Basic Control of SMVector over Ethernet IP using RSLogix™5000

This application example illustrates the basic control of an SMVector Drive over Ethernet IP using an RSLogix™5000 programmed PLC. A CompactLogix® controller was used for the development of this application and its export file is attached to this application note. The application cycles the drive between two moves: First a 60 second forward move at 40Hz with a 1 second acceleration and a 5 second deceleration, and then a second move for 30 seconds at 60Hz in the reverse direction with a 5 second acceleration and a 1 second deceleration.

This logic example uses Assemblies 100 and 101 to control the SMVector drive over Ethernet IP.

The controller tags in this project have aliasing to break out the relevant SMV data from the assemblies.

Output Assembly 100 - Speed (Hz) & Digital and Analog Output

Word 0	Bit 0	0 = NOT Run Forward 1 = Run Forward			
	Bit 1	0 = NOT Run Reverse 1 = Run Reverse			
	Bit 2	Fault reset on transition from 0 to 1			
	Bit 3	Reserved			
	Bit 4	Reserved			
	Bit 5	0 = Local Control 1 = Network Control			
	Bit 6	0 = Local Speed reference 1 = Network Speed reference			
	Bit 7	Reserved			
	Bit 8 Bit 9 Bit 10 Bit 11	Network Speed reference (valid when bit 6 set) 0 – Network 1 – keypad 2 – 0-10VDC	3 – 4-20mA 4 – Preset #1 5 – Preset #2	6 – Preset #3 7 – Preset #4 8 – Preset #5	9 – Preset #6 10 – Preset #7 11 – MOP
	Bit 12	0 = No Action 1 = Inhibit (Coast to STOP)			
	Bit 13	0 = No Action 1 = Activate Quick STOP			
	Bit 14	0 = No Action 1 = Force Manual Mode (active only in Network Control, in PID mode will force open loop)			
	Bit 15	0 = DC brake active 1 = DC brake NOT active			
Word 1	Unsigned speed 0.1Hz resolution • received value = 0x01F0 = 49.6Hz				
Word 2	Digital Output + Relay – Active when parameter P140, P142 = 25 Network Control Bit 9 – Open Collector Bit 10 - Relay Others – reserved for future use				
Word 3	Analog Output [0.01VDC] – Active when parameter P150 = 9 Network Control • received value = 0x024B = 5.87[VDC]				

Input Assembly 101 - Speed (Hz) & Digital and Analog Input

Word 0	Bit 0	1 = Faulted			
	Bit 1	Reserved			
	Bit 2	1 = Running Forward			
	Bit 3	1= Running Reverse			
	Bit 4	1 = Ready			
	Bit 5	0 = Local Control 1 = Control from Network			
	Bit 6	0 = Local reference 1 = Reference from Network			
	Bit 7	1 = At reference			
	Bit 8 Bit 9 Bit 10 Bit 11	Actual set point source: 0 – keypad 1 – 0-10VDC 2 – 4-20mA	3 – Preset #1 4 – Preset #2 5 – Preset #3	6 – Preset #4 7 – Preset #5 8 – Preset #6	9 – Preset #7 10 – MOP 11 – Network
	Bit 12	1 = PID Active (closed loop)			
	Bit 13	1 = Torque mode active			
	Bit 14	1 = Current limit			
	Bit 15	1 = DC Braking			
Word 1	Unsigned actual frequency 0.1Hz resolution.				
Word 2	Digital Input/Output states (See Note 1 for details)				
Word 3	Analog Input 0-10V TB [0.01VDC] • received value = 0x024B = 5.87[VDC]				

The following parameters must be set first in the SMV:

- P100=3
- P101=6
- P111=2 to enable the deceleration logic to function
- P112=1
- P121=9
- P140=14
- P142=14
- P400=5
- P410-413 = 192.168.124.16
- P414-417 = 255.255.255.0
- Both TB1 and TB13A need to be asserted prior to running this logic.

