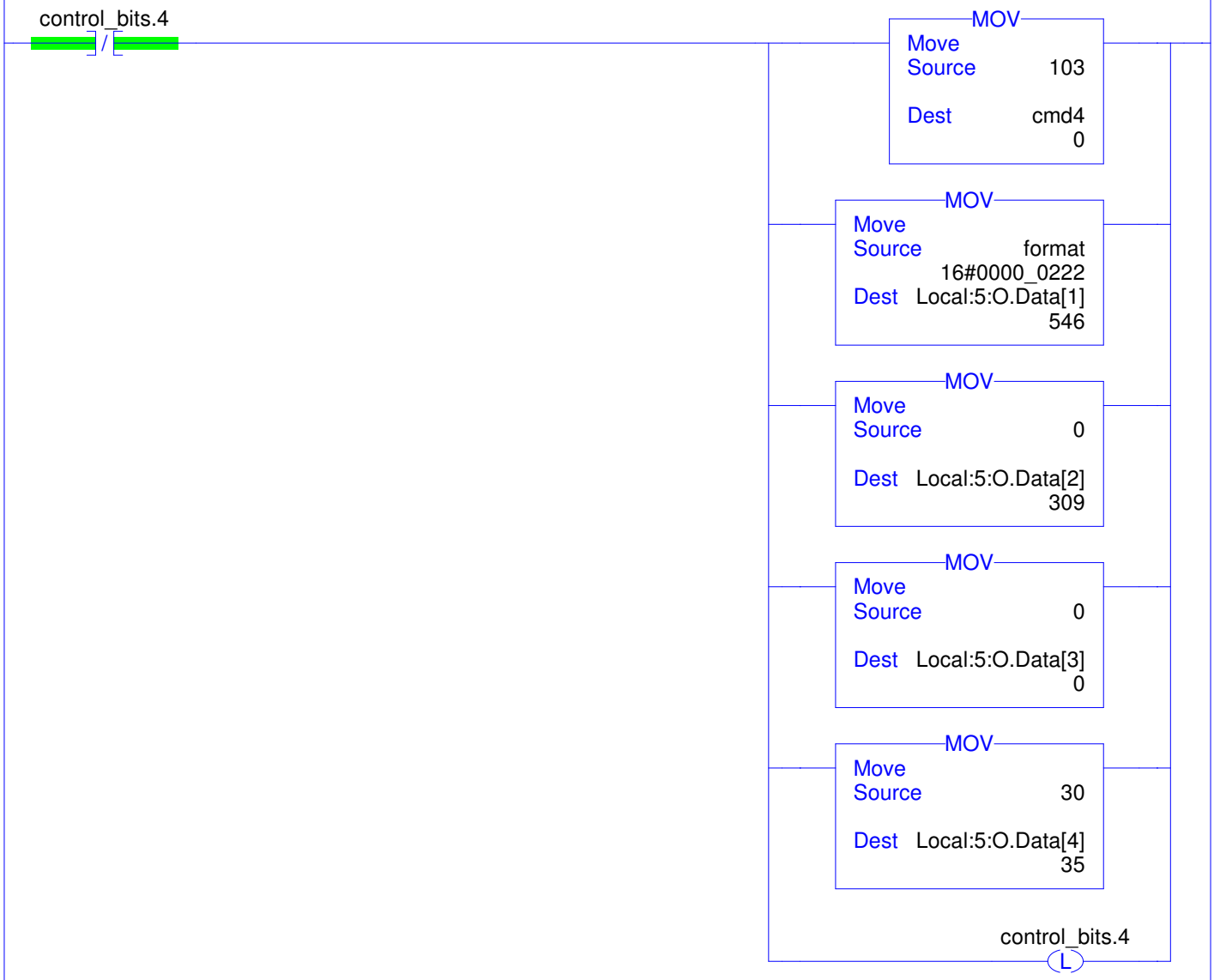
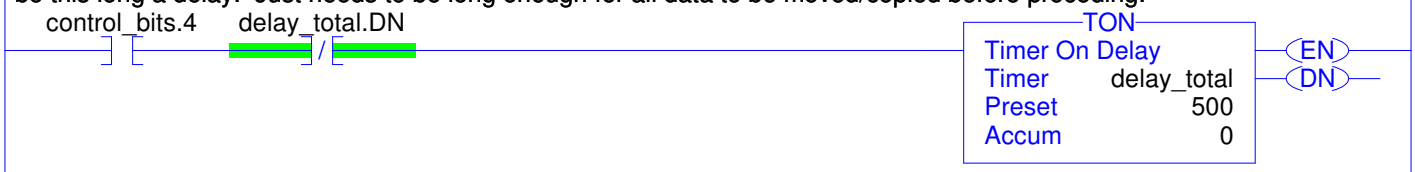
When the subroutine is triggered, this rung will move the write command, 103, to a com4 tag. It also moves the format value, xyz hex, Total parameter ID, 30, and the new total value, 0, into the Local:x:O table. Once triggered, the rung is blocked to prevent running again.

0



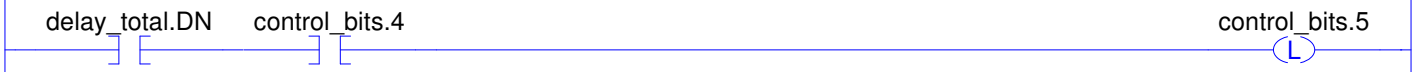
The ladder logic seems to run very fast, so once the routine is triggered and we start moving data around, I placed a delay here to prevent commands being sent before all the data movement is done. I set for 1 second, but do not think it needs to be this long a delay. Just needs to be long enough for all data to be moved/copied before proceding.

1



Once the delay time has elapsed, will latch a bit on to start next stage of command.

2



This rung will trigger the subroutine that actually runs the command and sends it to the module. This subroutine needs a command number as an input parameter. The command number in cmd4 tag will be given as the input parameter.

3

control_bits.5

JSR
Jump To Subroutine
Routine Name run_cmd
Input Par cmd4

The subroutine will turn on a bit indicating when it is done. Once that bit, cmd_sel.15, is set, this rung will then clear tags, reset delay timers, copy status and clear bits that triggered the routine.

4

cmd_sel.15

MOV
Move
Source 0
Dest cmd4
0

COP
Copy File
Source return_status.Data_0
Dest total
Length 1

delay_total
RES

MOV
Move
Source 0
Dest control_bits
0

MOV
Move
Source 0
Dest cmd_sel
0

clear_total
U

(End)

This routine will allow the user to trigger the commands to be sent to the module. Each of the first rungs, 0 – 12, will trigge

0

This rung will trigger the Start command.

cmd_sel.0

MOV
Move
Source 1
Dest cmd 0

1

This rung will trigger the Stop command.

cmd_sel.1

MOV
Move
Source 2
Dest cmd 0

2

This rung will trigger the Refill command.

cmd_sel.2

MOV
Move
Source 3
Dest cmd 0

3

This rung will trigger the Write Non-Volatile Command.

cmd_sel.3

MOV
Move
Source 4
Dest cmd 0

4

This rung will trigger the Manual Command.

cmd_sel.4

MOV
Move
Source 5
Dest cmd 0

5

This rung will trigger the Abort Command.

cmd_sel.5

MOV
Move
Source 6
Dest cmd 0

This routine will allow the user to trigger the commands to be sent to the module. Each of the first rungs, 0 – 12, will trigge

This rung will trigger the Reload Non-Volatile Command.

6 cmd_sel.6 — MOV: Move Source 16, Dest cmd 0

This rung will trigger the Rate Cal 5 Pt Command.

7 cmd_sel.7 — MOV: Move Source 96, Dest cmd 0

This rung will trigger the Rate Cal 2 Pt Command.

8 cmd_sel.8 — MOV: Move Source 97, Dest cmd 0

This rung will trigger the Cal Low Command.

9 cmd_sel.9 — MOV: Move Source 100, Dest cmd 0

This rung will trigger the Cal High Command.

