

ENHANCED INSTRUCTIONS

Bit Field Distribute: BTD

File Arithmetic and Logic: FAL

Copy File (**COP**) Synchronous And Copy File (**CPS**)

Digital Alarm Instruction:ALMD

Analog Alarm Instruction: ALMA

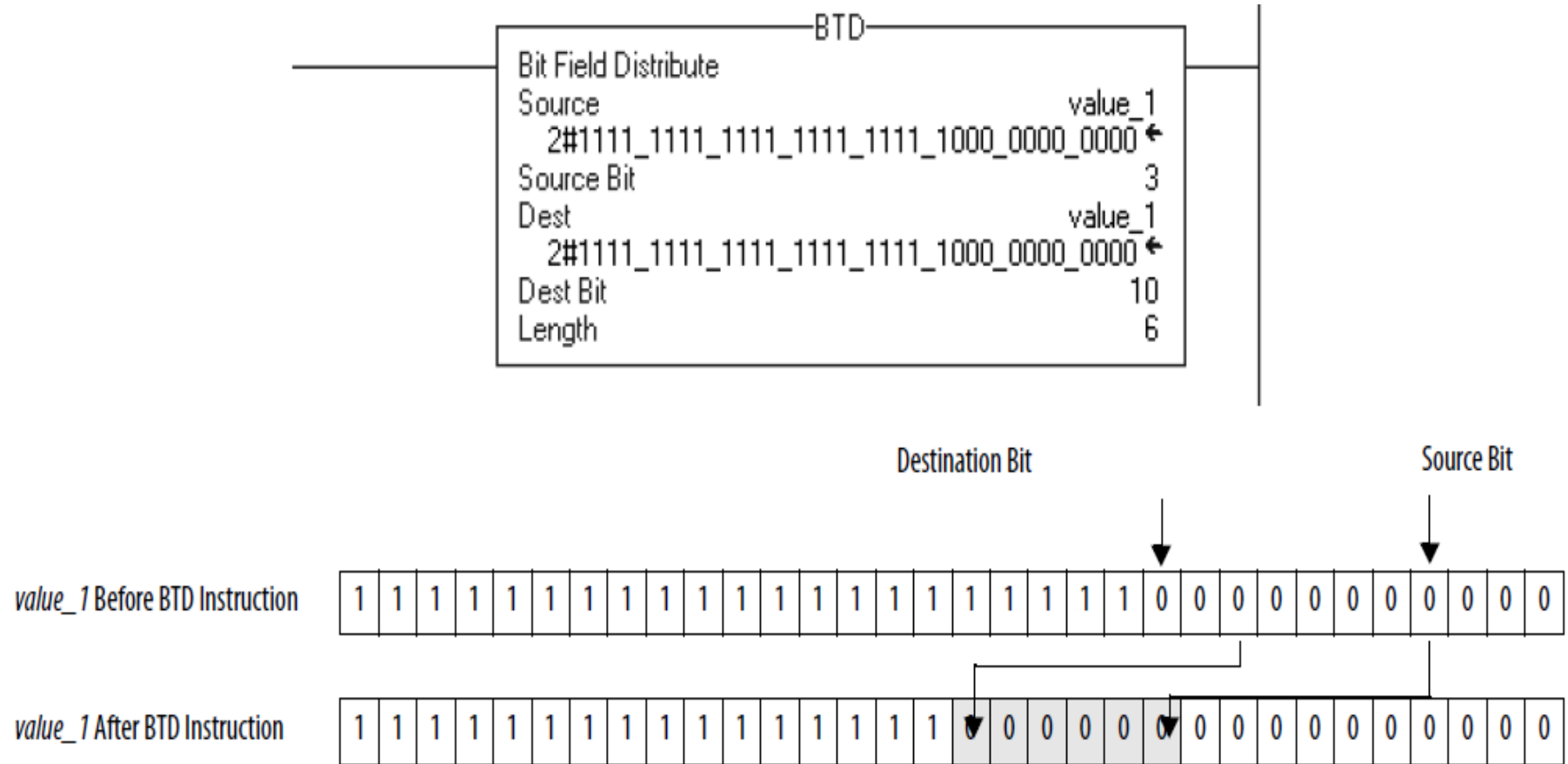
MSG, GSV, SSV Instructions.

Minor And MaJor Fault

MOV_LOGICAL INSTRUCTIONS

Bit Field Distribute: BTD

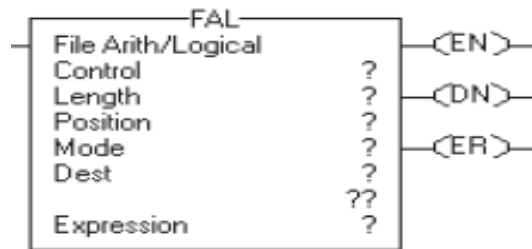
BTD copies specified bits from the source, shifts the bits to appropriate position and write the bits into Destination



ARRAY FILE INSTRUCTIONS

File Arithmetic and Logic: FAL

FAL performs copy, arithmetic, logic and function operations on data stored in an array.



Operand	Type	Format	Description
Control	CONTROL	Tag	Control structure for the operation
Length	DINT	Immediate	Number of elements in the array to be manipulated
Position	DINT	Immediate	Current element in array Initial value is typically 0
Mode	DINT	Immediate	How to distribute the operation Select INC, ALL, or enter a number
Destination	SINT INT DINT REAL	Tag	Tag to store the result
Expression	SINT INT DINT REAL	Immediate Tag	An expression consisting of tags and/or immediate values separated by operators
A SINT or INT tag converts to a DINT value by sign-extension.			

Selection Mode of operations.

If You Want To	Select This Mode
Operate on all of the specified elements in an array before continuing on to the next instruction	All
Distribute array operation over a number of scans Enter the number of elements to operate on per scan (1-2147483647)	Numerical
Manipulate one element of the array each time the rung-condition-in goes from false to true	Incremental

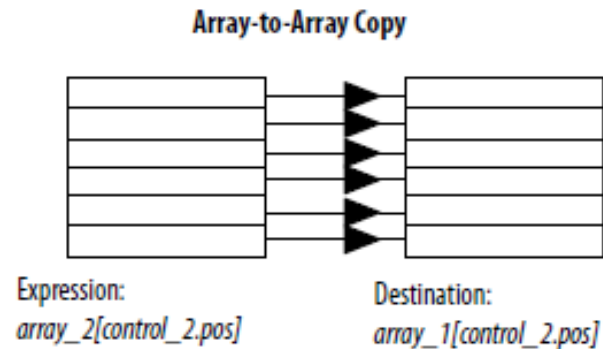
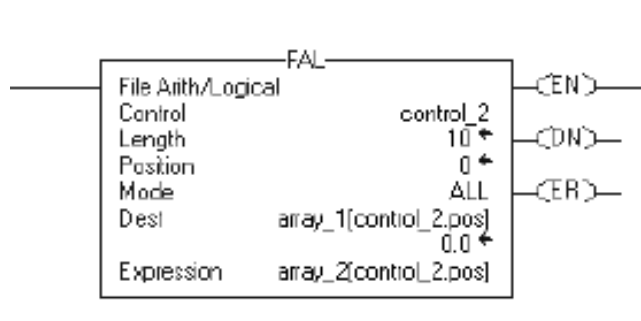
Structured Text

FOR position = 0 TO length DO
destination[position] := numeric_expression;
END_FOR;

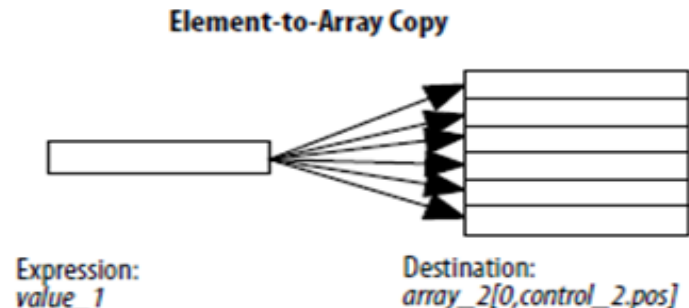
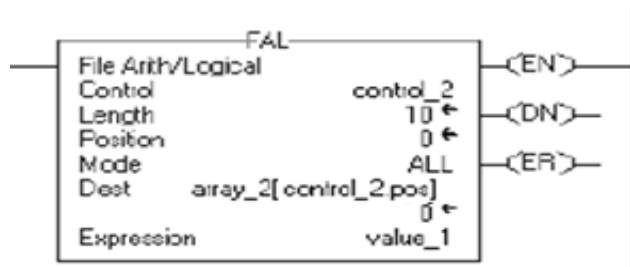
ARRAY FILE INSTRUCTIONS

FAL Examples

When enabled, FAL copies each element of array 2 into the same position within array 1

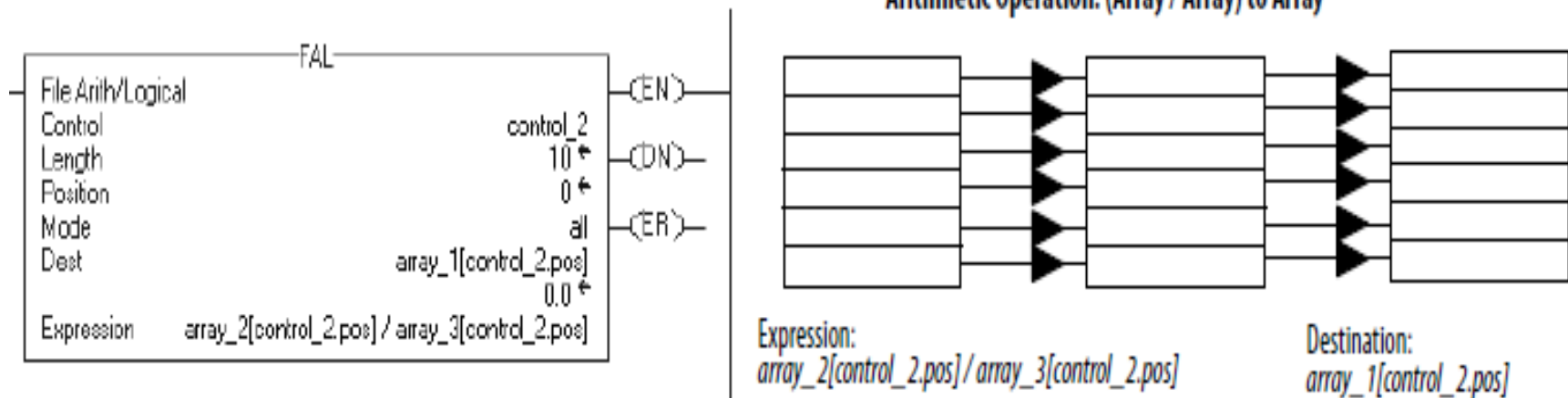


When enabled, FAL copies value_1 into the first 10 positions of array_2.



ARRAY FILE INSTRUCTIONS

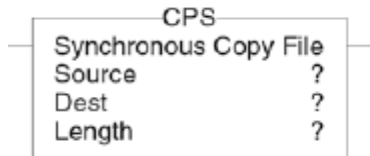
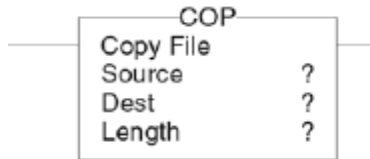
When enabled, FAL dives the value in the current position of array_2 with the value in the current position of array_3 and stores the result in the current position of array_1.



ARRAY FILE INSTRUCTIONS

Copy File (COP) Synchronous And Copy File (CPS)

The COP and CPS copy the value(s) in the Source to the Destination.
The Source remains unchanged



Operand	Type	Format	Description
Source	SINT INT DINT REAL string structure	Tag	Initial element to copy. Important: the Source and Destination operands should be the same data type, or unexpected results may occur.
Destination	SINT INT DINT REAL string structure	Tag	Initial element to be overwritten by the Source Important: the Source and Destination operands should be the same data type, or unexpected results may occur.
Length	DINT	Immediate Tag	Number of Destination elements to copy.

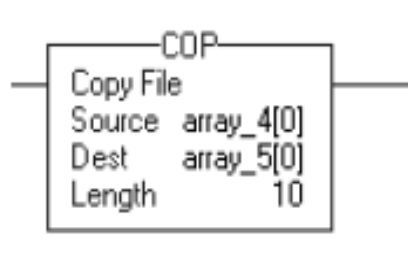
Difference between COP and CPS

If the source or destination is	And you want to	Then select	Notes
- Produced tag - Consumed tag - I/O data - Data that another task can overwrite	Prevent the data from changing during the copy operation	CPS	Tasks that attempt to interrupt a CPS instruction are delayed until the instruction is done.
	Allow the data to change during the copy operation	COP	
None of the above	—————→	COP	

ARRAY FILE INSTRUCTIONS

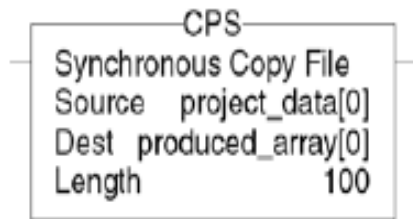
COP and CPS Examples

When enabled, COP copies the first 10 elements of array_4 into the first 10 elements of array_5



ST: COP(array_4[0],array_5[0],10);

When enabled, CPS copies 100 elements of project_data[0] into the 100 elements of produced_array[0]. No I/O Updates or other Tasks can change the data

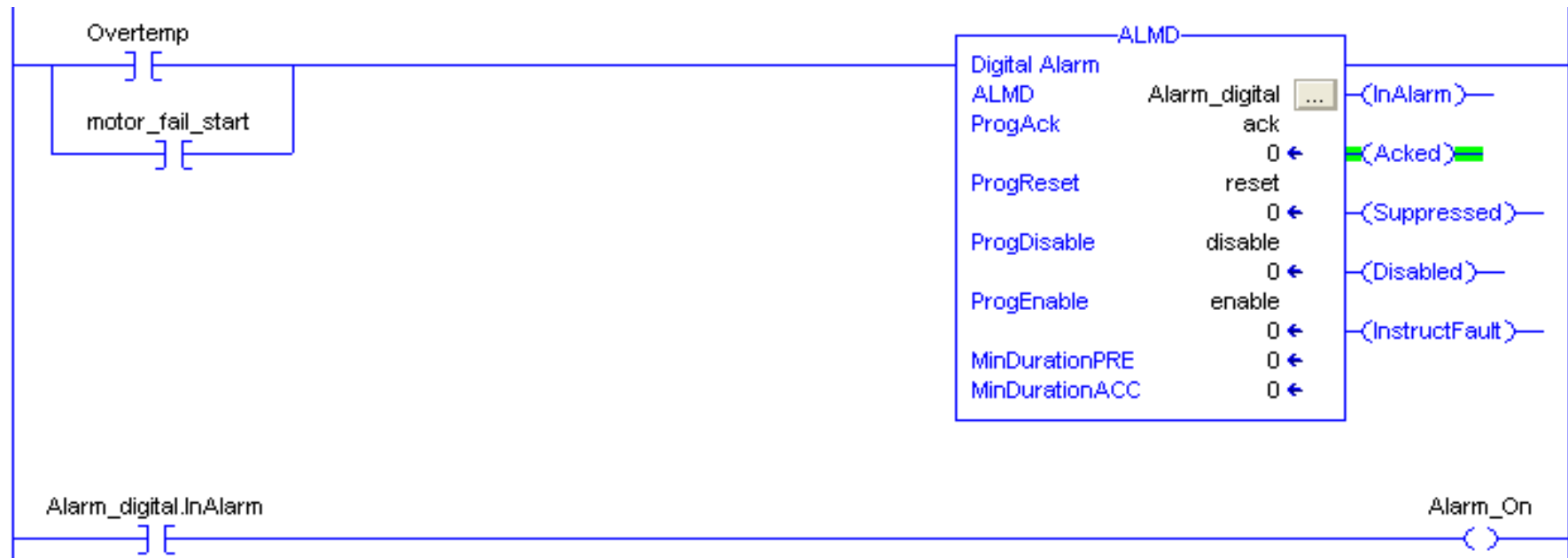


ST:CPS(project_data[0],produced_array[0],100);

DIGITAL ALARM INSTRUCTION

Digital Alarm Instruction:ALMD

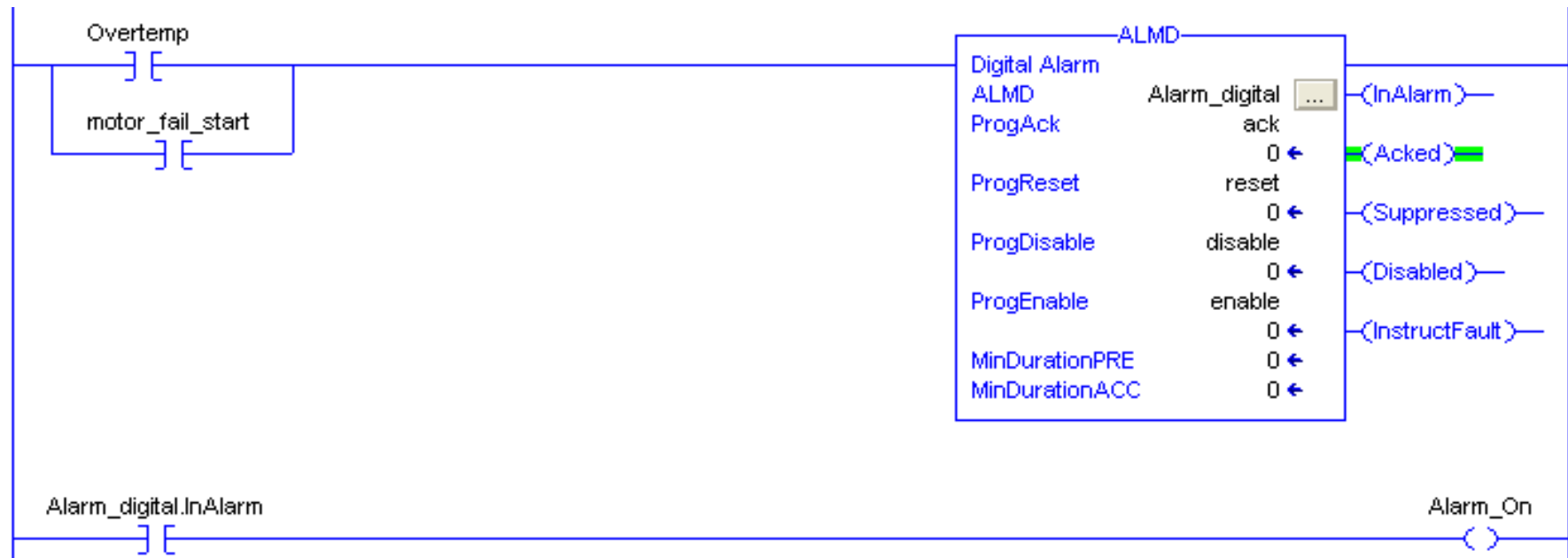
Scope: MainProgram Show... Show All					
	Name	Alias For	Base Tag	Data Type	Style
	ack			BOOL	Decimal
	Alarm_On			BOOL	Decimal
	disable			BOOL	Decimal
	enable			BOOL	Decimal
	motor_fail_start			BOOL	Decimal
	Overtemp			BOOL	Decimal
	reset			BOOL	Decimal
	Alarm_digital			ALARM_DIGITAL	