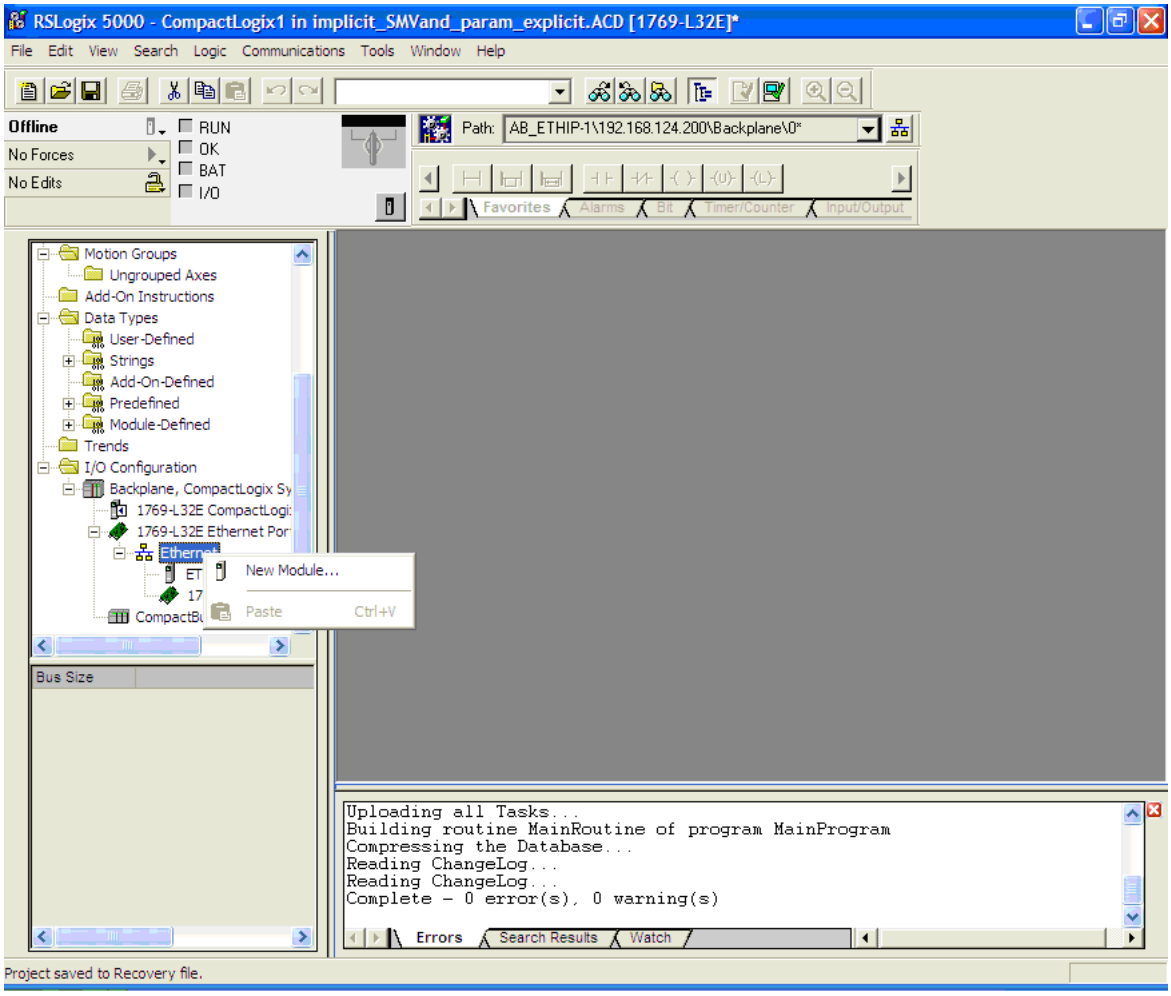
Ensure the SMVector drive is power cycled after programming prior to running this PLC application to ensure proper operation.

To map the drive to an Ethernet IP scanner in RSLogix™ 5000 for implicit messaging:

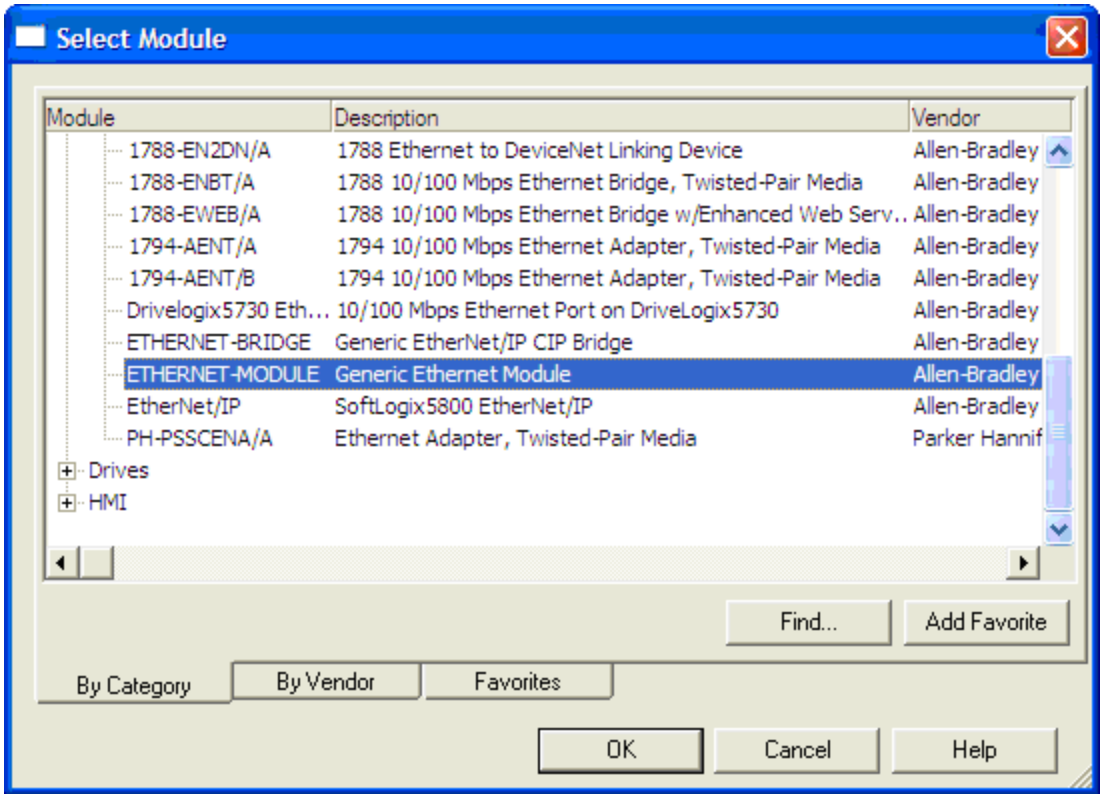
Click the [I/O Configuration] folder in the left-hand navigation window

