2500 Series®
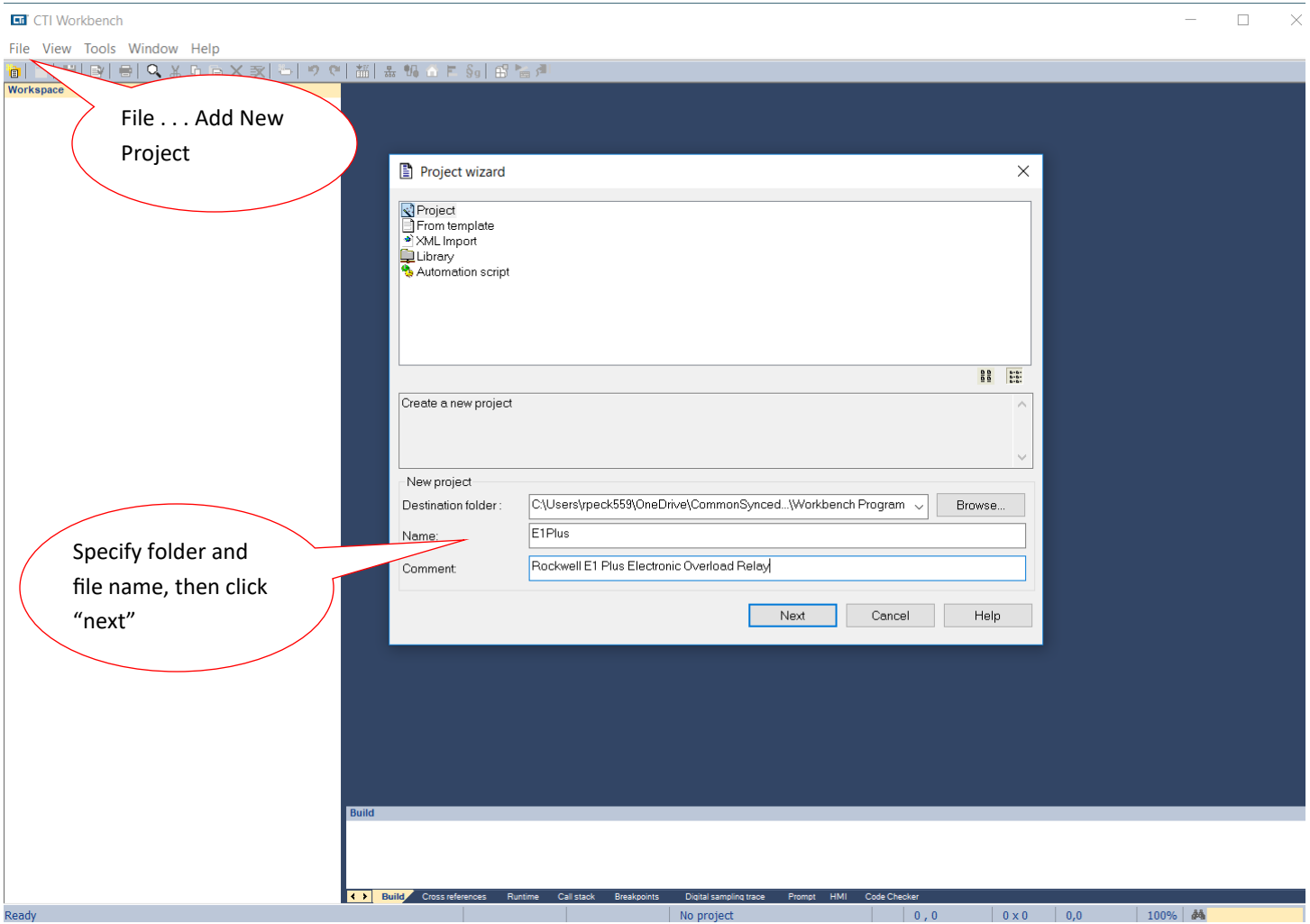## Programmable Automation Control System

Communicating between 2500 Series® Processors and E1 Plus Electronic Overload Relay using Ethernet/IP and 2500P-ACP1

The 2500P-ACP1 Application Coprocessor supports Ethernet/IP communications with up to 40 Ethernet/IP devices via I/O Scanner, I/O Adapter, Explicit Message Adapter, and Tag Client interfaces. This Application Note shows how to configure the ACP1 for communications with Rockwell E1 Plus Electronic Overload Relay using the 193-ETN Ethernet/IP Side Mount Module. In this example we will use a data structure to hold the drive variables. Use of a data structure allows us to quickly add more drives into the application.
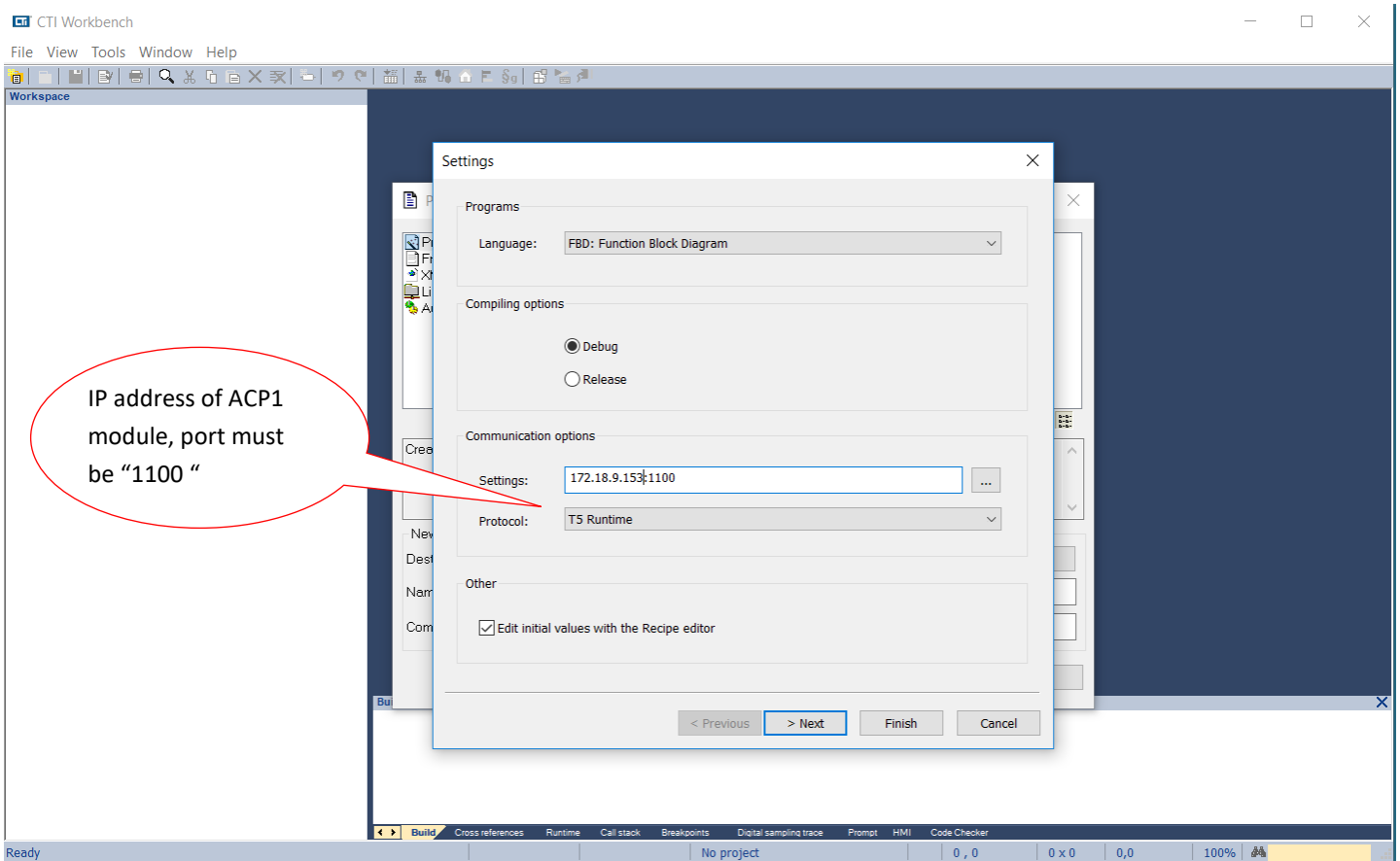
IMPORTANT NOTE: Configuring Ethernet/IP communications requires 2500P-ACP1 Firmware V3.03 or above, and Workbench V1.3 or above.

Step 1: Open a Project.

File . . . Add New Project

Specify folder and file name, then click "next"

CTI Workbench

File View Tools Window Help

Workspace

Project wizard

- Project
- From template
- XML Import
- Library
- Automation script

Create a new project

New project

Destination folder : C:\Users\rpeck559\OneDrive\CommonSynced...\Workbench Program [Browse...]

