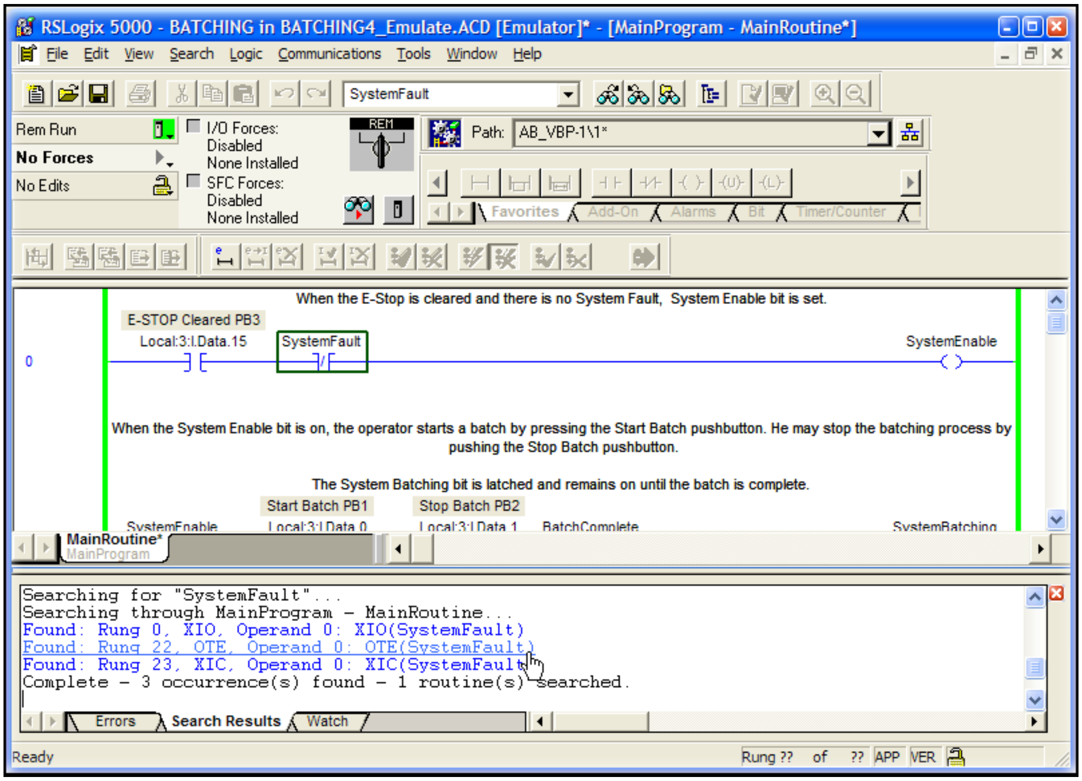
PLC Programming with

RSLogix 5000

How to Program Allen-Bradley ControlLogix and CompactLogix PLCs with Rockwell Automation's RSLogix 5000



When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

The System Batching bit is latched and remains on until the batch is complete.

```
Searching for "SystemFault"...
Searching through MainProgram - MainRoutine...
Found: Rung 0, XIO, Operand 0: XIO(SystemFault)
Found: Rung 22, OTE, Operand 0: OTE(SystemFault)
Found: Rung 23, XIC, Operand 0: XIC(SystemFault)
Complete - 3 occurrence(s) found - 1 routine(s) searched.
```

By Neal Babcock

engineer-and-technician.com

Contents

Introduction

The purpose of this book is to teach you how to set up, program and use an Allen-Bradley ControlLogix or CompactLogix. It will also familiarize you with the parts required for a common application.

It will tell you how to use RSLogix 5000 and how to write a ladder logic program.

Since I feel the best way to learn any programming language is by using a real-world example, there is a sample project included in this book. This sample project, which involves a chemical batching process, also contains a Project Scope. The Project Scope, or Functional Specification, or whatever your company might call it, defines in detail how the system is to operate when the project is finished.

You will learn, step by step, how to take a Project Scope and turn it into a working PLC program.

The book will show you how to go online with your PLC to monitor your program to verify your ladder logic and make sure it is functioning properly.

It will show you how to make changes to your program while you are online.

It will show you the keystrokes and mouse movements that you need to know to use RSLogix 5000.

Finally, it provides a number of tips and a Frequently Ask Questions section that will save you hours of frustration.

This book assumes you have a little background with PLCs – perhaps you have worked with other PLCs from other manufacturers or you have helped to install and wire PLCs. Perhaps you are a Mechanical, Chemical or Process Engineer and you need to learn how to use RSLogix 5000.

If you need a more thorough understanding of basic PLC concepts, you might want to try the Beginner's Guide to PLC Programming *How to Program a PLC (Programmable Logic Controller).* This ebook, along with the online tutorial, provides an example of how to automate a drill press, while explaining all the basic concepts of PLC programming that are necessary to write a solid PLC program.

The Beginner's Guide to PLC Programming works well in conjunction with this book, in that it concentrates on basic PLC programming methods that are common to all types of

PLCs. In addition, it provides an example of machine operation, whereas this book uses the example of a chemical batching process.

The Beginner’s Guide to PLC Programming is available from Modern Media for $9.95. Visit engineer-and-technician.com if you would like to learn more about this book.

PLCs

Nearly all the industrial equipment that you find in a modern manufacturing facility shares one thing in common - computer control. The most commonly used controller is the PLC, or the Programmable Logic Controller, using a programming language called Ladder Logic. The language was developed to make programming easy for people who already understood how switches, relay contacts and coils work. Its format is similar to the electrical style of drawing known as the “ladder diagram”.

The most popular and most widely used manufacturer of PLCs is Rockwell Automation, who produces the Allen-Bradley ControlLogix and CompactLogix series of PLCs. The ControlLogix and CompactLogix families of processors and I/O modules are all programmed using Rockwell’s proprietary software known as RSLogix 5000.

When you are finished with this book, you will be able to sit down in front of any computer running RSLogix 5000 and create a new program. You will be able to edit existing programs. You will be able to professionally document any changes you have made.

Rockwell Automation Technical Support

Unfortunately, we can’t anticipate all the problems you might face as you are troubleshooting a program on the factory floor. There are just too many variables. This is why you must establish a relationship with your local Rockwell Automation technical support team. Get to know them *before* you are in the final stages of a start-up and you run into a problem. They are very helpful and they can save you hours of frustration.

The Rockwell reps are not just technical support personnel; they are skilled engineers that are responsible for running their own projects and writing and troubleshooting their own programs. If you run into a problem, more than likely they have already seen it and have come up with a solution.

Hardware

One of the nice things about Allen-Bradley's smaller PLCs is the relative simplicity of assembling the hardware to create a system.

First, let's see what it takes to assemble a ControlLogix system. You only need to have a few components: a processor, a power supply, a rack and some I/O modules.

ControlLogix Processor

At the time of this writing, there are 15 ControlLogix processors available. For our application, the 1756-L55 processor will be fine.

For your future projects, you will have to consider a number of factors before you make the choice of your processor. Utilize your Rockwell representative and Rockwell's website (www.ab.com) to help you in your choice.

All the processors use RSLogix 5000, so any program you write for one processor could be adapted to run any other 1756 processor.

I/O Modules

For our system, we need discrete inputs, discrete outputs and analog inputs. These modules will work fine for our application:

1756-IA16 Digital AC Input Module (16 discrete inputs)

1756-OA16 Digital AC Outputs (16 discrete outputs)

1756-IF8 Analog Modules (8 single-ended analog inputs)

Software

All ControlLogix and CompactLogix processors use RSLogix 5000 software to program the PLCs. Admittedly, the software is a bit pricey, but in my opinion, it is worth it.

Getting The RSLogix 5000 Software

If you don't have access to a PLC, it would be well worth the effort to download the demo version of RSLogix 5000. The demo runs for 90 days, and has some limitations, but you will be gaining experience with the real thing. Currently, the software is here:

http://www.rockwellautomation.com/rockwellsoftware/design/rslogix5000/demo.html.

There are 7 sections to download, totaling slightly over 480MB. Yes, it's a big job to download and install it, but it is essential.

Ladder Logic

Before we open RSLogix 5000 and start programming, there are a few things you need to know about PLCs in general. I have summarized the basic terms and techniques required to work with ladder logic. It isn't a comprehensive summary, but if you are just starting out, the information presented here will be very helpful.

Every PLC programmer, no matter what skill level, must know the principles described in this section and the Equivalent Logic section. There is simply no way around it.

To effectively write a program, or even edit one, the programmer must know how to visualize the effects of the changes he will make.

In other words, you have to be able to look at the logic "on paper" and imagine how the logic will work when it is entered into the PLC.

The Dialect of PLCs

Lets' define some terms and symbols:

INSTRUCTION – RSLogix's Relay Ladder Logic command language is comprised of "instructions". An XIC (it looks like a normally open contact **--] [--**) is an instruction. A timer is an instruction. A few of the most common instructions are described below.

BIT - an address within the PLC. It can be an input, output or internal coil, among others.

RUNG - A section of the PLC ladder program that terminates in an output function of some type. Just like in an electrical ladder diagram, a rung has some type of output that is turned on or turned off by the preceding entities in the rung. The first rung in a ladder program is always 0.

HARDWIRED INPUT - a physical connection to the PLC from an input device (switch or sensor, etc.).

RSLogix 5000 defines the address of the input, based on the input cards that you configure.

We'll see how this works later on, but here is an example of a hardwired input:

Local:4:I.Data.3

Here is what each part of the address means:

Local:4:I.Data.3
"Local" means that the module is connected to a controller across a backplane or with a parallel link, keeping the module within a few inches of the controller.

Local:**4**:I.Data.3
"4" means that the module is module 4 (located in the 5th slot in the rack).

Local:4:**I**.Data.3
"I" means the bit is an input

Local:4:I.**Data**.3
"Data" indicates the type of data (this is the default for I/O)

Local:4:I.Data.**3**
“3” indicates that the bit is 4th input on the card (the bits start with 0).

By the way, don’t get the capital “I’s” confused with ones.

So, in evaluating our example, we would describe the bit as “Module 4, bit 3”.

Here is where some confusion comes in. Because the Rockwell numbering system starts with 0, and the processor resides in Slot 0, our example bit is actually in slot 5. Our bit 3 is actually the 4th bit. We could also describe the bit as “Slot 5, position 4”.

You will have to learn to transpose these ways of describing a bit back and forth in your head. If you are troubleshooting a problem, and you want someone to look for a signal on our example bit, you might have to tell him to look at the 4th position on the 5th slot. That will lead him to the physical point on the PLC.

However, you need to keep in mind that the corresponding bit in your program will be labeled Local:4:I.Data.3.

It can be confusing, but you will get used to it.

HARDWIRED OUTPUT - a physical connection from the PLC to an output device (relay or pilot light, etc.)

Outputs are addressed the same way.

Local:5:O.Data.4
“Local” means that the module is connected to a controller across a backplane or with a parallel link, keeping the module within a few inches of the controller.

Local:**5**:O.Data.4
“5” means that the module is module 5 (located in the 6th slot in the rack).

Local:5:**O**.Data.4
“O” means the bit is an output

Local:5:O.**Data**.4
“Data” indicates the type of data (this is the default for I/O)

Local:5:O.Data.**4**
“4” indicates that the bit is 5th output on the card (the bits start with 0).

INTERNAL COIL

This is a programmable bit used to simulate a relay within the PLC. The internal coil has no connection to the outside world. It does not connect to an output card. Internal coils are used to store information. The “contacts” of this “relay” can then be used multiple times in other parts of the program.

RSLogix 5000 has greatly simplified the process of describing an internal coil. We can simply give it a name, known as a ***tag***.

For example, if you have an internal coil that is the result of, say, three hardwired safety gate limit switches, we could label the coil “SafetyGatesClosed”.
Note the lack of spaces in the tag name. RSLogix 5000 does not allow spaces, or other special characters, in the tag name.

Some people use underscores, so the tag might be “Safety_Gates_Closed”. Either way is fine; it just depends on what your company or your client prefers.

TIMER

A timer is a programmable instruction that lets you turn on or turn off bits after a preset time.

The two primary types of timers are TON for “timer on delay” and TOF for “timer off delay”.

Timers in RSLogix 5000 use tag names for identification.

COUNTER

A counter is a programmable instruction that lets you turn on or turn off bits after a preset count has been reached.

There are different types of counters available in the RSLogix, but the CTU (counter up) instruction covers everything we will talk about here.

Counters in RSLogix 5000 use tag names for identification.

--] [-- Normally Open Contact

When used with a hardwired input, this instruction is off until there is a voltage applied to the input. The bit address then goes high, or on, and the instruction becomes “true.” It works the same way when it has the same address as an internal coil, except that the coil must be turned on by logic in the program.

Allen-Bradley calls these normally open contacts “XIC”, or “e**X**amine **I**f **C**losed” instruction.

An XIC instruction can reference a hardwired input, a hardwired output, an internal coil or a timer done bit, among others.

--]/[-- Normally Closed Contact
This is an inverted normally open contact.

When used with a hardwired input, this instruction is "true" until there is a voltage applied to the input. It then goes low, or off, and becomes “false.”

It also can be used with an internal coil, becoming true when the coil is off and becoming false when the coil is on.

Allen-Bradley calls these normally closed contacts “XIO”, or “e**X**amine **I**f **O**pen” instructions.