10 cmd_sel.10 — MOV: Move Source 101, Dest cmd 0

This rung will trigger the C2 Cal Command.

11 cmd_sel.11 — MOV: Move Source 102, Dest cmd 0

This routine will allow the user to trigger the commands to be sent to the module. Each of the first rungs, 0 – 12, will trigge

This rung will trigger the Set Default Parameters Command.

12

cmd_sel.12]

MOV
Move
Source 148
Dest cmd
0

This rung will check to see if a command has been triggered. If so, it will call the subroutine to actually send the command to the module. The tag cmd will be sent to the subroutine as an input parameter indicating which command to run.

13

NEQ
Not Equal
Source A cmd
0
Source B 0

cmd_delay.DN]/[

TON
Timer On Delay
Timer cmd_delay
Preset 500
Accum 0
EN
DN

14

cmd_delay.DN] [— control_bits.8 (L)

15

control_bits.8] [

JSR
Jump To Subroutine
Routine Name run_cmd
Input Par cmd

Once the subroutine is done, it will set bit cmd_sel.15 to on indicating it is done. At this point, this routing will clear the cmd tag, and reset all control bits.

16

cmd_sel.15] [

MOV
Move
Source 0
Dest cmd
0

MOV
Move
Source 0
Dest control_bits
0

cmd_delay (RES)

MOV
Move
Source 0
Dest cmd_sel
0

(End)

This rung will send the output percentage value from the HI 1756-FC module to the Analog Output module to control the feeder.

0

MOV
Move
Source Local:5:I.Data[5]
0
Dest Local:1:O.Ch0Data
0.0

These following rungs will take an input from the Digital input module and trigger a command from that input. These commands are: Input 0 is start; Input 1 is stop; Input 2 is refill; Input 3 will go to manual mode.

1

Local:2:I.Data.0 — NEQ (Not Equal, Source A Local:5:I.Data[12] 0, Source B 4) — once_5 ONS — cmd_sel.0 L

2

Local:2:I.Data.1 — once_6 ONS — cmd_sel.1 L

3

Local:2:I.Data.2 — once_7 ONS — cmd_sel.2 L

4

Local:2:I.Data.3 — once_8 ONS — cmd_sel.4 L

(End)

Vendor/ Address/ Phone

Hardy Instruments
3860 Calle Fortunada, San Diego, CA , 92123-1825
(800) 821-5831 USA

I am using some ONS bits in subroutines. During start up, these bits are set to prevent outputs going on during startup phase of PLC. Because these are in subroutines, then do not get cleared, which causes problem when trying to run the sub routines. This rung will clear these bits the first time for proper operation of the subroutine commands.

0 first_scan / once_0 U once_1 U once_2 U once_3 U once_4 U once_5 U once_6 U once_7 U once_8 U

1 first_scan L

Sample is for demonstration purposes only, showing an example of reading and writing commands and values to and from the module.

Main routine with bits tro trigger the start of all other routines.

On power up, will copy current_setpt value to new_setpt tag. This sets up the compare for the times when the operator enters a new setpoint.

2 once_4 / COP Copy File Source current_setpt Dest new_setpt Length 1 once_4 L

Vendor/ Address/ Phone

Hardy Instruments
3860 Calle Fortunada, San Diego, CA , 92123-1825
(800) 821-5831 USA

The operator will enter a new setpoint into the new_setpt tag. If the new setpoint is not equal to the current setpoint, and the module is not in manual mode, then it will jump to the new_setpt subroutine. This subroutine will download the new setpoint value into the module

3

NEQ
Not Equal
Source A Local:5:I.Data[12]
0
Source B 2

NEQ
Not Equal
Source A new_setpt
309
Source B current_setpt
309

JSR
Jump To Subroutine
Routine Name new_setpt

This rung will jump to the subroutine that checks for a command being triggered.

This routine will allow the user to trigger the commands to be sent to the module. Each of the first rungs, 0 – 12, will trigge

4

JSR
Jump To Subroutine
Routine Name cmd_select

The Local:x:C table is in SINT registers, so different values are difficult to read. This rung will copy the data located in the Local:x:C table into a tag the user can read.

5

COP
Copy File
Source Local:5:C.Data[0]
Dest readconfig
Length 1

This sample code does not use the AOP for the module, so writing the data values into the Local:x:C table can be a little difficult. This rung will copy the data from the tag a parameters read command fills and places it into the Local:x:C table.

6

copy_to_config_table

COP
Copy File
Source param_read.Ch_Active
Dest Local:5:C.Data[4]
Length 144

This rung will trigger the read command to read the parameters from the module. This will also be triggered if the user wishes to read the totals.

7

DO_READ_PARAM_0

do_read_totals

JSR
Jump To Subroutine
Routine Name read_param_msg

Vendor/ Address/ Phone

Hardy Instruments
3860 Calle Fortunada, San Diego, CA , 92123-1825
(800) 821-5831 USA

This rung will read the calibration values from the module.

8 DO_read_cal_0 — JSR: Jump To Subroutine, Routine Name Read_Cal_Values

This will start the write of the parameter values.

9 DO_write_PARAM_0 — JSR: Jump To Subroutine, Routine Name Write_param_MSG

This will write the calibration parameters. There are two different portions of the modules calibration, the weight calibration and the rate calibration. This rung allows you to select which portion or both portions to write.

10 only_rate_values OR only_weight_values OR both_wt_n_rate_values — JSR: Jump To Subroutine, Routine Name Write_Cal_Values

This will start the operation to set the manual operating percentage when in manual mode.

11 EQU: Equal, Source A Local:5:I.Data[12] 0, Source B 2 — set_man_op — JSR: Jump To Subroutine, Routine Name Set_man_op

This will clear the total from the memory.

12 clear_total — JSR: Jump To Subroutine, Routine Name Clear_Total

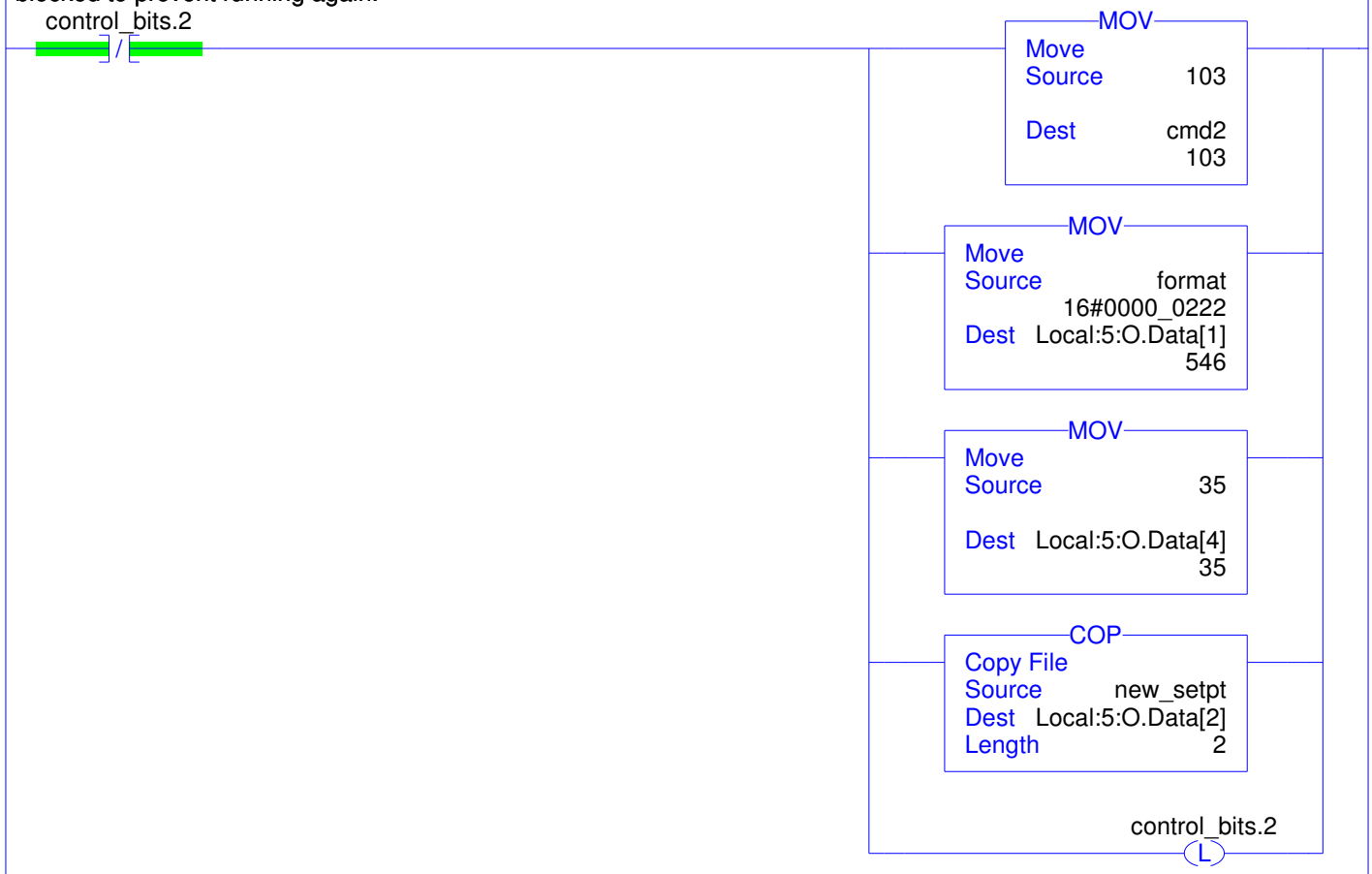
This jumps to the subroutine that controls the inputs and outputs from the module.