Name: E1Plus

Comment: Rockwell E1 Plus Electronic Overload Relay

[Next] [Cancel] [Help]

Build

Build   Cross references   Runtime   Call stack   Breakpoints   Digital sampling trace   Prompt   HMI   Code Checker

Ready   No project   0 , 0   0 x 0   0,0   100%

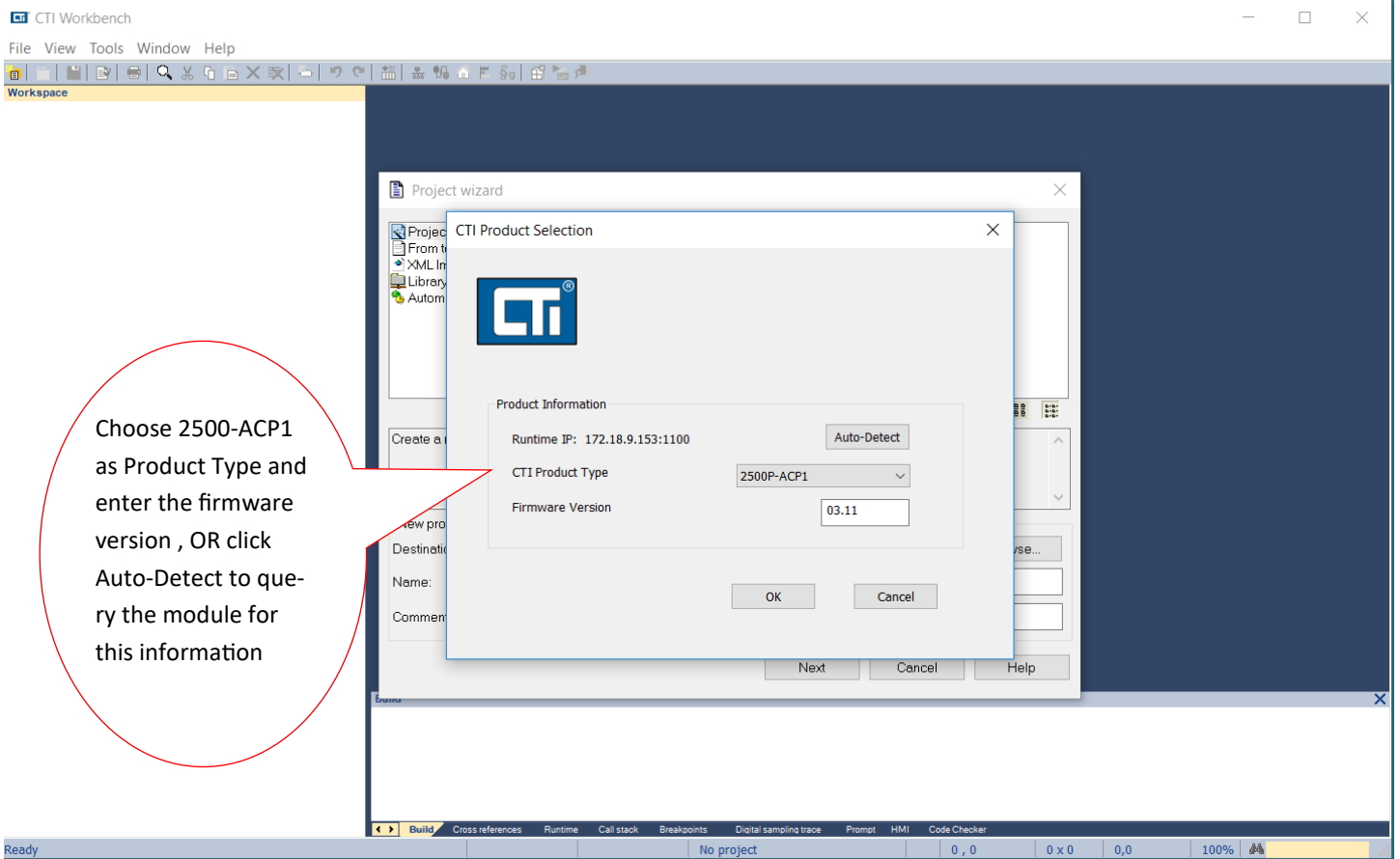ROCK SOLID PERFORMANCE. TIMELESS COMPATIBILITY.

Step 2: Specify Target's (ACP1 module) IP address. **Language** specifies the start-up mode and can be changed later. **T5 Runtime** is the protocol native to Workbench and the ACP1 module (and the Zenon HMI software as well). Port# 1100 is the defined port for interface between Workbench and the ACP1 module. Then click **Next**.

IP address of ACP1 module, port must be "1100 "

Settings

Programs

Language: FBD: Function Block Diagram

Compiling options

○ Debug
○ Release

Communication options

Settings: 172.18.9.153:1100

Protocol: T5 Runtime

Other

☑ Edit initial values with the Recipe editor

< Previous   > Next   Finish   Cancel

Step 3: The ACP1 is the Scanner and the E1Plus is the Adapter, so select **Ethernet/IP I/O Scanner (client)**. We are not using **Binding over Ethernet** but this can remain checked with the default Port# of 9000. (This is used for communications between ACP1 and similar devices using the Data Exchange protocol.) Then click **Finish**.


Select Ethernet/IP I/O Scanner

ROCK SOLID PERFORMANCE. TIMELESS COMPATIBILITY.
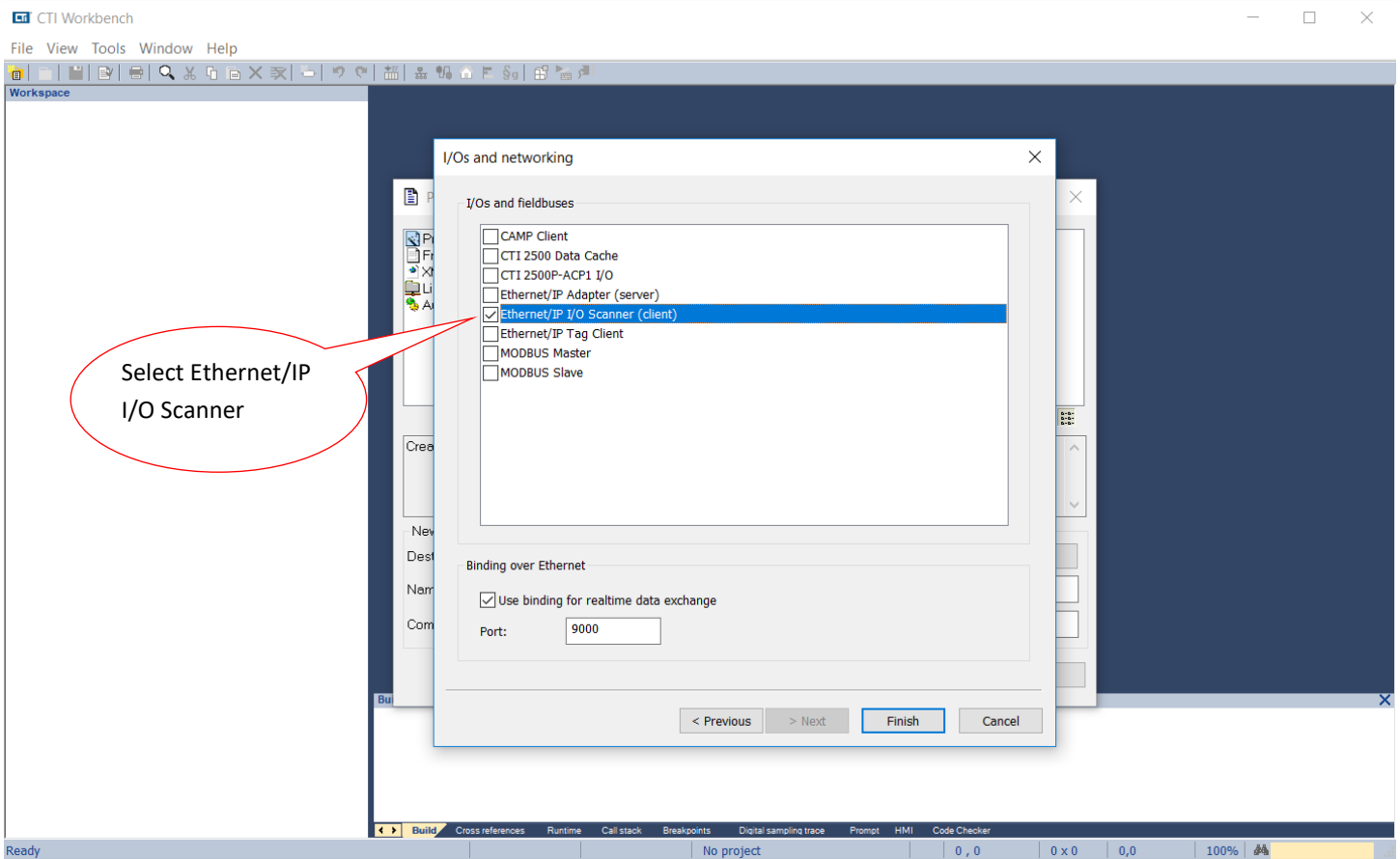
Step 4: If you are connected over the network, choosing **Auto-Detect** will connect to the specified IP address and return the **Firmware Version** of the ACP1 module. Choose 2500P-ACP1 as the **CTI Product Type** and then click **OK**.

