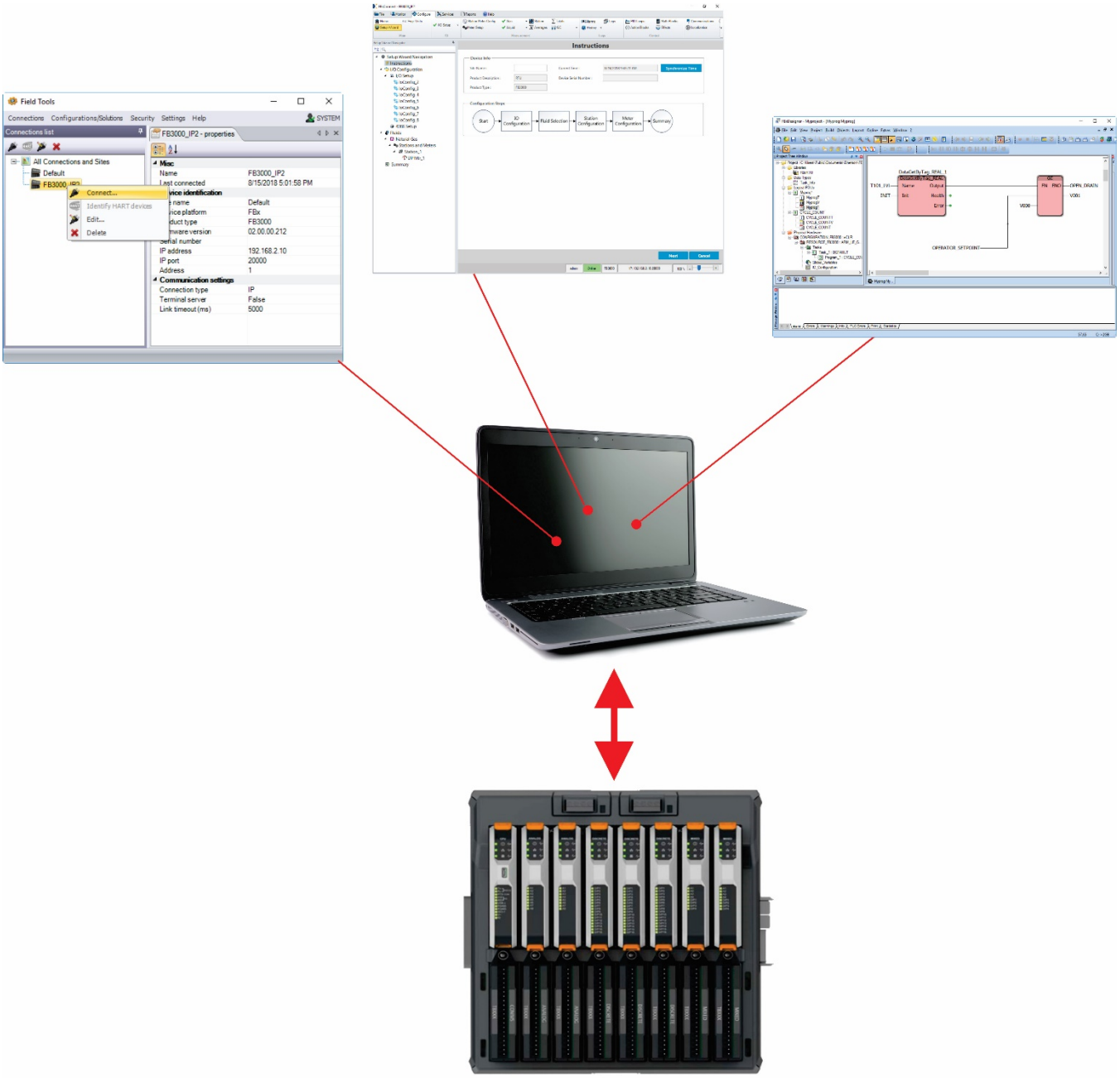
FBxDesigner™ Quick Start Guide



Energy and Transportation Solutions

EMERSON™

Application Safety Considerations

Protecting Operating Processes

A failure of this application – for whatever reason -- may leave an operating process without appropriate protection and could result in possible damage to property or injury to persons. To protect against this, you should review the need for additional backup equipment or provide alternate means of protection (such as alarm devices, output limiting, fail-safe valves, relief valves, emergency shutoffs, emergency switches, etc.).

System Training

A well-trained workforce is critical to the success of your operation. Knowing how to correctly install, configure, program, calibrate, and trouble-shoot your Emerson equipment provides your engineers and technicians with the skills and confidence to optimize your investment. Energy and Transportation Solutions offers a variety of ways for your personnel to acquire essential system expertise. Our full-time professional instructors can conduct classroom training at several of our corporate offices, at your site, or even at your regional Emerson office. You can also receive the same quality training via our live, interactive Emerson Virtual Classroom and save on travel costs. For our complete schedule and further information, contact the Energy and Transportation Solutions Training Department at 800-338-8158 or email us at *education@emerson.com*.

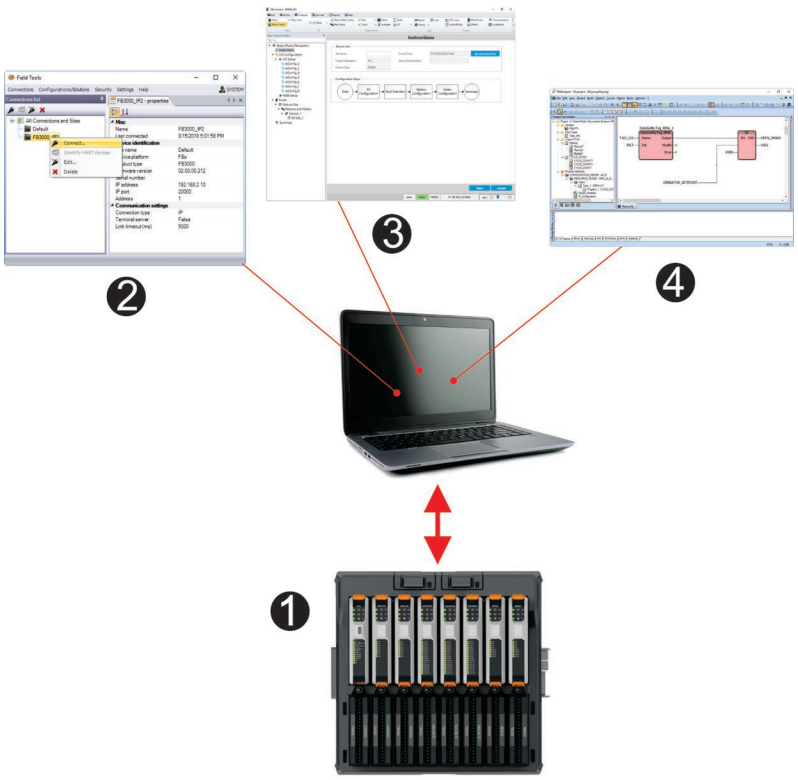
Contents

Introduction

This guide outlines the basic steps to create a simple FBxDesigner™ application for your FB3000 RTU, and also explains some relevant terminology used within FBxDesigner.

Field Tools, FBxConnect™, and FBxDesigner™

There are three PC-based software components used to configure an FB3000 RTU. They are represented as items 2, 3, and 4 in the figure, below.



1 FB3000 RTU

2 Field Tools

3 FBxConnect

4 FBxDesigner

Field Tools

The **Field Tools** software establishes communications with the RTU.

The Field Tools software suite also includes FBxConnect and FBxDesigner, and you must select them for installation when you install Field Tools.

Once Field Tools starts a communication connection with the FB3000 RTU, it automatically launches FBxConnect configuration software, so you can configure and monitor the RTU.

For each RTU you connect to, Field Tools creates a set of configuration folders at the PC for your activities in FBxConnect and FBxDesigner. This hierarchy of folders is called a **solution** and ultimately holds everything you create in FBxConnect, portions of what you create in FBxDesigner, plus all the communication parameters for your connection to this particular RTU.

Depending upon how many RTUs you have, you might have several different solutions on your PC, each with its own set of folders and files.

For full details on Field Tools, refer to the *Field Tools Quick Start Guide* (D301703X412) and the online help within the software.

FBxConnect

Once you establish communications through Field Tools, **FBxConnect** launches to allow you to configure the device. If you do not have an active communication connection, FBxConnect can also operate offline.

FBxConnect includes a series of standard menus which allow you to configure I/O, define meters and stations, set up historical data storage and alarming, and other common tasks for the RTU. FBxConnect also supports a set of action, math, and effect blocks to allow you to perform common logic operations. Whether used offline or online, the menu entries you make in FBxConnect are stored as the standard **configuration** for the RTU.

In addition, FBxConnect includes an integrated display builder (FBxVue™) to create customized graphical displays an operator can view within FBxConnect to monitor and control the RTU.

FBxConnect stores both the standard configuration and any custom displays as XML files in folders within the **solution** folder hierarchy.

For full details on FBxConnect, refer to the *FBxConnect Configuration Software User Manual* (D301850X012) and the online help within the software.

FBxDesigner

For programming and control operations beyond what is offered in FBxConnect, you use **FBxDesigner** software.

FBxDesigner is a licensed option in Field Tools which supports custom programming in accordance with the IEC61131 standard. FBxDesigner supports all five IEC61131 languages (FBD, ST, LD, SFC, and IL) by incorporating the industry-standard MULTIPROG IEC 61131 programming tool.

FBxDesigner also includes a library of pre-defined function blocks tailored specifically to referencing data in the FB3000 RTU. For information on how the various function blocks work, refer to the online help in the software.

Configurations, Projects, Applications, and Solutions

Configurations

As mentioned, before, all the choices you make in the standard FBxConnect menus are referred to as the standard **configuration** for the RTU. The configuration files are stored in a folder within the **solution** folder hierarchy for the RTU.

Projects

The programs you create and edit in FBxDesigner are stored as a **project**. Although the project consists of an .MWT file and associated sub-folders, we strongly recommend you save the project as a zipped file with an extension of *.ZWT; the ZWT includes all the source and related folders for the project.

When you finish creating your project, you must generate a machine-readable version of it, called a **boot project**, since that is the actual file that the FB3000 RTU executes.

The FBxDesigner project is a required component of any application.

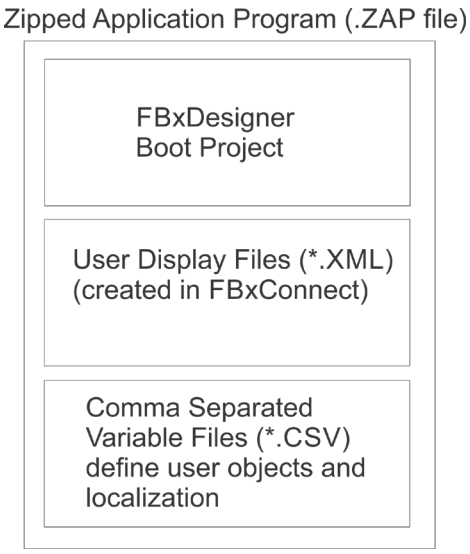
Applications

An application consists of up to three components.

FBxDesigner Project (REQUIRED)	A custom program (or programs) created in FBxDesigner that handle logic not available in the standard configuration for the RTU.
User-Defined Objects (OPTIONAL)	These are extensions to the RTU's database which define a particular object. You might, for example, create an object called a pump, or a compressor. Comma separated variable (CSV) files define the parameters for these objects. Typically, these objects are created in a text editor by a third-party integrator such as an Emerson Impact Partner to mimic field devices at a customer site. You also use CSV files for localization allowing you to change the language used on screens.
Displays (OPTIONAL)	You can use FBxVue to create displays for an operator to interact with the project.

You can combine the FBxDesigner project with any custom displays you create in FBxConnect and any user-defined objects to generate an **application**. Although display creation and user-defined objects are beyond the scope of this guide, their inclusion in an application can allow the application to be integrated seamlessly into FBxConnect, alongside the standard menus and functions.

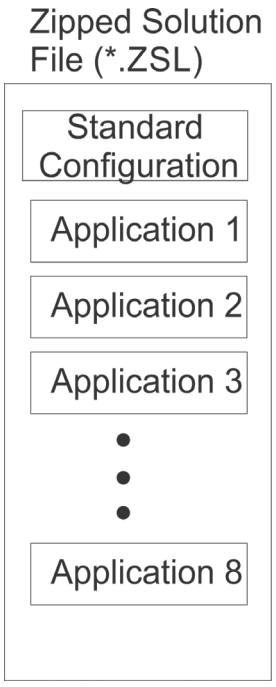
The various components of an application can be exported as a single *.ZAP file (application package) and can be downloaded into any FB3000 RTU. An application package can be loaded by field technicians from FBxConnect, without the use of FBxDesigner programming software.



Solutions

The solution folders contain the standard configuration you define in FBxConnect, plus any applications you create which include an FBxDesigner project, and optionally displays and CSV files for custom objects and localization.

The entire set of folders for the solution can be compressed into a single zipped solution file (*.ZSL) which you can use to distribute the same identical solution to multiple RTUs.



Overview of Steps

1. Install the Field Tools software suite including FBxConnect and FBxDesigner.
2. License FBxDesigner.
3. Create an FBxDesigner Project (including a Boot Project).
4. Create Displays (OPTIONAL – beyond the scope of this document).
5. Create User Defined Objects (OPTIONAL – beyond the scope of this document).
6. Create and Download an Application.

Installing Field Tools Software including FBxDesigner

You install Field Tools 3.x software (which launches FBxConnect) on your PC to configure the device.

Important

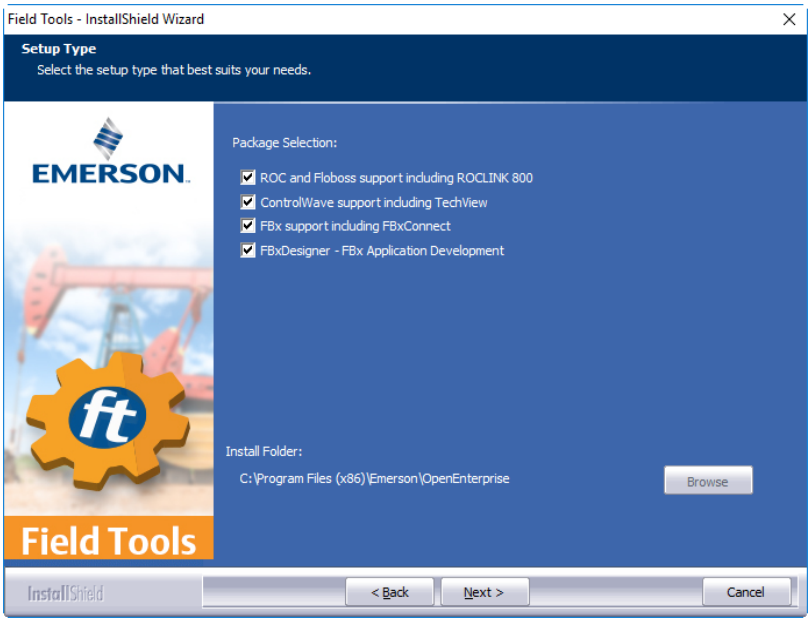
- Field Tools (including FBxConnect) is available as a free software download to registered Guardian users. If you are not a registered Guardian user, new accounts take up to 24 hours to process so plan accordingly.
- Although you install it as part of the Field Tools installation, you must purchase a software license for FBxDesigner.
- If installing TechView, close all other programs down before you begin installation. In particular Office 365 components must be closed because they can interfere with the Field Tools installer.
- Both FBxDesigner and ControlWave Designer can co-exist on the same PC, however, you can only run one or the other at any given time. Do not use ControlWave Designer with an FB3000 device or FBxDesigner with a ControlWave device; they are incompatible with the alternate platform.
- Field Tools **cannot** reside on a computer running any component of OpenEnterprise 2.x, OpenEnterprise 3.x, OpenEnterprise Client/Server, or ObjectServer software.
- TechView and other components of BSI_Config software **cannot** be installed on a computer running OpenBSI Network Edition versions older than 5.9 Service Pack 2.
- You must have administrative privileges to install Field Tools.
- You must disable User Account Control (UAC) prior to installing Field Tools (you can re-enable it after a successful installation).
- As part of the installation, software from Eltima is automatically installed. Depending upon your permissions, Windows may require you to confirm this installation before the installation can proceed.

1. Right click on the installer file, and choose **Run as administrator** from the pop-up menu.

2. Follow the on-screen instructions. For details on minimum hardware/software requirements as well as more details on the installation steps see the *Field Tools Quick Start Guide* (D301703X412).

Note

During Field Tools installation, you must select the **FBx support including FBxConnect** option and **FBxDesigner -FBx Application Development**.



3. After software installation re-boot, start Field Tools from either the Windows Start Programs menu or (if you created it) from the desktop icon.

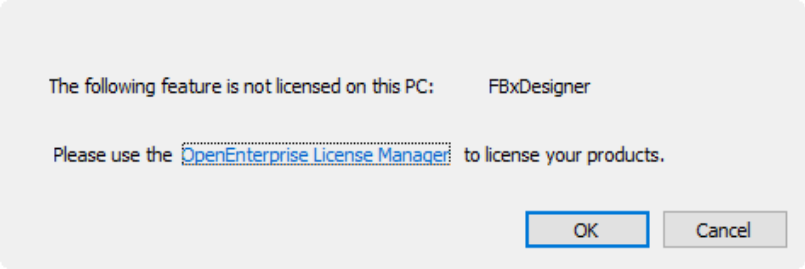
4. Log onto Field Tools.

Important

The very first time you login with Field Tools, use the default **username** of **admin** and leave the **password** field blank. Then assign a new password when prompted. See the *Field Tools Quick Start Guide* for any questions you have on changing default passwords after installation.