DIGITAL ALARM INSTRUCTION

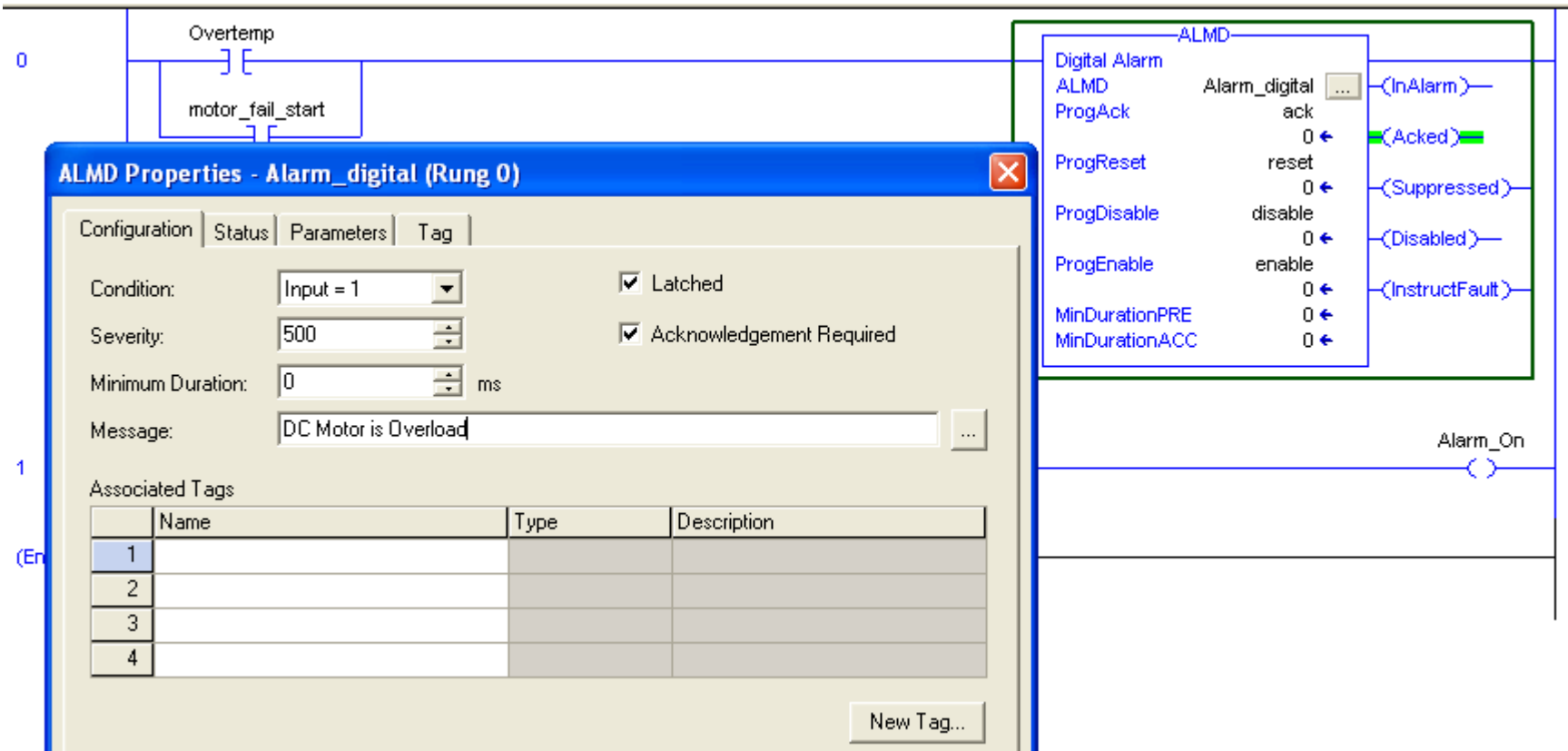
Digital Alarm Instruction:ALMD

Scope: MainProgram Show... Show All					
	Name	Alias For	Base Tag	Data Type	Style
	ack			BOOL	Decimal
	Alarm_On			BOOL	Decimal
	disable			BOOL	Decimal
	enable			BOOL	Decimal
	motor_fail_start			BOOL	Decimal
	Overtemp			BOOL	Decimal
	reset			BOOL	Decimal
	Alarm_digital			ALARM_DIGITAL	



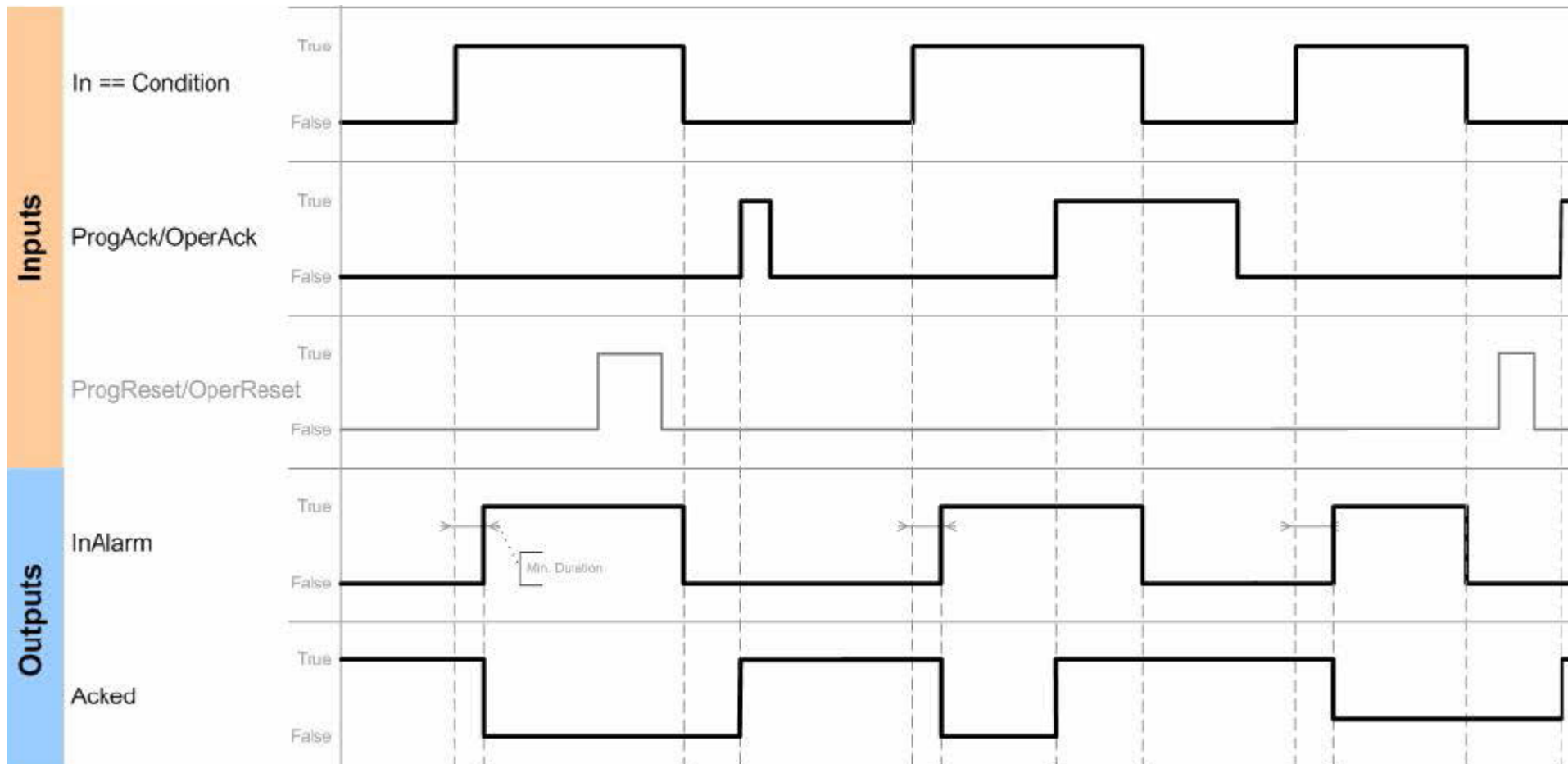
DIGITAL ALARM INSTRUCTION

ALMD:Creating Message to display in Factory Talk View



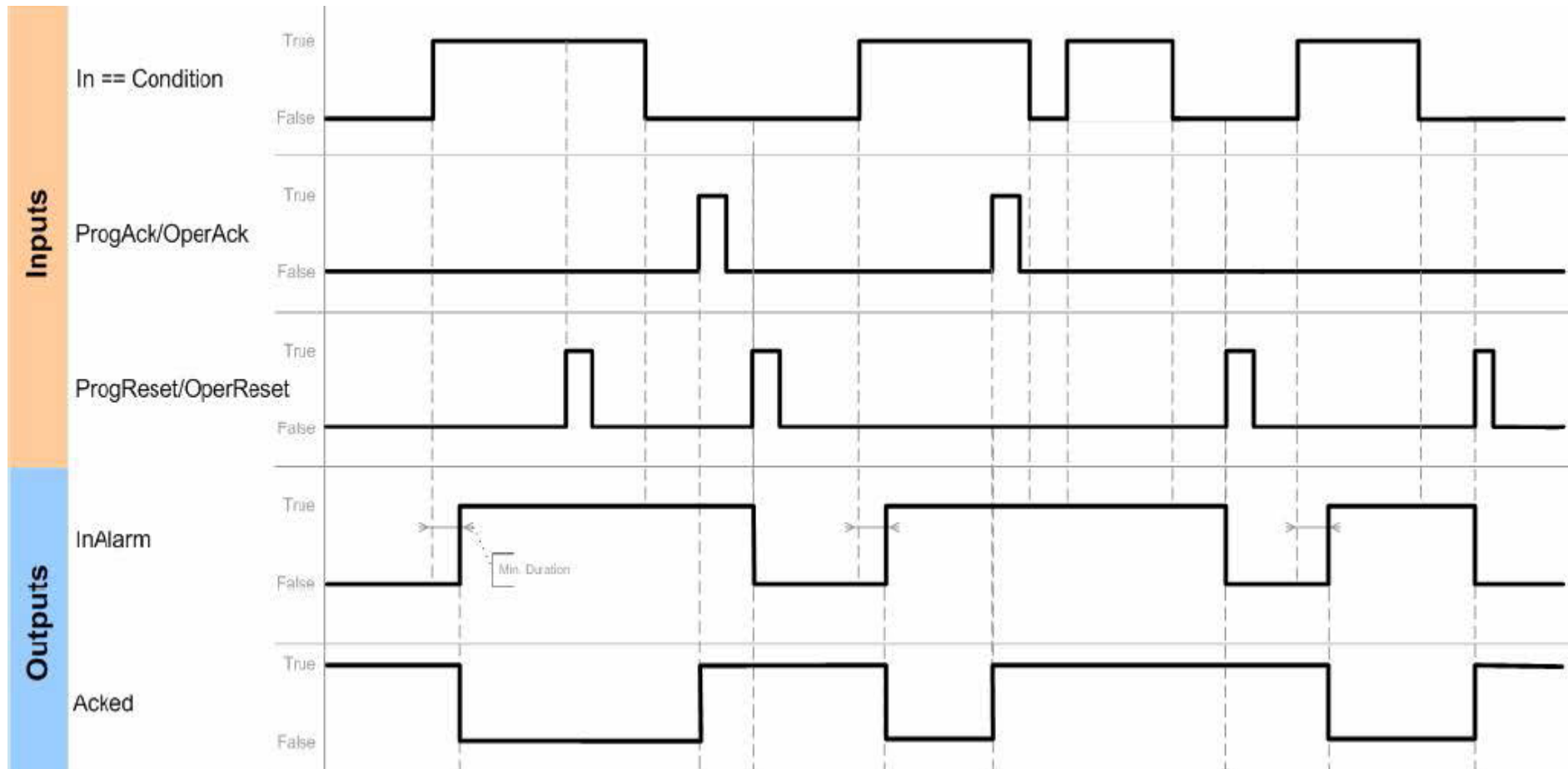
DIGITAL ALARM INSTRUCTION

ALMD Alarm Acknowledge Required and Not Latch



DIGITAL ALARM INSTRUCTION

ALMD Alarm Acknowledge Required and Latched

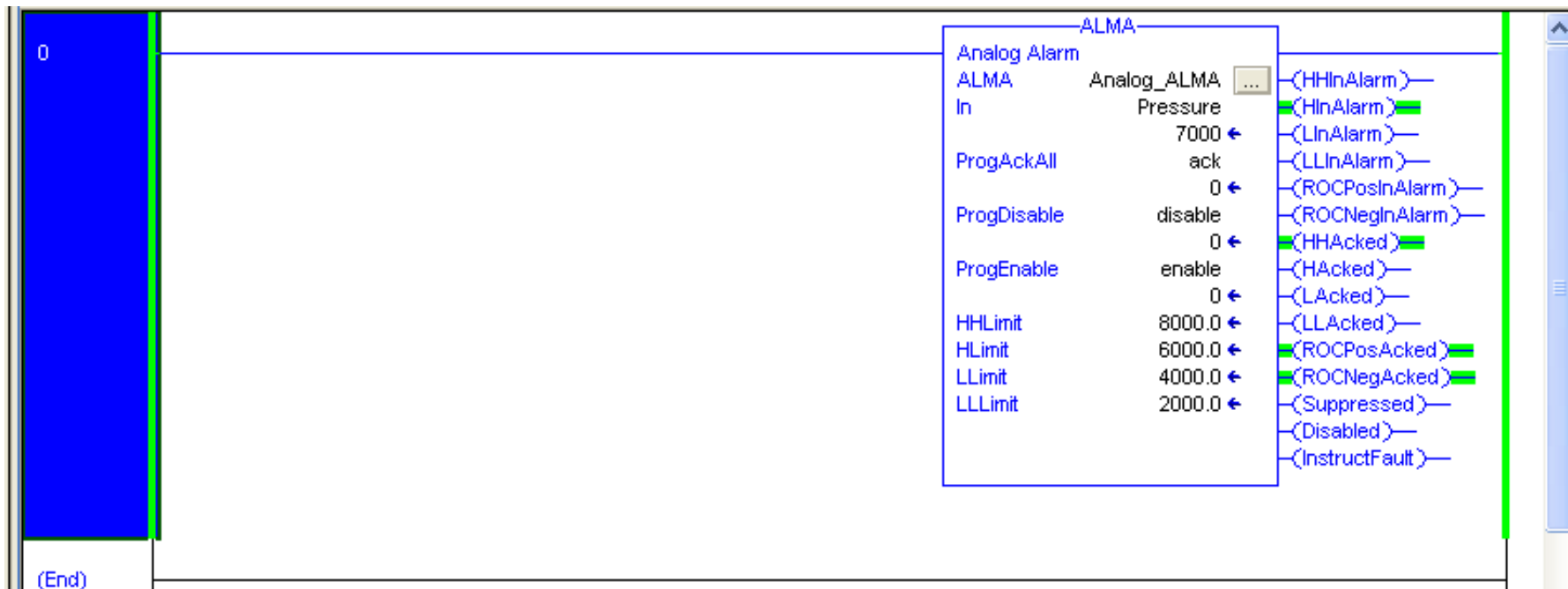


ANALOG ALARM INSTRUCTION: ALMA

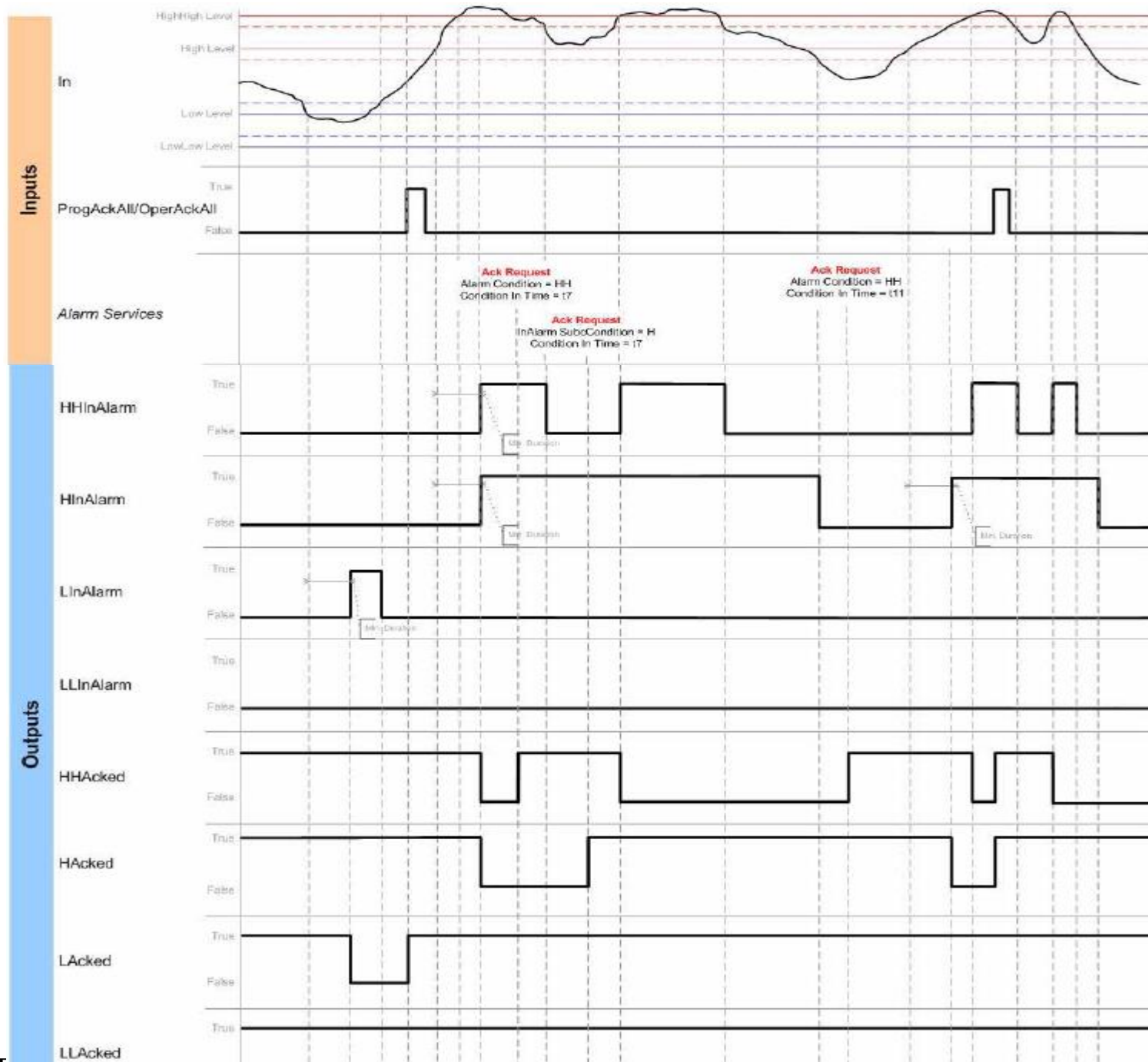
ALMA Alarm Acknowledge Required

Scope:  MainProgram   

	Name 	Alias For	Base Tag	Data Type	Style
	ack			BOOL	Decimal
	disable			BOOL	Decimal
	enable			BOOL	Decimal
	reset			BOOL	Decimal
	+ Analog_ALMA			ALARM_ANALOG	
	+ Pressure			DINT	Decimal
					



Inputs And Outputs Data Of ALMA



ANALOG ALARM INSTRUCTION: ALMA

FactoryTalk View Studio - Site Edition (Local) - [ALMD - /ALMA// (Display)]

File Edit View Settings Objects Arrange Animation Tools Window Help

Explorer - ALMA

- Local (SKY-A9A05C886A1)
 - ALMA
 - Runtime Security
 - ALMD
 - Communication S...
 - ALMA
 - System
 - Command Lin...
 - HMI Tags
 - Tags
 - Graphics
 - Displays
 - Global Object...
 - Symbol Factor...
 - Libraries
 - Images
 - Parameters
 - Recipes
 - Local Messag...
 - Trend Templa...
 - Trend Snapsh...

1 0 0 100

!		Event Time	Alarm Name	Condition N...	Message
!		6/25/1998 10:02:01 AM	[ALMD]Digital_alarm	TRIP_L	MOTOR FAIL OVER LOAC
!		6/25/1998 10:02:59 AM	...m:MainProgram.Analog_ALMA	HIHI	PRESSURE VERY HIGH
!		6/25/1998 10:03:56 AM	...m:MainProgram.Analog_ALMA	LO	PRESSURE LOW
!		6/25/1998 10:03:56 AM	...m:MainProgram.Analog_ALMA	LOLO	PRESSURE VERY LOW
!		6/25/1998 10:05:02 AM	...m:MainProgram.Analog_ALMA	LO	PRESSURE LOW
!		6/25/1998 10:05:02 AM	...m:MainProgram.Analog_ALMA	LOLO	PRESSURE VERY LOW
!		6/25/1998 10:05:02 AM	...m:MainProgram.Analog_ALMA	HI	PRESSURE HIGH
!		11/14/2014 5:12:49 PM	[ALMD]	STATUS	Mode of controller ALMD F

No message selected.