-( )- Output Coil
When used with a hardwired output, this function is off until the logic in the program allows it to turn on. It then becomes “true”, and will energize the device that is wired to the respective output.

If it is used as an internal coil, it will toggle the instructions associated with it. That is, it will close a normally open instruction and open a normally closed instruction.

Allen-Bradley calls these outputs “OTE”, or “**O**utpu**T** **E**nergize”.

An OTE may be used with a hardwired output or an internal coil.

TRUE – A state that indicates an instruction is allowing logic to “flow” through it.

Also, if the logic in a rung turns on the output of the rung, then the rung is said to be true.

FALSE - Without stating the obvious, this is the opposite of true.

OK, that was a lot to cover and for you to understand – don’t worry, this will start getting easier.

Equivalent Logic

Suppose we want to use a PLC to operate a pilot light. In its elementary form, PLC logic is very similar to the hard-wired logic you would find in an electrical ladder diagram.

For example, if you wanted to turn on a light with a momentary pushbutton, you would wire it like the circuit below. When you press PB1, the pilot light PL1 lights up.

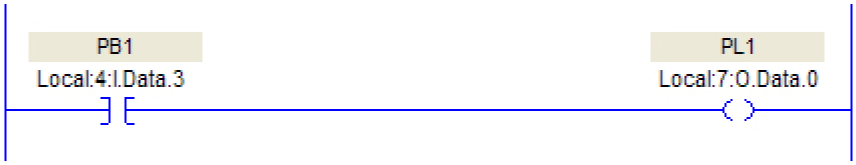
```
H                                              N
|                                        PILOT |
|                                        LIGHT |
|  PB1                                    PL1  |
|---] [-----------------------------------(L)----|
|                                              |
|
```

Now let's do the same thing in a PLC. To duplicate the hardwired circuit on a PLC, you would wire the switch PB1 to an input (let’s use Local:4:I.Data.3) and wire the light PL1 to an output (Local:7:O.Data.0).

The I/O (hardwired inputs and outputs) is set up like this:

- There is a “PB1” pushbutton switch wired to Local:4:I.Data.3 of the PLC.
- There is a “PL1” pilot light wired to Local:7:O.Data.0 of the PLC.

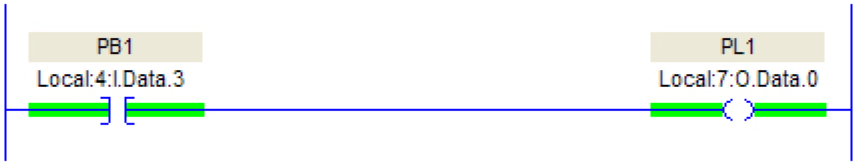
In RSLogix 5000, the screen would look like this.



Now let’s examine the sequence of events. When you first turn on the PLC, the PB1 pushbutton is off, or false. Therefore, the PL1 output is off. Pressing PB1 will make Local:4:I.Data.3 true, Local:7:O.Data.0 will come on and the light will be energized. It will stay on only as long as you hold the button in.

Just like electrical current has to flow through the switch to turn on the light in the hardwired circuit, the logic has to "flow" through the normally open instruction (which is “closed” when you press the switch) of Local:4:I.Data.3 to energize the output that turns on PL1.

The green highlight indicates the instruction, is “on” or “true”.



One nice feature of Allen-Bradley PLCs is that you can document each bit in the program. In the example above, “PB1” is somewhat meaningless on its own. After you add the descriptive text “Start Motor PB1”, things make more sense.

Project Scope

We will use a batching operation as an example. Batching, as you may know, is the term that describes the mixing of assorted ingredients to make a finished product.

There are techniques that are common to batching, whether you are making soap or cake mix. We are going to write a program that mixes a hypothetical window cleaner.

Someone has to define the batching procedure. Usually, this is done by a process engineer or a chemical engineer. If the job of defining the project is done well, a document called a Project Scope (or something similar) is generated.

It is extremely important that you clearly understand the entire process that is defined in the scope. If you have any questions or concerns, you need to resolve those before you begin programming. If you don't, then the responsibility of errors and omissions, and perhaps the blame, may be placed on you.

If you bring up questions that result in changes to the defined sequence of operations, ask the originator to revise the Project Scope. In fact, it is not uncommon for a Project Scope to undergo a number of revisions.

If there is a change that is not documented in the scope, you should document it by getting an email from the originator that explains the change. If nothing else, you want to make sure you understand what the change involves.

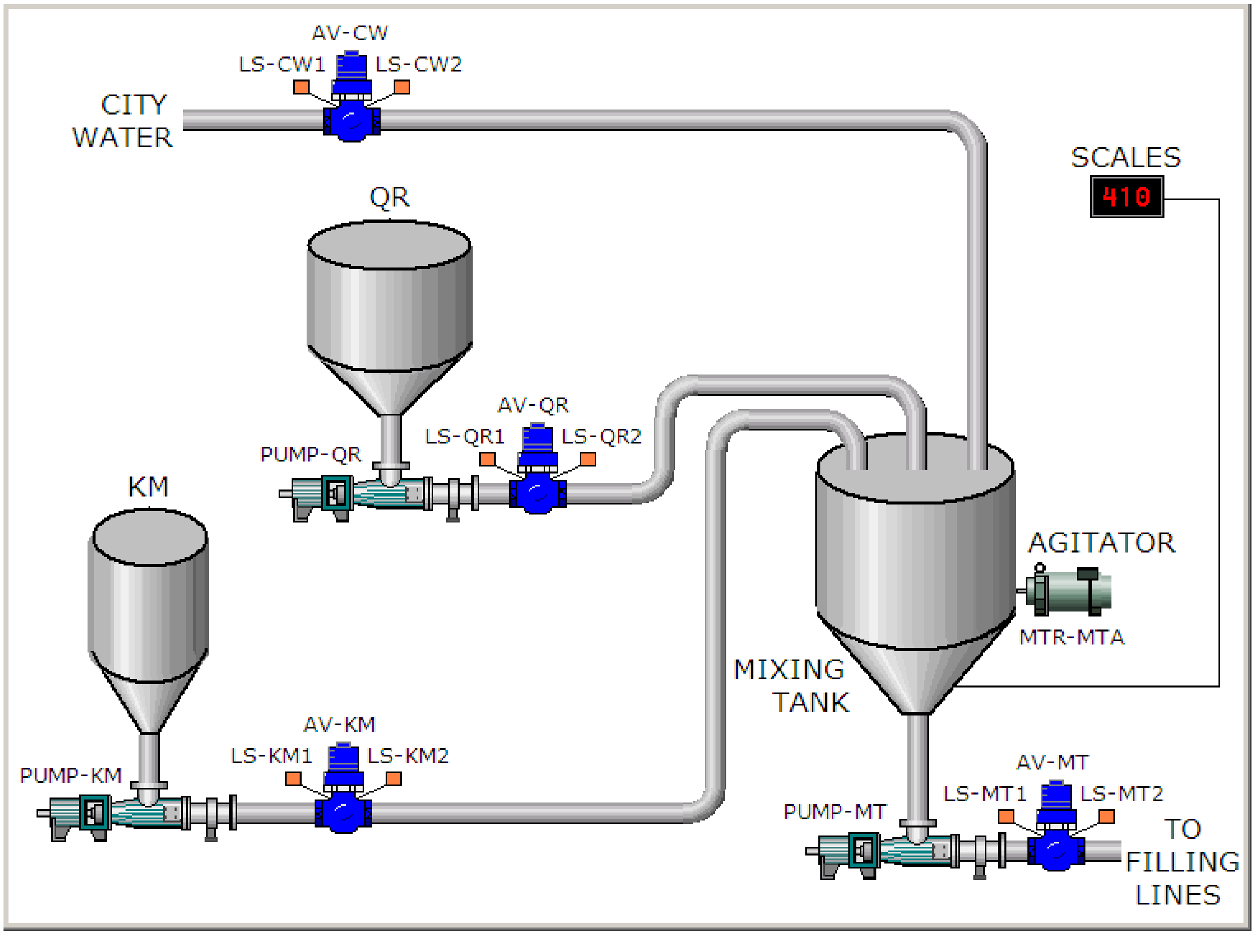
For our project, the project scope is as follows.

Hyper-Glass Cleaner
Batching Project Scope

Goal

The goal of this project is to install a new automated batching system for mixing Hyper-Glass Cleaner.

Overview

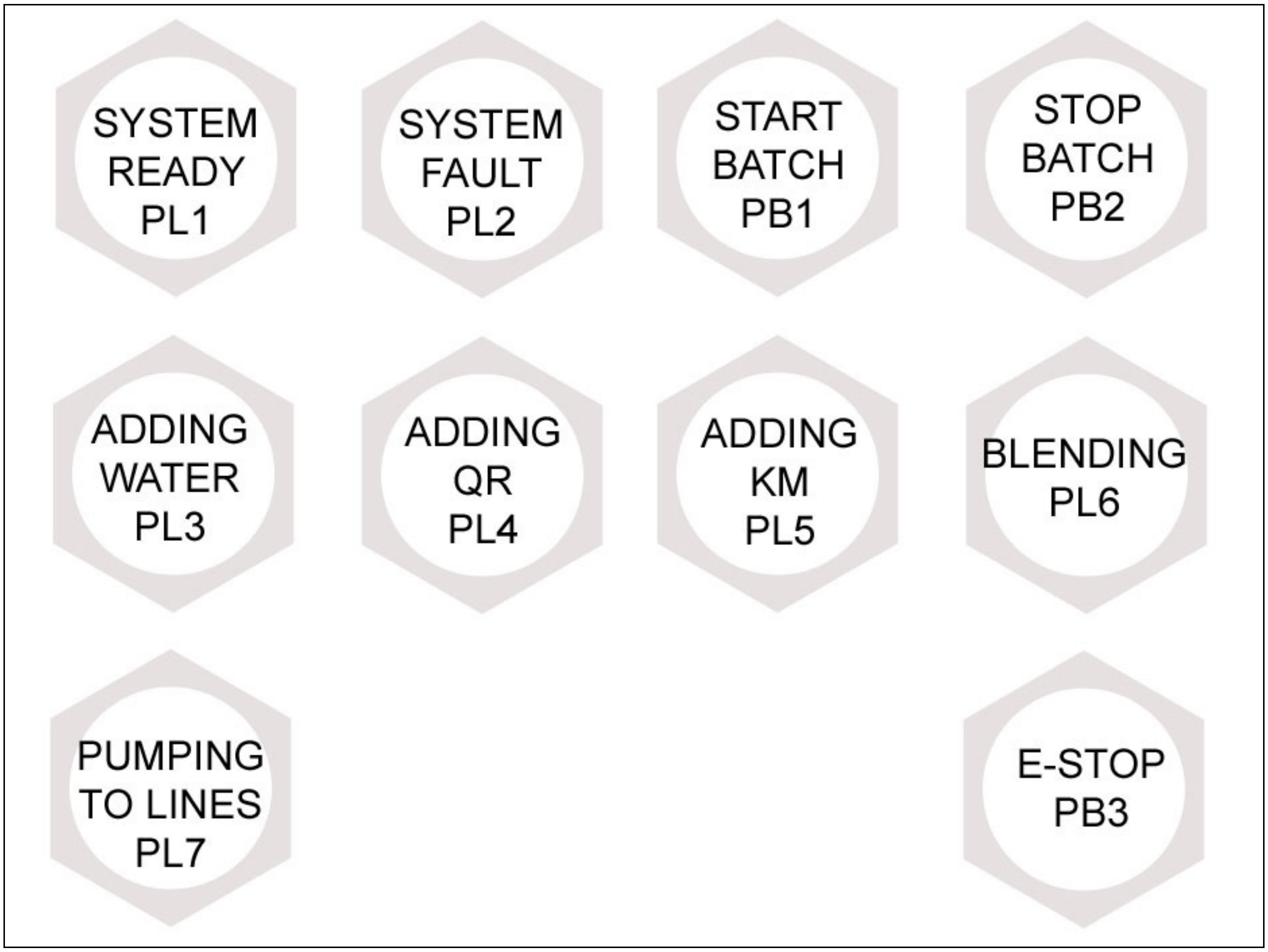


Three ingredients (city water, ingredient QR and ingredient KM) are added in specified amounts by weight to the Mixing Tank. After all the ingredients have been added to the Mixing Tank, the mixture is blended by running the agitator for a given time. When the blending time is complete, the finished product is pumped to the Filling Lines for bottling and final packaging.

