



Outputs

Blk	Type	Description
N	B	Output N = <S1> if <S3> = 0 Output N = <S2> if <S3> = 1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of first input
S2	N	0	I	Note 1	Block address of second input
S3	N	0	I	Note 1	Block address of select input; 0 = <S1>, 1 = <S2>

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

97.1 Applications

Function code 97 depends on external logic to select and output one of two redundant boolean inputs. The configuration shown in Figure Figure 97-1 illustrates the output of a function code 98 controlling the selection. If the active module in the function code 98 changes from module one to module two, then the active input in the function code 97 block switches from input one to input two. Configuring the blocks so that inputs one and two correspond to modules one and two insures that if one module goes bad, the input from the other module will automatically be read.

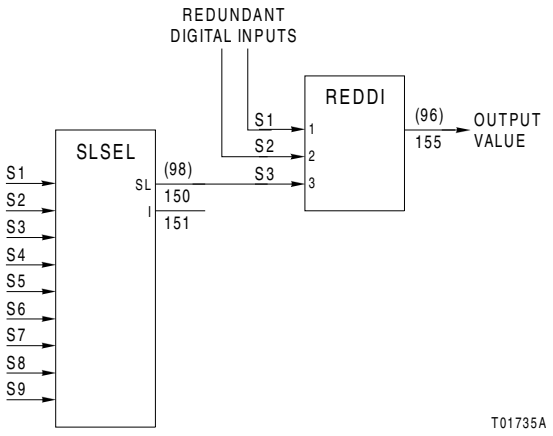
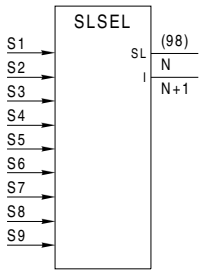


Figure 97-1 Choose Between Redundant Digital Signals Based on Module Quality

98. Slave Select

The slave select function code monitors a pair of redundant I/O module function blocks. Specification S9 is a toggle input that switches from one module to the other when both modules are good.
The block has two outputs, select (N) and interlock (N+1). The select output identifies the active module, and the interlock indicates when both modules are bad.



Outputs

Blk	Type	Description
N	B	Module selected: 0 = module 1 1 = module 2 Initial value is zero. Value changes only if the active module is bad and the inactive module is good, or if both modules are good and the toggle input makes a 0 to 1 transition.
N+1	B	Interlock: 0 = one or both modules good 1 = both modules bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of boolean status for module 1
S2	N	0	I	Note 1	Block address of boolean status for module 1
S3	N	0	I	Note 1	Block address of I/O, boolean or real with quality, for module 1
S4	N	0	I	Note 1	Block address of I/O, boolean or real with quality, for module 1
S5	N	0	I	Note 1	Block address of boolean status for module 2
S6	N	0	I	Note 1	Block address of boolean status for module 2
S7	N	0	I	Note 1	Block address of I/O, boolean or real with quality, for module 2
S8	N	0	I	Note 1	Block address of I/O, boolean or real with quality, for module 2
S9	N	0	I	Note 1	Block address of toggle input (if both modules are good, a 0 to 1 transition on this input changes state of the select output)

NOTE:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

98.1 Specifications

S1 – STTSA1
Block address of the boolean status for module one. Each module has a status associated with it that defines if the I/O carried by the module is valid. Function codes 79 and 83 provide the module status outputs.

S2 – STTSB1
Block address of the boolean status for module one. There are two status specifications for each module because IMDSO14 handles two groups of eight outputs. Function code 83 provides the status on groups of eight points only. Therefore, if redundant modules are to be monitored, there must be a function code 83 block configured for each group of eight outputs. A similar configuration exists for IMDSI12, IMDSI13, IMDSI14, IMDSI15, and IMDSI22 modules used for inputs. If either <S1> or <S2> goes to one, control transfers to module two.

S3 – QIOA1
Block address of the I/O quality for module one. Specifications S3 and S4 define the blocks containing the boolean or real I/O for module one. If the quality of <S3> or <S4> goes bad, control will be transferred to module two, providing it is good quality. If both modules are bad quality, the output is the value from module one, but output N+1 equals logic 1 (both modules are bad). Figure 98-1 shows how to transfer an active I/O module if an analog output goes bad.

S4 – QIOB1
Block address of the I/O quality for module one (same as S3).

S5 – STTSA2
Block address of the boolean status for module two (refer to S1 description).

S6 – STTSB2
Block address of the boolean status for module two (refer to S2 description).

S7 – QIOA2
Block address of the quality for module two (refer to S3 description).

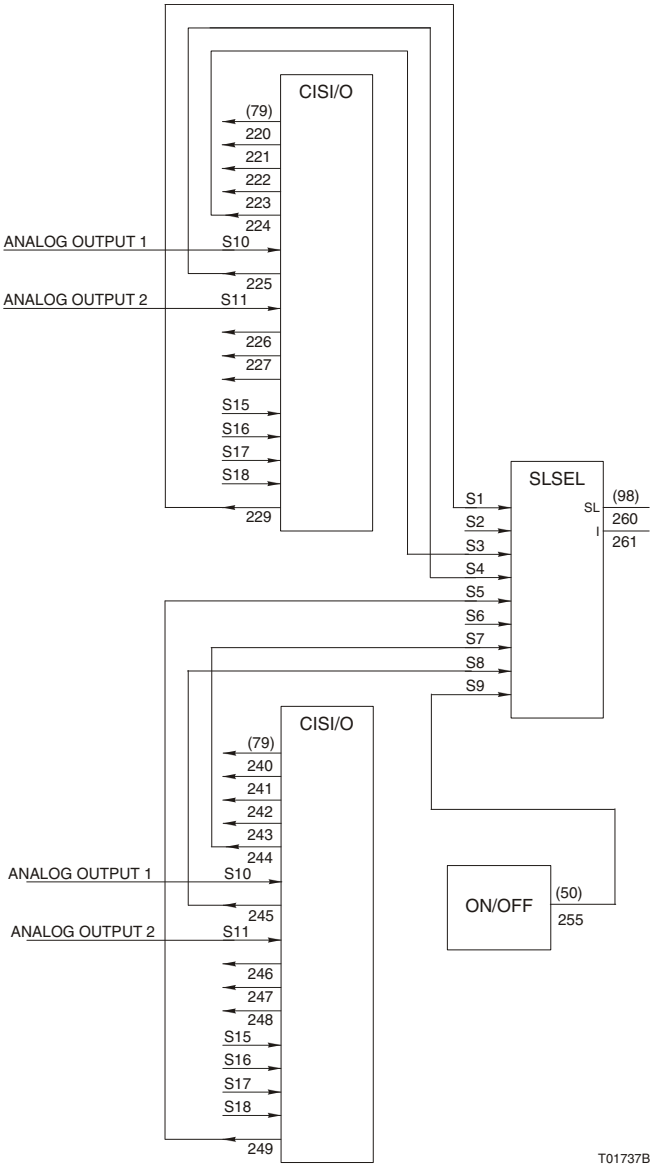


Figure 98-1 Transfer Active I/O Module if an Analog Output Goes Bad

S8 – QIOB2
Block address of the quality for module two (refer to S3 description).

S9 – TOGGLE
Block address of the toggle input. If both modules are good, a logic 0 to 1 transition of <S9> transfers control to the standby module.

98.2 Applications

Figure 98-1 shows the configuration of a slave select block controlling the redundant analog input (function code 96). The analog input selected depends on the module selected in the slave select. This configuration also shows the interlock (N+1) forcing the output value to a predetermined safe value in the event that both modules are bad.

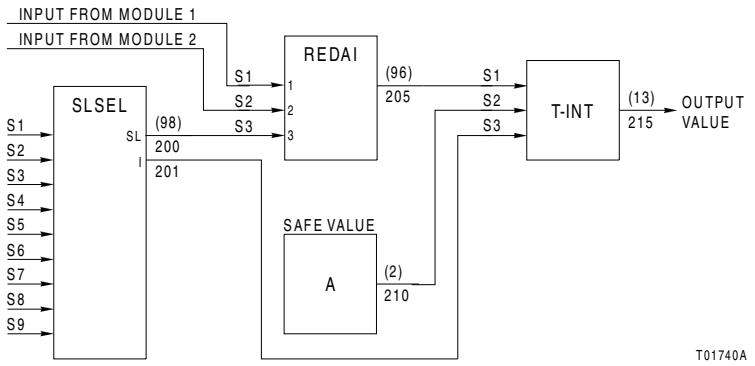


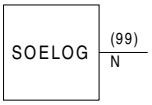
Figure 98-1 Controlling an Analog Output

99. Sequence of Events Log

This block enables a module to communicate with a sequential events recorder (SER). The sequence of events (SOE) log buffers event data received from the SER recorder for human system interface (HSI) access. Five types of SOE logs can be generated: standard, summary, prefault, postfault and snapshot. Each SOE log holds only one type of report, and the report includes data on all points configured in the SER recorder to be that type. Each block is independent and relies only on a common serial link to acquire event data. The SOE log function code specifies buffer size and the age of the event data in the buffer. Configure a single block for each log type, buffer size and age.

NOTE: This function code is supported only on the BRC-100 and the IMMFP11/12 controllers.

The sequence of events log sets local I/O module status to bad and generates a module problem report when the sequence of events recorder global I/O status is bad (i.e., input check failure). Output block N remains good quality on this failure.



Outputs

Blk	Type	Description
N	B	Events logged flag: 0 = no 1 = yes

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 4	Qualifier for sequence of events recorder data received buffer storage: 0 = standard 1 = summary 2 = prefault 3 = postfault 4 = snapshot
S2	N	256	I	0 - 3,500	Number of events that fit in buffer: buffer size = 9(S2+1)
S3	N	360	I	0 - 32,767	Age of event data (in secs) before discarded from buffer

99.1 Explanation

Function code 99 provides a communication link between the SER recorder and an interface device such as a console or computer. Data cannot be sent directly from the recorder to interface devices. The interface must be configured to print the event data received from the module. See the operators manual for the interface device to determine how to configure it to receive sequence of events data. Each sequence of events log in the module has an associated sequence of events configuration in the interface device. The sequence of events log is an exception report block with the same report type and size as a remote control memory (function code 62) exception report. The remote control memory command from the interface device requests a summary report.

NOTE: Dedicate the module used for sequence of events reporting to SER functions only. It cannot be configured for redundancy, BASIC, C language, batch or user defined functions.

The SER recorder is capable of handling up to 512 I/O points. The SER recorder defines point report types for all points. There are five report types available (standard, summary, prefault, postfault and snapshot).

99.1.1 Specifications

S1 – LOG TYPE
Qualifier for SER recorder event data buffer storage.

0 = standard
1 = summary
2 = prefault
3 = postfault
4 = snapshot

Standard
The standard type reports any standard SER recorder point change of state. Changes from normal-to-alarm or from alarm-to-normal result in a time-tagged report generated for that point. The results are stored in the module for the length of time specified by S3.
Summary
The summary report is generated on an operator demand from the interface device. This is a time-tagged report listing all SER recorder points that are not normal (i.e., inactive, deleted from scan, or in alarm). A remote control memory display type should be configured in the interface device to force the output of this function code to a one to request this summary data collection.
Prefault
The prefault mode stores the points when they change state. Up to 1,000 predefined events and a specified time window (up to 24 hours) limit the data stored. For example, this report can be configured to list the last 50 events prior to the prefault trigger, or all events occurring up to ten minutes before the prefault trigger. The trigger is a false to true signal resulting from an event or a series of events. When the trigger occurs, all of the stored data reports, with the oldest stored data reported first. The memory then begins saving new data. The new data will not be reported until the trigger outputs a false to true signal again.
Postfault
The postfault mode reports points when they change state, but only after the postfault trigger turns true. These points will be reported until the postfault trigger turns false, or the operator cancels the postfault state.
Snapshot
Snapshot inputs are typically a group of related points. These points, like any point in the system, can be in the alarm, normal, deleted, or inactive state at any one time. When the snapshot trigger turns true, a report generates detailing the status of the designated snapshot points in numerical point order. Another report generates the next time the snapshot trigger outputs a false to true signal.
All points defined in the SER recorder as a certain type will be saved in the buffer of the sequence of events log defined as the same type. For example, if this log is a standard log (zero), all points defined in the SER recorder as standard are sent to this buffer.

S2 – EVENT NO.
Number of events that fit in the buffer. Each log has a separate buffer that holds a specific number of point values. The buffer is in RAM memory, and the buffer size determines RAM memory utilization.

$$buffer\ size = 9(S2 + 1)$$

S3 – AGE
Maximum age in seconds of the event data in the buffer. After data has been in the buffer for this length of time, it is removed from the buffer. The buffer age is reset to zero every time an event for the specified log occurs.
A new and unique buffer age time is attached to each event saved in the buffer. Older events are therefore removed from the buffer before newer events when their buffer age time expires. Any data stored in the buffer is guaranteed to stay in the buffer for the configured age time.
Specifications S2 and S3 are closely related to the interface device configuration. These parameters control the amount of data the interface device can receive when a trip occurs. The interface device must be configured to provide a fast enough data transfer rate to remove the event data from the buffer before S3 is reached. Once S3 is reached, the data is erased.

99.1.2 Outputs

N
Events logged flag.

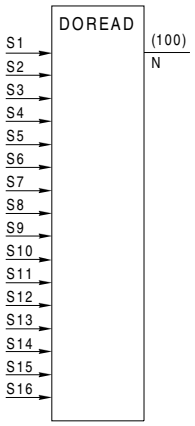
0 = no
1 = yes

Output N shows whether the buffer currently contains data received from the SER recorder. The output goes to one every time information is fed to the block from the SER recorder, and returns to zero when all the data has aged (S3) and is erased.
If multiple or oscillating events occur, the new data is saved in the buffer with a new age time (S3). Then, the output remains a one until all new data in the buffer is aged. If new data continues to filter in at a period less than S3, the output never goes to zero.

NOTE: A console report will not print until this output goes to zero.

100.Digital Output Readback Check

This function code checks the operation of a digital output group. It compares the digital output values with those typically read back via a digital input group.



Outputs

Blk	Type	Description
N	B	Status: 0 = all digital outputs match digital inputs 1 = one or more digital outputs do not match digital inputs

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of DO value 1
S2	N	0	I	Note 1	Block address of DI readback value 1
S3	N	0	I	Note 1	Block address of DO value 2
S4	N	0	I	Note 1	Block address of DI readback value 2
S5	N	0	I	Note 1	Block address of DO value 3
S6	N	0	I	Note 1	Block address of DI readback value 3
S7	N	0	I	Note 1	Block address of DO value 4
S8	N	0	I	Note 1	Block address of DI readback value 4
S9	N	0	I	Note 1	Block address of DO value 5
S10	N	0	I	Note 1	Block address of DI readback value 5
S11	N	0	I	Note 1	Block address of DO value 6
S12	N	0	I	Note 1	Block address of DI readback value 6
S13	N	0	I	Note 1	Block address of DO value 7
S14	N	0	I	Note 1	Block address of DI readback value 7
S15	N	0	I	Note 1	Block address of DO value 8
S16	N	0	I	Note 1	Block address of DI readback value 8

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

100.1Explanation

Figure 100-1 illustrates the internal logic of function code 100. Function code 100 performs internal exclusive OR functions on eight digital output and digital input pairs. The exclusive ORs require that both inputs be the same for the output to equal logic 0. If the inputs to the exclusive OR are not the same, the output equals logic 1.

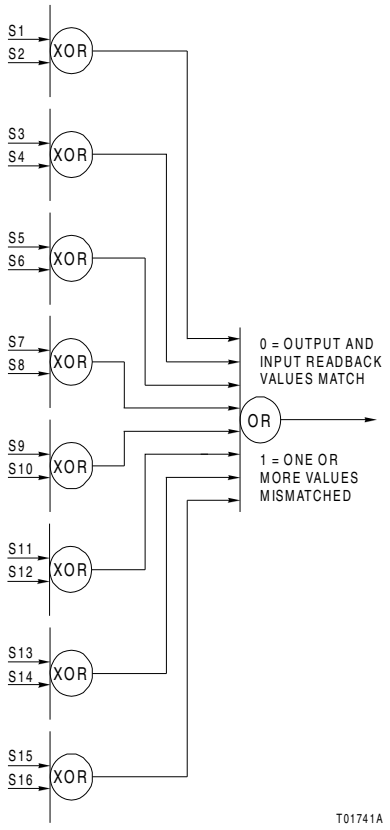
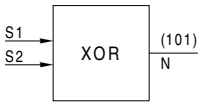


Figure 100-1 Internal Logic

101.Exclusive OR

The exclusive OR function code performs the logical exclusive OR function of two inputs. Function code 101 gives an output of logic 1 when one of two inputs equals logic 1, but not when both equal logic 1 or logic 0.



Outputs

Blk	Type	Description
N	B	Refer to Table 101-1

Table 101-1 Exclusive OR Truth Table

<S1>	<S2>	Output N
0	0	0
0	1	1
1	0	1
1	1	0

Specifications

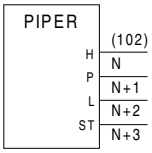
Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of first input
S2	N	0	I	Note 1	Block address of second input

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

102.Pulse Input/Period

The pulse input/period function code interfaces a controller to a selected channel of an IMDSM04 pulse input I/O module. The IMDSM04 module conditions, converts and processes pulse inputs into analog signals. It has eight input channels whose inputs can be processed in three modes: period, frequency or totalization. Function code 102 processes a signal in the period mode. Function codes 103 and 104 process signals in the frequency and totalization modes, respectively. Function code 109 processes the duration of the input pulses. Each input channel must have one and only one pulse input block configured.



Outputs

Blk	Type	Description
N	R	Period (seconds) × gain
N+1	B	Period high alarm: 0 = no alarm 1 = high alarm
N+2	B	Period low alarm: 0 = no alarm 1 = low alarm
N+3	B	I/O module communication status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus I/O module address of IMDSM04
S2	N	1	I	1 - 8	Selected channel
S3	N	0	B	0 or 1	Pulse trigger level: 0 = low to high transitions 1 = high to low transitions
S4	N	1	I	1 - 6	Expected period range: 1 = 0.1 msec - 6.553 secs, ± 0.1 msec 2 = 1.0 msec - 65.53 secs, ± 1.0 msec 3 = 10.0 msecs - 655.3 secs, ± 10.0 msecs 4 = 0.1 sec - 6.553 ksecs, ± 0.1 sec 5 = 1.0 sec - 65.53 ksecs, ± 1.0 sec 6 = 10.0 secs -1 655.3 ksecs, ± 10.0 secs
S5	Y	1.000	R	Full	Gain
S6	Y	9.2 E18	R	Full	High alarm value
S7	Y	-9.2 E18	R	Full	Low alarm value
S8	Y	0.000	R	Full	Spare

102.1Explanation

In the period mode, a counter measures the time between input pulses. Specification S4 configures the expected period range. There are six ranges available with range one having the highest resolution and range six having the lowest resolution. The pulse trigger transition can be configured to respond to either the rising or falling portion of the signal with S3. Specification S5 scales the determined period. The determined period is tested against specified high (S6) and low (S7) alarm values.

102.1.1 Specifications

S1 – SLVADR
Expander bus I/O module address of the IMDSM04 module. Valid addresses are zero to 63.

S2 – CHNL
Identifies the input channel being monitored. IMDSM04 modules have eight input channels. Refer to the IMDSM04 Pulse Input Module product instruction to identify the channel associated with each input.

S3 – PTRIG
Defines the type of signal transition that triggers the counter response.

0 = response on low to high transitions
1 = response on high to low transitions

S4 – RANGE
Defines the range within which the period of the input pulse is expected to fall. There are six period ranges. Resolution decreases from range one to six.

1 = 0.1 millisecond to 6.553 seconds, ± 0.1 millisecond
2 = 1.0 millisecond to 65.53 seconds, ± 1.0 millisecond
3 = 10.0 milliseconds to 655.3 seconds, ± 10.0 milliseconds
4 = 0.1 second to 6.553 kiloseconds, ± 0.1 second
5 = 1.0 second to 65.53 kiloseconds, ± 1.0 second
6 = 10.0 seconds to 655.3 kiloseconds, ± 10.0 seconds

NOTES:
1. If the period of the input pulse becomes larger than the expected range, output N goes to zero. This can be used to indicate a loss of input signal.

2. If the input pulse becomes smaller than the expected range, the output oscillates between 0.0 and the smallest measurable period for the selected range.

S5 – K
Defines the gain multiplier. The gain multiplier is the value used to scale the output to a value that is meaningful to the system.

S6 – HALRM
Value of the output that causes the high alarm output to go to one. Specification S6 is dependent on the values chosen with S4 and S5.

Output = period(gain)

S7 – LALRM
Value of the output that causes the low alarm output to go to one. Specification S7 is dependent on the values chosen with S4 and S5.

Output = period(gain)

S8
Reserved.

102.1.2 Outputs

N
Real value representing the period (in seconds) times the gain.

NOTES:
1. If the period of the input pulse becomes larger than the expected range, output N goes to zero. This can be used to indicate a loss of input signal.

2. If the input pulse becomes smaller than the expected range, the output oscillates between 0.0 and the smallest measurable period for the selected range.

N+1
High alarm indicator.

0 = no alarm
1 = high alarm

N+2
Low alarm indicator.

0 = no alarm
1 = low alarm

N+3
I/O module communication status.

0 = good
1 = bad

I/O modules are bad if:

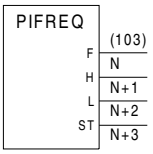
- There is an illegal I/O module response.
- There is no I/O module response.

or

- An input pulse has not been received.

103.Pulse Input/Frequency

The pulse input/frequency function code interfaces a controller to a selected channel of an IMDSM04 Pulse Input Module. The IMDSM04 module conditions, converts and processes pulse inputs into analog signals. It has eight inputs that can be processed in three modes: period, frequency and totalization. Function code 103 configures a signal to output a frequency value. Function codes 102 and 104 process signals in the period and totalization modes, respectively. Function code 109 processes the duration of the input pulses. Each input channel must have only one pulse input block configured.



Outputs

Blk	Type	Description
N	R	Frequency (Hz) × gain
N+1	B	Frequency high alarm: 0 = no alarm 1 = high alarm
N+2	B	Frequency low alarm: 0 = no alarm 1 = low alarm
N+3	B	I/O module communication status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus I/O module address of IMDSM04
S2	N	1	I	1 - 8	Selected channel
S3	N	0	B	0 or 1	Pulse trigger level: 0 = low to high transitions 1 = high to low transitions
S4	N	1	I	1 - 6	Expected frequency range: 1 = 0.15 mHz - 10 mHz, ± 0.01 mHz 2 = 1.5 mHz - 100 mHz, ± 0.1 mHz 3 = 15 mHz - 1.0 Hz, ± 1 mHz 4 = 0.1 Hz - 6.55 kHz, ± 0.1 Hz 5 = 1.0 Hz - 50 kHz, ± 1.0 Hz 6 = 10.0 Hz - 50 kHz, ± 10.0 Hz
S5	Y	1.000	R	Full	Gain
S6	Y	9.2 E18	R	Full	High alarm value
S7	Y	-9.2 E18	R	Full	Low alarm value
S8	Y	0.000	R	Full	Spare

103.1Explanation

In the frequency mode, a counter records the number of input pulses or cycles that occur per second. The expected frequency range is configurable with S4. There are six frequency ranges available with range one having the highest resolution (± 0.15 millihertz) but smallest span, compared to range six with the lowest resolution (10.0 hertz) but largest span. The pulse trigger transition can be configured to respond to either the rising or falling portion of the signal with S3. The determined frequency is scaled with S5 and tested against specified high and low alarm values specified by S6 and S7.

103.1.1Specifications

S1 – SLVADR
Expander bus I/O module address of the IMDSM04 module. Valid addresses are zero through 63.

S2 – CHNL
Identifies the input channel being monitored. IMDSM04 modules have eight input channels. Refer to the IMDSM04 Pulse Input Module product instruction to identify the channel associated with each input.

S3 – PTRIG
Defines the type of signal transition that triggers the counter response.

0 = response on low to high transitions
1 = response on high to low transitions

S4 – RANGE
Defines the range within which the frequency of the input pulses is expected to fall. There are six frequency ranges shown below. Resolution decreases from range one to six.

1 = 0.15 millihertz to 10 millihertz, ± 0.01 millihertz
2 = 1.5 millihertz to 100 millihertz, ± 0.1 millihertz
3 = 15 millihertz to 1.0 hertz, ± 1.0 millihertz
4 = 0.1 hertz to 6.55 kilohertz, ± 0.1 hertz
5 = 1.0 hertz to 50 kilohertz, ± 1.0 hertz
6 = 10.0 hertz to 50 kilohertz, ± 10.0 hertz

If the frequency goes above the expected range (ranges one through six), the output stops with the last valid frequency and the I/O module communication status becomes a logic 1. This also occurs on ranges one, two and three if the frequency goes below the expected range.

S5 – K
Defines the gain multiplier. Gain scales the output to a useful value.

S6 – HALRM
High alarm output (N+1 equals logic 1). Specification S6 is dependent on the values chosen with S4 and S5.

Output = frequency(gain)

S7 – LALRM
Low alarm output (N+2 equals logic 1). Specification S7 is dependent on the values chosen with S4 and S5.

Output = frequency(gain)

S8
Reserved.

103.1.2 Outputs

N
Real value representing the frequency (in hertz) times the gain.

N+1
High alarm.

0 = no alarm
1 = high alarm

N+2
Low alarm.

0 = no alarm
1 = low alarm

N+3
I/O module communication status.

0 = good
1 = bad

The I/O module is marked bad if:

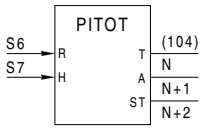
- There is an illegal I/O module response or no I/O module response.

- No pulses have been received yet.
- The frequency exceeds the expected range.
- The frequency goes below the expected range for S4 equal to one, two or three.

104.Pulse Input/Totalization

The pulse input/totalization function code interfaces a controller to a selected channel of an IMDSM04 pulse input slave module. The IMDSM04 module conditions, converts and processes pulse inputs into analog signals. It has eight inputs that can be processed in three modes: period, frequency and totalization. Function code 104 configures a signal to output a totalized value. Function codes 102 and 103 process signals in the period and frequency modes, respectively. Function code 109 processes the duration of the input pulses. Each input channel must have only one pulse input block configured.

Outputs



Blk	Type	Description
N	R	Current totalized value × gain
N+1	B	Totalized value alarm: 0 = no alarm 1 = alarm
N+2	B	I/O module communication status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus I/O module address of IMDSM04 module
S2	N	1	I	1 - 8	Selected channel
S3	N	0	B	0 or 1	Pulse trigger level: 0 = low to high transitions 1 = high to low transitions
S4	Y	0.000	R	Full	Starting value
S5	N	0	I	0 or 1	Totalization direction: 0 = positive 1 = negative
S6	N	0	I	Note 1	Block address of reset flag: 0 = continue totalization 1 = reset totalization
S7	N	0	I	Note 1	Block address of hold flag: 0 = continue totalization 1 = hold accumulated total
S8	Y	0	B	Full	Automatic reset on alarm limit flag: 0 = off 1 = when alarm, reset to S4 + alarm overrun
S9	Y	1.000	R	0 - 9.2 E18	Gain
S10	Y	9.2 E18	R	Full	Alarm limit

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

104.1Explanation

In the totalization mode, an internal counter records the number of input pulses up to 1.9×10^{19} or until the counter resets. The trigger transition level, totalization direction, and starting value are configurable. The reset input sets the totalizer to the starting value. This is useful for reset events based on time. An automatic reset on alarm can be specified. This allows for the totalizer to be set to the starting value plus the alarm overrun when an alarm condition occurs. This can be used for

reset events based on accumulated totals since the count is not lost. A configurable alarm limit determines alarm conditions. A hold input provides temporary stop totalization, and a gain parameter provides a scaled output.

104.1.1 Specifications

S1 – SLVADR
I/O module expander bus address of the IMDSM04 module. Valid addresses are zero through 63.

S2 – CHNL
Identifies the input channel being monitored. IMDSM04 modules have eight input channels. Refer to the ***IMDSM04 Pulse Input Module*** product instruction to identify the channel associated with each input.

S3 – PTRIG
Defines the type of signal transition that triggers the counter response.

0 = response on low to high transitions
1 = response on high to low transitions

S4 – STRT
Initial value of the count on startup and function code reset. Specification S4 plus the alarm overrun is the value of the count after an alarm if the alarm limit flag is set to automatic reset.

S5 – DIR
Direction of the count.

0 = positive
1 = negative

S6 – RST
Block address of the reset flag.

0 = continue totalization
1 = reset count to starting value (S4)

S7 – HOLD
Block address of the accumulated total hold flag. When a hold flag (logic 1) releases (logic 0), the counter resumes counting without resetting.

0 = continue totalization (normal)
1 = hold accumulated total

S8 – AUTO
Automatic reset on alarm flag. This value determines the counter action when the alarm limit is reached. The counter can either continue counting beyond the alarm limit, or reset the counter to S4 plus alarm overrun.

0 = off
1 = reset to S4 plus alarm overrun when alarm limit is exceeded

NOTE: Alarm overrun is the current count reset to the difference between the counter and the alarm limit. This allows the block to correctly detect the next alarm without losing track of the true counts.

S9 – K
Defines the gain multiplier. Gain scales the output to a useful value. A negative gain does not affect the output.

S10 – ALRM
Alarm limit. When the alarm limit is reached, the N+1 output becomes a logic 1, signaling the system that the totalized value is in alarm.

104.1.2 Outputs

N
Current totalized value times gain.

N+1
Total value alarm indicator.

0 = no alarm
1 = alarm

N+2
I/O module communication status.

0 = good
1 = bad

I/O modules are bad if:

- There is an illegal I/O module response.
- There is no I/O module response.
- There is an IMDSM04 counter overflow (count > 1.9×10^{19}).

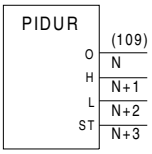
  or

- There is a counter overflow.

NOTE: A counter overflow indicates the counter input is transitioning too fast for the configured scan rate of the segment running the function code. This is a false indication of a bad I/O module and this indication should be considered when configuring this block.

109.Pulse Input/Duration

The pulse input/duration function code interfaces a controller to a selected channel of an IMDSM04 Pulse Input Slave Module. Function code 109 specifies the duration of the input pulses. The IMDSM04 module conditions, converts and processes pulse inputs into analog signals. It has eight input channels whose inputs can be processed in three modes (period, frequency and totalization). Function codes 102, 103 and 104 process signals in the period, frequency and totalization modes, respectively. Each input channel must have one and only one pulse input block configured.



Outputs

Blk	Type	Description
N	R	Pulse duration (seconds) × gain
N+1	B	Pulse duration high alarm: 0 = no alarm 1 = high alarm
N+2	B	Pulse duration low alarm: 0 = no alarm 1 = low alarm
N+3	B	I/O module communication status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus I/O module address of IMDSM04
S2	N	1	I	1 - 8	Selected channel
S3	N	0	B	0 or 1	Pulse trigger level (start of duration measurements): 0 = low to high transitions 1 = high to low transitions
S4	N	1	I	1 - 6	Expected pulse duration range: 1 = 0.1 msec to 6.55 secs 2 = 1.0 msec to 65.5 secs 3 = 10.0 msecs to 655.4 secs 4 = 0.1 sec to 6,553.5 secs
S4 *(cont.)*	N	1	I	1 - 6	5 = 1.0 sec to 65,535 secs 6 = 10.0 secs to 655,350 secs
S5	Y	1.000	R	Full	Gain
S6	Y	9.2 E+18	R	Full	High alarm value
S7	Y	-9.2 E+18	R	Full	Low alarm value
S8	Y	0.000	R	Full	Reserved

109.1Explanation

The pulse input/duration function code specifies an expected pulse duration. It outputs the duration (multiplied by a gain, S5) of the last pulse to occur during the configured pulse duration range S4. If no pulse occurs, the output remains at the last non-zero value. It provides alarms if the pulse input exceeds the selected high and low limits. The trigger transition level (begin count on low-to-high or high-to-low transition) and gain for the output can be selected.

109.1.1Specifications

S1 – SLVADR
Expander bus I/O module address of IMDSM04 module. Valid addresses are zero to 63.

S2 – CHNL
Identifies the input channel being monitored. IMDSM04 modules have eight input channels. Refer to the ***IMDSM04 Pulse Input Module*** product instruction to identify the channel associated with each input.

S3 – PTRIG
Defines the type of signal transition that triggers the counter response.

0 = response on low to high transitions
1 = response on high to low transitions

S4 – RANGE
Defines the expected range of the pulse duration. The ranges are:

1 = 0.1 millisecond to 6.55 seconds
2 = 1.0 millisecond to 65.5 seconds
3 = 10.0 milliseconds to 655.4 seconds
4 = 0.1 second to 6,553.4 seconds
5 = 1.0 second to 65,535 seconds
6 = 10.0 seconds to 655,350 seconds

If the input pulse duration is greater than the expected range, output block N freezes with the last valid pulse duration and the I/O module communication status changes to a one (bad).

S5 – K
Defines the gain multiplier: the value that scales the output to a useful value for control purposes.

S6 – HALRM
Pulse duration high alarm. When the measured pulse duration exceeds this value, output N+1 equals one.

S7 – LALRM
Pulse duration low alarm value. When the measured pulse duration is less than this value, output N+2 equals one.

S8
Reserved.

109.1.2Outputs

N
Pulse duration (seconds) times the gain.

N+1
Pulse duration high alarm indicator.

0 = no alarm
1 = pulse duration has exceeded high alarm limit set by S6

N+2
Pulse duration low alarm indicator.

0 = no alarm
1 = pulse duration is shorter than low alarm limit set by S7

N+3
I/O module communication status.

0 = good
1 = bad

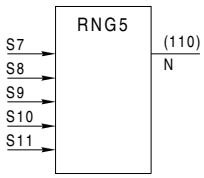
I/O module is marked bad if:

- There is an illegal or no I/O module response.
- No pulses have been received yet.
- The pulse duration exceeds the expected range.

110.Rung (5-Input)

Function code 110 implements a rung of a Ladder logic program (relay type). It accepts five boolean input signals and performs a fundamental operation on each input in sequential order. After each input is ***operated on***, the resulting value goes to the top of the stack. The output of the rung block is the value on the top of the stack unless S1 forces it to another value.

Outputs



Blk	Type	Description
N	B	Output value determined by S1 and value on top of stack If S1 = 0, output = value on top of stack If S1 = 1, output = previous value of output If S1 = 2, output = logic 0 If S1 = 3, output = logic 1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0	I	0 - 3	Output descriptor: 0 = normal 1 = hold previous value 2 = force output to logic 0 3 = force output to logic 1
S2	Y	0	I	0 - 242	Operation performed on input 1
S3	Y	0	I	0 - 242	Operation performed on input 2
S4	Y	0	I	0 - 242	Operation performed on input 3
S5	Y	0	I	0 - 242	Operation performed on input 4
S6	Y	0	I	0 - 242	Operation performed on input 5
S7	N	0	I	Note 1	Block address of input 1
S8	N	0	I	Note 1	Block address of input 2
S9	N	0	I	Note 1	Block address of input 3
S10	N	0	I	Note 1	Block address of input 4
S11	N	0	I	Note 1	Block address of input 5

NOTES:

1. Maximum values are: 9,998 for the BRC-100 , IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

110.1Explanation

This block accepts five boolean inputs, performs a specified fundamental operation on each input in turn, and provides an output dependent on the results of the operations and the output mode selected with S1.
The controller takes a Ladder program entered on a human system interface (HSI) and translates it to a group of rung blocks internally (refer to the ***Ladder Programming (SLAD)*** product instruction for operating instructions). Any changes made to the Ladder program after the conversion are easily saved by downloading them to the controller. This insures that all information is in the proper format when it is processed.
The function code the rung is translated to depends on the number of inputs to the rung. This is automatically defined in the Harmony controllers by the PC90 Ladder programmer. Figures 110-1 and 110-1 illustrate how to use the rung function codes (110, 111, 112) without the PC90 Ladder programmer.
AND, OR and PUT operations are performed sequentially on inputs one through five as specified with S2 through S6. The PUT operation places the specified value on top of a stack of values. The result of the most current operation always goes to the top of the stack. The value resulting from operation one is placed on a ***stack*** that has an initial value of one in the harmony controllers. Operation two is then performed, and the resulting value becomes the value on top of the stack, and so on through operation five. Once all operations have been performed, the controller reads S1 to determine the output

value. If it is zero, the value from the top of the stack (the result of operation five) is the output. Otherwise, the output is overridden and forced to zero or one or held from the previous output. The value on the top of the stack reverts to the initial value at the beginning of each controller execution cycle, so the first operation should always be a PUT to insure that the operations are performed on the desired values.

110.1.1 Specifications

S1 – OUT
(Output descriptor) Defines the output:

0 = Normal output. The value of the output will be the value on top of the stack when all operations on inputs are complete.

1 = Hold previous output. The value of the output will be the previous output value, regardless of the value on top of the stack.

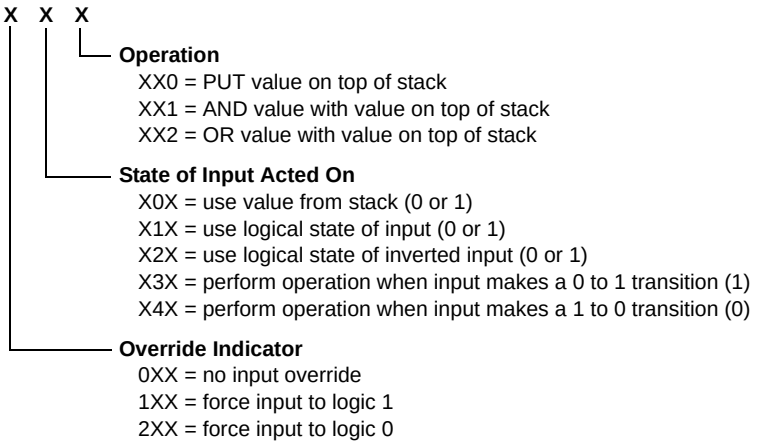
2 = Force output to logic 0. The value of the output will be logic 0 regardless of the value on top of the stack.

3 = Force output to logic 1. The value of the output will be logic 1 regardless of the value on top of the stack.

S2 to S6 – F1 to F5
Identify:

1. The operation to be performed on the input.
2. The input value the operation is performed on.
3. The input override.

Specification information for S2 through S6 is in the format:



S7 to S11 –
IN1 to IN5
Block addresses of inputs one through five.

110.1.2 Outputs

N
Dependent on value on top of the stack and the value of S1.

- If S1 = 0, output = value on top of the stack.
- If S1 = 1, output = previous value of output, regardless of the value on top of the stack.
- If S1 = 2, output = logic 0, regardless of the value on top of the stack.
- If S1 = 3, output = logic 1, regardless of the value on top of the stack.

110.2 Applications

Figures 110-1 and 110-1 illustrate the operations the Ladder programmer performs internally. Ladder logic uses the logic states of various inputs to drive devices. Figure 110-1 is one rung of a Ladder program. Figure 110-1 is the AND/OR logic

representing that rung. The specifications list shows the information entered by the operator in order to implement this logic in a controller when not using the Ladder programmer. The Ladder programmer translates the Ladder logic created by the operator directly into rung function blocks that can be downloaded to a controller.

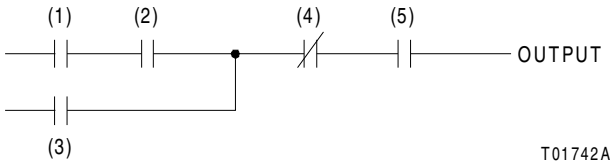


Figure 110-1 Four-Input Logic Rung

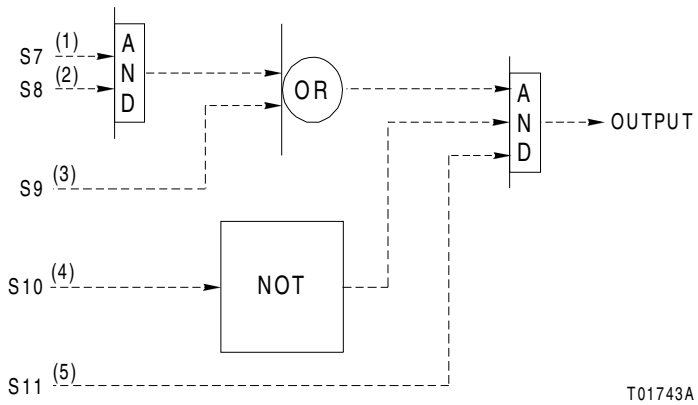


Figure 110-1 AND-OR Logic Representation of Ladder Rung in Figure 110-1

110.2.1 Specifications

S1 – 0
Output value on top of stack.

S2 – 010
No input override, use logical state of input, PUT value on top of stack.

S3 – 011
No input override, use logical state of input, AND with value on top of stack.

S4 – 012
No input override, use logical state of input, OR with value on top of stack.

S5 – 021
No input override, use logical state of inverted input, AND with value on top of stack.

S6 – 011
No input override, use logical state of inverted input, AND with value on top of stack.

S7
Block address of input one.

S8
Block address of input two.

S9
Block address of input three.

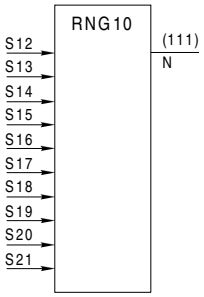
S10
Block address of input four.

S11
Block address of input five.
The circuit is complete when input five equals one and input four equals zero, and either inputs one and two, or input three is true (Fig. 110-1).

NOTE: The PC90 Ladder programmer is best suited for Ladder programming type logic. Refer to the ***Ladder Programming (SLAD)*** instruction.

111.Rung (10-Input)

Function code 111 implements a rung of a Ladder logic program (relay type). It accepts ten boolean input signals and performs a fundamental operation on each input in sequential order. After each input is ***operated on***, the resulting value goes to the top of the stack. The output of the rung block is the value on the top of the stack unless S1 forces it to another value.



Outputs

Blk	Type	Description
N	B	Output value determined by S1 and value on top of stack. If S1 = 0, output = value on top of stack If S1 = 1, output = previous value of output If S1 = 2, output = logic 0 If S1 = 3, output = logic 1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0	I	0 - 3	Output descriptor: 0 = normal 1 = hold previous value 2 = force output to logic 0 3 = force output to logic 1
S2	Y	0	I	0 - 242	Operation performed on input 1
S3	Y	0	I	0 - 242	Operation performed on input 2
S4	Y	0	I	0 - 242	Operation performed on input 3
S5	Y	0	I	0 - 242	Operation performed on input 4
S6	Y	0	I	0 - 242	Operation performed on input 5
S7	Y	0	I	0 - 242	Operation performed on input 6
S8	Y	0	I	0 - 242	Operation performed on input 7
S9	Y	0	I	0 - 242	Operation performed on input 8
S10	Y	0	I	0 - 242	Operation performed on input 9
S11	Y	0	I	0 - 242	Operation performed on input 10
S12	N	0	I	Note 1	Block address of input 1
S13	N	0	I	Note 1	Block address of input 2
S14	N	0	I	Note 1	Block address of input 3
S15	N	0	I	Note 1	Block address of input 4
S16	N	0	I	Note 1	Block address of input 5
S17	N	0	I	Note 1	Block address of input 6
S18	N	0	I	Note 1	Block address of input 7
S19	N	0	I	Note 1	Block address of input 8
S20	N	0	I	Note 1	Block address of input 9

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S21	N	0	I	Note 1	Block address of input 10

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

111.1 Explanation

This block accepts ten boolean inputs, performs a specified fundamental operation on each input in turn, and provides an output dependent on the results of the operations and the output mode selected with S1.
The controller internally translates a Ladder program entered on an engineering work station or another programming terminal and translates it to a group of rung blocks (refer to the ***Ladder Programming (SLAD)*** product instruction for operating instructions). Any changes made to the Ladder program after the conversion are easily saved by downloading them to the controller. This insures that all information is in the proper format when it is processed.
The function code that the rung is translated to depends on the number of inputs to the rung. This is automatically defined in the controller by the PC90 Ladder programmer. Refer to function code 110 for an example of using the rung function codes (110, 111, 112) without the PC90 Ladder programmer.
AND, OR and PUT operations are performed sequentially on inputs one through ten as specified with S2 through S11. The PUT operation places the specified value on top of a stack of values. The result of the most current operation always goes to the top of the stack. The value resulting from operation one is placed on a stack with an initial value of one in the Harmony controllers. Operation two is then performed, and the resulting value becomes the value on top of the stack, and so on through operation ten.
Once all operations have been performed, the controller reads S1 to determine the output value. If it is zero, the value from the top of the stack (the result of operation ten) is the output. Otherwise the output is overridden and forced to zero or one or held from the previous output. The value on the top of the stack reverts to the initial value at the beginning of each controller execution cycle, so the first operation should always be a PUT to insure that the operations are performed on the desired values.

111.1.1 Specifications

S1 – OUT
(Output descriptor) Defines the output:

0 = Normal output. The value of the output will be the value on top of the stack when all operations on inputs are complete.

1 = Hold previous output. The value of the output will be the previous output value, regardless of the value on top of the stack.

2 = Force output to logic 0. The value of the output will be logic 0 regardless of the value on top of the stack.

3 = Force output to logic 1. The value of the output will be logic 1 regardless of the value on top of the stack.

S2 to S11 – F1 to F10
Identify:

- The operation to be performed on the input.
- The input value the operation is performed on.
- The input override.

Specification information for S2 through S11 is in the format:

X X X

Operation
XX0 = PUT value on top of stack
XX1 = AND value with value on top of stack
XX2 = OR value with value on top of stack

State of Input Acted On
X0X = use value from stack (0 or 1)
X1X = use logical state of input (0 or 1)
X2X = use logical state of inverted input (0 or 1)
X3X = perform operation when input makes a 0 to 1 transition (1)
X4X = perform operation when input makes a 1 to 0 transition (0)

Override Indicator
0XX = no input override
1XX = force input to logic 1
2XX = force input to logic 0

S12 to S21 –
IN1 to IN10
Block addresses of inputs one through ten.

111.1.2 Outputs

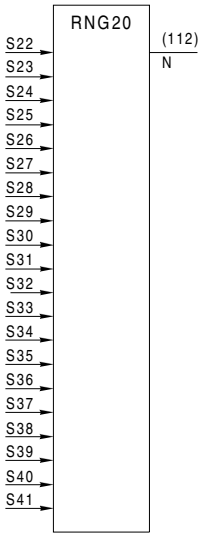
N
Dependent on value on top of the stack and the value of S1.
If S1 = 0, output = value on top of the stack.
If S1 = 1, output = previous value of output, regardless of the value on top of the stack.
If S1 = 2, output = logic 0, regardless of the value on top of the stack.
If S1 = 3, output = logic 1, regardless of the value on top of the stack.

111.2 Applications

Refer to the example in function code 110.

112.Rung (20-Input)

Function code 112 implements a rung of a Ladder logic program (relay type). It accepts 20 boolean input signals and performs a fundamental operation on each input in sequential order. After each input is ***operated on***, the resulting value goes to the top of the stack. The output of the rung block is the value on the top of the stack unless S1 forces it to another value.



Outputs

Blk	Type	Description
N	B	Output value determined by S1 and value on top of stack. If S1 = 0, output = value on top of stack If S1 = 1, output = previous value of output If S1 = 2, output = logic 0 If S1 = 3, output = logic 1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0	I	0 - 3	Output descriptor: 0 = normal 1 = hold previous value 2 = force output to logic 0 3 = force output to logic 1
S2	Y	0	I	0 - 242	Operation performed on input 1
S3	Y	0	I	0 - 242	Operation performed on input 2
S4	Y	0	I	0 - 242	Operation performed on input 3
S5	Y	0	I	0 - 242	Operation performed on input 4
S6	Y	0	I	0 - 242	Operation performed on input 5
S7	Y	0	I	0 - 242	Operation performed on input 6
S8	Y	0	I	0 - 242	Operation performed on input 7
S9	Y	0	I	0 - 242	Operation performed on input 8
S10	Y	0	I	0 - 242	Operation performed on input 9
S11	Y	0	I	0 - 242	Operation performed on input 10
S12	Y	0	I	0 - 242	Operation performed on input 11
S13	Y	0	I	0 - 242	Operation performed on input 12
S14	Y	0	I	0 - 242	Operation performed on input 13
S15	Y	0	I	0 - 242	Operation performed on input 14
S16	Y	0	I	0 - 242	Operation performed on input 15
S17	Y	0	I	0 - 242	Operation performed on input 16
S18	Y	0	I	0 - 242	Operation performed on input 17
S19	Y	0	I	0 - 242	Operation performed on input 18
S20	Y	0	I	0 - 242	Operation performed on input 19

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S21	Y	0	I	0 - 242	Operation performed on input 20
S22	N	0	I	Note 1	Block address of input 1
S23	N	0	I	Note 1	Block address of input 2
S24	N	0	I	Note 1	Block address of input 3
S25	N	0	I	Note 1	Block address of input 4
S26	N	0	I	Note 1	Block address of input 5
S27	N	0	I	Note 1	Block address of input 6
S28	N	0	I	Note 1	Block address of input 7
S29	N	0	I	Note 1	Block address of input 8
S30	N	0	I	Note 1	Block address of input 9
S31	N	0	I	Note 1	Block address of input 10
S32	N	0	I	Note 1	Block address of input 11
S33	N	0	I	Note 1	Block address of input 12
S34	N	0	I	Note 1	Block address of input 13
S35	N	0	I	Note 1	Block address of input 14
S36	N	0	I	Note 1	Block address of input 15
S37	N	0	I	Note 1	Block address of input 16
S38	N	0	I	Note 1	Block address of input 17
S39	N	0	I	Note 1	Block address of input 18
S40	N	0	I	Note 1	Block address of input 19
S41	N	0	I	Note 1	Block address of input 20

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

112.1Explanation

This block accepts 20 boolean inputs, and performs a specified fundamental operation on each input in turn. The block then provides an output dependent on the results of the operations and the output mode selected with S1.
The controller takes a Ladder program entered on an engineering work station or another programming terminal and translates it to a group of rung blocks internally (refer to the ***Ladder Programming (SLAD)*** product instruction for operating instructions). Any changes made to the Ladder program after the conversion are easily saved by downloading them to the controller. This insures that all information is in the proper format when it is processed.
The function code that the rung is translated to depends on the number of inputs to the rung. This is automatically defined in the controller by the PC90 Ladder programmer. Using the rung function codes (110, 111, 112) without the PC90 Ladder programmer is illustrated in function code 110.
AND, OR and PUT operations are performed sequentially on inputs one through 20 as specified with S2 through S21. The PUT operation places the specified value on top of a stack of values. The result of the most current operation always goes to the top of the stack. The value resulting from operation one is placed on a stack with an initial value of one in the Harmony controllers. Then, operation two is performed, and the resulting value becomes the value on top of the stack, and so on through operation 20. Once all operations have been performed, the controller reads S1 to determine the output value. If it is zero, the value from the top of the stack (the result of operation 20) is the output. Otherwise the output is overridden and forced to zero or one or held from the previous output. The value on the top of the stack reverts to the initial value at the beginning of each controller execution cycle, so the first operation should always be a PUT to insure that the operations are performed on the desired values.

112.1.1 Specifications

S1 – OUT
(Output descriptor) Defines the output:

0 = Normal output. The value of the output will be the value on top of the stack when all operations on inputs are complete.

1 = Hold previous output. The value of the output will be the previous output value, regardless of the value on top of the stack.

2 = Force output to logic 0. The value of the output will be logic 0 regardless of the value on top of the stack.

3 = Force output to logic 1. The value of the output will be logic 1 regardless of the value on top of the stack.

S2 to S21 – F1 to F20
Identify:

- The operation to be performed on the input.
- The input value the operation is performed on.
- The input override.

Specification information for S2 through S21 is in the format:

X X X

Operation
XX0 = PUT value on top of stack
XX1 = AND value with value on top of stack
XX2 = OR value with value on top of stack

State of Input Acted On
X0X = use value from stack (0 or 1)
X1X = use logical state of input (0 or 1)
X2X = use logical state of inverted input (0 or 1)
X3X = perform operation when input makes a 0 to 1 transition (1)
X4X = perform operation when input makes a 1 to 0 transition (0)

Override Indicator
0XX = no input override
1XX = force input to logic 1
2XX = force input to logic 0

S22 to S41 –
IN1 to IN10
Block addresses of inputs one through 20.

112.1.2 Outputs

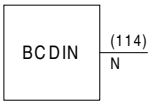
N
Dependent on value on top of the stack and the value of S1.
If S1 = 0, output = value on top of the stack.
If S1 = 1, output = previous value of output, regardless of the value on top of the stack.
If S1 = 2, output = logic 0, regardless of the value on top of the stack.
If S1 = 3, output = logic 1, regardless of the value on top of the stack.

112.2 Applications

Refer to the example in function code 110.

114.BCD Input

The BCD input block reads a boolean input from an IMDSI12 or IMDSI22 module in a binary coded decimal (BCD) format, and converts it to a real output with quality.



Outputs

Blk	Type	Description
N	R	Real value × gain

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus I/O module address of digital I/O module
S2	N	1	I	1 - 4	Zone containing ones digit of output: 1 and 2 = zones reside in group B 3 and 4 = zones reside in group A Input number on I/O module: 1 = 1 to 4 (B) (S4 can be any value) 2 = 5 to 8 (B) (S4 must be =<3) 3 = 1 to 4 (A) (S4 must be =<2) 4 = 5 to 8 (A) (S4 must be set to 1)
S3	N	0	I	0 or 1	Module action on I/O module failure: 0 = trip control module 1 = continue operation
S4	N	1	I	1 - 4	Number of digits in converted value
S5	N	1.000	R	Full	Gain multiplier

114.1Explanation

Each I/O module provides 16 inputs divided into two groups of eight as illustrated in Table 114-1. The two groups are divided into four zones, with each zone containing four inputs.

Table 114-1 BCD to Real Conversion Format

Description	Group A		Group B	
Position	8 7 6 5	4 3 2 1	8 7 6 5	4 3 2 1
Inputs	Zone 4	Zone 3	Zone 2	Zone 1
Internal BCD to Real Conversion				
Position	1000s	100s	10s	1s
Outputs	X	X	X	X

Figure 114-1 shows the binary coding of the four inputs in each zone. The boolean values convert internally to BCD values from zero to nine. A true input for any character in a zone causes the integer number associated with that character to be

output. If the inputs of a zone are 0 1 1 1, the output is 4+2+1=7. BCD values greater than nine are invalid since each zone provides only one digit of the output value.

```
| 8 4 2 1 |
| X X X X |
|  ZONE N |

EXAMPLE: INPUT = 0111
         OUTPUT = 4 + 2 + 1 = 7
                          T01744A
```

Figure 114-1 Binary Coding of Inputs

Specification S2 selects the zone that the ones digit of the BCD value will be read from. This arrangement enables a two digit integer to convert to only one group of inputs from any given I/O module, freeing the other inputs for other configuration uses. Specification S4 sets the number of digits the integer internally generates. Specifications S2 and S4 define the size and location of the integer.

114.1.1 Specifications

S1 – SLVADR
Expander bus I/O module address of the digital input I/O module that carries inputs to be converted. Valid addresses are zero to 63.

S2 – ZONE
Sets the zone containing the ones digit of the output. Zones three and four reside in group A, and zones one and two reside in group B.

S3 – TRIP
Defines the action of the control module on I/O module failure.

0 = trip control module
1 = continue operating

S4 – DNUM
Number of digits in converted value (one to four).

S5 – K
Gain multiplier. This real value is multiplied by the integer number generated by the conversion to provide a real output used for control purposes.

114.1.2 Outputs

N
Binary coded decimal value times gain, and quality.

114.2 Example

If the inputs to the BCD input function block are:
S2 = 1. Ones digit is found in zone 1.
S4 = 3. Output value is three digits (zones 1, 2 and 3).
S5 = 1.0. Gain is 2.0.
Then, the inputs to the I/O module are:

Description	Group A		Group B	
Position	1 0 0 1 (Zone 4)	0 0 1 1 (Zone 3)	1 0 0 0 (Zone 2)	0 1 0 1 (Zone 1)
Inputs	Zone 4	Zone 3	Zone 2	Zone 1
Internal BCD to Real Conversion				
Position	1000s	100s	10s	1s
Outputs	0	3	8	5

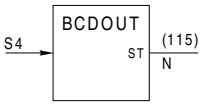
Integer value = 385

Output value = 385 × 2.0 = 770.0

NOTE: Even though some of the inputs in zone four are true (logic 1), the conversion does not use them (S2 equals one and S4 equals three).

115.BCD Output

This block converts a real input to boolean outputs by converting the input to binary coded decimal (BCD) format. The input can be scaled with a gain factor, then truncated to an integer value before the BCD conversion occurs. After conversion, the outputs write to an IMDSO14 or IMDSO15 module.



Outputs

Blk	Type	Description
N	B	I/O module status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus I/O module address of I/O module receiving outputs of function code 115
S2	N	0	I	0 - 120	I/O module definition: **hold + type + group** **Hold:** 0XX = go to default values on loss of control module 1XX = hold I/O module outputs on loss of control module **Type:** X0X = IMDSO15 module X1X = IMDSO14 module **Group:** XX0 = outputs 0 to 7 (group A) XX1 = outputs 8 to 15 (group B)
S3	N	0	I	0 or 1	Module action on I/O module failure: 0 = trip control module 1 = continue operation
S4	N	5	I	Note 1	Block address of real value to be converted to boolean
S5	N	1.000	R	Full	Gain multiplier
S6	N	2	I	2 or 4	Number of BCD digits to process

NOTE:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

115.1Explanation

This block converts real values to boolean outputs that are written to the output groups of an IMDSO14 or IMDSO15 module in BCD format. The real input is multiplied by the gain factor, then truncated to an integer value up to 9999. Specification S6 specifies the size of the integer value. The value can be defined to have either two or four digits. A two digit integer translates to eight boolean outputs. A four digit integer translates to 16 boolean outputs. Table 115-1 shows module output capacity for all I/O modules used with this function code.

Table 115-1 Module Output Capacity

Module	Outputs
IMDSO14	Two groups of eight
IMDSO15	One or two groups of eight

Figure 115-1 shows how each BCD digit provides four boolean outputs for a maximum of 16 boolean outputs.

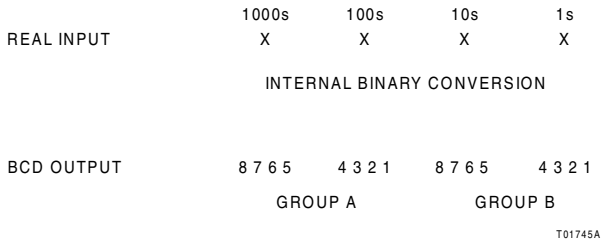


Figure 115-1 Real to BCD Conversion Format

An internal binary conversion is performed on each BCD digit, providing a one output for each power of two represented in the digit, and a zero output for each power of two not represented in the digit. For example, if the ones digit of the input is five, the boolean output representing it is 0101, since 5=0(8)+1(4)+0(2)+1(1). Figure 115-1 illustrates the binary breakdown.

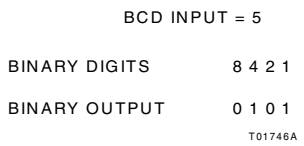


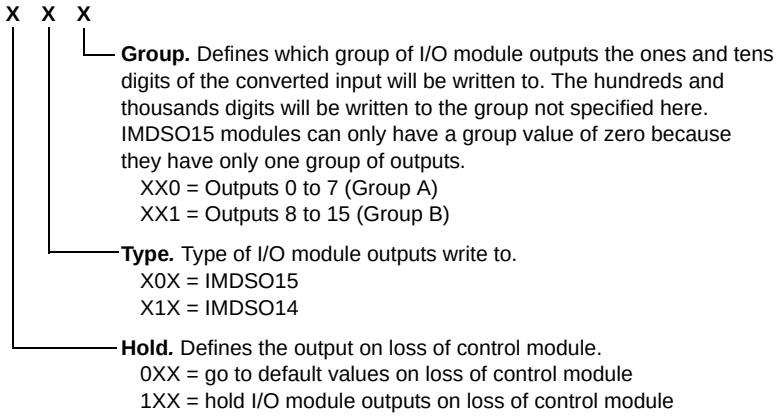
Figure 115-1 Binary Conversion

The output group that the ones and tens digits of the BCD value outputs is selectable with S2. If four digits are selected (IMDSO14 modules only), the hundreds and thousands digits write to the group not selected (S2; ones digit, group definition). Within a group, the least significant digit occupies outputs four through one, and the most significant digit occupies outputs eight through five.

115.1.1 Specifications

S1 – SLVADR
Expander bus I/O module address of the I/O module that boolean outputs write to. Valid addresses are zero to 63.

S2 – SLVDEF
I/O module definition.



S3 – TRIP
Defines action of control module on I/O module failure.

0 = trip control module
1 = continue operation

S4 – BCDINP
Block address of BCD value to be converted to boolean outputs.

S5 – K
Value of the gain multiplier used to scale a real input before truncation and conversion.

S6 – DNUM
Number of processed BCD digits. This number must be set to two or four. Three digit inputs must be specified as four.

115.1.2Output

N
Status of the I/O module.

0 = good
1 = bad

115.2Example

To convert 1293.6 to boolean outputs, set the specifications:

S2 = 1, group B contains ones and tens digits.

S5 = 1.0, scaling value is 1.0.

S6 = 4, input will be four BCD digits, therefore the output is 16 boolean digits.

BCD input = 1293 (after scaling and truncation).

Description	Group A		Group B	
Position	8 7 6 5	4 3 2 1	8 7 6 5	4 3 2 1
Binary value	8 4 2 1	8 4 2 1	8 4 2 1	8 4 2 1
Boolean output	0 0 0 1	0 0 1 0	1 0 0 1	0 0 1 1
BCD input	1	2	9	3

To convert 457.2 to boolean outputs, set the specifications:

S2 = 1, group B contains ones and tens digits.

S5 = 1.0, scaling value is 1.0.

S6 = 4, inputs are four BCD digits, therefore the output is 16 boolean digits. Although the input has only three digits, select four because S6 must be either a two or a four.

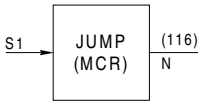
BCD input = 0457 (after scaling and truncation).

Description	Group A[1]		Group B	
Position	8 7 6 5	4 3 2 1	8 7 6 5	4 3 2 1
Binary value	8 4 2 1	8 4 2 1	8 4 2 1	8 4 2 1
Boolean output	0 0 0 0	0 1 0 0	0 1 0 1	0 1 1 1
BCD input	0	4	5	7

NOTE:
1.Digits five through eight in group A output zeros since the input is zero.

116.Jump/Master Control Relay

The jump/master control relay function code can be used as a jump or master control relay (MCR) function. Function code 116 transfers control past a series of blocks. Both functions transfer control from the jump/MCR block to a destination block specified with S2. Blocks between the jump/MCR block and its destination do not execute, but hold their previous values. The destination block number must be greater than the block number of the jump/MCR block where the transfer originated. Transfer across segment boundaries is not permitted.
Specification S3 sets the block to either jump or MCR. The jump function bypasses all blocks between the jump/MCR block and the destination block. The MCR function bypasses all blocks between the jump/MCR block and the destination block, and it de-energizes (zeros) the outputs of all rung blocks (i.e., function codes 110, 111 and 112) with normal outputs (S1 equals zero), that lie between the jump/MCR block and the destination block. Rung blocks without normal outputs (S1 not equal to zero) hold their previous output values.



Outputs

Blk	Type	Description
N	B	<S1> = 0 = jump/MCR activated <S1> = 1 = jump/MCR not activated, all blocks executed

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of control signal: 0 = jump/MCR function activated 1 = normal control
S2	N	0	I	Note 1	Block address of destination block for jump/MCR command
S3	N	0	I	0 or 1	Function selector: 0 = jump 1 = MCR

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

116.1Explanation

116.1.1Specifications

S1 – TRIP
Block address of the control input signal. This determines whether control is normal or if the jump/MCR function is activated.

0 = jump/MCR function activated – S3 selects the function implemented (jump or MCR)

1 = jump/MCR not implemented

S2 – DEST
Block address of destination block that jump/MCR function extends to. Normal block execution resumes with the function in this block.

S3 – FUNC
Defines which function the jump/MCR block performs.

0 = jump – bypasses all blocks between the jump/MCR block and the destination block

1 = MCR – bypasses all blocks between the jump/MCR block and the destination block and de-energizes the outputs of rung blocks with normal outputs between the jump/MCR block and destination block

116.1.2Output

N
Displays <S1>.

0 = jump/MCR implemented
1 = jump/MCR not implemented, all blocks executed

116.2Applications

Figure 116-1 shows the implementation of a jump/MCR block into a Ladder program. When the jump/MCR block is activated, all blocks between the jump block and block 900 are skipped. The jump/MCR can also be used with function code logic.

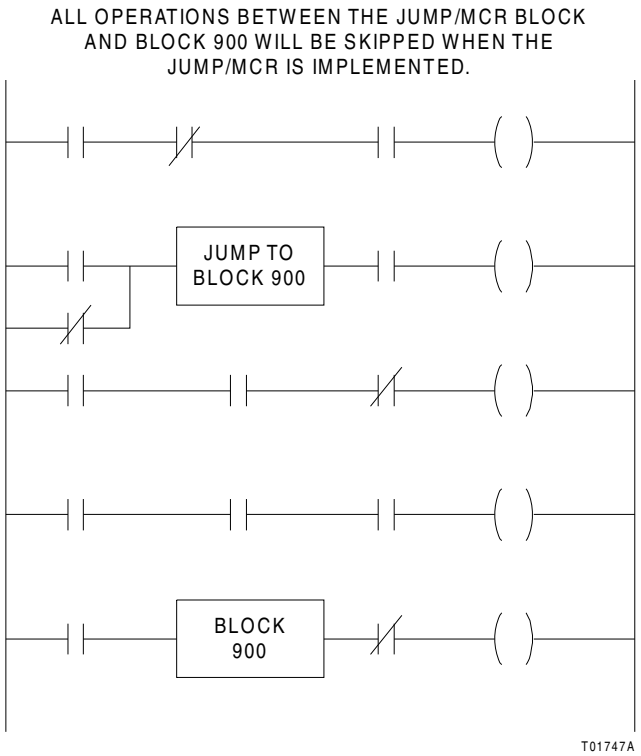
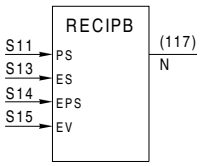


Figure 116-1 Jump/MCR Block Used in Ladder Program

117.Boolean Recipe Table

The boolean recipe table function code uses an externally generated value to select a parameter value from a recipe table. Blocks linked together, each containing up to ten boolean parameter values, make up the recipe table.
The first block in the link list is the group master. The master accepts the select input and is able to search the other blocks in the link list to find the selected parameter. The ten parameters in the master block are zero through nine. The range of parameters associated with the next block in the link list is ten through 19, and so on. The recipe value selected is always output from the master block. The outputs of all other blocks in the series remain unused.
This function code also includes an edit function that allows editing of selected parameters.
The recipe table function codes (117, 118) allow for recipe handling within the digital control system, eliminating the need for a centralized computer to store batch recipes.



Outputs

Blk	Type	Description
N	B	Parameter value selected. If the parameter number selected is invalid, the value of the nearest valid parameter is the output.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0	B	0 or 1	Value of parameter 0
S2	Y	0	B	0 or 1	Value of parameter 1
S3	Y	0	B	0 or 1	Value of parameter 2
S4	Y	0	B	0 or 1	Value of parameter 3
S5	Y	0	B	0 or 1	Value of parameter 4
S6	Y	0	B	0 or 1	Value of parameter 5
S7	Y	0	B	0 or 1	Value of parameter 6
S8	Y	0	B	0 or 1	Value of parameter 7
S9	Y	0	B	0 or 1	Value of parameter 8
S10	Y	0	B	0 or 1	Value of parameter 9
S11	N	5	I	Note 1	Block address of parameter select input
S12	N	0	I	Note 1	Block address of next recipe block (0 = no more recipe blocks)
S13	N	0	I	Note 1	Block address of edit signal: 0 = normal mode 1 = edit mode
S14	N	5	I	Note 1	Block address of edit parameter selector
S15	N	0	I	Note 1	Block address of edit value

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

117.1Explanation

117.1.1Specifications

S1 – P0
Value of parameter zero. If the block defined is not the master block for a particular recipe table, then S1 defines the value of parameter *n+0* where *n* equals the position of the block in the list of blocks making up the recipe table. For example, if the block defined is the first block after the master block, then S1 defines value ten.

S2 to S10 – P1 to P9
Values of parameters one through nine. If the block defined is not the master block for a particular recipe table, then S2 to S10 define the value of parameters *n+1* to *n+9* where *n* equals the position of the block in the list of blocks making up the recipe table. For example, if the block defined is the third block after the master block, then S2 defines value 31.

S11 – SEL
Block address of externally generated parameter selection signal. The number in this block selects the value from the recipe table that is output from the boolean recipe table block. If the input to the parameter selection block is a real number, it is rounded to the nearest integer before being used for selection. If several blocks are linked in series, the parameter selection signal <S11> should be defined only in the first block in the series, since the master block searches the other blocks in the series for the selected parameter value.

S12 – NXT
Block address of next recipe block in group. A value of zero for this specification indicates that there are no more recipe blocks in the group.

S13 – ESIG
Block address of edit signal. When this value is a one, the block is in edit mode. In edit mode, the value of the parameter selected by S14 changes to the value defined in S15. When the value is a zero, the block operates in the normal operating mode.

S14 – EPAR
Block address of the block containing the number of the parameter that changes when the block is in edit mode.

S15 – EVAL
Block address of the value that replaces the current value of the parameter specified in S14 when the block is in edit mode.

117.1.2Output

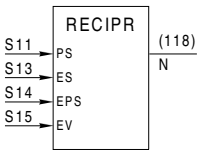
N
Value of the parameter selected for output.
The value of the selected parameter is always output from the first (master) block in a series of boolean recipe table blocks. The outputs of all other blocks in the series are unused.
If the parameter selected is invalid, the value of the valid parameter numerically closest to it will be output. For example: if there are 30 parameters specified (one to 30), and parameter 47 is requested, the value of parameter 30 will be output. Likewise, if parameter -3 is requested, the value of parameter zero will be output.

117.2Applications

Boolean recipe table blocks can be used with real recipe table blocks to control batch processes. Real recipe tables can be used to set ingredient quantity, and boolean recipe tables can define the operating states of the various devices used in the process for each step. The batch processing example found in function code 123 illustrates the use of recipe table blocks.

118.Real Recipe Table

The real recipe table function code uses an externally generated value to select a parameter value from a recipe table. Blocks linked together, each containing up to ten real parameter values, make up the recipe table.
The first block in the link list is the group master. The master accepts the select input and is able to search the other blocks in the link list to find the selected parameter. The ten parameters in the master block are zero through nine. The range of parameters associated with the next block in the link list is ten through 19, and so on. The recipe value selected is always output from the master block. The outputs of all other blocks in the series remain unused.
This function code also includes an edit function that allows editing of a selected parameter.
The recipe table function codes (117, 118) allow for recipe handling within the digital control system. This eliminates the need for a centralized computer to store batch recipes.



Outputs

Blk	Type	Description
N	R	Parameter value selected. If the parameter number selected is invalid, the value of the nearest valid parameter is the output.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0.000	R	Full	Value of parameter 0
S2	Y	0.000	R	Full	Value of parameter 1
S3	Y	0.000	R	Full	Value of parameter 2
S4	Y	0.000	R	Full	Value of parameter 3
S5	Y	0.000	R	Full	Value of parameter 4
S6	Y	0.000	R	Full	Value of parameter 5
S7	Y	0.000	R	Full	Value of parameter 6
S8	Y	0.000	R	Full	Value of parameter 7
S9	Y	0.000	R	Full	Value of parameter 8
S10	Y	0.000	R	Full	Value of parameter 9
S11	N	5	I	Note 1	Block address of parameter select input
S12	N	0	I	Note 1	Block address of next recipe block (next function code 118) 0 = no more recipe blocks
S13	N	0	I	Note 1	Block address of edit signal: 0 = normal mode 1 = edit mode
S14	N	5	I	Note 1	Block address of edit parameter selector
S15	N	5	I	Note 1	Block address of edit value

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

118.1Explanation

118.1.1Specifications

S1 – P0
Value of parameter zero. If the block defined is not the master block for a particular recipe table, then S1 defines the value of parameter *n+0* where *n* equals the position of the block in the list of blocks making up the recipe table. For example, if the block defined is the first block after the master block, then S1 defines value ten.

S2 to S10 – P1 to P9
Values of parameters one through nine. If the block defined is not the master block for a particular recipe table, then S2 to S10 define the value of parameters *n+1* to *n+9* where *n* equals the position of the block in the list of blocks making up the recipe table. For example, if the block defined is the third block after the master block, then S2 defines value 31.

S11 – SEL
Block address of externally generated parameter selection signal. The number in this block selects the value from the recipe table that is output from the real recipe table block. If the input to the parameter selection block is a real number, it is rounded to the nearest integer. If several blocks are linked in series, the parameter selection signal <S11> should be defined only in the first block in the series, since the master block searches the other block in the series for the selected parameter value.

S12 – NXT
Block address of next recipe block in group. A value of zero for this specification indicates that there are no more recipe blocks in the group.

S13 – ESIG
Block address of edit signal. When this value is a one, the block is in edit mode, and the value of the parameter selected by S14 changes to the value defined in S15. When the value is a zero, the block operates in normal mode.

S14 – EPAR
Block address of the block containing the parameter number that changes when the block is in edit mode.

S15 – EVAL
Block address of the value that replaces the current value of the parameter specified in S14 when the block is in edit mode.

118.1.2Output

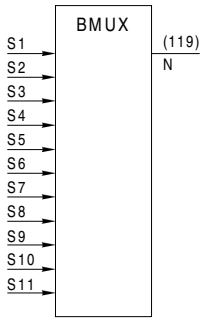
N
Value of the parameter selected for output. The value of the selected parameter is always output from the first (master) block in a series of real recipe table blocks. The outputs of all other blocks in the series are unused.
If the parameter selected is invalid, the value of the valid parameter numerically closest to it will be output. For example: if there are 30 parameters specified (one to 30), and parameter 47 is requested, the value of parameter 30 will be output. Likewise, if parameter -3 is requested, the value of parameter zero will be output.

118.2Applications

Real recipe table blocks can be used to handle both fixed batch sequences and variable batch sequences. In a fixed batch sequence, the order in which the steps are executed remains fixed from recipe to recipe, but the amounts of ingredients and the conditions under which they are added may vary. In a variable batch sequence, the order of step execution changes from recipe to recipe, as well as the amounts of ingredients and the conditions in which they are added. Refer to the applications section of the function code 123 description for an example of a real recipe table used in a batch process.

119.Boolean Signal Multiplexer

The boolean signal multiplexer function code selects a boolean signal from a group of inputs, and provides this signal as the block output. Any number of boolean signal multiplexer blocks can be joined to multiplex any number of signals. Each block contains ten boolean inputs.
The first block in the link list is the group master. The master accepts a select input and searches the other blocks in the link list to find the selected input. The ten values in the master block are zero through nine. The range of parameters associated with the next block in the link list is ten through 19, and so on. If several blocks are in series, the output always comes from the first block in the series. The outputs of the rest of the blocks in the series are unused.



Outputs

Blk	Type	Description
N	B	Input value selected. If the input number selected is invalid, the value of the nearest valid parameter is the output.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of input 0
S2	N	0	I	Note 1	Block address of input 1
S3	N	0	I	Note 1	Block address of input 2
S4	N	0	I	Note 1	Block address of input 3
S5	N	0	I	Note 1	Block address of input 4
S6	N	0	I	Note 1	Block address of input 5
S7	N	0	I	Note 1	Block address of input 6
S8	N	0	I	Note 1	Block address of input 7
S9	N	0	I	Note 1	Block address of input 8
S10	N	0	I	Note 1	Block address of input 9
S11	N	5	I	Note 1	Block address of input select signal
S12	N	0	I	Note 1	Block address of next block (next function code 119) 0 = no more multiplexer blocks

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

119.1Explanation

119.1.1Specifications

S1 – IN0
Block address of input zero. If the defined input is not in the master block for a particular input group, then S1 is the block address of input $n \times 10$ where n equals the position of the block in the list of blocks making up the link list. For example, if the defined block is the first block after the master block, then S1 defines input ten.
Input zero is reserved for executed stop (E-STOP) in batch operations. The normal starting input is input one which is <S2>.

S2 to S10 – IN1 to IN9
Block addresses of inputs one through nine. If the defined input is not the master block for a particular input group, then S2 to S10 define the block addresses of inputs $n \times 10 + 1$ to $n \times 10 + 9$ where n equals the position of the block in the list of blocks making up the input group. For example, if the defined input is in the third block after the master block, then S2 is the block address of input 31.

S11 – SEL
Block address of the externally generated input selection signal. The number in this block selects the input from the input group that is the block output. If the input to this block is a real number, it is rounded to the nearest integer. If more than one block is in series, the input selection signal is the master block only, since it searches all other blocks in the series for the selected parameter.

S12 – NXT
Block address of next input block in group. A value of zero for this specification indicates that there are no more input blocks in the group.

119.1.2Output

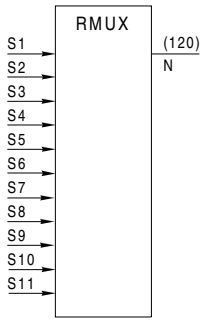
N
Selected input. The selected input is output from the master block. When several boolean signal multiplexer blocks are in series, the value selected for the output is the output from the master block. The outputs of the rest of the blocks in the series are unused.
If an invalid value is selected, the value of the parameter numerically closest to it is the output. For example, if there are 30 specified inputs, and input 47 is requested, then the value of input 30 will be output. Likewise, if the selected value is -3, the value of input zero will be output.

119.2Applications

Batch control uses signal multiplexers to select one input from a group, based on an externally generated signal. Refer to the applications section of function code 123 for an example of a boolean signal multiplexer used in a batch process.

120.Real Signal Multiplexer

The real signal multiplexer function code selects a real signal from a group of inputs, and provides this signal as the block output. Any number of real signal multiplexer blocks can be joined to multiplex any number of signals. Each block contains ten real inputs.
The first block in the link list is the group master. The master accepts a select input and searches the other blocks in the link list to find the selected input. The ten values in the master block are zero through nine. The range of parameters associated with the next block in the link list is ten through 19, and so on. If several blocks are in series, the output always comes from the first block in the series. The outputs of the rest of the blocks in the series are unused.



Outputs

Blk	Type	Description
N	R	Input value selected. If the input number selected is invalid, the value of the nearest valid parameter is the output.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input 0
S2	N	5	I	Note 1	Block address of input 1
S3	N	5	I	Note 1	Block address of input 2
S4	N	5	I	Note 1	Block address of input 3
S5	N	5	I	Note 1	Block address of input 4
S6	N	5	I	Note 1	Block address of input 5
S7	N	5	I	Note 1	Block address of input 6
S8	N	5	I	Note 1	Block address of input 7
S9	N	5	I	Note 1	Block address of input 8
S10	N	5	I	Note 1	Block address of input 9
S11	N	5	I	Note 1	Block address of input select signal
S12	N	0	I	Note 1	Block address of next block (next function code 120) 0 = no more multiplexer blocks

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

120.1Explanation

120.1.1Specifications

S1 – IN0
Block address of input zero. If the defined input is not in the master block for a particular input group, then S1 is the block address of input $n \times 10$ where n equals the position of the block in the list of blocks making up the link list. For example, if the defined block is the first block after the master block, then S1 defines input ten.

S2 to S10 – IN1 to IN9
Block addresses of inputs one through nine. If the defined input is not the master block for a particular input group, then S2 to S10 define the block addresses of inputs $n \times 10 + 1$ to $n \times 10 + 9$ where n equals the position of the block in the list of blocks making up the input group. For example, if the defined input is in the third block after the master block, then S2 is the block address of input 31.

S11 – SEL
Block address of the externally generated input selection signal. The number in this block selects the input from the input group that is the block output. If the input to this block is a real number, it is rounded to the nearest integer. If more than one block is in series, the input selection signal is the master block only, since it searches all other blocks in the series for the selected parameter.

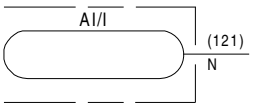
S12 – NXT
Block address of next input block in group. A value of zero for this specification indicates that there are no more input blocks in the group.

120.1.2 Output

N
Selected input. The selected input is output from the master block. When several real signal multiplexer blocks are in series, the value selected for the output is the output from the master block. The outputs of the rest of the blocks in the series are unused.
If an invalid value is selected, the value of the parameter numerically closest to it is the output. For example, if there are 30 specified inputs, and input 47 is requested, then the value of input 30 will be output. Likewise, if the selected value is -3, the value of input zero will be output.

121.Analog Input/Cnet

The analog input/Cnet function code acquires an analog input from another module in a different node via Cnet. If the source node is a process control unit, the input must come from an analog exception report block (function code 30), an analog in/channel (function code 222), or an analog out/channel (function code 223). If the remote node is a network interface unit, the input must come from an analog report index. Updates are on an exception report basis. The executive block for the controller module and the segment control block for the harmony controllers modules specify the exception report intervals.
To insure that the signal is successfully acquired from Cnet, the analog signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The quality of the point cannot be used as an input to any other type of block. However, the output of the test quality block, representing the quality, can be used as an input to other processing blocks. Refer to Appendix H, for a definition of point quality.



Outputs

Blk	Type	Description
N	R	Analog output value, quality and alarm state

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 31	Source module address
S2	N	0	I	Note 1	Source block address
S3	N	0	I	0 - 250	Source node address
S4	N	0	I	0 - 250	Source loop address
S5	N	0	I	0-2	Input block type. Defines the type of remote block address that is being referenced: 0=function code 30 1=function code 222 2=function code 223
S6	N	0.000	R	Full	Spare
S7	N	0	I	Note 1	Spare

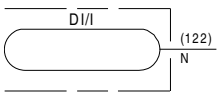
NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

122.Digital Input/Cnet

The digital input/Cnet function code acquires a digital input from another module in a different node via Cnet. If the source node is a process control unit, the input must come from a digital exception report block (function code 45), a digital in/channel (function code 224), or a digital out/channel (function code 225). If the remote node is a network interface unit, the input must come from an analog report index. Updates are on an exception report basis. Segment control blocks in Harmony controllers specify exception report intervals.
To insure that the signal is successfully acquired from the Cnet, the digital signal generates a point quality flag. To test the quality of the signal, include a function code 31 in the configuration. The quality of the point cannot be used as an input to any other type of block. However, the output of the test quality block, representing the quality, can be used as an input to other analog processing blocks. Refer to Appendix H, for a definition of point quality.