#8 3 0 3 0 Filter: No Sorted by: Evc

MSG, GSV, SSV INSTRUCTIONS

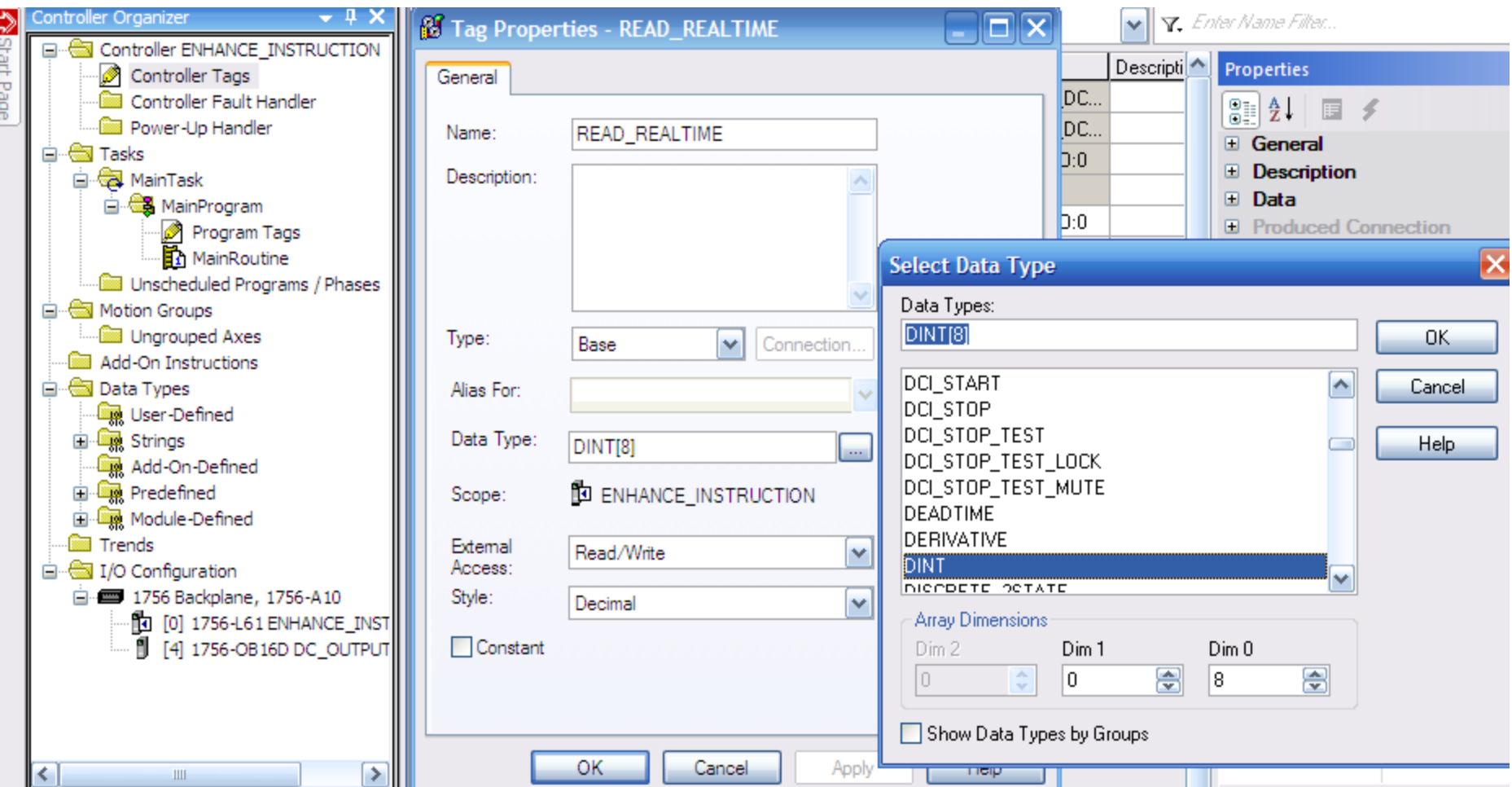
I/O Instructions: Message Control (MSG), Get System Value(GSV), Set system Value(SSV)

If You Want To	Use This Instruction	Available In These Languages
send data to or from another module	MSG	relay ladder structured text
get controller status information	GSV	relay ladder structured text
set controller status information	SSV	relay ladder structured text
- send output values to an I/O module or consuming controller at a specific point in your logic - trigger an event task in another controller	IOT	relay ladder structured text

MSG, GSV, SSV INSTRUCTIONS

Read Realtime in PLC

Create a tag to store DateTime data of PLC, the tag is 8(DINT) Array



MSG, GSV, SSV INSTRUCTIONS

Use GSV instruction to read and store Realtime in plc
Depend on your applications, which data in array is used

The screenshot displays the RSLogix 5000 software interface. The title bar indicates the project is 'ENHANCE_INSTRUCTION [1756-L61 20.3]* - [MainProgram - MainRoutine]'. The menu bar includes File, Edit, View, Search, Logic, Communications, Tools, Window, and Help. The toolbar contains various icons for file operations, logic editing, and communication. The 'Path' field shows 'AB_ETHIP-1\192.168.1.72\Backplane\0*'. The 'MSG' tab is selected in the instruction palette, and the 'GSV' instruction is highlighted. The 'Controller Organizer' on the left shows the project structure, including 'MainRoutine'. The main workspace shows a ladder logic program with a blue box labeled '0' and an 'End' instruction. A tooltip for the GSV instruction is visible, showing the following details:

GSV	
Get System Value	
Class Name	WallClockTime
Instance Name	
Attribute Name	DateTime
Dest	READ_REALTIME[0]
	2014

If DateTime data is wrong, use SSV to set DateTime to PLC

MSG, GSV, SSV INSTRUCTIONS

Choose Monitor Tags to view DateTime data of the controller

The screenshot displays the 'Controller Organizer' software interface. On the left is a tree view of the controller structure, including folders for 'Controller ENHANCE_INSTRUCTION', 'Tasks', 'Motion Groups', 'Add-On Instructions', 'Data Types', 'Trends', and 'I/O Configuration'. The 'Data Types' folder is expanded, showing 'User-Defined', 'Strings', 'Add-On-Defined', 'Predefined', and 'Module-Defined'. The 'I/O Configuration' folder is also expanded, showing '1756 Backplane, 1756-A10' and two tags: '[0] 1756-L61 ENHANCE_INST' and '[4] 1756-OB16D DC_OUTPUT'.

The main window on the right is titled 'Monitor Tags' and shows a table of tags for the selected scope 'ENHANCE_INST'. The table has columns for Name, Value, Force Mask, Style, and Data Type. The tags are listed as follows:

Name	Value	Force Mask	Style	Data Type
+ Local:4:C	{...}	{...}		AB:1756_DO_DC...
+ Local:4:I	{...}	{...}		AB:1756_DO_DC...
- Local:4:O	{...}	{...}		AB:1756_DO:O:O
+ Local:4:O.Data	0		Decimal	DINT
- READ_REALTIME	{...}	{...}	Decimal	DINT[8]
+ READ_REALTIME[0]	2014		Decimal	DINT
+ READ_REALTIME[1]	3		Decimal	DINT
+ READ_REALTIME[2]	1		Decimal	DINT
+ READ_REALTIME[3]	16		Decimal	DINT
+ READ_REALTIME[4]	29		Decimal	DINT
+ READ_REALTIME[5]	59		Decimal	DINT
+ READ_REALTIME[6]	547759		Decimal	DINT
+ READ_REALTIME[7]	0		Decimal	DINT

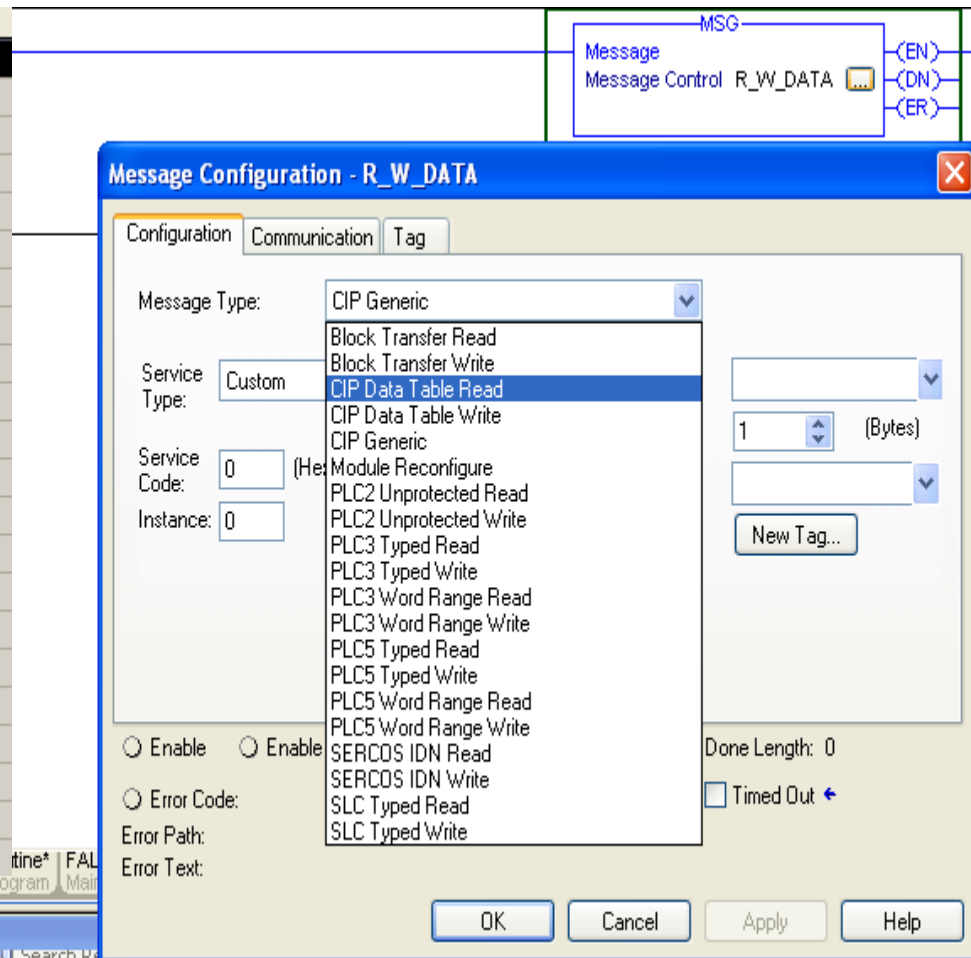
At the bottom of the window, there are buttons for 'Monitor Tags' and 'Edit Tags'.

MSG, GSV, SSV INSTRUCTIONS

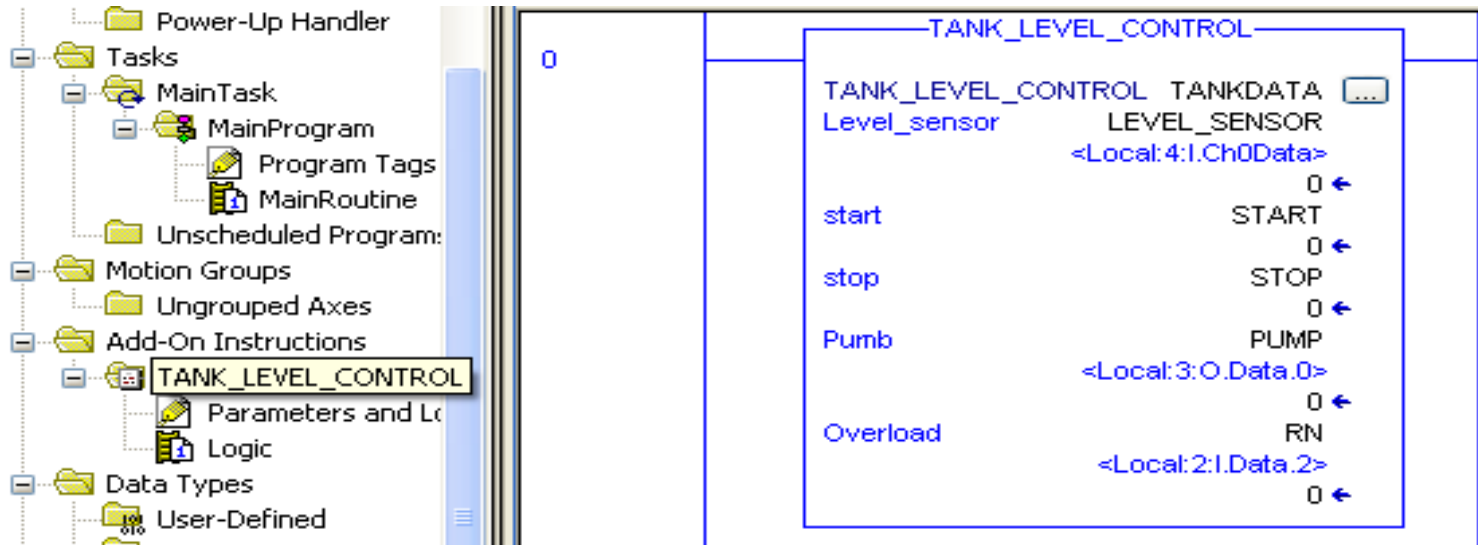
Message Control (MSG)

Read or write data to or from the controller or a block of data to or from another module on another network.

Name	Alias For	Base Tag	Data Type
R_W_DATA			MESSAGE
+ R_W_DATA.Flags			INT
+ R_W_DATA.EW			BOOL
+ R_W_DATA.ER			BOOL
+ R_W_DATA.DN			BOOL
+ R_W_DATA.ST			BOOL
+ R_W_DATA.EN			BOOL
+ R_W_DATA.TO			BOOL
+ R_W_DATA.EN_CC			BOOL
+ R_W_DATA.ERR			INT
+ R_W_DATA.EXERR			DINT
+ R_W_DATA.ERR_SRC			SINT
+ R_W_DATA.DN_LEN			INT
+ R_W_DATA.REQ_LEN			INT
+ R_W_DATA.DestinationLink			INT
+ R_W_DATA.DestinationNode			INT
+ R_W_DATA.SourceLink			INT
+ R_W_DATA.Class			INT
+ R_W_DATA.Attribute			INT
+ R_W_DATA.Instance			DINT
+ R_W_DATA.LocalIndex			DINT
+ R_W_DATA.Channel			SINT



ADDON INSTRUCTION

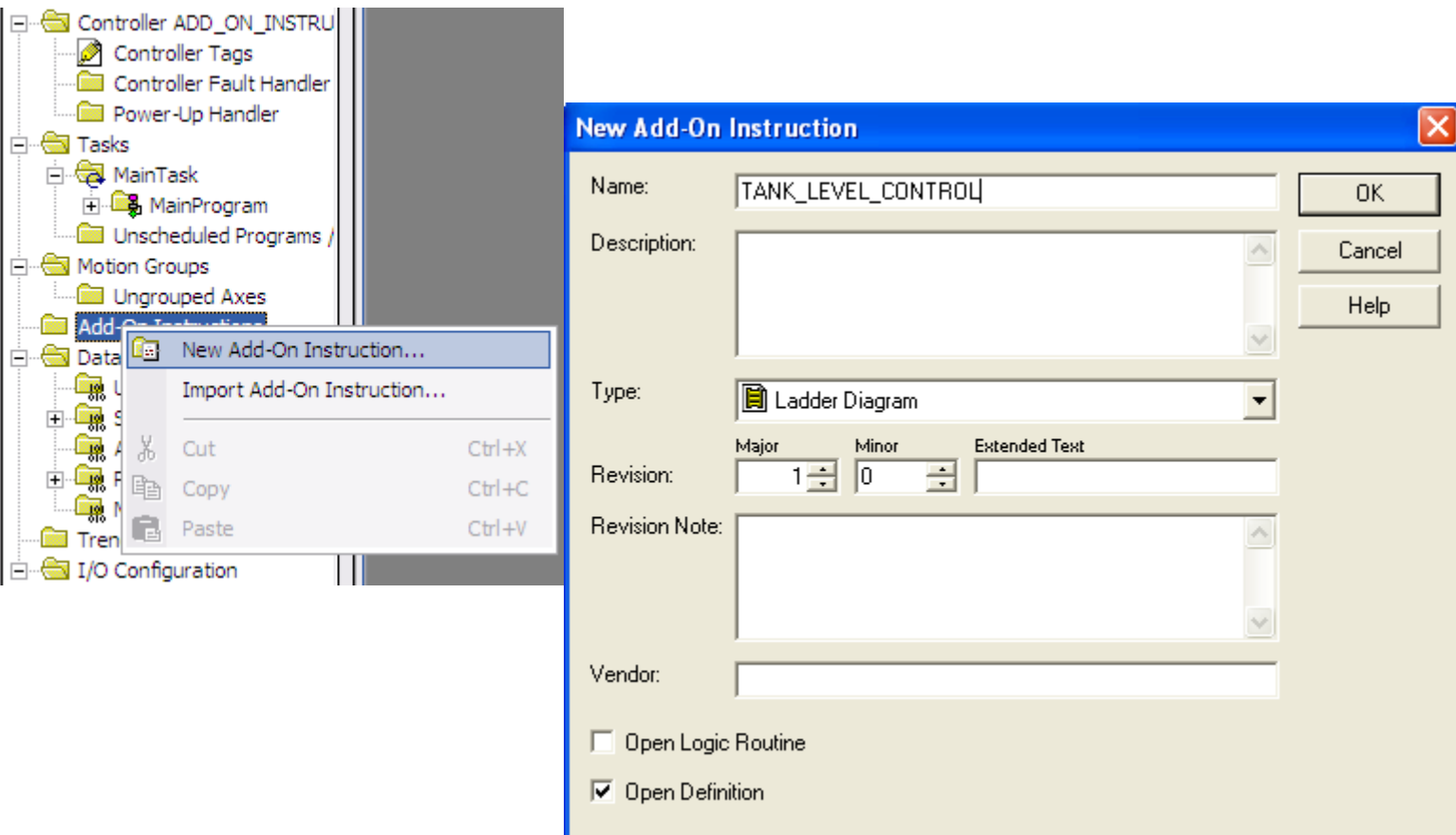


Add-on Instruction introduction

- Custom Instruction
- Reuse code
- Provide an easier to understand interface
- Export and Import an Add-on Instruction

ADDON INSTRUCTION

Creating Add-on Instruction



ADDON INSTRUCTION

Creating parameters and Local Tags

Add-On Instruction Definition - TANK_LEVEL_CONTROL v1.0

General Parameters* Local Tags Scan Modes Change History Help

Name	Usage	Data Type	Default	Style	Req	Vis	Description
EnableIn	Input	BOOL	1	Decimal	☐	☐	Enable Input - Sys...
EnableOut	Output	BOOL	0	Decimal	☐	☐	Enable Output - S...
+Level_sensor	Input	INT	0	Decimal	☒	☒	
start	Input	BOOL	0	Decimal	☒	☒	
stop	Input	BOOL	0	Decimal	☒	☒	
*Pumb	Output	BOOL	0	Decimal	☒	☒	
*Overload	Input	BOOL	0	Decimal	☒	☒	

