BOA Spot
Communication Guide

TELEDYNE IMAGING
Everywhereyoulook™

Notice

BOA Spot Communications Guide
Document Number 405-00061-00

Teledyne Digital Imaging US, Inc.
Information: TDI_Sales.ipd@teledyne.com
Support: TDI_Support.ipd@teledyne.com
Web: http://www.teledynedalsa.com/visionsystems

700 Technology Park Drive
Billerica, MA, USA 01821

Tel 1.978.670.2002 **Fax** 1.978.670.2010

Table Of Contents

In This Manual

Blue text such as **Tip** or **Note** indicates useful information.

Green text in a code example is a comment to the code, such as //Send result to PLC

Blue text underlined is a hyperlink. Black text underlined is for emphasis.

Red text such as **Note** indicates important or critical information.

This manual is a collection of Application Notes, Examples and Instructions written by several people for Teledyne Digital Imaging US, Inc. products. These have been edited and reformatted for this manual.

Introduction

Overview

This manual describes configuring the BOA Spot for communication with serial and TCP ports, and with several different types of PLCs. In some cases, instructions are also given for adding the BOA Spot into the PLC program environment.

Please refer to the *BOA Spot Quick Start Guide* for instructions on changing the BOA Spot IP Address using the iDiscover or Nexus for BOA utilities. You must configure the BOA Spot address to be compatible with your PLC network. The Quick Start Guide has a Getting Started section that introduces programming the BOA Spot.

Note: Every BOA Spot ships with a default address 192.168.0.100. This address is a default used by most network enabled devices including WIFI routers, which will interfere with communicating with the BOA Spot if the address is left at 192.168.0.x.

This manual does not cover programming the PLC. Advanced knowledge of the language specific to your PLC is required to program the factory floor environment. Please refer to the PLC Manufacturer or Distributor for training or tutorials.

Note: Most PLCs have a separate read space and write space. They do not use the same registers for both inputs and outputs. This means you cannot write a value to a register and read it back again. The PLC must be programmed to copy the value to a register you can read or write expected values into registers you can read.
Tip: Start small, by writing a single string or number to your destination or PLC. You can add more instructions and controls after you have successfully established communication between devices.

Tip: In most cases, it is useful to start with the BOA Spot using the Internal Timer, and later change the trigger source to suit your environment. You may need to increase the timing or timeout in the PLC program when you change to triggered acquisition.

- Internal Timer is useful for focusing the camera on a still object.
- External Trigger is used with moving parts and a part-in-place sensor.
- Software Trigger is used with a trigger signal from a PLC, or a command from another device.

Some examples show a “**heartbeat**” variable. This is to let the PLC program know the camera is “alive” and functioning. Some PLC programs require a heartbeat from all connected devices, and some do not. Some programmers prefer to always include the heartbeat, while others question its usefulness.

Some examples have a “**inspBusy**” variable, to signal the PLC the inspection program is executing. This variable is not fully implemented in the example. You could optionally send the busy status to the PLC if it is useful to your programming.

PLC Names and Protocols

There are several different types of PLC communication “Protocols” developed by different companies. This is only a partial list.

PLC stands for Programmable Logic Controller. Usually PLCs control a distributed system of sensors, relays, switches and controls on an industrial production line.

Protocol refers to a language or a communication standard. It can refer to software or hardware methods of communication.

EIP stands for **Ethernet IP**, a term used generally for communication over an Ethernet network.

SLMP (Seamless Message Protocol) for linking Ethernet IP to equipment from different manufacturers and different hardware, such as CC-Link. SLMP support introduced in firmware version 2224.

TCP stands for Transmission Control Protocol.

UDP stands for User Datagram Protocol.

AB or **Allen Bradley** systems may use either the **Ethernet IP** or **Control Logix** protocols on Ethernet. The Allen Bradley brand is owned by Rockwell.

FANUC Corp. makes systems which use **SNP** on Serial, and **SRTP/FINS** on Ethernet.

Siemens makes the **Simatec** systems, which use the **Profinet UDP** or **Profinet RT** protocol on Ethernet, supported by BOA Spot. **Profinet IRT** is a hardware-accelerated version, which is not supported on the BOA Spot.

Mitsubishi systems use the **MELSEC** protocol, available on both Serial and Ethernet.

Several manufacturers produce systems which use the **Modbus** protocol on Serial or EIP. Modbus is an open source, royalty free protocol used by several companies. The rights to development and maintenance were transferred to The Modbus Organization.

OMRON systems use multiple protocols. BOA Spot supports: Ethernet IP protocol, OMRON C protocol on Serial, OMRON C UDP on Ethernet.

Rockwell produces **Compact Logic** and **RSLogix** systems, which may use either the **Ethernet IP** or **Control Logix** protocols on Ethernet. BOA Spot software works with Rockwell Control Logix products using versions R18 through R30.

Yaskawa Electric Corp. and Yaskawa **Motoman** produces Robotic and automated systems which use the Motoman protocol, available on Serial.

EtherCAT is a hardware-accelerated system developed by **Beckhoff Automation**. EtherCAT is not supported by the BOA Spot.

Before You Begin

There are several important configuration changes to make and things to consider before you start. The *BOA Spot Quick Start Guide* contains detailed information on configuring the PC and the BOA Spot.

Windows Firewall The Windows Firewall should be **turned off** or customized on systems that communicate over the network (TCP or EIP). This does not apply to devices using Serial communication. The *BOA Spot Quick Start Guide* contains detailed information on configuring the PC, including the Windows Firewall.

BOA Spot Name and Address All BOA Spot models ship with the same Network Name or Alias “boaspot” and IP Address 192.168.0.100. **Change the address to be compatible with your PLC Network.** If you are connecting more than one BOA Spot on the same network or on the same PLC, you must change the IP Address so each BOA Spot has a different address. You should also change the name so each BOA Spot has a different name. This is important and even critical in some PLCs and network configurations, to avoid confusion. Use the Device Setup Page (before version 2150) or iDiscover utility to change the BOA Spot names and addresses. These are described in the *BOA Spot Quick Start Guide*. **Tip:** Names cannot start with a number.

BOA Spot Name in Profinet The GSD file that describes the BOA Spot to **Profinet** uses a default name “boa”. Earlier versions of the GSD file used the name “boavision”. The Profinet environment uses this as the default “Device name”. The TIAP software assigns a “Converted name”. If there is only one BOA Spot the two names are identical. If there are multiple BOA Spots connected, Profient appends numbers to the default Device name and examines the first four characters of a name. If they are not unique, a unique “Converted name” is created. You can edit the name in the Profinet environment. **Tip:** Change the Network names of the BOA Spot using iDiscover or the Device Setup page. They should be different in the first four characters (Top, Left, Right, First, Second, Third, etc.). When you configure the Profinet environment, add just one BOA Spot at a time and change the default names to match the Network names you used in iDiscover or the Device Setup Page. Names cannot start with a number.

Note: The Profinet factory development software may not function fully in Windows 10. We highly recommend using a Windows 7 system to run the Siemens software to create or monitor connections. Update your software with any available patches.

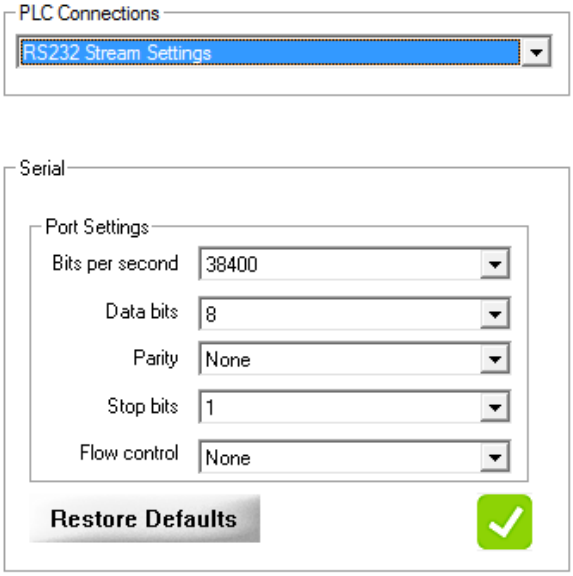
IMPORTANT: Editing the measurement Tools or Scripting changes the BOA Spot "run mode” and disables solution switching. Editing scripts can interrupt communication with PLCs and peripherals. If you encounter problems, Save your Solution and then reload it (same solution). Loading a solution resets the run mode, unless you have selected the "Load for Edit/Disable switching" option.

Serial Port

The Serial port or “RS232 Stream” is used for general communication such as messaging or data.

Configure the BOA Spot

1. Load or create a simple Solution with one or two measurements.
2. In the Navigation bar, click on Setup Connections.
3. In the drop-list, select RS232 Stream Settings. The center of the Setup panel changes, to configure BOA Spot for Serial communication. The standard serial port parameters are available for Baud Rate (bits per second), Data bits, Stop Bits, Parity and Flow Control.
4. Configure BOA Spot to match the settings used by your devices. Settings must match for correct operation.
5. Click the “Apply” button.



Note: Not all serial devices support all the available settings. BOA Spot does not support Baud Rates 110, 300, 1200 and 14400. Flow Control “software and “none” are supported, “hardware” is not supported.

To test your setup:

Requires a PC running HyperTerminal, Telnet or TeraTerm.

1.Create or load a simple Solution and configure the BOA Spot Serial port.

2.Click on the “Edit Scripts” button in the Navigation bar.

3.Click on the “Post Processing Function” and open the Edit window. Enter one (or both) of the following two statements:

COM3 = “Hello” // COM3 is the only port supported !

or

PutPortString(“\r\nHello”) // carriage return and new line

4. Click OK and Run the Solution.

5. Open HyperTerminal, Telnet or TeraTerm on a PC, and configure the Serial Port settings to match the BOA Spot settings.

6. When you connect to the BOA Spot, you should see the message “Hello” repeating on your monitor.

You can create more complex strings using the Free Edit window and the String Formatting window. You can use the following functions in script statements, to read and write to the Serial port:

PutPortString GetPortChar GetPortString WriteFormatString

Tip: The function PutPortString sends the exact string, without evaluating variables. The function WriteFormatString does evaluate variables.

Please refer to “Keyboard / Serial IO Functions” in the *BOA Spot Scripting User Manual* or the BOA Spot on-line Help for full syntax, explanations, and some examples.

Troubleshooting Serial Port Problems

Problem: No communication. Messages from BOA Spot do not appear in HyperTerminal, Telnet or TeraTerm.

Reason: Serial port problems are most often caused by (1) incorrect wiring or (2) incompatible settings between BOA and the connecting equipment.

Solution 1: Check your wiring carefully. **It is very important** to have a ground connection between the serial port on BOA Spot (or PL-101) and the other devices.

Solution 2: Check the serial port settings on your BOA Spot and on your other Serial Port devices. All settings must be identical on both devices. COM3 is the only port supported. The name must be COM3 in instructions that requre a “comVar”.

Note: When the BOA Spot application launches, the baud rate is set to **38400**, unless you are loading a saved Solution with different settings.

Solution 3: Save and reload your solution file. See “Important” on page 6.

Problem: Cannot find HyperTerminal, Telnet or TeraTerm on my PC.

Solution: HyperTerminal is not included on W7 or 10. TeraTerm is a third-party program not included in Windows. Telnet is not enabled by default. Please see page 11.

Problem: How can I verify the Serial port actually functions, or is not broken?

Solution: During the power-up sequence BOA Spot boots with the following serial port settings: 115200 bits per second
8 data bits
No parity
1 Stop bit
No flow control

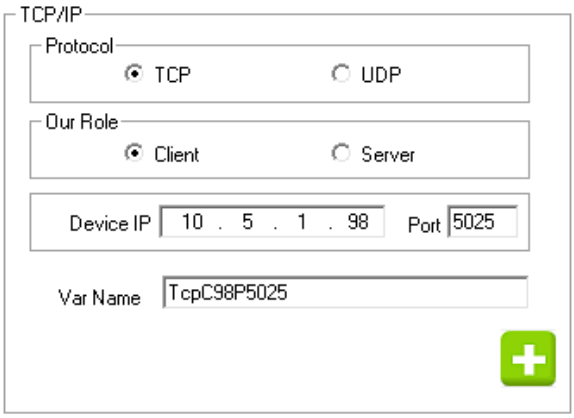
If you connect a PC through HyperTerminal with these settings, you should see activity during power-up. When the iNspect application loads, the baud rate is 38400.

TCP/IP Stream

The TCP/IP Stream is used for general communication with devices, such as output messaging or data. The TCP/IP Stream setup can be used with a PLC, however there are customized setups available for different PLC types and protocols.

Configure the BOA Spot

1. Load or create a simple Solution with one or two measurements.
2. In the Navigation Bar, click Setup Connections.
3. Use the drop list to select “TCP/IP Stream”. The center of the Setup panel changes, to configure the BOA Spot for TCP/IP communication.
4. Select TCP or UDP and select the BOA Spot to be a Client or Server. If the BOA Spot is a client, enter the **Server’s** IP Address and Port number (10.5.1.98 and port 5025 shown).



You can use the default variable name or change it. This variable is used in your script statements, when reading or writing to the server or client device.

5. Click the Add button. Your definition appears in the “Destination” drop list.

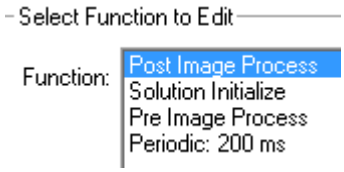
Note: The default on port 5024 is already defined. You can add more destinations.

To test your setup:

Requires a PC running HyperTerminal, Telnet or TeraTerm.

1. Create or load a simple Solution and configure the BOA Spot TCP/IP stream.
2. Click on the “Edit Scripts” button in the Navigation bar.
3. Click on the “Post Processing Function” and open the Edit window. Enter the following statement:

   WriteString(TcpP5025, “\n\rHello Spot”)



4. Click OK. Save, reload and Run the Solution. See “Important” on page 6.
5. Open HyperTerminal, Telnet or TeraTerm on a PC, and select TCP or Ethernet communication. Enter the BOA Spot IP Address and port number (5025 is shown above).
6. When you connect to the BOA Spot, you should see the message “Hello Spot” repeating on your monitor.

You can create more complex strings using the Free Edit window and the String Formatting window. Use the following functions to read or write to the TCP/IP network:

ReadByte, ReadString, WriteBytes, WriteString, WriteFormatString

Other useful TCP/IP functions available:

Disconnect, IsConnected, SendEmail, SetEmailInfo

Refer to “Comm / TCP/IP Functions” in the *BOA Spot Scripting User Manual* or in the BOA Spot on-line Help for full syntax, explanations, and some examples.

Note: The Windows Firewall should be **turned off** or customized on systems that communicate over the network.

TCP/IP Example

In this example, Port 5025 is used for communication. The BOA Spot-EL uses a Match Tool to find a part, and sends the image pixel X,Y coordinates [EL only] and Angle to a PC, robot arm, or “pick and place” system. Note: The robot arm or pick and place setup should have routines for calibrating to the corners of the image window.

In the **Post Process** function:

```
WriteFormatString( TcpP5025 , "\n\rX=[PX%3.2f], Y=[PY%3.2f], Theta=[MR%3.1f],
    Result  = [Result.0%d]\n\r" )      // X, Y, Angle & Result   Note: Not on the SL
```

The BOA Spot receives commands to trigger and change Solutions. The first character in the string is Trigger. The second character is a Solution number. The current Solution is 0. The string “T2” changes to Solution 2 and triggers the camera. The Solution test statement is different in each Solution. This test is: “not solution 0” (the current solution) and “a number less than 9” (valid solution numbers 0 through 8).

In the **Periodic: 200 ms** function:

```
ReadBuffer = ReadString( TcpP5025 , 13 )  // 13 is the end character
if(ReadBuffer != "")                      // if NOT an empty string
   CommandString = ReadBuffer
   Counter = Counter + 1                  // examine one character at a time
   CommandCharacter = Substring(CommandString, 1, 1)     // trigger command
   ProgramNumber = Substring(CommandString, 2, 1)        // Solution number
   CurrentSolution = GetSolutionID()
   if(INT(ProgramNumber) != CurrentSolution)             // test Solution number
      ChangeSolutionID(ProgramNumber)
    endif
   if(CommandCharacter = "T")
      trigger()
   endif
endif
```

When a Solution file is loaded, all variables should be cleared or initialized.

In the **Solution Initialize** function:

ReadBuffer = “”
CommandString = 0
Counter = 0
CommandCharacter = 0

Save, reload and run the Solution. See “Important” on page 6.

Troubleshooting TCP/IP Problems

Network Ports: BOA and iNspect Express require access to use network ports 80 and the range 5005 through 5025 plus any ports you configure for TCP communications.

Problem: No communication. Messages from BOA Spot do not appear in HyperTerminal or Telnet.

Reason: TCP/IP problems are most often caused by incompatible settings between BOA and the connecting equipment.

Solution 1: Check the settings on your BOA Spot and on your other TCP/IP devices. The Client device needs the IP Address and Port number of the Server device.

Check that the devices are on the same network, and are accessible (no address conflicts, and no security conflicts). On the PC, open a Command Prompt window and type “ping” and the IP Address of the BOA Spot, for example: **ping 10.5.1.100**

The Command Prompt window should show a number of bytes sent and received from BOA Spot. If data is not received, verify the BOA Spot is booted and the Ethernet cable is attached at both ends.

Solution 2: Save and reload your solution file. See “Important” on page 6.

Problem: Cannot find HyperTerminal, Telnet or TeraTerm on my PC.

Reason: HyperTerminal is not included in Windows 7 or 10. TeraTerm is a third-party program not included in Windows. Telnet is not enabled by default.

Solution 1: To enable Telnet on Windows 7, 10:

1. Click the Start button and select “Control Panel”.
2. Select “Programs and Features”.
3. Select “Turn Windows features on or off”.
4. Scroll down to find “Telnet” or “Telnet Client” and check the box beside it.
5. Click “OK”. Use the “Run” field or “Command Prompt” window to start Telnet.

Solution 2: Download TeraTerm.

Configure the PLC for BOA Spot

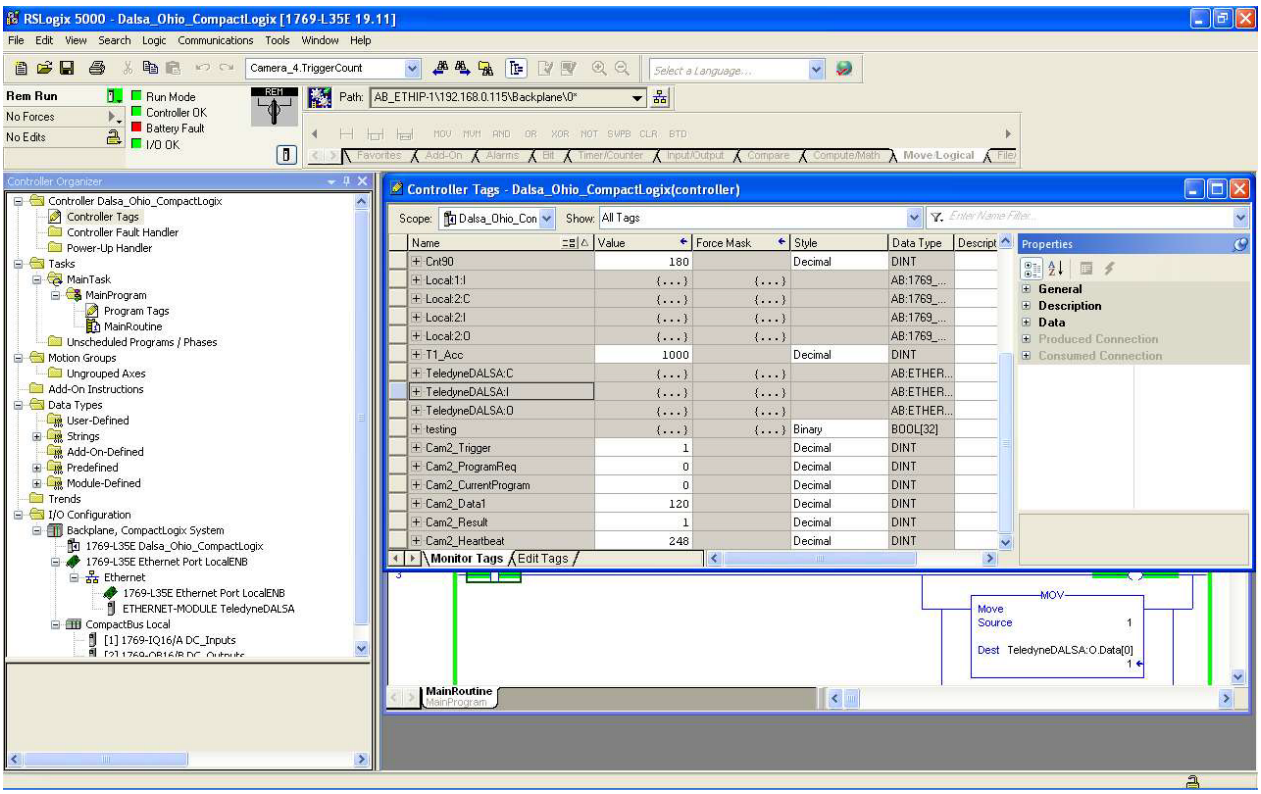
Rockwell Compact and Allen Bradley PLCs may use the Control Logix protocol for Tag based or “Explicit” Communication. You should create Tags in the Control Logix program before configuring BOA Spot, so there are tags to query and select.

Note: BOA Spot software has been verified with Rockwell Control Logix products using versions R18 through R30.

Note: The PLC should have a built-in Ethernet port for communication. Some add-on ports (like AB 1756-ENBT) and third-party ports (like Prosoft Tech. MVI56) do not use Control Logix protocol correctly, or relay (handle) Tag based queries correctly.

1. Open the PLC programming environment. This example shows the RSLogix5000.
2. Create Controller (global) Tags (a.k.a. User scoped) as needed for your application. These can be standard format or User Defined Tags.

Note: External access to Controller scoped tags is user selectable. If a tag’s External Access attribute is set to “none” then the tag cannot be accessed from outside the controller (PLC). Tags must have the “External Access” property set TRUE to be accessible to BOA Spot. BOA Spot can only access the Controller tags

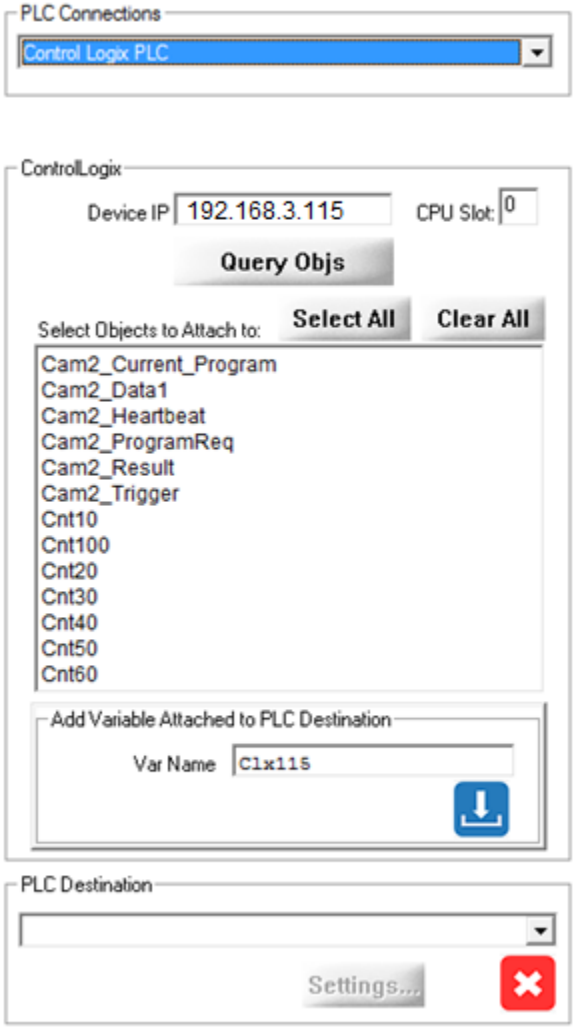


Tip: It is helpful to name the tags such that they are easy to recognize when imported to BOA Spot. In the image below, we are using Standard tags that all start with “Cam2”. In a factory environment there would be many tags for all the equipment attached to the PLC. Using a prefix like this makes it easier to see the ones you want.

3. Download the program to the PLC and run it. It should automatically compile before downloading.

Configure the BOA Spot

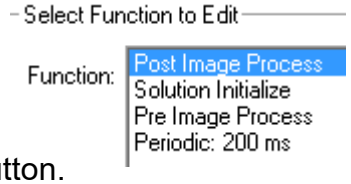
1. Load or create a simple Solution with one or two measurements.
2. In the Navigation bar, click on Setup Connections. 
3. Use the drop list to select Control Logix PLC. The center of the Setup panel changes, to configure BOA Spot for Control Logix communication.
4. Enter the IP Address of the **PLC**. Enter a CPU Slot number if your PLC has multiple CPUs. Use 0 if there are not multiple CPUs.
5. Click the “Query Objs” button and wait for the list to populate. The BOA Spot asks the PLC for Tags used in its program. This can take several minutes if the PLC program has many tags.
6. Scroll through the list and click on the variables your BOA Spot program needs. The list field allows multiple selections.
7. You can change the Variable name (in Var Name) or use the default. This name helps identify the PLC variables in your BOA solution program. This example shows the default variable “Clx115” (using the lowest byte of the IP Address).
8. Click “Add Control Logix PLC Destination” . Your PLC definition appears in the “PLC Destination” list box.



In an actual factory program there would be many more tags listed for all equipment the PLC communicates with. **Note:** The BOA Spot cannot access Program scope tags.