NOTE: If a module utilizes an imported digital value from the loop in several instances in its configuration, the function blocks that utilize that digital value must be connected to only one digital input/Cnet block. Exception reports from the same (loop, node, module and block) address cannot be imported to more than one destination within a single module configuration.



Outputs

Blk	Type	Description
N	B	Digital output value, quality and alarm state

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 31	Source module address
S2	N	0	I	Note 1	Source block address
S3	N	0	I	0 - 250	Source node address
S4	N	0	I	0 - 250	Source loop address
S5	N	0	I	0-2	Input block type. Defines the type of remote block address that is being referenced: 0=function code 45 1=function code 224 2=function code 225
S6	N	0.000	R	Full	Spare
S7	N	0	I	Note 1	Spare

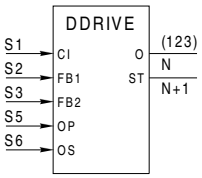
NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

123.Device Driver

Function code 123 interfaces the control system to a field device. It provides control and accepts feedback from its assigned control device. It normally receives a control input from the sequencing logic and sets the control output equal to the control input when the block is in auto. The control output then goes to the associated device.
The device driver also accepts up to two feedback inputs from the field that define the actual status of the device. The control output status represents the status of the device, determined from the feedback inputs as good, bad, or waiting. A waiting output is generated when the feedback waiting time has not elapsed after a state change. The device driver generates an exception report indicating output and alarm when:

- The device driver output changes ***or***
- The device driver generates an alarm ***or***
- The T_{max} for exception reporting is exceeded.

Device drivers can be used in batch and continuous control systems to control and observe a device. Any boolean signal can drive a device driver. It also can be operated from a console, or an output can be unconditionally set with interlock logic.



Outputs

Blk	Type	Description
N	B	Control output
N+1	R	Control output status: 0.0 = good 1.0 = bad 2.0 = waiting

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of control input
S2	N	0	I	Note 1	Block address of feedback input 1
S3	N	0	I	Note 1	Block address of feedback input 2
S4	Y	0	I	0 - 101	Control output status override signal
S5	N	1	I	Note 1	Block address of output override permissive
S6	N	0	I	Note 1	Block address of override output
S7	Y	0	I	0 - 11	Feedback status mask output = 0
S8	Y	0	I	0 - 11	Feedback status mask output = 1
S9	Y	0.000	R	Full	Feedback waiting time (secs)
S10	N	0	I	0 - 255	Device driver display type

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

123.1Explanation

The device driver, used with the other batch processing blocks, implements batch control. Normally, a sequence generator drives a device driver block by setting S1 to the output block address of the sequence generator block. The device driver may be controlled manually (from the console) or unconditionally set. The states of the output override and permissive

specifications indicate whether manual control from the console or remote control using unconditional logic is permitted. Table 123-1 identifies the permitted control modes for various states of the override specifications S4 through S6.

Table 123-1 Override Permissive/State Specifications

Control Output Status Override (S4)	Value of Output Override Permissive <S5>	Permitted Control Mode	Control Output	Control Output Status
Console Override Control (S6) = 0				
XX0	0	Auto	<S1>	Normal
XX0	1	Auto Manual	<S1> <console>	Normal Normal
XX1	0	Auto	<S1>	Forced good or waiting
XX1	1	Auto Manual	<S1> <console>	Forced good or waiting Forced good or waiting
Remote Override Control (S6) ≠ 0				
XX0	0	Auto	<S1>	Normal
XX0	1	Remote	<S6>	Normal
XX1	0	Auto	<S1>	Forced good or waiting
XX1	1	Remote	<S6>	Forced good or waiting

When the output of the block specified in S5 is one and S6 is zero, manual override from the console is permitted. Specification S6 unconditionally sets the output of the device driver when S6 is configured to any block address greater than zero. The boolean output of the block referenced by S6 sets the output of the device driver. This feature precludes manual override of the device from the operator console. When S5 equals zero, only automatic control is permitted.
The control output status is dependent on the feedback inputs, feedback waiting time, and the feedback status masks. Feedback status masks provide signals defining the normal states of the feedback inputs corresponding to the control outputs of zero (S7) and one (S8). If the values of the feedback signals do not match those of the feedback status masks for a given control output, an exception report generates and goes to the console. The exception report contains the value of the control output (zero or one) and an alarm indicator. When an operator command or logic overrides the control output status, the control output status goes to zero. This allows sequence logic to proceed normally. However, alarms still generate, exception report, and display on the console.
For example, when the device driver controlling an on/off valve generates an alarm, the sequence monitor block executes a fault step (e.g., shutdown). If the operator determines that the valve is working correctly, but the position feedback is not correct for some reason, the fault status can be overridden from the console by tuning the ones digit of S4 to a value of one. This overrides the control status output of the device driver and the sequence monitor block operates as if the device is acting correctly. The block address value specified by S6 determines whether control is set unconditionally or whether the operator has control. An external alarm message will still be transmitted to the console to indicate to the operator that the device is malfunctioning, but the sequence can be restarted or used repeatedly. The console can be used to configure both dynamic graphic displays and faceplate displays that allow access to all of the analog controllers, pushbutton stations and device drivers in the system.

123.1.1 Specifications

S1 – C1
(Block address of control input) A sequence generator block outputs up to eight boolean values for each step in a process. This output is called a mask. It represents the states of eight devices for each step of the process. Each of those eight values can be the control input for a different device driver block. The values are output from the sequence generator block in blocks N through N+7. The device driver block accesses the values if S1 is the block number of the sequence generator output defined for control of the associated device.
A block address of zero (default) for S1 configures the function code for Batch language only.

S2 – FEED1
(Block address of feedback input 1) Signals from the field that define the actual state of the device. The values of the feedback inputs are compared with feedback status masks to determine the control output status. If the feedback inputs do not match the feedback status masks for a given control output, an alarm is generated and the control output status is set to 1.0 (bad).

S3 – FEED2
(Block address of feedback input 2) Signals from the field that define the actual state of the device. The values of the feedback inputs are compared with feedback status masks to determine the control output status. If the feedback inputs do not match the feedback status masks for a given control output, an alarm is generated and the control output status is set to 1.0 (bad).

S4 – COSOV
(Control output status override) Refer to Table 123-1. When the hundreds digit equals one, the control output status is good as soon as the feedbacks match the feedback status mask.

X X X

Ones digit
0 = feedback determines control output status
1 = control output status is 0.0 (good) or 2.0 (waiting)

Hundreds digit
0 = normal operation
1 = early good status enable

NOTE: Early recognition of feedback cancels the feedback waiting time (S16) once feedback conditions have been met. This can result in bad status and alarm prior to feedback waiting time time-out.

S5 – OOPER
(Block address of output override permissive) When the output override permissive input value is a logic 1, the device driver control output can be controlled by a console (manual mode) or by external logic (remote mode) referenced by S6. When the output override permissive input is a logic 0, the device driver control output is controlled by the control input (automatic mode) referenced by S1. Refer to Tables 123-1 and 123-2 for more information.

S6 – OVOLIT
(Block address of override input) Determines whether the override state selected is manual or remote.

S6 = 0 and <S5> = 1, permitted control mode is auto or manual.

S6 ≠ 0 and <S5> = 1, permitted control mode is remote.

S7 – FDMSK0
(Feedback status mask output for control output of zero) The value found in this block defines the normal state of the feedback inputs corresponding to a control output of zero. If control output is a zero and the feedback inputs to the device driver do not agree with S7, an exception report with alarm is generated. The ones digit is feedback input one and the tens digit is feedback input two.

X X

feedback input 1 (S2)

feedback input 2 (S3)

S8 – FDMSK1
(Feedback status mask output for control output of 1) Defines the normal state of the feedback inputs corresponding to a control output of one. If control output is a one and the feedback inputs to the device driver do not agree with S8, an exception report with alarm is generated. The ones digit is feedback input one and the tens digit is feedback input two.

X X

feedback input 1 (S2)

feedback input 2 (S3)

S9 – FDWAIT
(Feedback waiting time) Defines the time in seconds that the device driver waits before comparing the feedback inputs with the feedback status masks after a change of state. For example, if the device driver controls a valve, feedback waiting time is the valve stroke time. This insures that measurements taken when the device is changing position or starting up are not used for control or indication.

S10 – DISPLAY
(Device driver display type) The console provides the capacity to create dynamic graphic and faceplate displays. This specification defines the display type used to represent this particular device. There is a special faceplate display created especially for the device driver block called the device driver mimic. Refer to the console instruction for display types.

0 = output and feedback indicators

123.1.2Outputs

N
(Control output) Drives the device associated with the device driver block. The control output is normally the control (auto) input received from a sequence generator block. Control outputs can be overridden by inputs from the console (manual) or unconditionally set to a value with interlock logic (remote). Specifications S4 through S6 define the override state, as shown in Table 123-1.

N+1
(Control output status) Output from the device driver to a device monitor block to inform the control system of the current state of the driven device.

0.0 = good
1.0 = bad
2.0 = waiting

A good output means that the feedback waiting time has elapsed and the inputs from the field agree with the feedback status masks defined in S7 and S8 as the normal states for a particular control output.
A bad output means that the feedback waiting time has elapsed and the inputs from the field do not agree with the feedback status masks for a particular control output.
A waiting output means that the feedback waiting time has not elapsed, and no comparisons between inputs from the field and feedback status masks has been made yet.
Control output status can be overridden and forced good when S4 equals XX1. If a device driver mimic display is configured in the console, it indicates an override state with *OVR* in the center of the bottom line of the display.

123.2Applications

Device driver blocks can be used in batch and continuous processes to control a piece of equipment from the console or to receive feedback on the state of the equipment. The following example shows a device driver used in a typical batch application. The example illustrates how all the batch function blocks interact in a batch process. These blocks can also be used for sequential control.
To be effective, a distributed batch control system should be partitioned for group control of interdependent unit operations. All interdependent unit operations should be controlled from a single module. Each independent unit operation should be located in a separate controller. This method maximizes system reliability and insures control system integrity.
Figure 123-1 shows a simple process requiring batch control capabilities. Reactants drawn from several storage tanks go into two batch chemical reactors. After the batch reaction is complete, the products transfer from the reactors to an intermediate storage tank, process through a liquid/liquid extractor, and then purify in a distillation column. This process is characterized by several interconnected unit operations, all of which must function properly in the proper sequence for the process to operate.

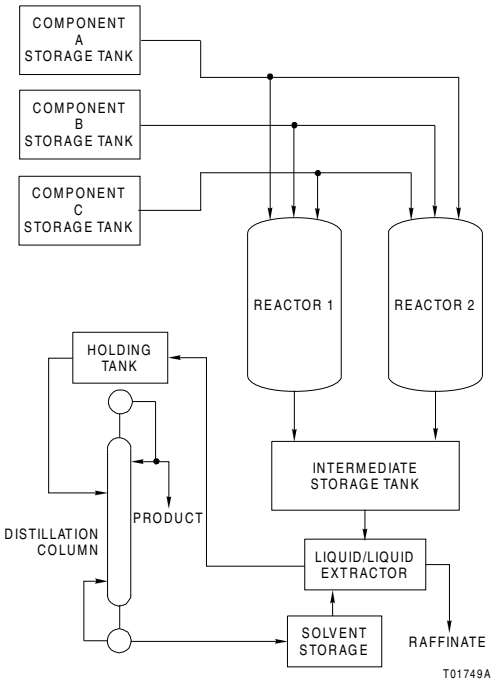


Figure 123-1 Sample Batch Application

Figure 123-1 shows typical partitioning of the example batch control system. Separate controllers are provided for each reactor, for reactant distribution, for liquid/liquid extraction, and for distillation. With the partitioning shown, if the control system for one of the batch reactors stops operation, the other reactor operates and the rest of the process continues to operate until immediate storage depletes, allowing time for the replacement of the malfunctioning component. If using a single controller for the entire process, the process would shut down abruptly, perhaps unsafely. It would stay shut down until the malfunction could be diagnosed and corrected.

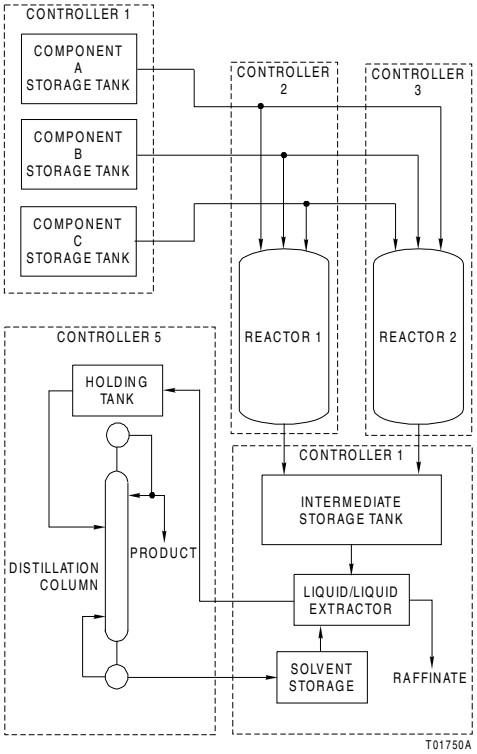


Figure 123-1 Partitioning of Sample Control System

Once the control system is effectively partitioned, each batch operation can be defined by identifying the devices used, the steps required for the batch sequence, the recipe parameters used, and the emergency actions required.
The basic approach to batch control is to create sequence control logic that generates a unique output state pattern for each step in a sequence. This logic must automatically and continuously verify that all of the devices are operating as directed. In this way, the process divides into a series of easy to construct auxiliary logics linked together through the sequence control logic.
To develop batch control logic:

1. The devices associated with control of the batch reactor must be identified. These devices include analog measurement sensors, modulating control valves, and discrete devices such as motors and on/off valves. Discrete device status feedback, such as motor starter holding contacts and valve limit switches, should also be

considered. The analog and discrete devices associated with the example batch reactor are shown in Figure 123-1.

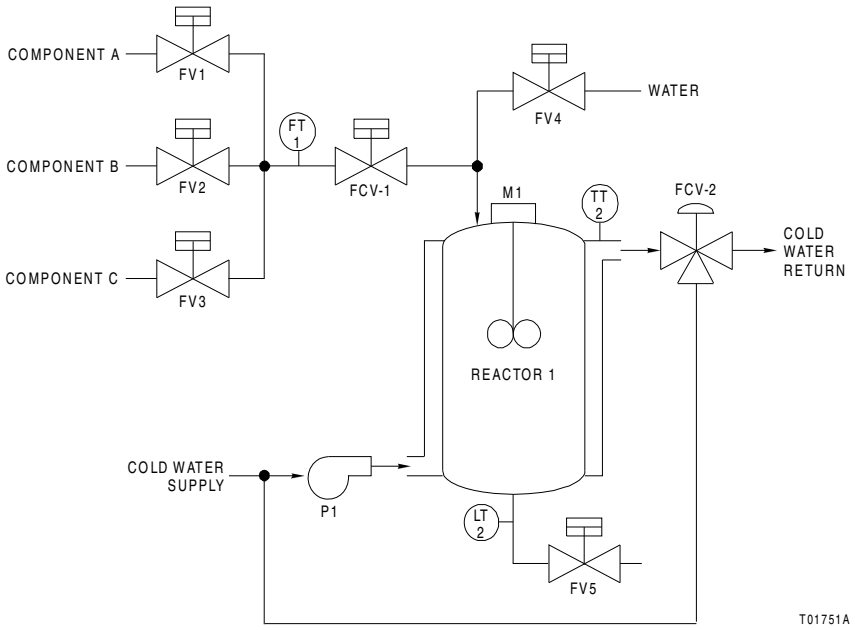


Figure 123-1 Control Devices Associated with Example Batch Reactor

2. A written description of the batch sequence must be prepared. This description should divide the sequence into a series of steps and fully describe both the actions taken during each step and the feedback required before the sequence can proceed. Use the following steps for the example batch sequence:

Step 1
Clean reactor. Open FV4, start M1 and P1, and set TIC-2 set point equal to recipe parameter A. Go to Step 2 when reactor is 80 percent full.

Step 2
Empty reactor. Open FV5. When reactor is less than five percent full, start ten minute timer. When timer times out, go to Step 3.

Step 3
Feed component A. Close FV5 and open FV1. Set FIC-1 set point equal to recipe parameter B. Integrate flow until the total amount added is greater than the amount indicated by recipe parameter C. Go to Step 4.

Step 4
Feed component B. Close FV1 and open FV2. Set TIC-2 set point equal to recipe parameter D. Integrate flow until total amount added is greater than the amount indicated by recipe parameter E. Go to Step 5.

Step 5
Feed component C. Close FV2 and open FV3. Integrate flow until total amount added is greater than the amount indicated by recipe parameter F. Go to Step 6.

Step 6
Cook reactants. Close FV3. Ramp reactor temperature up to recipe parameter G at a constant rate. When TT-2 is greater than recipe parameter G, go to Step 7.

Step 7
Empty reactor. Open FV5. When reactor level is below five percent, start ten minute timer. When timer times out, go to Step 8.

Step 8
Cue operator to review and print out batch report. After operator acknowledges that he has done this, go to Step 1.

NOTE: The process is not a continuous batch sequence. The process ends at Step 8, and requires operator action to restart the batch sequence.

3. The recipe parameters must be defined. These parameters should be selected to allow variations in time, temperature and relative amounts of reactants from recipe to recipe. The example batch reactor has seven recipe parameters:

   1 - starting reactor temperature

   2 - rate of addition of components to reactor (set point of FIC-1)

   3 - amount of component A to add to reactor

   4 - reactor temperature for the addition of components B and C

   5 - amount of component B to add to reactor

   6 - amount of component C to add to reactor

   7 - reaction temperature

4. The actions required upon device failure must be defined. One emergency action may be defined for a device failure in any step, or separate emergency actions may be defined for device failures in each step. For this example, a general fail safe condition exists for response to any device failure.

Step 0

E-STOP (executed stop). P1 and M1 are left running, and all valves are set to fail safe position. This step executes when the operator manually initiates an E-STOP, or the controller detects a device failure.

Figures 123-1 through 123-1 show the configuration used to achieve the control. Figure 123-1 shows the logic that controls the discrete devices. Figures 123-1 through 123-1 show the auxiliary logic required to implement each step. Cross referencing between the sequence definition and these figures provides a good indication of the flexibility and options available for implementing batch control.

In Figure 123-1, the device drivers control and monitor the discrete control devices used in the example. The example batch

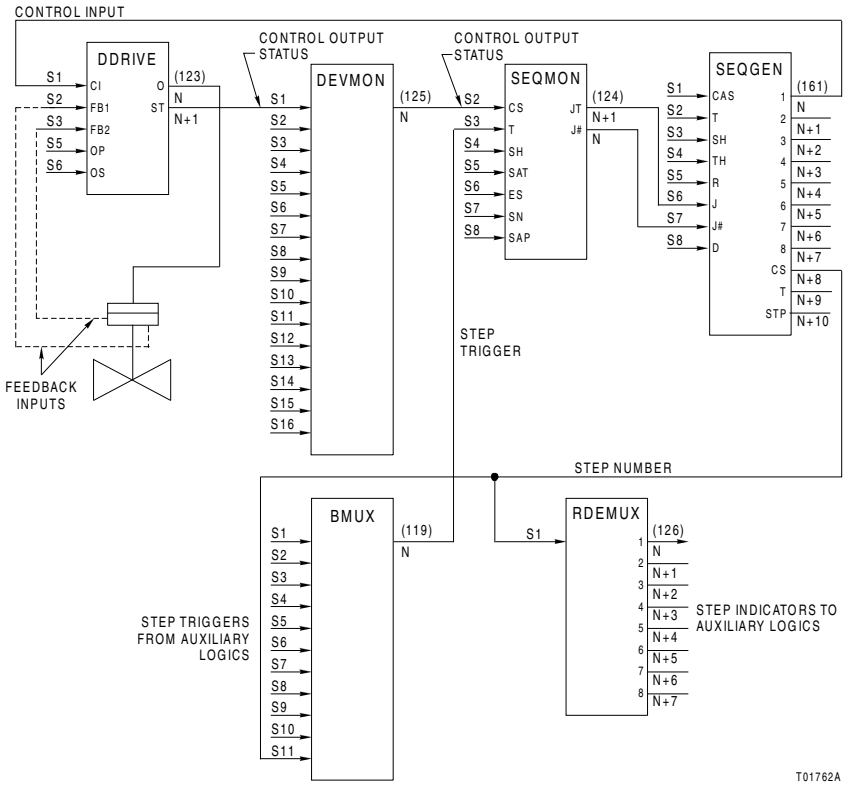


Figure 123-1 Logic Diagram for Example Batch Reactor

reactor has five on/off valves, a pump and an agitator. The on/off valves have two limit switches and require a maximum of five seconds for full travel. The states of the limit switches are used as the feedback indicators, and feedback waiting time is five seconds to insure that the valve has finished changing positions before the feedback values confirm valve operation.

The pump and agitator motor have only one feedback signal, and require a maximum of two seconds to confirm operation. The feedback waiting time for those devices will be defined as two seconds.

NOTE: Output N is Step 0 indicator. Input S1 to BMUX is Step 0 trigger.

The device driver has two outputs, a boolean output that drives the field device, and a real output that indicates if the field device is in the correct state based on the input and feedback values. The real output goes to a device monitor block that collects the outputs from all the device driver blocks and outputs a combined status for the process.
All device driver blocks connect to a device monitor block. Each device monitor block monitors the status of up to eight device driver blocks. It is, in effect, a specialized logic OR block. If any input to the device monitor block is bad or waiting, then the output is bad or waiting.
The output of the device monitor block is the control output status for the whole process. That value (good, bad or waiting) goes to a sequence monitor block. The sequence monitor block controls step execution. It determines which steps are to be executed and when. It selects the next step to be executed based on the value from the device monitor and a step trigger value. The sequence monitor block can be configured to act on either or both inputs. Each step is defined by three specifications: the next normal step, the next fault step, and a stop type. If the input from the device monitor block is bad (any field device not in proper state), the sequence monitor block will select a fault step (Step 0 in the example) to be executed next. If the input is good, the step chosen depends on the three specifications and the step selection configuration.
The number of the next step to be executed and a boolean step trigger are sent from the sequence monitor block to a sequence generator block. Each sequence generator block includes an array of up to eight outputs that can be used to control device driver blocks. The sequence monitor block (in its most commonly used form), with step type 00, initiates execution of the next step when the step trigger is on and the status of all devices for the batch unit is good.
The sequence monitor and sequence generator blocks are also used to control and monitor the auxiliary logics that must be executed with each step. In the sample configuration, when the sequence generator block initiates a step, the output mask for that step goes to the device driver blocks, and the step number goes to a real signal demultiplexer block. The real signal demultiplexer block converts the real step number value to a series of boolean outputs which are used to select the auxiliary logic corresponding to the current step. If the real input is a one, then outputs zero and two through seven will be zeros, and output one will be a one. Thus, the auxiliary logic corresponding to Step 1 will be initiated.

Figure 123-1 illustrates the auxiliary logic for Step 1. Step 1 includes opening FV4, starting M1 and P1, setting TIC-2 set point equal to recipe parameter A, and going to Step 2 when the reactor is 80 percent full. The output mask controls FV4,

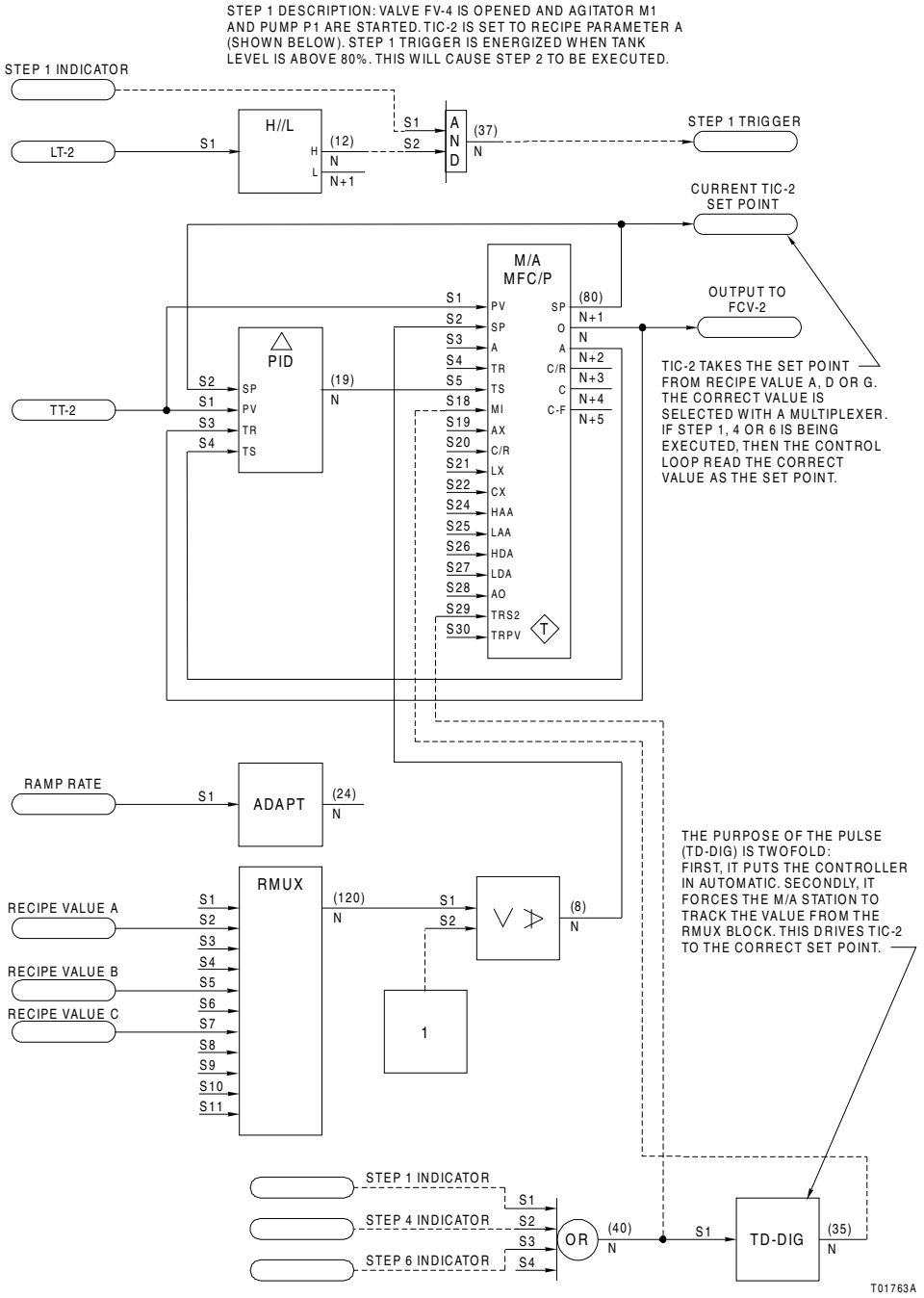


Figure 123-1 Auxiliary Logic for Step 1 in Example

M1 and P1; the auxiliary logic controls the set point and reactor level measurements.
The step number (one) goes to a real signal multiplexer, which outputs input value number one. Input value number one is a recipe value A, the starting reactor temperature. That value goes through a rate limiter and the rate the temperature ramps up (defined using an adapt block). The output rate then goes to a manual/auto station as the set point, and PID control is performed based on that set point and the current temperature read from a temperature indicator (TT-2).
The digital timer block shown at the bottom of Figure 123-1 serves two purposes. When a step requiring a change in reactor temperature executes, the time insures that the controller is in automatic, and it forces the manual/auto station to track the temperature value from the real signal multiplexer block. This drives the current set point to the correct value for each step. The digital timer is configured as a pulse to put the controller into auto and then to go off. This removes a lock in auto condition and allows putting the controller into manual from keyboard.
The logic at the top of Figure 123-1 illustrates the step trigger which signals the completion of Step 1. A tank level value goes to a high/low compare block. When the tank level exceeds the high limit (in this case 80 percent) the high alarm output goes to one. This value is ANDed with the Step 1 indicator from the real signal demultiplexer to create a step trigger. When

the high alarm output goes to one, the output of the AND block goes to one also. That zero to one transition of the output of the AND block is the step trigger shown in Figure 123-1 signaling the completion of the auxiliary logic for Step 1.
The values of all the completion step triggers go to a boolean signal multiplexer block that selects one of ten input signals and provides it as the output. The signal is selected with an input select signal. Figure 123-1 illustrates that the input select signal is the step number output from the sequence generator block (in this example one). Therefore, the value output from the boolean signal multiplexer block is the value of the completion flag for Step 1. The output of the boolean signal multiplexer block is the step trigger input for the sequence monitor block. When that value makes a zero to one transition, the sequence monitor selects the next step in the process. This arrangement insures that the current step runs to completion before the next step is initiated.

NOTE: All step logics have the same structure. They are triggered by a step indicator and when the step logic is complete a step trigger generates. All device checking (to insure correct operation) is done automatically via the device drivers through the device monitors by the sequence monitor.

Figures 123-1 through 123-1 illustrate the auxiliary logics associated with Steps 2 through 8 in the example process.

NOTE: Although not specifically shown in the drawings, all integrators have a default value of block address five for S3. This means the integrators track zero when not active. When active, they start from an initial value of zero.

Recipe values A through G are selected from seven real recipe table blocks as shown in Figure 123-1. A remote manual set constant block is used by the operator to select one of ten real values for each recipe value, enabling the operator to fine tune a recipe and create several products with one set of equipment.

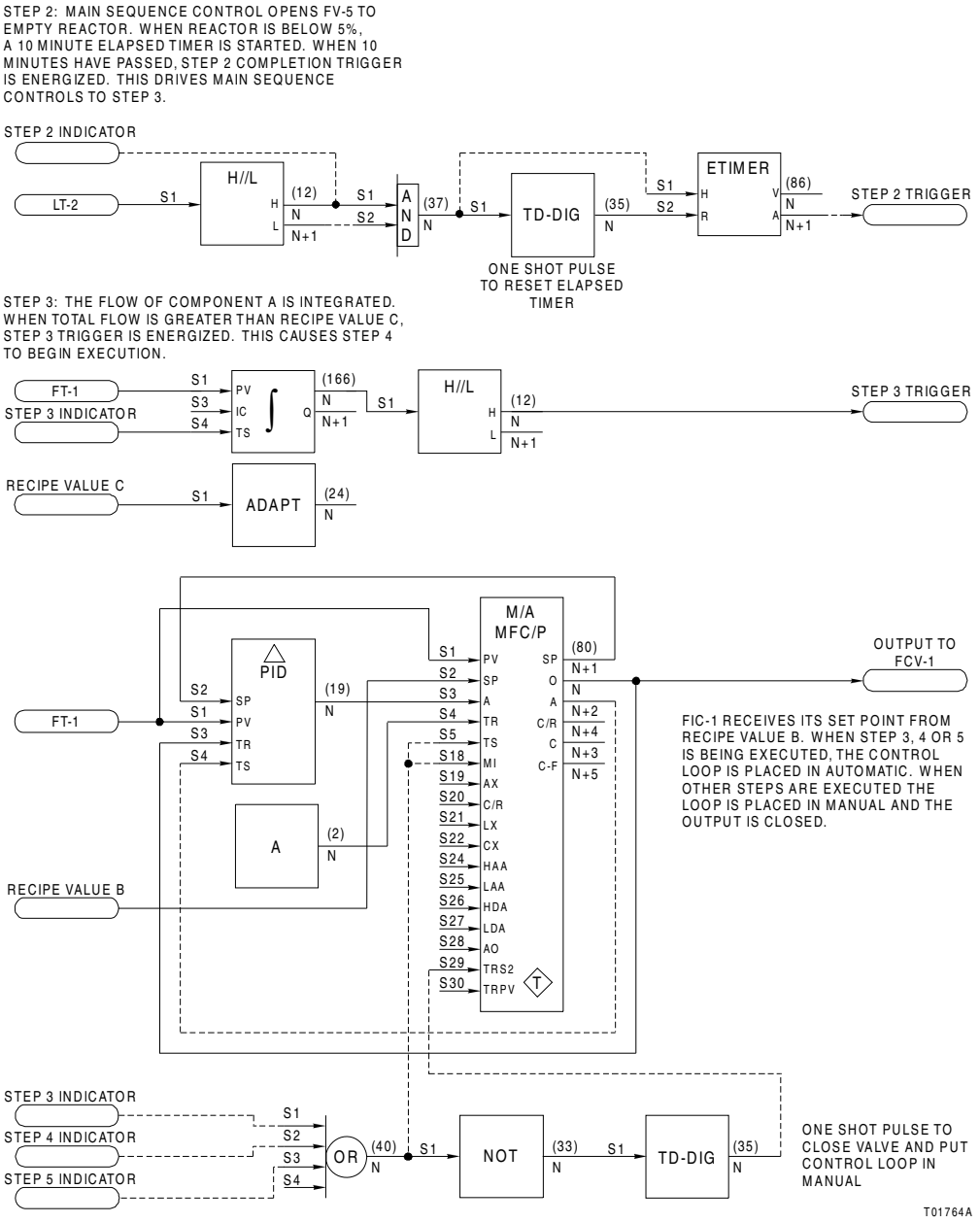


Figure 123-1 Auxiliary Logic for Steps 2 and 3

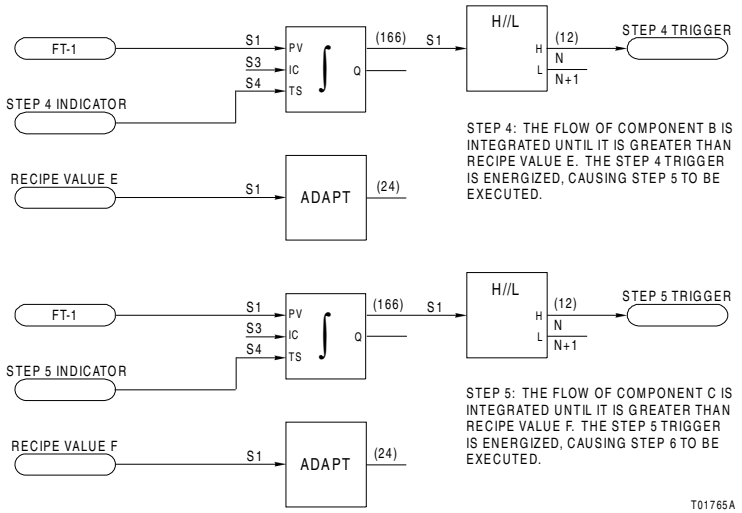


Figure 123-1 Auxiliary Logic for Steps 4 and 5

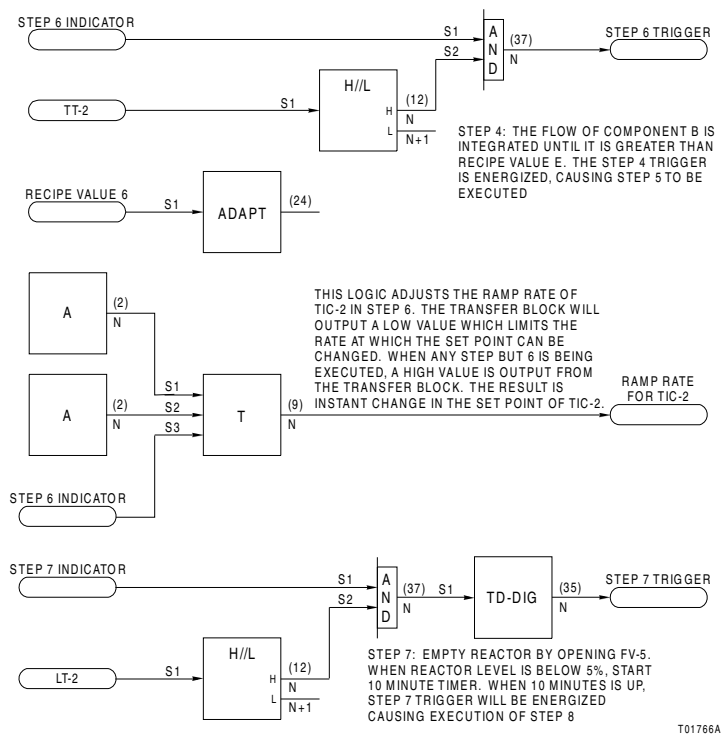


Figure 123-1 Auxiliary Logic for Steps 6 and 7

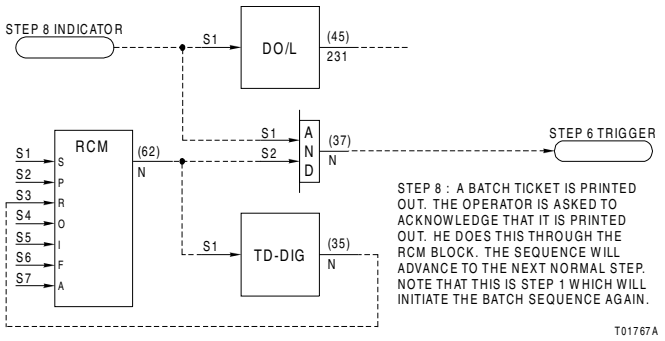


Figure 123-1 Auxiliary Logic for Step 8

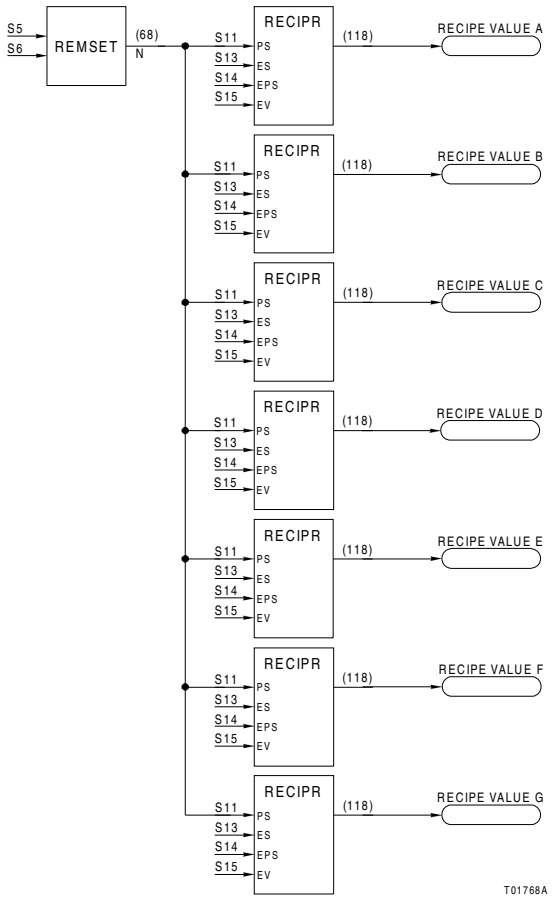


Figure 123-1 Recipe Tables for Example Process

123.3 Distributed Recipe Handling

Depending on the memory requirements for the actual batch control configuration, recipes can be stored in the same module as the control configuration, or in another module located either on the same Controlway/module bus or in a different process control unit on the communications highway.
This distributed recipe handling capability generally eliminates the requirement for a centralized computer to store batch recipes, and its associated costs, project complications, and operational unreliabilities. Further, recipe storage can be expanded incrementally as the need arises by simply adding another module to the control system.

123.3.1 Storing Recipe Data

A specialized function block has been provided in the module to simplify recipe handling. The real recipe table function block (RECIPR, function code 118) stores up to ten values of one recipe parameter. The output of the RECIPR block corresponds to the parameter value for the recipe number input to the RECIPR block. The RECIPR block can be used to change the value of any variable parameter from recipe to recipe. Further, the RECIPR block can change which steps are used for preparation of a batch, and the order that the steps are executed.
RECIPR blocks can be linked together in parallel (each block receives the recipe number as an input), with each block storing up to ten values for each of the recipe parameters. If more than ten values are required for each parameter, the RECIPR blocks can be linked together in series to provide the necessary number of values for each parameter (the second block is slaved to the first, the third is slaved to the second, etc.). Consequently, a batch process requiring eight recipes, each with five parameters, requires five RECIPR blocks to store the recipe data. A batch process requiring 28 recipes each with five parameters, requires 15 RECIPR blocks (two slave blocks connected to each of five master blocks).

123.4Recipe Handling for a Fixed Batch Sequence

In a fixed batch sequence, the order in which the steps are accomplished remains fixed from recipe to recipe, but the amounts of ingredients and the conditions they are added in may vary. The following lube oil blending example illustrates how fixed-sequence recipes can be edited. Table 123-2 shows the contents of a series of recipes.

Table 123-2 Recipe Contents

Parameter	Description
1	Percent component A in blend
2	Percent component B in blend
3	Percent component C in blend
4	Percent component D in blend
5	Percent component E in blend
6	Percent component F in blend
7	Maximum flow rate of blend
8	Parameter number of component to be used for viscosity trim
9	Viscosity set point (SUS) for blend
10	Total amount of blend production (barrels)

Each of these parameters is stored in a RECIPR block. The recipe is stored in ten RECIPR blocks tied together in parallel. The module configuration logic required to manipulate and store this recipe is shown in Figure 123-1.

NOTES:

1. Each RECIPR block contains one of the recipe parameters.

2. REMSET blocks load the new values into the RECIPR function blocks. These blocks show the current value of the recipe parameter before any changes are entered.

3. The recipe being selected for use is from the REMSET at block address 1000.

4. The RCM at block address 907 is used to write the new values into the RECIPR blocks through REMSETs at block addresses 908 through 917.

5. In this configuration, both the REMSET that selects the recipe parameter for editing and RCM that writes the new parameter value to the RECIPR block are interlocked to allow editing of the recipe only during Step 1 of the batch sequence. Without this interlock, the recipe can be edited at any time during the execution of the batch.

6. After the recipe is edited, it can be downloaded to the RECIPR blocks by triggering the RCM at block address 907 from the console.

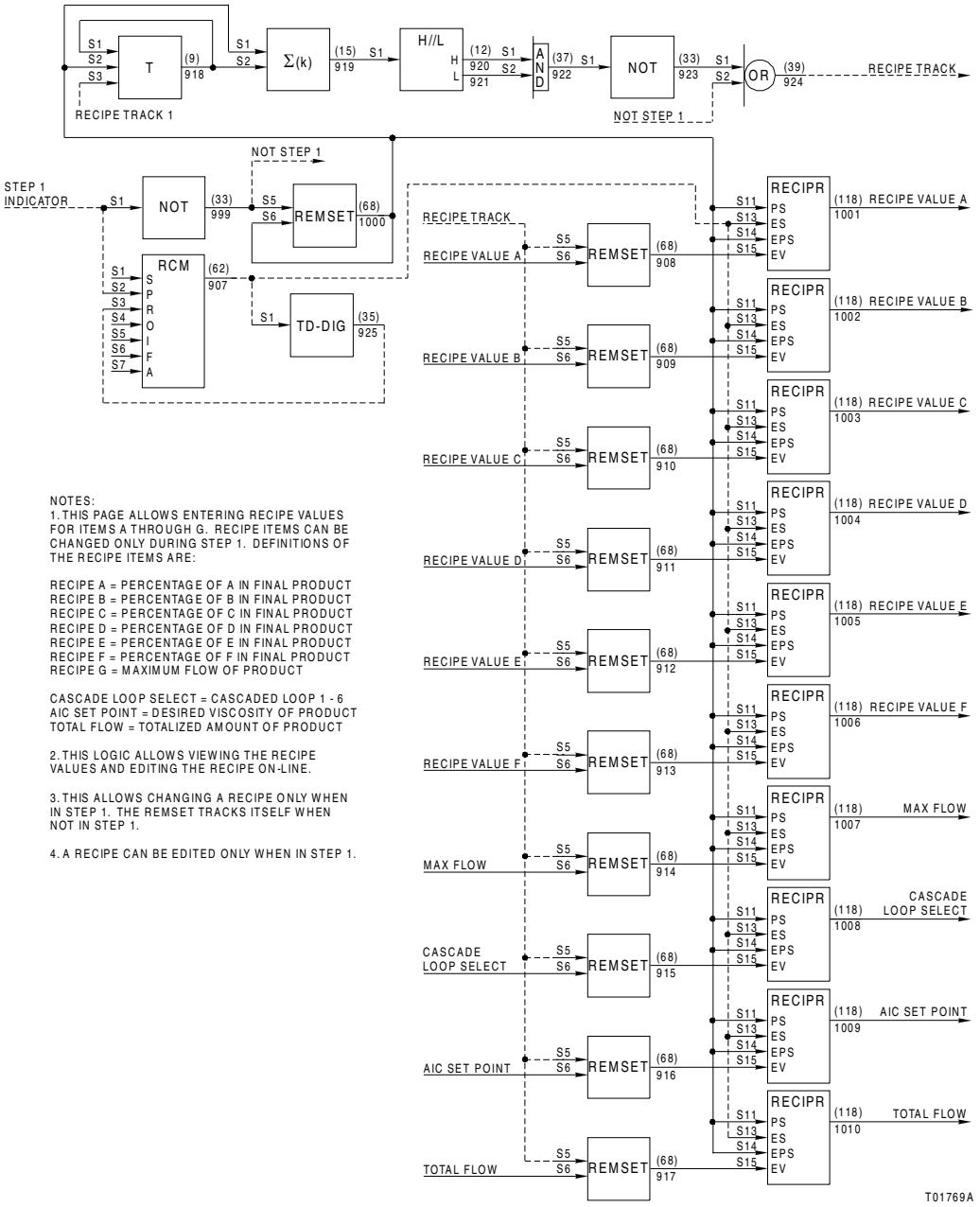
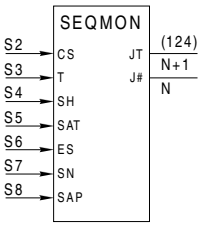


Figure 123-1 Fixed-Sequence Batch Control Example with Online Editing

124.Sequence Monitor

The sequence monitor function code controls the execution of a sequence generator block by selecting the order of step execution in a process based on system and external inputs. The sequence monitor block performs a logic action based on the value of the control status input from a device monitor block and a boolean step trigger input. Depending on the value of the control status input, the next step can be a fault step or a normal step determined by internal logic. The sequence monitor block can be placed in either automatic or semi-automatic mode, and includes hold/initialize and executed stop (E-STOP) inputs. Each sequence monitor block handles up to eight steps. If more than eight steps are required, a number of sequence monitor blocks can be chained together in series fashion with S1. The sequence monitor block defines step type, next step, and fault values for each step.



Outputs

Blk	Type	Description
N	R	Jump step number
N+1	B	Jump step trigger

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of next sequence monitor block associated with this sequence of steps: 0 = no more blocks
S2	N	5	I	Note 1	Block address of control status input
S3	N	0	I	Note 1	Block address of step trigger
S4	N	0	I	Note 1	Block address of hold/initialize input (hold current step on 0 to 1 transition of input)
S5	N	0	I	Note 1	Block address of semi-automatic trigger (initiate next step on a 0 to 1 transition of input)
S6	N	0	I	Note 1	Block address of E-STOP input (block will be reset to Step 0 when <S6>=1)
S7	N	5	I	Note 1	Block address of initial step number (reset step)
S8	Y	1	I	Note 1	Block address of semi-automatic permissive: <S8> = 0 = semi-automatic mode allowed <S8> = 1 = semi-automatic mode not allowed
S9	Y	0	I	0 - 22	Step 1 type: 0X = permit hold and semi-automatic modes 1X = permit hold only 2X = permit no option X0 = advance when <S2> = 0.0 and <S3> = 1only X1 = advance when <S2> = 0.0 X2 = advance when <S3> = 1 only
S10	Y	0	I	0 - 22	Step 2 type
S11	Y	0	I	0 - 22	Step 3 type
S12	Y	0	I	0 - 22	Step 4 type
S13	Y	0	I	0 - 22	Step 5 type
S14	Y	0	I	0 - 22	Step 6 type
S15	Y	0	I	0 - 22	Step 7 type
S16	Y	0	I	0 - 22	Step 8 type

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S17	Y	0.000	R	Full	Step to execute after Step 1 if input is good
S18	Y	0.000	R	Full	Step to execute after Step 2 if input is good
S19	Y	0.000	R	Full	Step to execute after Step 3 if input is good
S20	Y	0.000	R	Full	Step to execute after Step 4 if input is good
S21	Y	0.000	R	Full	Step to execute after Step 5 if input is good
S22	Y	0.000	R	Full	Step to execute after Step 6 if input is good
S23	Y	0.000	R	Full	Step to execute after Step 7 if input is good
S24	Y	0.000	R	Full	Step to execute after Step 8 if input is good
S25	Y	0.000	R	Full	Step 1 fault step (control status input bad)
S26	Y	0.000	R	Full	Step 2 fault step (control status input bad)
S27	Y	0.000	R	Full	Step 3 fault step (control status input bad)
S28	Y	0.000	R	Full	Step 4 fault step (control status input bad)
S29	Y	0.000	R	Full	Step 5 fault step (control status input bad)
S30	Y	0.000	R	Full	Step 6 fault step (control status input bad)
S31	Y	0.000	R	Full	Step 7 fault step (control status input bad)
S32	Y	0.000	R	Full	Step 8 fault step (control status input bad)

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

124.1Explanation

The sequence monitor block uses the values of two inputs to determine the next step number in a batch process. The first input is the control status input read from a device monitor block. This input defines the current state of the devices that the sequence generator block controls. This input can be 0.0 (good), 1.0 (bad), or 2.0 (waiting). If even one device is bad, this input is marked bad (value equals 1.0). The second input is the step trigger. The step trigger is dependent on the current step and the results of auxiliary logic associated with the device in question.

Each step of a batch sequence often requires auxiliary logic to perform functions in addition to activating/deactivating discrete devices (i.e., change controller set points, totalize flow, etc.). Figure 124-1 shows how this auxiliary logic ties into the batch execution configuration. The auxiliary logic executes according to the current step number read from the sequence generator

block. The current step number input then selects from the results of that auxiliary logic the boolean signal that will be output to the sequence monitor block as the step trigger.

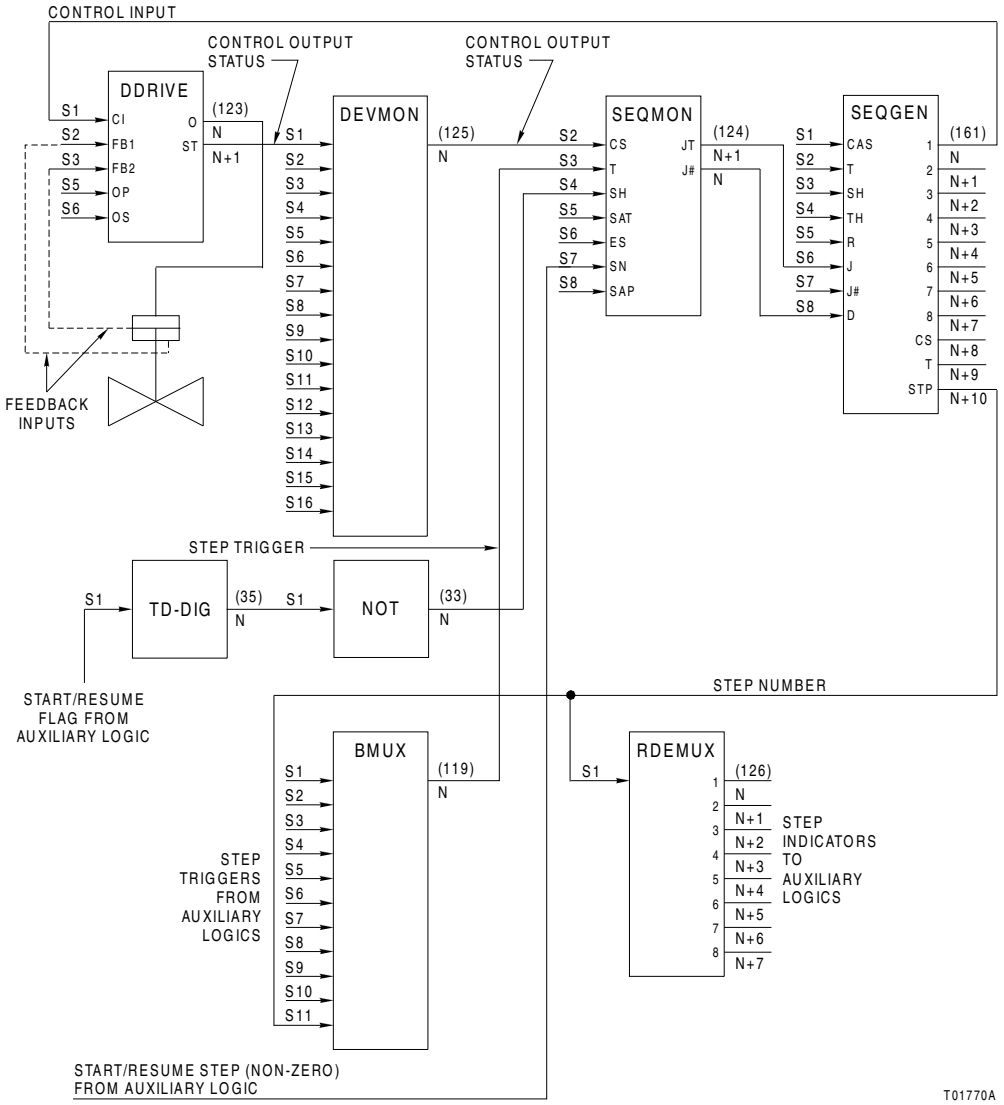


Figure 124-1 Sequence Monitor Block Used in a Batch Operation

The values of the step trigger and the control status input are tested against several operator defined parameters to determine which step executes next.
The sequence monitor block has both a next step and a fault step for each step number. When the control status input is bad (1.0), this function automatically selects the fault step number. The fault step number goes to the sequence generator block if the step type requires a good status input to continue with the sequence. If the control status input is good (0.0), or waiting (2.0), the output depends on the step type defined in the specification descriptions.
The sequence monitor block can operate in either an automatic or semi-automatic mode. In the automatic mode, the sequence is dependent on the values of the control status input and the step trigger. In the semi-automatic mode, the sequence is dependent on the values of the control status input, the step trigger, and the semi-automatic step trigger. The operator must activate the semi-automatic step trigger manually to proceed with the sequence as one of the conditions to go to the next step. The block can be placed in semi-automatic mode only if both the semi-automatic permissive and step type specifications permit it.

NOTE: For the sequence monitor to begin execution from Step 0, the old/initialize input <S4> must toggle from one to zero while the value of <S7> dictates the beginning step number to execute.

124.1.1Specifications

S1 – NXT
Block address of next sequence monitor block in the series of blocks used to execute the sequence. If this value equals zero, there are no more blocks in the sequence. Each sequence monitor block can accommodate eight steps of a

sequence. If the sequence contains more than eight steps, sequence monitor blocks can be linked in a series, with the last step of the first block initializing the first step of the next block, etc. Therefore, Steps 1 through 8 in the second sequence monitor block are Steps 9 through 16 in the sequence.

S2 – CSI
Block address of control status input (typically a device monitor block). The value in this block represents the control status of all the devices in the control loop. The output of this block is good when all the inputs are good, bad when any one input is bad, and waiting if any input is waiting for a reply from a device driver block and no inputs are bad. This value and the value of the step trigger are used to select the next step in the sequence. When the control status input is bad, the fault value for the next step is automatically output if the step type is one that requires a good status to continue with the sequence. If the control status input is good or waiting, output depends on the step type (S9) for the current step.

0.0 = good
1.0 = bad
2.0 = waiting

NOTE: Any step trigger (S3) or semi-automatic mode trigger (S5) that occurs while the control status input is 2.0 (waiting) is remembered and therefore acted upon when the control status input becomes 0.0 (good).

S3 – STEP
Block address of step trigger. The value in this block, used with the control status input, selects the next step in the control sequence. Auxiliary logic used for the batch process generates the step trigger. The sequence monitor recognizes the trigger when it goes from a zero to one state and when it is held high in its one state.

S4 – HOLD
Block address of hold/initialize value. The sequence monitor block can hold the sequence at certain steps. The hold/initialize function is active only when the step type specification is configured to permit it.

1 = the sequence holds at the current step.

1 to 0 transition = the sequence monitor block begins execution with the initial step specified by S7.

0 = sequence is in run mode.

NOTE: To enable the sequence monitor to resume running from the current step after the hold, it is necessary to feed the jump step number output N into input S7.

S5 – SEMI
Block address of semi-automatic mode trigger. When the semi-automatic mode is permitted by step type (S9) and the semi-auto permissive (S8), the operator must change this value from zero to one as one of the conditions for initiation of the next step of the sequence.

S6 – E-STOP
Block address of E-STOP (executed stop) input. The E-STOP specification drives the sequence to Step 0 (the reset step) whenever it has a value of one. The block referenced by this specification is normally a remote control memory function block used as an E-STOP. Thus, reserve the disable mask (Step 0) in sequence generator blocks driven by the sequence monitor block for E-STOP.

S7 – INIT
Block address of initial step number. When the block is initialized by a one to zero transition of <S4>, the sequence begins with the step identified in this block.

S8 – PERM
Block address of the semi-automatic permissive. When the value in the block is a zero, the semi-automatic mode is permitted unless step type (S9) overrides it.

<S8> = 0, semi-automatic mode permitted.
<S8> = 1, automatic mode permitted.

S9 – TYPE1
Step 1 type. Each step type is made up of two digits. The ones digit defines the states of the control status input and step trigger necessary for the block to initiate the next step of the sequence. The tens digit identifies the control options available to the operator for the step.

X X

Control status input state and step trigger.
X0 = advance when <S2> = 0.0 and <S3> = 1
X1 = advance when <S2> = 0.0
X2 = advance when <S3> = 1

Control options.
0X = permit hold and semi-automatic modes
1X = permit hold
2X = permit no option

S10 to S16 –
TYPE2 to TYPE8
Step types for Steps 2 through 8 defined in S9.

S17 to S24 –
STEP1 to STEP8
Define the number of the step the system should execute after Step *n* (*n* equals one to eight) is completed when the block is operating normally.

S25 to S32 -
FAULT1 to FAULT8
Define the number of the fault step the system should execute after Step *n* (*n* equals one to eight) is completed when the block receives a bad (1.0) input from the device monitor block and step type requires a good control status input.

124.1.2Outputs

N
(Jump step number) Identifies the number of the step performed at the conclusion of the current step.

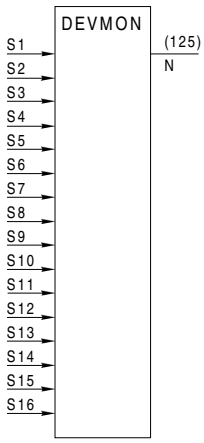
N+1
(Jump step trigger) Upon a zero to one transition of output N+1, initiates a step jump in the sequence generator block. The sequence generator block then executes the step identified by the jump step number. A zero to one transition of this value occurs when the sequence monitor block proceeds to the next step in the sequence (when the conditions defined by the step type specification are met). This output makes a one to zero transition one segment cycle after the zero to one transition.

124.2Applications

Refer to function code 123 for an example of the sequence monitor block used in a batch process. Sequence monitor blocks can also be used for sequential control.

125.Device Monitor

The device monitor block takes the control status outputs from up to 16 device drivers or device monitors to provide a common control output status. The control output status can be sent to another device monitor block, or to a sequence monitor block that determines the next step in a process. When all inputs are good (0.0), the output is good (0.0). When any input is bad (1.0), the output is bad (1.0). When any input is waiting (2.0) and no input is bad (1.0), the output is waiting.



Outputs

Blk	Type	Description
N	R	Control output status: 0.0 = good 1.0 = bad 2.0 = waiting

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of CO status 1
S2	N	5	I	Note 1	Block address of CO status 2
S3	N	5	I	Note 1	Block address of CO status 3
S4	N	5	I	Note 1	Block address of CO status 4
S5	N	5	I	Note 1	Block address of CO status 5
S6	N	5	I	Note 1	Block address of CO status 6
S7	N	5	I	Note 1	Block address of CO status 7
S8	N	1	I	Note 1	Block address of CO status 8
S9	N	5	I	Note 1	Block address of CO status 9
S10	N	5	I	Note 1	Block address of CO status 10
S11	N	5	I	Note 1	Block address of CO status 11
S12	N	5	I	Note 1	Block address of CO status 12
S13	N	5	I	Note 1	Block address of CO status 13
S14	N	5	I	Note 1	Block address of CO status 14
S15	N	5	I	Note 1	Block address of CO status 15
S16	N	5	I	Note 1	Block address of CO status 16

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

125.1Explanation

The device monitor block monitors a group of control output status values. It reads those values from a device driver or other device monitor block and outputs a value dependent on the values of the inputs.

For example, if a sequence generator block defines output states for ten field devices, the device monitor block associated with that control loop receives the control output status for the devices and generates a common output. That output goes to the sequence monitor block controlling the step output of the sequence generator block. The sequence monitor block then uses that information to select the next step in the process. It is used as a feedback from the field to alert the system of an abnormal state of the field devices.

Figure 125-1 illustrates this control scheme. If a control loop controls more than 16 devices, the output of additional device monitor blocks can be fed into another device monitor block to establish a common output. Therefore, no matter how many

devices are being controlled by a single sequence generator block, the control output status can be reduced to one value representing all inputs.

125.1.1 Specifications

S1 to S16 –
COS1 to COS16
Block addresses of the device drivers control status output or the device monitor block containing the control output status values for other monitored devices.

125.1.2 Output

N
Control output status based on the control output status of the devices monitored by the device monitor block.
0.0 = all inputs good
1.0 = any input bad
2.0 = no inputs bad and at least one input waiting

NOTE: Only one input must be bad for the output to be bad.

125.2 Applications

Device monitor blocks can be used to control any system that requires the output of a group of devices be a certain value before a control step is implemented. Refer to function code 123 for an example of the device monitor block used in a batch control process.

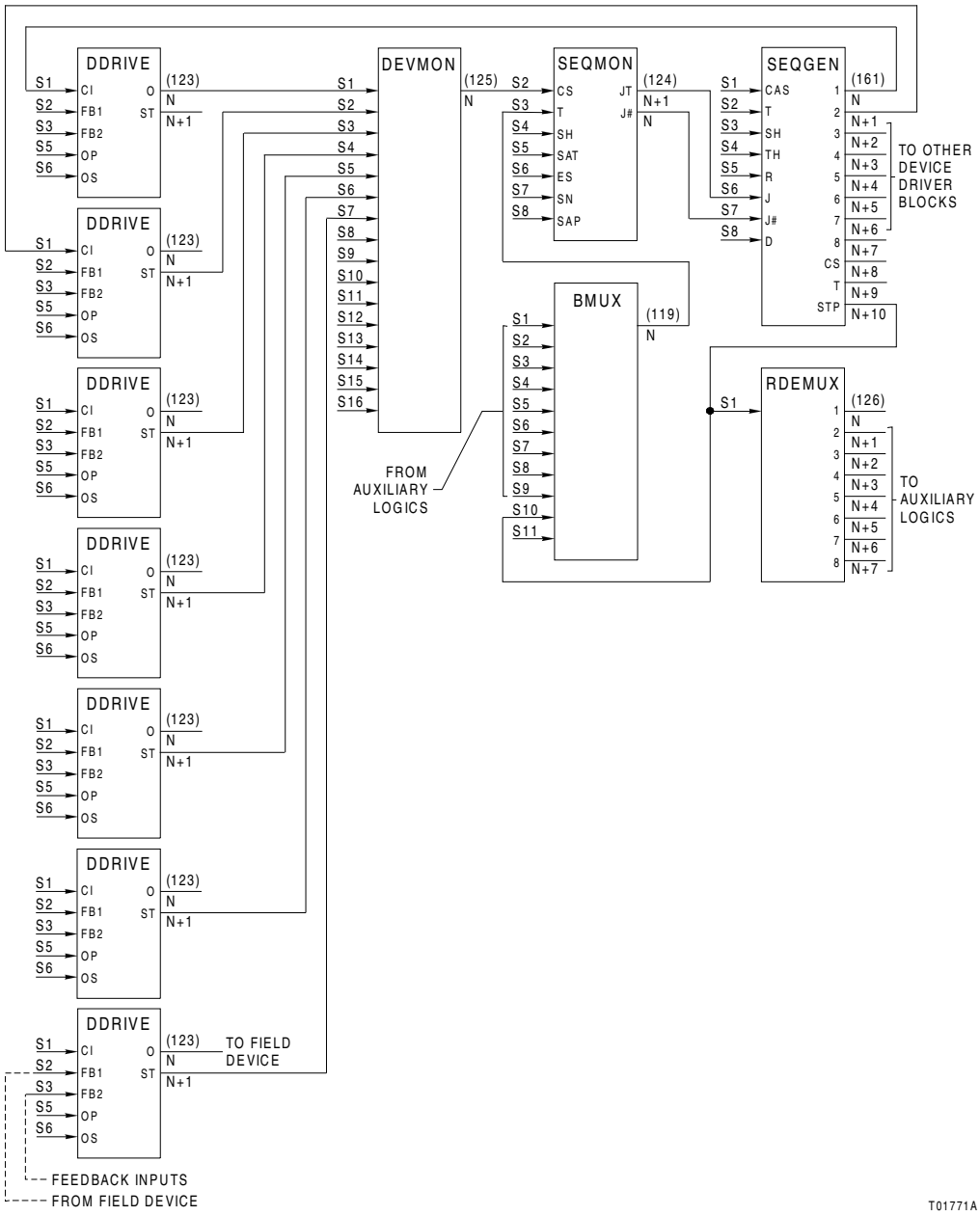
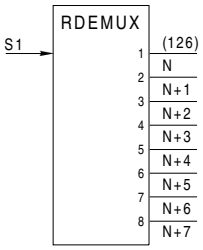


Figure 125-1 Controlling Multiple Field Devices

126.Real Signal Demultiplexer

The real signal demultiplexer function code outputs boolean signals based on a real input value. The real value can be converted to boolean values in three modes (select, integer and BCD). An unlimited number of real signal demultiplexer blocks may be linked together in the select mode (demultiplex the real input into the required number of boolean outputs). Up to four blocks may be linked together in the integer and BCD modes for demultiplexing. The first block in the link list is the group master. The master accepts the input signal and drives the outputs for the group.



Outputs

Blk	Type	Description
N	B	Output 0 (least significant bit)
N+1	B	Output 1
N+2	B	Output 2
N+3	B	Output 3
N+4	B	Output 4
N+5	B	Output 5
N+6	B	Output 6
N+7	B	Output 7 (most significant bit)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of select input
S2	N	0	I	0 - 2	Conversion mode: 0 = select (unlimited number of blocks in link list) 1 = integer (maximum of 4 blocks in link list) 2 = BCD (maximum of 4 blocks in link list)
S3	N	0	I	Note 1	Block address of next block in link list

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

126.1Explanation

126.1.1Select Mode

In the select mode, the block output is all zeros and a one in the position specified by the real input. For example, if the real input is a seven, the eighth boolean output is a one and all others are zeros. In this mode, an unlimited number of real signal demultiplexer blocks can be linked together. If <S1> is less than zero, output zero equals one. If <S1> is greater than the maximum, the last output is set. If a number outside of the available range (for example, zero through seven in the case of a master) is selected, the nearest output is set. For example, if the number selected is -1, the zero output changes to a one.

126.1.2Integer Mode

The integer mode converts the real input to a binary bit pattern. For example, if the input is 135, the binary output is 1000111, since 135=128+4+2+1. The least significant digit is output zero of the master block. This is true no matter how many blocks are in the link list. Table 126-1 shows the integer mode input to output relationship.

Up to four real signal demultiplexer blocks can be linked together in the integer mode, allowing the conversion of any real number up to 4.2×10^9 to binary digits.

Table 126-1 Integer Mode Input to Output Relationship

Output	MSB			Real to Binary				LSB
Output Number	7	6	5	4	3	2	1	0
Weighted Decimal Value	128	64	32	16	8	4	2	1
Boolean Outputs Example: 135 =	1	0	0	0	0	1	1	1

126.1.3BCD Mode

The BCD mode converts the real input to BCD digits. Each digit of the real number converts to four boolean digits by writing the real digit as the sum of the first four powers of two (eight, four, two, one). For example, if the real digit is a six, the boolean outputs for that digit are 0110 since 6=0(8)+1(4)+1(2)+0(1). Table 126-2 shows how each group of outputs represents two real digits.

Up to four real signal demultiplexer blocks can be linked together in the BCD mode allowing for the conversion of any real number up to eight digits in length. The least significant digit is always represented by outputs zero through three of the master block, no matter how many blocks are linked in series. Figure 126-2 and Table 126-3 show this arrangement.

Table 126-3 shows sample outputs for each of the three modes. There are two blocks in the link list. Outputs zero through seven are from the group master, and outputs eight through 15 are from the second block.

Table 126-2 Each Group of Outputs Represents Two Real Digits

Output	MSB			Real to Binary				LSB
Output number	7	6	5	4	3	2	1	0
Weighted decimal value	Tens Digit				Ones Digit			
	8	4	2	1	8	4	2	1
Boolean outputs example: 06	0	0	0	0	0	1	1	0

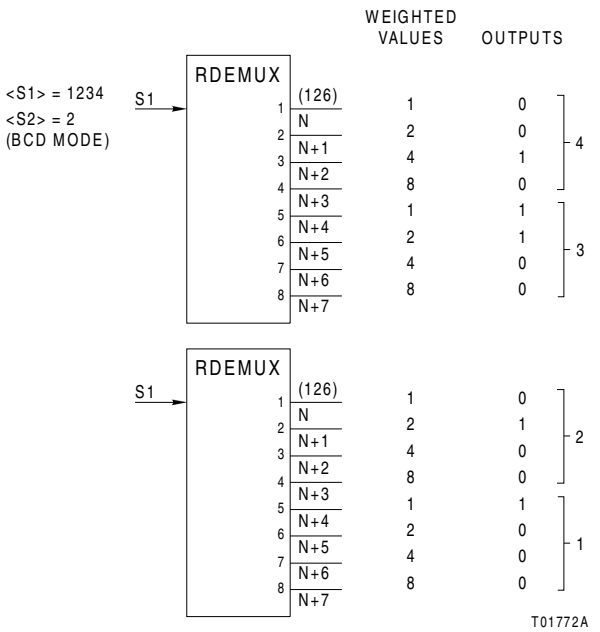


Figure 126-1 Output Values in the BCD Mode

Table 126-3 Sample Outputs of a Real Signal Demultiplexer Block

Input Value	Conversion Type	MSB								Output Number							LSB
		15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
6 (master only)	Select Integer BCD									0	1	0	0	0	0	0	0
										0	0	0	0	0	1	1	0
										0	0	0	0	0	1	1	0
15 (both blocks)	Select Integer BCD	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
		0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1
1234 (both blocks)	Integer BCD	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0
		0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0

126.2 Applications

Real signal demultiplexer blocks can be used for a variety of control purposes. Figure 126-2 shows how, in the select mode, the boolean outputs can be used to trigger execution of auxiliary logic sequences in a batch process.
In the select mode, the real signal demultiplexer block converts the real step number input from the sequence generator block into a series of boolean outputs. The boolean outputs act as triggers for the auxiliary logics associated with the sequence. A value of one is output as the trigger for the auxiliary logic associated with the current step, initiating the execution of that logic.

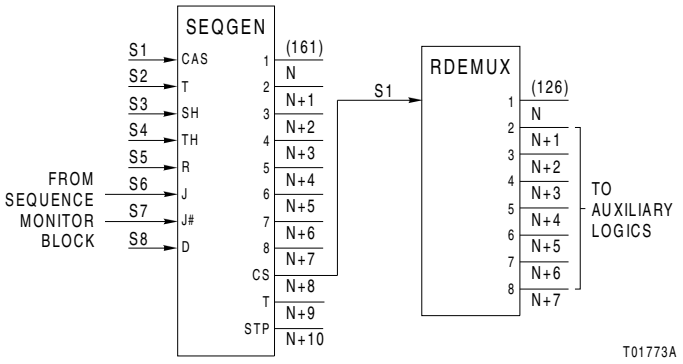
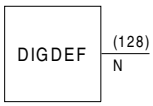


Figure 126-1 Recipe Selection

128.Slave Default Definition

The slave default definition function code provides the default values for function blocks that interface with two digital I/O modules (e.g., BCD output, function code 115).
There are four types of digital I/O modules. The IMDSO15 module has eight outputs, and the IMDSO14 module has 16 outputs. The slave default definition function code provides the default values for up to 16 outputs per I/O module. The outputs are in two groups (group A and B) of eight per module. When defining the default values for IMDSO15 modules, place them in group A. Group B remains unused since these modules have eight outputs.



Outputs

Blk	Type	Description
N	R	0.0 (no useful output)

Specifications

Spec	Tune	Default	Type	Range	Description	
S1	N	0	I	Note 1	Block number of first digital I/O module function block	Used with function code 83 ***or*** function code 115
S2	N	0	B	Full	Default state for output 1	
S3	N	0	B	Full	Default state for output 2	
S4	N	0	B	Full	Default state for output 3	
S5	N	0	B	Full	Default state for output 4	
S6	N	0	B	Full	Default state for output 5	
S7	N	0	B	Full	Default state for output 6	
S8	N	0	B	Full	Default state for output 7	
S9	N	0	B	Full	Default state for output 8	
S10	N	0	B	Full	Default state for output 9	Used with function code 115 only
S11	N	0	B	Full	Default state for output 10	
S12	N	0	B	Full	Default state for output 11	
S13	N	0	B	Full	Default state for output 12	
S14	N	0	B	Full	Default state for output 13	
S15	N	0	B	Full	Default state for output 14	Used with function code 115 only
S16	N	0	B	Full	Default state for output 15	
S17	N	0	B	Full	Default state for output 16	

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description	
S18	N	0	I	Note 1	Block number of second digital I/O module function block	Used with function code 83 ***or*** function code 115
S19	N	0	B	Full	Default state for output 1	
S20	N	0	B	Full	Default state for output 2	
S21	N	0	B	Full	Default state for output 3	
S22	N	0	B	Full	Default state for output 4	
S23	N	0	B	Full	Default state for output 5	
S24	N	0	B	Full	Default state for output 6	
S25	N	0	B	Full	Default state for output 7	
S26	N	0	B	Full	Default state for output 8	
S27	N	0	B	Full	Default state for output 9	Used with function code 115 only
S28	N	0	B	Full	Default state for output 10	
S29	N	0	B	Full	Default state for output 11	
S30	N	0	B	Full	Default state for output 12	
S31	N	0	B	Full	Default state for output 13	
S32	N	0	B	Full	Default state for output 14	
S33	N	0	B	Full	Default state for output 15	
S34	N	0	B	Full	Default state for output 16	

NOTE:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

128.1Explanation

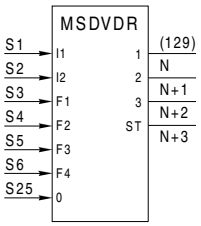
Function code 128 selects the default values for all outputs associated with any type of DSO module. If the control module goes bad, then the outputs can be forced to the default values until the module can be replaced or repaired.
The BCD output (function code 115) selects the failover to default settings for each group of eight outputs. Specification S2 of function code 115 defines the I/O module. If the hundreds digit of S2 equals logic 0, then the outputs go to the default values as configured by function code 128 on loss of the control module. Setting the hundreds digit to logic 1 holds the current output. Function code 128 provides default values for a maximum of two DSO modules. If one or both of the modules have eight outputs, then outputs one through eight are the default values, and outputs nine through 16 remain unused.

129.Multistate Device Driver

The multistate device driver (MSDD) function code provides a means of controlling field equipment (i.e., variable speed motor) or control schemes that have more than one control mode. The MSDD block provides four state controls with feedback. Two control inputs or an operator input selects one of four output masks for control action. The output masks provide three boolean signals simultaneously to the actual control logic. The MSDD block accepts four boolean feedback inputs that define the actual state of the control scheme. A control output status signal from the MSDD block can interface to sequencing logic to trigger the next step in a process. The control output status represents the status of the controlled process as defined by the feedback inputs and a feedback wait time. Control output status can be good (0.0), bad (1.0), or waiting (2.0).

NOTE: Because commands are buffered during module startup, some time may elapse between an operator action and the execution of that action during startup. The elapsed time is determined by the startup period specified by S4 of function code 90, and it is dependent on the startup in progress flag being set to 1 in function code 81.

The MSDD block is exception reported. Exception reports contain states: output, feedback, alarm; and statuses: control output and mode. An exception report generates when the block output mask changes, the block generates an alarm, a feedback input changes state, or T_{max} for exception reporting expires.



Outputs

Blk	Type	Description
N	B	Control output 1
N+1	B	Control output 2
N+2	B	Control output 3
N+3	R	Control output status: 0.0 = good 1.0 = bad 2.0 = waiting

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of control input 1
S2	N	0	I	Note 1	Block address of control input 2
S3	N	0	I	Note 1	Block address of feedback input 1
S4	N	0	I	Note 1	Block address of feedback input 2
S5	N	0	I	Note 1	Block address of feedback input 3
S6	N	0	I	Note 1	Block address of feedback input 4
S7	N	0	I	000 - 111	Default mask
S8	N	0	I	000 - 111	Output mask 1
S9	N	0	I	000 - 111	Output mask 2
S10	N	0	I	000 - 111	Output mask 3
S11	Y	0	I	0000 - 2222	Feedback corresponding to output mask 1
S12	Y	0	I	0000 - 2222	Feedback corresponding to output mask 2
S13	Y	0	I	0000 - 2222	Feedback corresponding to output mask 3. Each digit can be: 0 = input should be 0 1 = input should be 1 2 = input could be 0 or 1

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S14	Y	0	I	Full	Control output status override: **Status override:** XX0 = none XX1 = output status XX2 = output status and alarm **Control override:** X0X = go to manual mode and default control outputs X1X = hold current mode and control outputs X2X = go to manual mode and hold current control outputs X3X = go to auto mode and set outputs as selected by current control inputs X4X = go to manual mode and set outputs as selected by operator interface device **Operation:** 0XX = normal operation 1XX = early good status enable
S15	Y	1	B	0 or 1	Manual mode permissive: 0 = no 1 = yes
S16	Y	0.000	R	0 - 9.2 E18	Feedback waiting time (seconds)
S17	Y	0.000	R	0 - 9.2 E18	Fault waiting time
S18	N	0	I	0 - 255	MSDD display type
S19	Y	1	I	1 - 32	Next allowable mask numbers from output mask 1 in manual mode
S20	Y	2	I	1 - 32	Next allowable mask numbers from output mask 2 in manual mode
S21	Y	3	I	1 - 32	Next allowable mask numbers from output mask 3 in manual mode
S22	Y	0.000	R	0 - 9.2 E18	Length of pulsed outputs (if 0, sustained outputs)
S23	N	1	B	Full	Initial mode: 0 = manual 1 = auto
S24	Y	0.000	R	Full	Startup track flag: 0.0 = no track during startup 1.0 = control outputs track control inputs during startup
S25	N	0	I	Note 1	Block address of control override

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC

129.1 Explanation

The MSDD block controls output logic or other control schemes that can be operated in more than one state, for example, variable speed mixers and motors. The output state depends on two control input signals or an operator input. The values of these signals determine which of four output masks will be sent to the controlled process. Each output mask provides three user-defined boolean signals that drive the process.

The user defines the output masks based on which state the process should be in for given combinations of inputs. The user also defines feedback masks corresponding to the output masks. The block compares the feedback inputs from the process with the feedback mask containing ***normal*** feedback values for a given output mask. If the feedback inputs do not

match the feedback mask within the feedback waiting time, an alarm generates and is exception reported. Table 129-1 defines the output and feedback masks selected with various combinations of control inputs. Figure 129-1 illustrates how the block interprets the feedback masks.

Table 129-1 Truth Table for Selection of Output Masks in Auto Mode

Control Input		Mask Selected	Corresponding Feedback Mask	Next Allowable State Mask (Manual Mode Only)
1 <S1>	2 <S2>			
0	0	Default mask (S7)	None	None
0	1	Output mask 1 (S8)	Feedback mask 1 (S11)	Next state mask 1 (S19)
1	0	Output mask 2 (S9)	Feedback mask 2 (S12)	Next state mask 2 (S20)
1	1	Output mask 3 (S10)	Feedback mask 3 (S13)	Next state mask 3 (S21)

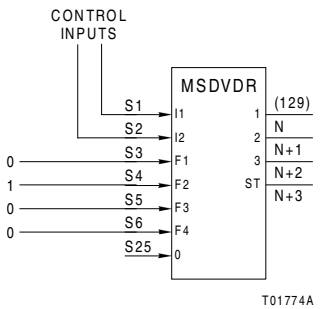


Figure 129-1 Relationship Between Feedback Inputs and Feedback Masks

If the control inputs are: <S1> = 0, <S2> = 1, then output mask one (S8), feedback mask one (S11), and next state mask one (S19) are implemented.

If feedback mask one (S11) = 0100, then the values of S3, S4, S5 and S6 must correspond to S11 as shown in Figure 129-1 to produce a good control output status (N+3).

NOTE: All feedback masks can input a four-digit number for the specification in which each digit is either a zero, one or two. Figure 129-1 illustrates this relationship.

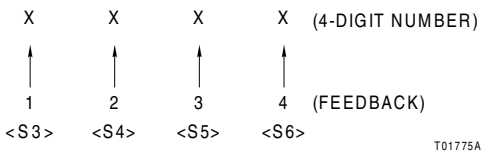


Figure 129-1 Four-Digit Feedback Mask

This block supports both automatic and manual modes. The user selects the initial mode with S23. In automatic mode, two boolean inputs from the control system select the mask that drives the outputs as shown in Table 129-1. The default mask may not be manually initiated. To operate the device in manual mode, the manual mode permissive (S15) must be one (yes).
Auto/manual status may be monitored internally by using the test alarm function (function code 69). Figure 129-1 shows this simplified configuration. A control output status value is generated based on the feedback inputs, feedback waiting time, and feedback masks. If the value of any feedback input does not match the value of the feedback mask for a given output

mask, an exception report generates and goes to the console, and the control output status value will be set to bad (1.0) unless overridden. The control output status can have one of three values (0.0 = good, 1.0 = bad, or 2.0 = waiting).