Move Up Move Down

Logic Data Type Size: 8 byte (s) OK Cancel Apply

Add-On Instruction Definition - TANK_LEVEL_CONTROL v1.0

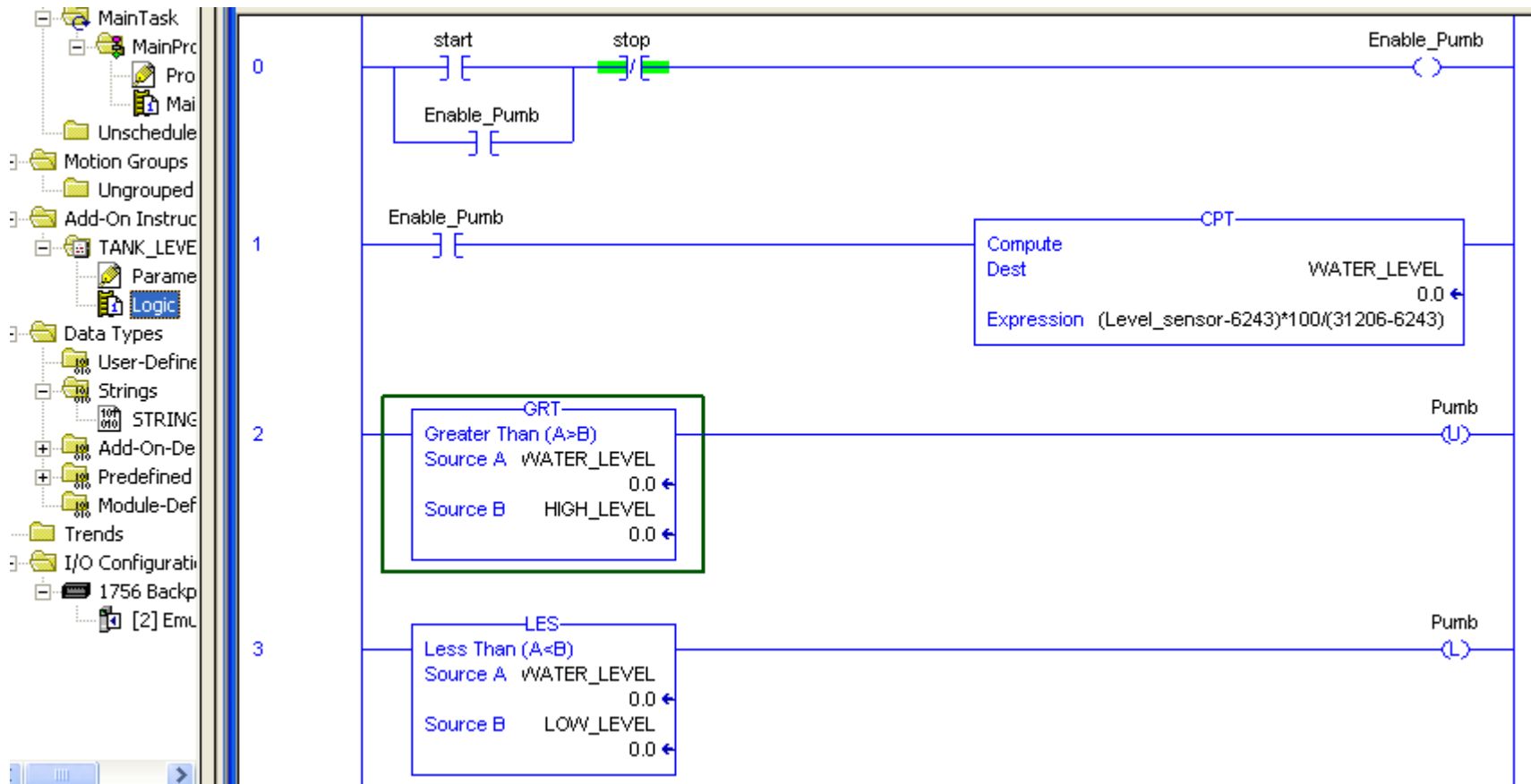
General Parameters Local Tags Scan Modes Change History Help

Name	△	Data Type	Default	Style	Description
Enable_Pumb		BOOL	0	Decimal	
HIGH_LEVEL		REAL	0.0	Float	
LOW_LEVEL		REAL	0.0	Float	
WATER_LEVEL		REAL	0.0	Float	

Logic Data Type Size: 20 byte (s) OK Cancel Apply Help

ADDON INSTRUCTION

Creating logic for the Add-on Instruction



ADDON INSTRUCTION

Creating I/O Tags and Adding the Add_on instruction to project.

Controller Organ... 

Scope:  TANK_CONTRC Show: All Tags

Name	Alias For	Base Tag	Data Type
+ Local:2:I			AB:1769_DI32:I:0
+ Local:3:C			AB:1769_DO32:C:0
+ Local:3:I			AB:1769_DO32:I:0
+ Local:3:O			AB:1769_DO32:O:0
+ Local:4:C			AB:1769_IF4:C:0
+ Local:4:I			AB:1769_IF4:I:0
RN	Local:2:I.Data.2	Local:2:I.Data.2	BOOL
START	Local:2:I.Data.0	Local:2:I.Data.0	BOOL
STOP	Local:2:I.Data.1	Local:2:I.Data.1	BOOL
PUMP	Local:3:O.Data.0	Local:3:O.Data.0	BOOL
+ LEVEL_SENSOR	Local:4:I.Ch0Data	Local:4:I.Ch0Data	INT

Controller Organizer 

TANK_LEVEL_CONTROL

0

TANK_LEVEL_CONTROL TANKDATA 

Level_sensor LEVEL_SENSOR
<Local:4:I.Ch0Data>

start START
0

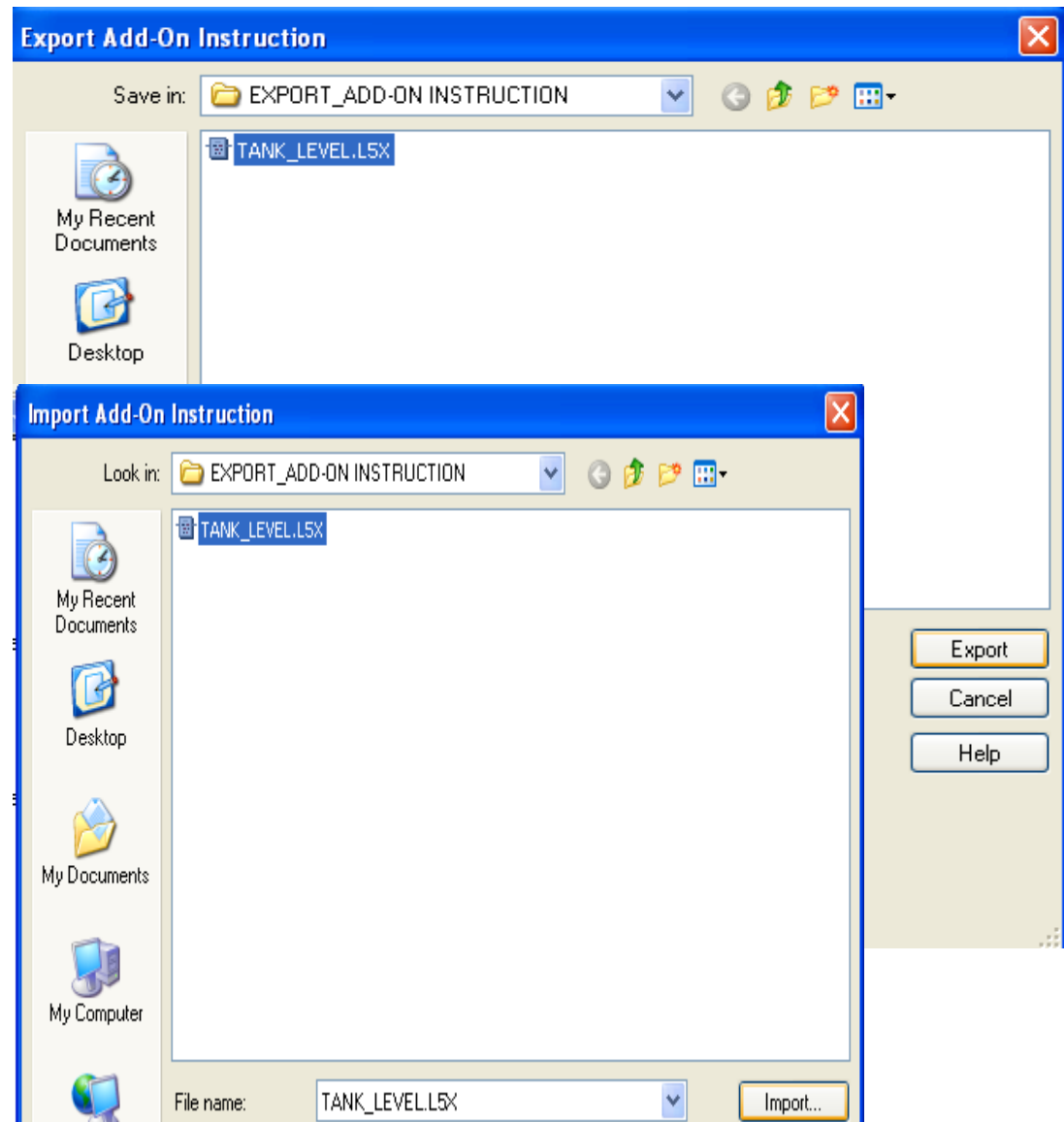
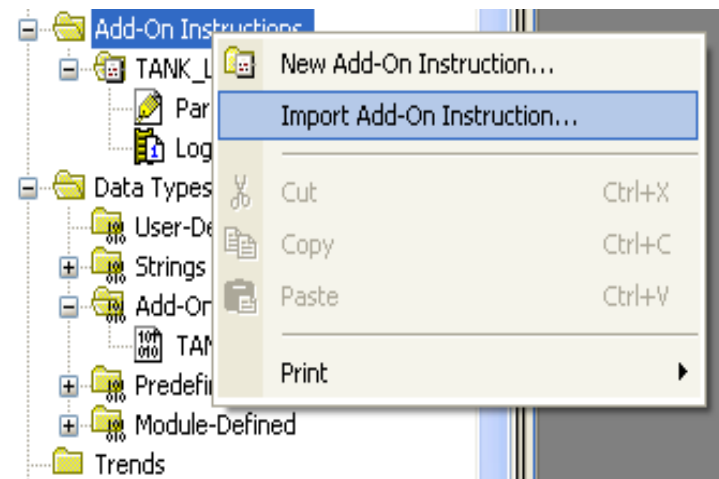
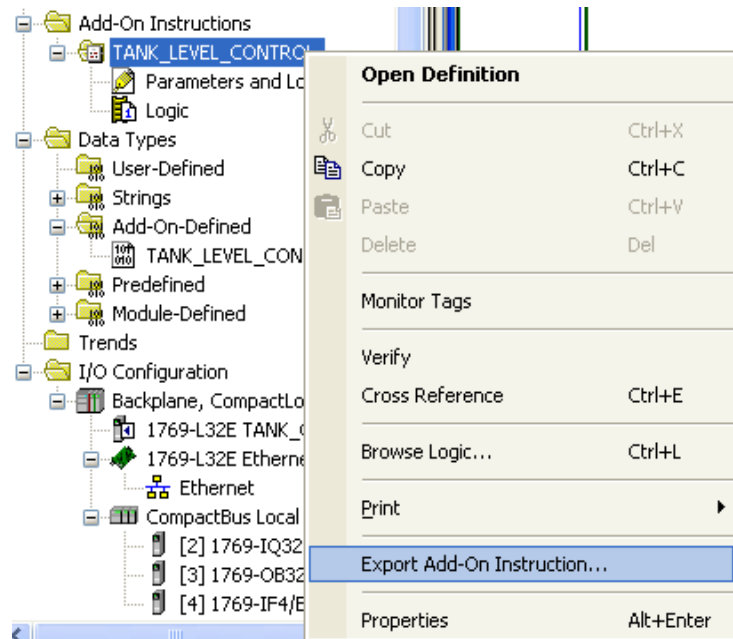
stop STOP
0

Pumb PUMP
<Local:3:O.Data.0>

Overload RN
<Local:2:I.Data.2>
0

ADDON INSTRUCTION

Export and Import the Add-on Instruction



MINOR AND MAJOR FAULT

Minor Fault: The CPU does not switch to faulted mode

- Periodic Task overlap.
- Load from nonvolatile memory.
- Problem with serial port.
- Low battery.....

Major Fault: The CPU switch to faulted mode

The CPU powered on in run mode.

A required I/O module connection failed.

Configuration fault occurred.....

MINOR FAULT CODES

Type	Code	Cause	Recovery Method
4	4	An arithmetic overflow occurred in an instruction.	Fix program by examining arithmetic operations (order) or adjusting values.
4	5	In a GSV/SSV instruction, the specified instance was not found.	Check the instance name.
4	6	In a GSV/SSV instruction, either: · specified Class name is <i>not</i> supported · specified Attribute name is <i>not</i> valid	Check the Class name and Attribute name.
4	7	The GSV/SSV destination tag was too small to hold all of the data.	Fix the destination so it has enough space.
4	35	PID delta time ≤ 0 .	Adjust the PID delta time so that it is > 0 .
4	36	PID setpoint out of range	Adjust the setpoint so that it is within range.
4	51	The LEN value of the string tag is greater than the DATA size of the string tag.	1. Check that no instruction is writing to the LEN member of the string tag. 2. In the LEN value, enter the number of characters that the string contains.
4	52	The output string is larger than the destination.	Create a new string data type that is large enough for the output string. Use the new string data type as the data type for the destination.
4	53	The output number is beyond the limits of the destination data type.	Either: · Reduce the size of the ASCII value.

MINOR FAULT CODES

4	56	The Start or Quantity value is invalid.	1. Check that the Start value is between 1 and the DATA size of the Source. 2. Check that the Start value plus the Quantity value is less than or equal to the DATA size of the Source.
4	57	The AHL instruction failed to execute because the serial port is set to no handshaking.	Either: · Change the Control Line setting of the serial port. · Delete the AHL instruction.
6	2	Periodic task overlap. Periodic task has not completed before it is time to execute again.	Simplify program(s), or lengthen period, or raise relative priority, etc.
7	49	Project loaded from nonvolatile memory.	
9	0	Unknown error while servicing the serial port.	Contact Rockwell Automaiton Technical Support.

MINOR FAULT CODES

Type	Code	Cause	Recovery Method
9	1	The CTS line is not correct for the current configuration.	Disconnect and reconnect the serial port cable to the controller. Make sure the cable is wired correctly
9	2	Poll list error. A problem was detected with the DF1 master's poll list, such as specifying more stations than the size of the file, specifying more than 255 stations, trying to index past the end of the list, or polling the broadcast address (STN #255).	Check for the following errors in the poll list: - total number of stations is greater than the space in the poll list tag - total number of stations is greater than 255 - current station pointer is greater than the end of the poll list tag - a station number greater than 254 was encountered
9	5	DF1 slave poll timeout. The poll watchdog has timed out for slave. The master has not polled this controller in the specified amount of time.	Determine and correct delay for polling.
9	9	Modem contact was lost. DCD and/or DSR control lines are not being received in proper sequence and/or state.	Correct modem connection to the controller.
10	10	Battery not detected or needs to be replaced.	Install new battery.

MINOR FAULT CODES

Ex1: Arithmetic overflow, result of arithmetic instruction is out of range(
Type =4, code =4)

The screenshot displays a PLC programming environment. On the left is a project tree with folders like 'Controller MA...', 'Tasks', 'Motion Groups', and 'Data Types'. The main workspace shows a ladder logic diagram with a vertical line representing a rung. A 'Controller Properties - MAJOR_MINOR' dialog box is open, showing the 'Major Faults' tab. It lists '3207 minor faults since last cleared.' and shows 'Recent Faults' with a specific error: '2/16/2014 2:22:30 AM (Type 04) Program Fault (Code 04) Arithmetic overflow. Result of an arithmetic instruction out of range.' The fault details are: Task: MainTask, Program: MainProgram, Routine: MainRoutine, Location: Rung 1. The dialog also has a 'Clear Minors' button and a 'Fault Bits' section with checkboxes for Powerup, I/O, Program (checked), and Watchdog. In the background, a ladder logic diagram is visible, showing an 'ADD' instruction with Source A: Source 6634.0, Source B: 2, and Dest: Source 6634.0. Below it is a 'MOV' instruction with Source: Source 6634.0 and Dest: Destination -22.

MINOR FAULT CODES

EX2: Periodic task overlap, Task scheduled again before it finished executing(Type =6, code =2)

The screenshot displays a software development environment. On the left is a project tree with folders like 'Controller MAJOR_MIN', 'Tasks', 'MainTask', 'Periodic_Task_ove', 'PROGRAM_CP', 'Motion Groups', 'Data Types', and 'Strings'. The main workspace shows a ladder logic diagram with a 'Compute' block containing the expression $(Source * Destination) / 10$ and a value of 140118.0. Below the workspace, the 'Controller Properties - MAJOR_MINOR' window is open, showing the 'Minor Faults' tab. It indicates '16280 minor faults since last cleared' and lists a recent fault: '2/16/2014 2:57:56 AM (Type 06) Watchdog Fault (Code 02) Periodic task overlap. Task scheduled again before it finished executing. Task: Periodic_Task_overlap'. To the right of the fault list are 'Fault Bits' for 'Powerup' and 'I/O'.

Controller MAJOR_MIN

- Controller Tags
- Controller Fault Ha
- Power-Up Handler
- Tasks
 - MainTask
 - MainProgram
 - Program Ta
 - MainRoutin
 - Periodic_Task_ove
 - PROGRAM_CP
 - Program Ta
 - routine_cp
- Unscheduled Progr
- Motion Groups
- Ungrouped Axes
- Add-On Instructions
- Data Types
 - User-Defined
 - Strings
 - Add-On-Defined
 - Predefined
 - Module-Defined

0

(End)

Compute
Dest
Expression $(Source * Destination) / 10$
CPT
CPT
140118.0

Controller Properties - MAJOR_MINOR

General	Serial Port	System Protocol	User Protocol	Major Faults
Minor Faults	Date/Time	Advanced	SFC Execution	File

16280 minor faults since last cleared. Clear Minors

Recent Faults:

2/16/2014 2:57:56 AM
(Type 06) Watchdog Fault
(Code 02) Periodic task overlap. Task scheduled again before it finished executing.
Task: Periodic_Task_overlap

Fault Bits:
☐ Powerup
☐ I/O

MAJOR FAULT CODES

Type	Code	Cause	Recovery Method
1	1	The controller powered on in Run mode.	Execute the power-loss handler.
1	15	- A 1769 power supply is connected directly to the controller's 1769 CompactBus, with an invalid configuration. - The 1768 power supply powering the controller has failed.	- Remove the power supply from the 1769 CompactBus and cycle power to the system. - Replace the power supply.
1	60	For a controller with <i>no</i> CompactFlash card installed, the controller: - detected a non-recoverable fault - cleared the project from memory	1. Clear the fault. 2. Download the project. 3. Change to remote run/run mode. If the problem persists: 1. Before you cycle power to the controller, record the state of the OK and RS232 LEDs. 2. Contact Rockwell Automation support. See the back of this publication.
1	61	For a controller with a CompactFlash card installed, the controller: - detected a non-recoverable fault - wrote diagnostic information to the CompactFlash card - cleared the project from memory	1. Clear the fault. 2. Download the project. 3. Change to remote run/run mode. If the problem persists, contact Rockwell Automation support. See the back of this publication.
3	16	A required I/O module connection failed.	Check that the I/O module is in the chassis. Check electronic keying requirements.

MAJOR FAULT CODES

3	20	Possible problem with the ControlBus chassis.	Not recoverable - replace the chassis.
3	23	At least one required connection was not established before going to Run mode.	Wait for the controller I/O light to turn green before changing to Run mode.
4	16	Unknown instruction encountered.	Remove the unknown instruction. This probably happened due to a program conversion process.
4	20	Array subscript too big, control structure .POS or .LEN is invalid.	Adjust the value to be within the valid range. Don't exceed the array size or go beyond dimensions defined.
4	21	Control structure .LEN or .POS < 0.	Adjust the value so it is > 0.
4	31	The Parameters of the JSR instruction do not match those of the associated SBR or RET instruction.	Pass the appropriate number of Parameters. If too many Parameters are passed, the extra ones are ignored without any error.
4	34	A timer instruction has a negative preset or accumulated value.	Fix the program to not load a negative value into timer preset or accumulated value.
4	42	JMP to a label that did not exist or was deleted.	Correct the JMP target or add the missing label.