Note: If you created User-Defined data types, that name would appear in the list of Objects, not all the “member” objects below it. Selecting the User Data Type object in the menu above will include all the member objects.

Now add scripts to communicate with the PLC.



1. In the Navigation bar, click on “Edit Scripts”.
2. In the setup panel (Left) click on “Post Image Process”.
3. In the bottom panel below the image area, click the “Edit” button.
4. Add equations or statements that read or write to the Control Logix Tags. You can use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

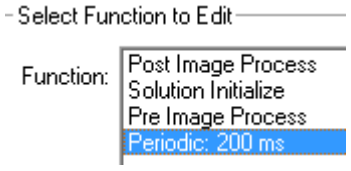
   In the **Post Image Process** function:

```
// send data to PLC
Clx115.Cam2_Data1 = IntenAvg   //Intensity Tool average intensity
Clx115.Cam2_Result = Result.0   // overall Pass Fail result
// inspection is done clear Busy flag
inspBusy = 0
```

5. Click the “Check Syntax” button to check for errors. 
6. Click the “Save” button, to save and close the Free Edit window. 

Tip: Start with writing one number to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

In most applications, you want the BOA Spot to change solutions and trigger on signals from the PLC.



1. In the Edit Scripts setup panel click on “Periodic: 200 ms”.
2. In the bottom panel below the image area, click the “Edit” button.
3. Add statements that will change the solution file, trigger the BOA Spot, clear and then rearm the trigger. (example shown on the next page)

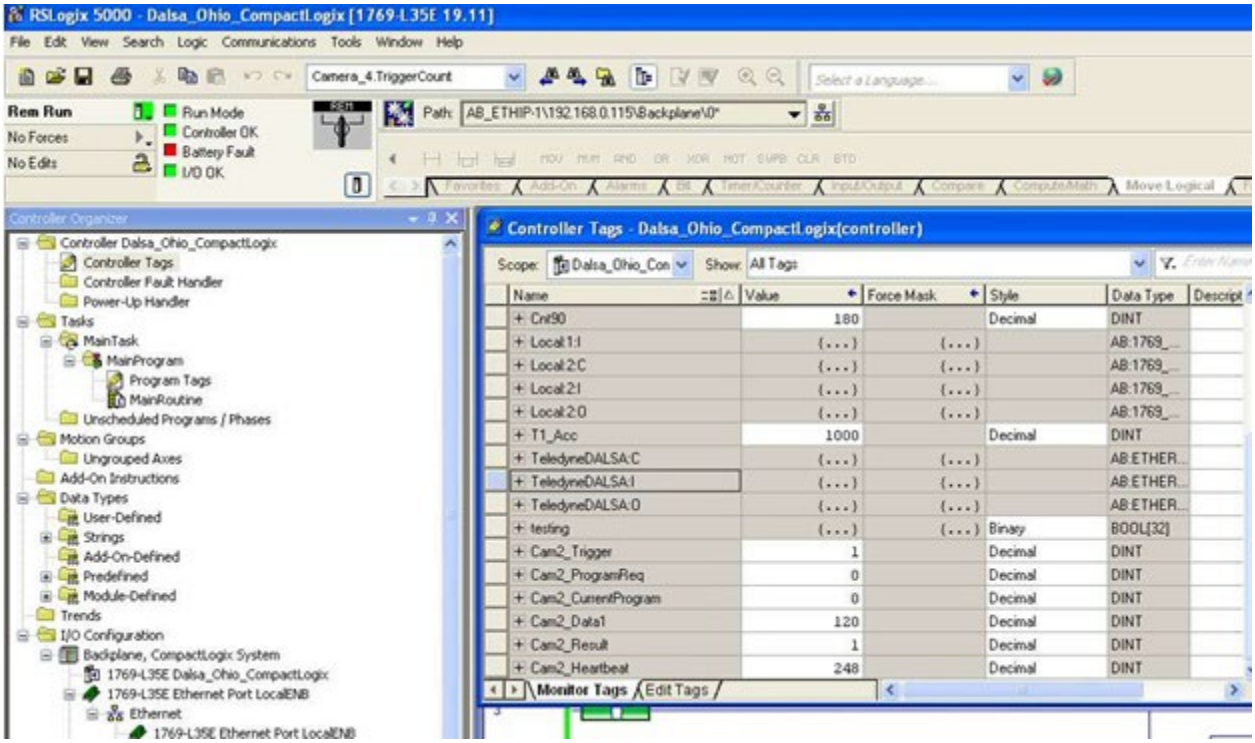
In the **Periodic: 200 ms** function:

```
// send current solution ID to PLC
MyRunningSolution = GetSolutionID()
Clx115.Cam2_CurrentProgram = MyRunningSolution
// get any solution change request from PLC
solReq = Clx115.Cam2_ProgramReq
if(solReq != MyRunningSolution    //If the request is a different Solution
    ChangeSolution(solReq)
endif
//
// get any trigger request from PLC
trigReq = Clx115.Cam2_Trigger
// only trigger on leading edge of register transition
if((trigReq = 1) AND (trigArmed = 1)
    trigArmed = 0
    inspBusy = 1
    trigger()
endif
// re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
//
// note PLC connection status
plcStat = IsConnected(Clx115)
//
// send heartbeat count
hb = hb + 1
if(hb > 999) hb = 1
Clx115.Cam2_Heartbeat = hb
```

4. Click the Check Syntax button. 
5. Click Save to save changes and close the Edit menu. 
6. Save, reload and Run the Solution. (See “Important” on page 6.)

Control Logix

With the Solution running on the BOA Spot, you can monitor communication between the PLC and BOA Spot in the RSLogix5000 program window.



Note: The Windows Firewall should be **turned off** or customized on systems that communicate over the network.

Note: Each tag read takes time. If a Control Logix Tag data is needed repeatedly in a script, it is advisable to read the tag once rather than several times in a script.

The script on the left contains 5 tag reads vs. 1 tag read in the script on the right. (Less communications traffic!)

WRONG METHOD
Reads the Tag 5 times.

```
// Fail inspection based on enabled tests
// in PLC configuration.
//
if(Clx115.Cam1_Test1Enabled = 1)
   if(Test1.Result != 1) FAIL = 1
endif
if(Clx115.Cam1_Test1Enabled = 1)
   if(Test2.Result != 1) FAIL = 1
endif
if(Clx115.Cam1_Test1Enabled = 1)
   if(Test3.Result != 1) FAIL = 1
endif
if(Clx115.Cam1_Test1Enabled = 1)
   if(Test4.Result != 1) FAIL = 1
endif
if(Clx115.Cam1_Test1Enabled = 1)
   if(Test5.Result != 1) FAIL = 1
endif
```

CORRECT METHOD
Reads the Tag only once.

```
// Fail inspection based on enabled tests
// in PLC configuration.
//
test1Enab = Clx115.Cam1_Test1Enabled
if(test1Enab = 1)
   if(Test1.Result != 1) FAIL = 1
endif
if(test1Enab = 1)
   if(Test2.Result != 1) FAIL = 1
endif
if(test1Enab = 1)
   if(Test3.Result != 1) FAIL = 1
endif
if(test1Enab = 1)
   if(Test4.Result != 1) FAIL = 1
endif
if(test1Enab = 1)
   if(Test5.Result != 1) FAIL = 1
endif
```

The point of this example is to illustrate the method of reading the tag once rather than several times in a script. (The script could certainly be optimized further!).

Note: We have observed that the Control Logix PLC takes 15 ms for each tag read during operation. Functions that read more than one tag must allow sufficient time for each tag to be read. For example, if a periodic function reads 5 tags, you must allow a minimum of 75 ms to read all 5 tags.

Ethernet IP PLC

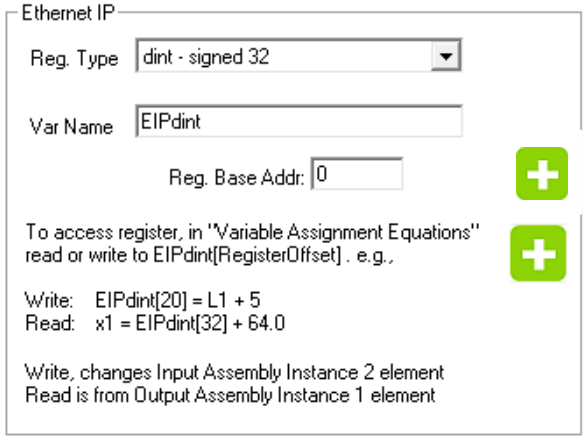
BOA Spot is compatible with PLCs that use Ethernet-IP communication. Rockwell and Allen Bradley PLCs use Ethernet-IP for “Implicit” communication (no Tags).

Note: For “Explicit” or Tag based communication, please see page 12. Do not use both!

Configure the BOA Spot

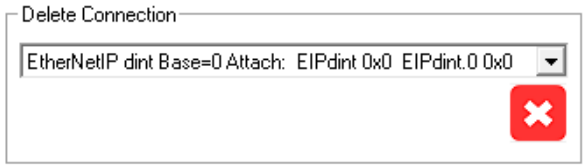
1. Load or create a simple Solution with one or two measurements.
2. In the Navigation bar, click on Setup Connections.

3. Use the drop list to select “Ethernet-IP PLC”. The center of the Setup panel changes to configure BOA Spot for Ethernet-IP communication.
4. Select a Register type (selected “dint” for this example), Use “real” for floating point.
5. Enter the base address of the registers assigned to the BOA Spot, in the PLC address space.



Tip: There are several different register types. Each performs a specific task and has a specific data type it will accept.

6. You can change the variable name or use the default.
7. Click the “Add” button. Your definition appears in the “Delete Connection” list box.



Notice the Read/Write examples just below the “Add” button. The examples are given using the default variable name “EIPdint”. These are the type of statements you would use in the Script Editor to read values from or write values to the PLC. The default name also indicates the register type. “dint” is for a signed 32-bit integer (double length integer). The brackets hold an index into the PLC register space.

EIPdint[20] = L1 + 5 – Translates to: Write the sum of L1 plus 5 to the PLC Input register at offset 20. **Note:** 20 is from iNspect Express perspective. If the PLC map is in bytes, each dint is 4 bytes so EIPdint[20] begins at N7:80 not N7:20.

X1 = EIPdint[32] + 64.0 – Translates to: Read the value in the PLC Output register at offset 32, add 64.0 to the value, and store it in variable X1.

Note: You can use Reg type “char array” to send a string to the PLC. You cannot use “char array” to read from the PLC. Use “sint - char” to read characters.

Now add scripts to communicate with the PLC.

1. In the Navigation bar, click on “Edit Scripts”.

2. In the setup panel (Left) click on “Post Image Process”.
3. In the bottom panel below the image area, click the “Edit” button.
4. Add equations or statements that read or write to the PLC. Use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

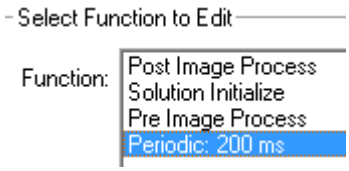
In the **Post Image Process** function:

```
//Send results to the PLC Input registers
EIPdint[0] = IntenAvg
EIPdint[1] = L
EIPdint[2] = Result.0
// inspection is done status
inspBusy = 0
```

5. Click the “Check Syntax” button to check for errors.
6. Click the “Save” button, to save and close the Free Edit window.

Tip: Start with writing just a few numbers to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

In most applications, you want the BOA Spot to switch Solution and trigger on a signal from the PLC.



7. In the Edit Scripts setup panel click on “Periodic: 200 ms”.
8. In the bottom panel below the image area, click the “Edit” button.
9. Add statements that tell the PLC which solution is running and read a solution change request from the PLC output registers. (example on the next page)
10. Add statements that trigger the BOA Spot, clear and then rearm the trigger. (example on the next page)

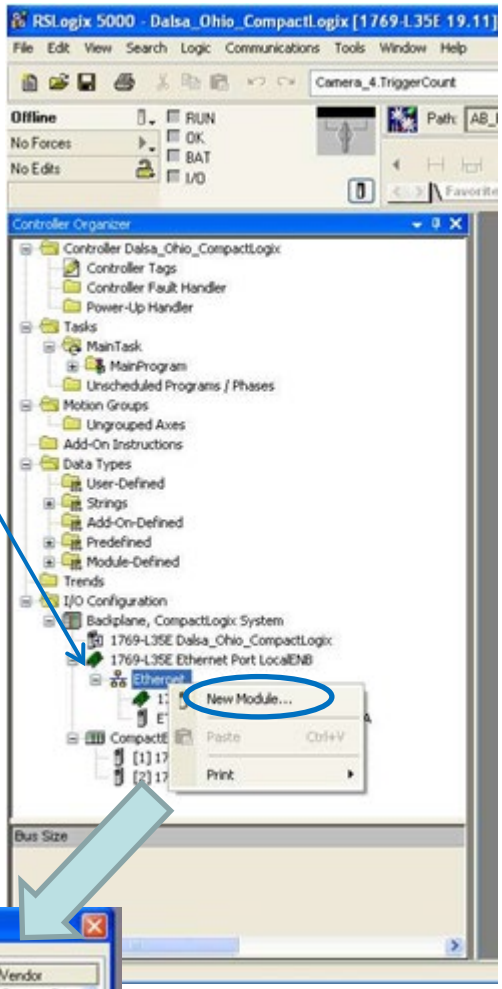
In the **Periodic: 200 ms** function:

```
//Send current solution to PLC
MyCurrentIDNum = GetSolutionID()
EIPdint[3] = MyCurrentIDNum
EIPdint[4] = Global.FrameCount
// get any solution change request from PLC Output registers
solReq = EIPdint[1]
if(solReq != MyCurrentIDNum)   //If the request is a different Solution
    ChangeSolution(solReq)     //Load the new Solution job file
endif
//
// get any trigger request from PLC output registers
trigReq = EIPdint[0]
// only trigger on leading edge of the register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0              // prevents multiple triggers
    inspBusy = 1               // the inspection is running
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
//note PLC connection status
plcStat = IsConnected(EIPdint)
//
//send heartbeat count
hb = hb + 1
if(hb > 999) hb = 1
EIPdint[5] = hb
```

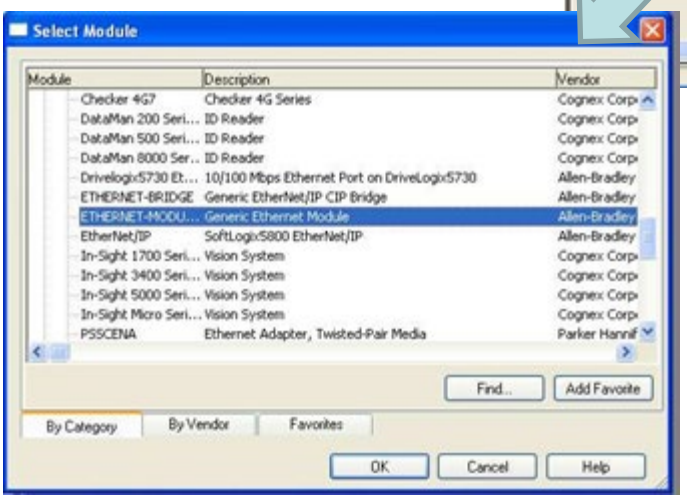
11. Check for syntax errors. 
12. Click “Save” to save changes and close the Edit window. 
13. Save, reload and then Run the Solution. (See “Important” on page 6.)

Configure RSLogix5000 for BOA Spot

1. Open the PLC programming environment. This example shows the RSLogix5000.
2. Right-click on the “Ethernet Connection” and select “New Module”.



3. In the pop-up menu, select “Generic Ethernet Module”.



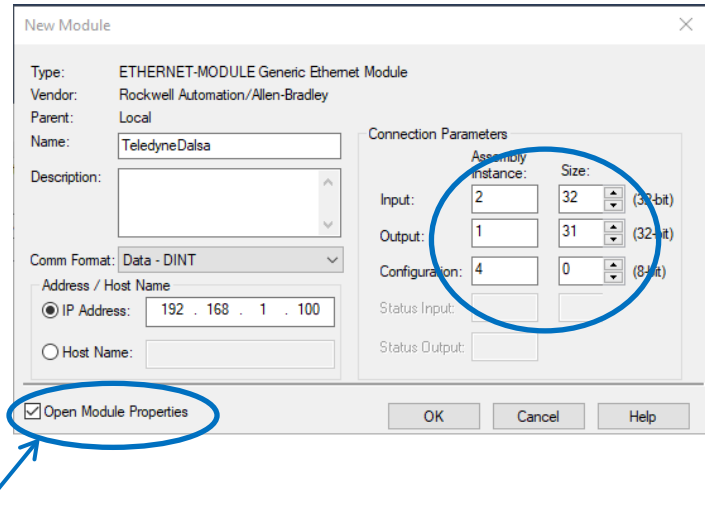
4. In the New Module menu, give the camera a name like TeledyneDalsa.
5. Enter the IP Address of the BOA Spot.
6. The Comm Format is used to specify the largest data size or type you want to send or receive. This is also defined in the EDS file, if you use one.
7. Enter the Input, Output and Configuration data for the “Assembly Instance” and “Size”. The values for Assembly Instance are always the same: Input = 2, Output = 1, Configuration = 4. The Size values set the maximum number of data values (or registers) you want to transfer.

   The Configuration Size is always 0 because iNspect software does not send Configuration information. The Assembly Instance is ignored.

   The Input Size is the number of values you want to write from ithe BOA Spot into the PLC.

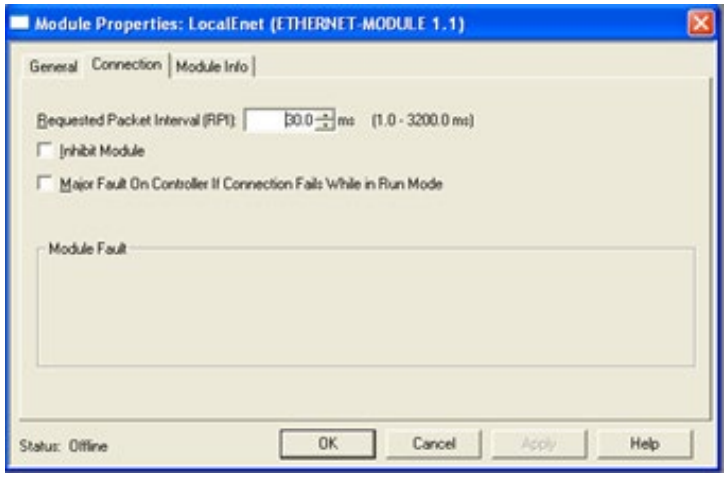
   The Output Size is the number of values you want the BOA Spot to read from the PLC.

   The Input and Output size is defined in the EDS file, if you use one.



8. Verify the “Open Module Properties” box is checked, then click “OK”

9. In the Module Properties menu, click on the Connection tab. Enter a “Requested Packet Interval (RPI)” not less than 30 ms (as shown). This is the time interval that the Logix processor will request new data from the BOA. Depending on network traffic, this number may need to be much larger (150 ms).



10. Click “OK”.

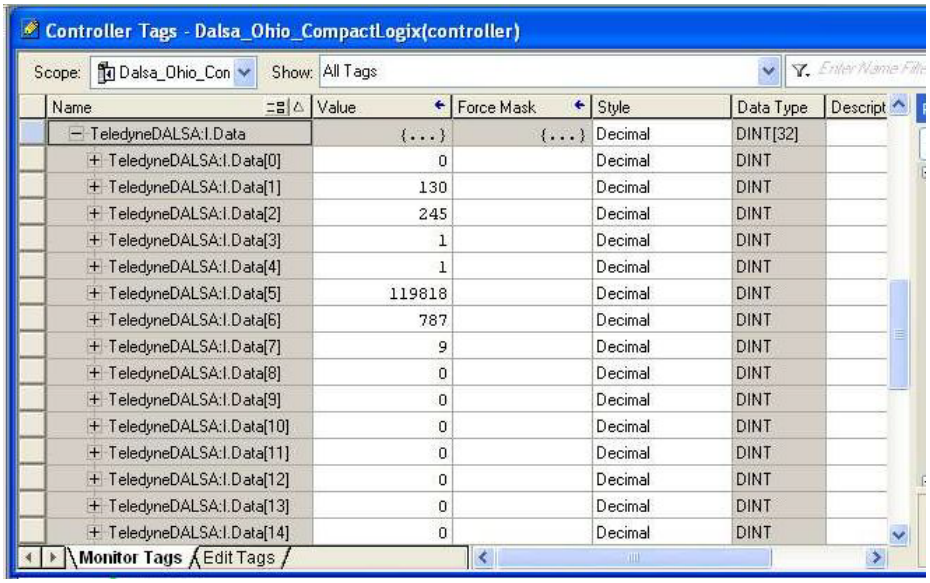
11. Compile the program and then download the result to the PLC.

If the BOA Spot is running, you should now be able to see values being updated, by monitoring the PLC Input registers.

Note: The BOA Spot write to EIP[0] maps to the PLC location I:DATA[1] (I for Input). This does not happen on a read. EIP[0] maps to PLC location O:DATA[0]. This is important to your PLC programming.

I:Data[1] = EIP[0] = IntenAvg, I:Data[2] = L, I:Data[3] = Result.0, I:DATA[4] = SOLUTION, I:Data[5] = Global.FrameCount, I:Data[6] = hb, I:Data [7] through [32] are not used in this example.

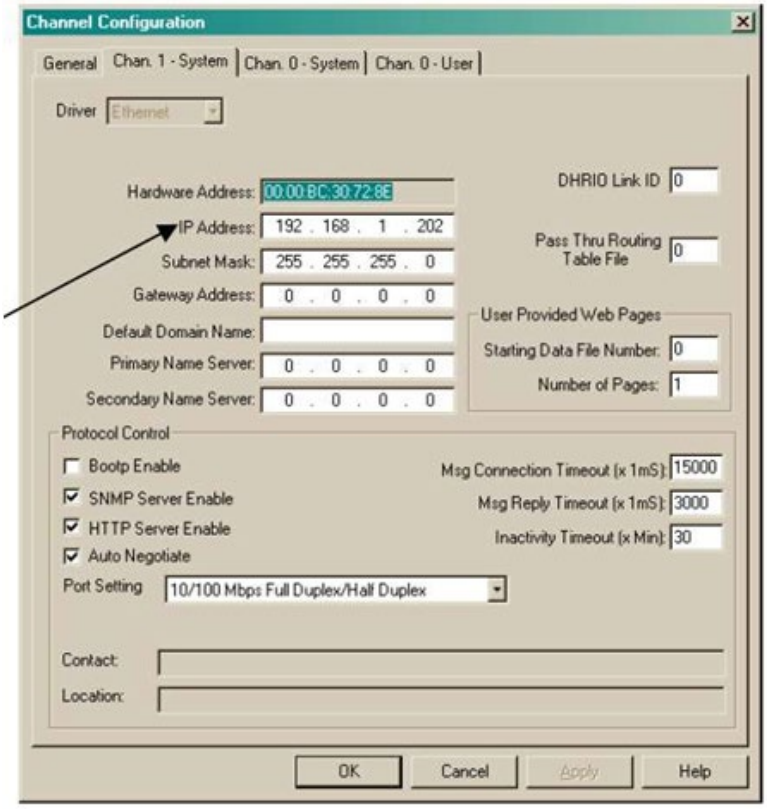
Note: This screen shows only the Input registers. The Output registers are at a different location. The Solution Change and Trigger request outputs do not appear in the Input registers.



Configure RSLogix500 for BOA Spot

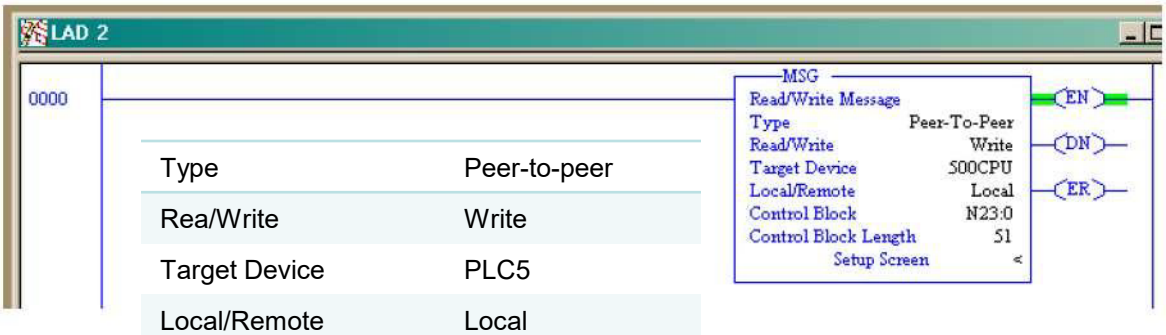
This example shows the RSLogix500 programming environment, for the SLC5-05 and MicroLogix 1100 PLCs.

1. Open the RSLogix500 programming environment.
2. In the Project Tree view, expand and navigate to: Project > Controller > Configuration.
3. Double click on Configuration to open the Channel Configuartion.
4. Click on the “Chan 1 System” tab.
3. Enter the IP Address and Submet Mask for the **PLC**. The PLC and BOA Spot must be in the same network “neighborhood”.



4. You may need to create a new Data File to get a message address.
   a. For the **SLC5-05:** Right-click on “Data Files” select New, select File: 11. Select Type: Message, Elements: 10 and click OK.
   b. For the **MicroLogix 1100:** Right-click on “Data Files”, select New, select File: 12. Select Type: Routing Information, and click OK. Right-click on Data Files a second time, select New, File: 13. Select Type: Routing Information, and click OK.