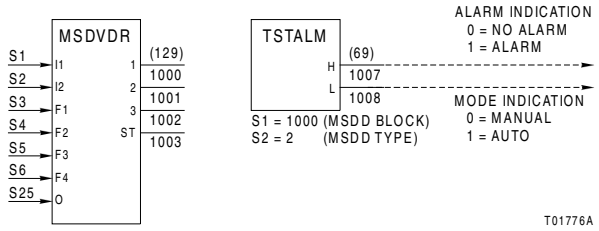


Figure 129-1 Auto/Manual Status Monitored with Test Alarm Block

The user can set a fault wait timer to allow a delay between the time the exception report indicates an alarm, and when the control output status reflects that alarm. The exception reported alarm occurs first. This provides a fixed interval of time during which the control output status can be overridden. The control output status override forces the control output to the good state. Specification S14 implements the control output status override (no override, override output status only, or override output status and alarm). Selecting override status and alarm disables exception report alarms.
A pulse output timer allows the selection of pulsed or maintained control outputs. When S22 is a non zero value, the outputs are pulsed to the selected output mask state for the time period selected, then set back to the default mask state. When S22 is zero, the outputs are sustained.

129.1.1 Specifications

S1 – CI1
Block address of control input one.

S2 – CI2
Block address of control input two.

NOTE: A block address of zero (default) in both S1 and S2 enables using this block with batch language only.

S3 – FB1
Block address of feedback input one. The feedback inputs are signals from the field that define the actual state of the device. The values of the feedback inputs are compared to the feedback status masks to determine the control output status. If the feedback inputs do not match the feedback status masks for given control outputs, an exception report with alarm generates.

S4 – FB2
Block address of feedback input two. Refer to ***S3 – FB1*** for an explanation.

S5 – FB3
Block address of feedback input three. Refer to ***S3 – FB1*** for an explanation.

S6 – FB4
Block address of feedback input four. Refer to ***S3 – FB1*** for an explanation.

S7 – DMASK
Default output mask. The default output mask is a value consisting of three boolean digits that control the output logic. This is configurable, and can be any combination of zeros and ones. This output mask will be sent to the output logic when the control inputs are both zero. The output mask configuration is shown as follows:

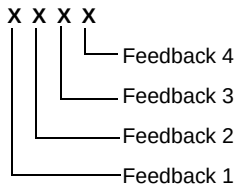
X X X
- **Ones digit** state of output N+2
- **Tens digit** state of output N+1
- **Hundreds digit** state of output N

S8 – MASK1
Output mask one. This value consists of three boolean digits that control the output logic. It can be configured as any combination of zeros and ones. This output mask is sent to the output logic when control inputs one and two are zero and one in that order (in auto mode) or the operator depresses the state one pushbutton (in manual mode). Refer to ***S7 – DMASK*** for output mask configuration.

S9 – MASK2
Output mask two. This value consists of three boolean digits that control the output logic. It can be configured as any combination of zeros and ones. This output mask is sent to the output logic when control inputs one and two are one and zero, respectively (in auto mode) or the operator depresses the state two pushbutton (in manual mode). Refer to ***S7 – DMASK*** for output mask configuration.

S10 – MASK3
Output mask three. This value consists of three boolean digits that control the output logic. It can be configured as any combination of zeros and ones. This output mask is sent to the output logic when control inputs one and two are both one (in auto mode) or the operator depresses the state three pushbutton (in manual mode). Refer to ***S7 – DMASK*** for output mask configuration.

S11 – FDMSK1
Feedback corresponding to output mask one. This value consists of four boolean values that identify the normal state of the feedback inputs when output mask one is the control. See Figure 129-1 for four-digit feedback masks.



S12 – FDMSK2
Feedback corresponding to output mask two. This value consists of four boolean values which identify the normal state of the feedback inputs when output mask two is the control. See Figure 129-1 for four-digit feedback masks. Refer to ***S11 – FDMSK1*** for output mask configuration.

S13 – FDMSK3
Feedback corresponding to output mask three. This value consists of four boolean values which identify the normal state of the feedback inputs when output mask three is the control. See Figure 129-1 for four-digit feedback masks. Refer to ***S11 – FDMSK1*** for output mask configuration.

S14 – COSOV
Control output status override. This value defines the override type currently implemented. The control output status override forces the control output to good, regardless of the feedback inputs and masks when the ones digit is set to a one or two. The alarm status is also overridden when the ones digit is set to a two. The tens digit overrides control output status and dictates block control only when <S25> is a logic 1.

NOTE: Early recognition of feedback cancels the feedback waiting time (S16) once feedback conditions have been met. This can result in bad status and alarm prior to feedback waiting time time-out.

S15 – MPERM
Sets the manual mode permissive. This tunable value defines if manual mode is permitted. This specification has no influence on these parameters and the output masks are never tunable while the feedback masks are always tunable.

0 = no (auto mode)
1 = yes (manual mode)

NOTE: If S15 changes from one to zero, then the block is forced to auto mode when S25 is a zero.

S16 – FDWAIT
Feedback waiting time. This value defines the time in seconds that the MSDD block waits before comparing the feedback inputs with the feedback masks. For example, if the block controls a variable speed motor, the feedback waiting time is the time it takes the motor to ramp from stopped to its top speed. This insures that measurements taken while the device is starting up or changing speeds are not used for control or indication. If at the end of the feedback time the feedbacks are not correct, then an alarm generates if not overridden by S14.

NOTES:
1. Without early good recognition implemented, the check for feedback follows the formula *S22* + *S16* = *check for feedback*.

2. With early good recognition implemented, updated control output status is updated after *S22* + *S16* expires or good feedback is met. Early good feedback cancels the feedback waiting time after the feedbacks match the state mask.

X X X

Status override
- 0 = no override.
- 1 = override output status only. Alarms are still exception reported to console.
- 2 = override output status and alarm (disable exception report alarms).

Control override
- 0 = go to manual mode (if manual mode permissive) and default control outputs.
- 1 = hold current mode and control outputs. If outputs are pulsed, go to default control outputs.
- 2 = go to manual mode (if permitted) and hold current control outputs. If outputs are pulsed, go to default control outputs.
- 3 = go to auto mode and set control outputs as selected by the current control inputs (S1 and S2).
- 4 = go to manual mode (if permitted) with current control outputs. Only applicable when S25 = 1. Control status will be forced to good while S25 = 1. If outputs are pulsed, go to default control outputs. Operator requests to change state are allowed subject to next allowable state (S19, S20, S21) constraints. Requests to change mode to auto are refused.

Operation - early recognition of feedbacks
- 0 = normal operation - wait for duration of S16.
- 1 = early good status enable. When the hundreds digit is set to one, the control output status is set to good as soon as the feedbacks indicate this condition.

NOTE: Early recognition of feedback cancels the waiting time (S16) once feedback conditions have been met. This can result in bad status and alarm prior to feedback waiting time time-out.

Example 1

Check for feedback at 5 seconds (S16 = 5). Maintained outputs (S22 = 0).

S22 + S16 = check feedback
0 + 5 = 5 seconds

Example 2

Check for feedback at 2 seconds (S16 = 2). Pulse output of 3 seconds (S22 = 3).

S22 + S16 = check feedback
3 + 2 = 5 seconds

S17 – FLTWAIT
Fault waiting time. This value defines the delay in seconds between an exception reported alarm and when the control output status indicates the fault condition to the rest of the control system. This allows a fixed interval of time during which the control output status override may be activated.

S18 – DDIS
MSDD display type. The console provides the capacity to create dynamic graphic and faceplate displays. This specification defines the console display type that represents the particular device. Custom displays can be created with the graphic display builder. Refer to the console operators instruction for information on building displays.

S19 – NXT1
Next allowable mask number for output mask one in manual mode. When the block is in manual mode, the operator selects the output mask by pressing the appropriate state pushbutton from the operator console. Table 129-1 identifies the masks selected for different combinations of inputs. This specification identifies which masks can be implemented after mask one. It is used to keep the operator from accidentally upsetting the process. For example, if S19 equals 23 or 32, then the operator can go to state two or state three directly.

Example:

S19 = 23 Either state two or three can be selected next.
S20 = 1 Only state one can be selected next.
S21 = 1 Only state one can be selected next.

S20 – NXT2
Next allowable mask number for output mask two in manual mode. Refer to ***S19 – NXT1*** for an explanation of NXT. If S20 equals 13 or 31, then the operator can go to state one or state three directly.

S21 – NXT3
Next allowable mask number for output mask three in manual mode. Refer to ***S19 – NXT1*** for an explanation of NXT. If S21 equals 12 or 21, then the operator can go to state one or state two directly.

S22 – PULSE
Length of the pulsed outputs in seconds. Output masks can be pulsed or sustained. If S22 is not equal to zero, the outputs are pulsed to the selected output state (determined by control inputs one and two or a console) for the length of time selected with S22, then set back to the default state (S7). If S22 equals zero, then the outputs are sustained.

S23 – INIT
Initial mode. This value defines the operating mode at startup.

0 = manual
1 = automatic

S24
Startup track flag. When this value is set to 0.0, the initial values for the control outputs are determined by S23 during startup. When this value is set to 1.0, the initial values for the control outputs will track the control input values regardless of the S23 setting. This specification is used when the state of function code 129 block must be synchronized with the state of an external process at the startup of the controller.

S25 – COVRD
Control override. When this input is one, the control status is forced good (0.0). The mode and output state control are selected with the tens digit in the control override definition specified by the control status override (S14).

129.1.2Outputs

N
Control output one. The three control outputs are grouped together to form output masks. Each control output is user configurable (zero or one). The control outputs are defined during configuration based on the function the control device is to perform for different combinations of control inputs. During execution, the values of the control inputs or an operator action determines the control output mask selected. This value is the current value of control output one being used for control.

N+1
Control output two. This value is the current value of control output two being used for control. Refer to output N explanation.

N+2
Control output three. This value is the current value of control output three being used for control. Refer to output N explanation.

N+3
Control output status. This value is output from the MSDD block to a sequence monitor block or other control logic to inform the control system of the current state of the driven device.

0.0 = good
1.0 = bad
2.0 = waiting

Good - feedback waiting time has elapsed or the inputs from the field agree with the feedback mask which corresponds to the current output mask.
Bad - feedback waiting time has elapsed and one or more inputs from the field do not agree with the feedback mask corresponding to the current output mask.
Waiting - feedback waiting time has not elapsed, and no comparisons between field inputs and feedback masks have been made yet.

NOTE: A logic 1 at S25 forces this output to 0.0 (good). This overrides a bad status (1.0). During output state transition, N+3 momentarily pulses from 0.0 to 2.0 then back to 0.0.

129.2 Applications

Figure 129-1 shows a sample faceplate. Figure 129-1 shows how to use function code 129 to control a variable speed motor. The allowable state transitions are off to low, low to high, low to off, high to low, and high to off.

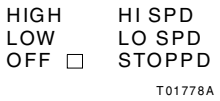


Figure 129-1 Sample Faceplate

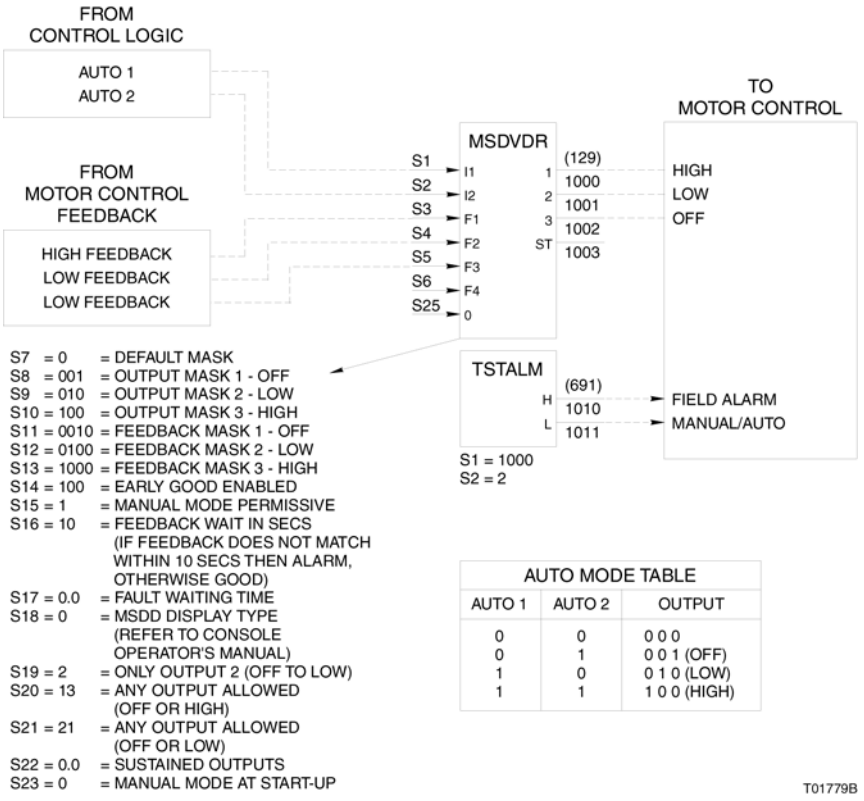


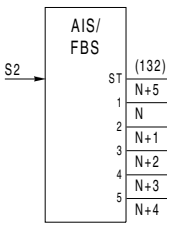
Figure 129-1 Controlling a Variable Speed Motor

132.Analog Input/Slave

The analog input/slave function code reads five analog inputs from an IMFEC12 Analog Input Slave or an IMFEC11 Field Bus Slave. Three analog input/slave function codes are required to utilize all 15 input channels on the module. When these function blocks are linked, only the status output of the primary block will be functional.
Function code 132 is also used with ABB FSK smart protocol and FSK bus applications. In the case of bus mode and multivariable bus mode applications, three instances of function code 132 are needed whenever there are 15 variables. This function block has six outputs. The first five are the inputs from the analog input module. The sixth output is the status of the analog input module.

The control system must be carefully evaluated to establish default values that will prevent personal injury and/or property damage in case of module failure.

Outputs



Blk	Type	Description
N	R	First analog input in engineering units
N+1	R	Second analog input in engineering units
N+2	R	Third analog input in engineering units
N+3	R	Fourth analog input in engineering units
N+4	R	Fifth analog input in engineering units
N+5	B	Input module status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Input module address
S2	N	2	I	Note 1	Block address of next analog input module
S3	N	0	I	0 or 1	Continue on input module error: 0 = trip master module 1 = continue to operate
S4	N	0	I	0 - 5	Input signal type of first input: 0 = 4 - 20 mA; also for digital mode 1 = 1 - 5 VDC 2 = -10 - +10 VDC 3 = 0 - +1 VDC 4 = 0 - +5 VDC 5 = 0 - +10 VDC
S5	Y	0.000	R	Full	Engineering unit zero for first input
S6	Y	0.000	R	Full	Engineering unit span for first input
S7	N	0	I	0 - 5	Input signal type of second input: 0 = 4 - 20 mA; also for digital mode 1 = 1 - 5 VDC 2 = -10 - +10 VDC 3 = 0 - +1 VDC 4 = 0 - +5 VDC 5 = 0 - +10 VDC
8	Y	0.000	R	Full	Engineering unit zero for second input

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S9	Y	0.000	R	Full	Engineering unit span for second input
S10	N	0	I	0 - 5	Input signal type of third input: 0 = 4 - 20 mA; also for digital mode 1 = 1 - 5 VDC 2 = -10 - +10 VDC 3 = 0 - +1 VDC 4 = 0 - +5 VDC 5 = 0 - +10 VDC
S11	Y	0.000	R	Full	Engineering unit zero for third input
S12	Y	0.000	R	Full	Engineering unit span for third input
S13	N	0	I	0 - 5	Input signal type of fourth input: 0 = 4 - 20 mA; also for digital mode 1 = 1 - 5 VDC 2 = -10 - +10 VDC 3 = 0 - +1 VDC 4 = 0 - +5 VDC 5 = 0 - +10 VDC
S14	Y	0.000	R	Full	Engineering unit zero for fourth input
S15	Y	0.000	R	Full	Engineering unit span for fourth input
S16	N	0	I	0 - 5	Input signal type of fifth input: 0 = 4 - 20 mA; also for digital mode 1 = 1 - 5 VDC 2 = -10 - +10 VDC 3 = 0 - +1 VDC 4 = 0 - +5 VDC 5 = 0 - +10 VDC
S17	Y	0.000	R	Full	Engineering unit zero for fifth input
S18	Y	0.000	R	Full	Engineering unit span for fifth input
S19	N	0.000	R	Full	Spare parameter
S20	N	0.000	R	Full	Spare parameter

NOTE:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

132.1Explanation

132.1.1Specifications

S1
Address of the analog input module.

S2
Block address of the next analog input block. The first block in the list is the master block. The rest of the blocks are under its control. Use the default value for the last block.

S3
Can be set to allow the module to continue to operate if there is an input module problem.

0 = trip master module
1 = continue to operate

S4
Defines the input signal type of the first input from the module.

0 = 4 to 20 milliamps, also used for FSK digital mode
1 = 1 to 5 VDC
2 = -10 to +10 VDC
3 = 0 to +1 VDC
4 = 0 to +5 VDC
5 = 0 to +10 VDC

S5
Defines the low limit of the first input in engineering units.

S6
Defines the range of the first input in engineering units. If this specification is set to zero, the input is considered unused.

NOTE: For any channel that is connected to an AVS smart positioner the low limit (S5) must be set to zero and the range value (S6) must equal 100. However, if the normal operating range of any of the outputs are expected to include numbers less than -5 or more than +105, the range value (S6) should equal 1 and the output of that channel should be treated as a normalized value. To unormalize the value, send the output of the block into a multiply block and multiply by 100. This is to get around the bad quality state which is automatically generated by function code 132 when the value is outside the -5 to +105 range.

S7 through S18
Define the input signal type, low limit and range for the other channels on the analog input module.

S19 and S20
Spare parameters.

132.2 Application

Figure 132-1 shows analog input blocks linked together. The inputs must be numbered consecutively.

NOTE: When using smartport, block numbering is critical when configuring analog inputs from smart transmitters. The block numbers for the three analog input/slave function codes must be numbered consecutively and the 15 smart field device definition function codes associated with the inputs must be numbered consecutive immediately following the inputs.

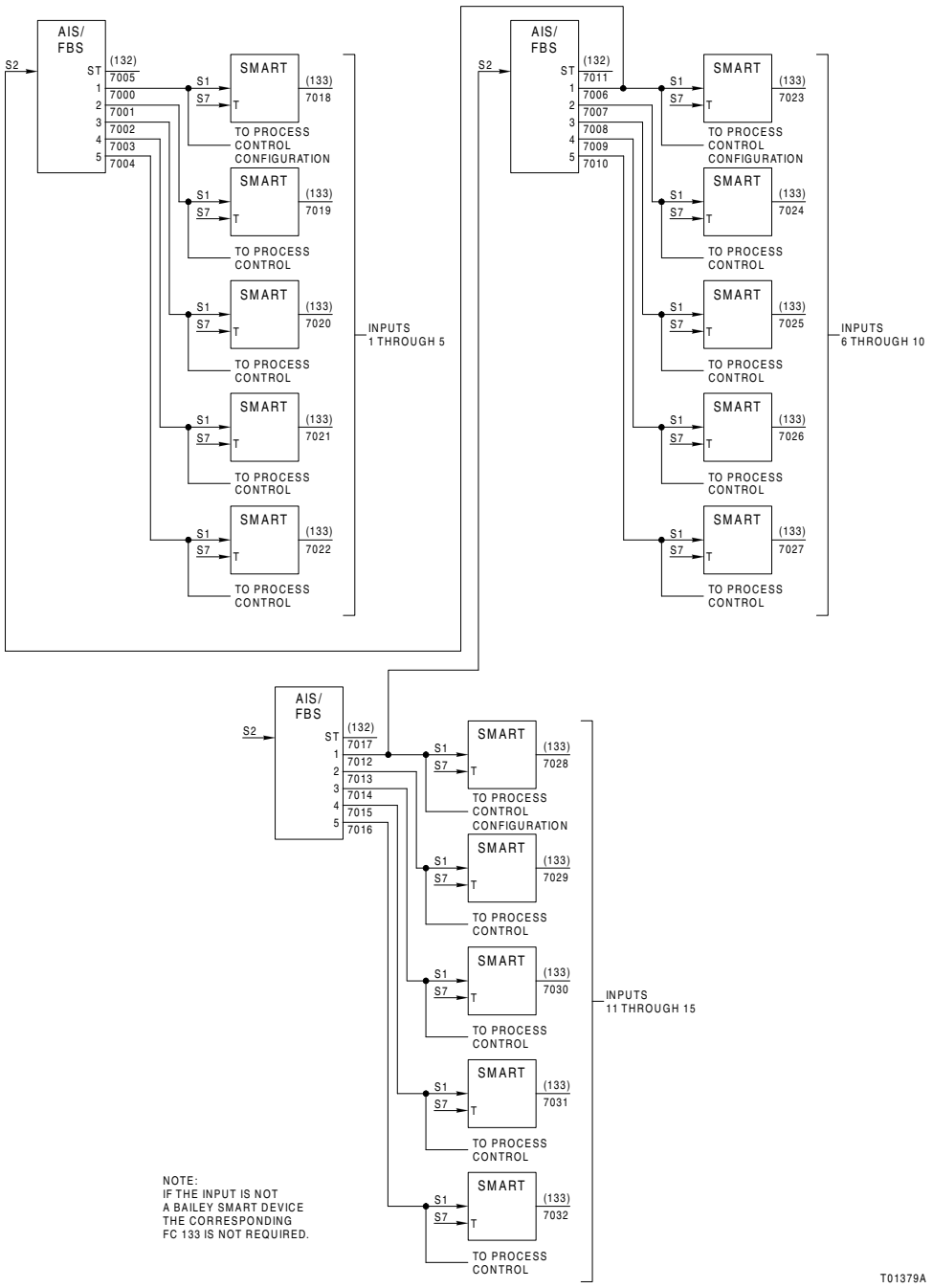
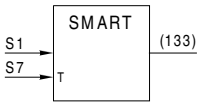


Figure 132-1 Linking Optional Smart Transmitters

133.Smart Field Device Definition

The smart field device definition function code implements ABB FSK smart transmitters, smart positioners, and other smart field devices with defined specifications. The controller verifies the configuration of the field device at startup. Three specifications can be modified while the controller is in execute mode. In execute mode, the controller gets and monitors the value of the process variable defined in this function code.



Outputs

Blk	Type	Description
N	B	Transmitter status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	2	I	Note 1	Block address of field device input
S2	N	0	I	0 - 2000	Field device definition
S3	N	0	I	Full	Engineering units of field device
S4	Y	0.000	R	0.0 - 32.0	Damping time (secs)
S5	N	0	I	0 - 12	Mode definitions
S6	Y	0	I	0 - 6	Operation select: 0 = normal 1 = zero up (uses operation trigger <S7>) 2 = zero down (uses operation trigger <S7>) 3 = span up (uses operation trigger <S7>) 4 = span down (uses operation trigger <S7>) 5 = fix output (fix position of type AVS positioner) 6 = download configuration
S7	N	0	I	Note 1	Block address of operation trigger
S8	Y	0.000	R	Full	Percentage of fixed output/position
S9	Y	0	I	0 - 102	Communication select (all) and temperature compensation definition (TBN only)
S10	N	0	I	0 - 17	Field device output definition
S11	N	0.000	R	Full	Spare
S12	Y	0	I	1 - 255	Field device address or channel number
S13	N	0.000	R	Full	Lower range
S14	Y	0.000	R	Full	Upper range

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

133.1Explanation

133.1.1Specifications

S1

Block address of the input. This input must be linked to the analog input/slave block (function code 132).

S2
Field device definition.
Field Device Type:

000 = unused
100 = PTS transmitter (pressure)
2XX = EQS transmitter (temperature)
3XX = AVS positioner
400 = BCN transmitter (pressure)
5XX = EQN transmitter (temperature)
6XX = TBN 580 transmitter (pH)
700 = TBN 581 transmitter (ORP/pION)
800 = TBN 480 transmitter (conductivity)
9XX = magnetic flowmeter 50XM/SM/XE
10XX = vortex flowmeter 10VT
11XX = vortex flowmeter 50VM
12XX = variable area flowmeter
13XX = mass flowmeter

where:

XX = sensor type

BCN, PTS, TBN 581, and TBN 480
00 = sensor type not applicable

XX = sensor type *(continued)*

Smart EQS and EQN transmitter (temperature)

01 = mV input thermocouple	18 = 3 wire RTD (Ω)
02 = dual RTD (Ω)	19 = 3 wire RTD (100 Ω, 385)
03 = 2 wire RTD (Ω)	20 = 3 wire RTD (100 Ω, 392)
04 = unused	21 = type D thermocouple nonlinear[1]
05 = type B thermocouple	22 = type B thermocouple nonlinear[1]
06 = type E thermocouple	23 = type E thermocouple nonlinear[1]
07 = type J thermocouple	24 = type J thermocouple nonlinear[1]
08 = type K thermocouple	25 = type K thermocouple nonlinear[1]
09 = type R thermocouple	26 = type R thermocouple nonlinear[1]
10 = type S thermocouple	27 = type S thermocouple nonlinear[1]
11 = type T thermocouple	28 = type T thermocouple nonlinear[1]
12 = type C thermocouple	29 = type C thermocouple nonlinear[1]
13 = type N thermocouple	30 = type N thermocouple nonlinear[1]
14 = dual RTD (100 Ω, 385)	31 = 4 wire RTD (100 Ω, 385)[2]
15 = dual RTD (100 Ω, 392)	32 = 4 wire RTD (100 Ω, 392)[2]
16 = 2 wire RTD (100 Ω, 385)	33 = 4 wire RTD (Ω)[2]
17 = 2 wire RTD (100 Ω, 392)	

[1]EQN only; [2]EQS only

Smart TBN 580 transmitter (pH)

40 = standard	42 = $Ir0_2$
41 = antimony	43 = reserved

Smart AVS positioner (digital multidrop mode only):

00 = position feedback (%)	09 = differential pressure
01 = setpoint (%)	10 = output pressure 1
07 = deviation (%)	11 = output pressure 2
08 = supply pressure	

NOTES:

1. Every type AVS positioner connected to the bus must have the first channel be position feedback. Specification S3 through the end are ignored on all channels other than position feedback. The engineering units of position feedback channel dictate the units used to report all pressures. Valid units for these pressures are psi and bars.

2. A maximum of five variables may be configured for any one type AVS positioner or other future multivariable device.

Flowmeters (digital multidrop mode only):
00 = Volume flow or mass flow rate
03 = Process or tube temperature (vortex/50VM or mass)
10 = Totalizer 1
11 = Totalizer 2

3. Every flowmeter connected to the bus must have the first channel be volume flow or mass flow rate. Specification S3 through the end are ignored on all channels other than volume flow or mass flow rate.

S3
Engineering units of the field device. Table 133-1 lists the available engineering units for specification S3.

Table 133-2 Specification S3 Engineering Units

Value	Engineering Units	Field Device Type	Value	Engineering Units	Field Device Type
0	Undefined	Any, TBN480	39	USGal/h	50XM/SM/XE
1	°C	EQS or EQN (temperature)	40	User-defined	50XM/SM/XE
2	°F	EQS or EQN (temperature)	41	l/s	10VT (actual)
3	°K	EQS or EQN (temperature)	42	l/m	10VT (actual)
4	mV	EQS or EQN (temperature) or TBN 581 (ORP/pION)	43	l/h	10VT (actual)
5	Ω	EQN or EQS	44	igps	10VT (actual)
10	%	None	45	igpm	10VT (actual)
11	Reserved	Reserved	46	igph	10VT (actual)
12	Reserved	Reserved	47	igpd	10VT (actual)
13	Reserved	Reserved	48	usgps	10VT (actual)
14	Reserved	Reserved	49	usgpm	10VT (actual)
15	Reserved	Reserved	50	usgph	10VT (actual)
20	Inches H_2O	BCN, PTS (pressure)	51	usmgd	10VT (actual)
21	mm H_2O	BC	52	ft^3/s	10VT (actual)
22	mm of mercury	BCN, PTS (pressure)	53	ft^3/m	10VT (actual)
23	psi	BCN, PTS, AVS (pressure)	54	ft^3/h	10VT (actual)
24	mPascals	BCN, PTS (pressure)	55	ft^3/d	10VT (actual)
25	kPascals	BCN, PTS (pressure)	56	m^3/s	10VT (actual)
26	Bar	BCN, PTS, AVS (pressure)	57	m^3/m	10VT (actual)
27	mBar	BCN, PTS (pressure)	58	m^3/h	10VT (actual)
28	cm H_2O	BCN, PTS (pressure)	59	m^3/d	10VT (actual)
29	kg per square cm	BCN, PTS (pressure)	61	bbl/s	10VT (actual)
30	pH	TBN 580 (pH)	62	bbl/m	10VT (actual)
31	l/s	50XM/SM/XE	63	bbl/h	10VT (actual)
32	l/m	50XM/SM/XE	64	bbl/d	10VT (actual)
33	l/h	50XM/SM/XE	65	g/s	10VT (mass)
34	m^3/s	50XM/SM/XE	66	g/m	10VT (mass)
35	m^3/m	50XM/SM/XE	67	g/h	10VT (mass)
36	m^3/h	50XM/SM/XE	68	kg/s	10VT (mass)
37	USGal/s	50XM/SM/XE	69	kg/m	10VT (mass)
38	USGal/m	50XM/SM/XE	70	kg/h	10VT (mass)

Table 133-2 Specification S3 Engineering Units (Continued)

Value	Engineering Units	Field Device Type	Value	Engineering Units	Field Device Type
71	t/m	10VT (mass)	75	lb/m	10VT (mass)
72	t/h	10VT (mass)	76	lb/h	10VT (mass)
73	t/d	10VT (mass)	77	lb/d	10VT (mass)
74	lb/s	10VT (mass)			

NOTE: TBN 480 devices must be configured for engineering units of 0.

S4
Defines the damping time and is adjustable from 0.0 to 32.0 seconds (0.0 to 5.0 seconds for AVS devices). The damping time is defined as the time required for an analog or digital response to a step input change to reach approximately 62% of its final value.

S5
Mode definition.

X X

Fail Mode (n/a to 50XM/SM/XE)
X0 = fail low
X1 = hold current values (n/a to flowmeters)
X2 = fail high

Initialize Mode
0X = initialize low (50XM/SM/XE: two forward ranges) (10VT: process fluid = gas)
1X = initialize high (50XM/SM/XE: one forward and one reverse range) (10VT: process fluid = liquid)

NOTE: Specification S5 is not applicable to the type AVS positioner. The type AVS positioner must be connected to the actuator such that an open (0 current) input provides the desired failure and initial position (0 or 100%). Specification S5 should always be 00.

S6
Operation select.

0 = normal
1 = zero up (uses operation trigger <S7>)
2 = zero down (uses operation trigger <S7>)
3 = span up (uses operation trigger <S7>)
4 = span down (uses operation trigger <S7>)
5 = fix output (fix position of type AVS positioner)
6 = download configuration

The operation select input allows the modification of the field device calibration. To adjust calibration:

1. Select level to adjust with S6 (S6=1, 2, 3 or 4).
2. Trigger the adjustment with <S7>. Each zero to one transition of <S7> changes the selected operation by approximately 0.025%. This allows adjustment of the 4 mA or 20 mA value when the span is at 0% or 100%.

Operation select can also fix the transmitter output or positioner internal set point at the percentage defined by S8. Setting S6 to five fixes the output. Setting S6 to zero resumes normal operation. The field device receives the downloaded configuration when S6 is set to six. When installing a new field device, this operation forces the field device configuration to match the defined specifications.
For type AVS positioner, zero and span adjusts the 4-20 mA position output option.

S7
Block address of the operation trigger. Each zero to one transition changes the selected operation (S6) by approximately 0.025%.

S8
Value of the fixed output (percentage). Refer to S6.

S9

Temperature compensation definition for TBN transmitters and communication select function for all field devices.

X X X

Temperature compensation algorithm (TBN58x only)
- XX0 = manual
- XX1 = nernstian
- XX2 = auto solution
- XX3 = pure water

Temperature compensation sensor type and units (TBN58x only)
- X0X = none
- X1X = Balco wire and °F
- X2X = Balco wire and °C

Communication select (all field devices)
- 0XX = communication to transmitter enabled (online)
- 1XX = communication to transmitter disabled (offline)

NOTE: Although specification S9 is tunable, tuning it does not affect field device configuration, only communication select.

S10

Transmitter output or positioner input characterization and normal/reverse definition.

X X

Output (common) only valid for EQS is linear and function generator
- X0 = output is linear with respect to input
- X1 = output is the square root of the input
- X2 = $^{3}/_{2}$ power flow mode AVS: square
- X3 = $^{5}/_{2}$ power flow mode AVS: equal percentage
- X4 = function generator
- X5 = volumetric (special tank) PTS (AVS: quick opening)
- X6 = volumetric (flat end tank) (AVS: quick opening)
- X7 = spare

Magflow 50XM/SM/XE settings

Flow direction	Response speed	Digital Filter
X0 = forward/reverse	normal	off
X1 = forward/reverse	normal	on
X2 = forward/reverse	fast	off
X3 = forward/reverse	fast	on
X4 = forward only	normal	off
X5 = forward only	normal	on
X6 = forward only	fast	off
X7 = forward only	fast	on

Vortex 10VT unit density with mass-flow

X0 =g/ml	X4 = kg/m^3
X1 = g/cm^3	X5 = kg/ft^3
X2 = g/l	X6 = kg/ugl
X3 = kg/l	X7 = spare

Action
- 0X = normal acting (TBN 480: diag spike option disabled) (10VT: flow mode = actual flow)
- 1X = reverse acting (TBN 480: diag spike option enabled) (10VT: flow mode = actual flow)

NOTE: Action for flowmeters is flow direction relative to arrow on primary.

S11

Spare.

S12

Field device address. Valid addresses are from one to 15. In point to point mode, this is the channel number the transmitter is connected to on function code 132. When this specification is zero, the engineering unit values stored in corresponding function code 132 blocks are used as field device lower and upper range values. EU zero is used as field device lower range and the difference of the EU span and EU zero is used as field device upper limit.

NOTE: When multiple values from one smart field device are brought into function code 132, specification S12 (the address) of function code 133 should be the same for each variable.

S13
Field device lower range. If S12 is a valid address (one to 15), this specification is compared with the lower range value stored in the field device. This specification value is downloaded to the field device when this specification is tuned. This value is also downloaded to the corresponding field device when the download configuration option is selected (S6).

S14
Field device upper range. This is the upper range value of the configuration. If S12 is a valid field device address (one to 15), this specification is compared with the upper range value stored in the field device. This specification value is downloaded to the field device when this specification is tuned. This value is also downloaded when the download configuration option is selected (S6).

133.2 Application

Figure 132-1 shows the block numbering configuration for analog input/slave (function code 132) linked to smart field device definition (function code 133) function blocks.
Figures 133-1 and 133-1 show how the smart field device definition block could be used in typical applications.

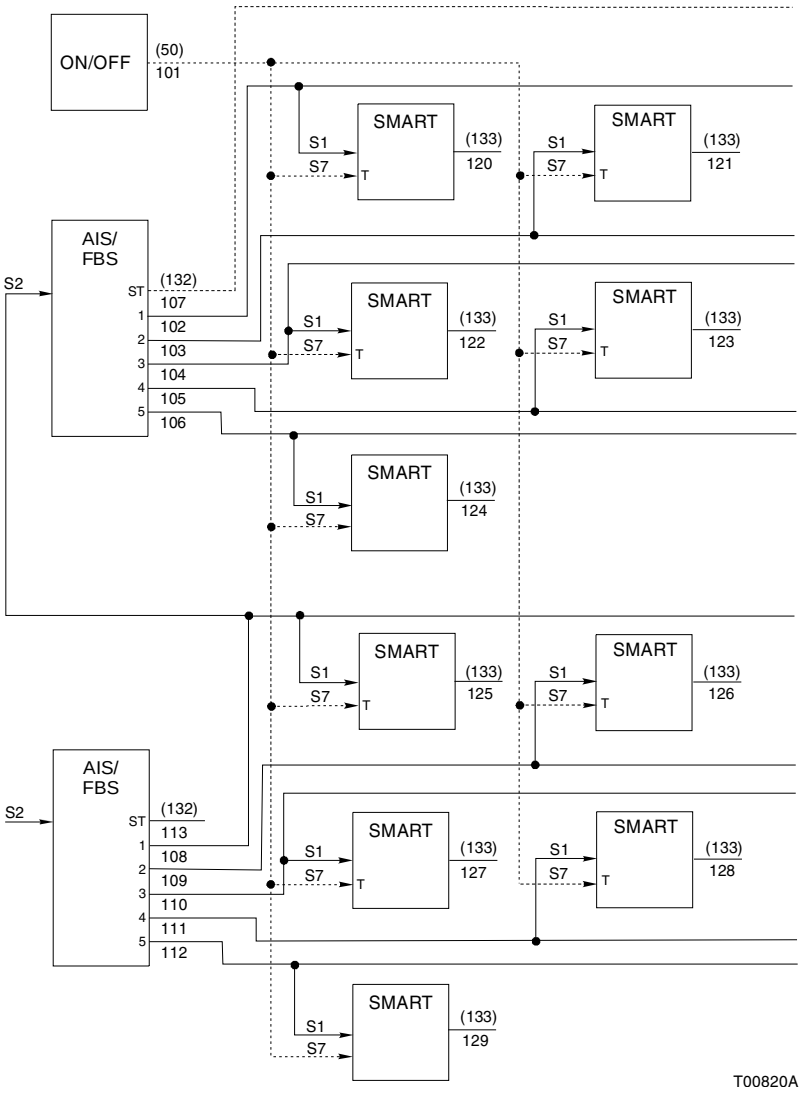


Figure 133-1 FC 132 and FC 133 Example

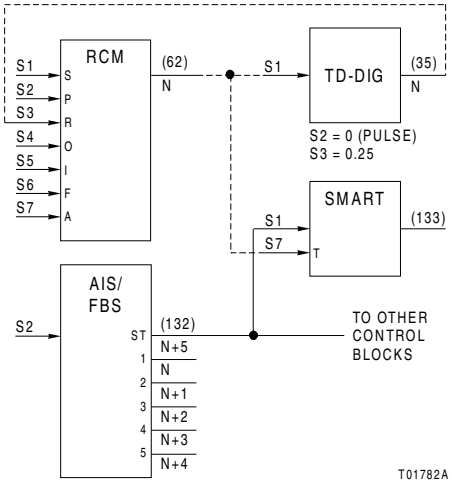


Figure 133-1 Smart Field Device Definition Example

In Figure 133-1, specification S1 of the smart field device definition block defines the input value and the I/O module to utilize for communications. When a tunable specification is changed, the new value is sent to the transmitter by this link. The remote control memory and timer blocks are set up to produce a zero to one transition on the operation trigger input <S7>. Each zero to one transition on <S7> changes the selected operation by 0.025 percent. This configuration allows the fine adjustment of the transmitter 4 mA or 20 mA values.
SMARTPORT requires 33 consecutive block addresses in the controller for each I/O module (up to 15 input devices per I/O module).
The default block address of 33 consecutive blocks for each I/O module is calculated as follows:

Default starting block address = (I/O module address × 33) + 7000

133.3Type AVS Positioner Application

The type AVS positioner is an output device and therefore has unique application abilities. The type AVS positioner device status information is reported through the position feedback block only. The only time the other outputs will go bad quality is if the device is not responding to the FBS module. The type AVS positioner is only supported in full digital field bus mode. When configuring a type AVS positioner, refer to the note on ranges located under specification S6 of function code 132.

Figure 133-1 shows how the smart field device definition block should be used with a type AVS positioner in a typical application.

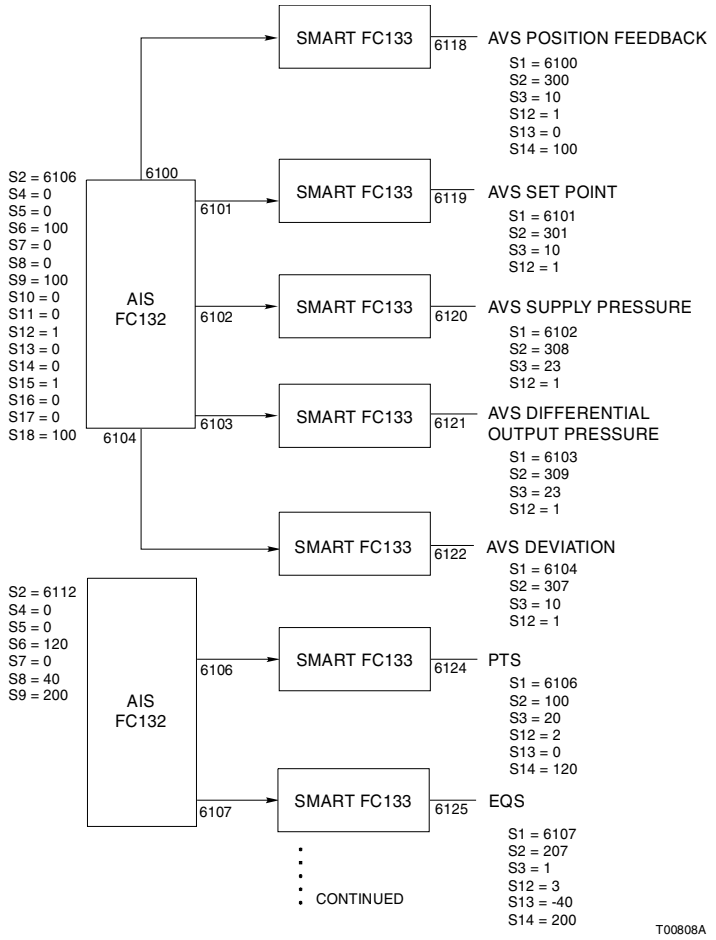
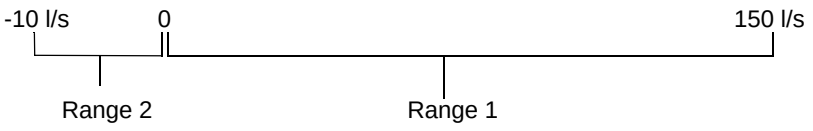


Figure 133-1 Type AVS Positioner Application Example

133.4Flowmeter Application

Figure 133-1 shows how to configure the analog input/slave (function code 132) linked to smart field device definition (function code 133) for the multirange capability of the ABB flowmeters. It requires the use of a digital signal generated by the flowmeter itself, indicating which range is active.
Assume the flowmeter is configured for one forward and one reverse range as follows:



The value of zero is fixed always. The flowmeter indicates which range is being used via a contact output. This must be tied into the DCS via a digital input function block. Refer to the function block algorithm shown in Figure 133-1.

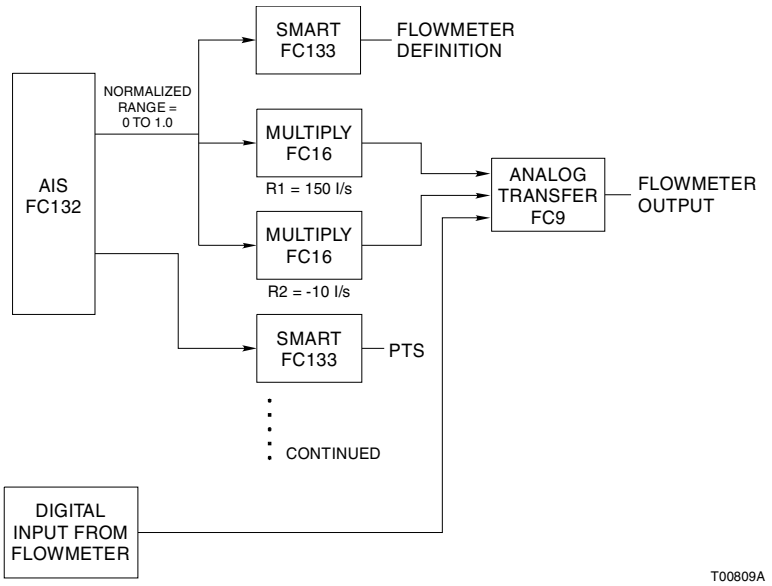
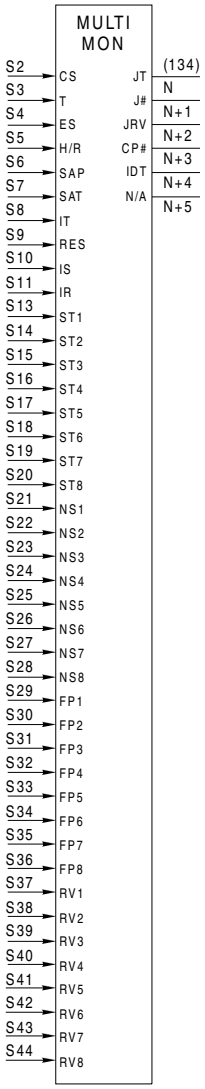


Figure 133-1 Flowmeter Application Example

134.Multi-Sequence Monitor

The multi-sequence monitor controls the execution of a sequence generator by selecting the order of step execution in a process. It operates like the sequence monitor (function code 124), but has enhanced functionality. The multi-sequence monitor block provides multi-batch recipe control, halts sequences and inserts steps. Each multi-sequence monitor block controls the execution of eight phases. A phase is a step that may vary from recipe to recipe. This block provides a means to change the order that predefined steps execute in different recipes. The phases always execute in numerical order from one to eight. Four parameters (step type, normal step number, fault step number, and recipe value) define each phase. The multi-sequence monitor block performs a logical action based on the value of a control status input from a device monitor block, and a boolean step trigger input. Depending on the value of the control status input, the next step can be either a fault step or one determined by internal logic. The multi-sequence monitor block can be placed in either automatic or semi-automatic mode, and includes hold/resume, executed stop (E-STOP), insert step and insert phase inputs. Each multi-sequence monitor block handles up to eight phases. If more than eight phases are required, multi-sequence monitor blocks can be linked together in series fashion with S1.



Outputs

Blk	Type	Description
N	R	Jump step number
N+1	B	Jump step trigger
N+2	R	Jump step recipe value
N+3	R	Current phase number
N+4	B	Insert step done
N+5	B	Run/hold state: 0 = run 1 = hold

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block number of next multi-sequence monitor block associated with this sequence of steps (0 = no more blocks)
S2	N	5	I	Note 1	Block address of control status input
S3	N	0	I	Note 1	Block address of step trigger
S4	N	0	I	Note 1	Block address of E-STOP
S5	N	0	I	Note 1	Block address of hold/resume input
S6	N	1	I	Note 1	Block address of auto permissive: 0 = manual 1 = auto
S7	N	0	I	Note 1	Block address of auto trigger
S8	N	0	I	Note 1	Block address of insert trigger
S9	N	5	I	Note 1	Block address of resume phase number
S10	N	5	I	Note 1	Block address of insert step number
S11	N	5	I	Note 1	Block address of insert recipe value
S12	N	0.000	R	Full	Reserved
S13	N	5	I	Note 1	Block address of phase 1 step type

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S14	N	5	I	Note 1	Block address of phase 2 step type
S15	N	5	I	Note 1	Block address of phase 3 step type
S16	N	5	I	Note 1	Block address of phase 4 step type
S17	N	5	I	Note 1	Block address of phase 5 step type
S18	N	5	I	Note 1	Block address of phase 6 step type
S19	N	5	I	Note 1	Block address of phase 7 step type
S20	N	5	I	Note 1	Block address of phase 8 step type
S21	N	5	I	Note 1	Block address of phase 1 normal step
S22	N	5	I	Note 1	Block address of phase 2 normal step
S23	N	5	I	Note 1	Block address of phase 3 normal step
S24	N	5	I	Note 1	Block address of phase 4 normal step
S25	N	5	I	Note 1	Block address of phase 5 normal step
S26	N	5	I	Note 1	Block address of phase 6 normal step
S27	N	5	I	Note 1	Block address of phase 7 normal step
S28	N	5	I	Note 1	Block address of phase 8 normal step
S29	N	5	I	Note 1	Block address of phase 1 fault phase
S30	N	5	I	Note 1	Block address of phase 2 fault phase
S31	N	5	I	Note 1	Block address of phase 3 fault phase
S32	N	5	I	Note 1	Block address of phase 4 fault phase
S33	N	5	I	Note 1	Block address of phase 5 fault phase
S34	N	5	I	Note 1	Block address of phase 6 fault phase
S35	N	5	I	Note 1	Block address of phase 7 fault phase
S36	N	5	I	Note 1	Block address of phase 8 fault phase
S37	N	5	I	Note 1	Block address of phase 1 recipe value
S38	N	5	I	Note 1	Block address of phase 2 recipe value
S39	N	5	I	Note 1	Block address of phase 3 recipe value
S40	N	5	I	Note 1	Block address of phase 4 recipe value
S41	N	5	I	Note 1	Block address of phase 5 recipe value
S42	N	5	I	Note 1	Block address of phase 6 recipe value
S43	N	5	I	Note 1	Block address of phase 7 recipe value
S44	N	5	I	Note 1	Block address of phase 8 recipe value

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

134.1Explanation

Each multi-sequence monitor block executes eight phases in numerical order. The step number executed in each phase comes from a different function block. The step type, fault phase, and recipe value for each phase also come from other function blocks. Since all the values are specified external to the multi-sequence monitor block, this arrangement enables the operator to vary those values either manually or through logic (for example, recipe table blocks). Thus, the multi-sequence monitor block can control the execution of many different sequences, allowing the operator to control several consecutive sequences with one block. Figure 134-1 shows a common configuration using a multi-sequence monitor block

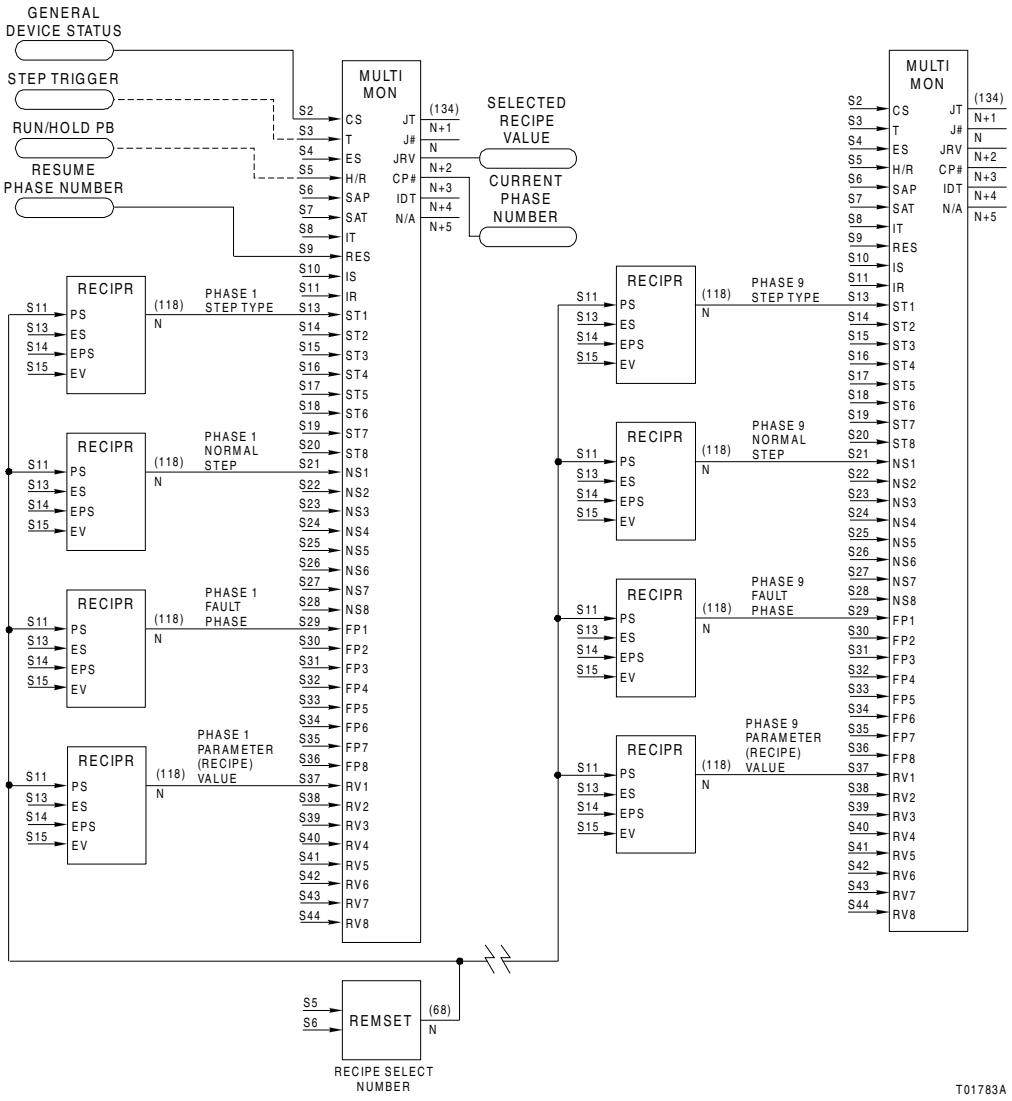


Figure 134-1 Multiple Sequence and Auxiliary Logic Connections

to control several sequences using recipe table blocks.

The multi-sequence monitor block uses the values of two inputs to determine the next step number in a batch process. The first is the control status input, which defines the current state of the devices controlled by the multi-sequence monitor block. This input can be 0.0 (good), 1.0 (bad), or 2.0 (waiting). This value determines whether the next step will be a normal step or a fault step. When the control status input is 1.0 (bad), a fault step initiates unless control is overridden.

The second input is the step trigger. The step trigger is dependent on the current step and the results of auxiliary logic associated with the device in question. Each step of a batch sequence often requires auxiliary logic to perform functions in addition to controlling the device (i.e., change controller set points, totalize flows, etc.). This auxiliary logic ties into the batch execution configuration. Execution of the auxiliary logic associated with the current step number initiates when the current step number is read from the sequence generator block. The current step number also selects from the results of that auxiliary logic the boolean signal output to the multi-sequence monitor block as the step trigger.

The values of the step trigger and the control status input are then tested against the step type to determine whether the next step can be executed. The step type defines the values of the step trigger and the control status that must exist for the sequence to proceed to the next step. The step type also defines whether semi-automatic control is permitted (operator intervention).

To initiate the multi-sequence monitor operation, configure the resume phase (S9) as some number other than zero, and place the sequence in hold first and run later.
The multi-sequence monitor block can operate in automatic or semi-automatic mode. In the automatic mode, the sequence is dependent on the values of the control status input and the step trigger. In the semi-automatic mode, the sequence is dependent on the values of the control status input, the step trigger, and the semi-automatic step trigger. The operator must (as one of the conditions to proceed to the next step) activate the semi-automatic step trigger manually to proceed with the sequence. The block can be placed in semi-automatic mode only if both the semi-automatic permissive and step type specifications are configured to permit it.
Execution of a phase in a multi-sequence monitor block can be halted when the step type and hold/resume specifications permit. When a phase halts, a step can be inserted into the phase sequence. On a zero to one transition of the insert trigger, the insert step number and the insert recipe value are output with a jump step trigger. The insert step done specification toggles from a zero to one when the insert trigger goes to zero and the step trigger makes a zero to one transition. If the hold/resume specification goes to zero (resume) during an insert step, the sequence remains in hold until the insert step is complete. When the insert step is complete, the insert step done output goes from zero to one and the sequence resumes at the resume phase number.
The E-STOP specification drives the sequence generator block and multi-sequence monitor block to the reset step when it is one. This is normally a remote control memory block set up as an E-STOP.

134.1.1 Specifications

S1 – NXT
Block address of next multi-sequence monitor block in the series of blocks used to execute the sequence. If this value equals zero there are no more blocks in the sequence. Each multi-sequence monitor block can accommodate eight phases. If the sequence contains more than eight steps, blocks can be linked in a series with the last phase of the first block initializing the first phase of the second block, etc. Therefore, phases one through eight in the second multi-sequence monitor block are phases nine through 16 in the sequence.

S2 – CSI
Block address of control status input. The value in this block represents the control status of the devices used in the control loop. The output of this block is good when all inputs are good, bad when any one input is bad, and waiting if any one input is waiting for a reply from a device driver or device monitor block. This value and the value of the step trigger select the next step in the sequence. When the control status input is bad, the fault step is automatically output. If the control status input is good or waiting, output depends on the step type for the current phase.

0.0 = good
1.0 = bad
2.0 = waiting

S3 – STEP
Block address of the step trigger. The value in this block, with the control status input, selects the next step in the control sequence. The step trigger generates by auxiliary logic used for the batch process. The step trigger input can either be one or zero. The state acted on depends on the step type for the current step.

S4 – E-STOP
Block address of E-STOP input. The E-STOP specification drives the block to step zero (the reset step) whenever it has a value of one. The block referenced by this specification is normally a remote control memory block set up as an E-STOP.

S5 – HOLD
Block address of hold/resume input. The multi-sequence monitor block can hold the sequence at certain steps. The hold/resume function is active only when the step type is configured to permit it (tens digit is zero). This specification enables the holding of the sequence or insertion of a step into the sequence. If the sequence holds at a step and the insert trigger makes a zero to one transition, then a step will be inserted into the sequence. Otherwise, the sequence holds until the hold/resume input makes a one to zero transition. The block then resumes operation at the phase specified with S9, the resume phase number.

0 = no hold

1 = hold the sequence at the current step

1 to 0 transition = resume operation at the phase specified by <S9>

S6 – PERM
Block address of the semi-automatic permissive. When the value in this block is zero, the semi-automatic mode is permitted unless <S9> overrules it.

0 = both automatic and manual modes permitted
1 = only automatic mode permitted

S7 – SEMI
Block address of the semi-automatic trigger. When the semi-auto mode is permitted by step type and the semi-auto permissive <S6> is zero, the operator must change this value from zero to one to initiate the next step of the sequence.

S8 – INTRG
Block address of insert trigger. The value in this block initiates the insertion of a step into a sequence. If the hold/resume input <S5> is at hold and this input makes a zero to one transition, the insert step number <S10> and insert recipe value <S11> will be output from the block. This function is applicable only when the hold/resume input is permitted.

S9 – RES
Block address of resume phase number. When the hold/resume input makes a one to zero transition, the block resumes operation at the phase specified here.

S10 – INSTP
Block address of insert step number. The step identified here will be inserted in the sequence on a zero to one transition of the insert trigger <S8>, when the hold/resume input <S5> is at hold.

S11 – INRCP
Block address of insert recipe value. The recipe value identified here is inserted in the sequence on a zero to one transition of the insert trigger <S8>, when the hold/resume input <S5> is at hold.

S12
Reserved.

S13 – TYPE1
(Block address of the step type for Step 1) Each step type is made up of two digits. The ones digit defines the state of the control status input and step trigger necessary for the block to initiate the next step of the sequence. The tens digit identifies the control options available for the step.

X X

Input states required to advance to next step
X0 = advance when control input = 0.0 and step trigger is (1)
X1 = advance when control input = 0.0
X2 = advance when step trigger makes zero to one transition

Options permitted for this step
0X = permit hold/resume <S4> and semi-auto <S6>
1X = permit semi-auto <S5>
2X = permit no option (auto control only)

S14 to S20 –
TYPE2 to TYPE8
Block addresses of the step types for steps two through eight. Refer to ***S13 – TYPE1*** for definitions.

S21 to S28 –
STEP1 to STEP8
Block addresses of the phase *n* normal steps. These specifications identify the block containing the number of the step the system should execute when it is in phase *n* (*n* equals one to eight) when the block is operating normally.

S29 to S36 –
FAULT1 to FAULT8
Block addresses of the phase *n* fault phases. These specifications identify the block containing the number of the phase the system should execute after phase *n* (*n* equals one to eight) when the block receives a bad input from the device monitor block.

S37 to S44 –
REC1 to REC8
Block addresses of the phase *n* recipe values. These specifications identify the block containing the recipe value used to implement phase *n* (*n* equals one to eight).

134.1.2Outputs

N
(Jump step number) Identifies the step that executes after the current step completes.

N+1
(Jump step trigger) A zero to one transition of this output initiates a jump step in the sequence generator block. The sequence generator block then executes the step identified by the jump step number. A zero to one transition of this value occurs when the multi-sequence monitor block proceeds to the next phase in the sequence (when the conditions defined by the step type are met).

N+2
(Jump step recipe value) Recipe value defined for the phase that is to follow the current phase. If the block is held and a step is inserted, this value reflects the insert recipe value <S11>.

N+3
(Current phase number) Phase number of the phase being executed. There are eight phases numbered one to eight. Although the step number in each phase can vary from sequence to sequence, the phases always execute in ascending numerical order.

N+4
(Insert step done) Zero when an inserted step is being executed. It will make a zero to one transition when the inserted step is finished. This triggers the block to resume execution with the phase selected in <S9>.

0 = a step has been inserted in the sequence and is currently running

1 = the sequence is proceeding normally with no inserted steps at this point

0 to 1 transition = the inserted step has been executed and the block is resuming normal operation

NOTE: An inserted step sequence is complete (done) when the insert step trigger goes to zero and the step trigger makes a zero to one transition.

N+5
(Run/hold state) Shows when the multi-sequence monitor block is in the hold mode.

1 = hold
0 = run

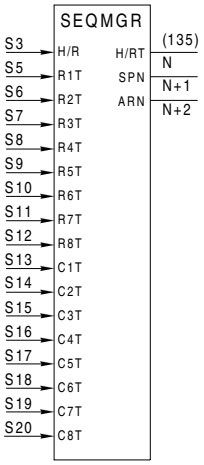
134.2Applications

Figure 134-1 shows a multi-sequence monitor block used to control a number of sequences through real recipe table blocks and a remote manual set constant block. The operator uses the remote manual set constant block to select the values that will be output from the real recipe table blocks. For example, if the operator selects zero, then the first parameter value defined in each real recipe table block will be output from the blocks. The real recipe table blocks are the normal steps for phases one through eight in the multi-sequence monitor block (S21 through S28). By ganging recipe table and multi-sequence monitor blocks, any number of sequences with any number of steps can be executed. The operator controls which sequence is executed with the remote manual set constant block. To select different sequences, enter different numbers.

135.Sequence Manager

The sequence manager function code controls the access to a subsequence from several main sequences. It identifies the step in a sequence monitor block or the phase in a multi-sequence monitor block that the other sequences access. Each sequence manager block can accommodate eight requests. For more than eight inserted steps in a sequence, link several sequence monitor blocks with S1. The block defines the manager type as ***first in first out*** (FIFO), or priority (lowest active request processed first) basis and executes requests accordingly. Requests become active on a zero to one transition of the request trigger.

NOTE: If multiple FC135s are linked together to be loaded into a IMMFP01/IMMFP02 controller, they must be linked in ascending block order to prevent a configuration error.



Outputs

Blk	Type	Description
N	B	Hold/reset trigger: 0 = reset 1 = hold
N+1	R	Starting phase number
N+2	R	Active request number

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block number of next sequence manager block
S2	N	0	I	0 or 1	Manager type: 0 = FIFO 1 = lowest active request number first
S3	N	0	I	Note 1	Block address of hold/reset input: 0 = reset 1 = hold
S4	N	0	I	Note 1	Hold output status for no active request: 0 = do not hold 1 = put sequence monitor into hold/reset (step 0)
S5	N	0	I	Note 1	Block address of request no. 1 trigger
S6	N	0	I	Note 1	Block address of request no. 2 trigger
S7	N	0	I	Note 1	Block address of request no. 3 trigger
S8	N	0	I	Note 1	Block address of request no. 4 trigger
S9	N	0	I	Note 1	Block address of request no. 5 trigger
S10	N	0	I	Note 1	Block address of request no. 6 trigger
S11	N	0	I	Note 1	Block address of request no. 7 trigger
S12	N	0	I	Note 1	Block address of request no. 8 trigger
S13	N	0	I	Note 1	Block address of no. 1 complete trigger
S14	N	0	I	Note 1	Block address of no. 2 complete trigger
S15	N	0	I	Note 1	Block address of no. 3 complete trigger
S16	N	0	I	Note 1	Block address of no. 4 complete trigger
S17	N	0	I	Note 1	Block address of no. 5 complete trigger

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S18	N	0	I	Note 1	Block address of no. 6 complete trigger
S19	N	0	I	Note 1	Block address of no. 7 complete trigger
S20	N	0	I	Note 1	Block address of no. 8 complete trigger
S21	Y	0.000	R	Full	Request no. 1 starting phase number
S22	Y	0.000	R	Full	Request no. 2 starting phase number
S23	Y	0.000	R	Full	Request no. 3 starting phase number
S24	Y	0.000	R	Full	Request no. 4 starting phase number
S25	Y	0.000	R	Full	Request no. 5 starting phase number
S26	Y	0.000	R	Full	Request no. 6 starting phase number
S27	Y	0.000	R	Full	Request no. 7 starting phase number
S28	Y	0.000	R	Full	Request no. 8 starting phase number
S29	N	0	I	Full	Spare
S30	N	0	I	Full	Spare
S31	Y	0.000	R	Full	Spare
S32	Y	0.000	R	Full	Spare

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

135.1Explanation

In batch processes, it is necessary to control both the primary reactors and peripheral equipment. A sequence that controls a reactor is a main sequence. One that controls peripheral equipment such as a header common to several reactors or a cooling system common to several units is a subsequence. Several main sequences use subsequences at different times in their operation. The sequence manager block manages requests to a single subsequence from several main sequences in an orderly and predictable manner.

Each sequence manager block can accommodate eight requests for a single subsequence. If more than eight main sequences need access to a subsequence, the sequence manager blocks can be ganged to provide as many as necessary.

The sequence manager block executes requests based on the manager type, and the values of the request and completion triggers for each of the eight requests.

Requests can be managed in two ways: ***first in, first out*** (FIFO) and on a priority basis (lowest active request processed first).

A request becomes active when its request trigger makes a zero to one transition. The sequence manager block chooses a request from all of the active ones based on the manager type. For example, if the manager is set to priority, and requests three and seven are active, request three will be processed first even if request seven was generated first.

The sequence manager will not process the next active request until the completion trigger for the current request makes a zero to one transition, signaling the step has been completed.

Figure 135-1 shows a configuration with the connection between the sequence monitor block for the main sequence, the sequence manager block, and the sequence monitor block for the subsequence. Either or both of the sequence monitor blocks can be a multi-sequence monitor block.

Each of the eight requests has a starting phase number. This number defines the phase (or step) in the subsequence that executes when the request activates. Each request can ask for a different step from the subsequence, or all requests can ask for the same one.

135.1.1 Specifications

S1 – NXT
(Block address of next sequence manager block) If more than eight main sequences need to acquire the same

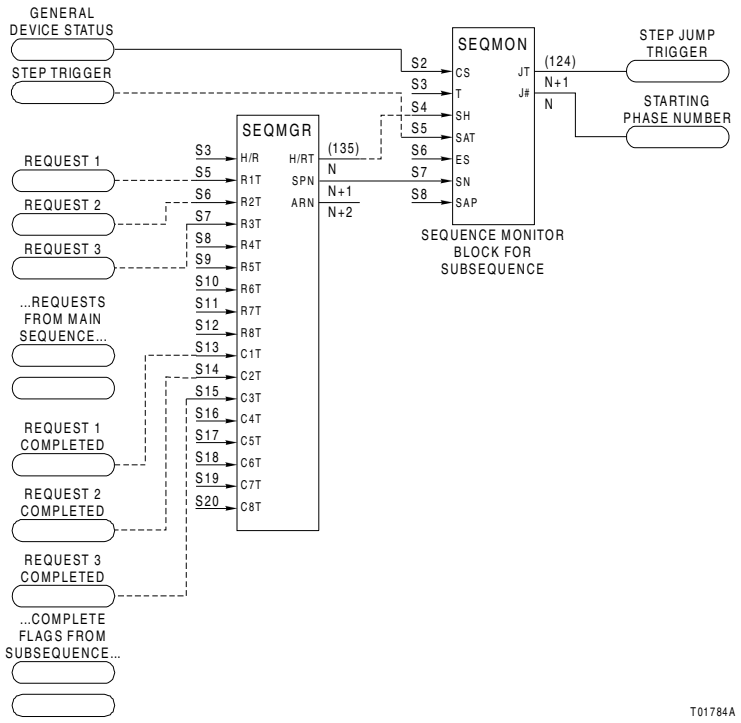


Figure 135-1 Sample Configuration Using Sequence Manager Block

subsequence, the sequence manager blocks can be ganged to enable any number of main sequences to access a subsequence. If this value equals zero, there are no more blocks in the sequence. Only the outputs from the first sequence manager block in the series will be connected to the sequence monitor or multi-sequence monitor controlling the subsequence.

S2 – TYPE
(Manager type) The sequence manager block executes the requests in one of two ways: first in first out, and priority. In the priority mode, the block always executes the lowest numbered request first, even if higher numbered requests were generated before it.

0 = first in first out (FIFO)
1 = priority (lowest number first)

S3 – HOLD
(Block address of hold/reset input) The hold/reset input can halt operation of the sequence manager block temporarily. If the hold/reset input goes to a one while a step is being executed, the block will finish executing the step, but it will not fill the next active request. It will not allow any more requests to be processed until the input goes to a zero. On a one to zero transition of the input, the sequence manager resets. This clears the FIFO stack and allows the sequence manager to begin again as if a power up had occurred.

0 = reset
1 = hold

S4
Hold output status for no active request.

0 = do not put sequence monitor into hold when there is no active request.

1 = put sequence monitor into hold when there is no active request. Hold means that the sequence manager outputs a zero value and trigger.

S5 to S12 –
RTRIG1 to RTRIG8
(Block address of request triggers one through eight) When a request trigger makes a zero to one transition, the request activates. Only active requests execute. The order of execution depends on the manager type (S2). If the request goes from zero to one without the request complete trigger being one, then the request is still waiting to be processed. If the request stays in the one state even after the completion trigger goes to one, a new request will be generated. The completed request will be skipped.

S13 to S20 –
CTRIG1 to CTRIG8
(Block address of completion triggers one through eight) The completion trigger makes a zero to one transition upon a completed requested step. When the completion trigger goes to one, the block is free to answer the next request.

S21 to S28 –
START1 to START8
(Starting phase number for requests one through eight) Identify the requested phase or step in the subsequence. All the requests can choose the same step or select different steps.

135.1.2Outputs

N
(Hold/reset trigger) Drives the hold/reset trigger of the sequence (or multi-sequence) monitor associated with the sequence manager.

0 = reset
1 = hold

N+1
(Starting phase number) Value of the subsequence phase (or step) that is being executed.

N+2
(Active request number) Number of the request that is being executed.

135.2Applications

The two types of control the sequence manager uses are parallel processing and common element control.
Parallel processing describes a situation in which several events must occur simultaneously. A common example is adding several components to a reactor simultaneously. The advantage of parallel addition over sequential addition is that parallel addition will maximize the throughput of the unit. A disadvantage of parallel addition is that it requires separate pipe runs, valves and flow meters for each component. This increases equipment costs.
Common element control is used when several different main sequences use the same equipment. Examples of this are common headers, pumps, cooling systems, etc. A problem inherent in common element control is prioritizing requests. In some cases, there is no priority, so requests are processed on a FIFO basis. In other cases, certain main sequences require access to the common elements more urgently than other main sequences. In this case, requests for common elements are handled on a priority basis, with the lowest numbered active request executed first. The sequence manager block provides a choice between these two types of control.
Figure 135-1 shows a control situation utilizing several common headers for a series of reactors. Reactor K1 must be simultaneously filled with components A and B in parallel through common headers. Components A and B do not share the same piping but the headers service more than one reactor. This example will illustrate both parallel addition and the use of

common headers. Logically, the process is subdivided into several sequences. Two main sequences control reactors K1 and K2. There is a subsequence for each common header. A sequence manager block controls each subsequence.

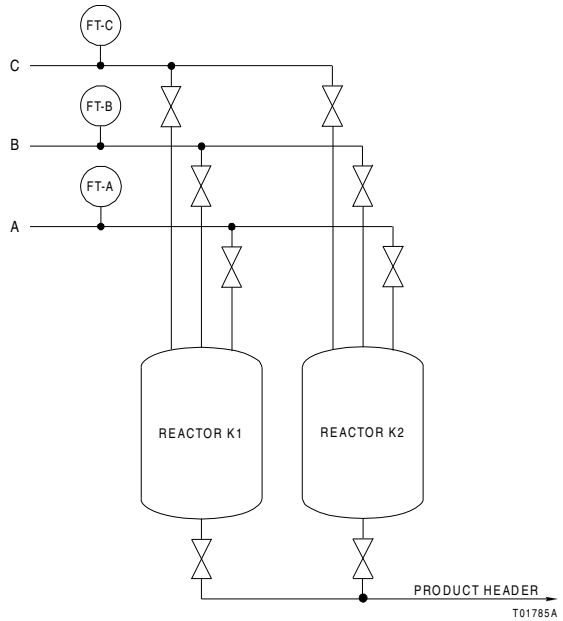


Figure 135-1 Parallel Addition through Common Headers

In this example, Step 4 of the main sequence for reactor K1 adds components A and B in parallel. Figure 135-1 shows the logic in the main sequence to perform this operation. This logic generates requests to add various components to K1, but it does not insure that the headers are both available. With this logic, it is not guaranteed that A and B will be added in parallel to reactor K1, since one of the headers could be in operation with reactor K2.

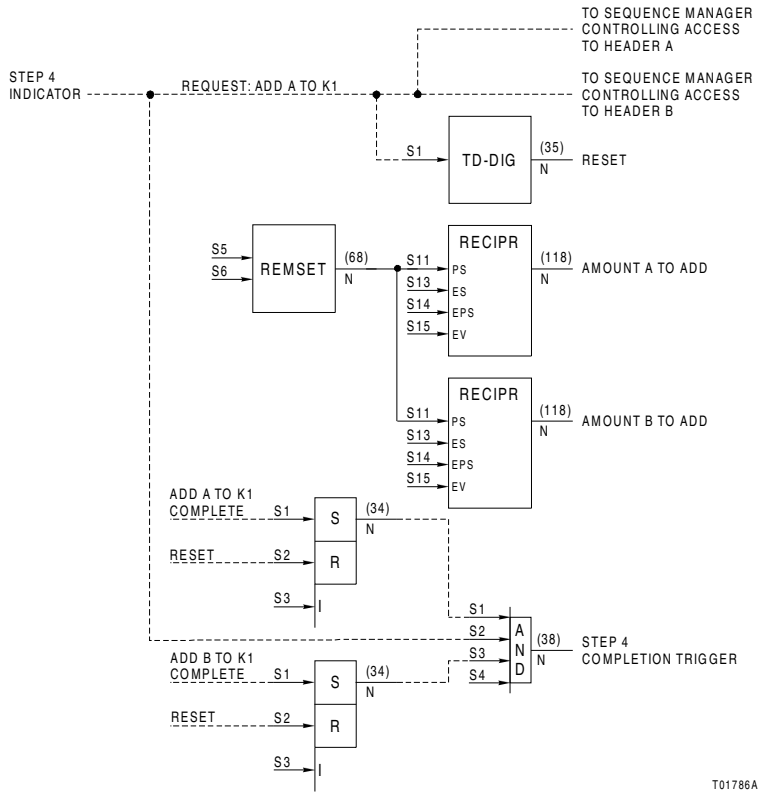


Figure 135-1 Parallel Addition of A and B

Figure 135-1 shows logic that checks to make sure that the sequence managers for both reactors are inactive. The logic checks the active request number for each reactor. If the active request number is zero, then the header is available. In this case, requests to add A and B will be processed only if both headers are available.

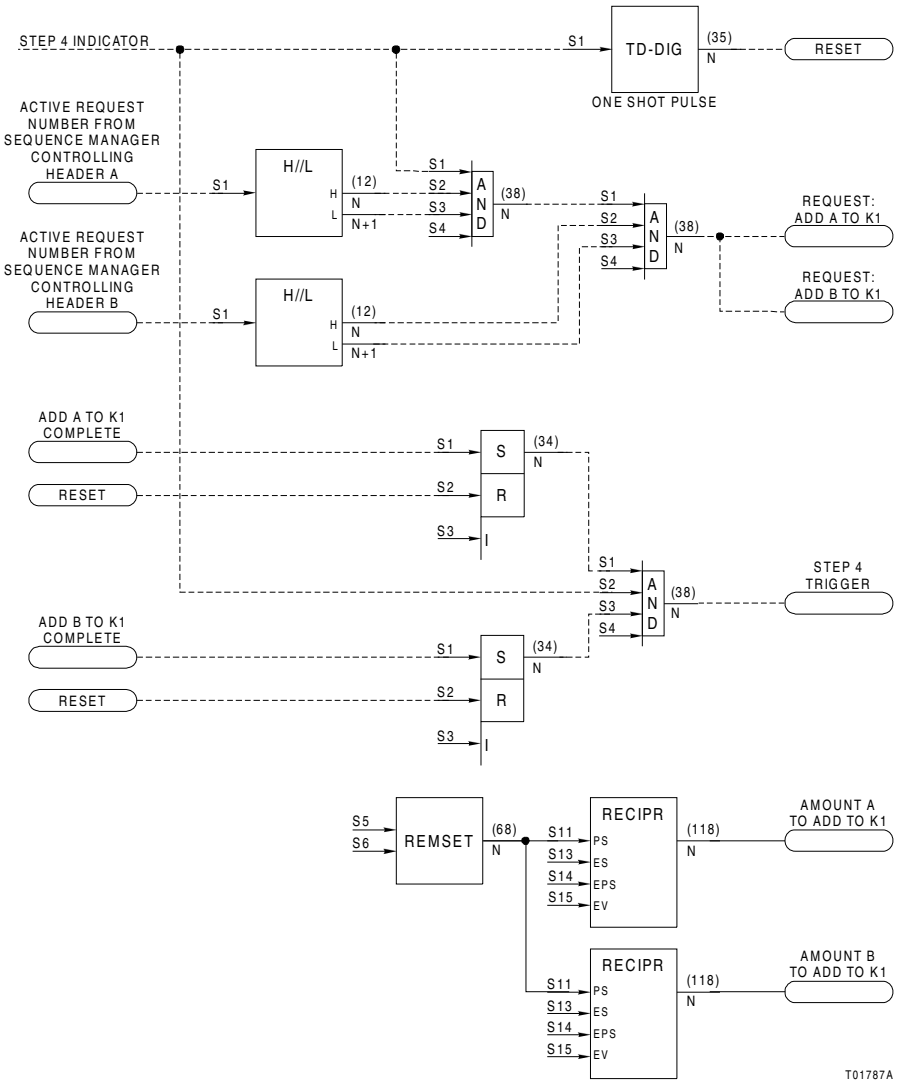


Figure 135-1 Simultaneous Addition of A and B

Figure 135-1 shows the logic used to control common header A (or B). Outputs from the sequence manager block control

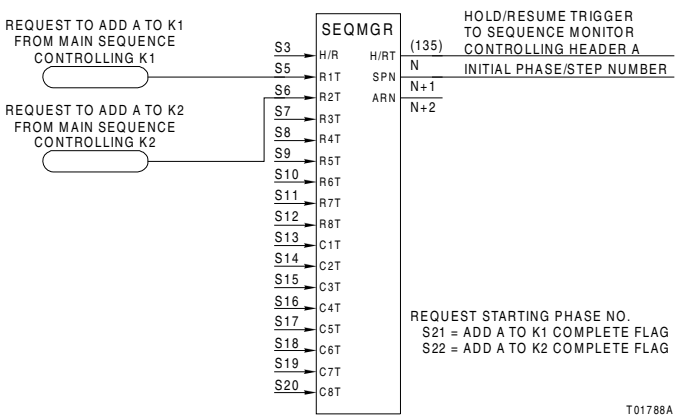


Figure 135-1 Control Common Header A

the sequence monitor block that controls and monitors the subsequence for the header. A hold/resume trigger puts the

sequence monitor block in the hold mode. A new number is loaded into the sequence monitor block as the initial step from the sequence manager block. The sequence manager then outputs a zero to the hold/resume trigger of the sequence monitor. This causes the sequence monitor block to go into run mode and begin execution with the step number selected with the sequence manager.
The sequence manager block selects Step 5 and is the step in the subsequence that adds component A to reactor K1. Figure 135-1 shows this logic. The amount of A added to the tank is integrated until it is greater than the amount called for, then the Step 5 completion trigger energizes. This causes the sequence to execute Step 6. Step 6, also shown in Figure 135-1, energizes the add A to K1 completion flag, which feeds back to the sequence manager (Figure 135-1), and to the main sequence logic (Figure 135-1).

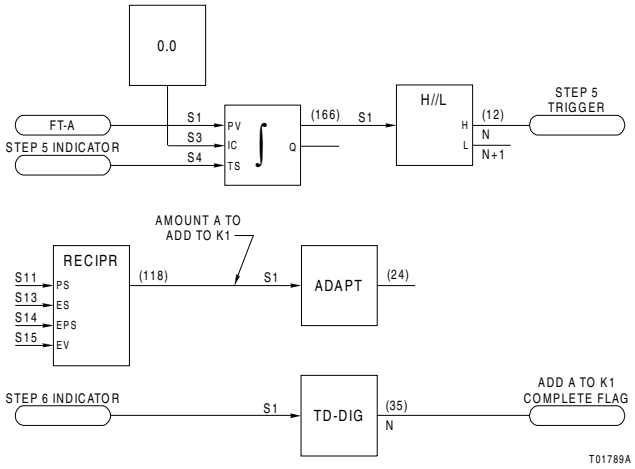


Figure 135-1 Logic to Add A to K1 through Common Header

In this application, all logic resides within a single module. Logic does not have to reside in a single module. The main sequence can be in one module, and a subsequence in another. When signals go between modules (on a polled basis), there could be several scans performed on one module before the data is received from other modules. In the case where a subsequence resides in another module, it is not much of a problem. However, if two requests for a common header come in, then the completion flag for one subsequence may be on for only one scan. With two requests for header A, the add A to K1 completion flag will be on for only one scan. If another module must see the flag in order for the sequence to continue, a timer block must be placed in the logic to insure that the flag stays on long enough to pass through the communications highway.
Figure 135-1 shows an application requiring request prioritizing. Four chemical reactors are fed through common header D. In this example, each reactor is making a different product, and these products have different levels of profitability. The priority is:

K3 > K4 > K5 > K6

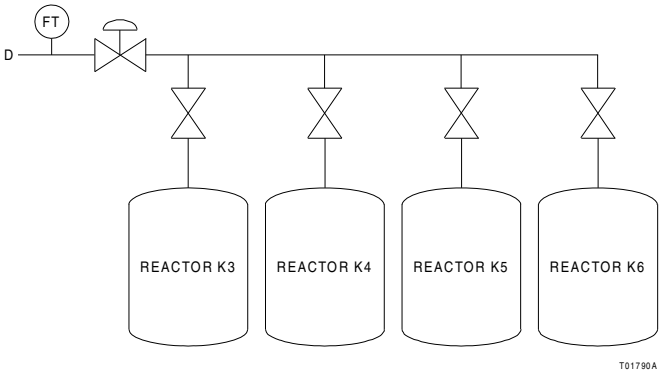


Figure 135-1 Request Prioritizing Example

Requests for header D should be prioritized to process requests from reactor K3, then K4, then K5, then K6. The sequence manager block will process requests on the basis of the lowest request number S2 equals one. Each request number is identified via a block address that is read into S5 through S12. The request for header D from K3 is identified in S5, the request from K4 is identified in S6, etc. With this arrangement, if two requests are received while the subsequence is

running a previous request, the request with the lowest request number is processed next. Figure 135-1 illustrates the logic required to implement the scheme.