MAJOR FAULT CODES

4	82	A sequential function chart (SFC) called a subroutine and the subroutine tried to jump back to the calling SFC. Occurs when the SFC uses either a JSR or FOR instruction to call the subroutine.	Remove the jump back to the calling SFC.
4	83	The data tested was not inside the required limits.	Modify value to be within limits.
4	84	Stack overflow.	Reduce the subroutine nesting levels or the number of Parameters passed.
4	89	In a SFR instruction, the target routine does not contain the target step.	Correct the SFR target or add the missing step.
6	1	Task watchdog expired. User task has not completed in specified period of time. A program error caused an infinite loop, or the program is too complex to execute as quickly as specified, or a higher priority task is keeping this task from finishing.	Increase the task watchdog, shorten the execution time, make the priority of this task "higher," simplify higher priority tasks, or move some code to another controller.
7	40	Store to nonvolatile memory failed.	1. Try again to store the project to nonvolatile memory. 2. If the project fails to store to nonvolatile memory, replace the memory board.
7	41	Load from nonvolatile memory failed due to controller type mismatch.	Change to a controller of the correct type or download the project and store it on the CompactFlash card.
7	42	Load from nonvolatile memory failed because the firmware revision of the project in nonvolatile memory does not match the	Update the controller firmware to the same revision level as the project that is in nonvolatile memory.

MAJOR FAULT CODES

EX3: Timer with a negative value preset for its Pre (Type =04, code =34)

The screenshot displays a PLC programming environment. On the left is a project tree with folders like 'Controller MAJOR_MINOR', 'Controller Tags', 'Controller Fault Hand', 'Power-Up Handler', 'Tasks', 'MainTask', 'MainProgram', 'Program Tag', 'MainRoutine', 'Unscheduled Program', 'Motion Groups', 'Ungrouped Axes', 'Add-On Instructions', 'Data Types', 'User-Defined', 'Strings', 'Add-On-Defined', 'Predefined', 'Module-Defined', 'Trends', 'I/O Configuration', and '1756 Backplane, 175'. The main workspace shows a ladder logic diagram with a 'TON' (Timer On Delay) block. The block's 'Timer' is set to 'T', 'Preset' is '-200', and 'Accum' is '0'. The diagram includes a 'T.DN' contact and a 'LIHGT' coil. A 'Controller Properties - MAJOR_MINOR' dialog box is open in the foreground, showing the 'Major Faults' tab. It indicates '1 major fault since last cleared' and provides details for a recent fault: '2/16/2014 3:47:21 AM (Type 04) Program Fault (can be trapped by a fault routine) (Code 34) A timer instruction had a negative value for its PRE or ACC. Task: MainTask Program: MainProgram Routine: MainRoutine Location: Rung 0'. A 'Clear Majors' button is also visible.

MAJOR FAULT CODES

EX4: JMP to a label that do not exists (Type =04, code =42)

The screenshot shows a PLC programming environment. On the left is a project tree with folders like 'Controller MAJOR_MINO', 'Tasks', 'MainTask', and 'MainProgram'. The main workspace shows a ladder logic program for 'MainProgram - MainRoutine'. Rung 0 contains a 'JMP' instruction. Rung 1 has a normally open contact labeled 'ON' and a coil labeled 'LIHGT'. Rung 2 has a coil labeled 'a' and a label '[LBL]'. A 'Controller Properties - MAJOR_MINOR' dialog box is open, displaying fault information. The 'Major Faults' tab shows '1 major fault since last cleared' and a 'Clear Majors' button. The 'Recent Faults' list shows a fault at '2/16/2014 4:24:31 AM' with details: '(Type 04) Program Fault (can be trapped by a fault routine) (Code 42) JMP to a label that did not exist or was deleted. Task: MainTask Program: MainProgram Routine: MainRoutine Location: Rung 0'.

MAJOR FAULT CODES

EX5: Task watchdog expired(Type =06, code =01)

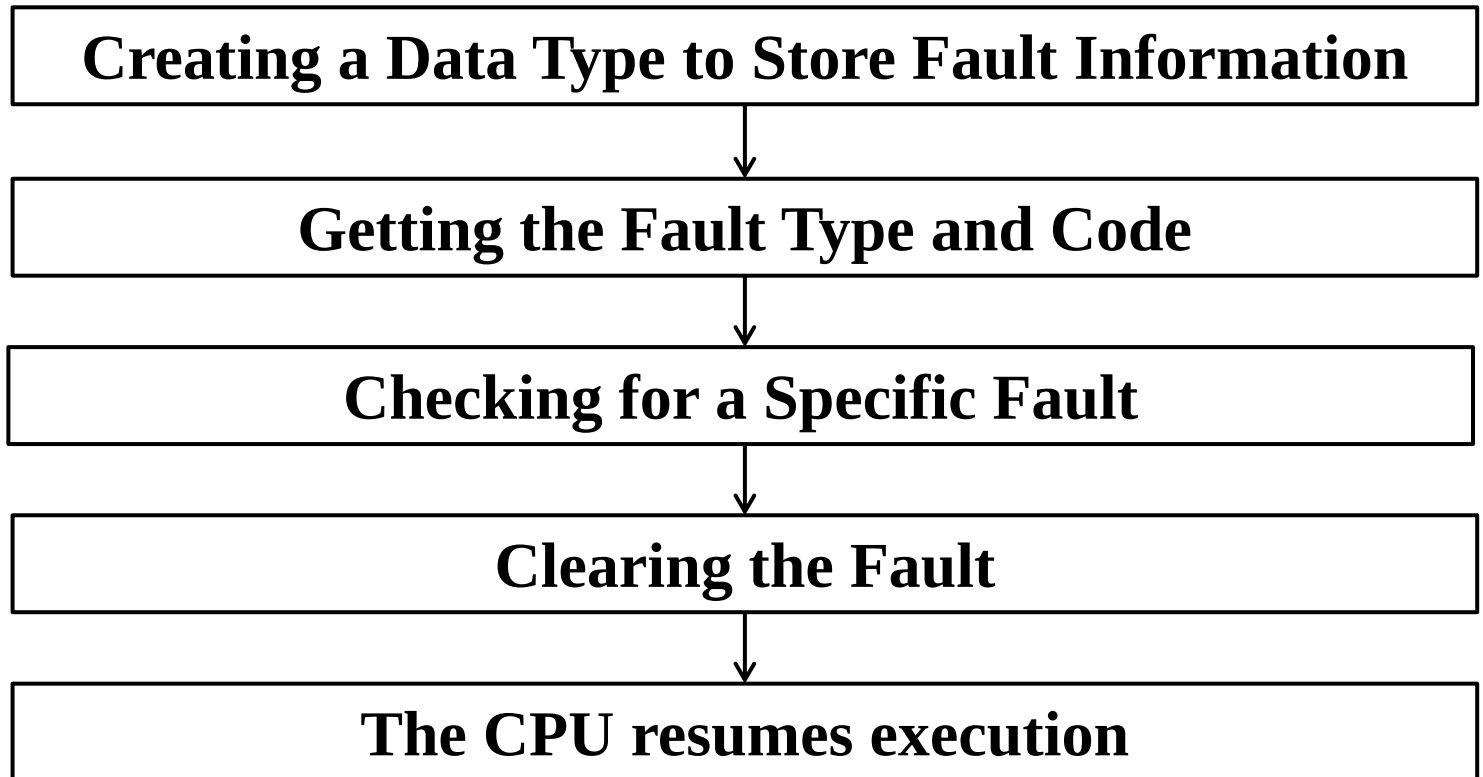
The screenshot displays a software interface for a controller, likely a PLC or motion controller, showing a major fault code. The main window displays a ladder logic diagram for a program named "MainProgram - fault_routine". The diagram shows a "loop" with a "LBL" (Label) and a "JMP" (Jump) instruction, indicating a loop structure. The "loop_bit" is shown as a green bar, indicating it is active. The "loop" instruction is labeled "loop (JMP)".

A "Controller Properties - MAJOR_MINOR" dialog box is open, showing the status of the controller. The "Minor Faults" tab is selected, and it displays "1 major fault since last cleared." and a "Clear Majors" button. The "Recent Faults" section shows a fault entry: "2/16/2014 4:34:22 AM (Type 06) Watchdog Fault (Code 01) Task watchdog expired. May have been caused by an infinite loop, a complex program, or a higher priority task. Task: MainTask Program: MainProgram Routine: fault_routine".

The interface also includes a "Faulted" section with checkboxes for "Program Mode", "Controller Fault", "Battery OK", and "I/O Not Present". The "Path" is set to "AB_VBP-1\1*". The "Sequencer" and "Equipment Phase" tabs are visible at the bottom of the main window.

HANDLING FAULTs

Programmatically Clear a Major Fault



HANDLING FAULTs

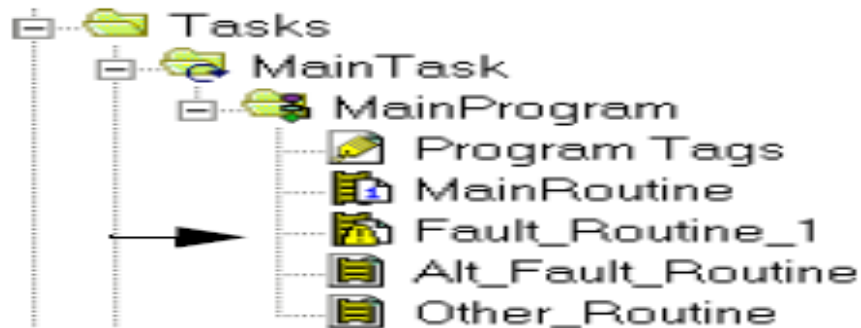
Choosing Where To Place The Fault Routine .

If you want take specific action/clear the fault when		Do this
Condition	Fault Type	
The execution of an instruction faults	4	Create a Fault Routine for a Program
Communication with an I/O module fails	3	Create a Routine for the Controller Fault Handler
Watchdog time for a task expires	6	
While a project is downloading to the controller, the keyswitch is placed in RUN	8	
The controller powers up in run/remote run mode	1	Create a Routine for the Power-Up Handler

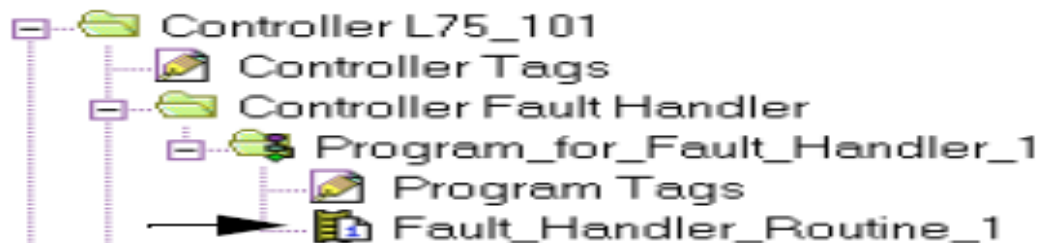
HANDLING FAULTS

Choosing Where To Place The Fault Routine .

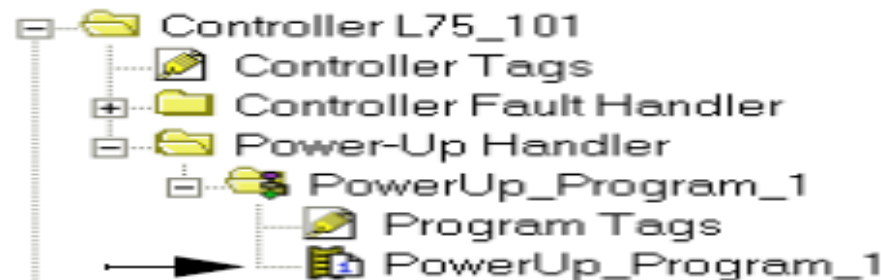
Program Fault Routine



Controller Fault Routine



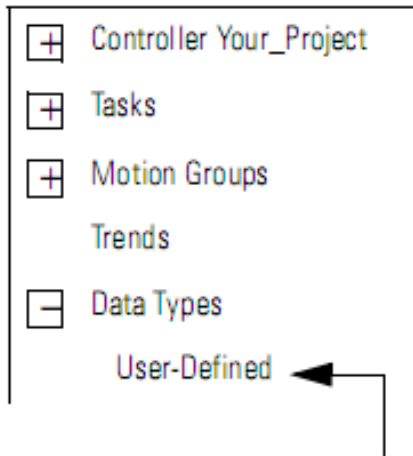
Power-Up Fault Handler Routine



HANDLING FAULTs

Creating a Data Type to Store Fault Information.

To create a new data type:



Right-click and choose *New Data Type*.

Data Type: FAULTRECORD				
Name	FAULTRECORD			
Description	Stores the MajorFaultRecord attribute or MinorFaultRecord attribute of the PROGRAM object.			
Members				
	Name	Data Type	Style	Description
	Time_Low	DINT	Decimal	lower 32 bits of the fault timestamp value
	Time_High	DINT	Decimal	upper 32 bits of the fault timestamp value
	Type	INT	Decimal	fault type (program, I/O, etc)
	Code	INT	Decimal	unique code for the fault
	Info	DINT[8]	Hex	fault specific information

- To access system information, use GSV(Get System Value) and SSV(Set System Value) Instruction.
- For status information about a program, access the program Objects.
- For fault information, access these attribute of the program Object

HANDLING FAULTs

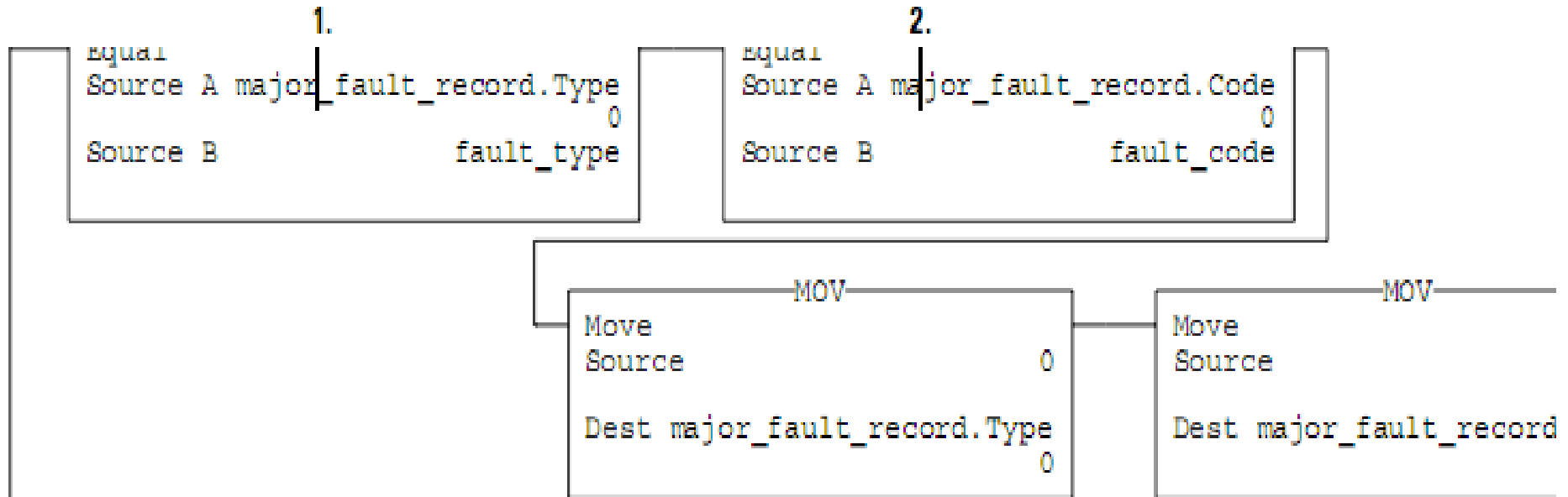
Getting the Fault Type and Code.



1. The GSV instruction accesses the MAJORFAULTRECORD attribute of this program. This attribute stores information about the fault.
2. The GSV instruction stores the fault information in the major_fault_record tag (of type FAULTRECORD). When you enter a tag that is based on a structure, enter the first member of the tag.

HANDLING FAULTs

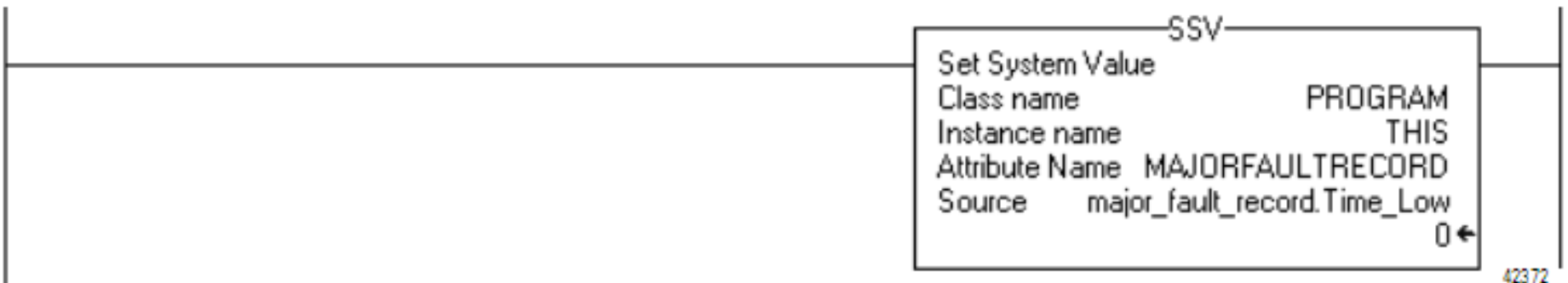
Checking for a Specific Fault.



1. The first EQU instruction checks for a specific type of fault, such as program, I/O. In Source B, enter the value for the type of fault that you want to clear.
2. The second EQU instruction checks for a specific fault code. In Source B, enter the value for the code that you want to clear.
3. The first CLR instruction sets to zero the value of the fault type in the `major_fault_record` tag.
4. The second CLR instruction sets to zero the value of the fault code in the `major_fault_record` tag.

HANDLING FAULTs

Clearing the Fault.



1. The SSV instruction writes new values to the MAJORFAULTRECORD attribute of this program.
2. The SSV instruction writes the values contained in the major_fault_record tag. Since the Type and Code member are set to zero, the fault clears and the controller resumes execution.

HANDLING FAULTs

Example: Checking and clearing the fault when CPU powered in run mode:

Type =1, Code = 1.