13 JSR: Jump To Subroutine, Routine Name control_IO

(End)

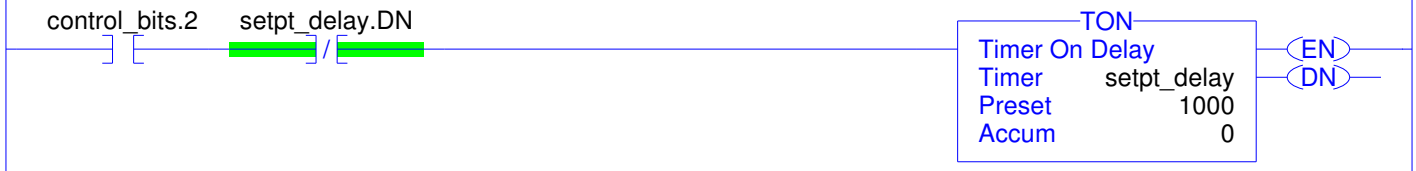
When the subroutine is triggered, this rung will move the write command, 103, to a com2 tag. It also moves the format value, xyz hex, setpoint parameter ID, 35, and the new setpoint value into the Local:x:O table. Once triggered, the rung is blocked to prevent running again.

0



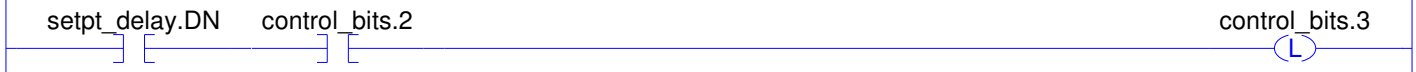
The ladder logic seems to run very fast, so once the routine is triggered and we start moving data around, I placed a delay here to prevent commands being sent before all the data movement is done. I set for 1 second, but do not think it needs to be this long a delay. Just needs to be long enough for all data to be moved/copied before proceding.

1



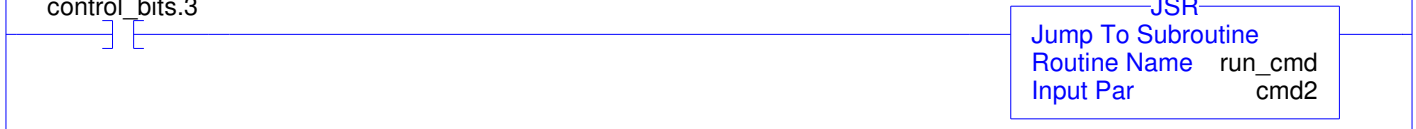
Once the delay time has elapsed, will latch a bit on to start next stage of command.

2



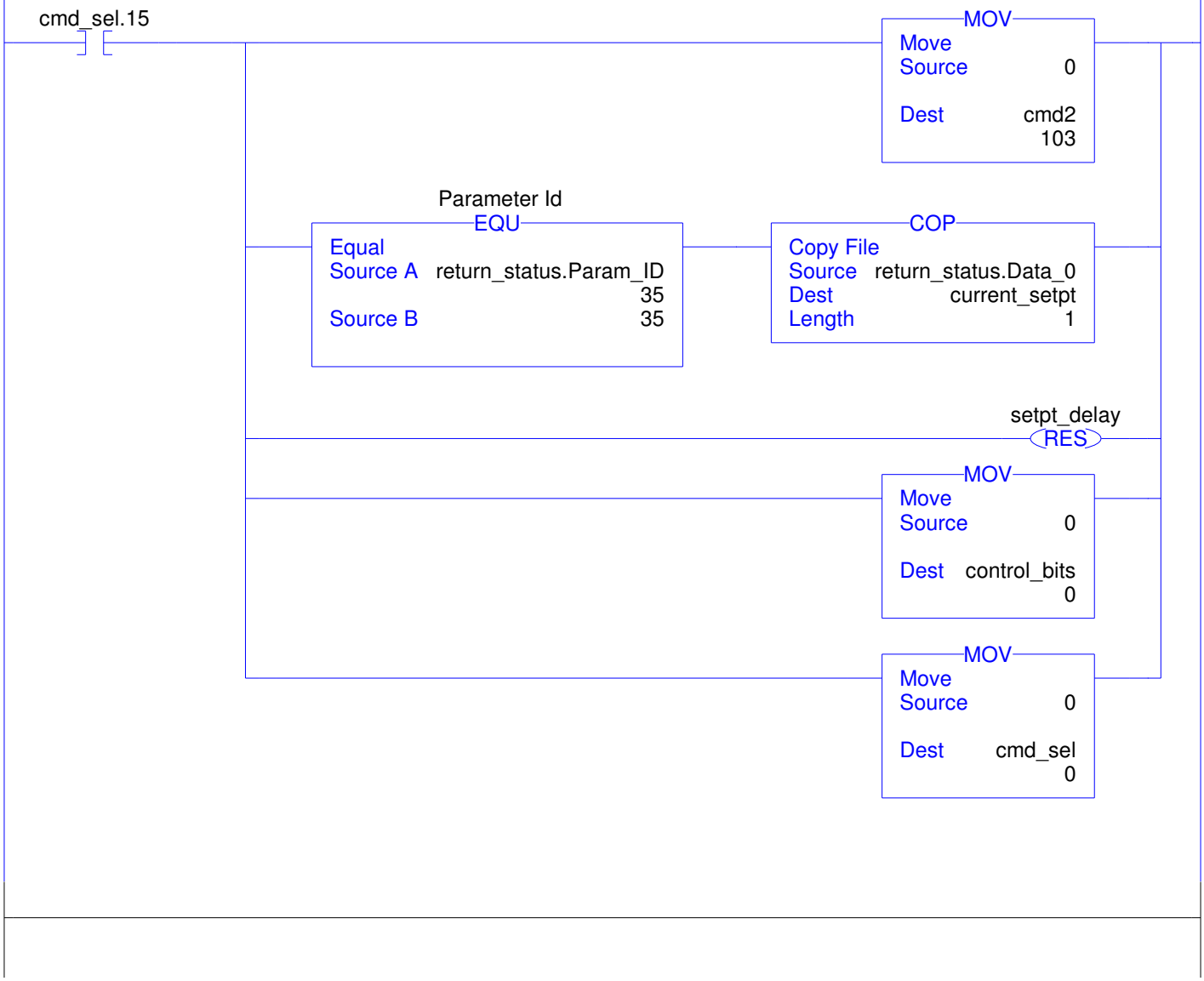
This rung will trigger the subroutine that actually runs the command and sends it to the module. This subroutine needs a command number as a input parameter. The command number in cmd2 tag will be given as the input parameter.