Licensing FBxDesigner

If you start FBxDesigner and receive the message below, you must license FBxDesigner.

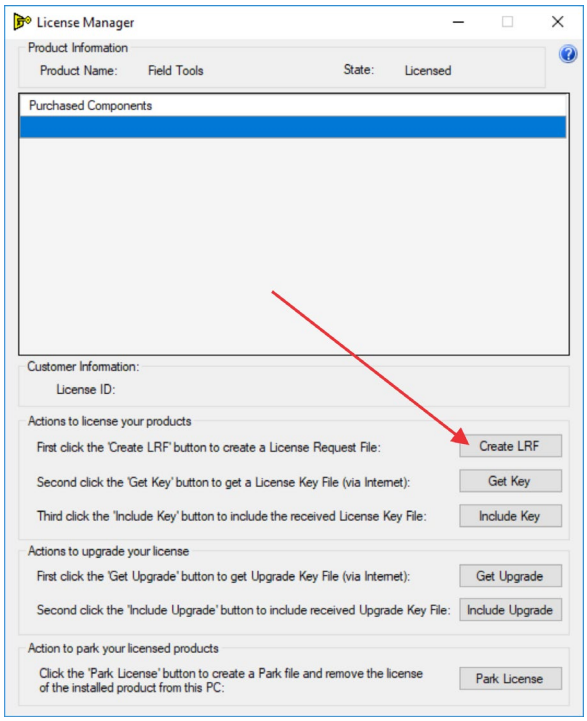


Note:

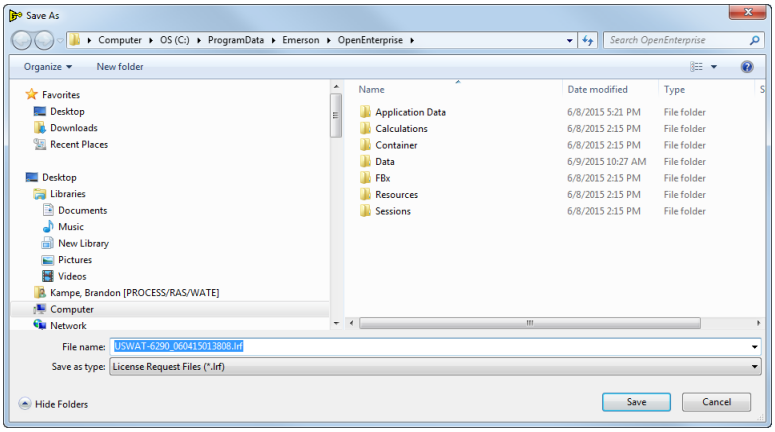
If launching the Licensing Manager through a newer browser such as Edge, you must run the browser in **IE mode**.

Starting License Manager and Licensing FBxDesigner

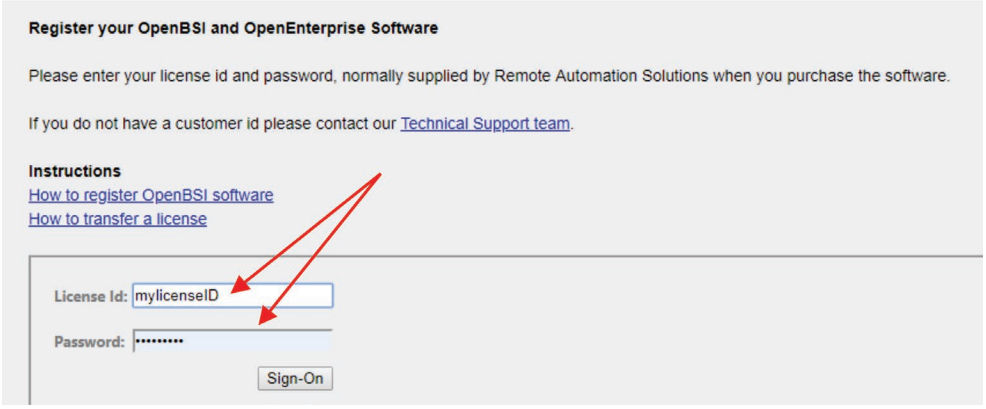
1. From the menu bar, in Field Tools, Click **Help > Licensing > Field Tools Licensing**. The License Manager screen opens.



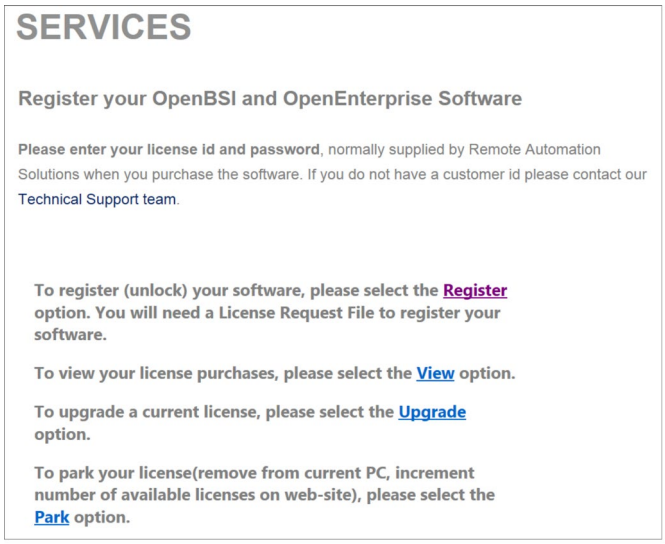
2. Click **Create LRF** to generate a License Request File. A message box notifies you when the LRF file is complete. Click **OK** to proceed.
3. Click **Save** and make note of the location where you store the file. You will need it later.



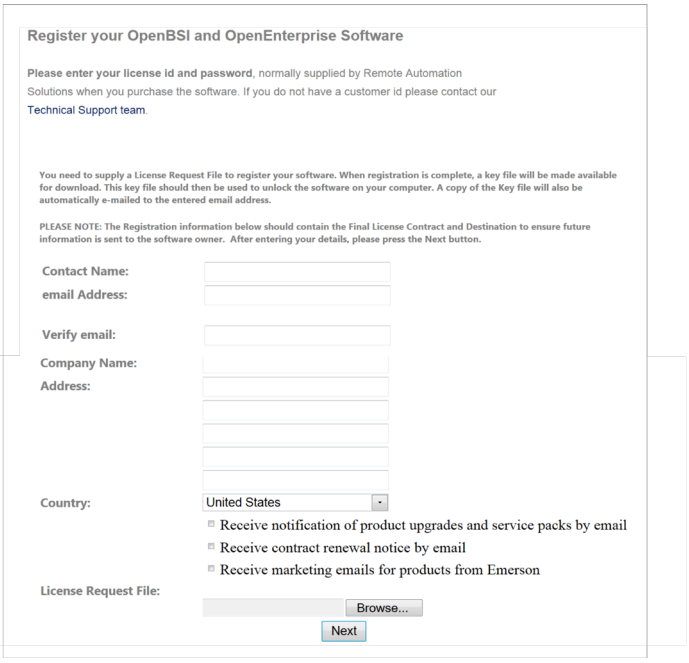
4. Now click **Get Key** in the License Manager to go to the software registration page of the Emerson Remote Automation Solutions website.
5. Sign onto the licensing website using the **License id** and **password** you received when you purchased FBxDesigner.



6. Click the **Register** option.

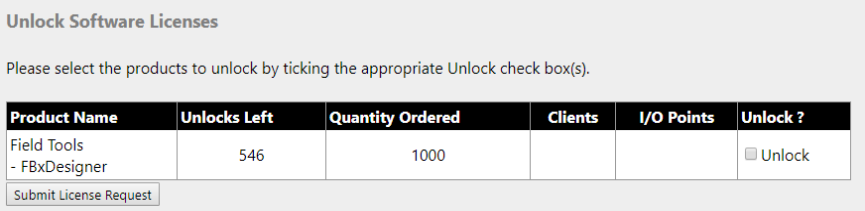


7. Enter your name in the **Your Name** field; and enter your e-mail address in both the **E-Mail Address** and **Verify E-Mail** fields. This is the address to which the website sends your key file. Enter your **Company Address** in the fields provided.
8. Scroll down to specify your **Country** and specify your preferences about receiving notifications of product updates, service packs, contract renewals, and marketing announcements by e-mail.

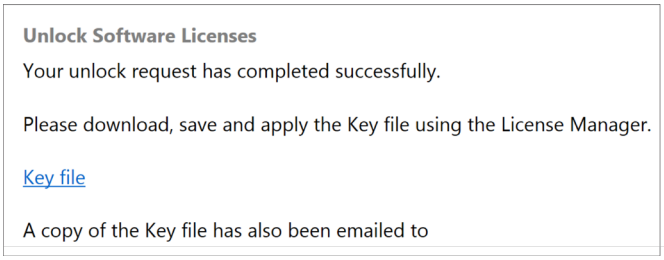


9. Use the **Browse** button to locate the **License Request File** you saved in Step 3 and then click **Next** to open the Unlock page.

10. The Unlock Software Licenses page shows a list of licenses purchased under your License Id. If the **Unlocks Left** entry for FBxDesigner is greater than zero, you can request one of the available licenses. Select **Unlock** for the FBxDesigner line and then click **Submit License Request**.



11. At this point, the website e-mails your key file to the address you specified in Step 7 along with a separate notification e-mail. You can check your e-mail for a copy of the key file and save it for the next step. Alternatively, you can click Key file to open the key file in a window; click **File** > **Save As** to save your key file.



Important

- If you use Microsoft® Internet Explorer 9, it automatically saves your key file with a .TXT extension. License Manager handles the .TXT extension; do not change the extension or the file may become unusable.
- If you right-click on the Key File link and select the **Save Target as** context menu item, the key file is saved with an .XML extension.
- The e-mailed key file has an extension of .KEY.
- License Manager handles .KEY, .TXT, and .XML extensions.

12. Click **Include Key** in the License Manager. Browse to the location of your key file and click **Open** to apply the key file. FBxDesigner is now licensed. You're done!

Re-assigning an FBxDesigner License to another PC (Parking a License)

If you license FBxDesigner on a particular PC and then decide you want to re-assign the FBxDesigner license to a different PC, you can remove the license from the first PC and

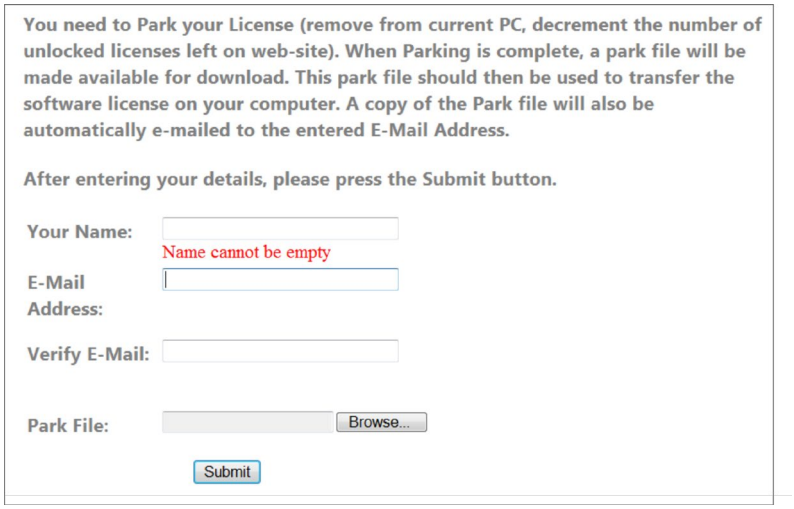
then temporarily “park” that license on the License Registration website. This restores the license to your total number of purchased licenses, and you can then assign it to the new PC through the normal license registration procedure.

Important

Once you park a license (which removes it from the original PC) **you cannot assign a new license to that same PC without first contacting our Technical Support personnel** for codes to restore the demo period for that PC. The technical support phone number in the U.S. is: 1-800-537-9313; for international numbers use this link: https://www.emerson.com/en-us/automation/guardian/technical-support-contact-information/ Alternately, log into Guardian at this link: https://guardian.emerson.com/login

1. To start the License Manager, click **Help > Licensing > Field Tools Licensing** from the menu bar in Field Tools.
2. Click **Park License** and save the PRK file. Make note of the location because you need to access the file in a later step.
3. Click **Get Key** to go to the License Registration website.
4. Enter your **License Id** and **Password** and then click **Sign-On**.
5. Click the **Park** option; this removes the license from the current PC.

Parking a License



6. Enter your name in the **Your Name** field; and enter your e-mail address in both the **E-Mail Address** and **Verify E-Mail** fields. This is the address to which the licensing website sends your key file.
7. Use the **Browse** button to locate the **Park File** you created in Step 2.
8. Click **Submit**. When the website accepts the park file, it shows the message **Park File Operation Completed Successfully**.
9. Exit the License Manager. You can now re-assign the license to a different PC by following the licensing procedure on the new PC.

Setting Up I/O in FBxConnect

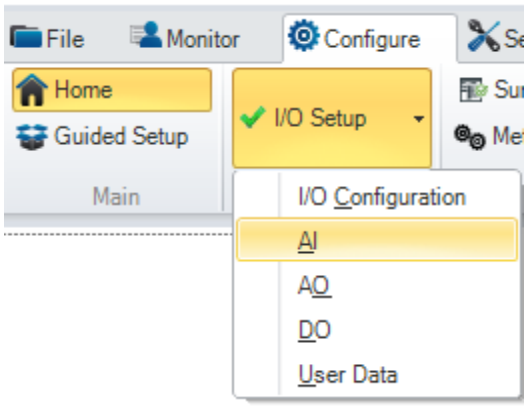
Note

For purposes of this guide, we assume you have already used Field Tools to establish communications with the FB3000 which then launches FBxConnect.

For full details on I/O configuration in FBxConnect, consult the FBxConnect online help.

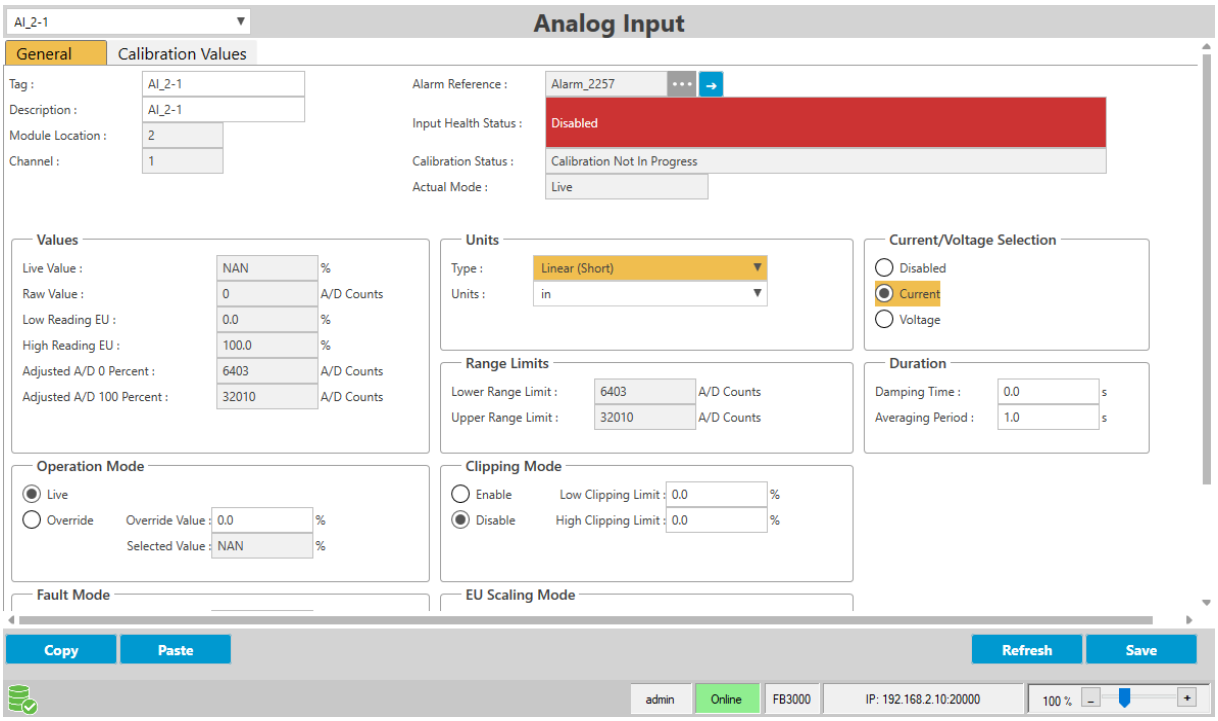
Now, we are going to configure some I/O points that we will use later. Our device has a single mixed I/O module in Slot 2 (IOConfig_2). **Note:** There can be no I/O module in Slot 1 because that's where the CPU module resides.

1. Click the **Configure** tab, then click **IO Setup > AI**.



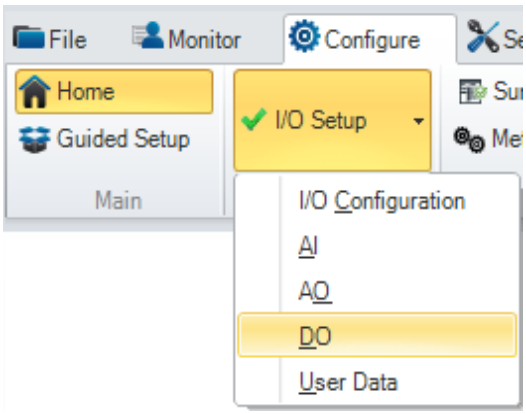
2. We are going to configure analog input #1 on the module (default tag of AI_2-1). It will represent the level of water in a tank. Refer to the screen shot, below, and change the following fields, as shown.