Creating a Data type to store fault information of program

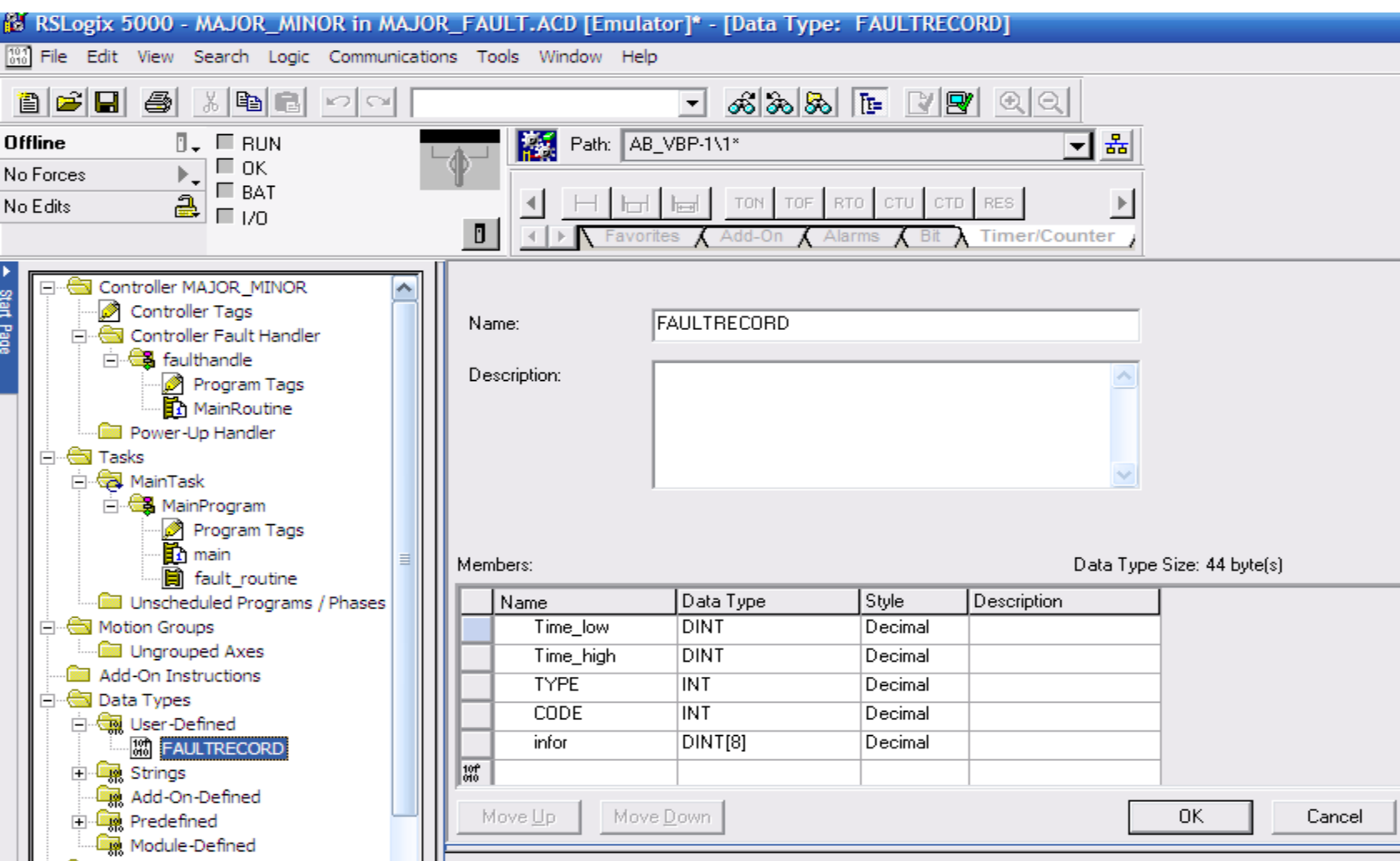
Use GSV instruction to read MAJORFAULTRECORD attribute of the program

Checking for a specific fault **Type** and **Code** then clearing the fault

Use SSV instruction to write new value to MAJORFAULTRECORD attribute of the program.

HANDLING FAULTs

➤ Creating a Data Type to store fault information of program



HANDLING FAULTs

➤ Creating a tag to store MAJORFAULTRECORD of the program

The screenshot displays a software development environment. On the left is a project tree with the following structure:

- Controller MAJOR_MINOR
 - Controller Tags
 - Controller Fault Handler
 - faulthandle
 - Program Tags
 - MainRoutine
 - Power-Up Handler
- Tasks
 - MainTask
 - MainProgram
 - Program Tags
 - main
 - fault_routine
 - Unscheduled Programs / Phases
- Motion Groups

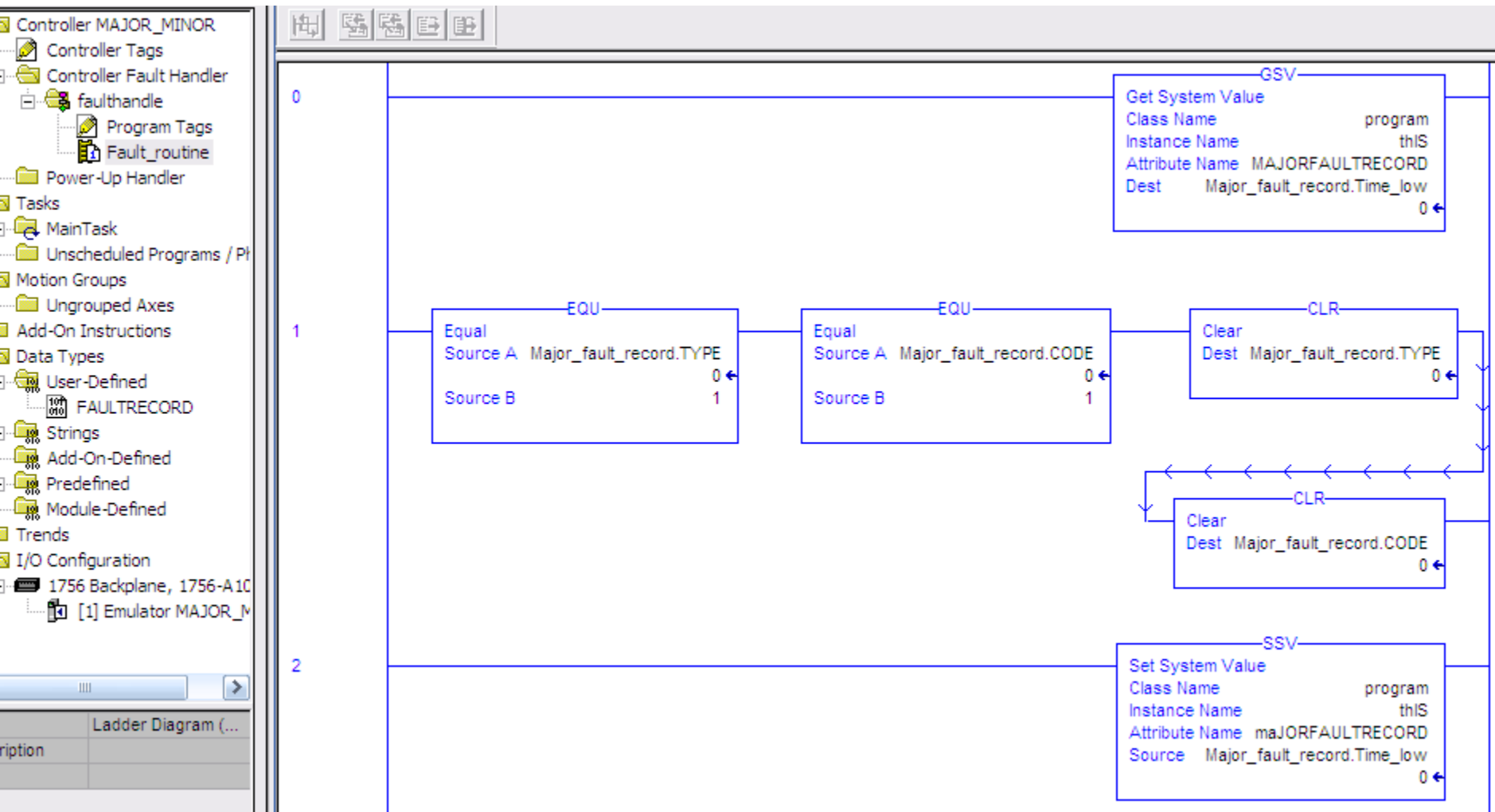
On the right, the 'Scope' is set to 'MAJOR_MINOR'. Below this is a table listing tags:

Name	Alias For	Base Tag	Data Type	Style
- Major_fault_record			FAULTRECORD	
+ Major_fault_record.Time_low			DINT	Decimal
+ Major_fault_record.Time_high			DINT	Decimal
+ Major_fault_record.TYPE			INT	Decimal
+ Major_fault_record.CODE			INT	Decimal
+ Major_fault_record.infor			DINT[8]	Decimal

At the bottom, there are tabs for 'Monitor Tags' and 'Edit Tags'.

HANDLING FAULTS

➤ Creating a routine in Controller Fault Handler and write a program as following



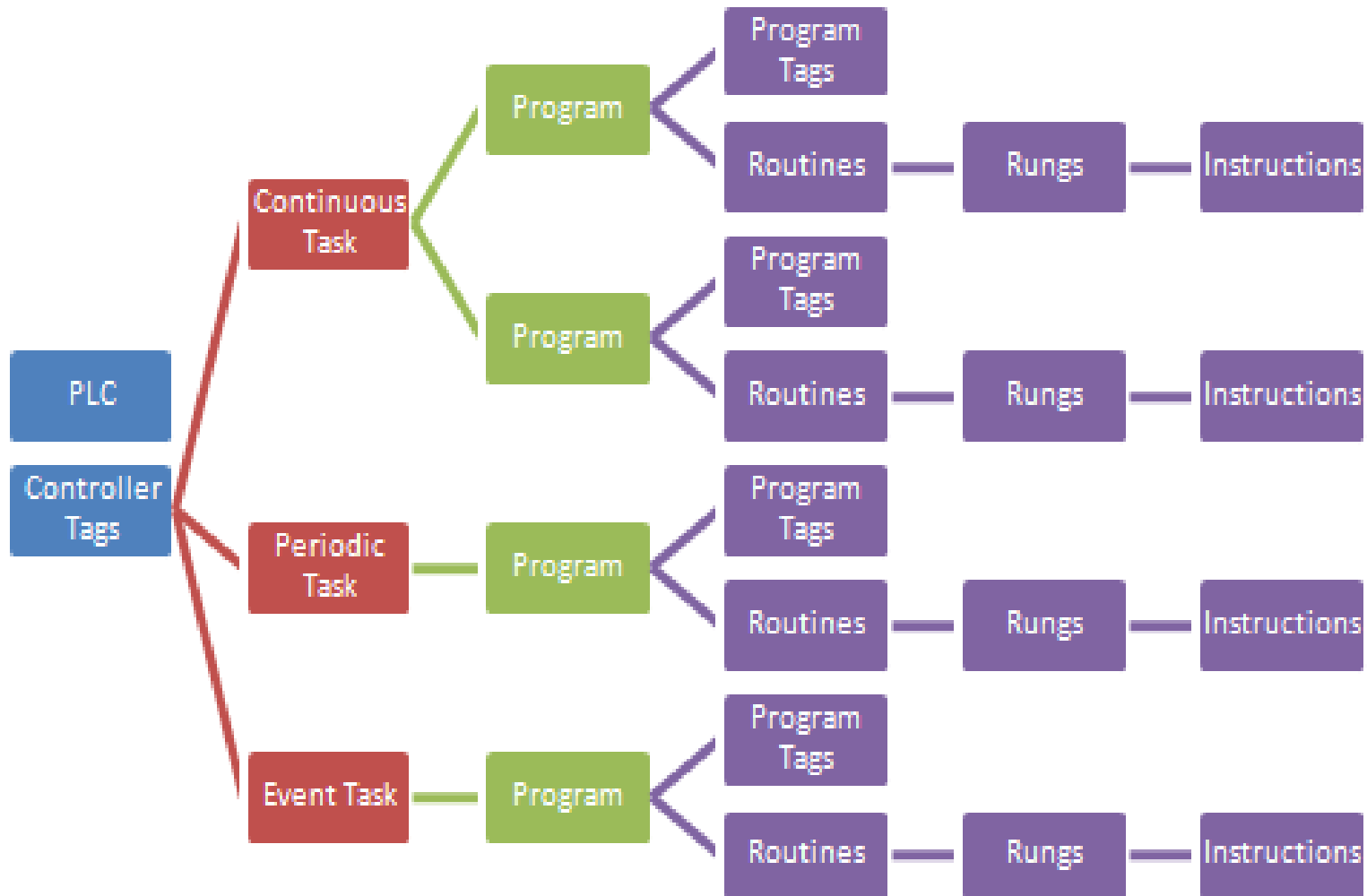
HANDLING FAULT

EX2: Handling faults for the CPU

Condition	Fault Type	
The execution of an instruction faults	4	Create a Fault Routine for a Program
Communication with an I/O module fails	3	Create a Routine for the Controller Fault Handler
Watchdog time for a task expires	6	
While a project is downloading to the controller, the keyswitch is placed in RUN	8	
The controller powers up in run/remote run mode	1	Create a Routine for the Power-Up Handler

RSLOGIX 5000 CONTROLLER TASKS

A RSLogix 5000 Controller supports three type of tasks



RSLOGIX 5000 CONTROLLER TASKS

A RSLogix 5000 Controller supports three type of tasks

- Continuous Tasks
- Periodic Task
- Event Task

Characteristic of Tasks

- The controller executes only one Task at one time
- A Task can interrupt a different task that is executing and take control if it has high priority
- In any given Task, only one program executes at one time.

RSLOGIX 5000 CONTROLLER TASKS

Function of Tasks

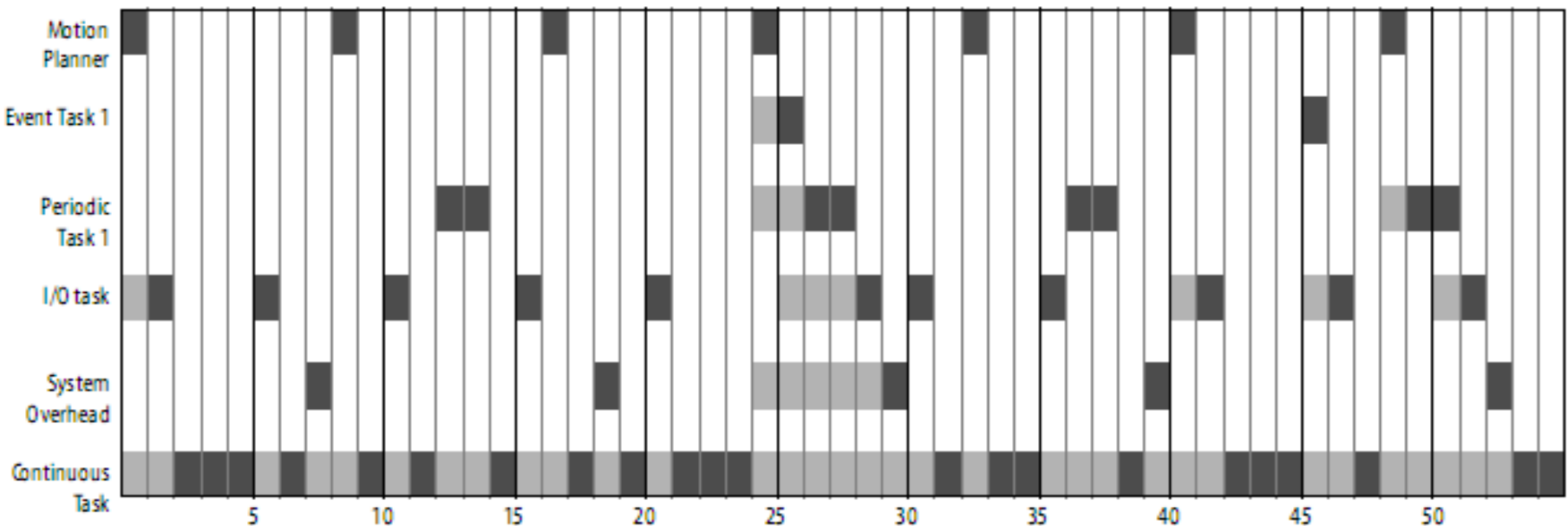
If you want to execute a section of your logic	Then use this type of task	Description
All of the time	Continuous Task	The continuous task runs in the background. Any CPU time not allocated to other operations (such as motion, communication, and periodic or event tasks) is used to execute the programs within the continuous task. - The continuous task runs all the time. When the continuous task completes a full scan, it restarts immediately. - A project does not require a continuous task. If used, there can be only one continuous task.
- At a constant period (example, every 100 ms) - Multiple times within the scan of your other logic	Periodic Task	A periodic task performs a function at a specific period. Whenever the time for the periodic task expires, the periodic task; - interrupts any lower priority tasks. - executes one time. - returns control to where the previous task left off. You can configure the time period from 0.1 ms. . . 2000 s. The default is 10 ms.
Immediately when an event occurs	Event Task	An event task performs a function only when a specific event (trigger) occurs. Whenever the trigger for the event task occurs, the event task; - interrupts any lower priority tasks. - executes one time. - returns control to where the previous task left off. The trigger can be a; - change of a digital input. - new sample of analog data. - certain motion operations. - consumed tag. - EVENT instruction. Important: Some Logix5000 controllers do not support all triggers.

RSLOGIX 5000 CONTROLLER TASKSt

This example depicts execution of a project with three tasks

Task	Priority	Period	Execution time	Duration
Motion planner	N/A	8 ms (course update rate)	1 ms	1 ms
Event task 1	1	N/A	1 ms	1...2 ms
Periodic task 1	2	12 ms	2 ms	2...4 ms
I/O task—n/a to ControlLogix and SoftLogix controllers. See page 11.	7	5 ms (fastest RPI)	1 ms	1...5 ms
System overhead	N/A	Time slice = 20%	1 ms	1...6 ms
Continuous task	N/A	N/A	20 ms	48 ms

Legend:  Task executes.  Task is interrupted (suspended).



RSLOGIX 5000 CONTROLLER TASKS

Examples for using Tasks

Fill a tank to its maximum level and then open a drain valve.	Continuous task
Collect and process system parameters and send them to a display.	Continuous task
Complete step 3 in a control sequence—reposition the bin diverter.	Continuous task
Your system must check the position of a field arm each 0.1 s and calculate the average rate of change in its position. This is used to determine braking pressure.	Periodic task
Read the thickness of a paper roll every 20 ms.	Periodic task
A packaging line glues boxes closed. When a box arrives at the gluing position, the controller must immediately execute the gluing routine.	Event task
In a high-speed assembly operation, an optical sensor detects a certain type of reject. When the sensor detects a reject, the machine must immediately divert the reject.	Event task
In an engine test stand, you want to capture and archive each analog data immediately after each sample of data.	Event task
Immediately after receiving new production data, load the data into the station.	Event task
In a line that packages candy bars, you have to make sure that the perforation occurs in the correct location on each bar. Each time the registration sensor detects the registration mark, check the accuracy of an axis and perform any required adjustment.	Event task
A gluing station must adjust the amount of glue it applies to compensate for changes in the speed of the axis. After the motion planner executes, check the command speed of the axis and vary the amount of glue, if needed.	Event task
In a production line, if any of the programs detect an unsafe condition the entire line must shut down. The shutdown procedure is the same regardless of the unsafe condition.	Event task

RSLOGIX 5000 CONTROLLER TASKS

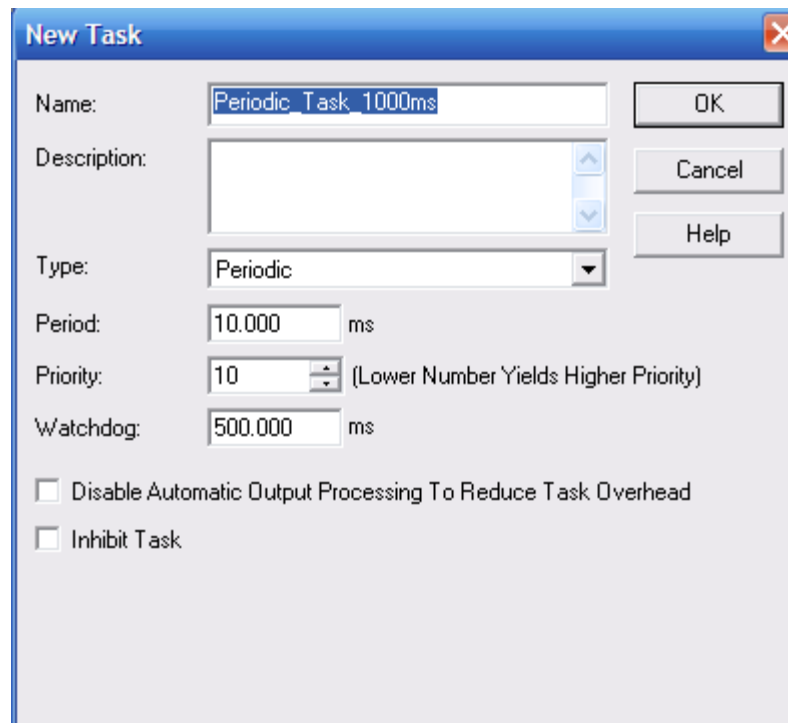
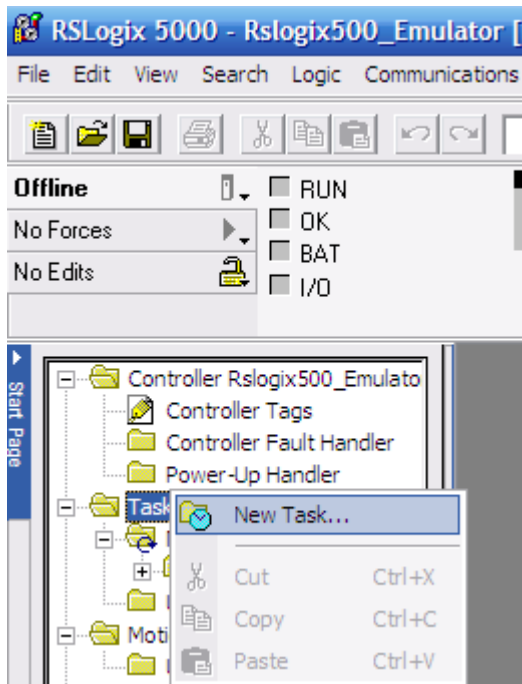
Priority Periodic and Event Tasks: The priority of each task tells the controller what to do

If you want	Then	Notes
This task to interrupt another task	Assign a priority number that is less than (higher priority) the priority number of the other task.	•A higher priority task interrupts all lower priority tasks. •A higher priority task can interrupt a lower priority task multiple times.
Another task to interrupt this task	Assign a priority number that is greater than (lower priority) the priority number of the other task.	
This task to share controller time with another task	Assign the same priority number to both tasks.	The controller switches back and forth between each task and executes each one for 1 ms.