5. Set up your PLC ladder logic. A simple Write Message example is shown here. Create a new rung with a MSG. Specify read/write and control block address. For a Rockwell SLC5-05, change the “Target Device” to “PLC5” (not “500CPU” shown in this figure). Do not change the values of Control Block and Control Block Length.



Type	Peer-to-peer
Rea/Write	Write
Target Device	PLC5
Local/Remote	Local
Control Block	Use the default
Control Block Length	Use the default

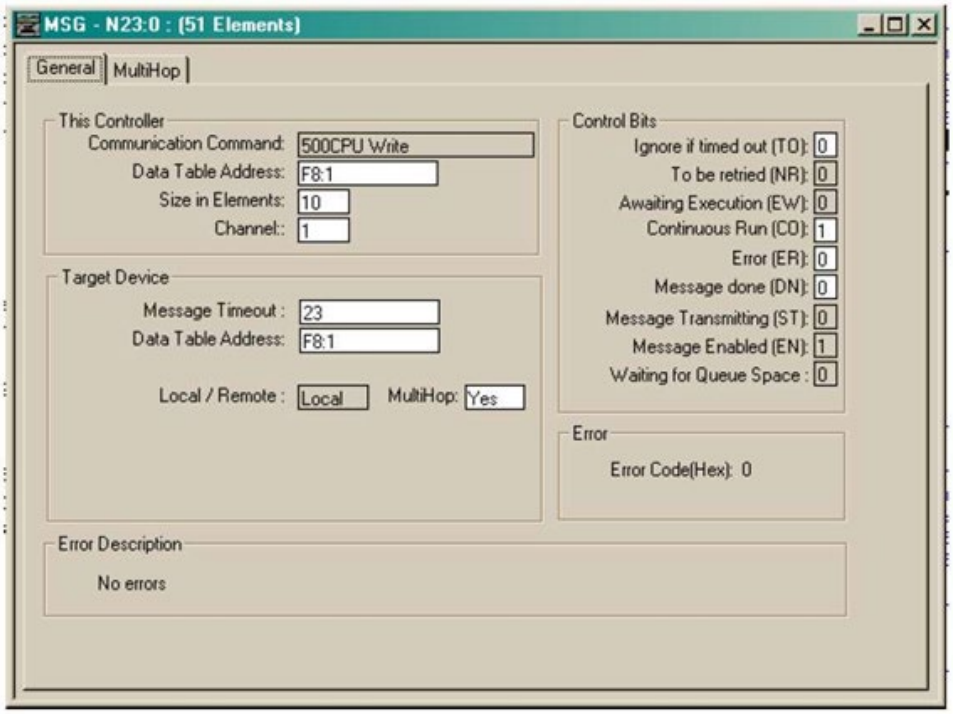
6. In the MSG menu, select channel 1 and **enable MultiHop**. For the Micrologix 1100, you will need to give some routing addresses from the routing files created (previous page). In “This Controller” specify the PLC source “table address” and “size in elements” for the data to write to the BOA Spot. In “MultiHop” select Yes.

The Data Table Address is predefined for the data types.

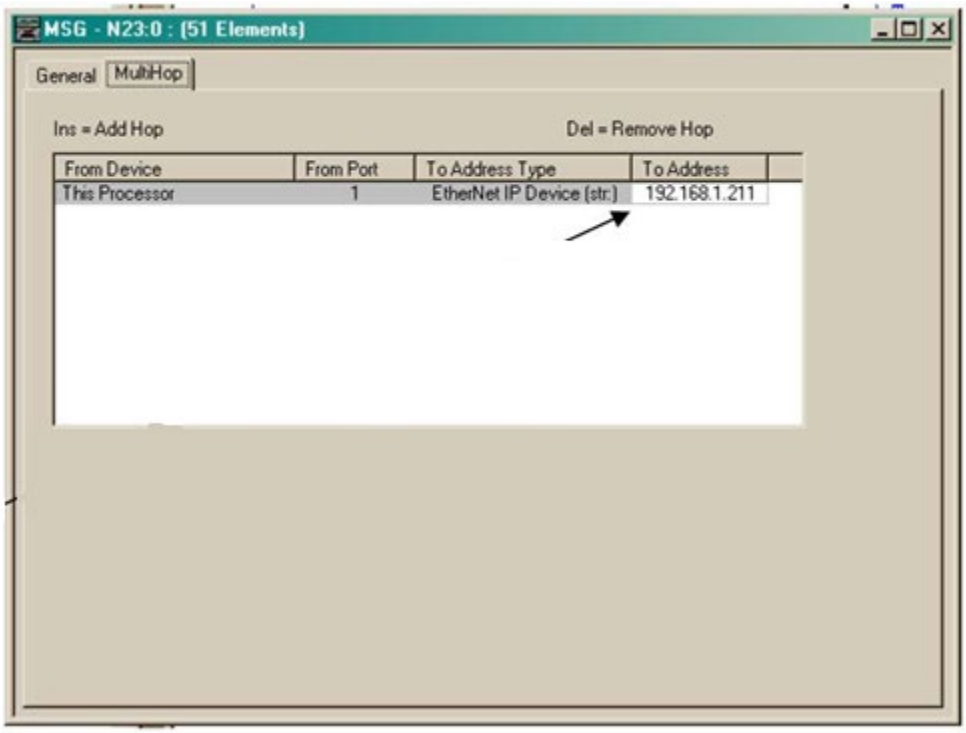
For int-signed 16 , Table address is N7.

For real-float 32, table address is F8.

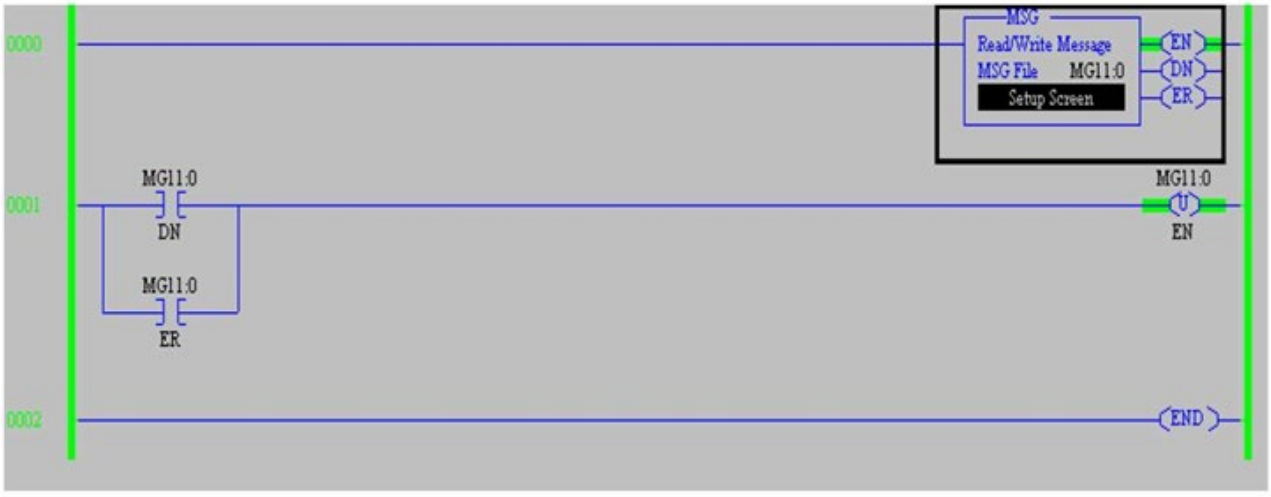
For char array-string, table address is ST9.



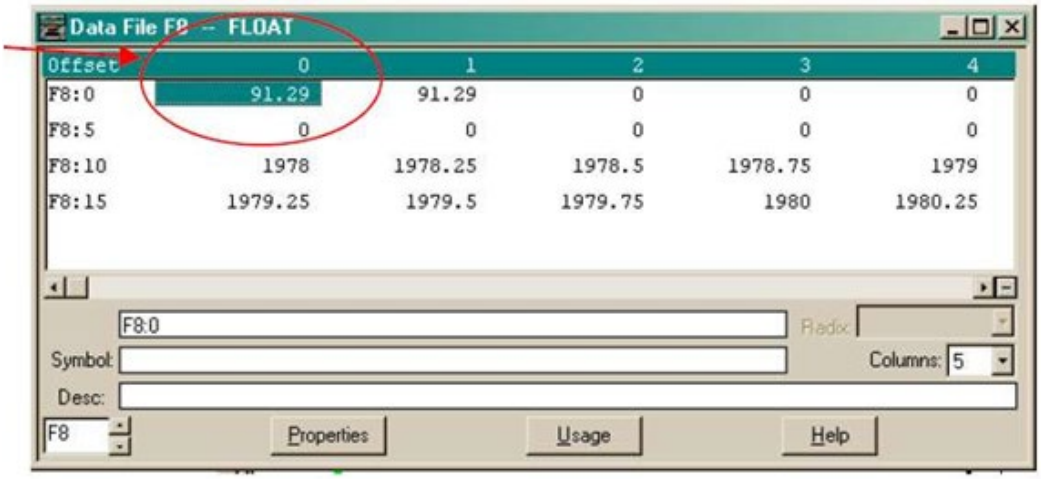
7. In the MultiHop tab, enter the IP Address of the BOA Spot.



8. Enter the run which will Unlatch EN if DN or ER is set.



Data will appear in the “Data File” associated with the Table address or data type used. For example, here is the Floating Point data; F8. F8:0 and F8:1 contain two floating point numbers received from the BOA Spot.



EIPreal[0] = A //angle measure from first Angle tool

EIPreal[1]=A1 // angle measure from second Angle tool

Data Table Assignments

EIPreal[0] is a Floating Point value, which is attached to F8:0

EIPint[0] is an Integer value, which is attached to N7:0

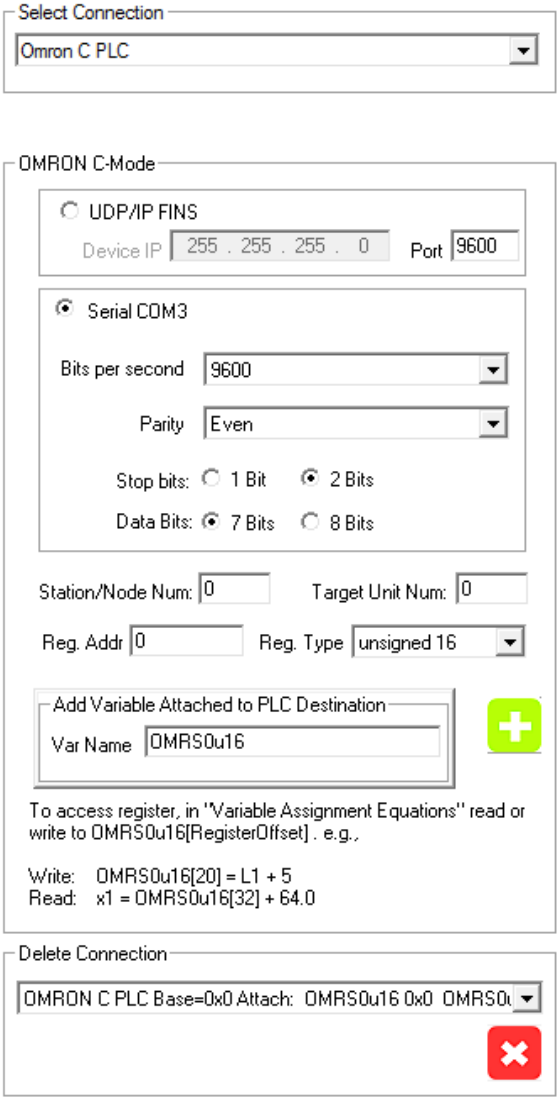
EIPsintArr[0] is a Character Array String value, which is attached to ST9:0

Note: The PLC map is displayed in bytes. It is important to remember that each Floating Point (real) value takes 4 bytes, each Integer value takes 4 bytes and each single character value takes one byte.

OMRON C PLC

BOA Spot is compatible with Omron PLC using a few different modes, depending on the Omron model and software capabilities: Omron C Serial, Omron C using UDC/IP FINS, and Ethernet-IP. For the Ethernet-IP setup (not an “Omron C”) see page 31.

Configure the BOA Spot



1. In the Navigation bar, click on Setup Connections.
2. Use the drop list to select “Omron PLC”.

The center of the Setup panel changes to configure the BOA Spot for Omron communication.

3. You can select a UDP/IP connection and enter the PLC’s IP Address. Or select a Serial connection and change the Serial port settings to match your PLC Host Link Interface. This example uses Serial.
4. Enter the Station or Node number and Target number (use 0 if you are not sure).
5. Select a Register type, Register address (index) Device (if applicable) and Variable Name.

Tip: There are several different register types. Each performs a specific task and has a specific data type it will accept.

6. Click “Add”. Your definition appears in the “Delete Connection” list box. Repeat steps 5 and 6 for different register types.

Notice the Read/Write examples just below the “Add” button. The examples are given using the default variable name “OMRS0u16”.
Reading and writing to an index of this variable causes a read or write to the same index location in the Omron C’s “D” memory area.

OMRS0u16[20] = L1 + 5 – Translates to: Write the sum of L1 plus 5 to the OMRON C D memory at offset 20.

X1 = OMRS0u16[32] + 64.0 – Translates to: Read the value in the OMRON C D memory at offset 32, add 64.0 to the value, and store it in variable X1.

Now add scripts to communicate with the PLC.

1. In the Navigation bar, click on “Edit Scripts”.
2. In the setup panel (Left) click on “Post Image Process”.
3. In the bottom panel below the image area, click the “Edit” button.
4. Add equations or statements that read or write to the PLC. Use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

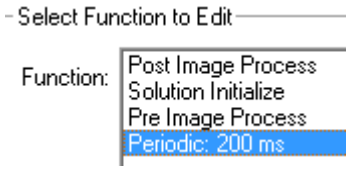
In the **Post Image Process** function:

```
//Send results to the PLC Input registers
OMRS0u16[0] = IntenAvg
OMRS0u16[2] = Result.0
```

5. Click the “Check Syntax” button to check for errors. 
6. Click the “Save” button, to save and close the Free Edit window. 

Tip: Start with writing just a few numbers to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

In most applications, you want the BOA Spot to switch Solution and trigger on a signal from the PLC (if this capability is available).



7. In the Edit Scripts setup panel click on “Periodic: 200 ms”.
8. In the bottom panel below the image area, click the “Edit” button.
9. Add statements that tell the PLC which solution is running and read a solution change request from the PLC output registers. (example on the next page)
10. Add statements that trigger the BOA Spot, clear and then rearm the trigger. (example on the next page)

Note: The Input Registers and Output Registers are at the same space in this method.

In the **Periodic: 200 ms** function:

```
//Send current solution to PLC
MyRunningID = GetSolutionID()
OMRS0u16[4] = MyRunningID
OMRS0u16[6] = Global.FrameCount
// get any solution change request from PLC
solReq = OMRS0u16[8]
if(solReq != MyRunningID)        //If the request is a different Solution
    ChangeSolution(solReq)       //Load the new Solution
endif
// get any trigger request from PLC
trigReq = OMRS0u16[10]
// only trigger on leading edge of register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0                // to prevent multiple triggers
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
```

11. Check for syntax errors. 
12. Click “Save” to save changes and close the Edit window. 
13. Save, reload and then Run the Solution. (See “Important” on page 6.)

Configure the PLC for BOA Spot

1. On the Omron C program development station, add a “Host Link Interface” to communicate over RS232.
2. Make sure the Serial port settings match those used to set up the BOA Spot.

For a UDP/IP or FINS connection the PLC setup would be more complex. Refer to the Omron documentation.

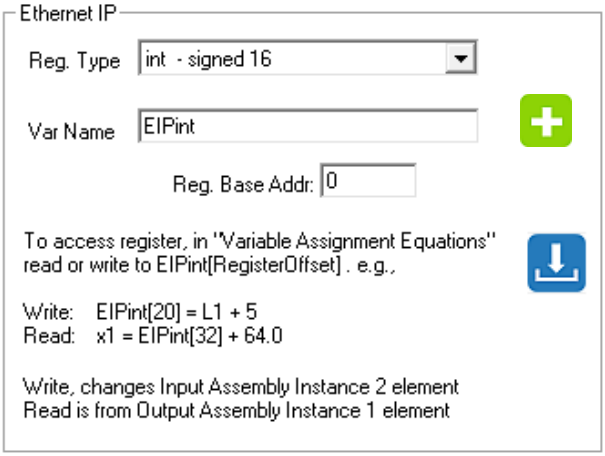
OMRON Ethernet-IP

Some OMRON PLCs use Ethernet-IP communication. This example uses the NJ301.

Configure the BOA Spot

1. Load or create a simple Solution with one or two measurements.
2. In the Navigation bar, click on Setup Connections. 
3. Use the drop list to select “Ethernet-IP PLC”. The center of the Setup panel changes to configure BOA Spot for Ethernet-IP communication.

4. Select a Register type (selected “dint” for this example) and enter the base address of the registers assigned to the BOA Spot, in the PLC address space.



Tip: There are several different register types. Each has a specific data type it will accept. Create a variable and destination for each register type you will use. This example uses 3 variable names.

5. Without changing the default variable name, click the “Add” button. The new variable destination appears in the Destination list.

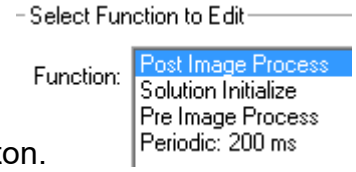
6. In the “Var Name” field, enter “EIPint_Read”. Click “Add” button again.
7. In the “Var Name” field, enter “EIPint_Write”. Click “Add” button again.

Notice the Read/Write examples just below the “Add” button. The examples are given using the default variable name “EIPint”. These are the type of statements you would use in the Script Editor to read from or write to the PLC. The default name also indicates the register type. “int” is for a signed 16-bit integer.

EIPdint[20] = L1 + 5 – Translates to: Write the sum of L1 plus 5 to the PLC register at offset 20 (from the base address).

X1 = EIPdint[32] + 64.0 – Translates to: Read the value in the PLC register at offset 32 (from the base address), add 64.0 to the value, and store it in variable X1.

Now add scripts to communicate with the PLC.



1.In the Navigation bar, click on “Edit Scripts”.

2.In the setup panel (Left) click on “Post Image Process”.

3.In the bottom panel below the image area, click the “Edit” button.

4.Add equations or statements that read or write to the PLC. Use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

In the **Post Image Process** function:

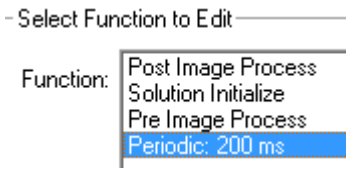
```
//send results to the PLC Input registers
EIP_Write[0] = IntenAvg
EIP_Write[1] = Global.PassCount
// send the overall result
EIP_Write[2] = Result.0
```

5. Click the “Check Syntax” button to check for errors. 

6. Click the “Save” button, to save and close the Free Edit window. 

Tip: Start with writing just a few numbers to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

In most applications, you want the BOA Spot to switch Solution and trigger on a signal from the PLC.



7. In the Edit Scripts setup panel click on “Periodic: 200 ms”.

8. In the bottom panel below the image area, click the “Edit” button.

9. Add statements that tell the PLC which solution is running and read a solution change request from the PLC output registers. (example on the next page)

10. Add statements that trigger the BOA Spot, clear and then rearm the trigger (example on the next page).

In the **Periodic: 200 ms** function:

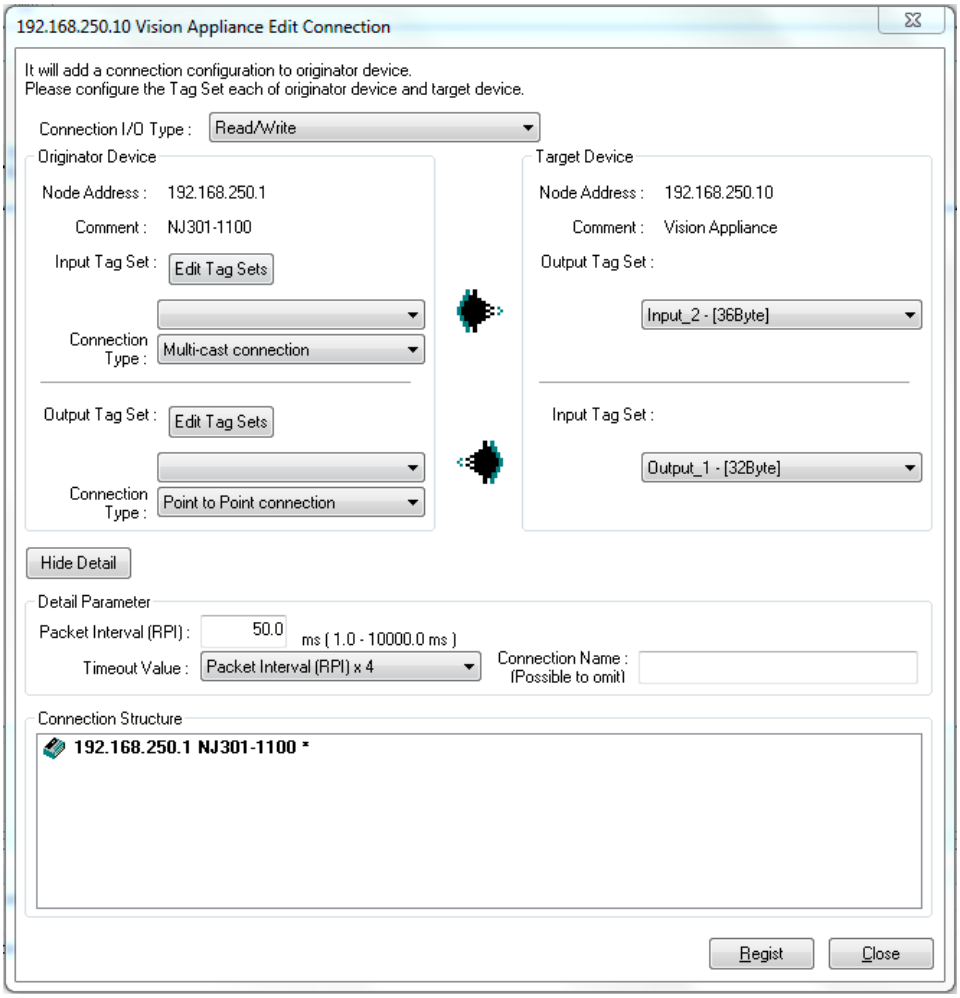
```
//Send current solution and total image count to PLC
MySolutionNow = GetSolutionID()
EIP_Write[3] = MySolutionNow
EIP_Write[4] = Global.FrameCount
// get any solution change request from PLC
solReq = EIP_Read[1]
if(solReq != MySolutionNow)      //If the request is a different Solution
    ChangeSolution(solReq)       //Load the new Solution
endif
//
// get any trigger request from PLC
trigReq = EIP_Read[0]
// only trigger on leading edge of register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0                // to prevent multiple triggers
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
```

11. Check for syntax errors. 
12. Click “Save” to save changes and close the Edit window. 
13. Save, reload and then Run the Solution. (See “Important” on page 6.)

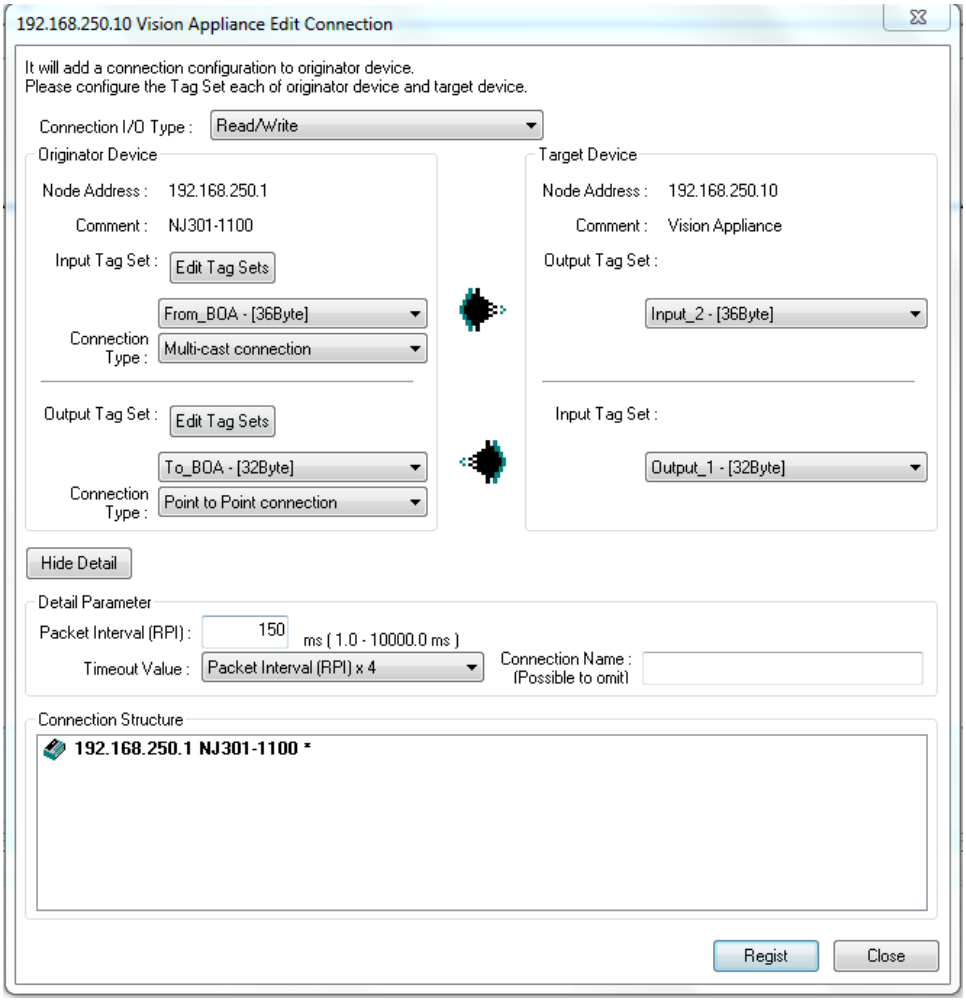
Configure the PLC for BOA Spot

The OMRON EIP requires an EDS file to define the BOA Spot resource to the PLC program. This EDS file is included in the BOA Spot software download file at \PLC Support Files\Omron.