   - Choose **Linear (short)** for the Unit's **Type**.
   - Choose **Current** for the **Current/Voltage S**election.

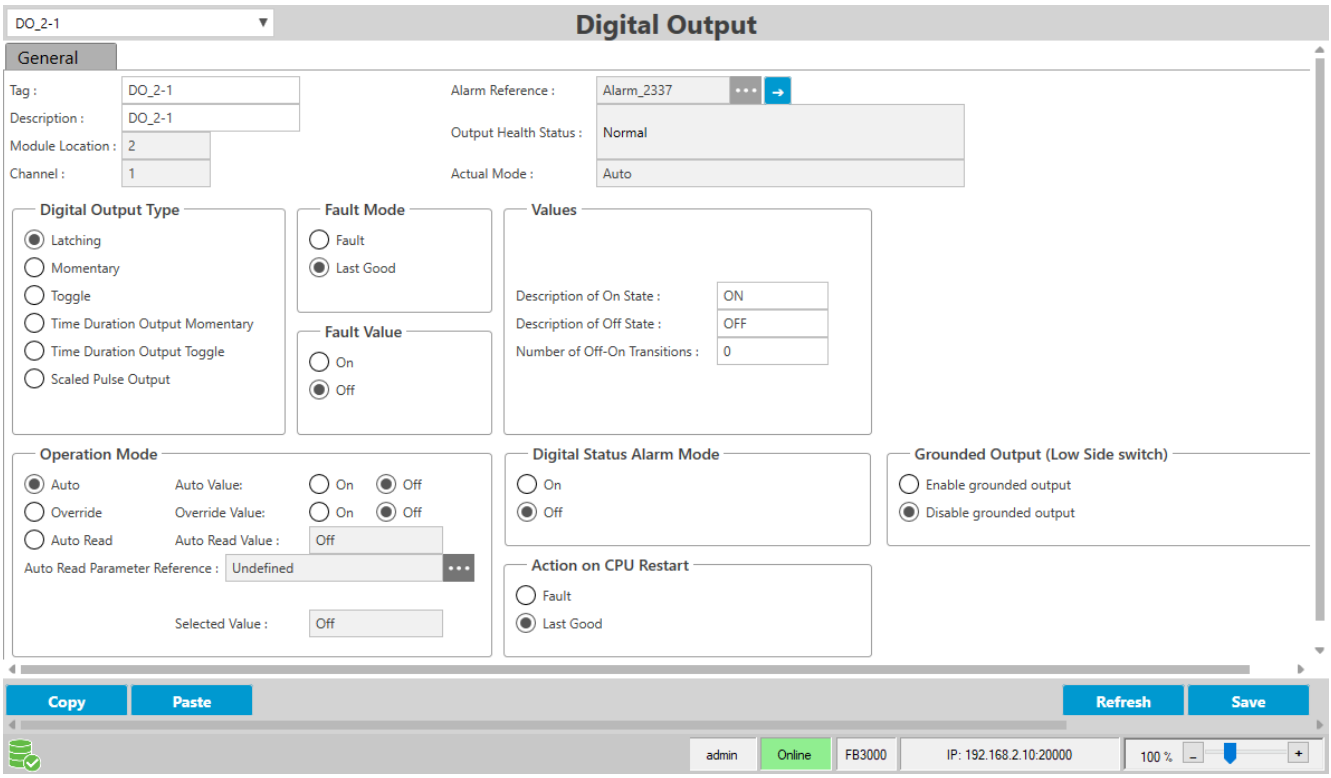


3. Click **Save**.

4. Click **IO Setup > DO**.



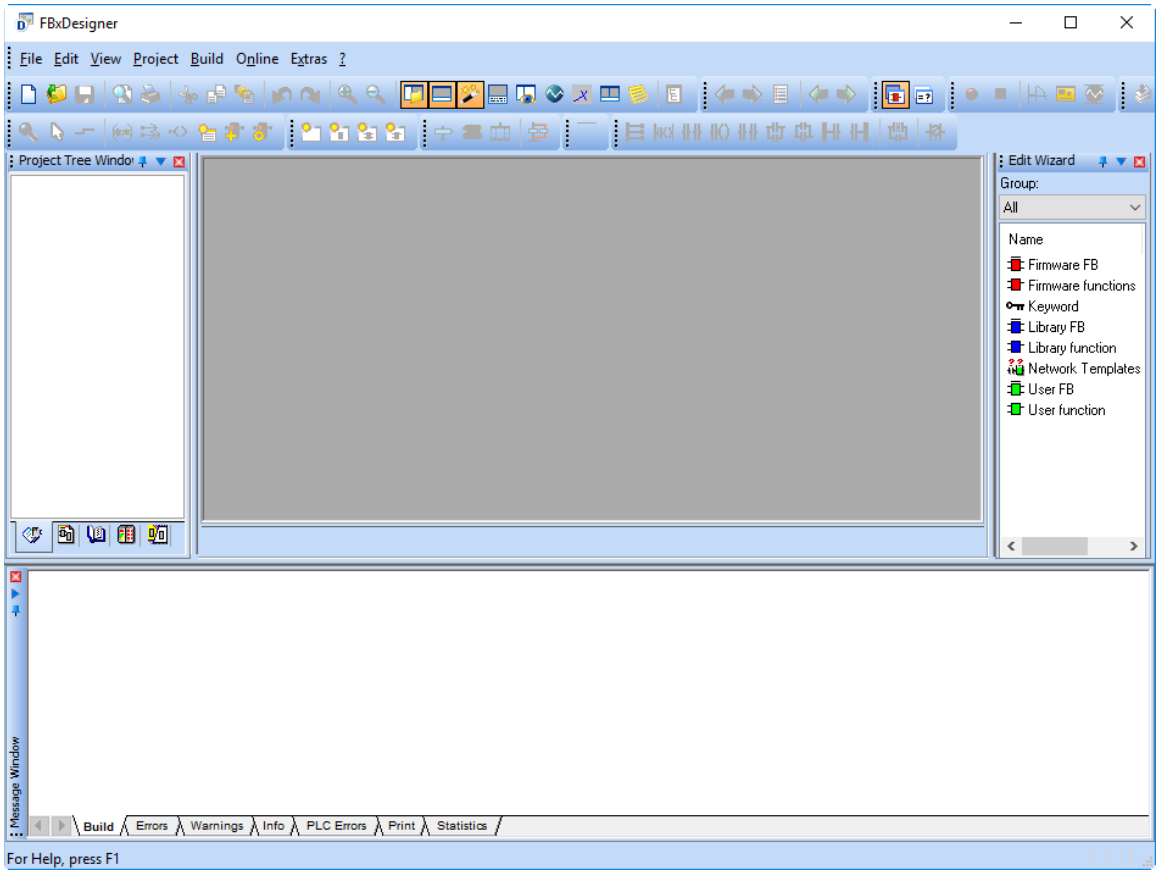
5. We are also going to use digital output #1 on the module (default tag of DO_2-1). When turned ON, it's intended to open a drain.



- Click **Save** to save the I/O configuration.

Starting FBxDesigner

To start FBxDesigner, click **Start > Emerson Field Tools > FBxDesigner**.



Note

If FBxDesigner immediately shuts down after you start it, you probably never licensed your copy of FBxDesigner. Please see *Licensing FBxDesigner*.

Note

You cannot start FBxDesigner by clicking on MWT.EXE in Windows Explorer unless you already have administrative privileges.

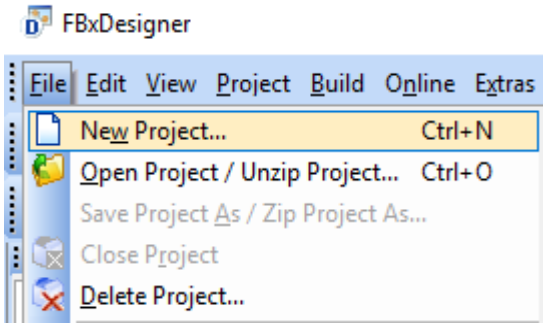
Creating a Simple FBxDesigner Project

In this example, we're going to create a simple project using FBD (Function Block Diagram) which is one of the IEC 61131 languages. This project compares an analog value to a setpoint, and based on the value, opens or closes a valve.

Note: This example requires you to look at variables used in FBxConnect through the Tag Browser, so you must establish an FBxConnect session with your device – either online or offline **before you begin.**

Creating a New Project

Click **File > New Project (**or click the New Project icon).



Choose **Template for FB3000** in the New Project dialog box, then click **OK**.**Note**
For this example, we are using the Template for FB3000. If you plan to create a very large application, you can select **Template for FB3000 Extended Memory**. Applications using this template can only be installed in application slot 1 of the FB3000; they cannot be used in application slots 2 through 8.

FBxDesigner opens the Project Tree Window. The following table talks about some of the more important items in the Project Tree Window.

Important Items in the Project Tree

The project tree includes a series of folders and worksheets which define the various elements of your project. This table discusses some of the most important elements of the project tree.

Logical POUs Program Organization Units (POUs) hold the program code for your project. They are defined on three separate worksheets. One worksheet can optionally contain a textual description of the POU, a second worksheet lists the variables associated with the POU, and a third worksheet defines the POU code. POUs can be up to 640 K bytes in size, and there is an enforced limit of 512 POUs per project. There are three types of POUs:

Functions - Sometimes the term function is abbreviated as 'FU'. These are the simplest type of POU. They can take multiple inputs, but they generate only one output. The memory allocated for a function's data is not persistent; i.e. once the function completes execution, that memory is released. Functions are generally used to perform mathematical operations; an example of a function is R_INT which truncates a real number to an integer. Functions can only call other functions. Functions can only be executed when included as part of a program POU.

Function Blocks - Sometimes the term function block is abbreviated as 'FB'. Function blocks can take multiple inputs, and they can generate multiple outputs. Unlike functions, they can have persistent memory. User-defined function blocks can be created which consist of logical connections of various functions and function blocks. Function blocks can only be executed when included as part of a program POU.

Programs – Programs are essentially groups of functions and function blocks which have been logically connected together to perform some task. Programs are the only type of POU which can actually be executed. Users can create more than one program in a project, and in fact, they can create multiple instances of the same program. Programs can contain persistent memory. The project tree includes a simple program called CYCLE_COUNT which just adds 1 to a total every time it executes.

Libraries There are two types of libraries: Firmware libraries are libraries of functions and function blocks created specifically for this controller model. The FBxRTU library is a firmware library containing functions and function blocks created specifically for the FB3000-series of controllers. It is automatically loaded when you choose the FB3000 Template in the New Project dialog box. User libraries are optionally created by the user. They are libraries of programs, functions, and function blocks called in from *other* previously saved projects.

Data Types The Data Type worksheets can be used to define customized data type structures. For example, arrays of numbers. NOTE: The data entered here only defines the data type, it doesn't actually set aside memory for storing the data - - that occurs in a variable declaration.

Important Items in the Project Tree (continued)

For reference, the right-most column of the table, below, shows the standard datatypes supported in FBxDesigner that can be part of custom datatypes you might create, and corresponding data types used in the RTU's database.

IEC61131 Data Type	Description	Native Database Data Type
SINT	Short Integer	INT8
INT	Integer	INT16
DINT	Double Integer	INT32
LINT	Long Long Integer	INT64
USINT	Unsigned Short Integer	UINT8, BIN8, BOOL
UINT	Unsigned Integer	UINT16, BIN16
UDINT	Unsigned Double Integer	UINT32, BIN32, TIME
REAL	Real Numbers	FLOAT
LREAL	Long Real Numbers	DOUBLE
STRING	String	UC10, UC20, UC30, UC40

Physical Hardware Physical hardware defines details of the actual controller which will execute the project. It is divided into several sections:

CONFIGURATION is the type of code generation required for the controller. For FB3000, it is always FB3000 eCLR.

RESOURCE defines the type of run-time system used – in this case, it is always "FB3000" with the "ARM_LE_GCC3" processor.

Tasks are the actual mechanism by which programs are executed. When you have completely defined a program, you must associate a program instance with an executing task, and define the rate of execution.

Global_Variables is where any user-created global variables are defined.

IO_Configuration defines drivers used to communicate with the FB3000 database.

Additional worksheets are added to the project tree as you build your project. Some of these are

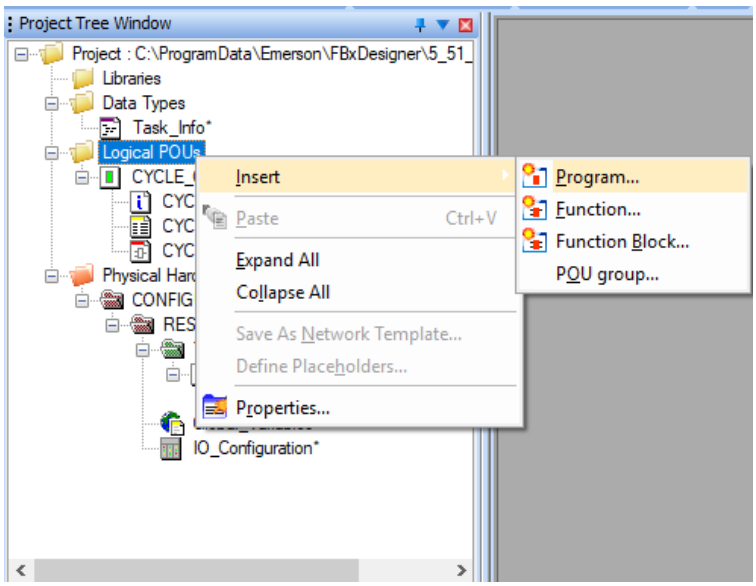
Important Items in the Project Tree (continued)

Additional worksheets are added to the project tree as you build your project. Some of these are added automatically, others must be manually added by the user.

Do not rename items in the project tree created automatically by FBxDesigner. FBxDesigner looks for worksheets under specific items of the tree, and if you rename those items, FBxDesigner will be unable to locate those worksheets.

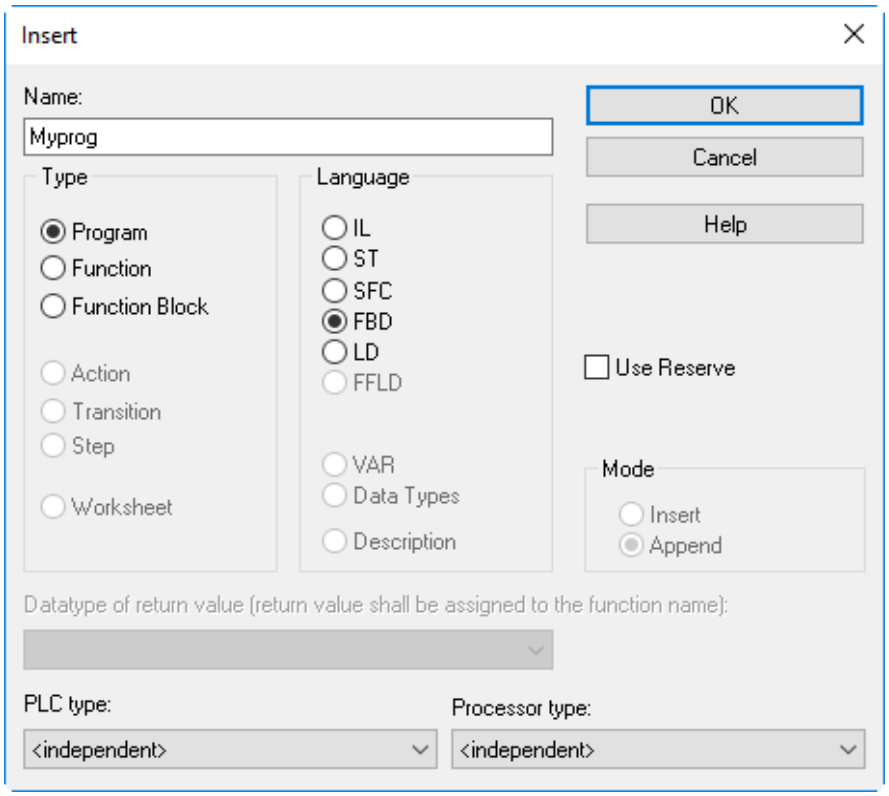
Creating a Program POU For Your Project

Right click on **Logical POUs** and choose **Insert > Program**.