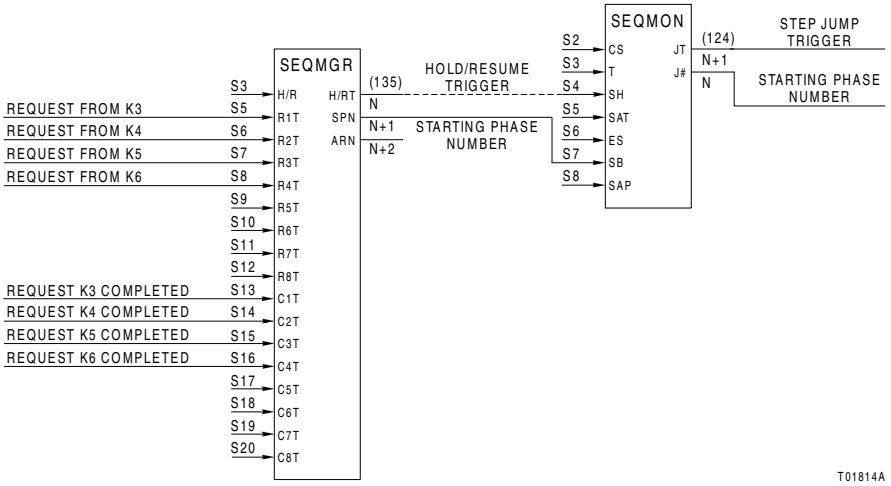
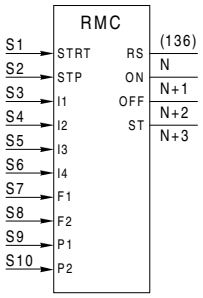


Figure 135-1 Request Priority Logic

136.Remote Motor Control

The remote motor control (RMC) function code has two basic functions. First, it performs the logic necessary to control a digital output. Second, it communicates the result of that logic to the human system interface (HSI).



Outputs

Blk	Type	Description
N	B	Run state: 0 = stopped 1 = running
N+1	B	Pulse on
N+2	B	Pulse off
N+3	B	Control output status: 0.0 = good 1.0 = alarm 2.0 = waiting for feedback

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of start
S2	N	0	I	Note 1	Block address of stop
S3	N	1	I	Note 1	Block address of interlock 1
S4	N	1	I	Note 1	Block address of interlock 2
S5	N	1	I	Note 1	Block address of interlock 3
S6	N	1	I	Note 1	Block address of interlock 4
S7	N	0	I	Note 1	Block address of feedback 1
S8	N	0	I	Note 1	Block address of feedback 2
S9	N	1	I	Note 1	Block address of start permissive 1
S10	N	1	I	Note 1	Block address of start permissive 2
S11	Y	0.000	R	Full	Startup feedback time (secs)
S12	Y	0.000	R	Full	Pulse on time (secs)
S13	Y	0.000	R	Full	Pulse off time (secs)
S14	N	0	I	Full	Display type
S15	N	0	I	Full	Spare
S16	N	0.000	R	0.0, 1.0 or 2.0	Status control: 0.0 = hold status disabled 1.0 = hold status enabled 2.0 = hold status enabled (reset status on exit)
S17	N	0.000	R	Full	Shutdown feedback time (shutdowns use the startup feedback time when set to 0.0)

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

136.1Explanation

Figure 136-1 shows the basic logic of a remote motor control block.

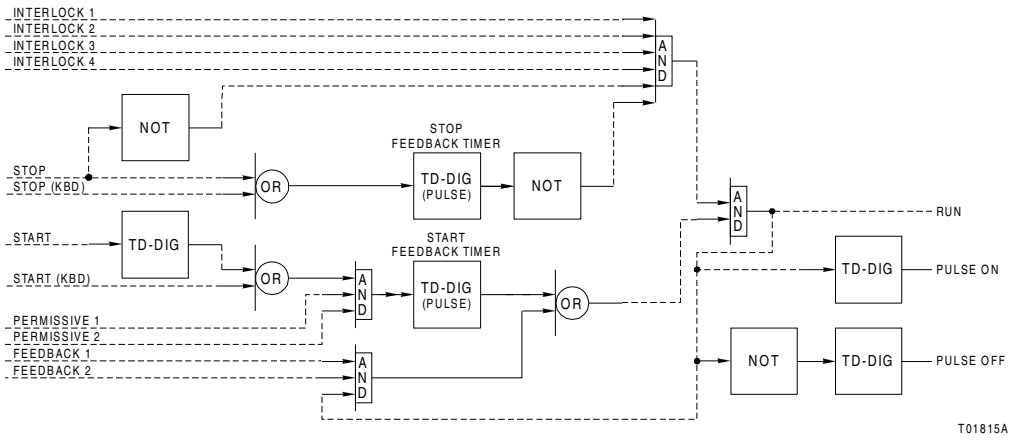


Figure 136-1 Basic Logic of Remote Motor Control Block

136.1.1Normal Operation of the Remote Motor Control

Start a device (energize output)
When <S1> makes an off to on transition or receives a start command from the HSI keyboard, the block undergoes a start. The output of the block is energized when all interlocks <S3> to <S6> and permissive inputs <S9> and <S10> are logic1.

Maintain a start
When <S1> makes the transition from off to on or receives a start command from the keyboard, the feedback timer starts. When the startup feedback time is exceeded (S11), the feedback inputs (<S7> and <S8>) must be logic 1. If they are not, then a ***bad start*** occurs and the output (N) goes to the stopped state (N equals zero).

NOTE: After the output of the block goes on, the permissives are not used. Permissives are not used to maintain an output.

Normal stop (de-energize output)
When <S2> makes an off to on transition or the keyboard sends a stop, the output of the block is de-energized and a shutdown is initiated. When a shutdown occurs, the RMC cannot be restarted until the stop feedback time (S17 or S11 when S17 equals 0.0) expires.

136.1.2Abnormal Operation

Bad start
A ***bad start*** occurs when:

1. Either interlocks <S3> to <S6> or permissives <S9> and <S10> are not logic 1 when <S1> makes an off to on transition or a keyboard start command has been received. In this case, block output N never goes on.
2. If interlocks and permissives are logic 1 and <S1> makes an off to on transition or a keyboard start is received, then the output of the block is energized. However, if the feedbacks do not come on within the allotted feedback time, then a bad start has occurred. Output N is de-energized and the shutdown sequence is initiated.

Fault
After a successful start, it is possible that one of the interlocks or feedbacks de-energize. When this happens, a ***fault*** has occurred and the output of block N is de-energized and the shutdown sequence is initiated.

136.1.3Pulsed Outputs

Outputs N+1 and N+2 are pulsed outputs that coordinate with output N. When output N makes an off to on transition, N+1 stays on for the length of time specified by S12. When output N makes an on to off transition, output N+2 stays on for the length of time specified by S13.

136.1.4 Specifications

S1
Block address of the start input. This input triggers on the rising edge of this signal.

S2
Block address of the stop input. This input must be momentary. A maintained signal at S2 causes an alarm condition.

S3 through S6
Block addresses of the interlock inputs.

S7 and S8
Block addresses of the feedback inputs.

S9 and S10
Block addresses of the start permissive inputs.

S11
Startup feedback time. This is the amount of time the remote motor control holds the run state to a logic 1 while waiting for a logic 1 to appear on both feedback inputs when a startup is initiated. If this feedback (logic 1) does not reach the RMC within this time, a bad start is assumed and the controller initiates a shutdown with an alarm status.

S12
Pulse on time. This is the amount of time the pulse on output stays at a logic 1 after a startup is initiated.

S13
Pulse off time. This is the amount of time that the pulse off output stays at a logic 1 after a shutdown is initiated.

S14
Display type.

S15
Spare.

S16
Status control specification. It controls the control output status when an alarm condition forces the RMC to initiate a shutdown. After the shutdown feedback timer expires, S16 determines if:

1. The RMC goes to stopped state (S16 equals 0.0).
2. The RMC enters a holding state that holds the control output status to an alarm value (1.0) until a there is a start or stop initiated. The control output status is set to a good value (0.0) when the holding state is terminated (S16 equals 1.0).
3. The RMC enters a holding state that holds the control output status to an alarm value (1.0) until there is a start or stop initiated. The control output status is set to a good value (0.0) when the holding state is terminated. When the hold state is terminated, any bad start or fault (S16 equals 2.0) condition present will be cleared.

S17
Shutdown feedback time. This is the amount of time the RMC waits for a motor to shutdown after a shutdown is initiated. If this specification is set to 0.0, the value in S11 is the shutdown feedback time.

136.1.5 Outputs

N
Run state. A logic 1 means the block is running and a logic 0 is stopped. Output N has an alarm status associated with it. The alarm status is set when a bad start, fault, or error condition exists.

N+1
Displays the pulse on. A logic 1 means the pulse on is on and a logic 0 is off.

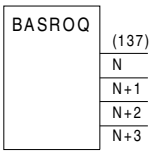
N+2
Displays the pulse off. A logic 1 means the pulse off is off and a logic 0 is on.

N+3
Displays the control output status.

0.0 = good
1.0 = alarm
2.0 = waiting for feedback

137.C and BASIC Program Real Output With Quality

The C and BASIC real output with quality function code provides a means of transmitting a real data value and its quality from a C or BASIC program to other function blocks. There are no specifications for this block. The BOUT command in the C or BASIC program defines the four outputs. This function code is similar to function code 93, except this code has a quality indicator. Quality can be measured only with a test quality block (function code 31). Quality is either good (zero) or bad (one). There are no specifications for this block. The BOUT command in the C or BASIC program defines the four outputs. Refer to the ***C Utility Program*** or the ***MFC BASIC Programming Language Reference*** for instructions on programming the modules.



Outputs

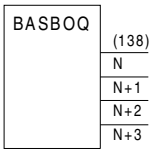
Blk	Type	Description
N	R	C or BASIC program command BOUT sets the output value
N+1	R	
N+2	R	
N+3	R	

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Full	Not used

138.C or BASIC Program Boolean Output With Quality

The C or BASIC program boolean output with quality function code provides a means of transmitting a boolean data value and its quality from the C program or BASIC program to other function blocks. There are no specifications for this block. The BOUT command in the C program or BASIC program defines the four outputs. This function code is similar to function code 93, except this code has a quality indicator. Quality can only be measured with a test quality block (function code 31). Quality is either good (zero) or bad (one). There are no specifications for this block. The BOUT command in the C program or BASIC program defines the four outputs. Refer to the ***C Utility Program*** or the ***MFC BASIC Programming Language Reference*** for instructions on programming the module.



Outputs

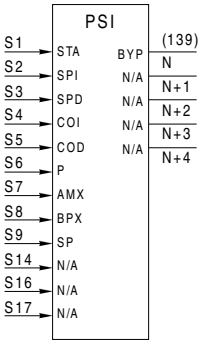
Blk	Type	Description
N	B	C program or BASIC program command BOUT sets the output value
N+1	B	
N+2	B	
N+3	B	

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Full	Not used

139.Passive Station Interface

The passive station interface function code emulates a control station using boolean and real inputs. Each passive station interface block links to an associated station block (function code 80). Specification S16 of the station block must be set to 254. This value defines a passive station interface.



Outputs

Blk	Type	Description
N	B	Auto bypass request
N+1	B	Reserved
N+2	B	Reserved
N+3	B	Reserved
N+4	B	Reserved

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of station block (function code 80)
S2	N	0	I	Note 1	Block address of set point raise
S3	N	0	I	Note 1	Block address of set point lower
S4	N	0	I	Note 1	Block address of control output raise
S5	N	0	I	Note 1	Block address of control output lower
S6	N	1	I	Note 1	Block address of auto/manual permissive
S7	N	0	I	Note 1	Block address of auto/manual transfer
S8	N	0	I	Note 1	Block address of manual bypass request
S9	N	5	I	Note 1	Block address of bypass control output
S10	N	0	I	Full	Check input quality: 0 = disable 1 = enable
S11	Y	2.500	R	Full	Set point ramp rate (units/sec)
S12	Y	2.500	R	Full	Control output ramp rate (units/sec)
S13	N	1	I	Full	Bad quality bypass control output option: 0 = prevent bypass exit 1 = permit bypass exit
S14	N	5	I	Note 1	Spare real input
S15	Y	1.000	R	Full	Maximum rate of change in bypass control output (units)
S16	N	5	I	Note 1	Spare real input
S17	N	0	I	Note 1	Spare boolean input

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

139.1Explanation

139.1.1Specifications

S1
Block address of the station block (function code 80). Specification S16 of the station block must be set to 254. This value defines a passive station interface.

S2
Block address of set point raise. This input can simulate set point ramping. Specification S11 adjusts the set point ramp rate.

S3
Block address of set point lower. This input can simulate set point ramping. Specification S11 adjusts the set point ramp rate.

S4
Block address of control output raise. This input can simulate control output ramping. Specification S12 adjusts the control output ramp rate.

S5
Block address of control output lower. This input can simulate control output ramping. Specification S12 adjusts the control output ramp rate.

S6
Block address of auto/manual permissive. A mode change is possible when this input is active (logic 1). Refer to S7.

S7
Block address of the auto/manual transfer. A zero to one transition on this input will cause a mode change (auto/manual) if S6 is active.

S8
Block address of the manual bypass request. The station is put in bypass when this input is active (logic 1).

S9
Block address of the bypass control output. This input allows the aligning of the station output to the actual output (in bypass).

NOTES:
1. The BRC-100 controller will enter the error mode when S13 equals zero and S9 references a block output that does not have a defined quality status.

2. The station will always enter auto bypass when the quality of the associated analog output defined by S28 of function code 80 is bad.

S10
Enables input quality checking:

0 = disable input quality checking
1 = enable input quality checking

NOTE: The station enters auto bypass when an input of function code 139 has bad quality when S10 equals one.

S11
Controls the set point ramp rate. The rate is expressed in units per second.

S12
Determines the control output ramp rate. The rate is expressed in units per second.

S13
Bad quality bypass control output option.

0 = bypass exiting not permitted when <S9> has bad quality
1 = bypass exiting always permitted

S14
Spare real input.

S15
Maximum change in the bypass control output input <S9> that is allowed for exiting bypass. This specification is only in effect while the station is in auto bypass. The station will be prevented from exiting auto bypass if the change in the <S9> input from one segment cycle execution to the next is greater than this value. To derive the S15 value from a rate of change value, use the formula:

FC139 S15 setting = rate of change (units/sec or units/min) × S2 of FC82

The <S9> input is typically connected to a feedback from the control output of the station block (function code 80). When this is the case, the use of this specification allows the station to maintain itself in the bypass condition until it is detected that the control output is not changing rapidly. The use of this specification prevents the auto bypass output from switching control of the control output analog output signal back to the AOT, CIO, CIS, or ASO hardware while the control output is changing.

S16 and S17
Spare real and boolean inputs respectively.

139.1.2Outputs

N
(Auto bypass request) This boolean output signals auto bypass when active (logic 1).

N+1 through N+4
Reserved.

139.2Application

Figure 139-1 shows a configuration using the passive station interface block. This example allows simultaneous control of a station function from both a console and external pushbuttons. A hard station cannot be used with this configuration.

NOTE: The inputs to the passive station interface do not have to come from external inputs. Internal logic can generate these inputs. For example, the auto permissive could be generated from logic that only allows automatic during certain portions of the process operation.

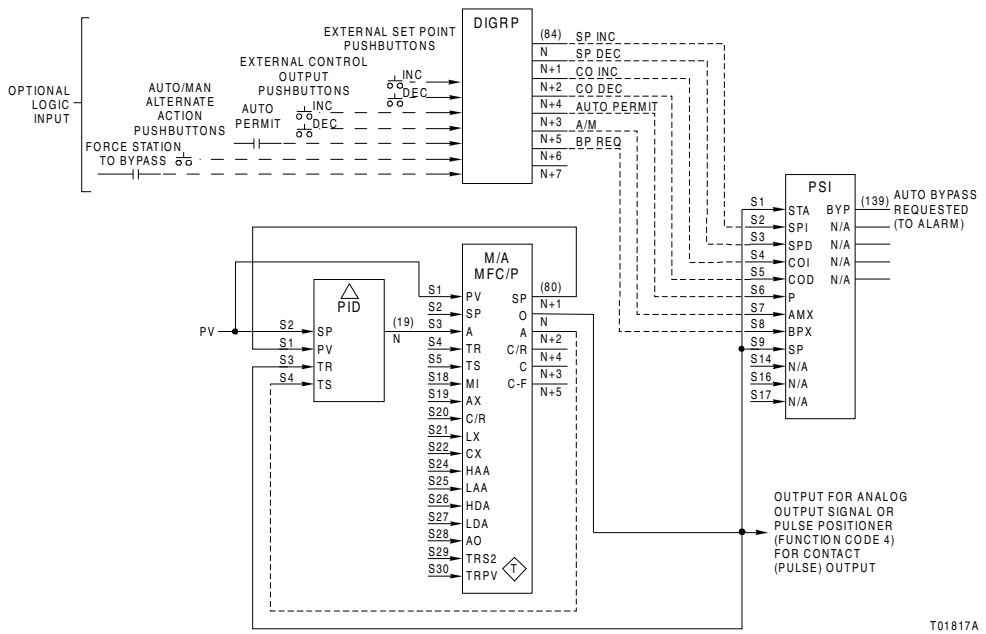
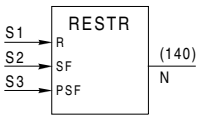


Figure 139-1 Passive Station Interface Example

140.Restore

The restore function code saves and restores critical block values (e.g., totalizers, counters, timers) to and from nonvolatile random access memory (NVRAM). Tables 140-4 and 140-5 show NVRAM and checkpoint utilization. During normal execution after startup, the internal and output block data referenced by this block goes to NVRAM. After a module power loss or recovery, the block referenced by this function block restores to the last saved state. Saving the block data can be disabled by either boolean input (S2 and S3). Both inputs must be logic 1 for the save to occur.
Data restore can be selected upon power restore, mode changes to execute, or both. If the module is offline longer than the maximum downtime, an external timer can be used to trip a digital input on an IMCIS12, IMCIS22, IMQRS12, IMQRS22, IMDSI12, IMDSI13, IMDSI14, IMDSI15 or an IMDSI22 module to prevent a restore after the timer expires. The expired logic state for the digital input is configurable. The data restores to the saved value before the first execution cycle of the configuration. The restore function block must be located at a block number greater than the function block being restored (S1).
When restoring a multiple output block (e.g., M/A station function code 80), the lowest output number (N) of that block must be specified by S1.



Outputs

Blk	Type	Description
N	B	No meaning, N must be greater than S1

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address to be restored
S2	N	0	I	Note 1	Block address of save flag: 0 = no save 1 = save data to NVRAM if <S3> = 1
S3	N	0	I	Note 1	Block address of save permissive: 0 = disabled 1 = enabled
S4	N	0	I	00, 01, 10 or 11	Restore condition: X0 = restore on power up X1 = no restore on power up 0X = restore on mode changes 1X = no restore on mode changes
S5	N	0	I	0 - 63	I/O module address of timer input
S6	N	0	I	0 - 123	Point number of timer input: unused = 000 DSI = X0Z = Group A (Z = 1 - 8) = X1Z = Group B (Z = 1 - 8) CIS = X2Z (Z = 1 - 3) Expire logic state: 0YZ = logic 0 1YZ = logic 1
S7	N	0	I	Full	Spare
S8	Y	0.000	R	Full	Spare
S9	Y	0	I	Full	Spare

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

140.1Specifications

S1
Block address of the block to be restored.

S2
Block address of the save flag. This specification is used in conjunction with <S3> to cause the internal and external output data to be saved to NVRAM every time the restore block executes while <S2> and <S3> equal one.

S3
Block address of the save permissive. The save permissive enables or disables the save flag input <S2>. Table 140-3 shows the save action dependent upon <S2> and <S3>.

S4
Restore condition. This specification determines which conditions cause the saved data in NVRAM to be restored to the

Table 140-3 Save Action

<S3>	<S2>	Save Action
0	0	No save
0	1	No save
1	0	No save
1	1	Save data to NVRAM

internal and external outputs in RAM. The ones digit of S4 determines if the data is to be restored when a power up of the module occurs. The tens digit of S4 determines if the data is to be restored upon a mode change or module reset via the reset button.

X0 = restore on power up
X1 = no restore on power up
0X = restore on mode changes
1X = no restore on mode changes

S5
I/O module address of the timer input. This specification is the expander bus address of the digital or control I/O module that provides the timer input signal.

S6
Point number of the timer input. This specification defines the digital or control I/O module input point number of the time input signal. The timer input prevents the module from restoring outdated information after it has been powered down for an extended period of time. The hundreds digit of S6 defines the expired logic state when a time-out condition occurs.

unused	=	000			
DSI	=	X0Z	=	Group A	(Z = 1 - 8)
	=	X1Z	=	Group B	(Z = 1 - 8)
CIS	=	X2Z			(Z = 1 - 3)

Expire logic state:

0YZ = logic 0
1YZ = logic 1

S7, S8 and S9
Spare.

140.2Module Memory Utilization

NOTE: When the NVRAM module memory utilization of a function block being restored is modified (i.e., a function block which has a variable usage equation associated with it in Table 140-4 or 140-5), it is necessary to manually perform the following operation in the configuration mode:
1. Modify S1 of the FC 140 function block to zero.
2. Modify S1 of the FC 140 function block to re-reference the block address of the function block being restored.
This operation allows the FC 140 function block to recognize the NVRAM changes of the function block being restored and to then adjust itself accordingly. Failure to perform this operation will result in the module entering error mode when function block configuration has been completed.

To determine NVRAM module memory utilization, use Appendix B, in conjunction with the formula:

$$NVRAM = 40 + N$$

where:

N = Applicable function code size from Tables 140-4 or 140-5. NVRAM is set to 46 when *N* is less than six.

Table 140-4 BRC-100/200/300/400 and IMMFP11/12 Additional NVRAM and Checkpoint Utilization Byte Size

Function Code	Size[1]	Function Code	Size[1]	Function Code	Size[1]	Function Code	Size[1]
1	4	15	4	36	2	61	6
2	4	16	4	37	2	62	10
3	14	17	4	38	2	63	118
4	12	18	18	39	2	64	70
5	10	19	22	40	2	65	4
6	4	24	4	41	28	66	80 slow
7	4	25	34	42	4	66	332 fast
8	10	26	10	45	8	68	24
9	14	30	14	50	2	69	4
10	4	31	2	51	4	79	60
11	4	32	2	52	2	80	68
12	4	33	2	55	80	81	52
13	2	34	2	58	Equation 1	82	36
14	4	35	10	59	2	83	2
84	34	120	4	152	18	191	88
85	10	121	12	153	22	192	104
86	16	122	4	154	20	193	62 + S5
89	4	123	20	155	52	194	50 + S4
90	52	124	12	156	40	198	16
91	4	125	4	157	Equation 3	199	8
92	4	126	16	160	114	210	178
93	16	128	4	161	38	211	70

Table 140-4 BRC-100/200/300/400 and IMMFP11/12 Additional NVRAM and Checkpoint Utilization Byte Size (Continued)

Function Code	Size[1]	Function Code	Size[1]	Function Code	Size[1]	Function Code	Size[1]
94	20	129	26	162	16	212	24
95	26	132	40	163	24	215	18
96	26	133	12	165	Equation 4	216	8
97	8	134	24	166	16	217	32
98	6	135	24	167	4	218[2]	294+S12
99	4	136	38	168	6	219	Equation 5
100	18	137	24	169	36	220	214 + S11
101	2	138	16	170	36	221[2]	54
102	12	139	24	171	4	222[2]	54
103	12	140	4	172	4	223[2]	54
104	24	141	16	173	4	224[2]	54
109	12	142	2	174	4	225[2]	54
110	6	143	14	177	82	226[2]	8
111	6	144	2	178	36	227[2]	64
112	6	145	14	179	Equation 12	228[2]	106
114	10	146	46	184	56	229[2]	72
115	4	147	66	185	52	241	18
116	2	148	Equation 2	186	68	242	88
117	2	149	44	187	24	247	70
118	4	150	28	188	24		
119	2	151	14	190	4		

NOTES:
1. Add 8 bytes to the byte size for BRC-300/400 controllers.
2. This function code is not supported by the IMMFP11/12.

Table 140-5 HAC Additional NVRAM and Checkpoint Utilization Byte Size

Function Code	Size	Function Code	Size	Function Code	Size	Function Code	Size
1	12	40	10	97	16	138	24
2	12	41	33	98	14	139	32
3	22	42	14	100	26	140	12
4	20	45	32	101	10	141	24
5	18	50	10	102	20	142	10
6	12	51	8	103	20	143	22

Table 140-5 HAC Additional NVRAM and Checkpoint Utilization Byte Size (Continued)

Function Code	Size	Function Code	Size	Function Code	Size	Function Code	Size
7	12	52	8	104	32	144	Equation 7
8	18	55	88	109	20	145	22
9	22	57	Equation 6	110	14	146	54
10	12	58	22	111	14	147	74
11	12	59	10	112	14	148	Equation 8
12	12	61	14	114	18	149	52
13	10	62	22	115	12	150	42
14	12	63	110	116	10	151	26
15	12	64	62	117	10	152	26
16	12	65	12	118	12	153	30
17	12	66	340	119	10	154	28
18	26	68	36	120	12	155	Equation 9
19	30	69	12	121	22	156	48
24	12	79	68	122	14	157	Equation 10
25	34	80	80	123	32	160	122
26	20	81	66	124	20	161	46
30	40	82	56	125	12	162	24
31	10	83	10	126	24	163	32
32	10	84	42	128	22	165	Equation 11
33	10	85	18	129	38	166	24
34	10	86	24	132	48	167	12
35	18	87	10	133	20	168	14
36	10	88	26	134	36	169	44
37	10	90	58	135	32	170	44
38	10	95	30	136	50	171	12
39	10	96	34	137	32	172	12
173	12	188	32	212	34	223	64
174	14	190	12	215	26	224	64
177	94	191	96 + S16	216	16	225	64
178	52	192	112 + S24	217	40	226	16
179	Equation 12	193	72 + S5	218	308 + S12	227	Note 1
184	84	194	62 + S4	219	Equation 13	228	Note 1
185	60	198	24	220	226 + S11	241	30

Table 140-5 HAC Additional NVRAM and Checkpoint Utilization Byte Size (Continued)

Function Code	Size	Function Code	Size	Function Code	Size	Function Code	Size
186	76	199	16	221	50	242	96
187	32	211	84	222	64	247	Note 1

NOTE:
1. This function code is not supported by the HAC01.

140.3 Memory Usage Equations

1. 14 + S5 x 4
2. 188 + S12 x b

   where:

   b = 1 (if S12 is positive)
   - or -
   b = 1024 (if S12 is negative)
3. 12 + 4 x (S21 + 5) + 4 x (S22 + 5)
4. 26 + 4 x S2
5. 156 + S13 x 56
6. 182 + S2 + (S8 x 16)
7. 10 + (S1 x 1024)
8. 248 + (S12 x b)

   where:

   b = 1 (if S12 is positive)
   - or -
   b = 1024 (if S12 is negative)
9. 60 + (S7 x 20)
10. 20 + [4 x ({5 + S21} + {5 + S22})]
11. 34 + (4 x S2)
12. a + [b x (S4)] + [c x (n{S3})] + d

    where:

    n{S3} = Number of modes selected for S3

    if S2 = 0, 3, 10, or 11, then a = 106, b = 12, and c = 8

    if S2 = 1, 2, 5, 6, 7, 9, 12, or 13, then a = 100, b = 10, and c = 4

    if S2 = 4 or 8, then a = 128, b = 25, and c = 32

    if IMMFP11/12 or BRC-100, then d = 0
    if HAC, then d = 4
13. 378 + (142 x S13)

140.4 Applications

Figure 140-1 shows an example of using function code 140. In the application, the moving average block (function code 165) calculates the average of the square root value at block 400. The moving average is calculated every 15 seconds and is calculated based on one hour's worth of collected data. The average calculation is suspended whenever the quality of the analog input is bad.

The restore block (function code 140) insures that the hour's worth of collected data is not lost in the event that the controller module loses power. The restore block is configured for restore on power up. The elapsed timer block controls the update rate of NVRAM data. The NVRAM data is saved once every 15 seconds.

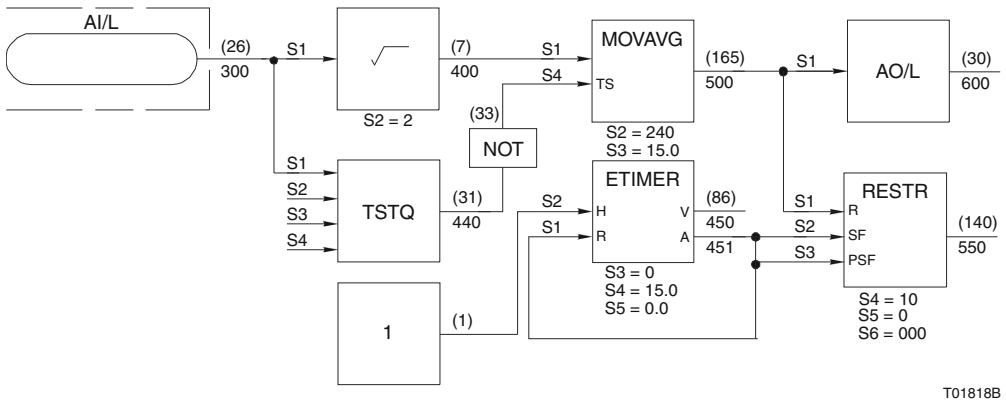
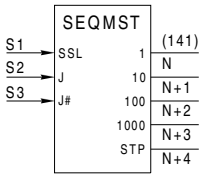


Figure 140-1 Using the Restore Block

141.Sequence Master

The sequence master function code contains a series of masks selected by the step jump input. The masks define the states of the four boolean outputs for each step. A mask is specified as a series of zeroes, ones or twos. Each digit in the mask is associated with a specific output. A value of two represents the ***don't care*** state. When this state is selected, the output value assumes the value of the previous step's state. There is a disable mask in addition to the step masks. The disable mask defines the safe state of the outputs. The default mask is output for step zero. If additional steps are required, a sequence master block can be linked to a sequence slave block. Sequence slave blocks can also be linked together. Series of sequence master and slave blocks can be run in parallel to provide multiples of four boolean outputs.



Outputs

Blk	Type	Description
N	B	Ones digit of current mask
N+1	B	Tens digit of current mask
N+2	B	Hundreds digit of current mask
N+3	B	Thousands digit of current mask
N+4	R	Current step number

NOTE: If mask digit is two, then output equals previous step output value.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of sequence slave (0 = end)
S2	N	0	I	Note 1	Block address of step jump trigger. Requires a 0 to 1 transition after start-up is complete.
S3	N	5	I	Note 1	Block address of step jump number
S4	Y	0	I	0 - 2222	Default mask
S5	Y	0	I	0 - 2222	Step 1 mask
S6	Y	0	I	0 - 2222	Step 2 mask
S7	Y	0	I	0 - 2222	Step 3 mask
S8	Y	0	I	0 - 2222	Step 4 mask
S9	Y	0	I	0 - 2222	Step 5 mask
S10	Y	0	I	0 - 2222	Step 6 mask
S11	Y	0	I	0 - 2222	Step 7 mask
S12	Y	0	I	0 - 2222	Step 8 mask
S13	Y	0	I	0 - 2222	Step 9 mask
S14	Y	0	I	0 - 2222	Step 10 mask
S15	Y	0	I	0 - 2222	Step 11 mask
S16	Y	0	I	0 - 2222	Step 12 mask
S17	Y	0	I	0 - 2222	Step 13 mask
S18	Y	0	I	0 - 2222	Step 14 mask
S19	Y	0	I	0 - 2222	Step 15 mask

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S20	Y	0	I	0 - 2222	Step 16 mask
S21	Y	0	I	0 - 2222	Step 17 mask
S22	Y	0	I	0 - 2222	Step 18 mask
S23	Y	0	I	0 - 2222	Step 19 mask
S24	Y	0	I	0 - 2222	Step 20 mask
S25	Y	0	I	0 - 2222	Step 21 mask
S26	Y	0	I	0 - 2222	Step 22 mask
S27	Y	0	I	0 - 2222	Step 23 mask
S28	Y	0	I	0 - 2222	Step 24 mask
S29	Y	0	I	0 - 2222	Step 25 mask
S30	Y	0	I	0 - 2222	Step 26 mask
S31	Y	0	I	0 - 2222	Step 27 mask
S32	Y	0	I	0 - 2222	Step 28 mask
S33	Y	0	I	0 - 2222	Step 29 mask
S34	Y	0	I	0 - 2222	Step 30 mask
S35	Y	0	I	0 - 2222	Step 31 mask
S36	Y	0	I	0 - 2222	Step 32 mask

NOTES:

1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

141.1Explanation

141.1.1Specifications

S1
Block address of the sequence slave (function code 142).

0 = none (Steps zero to 32 supported)

Each additional sequence slave adds another 32 step masks to the selection.

S2
Block address of the step jump trigger. When <S2> transitions from zero to one, the step jump number <S3> is loaded as the current step number (N+4) and the appropriate mask is loaded to update the outputs N, N+1, N+2 and N+3.

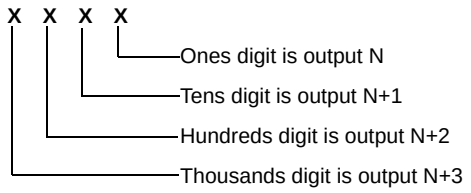
S3
Block address of the step jump number. The input value is forced equal to zero when it is less than or equal to zero. If the input value is greater than the maximum step number, the maximum step number is used. The maximum step number is determined from the number of sequence slaves linked to the sequence master block.

$$\textit{maximum step number} = (\textit{master} + N)\ 32$$

where:

N = 0	32 steps
N = 1	64 steps

N = 2	96 steps
N = 3	128 steps
etc.	



S4
Specification S4 is the default mask.

Step 0 = default mask

S5 through S36
Step 1 to Step 32 masks.
where:

X = 0	Output equals zero
X = 1	Output equals one
X = 2	Output equals hold previous output

Figure 141-1 and Table 141-6 show a series of sequence master and slave blocks run in parallel to provide multiple outputs.

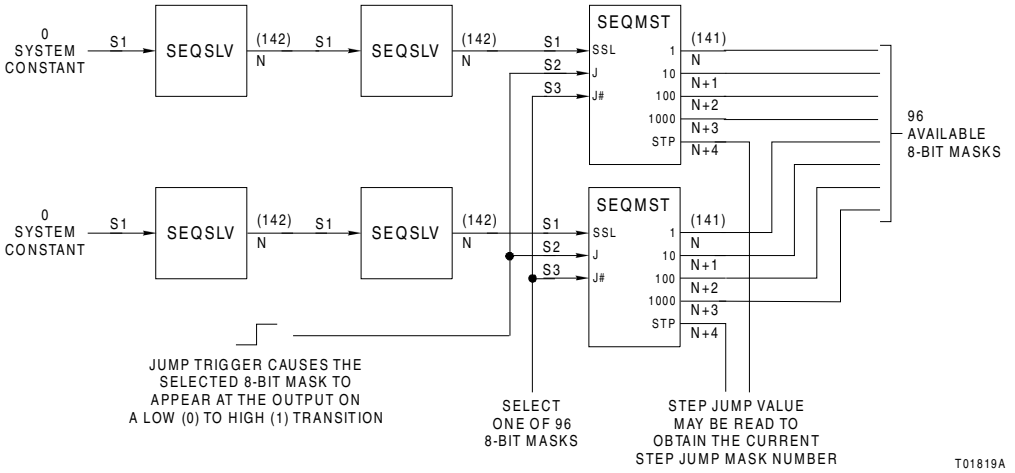


Figure 141-1 Parallel Sequence Master to Slave

Table 141-6 Output Descriptions

Step No. <S3>	Mask	Output				
		N+4	N+3	N+2	N+1	N
0.0	S4 = 0000	0.0	0	0	0	0
1.0	S5 = 0101	1.0	0	1	0	1
2.0	S6 = 1200	2.0	1	1	0	0
3.0	S7 = 2012	3.0	1	0	1	0
4.0	S8 = 0001	4.0	0	0	0	1

Figure 141-1 shows a logic diagram with the sequence master block used in a batch operation. The sequence master block can be thought of as a simpler version of the sequence generator block (function code 161). In most batch applications of the sequence generator block, the sequence master block may be used instead.

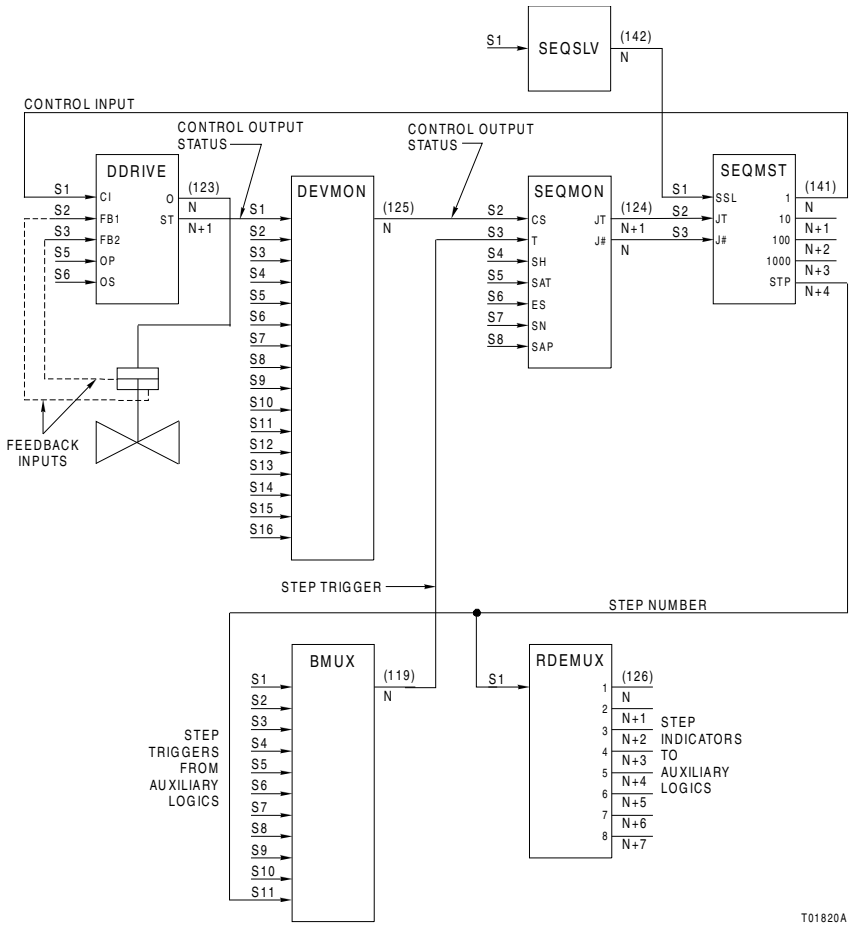


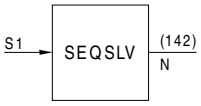
Figure 141-1 Using Sequence Master Block in Batch

The number of steps for an application can be expanded in multiples of 32 by linking sequence slave blocks (function code 142). Additional sequence master blocks can be run in parallel to provide expanded step masks in multiples of four output states per step number.

142.Sequence Slave

The sequence slave function code contains a series of masks that a sequence master function code steps through on a selected step basis. The purpose of the sequence slave function code is to expand the step number capability of the sequence master function code. The masks define the states of the sequence master block's four boolean outputs for each step. A series of zeroes, ones, or twos specifies a mask. Each digit in the mask is associated with a specific output. A value of two represents the ***don't care*** state. When this state is selected, the output value assumes the value of the previous step's state. If additional steps are required, the sequence slave block can be linked to another sequence slave block.

NOTE: Function code 142 must be used in combination with a sequence master block (function code 141). Refer to the applications section of function code 141 for more information.



Outputs

Blk	Type	Description
N	B	No meaning

Specifications

Spec	Tune	Default	Type	Range	Description[1]
S1	N	0	I	Note 2	Block address of next sequence slave (0 = end)
S2	Y	0	I	0 - 2222	Step N+1 mask
S3	Y	0	I	0 - 2222	Step N+2 mask
S4	Y	0	I	0 - 2222	Step N+3 mask
S5	Y	0	I	0 - 2222	Step N+4 mask
S6	Y	0	I	0 - 2222	Step N+5 mask
S7	Y	0	I	0 - 2222	Step N+6 mask
S8	Y	0	I	0 - 2222	Step N+7 mask
S9	Y	0	I	0 - 2222	Step N+8 mask
S10	Y	0	I	0 - 2222	Step N+9 mask
S11	Y	0	I	0 - 2222	Step N+10 mask
S12	Y	0	I	0 - 2222	Step N+11 mask
S13	Y	0	I	0 - 2222	Step N+12 mask
S14	Y	0	I	0 - 2222	Step N+13 mask
S15	Y	0	I	0 - 2222	Step N+14 mask
S16	Y	0	I	0 - 2222	Step N+15 mask
S17	Y	0	I	0 - 2222	Step N+16 mask
S18	Y	0	I	0 - 2222	Step N+17 mask
S19	Y	0	I	0 - 2222	Step N+18 mask
S20	Y	0	I	0 - 2222	Step N+19 mask
S21	Y	0	I	0 - 2222	Step N+20 mask
S22	Y	0	I	0 - 2222	Step N+21 mask
S23	Y	0	I	0 - 2222	Step N+22 mask

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description[1]
S24	Y	0	I	0 - 2222	Step N+23 mask
S25	Y	0	I	0 - 2222	Step N+24 mask
S26	Y	0	I	0 - 2222	Step N+25 mask
S27	Y	0	I	0 - 2222	Step N+26 mask
S28	Y	0	I	0 - 2222	Step N+27 mask
S29	Y	0	I	0 - 2222	Step N+28 mask
S30	Y	0	I	0 - 2222	Step N+29 mask
S31	Y	0	I	0 - 2222	Step N+30 mask
S32	Y	0	I	0 - 2222	Step N+31 mask
S33	Y	0	I	0 - 2222	Step N+32 mask

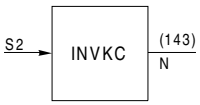
NOTES:

1.N = 32 (for first sequence slave in linked list)
N = 64 (for second sequence slave in linked list)
N = 96 (for third sequence slave in linked list)
etc.

2. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

143.Invoke C

Function code 143 invokes the C program. Specifically, the invoke C block calls a function within the C program designated as a ***segment entry point***.
The C program can have as many as eight segment entry points. Each entry point has a function block segment (segment entry point zero with function block segment zero, etc.). A function name (i.e., the name of the functions that compose the C program) connects with each entry point. The same function name may connect with more than one entry point.
During function block execution, an encounter with an invoke C block transfers control to the C program at the associated entry point (i.e., calls the appropriate function). When that program returns, function block execution continues normally at the next block. The entry point program function may call other functions of the C program before returning.
Besides the normal utilization constraints, there are no further restrictions on the configuring of invoke C blocks. Specifically, invoke C blocks may appear in any or all function block segments. Since each function block segment is a separate task, more than one task may execute the C program. When the C program is subject to multitasking, all shared functions among tasks must be coded so as to be reentrant.
For more information on C program development, consult the ***C Utility Program*** product instruction.



Outputs

Blk	Type	Description
N	R	Value may be set by C program using the ***putargs*** function

Specifications

Spec	Tune	Default	Type	Range	Description
S1	Y	0	I	0, 1 or 2	Program operating mode: 0 = C program will execute normally 1 = C program execution is inhibited, blocks will run
S1 *(cont.)*	Y	0	I	0, 1 or 2	2 = C program is in debug mode for use with the EWS C debugger
S2	N	0	I	Note 1	Block address of inhibit execution flag
S3	Y	0	I	Full	Program readable parameter
S4	Y	0	I	Full	Program readable parameter
S5	Y	0	I	Full	Program readable parameter
S6	Y	0	I	Full	Program readable parameter
S7	Y	0.000	R	Full	Program readable parameter
S8	Y	0.000	R	Full	Program readable parameter
S9	Y	0	I	Full	Spare
S10	Y	0	I	Full	Spare
S11	Y	0.000	R	Full	Spare

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

143.1 Applications

Figure 143-1 shows a configuration of a timer used to monitor and provide alarm capability. Figure 143-1 shows the entry in the C utility program for this example. There are numerous ways to use the invoke C command block and several ways to configure a timer. This is just one example.

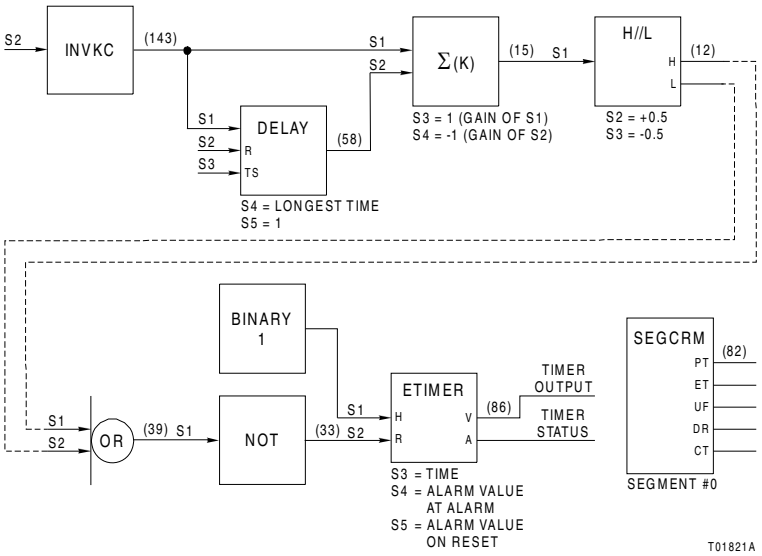


Figure 143-1 Watchdog Timer Configuration

EDIT USER RAM SPECIFICATIONS

Max size of C idata section	0
Max size of C udata section	0
Min size of dynamic memory pool	0

Segment Number	Name of Entry Point	Stack Size
0	Main	1024
1		0
2		0
3		0
4		0
5		0
6		0
7		0

Alt-H Help

T01822A

Figure 143-1 Entry in C Utility Program

The example in Figure 143-1 shows a C program called main being executed in segment zero of the module. The invoke C function code is placed in numerical order after the segment control block.
The C program has an output to the invoke C specification which triggers a time delay command (function code 58) in the succeeding configuration.
This timer configuration starts with a time delay command (function code 58). The input comes from the invoke C command block. The delay is equal to the longest inference cycle time period anticipated, and the number of intervals set to one. A 2-input summer (function code 15) is next. The summer's inputs come from the invoke C command block (gain equals one) and the time delay block output (gain equals -1). Specification <S2> of the summer block subtracts from <S1> and the value goes to the high/low compare (function code 12).

The high/low compare block compares the input with the high and low limits specified. The high limit equals +0.5 and the low limit equals -0.5. If the input is equal to or greater than the high limit, the high output is logic 1 and the low output a logic 0. If the input is equal to or less than the low limit, the low output is logic 1 and the high output a logic 0. If the input value is between the high and low limits, both outputs are logic 0.

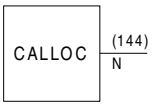
Both of the high/low compare outputs go to the 2-input OR block (function code 39). If either or both inputs to the OR block equal logic 1, the output equals logic 1. When both inputs equal logic 0, the output is logic 0.

The output of the OR block goes to the NOT block which reverses the signal (i.e., logic 0 input equals logic 1 output). The NOT output signal goes to the reset input of the elapsed timer block (function code 86). A constant logic 1 signal goes to the hold input. The elapsed timer block sets the alarm output to a logic 1 when the reset input drops to a logic 0 for the time specified.

This configuration compares the present vital sign with the previous cycles vital sign. If there is no significant change (plus or minus 0.5), the elapsed timer starts timing. When the specified time elapses, the alarm output goes to logic 1. Upon a significant change, the configuration continues comparing vital signs.

144.C Allocation

The C allocation function code declares the amount of volatile and nonvolatile memory allocated for C programs in the module. The declarations serve as a marker for module memory utilization calculations.
Specification S1 sets the amount of random access memory (RAM) in one-kilobyte increments. Specification S1 should be greater than or equal to the total RAM allocated to C specification for memory format defined in the C utility program. For more information on memory specification refer to the product instruction ***C Utility Program***.
Specification S2 sets the amount of nonvolatile random access memory (NVRAM). This should be greater than or equal to the total NVRAM allocated to C specification for memory format defined in the C utility program. For more information on memory specification refer to the product instruction ***C Utility Program***.



Outputs

Blk	Type	Description
N	B	Unused

Specifications

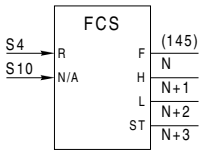
Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 320	RAM allocation in 1 kbyte increments
S2	N	0	I	0 - 80	NVRAM allocation in 1 kbyte increments

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

145.Frequency Counter/Slave

The frequency counter/slave function code interfaces the frequency counter slave to a multifunction controller or multifunction processor. Output N provides the frequency in hertz times the gain (S5). The quality and alarms on this output reflect the current state of the block. This includes the rate alarms and the high and low rate alarms. The high and low rate alarms display on operator devices as *HD* and *LD* respectively. The high and low alarms and quality are included in the frequency output N. These alarms can be detected using the test alarm function (function code 69).
Quality is detected using the test quality function (function code 31). High rate of change occurs when the frequency output N changes in a positive direction from the previous output in excess of S9. Low rate of change occurs when the frequency output has changed in a negative direction from the previous output in excess of S9. The configurable high and low alarm limits (S6 and S7) determine when the alarm outputs N+1 and N+2 are set. To prevent alarm ***chatter***, the alarm deadband (S8), expressed in engineering units, is provided for S6 and S7. The last output N+3 reports the status of I/O module communications.
The IMFCS01 module typically measures rotational speed of a turbine utilizing a toothed wheel attached to the shaft and a magnetic pickup.



Outputs

Blk	Type	Description
N	R	Frequency (hertz) × gain (with alarms)
N+1	B	High alarm: 0 = good 1 = alarm
N+2	B	Low alarm: 0 = good 1 = alarm
N+3	B	I/O module communication: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus address of I/O module
S2	N	0	I	Full	Spare
S3	N	0	I	Full	Spare
S4	N	0	I	Full	Spare boolean input
S5	Y	1.000	R	Full	Engineering unit gain
S6	Y	9.2 E18	R	Full	Engineering unit high alarm
S7	Y	-9.2 E18	R	Full	Engineering unit low alarm
S8	Y	0.000	R	Full	Absolute alarm deadband
S9	Y	9.2E18	R	Full	Engineering unit rate of change alarm.
S10	N	0	I	Note 1	Spare boolean input

NOTES:
1. Maximum values are: 9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

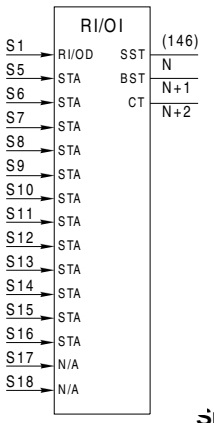
146.Remote I/O Interface

The remote I/O interface function code defines the interface between:

- A BRC-100/200/300/400/410 controller or HAC controller and a local remote I/O module.
- RIO-BRC (BRC-300/400/410 configured for HNET communication) controllers.

Expander Bus Communication
The remote I/O interface function code defines the interface between a BRC-100 controller or HAC controller and a local remote I/O module. The local remote I/O module is the remote master processor (RMP). Function code 146 defines the expander bus addresses for both the primary and secondary remote master processors.

NOTE: In order to minimize the possibility of malfunctions, it is essential that the RMP function block (FC146), the RSP function blocks (FC147), and all the slave definition blocks for the remote slaves (FC79, FC83, FC84, FC132, etc.) all reside within the same segment. In addition, these blocks must be sequenced in the following order: FC146 must reside at a lower block number than all FC147s for that link, and the FC147s must reside at lower block addresses than the slave definition blocks and station blocks (FC80) for the remote slaves.



Outputs

Blk	Type	Description
N	B	Primary RMP or RIO-BRC status: 0 = good 1 = bad
N+1	B	Secondary RMP or RIO-BRC status: 0 = good 1 = bad
N+2	R	Cycle time of remote master processor (secs)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of remote I/O definition
S2	N	0	I	0 - 63	Expander bus address of primary RMP or HNET address of primary RIO-BRC
S3	N	0	I	0 - 63	Expander bus address of secondary RMP or any valid number not equal to S2 (HNET communication)
S4	N	0	I	Full	Remote block input allocation: 0 = IMRIO02 module being used (expander bus communication) >0 = total remote I/O blocks to allocate to RIO-BRC controller (HNET communication) Refer to **Calculating Specification S4** for more information.

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S5	N	2	I	Note 1	S5 - S16 are block addresses of control or indicator stations (expander bus communication) or spare parameters (HNET communication)
S6	N	2	I	Note 1	
S7	N	2	I	Note 1	
S8	N	2	I	Note 1	
S9	N	2	I	Note 1	
S10	N	2	I	Note 1	
S11	N	2	I	Note 1	
S12	N	2	I	Note 1	
S13	N	2	I	Note 1	
S14	N	2	I	Note 1	
S15	N	2	I	Note 1	
S16	N	2	I	Note 1	
S17	N	0	I	Note 1	Spare boolean input
S18	N	0	I	Note 1	Spare boolean input

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the BRC-400 and HAC

Calculating Specification S4

The value used for Specification S4 is calculated based on the number of input block references defined by all linked I/O function codes attached to the FC146/FC147 linked list. The S4 value must non-zero when the BRC-300/400/410 controller is being used for remote I/O. Use the information in Figure 146-1 to calculate the S4 value.

Table 146-1 Block Inputs to be Reserved

Function Code	Block Inputs to be Reserved
55	9
79	6
83	8
84	0
102	0
103	0
104	2
109	0
114	0
115	1
128	0

Table 146-1 Block Inputs to be Reserved

Function Code	Block Inputs to be Reserved
132	1
145	2
149	7
215	1
247	24

Table 146-2 provides an example of how Specification S4 is calculated.

Table 146-2 Example S4 Calculation

Function Code	Quantity Used in Configuration	Blocks to be Allocated
55	3	3 x 9 = 27
83	3	3 x 8 = 24
84	3	3 x 0 = 0
149	1	1 x 7 = 7
Blocks to be reserved for future expansion (optional)		20
Total blocks reserved (value to be used in Specification S4)		78

HNET Communication

FC146 (remote I/O interface) and FC147 (remote I/O definition) can use HNET as a remote I/O link. The BRC-300/400/410 controller is capable of running as both the RIO-BRC master and RIO-BRC slave. Minimal configuration alterations are required to use the HNET functionality. Existing IMRIO02 modules and RIO-BRC controllers can be used simultaneously in the same controller configuration. Porting an RIO02 configuration to a RIO-BRC configuration can be easily done, due to the small differences in configuration. With HNET functionality, the RMP/RSP is effectively replaced by a single RIO-BRC on HNET. Redundant communications is inherently part of HNET and the BRC also supports redundancy as a RIO-BRC. The RIO-BRC scans its I/O using the same function codes that would run on normal BRC controllers, but can be located remotely.

NOTE: In order to minimize the possibility of malfunctions, it is essential that function block (FC146), function blocks (FC147), and all the slave definition blocks for the remote slaves all reside within the same segment. In addition, these blocks must be sequenced in the following order: FC146 must reside at a lower block number than all FC147s for that link, and the FC147s must reside at lower block addresses than the slave definition blocks and station blocks (FC80) for the remote slaves.

Configuration is performed on the local BRC-300/400/410. Similar to IMRIO02 operation, the local BRC-300/400/410 downloads the associated I/O function codes defined in FC146/FC147 into the RIO-BRC. If the I/O device is defined with more than one linked function code (that is, 14 AO channels require two FC149s), only the first FC needs to be referenced in FC146/FC147 for all associated/linked function codes to be downloaded into the RIO-BRC. The complete I/O slave configuration executes on the RIO-BRC.

Online configuration has one small enhancement, The RIO-BRC operation is affected only when changes are made to the function codes it is using. The I/O on the RIO-BRC holds the last value during the configuration download and then resumes updating with dynamic data as soon as the update is complete (same as IOR800). If the configuration changes are not associated with the RIO-BRC then dynamic data updates continue unchanged. Multiple RIO-BRCs can be configured, and each FC146/147 configuration is managed independently. A configuration change on one RIO-BRC has no impact on another RIO-BRC configuration.

Block number order is enforced via FC146/FC147. The base block number of FC146 must be less than any I/O it is linked to including FC147. The linked list of FC147 must be in ascending block number order (that is, S1 block number reference

must be greater than its own block number). I/O function codes must have block numbers greater than the FC147 they are linked to. Figure 146-1 shows an example of expander bus and HNET communication configurations.

Figure 146-1 Example Configurations

146.1Explanation

The following explanation adds detail to the specifications for the remote I/O interface block.

Specifications

S1
Block address of the first remote I/O definition (function code 147). This specification links the remote I/O definition block to the remote I/O interface block. This link defines the remote I/O module processors (or RIO-BRC controller) and their associated I/O modules (or allocated remote I/O blocks). Specification S1 can be set to zero if there is no remote I/O definition block (or allocated remote I/O blocks).

S2
Expander bus address of the primary remote master processor. The local remote I/O module is the remote master processor. Remote master processors can address up to eight control stations and four indicator stations. Stations addressed by remote master processors are configured as inputs to this block. This specification also can be used to define the HNET address of the primary RIO-BRC module.

S3
Expander bus address of the secondary remote master processor (expander bus communication) or any valid number not equal to S2 (HNET communication).

NOTE: If redundancy is not implemented, the definition of the secondary expander bus address must be set equal to the primary (S2).

When HNET communication is being used, the secondary RIO-BRC controller has the same address as the primary (S2 would be equal to S3 incorrectly telling the module redundancy is not implemented) so set this specification to any valid number that is not equal to the number in specification S2.

S4
Remote block input allocation. Set to zero when IMRIO02 modules (expander bus communication) are supplying remote values or set to the total number of remote I/O blocks to allocate to the remote RIO-BRC module (HNET communication) when it is supplying remote values.
When HNET communication is being used, The RIO-BRC is configured with the I/O function codes referenced in FC147 and executes them to update the I/O data. Only the base function code is defined in FC147 (S5-S36). When the I/O slave definition uses multiple function codes, all associated function codes (block number linking specifications that is) for a particular slave interface are downloaded. The function codes downloaded to the RIO-BRC need their block input values in order to execute correctly. Each I/O function block has a certain number of block inputs that must be updated. S4 allocates the necessary memory in both the local BRC and remote RIO-BRC for the input blocks to be updated. Add up the number of inputs on all of the RIO-BRC function codes being referenced (excluding FC80/146/147) for a simple input total and enter the total into S4. A larger number can be used if future changes are expected. If optimization of the S4 value is desired, three additional rules can be used to subtract from the previous simple input total:

1. Block number addresses that reference the base blocks (block #0 to #29) do not need allocation.
2. Duplicated references only need one allocation.
3. Linking references can be excluded.

S5 through S16
Block addresses of the control stations or indicator stations (expander bus communication) or spare parameters (HNET communication). Valid station addresses are zero through seven. Valid indicator addresses are eight through 11.

S17 and S18
Spare boolean inputs.

146.2RMP/RSP Memory Usage Calculation (Expander Bus)

The number of slaves an RMP can support is determined by memory usage only. To calculate RMP memory requirements, refer to ***RMP/RSP Memory Usage Calculation (Expander Bus)*** in the function code 147 section of this manual.

146.3Applications

Figure 146-1 shows the block diagram configuration detailing the relationship between a harmony controller and a remote I/O module interface and the control station and indicator station. The associated module function codes required to define this interface are also shown.

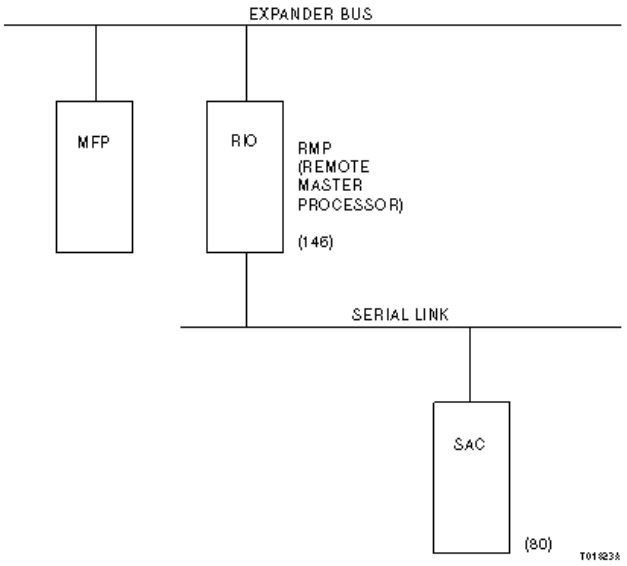


Figure 146-1 Remote I/O Interface Example (RIO02 Aplications Only)

The remote I/O interface block defines the interface between the harmony controller and the remote I/O module. This interface links the station functions in the module to their respective remote I/O modules. Refer to function code 147 for a block diagram example showing the remote I/O block working in conjunction with the remote I/O definition block.
Figure 146-1 illustrates a configuration using the remote I/O interface and the remote I/O definition blocks (function codes 146 and 147 respectively). The control interface slave (function code 79) references the I/O from a control interface slave

connected to the remote slave processor. This configuration has the control stations interfaced via the remote master processor.

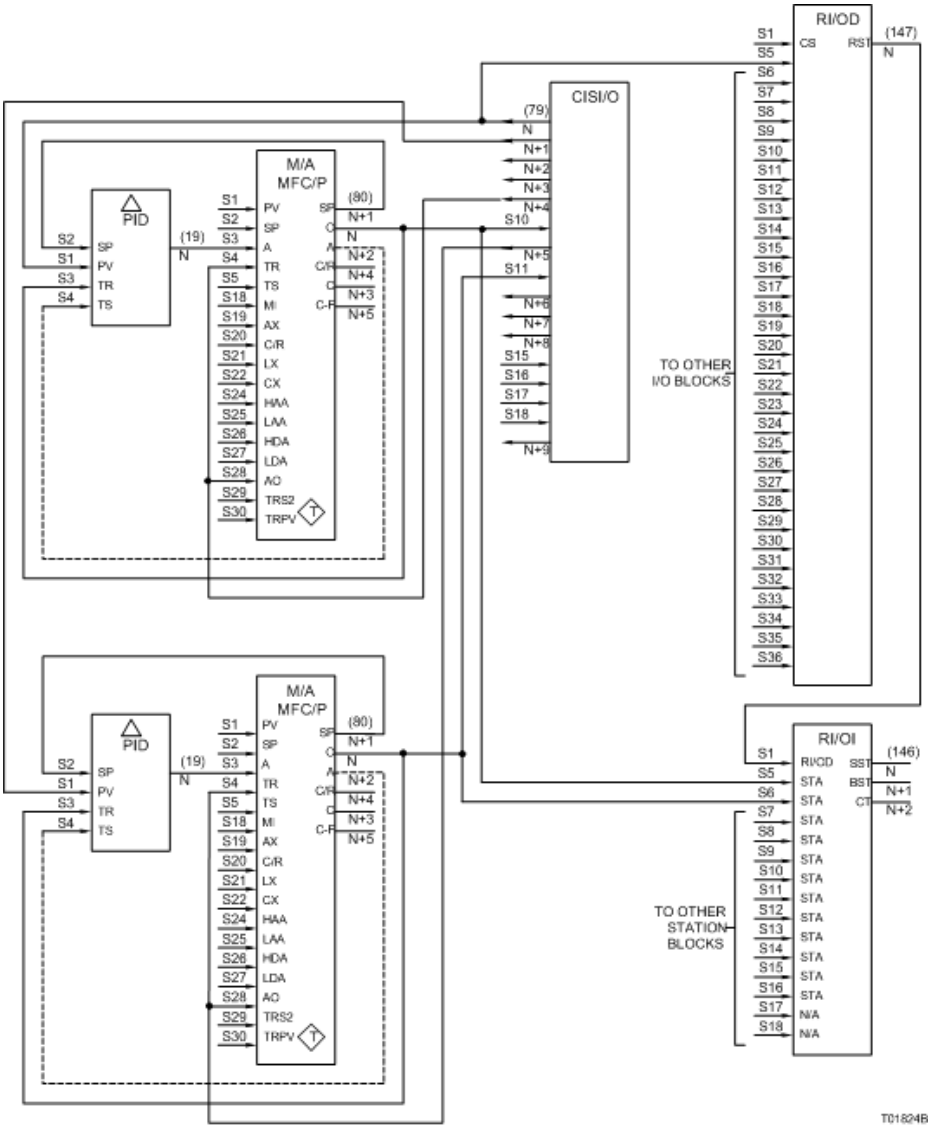


Figure 146-1 Remote I/O Interface Configuration

NOTE: The configuration shown in Figure 146-1 is only valid for applications using IMRIO02 modules.

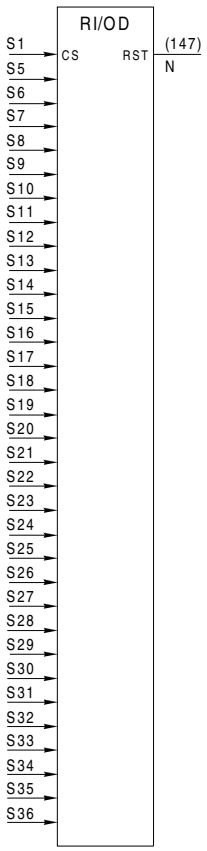
147.Remote I/O Definition

The remote I/O definition function code defines:

- The communication address (expander bus communication) and environment of remotely located I/O modules.
- The block address of the next remote I/O definitions allocated for more than 32 remotely located I/O function codes.

Expander Bus Communication
The remote I/O definition function code defines the communication address and environment of remotely located I/O modules. This module is the remote slave processor (RSP). Each processor has a unique serial link communication address. This block defines that address.
The remote I/O definition block also defines the digital and analog I/O module types interfaced with the remote processor.

NOTE: In order to minimize the possibility of malfunctions, it is essential that the RMP function block (FC146), the RSP function blocks (FC147), and all the slave definition blocks for the remote slaves (FC79, FC83, FC84, FC132, etc.) all reside within the same segment. In addition, these blocks must be sequenced in the following order: FC146 must reside at a lower block number than all FC147s for that link, and the FC147s must reside at lower block addresses than the slave definition blocks and station blocks (FC80) for the remote slaves.



Outputs

Blk	Type	Description
N	B	Remote processor status: 0 = good 1 = bad

NOTE: If running a redundant remote I/O configuration, block N indicates good (zero) when both remote processors function properly. Block N indicates bad (one) when one or both of the remote processors are offline.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of next remote I/O definition
S2	N	0	I	0 - 63	Remote processor serial link communication address (expander bus communication, FC146 S4 = 0) Spare parameter (HNET communication, FC146 S4 > 0)
S3	N	0	I	Full	Spare parameter
S4	N	0	I	Full	Spare parameter
S5	N	2	I	Note 1	Block address of I/O block or station
S6	N	2	I	Note 1	Block address of I/O block or station
S7	N	2	I	Note 1	Block address of I/O block or station
S8	N	2	I	Note 1	Block address of I/O block or station
S9	N	2	I	Note 1	Block address of I/O block or station
S10	N	2	I	Note 1	Block address of I/O block or station
•	•	•	•	•	•
•	•	•	•	•	•
•	•	•	•	•	•
S34	N	2	I	Note 1	Block address of I/O block or station

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S35	N	2	I	Note 1	Block address of I/O block or station
S36	N	2	I	Note 1	Block address of I/O block or station

NOTES:
1. Maximum values are:9,998 for the BRC-100200/300, IMMFP11/12
31,998 for the BRC-400 and HAC

HNET Communication
The remote I/O definition function code defines the block address of the next remote I/O definition allocated in the RIO-BRC.
The remote I/O definition block also defines the digital and analog I/O module types interfaced with the RIO-BRC.

NOTE: In order to minimize the possibility of malfunctions, it is essential that function block (FC146), function blocks (FC147), and all the slave definition blocks for the remote slaves all reside within the same segment. In addition, these blocks must be sequenced in the following order: FC146 must reside at a lower block number than all FC147s for that link, and the FC147s must reside at lower block addresses than the slave definition blocks and station blocks (FC80) for the remote slaves.

147.1Explanation

The following explanation adds detail to the specifications and shows an application using the remote I/O definition block.

147.1.1Specifications

S1
Block address of the next remote I/O definition. This should be set to zero if all the I/O for all remote processors can be defined by one remote I/O definition.

S2
Remote processor serial link communication address (if FC146 S4 = 0, expander bus communication) or spare parameter (if FC146 S4 > 0, HNET communication).

S3 and S4
Spare parameters.

S5 through S36
Block addresses of the I/O blocks.
The remote I/O definition block defines the type of I/O module blocks in the remote processor interface (or RIO-BRC). Specifications S5 through S36 point to I/O related function blocks such as the control interface slave (function code 79), digital output group (function code 83), station (function code 80), and the digital input group (function code 84). This link defines the type of I/O module each remote processor handles. If a DCS or SAC station is to be connected to a station link at the remote end (remote processor end), then function code 80 must be defined for that station in S5 through S36. Function code 147 can select up to 32 I/O blocks. Each remote processor can address up to 64 modules. Many remote I/O definition blocks may be required for each remote processor.
The analog/input slave (function code 132), the analog/output slave (function code 149), and the enhanced analog slave definition (function code 215) use a linked list of blocks with the base block being the control block. Only the control block for a multi-block I/O module definition should be defined in S5 through S36. When the control block is a remote I/O module, all associated linked list blocks are remote blocks. This includes any smart transmitter definition blocks (function code 133) linked to an analog input/slave (function code 132) or any enhanced analog input definitions function code 216) or the enhanced calibration command (function code 217) linked to an enhanced analog slave definition (function code 215). Function codes 102, 102, 104 and 109 each define only one input. The modules associated with these function codes have up to eight inputs. Each input for the IMDSM04 module is defined by one of these function codes. All function codes associated with the remote IMDSM04 module must be individually allocated a unique entry in S5 through S36.

Multiple blocks defining the same remote processor should be arranged consecutively in the linked list. This implementation results in an efficient configuration. The remote I/O definition block supports the I/O related blocks shown in Table 147-1 and Table 148-2.

Table 147-1 Function Blocks Supported by Remote I/O Definition Block (Expander Bus)

Function Code	Description	Function Code	Description
20	Indicator station	104	Pulse input/totalization
79	Control interface slave	109	Pulse input/duration
80	Control station	114	BCD input
83	Digital output group	115	BCD output
84	Digital input group	132	Analog input/slave
102	Pulse input/period	149	Analog output/slave
103	Pulse input/frequency	215	Enhanced analog slave

Table 147-2 Function Blocks Supported by Remote I/O Definition Block (HNET)

Function Code	Description	Function Code	Description
55	Hydraulic servo	109	Pulse input/duration
79	Control interface slave	114	BCD input
80	Control station	115	BCD output
83	Digital output group	132	Analog input/slave
84	Digital input group	145	Frequency counter/slave
102	Pulse input/period	149	Analog output/slave
103	Pulse input/frequency	215	Enhanced analog slave
104[1]	Pulse input/totalization	247	Condition monitoring

NOTE:

1.FC104 Pulse Input/totalization performs a reset operation when the RIO-BRC has an online configuration performed.

147.2RMP/RSP Memory Usage Calculation (Expander Bus)

The number of slaves an RMP can support is determined by memory usage only. To calculate RMP memory requirements:

1. Use the values in Table 147-3 to sum the memory required by all the remote I/O slaves.

Table 147-3 RMP Shared Memory Usage

Function Code	Description	Memory Usage (Bytes)[1]
20	Indicator station (DIS)	48
79	Control interface slave (CIS)	29
80	Control station	38
83	Digital output group (DO)[2]	14

Table 147-3 RMP Shared Memory Usage (Continued)

Function Code	Description	Memory Usage (Bytes)[1]
84	Digital input group (DI)[3]	10
102-104, 109	Pulse input[4]	11
114	Binary coded decimal (BCD) input[5]	10
115	Binary coded decimal (BCD) output[6]	14
132	Analog input/slave (ASI02/FBS)	125
147	Remote slave processor (RSP)	50
149	Analog output/slave (ASO)	65
215	Enhanced analog slave (ASI03)	174

NOTES:

1. Add the memory for each function code on the link regardless of the number of RSPs.
   2.If two DO groups or an FC83/115 pair are used on a single digital output slave, no additional memory is required (i.e., one DSO slave = 14 bytes).
   3.If two DI groups or an FC84/114 pair are used on a single digital input slave, no additional memory is required (i.e., one DSI slave =10 bytes).
   4.The value listed is for one channel. Additional channels require additional memory (e.g., 8 channels = 88 bytes).
5. If two BCD inputs or an FC84/114 pair are used on a single digital input slave, no additional memory is required (i.e., one DSI slave =10 bytes).
6. If two BCD outputs or an FC83/115 pair are used on a single digital output slave, no additional memory is required (i.e., one DSO slave = 14 bytes).

2. Subtract the sum from the total memory available in an RMP (5,552 bytes).

NOTE: The RSP has the same expander bus limitations as any master module (MFC/MFP).

147.3Applications

Figure 147-1 shows a block diagram configuration using the remote I/O interface, remote I/O definition blocks (function codes 146 and 147 respectively), and expander bus communication. Figure 147-2 shows a block diagram configuration using the RIO-BRC module, remote I/O definition blocks (function codes 146 and 147 respectively), and HNET communication.

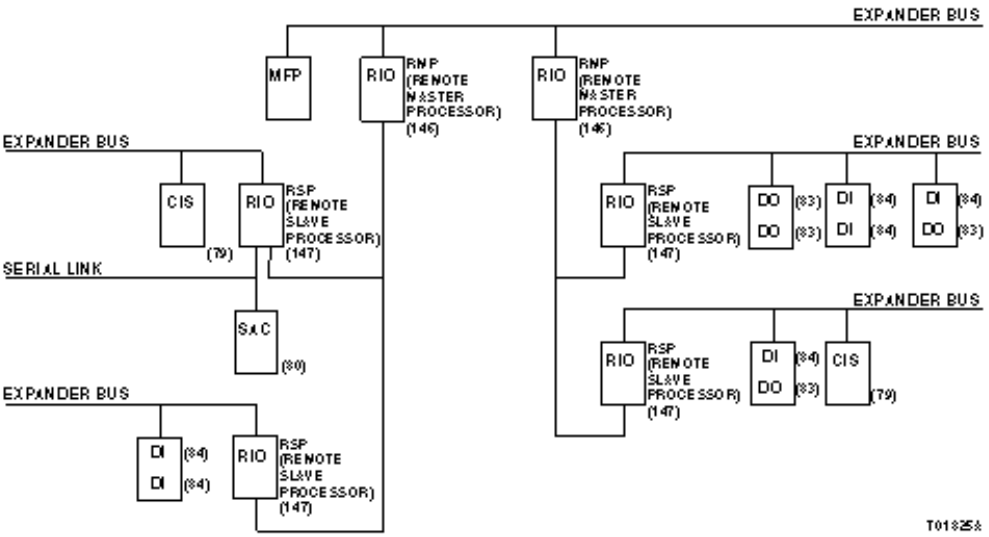


Figure 147-1 Remote I/O Definition Example (Expander Bus)

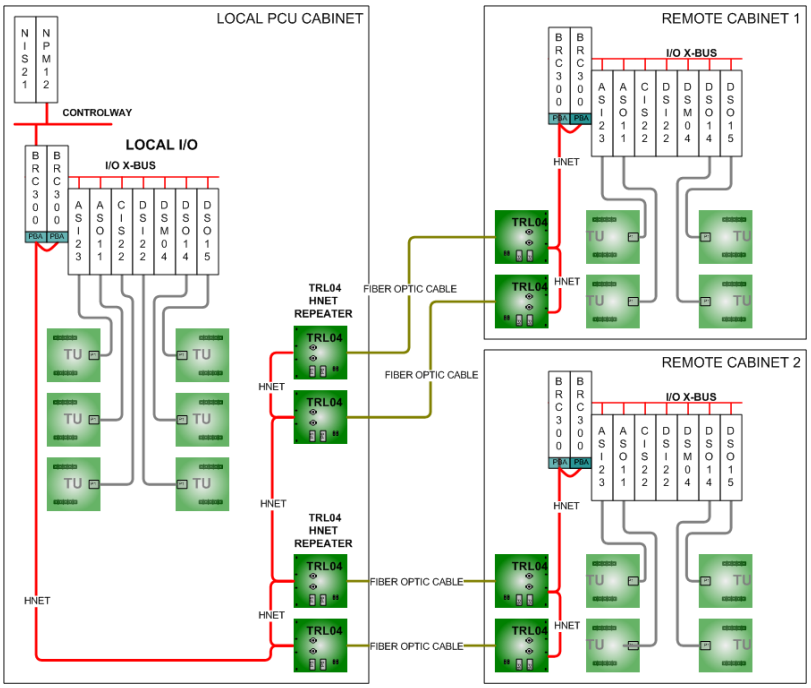
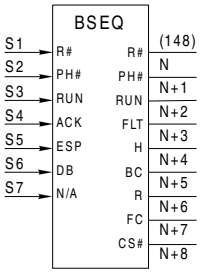


Figure 147-1 Remote I/O Definition Example (HNET)

The first example uses the remote I/O definition block to support various I/O related blocks. The remote I/O definition block defines the related I/O blocks. The remote I/O interface block defines the interface between the harmony controller and the remote I/O module. Refer to function code 146 for remote I/O interface details.
The second example is similar except that RIO-BRC modules are used. These examples are just two of many possible configurations.

148.Batch Sequence

The batch sequence (BSEQ) function code coordinates sequence activities for a batch process.



Outputs

Blk	Type	Description
N	R	Current recipe ID Number
N+1	R	Current phase number
N+2	B	Current status: 0 = hold 1 = run
N+3	B	Fault logic active: 0 = no 1 = yes
N+4	B	Hold logic active: 0 = no 1 = yes
N+5	B	Batch complete: 0 = no 1 = yes
N+6	B	Reset operator acknowledge: 0 = no 1 = yes
N+7	R	Fault code
N+8	R	Current statement number

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of recipe ID number
S2	N	5	I	Note 1	Block address of phase number
S3	N	0	I	Note 1	Block address of run/hold pushbutton
S4	N	0	I	Note 1	Block address of operator acknowledge pushbutton
S5	N	0	I	Note 1	Block address of E-STOP
S6	N	0	I	Note 1	Spare boolean input
S7	N	0	I	Note 1	Spare boolean input
S8	N	0	I	Full	Spare parameter
S9	Y	0.000	R	Full	Batch program ID number
S10	Y	1	I	Full	Debug operation
S11	N	1	I	Full	RAM allocation for object file (1 kbyte increments)
S12	N	256	I	Full	RAM allocation for data (Pos = bytes, Neg = kbytes)
S13	N	0	I	Full	Spare parameter

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S14	N	0	I	Full	Spare parameter

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

148.1Explanation

148.1.1Specifications

S1
Identifies the current recipe ID number for the BSEQ block. The recipe ID number is defined by the operator. A new recipe cannot be loaded into the BSEQ block until the previous recipe is complete or the sequence is in hold.

NOTE: Alphanumeric Recipe IDs are supported by connecting S1 of the BSEQ block to the N+1 ST output of a DATA EXPT (FC 194) or DATA IMPT (FC 193) function block. The batch program must include the #ALPHA compiler directive in order to support Alphanumeric Recipe IDs.

S2
Defines the start/restart phase number for the BSEQ block.

S3
Controls the status of the batch sequence. A zero to one transition causes the sequence to begin or resume running. A zero input causes the sequence to hold unless external hold is disabled.

S4
Operator acknowledge input. It latches in the true state by a zero to one transition. It remains true until read by an ACK statement.

S5
Block address of the E-STOP (executed stop).

S6 and S7
Spare boolean input.

S8
Spare parameter.

S9
Defines the ID number of the batch program that the BSEQ block will execute. The program ID number is determined by the last two digits of the program name or the compiler directive #PROGRAMID.

S10
Defines a debug operation. Under normal program execution requirements, use the default value (one).

1000 = transition to 1000 (from any other value) causes a debugger STOP command **S** to be executed

1001 = transition to 1001 (from any other value) causes a debugger GO command **G** to be executed

9000 = value of 9000 causes a debugger STOP command **S** to be executed whenever the program is started (complete to running) or restarted (holding to running)

S11
Controls the RAM allocation for the object file in one kilobyte increments. The program listing file *name**.LST*** shows the minimum value for this specification in its last comment lines.

S12
Controls the RAM allocation for the dynamic data. This data consists of batch, local and stack data for the active step subroutine. The program listing file *name**.LST*** shows the minimum value for this specification in its last comment lines. For

redundant configurations, the primary controller copies the entire dynamic data space to the backup. Therefore, to minimize update time, the smallest amount of memory should be allocated.

NOTE: Do not use online configuration to change the value of specification S12 in redundant configuration. Online configuration of S12 while a batch program is running forces the program state to Batch Complete.