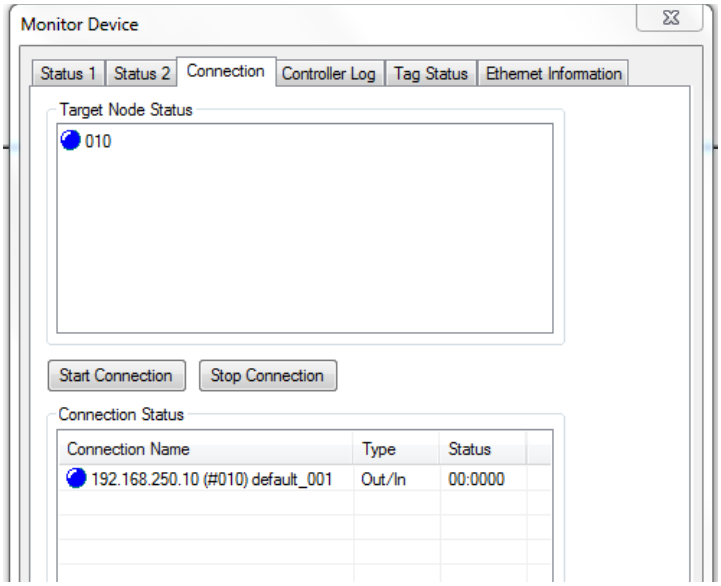
1. Open the OMRON Network Configurator. This example uses the Sysmac Studio.
2. Connect the PLC, and verify it is Online. This example used the NJ301.
3. If you already have a Network Configuration (nvf) upload it (Ctrl+U).
4. Add the “DalsaVisionV3_0814_01.eds” file to your system.
5. Insert the “BOA” into the network. You can drag and drop from “Teledyne DALSA”.
6. Drag the BOA icon onto the NJ Icon in the Network Configurator. The following screen should appear. The BOA Spot has the address 192.168.250.10.



7. Beside “Input Tag Set” click the “Edit Tag Sets” button.
8. Create the tag name “From_BOA” and make sure the input byte size is 36 bytes. This matches the value in the EDS file.
9. Beside “Output Tag Set” click the “Edit Tag Sets” button.
10. Create the tab name “To_BOA” and make sure the output byte size is 32 bytes, to match the value in the EDS file.
11. Insert the “BOA” into the network. You can drag and drop from “Teledyne DALSA”.
12. Change the “Packet Interval (RPI)” to 150.
13. Register and close.



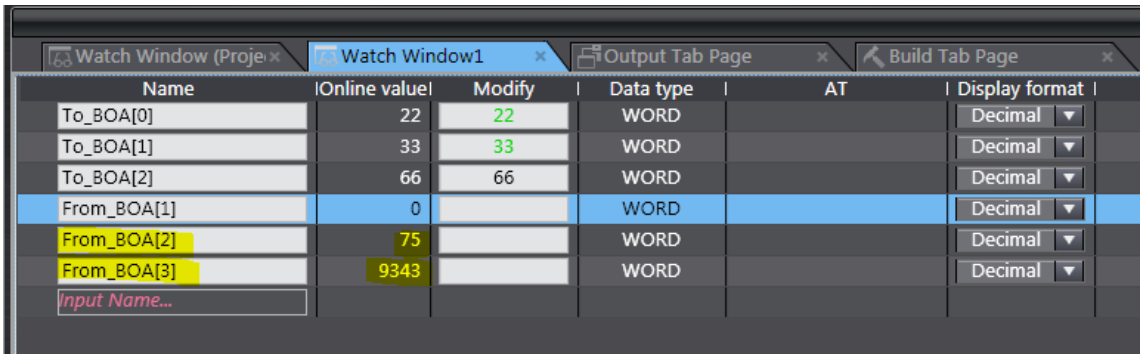
14. In the Network Configurator, right-click on the NJ301 and select “Monitor”. You should see a blue indicator next to the IP Address of the BOA Spot, as shown.



15. Create Global Variable in Sysmac Studio, that match the Tag names you just created in the Network Configurator, in this example “To_BOA” and “From_BOA”.

16. Download the program to the PLC. It should automatically compile.

17. You can use the Watch Window to view data being transferred between the PLC and the BOA Spot.



Note: The BOA Spot write to EIPint[0] or to EIP_Write[0] maps to the PLC location “From_BOA[2]”. This is important to your PLC programming.

Note: The Windows Firewall should be **turned off** or customized.

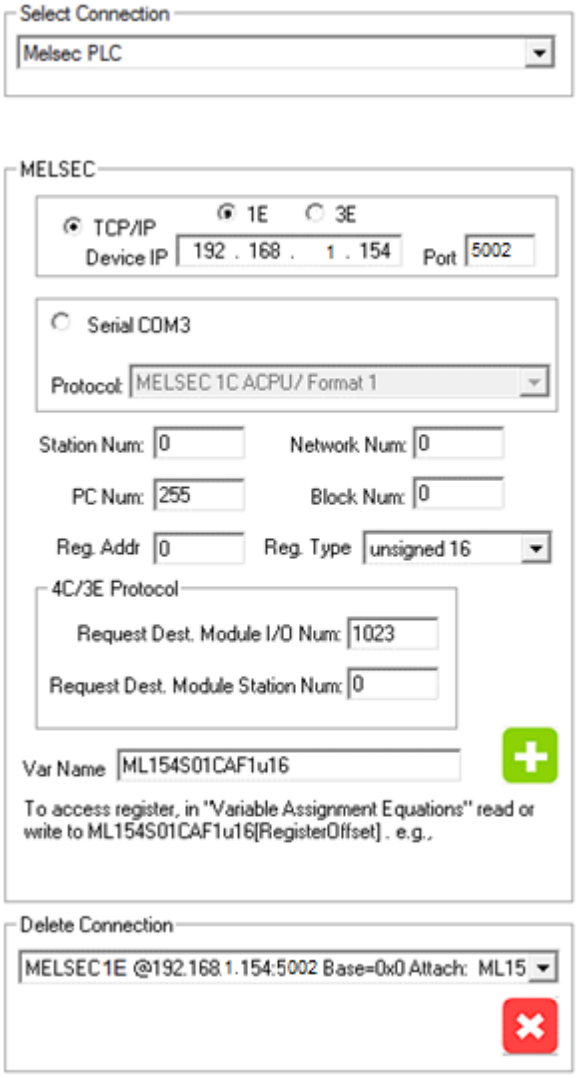
MELSEC PLC

BOA Spot is compatible with PLCs using Melsec communication over Serial or EIP. This example uses the Mitsubishi Series Q PLC - model QJ1E71 with EIP.

Configure the BOA Spot

1. Load or create a simple Solution with one or two tools.
2. In the Navigation bar, click on Setup Connections. 
3. Use the drop list to select “Melsec PLC”.

The center of the Setup panel changes to configure BOA Spot for Melsec communication.



4. Select a TCP/IP connection, enter the **PLC**’s IP Address, (192.168.1.154) and Port number to match your PLC. Note: The QJ1E71 uses Port 5002. The FX3u uses Port 5001.
5. Select 1E or 3E to match your PLC. Note: The QJ7aE71 and FX3u both use 1E communication.
6. Enter the Station, Network, PC and Block numbers for your PLC.
7. Select a Register type and Register address (index).

Tip: There are several different register types. Each performs a specific task and has a specific data type it will accept.

8. Enter the Request information if you are using 4C/3E communication.
9. You can enter a variable name or use the defaults. The default variable names reflect the register type and data type.
10. Click “Add”. Your definition appears in the “Delete Connection” list box. Repeat steps 5, 7 and 8 to add other register types.

Tip: It may be easier to create all your variable types (Reg types) before leaving this panel, because the settings revert back to 3E and Serial when you exit this panel. If you return, make sure you set these options correctly.

Tip: The Input Registers and Output Registers are usually at the same space in a MELSEC system.

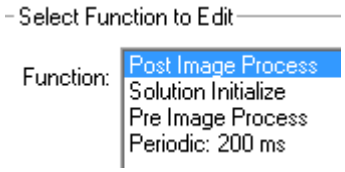
Notice the Read/Write examples just below the “Add” button. The examples are given using the default variable name “ML154S01CAF1u16”. “u16” for unsigned 16-bit data.

ML154S01CAF1u16[20] = L1 + 5 – Translates to: Write the sum of L1 plus 5 to the PLC D memory at offset 20.

X1 = ML154S01CAF1u16[32] + 64.0 – Translates to: Read the value in the D memory at offset 32, add 64.0, and store it in variable X1.

Now add scripts to communicate with the PLC.

1.In the Navigation bar, click on “Edit Scripts”. 



2. In the setup panel (Left) click on “Post Image Process”.
3. In the bottom panel below the image area, click the “Edit” button.
4. Add equations or statements that read or write to the PLC. Use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

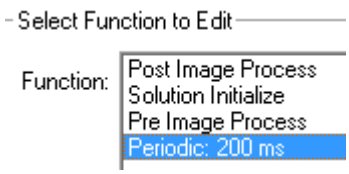
In the **Post Image Process** function:

```
//Send results to the PLC Input registers
ML154S01CAF1u16[0] = IntenAvg
ML154S01CAF1u16[1] = Result.0
inspBusy = 0 // this is an optional status flag
```

5. Click the “Check Syntax” button to check for errors. 
6. Click the “Save” button, to save and close the Free Edit window. 

Tip: Start with writing just a few numbers to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

In most applications, you want the BOA Spot to switch Solution and trigger on a signal from the PLC.



7. In the Edit Scripts setup panel click on “Periodic: 200 ms”.
8. In the bottom panel below the image area, click the “Edit” button.

9. Add statements that tell the PLC which solution is running and read a solution change request from the PLC output registers. (example on the next page)
10. Add statements that trigger the BOA Spot, clear and then rearm the trigger. (example on the next page)

Note: The “inspBusy” variable is not fully implemented in this example. You could send it to the PLC to tell it the Solution is running.

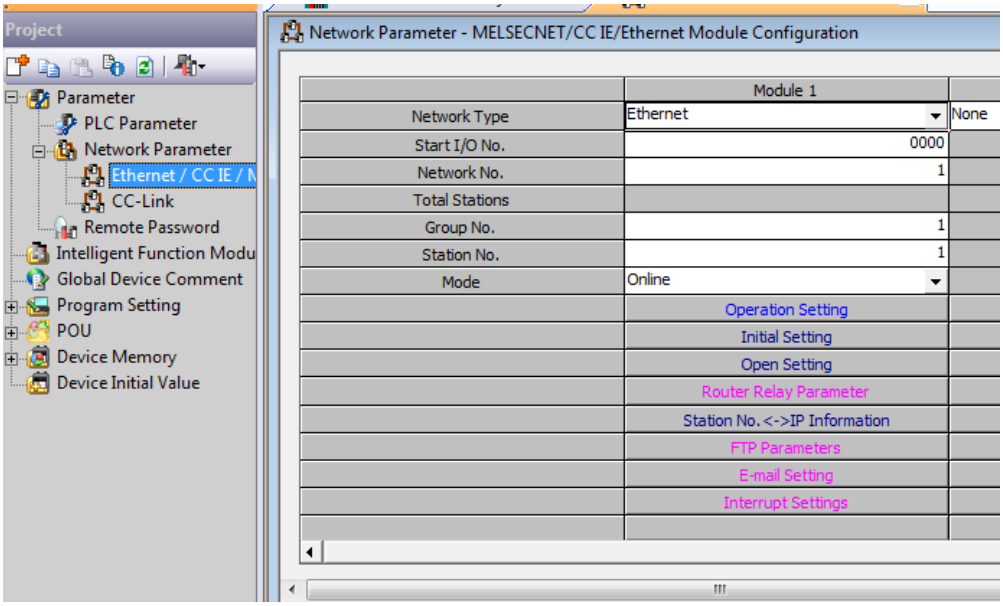
In the **Periodic: 200 ms** function:

```
//Send current solution and total image count to PLC
MyRunningSolution = GetSolutionID()
ML154S01CAF1u16[2] = MyRunningSolution
ML154S01CAF1u16[3] = Global.FrameCount
//
// get any solution change request from PLC
solReq = ML154S01CAF1u16[4]
if(solReq != MyRunningSoluiton)   //If the request is a different Solution
    ChangeSolution(solReq)        //Load the new Solution
endif
// get any trigger request from PLC
trigReq = ML154S01CAF1u16[5]
// only trigger on leading edge of register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0                 // to prevent multiple triggers
    inspBusy = 1                  // the inspection is running
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
```

11. Always check for syntax errors. 
12. Always click “Save” to save changes and close the Edit window. 
13. Save, reload and then Run the Solution. (See “Important” on page 6.)

Configure the PLC for BOA Spot

1. Open the MELSEC program development software. This example is using MELSOFT Series GX Works2.
2. In the Project Data list, expand the Network Parameter group, and click on “Ethernet/CC IE/MELSECNET”
3. Configure the Ethernet module settings as follows:
   Set Network Type to **Ethernet**
   Set Starting IO to **0000**
   Set Network No. to **1**
   Set Group No. to **1**
   Set Station No. to **1**



Note: The Windows Firewall should be **turned off** or customized on systems that communicate over the network.

4. Click on “Operation Setting”.

Set Communication Data Code to:
Binary Code

Set Initial Timing to:
Always wait for OPEN

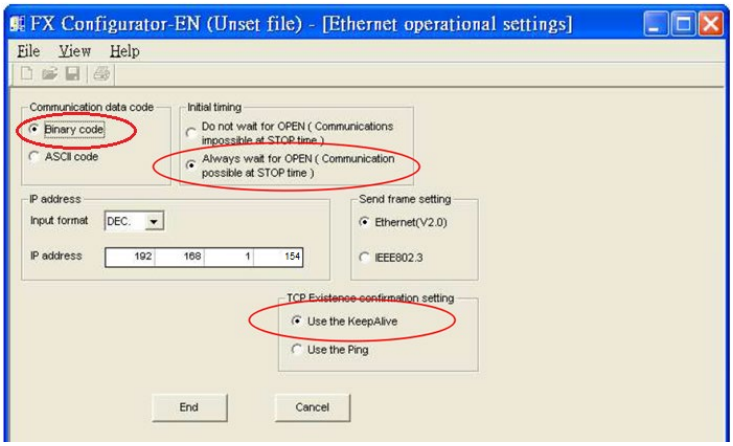
Set Input Format to: **DEC**.

The IP **Address** shown is for **this PLC**, the same value entered in the BOA Spot.

If available, check the box beside:
Enable Online Change

Send frame setting should be:
Ethernet(V2.0)

TCP Existence Confirmation Setting should be: **Use the KeepAlive**



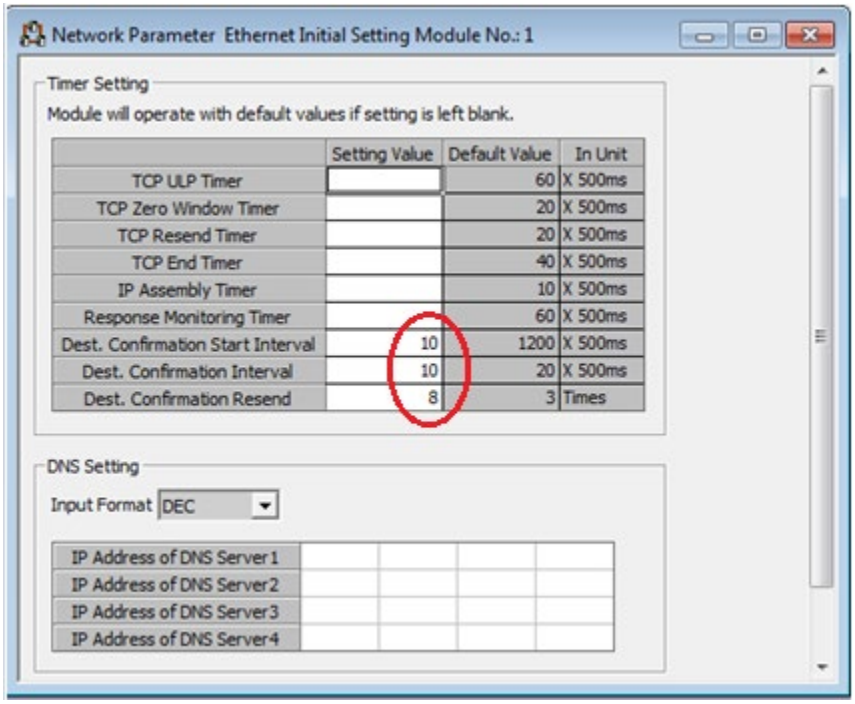
5. Click **End** to accept changes and return to the Module Configuration.

6. Click on “Initial Setting”.

Set “Dest. Confirmation Start Interval” to **10**

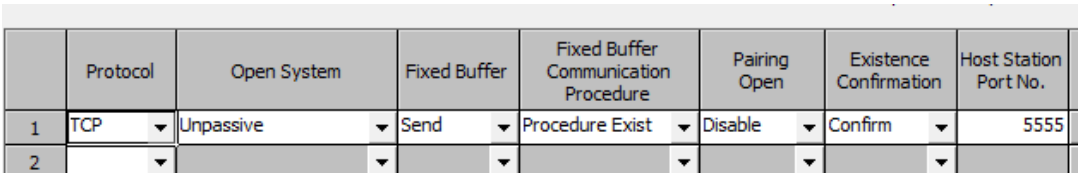
Set “Des. Confirmation Interval” to **10**

Set “Des. Configuration Resend” to **8**



7. Close this window and return to the Module Configuration.

8. Click On “Open Settings”
   Set Protocol to **TCP**
   Set Open System to **Unpassive**
   Set Fixed Buffer to **Send**
   Set Fixed Buffer Communication Procedure to **Procedure Exist**
   Set Paring Open to **Disable**
   Set Existence Confirmation to **Confirm**
   Set Host Station Port No. to 5002, to match the BOA Spot
9. Close this window to return to Module Settings.



	Protocol	Open System	Fixed Buffer	Fixed Buffer Communication Procedure	Pairing Open	Existence Confirmation	Host Station Port No.
1	TCP	Unpassive	Send	Procedure Exist	Disable	Confirm	5555
2							

10. Click “End” to accept changes and close the Module Configuration.
11. In the Menu bar, select **Online => Write to PLC** to write your Configuration to the PLC.
12. Check **Write** and check **Parameters**
13. Click on “Execute”.
14. Cycle power to the PLC to reset it.

Use the Batch Monitor (In GX Works 2) to view the D Memory area, which BOA Spot reads or writes to:

15. In the Menu bar, select **Online->Monitor->Device Buffer Memory Batch**
16. In “Device Name” enter **D0** to monitor the first D memory locations.

The script in BOA Spot writes a measured value to offset 0. This corresponds to D0. The pass count, total frame counts and solution number are written to offsets 1, 2, and 3. These correspond to locations D1, D2, and D3. The script reads from offsets 4 and 5, which correspond to D4 and D5.

Our register definition (in BOA Spot) was limited to unsigned 16-bit values. You can add other register types. Make sure the display format settings in the Batch Monitor match the register types you send.

Profinet PLC

BOA Spot is compatible with PLCs that use the Profinet UDP or Profinet RT protocol.

Note: Please refer to page 6, “BOA Spot Name in Profinet” for critical information on device names in Profinet. When more than one BOA Spot is connected to the PLC.

Note: Configure the BOA Spot, and run the Solution at least once, to initialize Profinet communication. If the BOA Spot Solution has not run, the BOA Spot will not appear in your Profinet programming environment.

Configure the BOA Spot

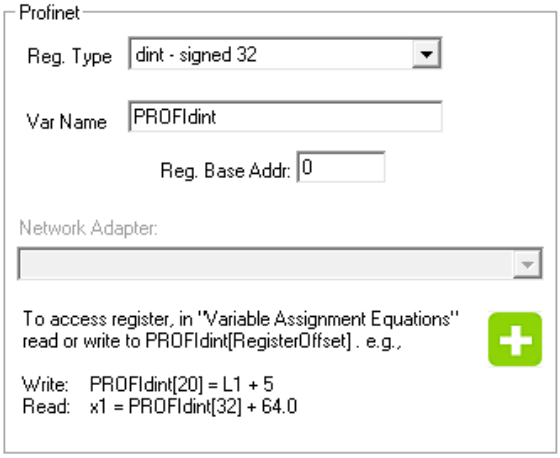
1. Load or create a simple Solution with one or two tools.
2. In the Navigation bar, click on Setup Connections.

3. Use the drop list to select “Profinet RT IO”.

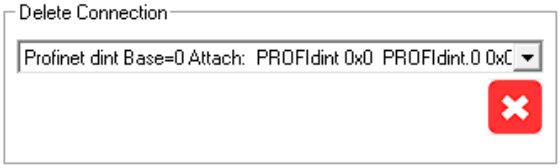
The center of the Setup panel changes to configure BOA Spot for Profinet communication.

4. Select the Register type and enter the base address of the registers assigned to the BOA Spot, in the PLC address space.

There are several different register types. Each performs a specific task and has a specific data type it will accept.



5. You can change the Variable name or use the default.
6. Click the “Add” button. Your definition appears in the “Delete Connection” list box.



7. Repeat steps 3, 4, & 5 for each different register type.

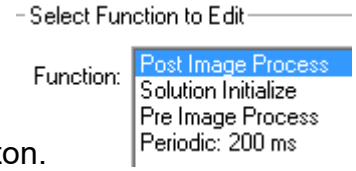
Notice the Read/Write examples just below the “Add” button. The examples are given using the default variable name “PROFIdint”. These are the type of statements you would use in the Script Editor to read values from or write values to the PLC. The default name also indicates the register type. “dint” is for a signed 32-bit integer (double int).

PROFIdint[20] = L1 + 5 – Translates to: Write the sum of L1 plus 5 to the PLC’s Holding register at index location 20.

X1 = PROFIdint[32] + 64.0 – Translates to: Read the value in Holding register index location 32, add 64.0 to the value, and store it in variable X1.

Now add scripts to communicate with the PLC.

1.In the Navigation bar, click on “Edit Scripts”.



2.In the setup panel (Left) click on “Post Image Process”.

3.In the bottom panel below the image area, click the “Edit” button.

4.Add equations or statements that read or write to the PLC. Use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

In the **Post Image Process** function:

```
//Send results to the PLC Input registers
PROFIdint[0] = IntenAvg
PROFIdint[1] = L
PROFIdint[2] = Result.0
```

5. Click the “Check Syntax” button to check for errors.

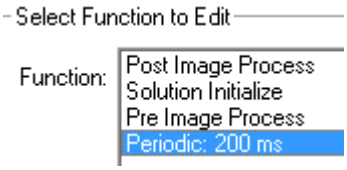
6. Click the “Save” button, to save and close the Free Edit window.

Tip: Start with writing just a few numbers to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

Note: When BOA Spot writes to the attached variable PROFIdint, it is updating the **input module** on the Profinet controller (I address range 256..509). PROFIdint is a 32-bit signed value; meaning that BOA Spot references are indexing a dint sized array.

PROFIdint[0] = x, writes the value which appears at location %ID256 on the controller, PROFIdint[1] = x, writes the value which appears at location %ID260 on the controller, which is the next double size index. Note: These numbers were with TIA-11.

In most applications, you want the BOA Spot to switch Solution and trigger on a signal from the PLC.



7. In the Edit Scripts setup panel click on “Periodic: 200 ms”.
8. In the bottom panel below the image area, click the “Edit” button.
9. Add statements that tell the PLC which solution is running and read a solution change request from the PLC output registers. (example on the next page)
10. Add statements that trigger the BOA Spot, clear and then rearm the trigger. (example on the next page)

In the **Periodic: 200 ms** function:

```
//Send current solution number to the PLC
MySolutionIs = GetSolutionID()
PROFIdint[3] = MySolutionIs
PROFIdint[4] = Global.FrameCount
// get any solution change request from the PLC
solReq = PROFIdint[1]
if(solReq != MySolutionIs)        //If the new request is a different Solution
    ChangeSolution(solReq)    //Load the new Solution
endif
// get any trigger request from the PLC
trigReq = PROFIdint[0]
// only trigger on leading edge of register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0                // to prevent multiple triggers
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
```

11. Check for syntax errors. 
12. Click “Save” to save changes and close the Edit window. 
13. Save, reload and then Run the Solution. (See “important” in page 6.)

Note: When BOA Spot reads the attached variable PROFIdint, it is reading the **output module** on the Profinet controller (Q address range 256..509).

profiCmd= PROFIdint[0], is reading from location %QD256 on the controller,
profiCmd= PROFIdint[1], is reading from location %QD260 on the controller,
…….
profiCmd= PROFIdint[62], is reading from location %QD504 on the controller,

Note: You can use Reg type “char array” to send a string to the PLC. You cannot use “char array” to read from the PLC. Use “sint - char” to read characters.

Configure the PLC for BOA Spot

Note: The Profinet factory development software did not function fully in Windows 10 until Siemens released a patch in May/June 2020. Windows 7 operated correctly. Make sure your software is up to date.