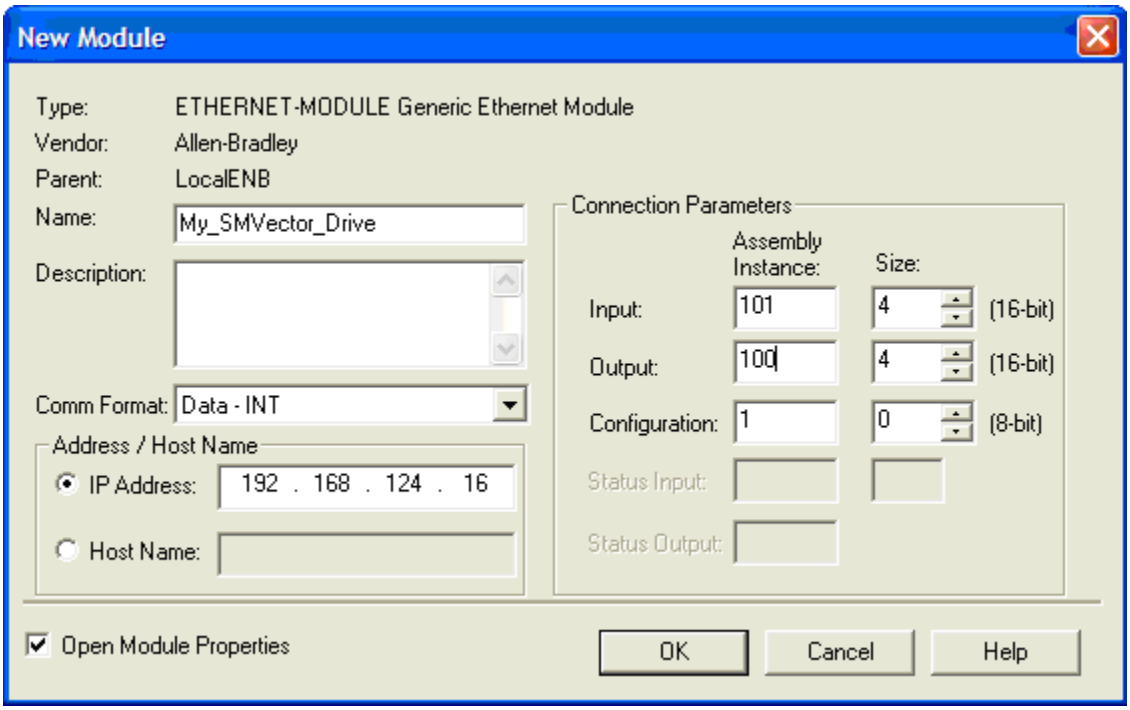
Click the appropriate Ethernet Port folder, [1769-L32E Ethernet Port] in this example.

Right click on the [Ethernet] network icon and select [New Module].



Expand “Communications” and select [ETHERNET-MODULE Generic Ethernet Module].





Enter a name for the drive -usually relating to the process (i.e. booster_pump_4, or an equipment tag number such as PP105).

Enter the IP address of the SMV drive. Ensure that it is on the same subnet as the PLC (the first 3 octets of the IP address match).

For this basic application, enter “Data – INT” for the Comm format.

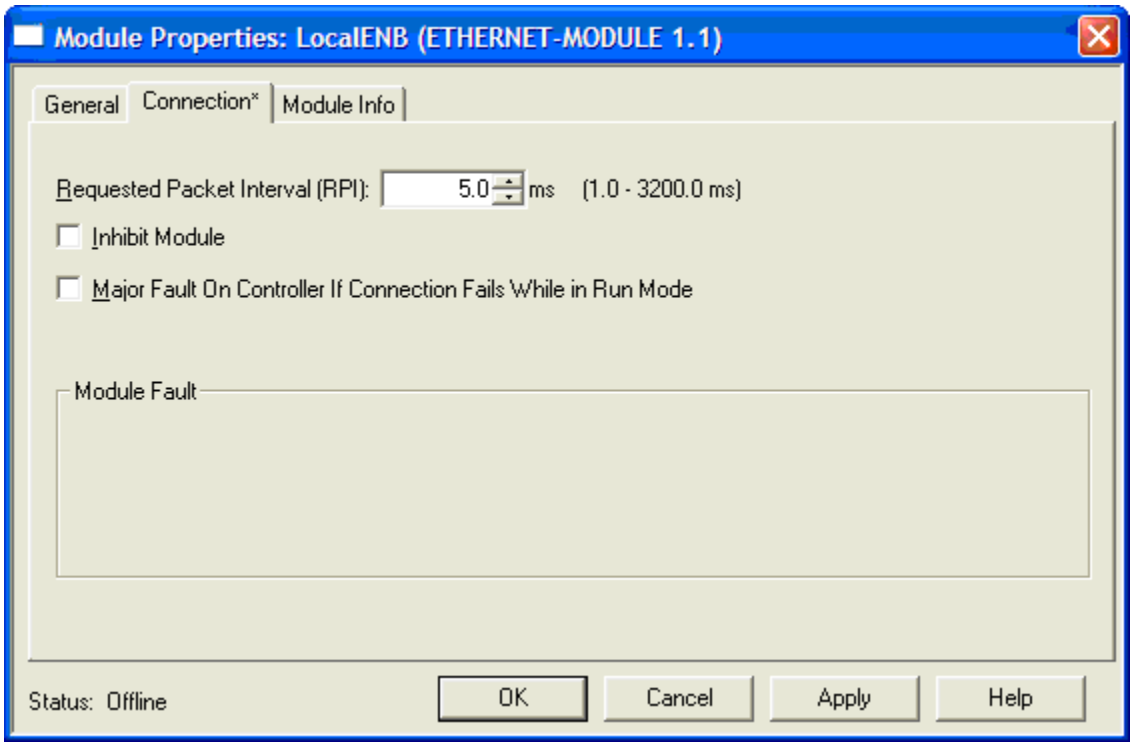
Enter the desired Input and Output Assembly numbers and their corresponding lengths. Remember the size must be set to the number of words that actually make up the assembly you want to use.