148.1.2Outputs

The block outputs show the status of the batch sequence (BSEQ) function block. These outputs can be linked to any other block in the configuration to monitor the batch process. The output of block number N+7 is a fault code.

148.1.3Run-Time Fault Code Explanation

The BSEQ function block executes a series of diagnostic tests that detect errors that cannot be detected by the compiler. The fault codes can be seen by using the batch debugger or by viewing the BSEQ block output (N+7). The errors are detectable only while the controller is in execution and are, therefore, called run-time errors. Table 148-1 lists the possible error codes and an explanation of each.

Table 148-1 BSEQ Run-Time Fault Codes

Fault Codes	Explanation
Any positive number	Assigned (any positive number) by the user in the batch language program and is used to indicate what type of fault has occurred. There is no limit to the number of fault codes the user can assign.
-1.0 Hold command	The batch sequence is in hold, through either the BSEQ function block or a command in the batch language. Going to hold suspends normal logic and starts execution of hold logic.
-3.0 Stack overflow	Contact ABB technical support.
-4.0 Error reading object file	No batch program exists in the NVRAM memory that matches the number indicated in specification S9 of the BSEQ function block. Normally this means that the batch object file has not been downloaded to the controller, or specification S9 of the BSEQ function block references an undefined program number.
-5.0 Object file exceeds memory allocation	Batch program size exceeds the amount of controller volatile memory specified by specification S11 of the BSEQ function block. Correct this problem by increasing specification S11.
-7.0 Phase data size exceeds memory allocation	Amount of data used by a step/phase exceeds the amount of memory specified by specification S12 of the BSEQ function block. To correct, increase the value of S12 in the BSEQ function block.
-8.0 Recipe refers to undefined phase subroutine	Unit recipe contains a phase subroutine name that is not contained within the batch program. This situation can happen when a batch program is edited so that a phase subroutine is removed, but the corresponding recipes are not changed. To correct, add the undefined phase subroutine or remove the called (undefined) phase subroutine from the unit recipe.
-9.0 Batch directory error	Format the controller and reload necessary programs, recipes, and data files.
-10.0 Recipe error	Execution of a unit recipe was attempted that does not exist within the NVRAM memory of the controller. To correct, create or download a unit recipe to the controller, or input a valid recipe ID. Then restart the sequence.
-12.0 Illegal parameter type	Argument data type conflict between the unit recipe and the batch program. To correct, recompile the batch program and the recipe. Then download both the recompiled batch program and the unit recipe.
-13.0 ESTOP/ Aborting from block input	Emergency input to the BSEQ function block (specification S5) is ON. This drives the batch program unconditionally to operation 0 of the current unit recipe. To correct, find out why the emergency shutdown input is being set to ON and correct it.

Table 148-1 BSEQ Run-Time Fault Codes (Continued)

Fault Codes	Explanation
-15.0 Invalid operation number	Starting of a batch sequence was attempted at an operation number not defined within the unit recipe being run. Create a unit recipe with an operation number that matches the one to be executed, or change the operation number.
-16.0 Bad function block reference	In the batch data declaration sections of the batch language, the program is trying to reference a function block that does not exist or one whose type does not match the function code type in the declaration. The batch debugger will provide the function block number within the batch data section that is making the reference. To correct, change the function block number to a valid one, erase the reference from the program, or correct the type to match the function block in the controller.
-17.0 Array error	Array subscript is out of bounds. Normal logic is suspended and execution of fault logic begins. Note that it is possible to inspect the value of the fault code to detect when this fault has occurred.
-18.0 BCODE revision mismatch	Batch program was compiled using firmware that does not match the firmware in the controller. Recompile the batch program with the compiler that matches the firmware within the controller.
-19.0 Recipe requires too many parallel phases	Execution of a recipe was attempted that contains more parallel phase subroutines than are allowed in the target program. To correct, edit and recompile the unit recipe to contain less parallel phase subroutines, or edit the #MAXPARALLEL statement.
-20.0 Invalid number of parameters in phase data	Unit recipe contains the wrong number of recipe parameters compared to the target program. Normally, the recipe must be corrected. Otherwise, the program must be corrected.
-21.0 Invalid online program change	Execution of a new program was attempted that differs from the previous one because of a change in the batch data area or the local declaration section of the active phase subroutine. Such online changes are not permitted.
-22.0 Batch descriptor not unique	Batch and lot number in the BHIST function block are not unique to the batch historian. Change the batch and/or lot number and restart the program.
-23.0 Wait for batch historian	Batch historian is busy, and the program may not proceed until it is available. No corrective action is required.
-24.0 Batch historian offline	Batch historian is offline, and the program may not proceed until the batch historian is online and the program is restarted.
-25.0 Bad block reference in phase data	Unit recipe used contains a reference to an incorrect or nonexistent block. Correct the block number in the unit recipe.
-26.0 Bad data reference in recipe	Data entry in the unit recipe does not match the program. This most commonly happens when a unit recipe argument value was selected from a selection list, and the program was changed to no longer include that selection. Resolve any discrepancies and recompile the unit recipe with the batch program.
-27.0 Bad block reference in Unit Data	Function block declaration in the unit data file does not match the controller configuration. (Either the function block address or the function code type is in error). Resolve any discrepancies and recompile the unit data file against the batch program.
-28.0 Bad CSEQ reference in Unit Data	CSEQ reference in the unit data file does not match the program. Resolve any discrepancies and recompile the unit data file against the batch program.

Table 148-1 BSEQ Run-Time Fault Codes (Continued)

Fault Codes	Explanation
-29.0 Unit Data does not match B90 program	Unit data file does not match the batch program structurally (the number or the type of the declarations does not match). Resolve any discrepancies and recompile the unit data file against the batch program.
-30.0 Error reading UNIT.DEF file	No unit data file exists in the NVRAM memory that matches the number indicated by specification S9 of the BSEQ function block. Normally, this means that the unit data object file has not been downloaded to the controller.
-31 ID type mismatch	Recipe ID type connected to the BSEQ function block is not the same as the program file type selected by BSEQ specification S9. This error is caused by the BSEQ input specification S1 connected to a DATAEXPT (FC144) and the program referenced by specification S9 being *Numeric*, or specification S1 connected to a READ and the program referenced is #alpha.
-32 String subscript error	String position specified in the program is negative or larger than the maximum size of the string. This error occurs during program execution and transfers the program to fault logic.
-33 Restart error, Historian queue full	Hold-to-run command was received but ignored. This is due to a historian queue-full condition with a good Historian status. This state is a continuation of the Hold Command state. (Refer to fault code -1). Even though the run input may still be active, a new Hold-to-Run transition must be initiated to attempt another restart. The program will not restart unless the queue-full condition was rectified prior to the Hold-to-Run transition request, regardless of the setting in FC220, specification 9. Note that if and when the historian is marked bad, the queue is cleared.

148.2 Application

Figure 148-1 is a batch sequence example. Specification S1 links the BSEQ block to the first remote manual set constant (REMSET) block. This link gives the BSEQ block the recipe ID number. Specification S2 links the BSEQ block to the second REMSET block. This link gives BSEQ block the phase number.

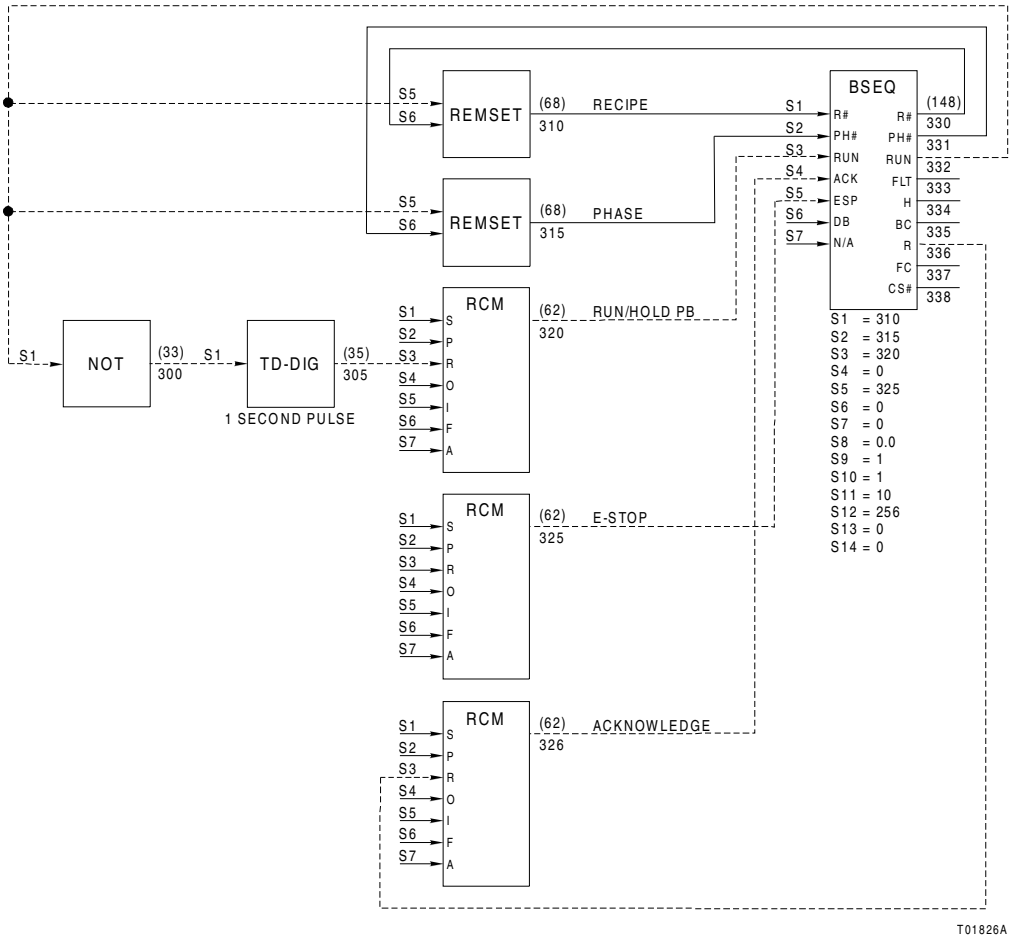


Figure 148-1 Batch Sequence Example

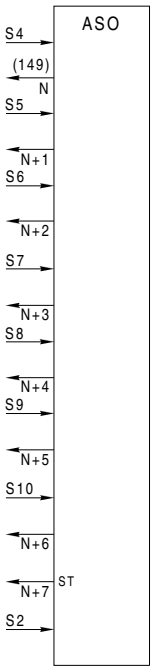
The output of the first remote control memory (RCM) block is linked to S3 of the BSEQ block. This input value describes the status of the run/hold pushbutton. The second RCM block output supplies the BSEQ block with the status of the E-STOP pushbutton.

Output N of the BSEQ block is the current recipe ID number. This output functions as a feedback signal for the first REMSET block. The N+1 output of the BSEQ block is the current phase number. This output functions as a feedback signal for the second REMSET block. The N+2 output identifies the run/hold status of the BSEQ block. If this output is one, the BSEQ block is running. If the output is zero, the BSEQ block is in hold mode.

149.Analog Output/Slave

The analog output function code writes seven analog outputs to an analog output module. Two blocks of this type may be linked together to utilize all 14 channels on the analog output module. When these function blocks are linked, only the status output of the primary block will be functional.
The analog output function block has eight outputs. The first seven are the calculated output values in percent. The eighth output is the status of the I/O module.

The control system must be carefully evaluated to establish default values that will prevent personal injury and/or property damage in case of module failure.



Outputs

Blk	Type	Description
N	R	First analog output in percent
N+1	R	Second analog output in percent
N+2	R	Third analog output in percent
N+3	R	Fourth analog output in percent
N+4	R	Fifth analog output in percent
N+5	R	Sixth analog output in percent
N+6	R	Seventh analog output in percent
N+7	R	I/O module status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	I/O module address
S2	N	2	I	Note 1	Block address of next analog output
S3	N	0	I	0 or 1	0 = trip control module on I/O module error 1 = continue to operate on I/O module error
S4	N	2	I	Note 1	Block address of output N
S5	N	2	I	Note 1	Block address of output N+1
S6	N	2	I	Note 1	Block address of output N+2
S7	N	2	I	Note 1	Block address of output N+3
S8	N	2	I	Note 1	Block address of output N+4
S9	N	2	I	Note 1	Block address of output N+5
S10	N	2	I	Note 1	Block address of output N+6
S11	N	2	I	0 - 2	Default state of output N
S12	N	2	I	0 - 2	Default state of output N+1
S13	N	2	I	0 - 2	Default state of output N+2
S14	N	2	I	0 - 2	Default state of output N+3

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S15	N	2	I	0 - 2	Default state of output N+4
S16	N	2	I	0 - 2	Default state of output N+5
S17	N	2	I	0 - 2	Default state of output N+6
S18	Y	0.000	R	Full	Engineering unit zero for output N
S19	Y	0.000	R	Full	Engineering unit span for output N
S20	Y	0.000	R	Full	Engineering unit zero for output N+1
S21	Y	0.000	R	Full	Engineering unit span for output N+1
S22	Y	0.000	R	Full	Engineering unit zero for output N+2
S23	Y	0.000	R	Full	Engineering unit span for output N+2
S24	Y	0.000	R	Full	Engineering unit zero for output N+3
S25	Y	0.000	R	Full	Engineering unit span for output N+3
S26	Y	0.000	R	Full	Engineering unit zero for output N+4
S27	Y	0.000	R	Full	Engineering unit span for output N+4
S28	Y	0.000	R	Full	Engineering unit zero for output N+5
S29	Y	0.000	R	Full	Engineering unit span for output N+5
S30	Y	0.000	R	Full	Engineering unit zero for output N+6
S31	Y	0.000	R	Full	Engineering unit span for output N+6

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

149.1Explanation

149.1.1Specifications

S1
Address of the analog output module.

S2
Block address of the next analog output block. The first block in the list is the control module block. The second block is the I/O module.

S3
Can be set to allow the module to continue to operate if there is an analog output module problem.

0 = trip control module
1 = continue to operate

S4 through S10
Block addresses of the outputs N through N+6.

S11 through S17
Define the default state of outputs N through N+6.

0 = -6.25 percent
1 = 106.25 percent
2 = hold

When power is supplied to the analog output module, the outputs are at zero percent. The default states defined by S11 through S17 are active after the multi-function controller completes initialization in execute mode. Initialization is complete when the green LED on the I/O module is on. The outputs will now enter the configured default state when a loss of the bus clock is detected. The outputs are held in their current state upon entry to configure mode.

S18 through S31
Define the low limit (EU zero) and range (EU span) of outputs N through N+6.

$$\textit{Output percent} = \frac{\textit{<Output>} - \textit{EU Zero}}{\textit{EU Span}} \times 100.0$$

If the EU span of an output is set to zero, the output is considered unused.

149.2 Applications

Figure 149-1 shows two analog output blocks linked together to utilize all 14 channels on an analog output module.

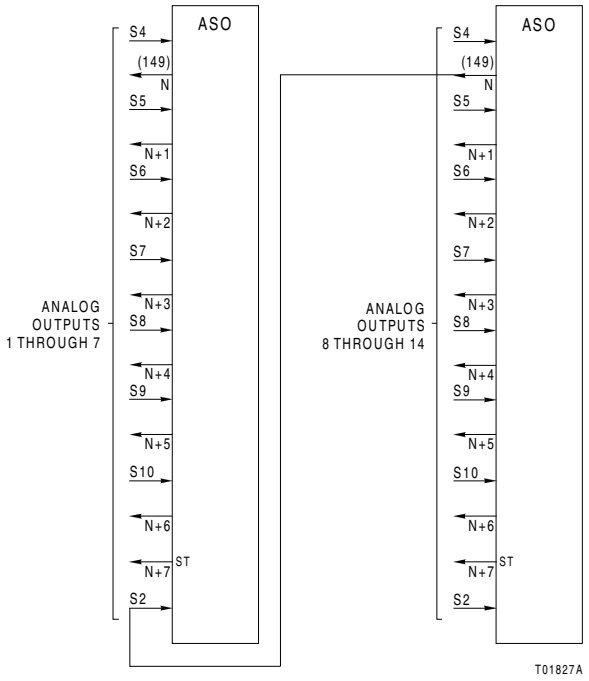
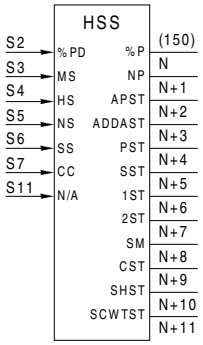


Figure 149-1 Linking Analog Output Blocks

150.Hydraulic Servo Slave

This function code interfaces a hydraulic servo slave (HSS) to a Harmony controller.
The hydraulic servo slave provides the necessary inputs and outputs to position a hydraulic actuator.
The value of the linear variable differential transformer (LVDT) differential voltages determined during the I/O module calibration cycle are stored permanently in S8 and S9 replacing the initial values or previous calibration values. On HSS module or controller restart, the calibration data from S8 and S9 is downloaded from the module to the HSS module. If necessary, new S8 and S9 values can be established via an interface device such as a HSI.



Outputs

Blk	Type	Description
N	R	Percent actuator position with quality
N+1	B	LVDT at null point: 0 = no 1 = yes
N+2	B	Actuator positioning status: 0 = good 1 = bad
N+3	B	A/D or D/A status: 0 = good 1 = bad
N+4	B	LVDT primary status: 0 = good 1 = bad
N+5	B	LVDT secondary status: 0 = good 1 = bad
N+6	B	Output 1 status: 0 = good 1 = bad
N+7	B	Output 2 status: 0 = good 1 = bad
N+8	B	I/O module mode: 0 = normal 1 = E-STOP manual
N+9	B	Calibrate status: 0 = in progress 1 = complete
N+10	B	I/O module hardware status: 0 = good 1 = bad
N+11	B	I/O module communication and watchdog timer status: 0 = good 1 = bad

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	0 - 63	Expander bus address of I/O module
S2	N	5	I	Note 1	Block address of the percent position demand

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S3	N	0	I	Note 1	Block address of calibrate mode select: 1 = calibrate
S4	N	0	I	Note 1	Block address of the calibration ***go/hold*** select: 0 = hold 1 = go
S5	N	0	I	Note 1	Block address of the LVDT null check mode select: 0 = no 1 = yes
S6	N	6	I	Note 1	Block address of the calibration stroke time selection: 1.0 = 30 secs 2.0 = 60 secs 3.0 = 35 min 4.0 = 70 min
S7	N	6	I	Note 1	Block address of the calibration cycles count (1.0 to 8.0 cycles)
S8	Y	-10.000	R	Full	LVDT differential voltage at 0% actuator position
S9	Y	+10.000	R	Full	LVDT differential voltage at 100% actuator position
S10	Y	0.000	R	Full	Spare real parameter
S11	N	0	I	Note 1	Spare boolean block input

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

150.1Explanation

The position of the actuator is measured using a linear variable differential transformer (LVDT). The HSS module provides an adjustable AC frequency excitation signal for the LVDT and demodulates the two LVDT outputs. The difference between the LVDT outputs is a measure of the actuator position. The actuator is positioned by a servo valve which moves hydraulic fluid into and out of the actuator drive cylinder. The HSS module compares the actuator position, from the LVDT, to the position demand from the module and modulates the current flow in the servo valve coils to move the actuator to the desired position. The positioning function is performed by the analog controller on the HSS module. Other hydraulic servo slave functions are performed by its on-board microprocessor. In the normal operating mode, the actuator is positioned to match the position demand from the module.

For maximum LVDT linearity, the LVDT null point should occur at the actuators mid-travel position. To check and adjust the LVDT null point, the calibrate and null check modes are selected. When the ***go*** mode is selected, the actuator ramps to and holds at the LVDT null point (LVDT secondaries have equal voltages). While the actuator is held at the LVDT null point, the mechanical zero on the LVDT can be adjusted until the actuator is at its mid-travel position. After LVDT zeroing, turn the null check mode off, select ***hold*** and proceed with calibration.

In the calibrate mode, the actuator is held at its last position until the ***go*** mode is selected. When in the calibrate ***go*** mode, the actuator moves to the 100 percent position at the selected stroke time <S6>. The HSS module drives the actuator to the 100 percent end of travel stop as verified by no further position movement and saturated servo valve coil current. The LVDT differential voltage is recorded in (S9) while the actuator is held against the 100 percent end of travel stop.

After recording the 100 percent reading, the actuator moves at the selected stroke time to the zero percent actuator position. The HSS module drives the actuator to the zero percent end of travel stop as verified by no further position movement and saturated servo valve coil current. The LVDT differential voltage is recorded in S8 while the actuator is held against the zero percent end of travel stop. After recording the zero percent reading, the actuator moves at the selected stroke time to the position demand from the module. If the number of calibration cycles (S7) to be performed is greater than one, the 100 percent to zero percent cycle will be repeated the specified number of times before ramping to the position demand <S2>.

The HSS module provides a number of status outputs. The actuator positioning status will be bad if the measured actuator position deviates beyond an established deadband from the position set point established in the HSS module. For steam turbine valve control applications, this bad status is called a ***valve contingency***. The A/D or D/A status is bad when the A/D reading of the voltage reference is outside specifications or the D/A output as read by the A/D is outside specifications.

The LVDT primary status is bad if both LVDT secondaries read approximately zero volts. LVDT secondary status is bad if one secondary reads approximately zero volts, while the other is reading a nonzero value.

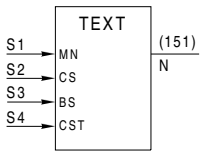
NOTE: These alarms are enabled only when the actuator position status is bad. For more information, refer to the ***IMHSS03 Hydraulic Servo*** product instruction.

The output one or two status is bad if its servo valve coil voltage indicates the coil is open or shorted. The HSS module can continue to control the actuator even if one coil is open or shorted.
The current output to the servo valve coils is set to zero if A/D or D/A status is bad, or the I/O module hardware watchdog timer times out. The output is also set to zero until the HSS module restart is complete. The servo valve will hold the actuator approximately in place if the coil current is zero. The servo valve will drift based on its mechanical biasing when the coil is zero.
If the watchdog timer (time-out condition) determines that communication with the module is lost, the HSS module reverts to and indicates E-STOP (executed stop) manual mode. In E-STOP mode, the actuator position can be changed by activating the module's raise and lower contact inputs. When communications are re-established with the module, the HSS module will not transfer to position control from the module until the module's position demand equals the present actuator position. This is to prevent an undesired actuator position change.
The HSS module is typically used in the positioning of steam turbine, throttle and control valves, gas turbine fuel valves, inlet guide vanes and nozzle angle.

151.Text Selector

Function code 151 sends message numbers and color (selected by logic) to a human system interface (HSI). The message numbers correspond to text strings configured in the HSI message list. Message number and color reporting occur using Symphony exception reporting techniques.
The text selector can operate in one of two modes. The first mode uses the message number input, color input and blink input to select the message number and attributes. The second mode uses the value of a control status input to select one of three predefined message numbers and attributes. Control status reflects the current operating state of a device as good (0.0), bad (1.0) or waiting (2.0). The text selector block references a control status output contained in function codes 123, 125, 129 and 136. This reference automatically selects the second mode of operation. A set of specifications that define the message number, color and blink are associated with each state of the device control status. The message number, color and blink are then set according to the current status of the device. Table 151-1 lists the HSI color codes for a Conductor VMS HSI.

NOTE: In the HSI tag database, the text selector tag number must be defined as type real.



Outputs

Blk	Type	Description
N	R	Message number

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of message number
S2	N	5	I	Note 1	Block address of color select
S3	N	0	I	Note 1	Block address of blink select: 0 = solid 1 = blink
S4	N	2	I	Note 1	Block address of control status: 0.0 = good 1.0 = bad 2.0 = wait
S5	N	0.000	R	0 - 65535.000	Good control status message
S6	N	0.000	R	0 - 127.000	Good control status color
S7	N	0	I	0 or 1	Good control status blink
S8	N	0.000	R	0 - 65535.000	Bad control status message
S9	N	1.000	R	0 - 127.000	Bad control status color
S10	N	1	I	0 or 1	Bad control status blink
S11	N	0.000	R	0 - 65535.000	Wait control status message
S12	N	3.000	R	0 - 127.000	Wait control status color
S13	N	0	I	0 or 1	Wait control status blink
S14	N	0	I	Full	Spare integer parameter
S15	N	0.000	R	Full	Spare real parameter
S16	Y	0.000	R	Full	Spare real parameter

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

Table 151-1 HSI Color Codes

Code	Color	Code	Color	Code	Color	Code	Color
0.0	Black	4.0	Blue	8.0	Orange	12.0	Blue-Magenta
1.0	White	5.0	Cyan	9.0	Yellow-Green	13.0	Magenta-Red
2.0	Red	6.0	Magenta	10.0	Green-Cyan	14.0	Dark Gray
3.0	Green	7.0	Yellow	11.0	Cyan-Blue	15.0	Light Gray

NOTE: The HSI may have color display limitations. Refer to the HSI instruction for possible display colors.

151.1Specifications

S4
Determines the control status. If good, message (S5), color (S6) and blink (S7) are reported. If bad, message (S8), color (S9) and blink (S10) are reported. If wait, message (S11), color (S12) and blink (S13) are reported. If the control status input (S4) is not defined, S5 through S13 are ignored and S1 through S3 specify the message number, color and blink to be reported.

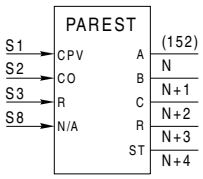
NOTE: When implementing function code 151 for Batch 90 use, S1, S2 and S3 must remain at default values. All other specifications are ignored.

152.Model Parameter Estimator

The model parameter estimator function block uses a recursive least-squares algorithm to identify a mathematical model of a process. This function block calculates the parameters for a linear, first-order dynamic model with deadtime of the specific form.

$$Y_t = -aY_{t-1} + bu_{t-k} + c$$

where:



Y_t = Value of the process variable at time t.

Y_{t-1} = Value of the process variable at one sample time before time t.

u_{t-k} = Value of the control output one process deadtime (expressed as k sample time increments) before t.

a, b, c = Model parameters.

Outputs

Blk	Type	Description
N	R	Model parameter *a*
N+1	R	Model parameter *b*
N+2	R	Model parameter *c*
N+3	R	Residual between actual and calculated process variable data
N+4	B	Quality of model parameter estimates: 0 = parameter estimator locked on 1 = new parameter estimation in progress

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of controlled process variable
S2	N	5	I	Note 1	Block address of control output
S3	N	0	I	Note 1	Block address of reset trigger (resets on a 0 to 1 transition)
S4[2]	Y[3]	0.250	R	0.25 - 9.20 E18	Sample time (secs)
S5[2,4]	Y[3]	1.000	R	0.25 - 9.20 E18	Process deadtime (secs); seconds must be < S4 × 40
S6	Y[3]	0.000	R	0.00 - 9.20 E18	Expected noise level in process variable (p-p)
S7	Y	0.000	R	Full	Spare real parameter
S8	N	0	I	Note 1	Spare boolean input

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
   31,998 for the HC800, BRC-400/410, HPG800 and HAC
2. The relationship between S4 and S5 must be valid for proper operation. If adding this block with an inferential smith controller (ISC) (function code 160) the sample time S4 must be greater than the ISC deadtime (function code 160, S8) divided by 40.
3. The initialization routine in the ISC parameter converter (function code 153) automatically specifies S4, S5 and S6.
4. Specification S5 is used only when the model parameter estimator is not used with the ISC parameter converter (function code 153).

152.1Explanation

The model parameter estimator continuously monitors the value of the controlled process variable and the control output. The value of the model parameters are calculated whenever the process behavior deviates significantly from the established parameters. The model parameter estimator outputs the value of the calculated process model parameters, the statistical residual between the actual data and the calculated model, and the status of the parameter estimator.
The model parameter estimator contains a set of heuristic rules to eliminate the practical difficulties of estimation theory. These rules prevent long-term drift of the model parameter estimates during consistent process performance, and inappropriate reaction of the model parameter estimator to external process disturbances.

152.2Specifications

S1
Block address of the process variable. This identifies the controlled process variable used by the model parameter estimator.

S2
Block address of the control output. This identifies the controller output used by the model parameter estimator.

S3
Block address of the reset trigger. When this trigger changes from zero to one, the model parameter estimator is initialized. The reset trigger also updates the ISC parameter converter (function code 153) to the default settings (process gain and process lag) stored in NVRAM of the inferential smith controller (function code 160). These settings can be updated manually by tuning the corresponding ISC specifications.

NOTE: The estimator does not stop when the loop is in manual or the process is shut down. Reset trigger must be used on startup of process.

S4
Sample time. This provides time scaling for the estimation algorithm. To assure proper operation of the model parameter estimator, the sample time should be selected so that it is between 20 percent and 50 percent of the process lag time. Because of the strong dependency of the calculated model coefficients on the selected sample time, when the sample time is changed more than ten percent or in excess of 0.5 seconds, the model coefficients are automatically initialized.

S5
Process deadtime. This defines the deadtime or transport delay exhibited by the process. Underestimation of deadtime adversely affects parameter estimation more severely than overestimation. When the model parameter estimator is linked with an ISC parameter converter (function code 153), the process deadtime is automatically updated by the ISC parameter converter.

S6
Expected process noise level. The model parameter estimator uses S6 in its identification of process upsets. This value indicates the maximum deviation from set point that can be attributed to noise in the process. The model parameter estimator treats deviations greater than this value as process upsets.

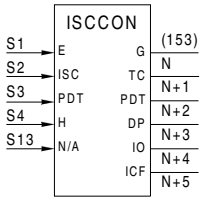
152.3Applications

The specialized function blocks required for self-tuning of the inferential smith controller (function code 160) are the model parameter estimator (function code 152) and the ISC parameter converter (function code 153). The use of an adaptive parameter scheduler (function code 154) is optional.
The model parameter estimator configuration is shown in the applications section of function codes 153 and 154. The ISC parameter converter (function code 153) application is a self-tuning configuration. The adaptive parameter scheduler

(function code 154) application is advanced self-tuning configuration with deadtime scheduling and adaptive gain and lag scheduling.
For more application information on self-tuning control, refer to the ***Self Tuning Control*** application guide.

153.ISC Parameter Converter

The ISC parameter converter function block calculates optimal tuning parameters for the associated inferential smith controller (ISC controller, function code 160) using the output of the model parameter estimator (function code 152). The tuning values are the process gain, process deadtime and process lag time. The outputs describe process dynamics at one operating point.
Through a direct link with the ISC controller, the tuning parameters may be directly adapted. However, they will only be adapted when:



- The appropriate adapt option is selected in S9.

  and

- The quality output of the associated model parameter estimator is good (zero).

The ISC parameter converter also supervises an automated initialization routine for establishing initial estimates for the associated ISC and model parameter estimator. After completion of initialization, the converter tunes:

- The ISC, gain and lag time.
- The ISC tuning time constant.
- The minimum and maximum process gain (S5 and S6).
- The minimum and maximum process lag time (S7 and S8).
- The sample time and expected noise level for the model parameter estimator.

The ISC parameter converter also updates the estimated process deadtime for the associated ISC controller and model parameter estimator whenever input S3 is connected to a function block other than number five, and the initialization trigger equals zero. The initialization value of deadtime will be used as long as the initialization trigger equals one. If S3 is set to five then, the ISC parameter converter updates the model parameter estimator with the value used by the ISC controller.

Outputs

Blk	Type	Description
N	R	Estimated process gain
N+1	R	Estimated process time constant
N+2	R	Adjusted process deadtime
N+3	R	Estimated process operating point
N+4	R	Initialization output: 0 = initialization complete, not in progress (normal) 1 = initialization failed or aborted 2 = initialization in progress
N+5	B	Pulse output from 0 to 1 and after 5 cycles, back to 0, initiated after completion of the automated initialization routine.

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of associated model parameter estimator
S2	N	5	I	Note 1	Block address of associated ISC controller
S3	N	5	I	Note 1	Block address for deadtime
S4	N	0	I	Note 1	Block address of hold signal
S5	Y[2]	-9.2 E18	R	Full	Minimum allowable value for process gain
S6	Y[2]	9.2 E18	R	Full	Maximum allowable value for process gain
S7	Y[2]	0.000	R	0.0 - 9.2 E18	Minimum allowable value for process lag time
S8	Y[2]	9.2 E18	R	0.0 - 9.2 E18	Maximum allowable value for process lag time
S9	N	0	I	0 - 3	Adapt option: 0 = no adapt 1 = adapt process gain only 2 = adapt lag time only 3 = adapt both process gain and lag time
S10	Y	0	B	0 or 1	Initialization trigger (on 0 to 1 transition)
S11	Y	5.000	R	Full	Maximum control output change for initialization
S12	Y	0.000	R	Full	Spare real parameter
S13	N	0	I	Note 1	Spare boolean input

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC
2. Value automatically specified by the initialization routine.

153.1Explanation

The ISC parameter converter function block calculates optimal tuning parameters for the associated inferential smith controller using the outputs of the model parameter estimator. Direct links between the function blocks simplify implementation.

The model parameter estimator generates the value for the process gain and process lag time. The outputs describe process dynamics at one operating point. This information directly converts to optimal tuning parameters for the inferential smith controller at this operating point using simple algebraic equations.

The tuning parameters for the ISC controller (process gain and lag time) are automatically adjusted by the ISC parameter converter as the model parameter estimator changes its estimates. However, the controller tuning time constant (S10) for the ISC controller is not automatically adjusted; this tuning time constant provides a mechanism for establishing the desired controller performance.

The ISC parameter converter also supervises an automated initialization routine for the self-tuning inferential smith controller. When the control station is set to automatic mode after the initialization trigger is changed from zero to one, the ISC parameter converter exercises the control output by a series of two step changes (in opposite directions) of a size previously established (typically ±5 percent), and monitors the reaction of the controlled process variable to estimate the process deadtime, gain and lag time. The automated initialization routine is immediately aborted if the control station for the ISC controller is set to manual mode.

After enough data has been collected to establish statistically valid estimates, the initialization routine is automatically terminated and the control station for the ISC controller is set automatically to manual mode.

The process deadtime estimated from the initialization routine is used by the model parameter estimator to determine on-line values of the process gain and lag time whenever S3 specifies block address five. If the process deadtime is externally calculated as a function of some process variable, then this value is connected to S3 and is used by the model parameter estimator and ISC controller.

The estimated values of process deadtime, gain and lag time from the initialization routine are used by the ISC parameter converter to automatically establish the initial values of a number of other specifications:

- Minimum process gain (S5) is set to 50 percent of the initial process gain observed during the initialization routine.
- Maximum process gain (S6) is set to 200 percent of the initial process gain observed during the initialization routine.
- Minimum process lag time (S7) is set to 50 percent of the initial process lag time observed during the initialization routine.
- Maximum process lag time (S8) is set to 200 percent of the initial process lag time observed during the initialization routine.
- Controller time constant for the inferential smith controller (function code 160, S10) is set to 100 percent of the initial process lag time observed during the initialization routine.
- Sample time for the model parameter estimator (function code 152, S4) is set to 20 percent of the process lag time observed during the initialization routine.
- Expected noise level for the model parameter estimator (function code 152, S6) is set based on the peak to peak value of the noise on the controlled process variable observed during the initialization routine for constant valve position.

The initialized specifications can be manually changed after the initialization routine is complete. However, they should be valid for most applications.
Upon completion or failure of the automated initialization routine, the ISC parameter converter automatically returns the control station for the ISC controller to manual mode, and sets the value of the appropriate controller specifications. At this point, the initial settings can be monitored and validated before they are actually used by the controller. The initialization trigger must be manually set to zero for normal operation.

Specifications

S1
(Block address of associated model parameter estimator) Establishes the link between the ISC parameter converter and the associated model parameter estimator. The ISC parameter converter obtains the estimated value of the process model parameters and the status of the estimates through this link. The process deadtime for the model parameter estimator is updated through this link.

S2
(Block address of associated inferential smith controller) Links the ISC parameter converter with the associated ISC controller. Updating of the ISC controller tuning parameters and the process deadtime occur through this link.

S3
(Block address for process deadtime) Locates the value of the deadtime the ISC controller and model parameter estimator use. If the deadtime is not predicted as a function of a process variable, use the default address to permit the process deadtime setting of the ISC controller to be used by the ISC parameter converter.

S4
(Block address of hold signal) Identifies a hold switch for the ISC parameter converter. If the value of this switch is set to one, parameter conversion continues but the tuning parameters of the ISC controller are not automatically updated. New values for the calculated tuning parameters are available at the block outputs whenever the status of the model parameter estimator indicates *parameter estimator locked on*. When the estimator status indicates *new parameter estimation in progress*, the tuned ISC parameter will be displayed and the operator can manually tune the ISC controller. If the value of the switch is set to zero, the ISC parameter converter automatically tunes the parameter of the ISC specified by the adapt option. The operator can manually tune the ISC controller when the status from the model parameter estimator indicates *new parameter estimation in progress*.

S5 through S8
(Minimum and maximum tuning parameters) Required for commissioning of the self-tuning ISC controller and to increase the fault tolerance of ISC controller operation. Minimum and maximum values are preset by the automated initialization routine, but can be adjusted to match the process. In the event that the ISC parameter converter generates values for the tuning parameters outside of the previously specified constraints, the tuning parameters for the controller are limited to the constrained values.

S9
(Adapt option) Permits selection of self-tuning for either or both controller tuning parameters. If using the adaptive parameter scheduler (function code 154), the scheduled parameters should not be selected for self-tuning with this specification.

S10
(Initialization trigger) Provides the trigger for the automated initialization routine. When the trigger changes from zero to one and the ISC controller is in manual mode, the initialization routine is activated. As a safeguard, the station associated with the ISC controller must then be placed in automatic mode for initialization to proceed. The routine is automatically terminated when adequate data has been generated for process identification. As long as the initialization trigger equals one, the process deadtime will be that estimated by the initialization routine. This value can be changed manually by tuning the ISC controller. The model parameter estimator is automatically updated to this value. When the initialization trigger is set equal to zero, <S3> (block address for process deadtime) is utilized for process deadtime if S3 does not equal five.

S11
(Maximum control output change for initialization) Establishes the maximum change from the manually set valve position to be permitted during the automated initialization routine. Either a positive or negative step change can be specified.

153.2 Applications

The specialized function blocks required for self-tuning of the inferential smith controller are the model parameter estimator (function code 152), ISC parameter converter (function code 153) and the smith predictor (function code 160).
Figure 153-1 shows a basic self-tuning configuration. For more application information on self-tuning control, reference the ***Self Tuning Control*** application guide.

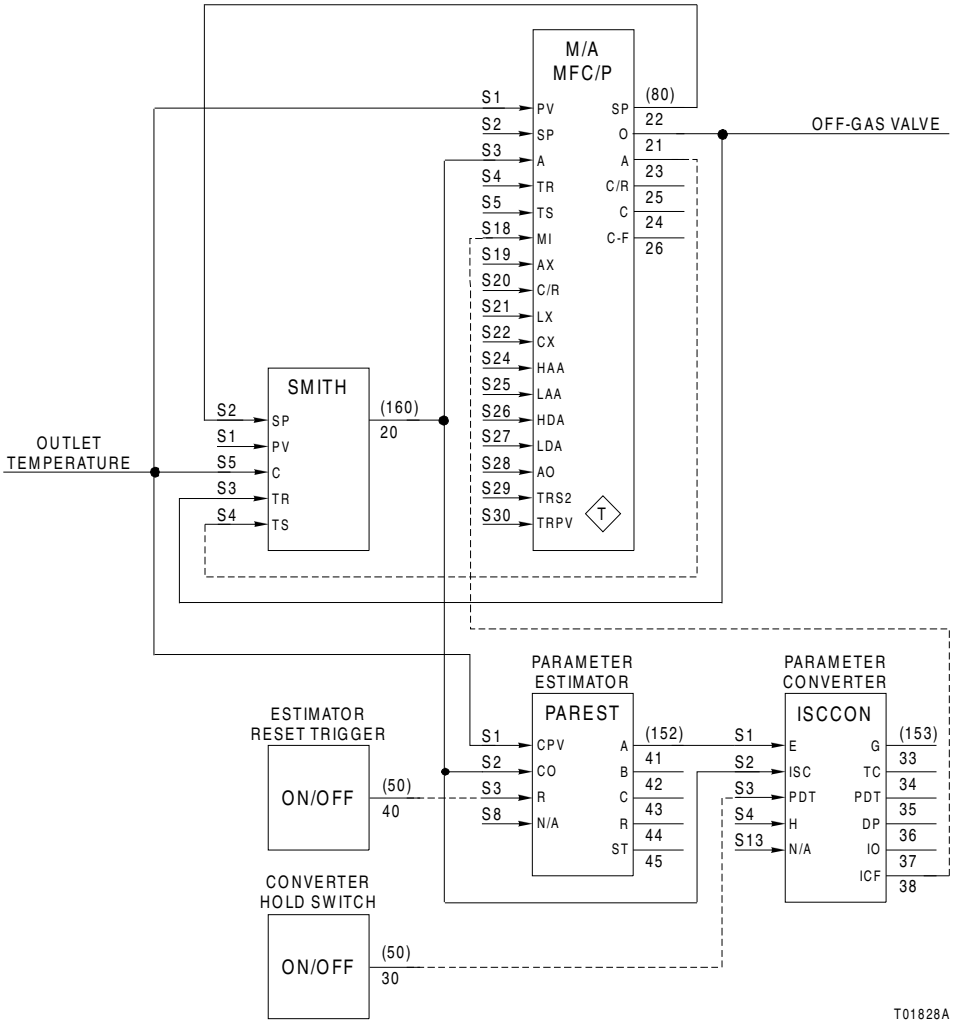
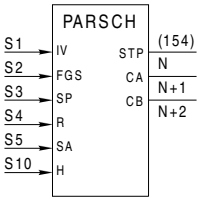


Figure 153-1 Basic Self-Tuning Configuration

154.Adaptive Parameter Scheduler

The adaptive parameter scheduler function allows process characteristics such as a measured or calculated index variable to be used to adjust the tuning parameters for the associated inferential smith controller (ISC function code 160). This feature optimizes controller performance for predictable changes in process operation and prevents periods of potentially unacceptable control while the ISC controller is being retuned by the model parameter estimator (function code 152) via the ISC parameter converter (function code 153).
The adaptive parameter scheduler can automatically establish the relationship between an ISC tuning parameter and a measured or calculated index variable using linear regression. The adaptive parameter scheduler uses this relationship to automatically adjust the specified ISC tuning parameter based on the value of the specified index variable.
Alternatively, this function can automatically determine the correction bias required for a pre-established gain schedule. This permits a nonlinear relationship between the ISC tuning parameter and the index variable, with automatic correction of the relationship for design inaccuracies and changes in process behavior.



Outputs

Blk	Type	Description
N	R	Scheduled tuning parameter
N+1	R	Coefficient A of correction equation
N+2	R	Coefficient B of correction equation

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of index variable
S2	N	5	I	Note 1	Block address of fixed gain schedule
S3	N	5	I	Note 1	Block address of scheduled parameter
S4	N	0	I	Note 1	Block address of reset trigger
S5	N	5	I	Note 1	Address block containing specification to be adapted
S6	N	0	I	0 - 255	Specification to be adapted
S7	N	0.000	R	Full	Minimum index value
S8	N	0.000	R	Full	Maximum index value
S9	Y	0.000	R	Full	Spare real parameter
S10	N	0	I	Note 1	Block address of coefficient update hold: 0 = update A and B 1 = hold updates of A and B

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

154.1Explanation

When a process controlled by an inferential smith controller (function code 160) shifts from one operating point to another, the inferential smith controller (ISC) is automatically retuned to maintain the desired controller performance at the new operating point. However, during self-tuning, the ISC controller performance can be temporarily less than desirable. In applications where the specific value of an ISC tuning parameter (process gain or lag time) is related to some process variable or discrete event (an index variable), these periods of suboptimum controller performance during self-tuning can be eliminated by adaptive scheduling of the tuning parameter.
The adaptive parameter scheduler utilizes a least-squares technique to automatically correlate a preselected index variable with one controller tuning parameter output by the ISC parameter converter. Once an effective linear correlation has been established, the adaptive parameter scheduler adjusts the tuning parameter for the ISC controller as a function of this index variable. If more than one tuning parameter must be scheduled, more than one adaptive parameter scheduler must be used.

The adaptive parameter scheduler utilizes a bin data structure for regression of the linear relationship between the index variable and the correction bias. The range of the index value is divided into ten bins, and when a valid data set becomes available, it goes into the bin corresponding to the value of the index variable for the data set. Only one data point is stored in each bin. As new data becomes available for a bin, the old data is replaced and the regression is recalculated. To facilitate commissioning of the adaptive parameter scheduler when there is only one data set, a line passing through the data point with zero slope is assumed.

Scheduled tuning parameter =
Output from fixed gain schedule <S2> + correction bias
Correction bias = Ax + B

where:

x = Value of the index variable <S1>.

A and B coefficients are updated by the regression algorithm.

154.1.1 Specifications

S1
(Block address of index variable) Identifies the index variable used by the adaptive parameter scheduler.

S2
(Block address of fixed gain schedule) Identifies the output of the associated fixed gain schedule. If not using a pre-established gain schedule, this specification should be set to block address five (default value), which provides a constant value of zero.

S3
(Block address of scheduled parameter) Identifies the estimated value of the scheduled tuning parameter. This value determines the relationship between the tuning parameter and index variable. The instantaneous correction bias (S2-S3) is used with the index variable (S1) as a data point set for regression determination of A and B.

S4
(Block address of reset trigger) Identifies an external trigger used to reset the regression data. When the trigger changes from zero to one, all historic data used for determining the correlation equation is erased and the correction bias is set to zero.

S5
(Address of block containing parameter to be adapted) Identifies the block address for the parameter adjusted by the adaptive parameter scheduler.

S6
(Specification to be adapted) Identifies which specification of the identified block is adjusted by the adaptive parameter scheduler.

S7 and S8
(Minimum and maximum values for the index variable) Define the allowable range for the index variable.

S10
(Block address of coefficient update flag) Allows suspension of the recalculation of the A and B coefficients. The correction bias will still be computed and the output updated. Also, the parameter in the target block and specification continue to update.

154.2 Applications

The specialized function blocks required for self-tuning of the inferential smith controller are the model parameter estimator (function code 152), ISC parameter converter (function code 153), and the smith predictor (function code 160). The use of the adaptive parameter scheduler (function code 154) is optional.

Figure 154-1 shows an advanced self-tuning configuration with deadtime scheduling and adaptive gain/lag scheduling. For more application information on self-tuning control, reference the ***Self Tuning Control*** application guide.

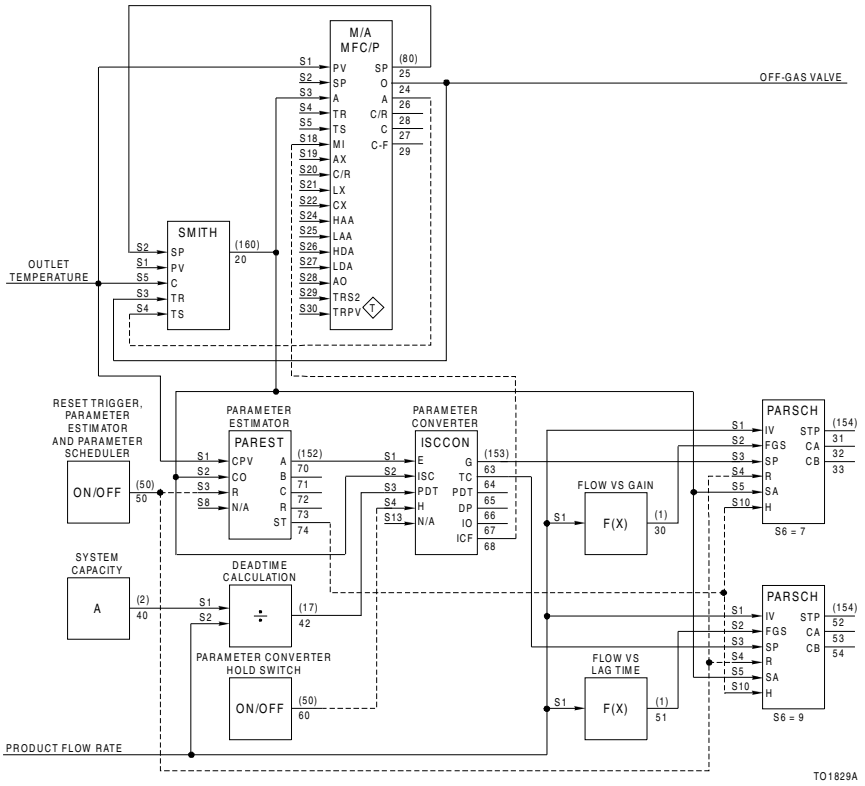
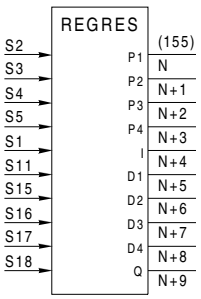


Figure 154-1 Adaptive Parameter Scheduler Example

155.Regression

The regression block correlates up to four independent variables to a single dependent variable. Data can be collected on a time or trigger basis, and buffered either sequentially or in bins. The size of the data buffer is configurable.
A ***goodness of fit*** is specified. This output disables the updating of parameter estimates when a mismatch between the collected data and the estimated curve is beyond the specified goodness of fit.
A reset input provides the ability to suspend the start data collection when flagged.
Calculation and edit are the two operating modes. In both modes, the first four outputs are dedicated to the computed coefficients. The remaining outputs are dependent on the mode. The calculation mode outputs information about the current calculation. The edit mode identifies the inputs to the calculation and allows the operator to change the quality of a row of data in the regression matrix.



Outputs

Blk	Type	Description	
		Calculate Mode	**Edit Mode**
N	R	Parameter 1	Parameter 1
N+1	R	Parameter 2	Parameter 2
N+2	R	Parameter 3	Parameter 3
N+3	R	Parameter 4	Parameter 4
N+4	R	Goodness of fit	Dependent variable, (y)
N+5	R	Maximum model mismatch	First independent variable, x_1
N+6	R	Row no. producing maximum model mismatch	Second independent variable, x_2
N+7	R	No. of rows with good quality	Third independent variable, x_3
N+8	R	Time of last computation in mmddhh format	Fourth independent variable, x_4
N+9	B	State of outputs: 0 = computed 1 = default	Quality of data point: 0 = bad, excluded from computation 1 = good, included in computation

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of dependent variable
S2	N	5	I	Note 1	Block address of first independent variable
S3	N	5	I	Note 1	Block address of second independent variable
S4	N	5	I	Note 1	Block address of third independent variable
S5	N	5	I	Note 1	Block address of fourth independent variable
S6	N	1	I	1 - 4	Number of independent variables to use in calculation
S7	N	5	I	5 - 32	Number of sets of data to buffer
S8	N	5	I	5 - 32	Minimum number of good sets of data required for calculation
S9	N	1	B	Full	Trigger or time mode flag: 0 = trigger 1 = time
S10	N	1.000	R	0.0 - 9.2 E18	Time interval between calculations (minutes)

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S11	N	0	I	Note 1	Block address of external trigger
S12	N	0	B	Full	Data storage mode flag: 0 = sequential 1 = bins
S13	N	100.000	R	Full	High range of first independent variable for bin storage
S14	N	0.000	R	Full	Low range of first independent variable for bin storage
S15	N	0	I	Note 1	Block address of edit mode switch: 0 = calculate 1 = edit
S16	N	5	I	Note 1	Block address of set to edit value
S17	N	0	I	Note 1	Block address of set quality toggle flag: 1 = toggle quality
S18	N	0	I	Note 1	Block address of reset flag: 1 = reset
S19	Y	0.000	R	Full	Initial default for first parameter
S20	Y	0.000	R	Full	Initial default for second parameter
S21	Y	0.000	R	Full	Initial default for third parameter
S22	Y	0.000	R	Full	Initial default for fourth parameter
S23	Y	1.000	R	0.000 - 1.000	Desired goodness of fit
S24	N	0.000	R	Full	Default update
S25	N	0.000	R	Full	Full spare
S26	N	0.000	R	Full	Full spare
S27	N	0.000	R	Full	Full spare

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

155.1Explanation

The regression block has two modes of operation. First is calculation of parameter estimates, and second is editing of data contained in the data table. Specification S15 selects the mode of operation (the edit mode switch). Setting S15 to zero selects the calculation mode and one selects the edit mode.

In the calculation mode, the regression block stores the measurements x_1, x_2, x_3, x_4 (independent variable) and y (dependent variable) in a data table. The matrix X and the vector Y represent this data table as shown below. Each row of the matrix and the corresponding element in Y contain data from one sampling period.

$$a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} \qquad X = \begin{bmatrix} x_{11} & x_{21} & x_{31} & x_{41} \\ x_{12} & x_{22} & x_{32} & x_{42} \\ \neg & \neg & \neg & \neg \\ x_{1N} & x_{2N} & x_{3N} & x_{4N} \end{bmatrix} \qquad Y = \begin{bmatrix} y_1 \\ y_2 \\ \neg \\ y_N \end{bmatrix}$$

where:

a_n = The values of parameter n where n = 1 to 4

X	=	Matrix of input values for independent variables. Each column contains the group of samples for one of the four independent variables; x_{1N} = values for independent variable x_1, etc.
Y	=	Matrix of values for the dependent variable. The number of rows is the number of samples taken.

The regression algorithm solves the equation *Xa=Y*. If the number of samples (rows in *X*) equals the number of parameters to find (columns in *X*), creating a square matrix, the solution is *a=X-1Y*. However, the matrix *X* is not always invertible. If the rows of *X* are not unique, the matrix is singular and the inverse does not exist. The internal logic of the regression block prevents entry of data that creates a singular matrix.
When collecting live data, there is always uncertainty in the values collected, resulting from the influence of uncontrollable effects in the surrounding environment. To counteract this influence, more data points are collected to increase confidence in the model parameters. When this is done, the matrix *X* is not square. This leaves more equations than unknown parameters to specify, and the simple algebraic solution explained above is not possible.
Rearranging the equation *Xa=Y* gives *X(a–y)=r* where *r* is the vector of residuals. Generally, any *a* selected leaves a non-zero vector of residuals, indicating the mismatch between model and data. To solve this problem, the regression block uses the least squares method to minimize the square of the residuals. The solution takes the form *X´(X(a–Y))=0*. This is a set of linear equations, solved by the Gaussian Elimination method. This method provides numerically stable solutions while requiring less processing time than more direct solution techniques.
A minimum number of sets of data with good quality must be present in the data table before the parameters may be calculated. Specification S8 specifies the minimum number. But, the minimum number must be equal to or greater than five. The data set can be viewed and changed in the edit mode. Each time a good quality data set is entered, the values of a_1 through *a4* are recalculated.
If the calculation of *a* is valid, and the goodness of fit is less than that specified with S23, then the values of *a1* through *a4* are output from the block. The goodness of fit is defined as the mean relative residual:

$$GF = \sum_{j=1}^{J} \frac{\left| \left(\sum_{k=1}^{K} a(k) \times X(j,k) \right) - y(j) \right|}{max\,(1, |y(j)|) \times J}$$

where:

J	=	Number of independent variables used in the calculation (S6).
K	=	Number of data sets used in the calculation (S7).
a(K)	=	Value of *a* determined when k data sets are used for calculation.
X	=	Matrix of input values for independent variables.
y(j)	=	Value of the dependent variable associated with independent variable number j.

The block also performs a test on residuals *r* after accepting new data. The old set of data is always buffered for the duration of the calculation, and replaces the new set of data in the event that the block is unable to calculate valid parameters. When the computed goodness of fit is greater than the tolerance limit (S23), the new data set is removed from the data table *X* and the old data is reinstated.
Data can be collected in two ways (time basis and transition of external trigger). Specification S9 selects the mode. If data is collected on a time basis, the collection frequency is specified in minutes with S10.
The data can be stored in the data table in one of two ways. Data can be stored either in bin mode or sequentially. Specification S12 selects the mode.

S12 = one = bin storage

The bin mode of data storage allows the system to maintain a spread of data over a range of the independent variable *X*1. The bin mode of data storage should be used when the correlation is not expected to change (e.g., due to sensor contamination or independent variable changes over a wide range). In this mode the data is sent to the appropriate bin and calculated as shown:

$$Bin\ No. = S7 \frac{<S2> - <S14>}{<S13> - <S14>}$$

where:

<S2>	=	First independent variable.
S7	=	N, the number of data sets used for calculation.
<S13>	=	High end of range of allowable *X1* values.

<S14> = Low end of range of allowable *X1* values.

Any input greater than S13 or less than S14 is discarded. In the sequential mode of data collection, the data table is first in first out (FIFO queue). The new points fill the data table row by row, and the newest data set replaces the oldest data set. This mode is used when the correlation is expected to change.

In the edit mode, a set of data from the data table can be selected and displayed by entering a number between one and N as S16. The set of data selected is then available in output blocks N+4 through N+8. Block N+9 contains a quality bit that indicates if the data selected is currently in use in the calculation.

0 = good data
1 = bad data

The operator can change the quality associated with a set of data by toggling S17 to one. By changing the quality associated with an erroneous set of data as bad, it is eliminated from the parameter calculation.

Default values for each of the four parameters are specified with S19 through S22. The default values can be periodically updated from the data tables by selecting the update time in hours with S24. If S24 is set to 0.0 there is no updating of the default parameters. The minimum update time is 18.0 hours. The default update is an important feature because the data table is stored in RAM and is lost on power down, module reset, or entering configuration mode. The default parameters are stored in NVRAM which is not affected by these interruptions of normal operation. Thus, when the module is started, real values are available. The default parameters are output after start-up, and until there are the specified number of good quality data sets (S9).

A reset input is also available. If it is set to one it marks all sets of data in the table to bad and makes the default parameters S19 through S22 available at the output to the block.

155.1.1 Specifications

S1 – Y
Block address of dependent variable.

S2 – X1
Block address of independent variable X1.

S3 – X2
Block address of independent variable X2.

S4 – X3
Block address of independent variable X3.

S5 – X4
Block address of independent variable X4.

S6 – J
Number of independent variables (one to four) used for calculation. Select the number of variables from one to four used in the calculation.

S7 – K
Number of sets of data used for calculation. This identifies the number of sets of data to be drawn from to perform the calculation. There can be up to 32 sets.

S8 – MD
Minimum number of good sets of data required for calculation. The minimum number of good data sets required to perform the calculation is five.

S9 – MD1
Time and trigger mode flag. This specification defines the mode of data collection used. In the time mode, data is collected at a fixed interval of time specified with S10. In the trigger mode, data is collected each time the externally controlled collection trigger (S11) goes to one.

0 = trigger mode
1 = time mode

S10 – DT
Time in minutes between collections of data when the regression block is in the time collection mode (S9 equals one).

S11 – ET
Block address of the external collection trigger. This input determines when collections of data occur in the trigger mode (S9 equals zero). When this input makes a zero to one transition, the block reads the incoming data.

S12 – MD2
Data storage mode flag. This specification defines the data collection mode. In the bin mode, the system maintains a spread of data over a range of the independent variable X1,(S2). In the sequential mode, the newest set of data replaces the oldest set of data in the data table.

0 = sequential
1 = bin

S13 – HR
High end of the range of X1 for bin storage. If there is data stored in the bin mode, any input values greater than this number are discarded. If data storage is in the sequential mode, retain the default value.

S14 – LR
Low end of the range of X1 for bin storage. If storing data in the bin mode, input values less than this number are discarded. If storing data in the sequential mode, retain the default value.

S15 – MD3
Block address of calculate and edit mode switch. This value controls the operating mode of the regression block.

0 = calculate mode
1 = edit mode

S16 – EDN
Block address of the number of data sets from one to n viewable in the edit mode. This specification is only activated in the edit mode (S15 equals one). When in edit mode, the variables in the set selected with S16 output to blocks N+4 through N+8.

S17 – SQ
Block address of the quality switch. This specification is active only in edit mode. When <S17> changes from zero to one, the quality value of the row S16 specifies changes to the opposite quality. Good quality can be forced bad or bad quality, likewise, can be forced good.

1 = change quality

S18 – RS
Block address of the reset switch. When this value goes to one, all rows in the data table are marked bad quality, and the default parameter values from S19 through S22 are output from the block.

1 = reset
0 = normal

S19 – D1
Initial default value for parameter a_1. If S24 is not equal to zero, the calculated value replaces the initial value at the interval specified with S24. If S24 equals zero, S19 equals default value.

S20 – D2
Initial default value for parameter a_2. If S24 is not equal to zero, the calculated value replaces the initial value at the interval specified with S24. If S24 equals zero, S20 equals default value.

S21 – D3
Initial default value for parameter a_3. If S24 is not equal to zero, the calculated value replaces the initial value at the interval specified with S24. If S24 equals zero, S21 equals default value.

S22 – D4
Initial default value for parameter a_4. If S24 is not equal to zero, the calculated value replaces the initial value at the interval specified with S24. If S24 equals zero, S22 equals default value.

S23 – GF
Desired goodness of fit parameter. If the calculated values are not less than this value, they will not be output from the block. The calculated values will be discarded and the last set of successfully calculated values will be output. This input can be used to reject noisy data.

S24 – DEFUP
Default update period. At the end of this time, the calculated values of the parameters a_1 to a_4 are copied to the default parameters. The minimum update period is 18 hours.

$S24 = 0$, no update of default values.

$S24 \neq 0$ and ≥ 18, the default values of the parameters will update at the end of the update period.

S25 to S27
Spare.

155.1.2 Outputs

N
Value of the first calculated parameter in both calculation and edit modes.

N+1
Value of the second calculated parameter in both calculation and edit modes.

N+2
Value of the third calculated parameter in both calculation and edit modes.

N+3
Value of the fourth calculated parameter in both calculation and edit modes.

N+4
Calculation Mode

$$\textit{goodness of fit} = \sum \frac{\left|\frac{(Y_i - Y_i)}{Y_i}\right|}{n}$$

Edit Mode
Value of dependent variable *Y*.

N+5
Calculation Mode

$$\textit{maximum model mismatch} = \textit{MAX}\left(\frac{(Y_i) - Y_i}{Y_i}\right)$$

Edit Mode
Value of first independent variable *X1*.

N+6
Calculation Mode
Row number of maximum mismatch.
Edit Mode
Value of second independent variable *X2*.

N+7
Calculation Mode
Number of data rows with good quality.
Edit Mode
Value of third independent variable *X3*.

N+8
Calculation Mode
Time of last successful computation in mmddhh format with hours in military time.
Edit Mode
Value of fourth independent variable *X4*.

N+9
Calculation Mode
State of outputs:

1 = computed

0 = default; when the module is reset, all values in the data table are marked bad quality and the default values specified by S19 through S22 are output.

Edit Mode

Quality of the current data set (selected with S16):

1 = good quality included in computation
0 = bad quality excluded from computation

The current quality can be changed by setting S17 to one. This toggles the quality input to the opposite value.

155.2Applications

The regression block can be used for economic optimization. It operates on functions described as linear, which means *y* is a linear function of *a*. This does not imply that *y* is a linear function of the measurements forming *X*. For instance, to identify the cost function of a steam generating unit, a quadratic form is employed.

y = cost
X1 = steam flow
X2 = steam flow^2
X3 = 1

Y = a(1)X1 + a(2)X2 + a(3)1

The equation provides a steady state economic model used by an optimization program to minimize operating expenses. Another application is modeling of the kinetic parameters in a batch reactor.

extent = a(1) x (time,temperature,concentration) + a(2)

The extent of the reaction is a laboratory measurement, and *f(t,T,C)* is a dimensionless group representing relative reaction rate computed by the module. The lab data is entered through the console or control station. The identification of the parameters allows on-line prediction of required batch reaction time, given measurements for temperature and initial concentration.

The two preceding applications represent models as power series. Linear regression can also compute model parameters for more complex function forms. The regression block correlates up to four independent variables to a single dependent variable.

For instance, the model $m = b(1) \times (pb(2)) \times (qb(3))$ contains two independent variables in a nonlinear relationship. The following equation results after taking the log of both sides, making the model linear in the parameters:

$$\log(m) = \log(b(1)) + b(2) \log p + b(3) \log(q)$$

making the definitions:

$$y = \log(m),\ X1 = 1,\ X2 = \log(p),\ \text{and}\ X3 = \log(q)$$

The regression block finds the best parameter set *a* for the equation:

$$y = a(1)X1 + a(2)X2 + a(3)X3$$

Figure 155-1 shows the contours of equal *m* plotted in the *p – q* plane.

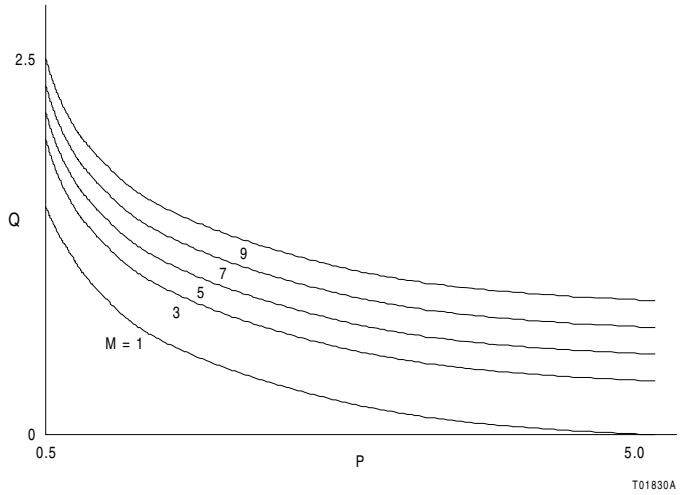


Figure 155-1 Contours of Equal m Plotted in the p – q Plane

The type of data storage used depends on the situation. Sequential storage retains the last seven data sets and calculates the parameters from them. Use this mode when the correlation is expected to change. Bin storage retains the data sets that give evenly spaced sets across the entire range of the independent variable. It should be used whenever the correlation is not expected to change due to such things as sensor contamination. Bin storage should also be used when the independent variable changes over a wide range, but is not expected to assume all or nearly all of the range of values. For example, a machine that commonly runs at 60 to 80 percent load for extended periods of time would run most efficiently with bin storage. Bin storage retains values that fall within zero to 60 percent and 80 to 100 percent load while frequently updating the fall between values 60 to 80 percent load.

155.2.1 Regression Block Application Considerations

The regression block is very flexible. Determine which combination of data collection and storage techniques is needed for the application. A balance must be maintained when setting data collection, storage and acceptance specifications. More certainty and stability in the calculated coefficients is generally obtained at the expense of speedy adaptation to significant changes in process behavior.

Number of Data Sets

The specification of 32 data sets in the data table (the maximum for S7) provides more information about the process and stabilizes the value of calculated model coefficients. However, specification of 16 data sets would speed up adaptation of the coefficients if process behavior was expected to change rapidly.

Binned Data Storage

Segregating the data into bins according to the value of *x1* (at S2) significantly increases the reliability of the calculated model coefficients. The bins insure that data collected over the entire range of interest for *x1* is included in the coefficients, not just the most recent data. The most recent data may be concentrated around a long term operating point of the process, and the coefficients calculated from this data may not be representative of the process outside this operating point.

Figure 155-1 shows the effect of the data storage technique. This example shows two models of sampling 20 points of a known function. The function was distorted by the addition of random noise. The first model uses bins for data storage, the second model uses sequential data storage. The first model adequately represents the actual function over the entire range of interest while the second only represents the function in the local range of the data collected.

The increased reliability provided by binned data storage comes at the price of slower adaptation to process changes. If some data points are collected at operating ranges entered rarely, they corrupt the currentness of the curve fit. Narrowing the collection range speeds the adaptation process, but renders the coefficients inaccurate outside the range of data collection.

Maximum Residual

The maximum residual specified in S23 also has a strong effect on the ability of the regression block to adapt and its rate of adaptation. To reject noisy data, a small residual is desirable. However, if the residual is set too small, all new data will be rejected. To give the regression block some pliancy, a larger residual must be specified.

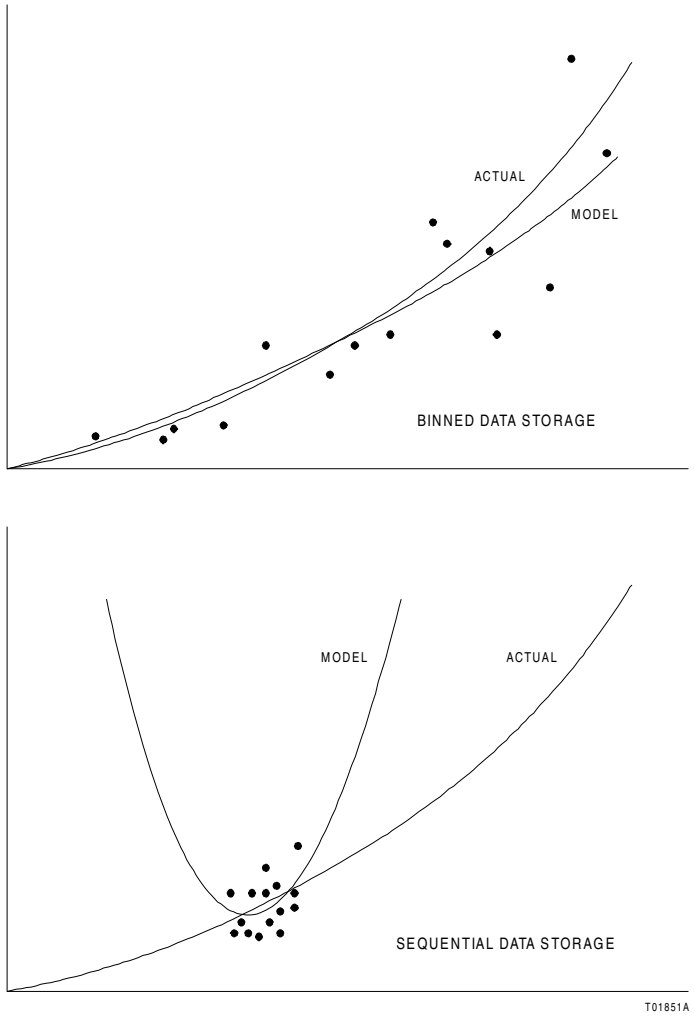
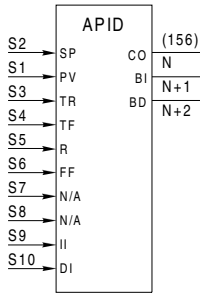


Figure 155-1 Effect of Data Storage Technique

156.Advanced PID Controller

The advanced PID controller function code implements a proportional integral derivative controller. Some of the advanced features of this function code above the other PID controllers (function codes 18 and 19) are:



- Direct use of a feedforward signal into a PID controller.
- Improved algorithm for derivative action calculation.
- Improved algorithm for incorporating an external reset or manual reset signal.
- Control output increase and decrease inhibit signals to constrain controllers in cascade configurations when limits are encountered. This prevents the primary loop from wind-up when the secondary loop control output saturates.
- Provides digital implementation of a noninteracting PID control algorithm or a classical (analog) series interacting PID controller.
- Provides a quick saturation recovery option.

In addition to the advanced features, function code 156 has the normal features of a PID, including:

- Bumpless manual-to-auto transfer.
- Bumpless proportional band tuning.
- Anti-reset (integral) wind-up function.
- Reverse acting or direct acting direction switch.
- Set point modifier option allowing bumpless set point changes.

NOTE: PID reset mode (S5) and PID gain (S6) of the segment control block (function code 82) do not affect the advanced PID controller. Maximum derivative gain for PID (S11) and external reset for PID (S12) of the executive block (function code 53) do not affect the advanced PID controller. These functions are controlled within the advanced PID controller. This feature allows PID controllers with and without external reset to be included in the same segment.

Outputs

Blk	Type	Description
N	R	Control output with feedforward
N+1	B	Block increase flag: 0 = permit increase 1 = inhibit increase
N+2	B	Block decrease flag: 0 = permit decrease 1 = inhibit decrease

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of process variable
S2	N	5	I	Note 1	Block address of set point
S3	N	5	I	Note 1	Block address of track reference

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S4	N	0	I	Note 1	Block address of track flag: 0 = track 1 = release
S5	N	5	I	Note 1	Block address of external or manual reset
S6	N	5	I	Note 1	Block address of feedforward signal
S7	N	5	I	Note 1	Spare real input
S8	N	0	I	Note 1	Spare boolean input
S9	N	0	I	Note 1	Block address of increase inhibit: 0 = normal 1 = prevent increase
S10	N	0	I	Note 1	Block address of decrease inhibit: 0 = normal 1 = prevent decrease
S11	Y	1.000	R	Full	Gain multiplier *K*
S12	Y	1.000	R	Full	Proportional gain K_P
S13	Y	0.000	R	Full	Integral reset resets/min. or manual reset time constant $K_I \times min.$
S14	Y	0.000	R	Full	Derivative rate action $K_D \times min.$
S15	Y	10.000	R	Full	Derivative lag constant K_A (typically = 10)
S16	Y	105.000	R	Full	High output limit
S17	Y	-5.000	R	Full	Low output limit
S18	N	10	I	00 - 03 or 10 - 13 (Note 2)	Algorithm **Version:** 0X=original 1X=new[3] **Type:** X0 = classical X1 = noninteracting X2 = classical with external reset X3 = manual reset noninteracting
S19	N	0	I	0 or 1	Integral limit type: 0 = quick saturation recovery 1 = conventional saturation recovery
S20	Y	0	I	0 or 1	Set point modifier: 0 = normal 1 = integral only on set point change
S21	Y	0	I	0 or 1	Direction switch: 0 = reverse mode *error = SP – PV* 1 = direct mode *error = PV – SP*
S22	Y	0.000	R	Full	Spare real parameter

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S23	Y	0	I	Full	Spare integer parameter

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC
2. Input values other than those defined may result in unpredictable APID behavior.
3. The new version has been revised to improve the internal behavior of the algorithm. The original version is retained for backward compatibility of old configurations.

156.1Explanation

156.1.1Specifications

S1
Block address of the process variable. This identifies the process variable controlled by the PID algorithm.

S2
Block address of the set point.

S3
Block address of the track reference. This identifies the signal the PID outputs when in track mode.

S4
Block address of the track flag. This input signal controls the track or release mode:

0 = track mode
1 = release mode

In track mode, the output is forced to the value of the track reference (S3). In release mode, the block output is calculated as a function of the process variable and the set point.

S5
Block address of the external or manual reset. This input has two functions. It can link the external reset signal that is used in the integral calculation (S18, algorithm type two). It also can link the manual reset signal for manual reset control (S18, algorithm type three).

S6
Block address of the feedforward signal. This input is the signal added to the output calculated by the PID algorithm. This combined output becomes the block output in the release mode.

S7 and S8
Spare real and boolean input.

S9
Block address of increase inhibit signal. An input of one prevents the control output from increasing beyond its current value when the controller is not in the track mode. An input of zero does not affect the PID controller. If the S9 and S10 inputs are both one (in release mode), the output of the PID controller is held at its current value. These inputs also go to outputs N+1 and N+2.

S10
Block address of the decrease inhibit signal. An input of one prevents the control output from decreasing below its current value when the controller is not in the track mode. An input of zero does not affect the PID controller. If the S9 and S10 inputs are both one (in release mode), the output of the PID controller is held at its current value. These inputs also go to outputs N+1 and N+2.

S11
Gain multiplier K. The gain multiplier is one of the terms in the PID calculation.

S12
Proportional gain KP. The proportional gain is one of the terms in the PID calculation.

S13
Integral reset KI resets/min. The integral reset (controller type zero, one or two of S18) is a term in the PID calculation. Specification S13 is the manual reset time per minute for manual reset controllers (type three of S18).

S14
Derivative rate action KD min. The derivative rate action is one of the terms in the PID calculation.

NOTE: The derivative rate action is calculated based on changes in the process variable only. To calculate the derivative rate action on set point changes as well, calculate an error signal external to the advanced PID controller. This error signal can then be introduced as the process variable with a set point of zero.

S15
Derivative lag constant KA. The derivative lag constant is one of the terms in the PID calculation.
Controllers refrain from directly implementing derivative control in favor of filtering the derivative contribution. The derivative lag constant allows specifying the extent of this filtering. This filtering is a simple first order lag with a time constant of KD/KA. For the default setting of KA = 10.0, the filter has a time constant of 1/10 the derivative time. Typical values are from ten to 20 for KA.

NOTE: In order to effectively disable the derivative filtering action, set KA to a very high value (such as 9E18). Note that KA has no affect if the derivative constant (KD) is equal to 0.0. Also, if KA is configured by the user to 0.0, the function code will internally substitute a value of 1.0 for all calculations.

S16
High output limit. The output (PID algorithm plus feedforward signal S6) is limited by this value before it is transferred to the block output.

S17
Low output limit. The output (PID algorithm plus feedforward signal S6) is limited by this value before it is transferred to the block output.

S18
Tens digit, selects the algorithm version:

0X = original **-** PID output is calculated using the original algorithm implementation. Existing configurations may use this version for backward compatibility.

1X = new **-** PID output is calculated using a new version of the algorithm implementation.

NOTE: The algorithm version is not permitted to be modified via an on-line configuration operation.

It is recommended that new configurations use the new version of the algorithm (S18 = 1X). However, for existing configurations, the original algorithm (S18 = 0X) is available for full backward compatibility.
Ones digit, selects the type of algorithm for the PID calculation:

X0 = classical **-** PID output is calculated using a classical interactive controller. Tuning any of the proportional, integral or derivative terms changes the effective value of the other terms.

X1 = noninteracting **-** PID output is calculated using a noninteracting control algorithm. Tuning the proportional, integral or derivative terms individually has no effect on the other terms. This is the same type as function code 19.

X2 = classical with external reset **-** cascade and override configurations use this type of algorithm. The PID output is calculated using the classical interactive control algorithm. The integral contribution is calculated as a function of the external reset signal.

X3 = manual reset noninteracting **-** PID output is calculated from the proportional and derivative terms with manual reset. For manual reset control, a manual reset time constant (S13) is used for bumpless transfer between the track and release states.

NOTE: The transfer is not bumpless if the manual reset time constant (S13) is set to zero. Any change in the manual reset is filtered by a first order lag with the manual reset time specified.

All versions of the algorithm provide bumpless auto or manual transfer and tuning of the proportional band.

S19
Integral limit type. This input specifies the limiting type applied to the integral calculation. Both forms of limiting prevent controller wind-up during saturation of the control output.

0 = quick saturation recovery limiting - integral limiting equals (specified limits minus feedforward signal minus proportional action).

Proportional action = $K \times K_P \times (SP - PV)$ *(reverse acting)*

Proportional action = $K \times K_P \times (PV - SP)$ *(direct acting)*

The quick saturation recovery limitation uses the proportional action calculation for all algorithm types specified in S18. This type of limiting prevents the integral action from duplicating the efforts of the feedforward signal and the proportional action when the control output is saturated. When a decrease in error between the process variable and set point occurs, the control output immediately moves out of saturation minimizing the possibility of set point overshoot.

NOTE: This action is not desirable if the PID controller is used as a limiting controller holding a valve or other device at a limit (e.g., holding a valve closed) and set point changes are made that result in a decrease, but not a change in sign, in the error. In addition, the use of this option may not be desirable when the integral reset (S13) is set to zero (i.e., a P or PD controller).

Example

The system is initially at a steady state with zero offset, an error signal between the process variable and set point develops, resulting in control output saturation. Then, the use of this option shifts the integral value to the integral limit. This results in an offset from the initial steady-state when the error signal reduces to zero.

1 = conventional saturation recovery limiting - integral limiting equals (specified limits minus feedforward signal).

With this type of limiting, the control output moves out of saturation only after the error between the process variable and the set point has changed sign; this may result in significant overshoot of the set point by the process variable.

S20
Set point modifier. This input defines the action to be taken on a set point change.

0 = normal - this typically results in a jump in the control output due to the proportional contribution from the error created by a set point change.

1 = integral only on set point change - proportional contribution of the error is subtracted from the integral contribution. This action eliminates the jump in control output and results in only integral action on a change in set point.

S21
Direction switch. This input defines the direction the control output must move to compensate for an error between the process variable and the set point.

0 = reverse mode controller - an increase in the control output results in an increase in the process variable. The controller error signal is equal to the set point minus the process variable.

error = SP – PV

1 = direct mode controller - an increase in the control output results in a decrease in the process variable. The controller error signal is equal to the process variable minus the set point.

error = PV – SP

S22 and S23
Spare parameters.

156.1.2 Outputs

N
Control output with feedforward.

N+1
Block increase flag.

0 = permit increase
1 = inhibit increase

Figure 156-1 shows an advanced PID output example. Output N+1 from block B should be linked directly to S9 of an advanced PID block A, whose output forms the set point to block B.

NOTE: Do not use the output block increase flag if using the quick saturation recovery option (S19 equals zero).

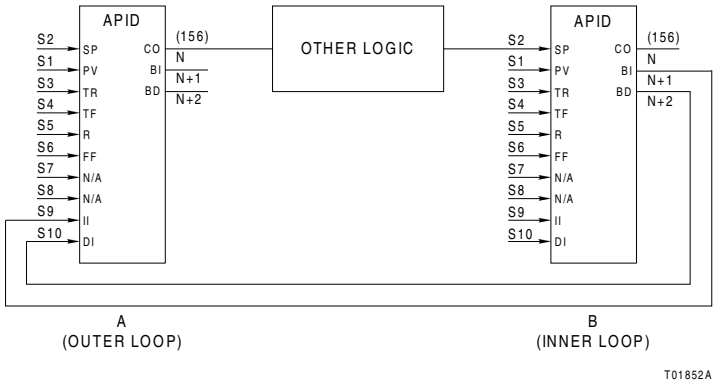


Figure 156-1 Advanced PID Output Example

N+2
Block decrease flag.

0 = permit decrease
1 = inhibit decrease

NOTE: Do not use the output block decrease flag if using the quick saturation recovery option (S19 equals zero).

Refer to Figure 156-1. The N+2 output from block B should be linked directly to S10 of an advanced PID block A, whose output forms the set point to block B.

NOTE: If block B is placed in track, both of its status flags are set to one. This limits control action of block A in both directions.

The advanced PID controller uses a limit checking and status passing mechanism. This feature is designed to constrain controllers in cascade configurations when limits are met.
Two boolean status flags implement this feature. The outputs N+1 and N+2 reflect two conditions:

1. The limit status of the local advanced PID controller.
2. The limit status of downstream controllers. This information is supplied by S9 and S10.