Choose 2500-ACP1 as Product Type and enter the firmware version , OR click Auto-Detect to query the module for this information

Step 5: Double-click on **Fieldbus Configurations**. Because we already specified the Ethernet/IP I/O Scanner (client) in Step 3, this driver automatically appears in the configuration window.
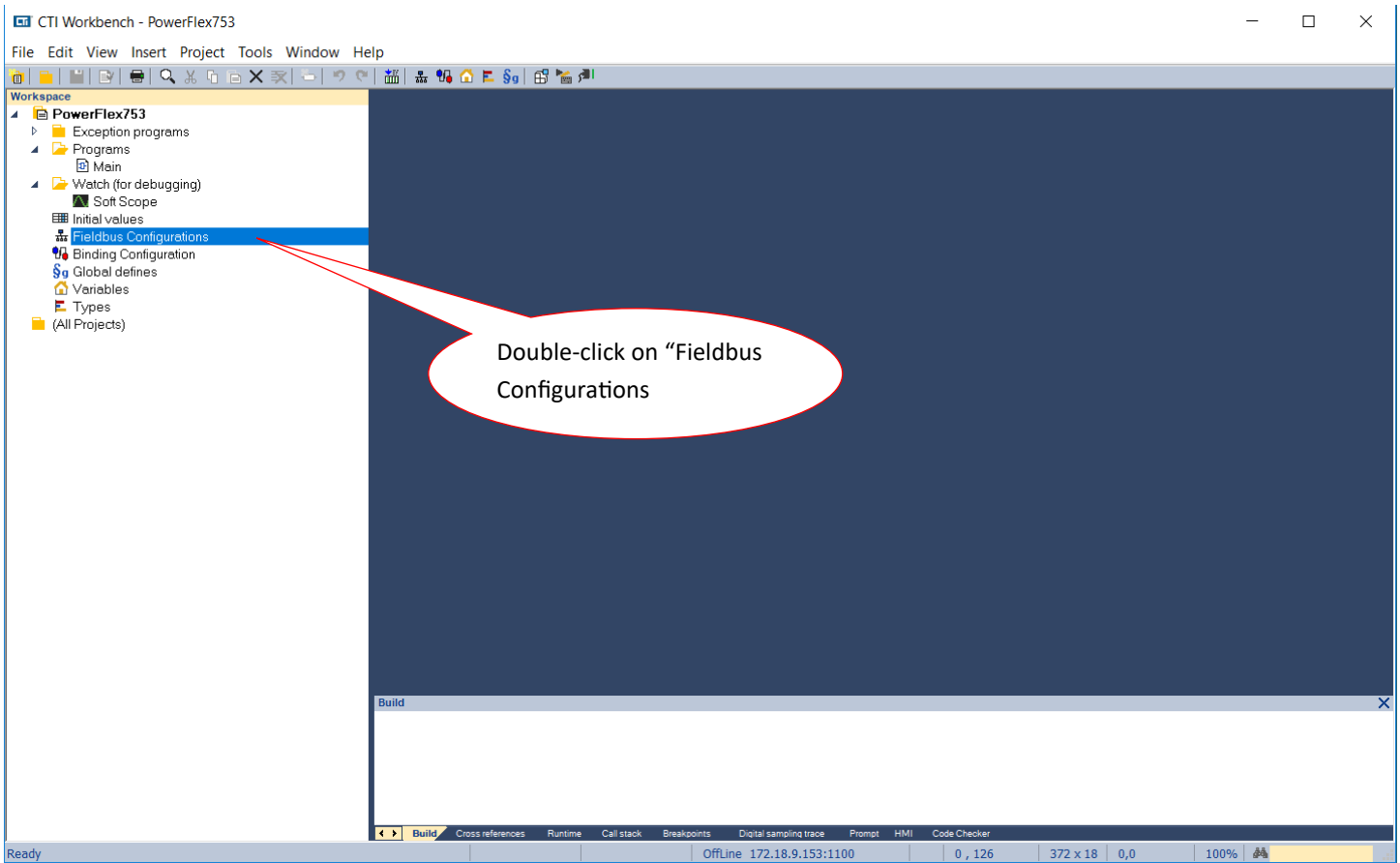
Step 6: Click on the **Insert Master/Port** symbol, then type in the **Address** of the PowerFlex drive in the Server pop-up box. Add optional **Description.** The **Configuration Instance** is defined by Rockwell as "120" (reference "Bulletin 193 E1 Plus EtherNet/IP Side Mount Module - 192-UM012B-EN-P June 2011" page 42—excerpt below). Then click **OK**.

Click on "Insert Master/Port

Enter the IP address, a name for the drive, and set Configuration Instance =6
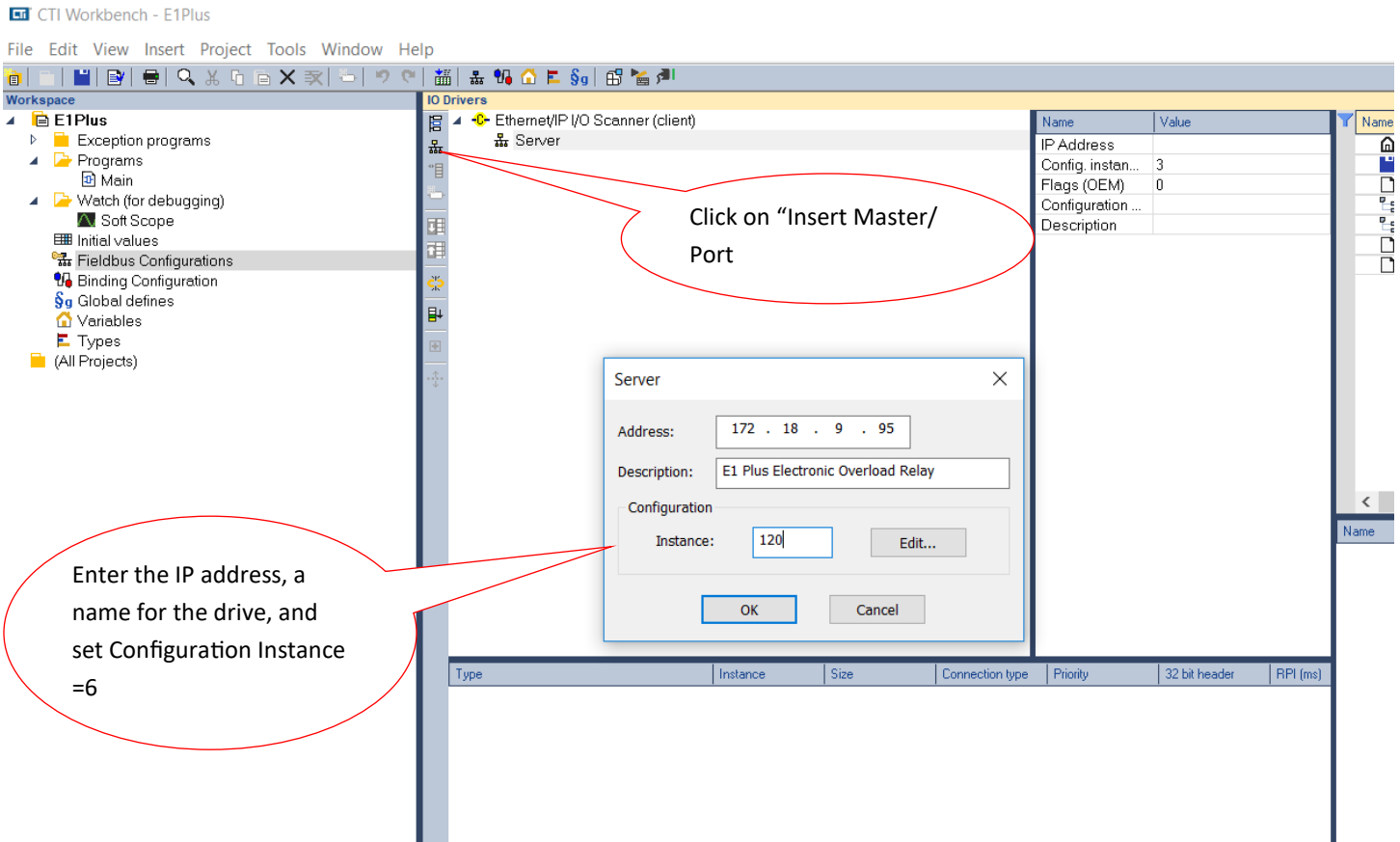
6. Set the Connection Parameters. I/O data is accessed using Input Instances 50, 51, 106, 110 or 111 and Output Instances 2, 101 or 103. The size of the input connection and the output connection shall correspond to the size of the chosen instance. The E1 Plus configuration assembly instance is 120. In this example configuration data is not used, so the data size is set to 0.

Step 7: Expand the **Server** and double-click the **Target to Originator** (Input). In the **IO/Object** pop-up box, change the **Instance** to "111" and the **Size** (in bytes) to "22". Referring to the table on page 10 ("Bulletin 193 E1 Plus EtherNet/IP Side Mount Module - 192-UM012B-EN-P June 2011", page 84). Change the **Priority** to "High" and leave the **32 bit idle header** unchecked. In "Description" we usually enter "E1Plus to ACP1" to make it easy to remember the direction of this data. Then click **OK**.

Step 8: Double-click the **Originator to Target f**or the Output definition. The 32 bit header is assumed here and therefore the **32 bit idle header** box is checked. The **Instance** and **Size** are similarly derived from the table referenced in the previous Step. Enter "ACP1 to Drive" for description, then click **OK**.