For this example application we will use Assemblies 101 (Input) and 100 (Output).

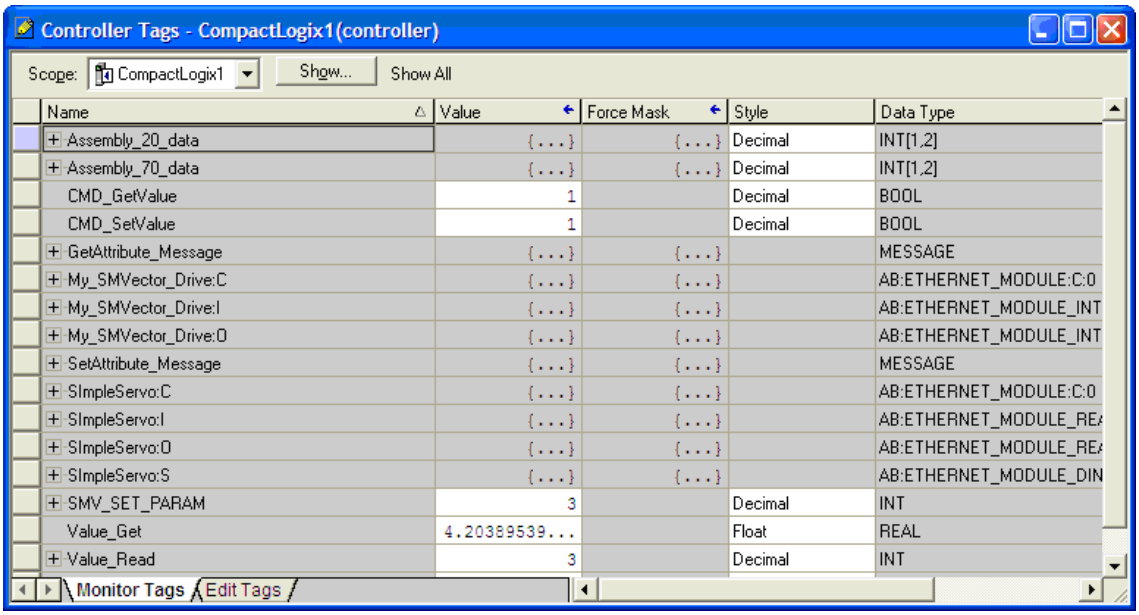
For Configuration enter assembly instance 1 and a size of 0. This value is required.

Under the connection tag enter the desired RPI rate. This is how frequently the drive will be polled by the PLC. The minimum recommended value is 5.0 milliseconds.

From this screen you can also optionally set the controller to fault if the Ethernet IP connection is lost to the drive while the controller is running. This selection is the [Major Fault On Controller If Connection Fails While in Run Mode].



The corresponding tags will then be created in the controller tags of the project as shown herein.



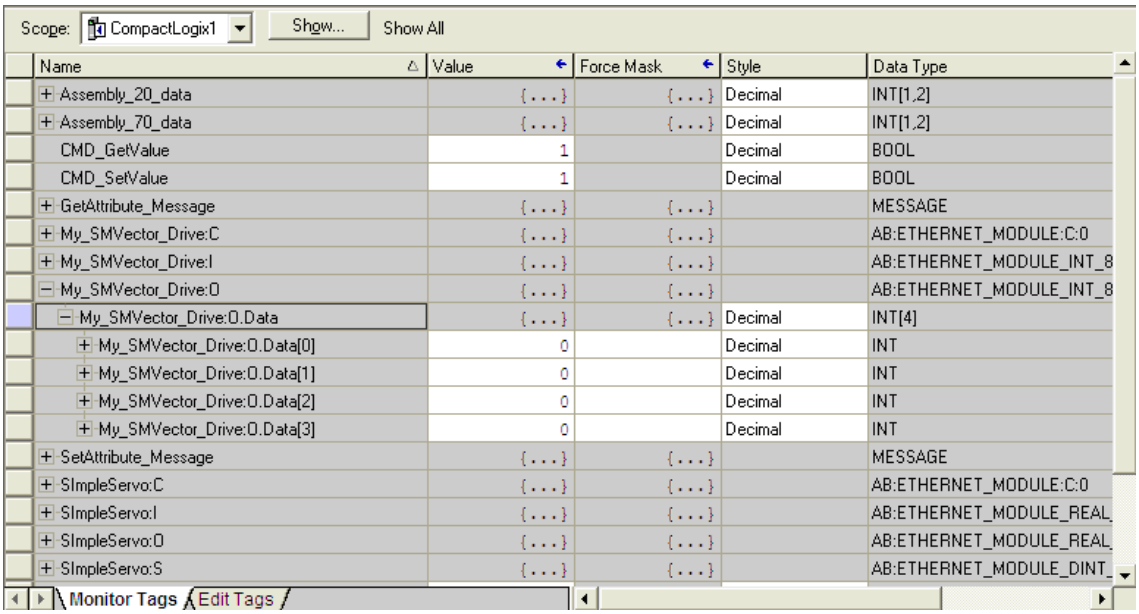
To understand this from the above configuration we named the drive “My_SMVector_Drive”. There are three sets of tags labeled “My_SMVector_Drive”:

[:C] for the Configuration assembly (1)