3



The subroutine will turn on a bit indicating when it is done. Once that bit, cmd_sel.15, is set, this rung will then clear tags, reset delay timers, copy status and clear bits that triggered the routine.

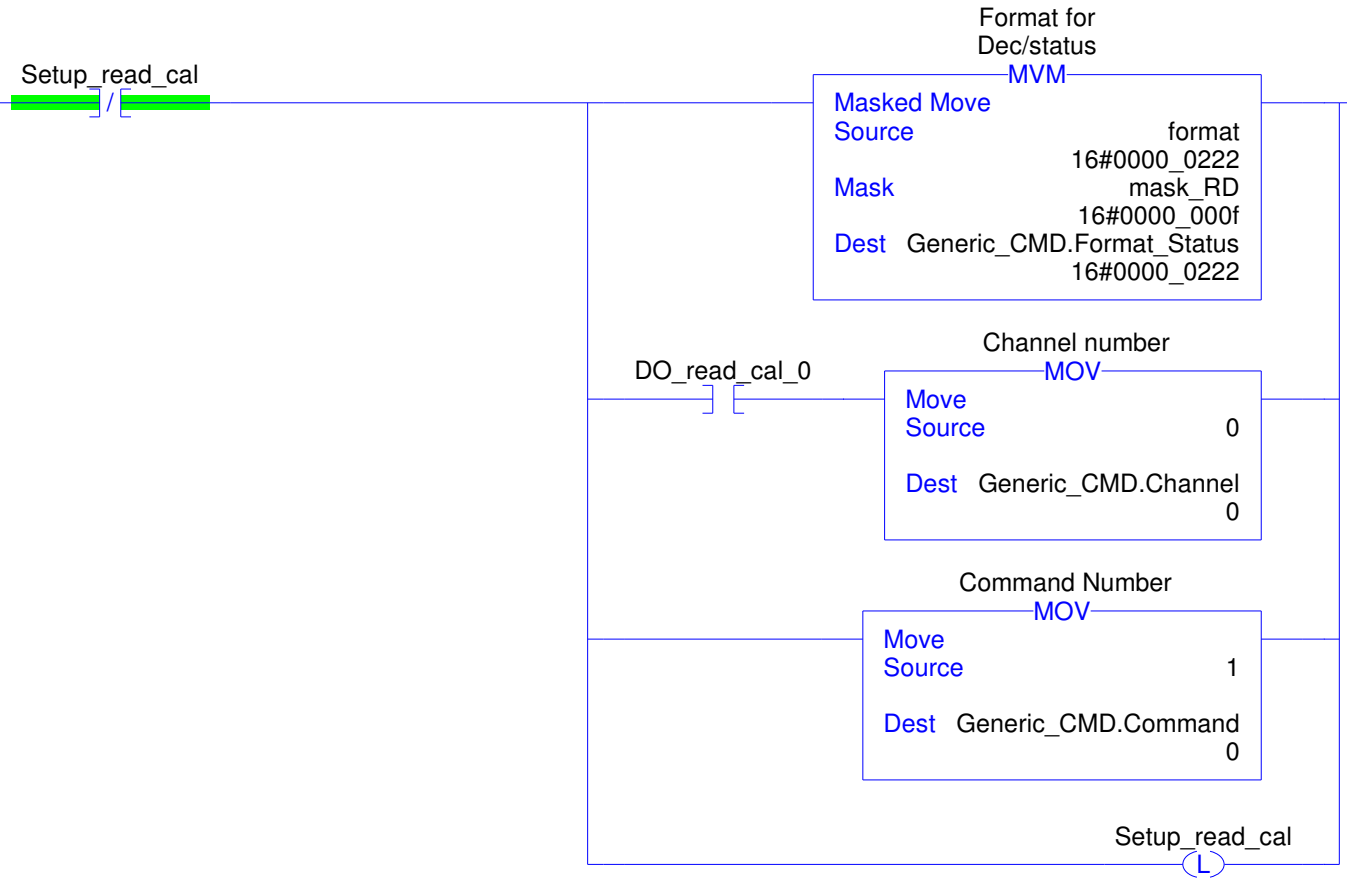
4



(End)

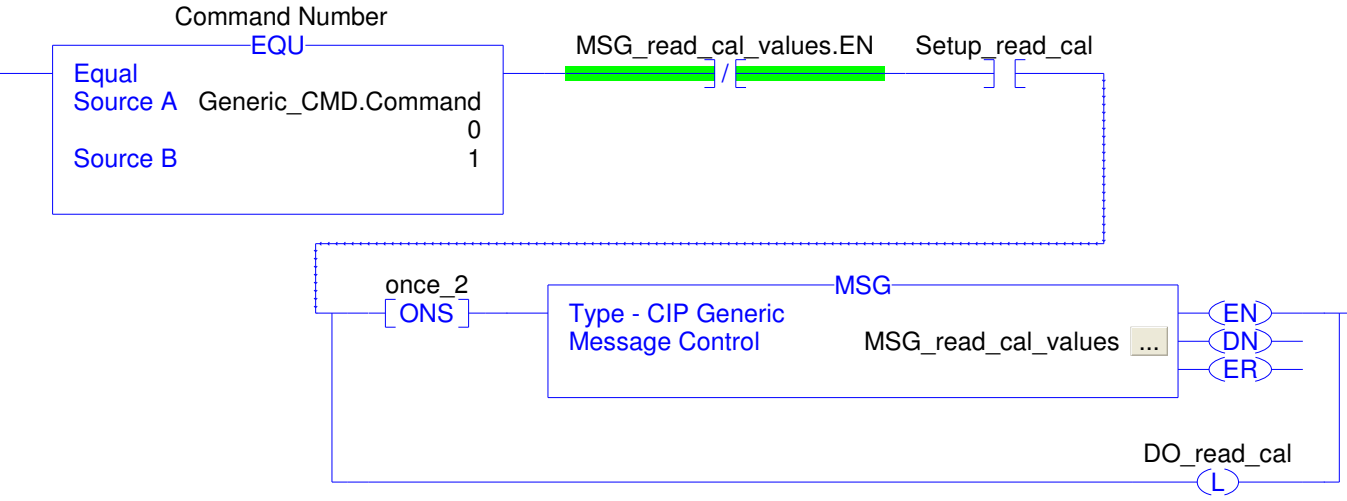
When routine started, this rung will, using the mask move, move the number of decimal places applicable for the rate cal numbers, into the Generic_CMD tag. Also moved into this tag is the channel number and the command number. This tag will be used for several commands, the command number indicating the actual command that is being run.

0



Once the command has been entered into the Generic_CMD tag, the MSG instruction will be started. It will use the Generic_CMD tag as the source tag for this instruction. The destination tag for this instruction will be a user defined tag holding the calibration values named Cal_values_read.

1



When the MSG instruction is complete, the setup bits and tags are cleared.