When the advanced PID controller saturates at one of its limits, the appropriate output is set (i.e., block increase or decrease). A further increase or decrease of the set point will attempt to drive the local advanced PID controller further into saturation. The N+1 and N+2 outputs can be monitored by any advanced PID controller whose output forms the set point for the loop. Setting the N+1 or N+2 outputs prevents the higher level advanced PID controller from increasing or decreasing its output if this action causes further saturation.

NOTES:
1. The N+1 and N+2 outputs are adjusted for the direct or reverse mode of each advanced PID controller.

2. The use of the quick saturation recovery option (S19 equals zero) in the inner loop APID may result in ringing of the increase or decrease inhibit flag value as the inner loop saturates at one of its limits. Ringing is where the inhibit flag value flips between zero and one frequently. This may result in creeping of the output loop control output (i.e., instead of holding the outer loop control output (inner loop set point) constant when the inner loop saturates, the outer loop control output may move, or creep, as ringing of the inhibit flag occurs). The quick saturation recovery option is designed to have the control output move immediately out of saturation when a decrease in error between the process variable and set point occurs. Therefore, if the inner loop error value increases and decreases frequently near saturation, its control output moves in and out of saturation resulting in ringing of the inhibit output flag. In such instances, it is recommended that the conventional saturation recovery limiting option (S19 equals one) be used in the inner loop APID.

For example, if an inner loop controller (reverse mode) saturates at its upper limit, the block increase flag will be set. Setting this flag indicates to the outer loop controller that it should not increase its output (which acts as the set point to the inner loop).

NOTE: A direct mode controller sets the block decrease flag when it saturates at its upper limit.

156.2Classical PID Controller

Standard form of a classical PID controller.

$$Output = K\left(1+\frac{1}{T_i s}\right)\left(\frac{T_D s+1}{T_D a s+1}\right)error$$

where:

error	SP – PV (reverse mode)
error	PV – SP (direct mode)

Advanced PID controller block parameters.

S11 = Gain multiplier K.
S12 = Proportional gain K_P.
S13 = Integral reset (resets per min.) K_I.
S14 = Derivative rate action (min.) K_D.
S15 = Derivative lag constant (typically equals ten) K_A.

NOTE: This controller works in seconds internally. It is assumed that K_I and K_I are in resets per minute and minutes, respectively. The 60 term converts K_I and K_I into resets per second and seconds, respectively.

Substituting block parameters into the original equation.

$$Output = K \times K_P\left(1+\frac{K_I/60}{s}\right)\left(\frac{60K_D s+1}{\frac{60K_D}{K_A}s+1}\right)error$$

Standard convention excludes the effects of derivative action on set point changes. Using superposition, this is achieved for reverse mode.

$$Output = KK_P\left(1+\frac{K_I/60}{s}\right)SP - KK_P\left(1+\frac{K_I/60}{s}\right)\left(\frac{60K_D s+1}{\frac{60K_D}{K_A}s+1}\right)PV$$

$$Output = KK_P\left(1+\frac{K_I/60}{s}\right)\left[SP - \frac{60K_D s+1}{\frac{60K_D}{K_A}s+1}PV\right]$$

To make the algorithm suitable for external reset, the proportional and integral section is implemented using positive feedback of a first order lag filter. Figure 156-1 is a block diagram of a reverse mode classical controller. Figure 156-1 is a detailed block diagram of the reverse mode classical controller.

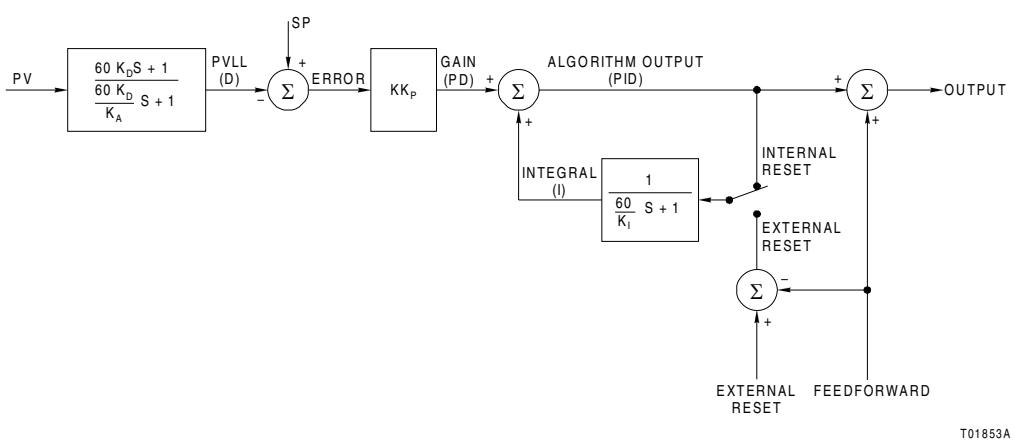


Figure 156-1 Classical Controller

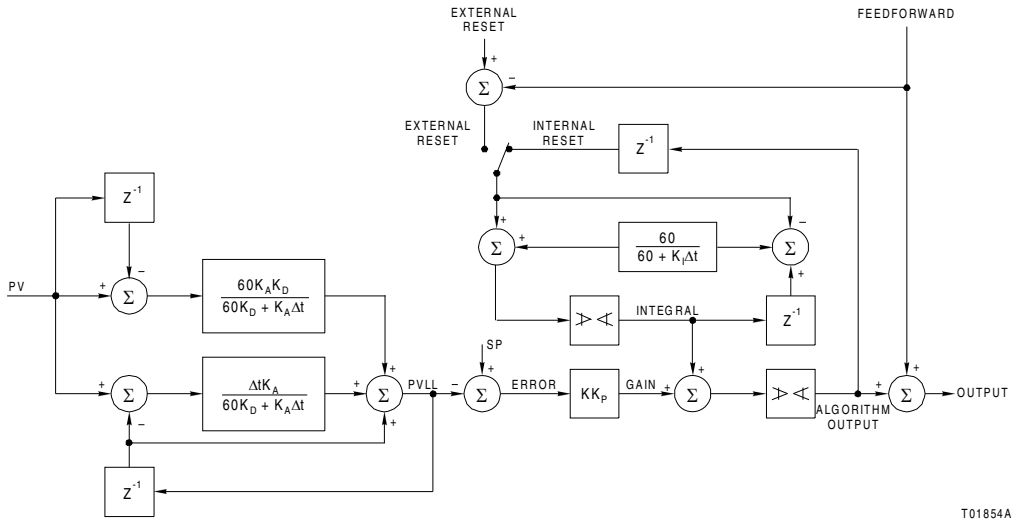


Figure 156-1 Classical Controller - Detail

Equations to implement the reverse mode classical controller.

Block output = feedforward + algorithm output

$$PVLL = \text{Previous PVLL} + \frac{60K_A K_D(PV - \text{previous } PV)}{60K_D + K_A Dt} + \frac{DtK_A(PV - \text{previous } PVLL)}{60K_D + K_A Dt}$$

PVLL = PV during initialization or track mode or $K_A = 0$

error' = SP – PVLL reverse mode

error' = PVLL – SP direct mode

Algorithm output = gain + PI

NOTE: Algorithm output is limited to the specified limits minus the feedforward value.

Gain = $(K\ K_P)$ *error'*

$$PI = \left(\frac{60}{60 + K_I Dt}\right)(\text{previous } PI) + \left(\frac{K_I Dt}{60 + K_I Dt}\right)(\text{previous algorithm output})$$

or

$$PI = \left(\frac{60}{60 + K_I Dt}\right)(\text{previous } PI) + \left(\frac{K_I Dt}{60 + K_I Dt}\right)(\text{ext. reset} - \text{feedforward})$$

NOTES:

1. The integral value is limited to the specified limits minus the feedforward and minus the proportional component when quick saturation recovery limiting is selected. The integral is limited to the specified limits minus the feedforward when conventional saturation recovery limiting is selected.

2. The *PI* value is adjusted to compensate for bumpless transfer, bumpless proportional tuning and implementation of the set point modifier option (S20). The *PI* term can be forced outside its normal limits due to one of these conditions. After this happens, a newly computed value is only allowed to move toward the region between its limits. The *PI* value is not forced within the limits, but it is also not allowed to move further from its limits.

3. Using external reset and feedforward control simultaneously may cause controller instability. This is possible due to the interaction between the feedforward and external reset signals. If there is a significant lag between a change in the feedforward signal and the resultant change in external reset, the control output will first respond to the feedforward change and then tend to converge to the external reset signal. The external reset signal will dominate whenever the controller output is saturated.

4. The classical control algorithm cannot be used for integral only control with internal reset and K_P equal to zero. With K_P equal to zero and external reset not specified, the block will automatically default to a noninteracting PID controller.

156.3Noninteracting PID Controller

Standard form of a noninteracting PID controller.

$$Output = \left(K + \frac{1}{T_1 s} + \frac{T_D s}{T_D a s + 1}\right) error$$

where:

error	SP – PV reverse mode
error	PV – SP direct mode

Advanced PID controller block parameters.

S11 = Gain multiplier K.
S12 = Proportional gain K_P.
S13 = Integral reset (resets per min.) K_I.
S14 = Derivative rate action (min.) K_D.
S15 = Derivative lag constant (typically equals ten) K_A.

NOTE: This controller works in seconds internally. It is assumed that K_I and K_D are in resets per minute and minutes, respectively. The *60* term converts K_I and K_D into resets per second and seconds, respectively.

Substituting block parameters into the original equation.

$$Output = K\left(K_P + \frac{K_I/60}{s} + \frac{60K_D s}{\frac{60K_D}{K_A}s + 1}\right) error$$

The standard convention is to exclude the effects of derivative action on set point changes. Using superposition, this is achieved for reverse mode as:

$$Output = K\left(K_P + \frac{K_I/60}{s}\right) error - K\left(\frac{60K_D s}{\frac{60K_D}{K_A}s + 1}\right) PV$$

Refer to Figure 156-1 for a block diagram of a reverse mode noninteracting controller. Figure 156-1 illustrates a detailed block diagram of the reverse mode noninteracting controller.

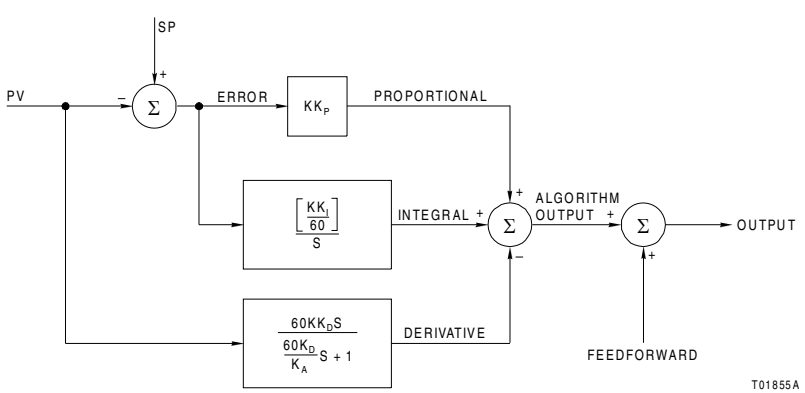


Figure 156-1 Noninteracting Controller

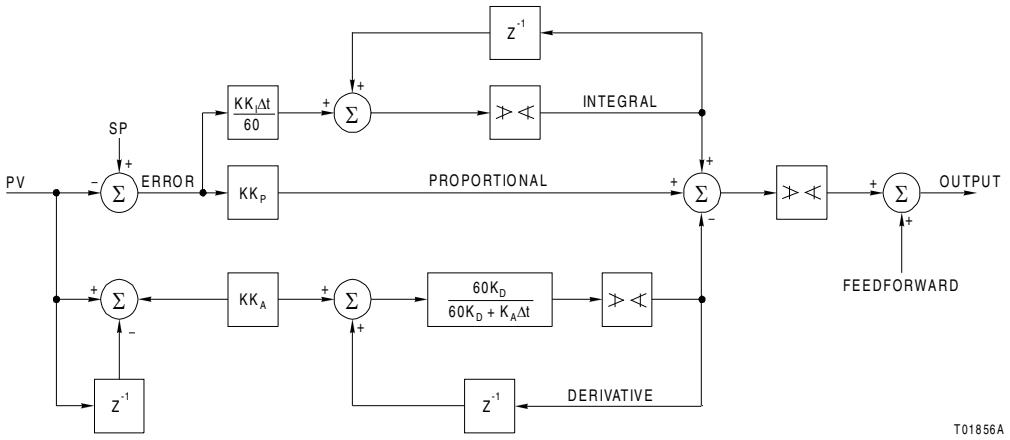


Figure 156-1 Noninteracting Controller - Detail

Equations to implement the reverse mode noninteracting controller.

Block output = *feedforward* + *algorithm output*

Algorithm output = *proportional* + *integral* – *derivative*

NOTE: Algorithm output is limited to the specified limits minus the feedforward value.

Proportional = *(KK$_P$) error*

$$Integral = \left(\frac{KK_I Dt}{60}\right) error + previous\ integral$$

NOTES:

1. The integral value is limited to the specified limits minus the feedforward value and minus the proportional term when the quick saturation recovery limiting option is selected. The integral value is limited to the specified limits minus the feedforward value when conventional saturation recovery limiting is selected.

2. The integral value is adjusted to compensate for bumpless transfer, bumpless proportional tuning and implementation of the set point modifier option (S20). The integral term can be forced outside its normal limits due to one of these conditions. After this happens, a newly computed value is only allowed to move toward the region between its limits. The integral value is not forced within the limits, but it is also not allowed to move further from its limits.

$$Derivative = \left(\frac{60KK_D K_A}{60K_D + K_A Dt}\right)(PV - previous\ PV) + \left(\frac{60K_D}{60K_D + K_A Dt}\right)(previous\ derivative)$$

NOTE: Derivative limiting equals ± span of specified limits. This is shown for a reverse acting controller. For direct action, change (*PV – previous PV*) to (*previous PV – PV*).

156.4Manual Reset PID Controller

The manual reset PID control algorithm provides a noninteracting controller with a fixed integral (reset) term. Use this type of controller when only proportional or derivative action is required. The manual reset enables the balance of the control loop at a specific operation point. This reduces the steady state offset between the process variable and set point.
Taking the transfer function for a noninteracting PID controller (with derivative action on the process variable) and replacing the integral term with a first order lag yields the following transfer function:

$$Output = (KK_P)error + \left(\frac{1}{60K_I s + 1}\right) manual\ reset - K\left(\frac{\frac{60K_D s}{60K_D}}{\frac{K_A}{K_A}s + 1}\right)PV$$

NOTE: This controller works in seconds internally. It is assumed that K_I and K_D are in resets per minute and minutes, respectively. The *60* term converts K_I and K_D into resets per second and seconds, respectively.

where:

error	SP – PV reverse acting
error	PV – SP direct acting

NOTE: K_I is a time constant (min.), not a rate setting (resets per min.). The first order lag on the manual reset input avoids bumping the process whenever the manual reset value is changed. Manual or auto (track or release) transitions also utilize this time constant to perform bumpless transfer and ramping to the manual reset value.

Refer to Figure 156-1 for a block diagram of a reverse mode manual reset PID controller.

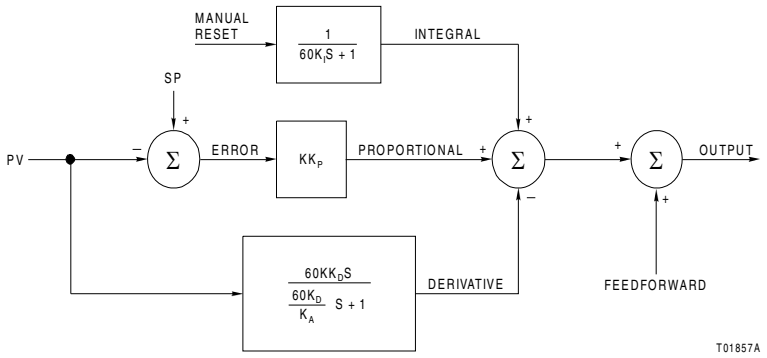


Figure 156-1 Manual Reset PID Controller

Figure 156-1 illustrates a detailed block diagram of the reverse mode manual reset PID controller.

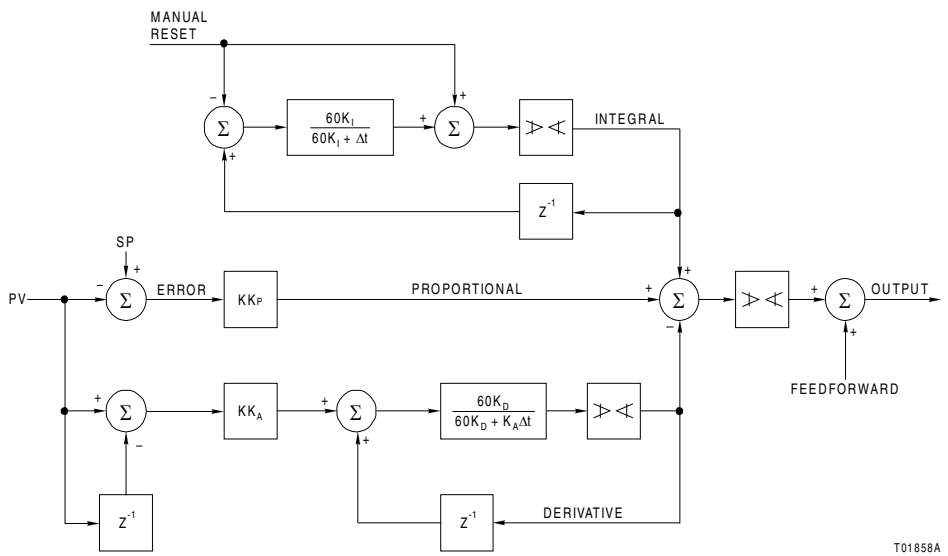


Figure 156-1 Manual Reset PID Controller - Detail

The equations to implement the reverse mode manual reset PID controller are:

Block output = feedforward + algorithm output
Algorithm output = proportional + integral – derivative

NOTE: Algorithm output limiting equals specified limits minus feedforward.

Proportional = (KK_P) error

$$Integral = manual\ reset + \left(\frac{60K_I}{60K_I + Dt}\right)(previous\ integral - manual\ reset)$$

NOTE: The integral value is adjusted to compensate for bumpless transfer, bumpless proportional tuning and implementation of the set point modifier option (S20). The integral term can be forced outside its normal limits due to one of these conditions. After this happens, a newly computed value is only allowed to move toward the region between its limits. The integral value is not forced within the limits, but it is also not allowed to move further from its limits.

$$Derivative = \left(\frac{60KK_DK_A}{60KK_DK_ADt}\right)(PV - previous\ PV) + \left(\frac{60K_D}{60K_D + K_ADt}\right)(previous\ derivative)$$

NOTE: Derivative limiting equals ± span of specified limits. This is shown for a reverse acting controller. For direct action, change (*PV – previous PV*) to (*previous PV – PV*).

156.5Applications

Figure 156-1 shows the use of the advanced PID controller in a single input/single output control loop.

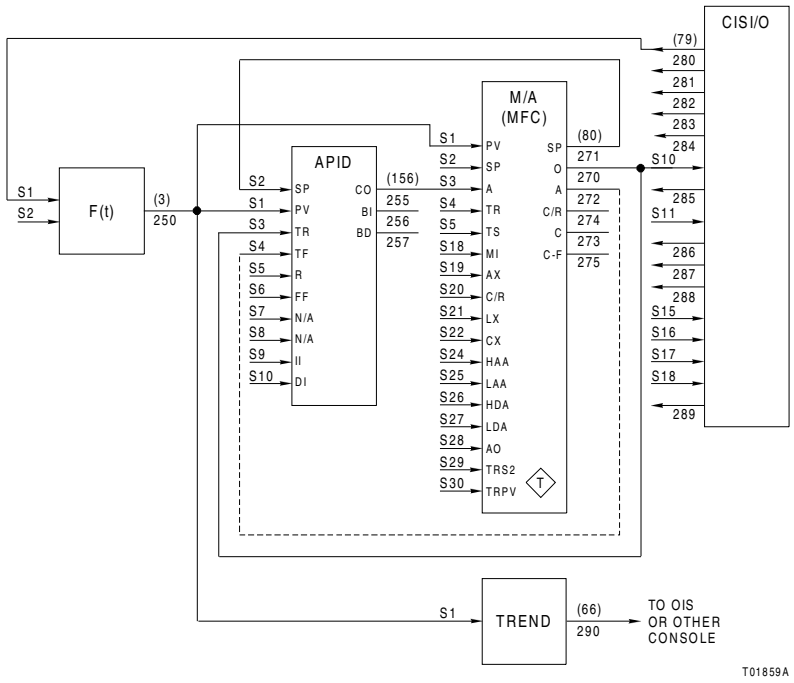
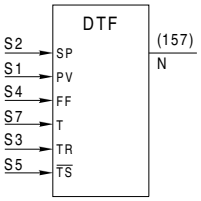


Figure 156-1 Single Input/Single Output Control Loop

157.General Digital Controller

The general digital controller function code implements a fourth order difference equation with variable deadtime. This function block uses previous outputs and error signals (SP minus PV) to calculate the present output. The starting point for calculations can be initiated by time or trigger. This block can function as a digital controller, digital filter, or a digital process model for the implementation of sophisticated control schemes.



Outputs

Blk	Type	Description
N	R	U(t)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of process variable
S2	N	5	I	Note 1	Block address of set point
S3	N	5	I	Note 1	Block address of track value
S4	N	5	I	Note 1	Block address of feedforward
S5	N	0	I	Note 1	Block address of release/track flag: 0 = track 1 = release
S6	N	1	B	0 or 1	Execution mode: 0 = trigger 1 = time
S7	N	0	I	Note 1	Block address of external trigger flag: 1 = run
S8	N	1.000	R	0 - 9.2E18	Time between runs (in secs)
S9	Y	105.000	R	Full	High output limit
S10	Y	-5.000	R	Full	Low output limit
S11	Y	0.000	R	Full	Coefficient a0
S12	Y	0.000	R	Full	Coefficient a1
S13	Y	0.000	R	Full	Coefficient a2
S14	Y	0.000	R	Full	Coefficient a3
S15	Y	0.000	R	Full	Coefficient a4
S16	Y	1.000	R	Full	Coefficient b0
S17	Y	0.000	R	Full	Coefficient b1
S18	Y	0.000	R	Full	Coefficient b2
S19	Y	0.000	R	Full	Coefficient b3
S20	Y	0.000	R	Full	Coefficient b4
S21	N	0	I	0 - 255	Numerator deadtime expressed as a number of sample intervals
S22	N	0	I	0 - 255	Denominator deadtime expressed as a number of sample intervals

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S23	Y	0.000	R	Full	Spare parameter
S24	Y	0.000	R	Full	Spare parameter

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

157.1Explanation

157.1.1Specifications

S1
(Block address of process variable) Current value of the input from the process. Specifications S1 and S2 define the error term in the difference equation:

e(t) = <S2> – <S1>

S2
(Block address of set point) Block address of the set point. This is the current value of the set point input. It defines the desired value of the process variable. Specifications S1 and S2 define the error term in the difference equation:

e(t) = <S2> – <S1>

S3
(Block address of track value) Supplies the block output N when the controller is tracking. The output is limited before it is output to the field and before it is used to update the output buffer. The final output from the general digital controller is the sum of this limited internal control output value and the feedforward signal.

S4
(Block address of feedforward input) Block address of the feedforward input. This input biases the output of the general digital controller based on the changing value of some other variable. The feedforward input is an externally generated signal.

S5
(Block address of input selecting controller tracking) Block address of the input that selects controller tracking. When tracking is selected, the output N tracks the value referenced by S3.

0 = track <S3>
1 = release

S6
(Time and trigger mode select) Defines the mode of data collection. In the time mode, data is collected at fixed intervals of time defined by S8. In the trigger mode, data is collected when <S7> makes a zero to one transition.

0 = trigger mode
1 = time mode

S7
(Block address of external trigger) Block address of the external trigger. If the trigger mode of execution is selected, the calculation is initiated each time this input makes a zero to one transition. The calculation is also initiated when the track switch <S5> is set to zero to force the output to track the desired value.

S8
(Interval between executions) Identifies the interval between executions in seconds if the time based mode of execution is selected.

S9
(High output limit) Actual control output will not exceed this value.

NOTE: The actual control output equals the sum of the low limited internally calculated control output plus the feedforward input value, then high limited if necessary.

S10
(Low output limit) Internally calculated control output will not be less than this value. The actual control output will not be less than this value plus the feedforward input value.

NOTE: The actual control output equals the sum of the low limited internally calculated control output plus the feedforward input value, then high limited if necessary.

S11 through S20
(Coefficients used in difference equation) Values of the coefficients used in the difference equation. The coefficient values are determined from the relationships between the operating parameters of the device being controlled (control theory modeling of process).

S21
(Numerator deadtime) Numerator deadtime expressed as a number of sample intervals. The error signal entering the general digital controller block is not acted on by the block until this number of sample intervals has passed.

S22
(Denominator deadtime) Denominator deadtime expressed as a number of sample intervals. The outputs of the general digital controller are recycled back into the equation after they are calculated. The outputs being fed back into the controller are not acted on until the sample intervals have passed.

NOTE: The value of S21 and S22 affects memory utilization. Refer to Appendix B for details.

S23 and S24
Spare parameters.

157.1.2Output

N
Calculated from previous outputs, current and previous errors, and the feedforward value. During module startup or tracking, the output is controlled by the track value input.

157.2Application

General Information
The general digital controller is used to implement control algorithms that are based on discrete time sampled data that are sampled at a rate that can be internally generated (<S8> when S6 equals one) or externally generated (<S7> when S6 equals zero).
Discrete time based functions are simple continuous time based functions that have been sampled at some periodic rate. Just as Laplace transforms are useful to represent complex continuous time functions in a simple manner, Z-domain transforms are used to represent complex discrete time sampled functions in an analogous simple manner. Like Laplace transforms, Z-domain transforms can be used to simplify a complex continuous function into a simple equation. Since in the world of digital control systems all continuous time based data is sampled into discrete time based data, it is more relevant to perform complex continuous time based control algorithms using their equivalent discrete time based algorithms. These continuous time based control algorithms can be converted into discrete time based algorithms via the use of Z-domain transforms. As in Laplace transforms, Z-domain transforms can be algebraically manipulated and simplified into simple equations consisting of simple terms. Conversion tables exist for converting these simple terms back and forth between their equivalent continuous time and Z-domain functions. Further information on this can be found in any good basic control textbook.
Any complex controller algorithm can be implemented by first determining its continuous time base transfer function. With the use of the knowledge of Z-domain transforms, the continuous time based function can be translated into the Z-domain function representation of the equivalent discrete time based function. The Z-domain equation can then be algebraically manipulated into one or more terms that is equivalent to the Z-domain transfer function of function code 157. Therefore, one or more function code 157 general digital controller function blocks can be used to implement the original complex time based control algorithm.
The Z-domain transfer function representation of the general digital controller is:

$$G_{(Z)} = \frac{U_{(Z)}}{E_{(Z)}} = \frac{Z^{-N}(a_4Z^{-4} + a_3Z^{-3} + a_2Z^{-2} + a_1Z^{-1} + a_0)}{Z^{-D}(b_4Z^{-4} + b_3Z^{-3} + b_2Z^{-2} + b_1Z^{-1} + b_0)}$$

where:

$U_{(Z)}$ = Z – transfer function of *u(t).*

$E_{(Z)}$ = Z – transfer function of *e(t).*

Specific Information

The general digital controller block calculates an output based on previous outputs and error signals. The calculation uses the discrete function:

$$u(t) = \frac{1}{b_0}[a \times e(t-N) + a_1 \times e(t-N-1) + a_2 \times e(t-N-2)$$
$$+ a_3 \times (et-N-3) + a_4 \times e(t-N-4)] - \frac{1}{b_0}[b_1 \times u(t-D-1)$$
$$+ b_2 \times u(t-D-2) + b_3 \times u(t-D-3) + b_4 \times u(T-D-4)]$$

where:

a_0 - a_4 b_0 - b_4	=	Coefficients (specified in S11 through S20)
N	=	Numerator (input) deadtime expressed as a number of sample intervals (S21)
D	=	Denominator (output feedback) deadtime expressed as a number of sample intervals (S22)
e(t)	=	Present error = (<S2> – <S1>)
e(t - N - n)	=	Error from (*N*+*n*)th previous run of the algorithm
u(t)	=	Current internal control output value
u(t - D - n)	=	Internal control output from the (*D*+*n*)th previous run of the algorithm

The general digital controller takes inputs and holds them for a specified number of time intervals for each step before releasing them to the next step as shown in Figure 157-1. On start-up, the error queue is filled with the error signal and the output queue is filled with the track value minus the feedforward value.

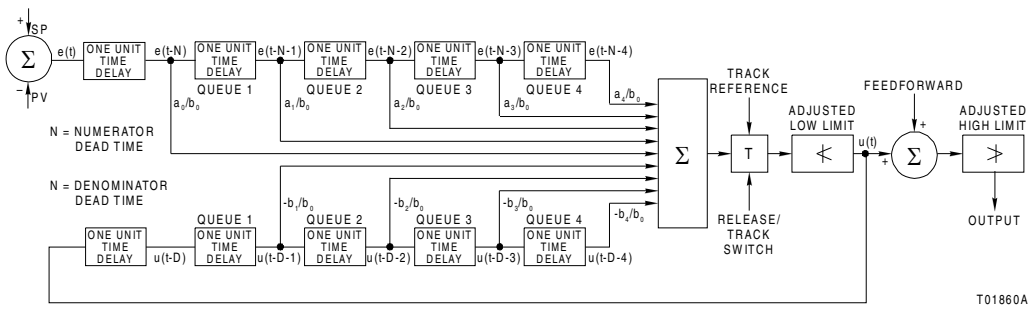


Figure 157-1 Internal Logic

The general digital controller block implements a deadtime queue for the error signal and previous output values. The length of these queues are specified as integer multiples of sample time. On start-up and transfer from manual to automatic, all elements of the error queue are initialized with the current error value, and the output queue is initialized with the track value minus the feedforward. Both queues are of the first-in, first-out (FIFO) type. A new value is placed in the queue at each execution time; values already in the queue are shifted one element to make room for the new value, and the oldest value in the queue is discarded.

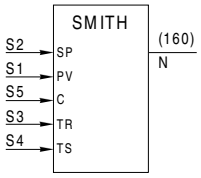
The internal control output of the general digital controller block is formed by adding together the following values:

a_0/b_0X	(fifth oldest value in error queue)
a_1/b_0X	(fourth oldest value in error queue)
a_2/b_0X	(third oldest value in error queue)
a_3/b_0X	(second oldest value in error queue)
a_4/b_0X	(oldest value in error queue)
$-b_1/b_0X$	(fourth oldest value in output queue)
$-b_2/b_0X$	(third oldest value in output queue)
$-b_3/b_0X$	(second oldest value in output queue)
$-b_4/b_0X$	(oldest value in output queue)

The output is limited before it is output to the field and before it is used to update the output queue. The final output from the general digital controller is the sum of this limited internal control output value and the feedforward signal.

160.Inferential Smith Controller

The inferential smith controller (ISC) function code provides predictive control on an error signal developed from the process variable and set point inputs measured against an internal model of the process. The ISC controller utilizes a first order dynamic model with deadtime to predict the current value of the process variable based on past values of the control output. The ISC controller function block provides regulatory process control similar to a PID algorithm. However, the ISC controller has the added advantage of effective control for processes with a significant transport delay (deadtime). The ISC controller prevents controller windup by limiting control output to operator specified high and low limits. The ISC controller also prevents windup in cascade configurations with the use of an external reference value.
Processes with long deadtimes are difficult to control with PID controllers using traditional tuning methods. The ISC controller algorithm functionally replaces the standard PID controller function. The ISC controller easily deals with process deadtime and tunes with a single tuning parameter.



Outputs

Blk	Type	Description
N	R	Control output (CO)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of process variable
S2	N	5	I	Note 1	Block address of set point
S3	N	5	I	Note 1	Block address of track reference value
S4	N	0	I	Note 1	Block address of track switch signal: 0 = track 1 = release
S5	N	5	I	Note 1	Block address of (cascade) external reference value
S6	N	0	I	0 or 1	External reference flag: 0 = normal 1 = use external reference
S7	Y	1.000	R	Full	Process model gain
S8	Y	0.000	R	0 - 9.2 E18	Process model deadtime (in secs)
S9	Y	0.000	R	0 - 9.2 E18	Process model lag time constant (in secs)
S10	Y	9.2 E18	R	0 - 9.2 E18	Controller tuning time constant (in secs)
S11	Y	105.000	R	Full	Control output high limit
S12	Y	-5.000	R	Full	Control output low limit

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

160.1Explanation

Many processes have open-loop step responses similar to the one shown in Figure 160-1. In an open-loop step test, the controller is in manual and the controller output (CO) increases or decreases in a single step. The process response is the behavior of the process variable (PV) in response to the CO change. In the test, PV and CO are initially at steady-state near the desired operating point (i.e., the values of PV and CO are constant over a reasonable period prior to the CO step

change). The value of CO stays constant after the step change, and PV is monitored until it has reached a new constant value.

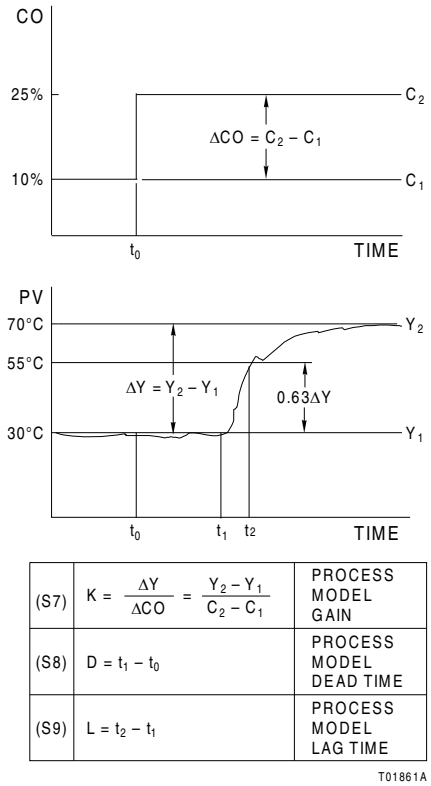


Figure 160-1 Process Model Parameters

The ISC controller uses three parameters to characterize the open-loop step process response: S7, process model gain (K); S8, process model deadtime (D); and S9, process model lag time constant (L). Figure 160-1 shows an example of these parameters. In this example, PV is initially at steady-state at 30 degrees Celsius and CO is at ten percent. CO changes from ten percent to 25 percent at time t_0. PV starts to move from its initial value at time t_1 and reaches a new steady-state temperature of about 70 degrees Celsius. Process model gain is the ratio of the steady-state change in PV to the change in CO, for example,

$$K = \frac{70°C - 30°C}{25\% - 10\%} = 2.67°C/\%$$

In this example, *K* is a positive value (i.e., PV increases as CO increases and PV decreases as CO decreases). In other cases, *K* may be a negative value (i.e., PV increases as CO decreases and PV decreases as CO increases).
Process model deadtime is the time of a change in the control output until a change in the process variable. Process model lag time is the time to reach 63 percent of the final value after the response begins.
Inside the ISC controller function code calculations, the algorithm tries to predict the behavior of the real process based on process model parameters S7, S8 and S9. Because these parameters are only approximations of the real process, there will generally be errors in the prediction. A controller tuning parameter (S10), T, takes into account the effects of the prediction error. Smaller values of T would result in more rapid changes in CO; whereas larger values of T would result in slower changes in CO.
More specifically, if T is less than L, there is more lead action in the control output; if T is greater than L, there is more lag action in the control output. Control output is limited to high and low limits specified in S11 and S12. Qualitatively, larger values of T should be used if the model representation is poor, or quick and large movements of CO are undesirable. One method for tuning is to initially set T to the sum of the process model deadtime (S8) and the process model lag time constant (S9). Place the controller in auto, perform set point changes and adjust T to get a desirable response. Decrease T if the closed-loop response appears too sluggish. Increase T if the closed-loop response is too oscillatory.
The process model parameters are approximate descriptions of the real process about a single operating point. The model becomes less accurate as operating conditions move away from the initial point. Process model parameters may have to be re-estimated as operating conditions change.
The ISC controller should not be used in highly nonlinear processes (example, pH control), or in very fast processes (i.e., processes with dynamics dominated by process gains with negligible deadtime and lag effects). The ISC controller is useful for regulatory control with step-type disturbances (e.g., load disturbances through processes where deadtime is dominant

over lag effects). The ISC controller provides bumpless tuning (i.e., CO will not jump as a result of changing the value of S7, S8, S9 or S10). In addition, the ISC controller provides bumpless manual-to-auto transfer.
Figure 160-1 shows the use of an external reference signal in a cascade configuration. In this case, the outer loop model refers to the effects of the inner loop PV on the outer loop PV; the inner loop model refers to the effects of the inner loop manipulated variable (example, valve position) on the inner loop PV. The inner loop PV is the external reference signal to the outer loop ISC controller. This prevents controller windup in the outer loop ISC controller should the inner loop saturate. Specification S6 of the outer loop ISC controller must equal one.

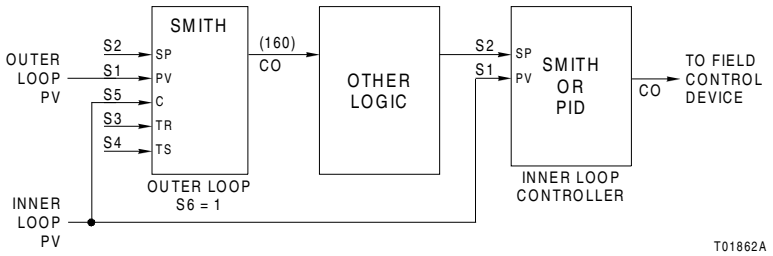


Figure 160-1 External Reference for Cascade Control

160.1.1 Specifications

S1 – PV
Block address of process variable (PV).

S2 – SP
Block address of set point (SP).

S3 – TR
Block address of track reference (TR). The ISC control output (CO) will track the value in this block when the track switch (TS) signal is zero.

S4 – TS
Block address of track switch (TS). The ISC control output (CO) will track <S3> when the value of TS is zero.

0 = track
1 = release

S5 – C
Block address of the cascade (C) external reference value. When the ISC controller is a control module or outer loop controller in a cascade configuration, the control loop uses the external reference value to prevent controller windup should the I/O module or inner loop controller saturate. Typically, the external reference value is the ***inner*** loop process variable. To use the external reference value, S6 of the ***outer*** loop ISC controller must equal one.

S6
External reference flag. Set the value of S6 to one to use the external reference value defined in S5; otherwise, S6 should always equal zero.

0 = normal configuration, external reference not used
1 = use external reference

S7, S8 and S9
Characterize the response of the process variable to a step change in control output. The ISC controller uses a first-order lag with deadtime approximation of the actual process in its internal calculations. These parameters are the ISC process model parameters. Refer to Figure 160-1 for sample calculations of these parameters.

S7 – K
Process model gain (K). K can be positive or negative.

S8 – D
Process model deadtime (D).

S9 – L
Process model lag time constant (L).

S10 – T
ISC controller tuning parameter (T). T must be greater than zero. Without information on model uncertainty, a starting point for tuning T is to set it to the sum of D (S8) and L (S9).
For control with an accurate model, this parameter may be set to 30 percent of process lag time, L (S9). For slower controller response, or when the process model is not considered accurate, the value of this parameter can be increased to the process deadtime, D (S8) plus 300 percent of the process lag time, L (S9).

S11
High control output limit. This specifies the maximum output of the ISC controller block.

S12
Low control output limit. This specifies the minimum output of the ISC controller block.

160.2ISC Structure

Figure 160-1 shows a block diagram representation of the ISC controller structure. In the diagram, U represents the effects of disturbances on the process. U_e is an estimate of the disturbances and effects of modeling error. The ISC controller uses a first-order with deadtime approximation of the process. If there are no modeling errors (i.e., process model equals process), the process output is:

$$PV(s) = F(s)\ SP(s) + [1 - F(s)]\ U(s)$$

where:

$$F(s) = \frac{1}{Ts+1} exp(-sD)$$

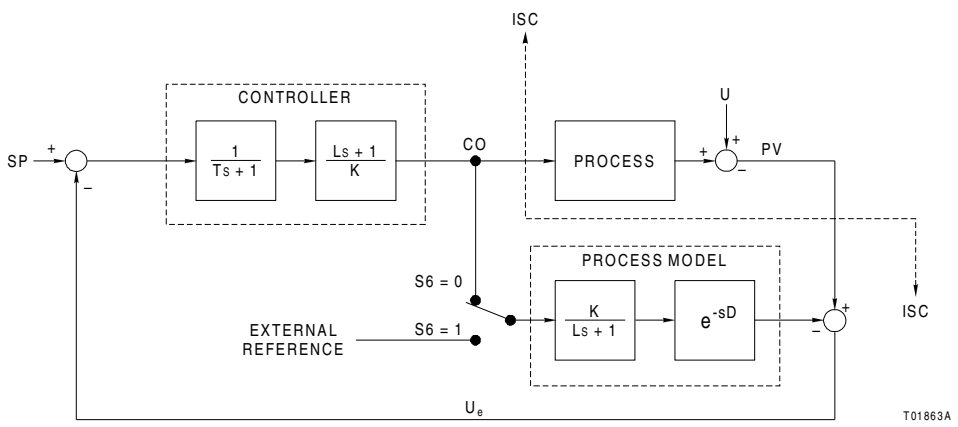


Figure 160-1 ISC Structure

F(s) is the closed-loop response of the system to a set point change, and T is a measure of the closed-loop response speed. The controller is basically a lead/lag feedforward controller with the disturbance estimated by subtracting the model output from the actual measure process value. In real applications, there are always modeling errors and T is a tuning parameter in the lead/lag controller.
Control output constraints and process constraints must be considered in any controller design. In the ISC controller algorithm, this is done by constraining the controller output to within high and low limits (S11 and S12), and by taking into account the predicted model output in the controller calculations. If the control output saturates at a control limit, the input to the model will be a constant value (CO = high limit or low limit) and hence the predicted model output also reaches a constant value.
The controller sees the saturated predicted model output and prevents the control calculations from growing beyond the control limits (i.e., prevents controller windup). The same reasoning applies to cascade control. When using the ISC controller as the control loop, the inner loop PV is the external reference signal (Figure 160-2) and the input to the process model (Figure 160-3, S6=1). This external reference feedback prevents controller windup should the inner loop saturate.

160.3Applications

Figure 160-1 shows how to use the inferential smith controller with a manual/automatic station (function code 80).

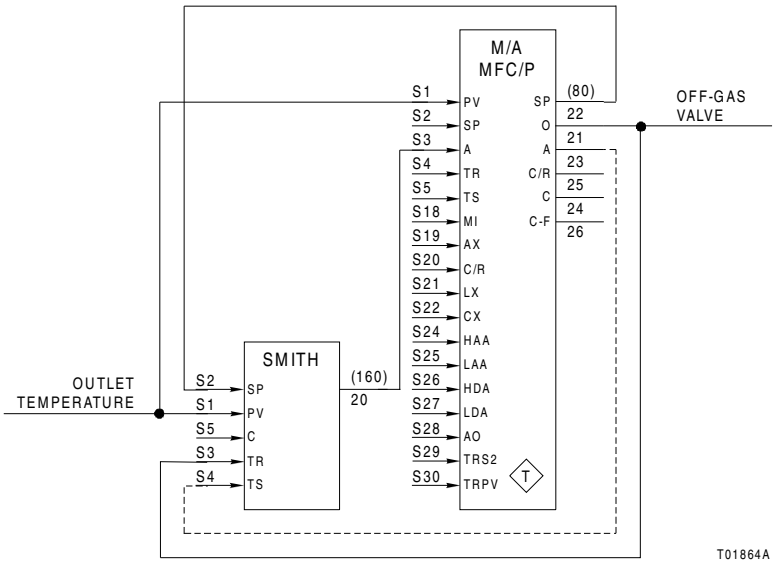
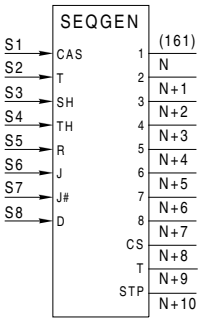


Figure 160-1 Using the ISC with the M/A Station

161.Sequence Generator

The sequence generator function code contains a series of masks that are stepped through on an event or time basis. The masks define the states of eight boolean outputs for each step.
In addition to the eight step masks, there is a default mask often called Step 0 or E-STOP (executed stop). If the block resets or jump to Step 0 executes, E-STOP is the output. E-STOP defines the state of the outputs. Sequential stepping in ascending numerical order, and jumping to specific step numbers are the two methods of event stepping.
If a sequence has more than eight steps, sequence generator blocks can link in series. If there are more than eight outputs per step, sequence generator blocks can be linked in parallel to provide the required number of outputs.
Unlike the other batch function blocks (123, 124, 129, etc.), the last sequence generator in the series is the output.



Outputs

Blk	Type	Description
N	B	Output 1 of current step
N+1	B	Output 2 of current step
N+2	B	Output 3 of current step
N+3	B	Output 4 of current step
N+4	B	Output 5 of current step
N+5	B	Output 6 of current step
N+6	B	Output 7 of current step
N+7	B	Output 8 of current step
N+8	R	Current step number
N+9	R	Seconds remaining in current step
N+10	B	Step taken (logic 0 to 1 transition)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Starting block address of previous sequence generator in series (0 = first in series)
S2	N	0	I	Note 1	Block address of step trigger (0 to 1 transition)
S3	N	0	I	Note 1	Block address of step hold: 0 = release 1 = hold
S4	N	0	I	Note 1	Block address of step timer hold: 0 = release 1 = hold
S5	N	0	I	Note 1	Block address of reset trigger (0 to 1 transition)
S6	N	0	I	Note 1	Block address of step jump trigger (0 to 1 transition)
S7	N	5	I	Note 1	Block address of step jump number
S8	N	0	I	Note 1	Block address of output disable flag
S9	Y	0.000	R	Full	Disable mask
S10	Y	0.000	R	Full	Step 1 output mask
S11	Y	0.000	R	Full	Step 1 time in secs

Specifications

Spec	Tune	Default	Type	Range	Description
S12	Y	0.000	R	Full	Step 2 output mask
S13	Y	0.000	R	Full	Step 2 time in secs
S14	Y	0.000	R	Full	Step 3 output mask
S15	Y	0.000	R	Full	Step 3 time in secs
S16	Y	0.000	R	Full	Step 4 output mask
S17	Y	0.000	R	Full	Step 4 time in secs
S18	Y	0.000	R	Full	Step 5 output mask
S19	Y	0.000	R	Full	Step 5 time in secs
S20	Y	0.000	R	Full	Step 6 output mask
S21	Y	0.000	R	Full	Step 6 time in secs
S22	Y	0.000	R	Full	Step 7 output mask
S23	Y	0.000	R	Full	Step 7 time in secs
S24	Y	0.000	R	Full	Step 8 output mask
S25	Y	0.000	R	Full	Step 8 time in secs

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

161.1Explanation

The sequence generator block controls a batch process by providing outputs that define the operating states for each step in a process for all the devices used. Figure 161-1 illustrates a typical arrangement. A sequence monitor or multi-sequence monitor block monitors the process and selects the order in which the process steps execute. The step number then goes to

the sequence generator block that outputs the values identified with that step. Any number of sequence generator blocks can be linked in series or parallel to provide the required number of outputs or steps.

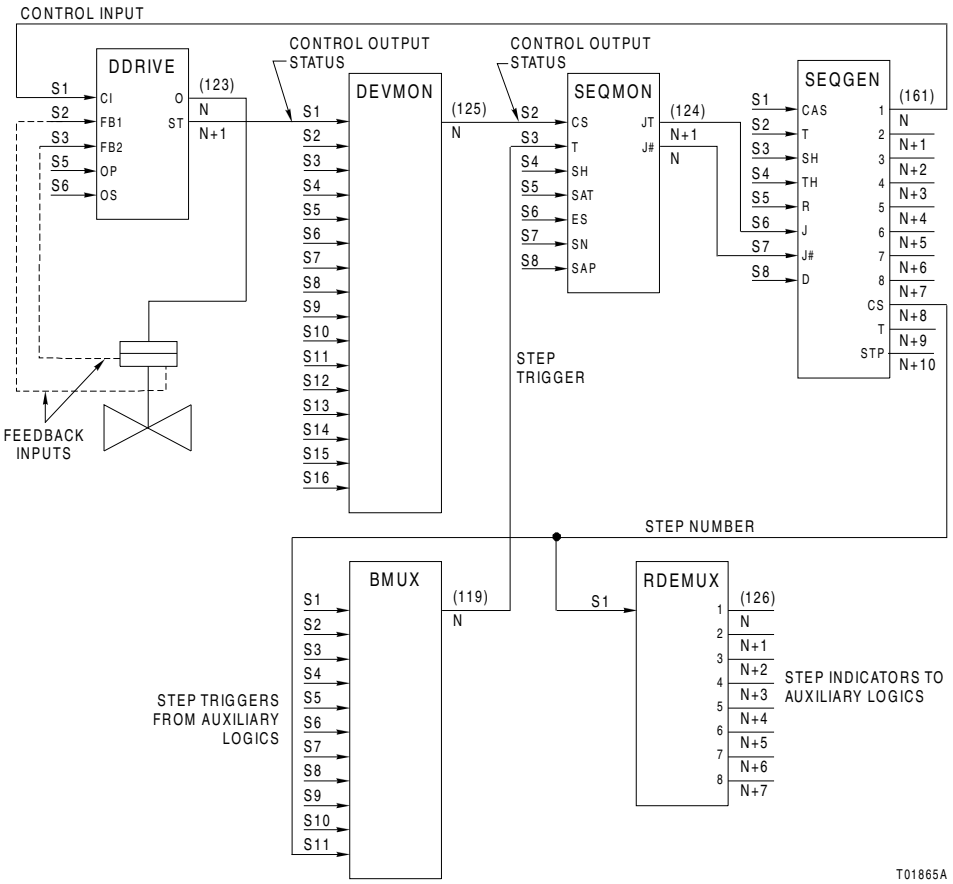


Figure 161-1 Controlling a Batch Process

Figure 161-1 illustrates a configuration with sequence generator blocks ganged in series and in parallel. Sequence generators in series increase the step range by factors of eight. In Figure 161-1, blocks 60, 71 and 82 are in series, providing 24 steps. Running parallel to this sequence is the block series 93, 104 and 115. The first sequence generator has the value of zero for S1, identifying it as the first block in the series, with outputs zero through eight. In the figure, blocks 60 and 93 are the first in the series. The second block in the series must be greater in number than the first. Blocks 71 and 104 are the second blocks in the series, they have values of 60 and 93, respectively, for S1. This shows that they are not the first

blocks in the series. They have step outputs nine through 16. This series relationship continues until the maximum number of steps needed is reached.

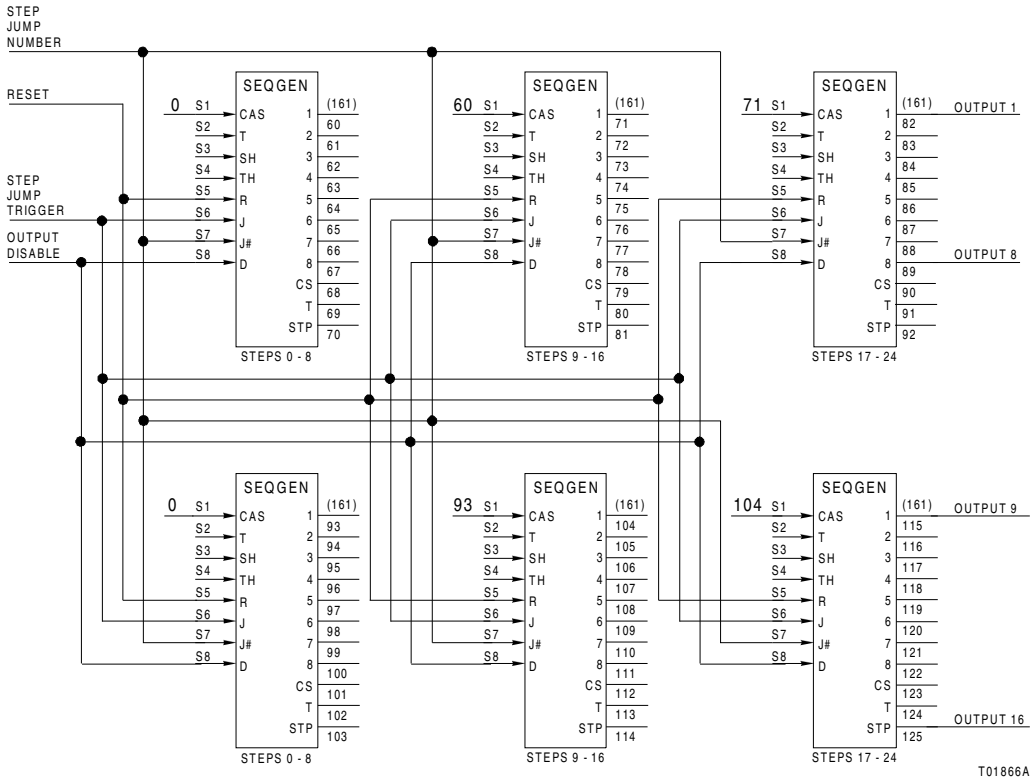


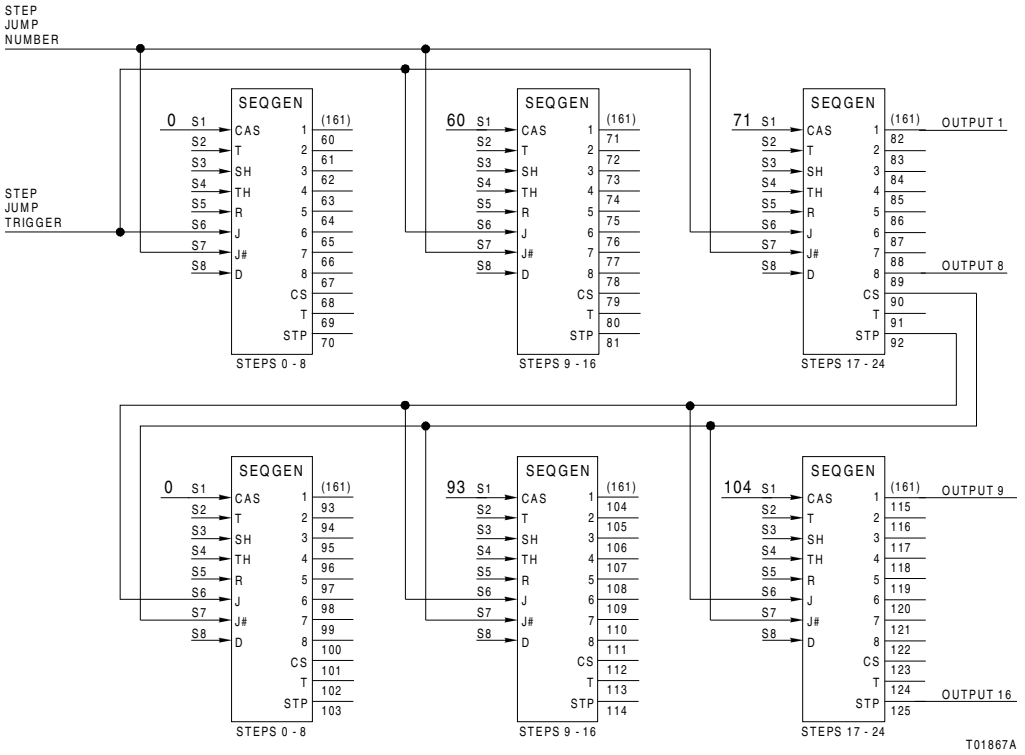
Figure 161-1 Series and Parallel to Provide 24 Steps for 16 Outputs

Sequence generators in series monitor the step number and step taken outputs from the preceding sequence generator block. Therefore, the outputs of the last sequence generator in a series represent the outputs used to drive device driver blocks.

In batch control applications, it is necessary to run the outputs (step jump trigger and step jump number) from the lead sequence monitor block (function code 124) to all of the sequence generator blocks used by the sequence. In this way, all blocks will move from step to step in unison. Furthermore, when using a sequence monitor (function code 124) or multi-monitor block (function code 135), leave all step timers in the sequence generator blocks at the default (zero) value.

Figure 161-1 shows an application not using sequence monitor or multi-sequence monitor blocks. When sequence monitor or multi-sequence monitor blocks are not used, all sequence generator blocks ganged together must reference the same reset, jump trigger, jump step number, and disable inputs. This will force the sequence generator blocks to work in unison. If the step timer is being used, a good technique is to take the step taken indicator and current step output from the last sequence generator block for the first eight outputs, and use them as the step trigger and step number inputs for the other

sequence generator blocks used in the sequence. Set the step timers for outputs greater than eight to zero. This way, the first series ***chain*** of sequence generator blocks will be timer controlled, and will drive the remaining blocks in unison.



NOTE: S11, S13, S15, S17, S19, S21, S23 and S25 of blocks 60, 71 and 82 are all nonzero. S11, S13, S15, S17, S19, S21, S23 and S25 of blocks 93, 104 and 115 are all zero.

Figure 161-1 Series and Parallel to Provide Automatic Timed Stepping

With series sequence generator blocks, the sequence generator block handling outputs one through eight must have a lower block address than the block for Steps 9 through 16. Likewise, the sequence generator block Steps 9 through 16 must have a lower block address than the one for Steps 17 through 24. This is true for all chained blocks.
Processing of inputs to sequence generator blocks is done in the following order: reset, disable, jump, step hold, next step, and time hold.
To define the output masks, the operator enters a real value that internally converts to binary digits. The operator defines the output values needed for a given step, then converts them to a real number as shown in Table 161-2. The real number representing the desired output is the step mask for the desired step output.

Table 161-2 Definition of Step Inputs for Sequence Generator Blocks

Types of Values	MSB			Bit number				LSB
	1	2	3	4	5	6	7	8
Weighted decimal value	128	64	32	16	8	4	2	1
Example output values needed for a given step	1	0	1	0	0	1	0	1

161.1.1 Specifications

S1 – PREV
(Block address of previous sequence generator block in series) If the value equals zero, then the block is the first in the series. Each sequence generator block can output eight values for eight steps of a process. If more steps are required, the blocks can be ganged in series. If more than eight outputs are required, the blocks can be ganged in parallel.

S2 – STRIG
(Block address of step trigger) The value in this block controls sequence generator step execution when the sequence generator block is performing sequential stepping. On a zero to one transition of this input, the block will execute the step number which immediately follows the current step number.

S3 – HOLD
(Block address of step hold input) When this input is a one, and steps are being executed sequentially, the advance to the next step is disabled. When the input goes to zero, the next step will be executed. The step hold will not disable the step timer, but will prevent the block from executing the next step once the timer has expired. All blocks in series must have the same step hold input.

0 = execute next step
1 = hold

S4 – THOLD
(Block address of timer hold input) When this input is a one, the step timer is frozen at its current value. An input of zero will cause the timer to resume timing exactly where it left off. All blocks in series must have the same timer hold input.

0 = release
1 = hold

S5 – RESET
(Block address of reset trigger) On a zero to one transition of this input, the block will output the disable mask and reset to Step 0. All sequence generators ganged together in series or parallel must have the same reset input.

S6 – JMPTRG
(Block address of step jump trigger) When the value in the block indicated by this specification makes a zero to one transition, the step indicated by the step jump number (S7) will be executed.

S7 – JMPSTP
(Block address of the step to be executed when the jump step trigger makes a 0 to 1 transition) Jumping to a step number less than zero will cause a jump to Step 0 (same as reset). Jumping to a number higher than the highest step number available in the series ***chain*** will cause the highest available step number to be executed. All sequence generator blocks in a series or parallel chain must have the same jump step number and jump step trigger to operate in unison.

S8 – DISFLG
(Block address of output disable trigger) When this output is equal to one, it sets the outputs equal to the disable mask, but does not move the step number. If the steps are timed, the output disable trigger will hold the block at the current step and freeze the step timer. The output disable trigger does not disable step triggers; the block will still execute the sequence and advance the step number, but the output will be the disable mask. All sequence generators ganged together in series or parallel must have the same disable trigger.

S9 – DISMSK
(Disable mask) Real value which, when subjected to a binary conversion as shown in Table 161-1, provides safe outputs for all devices controlled by the sequence generator block. The disable mask is output in three situations: when the disable input is set, when the block is reset, and after all steps have been completed. In batch control, the disable mask is the same as Step 0 and is reserved as an E-STOP (executed stop).

S10, S12, S14, S16, S18, S20, S22 and S24
(Mask 1 through mask 8) Real values representing the output mask associated with Step 1 (mask one) through Step 8 (mask eight). The output mask is a series of binary digits defining the operating states for all devices associated with the sequence generator block. Refer to Table 161-1 for the definition of step inputs for sequence generator blocks.

S11, S13, S15, S17, S19, S21, S23 and S25
(Time 1 through time 8) Represents the time in seconds that it will take to run the step. The individual step timers start whenever the corresponding step is entered. If a step timer is enabled for a particular step, the sequence generator automatically advances to the next sequential step when the step timer expires. Step timers are disabled when less than or equal to zero. The timer hold and disable flags freeze the step timer. The step hold input does not freeze the step timer, but does disable the advance to the next step when the timer expires. When the step hold is removed, the block will advance to the next step. Adapting the value of a currently running step timer causes the timing to begin again at the adapted value. Tuning any specification during an active step restarts the current step timer to its beginning value.

161.1.2Outputs

N through N+7
Outputs (one through eight) of the current step. The output is a boolean value representing the operating state of the device controlled by the sequence generator block. All outputs should be read from the last sequence generator block in the series.

N+8
Current step number.

N+9
Time remaining in current step in seconds.

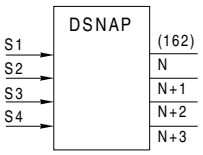
N+10
Step taken indicator. This output reflects whether the next step has been taken. A zero to one transition of this output indicates that the block has begun the next step in the sequence.

161.2Applications

Refer to the applications section of function code 123 for an example of a sequence generator block used in batch control.

162.Digital Segment Buffer

The digital segment buffer function code takes a snapshot of four digital values simultaneously to eliminate inconsistent data within a segment. Any series of segment buffer blocks (analog or digital) unbroken by nonbuffer blocks is an uninterruptible sequence within a segment. Thus, all outputs of the series of blocks are consistent for the duration of the segment cycle.



Outputs

Blk	Type	Description
N	B	Value of first input
N+1	B	Value of second input
N+2	B	Value of third input
N+3	B	Value of fourth input

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	0	I	Note 1	Block address of first input
S2	N	0	I	Note 1	Block address of second input
S3	N	0	I	Note 1	Block address of third input
S4	N	0	I	Note 1	Block address of fourth input

NOTES:

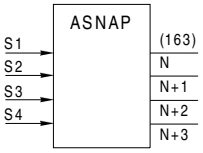
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

162.1Explanation

Use function code 162 in segments that are not high priority. This insures a higher priority segment cannot interrupt the current segment until all four specified inputs update. If a higher priority segment tries to interrupt the current segment after one or more of these inputs update in the current cycle, this block suspends execution of the higher priority segment until it receives the updated values for all four of the inputs. If the higher priority segment interrupts the lower priority segment before any of the values update and while the function is executing, the previous outputs are used.

163.Analog Segment Buffer

The analog segment buffer function code takes a snapshot of four analog values simultaneously. Simultaneous snapshots eliminate the possibility of inconsistent data within a segment. Any series of segment buffer blocks (analog or digital) unbroken by nonbuffer blocks is an uninterruptible sequence within a segment. Thus, all outputs of the series of blocks are consistent for the duration of the segment cycle.



Outputs

Blk	Type	Description
N	R	Value of first input
N+1	R	Value of second input
N+2	R	Value of third input
N+3	R	Value of fourth input

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of first input
S2	N	5	I	Note 1	Block address of second input
S3	N	5	I	Note 1	Block address of third input
S4	N	5	I	Note 1	Block address of fourth input

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

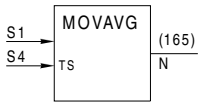
163.1Explanation

Use the analog segment buffer function code in segments that are not the highest priority. This insures a higher priority segment cannot interrupt the current segment until all four of the specified inputs update. If a higher priority segment tries to interrupt the current segment after one or more of these inputs update in the current cycle, this block suspends execution of the higher priority segment until the four values update. If the higher priority segment interrupts the lower one before any of the values update while this function is executing, the previous values are output.

NOTE: The output update is protected from interruption, but the function code execution could be interrupted while reading the input values.

165.Moving Average

The moving average function code computes a moving average from n samples taken at a defined interval. The sample buffer fills with an input value at the end of each cycle. At every sample interval, the current input value replaces the oldest value in the buffer. At each sample interval, a new average is calculated.



Outputs

Blk	Type	Description
N	R	Sum of samples divided by the number of samples

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of averaged value
S2	N	1	I	1-255	Number of samples in moving average
S3	Y	1.000	R	0.0 - 9.2 E18	Sample interval in seconds
S4	N	1	I	0 or 1	Output track switch: 0 = track 1 = normal

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

165.1Explanation

The moving average block calculates the average of *n* values by dividing the sum of the samples by the number of samples. It operates in two modes, normal and track. In the normal mode, the block reads a new input, discards the oldest sample, and calculates a new average each time the sample interval passes. In the track mode, the output equals the input. At each cycle, the buffer fills with the tracked value.
The sample interval and the number of samples used are configurable. The samples are stored in a first in first out (FIFO) buffer. During module startup, the FIFO buffer fills with the input values from each scan cycle. After startup, the moving average function code computes the starting average with the first sample interval. Therefore, the first output is the same as the input (first sample divided by one), the second output is the average of the first two inputs (first and second sample divided by two), and so on.

Example:

If *n* equals five at power up and the input equals 2.0:

2.0	0.0	0.0	0.0	0.0

n = 1 valid sample

$$Output = \frac{2.0 + 0.0 + 0.0 + 0.0 + 0.0}{1} = \frac{2.0}{1} = 2.0$$

On second scan the input equals 4.0:

4.0	2.0	0.0	0.0	0.0

n = 2 valid samples

$$Output = \frac{4.0 + 5.0 + 0.0 + 0.0 + 0.0}{2} = \frac{6.0}{2} = 3.0$$

165.1.1 Specifications

S1 – X
Value of the current input.

S2 – N
Number of samples used in the average calculation.

S3 – INT
Sample interval in seconds. This specification defines the length of time between inputs. It also defines the length of time between calculations, because a new average is calculated each time a new value is added.

S4 – TRCK
Block address of track input. In the track mode, the output tracks the input; there is no operation performed on the input. The tracked values fill the input buffer, but they are not averaged. In the normal mode, the input feeds into a sample buffer and a new output is calculated each time an input is entered.
0 = track
1 = normal

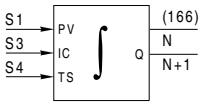
165.1.2 Output

N
Sum of the samples divided by the number of samples in the normal mode. In the track mode, the output equals the input. In both modes, the current calculated values are not retained.

NOTE: In normal mode, the moving average output is valid only after *n* samples have been processed.

166.Integrator

The integrator function code computes the integral of an input signal using the trapezoidal rule of integration and double precision arithmetic. The result of the integration times the gain (S7) is the output. Integration begins at an initial value (S3). High (S5) and low (S6) limits affect the output (N).
Specification S2 specifies the time units: seconds, minutes or hours.
The automatic reset option (S8 equals one) restarts the integration from the initial value when the integrator reaches the high or low limit. In the automatic reset mode, the limit flag is set for one cycle after the integrator reaches a limit. In automatic reset mode, a counter or second integrator can count overflows from the integrator, forming a multistage integration.



Outputs

Blk	Type	Description
N	R	Value of integral
N+1	B	Limit flag: 0 = good 1 = high or low limit reached

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of process variable
S2	N	1	I	0 - 2	Time base of integration: 0 = seconds 1 = minutes 2 = hours
S3	N	5	I	Note 1	Block address of initial value
S4	N	0	I	Note 1	Block address of reset signal: 0 = reset 1 = run
S5	Y	9.2 E18	R	Full	High limit
S6	Y	-9.2 E18	R	Full	Low limit
S7	Y	1.000	R	Full	Gain
S8	Y	0	B	0 or 1	Automatic reset: 0 = off 1 = on
S9	Y	0.000	R	Full	Spare

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

166.1Explanation

The integrator block computes the area under a curve using the trapezoid rule:

$$\int_{t=0}^{t=nh} x(t)dt = h\left[\frac{Y_0}{2} + Y_1 + Y_2 + \ldots + Y_{n-1} + \frac{Y_n}{2}\right]$$

where:

h = Time between sample points on the curve (step size). The integrator block reads the segment control block to determine the scan time for the segment that the integrator block is in.

n = Number of samples taken. The samples are always taken at equal intervals (*h*).

Y = Value of the integrated signal. Y_0 equals the signal at the start of the integration, and Y_n equals the signal at interval *n* of the integration.

The trapezoid rule executes by the integrator block with *h* equal to scan time. This provides the smallest possible step size. Since the error in the trapezoid rule is a function of step size, the smallest step size results in the smallest error. The integrator block then scales the result of the integration based on the value of S2. Specification S2 reflects the time base of the integrated signal. For example, if input flow is in gallons per minute, S2 equals one (minute).

The integrator block operates in two modes: normal and track. In normal mode, the block integrates the values from the input block S1 on the time basis selected with S2. In track mode, the output tracks the input and integration does not take place.

NOTE: The two modes, normal and track, are based on the reset input signal set in S4. When S4 is set to one, integration takes place. When S4 is set to zero, it tracks the S3 input.

166.1.1 Specifications

S1 – X
Block address of the desired signal.

S2 – TB
Time base of the integration.

0 = seconds
1 = minutes
2 = hours

S3 – INIT
Block address of the initial value. The initial value is the output value during start-up and after an automatic or forced reset.

S4 – RESET
Block address of the reset signal. When <S4> equals zero, the integral output N initializes to the initial value <S3>. When <S4> equals one, integration takes place.

S5 – HL
High output limit. When the output reaches or exceeds this value and S8 equals one, the block resets to the initial value.

S6 – LL
Low output limit. When the output reaches or falls below this value and S8 equals one, the block resets to the initial value.

S7 – K
Gain value. This specification scales the output N to a desired value range.

S8 – AR
Automatic reset flag. In the automatic reset mode, the block resets to a defined initial input <S3> value after the output value reaches either limit.

0 = off (no reset)
1 = on (automatic reset)

166.1.2 Outputs

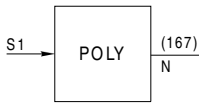
N
Value of the integral. If the value of the integral reaches or exceeds either of the limits and S8 equals zero, the output holds at the limit and the limit flag is set.

N+1
Limit flag. This output is a boolean signal that indicates when the integral output N has reached or exceeded the high or low limit specified by S5 or S6. If the block is in automatic reset mode, this value goes to one for one cycle when the integral reaches either limit. After the cycle, the value returns to zero. If the block is not in automatic reset mode, this value goes to one and remains there as long as the integral is at the limit.

0 = good
1 = limit reached or exceeded

167.Polynomial

The polynomial function code implements a seventh order polynomial.



$$Output = Ax^7 + Bx^6 + Cx^5 + Dx^4 + Ex^3 + Fx^2 + Gx + H$$

Two specifications define the coefficients *A* through *H*:

- Tunable mantissa value.
- Nontunable power of ten.

The actual value of the coefficient is the product of the mantissa and the power of ten. For example:

$$A = S16 \times 10^{S17} = A_{man} \times 10^{Aexp}$$

NOTES:

1. When function code 167 is utilized as a shaping algorithm for analog in/channel (function code 222), its tunable specifications are not adaptable.

2. When function code 167 is used as a shaping algorithm, it can not at the same time also be used as a logic function because the block output will not respond to the specification S1 input.

3. Multiple instances and combinations of function code 177 and 222 function blocks may utilize the same function code 167 function block as a shaping algorithm. The function code 167 shaping algorithm function block is not required to be in the same segment as the function code 177 or function code 222 blocks.

Outputs

Blk	Type	Description
N	R	Result of polynomial evaluation

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of X input
S2	Y	0.000	R	Full	Mantissa of H coefficient
S3	N	0.000	R	Full	Power of 10 of H coefficient
S4	Y	0.000	R	Full	Mantissa of G coefficient
S5	N	0.000	R	Full	Power of 10 of G coefficient
S6	Y	0.000	R	Full	Mantissa of F coefficient
S7	N	0.000	R	Full	Power of 10 of F coefficient
S8	Y	0.000	R	Full	Mantissa of E coefficient
S9	N	0.000	R	Full	Power of 10 of E coefficient
S10	Y	0.000	R	Full	Mantissa of D coefficient
S11	N	0.000	R	Full	Power of 10 of D coefficient
S12	Y	0.000	R	Full	Mantissa of C coefficient
S13	N	0.000	R	Full	Power of 10 of C coefficient
S14	Y	0.000	R	Full	Mantissa of B coefficient

Specifications (Continued)

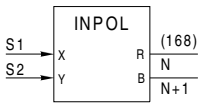
Spec	Tune	Default	Type	Range	Description
S15	N	0.000	R	Full	Power of 10 of B coefficient
S16	Y	0.000	R	Full	Mantissa of A coefficient
S17	N	0.000	R	Full	Power of 10 of A coefficient

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

168.Interpolator

The interpolator function code outputs a value determined by a two dimensional linear interpolation of the two inputs. The coordinates of the values within a defined five-by-five table are the basis of the interpolation. If either input is outside the table, the first output becomes the largest possible number. The second output is a boolean value that indicates if the input points are in the five-by-five table range. If either input is outside the table, the second output becomes a one.



Outputs

Blk	Type	Description
N	R	Interpolated output value
N+1	B	Range of inputs: 0 = good 1 = at least one input is out of range

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	6	I	Note 1	Block address of *X* value
S2	N	6	I	Note 1	Block address of *Y* value
S3	N	0.000	R	Full	Minimum value of *X* (*X* coordinate of Z_{11})
S4	N	100.000	R	Full	Maximum value of *Y* (*Y* coordinate of Z_{11})
S5	N	100.000	R	Full	Maximum value of *X* (*X* coordinate of Z_{55})
S6	N	0.000	R	Full	Minimum value of *Y* (*Y* coordinate of Z_{55})
S7	Y	0.000	R	Full	Z_{11} table entry
S8	Y	0.000	R	Full	Z_{12} table entry
S9	Y	0.000	R	Full	Z_{13} table entry
S10	Y	0.000	R	Full	Z_{14} table entry
S11	Y	0.000	R	Full	Z_{15} table entry
S12	Y	0.000	R	Full	Z_{21} table entry
S13	Y	0.000	R	Full	Z_{22} table entry
S14	Y	0.000	R	Full	Z_{23} table entry
S15	Y	0.000	R	Full	Z_{24} table entry
S16	Y	0.000	R	Full	Z_{25} table entry
S17	Y	0.000	R	Full	Z_{31} table entry
S18	Y	0.000	R	Full	Z_{32} table entry
S19	Y	0.000	R	Full	Z_{33} table entry
S20	Y	0.000	R	Full	Z_{34} table entry
S21	Y	0.000	R	Full	Z_{35} table entry
S22	Y	0.000	R	Full	Z_{41} table entry
S23	Y	0.000	R	Full	Z_{42} table entry
S24	Y	0.000	R	Full	Z_{43} table entry

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S25	Y	0.000	R	Full	Z_{44} table entry
S26	Y	0.000	R	Full	Z_{45} table entry
S27	Y	0.000	R	Full	Z_{51} table entry
S28	Y	0.000	R	Full	Z_{52} table entry
S29	Y	0.000	R	Full	Z_{53} table entry
S30	Y	0.000	R	Full	Z_{54} table entry
S31	Y	0.000	R	Full	Z_{55} table entry

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

168.1Explanation

Interpolation determines the value of the dependent variable *Z* based on the values of the two independent variables *X,Y*. Specifications S3 through S6 specify the high and low limits for the independent variables. The five-by-five table shown in Figure 0-1 is for the interpolation. Input Z_{11} is the value of the dependent variable *Y* that corresponds to the maximum value of the *Y* variable and the minimum value of the *X* variable. Likewise, input Z_{55} is the value of the dependent variablethat

$$\begin{matrix} Z_{11} & Z_{12} & Z_{13} & Z_{14} & Z_{15} \\ Z_{21} & Z_{22} & Z_{23} & Z_{24} & Z_{25} \\ Z_{31} & Z_{32} & Z_{33} & Z_{34} & Z_{35} \\ Z_{41} & Z_{42} & Z_{43} & Z_{44} & Z_{45} \\ Z_{51} & Z_{52} & Z_{53} & Z_{54} & Z_{55} \end{matrix}$$

Figure 0-1. Five-by-Five Array Used by Interpolation Block

corresponds to the minimum value of the *Y* variable and the maximum value of the *X* variable. If either of the dependent variable values go outside the maximum value in the table, the output becomes the largest possible number (Z_{11} and Z_{55}). The increments of *X* must be equal and the increments of *Y* must be equal. *X* and *Y* need not be equal to each other.
This block can easily handle any situation requiring a two dimensional linear interpolation, for example, steam tables. Using temperature and pressure as the *X* and *Y* variables, and enthalpy as the *Z* variable, the user could find the enthalpy associated with any temperature pressure combination that is within the confines of the table.
Implementing this block requires the creation of a lookup table and entering it into the block via S7 through S31. For example, if using this block for enthalpy values, go to the steam tables, and transfer the information needed into the block via the specifications. The values in the table are pre-defined values of the dependent variable that correspond to the designated values of the independent variables.
The block calculates the interpolated output value with the equations:

$$Output = \frac{(Z_{Y1} - Z_{Y2}) \times (Y_{act} - Y_{min})}{Y_{span}} + Z_{Y1}$$

where:

$$Z_{Y1} = \frac{(X_{act} - X_{min}) \times [Z(X_{max}, Y_{min}) - Z(X_{min}, Y_{min})]}{X_{span}} + Z(X_{min}, Y_{max})$$

$$Z_{Y2} = \frac{(X_{act} - X_{min}) \times [Z(X_{min}, Y_{max}) - Z(X_{max}, Y_{max})]}{X_{span}} + Z(X_{min}, Y_{max})$$

X_{act} = Actual value of *X*.

X_{min} = Value of *X* for column before X_{act}.

X_{max} = Value of *X* for column after X_{act}.

X_{span} = Range of *X* values between columns. The range between columns is automatically defined when selecting the minimum and maximum values for *X*. Since the data table is divided into five columns, the range between columns = $\frac{X_{max} - X_{min}}{4}$.

Y_{act} = Actual value of *Y*.

Y_{min} = Value of *Y* for row before Y_{act}.

Y_{max} = Value of *Y* for row after Y_{act}.

Y_{span} = Range of *Y* values between rows. The range between rows is automatically defined when selecting the minimum and maximum values for *Y*. Since the data table is divided into five rows, the range between rows = $\frac{Y_{max} - Y_{min}}{4}$.

$Z(X_{nnn}, Y_{mmm})$ = The *Z* value in the array corresponding to the *X* and *Y* values defined by nnn and mmm.

168.2 Applications

Use this block in any situation requiring the linear interpolation of one variable from the values of two others. External logic is easily implemented to form a large lookup table from multiple interpolator blocks. The following examples illustrate use of the interpolator block for both two dimensional and single dimensional interpolation.

The interpolator block can be used to determine steam properties as mentioned earlier. First, select the range of *X* and *Y*, in this case, pressure (P) and temperature (T). Since steam table data is in absolute pressure, Table 168-3 uses pounds per square inch absolute, and the range of pressure is altered for input to the interpolator block.

Table 168-3 Interpolator Block Determines Steam Properties

P_{gage}	P_{abs} 0 - 600 psia	T 200° - 800°F	s	h
0.304	15 (T_{sat} = 213.0)	200	1.7472	1144.70
		350	1.8437	1216.20
		500	1.9242	1287.30
		650	1.9940	1359.40
		800	2.0563	1433.20
150.304	165 (T_{sat} = 366.0)	200	0.2938	168.44
		350	1.5518	1186.60
		500	1.6485	1272.80
		650	1.7240	1370.70
		800	1.7885	1427.00

Table 168-3 Interpolator Block Determines Steam Properties (Continued)

P_{gage}	P_{abs} 0 - 600 psia	T 200° - 800°F	s	h
300.304	315 (T_{sat} = 421.0)	200	0.2936	168.78
		350	1.4552	1155.90
		500	1.5635	1255.90
		650	1.6465	1341.40
		800	1.6922	1420.70
450.304	465 (T_{sat} = 460.0)	200	0.2935	169.11
		350	1.3829	1121.50
		500	1.5043	1236.20
		650	1.5968	1331.50
		800	1.6667	1414.20
600.304	615 (T_{sat} = 489.0)	200	0.2933	169.45
		350	1.3195	1083.20
		500	1.4542	1213.40
		650	1.5587	1320.90
		800	1.6320	1407.60

The abrupt changes in the original s and h values show that the values are on both sides of the saturation line. Verify this by looking at the saturation temperature (T_{sat}). In order to have valid data down to the saturation line (only in the superheated region), extrapolate valid superheated data down to the next lower temperature. At 15 pounds per square inch absolute, the superheated region permits temperatures down to 213 degrees Fahrenheit. When using the 200 degrees Fahrenheit value of h (168.09) for extrapolation, the interpolator block calculates h at 275 degrees Fahrenheit as 692.1. The correct value is 1180.7.

Obviously, there is a need to determine a new value of the minimum allowable temperature for superheated steam (from steam tables: T = 220, h = 1154.2). By using these two points (1216.2 and 1154.2), extrapolate linearly to get h = 1144.7 at T = 200.

Similar calculations are made for the remaining pressures. The below saturation line value remaining (200 degrees Fahrenheit for P ≥ 165 psia) can be made equal to the 350 degrees Fahrenheit value or remain as they are (current calculations never permit their use). Additional function blocks may be added to monitor below superheated values. The second output of the interpolator block (status) determines if the pressure is less than 15 pounds per square inch absolute. Low pressures cause fixed values of h and s to be output from the block.

Use the interpolator block for single dimensional interpolation by setting the range of y equals zero to ten, and fixing the y input (S2) at 0.0. Then only one row (y = 0.0) needs filled in. This function block will linearly interpolate between two values in its table. For curve segments that do not have inflection points (second derivative crossing through zero), the resulting error always has the same sign as shown in Figure 168-1. For applications requiring greater accuracy, use a function generator block. When linearizing the curve with a function generator block, the straight lines are selected such that errors are equal on both sides of the curve as shown in Figure 168-1. If a segment requires greater accuracy (such as around an operating point), the function generator breakpoints can be spaced closer together.