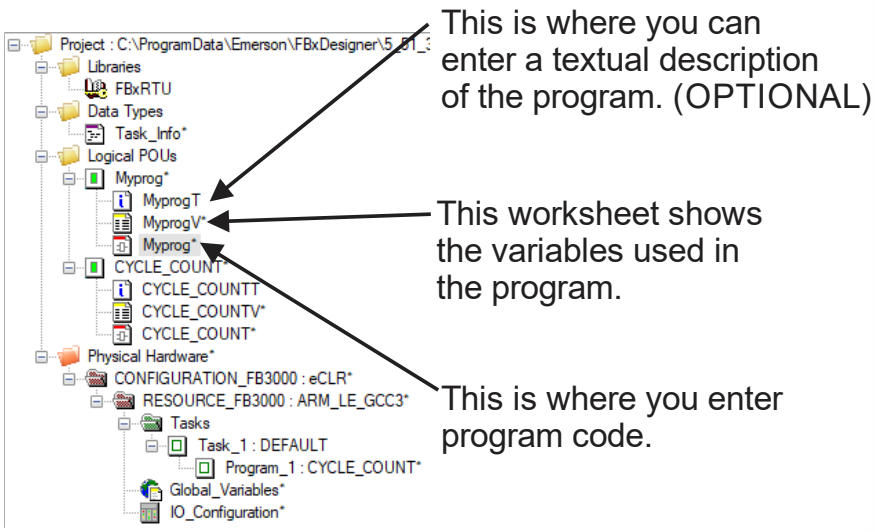


In the Insert dialog box:

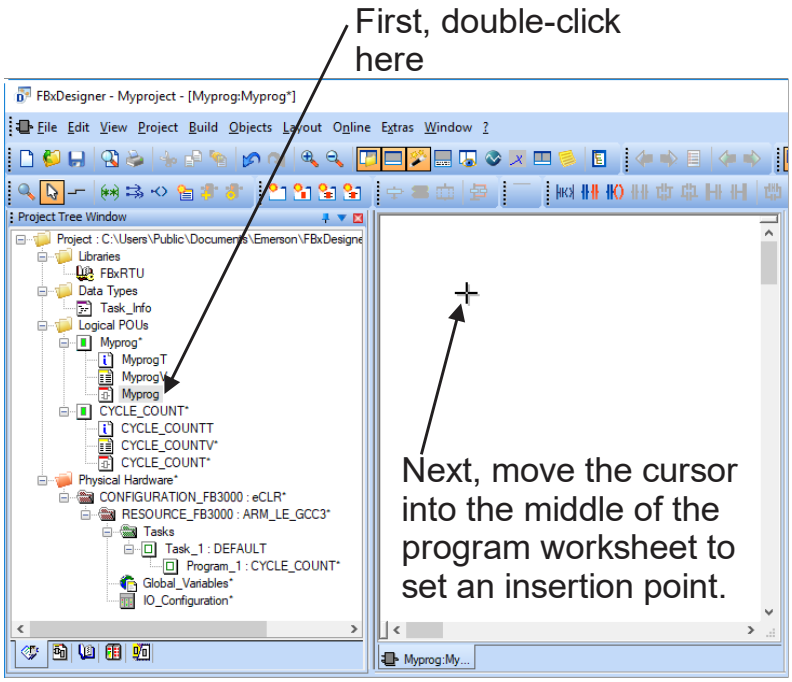
- Enter **Myprog** in the **Name.**
- Choose **Program** for the **Type.**
- Choose **FBD** for the **Language**.
- Click **OK.**



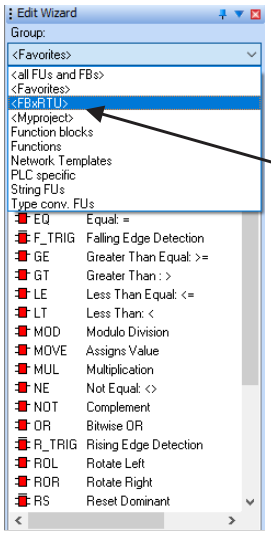
If you look at the Project Tree, you'll see we have created a Myprog item in the Project Tree with three worksheets.



Double-click on the third item which is where you enter program code and then click in the program code worksheet to set an insertion point for functions and function blocks.



If it's not already visible, click the Edit Wizard icon  to open the Edit Wizard pane. The Edit Wizard pane contains a list of functions and function blocks you can use in your program.

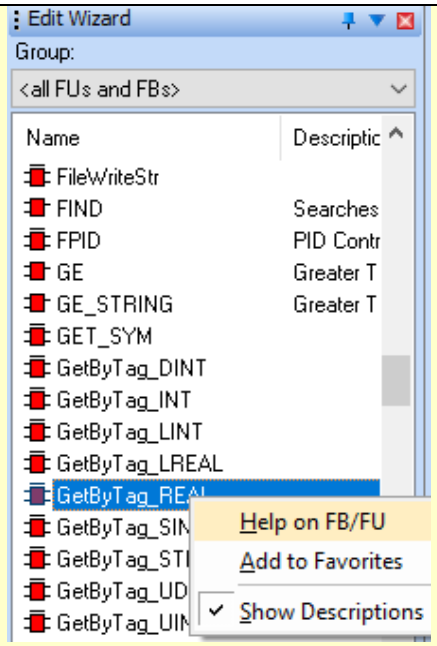


Click here to select which groups of functions and function blocks you want. In this case, we chose all of them "<all FUs and FBs>"

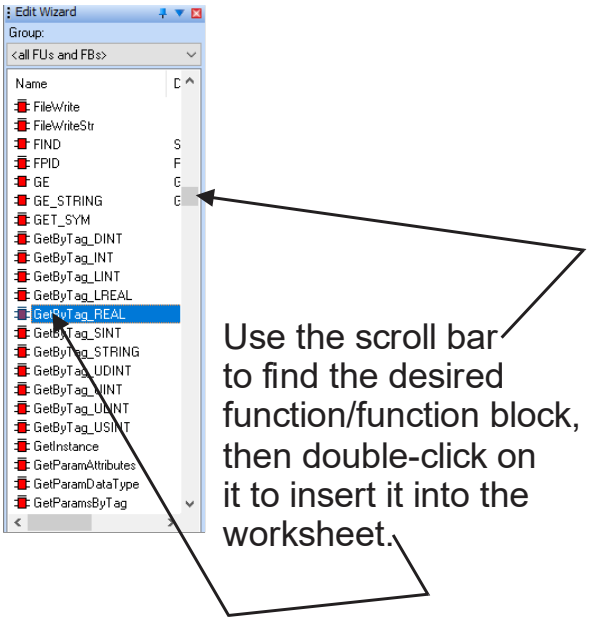
If you click at the top of the wizard, you can select **"<all FUs and FBs>"** so all possible functions and function blocks will be available in the wizard. Use the scroll bar to locate the function or function block you want to insert.

How do I find out what a function / function block does?

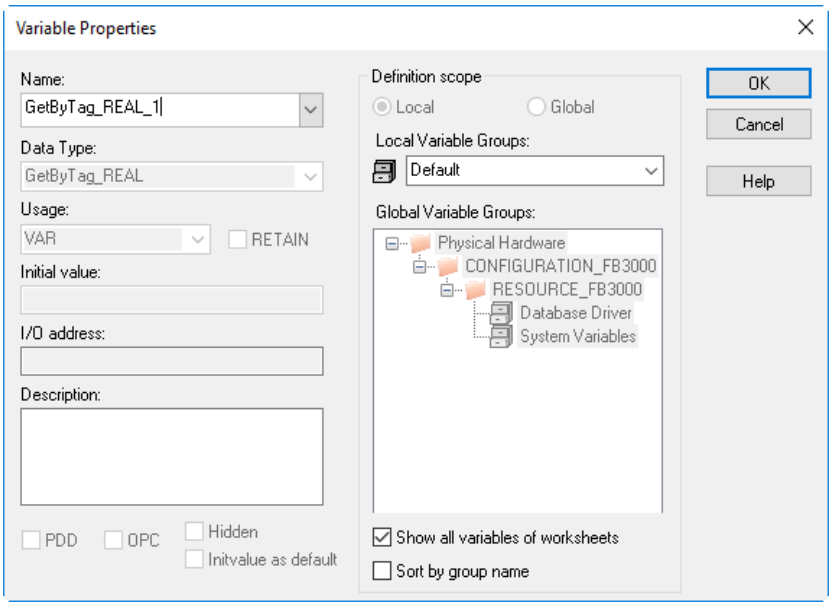
For more information about what a function/function block does, right-click on it in the Edit Wizard and choose **Help on FB/FU** in the pop-up menu to open up FBxDesigner's online help.You can also right-click on a function or function block that you've already placed in the worksheet to call up its associated online help page.



To start, we need to insert a **GetByTag_REAL** function block so scroll to that and double-click on it to insert it in the worksheet. (The reason we need a GetByTag_REAL function block is that we want to retrieve a REAL value (also known as a floating point value) from the RTU's database – in this case, the tank level – and bring it into the application. If we were writing to the database, we'd use a SetByTag function block.)



The Variable Properties dialog box, shown below, opens.

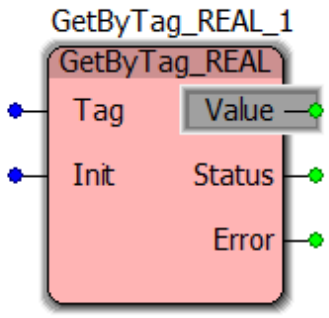


You might ask at this point, 'Why is the function block being called a variable?' The reason is that FBxDesigner doesn't know it's a function block yet, but the function block is referred to in the program by an instance name, which is like a variable name.

The instance name allows the program to distinguish between different instances of the same type of function block, for example, if you had two different GetByTag_REAL function blocks in the same POU. You can optionally enter a name of your choice for the

function block, or just use the default. You can also enter a comment in the “Description” field, if you want to. Now click on the **OK** push button.

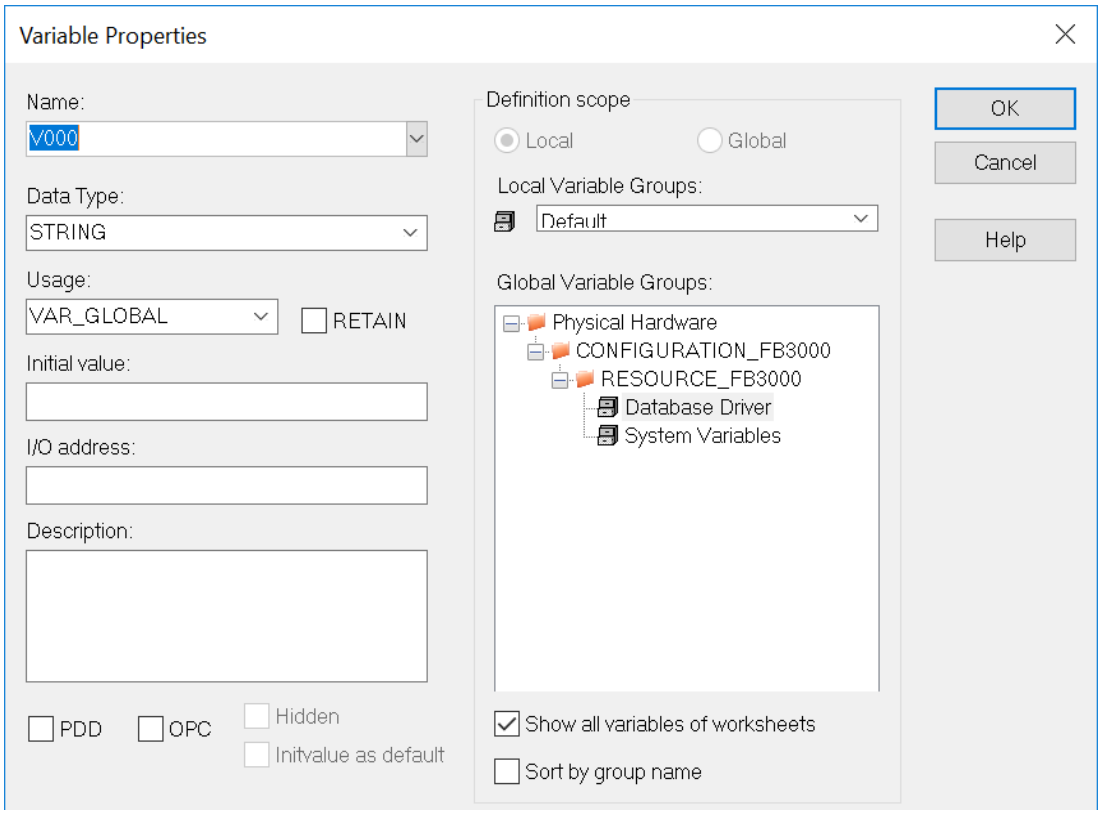
FBxDesigner inserts a graphical representation of the GetByTag_REAL function block into the program worksheet at the location of your cursor.



Once it’s inserted, you can drag it around with your mouse if you want to re-position it.

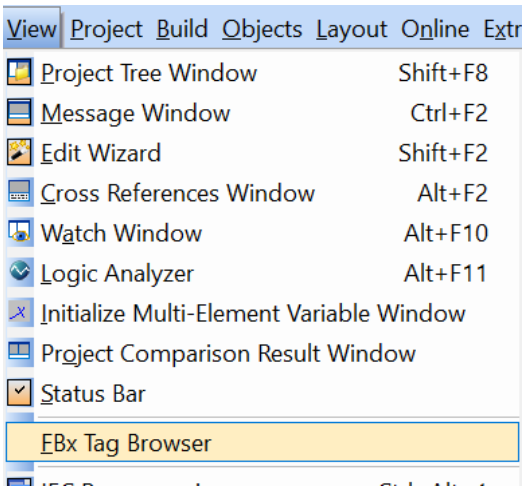
Double-click on the dot feeding into the **Tag** parameter of the GetByTag_REAL function block; alternatively, click once on the dot then go to the menu bar and click: **Objects→Variable.**

The Variable Properties dialog box opens. A default variable name beginning with ‘V00’ appears as well.



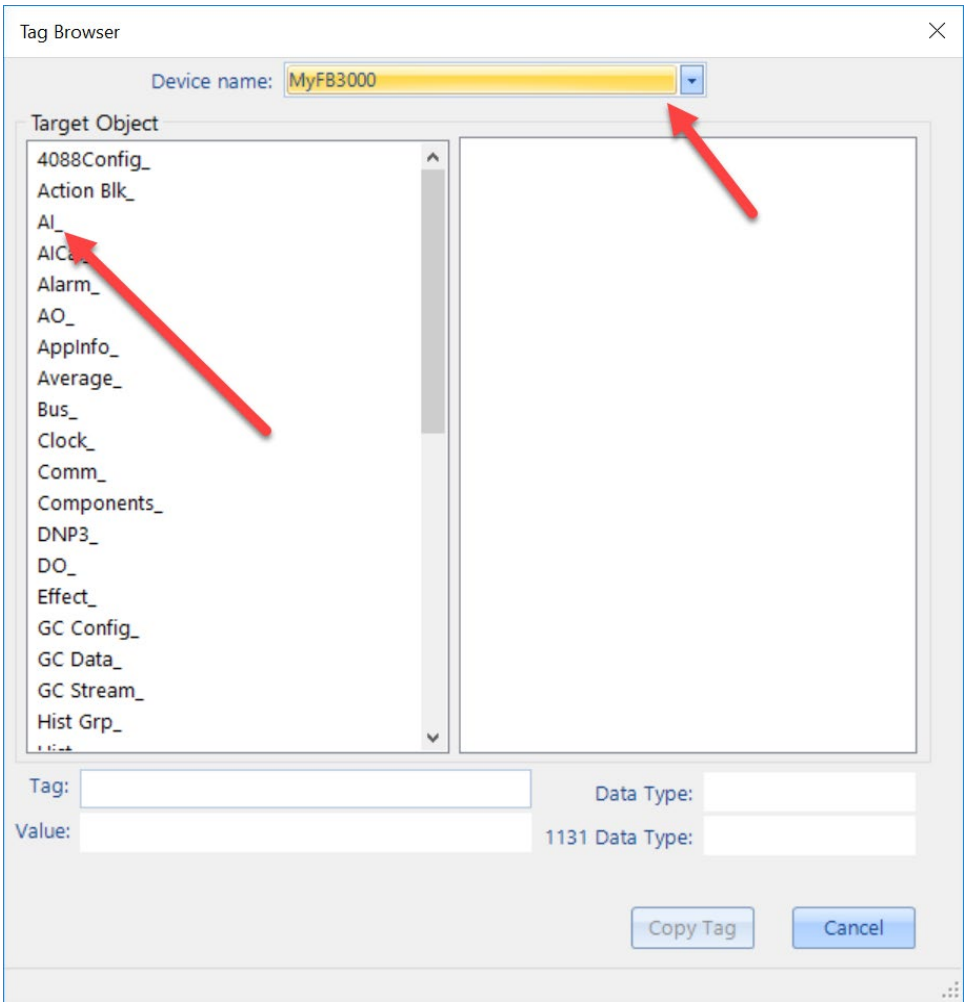
However, the GetByTag_REAL function block must receive its input from the point on the mixed I/O module that has the tank level (defined earlier in this guide in the section *Setting Up the I/O in FBxConnect*). If you look back at that section, you will remember that the mixed I/O module is in slot 2 of the RTU, and the tank level comes in on the first AI on that mixed I/O module.

To get the correct parameter, click **View > FBx Tag Browser**



When the Tag Browser opens, use the **Device name** field to select the RTU you want to get data from, and the Tag Browser displays a list of database objects associated with that RTU in the **Target Object** pane.

Notice that there are many database objects associated with the RTU; you can use the scroll bar to reach items not shown in the pane. Most of the items correspond directly to items you configure in FBxConnect.