2

DO_read_cal MSG_read_cal_values.DN Setup_read_cal (U)

Command Number
MOV
Move
Source 0
Dest Generic_CMD.Command
0

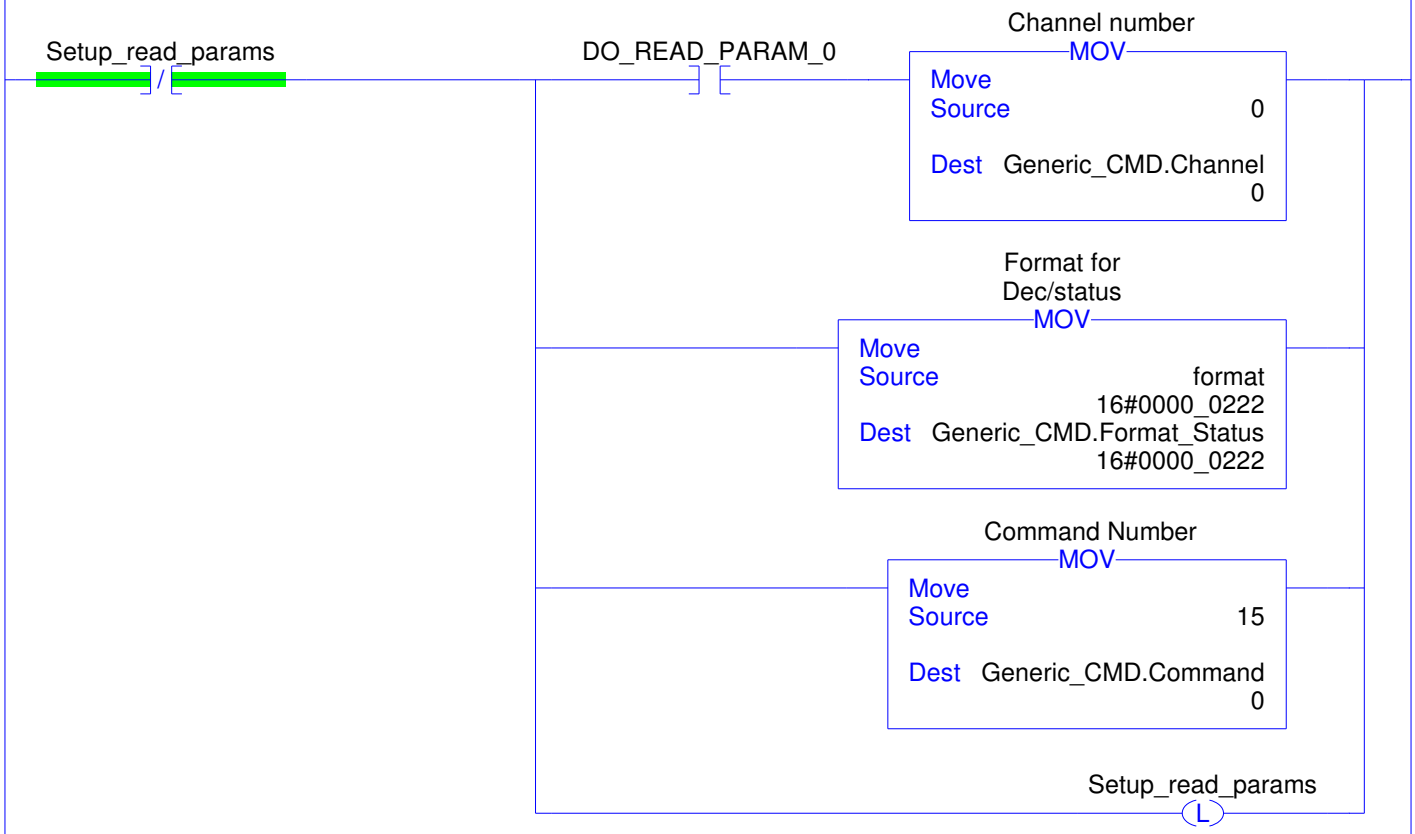
DO_read_cal (U)

DO_read_cal_0 (U)

(End)

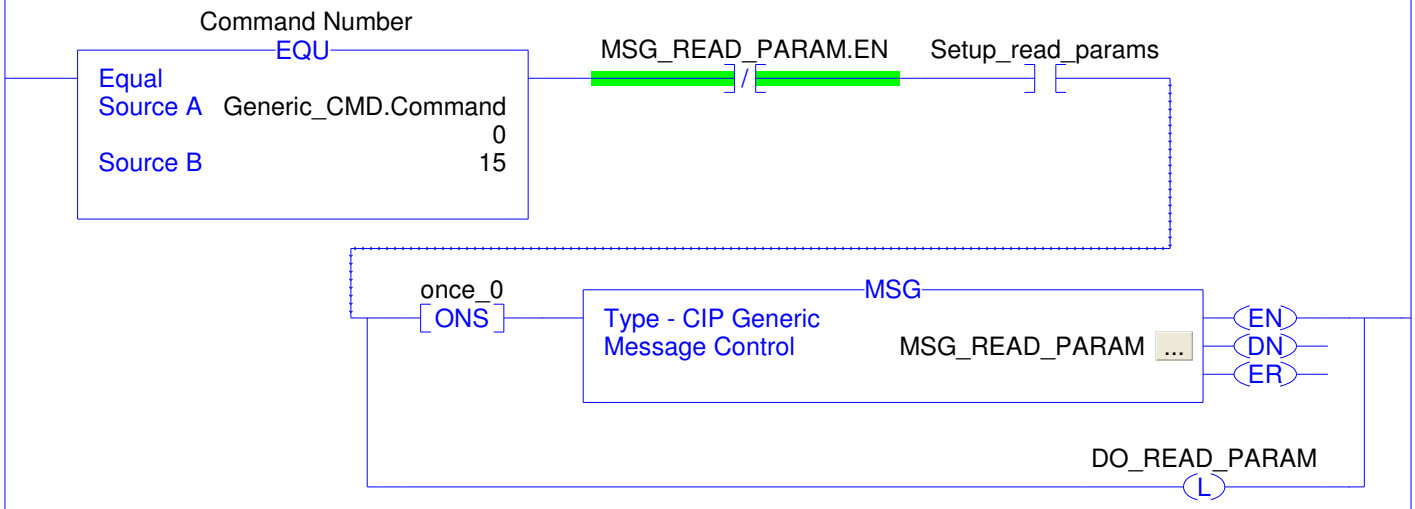
When routine started, this rung will move the format value, which holds the decimal place data applicable for the Rate, Weight and Total type data into the Generic_CMD tag. Also moved into this tag is the channel number and the command number.
This tag will be used for several commands, the command number indicating the actual command that is being run.

0



Once the command has been entered into the Generic_CMD tag, the MSG instruction will be started. It will use the Generic_CMD tag as the source tag for this instruction. The destination tag for this instruction will be a user defined tag holding the calibration values named param_read.

1



2

When the MSG instruction is complete, the setup bits and tags are cleared.

DO_READ_PARAM MSG_READ_PARAM.DN

Setup_read_params (U)

COP
Copy File
Source param_read.Total
Dest total
Length 1

COP
Copy File
Source param_read.BatchTotal
Dest batch_total
Length 1

do_read_totals (U)

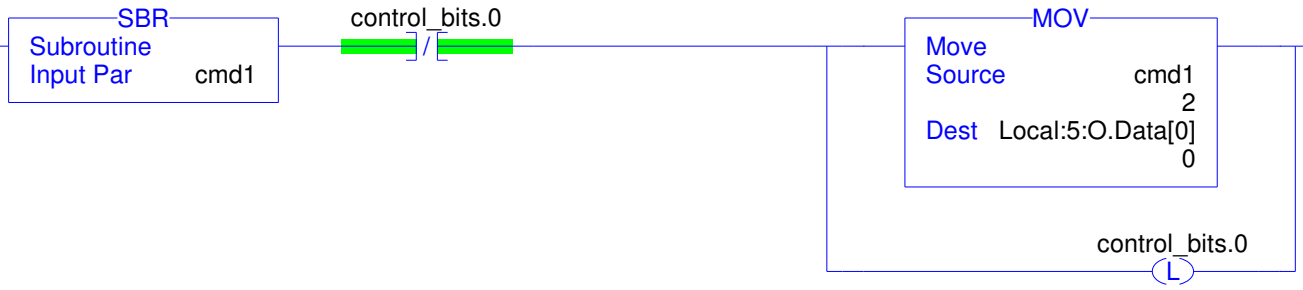
DO_READ_PARAM (U)

DO_READ_PARAM_0 (U)

(End)

