

Hyperion Code Guide

Table of Contents

FOC Mask Setting 3

Calibration 9

Multiple Connections 10

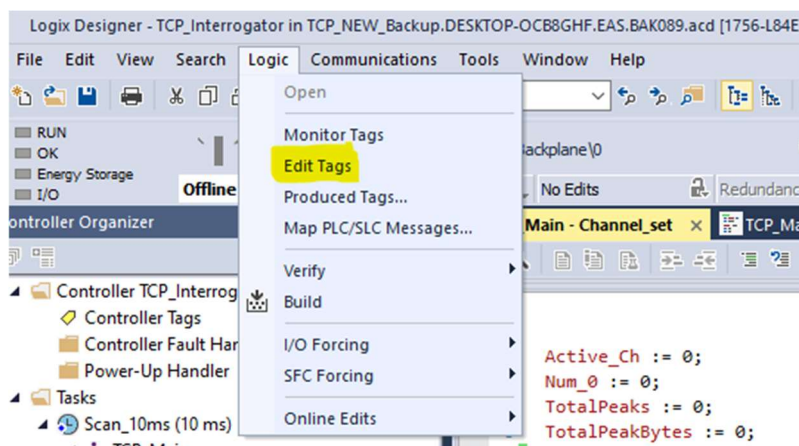
Updating Temp Rate 14

Troubleshooting/ Server Connections..... 15

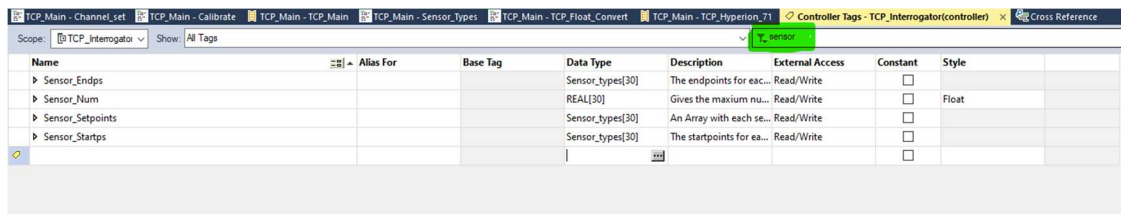
Space Intentionally Left Blank

FOC mask setting

1. Navigate to logic then to edit tags.

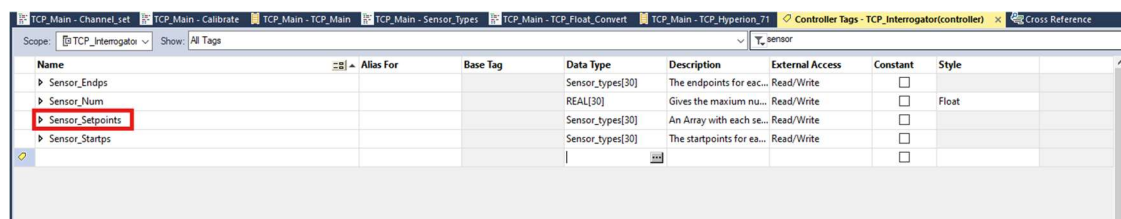


2. After clicking edit tags, search the word 'sensor.'



- 3.

4. Click on the 'Sensor Setpoints' tag. This will allow you to input the spacing of the cable's specific sensors. Be sure to not input more than 30 of the setpoints even if there are more.



- | Scope: [B]TCP_Integrator | | Show: All Tags | | T_Sensor | | | |
|--------------------------|-------|----------------|-------------|------------------|-------------|-------------------|--|
| Name | Value | Force Mask | Style | Data Type | Description | Properties | |
| ▶ Sensor_Endps | | (...) | (...) | The endpoi | | Extended Properti | |
| ▶ Sensor_Num | | (...) | (...) Float | REAL[30] | Gives the n | | |
| ▶ Sensor_Setpoint | | (...) | (...) | Sensor_types[30] | An Array w | | |
| ▶ Sensor_Setpoints[0] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[1] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[2] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[3] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[4] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[5] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[6] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[7] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[8] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[9] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[10] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[11] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[12] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[13] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[14] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Setpoints[15] | | (...) | (...) | Sensor_types | An Array w | | |
| ▶ Sensor_Geogrids[16] | | (...) | (...) | Sensor_types | An Array w | | |