PROGRAM FOR PERIODIC TASKS

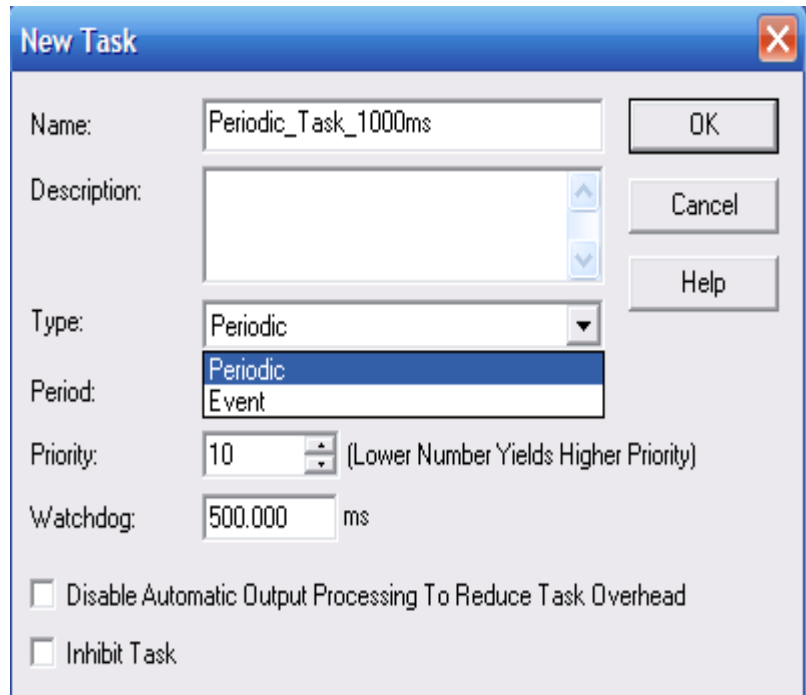
Creating a Periodic Task, Putting an appropriate name, selecting Task Type, Periodic and Priority, creating a program and writing a logic program

➤ **Creating a Periodic Task, enter an appropriate name**



PROGRAM FOR PERIODIC TASKS

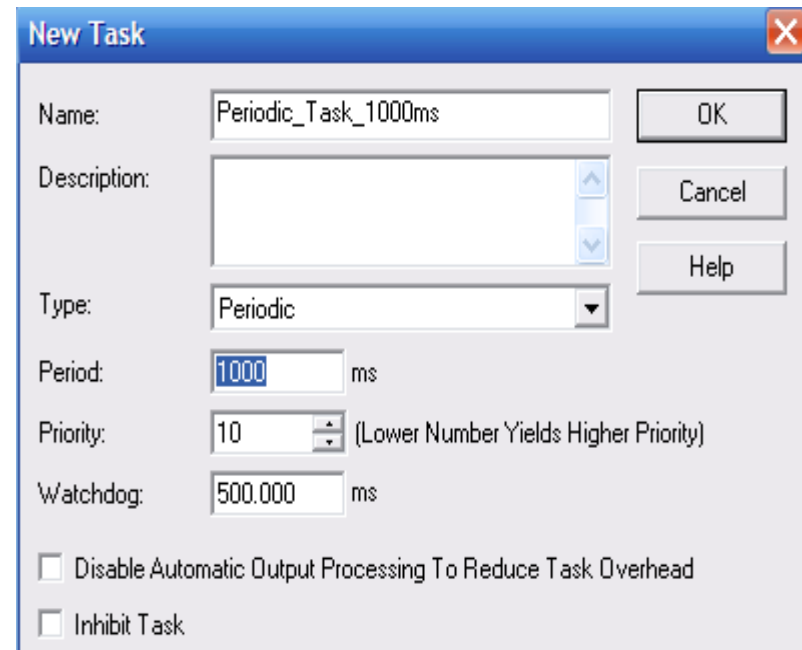
➤ Selecting Task Type, Periodic and Priority



The 'New Task' dialog box is shown with the following fields and options:

- Name: Periodic_Task_1000ms
- Description: (empty text box)
- Type: Periodic (selected in the dropdown menu)
- Period: (empty text box)
- Priority: 10 (Lower Number Yields Higher Priority)
- Watchdog: 500.000 ms
- ☐ Disable Automatic Output Processing To Reduce Task Overhead
- ☐ Inhibit Task

Buttons: OK, Cancel, Help



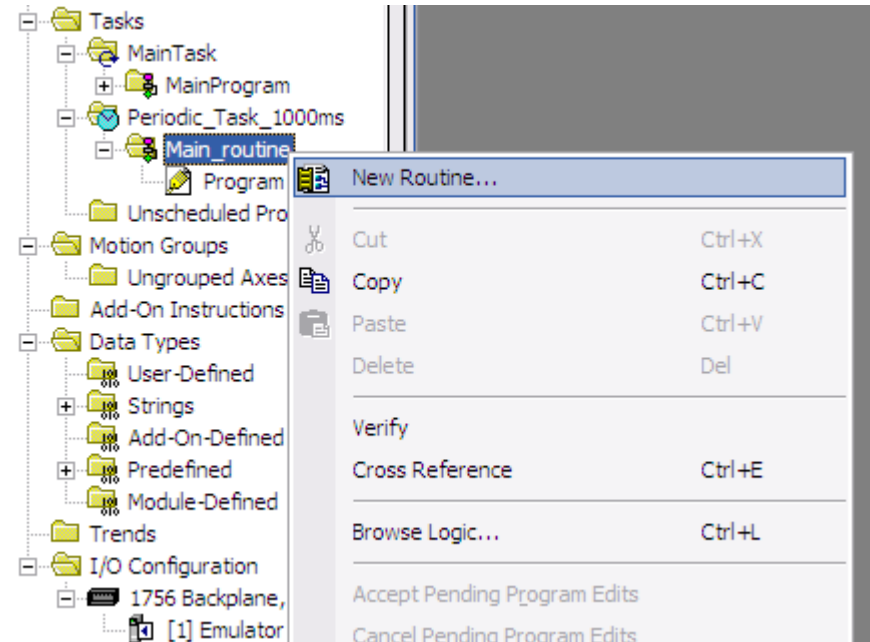
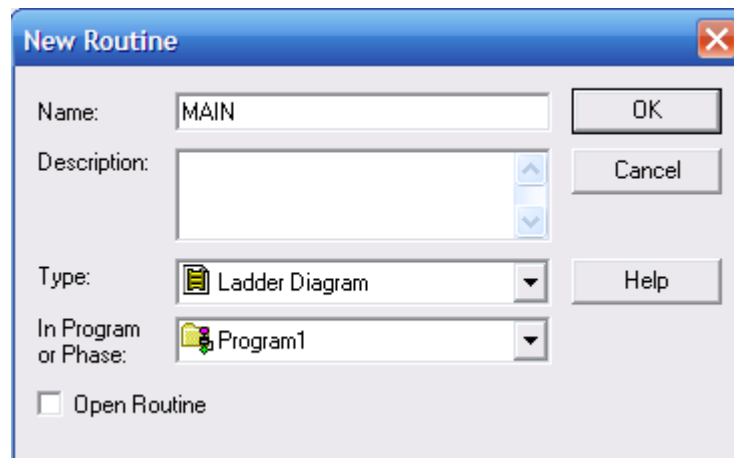
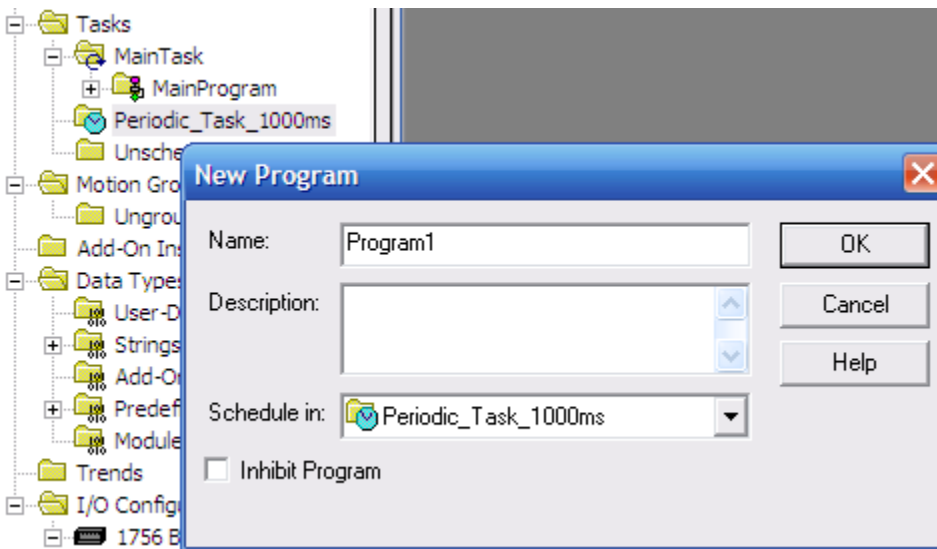
The 'New Task' dialog box is shown with the following fields and options:

- Name: Periodic_Task_1000ms
- Description: (empty text box)
- Type: Periodic (selected in the dropdown menu)
- Period: 1000 ms
- Priority: 10 (Lower Number Yields Higher Priority)
- Watchdog: 500.000 ms
- ☐ Disable Automatic Output Processing To Reduce Task Overhead
- ☐ Inhibit Task

Buttons: OK, Cancel, Help

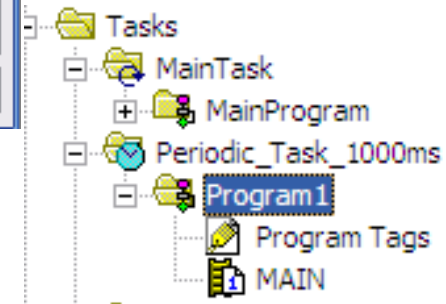
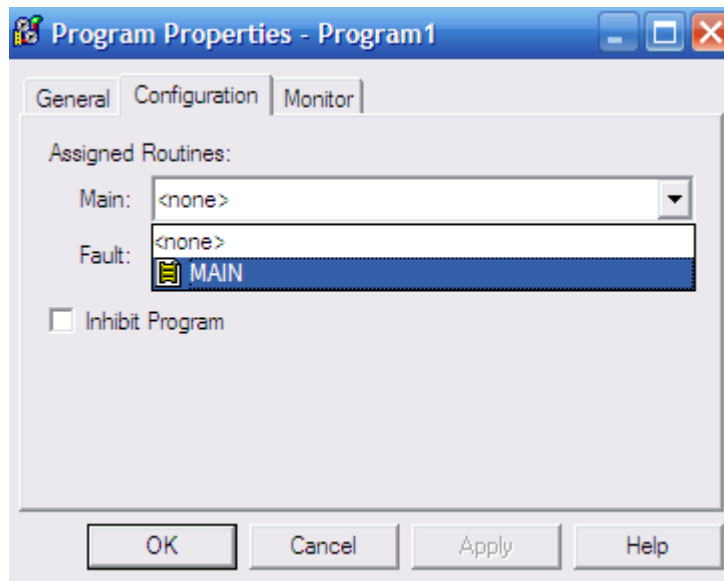
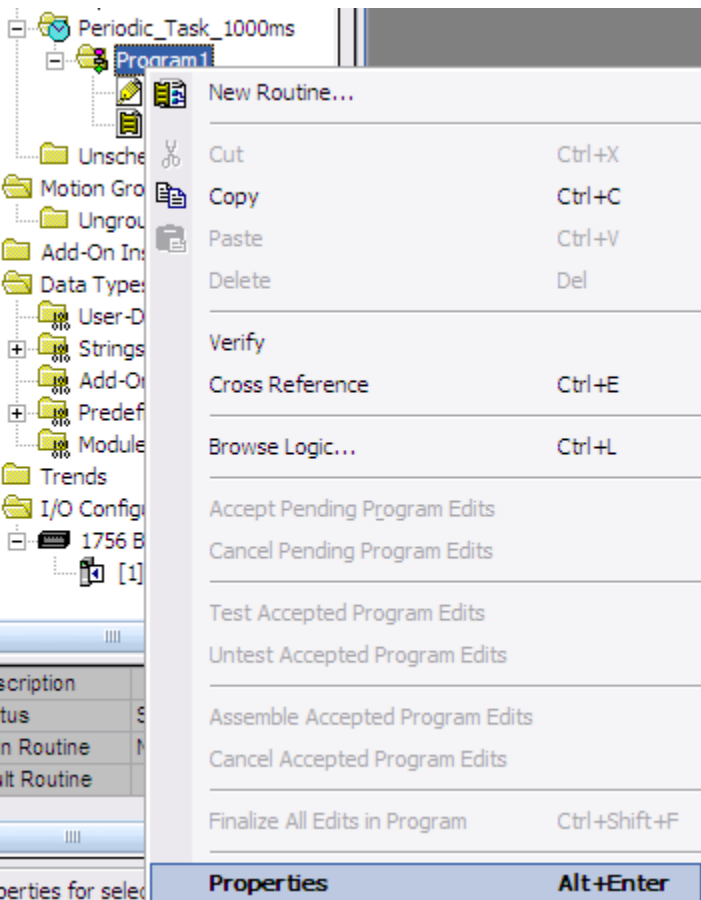
PROGRAM FOR PERIODIC TASKS

➤ **Creating a new Program with appropriate name and a new routine**



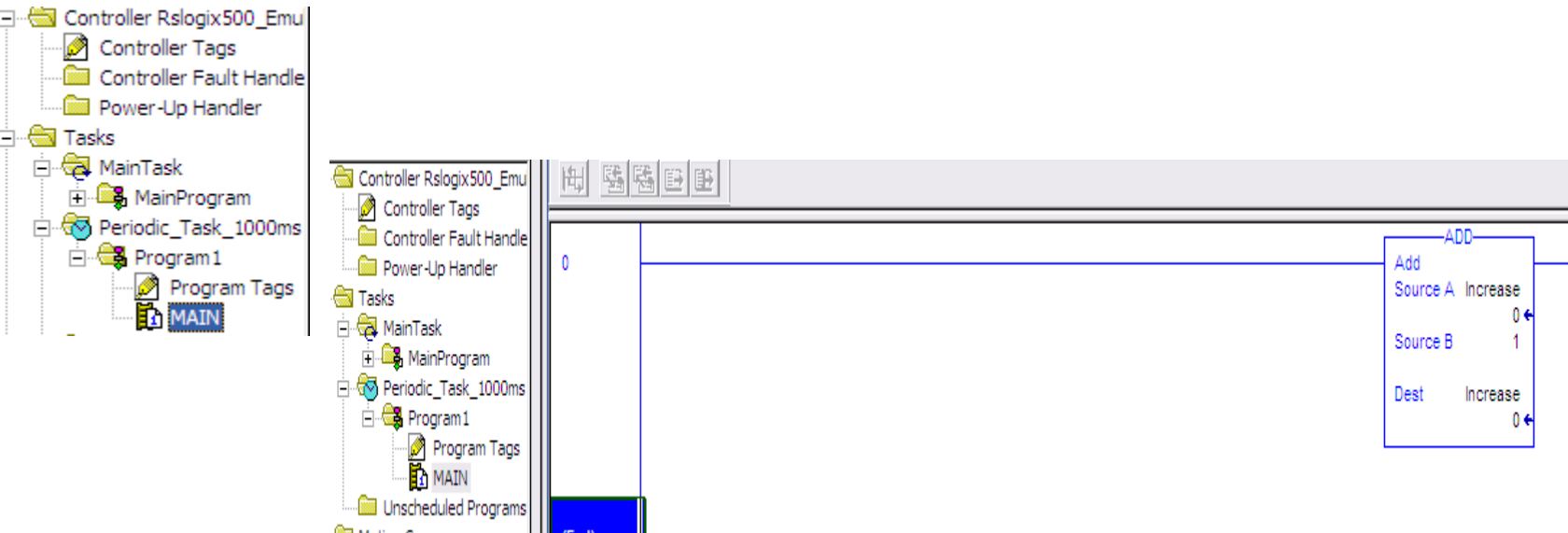
PROGRAM FOR PERIODIC TASKS

➤ Select Main Routine for writing logic program



PROGRAM FOR PERIODIC TASKS

➤ Selecting Main Routine for writing a Program



Add Instruction will executed one every 1000ms

MANAGE EVENT TASKS

Choosing the Trigger for an Event Task

To trigger an event task when	Use this trigger	With these considerations
Digital input turns On or Off	Module Input Data State Change	• Only one input module can trigger a specific event task. • The input module triggers the event task based on the change of state (COS) configuration for the module. The COS configuration defines which points prompt the module to produce data if they turn On or Off. This production of data (due to COS) triggers the event task. • Typically, enable COS for only one point on the module. If you enable COS for multiple points, a task overlap of the event task may occur.
Analog module samples data	Module Input Data State Change	• Only one input module can trigger a specific event task. • The analog module triggers the event task after each real time sample (RTS) of the channels. • All the channels of the module use the same RTS.
Controller gets new data via a consumed tag	Consumed Tag	• Only one consumed can trigger a specific event task. • Typically, use an IOT instruction in the producing controller to signal the production of new data. The IOT instruction sets an event trigger in the producing tag. This trigger passes to the consumed tag and triggers the event task. • When a consumed tag triggers an event task, the event task waits for all the data to arrive before the event task executes.
Specific condition or conditions occur within the logic of a program	EVENT instruction	Multiple EVENT instructions can trigger the same task. This lets you execute a task from different programs.

MANAGE EVENT TASKS

Module Input Data State Change Trigger

Let an event trigger this task. →

Let data from an input module trigger the task. →

Let this input tag trigger the task. →

When the task is done, do not update digital outputs in the local chassis. →

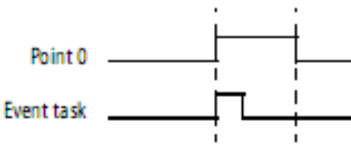
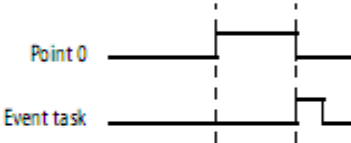
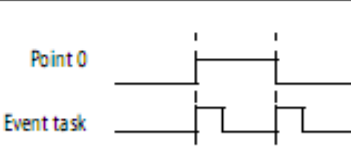
The screenshot shows the 'Task Properties - Task_1' dialog box with the 'Configuration' tab selected. The 'Type' is set to 'Event'. The 'Trigger' is set to 'Module Input Data State Change'. The 'Tag' is set to 'Local:4:I'. The 'Execute Task If No Event Occurs Within' is set to '1000.000' ms. The 'Priority' is set to '1' (Lower Number Yields Higher Priority). The 'Watchdog' is set to '500.000' ms. The checkbox 'Disable Automatic Output Processing To Reduce Task Overhead' is checked.