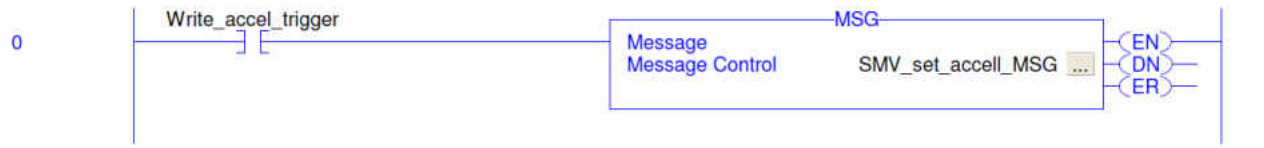
[:I] for the Input Assembly (101 in this example)

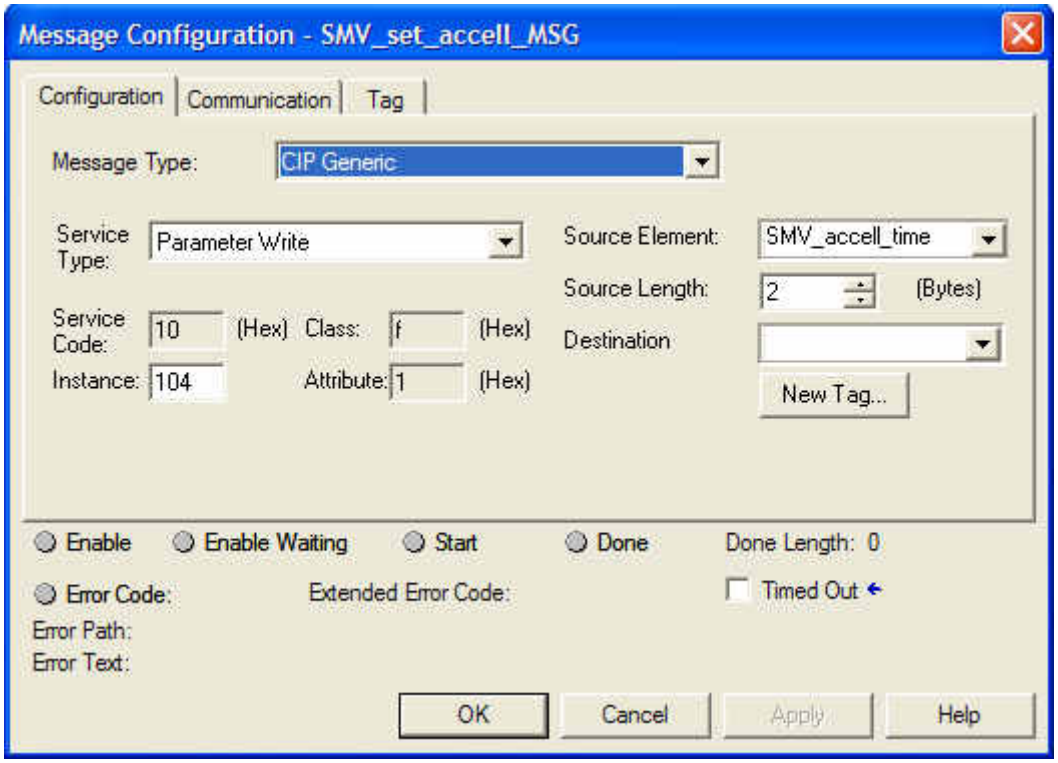
[:O] for the Output assembly (100 in this example)

Click on the [+] and expand the [My_SMVector_Drive:O] data to reveal all four words that make up the Output assembly.

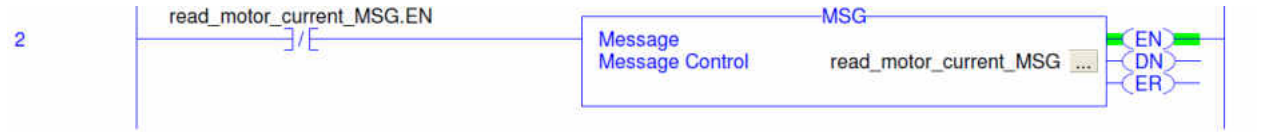


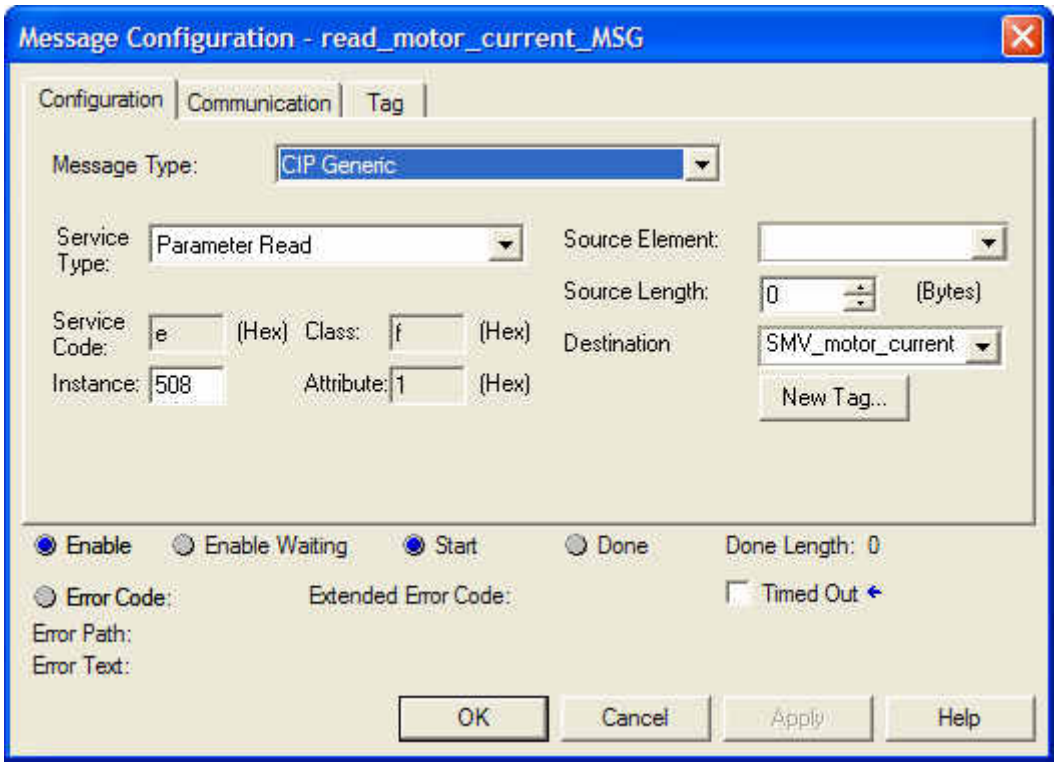
Rung 0 illustrates the basic MSG instruction to write to the drive's Acceleration time (Parameter 104)