TCP_Main - Channel_set TCP_Main - Calibrate TCP_Main - TCP_Nums TCP_Main - Sensor_Types TCP_Main - TCP_Float_Convert TCP_Main - TCP_Hyperion

Scope: Show:

Name	Value
▶ Sensor_Endps	{...}
▶ Sensor_Num	{...}
▲ Sensor_Setpoints	{...}
▲ Sensor_Setpoints[0]	{...}
▶ Sensor_Setpoints[0].Array	{...}
▲ Sensor_Setpoints[1]	{...}
▶ Sensor_Setpoints[1].Array	{...}
▲ Sensor_Setpoints[2]	{...}
▶ Sensor_Setpoints[2].Array	{...}
▲ Sensor_Setpoints[3]	{...}
▶ Sensor_Setpoints[3].Array	{...}
▲ Sensor_Setpoints[4]	{...}
▶ Sensor_Setpoints[4].Array	{...}
▲ Sensor_Setpoints[5]	{...}
▶ Sensor_Setpoints[5].Array	{...}
▶ Sensor_Setpoints[6]	{...}
▶ Sensor_Setpoints[7]	{...}
▶ Sensor_Setpoints[8]	{...}
▶ Sensor_Setpoints[9]	{...}
▶ Sensor_Setpoints[10]	{...}

‘Setpoints’ are the Wavelengths being measured by the fiber optic and there may be more than 30 different wavelengths that a fiber optic cable has sensors for. If there are more than 30,

ONLY input the first 30 wavelengths.

Do not change the size of Setpoint Array.

5.) Check each Index of Sensor_Setpoints until you find the first empty array.

In red is an example of a full array, there may be zeros at the end of the array, it is still being used. In green is an example of what to look for when you are about to input the wavelengths.

Make sure to input the wavelengths in ascending order (lowest at the 0 index and largest at the last inputted index)

Name	Value
Sensor_Setpoints[0].Array	{...}
Sensor_Setpoints[1]	{...}
Sensor_Setpoints[1].Array	{...}
Sensor_Setpoints[1].Array[0]	1511.0
Sensor_Setpoints[1].Array[1]	1516.5
Sensor_Setpoints[1].Array[2]	1522.0
Sensor_Setpoints[1].Array[3]	1527.5
Sensor_Setpoints[1].Array[4]	1533.0
Sensor_Setpoints[1].Array[5]	1538.5
Sensor_Setpoints[1].Array[6]	1544.0
Sensor_Setpoints[1].Array[7]	1549.5
Sensor_Setpoints[1].Array[8]	1555.0
Sensor_Setpoints[1].Array[9]	1560.5
Sensor_Setpoints[1].Array[10]	1566.0
Sensor_Setpoints[1].Array[11]	1571.5
Sensor_Setpoints[1].Array[12]	1577.0
Sensor_Setpoints[1].Array[13]	1582.5
Sensor_Setpoints[1].Array[14]	0.0
Sensor_Setpoints[1].Array[15]	0.0
Sensor_Setpoints[1].Array[16]	0.0

Name	Value
Sensor_Setpoints[2]	{...}
Sensor_Setpoints[2].Array	{...}
Sensor_Setpoints[3]	{...}
Sensor_Setpoints[3].Array	{...}
Sensor_Setpoints[3].Array[0]	0.0
Sensor_Setpoints[3].Array[1]	0.0
Sensor_Setpoints[3].Array[2]	0.0
Sensor_Setpoints[3].Array[3]	0.0
Sensor_Setpoints[3].Array[4]	0.0
Sensor_Setpoints[3].Array[5]	0.0
Sensor_Setpoints[3].Array[6]	0.0
Sensor_Setpoints[3].Array[7]	0.0
Sensor_Setpoints[3].Array[8]	0.0
Sensor_Setpoints[3].Array[9]	0.0
Sensor_Setpoints[3].Array[10]	0.0
Sensor_Setpoints[3].Array[11]	0.0
Sensor_Setpoints[3].Array[12]	0.0
Sensor_Setpoints[3].Array[13]	0.0
Sensor_Setpoints[3].Array[14]	0.0
Sensor_Setpoints[3].Array[15]	0.0