The Profinet programming station (PC) uses a GSD file to describe the BOA Spot resources. There are two GSD files included in the BOA Spot software download file. There is also an image file, to add a BOA Icon to your programming station. This icon appears when you add a BOA Spot device in your program. Copy the files to your PC.

The GSD file “GSDML-V2.2-Dalsa-IpdVisionBOA254-20140910.xml“ defines 256 bytes for each BOA. This means that a maximum of four BOA Spots can be used.

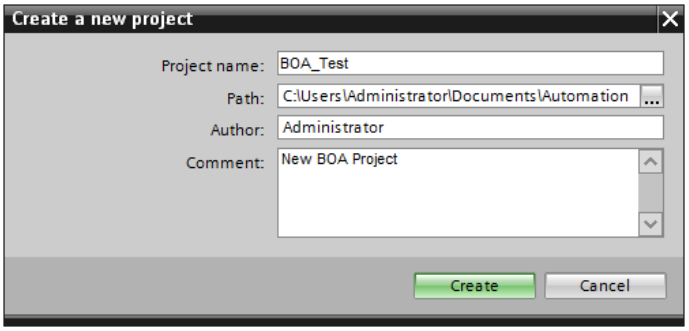
The GSD file “GSDML-V2.2-Dalsa-IpdVisionBOA64-20140909.xml” defines 64 bytes for each BOA Spot. In theory up to 15 BOA Spots could be defined, but this is not proven.

Note: Please refer to page 6, “BOA Spot Name in Profinet” for critical information on device names used in Profinet.

Note: You must first configure the BOA Spot, and run the Solution at least once, to request and initialize Profinet communication. If the BOA Spot Solution has not run, the BOA Spot will not appear in your Profinet programming environment.

1. Copy the GSD and BMP files to your PC that runs the development environment.
2. Open the Profinet program development software.

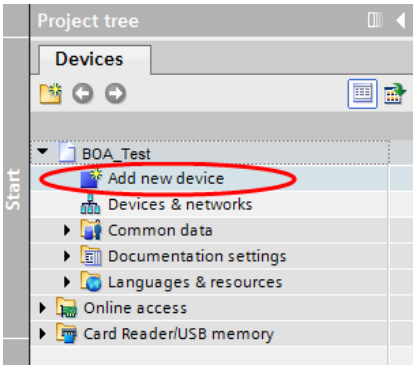
Note: You can add the BOA Spot to an existing project or create a new project for testing.



To Create a New Project

This example is using Siemens "Totally Integrated Automation Panel" TIAP-V11.

1. In the TIAP toolbar, select “Create Project”.
2. Give your project a name and click “Create”.
3. In the Devices tab, click “Add new device”.
4. Enter a “Device Name” (optional) and select the correct CPU for your PLC from the list of Controllers. Our example uses the name “lilwes” for a 1212C CPU.
5. Click “OK”.

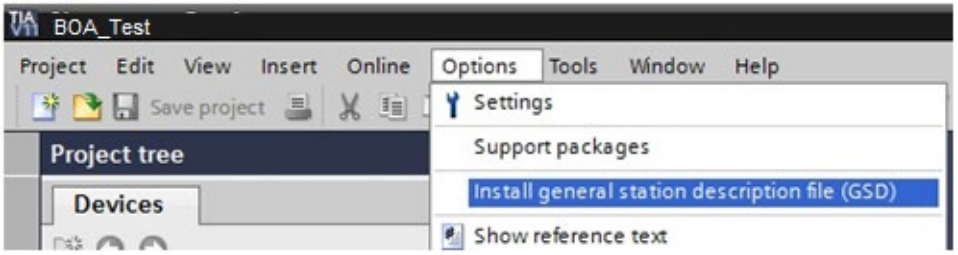


Add the BOA Spot to a new or existing TIAP Project

Instructions for the Simatic Manager or STEP7 environment begin on page 51.

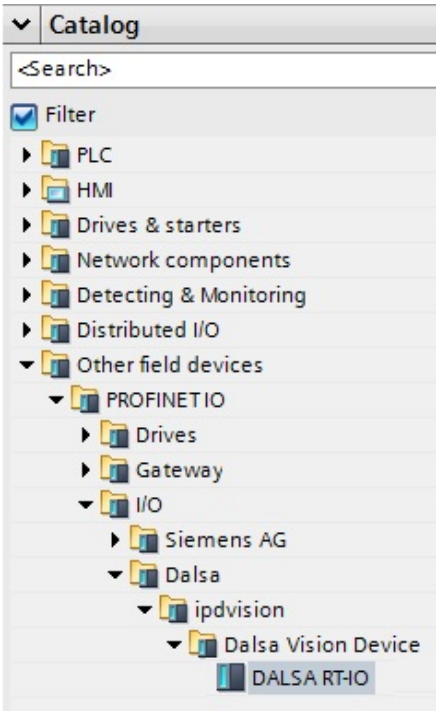
Note: The BOA Spot IP Address and Device Name must be already configured. You cannot set the address and name from the Profinet environment.

1. In the Options menu, select “Install general station description file”.



2. Navigate to find the GSD file. Select the file and click “Install”. In this example we copied it to D:\Profinet.

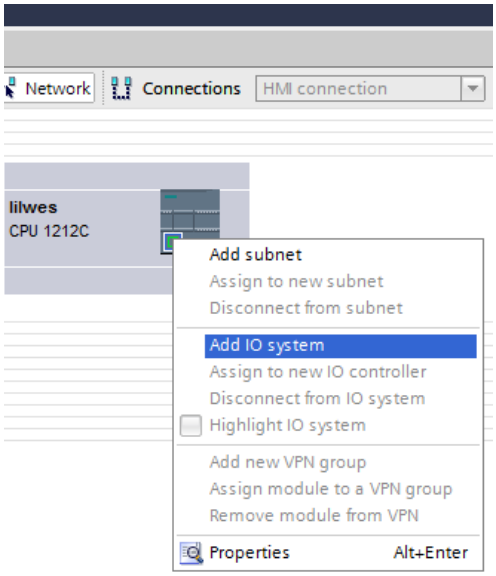
3. Click on “Devices and Networks”.
4. Open the Hardware catalog (shown at right). The DALSA RT-IO Device is automatically added to the Hardware Catalog when you install the GSDML file.
5. You can drag and drop the “DALSA RT-IO” object into the Devices and Networks view.



Note: The BOA Spot does not appear, unless the BOA Spot Solution has been run (at least once).

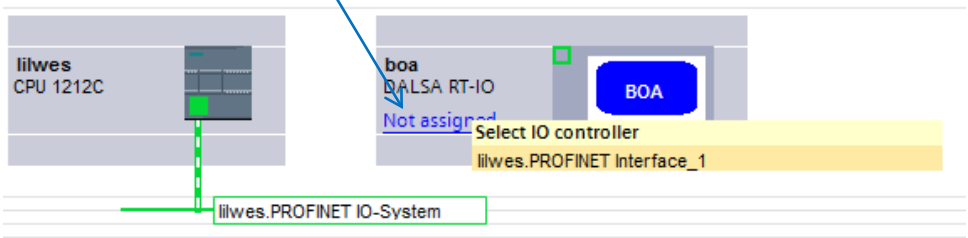
Tip: It can be much easier to add and configure one BOA Spot at a time, rather than add multiple BOA Spots and configure them to be name unique.

7. In the Network View, click on the green square on the CPU, and select “Add IO system”.



Tip: If you are familiar with the Profinet TIA Panel, you will know there are a few ways to create and configure the connections outlined on this page.

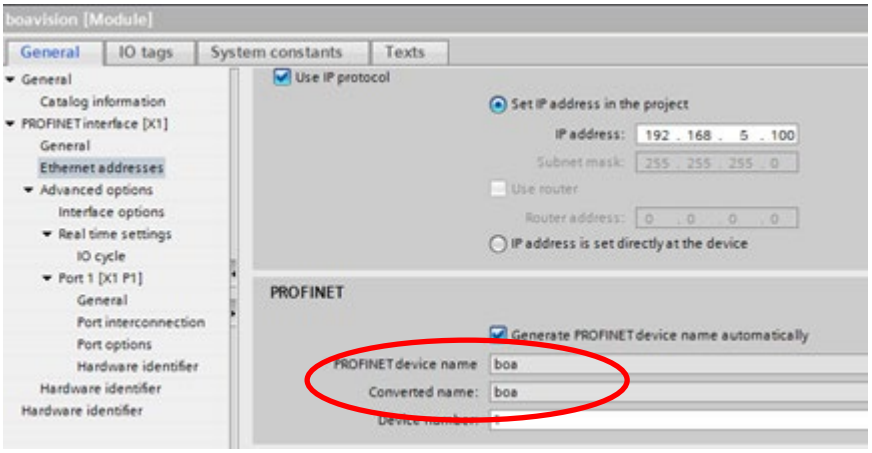
8. Click on “Not assigned” on the “BOA” and select the CPU, named “lilwes” in this image.



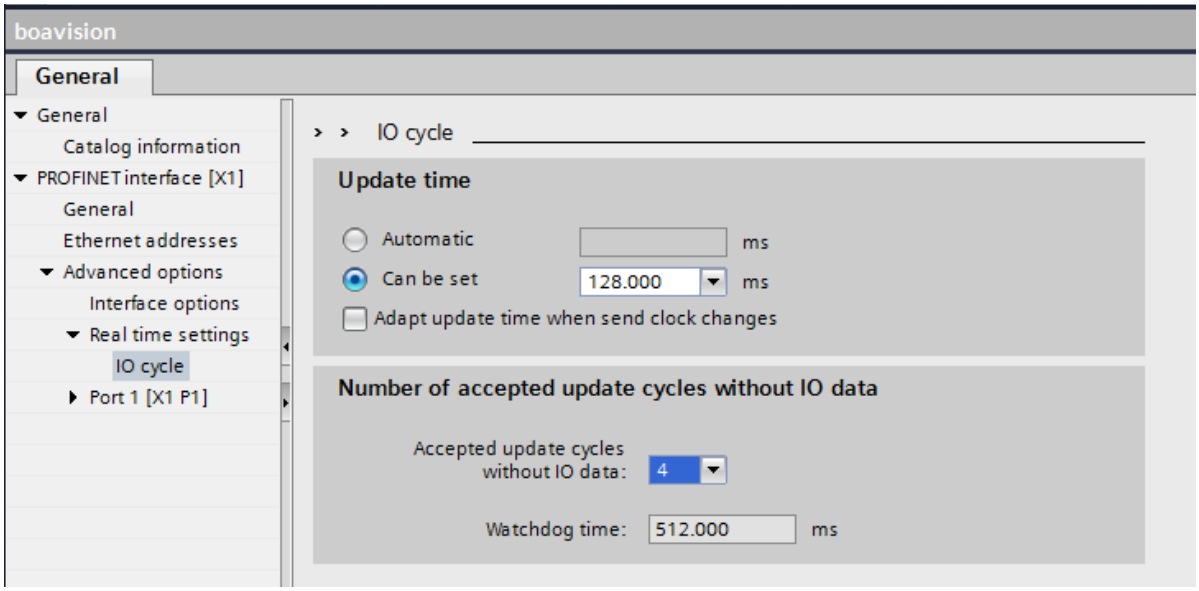
9. Click on the connection and verify or assign PN/IE_1 communication.
10. Click on the green square on the “BOA” or double-click on the name “boa”.
11. Select Ethernet address. If tabs appear, click the Properties tab.
12. Enter the IP Address of your BOA Spot. If there is only one BOA Spot, the “Profient device name” and “Converted name” fields are the same. You can use the default name “boa” or change it in the Device Overview or the Network Overview.

Note: You cannot change the device name in this view.

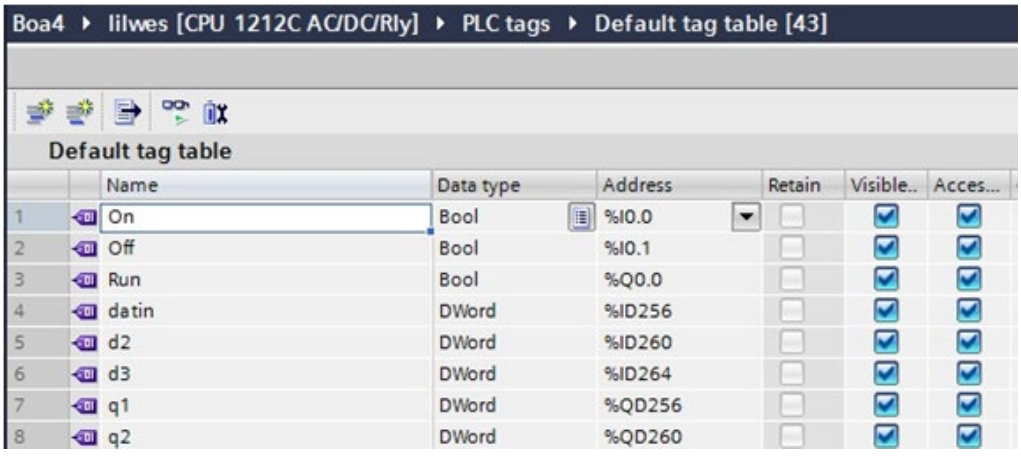
The “Device name” in the BOA Spot (in Device setup or iDiscover) must match the name in the “Converted name” field for Profinet to connect correctly. Correct the BOA Spot if needed.



12. Under the “Real-time settings” select “I/O cycle”. Timers.
13. Set the Update Time and Watchdog Timer. The default values seemed to work for a small read/write program running continuously. Values 128 ms and 4 cycles worked well in our tests. **Note:** When you get to production time, you may need to experiment with what Watchdog and cycle time works best for your program and environment. Larger values may be necessary for a more complex program, slow trigger rate, or driven by a slow product line.

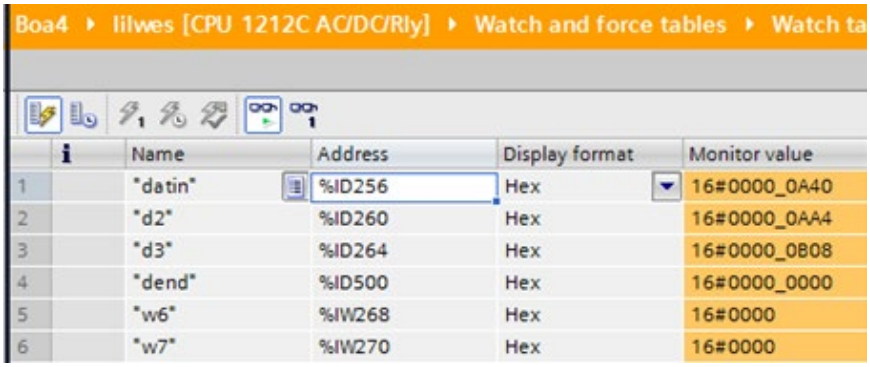


14. Add controller logic to perform IO with the BOA Spot. Configure the tag names and data types to match what your PLC and BOA Spot programs will read and write. The image below shows where you make changes, but this image has not yet been configured for the BOA Spot variables in our script example.



15. Compile (Toolbar button) and download (Toolbar button) your program to the PLC. The PLC will be halted to accept the new program. If you created a new project, you may need to use the Search option to identify the PLC (lilwes) and then issue a separate Load.

16. You can open the “Watch Table” to monitor your tags being written to by the BOA Spot. The image below has not been configured for the BOA Spot variables shown in the Scripting example.



	Name	Address	Display format	Monitor value
1	"datin"	%ID256	Hex	16#0000_0A40
2	"d2"	%ID260	Hex	16#0000_0AA4
3	"d3"	%ID264	Hex	16#0000_0B08
4	"dend"	%ID500	Hex	16#0000_0000
5	"w6"	%IW268	Hex	16#0000
6	"w7"	%IW270	Hex	16#0000

If you add more than one BOA Spot, the device names must be changed (page 48) to be different. The default name in the GSD file is “boa”. Earlier versions used the name “boavision”. Please refer to page 6, “BOA Spot Name in Profinet” for special information on Profinet device names. The BOA Spot “Device name” must match Profinet’s “Converted name”.

Troubleshooting Profinet Problems

Problem: Profinet system does not find the BOA Spot.

Reason 1: BOA Spot program must be run first, to request Profinet communication.

Solution 1: Run the BOA Spot Solution before adding BOA Spot to the Profinet environment.

Reason 2: The BOA's Device Name does not match the Profinet Converted Name. IP Address is not compatible or not correct in Profinet setup.

Solution 2: Change the BOA to match the Converted name. Verify the address on both sides. Refer to page 48.

Problem: BOA Spot does not accept changing the IP Address from TIAP environment.

Reason: Devices with their own CPU and OS usually do not support this.

Solution: Configure the BOA Spot IP Address from iDiscover or Device Setup.

Problem: BOA Spot keeps disconnecting from the PLC.

Reason 1: Trigger too fast for processing or image display update. Image display takes 1 byte per pixel.

Solution 1: Trigger slower. Increase the time between triggers.

Reason 2: Communication timeout;. Inspection time may be longer than your first tests if you have added more measurements. Image logging will increase the execution time. Some inspection tools require more time.

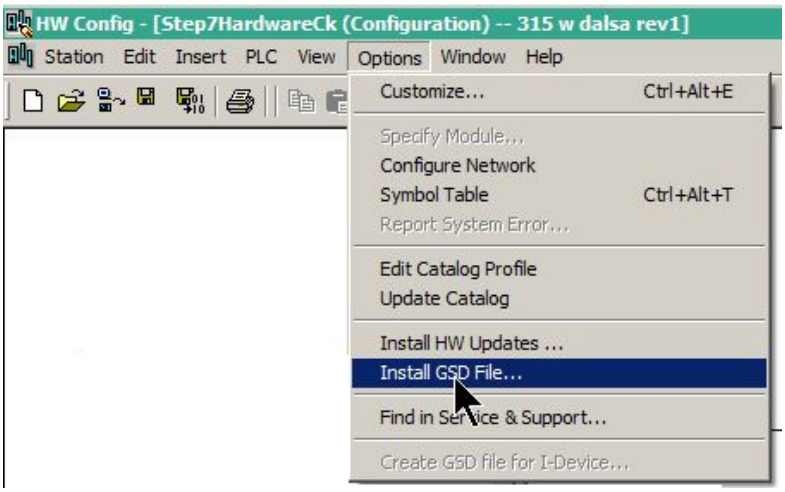
Solution 2: Try increasing the "Number of accepted cycles without I/O data". See "Note" on page 49.

Add the BOA Spot to an existing Simatic Manager Step7 Project

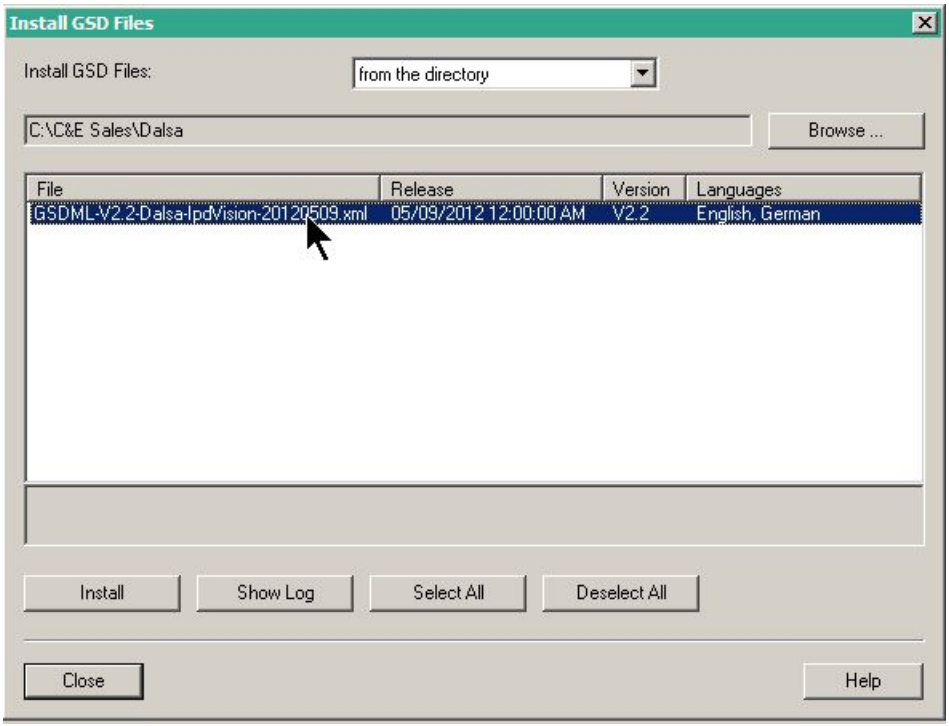
Instructions for the TIAP environment begin on page 47.

Note: The BOA Spot IP Address and Device Name must be already configured. You cannot set the address and name from the Profinet environment.

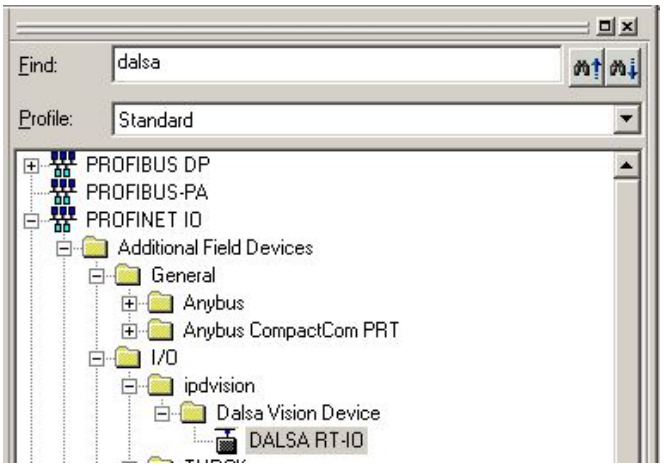
1. In the Options menu, select “Install GSD file”.



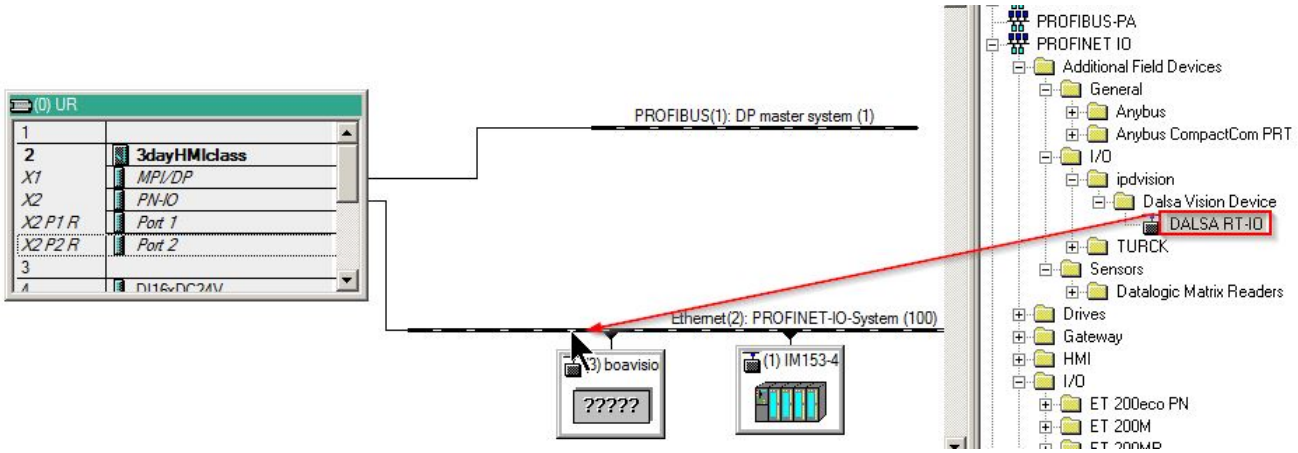
2. Navigate to find the GSD file. Select the file and click “Install”.



3. Open the Hardware catalog (shown at right). The “DALSA RT-IO” Device is automatically added to the Hardware Catalog when you install the GSDML file.



4. Drag and drop the “DALSA RT-IO” object onto the “Ethernet PROFINET-IO System” bus.

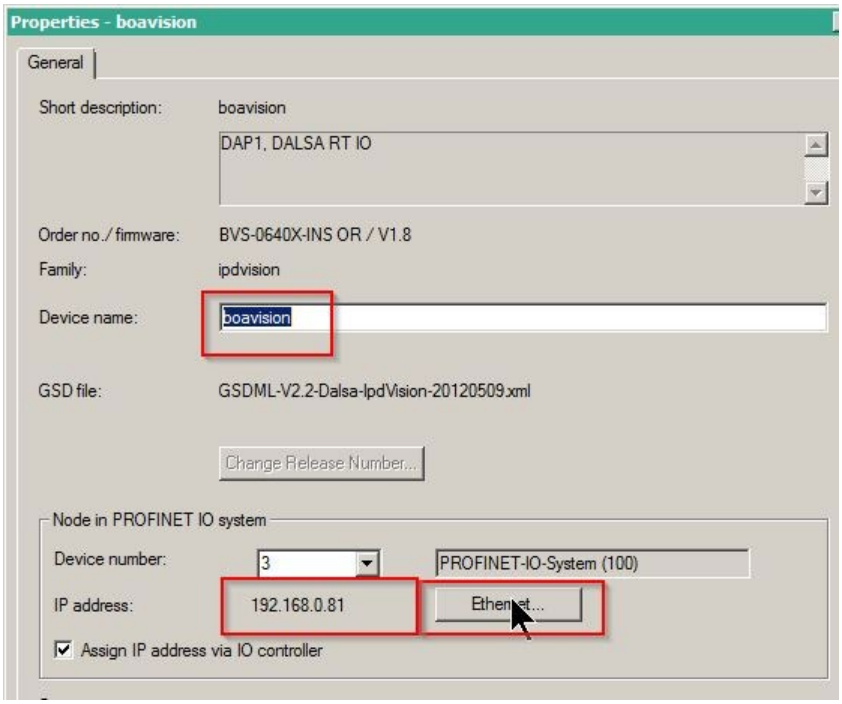


Note: The BOA Spot does not appear, unless the BOA Spot Solution has been run (at least once). The default name is “boavision” in the 2014 version of the GSD file, or “boa” in the 2016 version of the file.