System Components

Component	Function
Valve AV-CW	Supplies city water to the Mixing Tank
Limit Switch LS-CW1	Indicates when valve AV-CW is closed
Limit Switch LS-CW2	Indicates when valve AV-CW is open
Pump PUMP-QR	Pumps ingredient QR to the Mixing Tank
Valve AV-QR	Supplies QR to the Mixing Tank
Limit Switch LS-QR1	Indicates when valve AV-QR is closed
Limit Switch LS-QR2	Indicates when valve AV-QR is open
Pump PUMP-KM	Pumps ingredient KM to the Mixing Tank
Valve AV-KM	Supplies KM to the Mixing Tank
Limit Switch LS-KM1	Indicates when valve AV-KM is closed
Limit Switch LS-KM2	Indicates when valve AV-KM is open
Scales	Provides the current weight of the ingredients in the tank to the PLC
Agitator MTR-MTA	Blends the ingredients in the Mixing Tank
Pump PUMP-MT	Pumps ingredient MT from the Mixing Tank
Valve AV-MT	Supplies the finished product to the Filling Lines
Limit Switch LS-MT1	Indicates when valve AV-MT is closed
Limit Switch LS-MT2	Indicates when valve AV-MT is open
Ultrasonic Level Sensor ULS-1	Indicates the level in the Mixing tank

Operator Panel Layout



Operator Panel Components

Component	Function
SYSTEM READY pilot light PL1	Indicates the system is ready for batching
SYSTEM FAULT pilot light PL2	Indicates the system has a fault and is stopped
START BATCH pushbutton switch PB1	Starts a new batch
STOP BATCH pushbutton switch PB2	Stops the batching process
ADDING WATER pilot light PL3	Indicates the system adding water to the Mixing Tank
ADDING QR pilot light PL4	Indicates the system adding ingredient QR to the Mixing Tank
ADDING KM pilot light PL5	Indicates the system adding ingredient KM to the Mixing Tank
BLENDING pilot light PL6	Indicates the system is blending the ingredients
PUMPING TO LINES pilot light PL7	Indicates the system is pumping the batch to the Filling Lines
E-STOP PB3	Immediately stops the entire system

Electrical Specifications

The Ultrasonic Level Sensor ULS-1 provides a 0-10VDC signal to the PLC.

The Scales provide a 0-10VDC signal to the PLC.

All other input signals are 120VAC.

All output signals are 120VAC.

Detailed Sequence of Operations

There are 5 steps in the Batching process:

1. Add City Water
2. Add Ingredient QR
3. Add Ingredient KM
4. Mix the batch
5. Pump the batch to the filling lines

To begin a new batch, the operator will verify that the “SYSTEM READY” pilot light is on and that the Mixing Tank is ready to receive ingredients.

The operator will then press the “START BATCH” pushbutton to begin the batching process. The “SYSTEM READY” pilot light will turn off. No further operator input is required.

Step 1 – City Water
Automatic valve AV-CW will open. The “ADDING WATER” pilot light will illuminate.

Valve AV-CW will remain open until 1275 lbs. of City Water is in the Mixing Tank. Valve AV-CV will close.

The state of AV-CW will be verified by limit switch LS-CW2. If LS-CW2 is not made within 2 seconds after the valve was told to open, a fault will be generated and the system will shut down. The pilot light “SYSTEM FAULT” PL2 will illuminate indicating that a fault has occurred.

LS-CW1 will verify that the valve is closed within 2 seconds after the valve was told to close. If the valve closure is not verified within 2 seconds, a fault will be generated, the system will shut down and PL2 will illuminate.

All valves and their respective limit switches will work in the manner described above.

After the City Water has been added, valve AV-CW will close and the “ADDING WATER” pilot light will turn off.

Step 2 – Ingredient QR
Valve AV-QR will be opened. After the valve position has been verified by LS-QR2, PUMP-QR will pump 390 lbs. of ingredient QR into the Mixing Tank. The “ADDING QR” pilot light will be illuminated while the pump is running.

After the ingredient QR has been added to the Mixing Tank, PUMP-QR stops and the “ADDING QR” pilot light will turn off. Valve AV-QR will close.

Step 3 – Ingredient KM
Valve AV-KM will be opened. After the valve position has been verified by LS-KM2, PUMP-KM will pump 173 lbs. of ingredient KM into the mixing tank. The “ADDING KM” pilot light will be illuminated while the pump is running.

After the ingredient KM has been added to the Mixing Tank, valve AV-KM will close. PUMP-KM will stop. The “ADDING KM” pilot light will turn off.

After LS-KM1 indicates the valve has been closed, the agitator motor MTR-MTA will start. The “BLENDING” pilot light will illuminate.

Step 4 – Mixing
The agitator will run for 3 minutes. The "BLENDING" pilot light will illuminate.

After the agitator is finished, The "BLENDING" pilot light will turn off.

Step 5 – Pump to filling lines
Valve AV-MT will open. After LS-MT1 indicates the valve is open, the "PUMPING TO LINES" pilot light will illuminate.

PUMP-MT will pump the entire batch to the filling lines. When the Ultrasonic Level Sensor ULS-1 indicates that the tank is empty, PUMP-MT will turn off, valve AV-MT will close and the batching cycle is complete. The "PUMPING TO LINES" pilot light will turn off and the "SYSTEM READY" pilot light will illuminate.

During every phase of the batching process, the liquid level must be monitored by the PLC. If the level rises to greater than 95% of that Mixing tank's capacity, the system will generate a fault and the batching process must be halted.

The operator may press the "E-STOP" pushbutton PB3 to stop the process at any time.

END OF HYPER-GLASS CLEANER BATCHING PROJECT SCOPE

Summarizing the Scope

So, what did we get from the scope? Let's summarize:

First, 1275 lbs. of water will be added to the Mixing Tank. Then, 390 lbs. of QR will be added. The last ingredient is KM, of which we will add 173 lbs.

After all the ingredients are in the Mixing Tank, we have to blend it for 3 minutes.

After the batch is blended, we will pump the finished product in the tank to the filling lines.

We have to make sure all the valves open or close in less than 2 seconds. If they do not, then we need to shut down the process.

We need to turn on the appropriate pilot lights to indicate what stage the batching process is in.

We need to make sure the level in the Mixing Tank doesn’t get too high. If it does, we must shut down everything.

We need to make sure that the respective valves for the pumps are open before we turn on the pumps.

Which PLC?

There are certainly a number of factors that will determine which PLC you need. Without getting into all of those, let’s just say that the 1756-L55 processor has plenty of processing power for this project and the cost is reasonable, so we will use one.

Before you can determine what modules, rack or power supply you need to buy, you will have to know what your I/O requirements are. This involves the very critical step of laying out your I/O.

A bit of advice here: Don’t skimp on this step. Make sure the I/O is right before you begin programming. A mistake or omission here will cost you ten-fold further down the road.

Lay Out The I/O

Now we need to layout the I/O. This will tell us the addresses for the I/O points, what PLC modules we need and how the PLC modules need to be wired.

There are three types of signals in the batching system: 120VAC digital inputs (limit switches and pushbutton switches) 120VAC digital outputs[1] (valves, motors and pilot lights) and analog 0-10VDC inputs.

List all of the components in the system that are connected to the PLC. Categorize each component according its type (digital input, digital output or analog 0-10VDC). It is best

[1] Technically, the valves themselves are not 120VAC devices, but in this case, the solenoids that subsequently drive the valves are. Likewise, the motors that run the pumps and the agitators may not be 120VAC, but the control circuitry that operates the motors is 120VAC.

to do this in an Excel spreadsheet. I have provided one for this project – it is called IO_List.xls and is included in the files you downloaded.

Try to keep associated devices together. For example, the “ADDING WATER” pilot light should be near Valve AV-CW. This will make the electrical prints easier to read and also help to keep the PLC program organized.

Notice the “Descriptor” column. This is a statement providing a shorthand description of the device when the associated input is on, or true. We will use these descriptors in the actual PLC program.

I can’t stress how important it is to get the verbiage right in a descriptor. For example, let’s look at LS-CW1. This particular limit switch is normally open, but held closed when the valve is closed.

When the limit switch is closed, the input to the PLC will be on.

If we used the descriptor

Limit Switch
LS-CW1

that wouldn’t tell us too much without referring to the prints. Plus, it is a little redundant, as we know it is a limit switch based on the “LS” prefix in the device name.

If, however, we use the descriptor

City Water
Valve AV-CW
Closed
LS-CW1

then that tells us immediately, without referring to the prints, that the City Water valve is closed as indicated by the limit switch LS-CW1.

After you go online with a PLC, if an input is energized (when used with a normally open instruction), the symbol for the bit is highlighted. You can quickly realize the descriptor statement is currently true.

TIP This is a good time to call your local Allen-Bradley representative and have him assist you in selecting the parts you need. He can work directly from your I/O listing and probably save you a bunch of time.

Now we must determine what input and output modules we need.

The Project Scope said that the Scales and the Ultrasonic Level Sensor provide 0-10VDC signals. We can use an Allen-Bradley 1756-IF8 Analog Module.

For the inputs, we can use the 1756-IA16 Digital AC Input Module. Since 11 inputs are needed for the system, this card will provide 5 spares.

For the outputs, we will use a 1756-OA16 Digital AC Outputs Module.

We still need a rack to hold our processor, the I/O cards and a power supply. We are not going concern ourselves here with the rack or the power supply, as this doesn’t have much effect on our programming. Suffice to say, try to select components that will provide the space and flexibility for future expansion.

Assigning I/O Addresses

Here is the final layout for the cards in the rack:

Slot 0 – 1756-L55 processor
Slot 1 – 1756-IF8 Analog Modules (8 single-ended analog inputs)
Slot 2 – 1756-IA16 Digital AC Input Module (16 discrete inputs)

Slot 3 – 1756-OA16 Digital AC Outputs (16 discrete outputs)

Please refer to the I/O List spreadsheet and you will see how the I/O has been assigned.

A final note about the I/O list – take the time to do it right and keep it updated as the project progresses.

Running RSLogix

To run RSLogix, click:

Start > All Programs > Rockwell Software > RSLogix 5000 Enterprise Series > RSLogix 5000

A quick side note about conventions used in this book:

We are going to use the format shown above to indicate what menu items you should click on as you navigate the menus and sub-menus.

For example, the line above means:

Click on “Start”.

Click on “All Programs”.

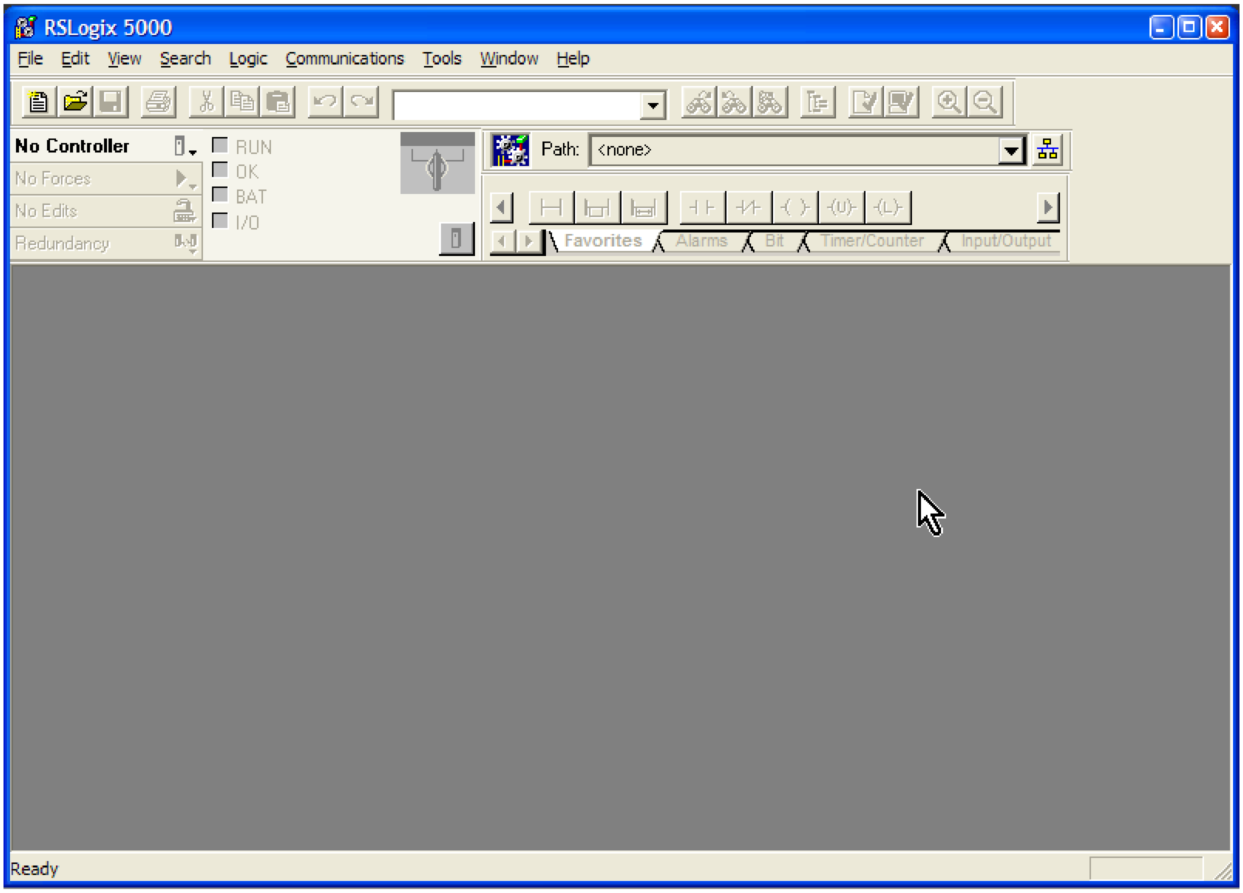
Click on “Rockwell Software”.

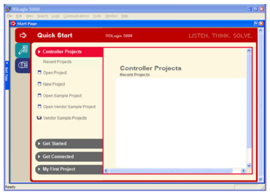
Click on “RSLogix 5000 Enterprise Series”.

Click on “RSLogix 5000”.

Your path to start RSLogix, depending on the version you have installed, may be slightly different.

You will see this on your monitor.





TIP

By default, RSLogix 5000 displays a “Start Page” every time the program is started. Most people don’t use this and choose to turn it off.