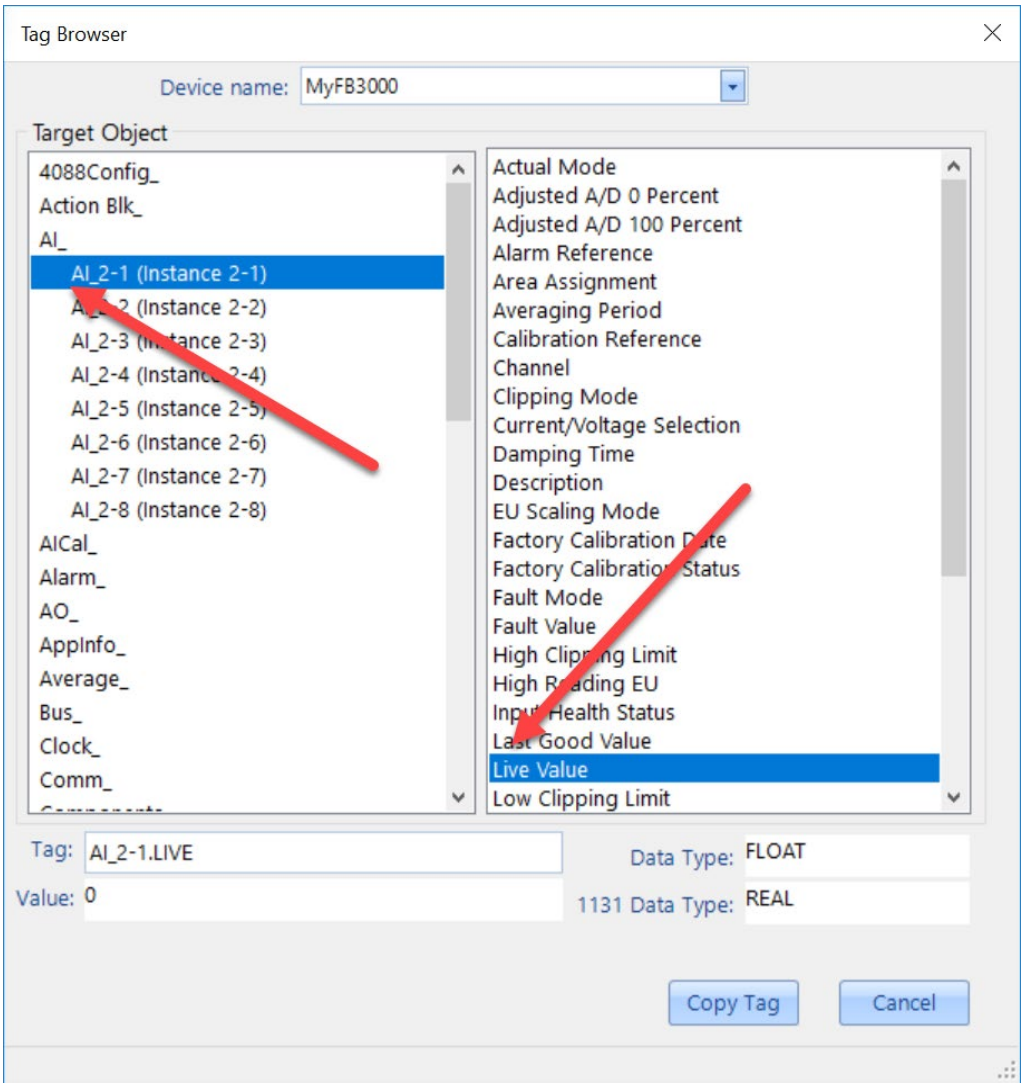


Since we want an AI, click the AI object. Since the I/O module has multiple AIs, each AI has its own copy (or instance) of the AI database object.

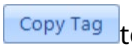
The Tag Browser shows you a list of the database instances. In this case, the I/O module in slot 2 has eight (8) individual AIs.

We want the first AI on the module, so click on the AI_2-1 instance.

Once you select the instance, the right pane shows all the different attributes of that database instance. You can use the scroll bar to see which attributes are available.



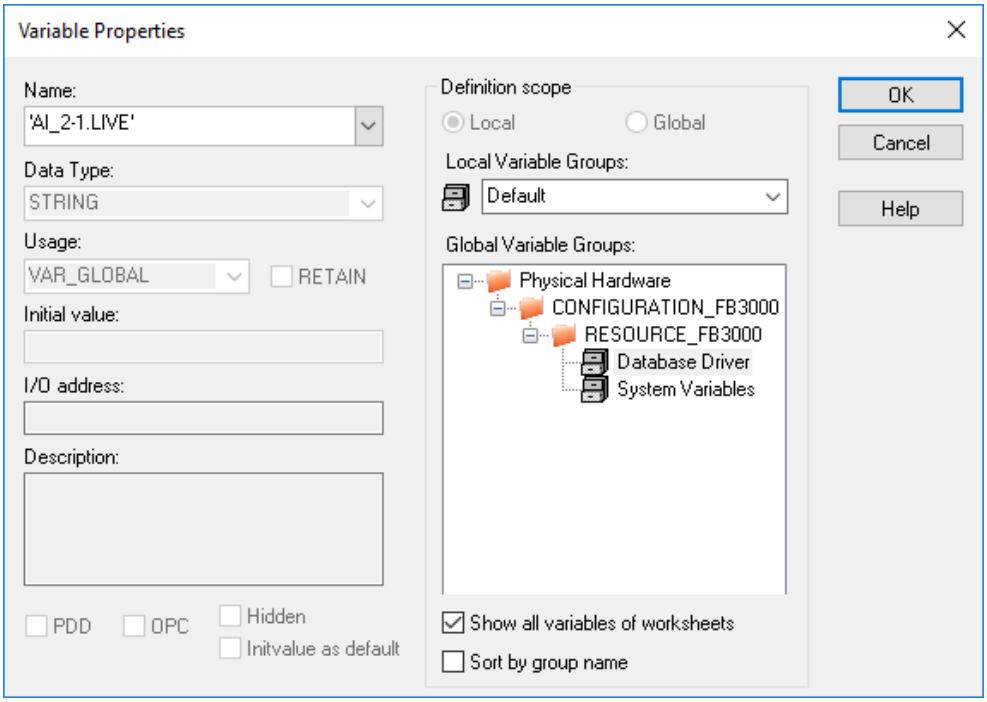
We want the live value of the AI, so make that selection and you will notice the **Tag** field updates to show the correct database tag name for the live value – **AI_2-1.LIVE**. You will also notice that the dialog box shows the database **Value** for the tag, the **Data ype** used in the FBxConnect database, and the corresponding (IEC) **1131 Data Type** used in FBxDesigner.

Click the **Copy Tag** button 

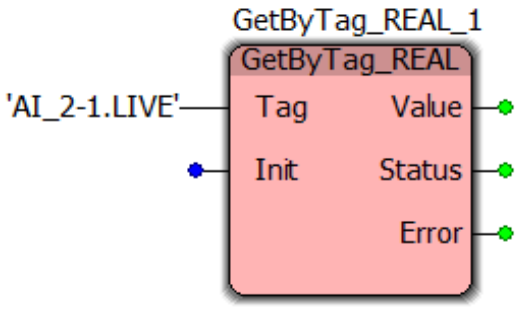
to store that name in the Windows clipboard.

Now double-click on the GetTagBy_REAL function block's Tag input to reopen the Variable Properties dialog box and go to the **Name** field. You can either backspace over the V000 to erase it, or just highlight that name, then press **[Ctrl-V]** on the keyboard (or click **Edit > Paste**) to paste the tag we want ('AI_2-1.LIVE') from the

clipboard. Click **OK** to exit the dialog box.

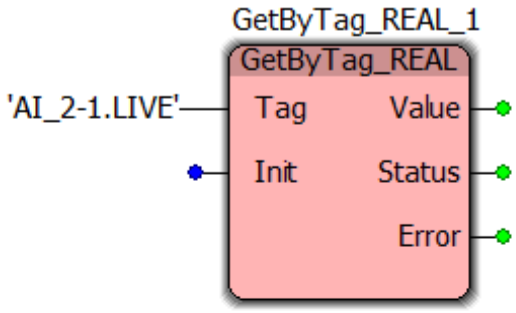


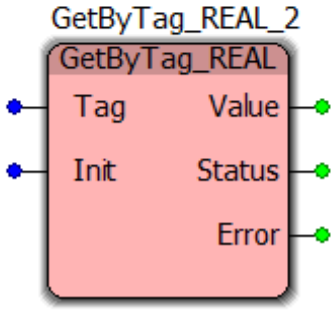
'AI_2-1.LIVE' will now be displayed on the **Tag** parameter of the GetByTag_REAL function block.



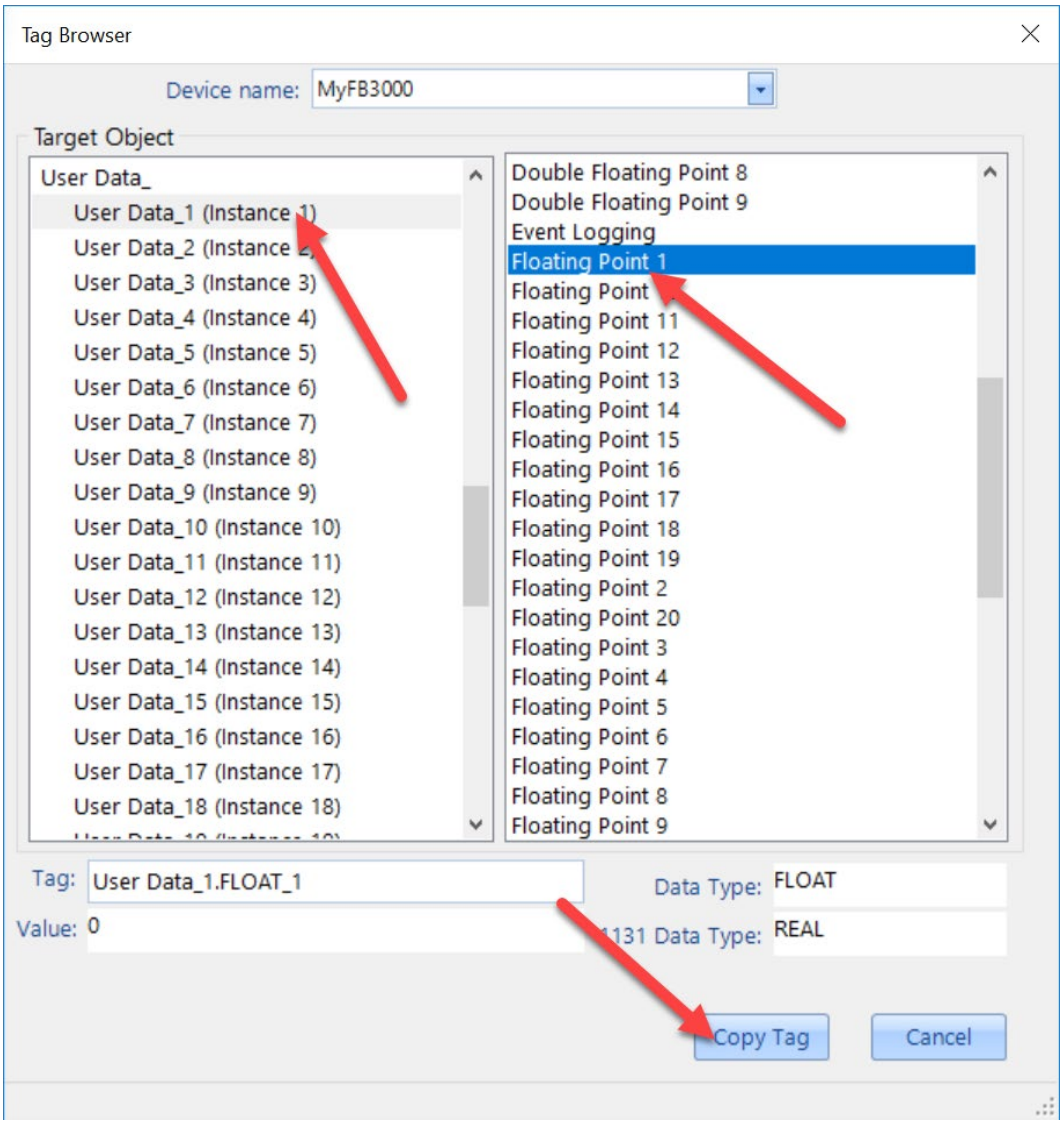
We want another value against which the tank level value gets compared to; like a setpoint or limit. Typically, there would be a field on a display where an operator would enter such a value, however, creating displays is beyond the scope of this guide, so we're going to just reference another value in the RTU's database. In this case, we'll reference the first user data point; which is one place a value someone enters from a screen might be stored.

So let's add another GetByTag_REAL function block to retrieve that user data point. Notice that when you add a second GetByTag_REAL function block, it's appended with a number two to distinguish it from the first instance of GetByTag_REAL.

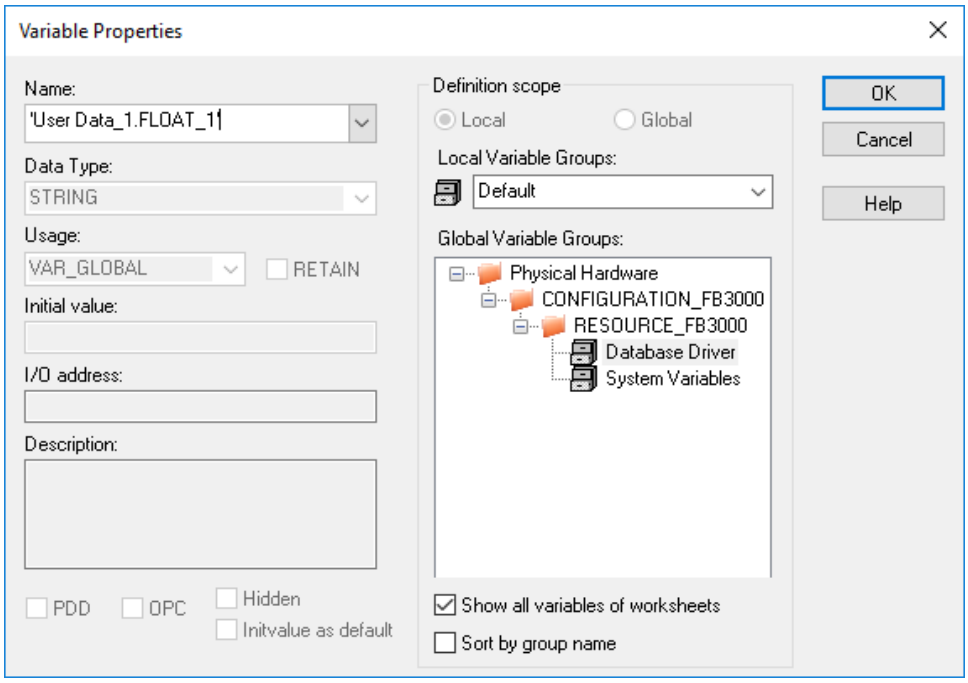




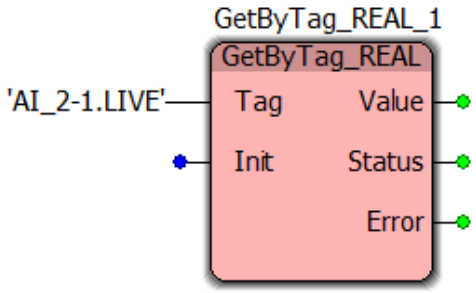
Now to get the name for the data point, go to the Tag Browser and in the **Target Object** field, scroll down to the User Data object, and click on it to see all the instances. We want **User Data 1** – the first instance, so click on that, and then in the right-most pane you can see all the choices for that object instance. We want to store a floating point value (real number) so click on **Floating Point 1** to populate the **Tag** field in the Tag Browser, and click **Copy Tag**.

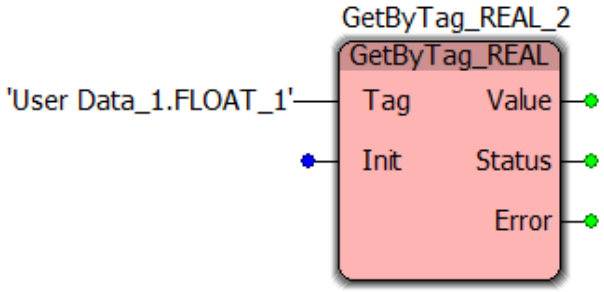


Now go back to the second GetByTag_REAL function block and double-click on the **Tag** point to bring up the Variable Properties dialog box. Either erase what's in the **Name** field or highlight it, and then press **[Ctrl-V]** or click **Edit > Paste** to fill the **Name** field with the tag name we copied - 'User Data_1.FLOAT_1'. Finally, click **OK**.



Here's what the result looks like:



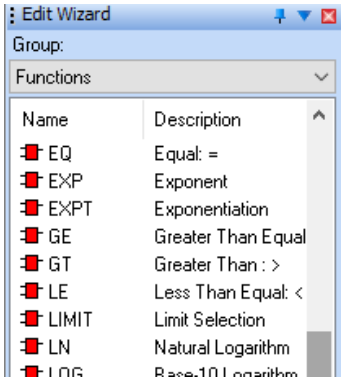


Now the REAL values of the two function blocks, which get retrieved from the RTU are

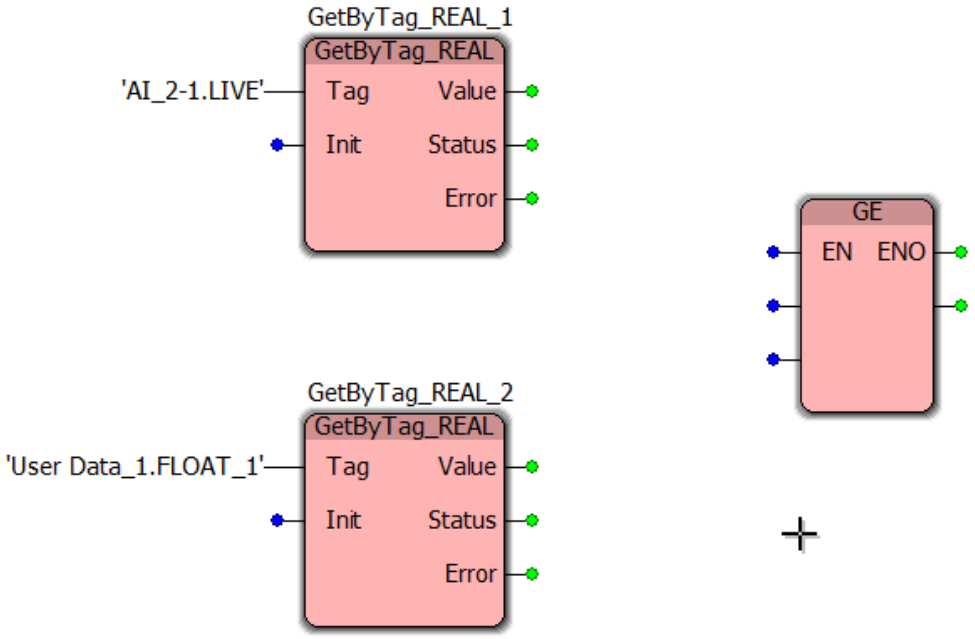
what we want to compare. They will serve as *inputs* to another function– the GE function.