Table 22 - Instance 111 —Complete Motor Starter Input Assembly

| Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Motor Current		Input 2	Input 1		Out A Stat	Warning	Tripped
1	Unused							
2	Average % FLA (low byte)							
3	Average % FLA (high byte)							
4	%Therm Utilized (low byte)							
5	%Therm Utilized (high byte)							
6	Trip Status (low byte)							
7	Trip Status (high byte)							
8	Warning Status (low byte)							
9	Warning Status (high byte)							
10	Device Status (low byte)							
11	Device Status (high byte)							
12	Trip Log 0 (low byte)							
13	Trip Log 0 (high byte)							
14	Trip Log 1 (low byte)							
15	Trip Log 1 (high byte)							
16	Trip Log 2 (low byte)							
17	Trip Log 2 (high byte)							
18	Trip Log 3 (low byte)							
19	Trip Log 3 (high byte)							
20	Trip Log 4 (low byte)							
21	Trip Log 4 (high byte)							

From Rockwell publication 193-UM012B-EN-P June 2011, pages 83-85

Table 17 - Instance 103 — Similar to Basic Starter Output Assembly from ODVA Starter Profile

Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0						Trip Reset		Output A

Step 9: There are three system variables we want to add. Highlight the **E1Plus to ACP1** connection, right click and select "insert variable". Click the Diagnostic/Control radio button and select "Server OK", then click the OK button. Repeat this process to add the "I/O Connection OK" and "Last UCMM error" variables.
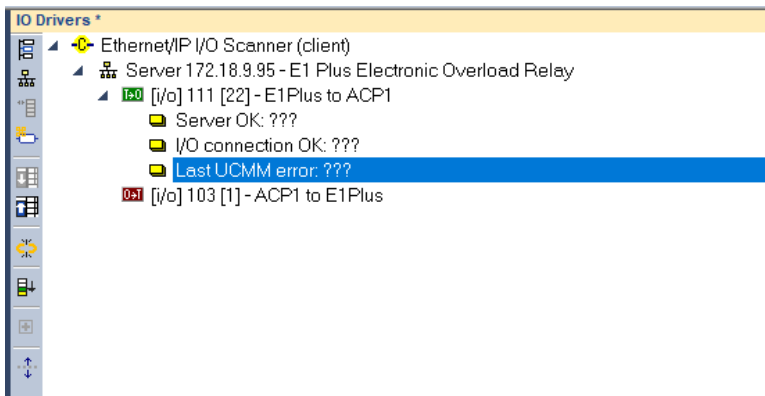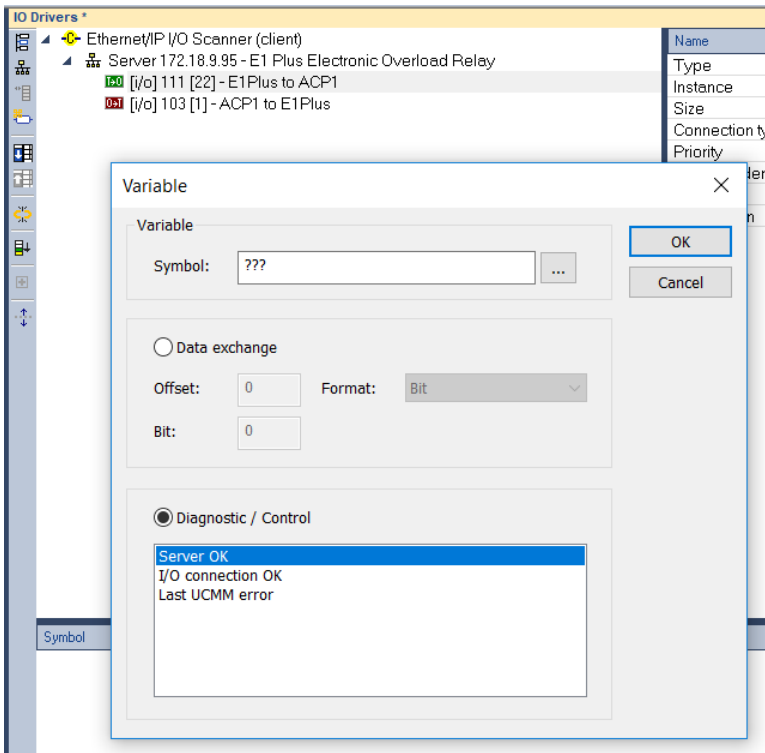

Step 10. Next we want to set all these driver variables as integers. To do this, highlight each variable in turn, and in the editing pane to the right, double-click the "format" field and set the format as "16-bit unsigned". Do this for each of the three driver variables.
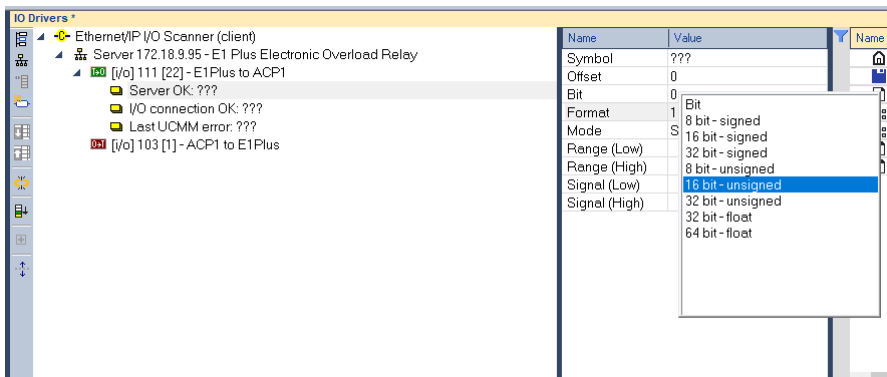
Step 11. Now we will create a data structure which holds all the variables for the E1Plus. Using a structure allows us to rapidly add multiple E1Plus devices into the application without creating a new set of variables for each instance.
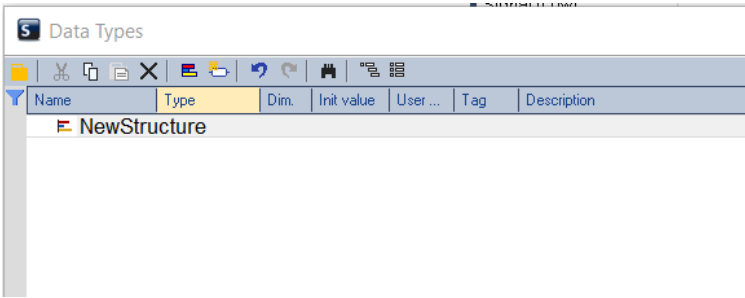
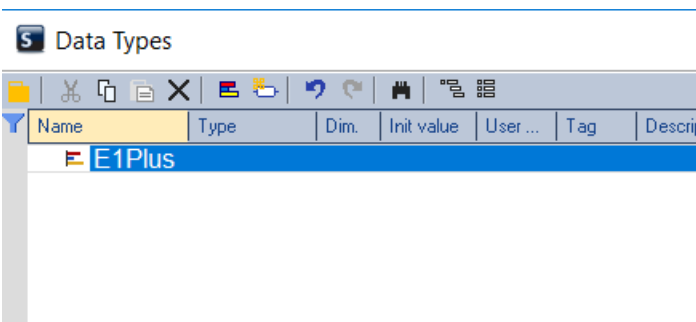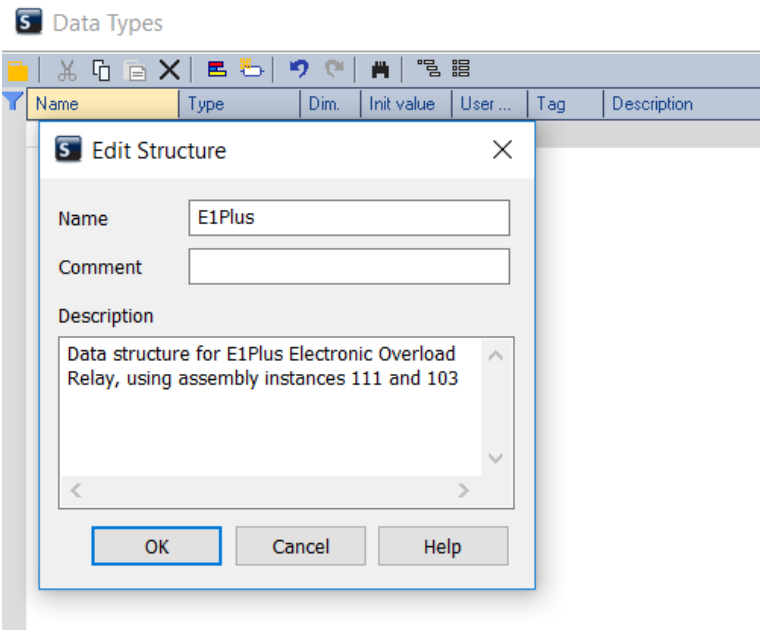Open the data structure window by clicking on the icon in the toolbar. The "Data Types" window will open:
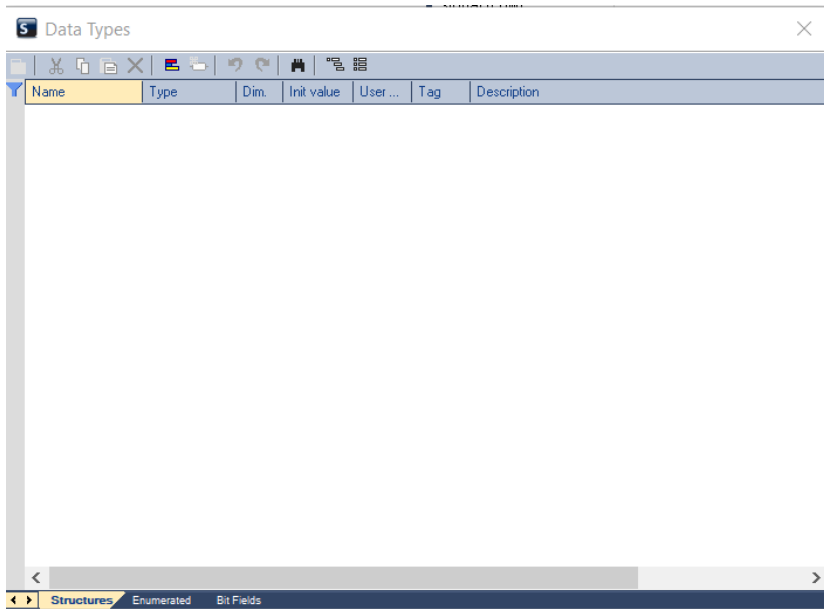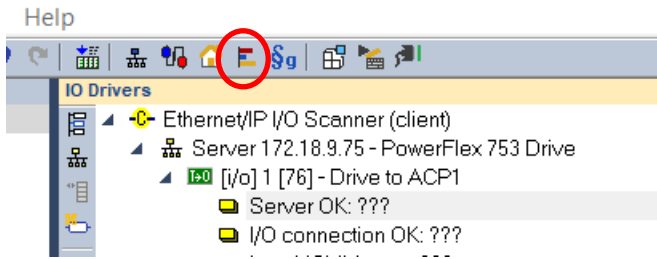
Click the "Insert Type" icon in the toolbar in the Data Type window.
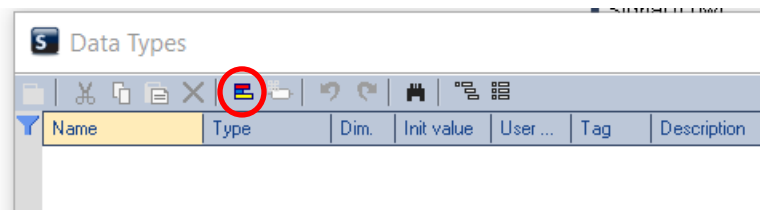
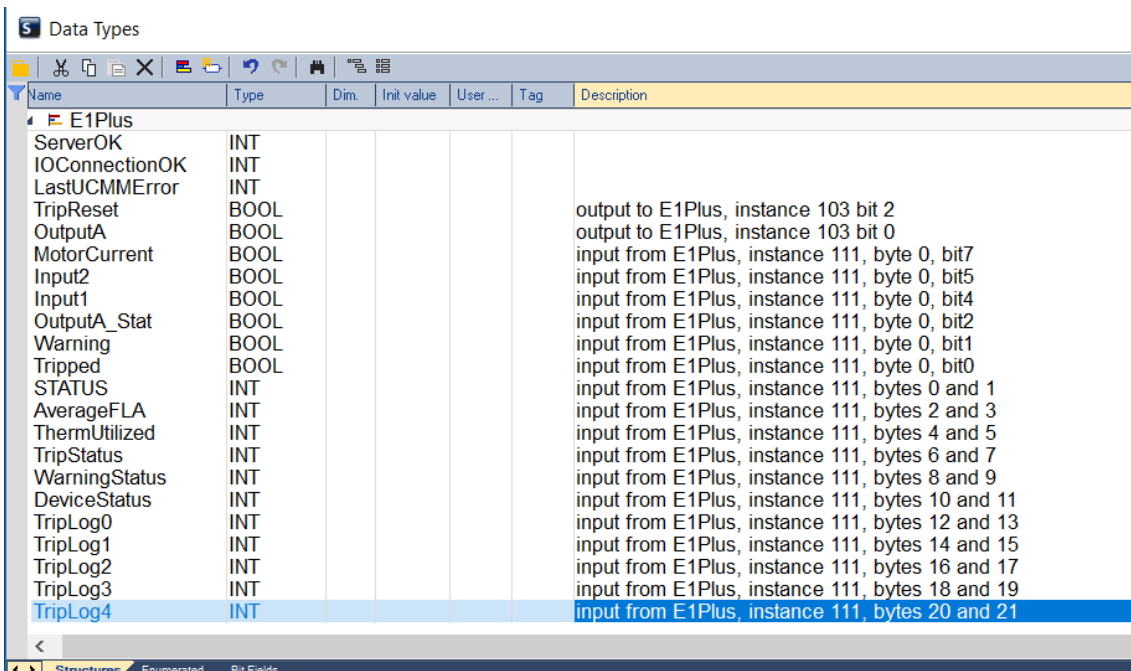A NewStructure type will be created.