Select Tools > Options and uncheck “Show Start Page on Start Up”.

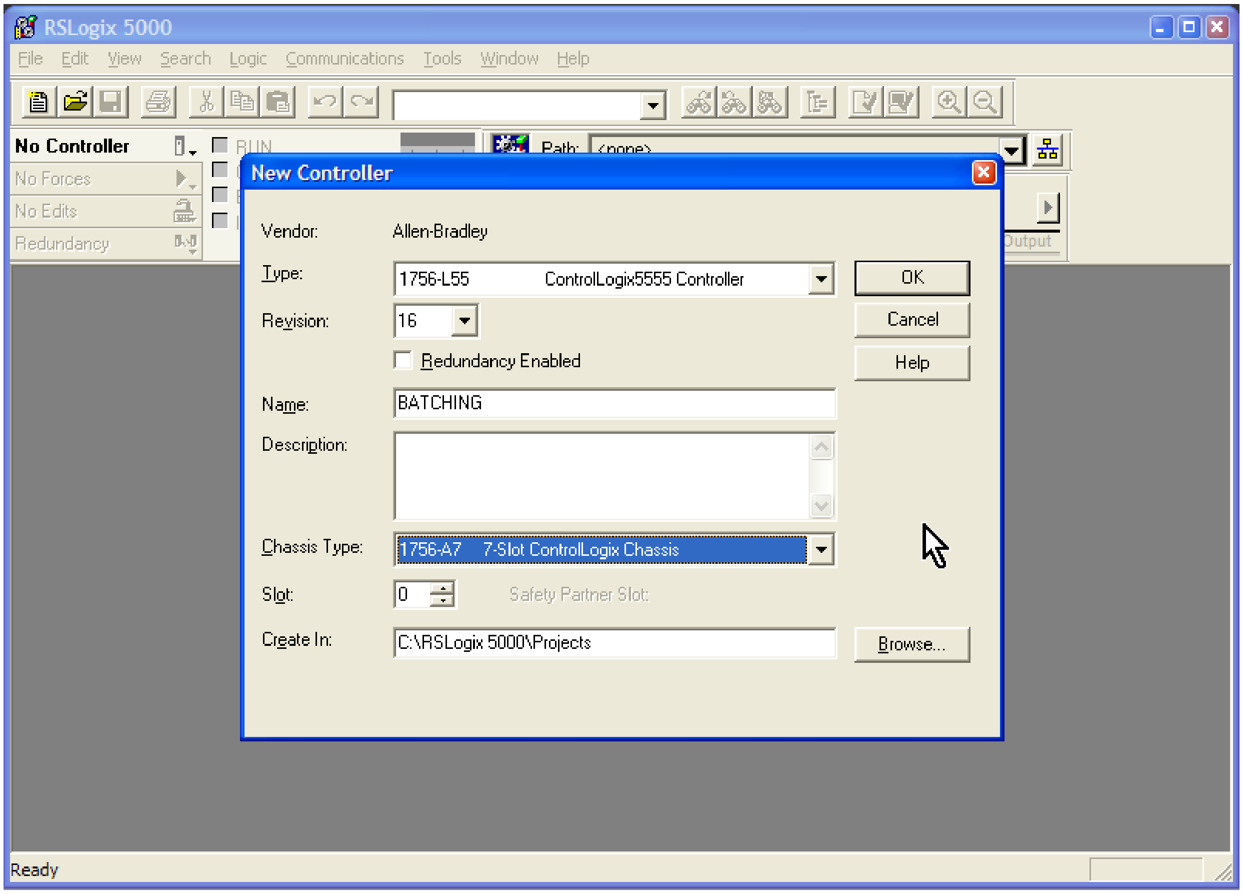
To open a new programming, choose

File > New

In the Type: dropdown, select “1756-L55”.

In the Name field, type in “BATCHING”.

In the Chassis Type: dropdown, select the seven slot “1756-A7” chassis.



Vendor: Allen-Bradley

Type: 1756-L55 ControlLogix5555 Controller

Revision: 16

□ Redundancy Enabled

Name: BATCHING

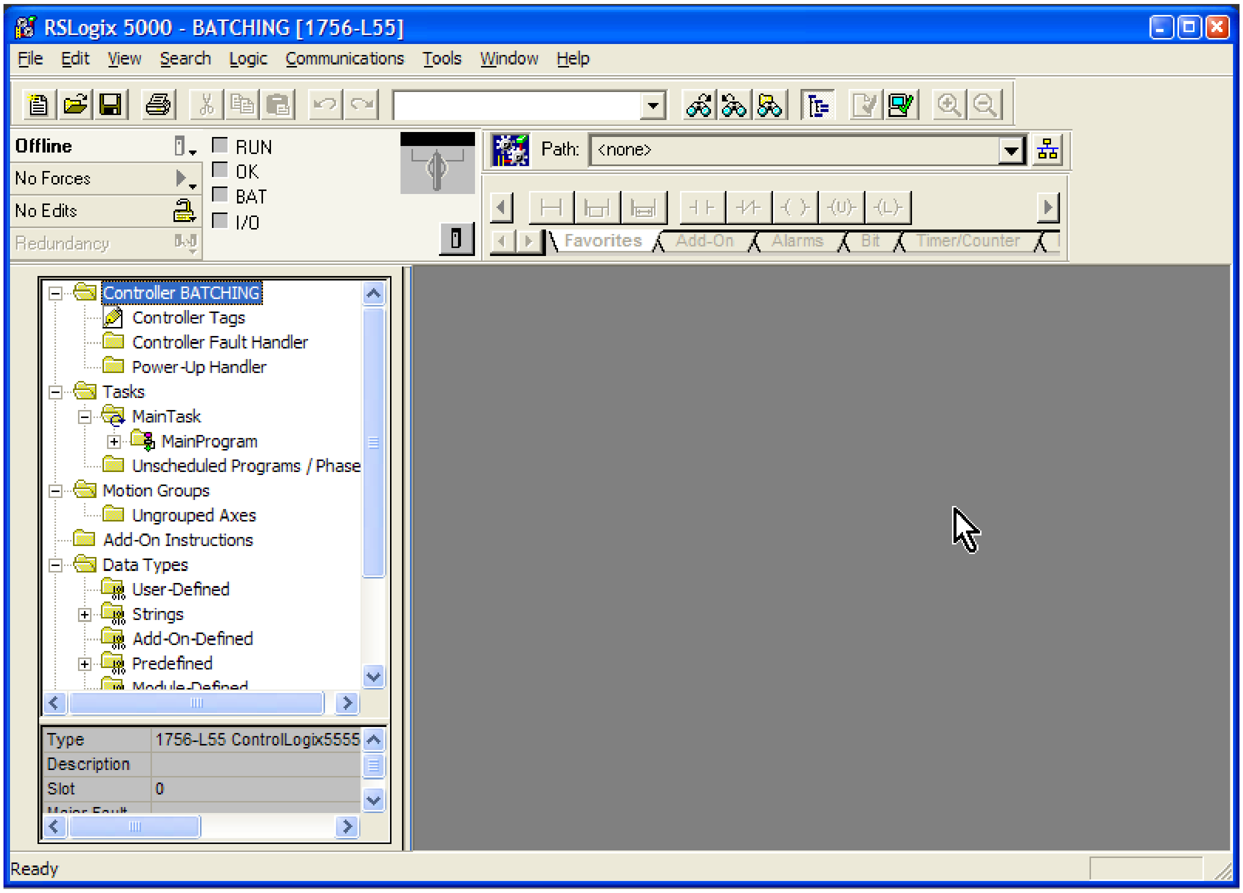
Description:

Chassis Type: 1756-A7 7-Slot ControlLogix Chassis

Slot: 0

Create In: C:\RSLogix 5000\Projects

Click “OK” and this screen will appear.



Path: <none>

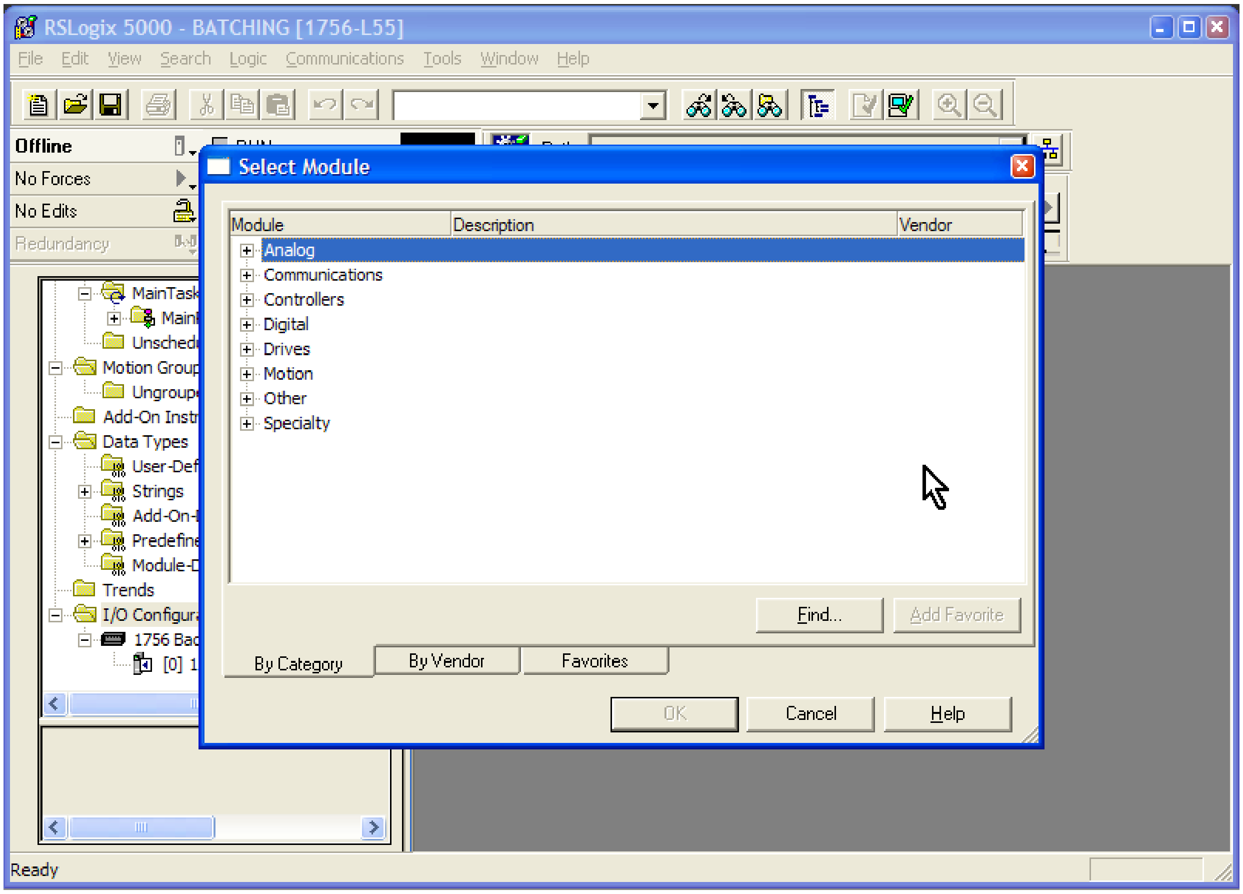
Type	1756-L55 ControlLogix5555
Description	
Slot	0

On the left, you see an explorer-type menu. This is called the Controller Organizer. All of these folders and files allow you to configure or view properties of the PLC or data files within the PLC.