The GE (for **G**reater than or **E**qual to) function compares two values and sets a BOOL value based on the comparison. Let's add that function now.

To insert the GE function, click in an open area of the window pane. Again, in the Edit Wizard you must choose "<All FBs and FUs>" for the **Group** to see all the choices, then scroll down to locate the GE function block, and double-click on it to place it at our insertion point in the worksheet.

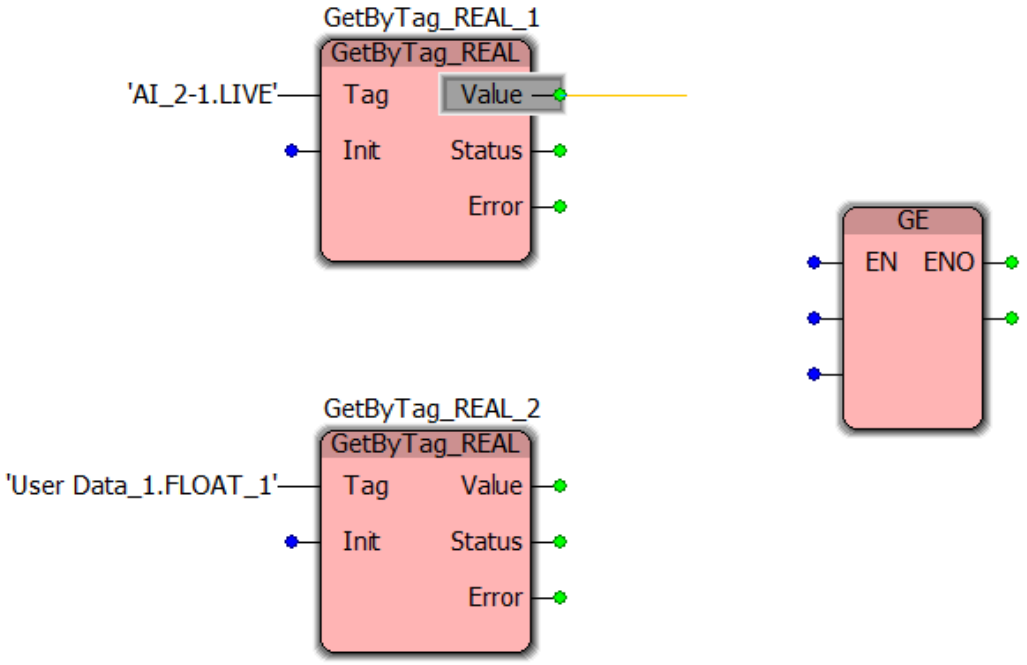


Now we have two GetByTag_REAL function blocks and a GE function, but they are not connected.

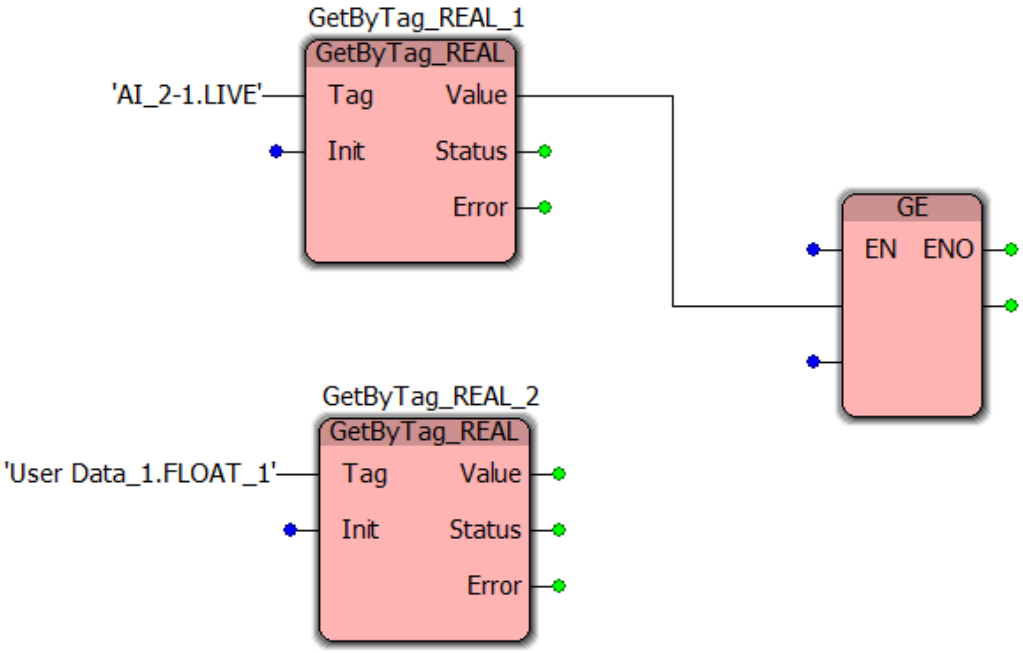


We want to take the output of the top GetByTag_REAL function block (which is the tank level retrieved from the RTU's database) and feed that into the input of the GE (greater than or equal to) function.

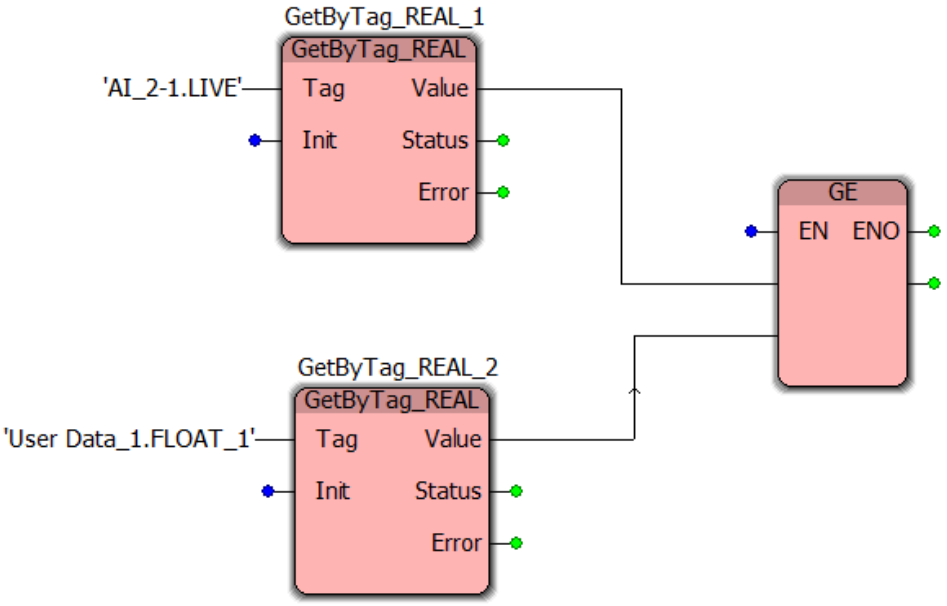
To do this; click on the green dot representing the **Value** parameter of the GetByTag_REAL function block. Move the mouse horizontally and you will notice that a line is drawn as you move the cursor.



Drag the mouse down to the level of one of the middle input points of the GE function, then without releasing the mouse, drag to the input point to form a connection.



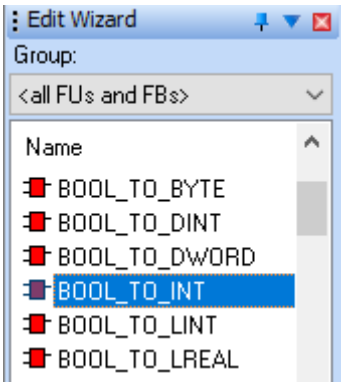
Repeat this process with the Value output of the GetByTag_REAL2 function block, except this time, drag the mouse upwards to the level of the bottom-most input point on the GE function, and make that connection.



Now we have two inputs to the GE function. We want to open a drain valve on a tank when the tank level value (‘AI_2-1.LIVE’) is greater than or equal to some specified value

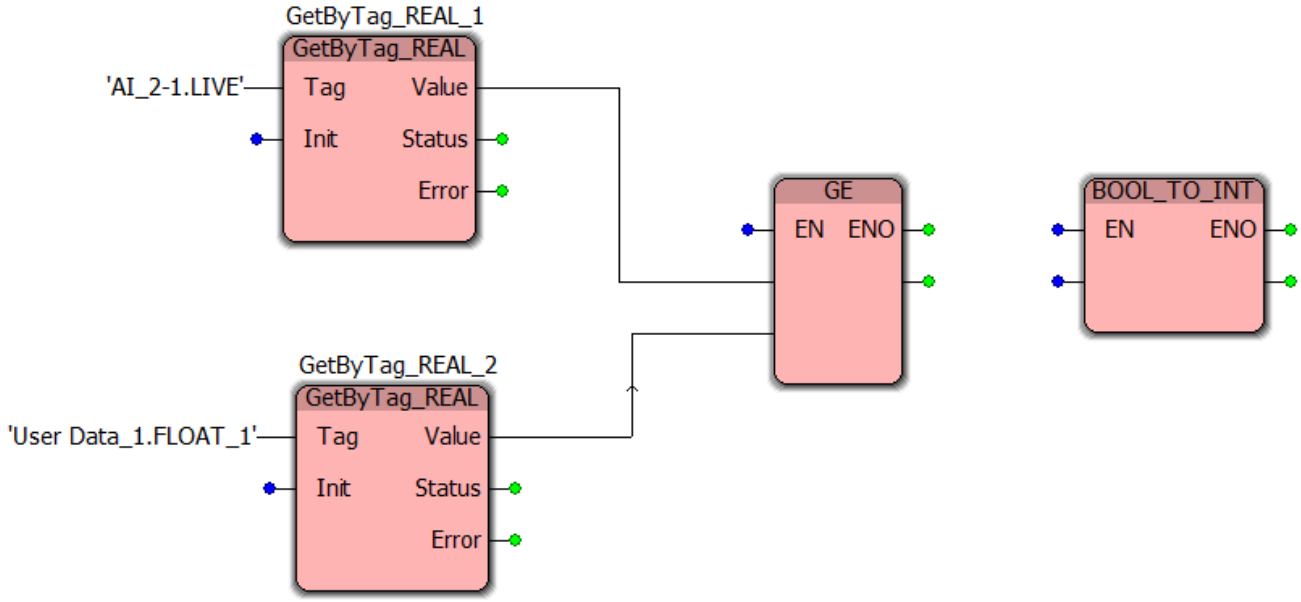
('User Data_1.FLOAT_1'). The GE function turns an output of ON (TRUE), whenever 'AI_2-1.LIVE' ≥ 'User Data_1.FLOAT_1' so we want to take that output (a BOOL value) and send it to the RTU.

The database in the RTU, however, stores BOOL values as integer values 0 (FALSE or OFF) or 1 (TRUE or ON), so we have a little more work to do.

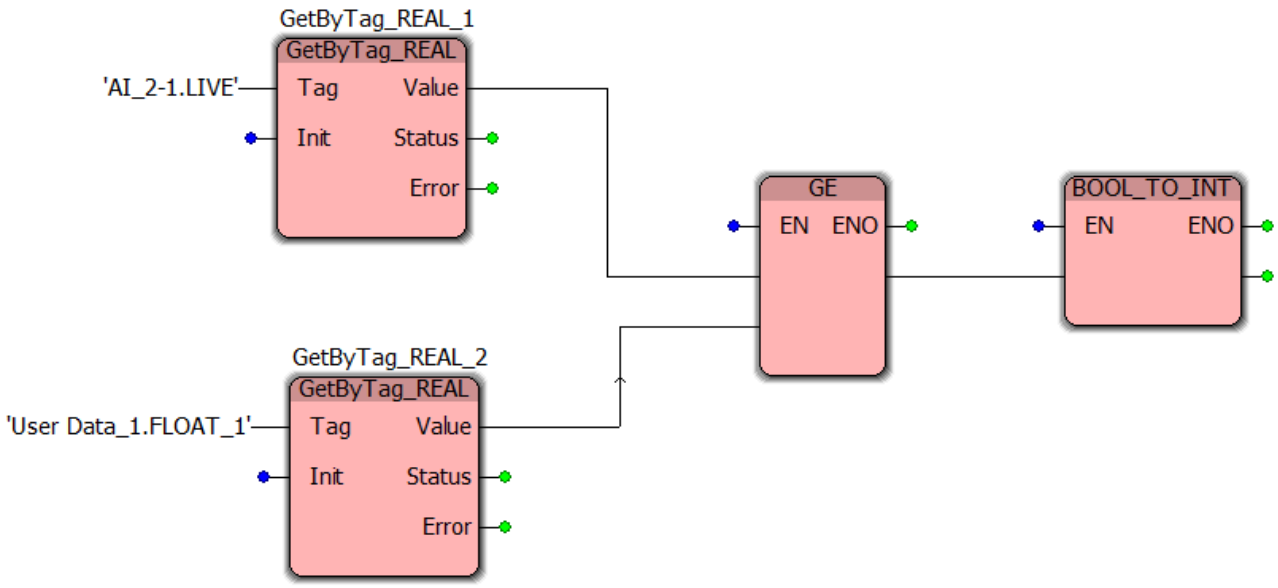


We have to take the output of the GE function, and convert it from a BOOL to an INT.

Luckily, there's a function that does that for us called BOOL_TO_INT so locate it in the Edit Wizard and insert it into our worksheet at our insertion point.

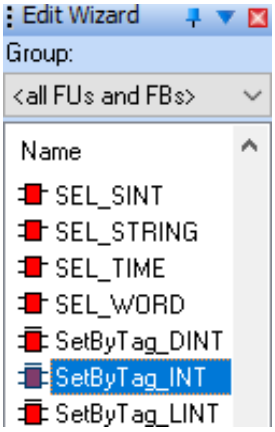


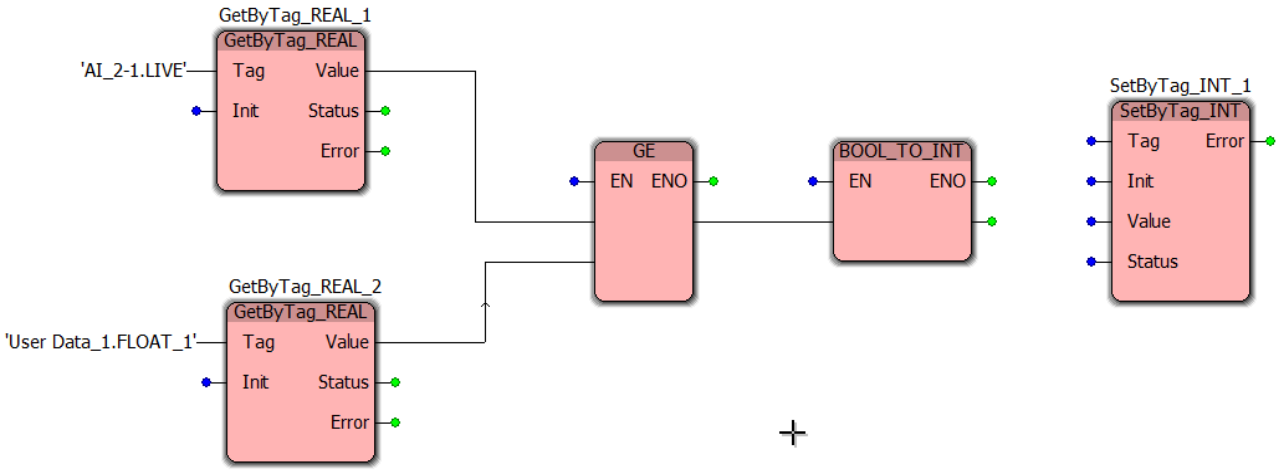
Once we add the BOOL_TO_INT, we need to connect the output of the GE function to the input of the BOOL_TO_INT function so it looks like this:



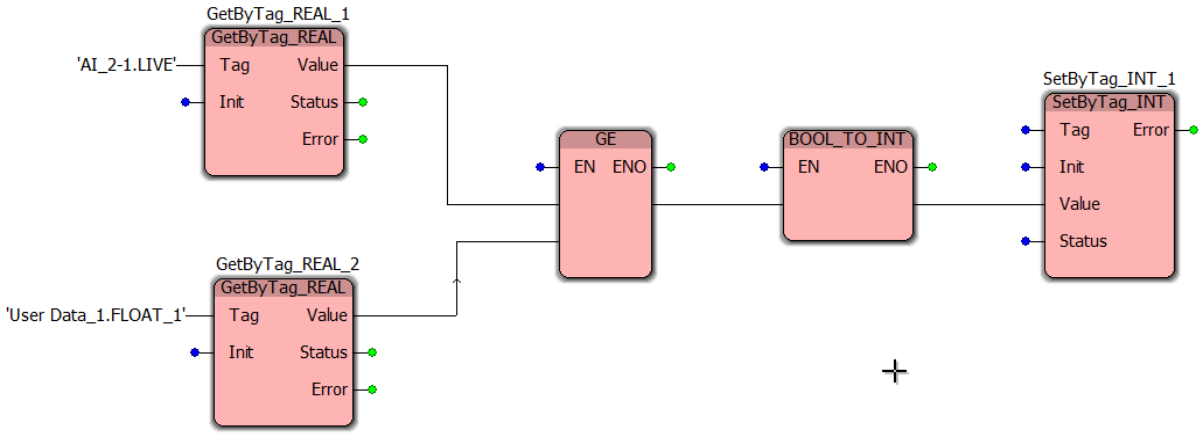
Now that we have a way to convert the BOOL to an INT, we need a way to get that converted value into the RTU's database, so we need to add a SetByTag_INT function block.