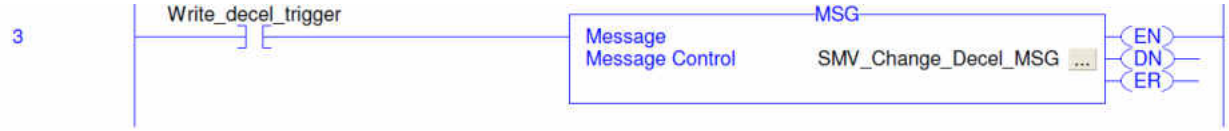


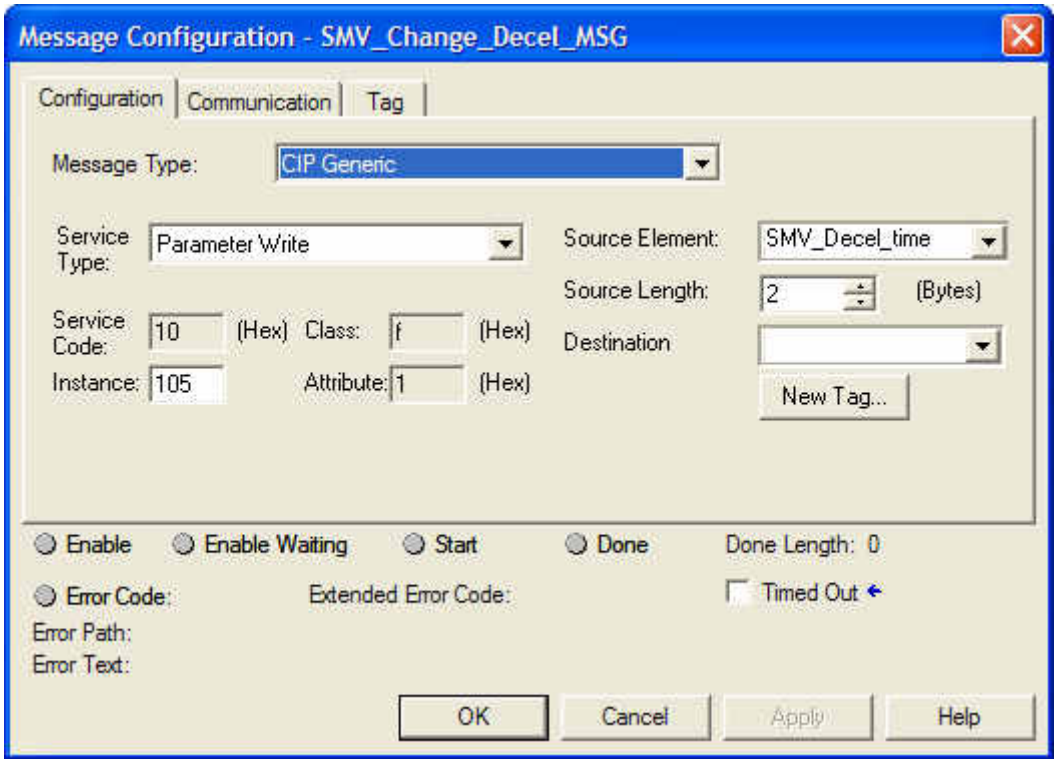
Here we use the Examine if Off (Normally Closed Contact) expression evaluating the MSG's own enable bit so that it will continuously read from the SMV the Motor Current (Parameter 508)





This MSG changes the value of the drive's DECEL time (parameter 105)

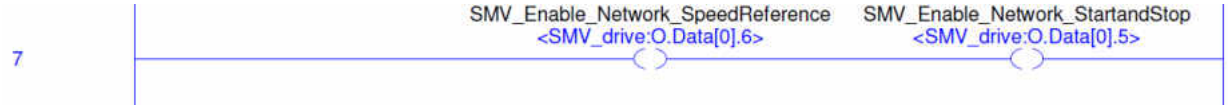




As the output frequency command in the drive assembly 100 is an integer value in tenths of a Hz (i.e. 412 =41.2 Hz), this MUL instruction converts the REAL format value (Source A) to the scaled integer value for use in the drive assembly by multiplying by 10 (Source B).