TIP

Tip: You can toggle the Controller Organizer by pressing ALT-0

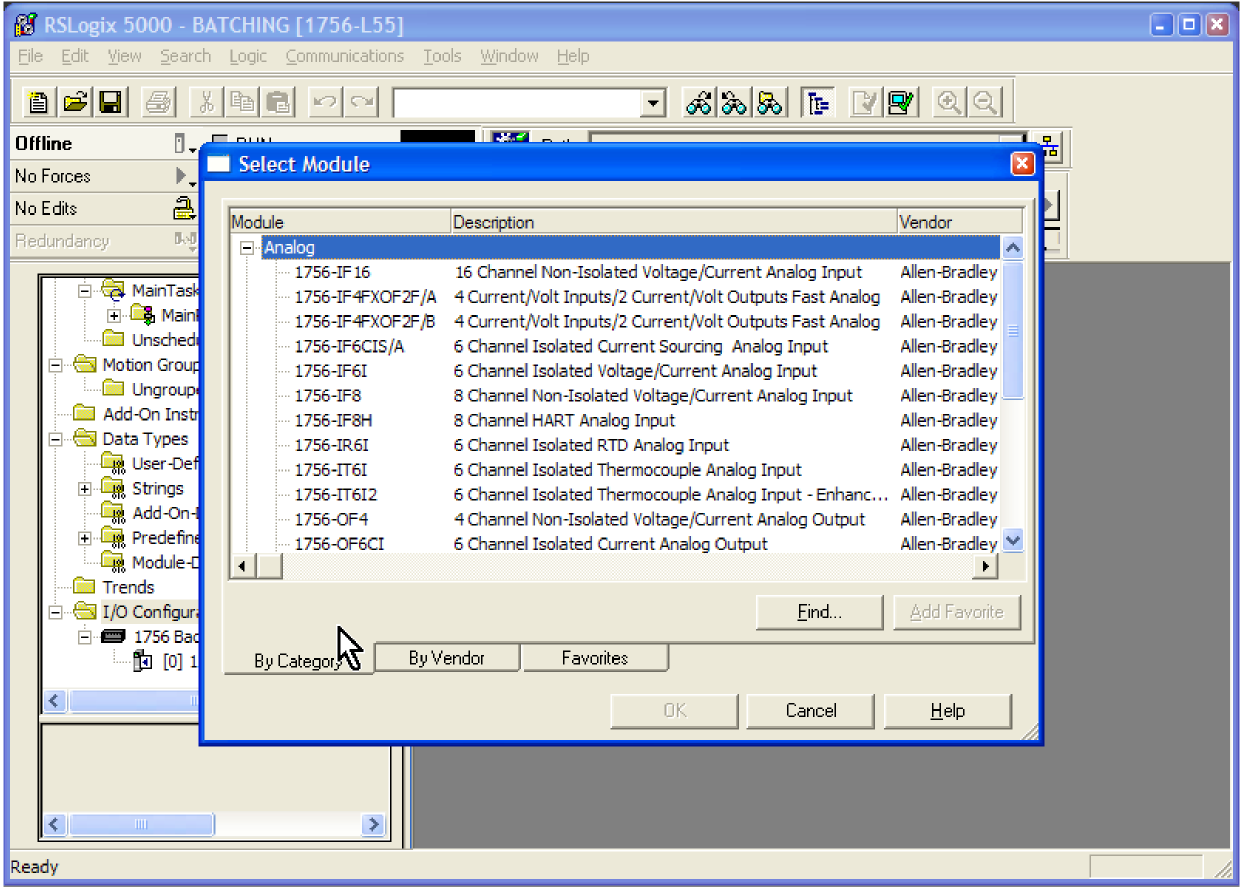
The first thing we need to do is configure the I/O. Scroll down in the Controller and right-click on “I/O Configuration”. Choose “New Module”.



Select Module

Module	Description	Vendor
Analog		
Communications		
Controllers		
Digital		
Drives		
Motion		
Other		
Specialty		

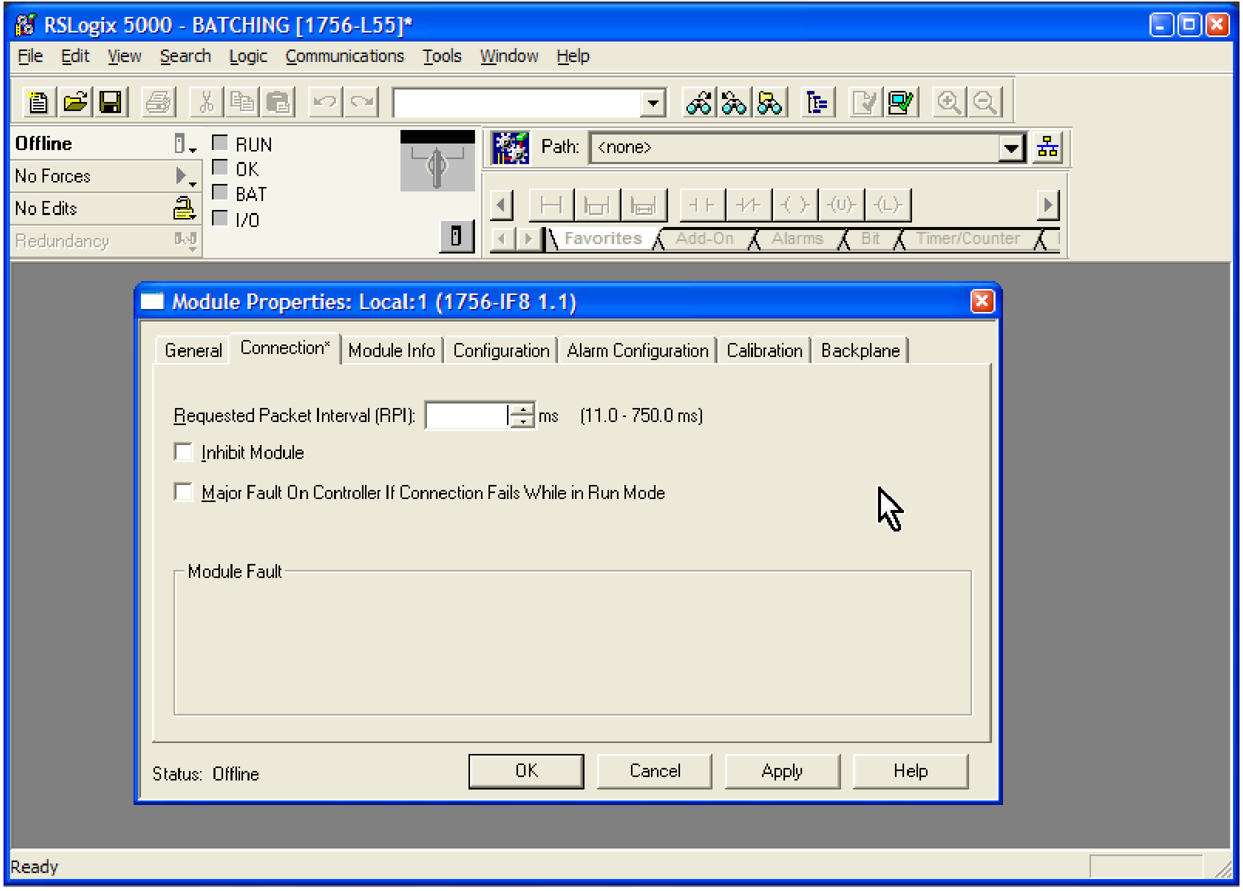
Expand the “Analog” section.



Module	Description	Vendor
⊟ Analog		
1756-IF16	16 Channel Non-Isolated Voltage/Current Analog Input	Allen-Bradley
1756-IF4FXOF2F/A	4 Current/Volt Inputs/2 Current/Volt Outputs Fast Analog	Allen-Bradley
1756-IF4FXOF2F/B	4 Current/Volt Inputs/2 Current/Volt Outputs Fast Analog	Allen-Bradley
1756-IF6CIS/A	6 Channel Isolated Current Sourcing Analog Input	Allen-Bradley
1756-IF6I	6 Channel Isolated Voltage/Current Analog Input	Allen-Bradley
1756-IF8	8 Channel Non-Isolated Voltage/Current Analog Input	Allen-Bradley
1756-IF8H	8 Channel HART Analog Input	Allen-Bradley
1756-IR6I	6 Channel Isolated RTD Analog Input	Allen-Bradley
1756-IT6I	6 Channel Isolated Thermocouple Analog Input	Allen-Bradley
1756-IT6I2	6 Channel Isolated Thermocouple Analog Input - Enhanc...	Allen-Bradley
1756-OF4	4 Channel Non-Isolated Voltage/Current Analog Output	Allen-Bradley
1756-OF6CI	6 Channel Isolated Current Analog Output	Allen-Bradley

Select “1756-IF8” and click “OK”.

Click “OK” on the next dialog box and the “Module Properties” window appears.



Module Properties: Local:1 (1756-IF8 1.1)

Requested Packet Interval (RPI): ms (11.0 - 750.0 ms)

□ Inhibit Module

□ Major Fault On Controller If Connection Fails While in Run Mode

Module Fault

Status: Offline

This window allows you to configure many aspects of the analog card, including:

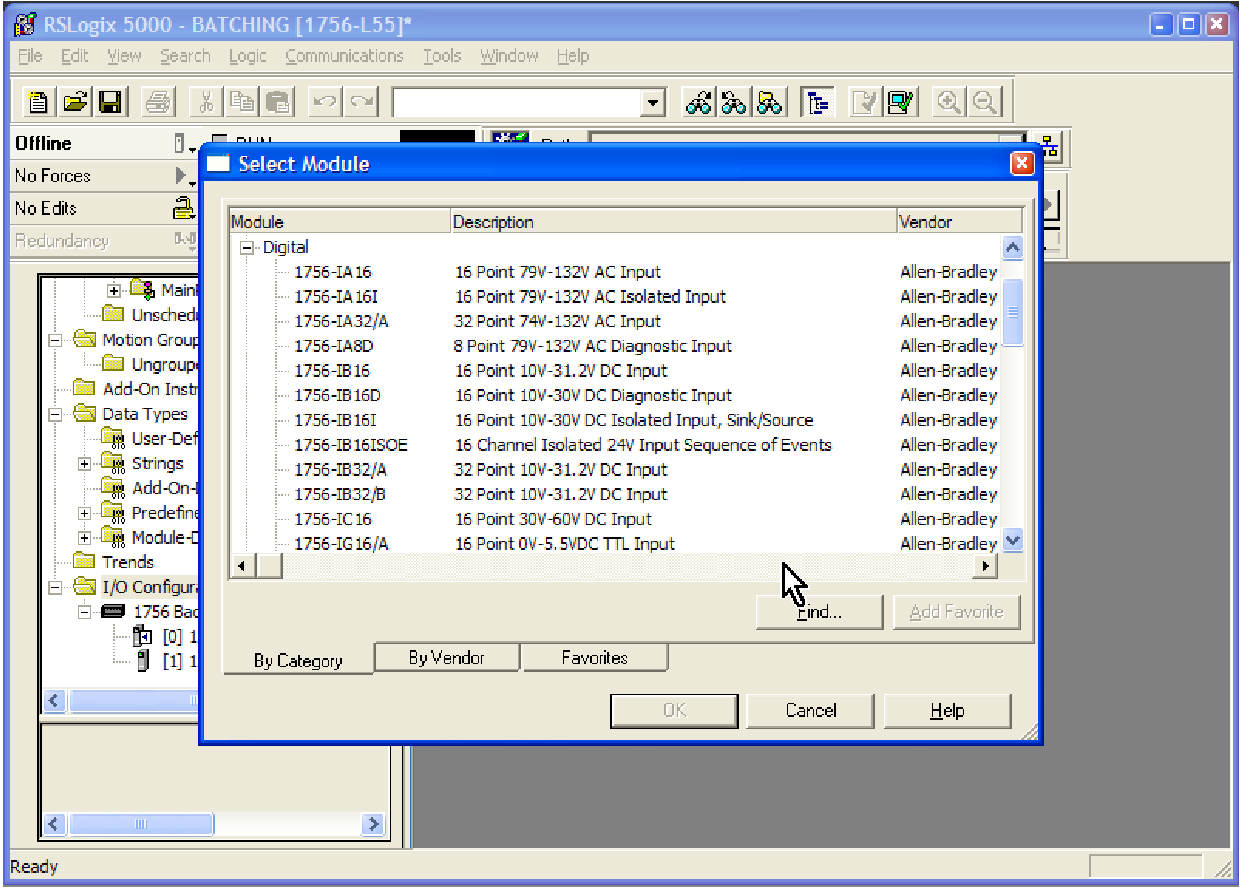
- scaling (per channel)
- input range (voltage or current, per channel)
- alarm configuration (per channel)
- calibration gain and offset (per channel)

We will adjust these later; for the time being click “OK” to accept the default values.

Let’s add the discrete input card. Right-click on “I/O Configuration” and select “New Module”.

Make sure that the module is assigned to Slot 1.

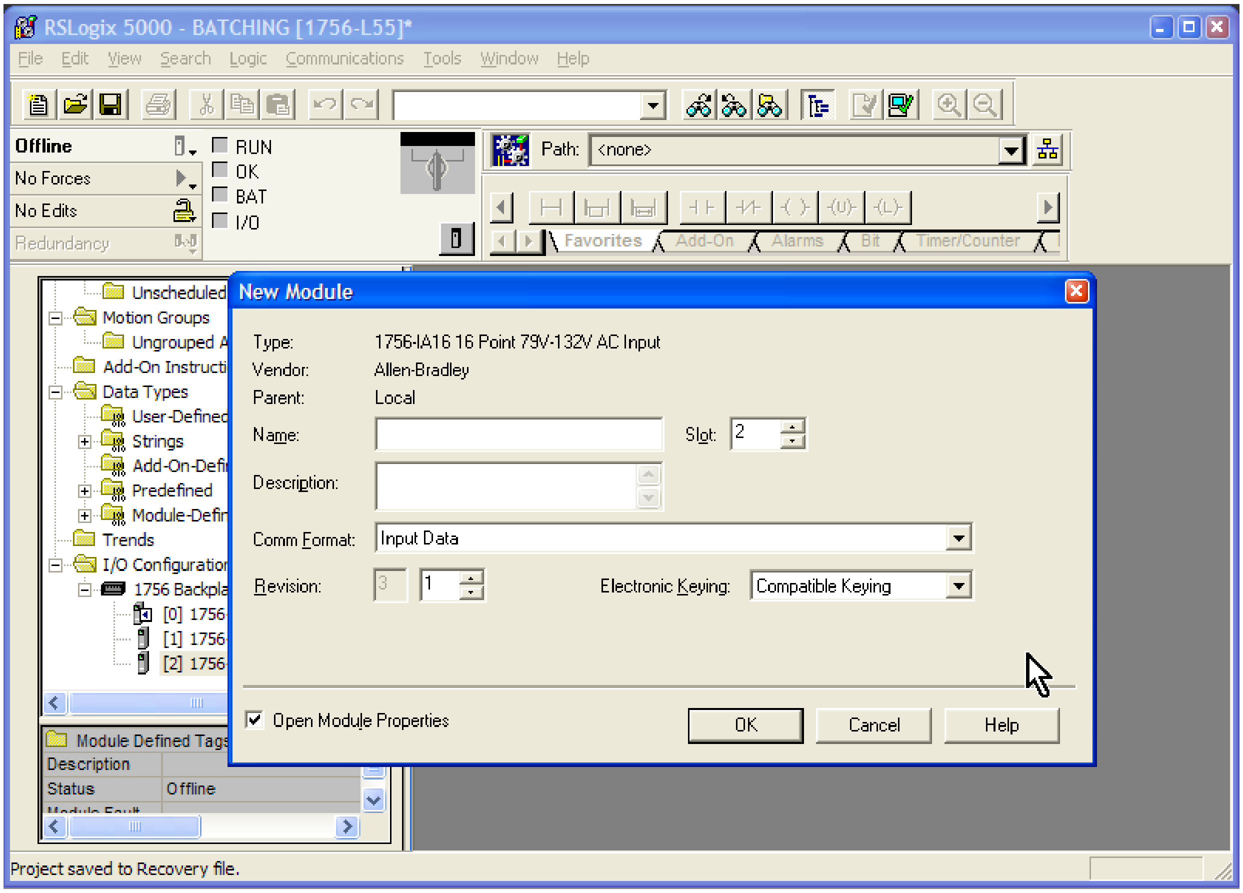
Expand the “Digital” section.