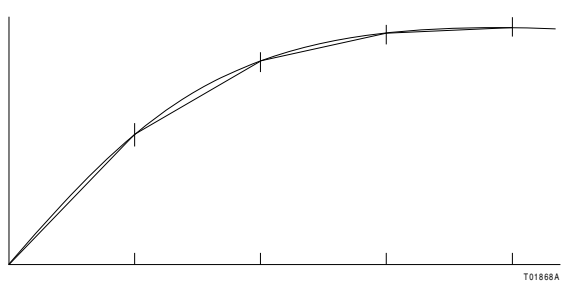


Figure 168-1 Approximation of Curve Using Interpolator

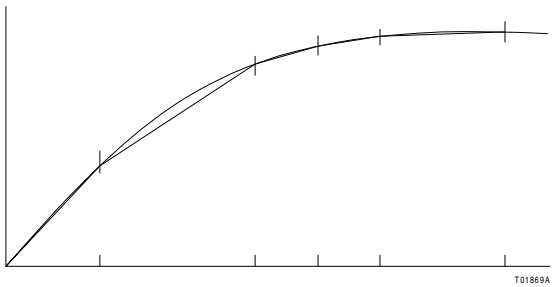
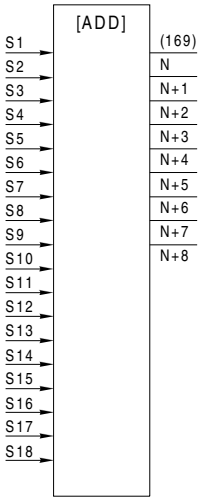


Figure 168-1 Approximation of Curve Using Function Generator

169.Matrix Addition

The matrix addition function code adds two three-by-three matrices to provide a three-by-three matrix output. Use this operation for modeling and simulations, and advanced control strategies.



Outputs

Blk	Type	Description
N	R	Sum of A_{11} + B_{11} or <S1> + <S10>
N+1	R	Sum of A_{12} + B_{12} or <S2> + <S11>
N+2	R	Sum of A_{13} + B_{13} or <S3> + <S12>
N+3	R	Sum of A_{21} + B_{21} or <S4> + <S13>
N+4	R	Sum of A_{22} + B_{22} or <S5> + <S14>
N+5	R	Sum of A_{23} + B_{23} or <S6> + <S15>
N+6	R	Sum of A_{31} + B_{31} or <S7> + <S16>
N+7	R	Sum of A_{32} + B_{32} or <S8> + <S17>
N+8	R	Sum of A_{33} + B_{33} or <S9> + <S18>

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of A_{11} value
S2	N	5	I	Note 1	Block address of A_{12} value
S3	N	5	I	Note 1	Block address of A_{13} value
S4	N	5	I	Note 1	Block address of A_{21} value
S5	N	5	I	Note 1	Block address of A_{22} value
S6	N	5	I	Note 1	Block address of A_{23} value
S7	N	5	I	Note 1	Block address of A_{31} value
S8	N	5	I	Note 1	Block address of A_{32} value
S9	N	5	I	Note 1	Block address of A_{33} value
S10	N	5	I	Note 1	Block address of B_{11} value
S11	N	5	I	Note 1	Block address of B_{12} value
S12	N	5	I	Note 1	Block address of B_{13} value
S13	N	5	I	Note 1	Block address of B_{21} value
S14	N	5	I	Note 1	Block address of B_{22} value
S15	N	5	I	Note 1	Block address of B_{23} value
S16	N	5	I	Note 1	Block address of B_{31} value
S17	N	5	I	Note 1	Block address of B_{32} value

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S18	N	5	I	Note 1	Block address of B_{33} value

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

169.1Explanation

Figure 169-1 shows how this block adds the two matrices.
The results of the matrix addition are real values that are output from the block as shown in Figure 169-1. The values in the matrices are defined by their block addresses. The *A* matrix is filled in numerical order, followed by the *B* matrix.

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} + \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} = \begin{bmatrix} A_{11}+B_{11} & A_{12}+B_{12} & A_{13}+B_{13} \\ A_{21}+B_{21} & A_{22}+B_{22} & A_{23}+B_{23} \\ A_{31}+B_{31} & A_{32}+B_{32} & A_{33}+B_{33} \end{bmatrix}$$

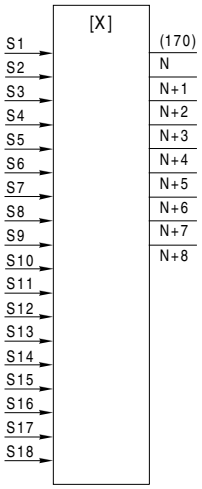
or

$$\begin{bmatrix} \langle S1\rangle & \langle S2\rangle & \langle S3\rangle \\ \langle S4\rangle & \langle S5\rangle & \langle S6\rangle \\ \langle S7\rangle & \langle S8\rangle & \langle S9\rangle \end{bmatrix} + \begin{bmatrix} \langle S10\rangle & \langle S11\rangle & \langle S12\rangle \\ \langle S13\rangle & \langle S14\rangle & \langle S15\rangle \\ \langle S16\rangle & \langle S17\rangle & \langle S18\rangle \end{bmatrix} = \begin{bmatrix} \mathrm{N} & N+1 & N+2 \\ N+3 & N+4 & N+5 \\ N+6 & N+7 & N+8 \end{bmatrix}$$

Figure 169-1 Matrix Addition

170.Matrix Multiplication

The matrix multiplication function code multiplies a pair of three-by-three matrices. The output is a three-by-three matrix. Like matrix addition (function code 169), use matrix multiplication for system modeling and simulation. The outputs show the effects of various values of several different parameters.
Matrix multiplication can implement advanced control strategies that incorporate several independent variables.



Outputs

Blk	Type	Description
N	R	Product $(A_{11} \times B_{11}) + (A_{12} \times B_{21}) + (A_{13} \times B_{31})$ or (<S1> × <S10>) + (<S2> × <S13>) + (<S3> × <S16>)
N+1	R	Product $(A_{11} \times B_{12}) + (A_{12} \times B_{22}) + (A_{13} \times B_{32})$ or (<S1> × <S11>) + (<S2> × <S14>) + (<S3> × <S17>)
N+2	R	Product $(A_{11} \times B_{13}) + (A_{12} \times B_{23}) + (A_{13} \times B_{33})$ or (<S1> × <S12>) + (<S2> × <S15>) + (<S3> × <S18>)
N+3	R	Product $(A_{21} \times B11) + (A_{22} \times B_{21}) + (A_{23} \times B_{31})$ or (<S4> × <S10>) + (<S5> × <S13>) + (<S6> × <S16>)
N+4	R	Product $(A_{21} \times B_{12}) + (A_{22} \times B_{22}) + (A_{23} \times B_{32})$ or (<S4> × <S11>) + (<S5> × <S14>) + (<S6> × <S17>)
N+5	R	Product $(A_{21} \times B_{13}) + (A_{22} \times B_{23}) + (A_{23} \times B_{33})$ or (<S4> × <S12>) + (<S5> × <S15>) + (<S6> × <S18>)
N+6	R	Product $(A_{31} \times B11) + (A_{32} \times B_{21}) + (A_{33} \times B_{31})$ or (<S7> × <S10>) + (<S8> × <S13>) + (<S9> × <S16>)
N+7	R	Product $(A_{31} \times B_{12}) + (A_{32} \times B_{22}) + (A_{33} \times B_{32})$ or (<S7> × <S11>) + (<S8> × <S14>) + (<S9> × <S17>)
N+8	R	Product $(A_{31} \times B_{13}) + (A_{32} \times B_{23}) + (A_{33} \times B_{33})$ or (<S7> × <S12>) + (<S8> × <S15>) + (<S9> × <S18>)

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of A_{11} value
S2	N	5	I	Note 1	Block address of A_{12} value
S3	N	5	I	Note 1	Block address of A_{13} value
S4	N	5	I	Note 1	Block address of A_{21} value
S5	N	5	I	Note 1	Block address of A_{22} value
S6	N	5	I	Note 1	Block address of A_{23} value
S7	N	5	I	Note 1	Block address of A_{31} value
S8	N	5	I	Note 1	Block address of A_{32} value
S9	N	5	I	Note 1	Block address of A_{33} value
S10	N	5	I	Note 1	Block address of B_{11} value
S11	N	5	I	Note 1	Block address of B_{12} value
S12	N	5	I	Note 1	Block address of B_{13} value
S13	N	5	I	Note 1	Block address of B_{21} value

Specifications (Continued)

Spec	Tune	Default	Type	Range	Description
S14	N	5	I	Note 1	Block address of B_{22} value
S15	N	5	I	Note 1	Block address of B_{23} value
S16	N	5	I	Note 1	Block address of B_{31} value
S17	N	5	I	Note 1	Block address of B_{32} value
S18	N	5	I	Note 1	Block address of B_{33} value

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

170.1Explanation

The matrix multiplication function code multiplies two three-by-three matrices to form a three-by-three matrix of real values. Matrices multiply row by column. To form the first row of the product matrix, row one of matrix A multiplies by columns one, two and three of matrix B. The second and third rows of the product matrix form similarly. Row two of matrix A multiplies by columns one, two and three of matrix B to form the second row of the product matrix and row three of matrix A multiplies by columns one, two and three of matrix B to form the last row of the product matrix.
The row by column multiplication sums the products of the like elements to get one value. The first value in row one of matrix A multiplies by the first value in column one of matrix B. That product adds to the products of the second and third values to produce the value in the product matrix as Figure 170-1 shows.

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \times \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ B_{31} & B_{32} & B_{33} \end{bmatrix} = \begin{bmatrix} A_{11}B_{11} + A_{12}B_{21} + A_{13}B_{31} & A_{11}B_{12} + A_{12}B_{22} + A_{13}B_{32} & A_{11}B_{13} + A_{12}B_{23} + A_{13}B_{33} \\ A_{21}B_{11} + A_{22}B_{21} + A_{23}B_{31} & A_{21}B_{12} + A_{22}B_{22} + A_{23}B_{32} & A_{21}B_{13} + A_{22}B_{23} + A_{23}B_{33} \\ A_{31}B_{11} + A_{32}B_{21} + A_{33}B_{31} & A_{31}B_{12} + A_{32}B_{22} + A_{33}B_{32} & A_{31}B_{13} + A_{32}B_{23} + A_{33}B_{33} \end{bmatrix}$$

OR

$$\begin{bmatrix} (S1) & (S2) & (S3) \\ (S4) & (S5) & (S6) \\ (S7) & (S8) & (S9) \end{bmatrix} \times \begin{bmatrix} (S10) & (S11) & (S12) \\ (S13) & (S14) & (S15) \\ (S16) & (S17) & (S18) \end{bmatrix} = \begin{bmatrix} N & N+1 & N+2 \\ N+3 & N+4 & N+5 \\ N+6 & N+7 & N+8 \end{bmatrix}$$

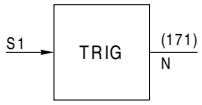
T01967A

Figure 170-1 Internal Operation of Matrix Multiplication Block

171.Trigonometric

The trigonometric function code calculates the standard trigonometric functions.

- Sine.
- Cosine.
- Tangent.
- Cotangent.
- Secant.
- Cosecant.



The input may be expressed in either degrees or radians. The output is the selected trigonometric function of the input value multiplied by the gain factor.
Use trigonometric functions for performance calculations and monitoring.

Outputs

Blk	Type	Description
N	R	Trigonometric value multiplied by the gain factor

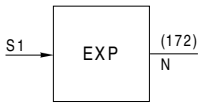
Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input
S2	N	0	I	0 or 1	Type in input <S1>: 0 = radians 1 = degrees
S3	N	0	I	0 - 5	Trigonometric function performed on input <S1>: 0 = sine 3 = cotangent 1 = cosine 4 = secant 2 = tangent 5 = cosecant
S4	Y	1.000	R	Full	Gain factor

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

172.Exponential

The exponential function code raises e to the power specified by the input <S1>. The result is multiplied by the gain factor (S2).
Use the exponential function code in process control for performance calculations and monitoring capability.



Outputs

Blk	Type	Description
N	R	Value = K × e^x

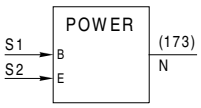
Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	5	I	Note 1	Block address of input value *X*
S2	Y	1.000	R	Full	Gain factor

NOTES:
1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

173.Power

The power function code raises the value of the first input <S1> to the power specified by the second input <S2>. The result is multiplied by the gain factor (S3) to provide the output.
Use the power function code in process control for performance calculations and monitoring.



Outputs

Blk	Type	Description
N	R	Value = K × $(y)^x$

Specifications

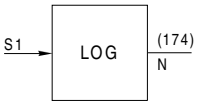
Spec	Tune	Default	Type	Range	Description
S1	N	6	I	Note 1	Block address of *y*
S2	N	6	I	Note 1	Block address of *x*
S3	Y	1.000	R	Full	Gain factor

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

174.Logarithm

The logarithm function code takes the logarithm of the input to the defined base. The result is multiplied by a gain factor to provide the output. The logarithm function code is not limited to natural logarithms. The log of any value in base zero (natural) to ten can be taken.
Use the logarithm function code in process control for performance calculations and monitoring.



Outputs

Blk	Type	Description
N	R	Value = K × [$\log_{(base)}$(X)]

Specifications

Spec	Tune	Default	Type	Range	Description
S1	N	6	I	Note 1	Block address of input value, *X*
S2	N	0	I	0 - 10	Base of logarithm: 0 = natural log
S3	Y	1.000	R	Full	Gain factor

NOTES:

1. Maximum values are:9,998 for the BRC-100/200/300, IMMFP11/12
31,998 for the HC800, BRC-400/410, HPG800 and HAC

A. List of Function Codes

A.1 Introduction

Appendix A contains cross references for all the function codes. Table A-1 lists the function codes numerically along with a description of each code.
Table A-2 is an alphabetical listing of the function codes by function code description and is followed by the function code number.
Tables A-3 through A-45 categorize the function codes by functionality. Note that in these tables function codes may be listed in multiple categories.

A.2 Cross Reference - Numerical

Table A-1 Numerical Listing

Function Code	Description	Function Code	Description
1	Function generator	38	AND (4-input)
2	Manual set constant (signal generator)	39	OR (2-input)
3	Lead/lag	40	OR (4-input)
4	Pulse positioner	41	Digital input (periodic sample)
5	Pulse rate	42	Digital input/loop
6	High/low limiter	45	Digital exception report
7	Square root	48	Analog Exception Report with High/Low Alarm deadband
8	Rate limiter	50	Manual set switch
9	Analog transfer	51	Manual set constant
10	High select	52	Manual set integer
11	Low select	55	Hydraulic servo
12	High/low compare	57	Reserved for future use
13	Integer transfer	58	Time delay (analog)
14	Summer (4-input)	59	Digital transfer
15	Summer (2-input)	61	Blink
16	Multiply	62	Remote control memory
17	Divide	63	Analog input list (periodic sample)
18	PID error input	64	Digital input list (periodic sample)
19	PID (PV and SP)	65	Digital sum
24	Adapt	66	Analog trend
25	Analog input (same PCU node)	67	Digital Exception Report with Alarm Deadband
26	Analog input/loop	68	Remote manual set constant
30	Analog exception report	69	Test alarm
31	Test quality	79	Control interface slave
32	Trip	80	Control station
33	Not	81	Execute
34	Memory	82	Segment control
35	Timer	83	Digital output group
36	Qualified OR (8-input)	84	Digital input group
37	AND (2-input)	85	Up/down counter

Table A-1 Numerical Listing (Continued)

Function Code	Description	Function Code	Description
84	Digital input group	123	Device driver
85	Up/down counter	124	Sequence monitor
88	Digital logic station	125	Device monitor
89	Last block	126	Real signal demultiplexer
90	Extended executive	128	Slave default definition
91	BASIC configuration (BRC-100/200)	129	Multistate device driver
92	Invoke BASIC	132	Analog input/slave
93	BASIC real output	133	Smart field device definition
94	BASIC boolean output	134	Multi-sequence monitor
95	Module status monitor	135	Sequence manager
96	Redundant analog input	136	Remote motor control
97	Redundant digital input	137	C and BASIC program real output with quality
98	Slave select	138	C or BASIC program boolean output with quality
99	Sequence of events log	139	Passive station interface
100	Digital output readback check	140	Restore
101	Exclusive OR	141	Sequence master
102	Pulse input/period	142	Sequence slave
103	Pulse input/frequency	143	Invoke C
104	Pulse input/totalization	144	C allocation
109	Pulse input/duration	145	Frequency counter/slave
110	Rung (5-input)	146	Remote I/O interface
111	Rung (10-input)	147	Remote I/O definition
112	Rung (20-input)	148	Batch sequence
114	BCD input	149	Analog output/slave
115	BCD output	150	Hydraulic servo slave
116	Jump/master control relay	151	Text selector
117	Boolean recipe table	152	Model parameter estimator
118	Real recipe table	153	ISC parameter converter
119	Boolean signal multiplexer	154	Adaptive parameter scheduler
120	Real signal multiplexer	155	Regression
121	Analog input/Cnet	156	Advanced PID controller
122	Digital input/Cnet	157	General digital controller

Table A-1 Numerical Listing (Continued)

Function Code	Description	Function Code	Description
160	Inferential smith controller	202	Remote transfer module executive block (INIIT02)
161	Sequence generator	203	INIPT02 executive block
162	Digital segment buffer	205	Analog Input List/CW800
163	Analog segment buffer	206	Digital Input List/CW800
165	Moving average	207	Module Status Monitor/CW800
166	Integrator	210	Sequence of events slave
167	Polynomial	211	Data acquisition digital
168	Interpolator	212	Data acquisition digital input/loop
169	Matrix addition	215	Enhanced analog slave definition
170	Matrix multiplication	216	Enhanced analog input definition
171	Trigonometric	217	Enhanced calibration command
172	Exponential	218	Phase execution
173	Power	219	Common sequence
174	Logarithm	220	Batch historian
177	Data acquisition analog	221	I/O Device definition
178	Data acquisition analog input/loop	222	Analog in/channel
179	Ehanced trend	223	Analog out/channel
184	Factory instrumentation protocol handler	224	Digital in/channel
185	Digital input subscriber	225	Digital out/channel
186	Analog input subscriber	226	Test status
187	Analog output subscriber	227	Gateway
188	Digital output subscriber	228	Foreign Device Definition
190	User defined function declaration	229	Pulse In/Channel
191	User defined function one	241	DSOE data interface
192	User defined function two	242	DSOE digital event interface
193	User defined data import	243	Executive block (SEM01/11)
194	User defined data export	244	Addressing interface definition
198	Auxiliary real user defined function	245	Input channel interface
199	Auxiliary digital user defined function	246	Trigger definition
201	Data point definition	247	Condition Monitoring

A.3 Cross Reference - Alphabetical

Table A-2 Alphabetical Listing

Description	Function Code	Description	Function Code
Adapt	24	BCD input	114
Adaptive parameter scheduler	154	BCD output	115
Addressing interface definition	244	Blink	61
Advanced PID controller	156	Boolean recipe table	117
Analog exception report	30	Boolean signal multiplexer	119
Analog exception report with high/low alarm deadband	48	C allocation	144
Analog in/channel	222	C and BASIC program real output with quality	137
Analog input/Cnet	121	Common sequence	219
Analog input list (same PCU)	63	Control interface slave	79
Analog Input List/CW800	205	Control station	80
Analog input/loop	26	C or BASIC program boolean output with quality	138
Analog input (same PCU node)	25	Data acquisition analog	177
Analog input/slave	132	Data acquisition analog input/loop	178
Analog input subscriber	186	Data acquisition digital	211
Analog out/channel	223	Data acquisition digital input/loop	212
Analog output/slave	149	Data point definition	201
Analog output subscriber	187	Device driver	123
Analog segment buffer	163	Device monitor	125
Analog transfer	9	Digital exception report	45
Analog trend	66	Digital exception report with alarm deadband	67
AND (2-input)	37	Digital in/channel	224
AND (4-input)	38	Digital input (periodic sample)	41
Auxiliary digital user defined function	199	Digital input group	84
Auxiliary real user defined function	198	Digital input/Cnet	122
BASIC boolean output	94	Digital input list (periodic sample)	64
BASIC configuration	91	Digital Input List/CW800	206
BASIC real output	93	Digital input/loop	42
Batch historian	220	Digital input subscriber	185
Batch sequence	148	Digital logic station	88

Table A-2 Alphabetical Listing (Continued)

Description	Function Code	Description	Function Code
Digital logic station interface	87	Inferential smith controller	160
Digital out/channel	225	INIPT02 executive block	203
Digital output group	83	Input channel interface	245
Digital output readback check	100	Integer transfer	13
Digital output subscriber	188	Integrator	166
Digital segment buffer	162	Interpolator	168
Digital sum with gain	65	Invoke BASIC	92
Digital transfer	59	Invoke C	143
Divide	17	I/O device definition	221
DSOE data interface	241	ISC parameter converter	153
DSOE digital event interface	242	Jump/master control relay	116
Elapsed time	86	Last block	89
Enhanced analog input defintion	216	Lead/lag	3
Enhanced analog slave defintion	215	Logarithm	174
Enhanced calibration command	217	Low select	11
Enhanced trend	179	Manual set constant	51
Exclusive OR	101	Manual set constant (signal generator)	2
Executive	81	Manual set integer	52
Executive block (SEM01/11)	243	Manual set switch	50
Exponential	172	Matrix addition	169
Extended executive	90	Matrix multiplication	170
Factory instrumentation protocol handler	184	Memory	34
Foreign Device Definition	228	Model parameter estimator	152
Frequency counter/slave	145	Module Status Monitor	95
Function generator	1	Module Status Monitor/CW800	207
Gateway	227	Moving average	165
General digital controller	157	Multiply	16
High/low compare	12	Not	33
High/low limiter	6	OR (4-input)	40
Hydraulic servo	55	OR (2-input)	39
Hydraulic servo slave	150	Passive station interface	139
High select	10	Phasex	218

Table A-2 Alphabetical Listing (Continued)

Description	Function Code	Description	Function Code
PID error input	18	Segment control (IMCOM04)	164
PID (PV and SP)	19	Sequence generator	161
Polynomial	167	Sequence manager	135
Power	173	Sequence master	141
Pulse In/Channel	229	Sequence monitor	124
Pulse input/duration	109	Sequence of events log	99
Pulse input/frequency	103	Sequence of events slave	210
Pulse input/period	102	Sequence slave	142
Pulse input/totalization	104	Slave default definition	128
Pulse positioner	4	Slave select	98
Pulse rate	5	Smart field device definition	133
Qualified OR (8-input)	36	Square root	7
Rate limiter	8	Summer (2-input)	15
Real recipe table	118	Summer (4-input)	14
Real signal demultiplexer	126	Test alarm	69
Real signal multiplexer	120	Test quality	31
Redundant analog input	96	Test status	226
Redundant digital input	97	Text selector	151
Regression	155	Time delay (analog)	58
Remote control memory	62	Timer	35
Remote I/O definition	147	Trigger definition	246
Remote I/O interface	146	Trigonometric	171
Remote manual set constant	68	Trip	32
Remote motor control	136	Up/down counter	85
Remote transfer module executive block (INIIT02)	202	User defined data export	194
Restore	140	User defined data import	193
Rung (5-input)	110	User defined function declaration	190
Rung (10-input)	111	User defined function one	191
Rung (20-input)	112	User defined function two	192
Segment control	82		

A.4 Cross Reference - Categorization

Function codes may be listed in multiple categories.

Table A-3 Adapt

Function Code	Description
24	Adapt

Table A-4 Advanced Functions

Function Code	Description	Function Code	Description
152	Model parameter estimator	167	Polynomial
153	ISC parameter converter	168	Interpolator
154	Adaptive parameter scheduler	169	Matrix addition
157	General digital controller	170	Matrix multiplication
160	Inferential smith controller	171	Trigonometric
162	Digital segment buffer	172	Exponential
163	Analog segment buffer	173	Power
165	Moving average	174	Logarithm
166	Integrator		

Table A-5 BASIC Language

Function Code	Description	Function Code	Description
91	BASIC configuration	94	BASIC boolean output
92	Invoke BASIC	137	C and BASIC program real output with quality
93	BASIC real output	138	C or BASIC program boolean output with quality

Table A-6 Batch Functions

Function Code	Description	Function Code	Description
117	Boolean recipe table	129	Multistate device driver
118	Real recipe table	134	Multi-sequence monitor
119	Boolean signal multiplexer	135	Sequence manager
120	Real signal multiplexer	141	Sequence master
123	Device driver	142	Sequence slave
124	Sequence monitor	161	Sequence generator
125	Device monitor	218	Phase execution
126	Real signal demultiplexer		

Table A-7 Batch Language

Function Code	Description	Function Code	Description
93	BASIC real output	148	Batch sequence
94	BASIC boolean output	218	Phase execution
137	C and BASIC program real output with quality	219	Common sequence
138	C or BASIC program boolean output with quality	220	Batch historian

Table A-8 C Language

Function Code	Description	Function Code	Description
93	BASIC real output	138	C or BASIC program boolean output with quality
94	BASIC boolean output	143	Invoke C
137	C and BASIC program real output with quality	144	C allocation

Table A-9 Communications

Function Code	Description
201	Data point definition
203	INIPT02 executive block
202	Remote transfer module executive block (INIIT02)

Table A-10 Computing

Function Code	Description	Function Code	Description
1	Function generator	17	Divide
2	Manual set constant (signal generator)	51	Manual set constant
3	Lead/lag	52	Manual set integer
5	Pulse rate	58	Time delay (analog)
6	High/low limiter	65	Digital sum with gain
7	Square root	155	Regression
8	Rate limiter	171	Trigonometric
14	Summer (4-input)	172	Exponential
15	Summer (2-input)	173	Power
16	Multiply	174	Logarithm

Table A-11 Controlway/Module Bus/CW800 and Peer-to-Peer Network I/O

Function Code	Description	Function Code	Description
25	Analog input (periodic sample)	95	Module status monitor
41	Digital input (periodic sample)	205	Analog Input List/CW800
63	Analog input list (periodic sample)	206	Digital Input List/CW800
64	Digital input list (periodic sample)	207	Module Status Monitor/CW800

Table A-12 Control Function Blocks

Function Code	Description	Function Code	Description
4	Pulse positioner	24	Adapt
18	PID error input	156	Advanced PID controller
19	PID (PV and SP)	160	Inferential smith controller

Table A-13 Exception Report

Function Code	Description	Function Code	Description
26	Analog input/loop	136	Remote motor control ((RMC)
30	Analog exception report	193	User defined data import
42	Digital input/loop	194	User defined data export
45	Digital Exception Report	211	Data acquisition digital
48	Analog exception report with high/low deadband	212	Data acquisition digital input/loop
62	Remote control memory	218	Phase execution
67	Digital exception report with alarm deadband	221	I/O device definition
68	Remote manual set constant	222	Analog in/channel
80	Control station	223	Analog out/channel
121	Analog input/Cnet	224	Digital in/channel
122	Digital input/Cnet	225	Digital out/channel
123	Device driver	229	Pulse in/channel
129	Multistate device driver		

Table A-14 Executive

Function Code	Description	Function Code	Description
57	Node statistics block	89	Last block

Table A-14 Executive

Function Code	Description	Function Code	Description
81	Executive	90	Extended executive
82	Segment control		

Table A-15 Factory Instrumentation Protocol

Function Code	Description	Function Code	Description
184	Factory instrumentation protocol handler	187	Analog output subscriber
185	Digital input subscriber	188	Digital output subscriber
186	Analog input subscriber		

Table A-16 Field I/O

Function Code	Description	Function Code	Description
55	Hydraulic servo	128	Slave default definition
79	Control interface slave	132	Analog input/slave
83	Digital output group	133	Smart field device definition
84	Digital input group	145	Frequency counter/slave
96	Redundant analog input	146	Remote I/O interface
97	Redundant digital input	147	Remote I/O definition
102	Pulse input/period	149	Analog output/slave
103	Pulse input/frequency	150	Hydraulic servo slave
104	Pulse input/totalization	210	Sequence of events slave
109	Pulse input/duration	215	Enhanced analog slave definition
114	BCD input	216	Enhanced analog input definition
115	BCD output	217	Enhanced calibration command

Table A-17 Harmony I/O and Foreign Devices (HART & PROFIBUS)

Function Code	Description	Function Code	Description
221	I/O device definition	226	Test status
222	Analog in/channel	227	Gateway
223	Analog out/channel	228	Foreign device definition
224	Digital in/channel	229	Pulse in/channel (S800 - DP820)
225	Digital out/channel		

Table A-18 Harmony Bridge Controller Executive

Function Code	Description	Function Code	Description
81	Executive	89	Last block
82	Segment control	90	Extended executive

Table A-19 Ladder Logic

Function Code	Description	Function Code	Description
110	Rung (5-input)	112	Rung (20-input)
111	Rung (10-input)	116	Jump/master control relay

Table A-20 Logic

Function Code	Description	Function Code	Description
33	Not	61	Blink
34	Memory	85	Up/down counter
35	Timer	86	Elapsed timer
36	Qualified OR (8-input)	101	Exclusive OR
37	AND (2-input)	110	Rung (5-input)
38	AND (4-input)	111	Rung (10-input)
39	OR (2-input)	112	Rung (20-input)
40	OR (4-input)	123	Device driver
50	Manual set switch	129	Multistate device driver
59	Digital transfer		

Table A-21 I/O Expander Bus

Function Code	Description	Function Code	Description
55	Hydraulic Servo	147	Remote I/O definition
79	Control interface slave	149	Analog output/slave
83	Digital output group	150	Hydraulic servo slave
84	Digital input group	184	Factory instrumentation protocol handler
87	Digital logic station interface	185	Digital input subscriber
88	Digital input group	186	Analog input subscriber
102	Pulse input/period	187	Analog output subscriber
103	Pulse input/frequency	188	Digital output subscriber
104	Pulse input/totalization	210	Sequence of events slave

Table A-21 I/O Expander Bus (Continued)

Function Code	Description	Function Code	Description
109	Pulse input/duration	215	Enhanced analog slave definition
114	BCD input	216	Enhanced analog input definition
115	BCD output	217	Enhanced calibration command
128	Slave default definition	241	DSOE data interface
132	Analog input/slave	242	DSOE digital event interface
140	Restore[1]	247	Condition monitoring
145	Frequency counter/slave		
146	Remote I/O interface		

NOTE: 1. Restore uses expander bus only if a timer input is specified (S6 not equal to 000).

Table A-22 Restore

Function Code	Description
140	Restore

Table A-23 Sequence Command

Function Code	Description
82	Segment control
90	Extended executive

Table A-24 Sequence of Events

Function Code	Description	Function Code	Description
99	Sequence of events log	243	Executive block (SEM01/11)
210	Sequence of events slave	244	Addressing interface definition
241	DSOE data interface SEM to BRC	245	Input channel interface
242	DSOE digital event interface	246	Trigger definition

Table A-25 Signal Select

Function Code	Description	Function Code	Description
9	Analog transfer	59	Digital transfer
10	High select	162	Digital segment buffer
11	Low select	163	Analog segment buffer
13	Integer transfer		

Table A-26 Signal Status

Function Code	Description	Function Code	Description
12	High/low compare	98	Slave select
31	Test quality	100	Digital output readback check
69	Test alarm		

Table A-27 Station

Function Code	Description	Function Code	Description
62	Remote control memory	123	Device driver
68	Remote manual set constant	129	Multistate device driver
80	Control station	136	Remote motor control
87	Digital logic station interface	139	Passive station interface
88	Digital logic station		

Table A-28 Text Selector

Function Code	Description
151	Text selector

Table A-29 Trend

Function Code	Description
66	Analog trend
179	Enhanced trend

Table A-30 Trip

Function Code	Description
32	Trip

Table A-31 User Defined Function

Function Code	Description	Function Code	Description
190	User defined function declaration	194	User defined data export
191	User defined function one	198	Auxiliary real user defined function
192	User defined function two	199	Auxiliary digital user defined function
193	User defined data import		

B. Harmony Bridge Controller (BRC-300/400/410) and HPG800

B.1 Memory Utilization and Execution Times

This section lists the module memory requirements for each function code. Three quantities are given for the BRC-300, BRC-400, BRC-410 and HPG800 memory utilization:

- The number of bytes of nonvolatile random access memory (NVRAM). The BRC-300 module has a total configuration memory of 415 kilobytes of NVRAM. The BRC-400, BRC-410 and HPG800 modules have a total configuration memory of 1.9 megabytes of NVRAM.
- The number of bytes of random access memory (RAM). The BRC-300, BRC-400, BRC-410 and HPG800 modules have a total configuration memory of 7.56 megabytes of RAM.
- The checkpoint utilization byte size.

This section also lists the function code execution times (in microseconds) for the BRC-300, BRC-400, BRC-410 and HPG800.

NOTE: Except where otherwise noted, execution times are given for worst case conditions.

Table B-1 shows the BRC-300, BRC-400, BRC-410 and HPG800 memory requirements and the execution time for each function code.

NOTE: Refer to ***Memory Usage Equations*** in this section for the equations listed in Table B-1.

Table B-1 BRC-300/400/410 and HPG800 Memory Utilization and Execution Times

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
1	Function generator	46	96	12	6
2	Manual set constant	12	44	12	2
3	Lead/lag	18	66	22	8
4	Pulse positioner	24	84	24	4
5	Pulse rate	16	70	18	7
6	High/low limiter	16	52	12	3
7	Square root	14	52	12	24
8	Rate limiter	18	62	18	7
9	Analog transfer	20	70	22	14
10	High select	16	56	12	4
11	Low select	16	56	12	4
12	High/low compare	16	52	12	3
13	Integer transfer	14	50	10	2
14	Four input summer	16	56	12	6
15	Two input summer	18	56	12	7
16	Multiply	14	52	12	6
17	Divide	14	52	12	7
18	PID error input	34	98	26	44
19	PID process variable and set point	36	102	30	48
24	Adapt	12	54	12	2
25	Analog input (periodic sample)	12	110	38	3

Table B-1 BRC-300/400/410 and HPG800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
26	Analog input/loop	12	66	20	4
30	Analog exception report	26	120	40	8
31	Test quality	16	58	10	5
32	Trip	12	42	10	2
33	Not	12	42	10	2
34	Memory	14	52	10	2
35	Timer	14	56	18	7
36	Qualified OR (eight input)	26	74	10	7
37	AND (two input)	12	46	10	2
38	AND (four input)	16	54	10	3
39	OR (two input)	12	46	10	2
40	OR (four input)	16	54	10	3
41	Digital input/bus (periodic sample)	12	98	32	4
42	Digital input/loop	12	60	14	3
45	Digital exception report	12	88	32	7
48	Analog Exception Report with High/Low Alarm Deadband	32	126	72	9
50	Manual set switch	8	40	10	2
51	Manual set constant (non tunable)	12	36	8	2
52	Manual set integer	12	36	8	2
55	Hydraulic servo	62	430	88	1,723
58	Time delay (analog)	18	Equation 3	22	8
59	Digital transfer	14	50	10	2
61	Blink	12	50	14	8
62	Remote control memory	28	104	22	11
63	Analog input list (periodic sample)	28	294	122	11
64	Digital input list (periodic sample)	28	198	74	11
65	Digital sum with gain (four input)	28	72	12	8
66	Analog trend Normal mode (slow) Fast mode	 12 12	 376 376	 340 340	 6 4
67	Digital Exception Report with Alarm Deadband	16	102	48	8
68	Remote manual set constant (REMSET)	22	122	36	9

Table B-1 BRC-300/400/410 and HPG800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
69	Test alarm	12	48	12	4
79	Control interface slave First FC 79 block in segment Other FC 79 blocks in segment	 50 50	 354 354	 68 68	 15,430 570
80	Control station	84	382	76	24
81	Executive block	22	272	6	0
82	Segment control	64	300	48	0
83	Digital output group	32	120	12	98
84	Digital input group	16	144	44	97
85	Up/down counter	24	92	18	5
86	Elapsed timer	20	100	24	7
87	Digital logic station interface	12	72	10	2
88	Digital logic station	42	170	26	98
89	Last block	12	58	12	0
90	Extended executive	52	154	52	0
93	BASIC real output	12	72	20	1
94	BASIC boolean output	12	92	36	1
95	Module status monitor	22	80	30	6
96	Redundant analog input	22	88	34	9
97	Redundant digital input	14	58	16	5
98	Slave select	26	86	14	5
100	Digital output readback check	40	134	26	21
101	Exclusive OR	12	46	10	2
102	Pulse input/period	26	126	20	118
103	Pulse input/frequency	26	126	20	124
104	Pulse input/totalization	28	156	32	148
109	Pulse input/duration	26	126	20	117
110	Five input rung	24	76	14	5
111	Ten input rung	38	106	14	8
112	Twenty input rung	68	166	14	13
114	BCD input	20	98	18	82
115	BCD output	22	98	12	109
116	Jump/master control relay	12	58	10	2

Table B-1 BRC-300/400/410 and HPG800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
117	Boolean recipe table	28	74	10	5
118	Real recipe table	48	106	12	5
119	Boolean signal multiplexer	32	92	10	4
120	Real signal multiplexer	32	94	12	4
121	Analog input/Cnet	22	90	22	5
122	Digital input/Cnet	20	66	14	3
123	Device driver	30	116	32	9
124	Sequence monitor	80	158	20	3
125	Device monitor	40	106	12	16
126	Real signal demultiplexer	12	84	24	9
128	Slave default definition	44	98	12	1
129	Multistate device driver	60	160	38	11
132	Analog input/slave	54	366	48	252
134	Multi-sequence monitor	96	248	36	3
135	Sequence manager	82	192	32	14
136	Remote motor control	52	166	50	15
137	BASIC real output/quality	12	62	32	1
138	BASIC boolean output/quality	12	54	24	1
139	Passive station interface	44	130	32	8
140	Restore Largest NVM utilization (FC 165 with S2 = 249) Smallest NVM utilization (FC 33)	Refer to formula in Function Code 140.	92 92	12 12	4,500 108
141	Sequence master	80	134	24	9
142	Sequence slave	74	110	10	2
143	Invoke C	28	102	22	Program dependent
144	C allocation	Equation 4	Equation 5	Equation 6	1
145	Frequency counter/slave	32	148	22	45
146	Remote I/O interface	68	Equation 30	56	1,469
147	Remote I/O definition	144	588	108	20
148 [5]	Batch sequence	36	Equation 7	Equation 8	Program dependent

Table B-1 BRC-300/400/410 and HPG800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
149	Analog output/slave First FC 149 in segment Second FC 149 in segment (same ASO as first) Additional FC 149s in segment	 84 84 84	 344 344 344	 52 52 52	 16,015 16,015 702
150	Hydraulic servo slave	32	148	42	120
151	Text selector	46	156	26	5
152	Model parameter estimator	28	342	26	2
153	Inferential smith controller parameter converter	42	182	30	10
154	Adaptive parameter scheduler	36	212	28	5
155	Regression	68	Equation 9	Equation 10	4
156	Advanced PID controller	58	158	48	64
157	General digital controller	68	Equation 11	Equation 12	6
160	Smith predictor	36	236	122	20
161	Sequence generator	74	182	46	7
162	Digital segment buffer	16	70	24	6
163	Analog segment buffer	16	78	32	6
165	Moving average	16	Equation 13	Equation 14	13
166	Integrator	28	84	24	9
167	Polynomial Most complicated All zero	 68 68	 112 112	 12 12	 74 28
168	Interpolator	98	198	14	22
169	Matrix addition	44	144	44	12
170	Matrix multiplication	44	144	44	84
171	Trigonometric Sine Secant	 14 14	 52 52	 12 12	 20 22
172	Exponential	12	48	12	18
173	Power	14	52	12	31
174	Logarithm	14	54	14	20
177	Data acquisition analog	104	334	94	24
178	Data acquisition analog input/loop	20	124	52	8
179	Enhanced trend Normal sampling Normal and statistical sampling	 52 52	 FDB + FOB 180 + 2 x Equation 15	 Equation 15 Equation 15	 13 25

Table B-1 BRC-300/400/410 and HPG800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
190 [5]	User defined function declaration	30	Equation 16	12	1
191 [5]	User defined function one	56	440 + S16	96 + S16	Program dependent
192 [5]	User defined function two	70	508 + S24	112 + S24	Program dependent
193 [5]	User defined data import	48	Equation 19	72 + S5	9
194 [5]	User defined data export	48	Equation 20	62 + S4	15
198 [5]	Aux. real user defined function	38	142	24	1
199 [5]	Aux. digital user defined function	38	134	16	1
210	Sequence of events slave	100	586	182	246
211	Data acquisition digital	62	260	211	34
212	Data acquisition digital input/loop	20	106	212	5
215	Enhanced analog slave definition	32	232	26	558
216	Enhanced analog input definition	42	114	216	218
217	Enhanced calibration command	24	154	217	9
218 [5]	Phase execution	43	Equation 22	218	Program dependent
219 [5]	Common sequence	46	Equation 17	Equation 18	Program dependent
220 [5]	Batch historian	46	476 + S11	226 + S11	Program dependent
221	I/O device definition	Equation 23	966	50	Note 1
222	Analog in/channel	Equation 24	298	64	179 [2]
223	Analog out/channel	Equation 25	298	64	215 [2]
224	Digital in/channel	Equation 26	298	64	96 [2]
225	Digital out/channel	Equation 27	298	64	96 [2]
226	Test status	Equation 21	144	16	42
227	Gateway	Equation 31	288	68	41,194 [3]
228	Foreign device definition	Equation 28	356	74	42
229	Pulse In/Channel	Equation 29	472	80	257

Table B-1 BRC-300/400/410 and HPG800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
241	DSOE interface SEM to MFP No SED resynch performed SED resynch performed (1/sec.)	 18 18	 315 315	 30 30	257
242	DSOE digital event interface No data on SED I/O module Data on SED I/O module	 84 84	 246 246	 96 96	 77 343
247	Condition monitoring	134	592	82	Note 4

NOTES:

1. Time dependent on the I/O block type configured and on the presence or absence of redundant blocks. Refer to Table B-2.
2. With exception reports enabled.
3. Time for the maximum configuration.
4. Time dependent on the turbine instrumentation module type. Refer to the applicable product instruction manual.
5. Available in J0 and later releases.

Table B-2 FC221 Execution Times

I/O Block	Non-Redundant Blocks (µsecs)	Redundant Blocks (µsecs)
AIN-120	1,134	2,149
AOT-120	2,030	4,060
CIO-100	1,731	N/A
DIO-400	1,791	3,582
DOT-120	1,791	3,582

B.2 Memory Usage Equations

1. 485 + S2 + [16 x (S8)]
2. 400,174 + [16 x (S8)]
3. 72 + [8 x (S5)]
4. 12 + [1024 x (S2)]
5. 42 + [1024 x (S1)]
6. 10 + [1024 x (S1)]
7. 676 + [1024 x (S11)] + [a x (S12)]

   where:

   a = 2 (if S12 is positive)
   - or -
   a = 2048 (if S12 is negative)
8. 248 + [b x (S12)]

   where:

   b = 1 (if S12 is positive)
   - or -
   b = 1024 (if S12 is negative)
9. 294 + [20 x (S7)]
10. 60 + [20 x (S7)]
11. 144 + [8 x (S21 + S22)]

12. 60 + [4 x (S21 + S22)]
13. 79 + [4 x (S2)]
14. 34 + [4 x (S2)]
15. a + [b x (S4)] + [c x (n{S3})],
    where:

    n{S3} = Number of modes selected for S3
    if S2 = 0, 10, or 11, then a = 106, b = 12, and c = 8
    if S2 = 1, 2, 5, 6, 7, 9, 12, or 13, then a = 100, b = 10, and c = 4
    if S2 = 4 or 8, then a = 128, b = 25, and c = 32
16. 82 + [10 x (S2)]
17. 490 + [292 x (S13)]
18. 378 + [142 x (S13)]
19. 214 + [2 x (S5)]
20. 202 + [4 x (S4)]
21. 27 + Size of string data in S2 + S3 + S4 + S5 + S10.
22. 1458 + [1024 x (S11)] + [2 x (S12)]
23. 90 + Size of string data in S1 + Size of string data in S30.
24. 92 + Size of string data in S1 + Size of string data in S23.
25. 78 + Size of string data in S1+ Size of string data in S18.
26. 52 + Size of string data in S1+ Size of string data in S15.
27. 48 + Size of string data in S1+ Size of string data in S12.
28. 48 + Size of string data in S1 + Size of string data in S2
    + Size of string data in S7.
29. 108 + Size of string data in S1 + Size of string data in S23
30. 1008 + (16 x (S4)).
31. 48 + Size of string data in S1 + Size of string data in S2+ Size of string data in S12.

B.3 Function Blocks - BRC-300/400/410 and HPG800

Table B-3 contains function code block number information for the BRC-300, BRC-400, BRC-410 and HPG800 modules.

Table B-3 BRC-300/400/410 and HPG800 Modules

Block No.	Definition	Function Code
0	Logic 0	81
1	Logic 1	
2	0 or 0.0	
3	-100.0	
4	-1.0	
5	0.0	
6	1.0	
7	100.0	
8	-9.2 E18	
9	9.2 E18	
10	Startup flag (0 = no, 1 = yes)	
11	Memory display value	
12	System free time in percent	
13	Revision level	
14	Reserved	
15	Task 1 elapsed time since previous cycle	82
16	Task 1 elapsed time current cycle (sec/min)	
17	Task 1 processor utilization	
18	Task 1 check point overrun count	
19	Task 1 cycle time overrun (sec/min)	
20	Hours, time of day	90
21	Minutes, time of day	
22	Seconds, time of day	
23	No time synchronization flag: 0 = time of day invalid 1 = time of day valid	
24	Year (0 to 99)	
25	Month (1 to 12)	
26	Day (1 to 31)	
27	Day of week (1 to 7, Sunday = 1)	90
28	Reserved	
29	Reserved	

Table B-3 BRC-300/400/410 and HPG800 Modules (Continued)

Block No.	Definition	Function Code
30[1]	Configurable blocks	Any allowed function code - refer to Table B-1
31999	Loop type: 0.0 = Plant Loop 1.0 = Cnet 3.0 = Cnet with time-stamping	89

NOTE:
1. The highest configurable block number is 9998 for the BRC-300.

B.4 Module Status Information - BRC-300/400/410 and HPG800

Tables B-4 and B-5 explain Harmony bridge controller status bytes.

Table B-4 Bit Description - BRC-300/400/410 and HPG800

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	FTX	BAC	RIO	LIO	CFG	NVF	NVI	DSS
3	Error code							
4	Error code descriptor (1)							
5	Error code descriptor (2)							
6	ETYPE							
7	CWA	CWB	R1F	R2F	Reserved	Reserved	HnetA	HnetB
8	Reserved							
9	RA	RB	Reserved					
10	PRI	CFC	Reserved	CHK	RID	RDEXP	OCE	RDDET
11	Reserved	Reserved	Reserved	SOA	RNO	Reserved	Reserved	Reserved
12-13	Reserved							
14	Module nomenclature							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

Table B-5 Byte Description - BRC-300/400/410 and HPG800

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 00 = configure, 10 = error, 11 = execute
	TYPE	1F	Module type code: $(15)_{16}$ = Enhanced status

Table B-5 Byte Description - BRC-300/400/410 and HPG800 (Continued)

Byte	Field	Field Size or Value	Description
2	FTX	80	First time in execute: 0 = no, 1 = yes
	BAC	40	Backup status: 0 = good, 1 = bad
	RIO	20	Summary remote input status: 0 = good, 1 = bad
	LIO	10	Summary local input status: 0 = good, 1 = bad
	CFG	08	Online configuration changes being made
	NVF	04	Summary NVRAM failure status: 0 = good, 1 = fail
	NVI	02	Summary NVRAM initialized state: 0 = no, 1 = yes
	DSS	01	Digital station status: 0 = good, 1 = bad
3 - 5 Note 1	Error Code	3 4 5	
		01 01 — 02 — 03 — FF —	NVRAM error: Write failure Checksum failure Bad data Reset during write
		02 (1) (2)	Analog input reference error: (1), (2) = block number of control interface I/O module block
		03 (1) (2)	Missing I/O module or expander board: (1), (2) = block number of I/O module or station
		05 (1) (2)	Configuration error – undefined block: (1), (2) = block number making reference
		06 (1) (2)	Configuration error – input data type is incorrect: (1), (2) = block number making reference
		08 (1) (2)	Trip block activated: (1), (2) = block number of trip block
		09 — —	Segment violation.
		0F — —	Primary module has failed and the redundant module configuration is not current
		10 — —	Primary module has failed and the dynamic RAM data in the redundant module is not current
		09 — —	Segment violation - priority set the same in two segments or more than eight segments defined.
		11 — —	NVRAM write failure error
		1E (1) (2)	Duplicate device definition label – multiple function code 221 function blocks contain the same device label. (1), (2) = block number making reference.

Table B-5 Byte Description - BRC-300/400/410 and HPG800 (Continued)

Byte	Field	Field Size or Value	Description
3 - 5 *(cont)*	Error Code	2A (1) (2)	Not enough memory for UDF: (1), (2) = block number making reference
		20 — —	Program format error - inconsistent format table
		21 00 00 FF FE FF FF (1) (2)	File system error: Backup cannot take over due to uninitialized file system. Directory has not been configured. List of file system free memory is corrupted. (1), (2) = Number of files with errors.
		22 (1) (2)	Invoke C error: (1), (2) = block number making reference
		24 (1) (2)	C program stack overflow: (1), (2) = block number making reference
		28 (1) (2)	User defined function (UDF) reference is invalid: (1), (2) = block number making reference
		29 (1) (2)	UDF block cannot read program file: (1), (2) = block number making reference
		2B (1) (2)	Missing UDF declaration: (1), (2) = block number making reference
		2C (1) (2)	Wrong UDF type: (1), (2) = block number making reference
		2D (1) (2)	Missing UDF auxiliary block: (1), (2) = block number making reference
		2E (1) (2)	UDF compiler and firmware are incompatible: (1), (2) = block number making reference
6	ETYPE	1F	Enhanced module type = $(24)_{16}$
7	CWA	80	Controlway bus A failure: 0 = good, 1 = fail
	CWB	40	Controlway bus B failure: 0 = good, 1 = fail
	R1F	20	Redundancy link channel 1 failure: 0 = good, 1 = fail
	R2F	10	Redundancy link channel 2 failure: 0 = good, 1 = fail
	HnetA	02	Harmony net channel A failure: 0 = good, 1 = fail
	HnetB	01	Harmony net channel B failure: 0 = good, 1 = fail
8	—	—	Unused.
9	RA	80	Harmony net channel A relay fault: 0 = good, 1 = fail
	RB	40	Harmony net channel B relay fault: 0 = good, 1 = fail

Table B-5 Byte Description - BRC-300/400/410 and HPG800 (Continued)

Byte	Field	Field Size or Value	Description
10	PRI	80	Module is primary versus backup; set to 1 in the primary module.
	CFC	40	Configuration current (latched until backup is reset). Set when LED 7 is enabled (1 = on or blinking) on the backup module.
	CHK	10	Backup has completed checkpointing (latched until backup is reset). Always set to 0 on the primary module. Follows LED 8 (1 = on or blinking) on the backup module.
	RID	08	Redundancy ID. Follows setting of redundancy ID pole on the dip-switch.
	RDEXP	04	Redundancy expected. Always set to 1 on the backup module. Follows state of function code 90, specification S3, ones digit on the primary module.
	OCE	02	Online configuration is enabled. Follows setting of online configuration enable pole on dipswitch.
	RDDET	01	Redundancy detected (latched until module is reset or it changes from backup to primary or primary to backup). Set to 1 when a properly configured redundant module is detected.
11	SOA	10	Status output alarm. Indicates the status of the system +24 volt power and the I/O block's power (logic and field power for a single cabinet). 0 = OK, 1 = alarm.
	RNO	08	Redundancy NVM overrun (latched indication). Set to 1 in primary module if NVM checkpoint overruns have occurred. NVM checkpoint overruns cause the primary module to reset the backup module.
12-13	—	00	Reserved
14	—	FF	Module nomenclature:$(06)_{16}$ = BRC-300, $(07)_{16}$ = BRC-400
15	—	FF	Revision letter (in ASCII code), for example, $(47)_{16}$ = G, $(4A)_{16}$ = J
16	—	FF	Revision number (in ASCII code), for example, $(30)_{16}$ = 0

NOTE:

1. Byte 3 is displayed on the front panel LEDs when the module is in ERROR mode.

C. Harmony Bridge Controller (BRC-100/200)

C.1 Memory Utilization and Execution Times

This section lists the module memory requirements for each function code. Two quantities are given for the BRC-100 and BRC-200 memory utilization. The first value is the number of bytes of nonvolatile random access memory (NVRAM). The second quantity is the number of bytes of random access memory (RAM). The BRC-100 module has a total configuration memory of 441 kilobytes of NVRAM and 1.5 megabytes of RAM. The BRC-200 module has a total configuration memory of 1.8 megabytes of NVRAM and 7.86 megabytes of RAM.
This section also lists the function code execution times (in microseconds) for the BRC-100 and BRC-200 modules.

NOTE: Except where otherwise noted, execution times are given for worst case conditions.

Table C-1 shows the BRC-100 and BRC-200 memory requirements and the execution time for each function code.

NOTE: Refer to ***Memory Usage Equations*** in this section for the equations listed in Table C-1.

Table C-1 BRC-100/200 Memory Utilization and Execution Times

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Execution Times (in µsecs)
1	Function generator	46	92	17
2	Manual set constant	26	40	7
3	Lead/lag	26	72	60
4	Pulse positioner	26	88	27
5	Pulse rate	26	72	49
6	High/low limiter	26	48	16
7	Square root	26	48	70
8	Rate limiter	26	64	49
9	Analog transfer	26	76	68
10	High select	26	52	21
11	Low select	26	52	21
12	High/low compare	26	48	16
13	Integer transfer	26	44	9
14	Four input summer	26	52	26
15	Two input summer	26	52	36
16	Multiply	26	48	29
17	Divide	26	48	33
18	PID error input	34	108	213
19	PID process variable and set point	36	116	230
24	Adapt	26	50	12
25	Analog input (periodic sample)	26	102	12
26	Analog input/loop	26	54	14
30	Analog exception report	26	102	38
31	Test quality	26	52	18

Table C-1 BRC-100/200 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Execution Times (in µsecs)
32	Trip	26	36	8
33	Not	26	36	8
34	Memory	26	46	10
35	Timer	26	58	27
36	Qualified OR (eight input)	26	68	24
37	AND (two input)	26	40	8
38	AND (four input)	26	48	11
39	OR (two input)	26	40	8
40	OR (four input)	26	48	11
41	Digital input (periodic sample)	26	90	13
42	Digital input/loop	26	42	12
45	Digital exception report	26	66	26
50	Manual set switch	26	36	7
51	Manual set constant (nontunable)	26	40	7
52	Manual set integer	26	36	7
55	Hydraulic servo	60	404	2,363
58	Time delay (analog)	26	Equation 1	56
59	Digital transfer	26	44	9
61	Blink	26	48	40
62	Remote control memory	28	96	40
63	Analog input list (periodic sample)	28	286	40
64	Digital input list (periodic sample)	28	190	39
65	Digital sum with gain (four input)	28	68	33
66	Analog trend Normal mode (slow) Fast mode	 26 26	 196 700	 41 26
68	Remote manual set constant (REMSET)	26	128	33
69	Test alarm	26	44	13
79	Control interface slave First FC 79 block in segment Additional FC 79 blocks in segment	 48 48	 328 328	 15,710 1,054
80	Control station	84	366	118
81	Executive block	26	224	0
82	Segment control	64	284	0

Table C-1 BRC-100/200 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Execution Times (in µsecs)
83	Digital output group	32	90	134
84	Digital input group	26	114	133
85	Up/down counter	26	76	26
86	Elapsed timer	26	84	35
89	Last block	26	48	0
90	Extended executive	52	152	0
91	BASIC configuration	Equation 2	Equation 3	5
92	Invoke BASIC	26	46	Program dependent
93	BASIC real output	26	64	4
94	BASIC boolean output	26	72	4
95	Module status monitor	26	108	22
96	Redundant analog input	26	106	44
97	Redundant digital input	26	58	23
98	Slave select	26	84	26
99	Sequence of events log	26	Equation 4	97
100	Digital output readback check	40	144	77
101	Exclusive OR	26	40	8
102	Pulse input/period	26	100	162
103	Pulse input/frequency	26	100	170
104	Pulse input/totalization	26	130	203
109	Pulse input/duration	26	100	160
110	Five input rung	26	74	20
111	Ten input rung	38	104	30
112	Twenty input rung	68	164	49
114	BCD input	26	72	113
115	BCD output	26	72	150
116	Jump/master control relay	26	52	6
117	Boolean recipe table	28	68	24
118	Real recipe table	48	102	24
119	Boolean signal multiplexer	32	86	21
120	Real signal multiplexer	32	90	21
121	Analog input/Cnet	26	82	17

Table C-1 BRC-100/200 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Execution Times (in µsecs)
122	Digital input/Cnet	26	50	12
123	Device driver	30	118	33
124	Sequence monitor	80	162	15
125	Device monitor	40	102	77
126	Real signal demultiplexer	26	92	45
128	Slave default definition	44	72	4
129	Multistate device driver	60	168	40
132	Analog input/slave	54	340	346
133	Smart field device definition	36	144	14
134	Multi-sequence monitor	96	268	13
135	Sequence manager	82	208	66
136	Remote motor control	52	186	54
137	BASIC real output/quality	26	80	4
138	BASIC boolean output/quality	26	64	4
139	Passive station interface	44	146	37
140	Restore Restore largest NVM utilization (FC 165 with S2 = 249) Smallest NVM utilization (FC 33)	 Equation 19 Equation 19	 88 88	 5,141 123
141	Sequence master	80	142	33
142	Sequence slave	74	104	7
143	Invoke C	28	108	Program dependent
144	C allocation	Equation 5	Equation 6	5
145	Frequency counter/slave	30	120	62
146	Remote I/O interface	68	340	2,015
147	Remote I/O definition	144	480	27
148	Batch sequence	36	Equation 7	Program dependent
149	Analog output/slave First FC 149 in segment Second FC 149 in segment (same ASO as first) Additional FC 149s in segment	 82 82 82	 318 318 318	 16,414 16,414 1,298
150	Hydraulic servo slave	32	168	164
151	Text selector	46	152	20
152	Model parameter estimator	28	352	16

Table C-1 BRC-100/200 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Execution Times (in µsecs)
153	Inferential smith controller parameter converter	42	196	73
154	Adaptive parameter scheduler	36	224	38
155	Regression	68	Equation 8	28
156	Advanced PID controller	58	190	309
157	General digital controller	68	Equation 9	46
160	Smith predictor	36	342	146
161	Sequence generator	74	212	32
162	Digital segment buffer	26	78	22
163	Analog segment buffer	26	94	23
165	Moving average	26	Equation 10	96
166	Integrator	28	92	64
167	Polynomial Most complicated All zero	 58 58	 108 108	 216 81
168	Interpolator	98	196	167
169	Matrix addition	44	172	86
170	Matrix multiplication	44	172	624
171	Trigonometric Sine Secant	 26 26	 48 48	 149 161
172	Exponential	26	44	136
173	Power	26	48	228
174	Logarithm	26	50	152
177	Data acquisition analog	104	398	118
178	Data acquisition analog input/loop	26	134	30
179	Enhanced trend Normal sampling Normal and statistical sampling	 52 52	 Equation 11 Equation 11	 98 188
184	Factory instrumentation protocol handler	42	1,342	3,123
185	Digital input subscriber	32	220	88
186	Analog input subscriber	32	252	468
187	Analog output subscriber	48	232	835
188	Digital output subscriber	48	208	91
190	User defined function declaration	30	Equation 12	5
191	User defined function one	56	Equation 13	Program dependent

Table C-1 BRC-100/200 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Execution Times (in µsecs)
192	User defined function two	70	Equation 14	Program dependent
193	User defined data import	48	Equation 15	32
194	User defined data export	48	Equation 16	55
198	Auxiliary real user defined function	38	150	5
199	Auxiliary digital user defined function	38	134	5
210	Sequence of events slave	100	572	338
211	Data acquisition digital	62	310	123
212	Data acquisition digital input/loop	26	110	20
215	Enhanced analog slave definition	30	206	766
216	Enhanced analog input definition	42	114	299
217	Enhanced calibration command	26	178	12
218	Phase execution	44	Equation 25	Program dependent
219	Common sequence	46	Equation 17	Program dependent
220	Batch historian	46	Equation 18	Program dependent
221	I/O device definition	Equation 20	1124	Note 1
222	Analog in/channel	Equation 21	298	300[2]
223	Analog out/channel	Equation 22	298	360[2]
224	Digital in/channel	Equation 23	298	160[2]
225	Digital out/channel	Equation 24	298	160[2]
226	Test status	Equation 26	130	70
227	Gateway	Equation 27	19,982	69,000[3]
228	Foreign device definition	Equation 28	372	70
229	Pulse In/Channel	Equation 29	452	430
241	DSOE data interface SEM to MFP No SED resynch function performed SED resynch performed once every second	 26 26	 284 284	 105 470
242	DSOE digital event interface No data available on SED I/O module Data available on SED I/O module	 84 84	 338 338	 77 795

Table C-1 BRC-100/200 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Execution Times (in µsecs)
247	Condition monitoring	133	544	Note 4

NOTES:

1. Time dependent on the I/O block type configured and on the presence or absence of redundant blocks. Refer to Table C-2.
2. With exception reports enabled.
3. Time for the maximum configuration.
4. Time dependent on the turbine instrumentation module type. Refer to the applicable product instruction manual.

Table C-2 FC221 Execution Times

I/O Block	Non-Redundant Blocks (µsecs)	Redundant Blocks (µsecs)
AIN-120	1900	3600
AOT-120	3400	6800
CIO-100	2900	N/A
DIO-400	3000	6000
DOT-120	3000	6000

C.2 Memory Usage Equations

1. 78 + [8 x (S5)]
2. 38 + [1024 x (S5)]
3. 4994 + 1024 [(S3) + (S4) + (S5)]
4. 96 + [9 x (S2)]
5. 12 + [1024 x (S2)]
6. 36 + [1024 x (S1)]
7. 826 + [1024 x (S11)] + [a x (S12)]

   where:

   a = 2 (if S12 is positive)
   - or -
   a = 2048 (if S12 is negative)
8. 338 + [40 x (S7)]
9. 228 + [8 x (S21 + S22)]
10. 98 + [8 x (S2]
11. a + [b x (S4)] + [c x (n{S3})],

    where:

    n{S3} = Number of modes selected for S3

    if S2 = 0 or 3, then a = 396, b = 24, and c = 16

    if S2 = 1, 2, 5, 6, 7 or 9, then a = 380, b = 20, and c = 8

    if S2 = 4 or 8, then a = 456, b = 50, and c = 64
12. 78 + [10 x (S2)]
13. 520 + [2 x (S16)]
14. 604 + [2 x (S24)]
15. 256 + [3 x (S5)]

16. 256 + [5 x (S4)]
17. 424 + [222 x (S13)]
18. 666 + [2 x (S11)]
19. 40 + N,

    where:

    N = Applicable function code size from Table 140-2. NVRAM is set to 46 when N is less than six.
20. 90 + Size of string data in S1 + Size of string data in S30.
21. 92 + Size of string data in S1.
22. 78 + Size of string data in S1.
23. 52 + Size of string data in S1.
24. 48 + Size of string data in S1.
25. 1450 + [1024 x (S11)] + [2 x (S12)]
26. 27 + Size of string data in S2 + S3 + S4 + S5 + S10.
27. 208 + Size of string data in S1 + Size of string data in S2
    + Size of string data in S4 + Size of string data in S6
    + Size of string data in S8 + Size of string data in S10
    + Size of string data in S19 + S15
    + [18 if 0 < (S15) <= 16,374,
    else 36 if 16,374 < (S15) <= 32,748,
    else 54 if 32,748 < (S15)]
28. 48 + Size of string data in S1 + Size of string data in S2
    + Size of string data in S7
29. 108 + Size of string data in S1 + Size of string data in S23

C.3 Function Blocks - BRC-100/200

Table C-3 contains function code block number information for the BRC-100 and BRC-200 modules.

Table C-3 BRC-100/200 Module

Block No.	Definition	Function Code
0	Logic 0	81
1	Logic 1	
2	0 or 0.0	
3	-100.0	
4	-1.0	
5	0.0	
6	1.0	
7	100.0	
8	-9.2 E18	
9	9.2 E18	
10	Startup flag (0 = no, 1 = yes)	
11	Memory display value	81 *(continued)*
12	System free time in percent	
13	Revision level	
14	Reserved	

Table C-3 BRC-100/200 Module (Continued)

Block No.	Definition	Function Code
15	Task 1 elapsed time since previous cycle	82
16	Task 1 elapsed time current cycle (sec/min)	
17	Task 1 processor utilization	
18	Task 1 check point overrun count	
19	Task 1 cycle time overrun (sec/min)	
20	Hours, time of day	90
21	Minutes, time of day	
22	Seconds, time of day	
23	No time synchronization flag: 0 = time of day invalid 1 = time of day valid	
24	Year (0 to 99)	
25	Month (1 to 12)	
26	Day (1 to 31)	
27	Day of week (1 to 7, Sunday = 1)	90
28	Reserved	
29	Reserved	
30[1]	Configurable blocks	Any allowed function code - refer to Table C-1
9999	Loop type: 0.0 = Plant Loop 1.0 = Cnet 3.0 = Cnet with time-stamping	89

NOTE:

1. The highest configurable block number is 9998 for the BRC-100.

C.4 Module Status Information - BRC-100/200

Tables C-4 and C-5 explain Harmony bridge controller status bytes.

Table C-4 Bit Description - BRC-100/200

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	FTX	BAC	RIO	LIO	CFG	NVF	NVI	DSS
3	Error code							
4	Error code descriptor (1)							
5	Error code descriptor (2)							
6	ETYPE							

Table C-4 Bit Description - BRC-100/200 (Continued)

Byte	Bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
7	CWA	CWB	R1F	R2F	Reserved	Reserved	HnetA	HnetB
8	SIME	SIMR	SIMT	SIMM	Reserved			
9	RA	RB	Reserved					
10	PRI	CFC	Reserved	CHK	RID	RDEXP	OCE	RDDET
11	Reserved	Reserved	Reserved	SOA	RNO	Reserved	Reserved	Reserved
12-13	Reserved							
14	Module nomenclature							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

Table C-5 Byte Description - BRC-100/200

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 00 = configure, 10 = error, 11 = execute
	TYPE	1F	Module type code: $(15)_{16}$ = Enhanced status
2	FTX	80	First time in execute: 0 = no, 1 = yes
	BAC	40	Backup status: 0 = good, 1 = bad
	RIO	20	Summary remote input status: 0 = good, 1 = bad
	LIO	10	Summary local input status: 0 = good, 1 = bad
	CFG	08	Online configuration changes being made
	NVF	04	Summary NVRAM failure status: 0 = good, 1 = fail
	NVI	02	Summary NVRAM initialized state: 0 = no, 1 = yes
	DSS	01	Digital station status: 0 = good, 1 = bad

Table C-5 Byte Description - BRC-100/200 (Continued)

Byte	Field	Field Size or Value	Description
3 - 5 Note 1	Error Code	3 4 5	
		01 01 — 02 — 03 — FF —	NVRAM error: Write failure Checksum failure Bad data Reset during write
		02 (1) (2)	Analog input reference error: (1), (2) = block number of control interface I/O module block
		03 (1) (2)	Missing I/O module or expander board: (1), (2) = block number of I/O module or station
		05 (1) (2)	Configuration error – undefined block: (1), (2) = block number making reference
		06 (1) (2)	Configuration error – input data type is incorrect: (1), (2) = block number making reference
		08 (1) (2)	Trip block activated: (1), (2) = block number of trip block
		0F — —	Primary module has failed and the redundant module configuration is not current
		10 — —	Primary module has failed and the dynamic RAM data in the redundant module is not current
		09 — —	Segment violation - priority set the same in two segments or more than eight segments defined.
		11 — —	NVRAM write failure error
		1E (1) (2)	Duplicate device definition label – multiple function code 221 function blocks contain the same device label. (1), (2) = block number making reference.
		20 — —	Program format error - inconsistent format table
		21 00 00 FF FE FF FF (1) (2)	File system error: Backup cannot take over due to uninitialized file system. Directory has not been configured. List of file system free memory is corrupted. (1), (2) = Number of files with errors.
		22 (1) (2)	Invoke C error: (1), (2) = block number making reference
		24 (1) (2)	C program stack overflow: (1), (2) = block number making reference
		28 (1) (2)	User defined function (UDF) reference is invalid: (1), (2) = block number making reference
		29 (1) (2)	UDF block cannot read program file: (1), (2) = block number making reference
		2A (1) (2)	Not enough memory for UDF: (1), (2) = block number making reference

Table C-5 Byte Description - BRC-100/200 (Continued)

Byte	Field	Field Size or Value	Description
3 - 5 *(cont)*	Error Code	2B (1) (2)	Missing UDF declaration: (1), (2) = block number making reference
		2C (1) (2)	Wrong UDF type: (1), (2) = block number making reference
		2D (1) (2)	Missing UDF auxiliary block: (1), (2) = block number making reference
		2E (1) (2)	UDF compiler and firmware are incompatible: (1), (2) = block number making reference
		2F (1) (2)	BASIC program error: (1), (2) = line number of error
6	ETYPE	1F	Enhanced module type = $(24)_{16}$
7	CWA	80	Controlway bus A failure: 0 = good, 1 = fail
	CWB	40	Controlway bus B failure: 0 = good, 1 = fail
	R1F	20	Redundancy link channel 1 failure: 0 = good, 1 = fail
	R2F	10	Redundancy link channel 2 failure: 0 = good, 1 = fail
	HnetA	02	Harmony net channel A failure: 0 = good, 1 = fail
	HnetB	01	Harmony net channel B failure: 0 = good, 1 = fail
8	SIME	80	Simulation enabled: 0 = normal operation, 1 = simulation active
	SIMR	40	Simulation running/frozen: 0 = simulation frozen, 1= simulation running
	SIMT	20	Simulation time rate: 0 = real time, 1 = slow/fast time
	SIMM	10	Simulation mode: 0=INFI 90®/USM0x, 1=Harmony/SIM-100
9	RA	80	Harmony net channel A relay fault: 0 = good, 1 = fail
	RB	40	Harmony net channel B relay fault: 0 = good, 1 = fail
10	PRI	80	Module is primary versus backup; set to 1 in the primary module.
	CFC	40	Configuration current (latched until backup is reset). Set when LED 7 is enabled (1 = on or blinking) on the backup module.
	CHK	10	Backup has completed checkpointing (latched until backup is reset). Always set to 0 on the primary module. Follows LED 8 (1 = on or blinking) on the backup module.
	RID	08	Redundancy ID. Follows setting of redundancy ID pole on the dip-switch.
	RDEXP	04	Redundancy expected. Always set to 1 on the backup module. Follows state of function code 90, specification S3, ones digit on the primary module.
	OCE	02	Online configuration is enabled. Follows setting of online configuration enable pole on dipswitch.
	RDDET	01	Redundancy detected (latched until module is reset or it changes from backup to primary or primary to backup). Set to 1 when a properly configured redundant module is detected.

Table C-5 Byte Description - BRC-100/200 (Continued)

Byte	Field	Field Size or Value	Description
11	SOA	10	Status output alarm. Indicates the status of the system +24 volt power and the I/O block's power (logic and field power for a single cabinet). 0 = OK, 1 = alarm.
	RNO	08	Redundancy NVM overrun (latched indication). Set to 1 in primary module if NVM checkpoint overruns have occurred. NVM checkpoint overruns cause the primary module to reset the backup module.
12-13	—	00	Reserved
14	—	FF	Module nomenclature:$(05)_{16}$ = BRC-100, $(09)_{16}$ = BRC-200
15	—	FF	Revision letter (in ASCII code), for example, $(46)_{16}$ = F
16	—	FF	Revision number (in ASCII code), for example, $(37)_{16}$ = 7

NOTE:

1. Byte 3 is displayed on the front panel LEDs when the module is in ERROR mode.

D. Harmony Area Controller (HAC)

D.1 Memory Utilization and Execution Times

This section lists the module memory requirements for each function code. Three quantities are given for the HAC memory utilization:

- The number of bytes of nonvolatile random access memory (NVRAM). The Harmony area controller has a total configuration memory of 2,097,150 bytes of NVRAM.
- The number of bytes of random access memory (RAM). The Harmony area controller has a total configuration memory of 6,291,392 bytes of RAM.
- The checkpoint buffer RAM contained on the process I/O (PIO) board inside the HAC enclosure. This area is physically separate from the main memory board, and it can have an impact on the size of a configuration. The checkpoint buffer RAM has a total configuration memory of 2,070,000 bytes.

This section also lists the function code execution times (in microseconds) for the Harmony area controller.

NOTE: Except where otherwise noted, execution times are given for worst case conditions.

Table D-1 shows the HAC memory requirements and the execution time for each function code.

Table D-1 HAC Memory Utilization and Execution Times

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Buffer RAM (Bytes)	Execution Times (in µsecs)
1	Function generator	46	96	12	9
2	Manual set constant	12	44	12	4
3	Lead/lag	18	66	22	30
4	Pulse positioner	24	84	24	14
5	Pulse rate	16	70	18	25
6	High/low limiter	16	52	12	8
7	Square root	14	52	12	35
8	Rate limiter	18	62	18	25
9	Analog transfer	20	70	22	34
10	High select	16	56	12	11
11	Low select	16	56	12	11
12	High/low compare	16	52	12	8
13	Integer transfer	14	50	10	5
14	Four input summer	16	56	12	13
15	Two input summer	18	56	12	18
16	Multiply	14	52	12	15
17	Divide	14	52	12	17
18	PID error input	34	98	26	107
19	PID process variable and set point	36	102	30	115
24	Adapt	12	54	12	6
25	Analog input (peer-to-peer)	12	79	34	20
26	Analog input/loop	12	66	20	7

Table D-1 HAC Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Buffer RAM (Bytes)	Execution Times (in µsecs)
30	Analog exception report	26	120	40	19
31	Test quality	16	58	10	9
32	Trip	12	42	10	4
33	Not	12	42	10	4
34	Memory	14	52	10	5
35	Timer	14	56	18	14
36	Qualified OR (eight input)	26	74	10	12
37	AND (two input)	12	46	10	4
38	AND (four input)	16	54	10	6
39	OR (two input)	12	46	10	4
40	OR (four input)	16	54	10	6
41	Digital input/bus (peer-to-peer)	12	78	33	20
42	Digital input/loop	12	60	14	6
45	Digital exception report	12	88	32	13
50	Manual set switch	8	40	10	4
51	Manual set constant (nontunable)	12	36	8	4
52	Manual set integer	12	36	8	4
55	Hydraulic servo	60	332	88	1,182
57	Terminating executive	42	Equation 1	Equation 2	0
58	Time delay (analog)	18	Equation 3	22	28
59	Digital transfer	14	50	10	5
61	Blink	12	50	14	20
62	Remote control memory	28	104	22	20
63	Analog input list (peer-to-peer)	28	185	110	20
64	Digital input list (peer-to-peer)	28	137	62	20
65	Digital sum with gain (four input)	28	72	12	17
66	Analog trend Normal mode (slow) Fast mode	 12 12	 376 376	 340 340	 21 13
68	Remote manual set constant (REMSET)	22	122	36	17
69	Test alarm	12	48	12	7
79	Control interface slave First FC 79 block in segment Other FC 79 blocks in segment	 48 48	 276 276	 68 68	 7,855 527

Table D-1 HAC Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Buffer RAM (Bytes)	Execution Times (in µsecs)
80	Control station	84	316	80	59
81	Executive block	12	179	66	0
82	Segment control	66	260	56	0
83	Digital output group	32	96	10	67
84	Digital input group	16	88	42	67
85	Up/down counter	24	74	18	13
86	Elapsed timer	20	76	24	18
90	Extended executive	52	107	58	0
95	Module status monitor	22	80	30	11
96	Redundant analog input	22	88	34	22
97	Redundant digital input	14	58	16	12
98	Slave select	26	86	14	13
100	Digital output readback check	40	134	26	39
101	Exclusive OR	12	46	10	4
102	Pulse input/period	24	96	20	81
103	Pulse input/frequency	24	96	20	85
104	Pulse input/totalization	26	114	32	102
109	Pulse input/duration	24	96	20	80
110	Five input rung	24	76	14	10
111	Ten input rung	38	106	14	15
112	Twenty input rung	68	166	14	25
114	BCD input	20	70	18	57
115	BCD output	22	76	12	75
116	Jump/master control relay	12	58	10	3
117	Boolean recipe table	28	74	10	12
118	Real recipe table	48	106	12	12
119	Boolean signal multiplexer	32	92	10	11
120	Real signal multiplexer	32	94	12	11
121	Analog input/Cnet	22	90	22	9
122	Digital input/Cnet	20	66	14	6
123	Device driver	30	116	32	17
124	Sequence monitor	80	158	20	8
125	Device monitor	40	106	12	39

Table D-1 HAC Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Buffer RAM (Bytes)	Execution Times (in µsecs)
126	Real signal demultiplexer	12	84	24	23
128	Slave default definition	44	90	22	2
129	Multistate device driver	60	160	38	20
132	Analog input/slave	54	308	48	173
133	Smart field device definition	36	164	20	7
134	Multi-sequence monitor	96	248	36	7
135	Sequence manager	82	192	32	33
136	Remote motor control	52	166	50	27
137	BASIC real output/quality	12	62	32	2
138	BASIC boolean output/quality	12	54	24	2
139	Passive station interface	44	130	32	19
140	Restore Largest NVM utilization (FC 165 with S2 = 249) Smallest NVM utilization (FC 33)	Refer to formula in Function Code 140.	 92 92	 12 12	 2,571 62
141	Sequence master	80	134	24	17
142	Sequence slave	74	110	10	4
143	Invoke C	28	102	22	Program dependent
144	C allocation	Equation 4	Equation 5	Equation 6	3
145	Frequency counter/slave	30	114	22	31
146	Remote I/O interface	68	290	54	1,008
147	Remote I/O definition	144	426	74	14
148	Batch sequence	36	Equation 7	Equation 8	Program dependent
149	Analog output/slave First FC 149 in segment Second FC 149 in segment (same ASO as first) Additional FC 149s in segment	 82 82 82	 282 282 282	 52 52 52	 8,207 8,207 649
150	Hydraulic servo slave	32	148	42	82
151	Text selector	46	156	26	10
152	Model parameter estimator	28	342	26	8
153	Inferential smith controller parameter converter	42	182	30	37
154	Adaptive parameter scheduler	36	212	28	19
155	Regression	68	Equation 9	Equation 10	14

Table D-1 HAC Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Buffer RAM (Bytes)	Execution Times (in µsecs)
156	Advanced PID controller	58	158	48	155
157	General digital controller	68	Equation 11	Equation 12	23
160	Smith predictor	36	236	122	73
161	Sequence generator	74	182	46	16
162	Digital segment buffer	16	70	24	11
163	Analog segment buffer	16	78	32	12
165	Moving average	16	Equation 13	Equation 14	48
166	Integrator	28	84	24	32
167	Polynomial Most complicated All zero	 68 68	 112 112	 12 12	 108 41
168	Interpolator	98	198	14	84
169	Matrix addition	44	144	44	43
170	Matrix multiplication	44	144	44	312
171	Trigonometric Sine Secant	 14 14	 52 52	 12 12	 75 81
172	Exponential	12	48	12	68
173	Power	14	52	12	114
174	Logarithm	14	54	14	76
177	Data acquisition analog	104	334	94	59
178	Data acquisition analog input/loop	20	124	52	15
179	Enhanced trend Normal sampling Normal and statistical sampling	 52 52	FDB + FOB memory usage	 Equation 15 Equation 15	 49 94
184	Factory instrumentation protocol handler	42	1274	84	1,562
185	Digital input subscriber	32	178	60	44
186	Analog input subscriber	32	194	76	234
187	Analog output subscriber	48	214	32	418
188	Digital output subscriber	48	190	32	46
190	User defined function declaration	30	Equation 16	12	3
191	User defined function one	56	440 + S16	96 + S16	Program dependent
192	User defined function two	70	508 + S24	112 + S24	Program dependent

Table D-1 HAC Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Buffer RAM (Bytes)	Execution Times (in µsecs)
193	User defined data import	48	Equation 19	72 + S5	16
194	User defined data export	48	Equation 20	62 + S4	28
198	Aux. real user defined function	38	142	24	3
199	Aux. digital user defined function	38	134	16	3
211	Data acquisition digital	62	260	84	62
212	Data acquisition digital input/loop	20	106	34	10
215	Enhanced analog slave definition	30	196	26	383
216	Enhanced analog input definition	42	114	16	150
217	Enhanced calibration command	24	154	40	6
218	Phase execution	43	Equation 22	308 + S12	Program dependent
219	Common sequence	46	Equation 17	Equation 18	Program dependent
220	Batch historian	46	476 + S11	226 + S11	Program dependent
221	I/O device definition	Equation 23	966	50	Note 1
222	Analog in/channel	Equation 24	298	64	70[2]
223	Analog out/channel	Equation 25	298	64	85[2]
224	Digital in/channel	Equation 26	298	64	50[2]
225	Digital out/channel	Equation 27	298	64	50[2]
226	Test status	Equation 21	144	16	52
241	DSOE interface SEM to MFP No SED resynch performed SED resynch performed (1/sec.)	 18 18	 315 315	 30 30	 53 235
242	DSOE digital event interface No data on SED I/O module Data on SED I/O module	 84 84	 246 246	 96 96	 39 398

NOTES:

1. Time dependent on the I/O block type configured and on the presence or absence of redundant blocks. Refer to Table D-2.
2. With exception reports enabled.