Locate the SetByTag_INT function in the Edit Wizard and double-click to add it to your insertion point in the worksheet.

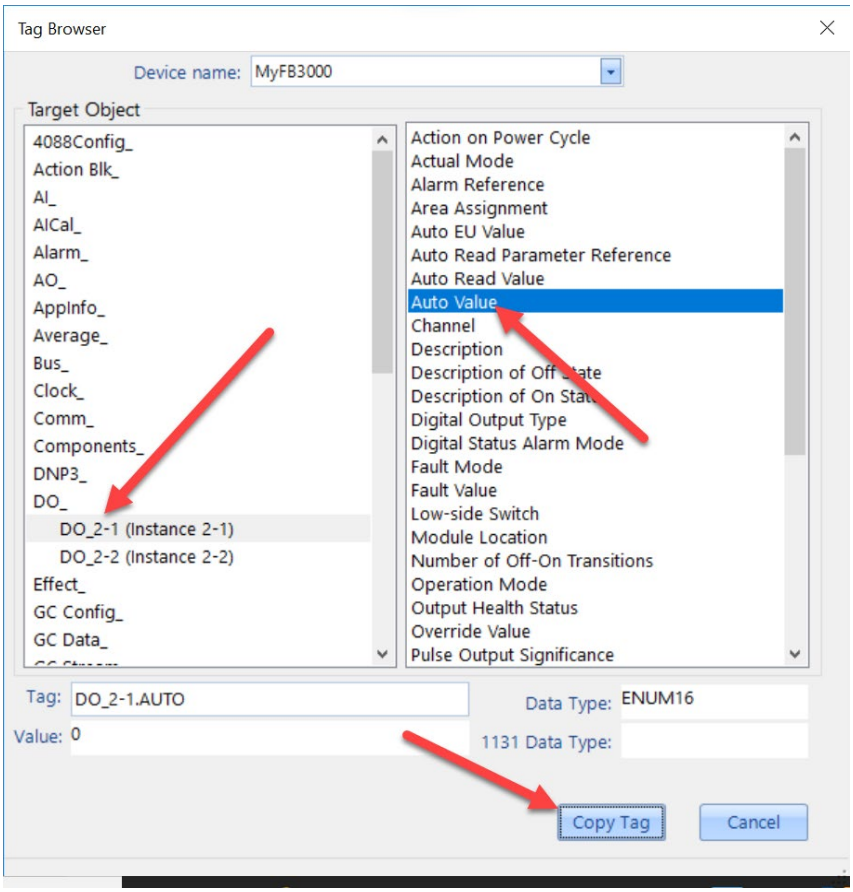




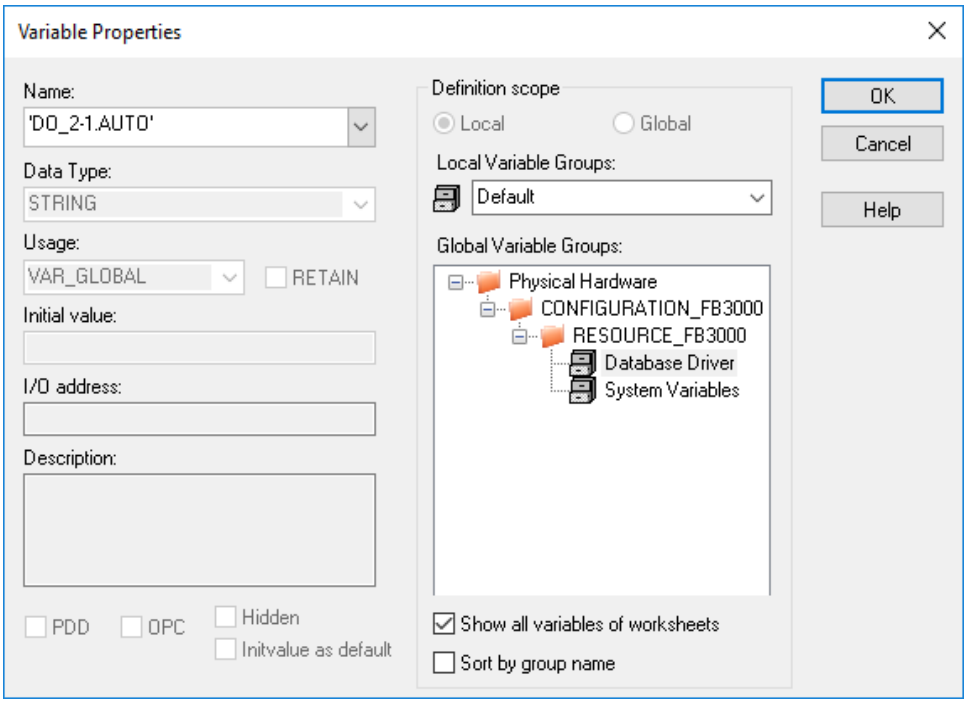
Now connect the output of BOOL_TO_INT to the **Value** parameter of SetByTag_INT:



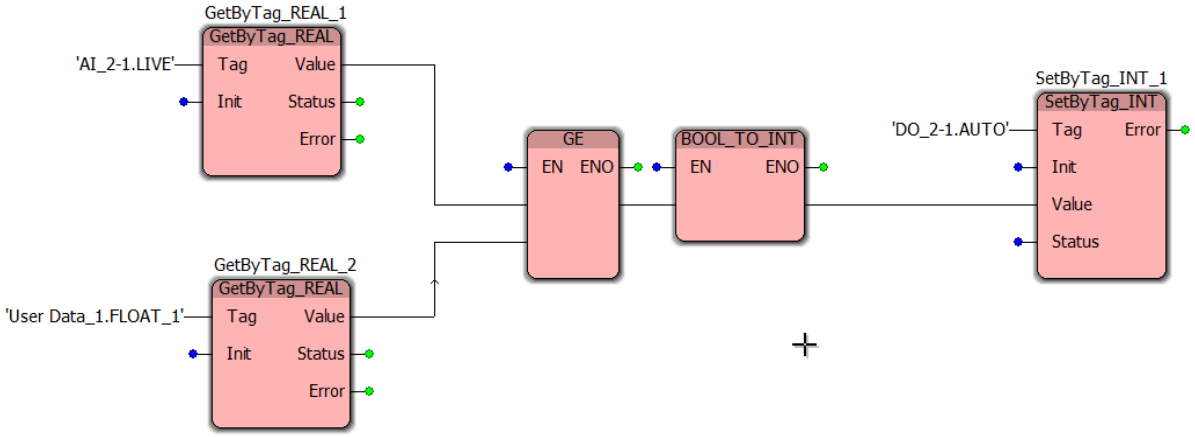
Now we need to specify the tag name in the RTU database that will receive the value. It has to go to digital output 2-1 of the mixed I/O module so go to the Tag Browser, and click on the DO **Target Object**, then choose the **Auto Value** attribute and click **Copy Tag**.



Now double-click on the Tag input to the SetByTag_INT_1 function block, and either erase or highlight whatever is in the **Name** field of the Variable Properties dialog box and press **[Ctrl-V]** or click **Edit > Paste** to fill the **Name** field with the 'DO_2-1.AUTO' tag. Finally, click **OK**.



Now the worksheet looks like this:



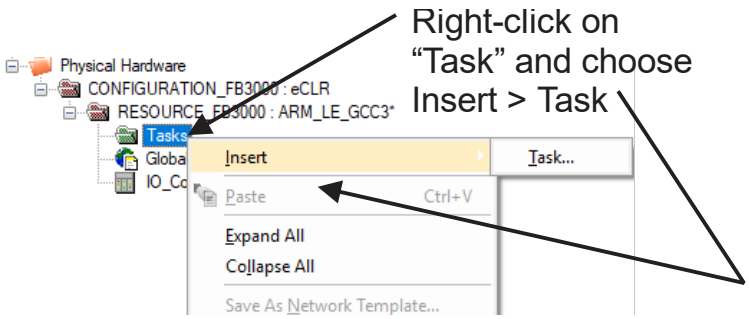
Click **Save**.

Our program (POU) is done! In order to make it do anything, however, it has to be associated with a task.

Creating a Task to Execute the Program

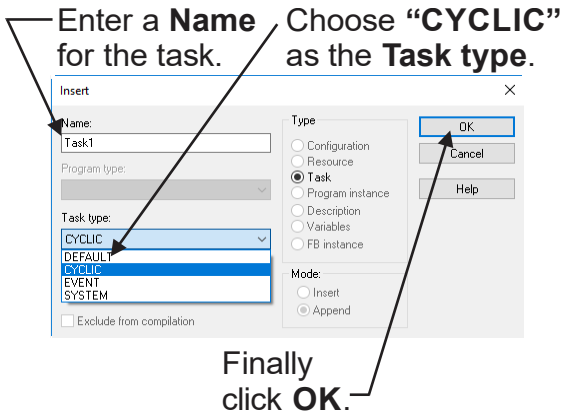
Tasks determine how fast a program, or program(s) are executed. When a task is started, it reads all inputs from the input process I/O boards, it then performs all calculations in the programs of a task, and then it writes output data out through the output points on the process I/O boards.

Tasks are found below the **"Physical Hardware"** branch of the Project Tree. To create a task, *right*-click on the **"Tasks"** icon in the project tree and choose **"Insert"** and **"Task'** from the pop-up menu.



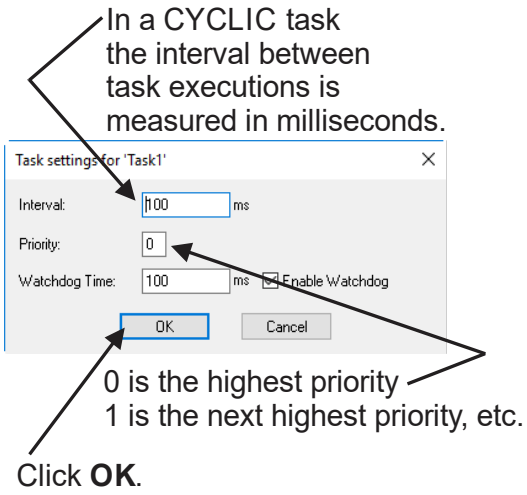
The Insert dialog box opens.

Enter a **Name** for the task. In this case we have chosen "Task1'. If you enter "DEFAULT' for the task type, the task will run whenever time is available - this is sometimes referred to as the "idle task." In general, "CYCLIC" should be chosen for the **Task type**. Click **OK** when finished.



The Task Settings dialog box opens. Choose the rate of execution of the task by entering a value in the **Interval** field. The interval is measured in milliseconds. Generally,

Watchdog Time is set to the same value as the **Interval**. A Watchdog condition is reported if the task does not execute within the watchdog time. This allows task slippage to be detected.

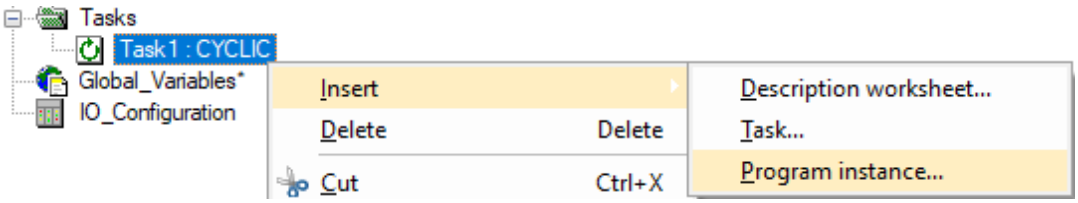


If desired, specify a **Priority** for the task. “0” is considered the highest priority. “1” is the next highest priority, and so on.

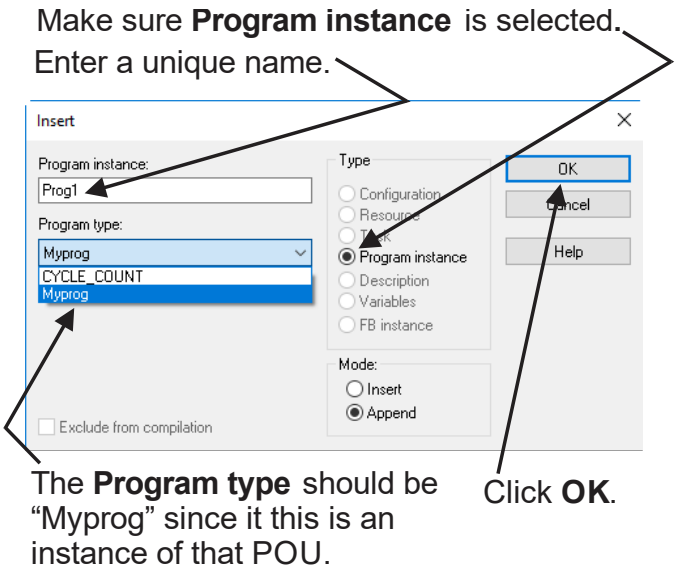
NOTE: If you chose “DEFAULT” as the task name, the task will **not** execute according to an interval, but instead, will be executed whenever time is available. It serves as the idle task.

When finished, click on **OK** and the task will have been defined.

Now that the task exists, however, it is necessary to associate the program that we wrote with the task. Otherwise we will just have an empty task executing. To associate the program with the task, right-click on the icon for the task in the project tree, and choose **Insert**>**Program instance** from the pop-up menu.



The Insert dialog box opens. Choose **Program instan**ce as the **Type**.



When we insert the program, we are actually creating something called a program instance. Each program instance must be assigned a unique name - in this case we chose "Prog1". The program instance is essentially a copy of the "Myprog" program POU we created, so we choose "Myprog" as the Program type. Copies are used because, theoretically, you might want to use the same program in different tasks, and if you didn't use a copy there would not be available memory set aside for internal (local) variables for each separate copy.

All local variables created in a POU are only for a particular instance. This becomes very important if you want to create a user defined function block from one of your POUs.

Click **OK** when finished.

Compile the Program, and if necessary, correct any errors

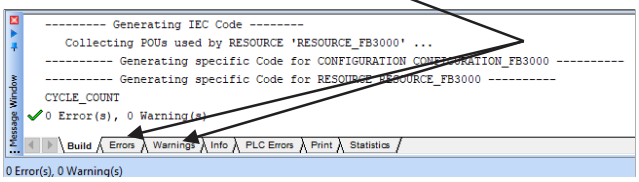
Now the program can be compiled. The compilation process takes your project (programs, function blocks, tasks, etc.) and generates machine-readable code from it, that can be run in the FB3000.

The compilation process checks for any syntactical errors in your program, and also issues warnings about possible problems with the structure of the program. It does **not** check for logic errors in your control strategy, however.

To compile the program, click on the icon shown at left, or go to the menu bar, and click as follows: **Build>Make**

Various messages scroll by on the screen.

If there are errors or warnings, click on the "Errors" or "Warnings" tabs.



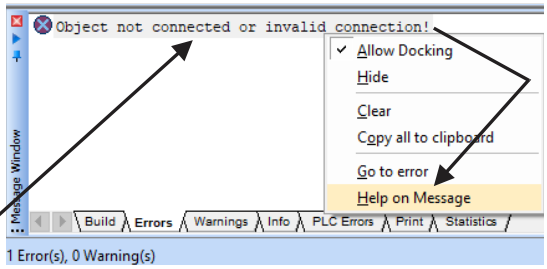
If there are errors or warnings generated during the compilation, you can view them by clicking on the "Errors" or "Warnings" tabs, respectively.

Often, you can double-click on the error listed in the error window, and the compiler identifies its location in the project.

For more information about what a particular error message means, right-click on the error message, then choose **Help on Message** from the pop-up menu (if it is available.)

Note: Help is not available for all error messages, and in some cases, the error might not be located in the exact location if you've edited text above it and did not recompile the project.

Double-click on the message to jump to the location in the file where the compiler detects the error.



Right-click on the error and choose "Help on Message" from the pop-up menu to view more information about the error.

Save Your Project

To save your project, click on **File>Save Project As/Zip Project As** and specify a path and filename for the project.

If you save the project "as is" i.e. uncompressed, it will have a file extension of *.Mwt, and will also have a sub-directory of files associated with it with the same name as the project. If you choose to zip the project (i.e. compress it), it will have a file extension of *.Zwt. To create an application package that you can download via FBxConnect, choose **Mwt**.