6.) After the wavelength setpoints have been inputted into the Sensor_Setpoints tag keep track of the Index that you inputted into. This will be important later.

When setting the type of wire that you are putting into which channel you need to go to a different routine, **TCP Float convert.**

It is required to know which channel you will be putting in and what type of Fiber optic cable.

Inside of the Routine go to line 70, this is where the first channel is. There will be a line of code underneath the given channel index. This is what you will change for the different sensor types at different channel indexes (Channel_I)

There will be letters at the end of the tags.

‘Peak_max ...’, ‘End_wL ...’, ‘Start_wL ...’

```

60 COP(CH_Peaks_DINT[0], ch_TEST[0],30);
61
62
63 (*
64  If else ladder has the purpose of ensuring that the peaks fall between each sensor and exclude incorrect peaks.
65  The purpose of repeated IF statment is to change specific channel specifications.
66  Change the The maximum Peak Index, and max/min wL depending on the Fiber optic cable specs
67 *)
68
69
70 If (Channel_I = 1) then;
71   If (Peak_Index < Peak_max_A) and (End_wL_A[Peak_Index] > ch_TEST[Peak_Index]) and (Start_wL_A[Peak_Index] < ch_TEST[Peak_Index]) then; // cha
72     COP(CH_Peaks_DINT[Peak_Index],CH1_Peaks[Peak_Index],1);
73     temp1[Peak_Index] := 9/5*((CH1_Peaks[Peak_Index] - CH1_Peaks_Cal[Peak_Index]) * wavelengthConversion + waterTemp)+32;
74   end_if;
75 End_if;
76
77
78 If (Channel_I = 2) then;
79   if (Peak_Index < Peak_max_B) and (End_wL_B[Peak_Index] > ch_TEST[Peak_Index]) and (Start_wL_B[Peak_Index] < ch_TEST[Peak_Index]) then;
80     COP(CH_Peaks_DINT[Peak_Index],CH2_Peaks[Peak_Index],1);
81     temp2[Peak_Index] := 9/5*((CH2_Peaks[Peak_Index] - CH2_Peaks_Cal[Peak_Index]) * wavelengthConversion + waterTemp)+32;
82   end_if;
83 End_if;
84
85
86 If (Channel_I = 3) then;
87   If (Peak_Index < Peak_max_A) and (End_wL_A[Peak_Index] > ch_TEST[Peak_Index]) and (Start_wL_A[Peak_Index] < ch_TEST[Peak_Index]) then;
88     COP(CH_Peaks_DINT[Peak_Index],CH3_Peaks[Peak_Index],1);
89     temp3[Peak_Index] := 9/5*((CH3_Peaks[Peak_Index] - CH3_Peaks_Cal[Peak_Index]) * wavelengthConversion + waterTemp)+32;
90   end_if;
91 End_if;
92

```

Depending on the index you input will determine the letter for the specific sensors.

It is in alphabetical order (A-J) and in numerical order (0-9).