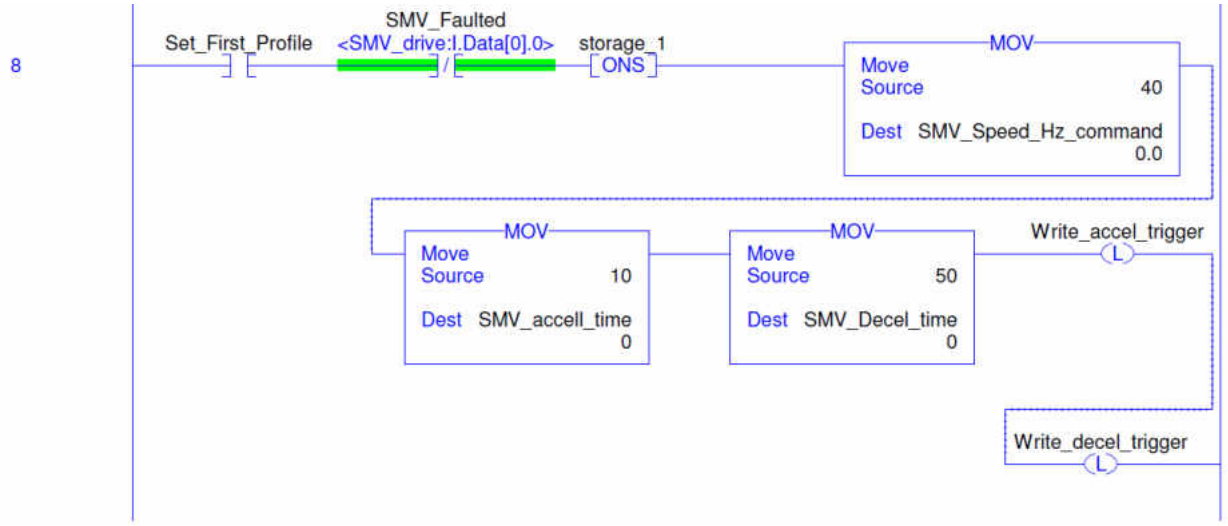
As the drive's actual frequency in assembly 101 is an integer value in tenths of a Hz (i.e. 412 =41.2 Hz), This DIV instruction converts the scaled integer value from the drive assembly (Source A) to a REAL value for use in the PLC by dividing by 10 (Source B).

This rung forces bits 5 and 6 of output assembly 100 to be on. This is required for network based start/stop and speed reference for the SMV.



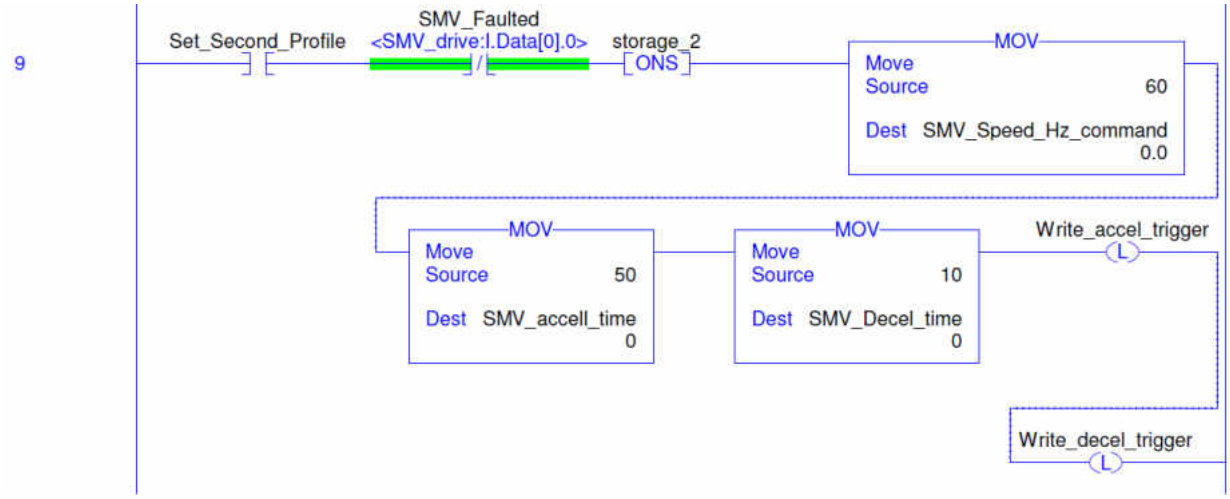
This rung loads target velocity (to SMV_Speed_Hz_command), then accel (to SMV_accel_time), and then decel (to SMV_Decel_time) values to create the first move profile of the program.

Note that the SMV_Speed_Hz_command is an alias in the controller tag for Assembly 100 word 1

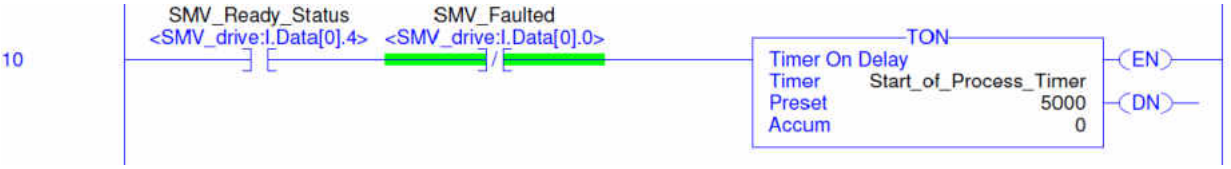


This rung loads target velocity (to SMV_Speed_Hz_command), then accel (to SMV_accel_time), and then decel (to SMV_Decel_time) values to create the second move profile of the program.