Module	Description	Vendor
1756-IA16	16 Point 79V-132V AC Input	Allen-Bradley
1756-IA16I	16 Point 79V-132V AC Isolated Input	Allen-Bradley
1756-IA32/A	32 Point 74V-132V AC Input	Allen-Bradley
1756-IA8D	8 Point 79V-132V AC Diagnostic Input	Allen-Bradley
1756-IB16	16 Point 10V-31.2V DC Input	Allen-Bradley
1756-IB16D	16 Point 10V-30V DC Diagnostic Input	Allen-Bradley
1756-IB16I	16 Point 10V-30V DC Isolated Input, Sink/Source	Allen-Bradley
1756-IB16ISOE	16 Channel Isolated 24V Input Sequence of Events	Allen-Bradley
1756-IB32/A	32 Point 10V-31.2V DC Input	Allen-Bradley
1756-IB32/B	32 Point 10V-31.2V DC Input	Allen-Bradley
1756-IC16	16 Point 30V-60V DC Input	Allen-Bradley
1756-IG16/A	16 Point 0V-5.5VDC TTL Input	Allen-Bradley

Choose “1756-IA16”.

A dialog box appears asking you to select the major revision. Select the default. Make sure Slot 2 is selected and click “OK”.



Description

Status Offline

Project saved to Recovery file.

New Module

Type: 1756-IA16 16 Point 79V-132V AC Input

Vendor: Allen-Bradley

Parent: Local

Name:

Slot: 2

Description:

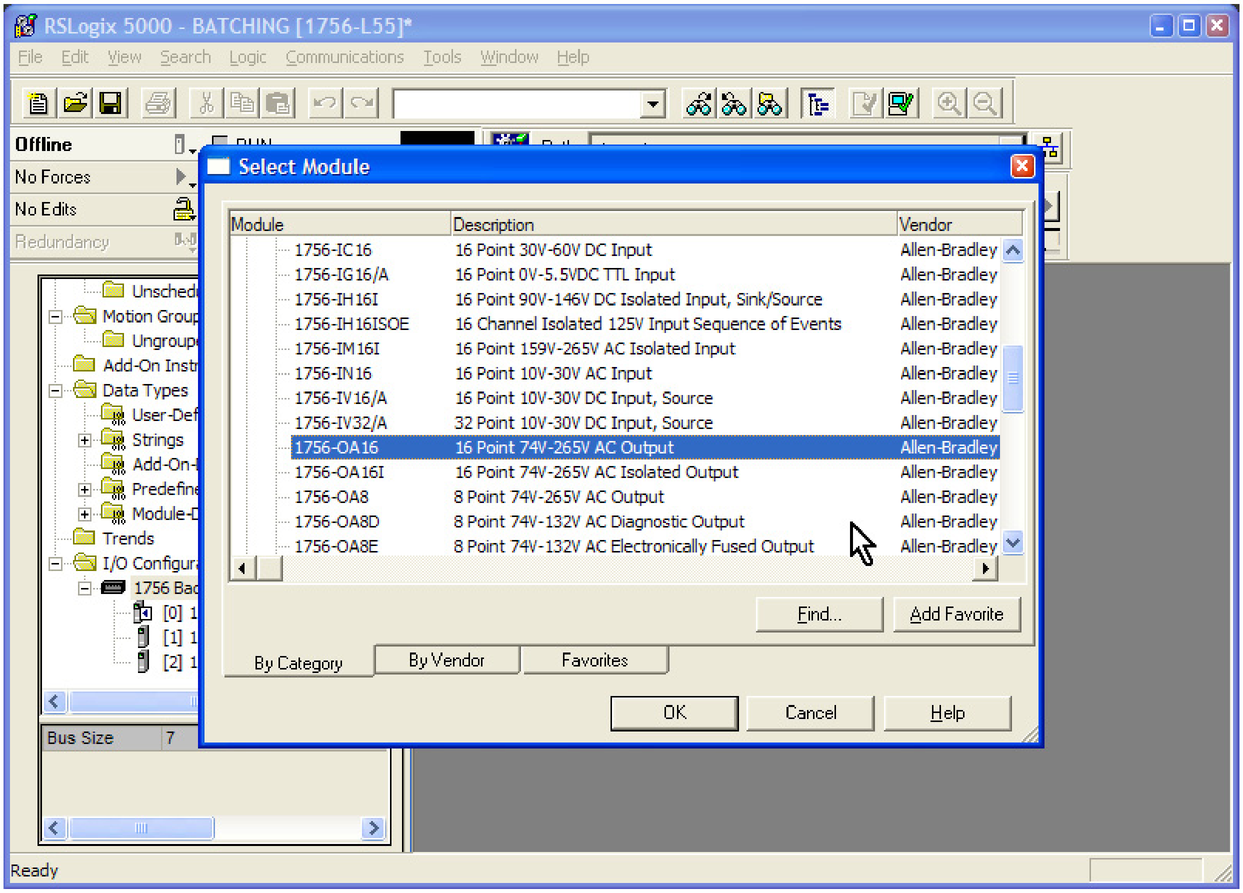
Comm Format: Input Data

Revision: 3 1

Electronic Keying: Compatible Keying

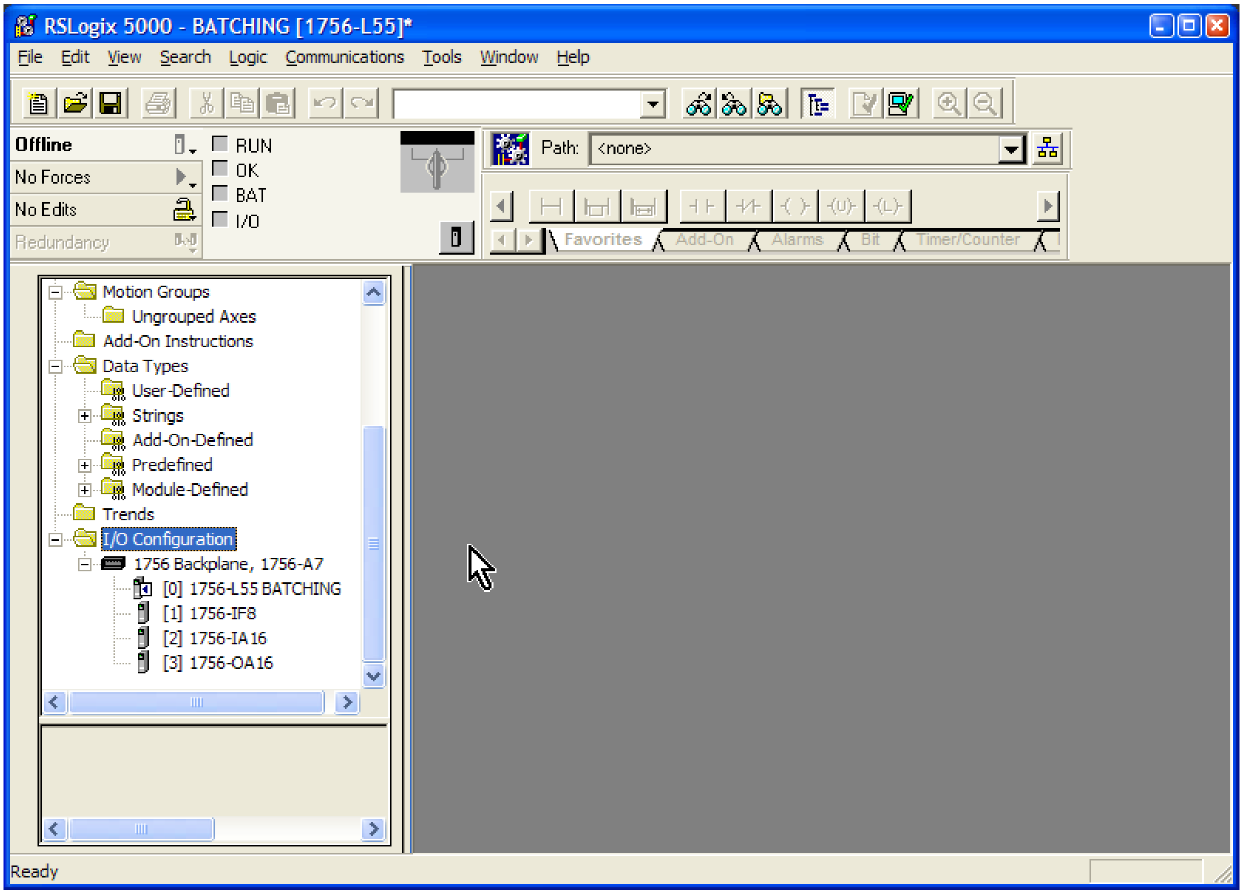
☑ Open Module Properties

Now we can add the discrete output card to complete our I/O configuration. Add a 1756-OA16.



Module	Description	Vendor
1756-IC16	16 Point 30V-60V DC Input	Allen-Bradley
1756-IG16/A	16 Point 0V-5.5VDC TTL Input	Allen-Bradley
1756-IH16I	16 Point 90V-146V DC Isolated Input, Sink/Source	Allen-Bradley
1756-IH16ISOE	16 Channel Isolated 125V Input Sequence of Events	Allen-Bradley
1756-IM16I	16 Point 159V-265V AC Isolated Input	Allen-Bradley
1756-IN16	16 Point 10V-30V AC Input	Allen-Bradley
1756-IV16/A	16 Point 10V-30V DC Input, Source	Allen-Bradley
1756-IV32/A	32 Point 10V-30V DC Input, Source	Allen-Bradley
1756-OA16	16 Point 74V-265V AC Output	Allen-Bradley
1756-OA16I	16 Point 74V-265V AC Isolated Output	Allen-Bradley
1756-OA8	8 Point 74V-265V AC Output	Allen-Bradley
1756-OA8D	8 Point 74V-132V AC Diagnostic Output	Allen-Bradley
1756-OA8E	8 Point 74V-132V AC Electronically Fused Output	Allen-Bradley

The Controller Organizer should look like this.



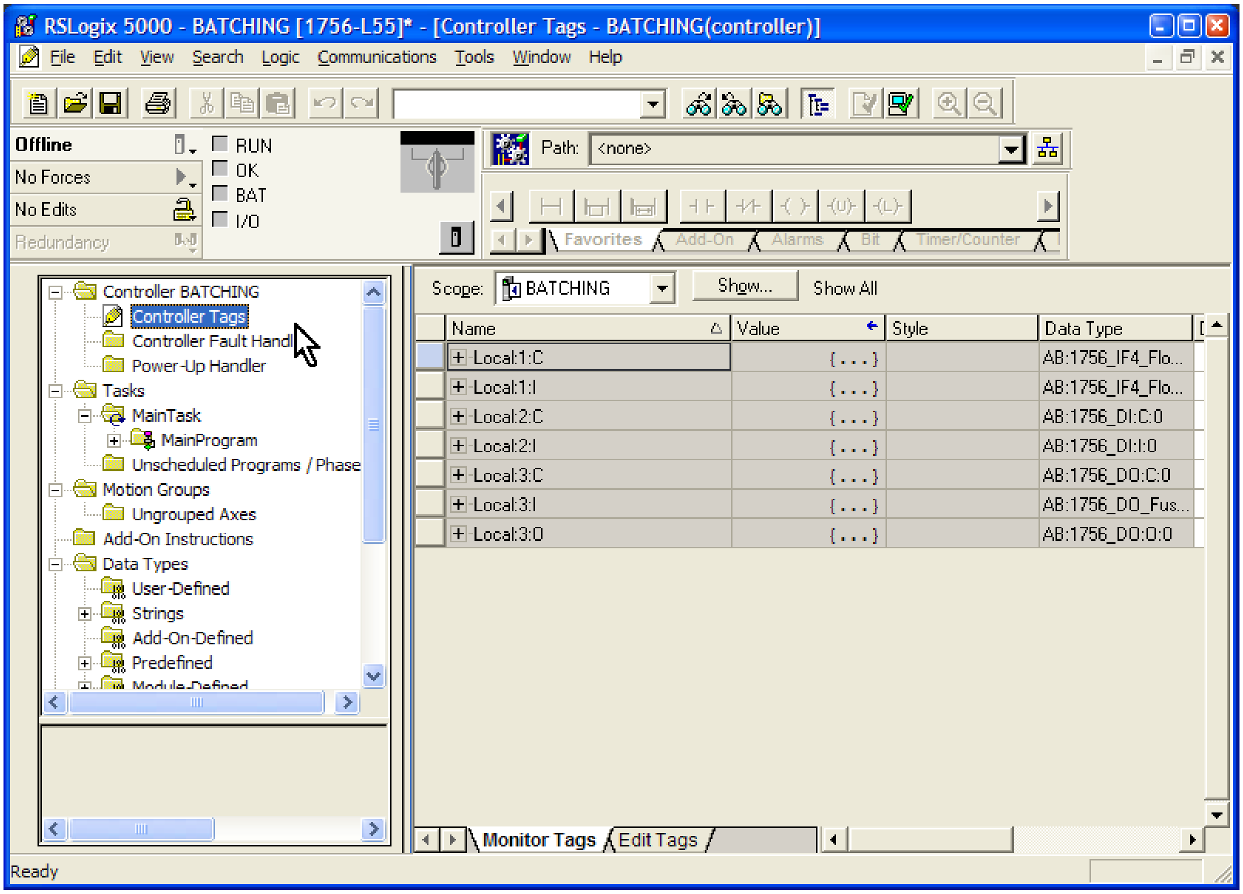
I have intentionally glossed over many of the configuration options for these cards, as I don’t want to get bogged down in these right now. In most cases, the default configuration for the cards will work just fine.