Tip: It can be much easier to add and configure one BOA Spot at a time, rather than add multiple BOA Spots and configure them to be name unique.

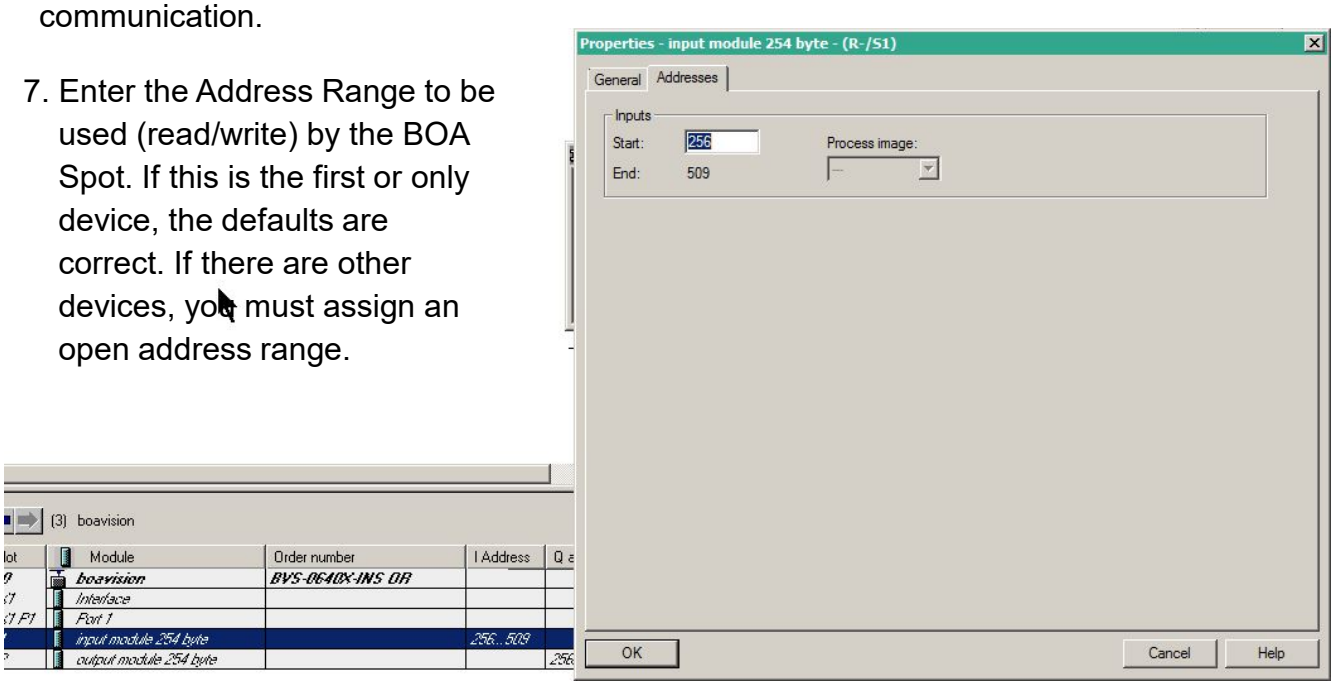
5. Double-click on the name “boa” or “boavision” or right click and select “Properties”.

6. Enter the Device Name and IP Address of your BOA Spot, as they appeared in “iDiscover” or in the BOA Spot Device Setup page.

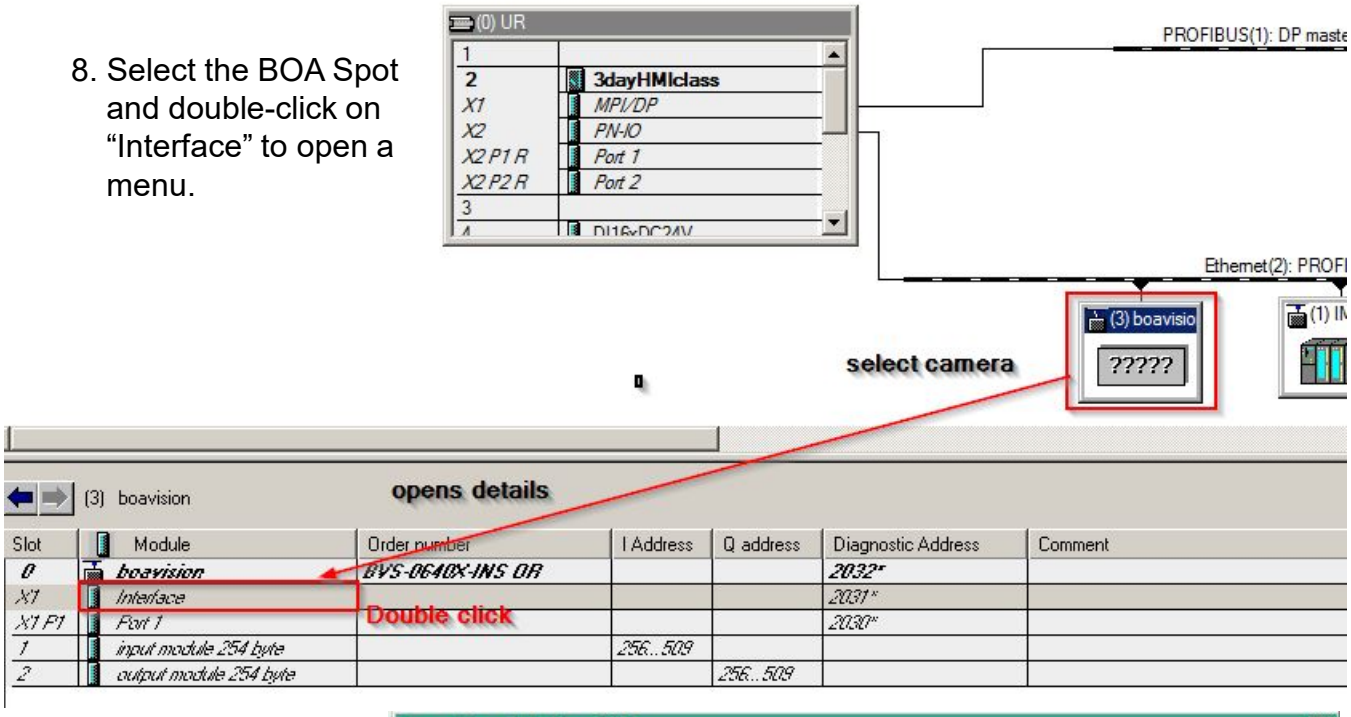


Note: The BOA Spot address and name cannot be changed by the Simatic Manager. You must change them in the BOA Spot (in the Device Setup page or iDiscover) The “device name” must be identical in both environments for successful communication.

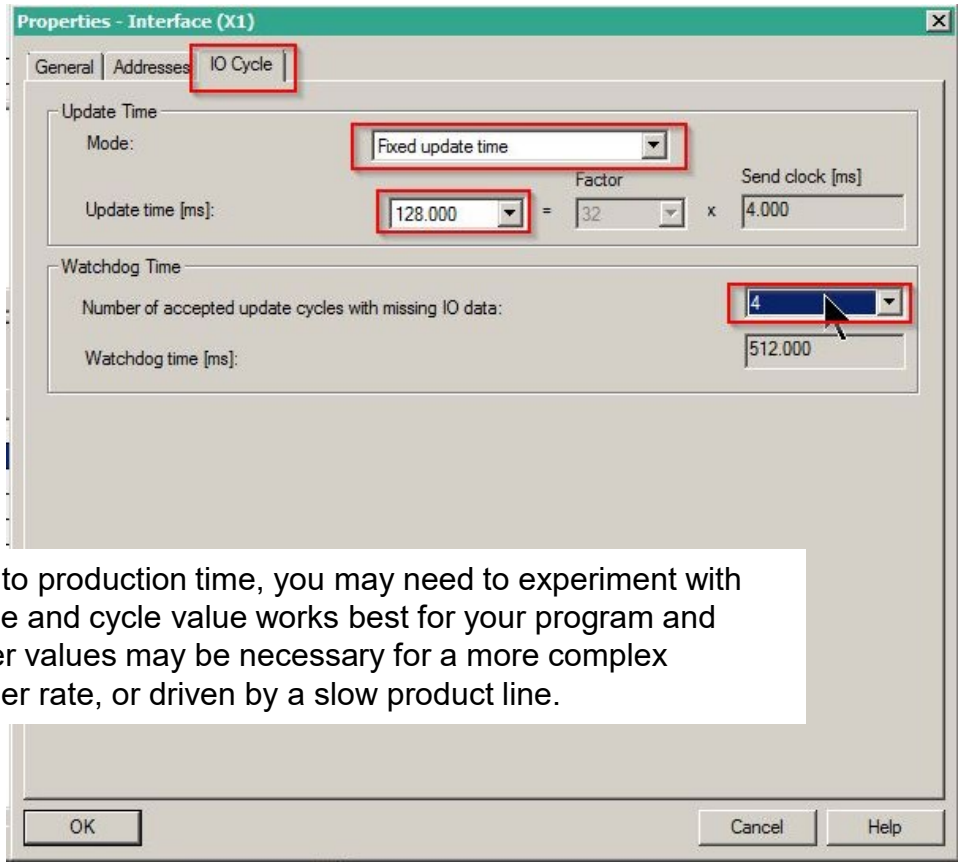
7. Enter the Address Range to be used (read/write) by the BOA Spot. If this is the first or only device, the defaults are correct. If there are other devices, you must assign an open address range.



8. Select the BOA Spot and double-click on “Interface” to open a menu.



9. Set the Update Time and Watchdog Timer. The default values seemed to work for a small read/write program running continuously. Values 128 ms and 4 cycles worked well in our tests.



Note: When you get to production time, you may need to experiment with what Watchdog time and cycle value works best for your program and environment. Larger values may be necessary for a more complex program, slow trigger rate, or driven by a slow product line.

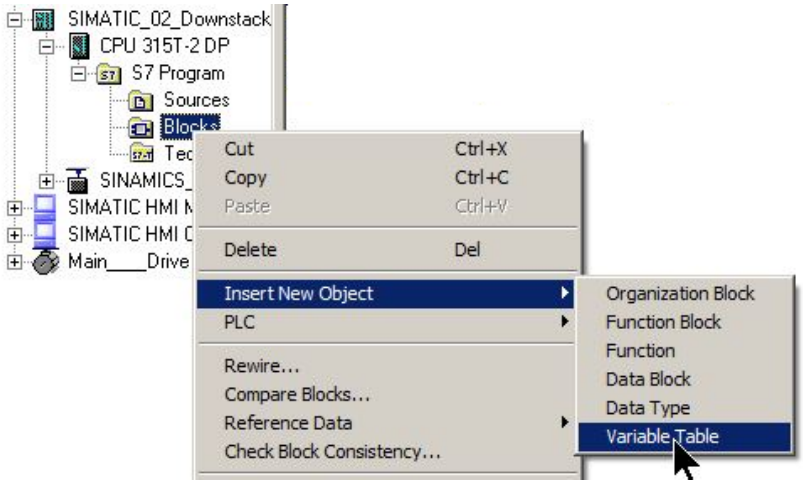
10. Open the Symbol Table and enter desired variables which perform IO with the BOA Spot. Configure the symbol names, addresses and data types to match what your PLC and BOA Spot programs will read and write.
    - BOA Spot writes to PIW 256 through 509
    - BOA Spot reads from PQW 256 through 509

S7 Program (Symbols) -- 26940-Ardargh Deeside\SIMATIC_02_Downstacker\CPU 315T-2 DP

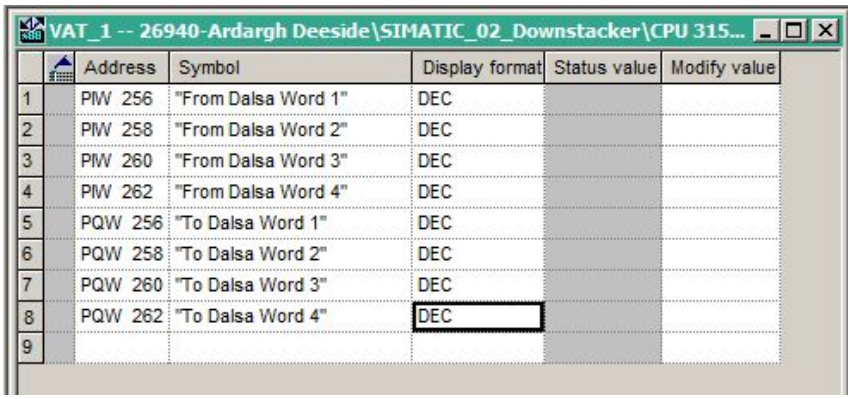
	Status	Symbol	Address	Data type	Comment
88		temporary jumper	M 250.0	BOOL	
89		TON	SFB 4	SFB 4	Generate an On Delay
90		Trace	DB 1	UDT 8	S7 data for Trace of the MC subsystem
91		TraceData	UDT 8	UDT 8	UDT: TraceData
92		Zero Flt to 319F	Q 4.5	BOOL	
93		Zero Master Encoder	DB 503	FB 432	
94		ZERO MSTER ENC ER...	MW 368	WORD	ZERO MASTER ENCODER ERROR ID MONITOR
95		From Dalsa Word 1	PIW 256	WORD	
96		From Dalsa Word 2	PIW 258	WORD	
97		From Dalsa Word 3	PIW 260	WORD	
98		From Dalsa Word 4	PIW 262	WORD	
99		To Dalsa Word 1	PQW 256	WORD	
100		To Dalsa Word 2	PQW 258	WORD	
101		To Dalsa Word 3	PQW 260	WORD	
102		To Dalsa Word 4	PQW 262	WORD	
103					

Note: Addresses are in the periphery area. Use the appropriate addressing.

11. Create a VAT Table. Right-click on “Blocks”, move your cursor to “Insert New Object” and click on “Variable Table”.



12. Populate the table with the variables you wish to monitor in the Step7 environment.



	Address	Symbol	Display format	Status value	Modify value
1	PIW 256	"From Dalsa Word 1"	DEC		
2	PIW 258	"From Dalsa Word 2"	DEC		
3	PIW 260	"From Dalsa Word 3"	DEC		
4	PIW 262	"From Dalsa Word 4"	DEC		
5	PQW 256	"To Dalsa Word 1"	DEC		
6	PQW 258	"To Dalsa Word 2"	DEC		
7	PQW 260	"To Dalsa Word 3"	DEC		
8	PQW 262	"To Dalsa Word 4"	DEC		
9					

13. Compile (Toolbar button) and download (Toolbar button) your program to the PLC. The PLC will be halted to accept the new program.

If you add more than one BOA Spot, the device names must be changed (page 53) to be different. The default name in the GSD file is “boa”. Earlier versions used the name “boavision”. Please refer to page 6, “BOA Spot Name in Profinet” for special information on Profinet device names. The BOA Spot’s “Device name” must match the Step7 “Device name”.

Troubleshooting Profinet Problems

Problem: Profinet system does not find the BOA Spot.

Reason 1: The BOA Spot Solution must be run first, to request Profinet communication.

Solution 1: Run the BOA Spot Solution before adding BOA Spot to the Profinet environment.

Reason 2: The BOA Spot's Device Name does not match the Profinet Device Name. IP Address not compatible or does not match in Profinet setup.

Solution 2: Change the BOA Spot Name to match the Profinet name. Verify the IP Address in Profinet setup.. See page 54.

Problem: BOA Spot does not accept changing the IP Address from TIAP environment.

Reason: Devices with their own CPU and OS usually do not support this.

Solution: You must configure the BOA Spot Device Name and IP Address using iDiscover or the Device Setup page.

Problem: BOA Spot keeps disconnecting from the PLC.

Reason 1: Trigger too fast for processing or image display update. Image display takes 1 byte per pixel.

Solution 1: Trigger slower. Increase the time between triggers.

Reason 2: Communication timeout;. Inspection time may be longer than your first tests if you have added more measurements. Image logging will increase the execution time. Some inspection tools require more time.

Solution 2: Increase "Number of accepted cycles without I/O data". See the "Note" page 55.

Modbus EIP

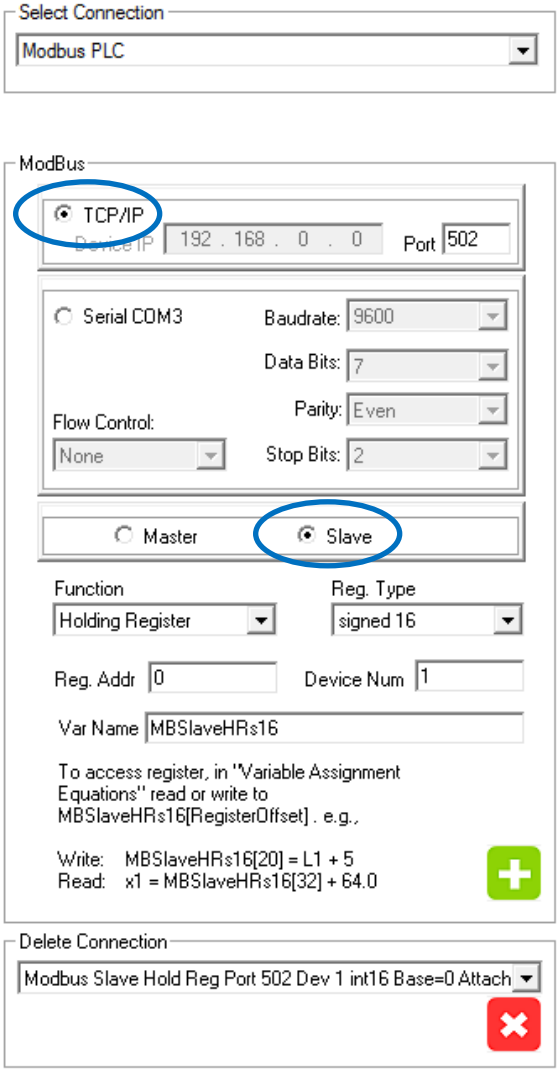
BOA Spot is compatible with PLCs and many other devices that use the Modbus protocol, over Ethernet or Serial port connections.

Configure the BOA Spot

1. Load or create a simple Solution with one or two tools.
2. In the Navigation bar, click on Setup Connections.

3. Use the drop list to select “Modbus PLC”.

The center of the Setup panel changes to configure BOA Spot for Modbus communication.



4. Select the TCP/IP connection. Select the BOA Spot to be a Modbus Slave.
5. When you configure the PLC, you will need the BOA Spot IP Address (not shown) and the Modbus Port number shown in this panel (502). This port number is used by most Modbus equipment.
6. We will use the default setting for this example: Holding Registers, signed 16, Address 0, Device 1.

There are several different register types. You should use the Holding registers only. Other registers are restricted to the PLCs use, mostly for coils and relays. These registers do not respond to RW access by other devices.

7. Click “Add”. Your definition appears in the “Delete Connection” list box.

Notice the Read/Write examples just below the “Add” button. The examples are given using the default variable name “MBSlaves16”. These are the type of statements you would use in the Script Editor to read values from or write values to the PLC. “HR” is for the Holding register. “s16” is for the “signed 16” 16-bit value.

MBSlaveHRs16[20] = L1 + 5 -- Translates to: Write the sum of L1 plus 5 to the PLC’s Holding register at BOA Spot index 20.

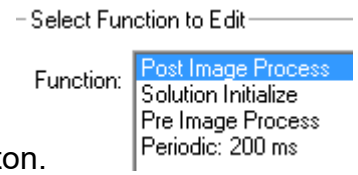
X1 = MBSlaveHRs16[32] + 64.0 – Translates to: Read the value in Holding register BOA Spot index 32, add 64.0 to the value, and store it in variable X1.

Tip: The Input and Output Registers are at the same space in a Modbus system. You can read back the same data that you write.

Note: The BOA Spot attached variable MBSlaveHRs16[0] maps to the **Holding Registers** which start at location 400001 in the Modbus memory. This is important to your Modbus device programming.

Now add scripts to communicate with the PLC.

1.In the Navigation bar, click on “Edit Scripts”.



2.In the setup panel (Left) click on “Post Image Process”.

3.In the bottom panel below the image area, click the “Edit” button.

4.Add equations or statements that read or write to the PLC. Use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

In the **Post Image Process** function:

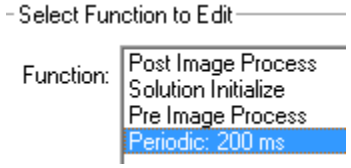
```
//Send measured results to the PLC Input registers
MBSlaveHRs16[0] = IntenAvg
MBSlaveHRs16[1] = Result.0
MBSlaveHRs16[2] = Global.PassCount
inspBusy = 0
```

5. Click the “Check Syntax” button to check for errors. 

6. Click the “Save” button, to save and close the Free Edit window. 

Tip: Start with writing just a few numbers to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

In most applications, you want the BOA Spot to switch Solution and trigger on a signal from the PLC.



7. In the Edit Scripts setup panel click on “Periodic: 200 ms”.

8. In the bottom panel below the image area, click the “Edit” button.

9. Add statements that tell the PLC which solution is running and read a solution change request from the PLC output registers. (example on the next page)

10. Add statements that trigger the BOA Spot, clear and then rearm the trigger. (example on the next page)

In the **Periodic: 200 ms** function:

```
//Send current solution and total image count to PLC
MyRunningID = GetSolutionID()
MBSlaveHRs16[3] = MyRunningID
MBSlaveHRs16[4] = Global.FrameCount
//
// get any solution change request from PLC
solReq = MBSlaveHRs16[5]
if(solReq != MyRunningID)       //if the new request is a different Solution
    ChangeSolution(solReq)      //Load the new Solution Job file
endif
// get any trigger request from PLC
trigReq = MBSlaveHRs16[6]
// only trigger on leading edge of register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0               // to prevent multiple triggers
    inspBusy = 1                // the inspection is running
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
```

11. Check for syntax errors. 
12. Click “Save” to save changes and close the Edit window. 
13. Save, reload and then Run the Solution. (See “Important” on page 6.)

Note: The Windows Firewall should be **turned off** or customized on systems that communicate over the network.

Configure the PLC for BOA Spot

The PLC is the Modbus master. The BOA Spot is configured as a Modbus slave. The Modbus master requires the slave device's IP Address and Port number.

1. Open your Modbus PLC programming software.
2. Add a new slave module connection.
3. Enter the BOA Spot's IP Address and Port number (Port 502 in the example).
4. Add the necessary number of variables and assign their data tags (if supported) locations (starting at 400001) names and properties as needed (data size, data format, read/write or read-only access). In this example the BOA Spot used signed 16-bit values. There are several other data types available.

Note: The BOA Spot attached variable MBSlaveHRs16[0] maps to the **Holding Registers** which start at location 400001 in the Modbus memory. This is important to your Modbus device programming.

MBSlaveHRs16[0] maps to 400001, MBSlaveHRs16[1] maps to 400002, MBSlaveHRs16[2] maps to 400003, MBSlaveHRs16[3] maps to 400004, etc..

In the Modbus register table specification:

Coil Status starts at location 000001, 16-bit R/W (you may see 0:00000 in the spec., 00001 is the first usable location)
Input Status starts at location 100001, 1-bit R-O
Input Registers start at location 300001, 16-bit R-O
Holding Registers start at location 400001, 1-bit R/W

Note: You can read and write to the same Holding register locations in the table. The BOA Spot and iNspect only communicate with the Holding registers. The other registers are dedicated to PLC communication with hardware, such as Coils and Relays.

5. Compile and download your program to the PLC.
6. Monitor the Holding Registers in your programming software, if supported on your system. Observe the BOA Spot writing to the Holding Registers.
7. Add the logic necessary to process the BOA Spot data, and to trigger and switch solutions as appropriate.

There are many different devices that use the Modbus standard. A few examples are robot arms, control modules, and HMI panels. The software for developing programs will be different for each, as well as the capabilities. Modbus is supported on both Serial and Ethernet.

Fanuc SNP

BOA Spot is compatible with PLCs that use the GE Fanuc SNP protocol, over a serial port connection.