Note that the SMV_Speed_Hz_command is an alias in the controller tag for Assembly 100 word 1



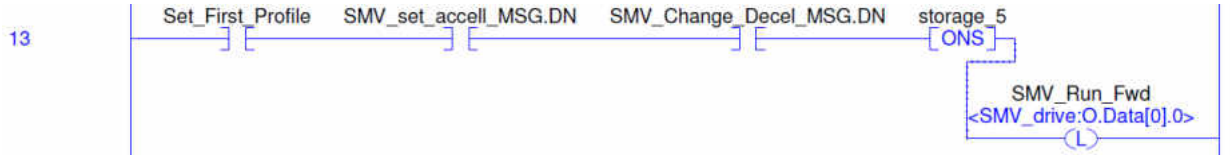
Note here that we first check to ensure that the drive reports a READY status and ensures the drive is not faulted prior to starting a 5 second timer and initiating the process. The timer allows for more than ample time for the I/O to become active at the start of the process.



Note that this logic checks to ensure that TB1 and TB13A are both asserted prior to starting the process. Both are required using the SMV Parameter settings listed at the top of this file.

Latch and Unlatch coils are used to control the program execution.

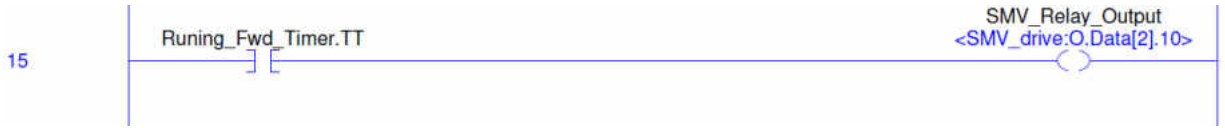
The .DN outputs of the MSG instructions are tested to ensure the accel and decel values have been loaded prior to starting this first move (forward).



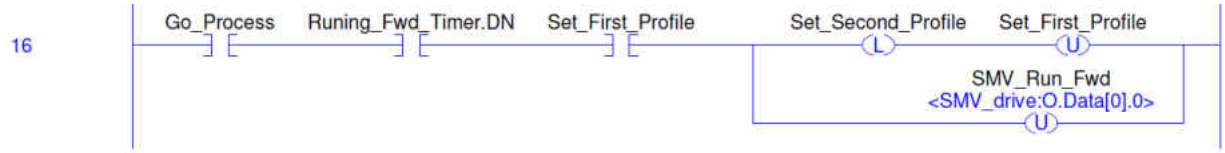
This rung starts the 60 second timer governing the first move.

Note SMV_at_set_speed status is aliased from Assembly 101 word 0 bit 7. This is checked to ensure the timer does not begin accumulating time until the drive has reached the commanded speed.

The .TT tag of the Running_Fwd_Timer is also used in this example to trigger the SMV drive's relay output. This output will be on while the drive is running forward at the set speed. It will not turn on until the acceleration is complete and will turn off prior to decelerating



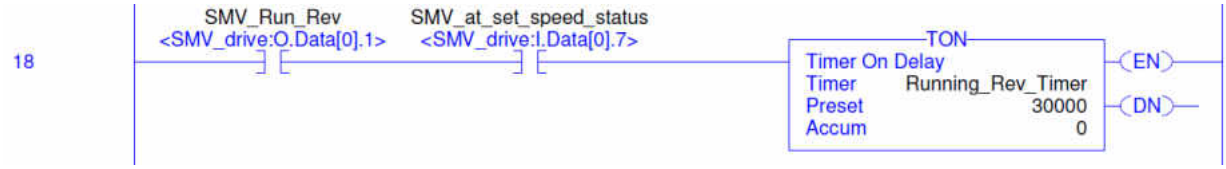
Once the Running_Fwd_Timer times out, the drive is stopped and the second move profile is sent to the drive (reference rung 9).



The .DN outputs of the MSG instructions are tested to ensure the accel and decel values have been loaded prior to starting this second move (reverse).

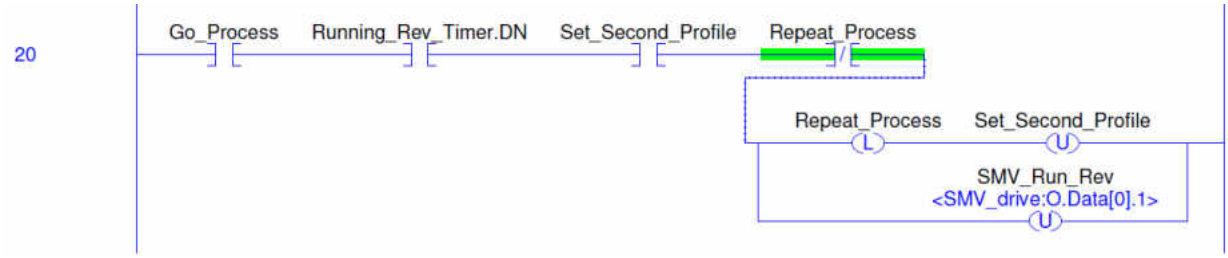
This rung starts the 30 second timer governing the second move.

Note SMV_at_set_speed status is aliased from Assembly 101 word 0 bit 7. This is checked to ensure the timer does not begin accumulating time until the drive has reached the commanded speed.



The .TT tag of the Running_Rev_Timer is also used in this example to trigger the SMV drive's open collector output. This output will be on while the drive is running reverse at the set speed. It will not turn on until the acceleration is complete and will turn off prior to decelerating

Once the Running_Rev_Timer times out, the drive is stopped and the process is repeated (reference rung 12).



In the event of a fault we turn off all process flow logic and enable TB13C to be read to reset the drive fault. The SMV_Fault_reset tag is aliased from Assembly 0 word 0 bit 2.