Double click the NewStructure name to bring up the editing box. Enter the name for the structure and a description. Click OK.

Now we will add the elements to the structure. Use the "Insert Variable" icon to add each element. First add the driver variable names which will be tied later to the driver variables. Set each type to "INT" (integer).


The "insert variable" icon is here
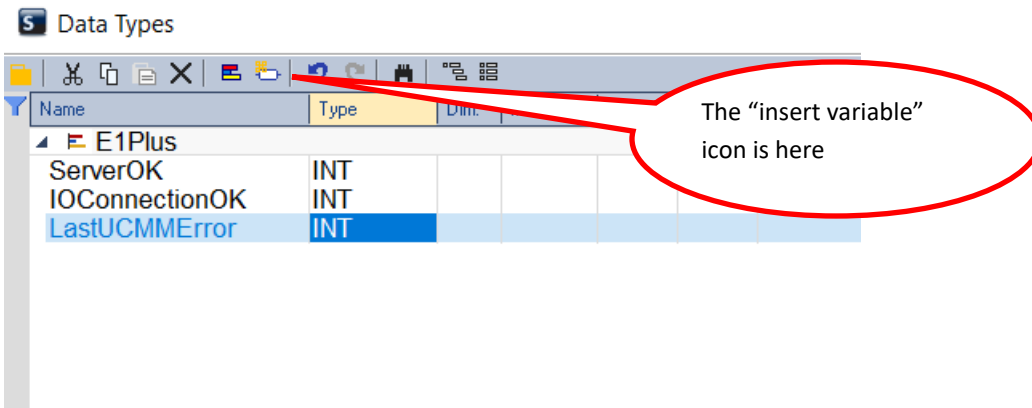
Referring to Table 22 and Table 17 on page 10, add all the other E1Plus parameters, first from Table 17 then from Table 22. "Output I/O" list, then from the "Input I/O—Generic" list. Set the type for each as shown below. Put a description if you want. Close the Data Type window when finished.

| Name	Type	Dim.	Init value	User ...	Tag	Description
⊿ ⋿ E1Plus						
ServerOK	INT					
IOConnectionOK	INT					
LastUCMMError	INT					
TripReset	BOOL					output to E1Plus, instance 103 bit 2
OutputA	BOOL					output to E1Plus, instance 103 bit 0
MotorCurrent	BOOL					input from E1Plus, instance 111, byte 0, bit7
Input2	BOOL					input from E1Plus, instance 111, byte 0, bit5
Input1	BOOL					input from E1Plus, instance 111, byte 0, bit4
OutputA_Stat	BOOL					input from E1Plus, instance 111, byte 0, bit2
Warning	BOOL					input from E1Plus, instance 111, byte 0, bit1
Tripped	BOOL					input from E1Plus, instance 111, byte 0, bit0
STATUS	INT					input from E1Plus, instance 111, bytes 0 and 1
AverageFLA	INT					input from E1Plus, instance 111, bytes 2 and 3
ThermUtilized	INT					input from E1Plus, instance 111, bytes 4 and 5
TripStatus	INT					input from E1Plus, instance 111, bytes 6 and 7
WarningStatus	INT					input from E1Plus, instance 111, bytes 8 and 9
DeviceStatus	INT					input from E1Plus, instance 111, bytes 10 and 11
TripLog0	INT					input from E1Plus, instance 111, bytes 12 and 13
TripLog1	INT					input from E1Plus, instance 111, bytes 14 and 15
TripLog2	INT					input from E1Plus, instance 111, bytes 16 and 17
TripLog3	INT					input from E1Plus, instance 111, bytes 18 and 19
TripLog4	INT					input from E1Plus, instance 111, bytes 20 and 21

Step 12. Now we will attach the variables we created to the Ethernet/IP drive connection. First we need to create an instance of the structure for this drive. In the variable editing window, highlight "Global Variables" and right click. Select "Add Variable"
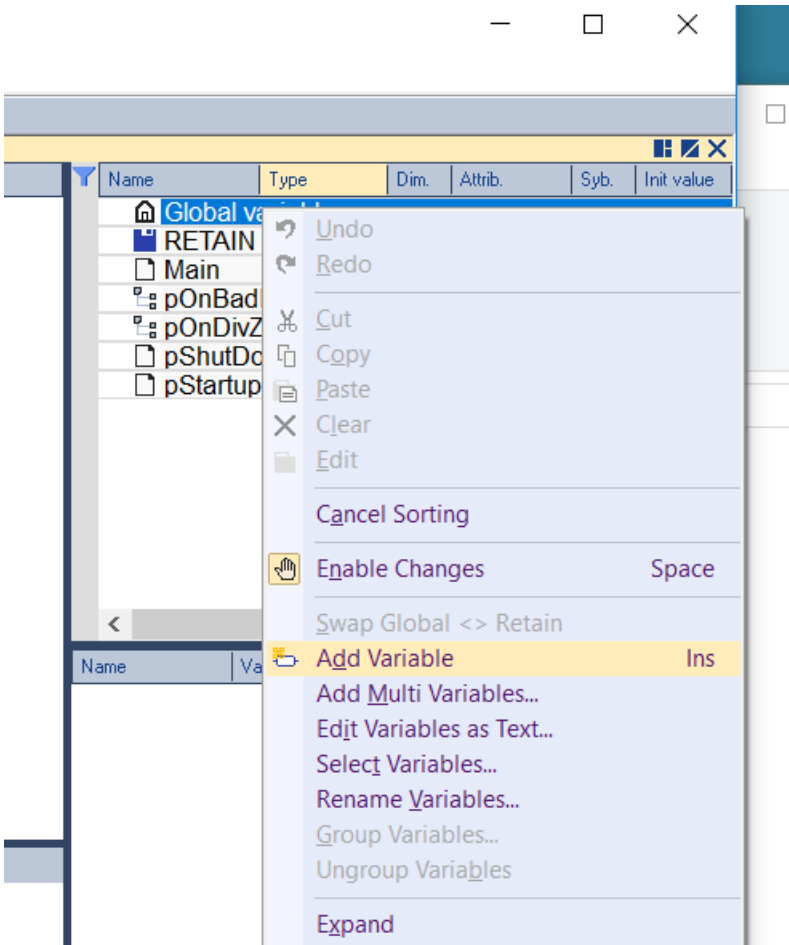