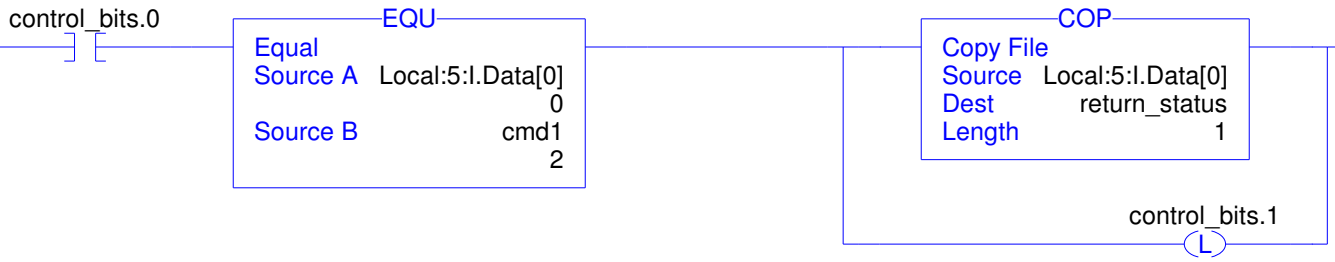
This routine will get command numbers from other routines and actually send it to the module, gather return status, and let calling routines know when it is done.
This rung will move the input command number to the Local:x:O data table to trigger the command.

0



Once it has written the command number out to the module in the output table, it will monitor the input table Local:x:I for a reflection of the command number sent. This would indicate the command is completed. Once completed, the rung will copy the command status information into the “return_status” tag. This is a location where the user can refer to for status and/or error codes.

1



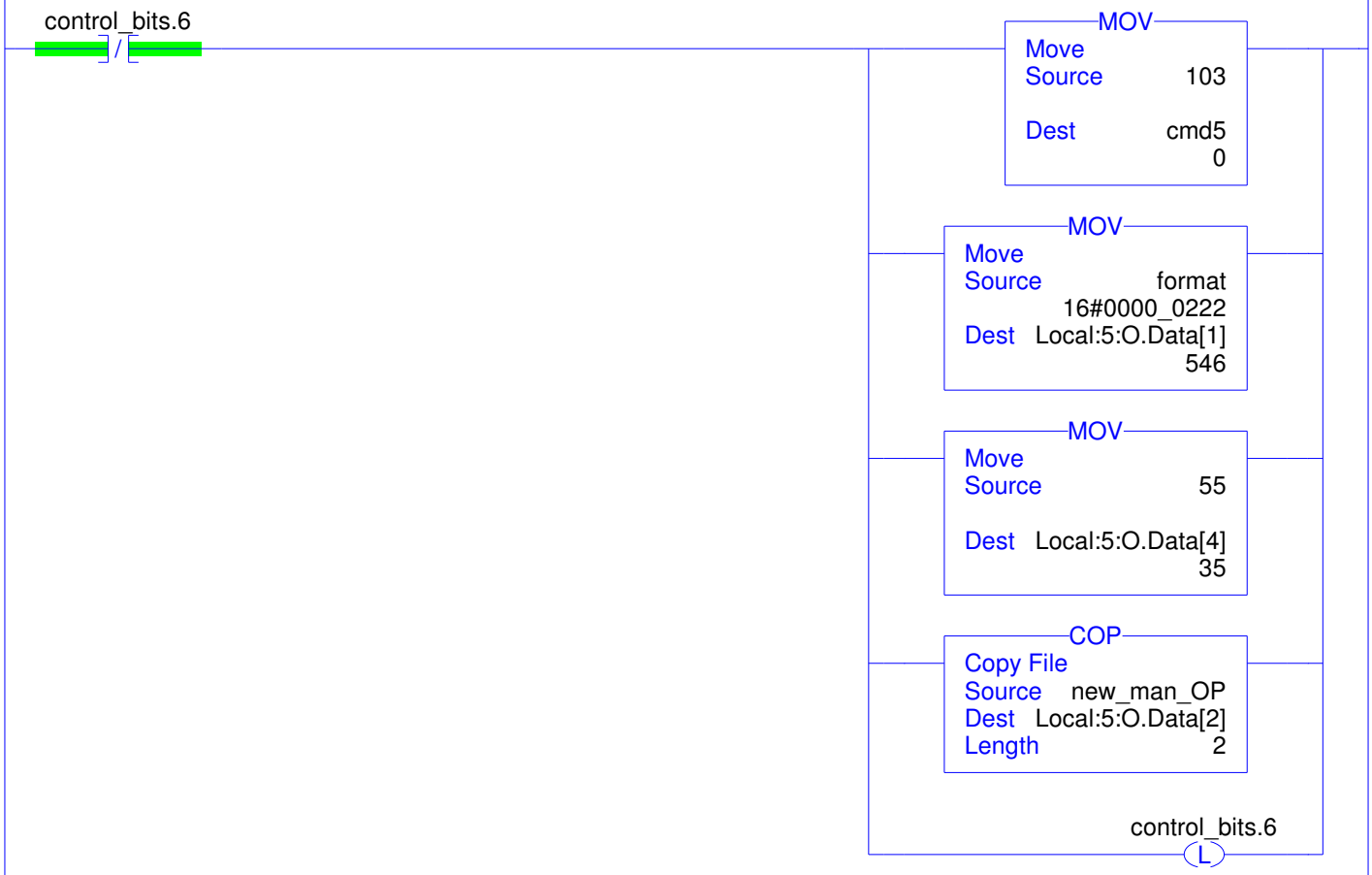
Once the command is complete, this rung will clear the output data location and set the bit indicating to the calling routine that it is done.

2

(End)

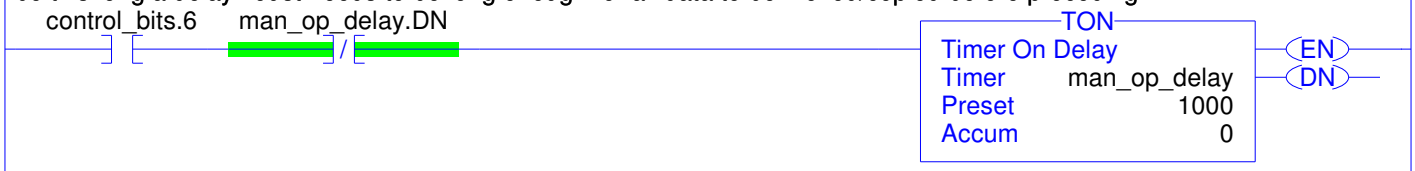
When the subroutine is triggered, this rung will move the write command, 103, to a com5 tag. It also moves the format value, xyz hex, manual OP parameter ID, 55, and the new OP value into the Local:x:O table. Once triggered, the rung is blocked to prevent running again.

0



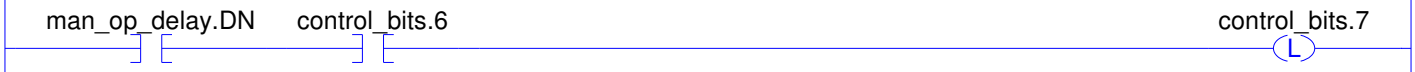
The ladder logic seems to run very fast, so once the routine is triggered and we start moving data around, I placed a delay here to prevent commands being sent before all the data movement is done. I set for 1 second, but do not think it needs to be this long a delay. Just needs to be long enough for all data to be moved/copied before proceding.

1



Once the delay time has elapsed, will latch a bit on to start next stage of command.