INDEX	0	1	2	3	4	5	6	7	8	9
LETTER	A	B	C	D	E	F	G	H	I	J

The specific Indexes and which letter they correspond to can be seen in the
‘Sensor Types’ routine as well

```

21     end_for; // loop for each setpoint in a given wire type
22 end_for; // loop for each wire type
23
24 COP(Sensor_Setpoints[0], Setpoint_wL_A[0], 30);
25 COP(Sensor_Endpts[0].Array[0], End_wL_A[0], 30); // information stored at INDEX 0 of 'Sensor' tags, 32 sensors ( A )
26 COP(Sensor_Startpts[0].Array[0], Start_wL_A[0], 30);
27 COP(Sensor_Num[0], Peak_max_A, 1);
28
29 COP(Sensor_Setpoints[1], Setpoint_wL_B[0], 30);
30 COP(Sensor_Endpts[1].Array[0], End_wL_B[0], 30); // information stored at INDEX 1 of 'Sensor' tags, 14 sensors ( B )
31 COP(Sensor_Startpts[1].Array[0], Start_wL_B[0], 30);
32 COP(Sensor_Num[1], Peak_max_B, 1);
33
34 COP(Sensor_Setpoints[2], Setpoint_wL_C[0], 30);
35 COP(Sensor_Endpts[2].Array[0], End_wL_C[0], 30); // information stored at INDEX 2 of 'Sensor' tags, -- sensors ( C )
36 COP(Sensor_Startpts[2].Array[0], Start_wL_C[0], 30);
37 COP(Sensor_Num[2], Peak_max_C, 1);
38
39 COP(Sensor_Setpoints[3], Setpoint_wL_D[0], 30);
40 COP(Sensor_Endpts[3].Array[0], End_wL_D[0], 30); // information stored at INDEX 3 of 'Sensor' tags, -- sensors ( D )
41 COP(Sensor_Startpts[3].Array[0], Start_wL_D[0], 30);
42 COP(Sensor_Num[3], Peak_max_D, 1);
43
44 COP(Sensor_Setpoints[4], Setpoint_wL_E[0], 30);
45 COP(Sensor_Endpts[4].Array[0], End_wL_E[0], 30); // information stored at INDEX 4 of 'Sensor' tags, -- sensors ( E )
46 COP(Sensor_Startpts[4].Array[0], Start_wL_E[0], 30);
47 COP(Sensor_Num[4], Peak_max_E, 1);

```

For example, if there was a set of wavelength setpoints stored at Index 3 for a specific Fiber Optic and channel 5 is the active channel with this sensor type, you would scroll down to Channel 5 (in the TCP_Float_Convert routine) and change the letters for the specified tags ('Peak_max_...', 'End_wL_...', 'Start_wL_...') to D.

After fully adding a Sensor, add any desired specification to the comment in the Sensor_Types routine.

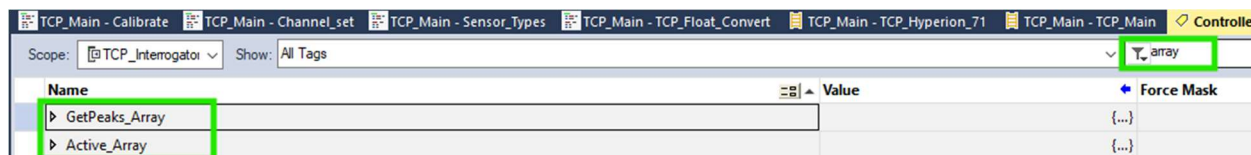
Calibration

The point of Calibration is to copy the currently measured wavelength into parallel arrays. This allows for the calculation of the difference of wavelength that occurs, which leads to being able to calculate a temperature. When Calibration occurs, the temperatures will read 70°F (or the water molds temperature) since the difference would calculate to 0 and not contribute to the temperature.

If a channel is not active, the respective temperature array will read 0. If a wavelength index is not being measured, then the temperature will read 0 for that corresponding index in the temperature array.

To see which Channels are active go to monitor Tags and search ‘array’

There you will find the tags Active Arrays and or GetPeaks Arrays.