A "NewVar" will be created. Double-click on "NewVar" to rename the variable for this first drive. We'll call it E1Plus00.
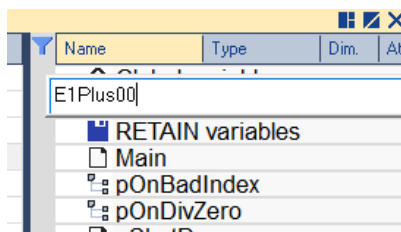
Now double-click on the "type" where it says BOOL, and select "E1Plus" from the drop-down box. This is the data structure we previously created.
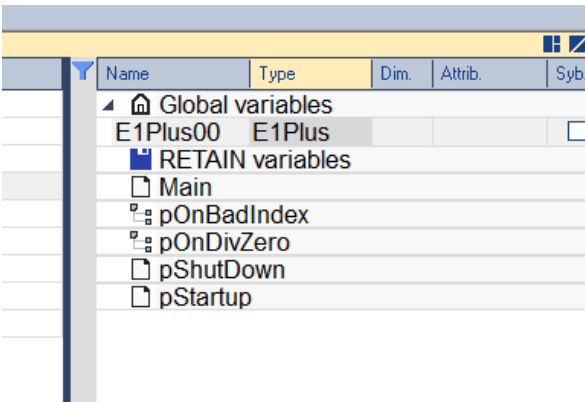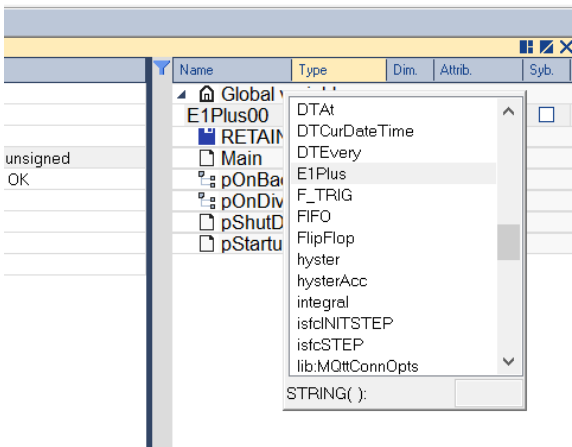
Next, click and hold on the E1Plus00 name and drag it to the box below the variable list. Then expand the list by clicking the arrow to the left of "E1Plus00".
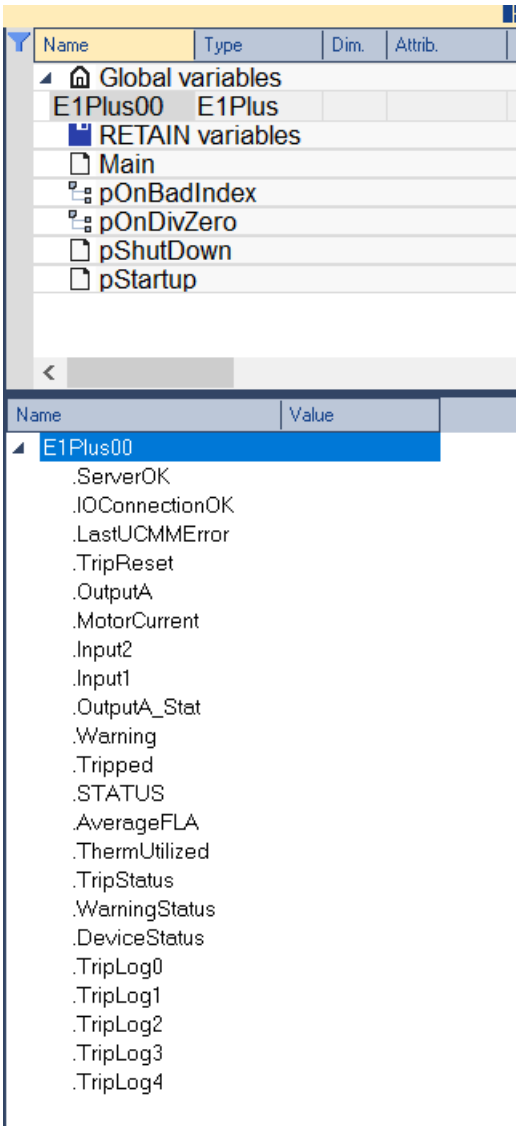
We will use this list to drag variable names over to our Ethernet/IP connections. First, drag the three driver variables (one at a time over to their corresponding locations in the "Drive to E1Plus" connection.
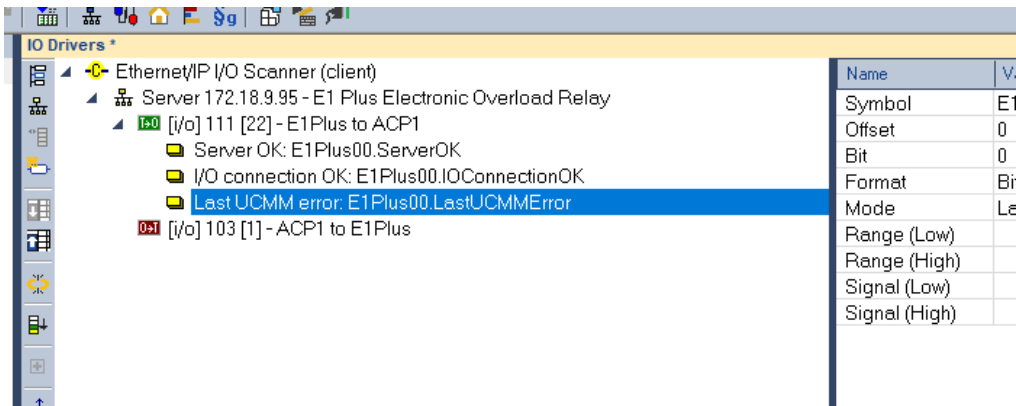
Next highlight the "ACP1 to E1Plus" connection. In the variable list, highlight the .TripReset and .OutputA variables. Drag these into the pane beneath the Ethernet/IP connections.

For the E1Plus00.TripReset variable, double-click in the "bit" column and change the bit from 0 to 2, since this is bit 2 in byte 0 (see Table 17 on page 10).

Step 13. Now repeat step 12, assigning the variables beginning with "MotorCurrent" and ending with "TripLog4" to the "E1Plus to to ACP1" connection. Set the "offsets", "bits", and "formats" as shown below (from Table 22 on page 10).

IO Drivers *

- ▲ **Ethernet/IP I/O Scanner (client)**
  - ▲ Server 172.18.9.95 - E1 Plus Electronic Overload Relay
    - ▲ [i/o] 111 [22] - E1Plus to ACP1
      - Server OK: E1Plus00.ServerOK
      - I/O connection OK: E1Plus00.IOConnectionOK
      - Last UCMM error: E1Plus00.LastUCMMError
      - 0.7: E1Plus00.MotorCurrent
      - 0.5: E1Plus00.Input2
      - 0.4: E1Plus00.Input1
      - 0.2: E1Plus00.OutputA_Stat
      - 0.1: E1Plus00.Warning
      - 0.0: E1Plus00.Tripped

Name	Value
Type	I/O: Inputs (Target to orig...
Instance	111
Size	22
Connection ty...	Point to point
Priority	High
32 bit header	☐
RPI (ms)	100
Description	E1Plus to ACP1

| Symbol	Offset	Bit	Format	Mode	Rang
E1Plus00.ServerOK	0	0	16 bit - unsigned	Server OK	
E1Plus00.IOConnectionOK	0	0	16 bit - unsigned	I/O connection OK	
E1Plus00.LastUCMMError	0	0	16 bit - unsigned	Last UCMM error	
E1Plus00.MotorCurrent	0	7	Bit	Data exchange	
E1Plus00.Input2	0	5	Bit	Data exchange	
E1Plus00.Input1	0	4	Bit	Data exchange	
E1Plus00.OutputA_Stat	0	2	Bit	Data exchange	
E1Plus00.Warning	0	1	Bit	Data exchange	
E1Plus00.Tripped	0	0	Bit	Data exchange	
E1Plus00.STATUS	0	0	16 bit - unsigned	Data exchange	
E1Plus00.AverageFLA	2	0	16 bit - unsigned	Data exchange	
E1Plus00.ThermUtilized	4	0	16 bit - unsigned	Data exchange	
E1Plus00.TripStatus	6	0	16 bit - unsigned	Data exchange	
E1Plus00.WarningStatus	8	0	16 bit - unsigned	Data exchange	
E1Plus00.DeviceStatus	10	0	16 bit - unsigned	Data exchange	
E1Plus00.TripLog0	12	0	16 bit - unsigned	Data exchange	
E1Plus00.TripLog1	14	0	16 bit - unsigned	Data exchange	
E1Plus00.TripLog2	16	0	16 bit - unsigned	Data exchange	
E1Plus00.TripLog3	18	0	16 bit - unsigned	Data exchange	
E1Plus00.TripLog4	20	0	16 bit - unsigned	Data exchange	

Step 14. OPTIONAL -do this step only if you want to read the drive setup data from a PLC and write the drive status data back to a PLC.

Open the fieldbus editing screen.