Configure the BOA Spot

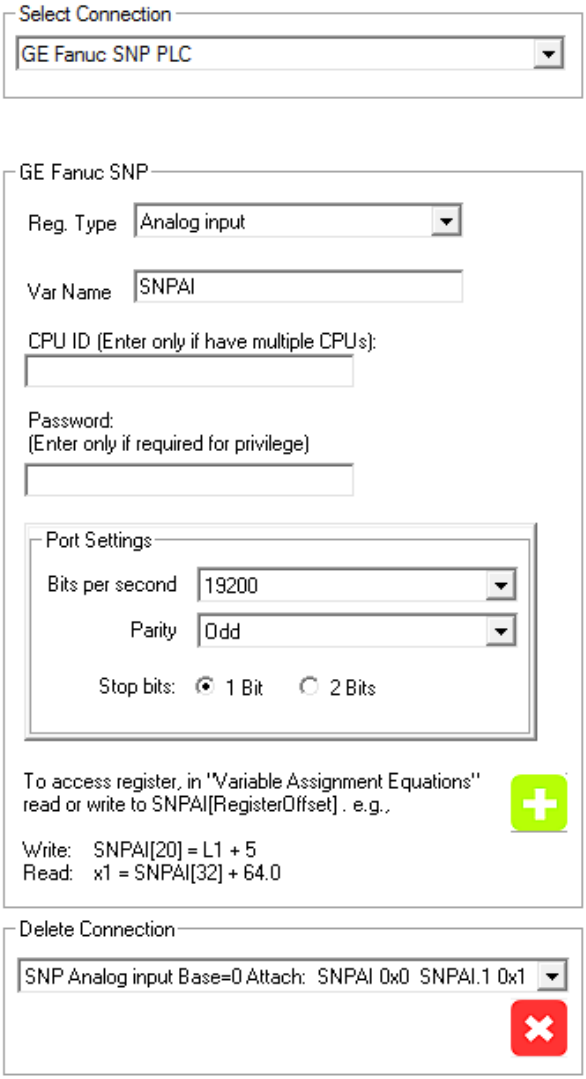
1. In the Navigation bar, click on Setup Connections.
2. Use the drop list to select “GE Fanuc SNP PLC”.

The center of the Setup panel changes to configure BOA Spot for SNP communication.

3. Select a Register type, and change the Serial Port settings, or verify that they match the PLC exactly.

There are several different register types. Each performs a specific task and has a specific data type it will accept.

4. You can change the variable name or use the defaults. This default name also signifies the type of register: The “AI” is for the Analog input register.
5. Click “Add”. Your definition appears in the “Delete Connection” list box.



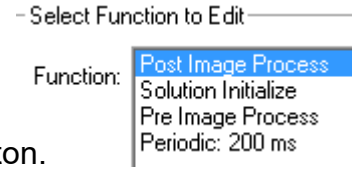
Notice the Read/Write examples just below the “Add” button. The examples are given for the default variable name “SNPAI”. These are the type of statements you would use in the Script Editor to read values from or write values to the PLC.

The default name also signifies the type of register: The “AI” for Analog input register.

SNPAI0[20] = L1 + 5 – Translates to: Write the sum of L1 plus 5 to the PLC’s Analog input register at index location 20.

X1 = SNPAI0[32] + 64.0 – Translates to: Read the value in Analog Input register index location 32, add 64.0 to the value, and store it in variable X1.

Now add scripts to communicate with the PLC.



1.In the Navigation bar, click on “Edit Scripts”.

2.In the setup panel (Left) click on “Post Image Process”.

3.In the bottom panel below the image area, click the “Edit” button.

4.Add equations or statements that read or write to the PLC. Use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

In the **Post Image Process** function:

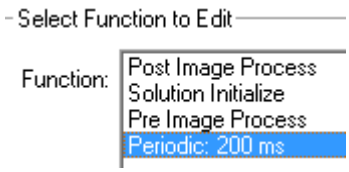
```
//Send results to the PLC Input registers
SNPAI0[0] = IntenAvg
SNPAI0[1] = Result.0
SNPAI0[2] = Global.PassCount
// inspection is done status
inspBusy = 0
```

5. Click the “Check Syntax” button to check for errors. 

6. Click the “Save” button, to save and close the Free Edit window. 

Tip: Start with writing one number to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

In most applications, you want the BOA Spot to switch Solution and trigger on a signal from the PLC.



7. In the Edit Scripts setup panel click on “Periodic: 200 ms”.

8. In the bottom panel below the image area, click the “Edit” button.

9. Add statements that tell the PLC which solution is running and read a solution change request from the PLC output registers. (example on the next page)

10. Add statements that trigger the BOA Spot, clear and then rearm the trigger. (example on the next page)

In the **Periodic: 200 ms** function:

```
//Send current solution to PLC
MyRunningID = GetSolutionID()
SNPAI0[3] = MyRunningID
SNPAI0[4] = Global.FrameCount
// get any solution change request from PLC Output registers
solReq = SNPAI0[1]
if(solReq !-MyRunningID)          //If the request is a different Solution
    ChangeSolution(solReq)        //Load the new Solution
endif
// get any trigger request from PLC output registers
trigReq = SNPAI0[0]
// only trigger on leading edge of the register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0                 // prevents multiple triggers
    inspBusy = 1                  // the inspection is running
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
//note PLC connection status
plcStat = IsConnected(SNPAI0)
//
//send heartbeat count - Optional
hb = hb + 1
if(hb > 999) hb = 1
SNPAI0[5] = hb
```

11. Check for syntax errors. 
12. Click “Save” to save changes and close the Edit window. 
13. Save, reload and then Run the Solution. (See “Important” on page 6.)

Configure the PLC for BOA Spot

Setup and programming information specific to the Fanuc SNP PLC was not available. A generalized setup is provided here. The Fanuc SNP uses Serial RS232 communication.

1. Open your PLC programming software.
2. Add a new RS232 connection. Make sure the setting match those of the BOA Spot.
3. Add the necessary number of variables. Configure variable data settings as needed.

Note: It is not known if you can read and write to the same locations in this PLC.

4. Compile and download your program to the PLC.
5. Monitor the PLC memory or registers in your programming software, if supported on your system. Observe the BOA Spot writing to the PLC.
6. Add the logic necessary to process the BOA Spot data, and to trigger and switch solutions as appropriate.

Fanuc SRTP

BOA Spot is compatible with PLCs that use the GE Fanuc SRTP protocol, over an Ethernet connection.

Configure the BOA Spot

1. In the Navigation bar, click on Setup Connections.
2. Use the drop list to select ”GE Fanuc SRTP Ethernet PLC”.

The center of the Setup panel changes to configure BOA Spot for SRTP communication.

3. Select the Register type, and enter the IP Address of the **PLC**.

There are several different register types. Each performs a specific task and has a specific data type it will accept.

4. You can change the variable name or use the defaults.
5. Click the “Add” button. Your definition appears in the “Delete Connection” list box.

Notice the Read/Write examples just below the “Add” button. The examples are given for the default variable name “SRTPAI0”. These are the type of statements you would use in the Script Editor to read values from or write values to the SRTP PLC.

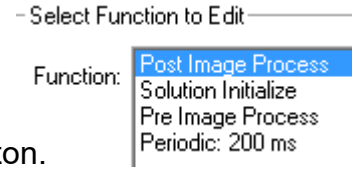
The default name also signifies the type of register: The “AI” for Analog input register.

SRTPAI0[20] = L1 + 5 – Translates to: Write the sum of L1 plus 5 to the PLC’s Analog input register at index location 20.

X1 = SRTPAI0[32] + 64.0 – Translates to: Read the value in Analog Input register index location 32, add 64.0 to the value, and store it in variable X1.

Depending on the design of the PLC, the read and write locations may be the same, or they may be different.

Now add scripts to communicate with the PLC.



1.In the Navigation bar, click on “Edit Scripts”.

2.In the setup panel (Left) click on “Post Image Process”.

3.In the bottom panel below the image area, click the “Edit” button.

4.Add equations or statements that read or write to the PLC. Use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

In the **Post Image Process** function:

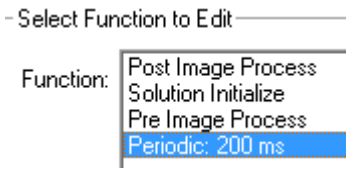
```
//Send results to the PLC Input registers
SRTPAI0[0] = IntenAvg
SRTPAI0[1] = Result.0
SRTPAI0[2] = Global.PassCount
// inspection is done status
inspBusy = 0
```

5. Click the “Check Syntax” button to check for errors. 

6. Click the “Save” button, to save and close the Free Edit window. 

Tip: Start with writing just a few numbers to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

In most applications, you want the BOA Spot to switch Solution and trigger on a signal from the PLC.



7. In the Edit Scripts setup panel click on “Periodic: 200 ms”.

8. In the bottom panel below the image area, click the “Edit” button.

9. Add statements that tell the PLC which solution is running and read a solution change request from the PLC output registers. (example on the next page)

10. Add statements that trigger the BOA Spot, clear and then rearm the trigger. (example on the next page)

In the **Periodic: 200 ms** function:

```
//Send current solution to PLC
MyRunningIDNum = GetSolutionID()
SRTPAI0[3] = MyRunningIDNum
SRTPAI0[4] = Global.FrameCount
// get any solution change request from PLC Output registers
solReq = SRTPAI0[1]
if(solReq != MyRunningIDNum)    //If the request is a different Solution
    ChangeSolution(solReq)      //Load the new Solution
endif
// get any trigger request from PLC output registers
trigReq = SRTPAI0[0]
// only trigger on leading edge of the register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0               // prevents multiple triggers
    inspBusy = 1                // the inspection is running
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
//note PLC connection status
plcStat = IsConnected(SRTPAI0)
//
//send heartbeat count
hb = hb + 1
if(hb > 999) hb = 1
SRTPAI0[5] = hb
```

11. Check for syntax errors. 
12. Click “Save” to save changes and close the Edit window. 
13. Save, reload and then Run the Solution. (See “Important” on page 6.)

Configure the PLC for BOA Spot

Setup and programming information specific to the Fanuc SRTP PLC was not available. A generalized setup is provided here. The Fanuc SRTP uses Ethernet communication.

1. Open your PLC programming software.
2. Add a new Ethernet connection. Enter the properties of the BOA Spot as needed.
3. Add the necessary number of variables. Configure variable data settings as needed.

Note: It is not known if you can read and write to the same locations in this PLC.

4. Compile and download your program to the PLC.
5. Monitor the PLC memory or registers in your programming software, if supported on your system. Observe the BOA Spot writing to the PLC.
6. Add the logic necessary to process the BOA Spot data, and to trigger and switch solutions as appropriate.

Note: The Windows Firewall should be **turned off** or customized on systems that communicate over the network.

Motoman

BOA Spot is compatible with devices that support the Motoman protocol, over a serial port connection.

Configure the BOA Spot

1. In the Navigation bar, click on Setup Connections.

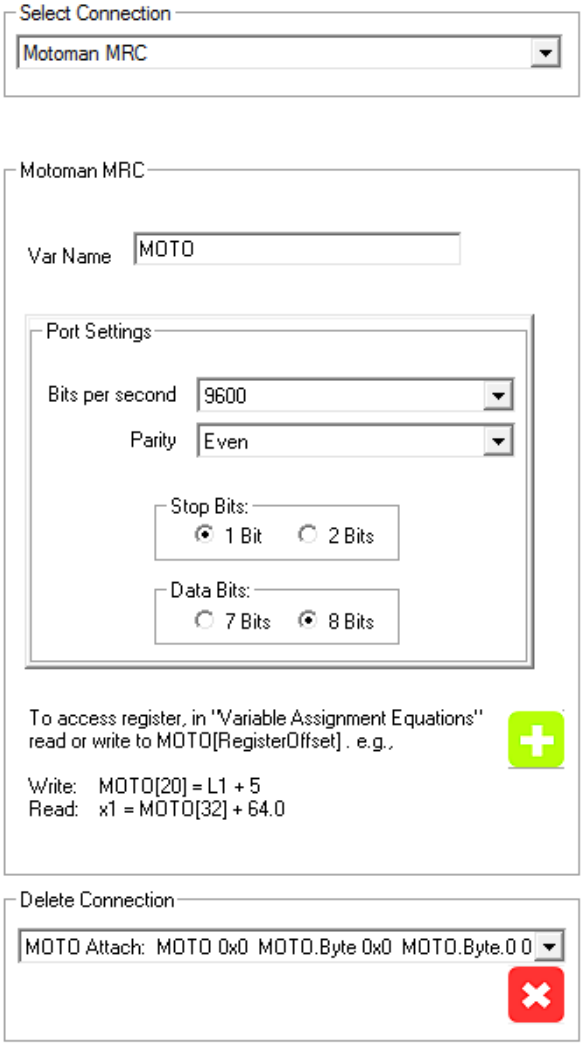
2. Use the drop list to select “Motoman MRC”.

The center of the Setup panel changes to configure BOA Spot for Motoman communication.

3. Change the Serial Port settings or verify that they match exactly the other Motoman systems you will communicate with.

4. You can change the Variable name or use the default.

5. Click “Add”. Your definition appears in the “Delete Connection” list box.



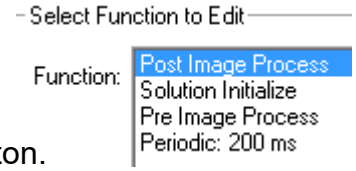
Notice the Read/Write examples just below the “Add” button. The examples are given for the default variable name “MOTO”. These are the type of statements you would use in the Script Editor to read values from or write values to a Motoman device.

MOTO[20] = L1 + 5 – Translates to: Write the sum of the L1 measurement plus 5, to the Motoman write location 20.

X1 = MOTO[32] + 64.0 – Translates to: Read the value at Motoman read location 32, add 64.0 to the value, and store it in variable X1.

Usually devices have a bank of register locations or addresses. There may be specific functions assigned to specific locations. This would be outlined in the documentation for your Motoman device or system.

Now add scripts to communicate with the PLC.



1.In the Navigation bar, click on “Edit Scripts”.

2.In the setup panel (Left) click on “Post Image Process”.

3.In the bottom panel below the image area, click the “Edit” button.

4.Add equations or statements that read or write to the PLC. Use the “Variable” button and menus to add your variables. Your Variable Name appears in the list of variables.

In the **Post Image Process** function:

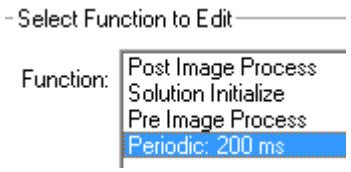
```
//Send results to the PLC Input registers
MOTO[0] = IntenAvg
MOTO[1] = Result.0
MOTO[2] = Global.PassCount
// inspection is done status
inspBusy = 0
```

5. Click the “Check Syntax” button to check for errors. 

6. Click the “Save” button, to save and close the Free Edit window. 

Tip: Start with writing just a few numbers to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

In most applications, you want the BOA Spot to switch Solution and trigger on a signal from the PLC.



7. In the Edit Scripts setup panel click on “Periodic: 200 ms”.

8. In the bottom panel below the image area, click the “Edit” button.

9. Add statements that tell the PLC which solution is running and read a solution change request from the PLC output registers. (example on the next page)

10. Add statements that trigger the BOA Spot, clear and then rearm the trigger. (example on the next page)

In the **Periodic: 200 ms** function:

```
//Send current solution to PLC
MyRunningIDNum = GetSolutionID()
MOTO[3] = MyRunningIDNum
MOTO[4] = Global.FrameCount
// get any solution change request from PLC Output registers
solReq = MOTO[1]
if(solReq != MyRunningIDNum)      //If the request is a different Solution
    ChangeSolution(solReq)        //Load the new Solution job file
endif
// get any trigger request from PLC output registers
trigReq = MOTO[0]
// only trigger on leading edge of the register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0                 // prevents multiple triggers
    inspBusy = 1                  // the inspection is running
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
//note PLC connection status
plcStat = IsConnected(MOTO)
//
//send heartbeat count
hb = hb + 1
if(hb > 999) hb = 1
MOTO[5] = hb
```

11. Check for syntax errors. 
12. Click “Save” to save changes and close the Edit window. 
13. Save, reload and then Run the Solution. (See “Important” on page 6.)

Configure the PLC for BOA Spot

Setup and programming information specific to the Motoman PLC was not available. A generalized setup is provided here. The Motoman uses Serial RS232 communication.

1. Open your PLC programming software.
2. Add a new RS232 connection. Make sure the setting match those of the BOA Spot.
3. Add the necessary number of variables. Configure variable data settings as needed.

Note: It is not known if you can read and write to the same locations in this PLC.

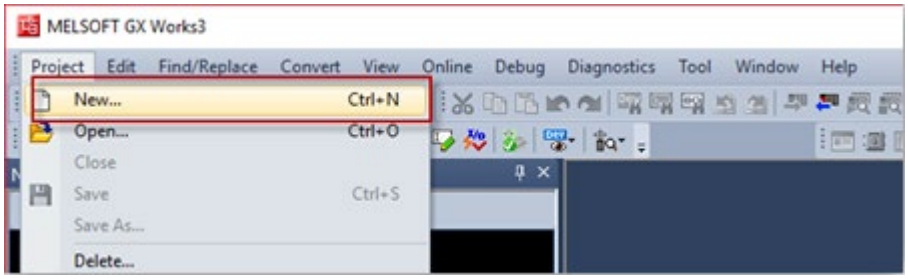
4. Compile and download your program to the PLC.
5. Monitor the PLC memory or registers in your programming software, if supported on your system. Observe the BOA Spot writing to the PLC.
6. Add the logic necessary to process the BOA Spot data, and to trigger and switch solutions as appropriate.

This example uses the FX5 PLC as an SLMP Server and the BOA Spot will be the SLMP Client.

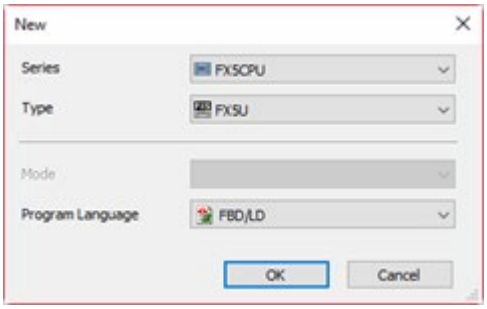
The BOA Spot IP Address is 192.168.3.100 and the FX5 IP Address is 192.168.3.50.

Configure the PLC for BOA Spot

1. Open the GX Works3 programming environment.
2. Create a new project.

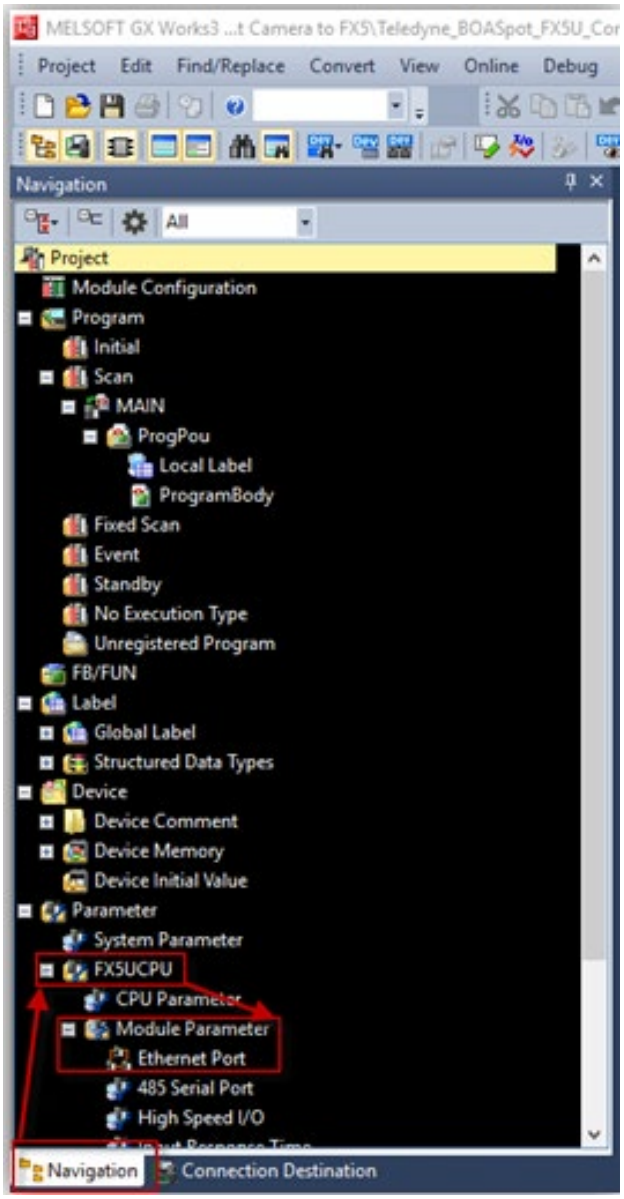


3. Select the Series as **FX5CPU** and the Type as **FX5U**.
4. Click OK.

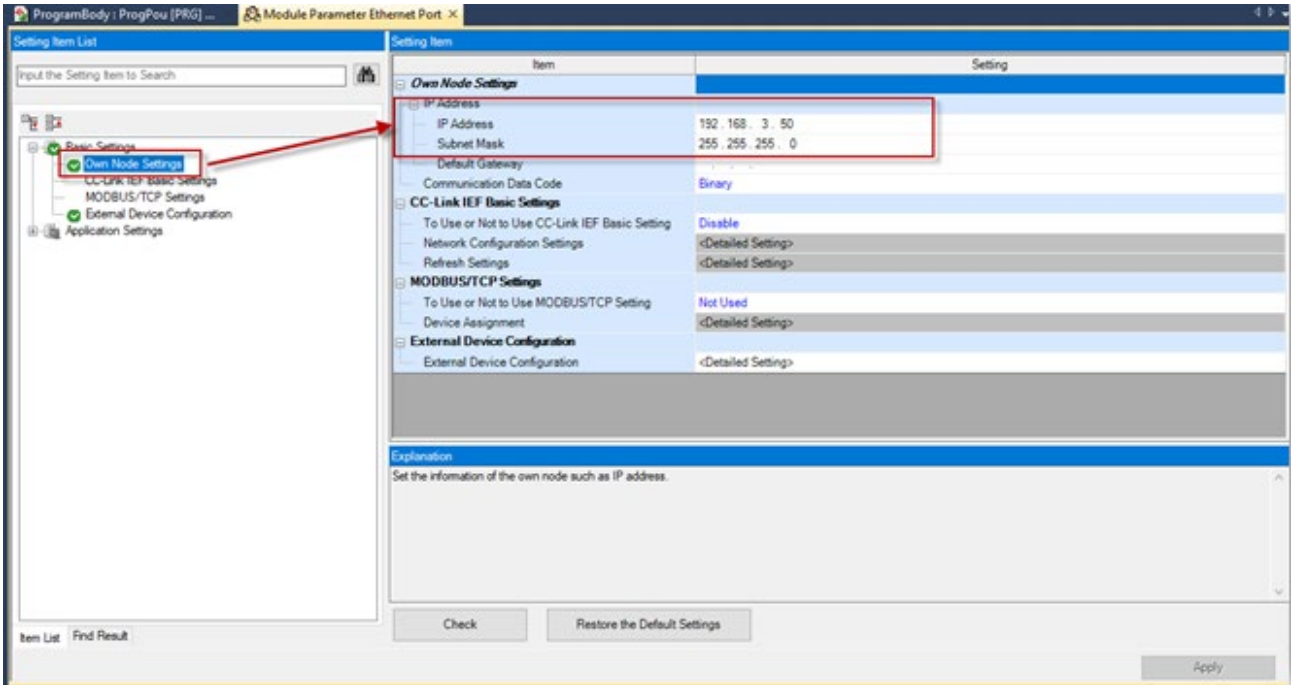


5. The next menu (not shown) allows you to set up module labels. These values are at your discretion and do not impact network communications. This setting allows you to take advantage of and use labels that are automatically assigned to special devices, buffer memory, etc..

6. Open the Navigation window.
7. Scroll down and expand the Parameter settings. Expand FX5UCPU, Module, and Parameter.
8. Double click on the Ethernet Port. This will direct you to settings that controls the Ethernet port that is built into the FX5.



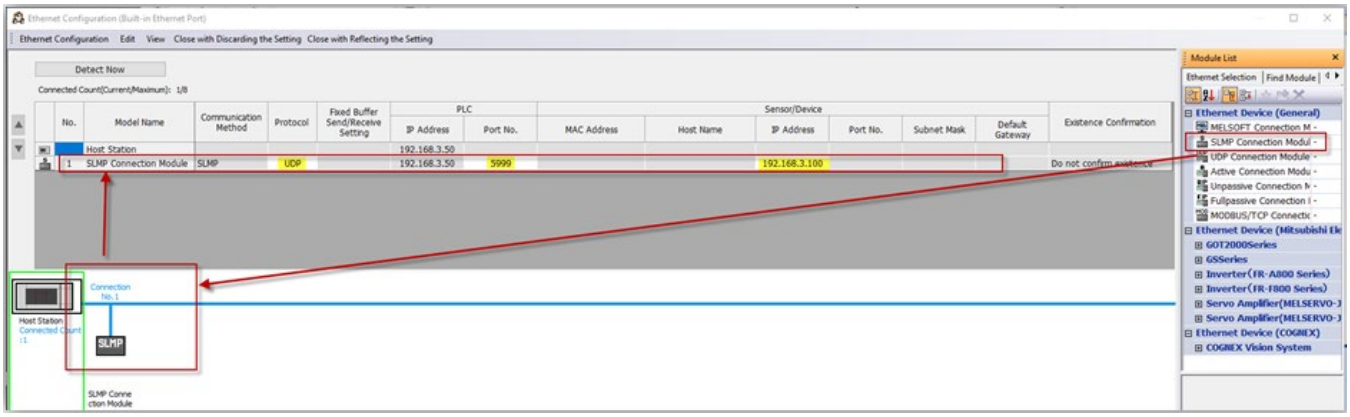
9. Under: Basic Settings -> Own Note Settings enter the IP Address and Subnet Mask for the FX5. This example shows Address 192.168.3.50 and Subnet Mask 255.255.255.0.