Property	Value
Type	Event
Trigger	Module Input Data State Change
Tag	Local:4:I
Execute Task If No Event Occurs Within	1000.000 ms
Priority	1 (Lower Number Yields Higher Priority)
Watchdog	500.000 ms
Disable Automatic Output Processing To Reduce Task Overhead	Checked

Event Task is triggered whenever data from input change

MANAGE EVENT TASKS

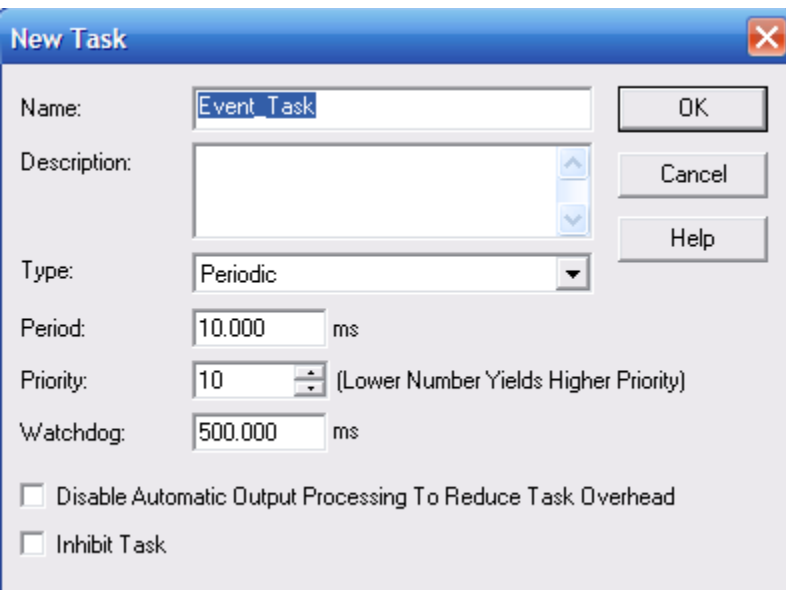
Choosing Trigger for Module Input State

If you want this	Then configure the input module like this (Point 0 is an example)												
	Change of State → No Change of State for Remaining Points → Module Properties: Local:1 (1756-IB32/B) General Connection Module Info Configuration Enable Change of State <table><thead><tr><th>Point</th><th>☐ Off -> On</th><th>☐ On -> Off</th></tr></thead><tbody><tr><td>0</td><td>☒</td><td>☐</td></tr><tr><td>1</td><td>☐</td><td>☐</td></tr><tr><td>2</td><td>☐</td><td>☐</td></tr></tbody></table>	Point	☐ Off -> On	☐ On -> Off	0	☒	☐	1	☐	☐	2	☐	☐
Point	☐ Off -> On	☐ On -> Off											
0	☒	☐											
1	☐	☐											
2	☐	☐											
	Change of State → No Change of State for Remaining Points → Module Properties: Local:1 (1756-IB32/B) General Connection Module Info Configuration Enable Change of State <table><thead><tr><th>Point</th><th>☐ Off -> On</th><th>☐ On -> Off</th></tr></thead><tbody><tr><td>0</td><td>☐</td><td>☒</td></tr><tr><td>1</td><td>☐</td><td>☐</td></tr><tr><td>2</td><td>☐</td><td>☐</td></tr></tbody></table>	Point	☐ Off -> On	☐ On -> Off	0	☐	☒	1	☐	☐	2	☐	☐
Point	☐ Off -> On	☐ On -> Off											
0	☐	☒											
1	☐	☐											
2	☐	☐											
	Change of State → No Change of State for Remaining Points → Module Properties: Local:1 (1756-IB32/B) General Connection Module Info Configuration Enable Change of State <table><thead><tr><th>Point</th><th>☐ Off -> On</th><th>☐ On -> Off</th></tr></thead><tbody><tr><td>0</td><td>☒</td><td>☒</td></tr><tr><td>1</td><td>☐</td><td>☐</td></tr><tr><td>2</td><td>☐</td><td>☐</td></tr></tbody></table>	Point	☐ Off -> On	☐ On -> Off	0	☒	☒	1	☐	☐	2	☐	☐
Point	☐ Off -> On	☐ On -> Off											
0	☒	☒											
1	☐	☐											
2	☐	☐											

PROGRAM FOR EVENT TASKS

Creating an *Event Task*, enter an appropriate name, Selecting Task Type, event
And Priority, creating a Program and writing a logic program

➤ Creating an **Event Task**, enter an appropriate **name**, **Type of Task** , **Trigger** and **Priority**



The 'New Task' dialog box shows the configuration for a periodic task. The 'Name' field is 'Event_Task'. The 'Type' is set to 'Periodic'. The 'Period' is 10.000 ms. The 'Priority' is 10, with a note '(Lower Number Yields Higher Priority)'. The 'Watchdog' is 500.000 ms. There are checkboxes for 'Disable Automatic Output Processing To Reduce Task Overhead' and 'Inhibit Task', both of which are currently unchecked. Buttons for 'OK', 'Cancel', and 'Help' are on the right.

Name: Event_Task

Description:

Type: Periodic

Period: 10.000 ms

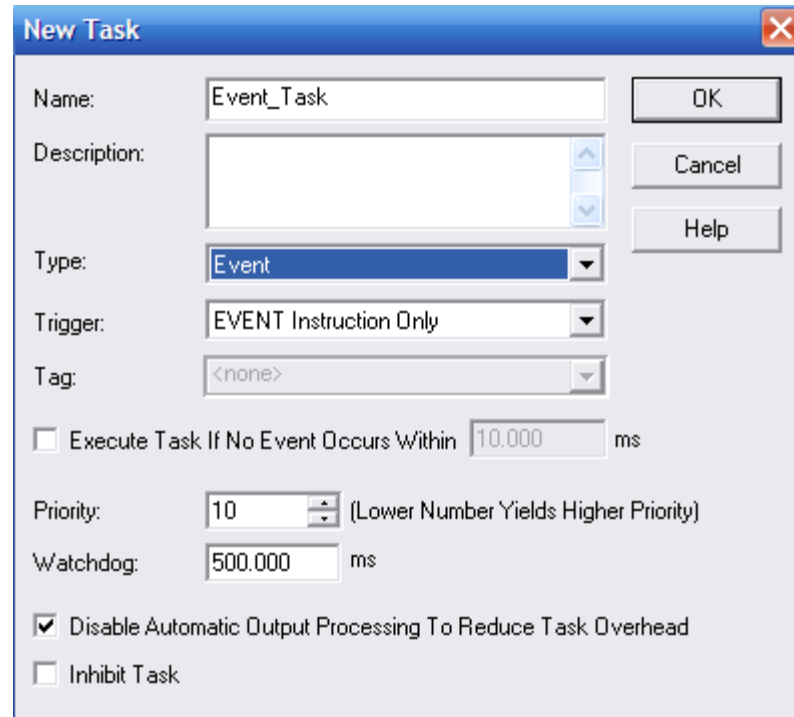
Priority: 10 (Lower Number Yields Higher Priority)

Watchdog: 500.000 ms

☐ Disable Automatic Output Processing To Reduce Task Overhead

☐ Inhibit Task

OK Cancel Help



The 'New Task' dialog box shows the configuration for an event task. The 'Name' field is 'Event_Task'. The 'Type' is set to 'Event'. The 'Trigger' is 'EVENT Instruction Only'. The 'Tag' is '<none>'. There is a checkbox for 'Execute Task If No Event Occurs Within 10.000 ms', which is unchecked. The 'Priority' is 10, with a note '(Lower Number Yields Higher Priority)'. The 'Watchdog' is 500.000 ms. There are checkboxes for 'Disable Automatic Output Processing To Reduce Task Overhead' (checked) and 'Inhibit Task' (unchecked). Buttons for 'OK', 'Cancel', and 'Help' are on the right.

Name: Event_Task

Description:

Type: Event

Trigger: EVENT Instruction Only

Tag: <none>

☐ Execute Task If No Event Occurs Within 10.000 ms

Priority: 10 (Lower Number Yields Higher Priority)

Watchdog: 500.000 ms

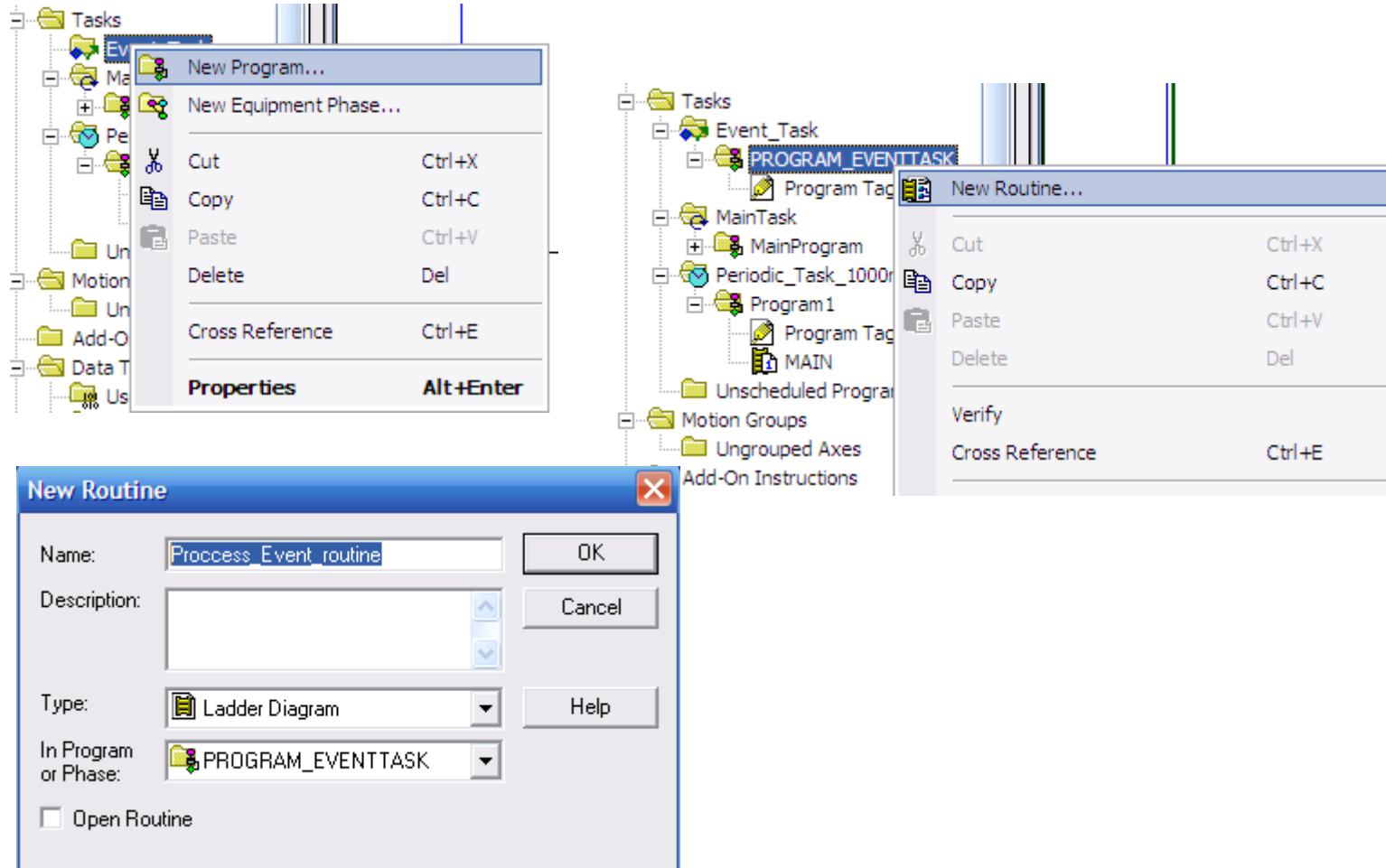
☒ Disable Automatic Output Processing To Reduce Task Overhead

☐ Inhibit Task

OK Cancel Help

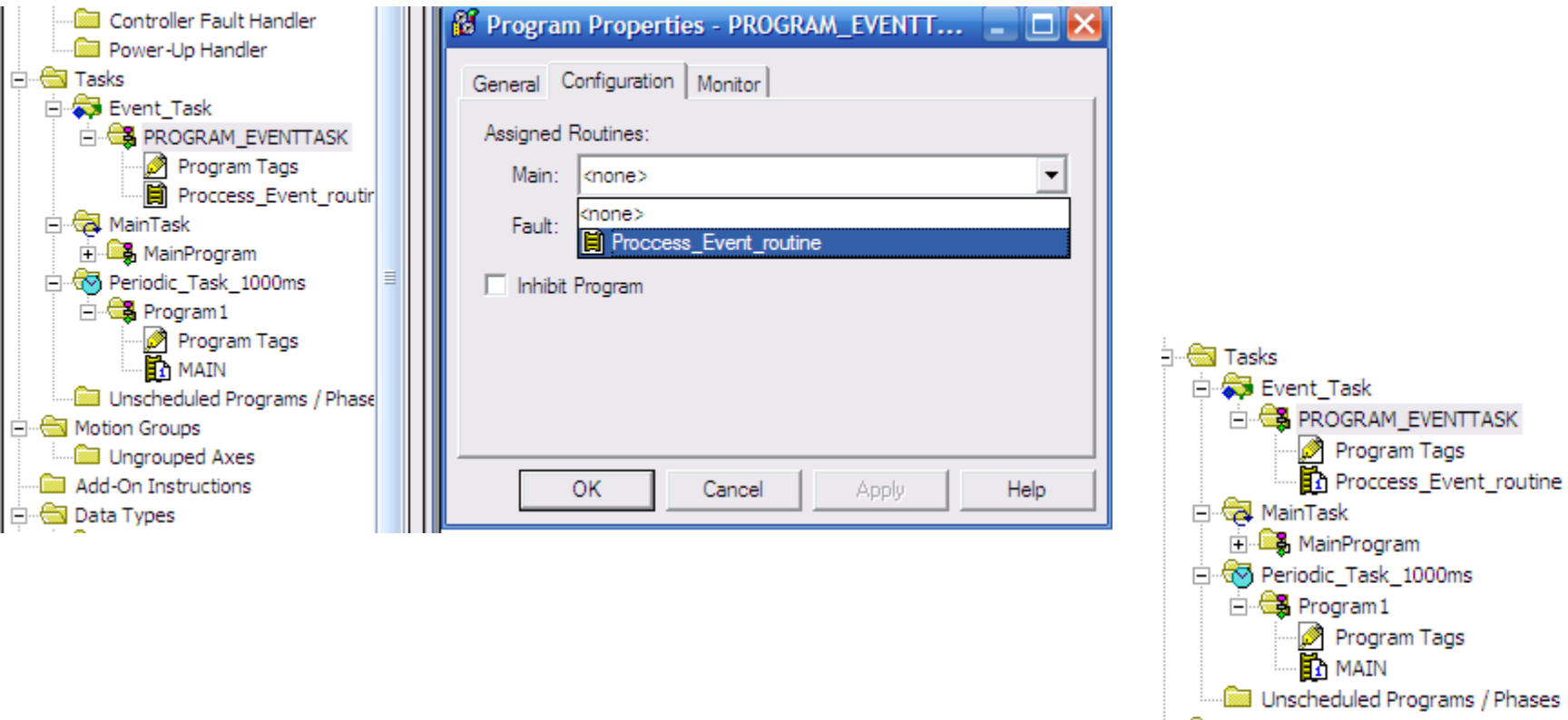
PROGRAM FOR EVENT TASKS

➤ Creating a new Program with appropriate name and a new routine



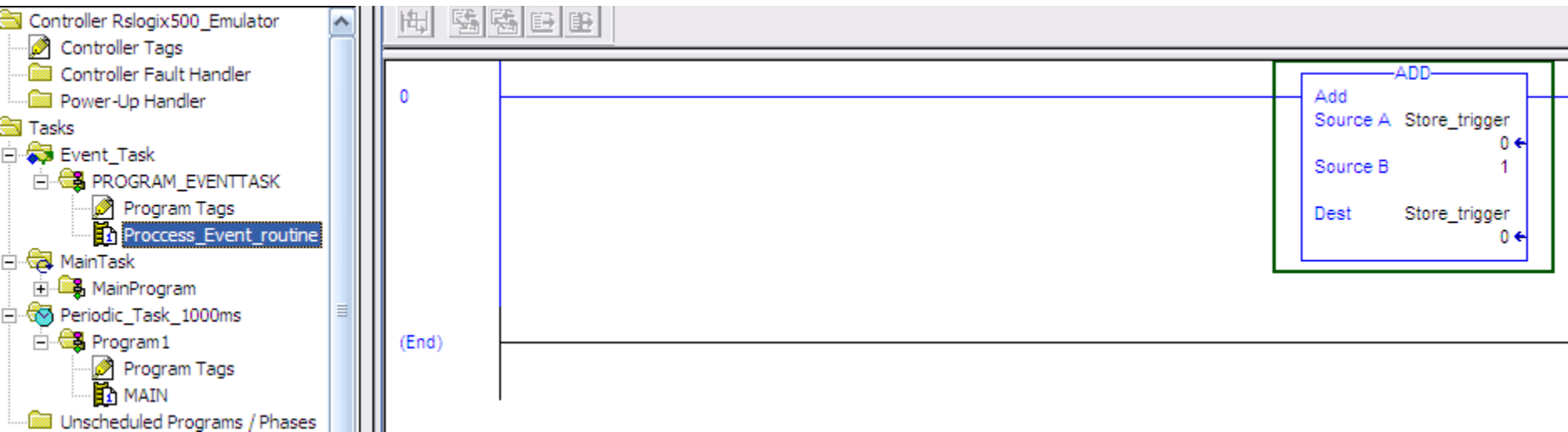
PROGRAM FOR EVENT TASKS

➤ Selecting Main Routine in Event Task to write logic program



PROGRAM FOR EVENT TASKS

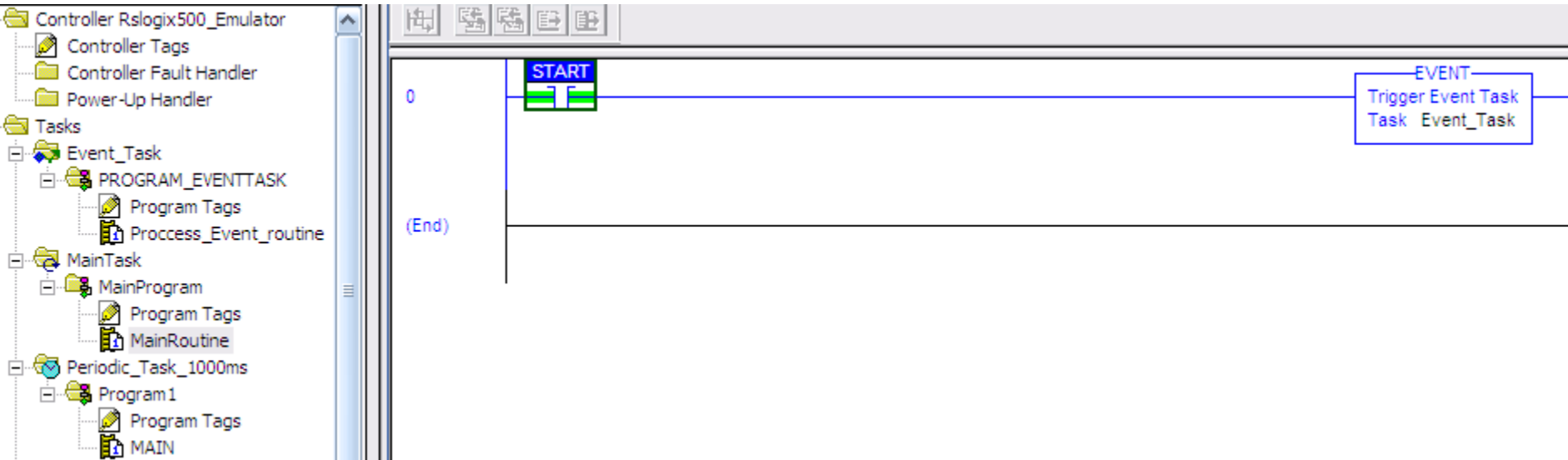
➤ Selecting Main Routine in Event Task to write a Program



Add Instruction will executed whenever Event Task is Called

PROGRAM FOR EVENT TASKS

Use Trigger Event Instruction to call Event_Task



Trigger Task Instruction is placed in another Task.