Click on the "Insert Configuration" icon. From the dropdown list, select CTI 2500 Data Cache. Then click OK.
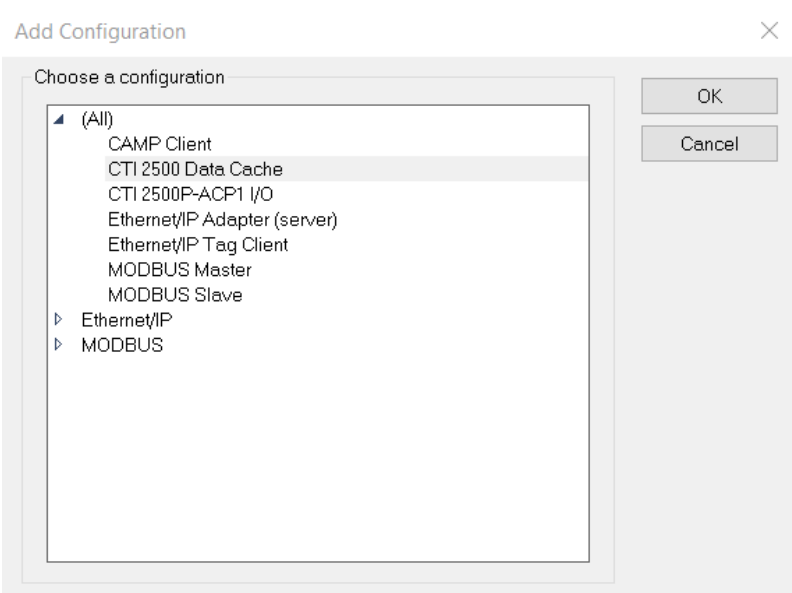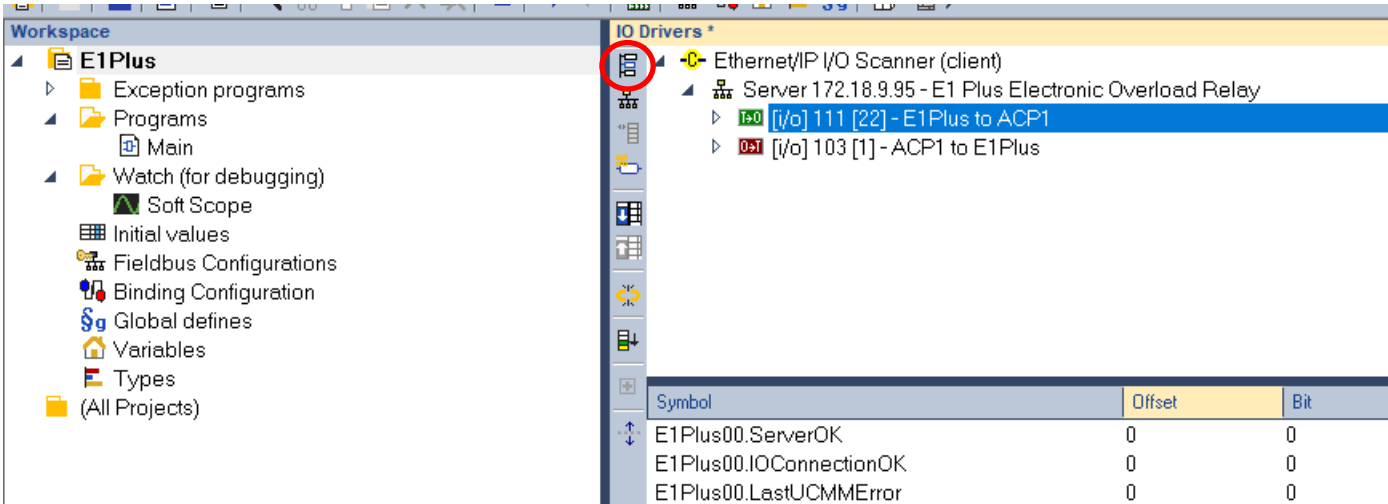
Next, click the "Insert Master/Port" icon.
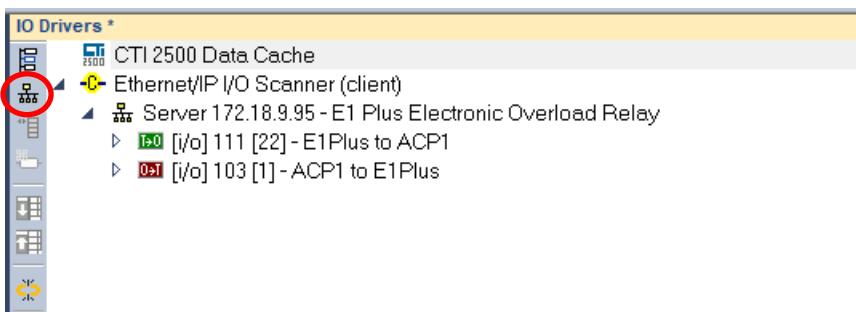
Complete the IP address information for the Host PLC you will be communicating with. Then click OK.

Click on the "Insert Slave/Data Block" icon.

Complete the information for memory type and starting address. Here we will read the drive command information from C-memory starting at C1000.

Click OK to proceed. Repeat the process to add a "Write to C-memory" block beginning at C1010. Repeat the process to add a "Write to V-memory" block beginning at V1000. This is where we'll store the data read from the drive.
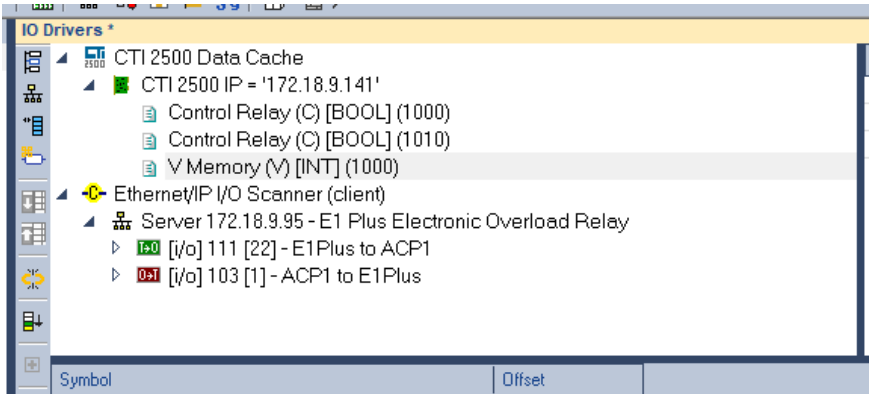

Now we will use the same "drag and drop" procedure we used in step 12, to populate the variables read from / written to the PLC. First highlight the "Read from Control Relay" block (the first block). From the variable list at the right, highlight the variables ".TripReset" and ".OutputA". Drag these into the box below the fieldbus configurations:
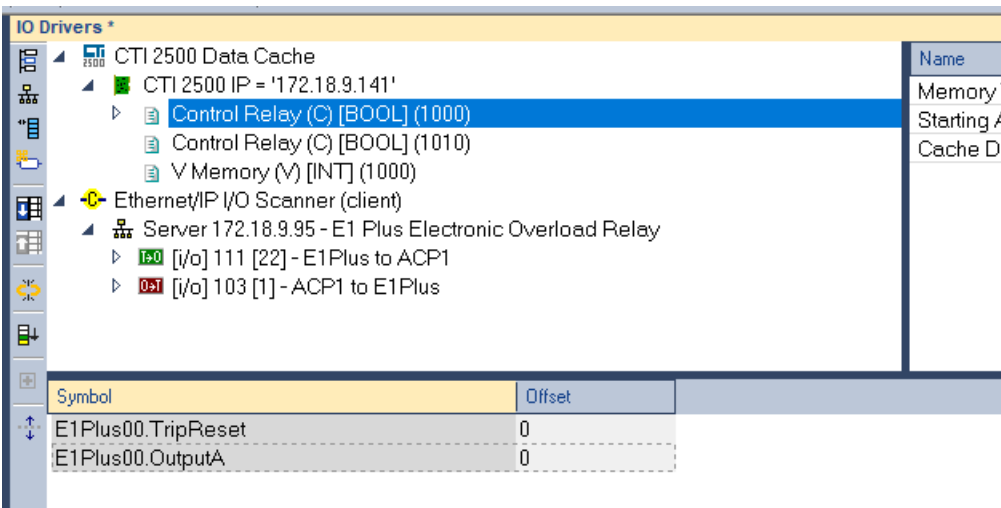