Name	Value	Force Mask
GetPeaks_Array		{...}
Active_Array		{...}



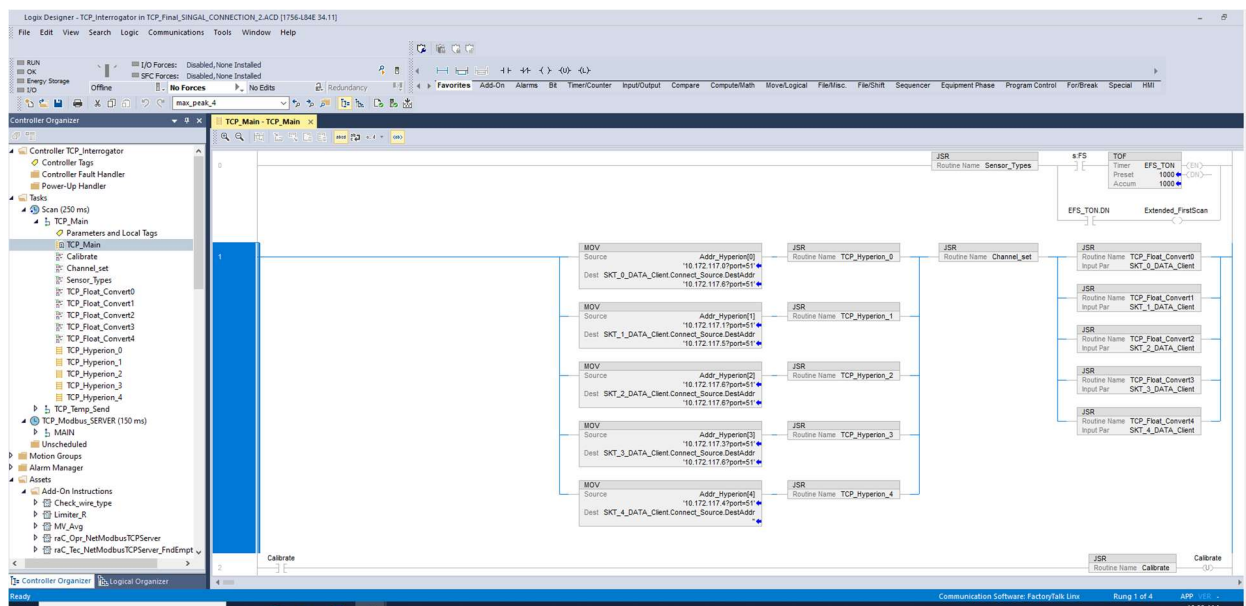
Active_Array	{...}
Active_Array[0]	3.0
Active_Array[1]	11.0
Active_Array[2]	14.0
Active_Array[3]	0.0
Active_Array[4]	0.0
Active_Array[5]	0.0
Active_Array[6]	0.0
Active_Array[7]	0.0
Active_Array[8]	0.0
Active_Array[9]	0.0
Active_Array[10]	0.0
Active_Array[11]	0.0
Active_Array[12]	0.0
Active_Array[13]	0.0
Active_Array[14]	0.0
Active_Array[15]	0.0



GetPeaks_Array	{...}
GetPeaks_Array[0]	0
GetPeaks_Array[1]	0
GetPeaks_Array[2]	32
GetPeaks_Array[3]	0
GetPeaks_Array[4]	0
GetPeaks_Array[5]	0
GetPeaks_Array[6]	0
GetPeaks_Array[7]	0
GetPeaks_Array[8]	0
GetPeaks_Array[9]	0
GetPeaks_Array[10]	18
GetPeaks_Array[11]	0
GetPeaks_Array[12]	0
GetPeaks_Array[13]	14
GetPeaks_Array[14]	0
GetPeaks_Array[15]	0

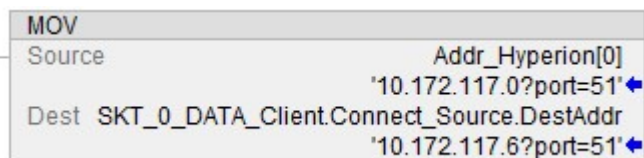
Setting Multiple Connections

In the Main routine, you will find 5 move blocks, moving information from the tag **'Addr_Hyperion'**, type: STRING [10]



Each block index from 0 through and including 4 are used to set the address of the five Hyperions.

Here is what one of the blocks looks like and the editing interface.