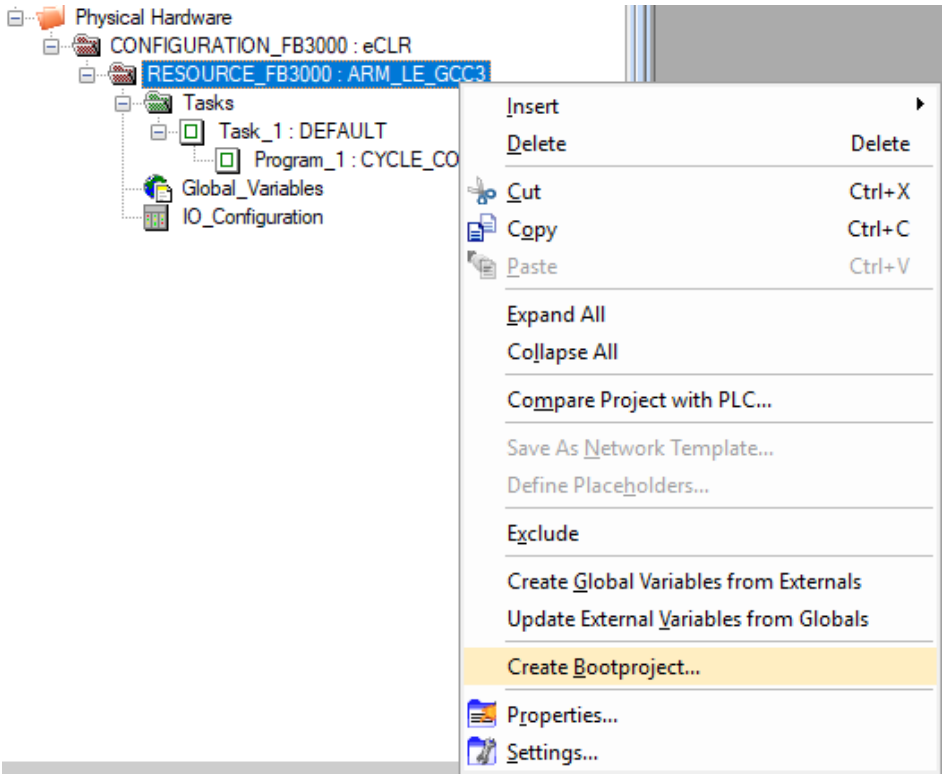
Notes

- If you intend to copy the project to a different location, or a different PC, it is strongly recommended you zip the project first to ensure that all files are copied.
- If, after unzipping a project, you try to save it and you encounter a message “Cannot copy file *file_path\filename*.TXT. Command aborted!”, go in Windows Explorer to the path and filename listed, and delete the named file(s). Then try to save the project again. If a similar message appears, repeat this process, as necessary. You will then be able to successfully save the project.

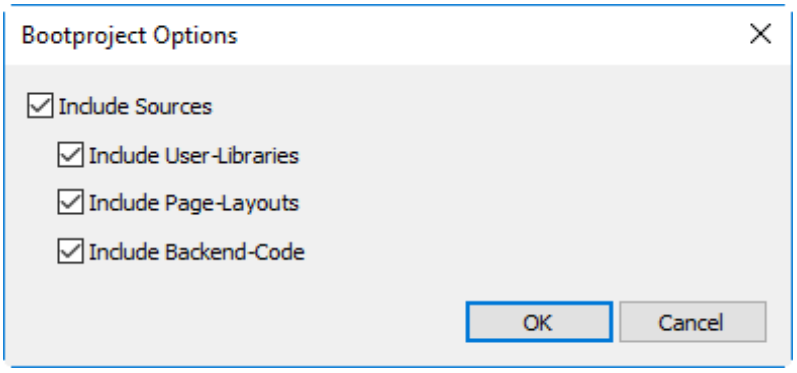
Creating the Boot Project

The FBxDesigner .MWT or .ZWT file cannot execute directly within the FB3000. You need to generate a machine-readable boot file that can run in the FB3000:

1. Right-click on the RESOURCE and choose **Create Bootproject.**



2. In the Bootproject Options dialog box, you can optionally select **Include Sources** and select individual elements of the project source files to include within the boot project. Then click **OK**.



3. FBxDesigner now reports various messages as it creates the boot project.

Creating and Downloading an Application into the Connected RTU

There are eight software “slots” in the FB3000 RTU which can hold applications.

An application consists of the following items:

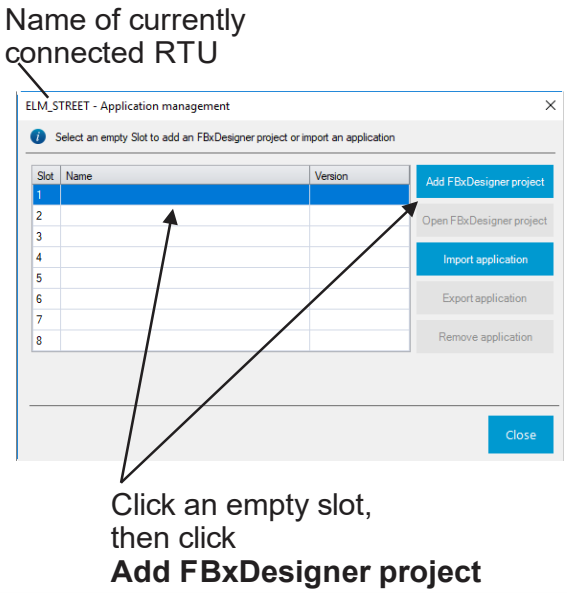
- An FBxDesigner project (which must include the boot project)
- Display file(s) created in FBxConnect - OPTIONAL
- User-Defined Database Objects - OPTIONAL

Note: The component of the FBxDesigner project folder extracted and loaded into the RTU is the machine-readable boot file known as the boot project. The boot project is **not** automatically created when you build a project in FBxDesigner, so any changes to an FBxDesigner project loaded via FBxConnect are only reflected if you re-created the Bootproject (see *Creating the Boot Project*).

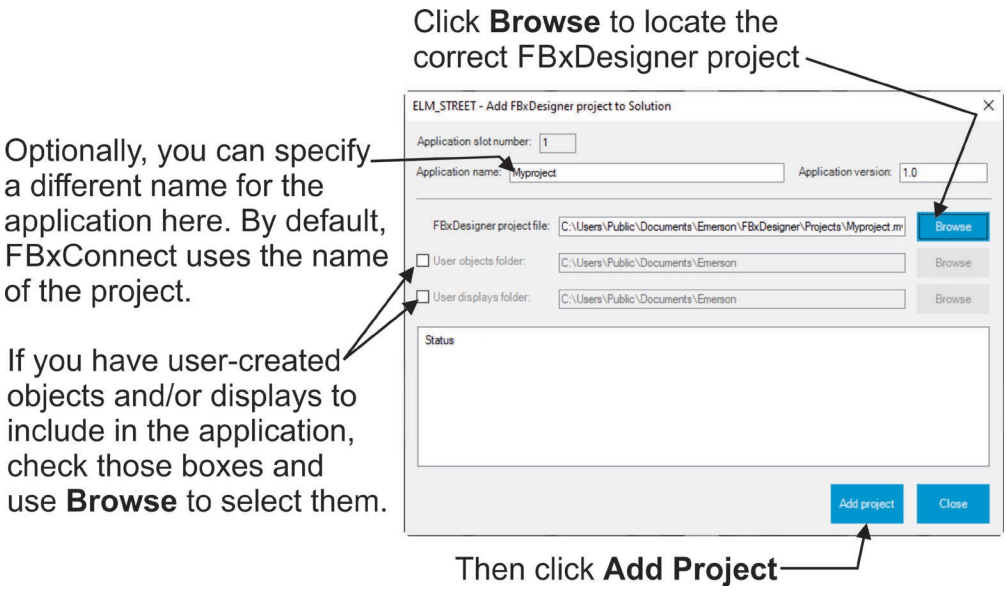
Users should never attempt to download *untested* software into a controller if the controller is currently connected to a running plant or industrial process. Safeguards

must be taken prior to downloading to ensure that the controller is isolated from the process and I/O is disconnected. Failure to take such precautions could result in injury to persons or damage to property.

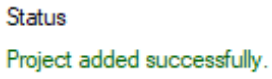
1. Connect to the RTU through Field Tools and when FBxConnect launches, click **Applications >Management**
2. Click an empty slot into which you want to download an application, then click **Add FBxDesigner Project**



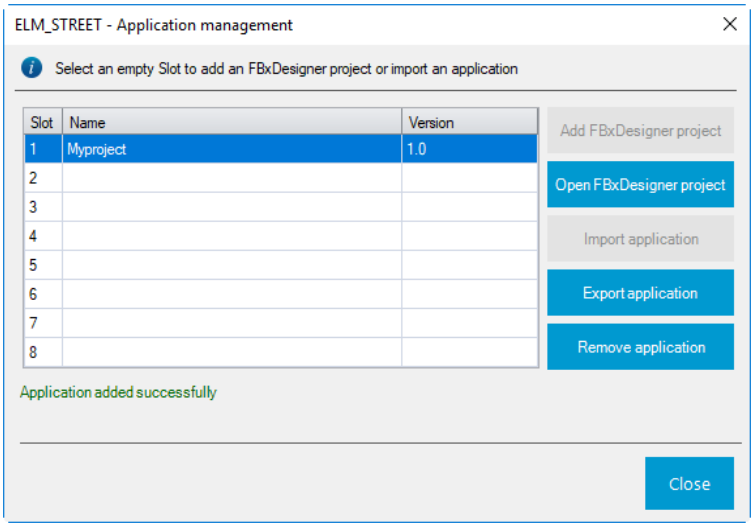
3. **Browse** to the appropriate folder and select the project's .MWT file; this automatically also selects the Bootproject you created earlier. By default, the application name for the project will match the project name, but you can optionally specify a different name that can be up to 40 characters long. **Note:** If you only have a ZWT file, unzip it first to generate the MWT. If we had displays or user-created objects to include with the project, we would check those boxes and browse to those items as well but creating those is beyond the scope of this guide. Since we don't have those, leave those boxes unchecked. Once the selected project is shown in the **FBxDesigner project file** field, click **Add project**.



The status pane now shows **Project added successfully.**



4. Click **Close** to exit the dialog box and FBxConnect automatically begins to download the project into the RTU. The Application Management dialog box returns to show the download progress and reports **Application added successfully** when the download completes, and the project starts executing.

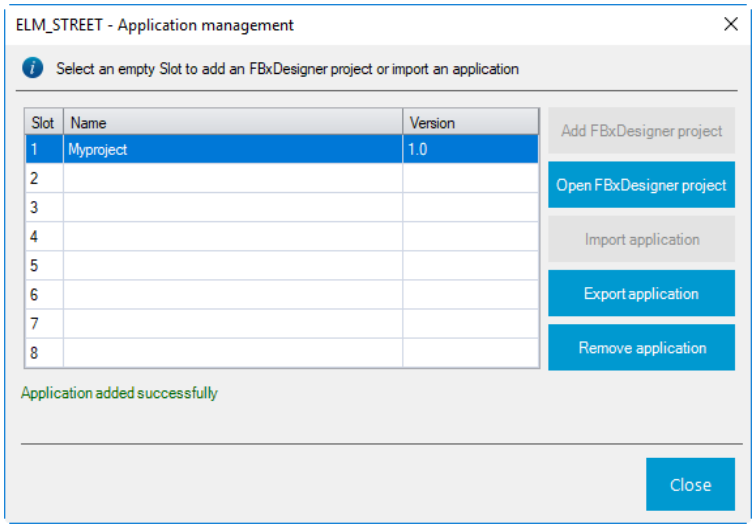


5. Click **Close** to exit the Application management dialog box.

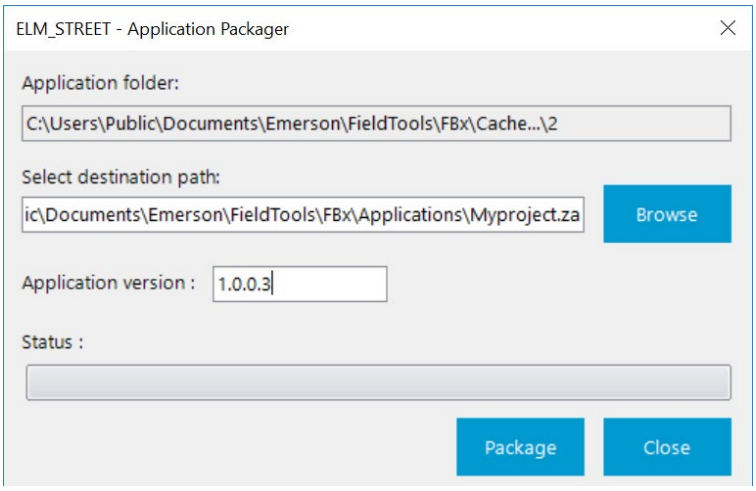
Packaging an Application for Reuse

Once you successfully download an application, you can package it into a .ZAP file that you can reuse on other RTUs.

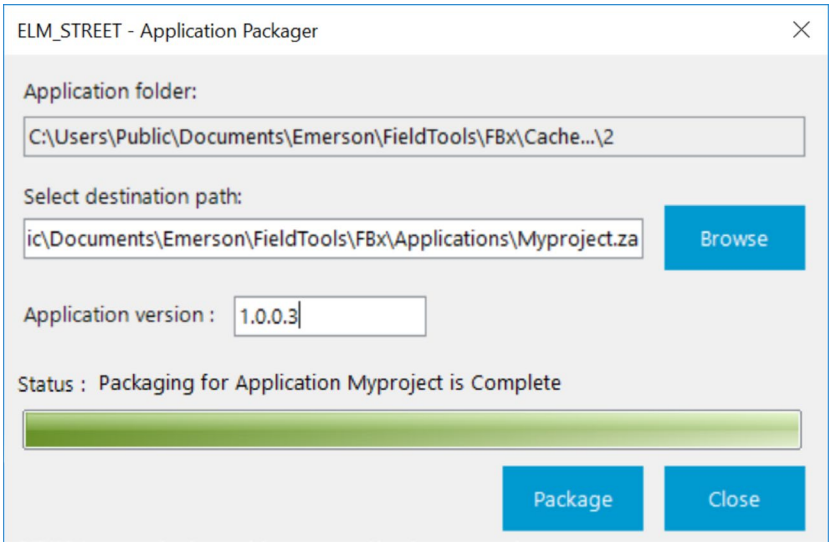
1. Connect to the RTU through Field Tools and when FBxConnect launches, click **Applications >Management**
2. Click the **Name** of the application in its slot that you want to package, then click **Export application** to launch the Application Packager.



3. In the Application Packager, the **Application folder** shows the location of the files you just selected, and the **Select destination path** field shows where the ZAP file will be created. (You can optionally specify a different path for the destination if you click **Browse**.) Also, you can optionally specify an **Application version** different than the default shown to distinguish between different versions of the application.



4. Click **Package** to generate the ZAP file. If you save it in the default location, you can find it at **\Users\Public\Documents\Emerson\FieldTools\FBx\Applications**. You can copy it to a memory stick and use it on any PC with FBxConnect.



5. Click **Close**. You can now download that ZAP file to any other FB3000 RTU.

Removing a Project/Application

If a slot is already occupied, you must remove the current application shown in the slot before you can download a different application into that slot. if you need to remove an existing application.

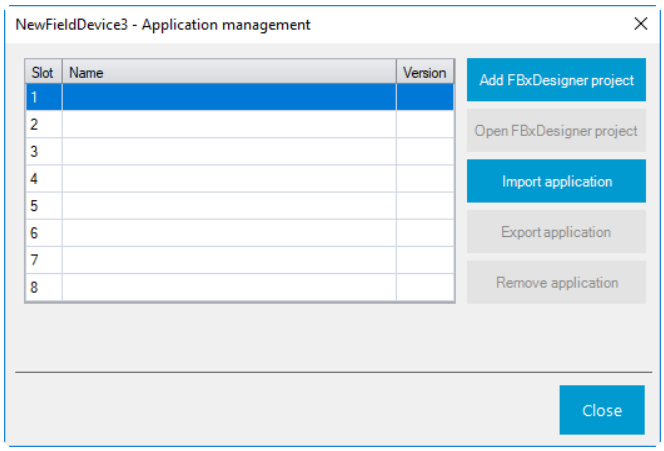
1. Start FBxConnect and click **Applications > Management**.
2. In the Application Management dialog box, click the application you want to remove, then click **Remove Application.**
3. A prompt asks you to confirm you want to remove the application. If you click **Yes**, FBxDesigner deletes the application from the device; if you click **No**, the application remains untouched.

Downloading an Application to an RTU

⚠ DANGER

Users should never attempt to download *untested* software into a controller if the controller is currently connected to a running plant or industrial process. Safeguards must be taken prior to downloading to ensure that the controller is isolated from the process and I/O is disconnected. Failure to take such precautions could result in injury to persons or damage to property.

1. Insert the memory stick containing your application(s) into the PC you use to communicate with the RTU.
2. Connect to the RTU through Field Tools and when FBxConnect launches, click **Applications >Management**



3. Click an empty slot into which you want to download an application, then click **Import application** and browse to the appropriate folder on the memory stick and select the application, then click **Open**.

> **Note**
> If you receive a prompt regarding hash codes, it means your system is configured to display hash codes for imported applications to allow you to verify the authenticity of the application if it was published by Emerson. In this case, you would compare the hash codes shown with those posted on the Guardian site for that particular application to verify the application was unchanged. In this guide, we are talking about creating your own application, so there are no published codes you need to compare it to. Since you created the application yourself, you can just click **Yes**.

4. FBxConnect automatically downloads the application into the RTU and starts it. Click **Close** to exit the dialog box.

FBxDesigner™ Quick Start Guide

D301860X012
November 2024

For customer service and technical support, visit *Emerson.com/Guardian*.

North America and Latin America:
Emerson Energy and Transportation Solutions
6005 Rogerdale Road
Houston, TX 77072 U.S.A.
T +1 281 879 2699 | F +1 281 988 4445
Emerson.com/SCADAforEnergy

United Kingdom:
Emerson Process Management Limited
Fosse House, 6 Smith Way
Grove Park, Enderby
Leicester LE19 1SX UK
T +44 0 870 240 1978

Europe:
Emerson S.R.L
Regulatory Compliance Shared Services Department
Company No. J12/88/2006
Emerson 4 Street
Parcul Industrial Tetarom 11
Romania
T +40 374 132 000

Middle East/Africa:
Emerson Energy and Transportation Solutions
Emerson FZE
P.O. Box 17033
Jebel Ali Free Zone – South 2
Dubai U.A.E.
T +971 4 8118100 | F +971 4 8865465

Asia-Pacific:
Emerson Energy and Transportation Solutions
1 Pandan Crescent
Singapore 128461
T +65 6777 8211| F +65 6777 0947

Energy and Transportation Solutions

EMERSON™