Right-click the "Read from Control Relay" block and select "Renumber Offsets". This will renumber the items into consecutive C-memory addresses.
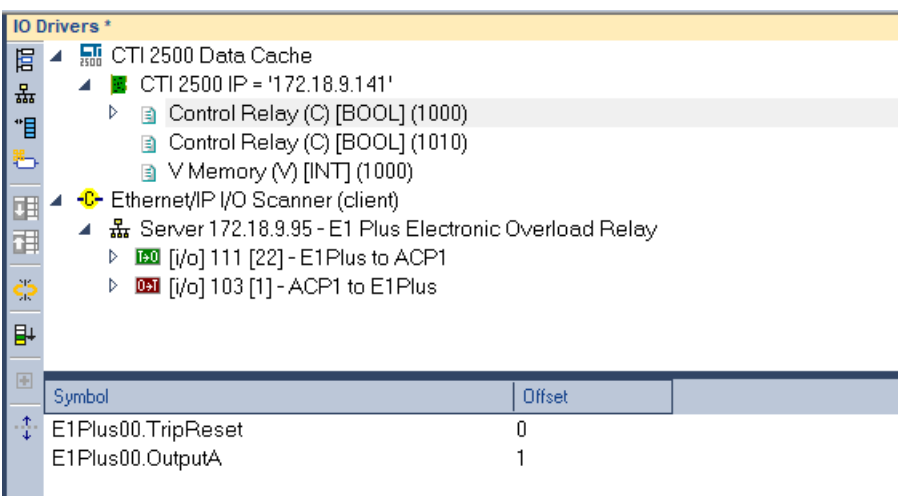
Now highlight the "Write to Control Relay" block (the second block). From the variable list at the right, highlight the variables ".MotorCurrent" through ".Tripped". Drag these into the box below the fieldbus configurations. Do a renumber as we did in the previous step. The result should look like this:
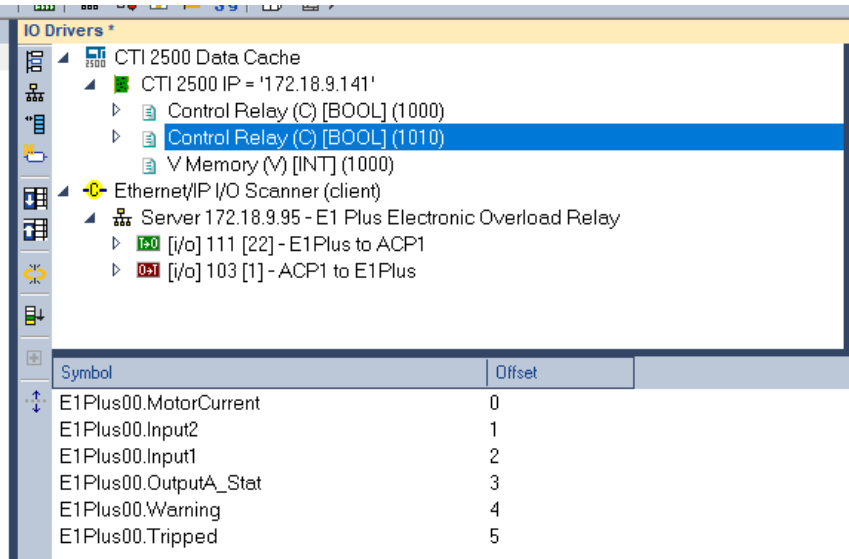
Repeat this process to populate the variables of "STATUS" through "TripLog4" to the Write to V-Memory block (third block). Do a renumber. The result should look like this:
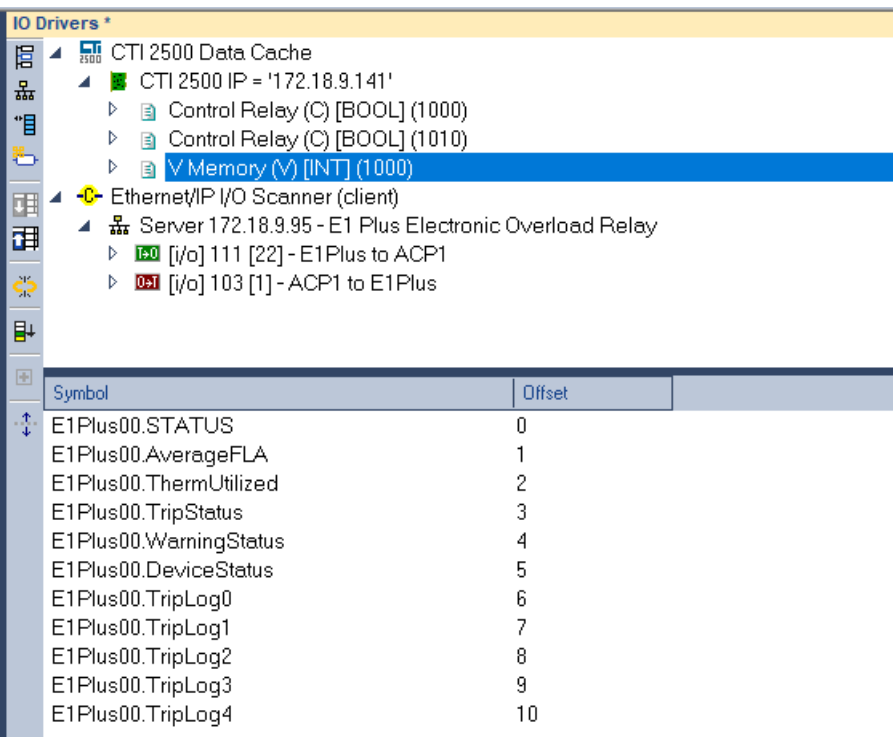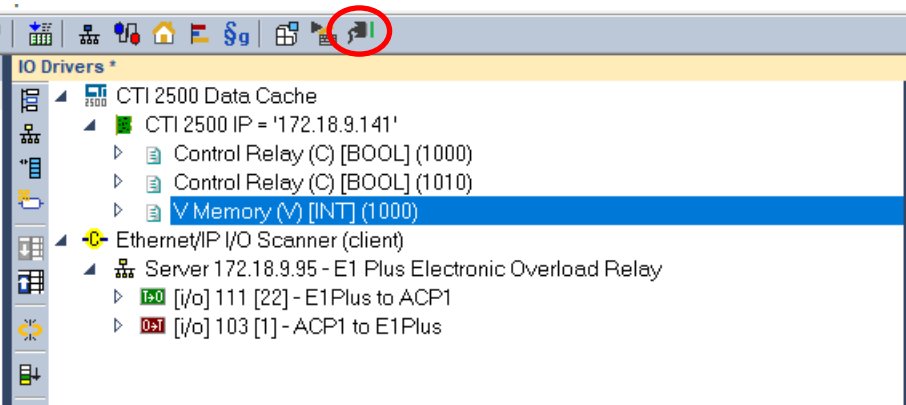
Step 15: Compile and Download. To compile, click the "Compile" icon at the top.
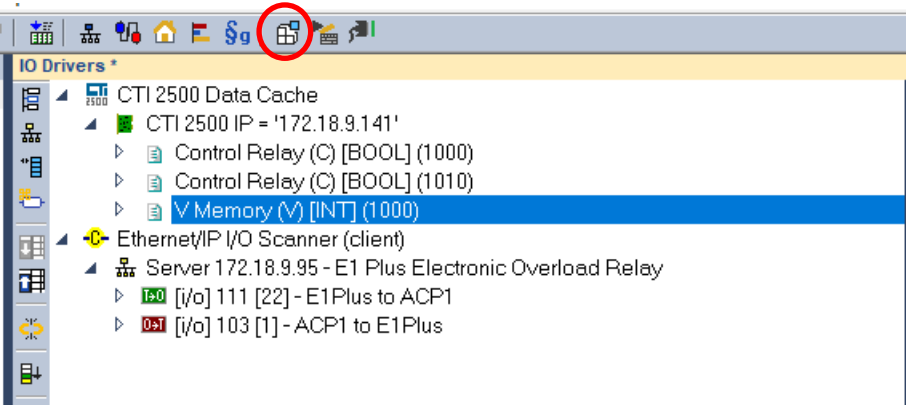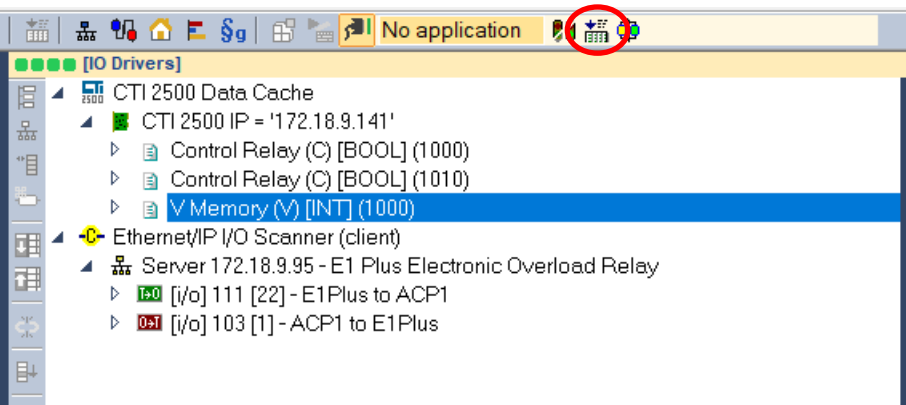

If there are no compile errors (shown in red in the "build" tab at the bottom), then we're ready to download and run.
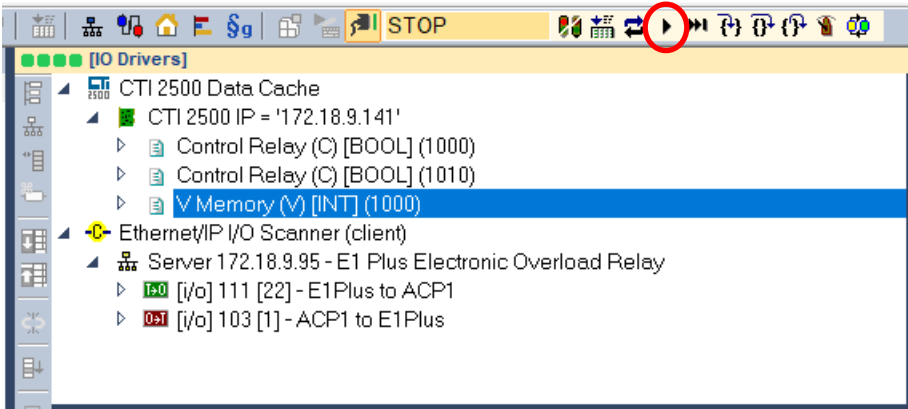
Click the "Online" icon at the top.

Then click the "download" icon:

After the download completes, click the "resume cycle to cycle" icon to start the program:

Control Technology Inc.
5734 Middlebrook Pike, Knoxville, TN 37921-5962
Phone: +1.865.584.0440   Fax: +1.865.584.5720
www.controltechnology.com

ROCK SOLID PERFORMANCE. TIMELESS COMPATIBILITY.