Later in the book, however, we will come back to the analog card to perform the necessary setup.

Tags

Rockwell introduced the concept of tags with RSLogix 5000. All of the addresses in the processor are tag based.

Scroll to the top of the Controller Organizer window and expand “Controller BATCHING”. Click on “Controller Tags” and this screen appears.



Name	Value	Style	Data Type
+ Local:1:C	{...}		AB:1756_IF4_Flo...
+ Local:1:I	{...}		AB:1756_IF4_Flo...
+ Local:2:C	{...}		AB:1756_DI:C:0
+ Local:2:I	{...}		AB:1756_DI:I:0
+ Local:3:C	{...}		AB:1756_DO:C:0
+ Local:3:I	{...}		AB:1756_DO_Fus...
+ Local:3:O	{...}		AB:1756_DO:O:0

Notice that all of the I/O cards we added are now listed in the Controller Tags section.

Local:1 is the analog input card.

Local:2 is the discrete input card.

Local:3 is the discrete output card.

There are different sections of tags for each card.

The “C” suffix (Local:1:**C**) stands for “configuration”.

The “I” (Local:1:**I**) stands for “input”.

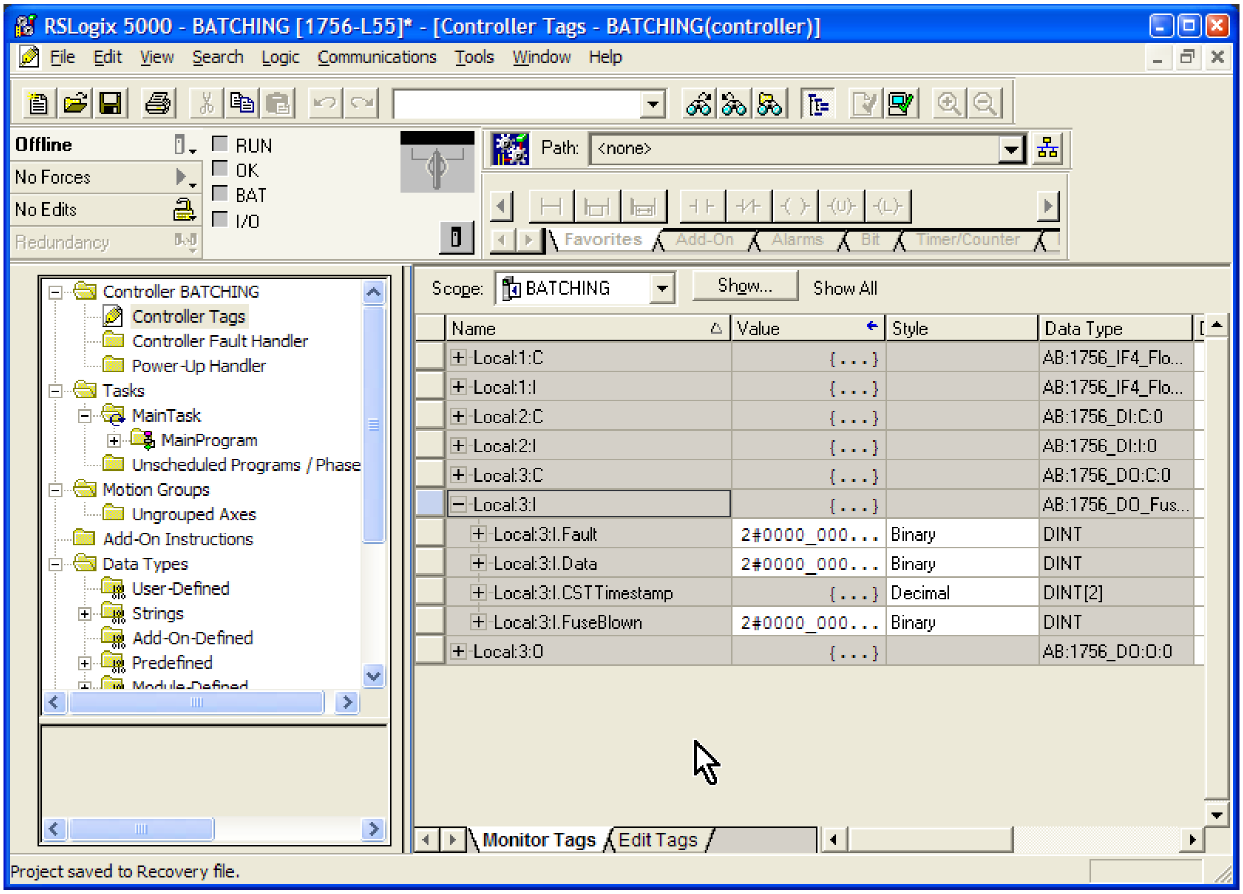
The “O” (Local:3:**O**) stands for “output”.

Each card has its own configuration section. These sections cover the parameters that we saw when we first added the cards.

As you might expect, the analog input card and the discrete input card have an “I” section. The discrete output has an “O” section.

Why, though, does the discrete output have an “I” section?

Expanding Local:3:I show us that this card returns information regarding faults, data, timestamps and fuse status.



Scope: BATCHING

Name	Value	Style	Data Type
+ Local:1:C	{...}		AB:1756_IF4_Flo...
+ Local:1:I	{...}		AB:1756_IF4_Flo...
+ Local:2:C	{...}		AB:1756_DI:C:0
+ Local:2:I	{...}		AB:1756_DI:I:0
+ Local:3:C	{...}		AB:1756_DO:C:0
− Local:3:I	{...}		AB:1756_DO_Fus...
+ Local:3:I.Fault	2#0000_000...	Binary	DINT
+ Local:3:I.Data	2#0000_000...	Binary	DINT
+ Local:3:I.CSTTimestamp	{...}	Decimal	DINT[2]
+ Local:3:I.FuseBlown	2#0000_000...	Binary	DINT
+ Local:3:O	{...}		AB:1756_DO:O:0

Project saved to Recovery file.

This information can be used in the program to help troubleshoot a problem.

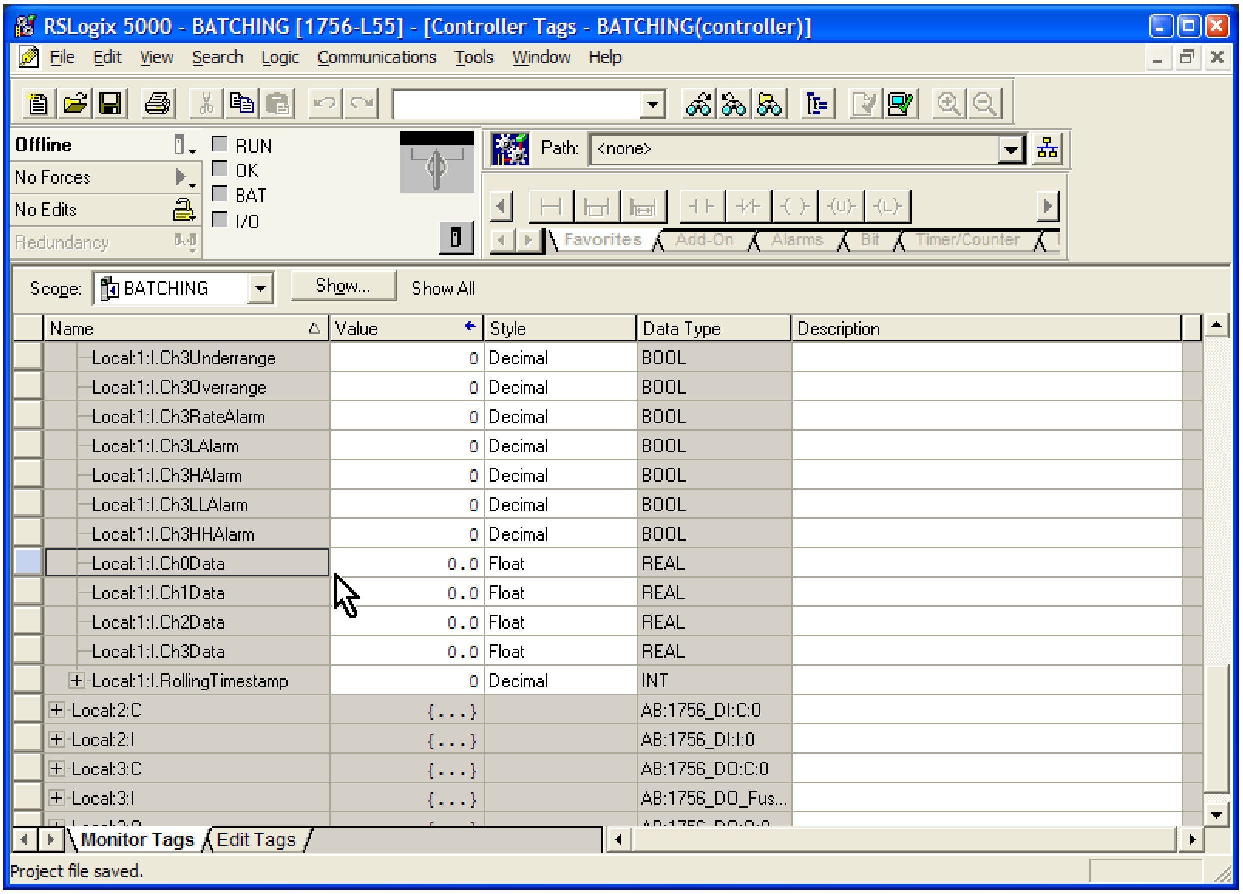
Adding Descriptors To Your I/O

A descriptor is the text that is associated with a tag. We could add a descriptor to every tag, but that may not be cost effective. We do, however, want to add descriptors to our I/O points.

In the controller tag window, expand "Local:1:I. Scroll down to the tag named Local:1:I.Ch0Data. *This tag holds the actual value of the signal on the first channel (channel 0) of the analog input card.*

Keep this in mind as you troubleshoot a ControlLogix processor. This is one place you can look to see if you are getting a signal at the input of the card.

Let's hide the Controller Organizer (ALT-0) for a moment and maximize the Controller Tags window. The screen looks like this.