Double click on the source value to change the IP address for a given Hyperion device. The Port value will also be located here, and it does not change between devices so leave it constant between all move blocks. **In the above example only '10.172.117.0' would be modified in the string '10.172.117.0?port=51971'. Only ever change the IP. Leave the ' ? ' and ' = '.**

NEVER HAVE MORE THAN ONE CLIENT
SET TO THE SAME IP WHILE THE PLC IS LIVE

To access the temperature values, go to monitor tags and look for which channel number you want to observe. Within this array there will be 10 arrays, 0 through and including 9. Select the Array that correlates with the SKT connection number.

temp15[3]	{...}
temp15[3].Array	{...}
temp15[3].Array[0]	70.24654
temp15[3].Array[1]	70.232025
temp15[3].Array[2]	70.203026
temp15[3].Array[3]	70.26103
temp15[3].Array[4]	70.232025
temp15[3].Array[5]	70.203026
temp15[3].Array[6]	70.24654
temp15[3].Array[7]	70.18852
temp15[3].Array[8]	70.21753
temp15[3].Array[9]	70.26103
temp15[3].Array[10]	70.17403
temp15[3].Array[11]	70.130516
temp15[3].Array[12]	70.18852
temp15[3].Array[13]	70.24654
temp15[3].Array[14]	70.24654
temp15[3].Array[15]	70.18852
temp15[3].Array[16]	70.203026
temp15[3].Array[17]	70.203026
temp15[3].Array[18]	70.18852

The above temperature example shows temperatures stored at Channel 15, hence temp15, at index 3 meaning this information stores SKT_3's info.

Below is another example of where to find temperature information.

temp10[4].Array	{...}
temp10[4].Array[0]	69.89848
temp10[4].Array[1]	69.91299
temp10[4].Array[2]	70.0435
temp10[4].Array[3]	69.9855
temp10[4].Array[4]	69.76797
temp10[4].Array[5]	69.81148
temp10[4].Array[6]	69.88399
temp10[4].Array[7]	69.9565
temp10[4].Array[8]	70.0435
temp10[4].Array[9]	69.73897
temp10[4].Array[10]	69.86948
temp10[4].Array[11]	70.11601
temp10[4].Array[12]	69.88399
temp10[4].Array[13]	70.11601
temp10[4].Array[14]	0.0
temp10[4].Array[15]	0.0
temp10[4].Array[16]	0.0
temp10[4].Array[17]	0.0
temp10[4].Array[18]	0.0
temp10[4].Array[19]	0.0

The above temperature array is correlated with the 10th channel of the fourth socket connection
(The IP address of the machine correlated to SKT_4 variables)

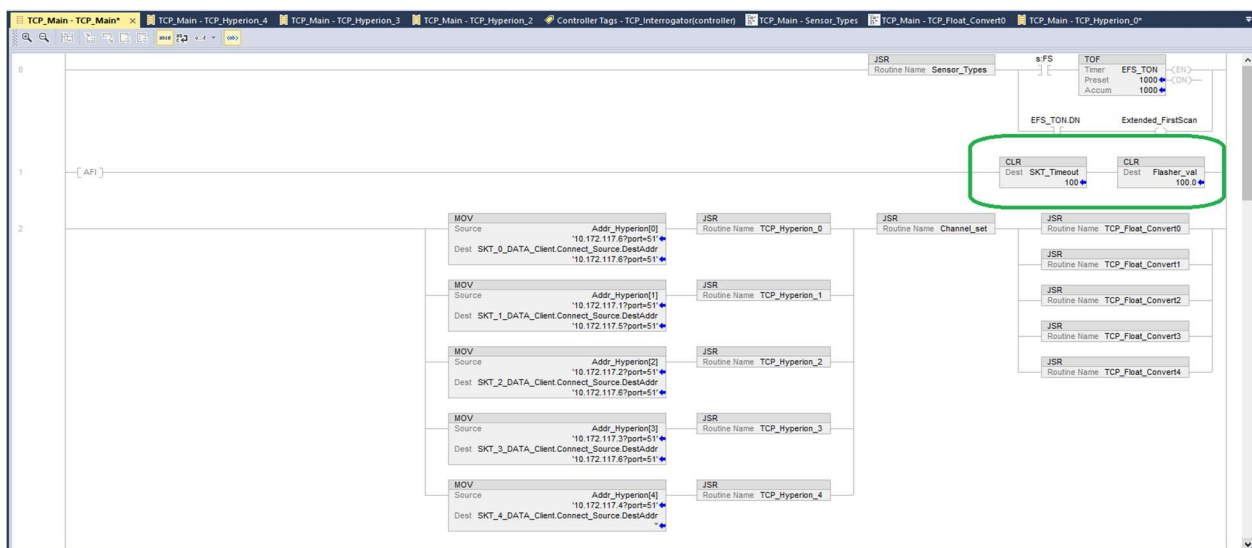
If a SKT connection is not reading or writing, try unplugging the ethernet cable on the correlated Hyperion device and plugging it back in **after** the device reads ‘ ** OBTAINING IP** ’. Other potential issues will be the timeout amount for a given Client and the timer values are not set higher than the routine scan rate.