Table D-2 FC221 Execution Times

I/O Block	Non-Redundant Blocks (µsecs)	Redundant Blocks (µsecs)
AIN-120	1500	2800
AOT-120	2500	5000
CIO-100	2500	N/A
DIO-400	2400	4300
DOT-120	2300	4200

D.2 Memory Usage Equations

1. 485 + S2 + [16 x (S8)]
2. 400,174 + [16 x (S8)]
3. 72 + [8 x (S5)]
4. 12 + [1024 x (S2)]
5. 42 + [1024 x (S1)]
6. 10 + [1024 x (S1)]
7. 676 + [1024 x (S11)] + [a x (S12)]

   where:

   a = 1 (if S12 is positive)
   - or -
   a = 1024 (if S12 is negative)
8. 248 + [b x (S12)]

   where:

   b = 1 (if S12 is positive)
   - or -
   b = 1024 (if S12 is negative)
9. 294 + [20 x (S7)]
10. 60 + [20 x (S7)]
11. 144 + [8 x (S21 + S22)]
12. 60 + [4 x (S21 + S22)]
13. 79 + [4 x (S2)]
14. 34 + [4 x (S2)]
15. a + [b x (S4)] + [c x (n{S3})],

    where:

    n{S3} = Number of modes selected for S3

    if S2 = 0, 10, or 11, then a = 106, b = 12, and c = 8

    if S2 = 1, 2, 5, 6, 7, 9, 12, or 13, then a = 100, b = 10, and c = 4

    if S2 = 4 or 8, then a = 128, b = 25, and c = 32
16. 82 + [10 x (S2)]
17. 490 + [292 x (S13)]
18. 378 + [142 x (S13)]
19. 214 + [2 x (S5)]
20. 202 + [4 x (S4)]
21. 27 + Size of string data in S2 + S3 + S4 + S5 + S10.
22. 1458 + [1024 x (S11)] + [2 x (S12)]
23. 90 + Size of string data in S1 + Size of string data in S30.
24. 92 + Size of string data in S1.

25. 78 + Size of string data in S1.
26. 52 + Size of string data in S1.
27. 48 + Size of string data in S1.

D.3 Function Blocks - HAC

Table D-3 contains function code block number information for the Harmony area controller.

Table D-3 Harmony Area Controller

Block No.	Definition	Function Code
0	Logic 0	81
1	Logic 1	
2	0 or 0.0	
3	-100.0	
4	-1.0	
5	0.0	
6	1.0	
7	100.0	81
8	-9.2 E18	
9	9.2 E18	
10	Startup flag (0 = no, 1 = yes)	
11	Memory display value	
12	System free time in percent	
13	Revision level	
14	Reserved	
15	Task 1 elapsed time since previous cycle	82
16	Task 1 elapsed time current cycle (sec/min)	
17	Task 1 processor utilization	
18	Task 1 check point overrun count	
19	Task 1 cycle time overrun (sec/min)	

Table D-3 Harmony Area Controller (Continued)

Block No.	Definition	Function Code
20	Hours, time of day	90
21	Minutes, time of day	
22	Seconds, time of day	
23	No time synchronization flag: 0 = time of day invalid 1 = time of day valid	
24	Year (0 to 99)	
25	Month (1 to 12)	
26	Day (1 to 31)	
27	Day of week (1 to 7, Sunday = 1)	
28	Reserved	
29	Reserved	
30[1]	Configurable blocks	Any allowed function code - refer to Table D-1
31999	Reserved	57

NOTE:
1. The highest configurable block number is 31,998 for the HC800, BRC-400/410, HPG800 and HAC.

D.4 Module Status Information - HAC

Tables D-4 and D-5 explain Harmony area controller module status bytes.

Table D-4 Bit Description - HAC

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	FTX	RDF	RIO	LIO	CFG	NVF	NVI	STA
3	Error code							
4	Error code descriptor byte 1							
5	Error code descriptor byte 2							
6	ETYPE							
7	CHnetA	CHnetB	RDA	RDB	PER	COM	IOHnetA	IOHnetB
8	SIME	Reserved	Reserved	SIMES	Reserved			
9	IOHnetRA	IOHnetRB	Reserved	MOV	NDT1	NDT2	NDC1	NDC2
10	PRI	CFC	Reserved	CHK	RID	RDEXP	OCE	RDDET
11	PA	PB	PS	SOA	RNO	NOL	FDNL	BFM
12	NODES	RER1	RER2	TER1	TER2	RID1	RID2	RCF
14	Hardware nomenclature (hex)							

Table D-4 Bit Description - HAC (Continued)

Byte	Bit							
	7	6	5	4	3	2	1	0
15	Firmware revision letter (ASCII)							
16	Firmware revision number (ASCII)							

Table D-5 Byte Description - HAC

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 00 = configure, 10 = error, 11 = execute
	TYPE	1F	Module type=0x15 enhanced status (ETYPE).
2	FTX	80	First time in execute: 0 = no, 1 = yes
	RDF	40	Redundant failure status: 0 = good; no error or not configured for redundancy. 1 = bad; primary halted, backup controller module expected and not present or halted.
	RIO	20	Summary remote I/O status: 0=good, 1=bad. Physical I/O that is not available over local I/O expander bus and Hnet, but available via some other communications channel.
2 *(cont)*	LIO	10	Summary local I/O status: 0 = good, 1 = bad. Physical I/O that is available over local I/O expander bus and Hnet.
	CFG	08	Online configuration: 0 = no change to configuration. 1 = backup module configuration changed.
	NVF	04	NVRAM status: 0=good, 1=bad.
	NVI	02	NVRAM initialized (default configuration): 0=no, 1=yes.
	STA	01	Station error status: 0=good, 1=bad (no response). Bad indicates station offline due to loss of station link communication.

Table D-5 Byte Description - HAC (Continued)

Byte	Field	Field Size or Value	Description
3 - 5 Note 1	Error Code	3 4^2 5^2	
		01 01 — 02 — 03 — FF —	NVRAM error: Write failure Checksum failure Bad data Reset during write
		02 (1) (2)	Analog input reference error: (1), (2) = block number of control I/O module function block
		03 (1) (2)	Missing rack I/O module: (1), (2) = block number of I/O module or station
		04 — —	Checkpoint buffer allocation error – configuration is too large to fit in available checkpoint memory.
		05 (1) (2)	Configuration error – undefined block: (1), (2) = block number making reference
		06 (1) (2)	Configuration error – input data type is incorrect. Specification S8 for function code 57 was set too low. (1), (2) = block number making reference
		08 (1) (2)	Trip block activated: (1), (2) = block number of trip block
		09 — —	Segment violation - priority set the same in two segments, or more than eight segments defined.
		0F — —	Primary module has failed, and the redundant module configuration is not current.
		10 — —	Primary module has failed, and the dynamic RAM data in the redundant module is not current.
		11 — —	NVRAM write failure error.
		1C — —	Network I/O hardware is offline. Online configuration test cannot be completed.
		1E (1) (2)	Duplicate device definition label – multiple function code 221 function blocks contain the same device label. (1), (2) = block number making reference.

Table D-5 Byte Description - HAC (Continued)

Byte	Field	Field Size or Value	Description
3 - 5 *(cont)*	Error Code	3 4^2 5^2	
		1F (1) —	Software licensing error. (1) equals: 01 - License key not valid. 02 - Function block license exceeded. 03 - C license not available. 04 - Batch 90/UDF license not available.
		20 — —	Program format error - inconsistent format table.
		21 00 00 FF FE FF FF (1) (2)	File system error: Backup cannot take over due to uninitialized file system. Directory has not been configured. List of file system free memory is corrupted. (1), (2) = Number of files with errors.
		22 (1) (2)	Invoke C error: (1), (2) = block number making reference.
		24 (1) (2)	C program stack overflow: (1), (2) = block number making reference
		28 (1) (2)	User defined function (UDF) reference is invalid: (1), (2) = block number making reference
		29 (1) (2)	UDF block cannot read program file: (1), (2) = block number making reference
		2A (1) (2)	Not enough memory for UDF: (1), (2) = block number making reference
		2B (1) (2)	Missing UDF declaration: (1), (2) = block number making reference
		2C (1) (2)	Wrong UDF type: (1), (2) = block number making reference
		2D (1) (2)	Missing UDF auxiliary block: (1), (2) = block number making reference
		2E (1) (2)	UDF compiler and firmware are incompatible: (1), (2) = block number making reference
6	ETYPE	1F	Enhanced module type = 0x24 (module address 2 & 3).
7	CHnetA	80	Controller peer-to-peer Hnet channel/relay A failure: 0 = good, 1 = fail
	CHnetB	40	Controller peer-to-peer Hnet channel/relay B failure: 0 = good, 1 = fail
	RDA	20	Redundancy link channel 1 failure: 0 = good, 1 = fail
	RDB	10	Redundancy link channel 2 failure: 0 = good, 1 = fail
	PER	08	Peripheral failure: 0 = good, 1 = fail
	COM	40	I/O Hnet communication established: 0=no, 1 = yes
	IOHnetA	02	I/O Hnet channel A failure: 0 = good, 1 = fail
	IOHnetB	01	I/O Hnet channel B failure: 0 = good, 1 = fail
8	SIME	80	Simulation enabled: 0 = no, 1 = yes. This field is set for both INFI 90 OPEN and Symphony simulation.
	SIMES	10	Hnet I/O block simulation enabled: 0 = no, 1 = yes. This field, in addition to the SIME field, is set when Symphony simulation is enabled.

Table D-5 Byte Description - HAC (Continued)

Byte	Field	Field Size or Value	Description
9	IOHnetRA	80	I/O Hnet channel A relay fault: 0 = good, 1 = fail
	IOHnetRB	40	I/O Hnet channel B relay fault: 0 = good, 1 = fail
	MOV	10	Module memory overflow: 0 = no, 1 = yes
	NDT1	08	Network I/O board drive transistor/relay one failure: 0 = no, 1 = yes
	NDT2	04	Network I/O board drive transistor/relay two failure: 0 = no, 1 = yes
	NCD1	02	Network I/O board channel one failure: 0 = no, 1 = yes
	NCD2	01	Network I/O board channel two failure: 0 = no, 1 = yes
10	PRI	80	Primary/backup status: 0 = backup module, 1 = primary module
	CFC	40	Configuration current (latched until backup is reset): 0 = configuration initialization not complete. 1 = all configuration initialization complete.
	CHK	10	Backup has completed check pointing (latched until backup is reset). 0 = no, always set to 0 on the primary module. 1 = yes
	RID	08	Redundancy ID. Indicates position on the backplane: 0 = left position, 1 = right position
	RDEXP	04	Redundant module expected. 0 = no. 1 = yes, always set to 1 on the backup module. Follows state of function code 90, specification S3 on the primary module.
	OCE	02	Online configuration is enabled: 0 = no, 1 = yes.
	RDDET	01	Redundant module detected: 0 = no, 1 = yes. Latched until module is reset or it changes from backup to primary or primary to backup.
11	PA	80	24 VDC logic power input A status: 0 = good, 1 = bad
	PB	40	24 VDC logic power input B status: 0 = good, 1 = bad
	PS	20	Internal power system non-fatal error: 0 = good, 1 = bad. Status display on the HAC must be checked for further information on the cause of the PS flag being set.
	SOA	10	Status output alarm: 0 = good; 1 = bad. Indicates the status of the system +24 volt power and the I/O block's power (logic and field power) for a single cabinet.
	RNO	08	Redundancy NVM overrun (latched indication): 0 = no. 1 = yes, set to 1 in primary module if NVM checkpoint overruns have occurred. NVM checkpoint overruns cause the primary module to reset the backup module.
	NOL	04	Network I/O board offline: 0 = no, 1 = yes.
	FDNL	02	Firmware download in progress: 0 = no, 1 = yes
	BFM	01	Backup firmware mismatch: 0 = no. 1 = yes, backup has determined the primary firmware revision is different from its own.

Table D-5 Byte Description - HAC (Continued)

Byte	Field	Field Size or Value	Description
12	NODES[3]	80	Nodes offline: 0 = no, 1 = yes
	RER1	40	Receive errors on loop channel 1: 0 = no, 1 = yes
	RER2	20	Receive errors on loop channel 2: 0 = no, 1 = yes
	TER1	10	Transmit errors on loop channel 1: 0 = no, 1 = yes
	TER2	08	Transmit errors on loop channel 2: 0 = no, 1 = yes
	RID1	04	Receiver idle on loop channel 1: 0 = no, 1 = yes
	RID2	02	Receiver idle on loop channel 2: 0 = no, 1 = yes
	RCF	01	Local Cnet loop communication failure: 0 = no, 1 = yes
14	—	FF	Module nomenclature: 0x08 = HAC
15	—	FF	Revision letter (in ASCII code). For example, 0x42 = B
16	—	FF	Revision number (in ASCII code). For example, 0x38 = 7

NOTES:

1. The error description corresponding to byte 3 is displayed on the front panel LCD display when the module is in ERROR mode.
2. All block numbers are encoded in hexadecimal, with (1) equaling the most significant digits and (2) equaling the least significant digits. Example: (1) = 4E, (2) = 20 is block number 20,000.
3. Active only when the controller is in execute mode.

E. Multi-Function Processors (IMMFP11/12)

E.1 Memory Utilization and Execution Times

This appendix lists the module memory requirements and execution times for each available function code for the IMMFP11/12 multi-function processors. Two quantities are given for these modules. The first value is the number of bytes of nonvolatile random access memory (NVRAM). The second quantity is the number of bytes of random access memory (RAM). The IMMFP11 has a total configuration memory of 62,656 bytes of NVRAM and 163,248 bytes of RAM. The IMMFP12 has a total configuration memory of 194,752 bytes of NVRAM and 347,568 bytes of RAM.

Table E-1 shows the module memory requirements and execution times for each function code.

NOTES:

1. Except where otherwise noted, execution times are given for worst case conditions.
2. Refer to ***Memory Usage Equations*** in this section for the equations listed in Table E-1.

Table E-1 Module Memory Utilization and Execution Times

Function Code	Description	NVRAM Bytes	RAM Bytes	Execution Times (in µsecs)
1	Function generator	46	88	38
2	Manual set constant	12	40	17
3	Lead/lag	18	72	193
4	Pulse positioner	24	88	105
5	Pulse rate	16	72	220
6	High/low limiter	16	48	39
7	Square root	12	44	183
8	Rate limiter	18	64	126
9	Analog transfer	20	76	204
10	High select	16	52	49
11	Low select	16	52	49
12	High/low compare	16	48	36
13	Integer transfer	14	44	22
14	Four input summer	16	52	72
15	Two input summer	18	52	96
16	Multiply	14	48	84
17	Divide	14	48	104
18	PID error input	34	108	658
19	PID process variable and set point	36	116	752
24	Adapt	12	50	28
25	Analog input (same PCU node)	12	100	38
26	Analog input/loop	12	54	42
30	Analog exception report	26	102	120
31	Test quality	16	52	43

Table E-1 Module Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM Bytes	RAM Bytes	Execution Times (in µsecs)
32	Trip	12	36	17
33	Not	12	3	19
34	Memory	14	46	23
35	Timer	14	58	54
36	Qualified OR (eight input)	26	68	60
37	AND (two input)	12	40	20
38	AND (four input)	16	48	27
39	OR (two input)	12	40	20
40	OR (four input)	16	48	27
41	Digital input/bus	12	88	38
42	Digital input/loop	12	42	33
45	Digital exception report	12	66	80
50	Manual set switch	12	36	16
51	Manual set constant (nontunable)	12	40	17
52	Manual set integer	12	36	17
55	Hydraulic servo	60	404	3,552
58	Time delay (analog)	18	Equation 1	141
59	Digital transfer	14	44	22
61	Blink	12	48	102
62	Remote control memory	28	96	91
63	Analog input list	28	286	167
64	Digital input list (module bus)	28	190	159
65	Digital sum with gain (four input)	28	68	89
66	Analog trend Normal mode Fast mode	 12 12	 196 700	 112 70
68	Remote manual set constant (REMSET)	22	128	75
69	Test alarm	12	44	29
79	Control interface slave First FC 79 block in segment Additional FC 79 blocks in segment	 48 48	 328 328	 16,302 2,724
80	Control station	84	366	350
81	Executive block (MFP)	22	140	0
82	Segment control	64	260	0

Table E-1 Module Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM Bytes	RAM Bytes	Execution Times (in µsecs)
83	Digital output group	32	90	275
84	Digital input group	16	114	295
85	Up/down counter	24	76	64
86	Elapsed timer	20	84	66
89	Last block	12	40	0
90	Extended executive (MFP)	52	152	0
91	BASIC configuration	Equation 2	Equation 3	15
92	Invoke BASIC	22	46	Program dependent
93	BASIC real output	12	64	15
94	BASIC boolean output	12	72	15
95	Module status monitor	22	108	161
96	Redundant analog input	22	106	105
97	Redundant digital input	14	58	56
98	Slave select	26	84	58
99	Sequence of events log	14	Equation 4	364
100	Digital output readback check	40	144	185
101	Exclusive OR	12	40	20
102	Pulse input/period	24	100	372
103	Pulse input/frequency	24	100	449
104	Pulse input/totalization	26	130	612
109	Pulse input/duration	24	100	379
110	Five input rung	24	74	45
111	Ten input rung	38	104	66
112	Twenty input rung	68	164	108
114	BCD input	20	72	300
115	BCD output	22	7	380
116	Jump/master control relay	12	52	17
117	Boolean recipe table	28	68	55
118	Real recipe table	48	102	56
119	Boolean signal multiplexer	32	86	48
120	Real signal multiplexer	32	90	49
121	Analog input/Cnet	22	82	80

Table E-1 Module Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM Bytes	RAM Bytes	Execution Times (in µsecs)
122	Digital input/Cnet	20	50	80
123	Device driver	30	118	110
124	Sequence monitor	80	162	45
125	Device monitor	40	102	359
126	Real signal demultiplexer	12	92	122
128	Slave default definition	44	72	15
129	Multistate device driver	60	168	137
132	Analog input/slave	54	340	1,065
133	Smart field device definition	36	144	60
134	Multi-sequence monitor	96	268	48
135	Sequence manager	82	208	192
136	Remote motor control	52	186	136
137	BASIC real output/quality	12	80	15
138	BASIC boolean output/quality	12	64	15
139	Passive station interface	44	146	160
140	Restore Largest NVM utilization (FC 165 with S2 = 249) Smallest NVM utilization (FC 33)	 Equation 5 Equation 5	 88 88	 30,650 635
141	Sequence master	80	142	120
142	Sequence slave	74	104	15
143	Invoke C	28	108	Program dependent
144	C allocation	Equation 6	Equation 7	17
145	Frequency counter/slave	30	120	122
146	Remote I/O interface	68	340	2,917
147	Remote I/O definition	144	480	243
148	Batch sequence	36	Equation 8	Program dependent
149	Analog output/slave First FC 149 in segment Second FC 149 in segment (same ASO as first) Additional FC 149s in segment	 82 82 82	 318 318 318	 18,494 18,494 3,071
150	Hydraulic servo slave	32	168	447
151	Text selector	46	152	47
152	Model parameter estimator	28	352	58

Table E-1 Module Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM Bytes	RAM Bytes	Execution Times (in µsecs)
153	Inferential smith controller parameter converter	42	168	284
154	Adaptive parameter scheduler	36	224	128
155	Regression	68	Equation 9	105
156	Advanced PID controller	58	190	900
157	General digital controller	68	Equation 10	138
160	Smith predictor	36	342	530
161	Sequence generator	74	212	78
162	Digital segment buffer	16	78	54
163	Analog segment buffer	16	94	56
165	Moving average	16	Equation 11	240
166	Integrator	28	92	168
167	Polynomial Most complicated All zero	 58 58	 104 104	 655 193
168	Interpolator	98	196	488
169	Matrix addition	44	172	230
170	Matrix multiplication	44	172	1,800
171	Trigonometric Sine Secant	 14 14	 48 48	 510 555
172	Exponential	12	44	400
173	Power	14	48	850
174	Logarithm	14	50	480
177	Data acquisition analog	104	398	384
178	Data acquisition analog input/loop	20	134	118
179	Enhanced trend Normal sampling Normal and statistical sampling	 52 52	 Equation 12 Equation 12	 360 734
184	Factory instrumentation protocol handler	42	1,342	5,222
185	Digital input subscriber	32	220	262
186	Analog input subscriber	32	252	1,009
187	Analog output subscriber	48	232	1,790
188	Digital output subscriber	48	208	241
190	User defined function declaration	30	Equation 13	16

Table E-1 Module Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM Bytes	RAM Bytes	Execution Times (in µsecs)
191	User defined function one	56	Equation 14	Program dependent
192	User defined function two	70	Equation 15	Program dependent
193	User defined data import	48	Equation 16	100
194	User defined data export	48	Equation 17	175
198	Auxiliary real user defined function	38	150	16
199	Auxiliary digital user defined function	38	134	16
210	Sequence of events slave	100	572	1,320
211	Data acquisition digital	62	310	448
212	Data acquisition digital input/loop	20	110	70
215	Enhanced analog slave definition	30	206	1,670
216	Enhanced analog input definition	42	114	584
217	Enhanced calibration command	24	178	33
219	Common sequence	46	Equation 18	Program dependent
220	Batch historian	46	Equation 19	Program dependent
241	DSOE data interface SEM to MFP No SED resynch function performed SED resynch performed once every second	 18 18	 284 284	 360 900
242	DSOE digital event interface No data available on SED I/O module Data available on SED I/O module	 84 84	 338 338	 300 1,600

E.2 Memory Usage Equations

1. 78 + [8 x (S5)]
2. 38 + [1024 x (S5)]
3. 4994 + 1024 [(S3) + (S4) + (S5)]
4. 96 + [9 x (S2)]
5. Refer to function code 140, module memory utilization
6. 12 + [1024 x (S2)]
7. 36 + [1024 x (S1)]
8. 826 + [1024 x (S11)] + [a x (S12)]

   where:

   a = 2 (if S12 is positive)
   - or -
   a = 2048 (if S12 is negative)
9. 338 + [40 x (S7)]
10. 228 + [8 x (S21 + S22)]
11. 98 + [8 x (S2)]

12. a + [b x (S4)] + [c x (n{S3})],
    where
    n{S3} = Number of modes selected by S3
    if S2 = 0 or 3, then a = 396, b = 24, and c = 16
    if S2 = 1, 2, 5, 6, 7 or 9, then a = 380, b = 20, and c = 8
    if S2 = 4 or 8, then a = 456, b = 50, and c = 64
13. 74 + [10 x (S2)]
14. 520 + [2 x (S16)]
15. 256 + [3 x (S5)]
16. 604 + [2 x (S24)]
17. 256 + [5 x (S4)]
18. 424 + [222 x (S13)]
19. 666 + [2 x (S11)]

E.3 Function Blocks - IMMFP11/12

Table E-2 contains function code block number information for the IMMFP11/12.

Table E-2 Function Blocks for the IMMFP11/12 Multi-Function Processors

Block No.	Definition	Function Code
0	Logic 0	81
1	Logic 1	
2	0 or 0.0	
3	-100.0	
4	-1.0	
5	0.0	
6	1.0	
7	100.0	
8	-9.2 E18	
9	9.2 E18	
10	Start-up flag (0 = no, 1 = yes)	
11	Memory display value	
12	System free time in percent	
13	Revision level	81
14	Reserved	
15	Task 1 elapsed time since previous cycle	82
16	Task 1 elapsed time current cycle (sec/min)	
17	Task 1 processor utilization	
18	Task 1 check point overrun count	
19	Task 1 cycle time overrun (sec/min)	

Table E-2 Function Blocks for the IMMFP11/12 Multi-Function Processors

Block No.	Definition	Function Code
20	Hours, time of day	90
21	Minutes, time of day	
22	Seconds, time of day	
23	No time synchronization flag: 0 = time of day invalid 1 = time of day valid	
24	Year (0 to 99)	
25	Month (1 to 12)	
26	Day (1 to 31)	
27	Day of week (1 to 7, Sunday = 1)	
28	Reserved	
29	Reserved	
30[1]	Configurable blocks	Note 2
Note 3	Loop type: 0.0 = Plant Loop 1.0 = Cnet 3.0 = Cnet with time-stamping	89

NOTES:
1. The highest configurable block number is 9998 for IMMFP11/12.
2. Any allowed function code - refer to Table E-1.
3. The Block number is 9998 for IMMFP11/12.

E.4 Module Status Information - IMMFP11/12

Table E-3 and E-4 explain IMMFP11/12 Multi-Function Processor module status bytes.

Table E-3 Bit Description - IMMFP11/12

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	FTX	BAC	RIO	LIO	CFG	NVF	NVI	DSS
3	Error code							
4	Error code descriptor (1)							
5	Error code descriptor (2)							
6	ETYPE							
7	CWA	CWB	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
8	SIME	SIMR	SIMT	Reserved				
9	Reserved	Reserved	Reserved					
10	PRI	CFC	Reserved	CHK	RID	RDEXP	OCE	RDDET
11	Reserved	Reserved	Reserved	Reserved	RNO	Reserved	Reserved	Reserved
12-13	Reserved							

Table E-3 Bit Description - IMMFP11/12 (Continued)

Byte	Bit							
	7	6	5	4	3	2	1	0
14	Module nomenclature							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

Table E-4 Byte Description - IMMFP11/12

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 00 = configure, 10 = error, 11 = execute
	TYPE	1F	Module type code: $(15)_{16}$ = Enhanced status
2	FTX	80	First time in execute: 0 = no, 1 = yes
	BAC	40	Backup status: 0 = good, 1 = bad
	RIO	20	Summary remote input status: 0 = good, 1 = bad
	LIO	10	Summary local input status: 0 = good, 1 = bad
	CFG	08	On-line configuration changes being made
	NVF	04	Summary NVRAM failure status: 0 = good, 1 = fail
	NVI	02	Summary NVRAM initialized state: 0 = no, 1 = yes
	DSS	01	Digital station status: 0 = good, 1 = bad

Table E-4 Byte Description - IMMFP11/12 (Continued)

Byte	Field	Field Size or Value			Description
3 - 5 Note 1	Error Code	3	4	5	
		01	01 02 03 FF	— — — —	NVRAM error: Write failure Checksum failure Bad data Reset during write
		02	(1)	(2)	Analog input reference error: (1), (2) = block number of control interface I/O module block
		03	(1)	(2)	Missing I/O module or expander board: (1), (2) = block number of I/O module or station
		05	(1)(	2)	Configuration error – undefined block: (1), (2) = block number making reference
		06	(1)	(2)	Configuration error – input data type is incorrect: (1), (2) = block number making reference
		08	(1)	(2)	Trip block activated: (1), (2) = block number of trip block
		09	—	—	Segment violation - priority set the same in two segments, or more than eight segments defined.
		0F	—	—	Primary module has failed and the redundant module configuration is not current
		10	—	—	Primary module has failed and the dynamic RAM data in the redundant module is not current
		11	—	—	NVRAM write failure error
		20	—	—	Program format error - inconsistent format table
		21	00 FF FF (1)	00 FE FF (2)	File system error: Backup cannot take over due to uninitialized file system. Directory has not been configured. List of file system free memory is corrupted. (1), (2) = Number of files with errors.

Table E-4 Byte Description - IMMFP11/12 (Continued)

Byte	Field	Field Size or Value	Description
3 - 5 Note 1 *(cont)*	Error Code	3 4 5	
		22 (1) (2)	Invoke C error: (1), (2) = block number making reference
		24 (1) (2)	C program stack overflow: (1), (2) = block number making reference
		28 (1) (2)	User defined function (UDF) reference is invalid: (1), (2) = block number making reference
		29 (1) (2)	UDF block cannot read program file: (1), (2) = block number making reference
		2A (1) (2)	Not enough memory for UDF: (1), (2) = block number making reference
		2B (1) (2)	Missing UDF declaration: (1), (2) = block number making reference
		2C (1) (2)	Wrong UDF type: (1), (2) = block number making reference
		2D (1) (2)	Missing UDF auxiliary block: (1), (2) = block number making reference
		2E (1) (2)	UDF compiler and firmware are incompatible: (1), (2) = block number making reference
		2F (1) (2)	BASIC program error: (1), (2) = line number of error
6	ETYPE	1F	Enhanced module type = $(24)_{16}$ = IMMFP11/12
7	CWA	80	Controlway bus A failure: 0 = good, 1 = fail
	CWB	40	Controlway bus B failure: 0 = good, 1 = fail
8	SIME	80	Simulation enabled: 0 = normal operation, 1 = simulation active
	SIMR	40	Simulation running/frozen: 0 = simulation frozen, 1= simulation running
	SIMT	20	Simulation time rate: 0 = real time, 1 = slow/fast time
10	PRI	80	Module is primary versus backup; set to 1 in the primary module.
	CFC	40	Configuration current (latched until backup is reset). Set when LED 7 is enabled (1 = on or blinking) on the backup module.
	CHK	10	Backup has completed checkpointing (latched until backup is reset). Always set to 0 on the primary module. Follows LED 8 (1 = on or blinking) on the backup module.
	RID	08	Redundancy ID. Follows setting of redundancy ID pole on the dipswitch.
	RDEXP	04	Redundancy expected. Always set to 1 on the backup module. Follows state of function code 90, specification S3, ones digit on the primary module.
	OCE	02	On-line configuration is enabled. Follows setting of on-line configuration enable pole on dipswitch.
	RDDET	01	Redundancy detected (latched until module is reset or it changes from backup to primary or primary to backup). Set to 1 when a properly configured redundant module is detected.

Table E-4 Byte Description - IMMFP11/12 (Continued)

Byte	Field	Field Size or Value	Description
11	RNO	08	Redundancy NVM overrun (latched indication). Set to 1 in primary module if NVM checkpoint overruns have occurred. NVM checkpoint overruns cause the primary module to reset the backup module.
12-13	—	00	Reserved
14	—	FF	Module nomenclature: $(01)_{16}$ = IMMFP11, $(02)_{16}$ = IMMFP12
15	—	FF	Revision letter (in ASCII code), for example, $(46)_{16}$ = F, $(47)_{16}$ = G
16	—	FF	Revision number (in ASCII code), for example, $(38)_{16}$ = 8

NOTE:

1. Byte 3 is displayed on the front panel LEDs when the module is in ERROR mode.

F. Sequence of Events Monitor (INSEM01 and INSEM11/SPSEM11)

F.1 Memory Utilization

This appendix lists the module memory requirements for each available function code for the INSEM01 and INSEM11/ SPSEM11 (SEM11) Cnet to computer communications and sequence of events monitor. Two quantities are given for the module. The first value is the number of bytes of nonvolatile random access memory (NVRAM). The second quantity is the number of bytes of random access memory (RAM).
Table F-1 lists the configuration memory available the module. Table F-2 shows the module memory requirements for each function code. Tables F-3 and F-4 explain module status.

Table F-1 Available Module Configuration Memory

Module	NVRAM Bytes	RAM Bytes
INSEM01	65,536	262,144

Table F-2 Module Memory Utilization for Function Codes

Function Code	Description	NVRAM Bytes	RAM Bytes
243	Executive block (INSEM01)	24	50
244	Addressing interface definition	20	50
245	Input channel interface	28	52
246	Trigger definition	72	92

F.2 Module Status Information

Table F-3 Bit Description - INSEM01 and SEM11

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	FTX		RIO	LIO	CFM			ERM
3	CSP	MOV		LOP	SDF	NODE TYPE		
4	NSF	LR1	LR2	LT1	LT2	RI1	RI2	RCF
5	HOST							
6	ETYPE							
7 - 8	Reserved							
9	NDT1	NDT2			NCD1	NCD2		
10 - 13	Reserved							
14	Module nomenclature							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

Table F-4 Byte Description - INSEM01

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 00 = configure, 01 = fail, 10 = error, 11 = execute
	TYPE	1F	Module type code: $(15)_{16}$ = enhanced status
2	FTX	80	First time in execute: 0 = no, 1 = yes
	RIO	20	Summary remote input status: 0 = good, 1 = bad
	LIO	10	Summary local input status: 0 = good, 1 = bad
	CFM	08	Module in configure mode (MODE = 00): 0 = no, 1 = yes
	ERM	01	Module in error mode (MODE = 10): 0 = no, 1 = yes
3	CSP	80	Communication status problem: 0 = no, 1 = yes
	MOV	40	Memory overflow: 0 = good, 1 = bad
	LOP	10	Off-line bridge for remote loop: 0 = no, 1 = yes
	SDF	01	Security device failure
	NODE TYPE	07	Node type: 0 = computer interface, 1 = operator interface unit or management command system, 2 = operator interface unit
4	NSF	80	Node environment status flag: 0 = good, 1 = bad
	LR1	40	Cnet 1 receive error: 0 = no, 1 = yes
	LR2	20	Cnet 2 receive error: 0 = no, 1 = yes
	LT1	10	Cnet 1 transmit error: 0 = no, 1 = yes
	LT2	08	Cnet 2 transmit error: 0 = no, 1 = yes
	RI1	04	Receiver idle on channel 1 of central loop: 0 = no, 1 = yes
	RI2	02	Receiver idle on channel 2 of central loop: 0 = no, 1 = yes
	RCF	01	Ring communication failure: 0 = no, 1 = yes
5	HOST	FF	Value set by the computer (when non-zero, ES = 1)
6	ETYPE	20	Enhanced module type $(2A)_{16}$ = INSEM01
7 - 8	—	00	Reserved
9	NDT1	01	NIS loop relay drive transistor 1 failure
	NDT2	01	NIS loop relay drive transistor 2 failure
	NCD1	01	NIS receive channel 1 disable
	NCD2	01	NIS receive channel 2 disable
10 - 13	—	—	Reserved
14	—	FF	Module nomenclature $(01)_{16}$ = INSEM01
15	—	FF	Revision letter (in ASCII code), for example, $(45)_{16}$ = E
16	—	FF	Revision number (in ASCII code), for example, $(30)_{16}$ = 0

G. Communication Modules

G.1 Introduction

This appendix describes the status bytes and their descriptions for the following communication modules:

- INICT03/03A Cnet to Computer Transfer Module
- INICT12 Cnet to Computer Transfer Module
- INICT13A and SPICT13A Cnet to Computer Transfer Module
- INIET800 and SPIET800 Cnet to Computer Transfer Module
- INIIT03/13 and SPIIT13 Local Transfer Module
- INNPM11/12/22 and SPNPM22 Network Processing Module
- PNI800 Plant Network Interface 800 Module
- CP800 Communications Processor Module (HPC800)
- SPIPT800 INFI-NET to PN800 Transfer Module

G.2 IINICT03/03A/13 and INICT12 Cnet to Computer Transfer Modules

Table G-5 lists the fields that make up the INICT03/03A/13 and INICT12 module status report. Table G-6 describes each field within the module status record.

Table G-5 INICT03/03A, INICT13 and INICT12 Status Bytes

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	FTX		RIO	LIO	CFM			ERM
3	PCUs (CSP)	MOV		RINGs (LOP)	SDF	NODE TYPE		
4	NSF	RER1 (LR1)	RER2 (LR2)	TER1 (LT1)	TER2 (LT2)	RI1	RI2	RCF
5	HOST							
6	ETYPE							
7 - 8	Reserved							
9	NDT1	NDT2			NCD1	NCD2		
10 - 12	Reserved							
13	Module type							
14	Module nomenclature							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

Table G-6 INICT03/03A, INICT13 and INICT12 Status Bits

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 00 = configure, 01 = fail, 10 = error, 11 = execute
	TYPE	1F	Module type code: $(15)_{16}$ = enhanced status
2	FTX	80	First time in execute: 0 = no, 1 = yes
	RIO	20	Summary remote input status: 0 = good, 1 = bad
	LIO	10	Summary local input status: 0 = good, 1 = bad
	CFM	08	Module in configure mode (MODE = 00): 0 = no, 1 = yes
	ERM	01	Module in error mode (MODE = 10): 0 = no, 1 = yes
3	PCUs (CSP)	80	Communication status problem: 0 = no, 1 = yes
	MOV	40	Memory overflow: 0 = good, 1 = bad
	RINGs (LOP)	10	Offline bridge for remote ring: 0 = no, 1 = yes
	SDF	01	Security device failure
	NODE TYPE	07	Node type: 0 = computer interface, 1 = operator interface unit or management command system, 2 = operator interface unit
4	NSF	80	Node environment status flag: 0 = good, 1 = bad
	RER1 (LR1)	40	Cnet 1 receive error: 0 = no, 1 = yes
	RER2 (LR2)	20	Cnet 2 receive error: 0 = no, 1 = yes
	TER1 (LT1)	10	Cnet 1 transmit error: 0 = no, 1 = yes
	TER2 (LT2)	08	Cnet 2 transmit error: 0 = no, 1 = yes
	RI1	04	Receiver idle on channel 1: 0 = no, 1 = yes
	RI2	02	Receiver idle on channel 2: 0 = no, 1 = yes
	RCF	01	Ring communication failure: 0 = no, 1 = yes
5	HOST	FF	Value set by the computer (when non-zero, ES = 1)
6	ETYPE	20	Enhanced module type $(20)_{16}$ = INICT12 or INICT03/03A/13
7 - 8	—	00	Reserved
9	NDT1	01	NIS ring relay drive transistor 1 failure
	NDT2	01	NIS ring relay drive transistor 2 failure
	NCD1	01	NIS receive channel 1 disable
	NCD2	01	NIS receive channel 2 disable
10 - 12	—	—	Reserved

Table G-6 INICT03/03A, INICT13 and INICT12 Status Bits (Continued)

Byte	Field	Field Size or Value	Description
13	—	FF	Module type: 1=SSM, 2=MCP
14	—	FF	Module nomenclature: $(0C)_{16}$ = INICT12, $(03)_{16}$ = INICT03/03A/13
15	—	FF	Revision letter (in ASCII code). For example, $(45)_{16}$ = E
16	—	FF	Revision number (in ASCII code). For example, $(30)_{16}$ = 0

G.3 INIET800 and SPIET800 Cnet to Computer Transfer Modules

Table G-7 lists the lists the fields that make up the INIET800 and SPIET800 (IET800) module status report. Table G-8 describes each field within the module status record.

Table G-7 IET800 Status Bytes

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	FTX		RIO	LIO	CFM			ERM
3	PCUs (CSP)	MOV	LFF	RINGs (LOP)	SDF	NODE TYPE		
4	NSF	RER1 (LR1)	RER2 (LR2)	TER1 (LT1)	TER2 (LT2)	RI1	RI2	RCF
5	HOST							
6	ETYPE							
7			RDA	RDB				
8	Reserved							
9	NDT1	NDT2			NCD1	NCD2		
10 - 12	Reserved							
13	Module type							
14	Module nomenclature							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

Table G-8 IET800 Status Bits

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 00 = configure, 01 = fail, 10 = error, 11 = execute
	TYPE	1F	Module type code: $(15)_{16}$ = enhanced status

Table G-8 IET800 Status Bits

Byte	Field	Field Size or Value	Description
2	FTX	80	First time in execute: 0 = no, 1 = yes
	RIO	20	Summary remote input status: 0 = good, 1 = bad
	LIO	10	Summary local input status: 0 = good, 1 = bad
	CFM	08	Module in configure mode (MODE = 00): 0 = no, 1 = yes
	ERM	01	Module in error mode (MODE = 10): 0 = no, 1 = yes
3	PCUs (CSP)	80	Communication status problem: 0 = no, 1 = yes One or more remote PCUs are offline
	MOV	40	Memory overflow: 0 = good, 1 = bad
	LFF	20	Licensing failure
	RINGs (LOP)	10	Offline bridge for remote ring: 0 = no, 1 = yes
	SDF	01	Security device failure
	NODE TYPE	07	Node type: 0 = computer interface, 1 = operator interface unit or management command system, 2 = operator interface unit
4	NSF	80	Node environment status flag: 0 = good, 1 = bad
	RER1 (LR1)	40	Cnet 1 receive error: 0 = no, 1 = yes
	RER2 (LR2)	20	Cnet 2 receive error: 0 = no, 1 = yes
	TER1 (LT1)	10	Cnet 1 transmit error: 0 = no, 1 = yes
	TER2 (LT2)	08	Cnet 2 transmit error: 0 = no, 1 = yes
	RI1	04	Receiver idle on channel 1: 0 = no, 1 = yes
	RI2	02	Receiver idle on channel 2: 0 = no, 1 = yes
	RCF	01	Ring communication failure: 0 = no, 1 = yes
5	HOST	FF	Value set by the computer (when non-zero, ES = 1)
6	ETYPE	20	Enhanced module type $(20)_{16}$ = SSM/CIU/ICT/IET
7	RDA	20	Redundancy link bus A error
	RDB	10	Redundancy link bus B error
8	—	00	Reserved
9	NDT1	01	NIS ring relay drive transistor 1 failure
	NDT2	01	NIS ring relay drive transistor 2 failure
	NCD1	01	NIS receive channel 1 disable
	NCD2	01	NIS receive channel 2 disable
10 - 13	—	—	Reserved

Table G-8 IET800 Status Bits

Byte	Field	Field Size or Value	Description
14	—	FF	Module nomenclature: $(03)_{16}$ = IET800 in serial mode $(04)_{16}$ = IET800 in Ethernet mode
15	—	FF	Revision letter (in ASCII code). For example, $(45)_{16}$ = E
16	—	FF	Revision number (in ASCII code). For example, $(30)_{16}$ = 0

G.4 INIIT03 and INIIT13/SPIIT13 Local Transfer Module

Table G-9 lists the fields that make up the INIIT03 and INIIT13/SPIIT13 (IIT13) module status report. Table G-10 describes each field within the module status record.

Table G-9 IIT03/13 Status Bytes

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2			SIO	CIO			NVI	
3	PCUs	MOV	NODS	BKCFG	BKSTS	RINGs		PSI
4[1]	NSFC	RER1C	RER2C	TER1C	TER2C	RID1C	RID2C	RCFC
5[1]	NSFS	RER1S	RER2S	TER1S	TER2S	RID1S	RID2S	RCFS
6	ETYPE							
7 - 13	Reserved							
14	Module nomenclature							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

NOTE: 1. The last letter of the codes (S or C) in bytes 4 and 5 indicates if the status applies to the satellite ring or the central ring.

Table G-10 IIT03/13 Status Bits

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 01 = fail, 11 = execute
	TYPE	1F	Module type code: $(15)_{16}$ = INIIT03

Table G-10 IIT03/13 Status Bits (Continued)

Byte	Field	Field Size or Value	Description
2	SIO	20	I/O status for data sourced by satellite ring PCUs: 0 = good, 1 = bad
	CIO	10	I/O status for data sourced by central ring PCUs: 0 = good, 1 = bad
	NVI	02	Default configuration: 0 = blocks configured, 1 = no blocks configured (INIPT01 only)
3	PCUs	80	Nodes off-line: 0 = no, 1 = yes
	MOV	40	Memory overflow: 0 = no, 1 = yes
	BKCFG	10	Redundant configuration: 0 = no, 1 = yes
	BKSTS	08	Secondary failed: 0 = no, 1 = yes
	RINGs	04	Rings off-line: 0 = on-line, 1 = off-line bridge for remote rings
	PSI	01	Primary/secondary indicator: 0 = primary, 1 = secondary
4 - 5[1]	NSF	80	Node environment status flag: 0 = good, 1 = bad
	RER1	40	Receive errors on channel 1: 0 = no, 1 = yes
	RER2	20	Receive errors on channel 2: 0 = no, 1 = yes
	TER1	10	Transmit errors on channel 1: 0 = good, 1 = errors
	TER2	08	Transmit errors on channel 2: 0 = good, 1 = errors
	RID1	04	Receiver idle of channel 1: 0 = no, 1 = yes
	RID2	02	Receiver idle of channel 2: 0 = no, 1 = yes
	RCF	01	Ring communication failure (IIT03 only)
6	ETYPE	FF	Extended module type: $(23)_{16}$ = IIT03
7 - 13	—	00	Reserved
14	—	FF	Module nomenclature: $(03)_{16}$ = IIT03/13
15	—	FF	Revision letter (in ASCII code), for example, $(42)_{16}$ = B
16	—	FF	Revision number (in ASCII code), for example, $(31)_{16}$ = 1

NOTE:

1. The last letter of the codes (S or C) in bytes 4 and 5 indicates if the status applies to the satellite ring or the central ring.

G.5 INNPM11/22, INNPM12 and SPNPM22 Network Processing Modules

The INNPM12 presents two different status summaries depending on its mode of operation (i.e. Cnet or Plant Loop mode) set with switch SW3. The INNPM11 and INNPM22/SPNPM22 (NPM22) operate only in Cnet mode. In Cnet mode the modules have 16-byte module status records. When in Plant Loop mode, the INNPM12 module has a five-byte module status record.

G.5.1 Cnet Mode

Table G-11 lists the fields that make up the status report for the INNPM11/12 and NPM22 modules in Cnet mode. Table G-12 describes each field within the module status record.

Table G-11 INNPM11/12 and NPM22 Cnet Mode Status Bytes

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	Reserved							
3	CSP	MOV		BKCFG	BKSTS			PSI
4	NSF	LR1	LR2	LT1	LT2	RI1	RI2	RCF
5	Reserved							
6	ETYPE							
7	CWA	CWB						
8	Reserved							
9	NDT1	NDT2			NCD1	NCD2		
10 - 13	Reserved							
14	Module nomenclature							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

Table G-12 INNPM11/12/22 Cnet Mode Status Bits

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors.
	MODE	60	Module mode: 10 = error, 11 = execute.
	TYPE	1F	Module type: 0x15 = enhanced status (ETYPE).
2	—	00	Reserved
3	CSP	80	Communication status problem: 0 = no, 1 = yes
	MOV	40	Memory overflow: 0 = good, 1 = bad
	BKCFG	10	Redundant configuration: 0 = no, 1 = yes
	BKSTS	08	Backup failed: 0 = no, 1 = yes
	PSI	00	Primary/backup indicator: 0 = primary, 1 = backup

Table G-12 INNPM11/12/22 Cnet Mode Status Bits (Continued)

Byte	Field	Field Size or Value	Description
4	NSF	80	Node environment status flag: 0 = good, 1 = bad
	LR1	40	Cnet 1 receive error: 0 = no, 1 = yes
	LR2	20	Cnet 2 receive error: 0 = no, 1 = yes
	LT1	10	Cnet 1 transmit error: 0 = no, 1 = yes
	LT2	08	Cnet 2 transmit error: 0 = no, 1 = yes
	RI1	04	Receiver idle on channel 1: 0 = no, 1 = yes
	RI2	02	Receiver idle on channel 2: 0 = no, 1 = yes
	RCF	01	ring communication failure: 0 = no, 1 = yes
5	—	00	Reserved
6	ETYPE	20	Enhanced module type: 0x25 = INNPM12 or INNPM11
7	CWA	80	Controlway channel A failure: 0 = no, 1 = yes
	CWB	40	Controlway channel B failure: 0 = no, 1 = yes
8	—	00	Reserved
9	NDT1	80	NIS ring relay drive transistor 1 failure: 0 = no, 1 = yes
	NDT2	40	NIS ring relay drive transistor 2 failure: 0 = no, 1 = yes
	NCD1	08	NIS receive channel 1 disable: 0 = no, 1 = yes
	NCD2	04	NIS receive channel 2 disable: 0 = no, 1 = yes
10 - 13	—	—	Reserved
14	—	FF	Module nomenclature: 0x01 = INNPM12 or INNPM11
15	—	FF	Revision letter (in ASCII code). For example, 0x41 = A
16	—	FF	Revision number (in ASCII code). For example, 0x30 = 0

G.5.2 Plant Loop Mode

Table G-13 lists the fields that make up the INNPM12 module status report when in Plant Loop mode. Table G-14 describes each field within the module status record.

Table G-13 INNPM12 Status Bytes (Plant Loop Mode)

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	Reserved							
3	CSP	MOV	NSF	BKCFG	BKSTS			PSI
4		LR1	LR2	LT1	LT2			
5	Reserved							

Table G-14 INNPM12 Status Bits (Plant Loop Mode)

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 10 = error, 11 = execute
	TYPE	1F	Module type: 0x08 = INNPM12 (Plant Loop mode)
2	—	00	Reserved
3	CSP	80	Communication status problem: 0 = no, 1 = yes
	MOV	40	Memory overflow: 0 = good, 1 = bad
	NSF	20	Node environment status flag: 0 = good, 1 = bad
	BKCFG	10	Redundant configuration: 0 = no, 1 = yes
	BKSTS	08	Backup failed: 0 = no, 1 = yes
	PSI	01	Primary/backup indicator: 0 = primary, 1 = backup
4	LR1	40	Plant Loop 1 receive error: 0 = no, 1 = yes
	LR2	20	Plant Loop 2 receive error; 0 = no, 1 = yes
	LT1	10	Plant Loop 1 transmit error: 0 = no, 1 = yes
	LT2	08	Plant Loop 2 transmit error: 0 = no, 1 = yes
5	—	00	Reserved

G.6 PNI800 Plant Network Interface 800 Module

Table G-17 lists the lists the fields that make up the IPNI800 module status report. Table G-16 describes each field within the module status record.

Table G-15 PNI800 Status Bytes

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2			RIO	LIO				
3	CSP	MOV	LFF					
4								
5	Reserved							
6	ETYPE							
7	Reserved							
8	Reserved							
9					NET SWAP	NET MIS-MATCH	ELANA	ELANB
10	Reserved							
11	PSA	PSB						

Table G-15 PNI800 Status Bytes

Byte	Bit							
	7	6	5	4	3	2	1	0
12-13	Reserved							
14	Module nomenclature							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

Table G-16 PNI800 Status Bits

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 00 = configure, 01 = fail, 10 = error, 11 = execute
	TYPE	1F	Module type code: $(15)_{16}$ = enhanced status
2	RIO	20	Summary remote input status: 0 = good, 1 = bad
	LIO	10	Summary local input status: 0 = good, 1 = bad
3	CSP	80	Communication status problem: 0 = no, 1 = yes (Remote node is offline)
	MOV	40	Memory overflow: 0 = good, 1 = bad
	LFF	20	Licensing failure
4-5	Reserved		
6	ETYPE	20	Enhanced module type $(20)_{16}$ = SSM/CIU/ICT/IET
7-8	Reserved		
9	NET SWAP	08	Network LAN A/B Swapped: 0 = no, 1 = yes If this bit is a '1' / YES, PNI800 has detected one or more nodes on the Network as having their (PN800) LAN A/B Ethernet connections swapped (i.e. LAN A is connected to LAN B, LAN B is connected to LAN A.
	LANM	04	Network LAN A/B Mismatch: 0 = no, 1 = yes If this bit is a '1' / YES, PNI800 has detected that its (PN800) LAN A/B Ethernet connections are operating in different modes (i.e. A= 10 MHZ, B= 100 MHZ, or A= Full Duplex, B= Half Duplex).
	ELANA	02	Error on LAN A. No connection on RJ45 LAN A
	ELANB	01	Error on LAN B. No connection on RJ45 LAN B
10	Reserved		
11	PSA	80	Power Status A: 0 = OK, 1 = failed/bad
	PSB	40	Power Status B: 0 = OK, 1 = failed/bad
12-13	Reserved		

Table G-16 PNI800 Status Bits

Byte	Field	Field Size or Value	Description
14	—	05	Module nomenclature: $(0x5)_{16}$ = PNI800
15	—	FF	Revision letter (in Hex). For example, $(0x45)_{16}$ = E
16	—	FF	Revision number (in Hex). For example, $(0x30)_{16}$ = 0

G.7 CP800 Communications Processor Module (HPC800)

Table G-17 lists the lists the fields that make up the CP800 module status report. Table G-18 describes each field within the module status record.

The following table (Table G-17) lists the fields that make up the CP800 module status report:

Table G-17 CP800 Status Byte Description

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	Reserved							
3	CSP	MOV		BKCFG	BKSTS			PSI
4								
5	Reserved							
6	ETYPE (0x25)							
7			R1F	R2F			HN800A	HN800A
8								
9	CW800A	CW800B	BAT LOW		NET SWAP	NET MISMATCH	NET IDLE 1	NET IDLE2
10	Reserved							
11	PSA	PSB						
12 - 13	Reserved							
14	Module nomenclature (0x1E for CP800)							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

The following table (Table G-18) describes each field within the module status record:

Table G-18 CP800 Status Bit Descriptions

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 10 = error, 11 = execute
	TYPE	1F	Module type: 0x15 = enhanced status (ETYPE)

Table G-18 CP800 Status Bit Descriptions (Continued)

Byte	Field	Field Size or Value	Description
2	—	00	Reserved
3	CSP	80	Communication status problem: 0 = no, 1 = yes
	MOV	40	Memory overflow: 0 = good, 1 = bad
	BKCFG	10	Redundant configuration: 0 = no, 1 = yes
	BKSTS	08	Backup failed: 0 = no, 1 = yes
	PSI	00	Primary/backup indicator: 0 = primary, 1 = backup
4 & 5	—	00	Reserved
6	ETYPE	20	Enhanced module type: 0x25 = CP800
7	R1F	20	Redundancy link channel 1 failure: 0 = good, 1 = fail
	R2F	10	Redundancy link channel 2 failure: 0 = good, 1 = fail
	HN800A	02	HN800 channel A failure: 0 = no, 1 = yes
	HN800B	01	HN800 channel B failure: 0 = no, 1 = yes
8	—	00	Reserved
9	CW800A	80	CW800 Bus A failure: 0 = good, 1 = fail
	CW800B	40	CW800 Bus B failure: 0 = good, 1 = fail
	BAT LOW	20	NVRAM Battery low: 0 = good, 1 = replace battery
	NET SWAP	08	Network LAN A/B Swapped: 0 = no, 1 = yes If this bit is a ‘1’ / YES, the CP800 detects one or more nodes on the Network having their (PN800) LAN A/B Ethernet connections swapped (i.e. LAN A is connected to LAN B, LAN B is connected to LAN A.
	NET MIS-MATCH	04	Network LAN A/B Mismatch: 0 = no, 1 = yes If this bit is a ‘1’ / YES, then CP800 detedts that its (PN800) LAN A/B Ethernet connections are operating in different modes (i.e. A= 10MHZ, B= 100MHZ, or A= Full duplex, B= Half Duplex)
	NET IDLE1	02	Error on PN800 LAN A. No connection on RJ45 for PN800 LAN A (labeled EN2 A on the MB810 base)
	NET IDLE2	01	Error on PN800 LAN B. No connection on RJ45 for PN800 LAN B (labeled EN2 B on the MB810 base)
10	—	—	Reserved
11	PSA	80	Power Status A: 0 = OK, 1 = failed/bad
	PSB	40	Power Status B: 0 = OK, 1 = failed/bad
12-13			Reserved
14	—	FF	Module nomenclature: 0x1E = CP800
15	—	FF	Revision letter (in ASCII code). For example, 0x41 = A
16	—	FF	Revision number (in ASCII code). For example, 0x30 = 0

G.8 SPIPT800 INFI-NET to PN800 Transfer Module

Table G-19 lists the lists the fields that make up the SPITP800 module status report. Table G-20 describes each field within the module status record.

The following table (Table G-19) lists the fields that make up the IPT800 module status report:

Table G-19 SPIPT800 Status Byte Description

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2			SIO	CIO				
3	PCUs	MOV						
4	NSF	RER1	RER2	BKCFG	BKSTS	RINGS		PSI
5	Reserved							
6	ETYPE							
7 - 8	Reserved							
9	CW800A	CW800B	BAT LOW		NET SWAP	NET MISMATCH	NET IDLE 1	NET IDLE2
10	Reserved							
11	PSA	PSB						
12 - 13	Reserved							
14	Module nomenclature (0x4)							
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

The following table (Table G-20) describes each field within the module status record:

Table G-20 SPIPT800 Status Bit Descriptions

Byte	Field	Field Size or Value	Description
1	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 10 = error, 11 = execute
	TYPE	1F	Module type: 0x4 = enhanced status (ETYPE)
2	SIO	20	I/O status for data sourced by satellite ring PCUs: 0 = good, 1 = bad
	CIO	10	I/O status for data sourced by central ring PCUs: 0 = good, 1 = bad
3	PCUs	80	Nodes off-line: 0 = no, 1 = yes
	MOV	40	Memory overflow: 0 = no, 1 = yes

Table G-20 SPIPT800 Status Bit Descriptions (Continued)

Byte	Field	Field Size or Value	Description
4	NSF	80	Node environment status flag: 0 = good, 1 = bad
	RER1	40	Receive errors on channel 1: 0 = no, 1 = yes
	RER2	20	Receive errors on channel 2: 0 = no, 1 = yes
	BKCFG	10	Redundant configuration: 0 = no, 1 = yes
	BKSTS	08	Backup failed: 0 = no, 1 = yes
	RINGS	04	Rings off-line: 0 = on-line, 1 = off-line bridge for remote rings
	PSI	01	Primary/backup indicator: 0 = primary, 1 = backup
5	Reserved		
6	ETYPE	20	Enhanced module type: 0x23= SPIPT800
7 - 8	Reserved		
9	CW800A	80	CW800 Bus A failure: 0 = good, 1 = fail
	CW800B	40	CW800 Bus B failure: 0 = good, 1 = fail
	BAT LOW	20	NVRAM Battery low: 0 = good, 1 = replace battery
	NET SWAP	08	Network LAN A/B Swapped: 0 = no, 1 = yes If this bit is a ‘1’ / YES, the SPIPT800 detects one or more nodes on the Network having their (PN800) LAN A/B Ethernet connections swapped (i.e. LAN A is connected to LAN B, LAN B is connected to LAN A.
	NET MIS-MATCH	04	Network LAN A/B Mismatch: 0 = no, 1 = yes If this bit is a ‘1’ / YES, then SPIPT800 detedts that its (PN800) LAN A/B Ethernet connections are operating in different modes (i.e. A= 10MHZ, B= 100MHZ, or A= Full duplex, B= Half Duplex)
	NET IDLE1	02	Error on PN800 LAN A. No connection on RJ45 for PN800 LAN A (labeled CH 0 on module front plate)
	NET IDLE2	01	Error on PN800 LAN B. No connection on RJ45 for PN800 LAN B (labeled CH 1 on module front plate)
10	—	—	Reserved
11	PSA	80	Power Status A: 0 = OK, 1 = failed/bad
	PSB	40	Power Status B: 0 = OK, 1 = failed/bad
12-13			Reserved
14	—	FF	Module nomenclature: 4 = SPIPT800
15	—	FF	Revision letter (in ASCII code). For example, 0x41 = A
16	—	FF	Revision number (in ASCII code). For example, 0x30 = 0

H. Point Quality Definition

H.1 General Description

This appendix defines the point quality flag states for analog I/O, digital I/O, and module bus I/O.

NOTE: The control module can be configured to trip or continue operation upon the loss of an I/O module.

H.2 Individual Analog Inputs

Each analog input is compared to a low value of -0.75 VDC and a high value of 5.25 VDC. If the input goes outside either limit, the value is held to the limit and the point quality flag indicates bad.

H.3 Group Analog Inputs

If an I/O module is not connected, all points in the group will have their point quality flag set to bad.

H.4 Group Analog Outputs

All outputs are read back by feedback input circuitry. If the feedback voltage signal does not match the requested software output value, the point quality is set to bad for that output.

H.5 Digital I/O

If the I/O definition block is configured to use an I/O module, but the I/O module is not installed, the point quality flag for the associated module I/O group will be set to bad, and the control module generates a problem report.

H.6 Peer-to-Peer and Module Bus I/O

If input data cannot be read for a period of 2.0 seconds (S2 of FC 90 = 0.250 sec.), the flag will be set to bad**.** A new attempt will be made to read the data at least every 20 seconds.

NOTE: After the point quality flag is set to indicate bad quality, the receiving module retains and uses the last good value that was obtained.

The Harmony controllers utilize a configurable retry period for determining bad quality. The base retry period is configured in the extended executive (function code 90) base periodic sample I/O period S2. This specification (S2) also determines the time resolution for all periodic sample I/O pole periods configured in the module. This includes the segment control (function code 82) periodic sample I/O period S13. All periodic sample I/O periods should be configured in multiples of function code 90, S2. To determine the typical time required to mark a periodic sample I/O point bad, multiply function code 90, S2 by eight.

Example:

function code 90, S2 = 0.250 sec.

0.250 sec. × 8.000 = 2.000 sec.

A point will not continue to retry at the function code 90, S2 period after being marked bad. The bad point retry is only performed once every 20 seconds.

I. Console Engineering Unit Descriptions

I.1 Engineering Unit Descriptions Tables

Table I-1 Human System Interface (Conductor VMS/NT)

Code	Description	Code	Description
0 and 1	Blank	9	CFS
2	%	10	CFM
3	DEG F	11	LB/HR
4	DEG C	12	GAL
5	PSIA	13	AMPS
6	PSIG	14	IN HG
7	IN H_2O	15	KLB/HR
8	GMP	16 - 255	Configurable

J. Symphony Plus HC800 Control Processor

J.1 Memory Utilization and Execution Times

This section lists the module memory requirements for each function code supported by the HC800 control processor. The HC800 is the component of the HPC800 controller in which the function codes configurations are stored and processed. Three quantities are given for the HC800 memory utilization:

- The number of bytes of nonvolatile random access memory (NVRAM). The HC800 module has 1.9 megabytes of NVRAM memory.
- The number of bytes of random access memory (RAM). The HC800 module's total RAM configuration memory is 16 megabytes.
- The checkpoint utilization byte size.

This section also lists the function code execution times (in microseconds) for the HC800.

NOTE: Except where otherwise noted, execution times are given for worst case conditions.

Table J-1 shows the HC800 memory requirements and the execution time for each supported function code.

NOTE: Refer to ***Memory Usage Equations*** in this section for the equations listed in Table J-1.

Table J-1 HC800 Memory Utilization and Execution Times

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
1	Function generator	46	108	12	2.80
2	Manual set constant	12	58	12	0.75
3	Lead/lag	18	88	22	5.00
4	Pulse positioner	24	180	56	3.00
5	Pulse rate	16	88	18	7.00
6	High/low limiter	16	64	12	1.00
7	Square root	14	64	12	7.26
8	Rate limiter	18	80	18	2.98
9	Analog transfer	20	92	22	5.60
10	High select	16	68	12	1.18
11	Low select	16	68	12	1.16
12	High/low compare	16	64	12	0.97
13	Integer transfer	14	60	10	0.76
14	Four input summer	16	68	12	2.14
15	Two input summer	18	68	12	2.22
16	Multiply	14	64	12	1.85
17	Divide	14	64	12	2.11
18	PID error input	34	124	26	19.15
19	PID process variable and set point	36	132	30	20.96
24	Adapt	12	66	12	1.25
25	Analog input (periodic sample)[5]	12	110	38	1.59

Table J-1 HC800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
26	Analog input/loop	12	72	18	1.33
30	Analog exception report	26	118	22	3.00
31	Test quality	16	68	10	2.33
32	Trip	12	54	10	0.93
33	Not	12	54	10	0.67
34	Memory	14	62	10	0.67
35	Timer	14	74	18	1.67
36	Qualified OR (eight input)	26	84	10	1.33
37	AND (two input)	12	56	10	0.67
38	AND (four input)	16	64	10	0.65
39	OR (two input)	12	56	10	0.56
40	OR (four input)	16	64	10	0.67
41	Digital input/bus (periodic sample)	N/A, see function code 206			
42	Digital input/loop	12	60	12	1.10
45	Digital exception report	12	82	16	2.76
48	Analog Exception Report with High/Low Alarm Deadband	32	126	22	3.00
50	Manual set switch	12	52	10	0.67
51	Manual set constant (nontunable)	12	58	12	0.67
52	Manual set integer	12	52	10	0.56
55	Hydraulic servo[5]	62	430	88	1,723
58	Time delay (analog)	18	94 or Equation 3	22	12.60
59	Digital transfer	14	60	10	0.73
61	Blink	12	64	14	1.71
62	Remote control memory	28	112	18	2.67
63	Analog input list (periodic sample)	N/A, see function code 205			
64	Digital input list (periodic sample)	N/A, see function code 206			
65	Digital sum with gain (four input)	28	84	12	1.74
66	Analog trend Normal mode (slow) Fast mode	 12 12	 212 212	 88 88	 1.67 2.33
67	Digital Exception Report with Alarm Deadband	16	102	22	2.00

Table J-1 HC800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
68	Remote manual set constant (REMSET)	22	144	32	2.17
69	Test alarm	12	60	12	0.67
79	Control interface slave[5] First FC 79 block in segment Other FC 79 blocks in segment	 50 50	 354 354	 68 68	 15,430 570
80	Control station	84	382	76	7.19
81	Executive block	12	3752	62	0
82	Segment control	64	316	56	0
83	Digital output group[5]	32	120	12	98.00
84	Digital input group[5]	16	144	44	97.00
85	Up/down counter	24	92	18	1.73
88	Digital logic station[5]	42	170	26	
89	Last block	12	66	16	0
90	Extended executive	38	64	12	0
93	BASIC real output	12	80	24	0.49
94	BASIC boolean output	12	88	28	0.45
95	Module status monitor	N/A, see function codes 207 and 226			
96	Redundant analog input	22	122	34	4.37
97	Redundant digital input	14	74	16	1.39
98	Slave select	26	100	14	1.73
100	Digital output readback check	40	160	26	3.55
101	Exclusive OR	12	56	10	0.63
102	Pulse input/period[5]	26	126	20	118.00
103	Pulse input/frequency[5]	26	126	20	124..00
104	Pulse input/totalization[5]	28	156	32	148.00
109	Pulse input/duration[5]	26	126	20	117.00
110	Five input rung	24	90	14	1.67
111	Ten input rung	38	120	14	2.16
112	Twenty input rung	68	180	14	3.33
114	BCD input[5]	20	98	18	82.00
115	BCD output[5]	22	98	12	109.00
116	Jump/master control relay	12	68	10	0.33
117	Boolean recipe table	28	84	10	1.40

Table J-1 HC800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
118	Real recipe table	48	118	12	1.60
119	Boolean signal multiplexer	32	102	10	1.30
120	Real signal multiplexer	32	106	12	1.39
121	Analog input/Cnet	22	98	20	1.98
122	Digital input/Cnet	20	66	12	2.78
123	Device driver	30	134	28	3.00
124	Sequence monitor	80	180	20	2.00
125	Device monitor	40	118	12	3.98
126	Real signal demultiplexer	12	108	24	1.30
128	Slave default definition[5]	44	98	12	1.00
129	Multistate device driver	60	184	34	3.90
132	Analog input/slave[5]	54	366	48	252.00
134	Multi-sequence monitor	96	284	36	0.89
135	Sequence manager	82	244	32	4.17
136	Remote motor control	52	202	46	6.55
137	BASIC real output/quality	12	96	32	0.56
138	BASIC boolean output/quality	12	80	24	0.49
139	Passive station interface	44	162	32	3.00
140	Restore Largest NVM utilization (FC 165 with S2 = 249) Smallest NVM utilization (FC 33)	Refer to formula in Function Code 140	104 104	12 12	1,600.38 25.93
141	Sequence master	80	158	24	1.77
142	Sequence slave	74	120	10	0.66
143	Invoke C	28	124	22	Program dependent
144	C allocation	Equation 4	Equation 5	Equation 6	1.00
145	Frequency counter/slave[5]	32	146	22	45.00
146	Remote I/O interface[5]	68	Equation 30	56	700
147	Remote I/O definition[5]	144	588	108	10.00
148	Batch sequence	36	Equation 7	Equation 8	Program dependent
149	Analog output/slave[5] First FC 149 in segment Second FC 149 in segment (same ASO as first) Additional FC 149s in segment	 84 84 84	 344 344 344	 52 52 52	 16,016 16,015 702

Table J-1 HC800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
151	Text selector	46	168	22	1.82
152	Model parameter estimator	28	368	26	3.33
153	Inferential smith controller parameter converter	42	212	30	10.00
154	Adaptive parameter scheduler	36	240	28	5.00
155	Regression	68	Equation 9	Equation 10	4.26
156	Advanced PID controller	58	206	48	27.88
157	General digital controller	68	Equation 11	Equation 12	3.21
160	Smith predictor	36	358	122	10.48
161	Sequence generator	74	228	46	2.53
162	Digital segment buffer	16	94	24	1.26
163	Analog segment buffer	16	110	32	1.52
165	Moving average	16	Equation 13	Equation 14	5.77
166	Integrator	28	108	24	5.26
167	Polynomial Most complicated All zero	 58 58	 124 124	 12 12	 37.84 26.85
168	Interpolator	98	212	14	5.71
169	Matrix addition	44	188	44	3.97
170	Matrix multiplication	44	188	44	23.86
171	Trigonometric Sine Secant	 14 14	 64 64	 12 12	 7.38 8.10
172	Exponential	12	60	12	7.22
173	Power	14	64	12	11.50
174	Logarithm	14	66	12	5.00
177	Data acquisition analog	104	414	90	10.04
178	Data acquisition analog input/loop	20	150	44	3.57
179	Enhanced trend Normal sampling Normal and statistical sampling	 52 52	 FDB + FOB 180 + 2 x Equation 15	 Equation 15 Equation 15	 5.68 10.92
190	User defined function declaration	30	Equation 16	12	1.00
191	User defined function one	56	532 + S16	94+ S16	Program dependent
192	User defined function two	70	616 + S24	110 + S24	Program dependent

Table J-1 HC800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
193	User defined data import	48	Equation 19	70 + S5	1.19
194	User defined data export	48	Equation 20	58 + S4	4.38
198	Aux. real user defined function	38	166	24	0.71
199	Aux. digital user defined function	38	150	16	0.67
205	Analog Input List/CW800 (Periodic Sample)	30	302	110	9.92
206	Digital Input List/CW800 (Periodic Sample)	30	206	62	0.43
207	Module Status Monitor/CW800	16	147	48	2.22
211	Data acquisition digital	62	326	78	6.30
212	Data acquisition digital input/loop	20	126	32	1.98
215	Enhanced analog slave definition[5]	32	232	26	558.00
216	Enhanced analog input definition[5]	42	130	16	218.00
217	Enhanced calibration command[5]	24	194	40	9.00
218	Phase execution	44	Equation 22	304	Program dependent
219	Common sequence	46	Equation 17	Equation 18	Program dependent
220	Batch historian	46	688 + S11	222 + S11	Program dependent
221	I/O device definition	Equation 23	1524	64	Note 1
222	Analog in/channel	Equation 24	332	64	35.80[2]
223	Analog out/channel	Equation 25	332	64	49.65[2]
224	Digital in/channel	Equation 26	332	64	24.00[2]
225	Digital out/channel	Equation 27	332	64	16.00[2]
226	Test status	Equation 21	148	16	3.06
227	Gateway	Equation 31	20556	76	41,194
228	Foreign device definition	Equation 28	404	120	21.00
229	Pulse In/Channel	Equation 29	480	84	126.00

Table J-1 HC800 Memory Utilization and Execution Times (Continued)

Function Code	Description	NVRAM (Bytes)	RAM (Bytes)	Checkpoint Utilization Byte Size	Execution Times (in µsecs)
241	DSOE interface SEM to MFP No SED resynch performed SED resynch performed (1/sec.)	 18 18	 340 340	 26 26	126.00
247	Condition monitoring[4,5]	134	592	82	Note 4

NOTES:

1. Time dependent on the I/O block type configured and on the presence or absence of redundant blocks.
2. With exception reports enabled.
3. Time for the maximum configuration.
4. Time dependent on the turbine instrumentation module type. Refer to the applicable product instruction manual.
5. These function codes are for use with remote IO applications only. They may be used only when accessing Harmony Rack IO modules via an RIO22 connected to an HPC800 using function codes 146 and 147.

J.2 Memory Usage Equations

1. 485 + S2 + [16 x (S8)]
2. 400,174 + [16 x (S8)]
3. 72 + [8 x (S5)]
4. 12 + [1024 x (S2)]
5. 42 + [1024 x (S1)]
6. 10 + [1024 x (S1)]
7. 676 + [1024 x (S11)] + [a x (S12)]

   where:

   a = 2 (if S12 is positive)
   - or -
   a = 2048 (if S12 is negative)
8. 248 + [b x (S12)]

   where:

   b = 1 (if S12 is positive)
   - or -
   b = 1024 (if S12 is negative)
9. 294 + [20 x (S7)]
10. 60 + [20 x (S7)]
11. 144 + [8 x (S21 + S22)]
12. 60 + [4 x (S21 + S22)]
13. 79 + [4 x (S2)]
14. 34 + [4 x (S2)]
15. a + [b x (S4)] + [c x (n{S3})],

    where:

    n{S3} = Number of modes selected for S3
    if S2 = 0, 10, or 11, then a = 106, b = 12, and c = 8
    if S2 = 1, 2, 5, 6, 7, 9, 12, or 13, then a = 100, b = 10, and c = 4
    if S2 = 4 or 8, then a = 128, b = 25, and c = 32
16. 82 + [10 x (S2)]
17. 490 + [292 x (S13)]
18. 378 + [142 x (S13)]

19. 214 + [2 x (S5)]
20. 202 + [4 x (S4)]
21. 27 + Size of string data in S2 + S3 + S4 + S5 + S10.
22. 1458 + [1024 x (S11)] + [2 x (S12)]
23. 90 + Size of string data in S1 + Size of string data in S30.
24. 92 + Size of string data in S1+ Size of string data in S23.
25. 78 + Size of string data in S1+ Size of string data in S18.
26. 52 + Size of string data in S1+ Size of string data in S15.
27. 48 + Size of string data in S1+ Size of string data in S12.
28. 48 + Size of string data in S1 + Size of string data in S2 + Size of string data in S7
29. 108 + Size of string data in S1 + Size of string data in S23
30. 1008 + (16 x (S4))
31. 48 + Size of string data in S1+ Size of string data in S2 + Size of string data in S24.

J.3 Function Blocks - HC800

Table J-2 contains function code block number information for the HC800 module.

Table J-2 HC800 Module

Block No.	Definition	Function Code
0	Logic 0	81
1	Logic 1	
2	0 or 0.0	
3	-100.0	
4	-1.0	
5	0.0	
6	1.0	
7	100.0	
8	-9.2 E18	
9	9.2 E18	
10	Startup flag (0 = no, 1 = yes)	
11	Memory display value	
12	System free time in percent	
13	Revision level	
14	Reserved	
15	Task 1 elapsed time since previous cycle	82
16	Task 1 elapsed time current cycle (sec/min)	
17	Task 1 processor utilization	
18	Task 1 check point overrun count	
19	Task 1 cycle time overrun (sec/min)	

Table J-2 HC800 Module (Continued)

Block No.	Definition	Function Code
20	Hours, time of day	90
21	Minutes, time of day	
22	Seconds, time of day	
23	No time synchronization flag: 0 = time of day invalid 1 = time of day valid	
24	Year (0 to 99)	
25	Month (1 to 12)	
26	Day (1 to 31)	
27	Day of week (1 to 7, Sunday = 1)	90
28	Reserved	
29	Reserved	
30	Configurable blocks	Any allowed function code - refer to Table J-1
31999	Loop type: 1.0 = PN800 3.0 = PN800 with time-stamping	89

NOTE:

J.4 Module Status Information - HC800

Tables J-3 and J-4 explain Symphony Plus HC800 controller status bytes.

Table J-3 Byte and Bit Description - HC800

Byte	Bit							
	7	6	5	4	3	2	1	0
1	ES	MODE		TYPE				
2	FTX	BAC	RIO	LIO	CFG	NVF	NVI	DSS
3	Error code							
4	Error code descriptor (1)							
5	Error code descriptor (2)							
6	ETYPE							
7			R1F	R2F	Reserved	Reserved	HN800A	HN800B
8	Reserved							
9	CW800A	CW800B	BATLOW	Reserved	RIOID	ETHER	EP1	EP2
10	PRI	CFC	Reserved	CHK	RID	RDEXP	OCE	RDDET
11	PSA	PSB	Reserved	SOA	RNO	Reserved	Reserved	Reserved
12-13	Reserved							
14	Module nomenclature							

Table J-3 Byte and Bit Description - HC800 (Continued)

Byte	Bit							
	7	6	5	4	3	2	1	0
15	Revision letter (ASCII)							
16	Revision number (ASCII)							

Table J-4 Byte Description - HC800

Byte	Field	Field Size or Value	Description
1[2]	ES	80	Error summary: 0 = good, 1 = errors
	MODE	60	Module mode: 00 = configure, 10 = error, 11 = execute
	TYPE	1F	Module type code: $(15)_{16}$ = Enhanced status
2	FTX	80	First time in execute: 0 = no, 1 = yes
	BAC	40	Backup status: 0 = good, 1 = bad
	RIO	20	Summary remote input status: 0 = good, 1 = bad
	LIO	10	Summary local input status: 0 = good, 1 = bad
	CFG	08	Online configuration changes being made
	NVF	04	Summary NVRAM failure status: 0 = good, 1 = fail
	NVI	02	Summary NVRAM initialized state: 0 = no, 1 = yes
	DSS	01	Digital station status: 0 = good, 1 = bad

Table J-4 Byte Description - HC800 (Continued)

Byte	Field	Field Size or Value	Description
3 - 5 Note 1	Error Code	3 4 5	
		01 01 — 02 — 03 — FF —	NVRAM error: Write failure Checksum failure Bad data Reset during write
		02 (1) (2)	Analog input reference error: (1), (2) = block number of control interface I/O module block
		03 (1) (2)	Missing I/O module or expander board: (1), (2) = block number of I/O module or station
		05 (1) (2)	Configuration error – undefined block: (1), (2) = block number making reference
		06 (1) (2)	Configuration error – input data type is incorrect: (1), (2) = block number making reference
		08 (1) (2)	Trip block activated: (1), (2) = block number of trip block
		09 — —	Segment violation.
		0F — —	Primary module has failed and the redundant module configuration is not current
		10 — —	Primary module has failed and the dynamic RAM data in the redundant module is not current
		09 — —	Segment violation - priority set the same in two segments or more than eight segments defined.
		11 — —	NVRAM write failure error
		1E (1) (2)	Duplicate device definition label – multiple function code 221 function blocks contain the same device label. (1), (2) = block number making reference.

Table J-4 Byte Description - HC800 (Continued)

Byte	Field	Field Size or Value	Description
3 - 5 *(cont)*	Error Code	2A (1) (2)	Not enough memory for UDF: (1), (2) = block number making reference
		20 — —	Program format error - inconsistent format table
		21 00 00 FF FE FF FF (1) (2)	File system error: Backup cannot take over due to uninitialized file system. Directory has not been configured. List of file system free memory is corrupted. (1), (2) = Number of files with errors.
		22 (1) (2)	Invoke C error: (1), (2) = block number making reference
		24 (1) (2)	C program stack overflow: (1), (2) = block number making reference
		28 (1) (2)	User defined function (UDF) reference is invalid: (1), (2) = block number making reference
		29 (1) (2)	UDF block cannot read program file: (1), (2) = block number making reference
		2B (1) (2)	Missing UDF declaration: (1), (2) = block number making reference
		2C (1) (2)	Wrong UDF type: (1), (2) = block number making reference
		2D (1) (2)	Missing UDF auxiliary block: (1), (2) = block number making reference
		2E (1) (2)	UDF compiler and firmware are incompatible: (1), (2) = block number making reference
6[2]	ETYPE	1F	Enhanced module type = $(24)_{16}$
7	—	—	Unused
	—	—	Unused
	R1F	20	Redundancy link channel 1 failure: 0 = good, 1 = fail
	R2F	10	Redundancy link channel 2 failure: 0 = good, 1 = fail
	HN800A	02	HN800 channel A failure: 0 = good, 1 = fail
	HN800B	01	HN800 channel B failure: 0 = good, 1 = fail
8	—	—	Unused
9	CW800A	80	CW800 Bus A failure: 0 = good, 1 = fail
	CW800B	40	CW800 Bus B failure: 0 = good, 1 = fail
	BATLOW	20	NVRAM Battery low: 0 = good, 1 = replace battery
	RIOID	08	Remote I/O Controller: 0 = no, 1 = yes
	ETHER	04	Ethernet Installed: 0 = no, 1 = yes
	EP1	02	The SOE Time Synchronization Ethernet Port (labeled EN1 A on the MB810 base): failure: 0 = good, 1 = fail
	EP2	01	The Foreign Device Interface Ethernet Port (labeled EN1 B on the MB810 base): 0 = good, 1 = fail

Table J-4 Byte Description - HC800 (Continued)

Byte	Field	Field Size or Value	Description
10	PRI	80	Module is primary versus backup; set to 1 in the primary module.
	CFC	40	Configuration current (latched until backup is reset). Set when LED 7 is enabled (1 = on or blinking) on the backup module.
	CHK	10	Backup has completed checkpointing (latched until backup is reset). Always set to 0 on the primary module. Follows LED 8 (1 = on or blinking) on the backup module.
	RID	08	Redundancy ID. Follows setting of redundancy ID pole on the dipswitch.
	RDEXP	04	Redundancy expected. Always set to 1 on the backup module. Follows state of function code 90, specification S3, ones digit on the primary module.
	OCE	02	Online configuration is enabled. Follows setting of online configuration enable pole on dipswitch.
	RDDET	01	Redundancy detected (latched until module is reset or it changes from backup to primary or primary to backup). Set to 1 when a properly configured redundant module is detected.
11	PSA	80	Power Status A: 0 = OK, 1 = failed/bad
	PSB	40	Power Status B: 0 = OK, 1 = failed/bad
	SOA	10	Status output alarm. Indicates the status of the system +24 volt power and the I/O block's power (logic and field power for a single cabinet). 0 = OK, 1 = alarm.
	RNO	08	Redundancy NVM overrun (latched indication). Set to 1 in primary module if NVM checkpoint overruns have occurred. NVM checkpoint overruns cause the primary module to reset the backup module.
12-13	—	00	Reserved
14[2]	—	FF	Module nomenclature number $(0A)_{16}$ =10 for the HC800
15[2]	—	FF	Revision letter (in ASCII code), for example, $(41)_{16}$ = A
16[2]	—	FF	Revision number (in ASCII code), for example, $(30)_{16}$ = 0

NOTE:

1. Byte 3 is displayed on the front panel LEDs when the module is in ERROR mode.
2. $(x)_{16}$ denotes a hexidecimal number. The equivalent decimal or ASCII character is to be used. Hex to ASCII Code conversion charts are readily available on the Internet.

ABB Ltd.
Business Unit Power Generation
P.O. BOX 8131
8050 Zürich
SWITZERLAND
Phone: +41 (0) 43 317-5380
Fax: +41 (0) 43 317-5382
www.abb.com/powergeneration

Notice
The information in this document is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this document.
In no event shall ABB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.
This document and parts thereof must not be reproduced or copied without written permission from ABB, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.
The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.
This product meets the requirements specified in EMC Directive 2004/108/EC and in Low Voltage Directive 2006/95/EC.

Copyright © July 2013 by ABB.
All rights reserved.
Release: July 2013

Document Number: 2VAA000844R0001 Vol. 1 Rev D