2

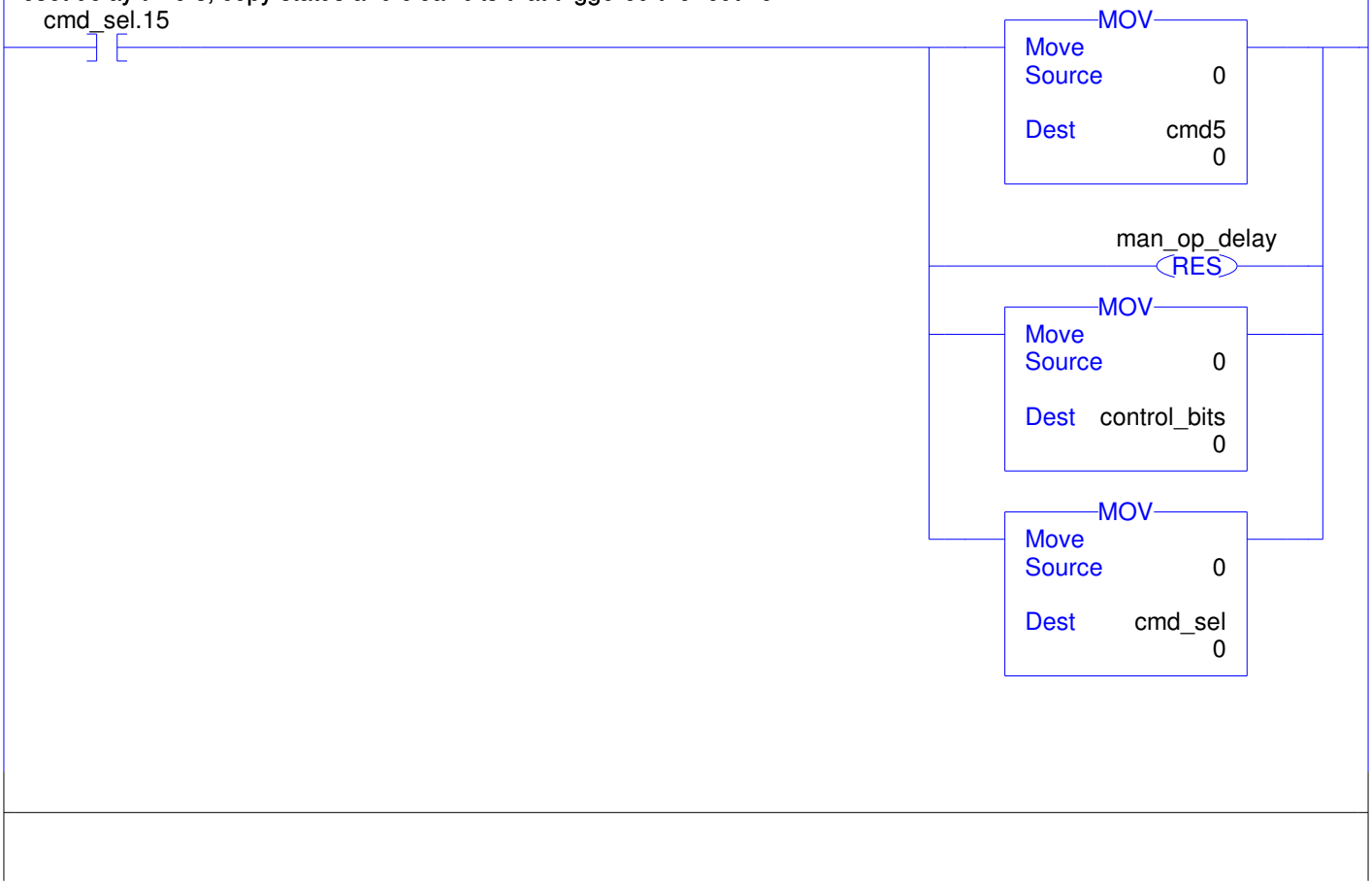


This rung will trigger the subroutine that actually runs the command and sends it to the module. This subroutine needs a command number as a input parameter. The command number in cmd5 tag will be given as the input parameter.

3

The subroutine will turn on a bit indicating when it is done. Once that bit, cmd_sel.15, is set, this rung will then clear tags, reset delay timers, copy status and clear bits that triggered the routine.

4



(End)

When routine started, this rung will, using the mask move, move the number of decimal places applicable for the rate cal numbers, into the Cal_values_write tag. Also moved into this tag are the channel number and the command number. The command number being written will depend on the selection of what portion of the calibration data the user wishes to write.

0

Setup_write_cal_values

Channel Number
MOV
Move
Source 0
Dest Cal_values_write.Channel
0

Rate Decimal Places
MVM
Masked Move
Source format
16#0000_0222
Mask mask_RD
16#0000_000f
Dest Cal_values_write.Rate_Dec_Pt
2

only_rate_values

Command Number
MOV
Move
Source 129
Dest Cal_values_write.Command
129

only_weight_values

Command Number
MOV
Move
Source 130
Dest Cal_values_write.Command
129

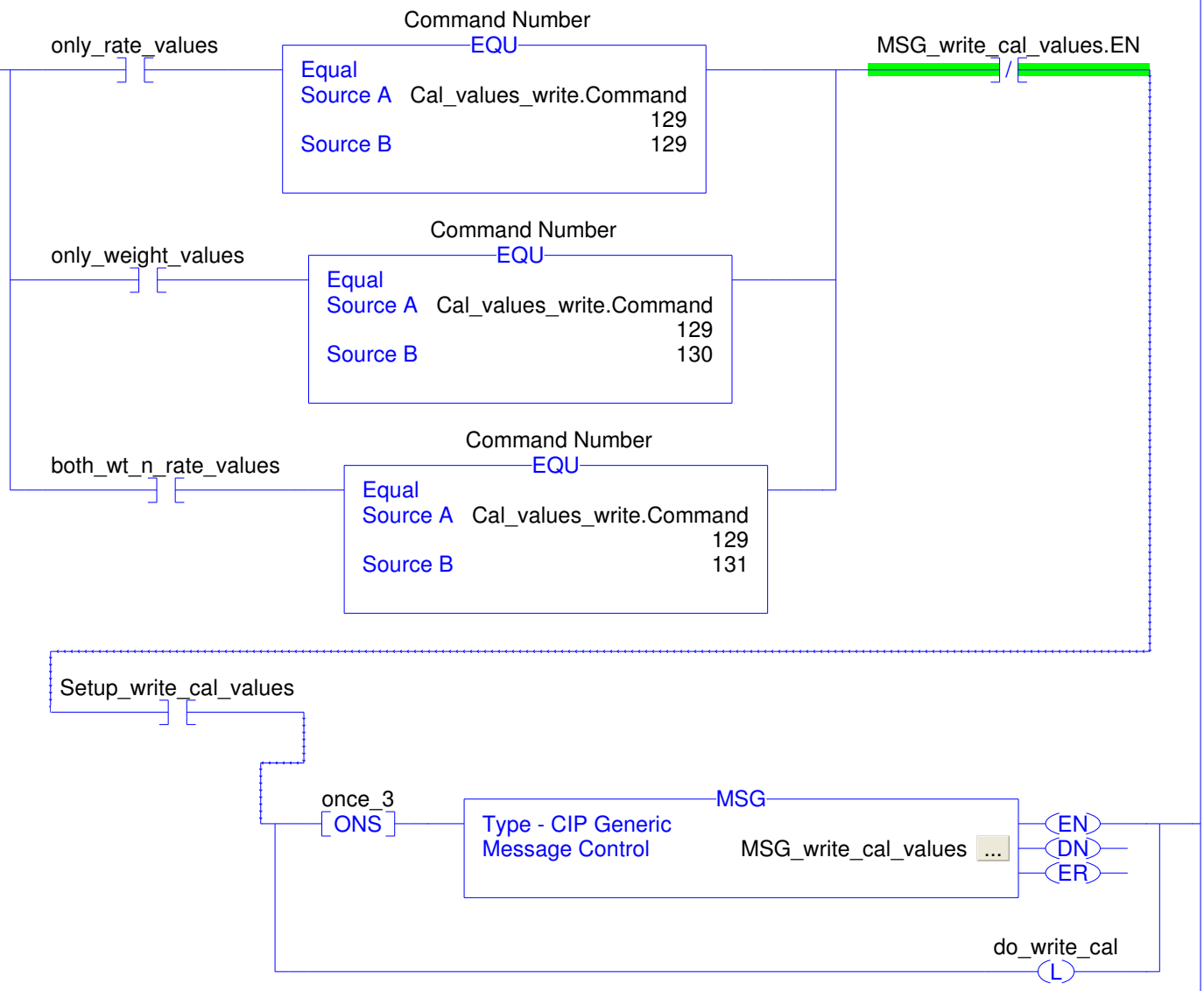
both_wt_n_rate_values

Command Number
MOV
Move
Source 131
Dest Cal_values_write.Command
129

Setup_write_cal_values
L

Once the command has been entered into the Cal_values_write tag, the MSG instruction will be started. It will use the Cal_values_write tag as the source tag for this instruction. The destination tag for this instruction will be a user defined tag holding the calibration values being read back, named Cal_values_read.