Updating Temp Rate

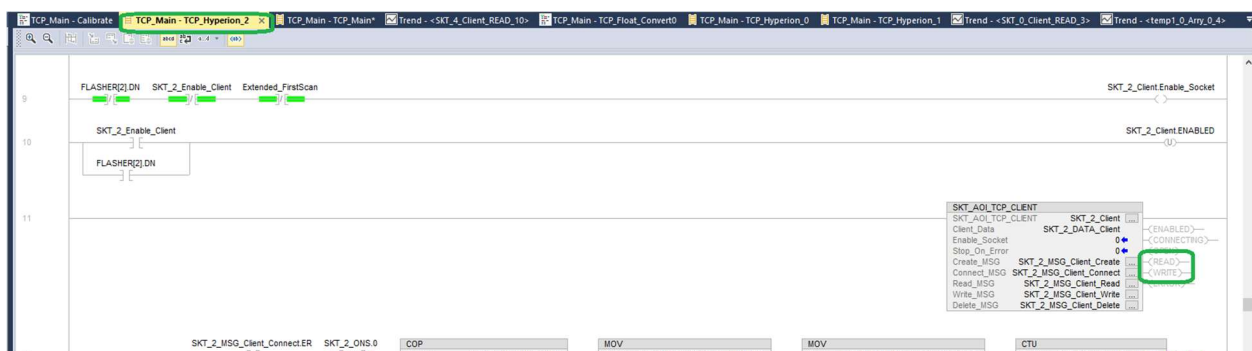
To update how fast the PLC tags are being updated with the temperature values.

Set the scan rate to a number less than 50ms, ideally about 10ms.

(increasing or decreasing the update rate). Go to main and in the rung labeled 1 you will find where to update the SKT_Timeout variable and the Flasher values. These variables are set to be universal across each Hyperion device connection. Set these to be much greater than the scan rate. The greater the scan rate the closer these values can be together.



If you are struggling to establish an initial connection to a server, set the scan rate lower and set the Flasher_set variable and SKT_Timeout timer higher. Verify that you are struggling to connect to a server if the client is stuck in the OPEN bit / not energizing READ or WRITE.



Troubleshooting / Server Connection

When troubleshooting the problem will most likely be in the initial connection to a server from a given client.

For an unstable connection/ struggling to connect

→ **DECREASE** SCAN RATE

→ **INCREASE** SKT_timeout, and Flasher_set

If a connection is still struggling to read, unplug the ethernet cable from the Hyperion device and wait for it to say “ ** OBTAINING IP ** “ and plug it back in