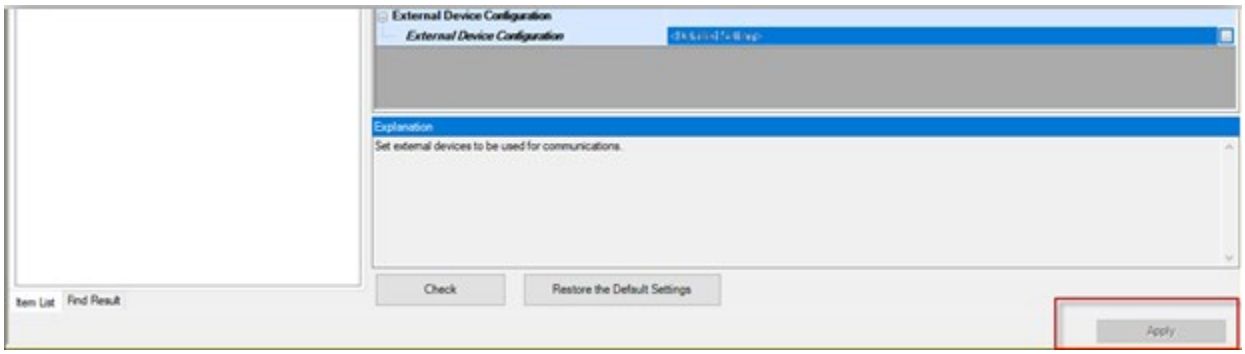
10. Under: Basic Settings -> External Device Configuration set the FX5 PLC as an SLMP server. Double-click on “Detailed Settings” or click on the ellipsis button at the far right.

The new window allows you to add a SLMP connection module.

11. Expand the Ethernet device (general) in the module window.
12. Drag the “SLMP Connection Module” from the list at the right and drop it on the network in the left.
13. Set the Protocol entry to **UDP** and enter a port number to use; in this example **5999**. Enter the IP Address of the **BOA Spot**; in this example 192.168.3.100.

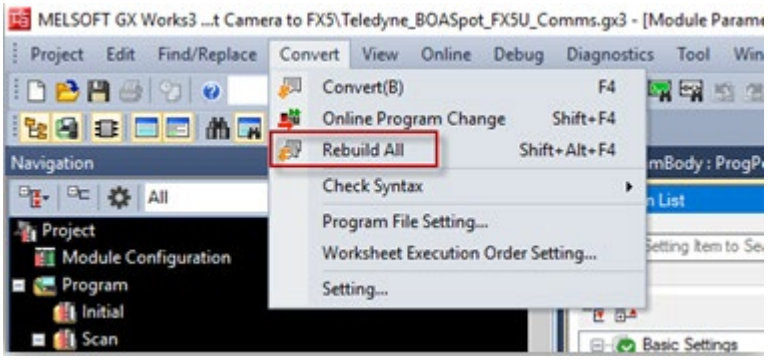


14. Click on **Close with Reflecting the Setting** to apply these changes.
15. Click on **Apply** in the main settings window of the Ethernet port.

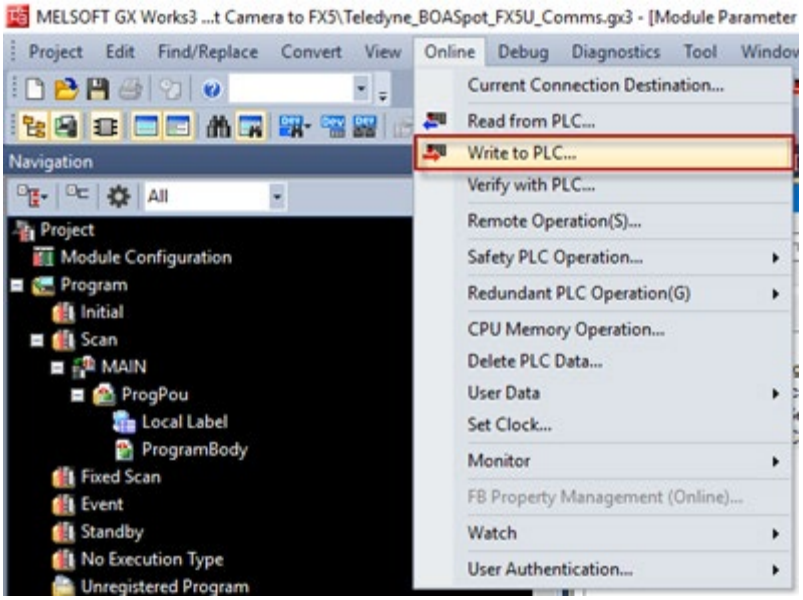


Note: If you forget step 14 or 15 the program will discard your settings.

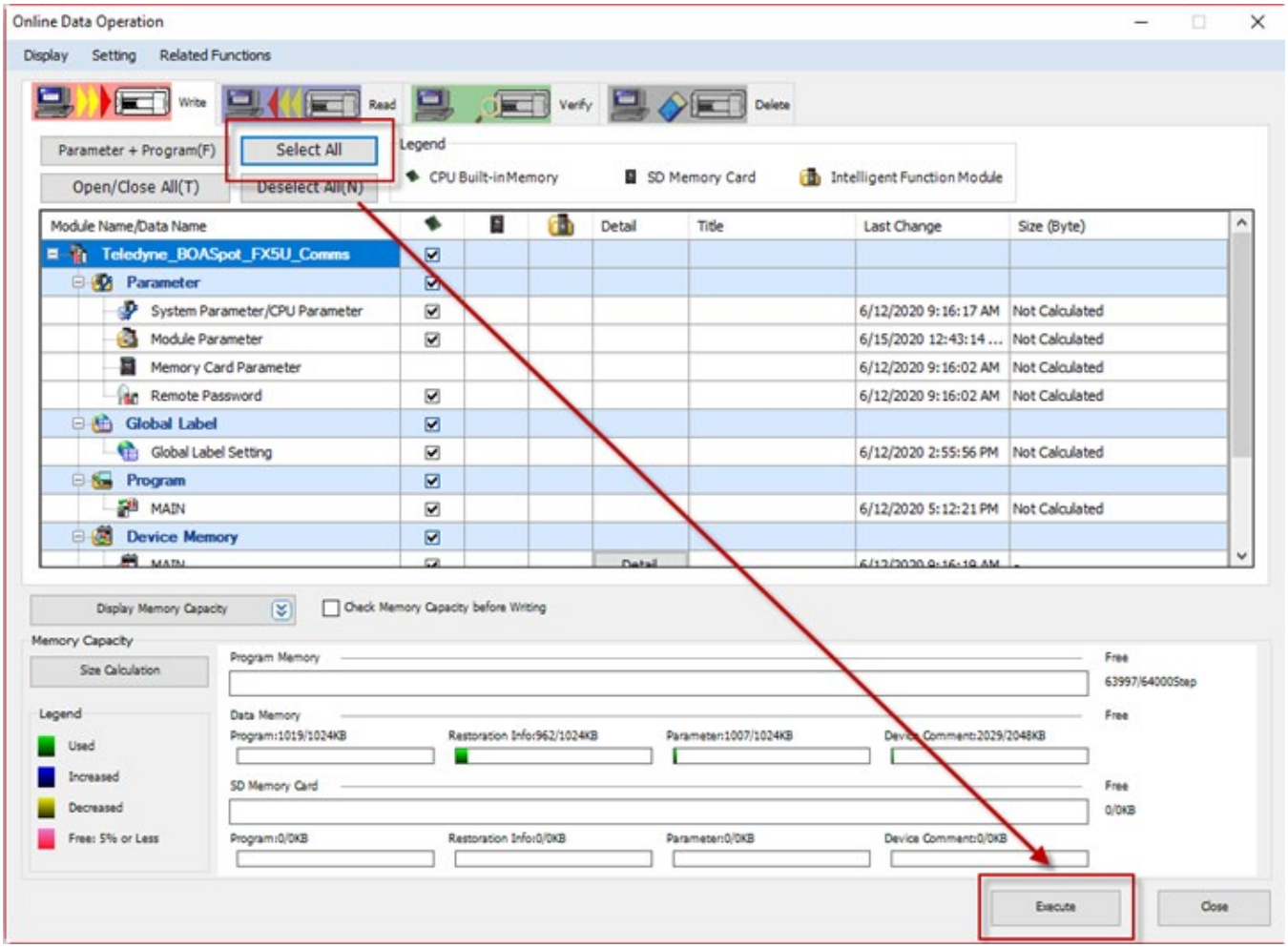
16. In the Main programming window, select **Convert** and **Rebuild All** to prepare the program for the FX5.



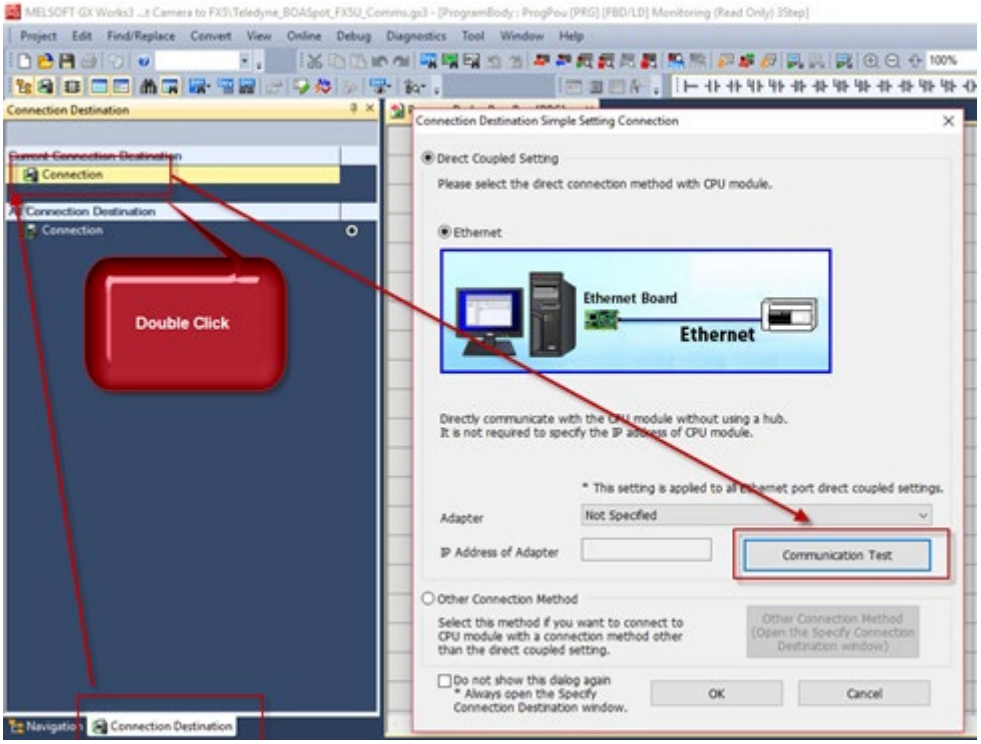
17. Select **Online** and **Write to PLC….**



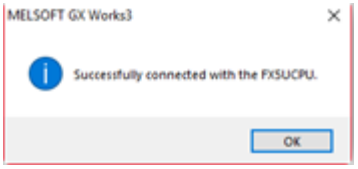
18. Click on **Select All** and then click **Execute**.



19. You may receive an error if you did not set up your connection to the FX5 before writing. Click on the **Connection Destination** tab at the bottom.
20. Double-click on **Connection** to open the connection wizard shown below.
21. Click on **Communication Test** to connect to he FX5.



22. Click OK to close the message window.
23. Click OK again to return to the main window.
24. Repeat steps 16, 17 and 18 to write your program to the FX5.



Configure the BOA Spot

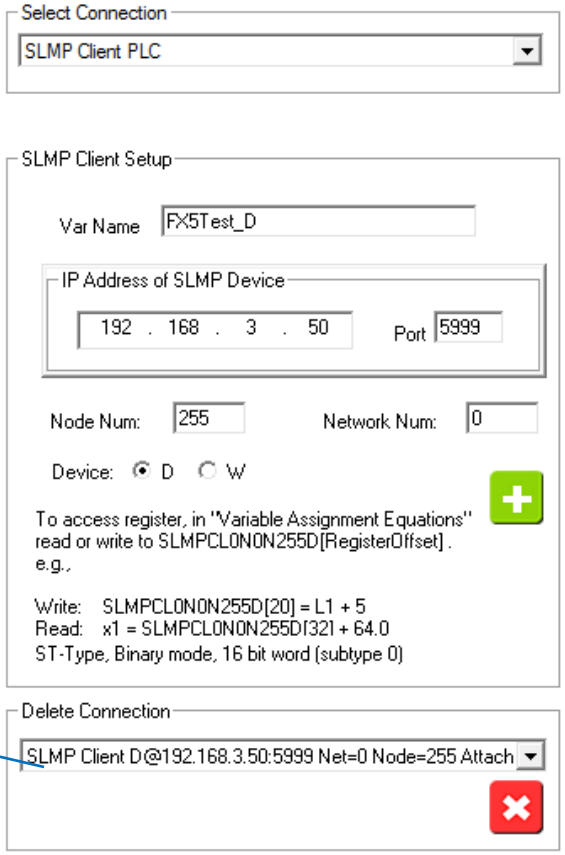
1. Load or create a Solution with one or two tools.
2. In the Navigation bar, click on Setup Connections.

3. Use the “Select Connection” list to select “SLMP Client”.

The center of the Setup panel changes to configure BOA Spot as the SLMP Client.

4. You can change the variable name or use the default. This example uses FX5Test_D.
5. Enter the IP Address of the **PLC** and the port number chosen in the PLC program. In this example, address 192.168.3.50 and port 5999.
6. Leave the other values at their defaults.
7. Click the “Add” button.

Your definition appears in the “Delete Connection” list box.



Notice the Read/Write examples just below the “Add” button. These are the type of statements you would use in the Script Editor to read values from or write values to the PLC. The examples are given using the default variable name “SLMPCL0N0N255D”. The brackets hold an index into the register space.

SLMPCLxxx[20] = L1 + 5 – Translates to: Write the sum of L1 plus 5 to the PLC’s register space at index location 20.

X1 = SLMPCLxxx[32] + 64.0 – Translates to: Read the value at the PLC’s register at index location 32, add 64.0 to the value, and store it in variable X1.

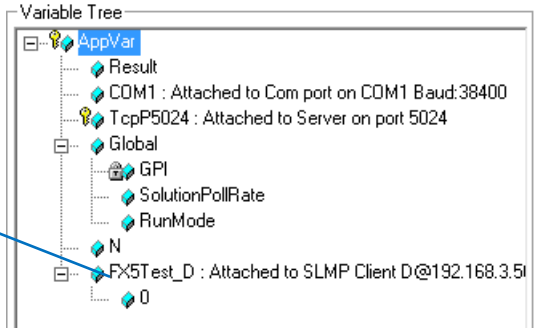
Depending on the design of the PLC, the read and write locations may be the same, or they may be different.

Verify Communication

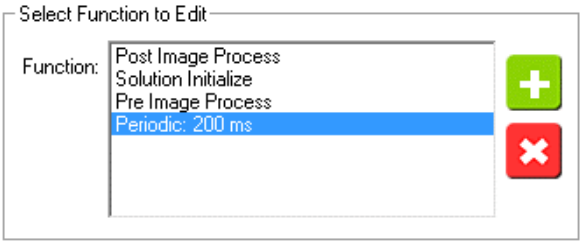
Now add scripts to communicate with the PLC. Tip: Start by writing just a few numbers to the PLC. You can add more instructions and controls after you have successfully established communication with the PLC.

1. In the Navigation bar, click on “Edit Scripts”. 

Our variable FX5Test_D appears in the list in the left panel.



2. At the bottom of the left panel double-click on “Periodic 200 ms” to open the Script Edit window. This is where we create script statements to read and write to the PLC.



The “Post Image Process” function is called after the image tools are run or executed. This is used to send measured results to the PLC or to tell the PLC the current image measurements are completed.

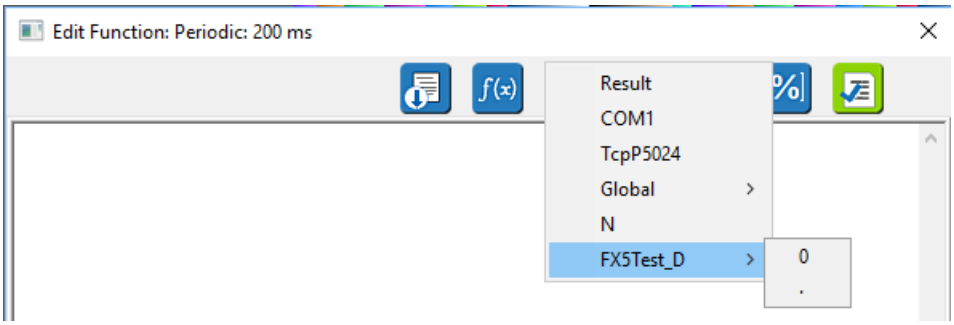
The “Pre Image Process” function is called after an image is acquired and before measurements are made.

The “Periodic: 200 ms” function is called every 200ms. This is a “background” script that runs independent of acquisition and measurement. This is used to monitor for commands or communication from the PLC.

The “Solution Initialize” function is run only once when the Solution is loaded.

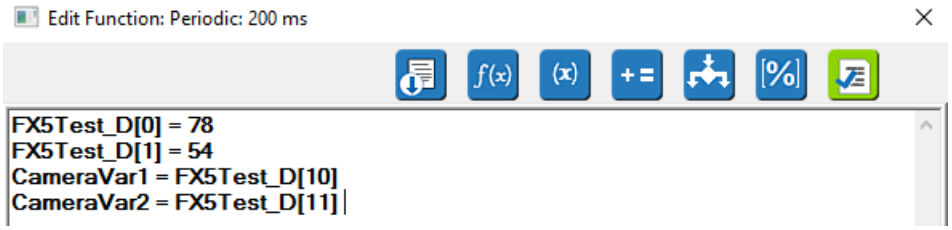
3. Click on the Variable button in the toolbar. 

4. Move your mouse to the variable for our PLC; FX5Test_D and click on 0. Until we save our program, the only PLC variable created here is index 0 or [0]. You must change the [0] for each different register index you want to access in the PLC.



5. Next to FX5Test_D[0] type space, an equal sign, another space, and the number 78. Press Enter. This statement writes the number 78 to the PLC at register index 0,

6. Click on the Variable button again. Continue typing to create the next 3 statements as shown here. Remember to change the number inside the square brackets.



The first statement writes 78 to index 0 in the PLC. The second statements writes 54 to index 1. The third statement reads from register index 10 in the PLC and writes or stores it in a new variable CameraVar1. The fourth statement reads from index 11 and stores it in variable CameraVar2. You create program variables “on the fly” by typing them in statements in this window.

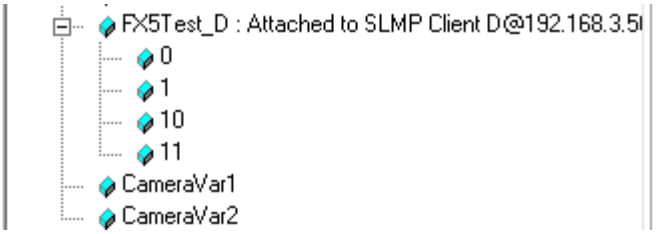
7. Click on the Check Syntax button. 
   Messages will appear in the lower part of this window if your statements have errors.

8. Click the Check button to save your edits and close this window 

Your register indexes are added in the left panel. The variable name is not repeated, only the index numbers appear. The new program variables CameraVar1 and CameraVar2 may not appear at first. Double-click on “Periodic: 200ms” to open the Edit window, and close it again. Or you can run the program.

9. Click the Solutions button in the Navigation bar. 

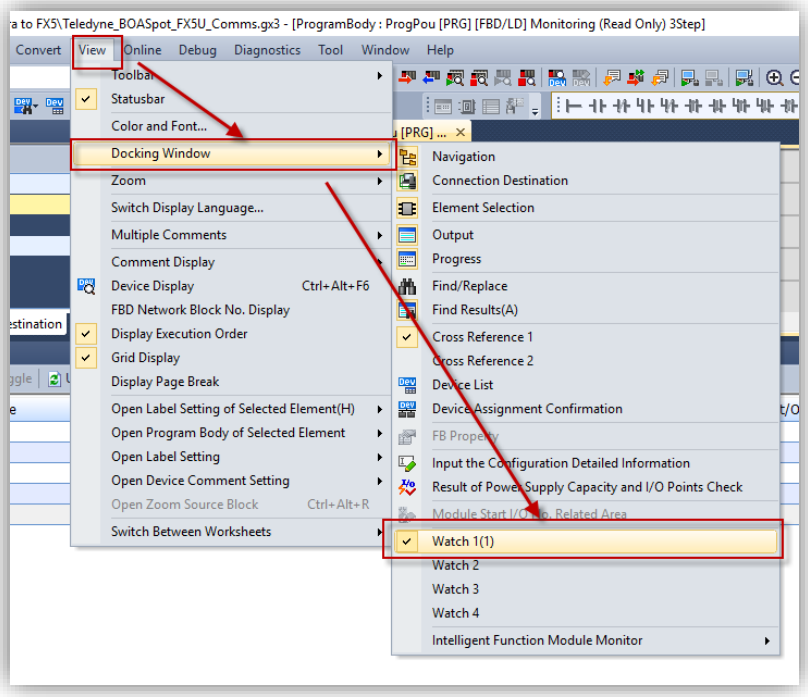
10. Save your Solution and re-load it.

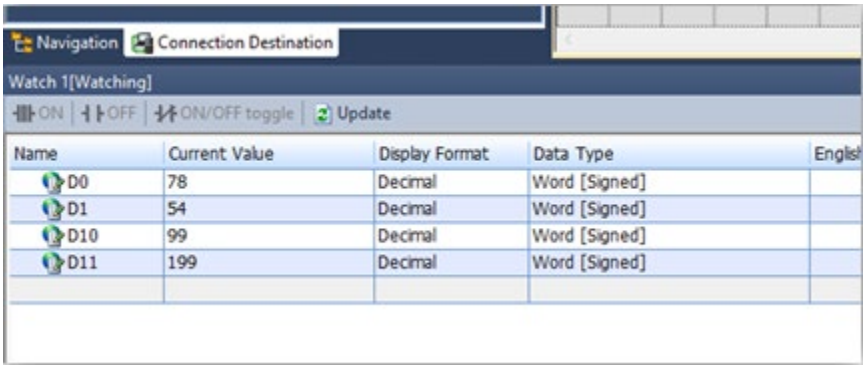


11. Click on Run button in the Navigation bar. 

The two new variables CameraVar1 and CameraVar2 will be 0 if there is no data entered in the PLC registers, or they may have values if the PLC already has stored data in those two locations.

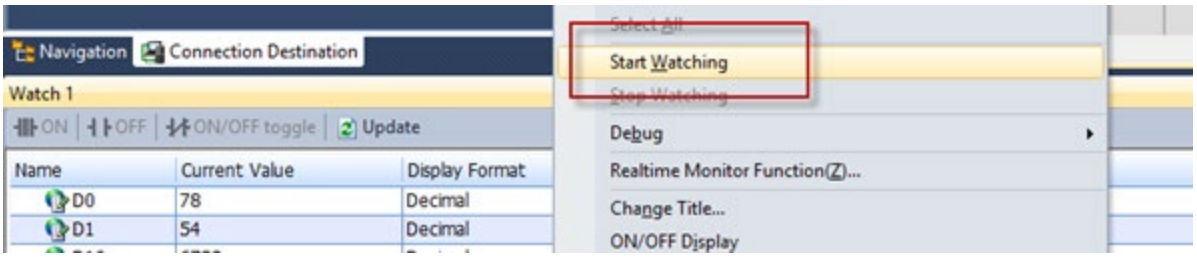
12. Return to the GX Works3 window.
    You need to open a Watch Window and add the devices for monitoring.





The BOA Spot is writing to locations D0 and D1. If you try to modify them, they will be overwritten by the BOA Spot.

Note: If you do not see values being updated in the Watch window, right-click in the window and select “Start Watching”.



13. You can modify the values in D10 and D11. They will appear in the Run panel on the BOA Spot as CameraVal1 and CameraVal2.

Now that you have established and verified communication between the PLC and BOA Spot, you can begin exchanging real measured data and PLC requests.

The next page shows a general example of program statements for communicating with the PLC. This example uses the post image process routine to pass the Pass/Fail result (Result.0), Intensity Tool measured Average intensity value (IntenAvg), and the total number of passed parts (Global.PassCount).

In the **Periodic: 200 ms** function:

```
//Send current solution to PLC
MySolutionNum = GetSolutionID()  //Get current running Solution number
FX5Test_D[3] = MySolutionNum     //Send Solution number to PLC
FX5Test_D[4] = Global.FrameCount  //Send total number of parts tested
// get a solution change request from PLC
solReq = FX5Test_D[11]
if(solReq != MySoutionNum)    //If the new request is a different number
    ChangeSolution(solReq)
endif
// get any trigger request from PLC
trigReq = SLMP[10]
// only trigger on leading edge of the register transition
if((trigReq = 1) AND (trigArmed = 1))
    trigArmed = 0                // prevents multiple triggers
    inspBusy = 1                 // the inspection is running
    trigger()
endif
//re-arm trigger when PLC register is 0
if(trigReq = 0) trigArmed = 1
//note PLC connection status
plcStat = IsConnected(FX5Test_D)
//send heartbeat count – optional Some PLCs need a heartbeat signal
hb = hb + 1
if(hb > 999) hb = 1
FX5Test_D[5] = hb
```

In the **Post Image Process** function:

```
//Send results to the PLC Input registers
FX5Test_D[0] = Result.0          //overall pass or fail result
FX5Test_D[1] = IntenAvg          //Intensity Tool measured value
FX5Test_D[2] = Global.PassCount   //Total Pass parts count
// inspection is done status
inspBusy = 0
```

1. Always check for syntax errors. 
2. Always click “Save” to save changes and close the Edit window. 
3. Always Save your Solution then Reload and Run the Solution. (See Important on page 6.)