1



When the MSG instruction is complete, the setup bits and tags are cleared.

2

do_write_cal MSG_write_cal_values.DN Setup_write_cal_values (U)

Command Number

MOV
Move
Source 0
Dest Cal_values_write.Command
129

do_write_cal_0 (U)

do_write_cal (U)

only_rate_values (U)

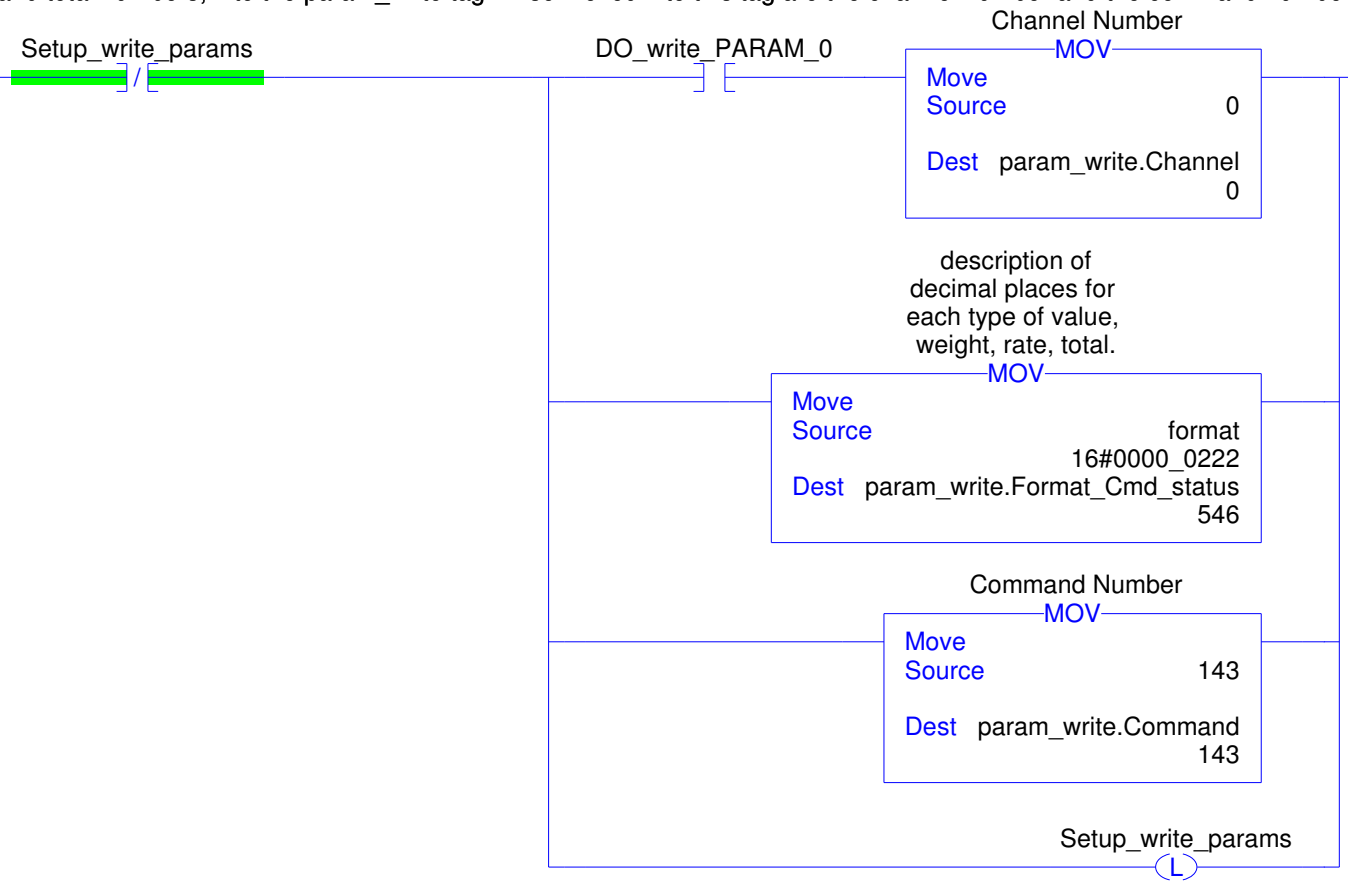
only_weight_values (U)

both_wt_n_rate_values (U)

(End)

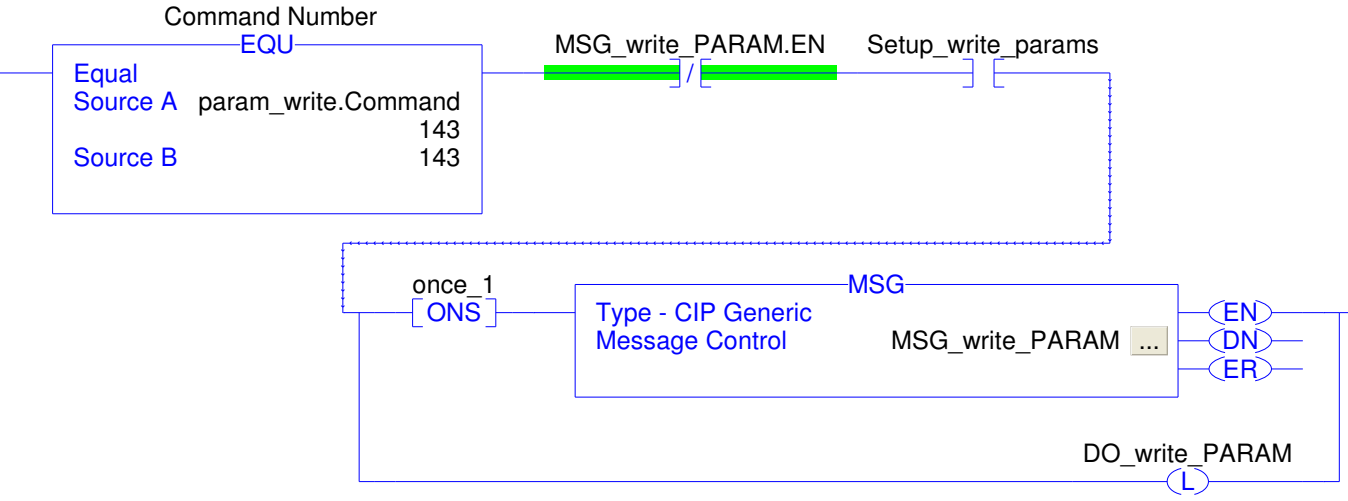
0

When routine started, this rung will move the format value with the number of decimal places applicable for the rate, weight, and total numbers, into the param_write tag. Also moved into this tag are the channel number and the command number.



1

Once the command has been entered into the param_write tag, the MSG instruction will be started. It will use the param_write tag as the source tag for this instruction. The destination tag for this instruction will be a user defined tag holding the parameter values being read back, named param_read.



2

When the MSG instruction is complete, the setup bits and tags are cleared.

DO_write_PARAM MSG_write_PARAM.DN Setup_write_params (U)

Command Number

MOV
Move Source 0
Dest param_write.Command 143

DO_write_PARAM (U)

DO_write_PARAM_0 (U)

(End)