For increasing temperature update rate

→ **DECREASE** SCAN RATE

→ **DECREASE** SKT_timeout, and Flasher_set

For reducing PLC processor load

→ **INCREASE** SCAN RATE

→ **INCREASE** SKT_timeout, and Flasher_set

Scan rate should always be less than SKT_timeout and Flasher_set magnitude

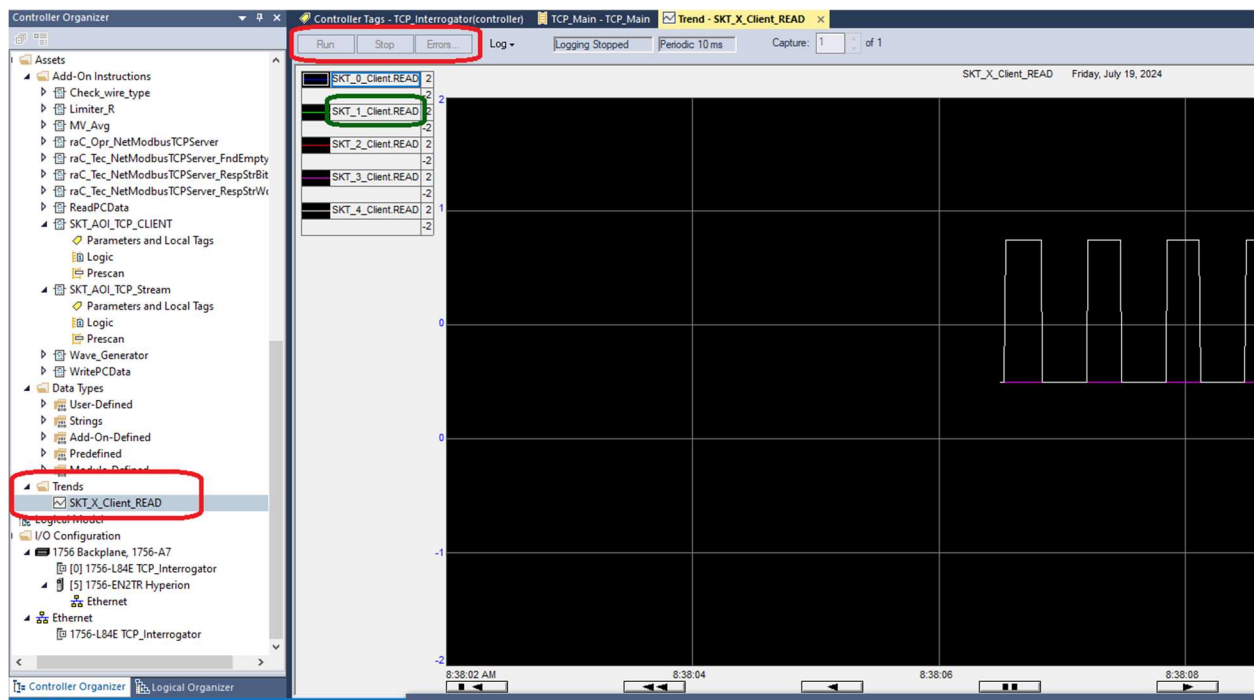
The SKT_timeout and Flasher_set should be relatively similar values, the SKT_timeout may be a slightly larger number.

Scan rate << Flasher_set =< SKT_timeout

An easy way to monitor how fast a client is getting a read buffer is by trending the read bit of a client.

Depending on the client that you want to monitor you will trend the SKT_X_Client.READ bit where X is any number from 0 through and including 4 (correlated to each client and IP address).

There is a trend that is already set for each of the current clients in the controller organizer.



Double click any of the SKT_X_Client.READ variables under the Run, Stop and Error buttons to open the menu that allows you to toggle the visibility of the bits being trended (if you only want to focus on one connection.)

Each rising Edge of the square wave is a read, if you change the Scan rate, Flasher_set, and Skt_Timeout , the period and properties of this square will change, this is normal as long as the square wave has a constant and consistent period.

If a client is reading but you are unable to read temperature values or are not receiving all the temperature values make sure you have the correct mask applied for the given channel of the given Client connection. If a mask is applied, for example, to channel 15 of SKT_1 but your connection is to SKT_0 then you will have to update the mask value if TCP_Float_Convert0

Space Intentionally Left Blank