RSLogix 5000 - BATCHING [1756-L55] - [Controller Tags - BATCHING(controller)]

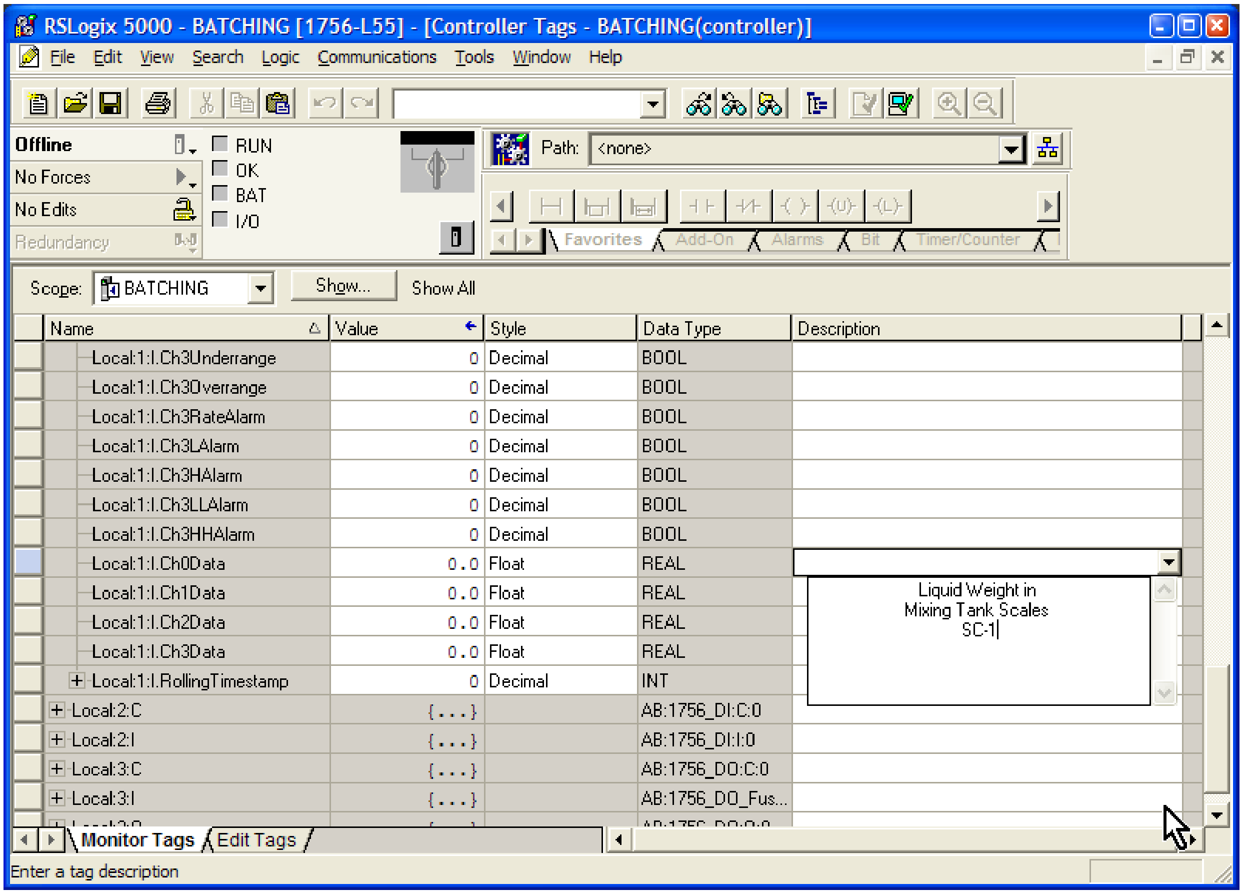
Scope: BATCHING

Name	Value	Style	Data Type	Description
Local:1:I.Ch3Underrange	0	Decimal	BOOL	
Local:1:I.Ch3Overrange	0	Decimal	BOOL	
Local:1:I.Ch3RateAlarm	0	Decimal	BOOL	
Local:1:I.Ch3LAlarm	0	Decimal	BOOL	
Local:1:I.Ch3HAlarm	0	Decimal	BOOL	
Local:1:I.Ch3LLAlarm	0	Decimal	BOOL	
Local:1:I.Ch3HHAlarm	0	Decimal	BOOL	
Local:1:I.Ch0Data	0.0	Float	REAL	
Local:1:I.Ch1Data	0.0	Float	REAL	
Local:1:I.Ch2Data	0.0	Float	REAL	
Local:1:I.Ch3Data	0.0	Float	REAL	
+ Local:1:I.RollingTimestamp	0	Decimal	INT	
+ Local:2:C	{...}		AB:1756_DI:C:0	
+ Local:2:I	{...}		AB:1756_DI:I:0	
+ Local:3:C	{...}		AB:1756_DO:C:0	
+ Local:3:I	{...}		AB:1756_DO_Fus...	

Project file saved.

Open your I/O List spreadsheet and find the description for Local:0:I.Ch0Data. It is "Liquid Weight in Mixing Tank Scales SC-1". Copy the text (CTRL-C)

Click on the description column for Local:0:I.Ch0Data and paste the description into the box.

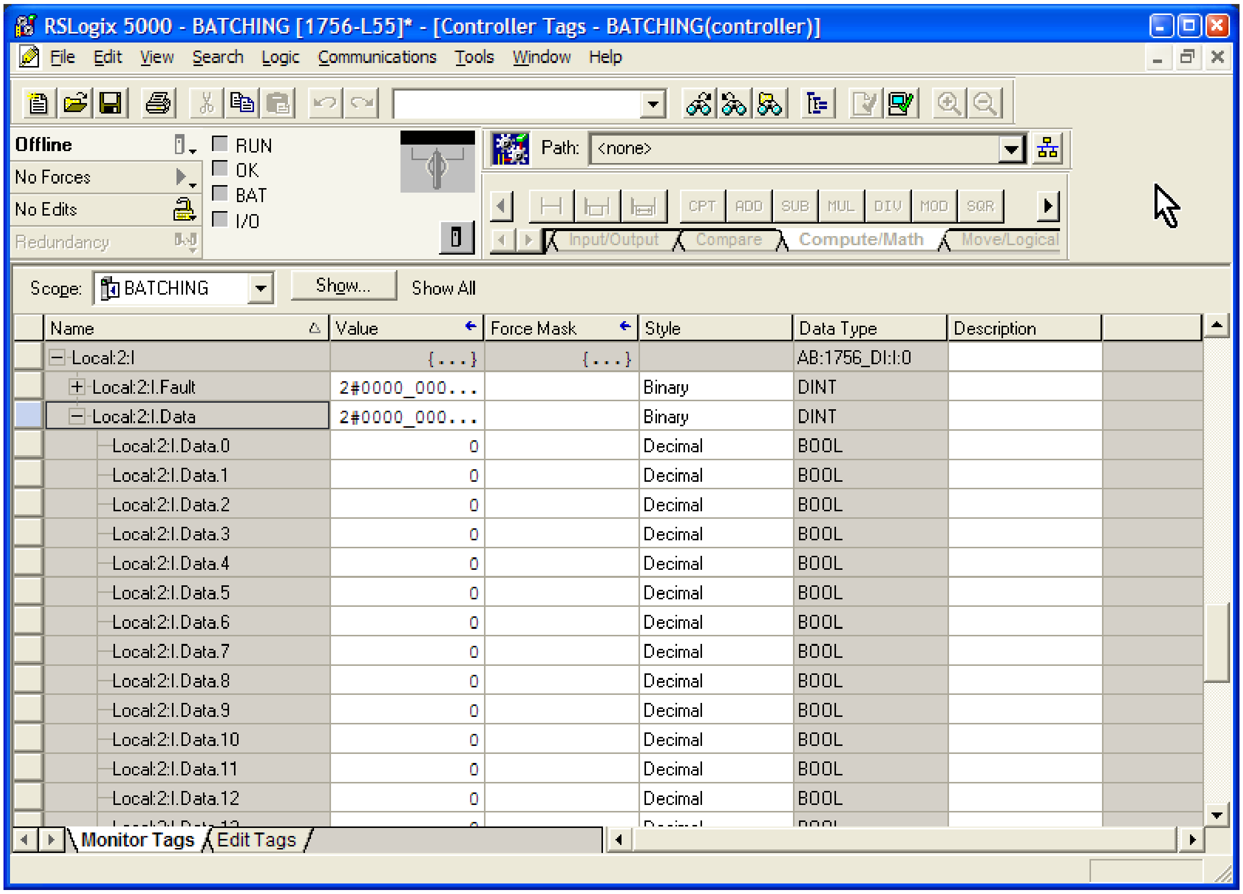


Name	Value	Style	Data Type	Description
Local:1:I.Ch3Underrange	0	Decimal	BOOL	
Local:1:I.Ch3Overrange	0	Decimal	BOOL	
Local:1:I.Ch3RateAlarm	0	Decimal	BOOL	
Local:1:I.Ch3LAlarm	0	Decimal	BOOL	
Local:1:I.Ch3HAlarm	0	Decimal	BOOL	
Local:1:I.Ch3LLAlarm	0	Decimal	BOOL	
Local:1:I.Ch3HHAlarm	0	Decimal	BOOL	
Local:1:I.Ch0Data	0.0	Float	REAL	Liquid Weight in Mixing Tank Scales SC-1
Local:1:I.Ch1Data	0.0	Float	REAL	
Local:1:I.Ch2Data	0.0	Float	REAL	
Local:1:I.Ch3Data	0.0	Float	REAL	
+ Local:1:I.RollingTimestamp	0	Decimal	INT	
+ Local:2:C	{...}		AB:1756_DI:C:0	
+ Local:2:I	{...}		AB:1756_DI:I:0	
+ Local:3:C	{...}		AB:1756_DO:C:0	
+ Local:3:I	{...}		AB:1756_DO_Fus...	

Enter a tag description

Repeat the process for the remaining analog inputs.

Let’s do the same with the descriptors for the first discrete input card. Expand “Local:2I”. Expand Local:2:I.Data.

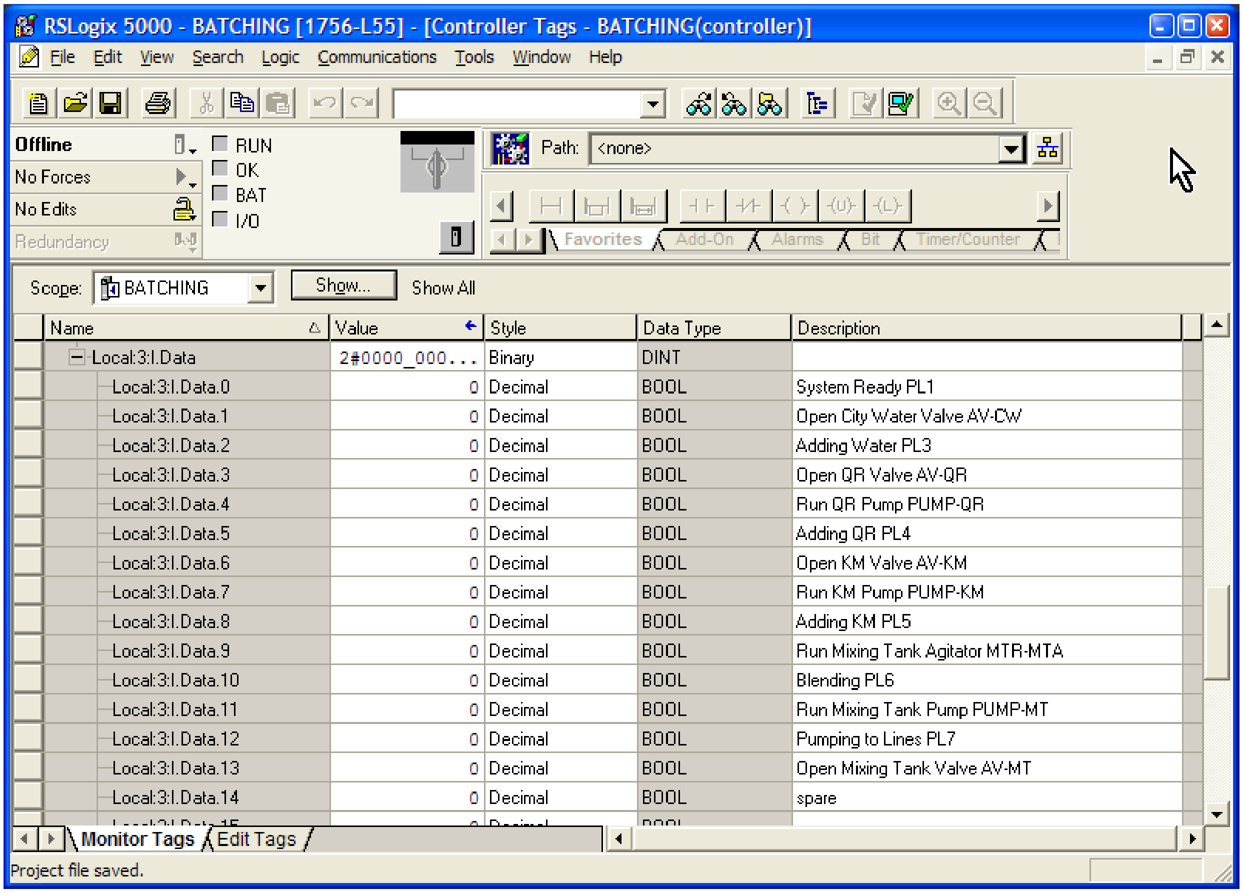

RSLogix 5000 - BATCHING [1756-L55]* - [Controller Tags - BATCHING(controller)]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
CPT ADD SUB MUL DIV MOD SQR
Input/Output Compare Compute/Math Move/Logical
Show...
Show All
Monitor Tags
Edit Tags

Scope: BATCHING

Name	Value	Force Mask	Style	Data Type	Description
Local:2:I	{...}	{...}		AB:1756_DI:I:0	
Local:2:I.Fault	2#0000_000...		Binary	DINT	
Local:2:I.Data	2#0000_000...		Binary	DINT	
Local:2:I.Data.0	0		Decimal	BOOL	
Local:2:I.Data.1	0		Decimal	BOOL	
Local:2:I.Data.2	0		Decimal	BOOL	
Local:2:I.Data.3	0		Decimal	BOOL	
Local:2:I.Data.4	0		Decimal	BOOL	
Local:2:I.Data.5	0		Decimal	BOOL	
Local:2:I.Data.6	0		Decimal	BOOL	
Local:2:I.Data.7	0		Decimal	BOOL	
Local:2:I.Data.8	0		Decimal	BOOL	
Local:2:I.Data.9	0		Decimal	BOOL	
Local:2:I.Data.10	0		Decimal	BOOL	
Local:2:I.Data.11	0		Decimal	BOOL	
Local:2:I.Data.12	0		Decimal	BOOL	

These are the tags that define the actual inputs on the card. Like with the analog card, this is where you look to see if you have a signal on an input.

Copy the descriptions from the spreadsheet for this card.



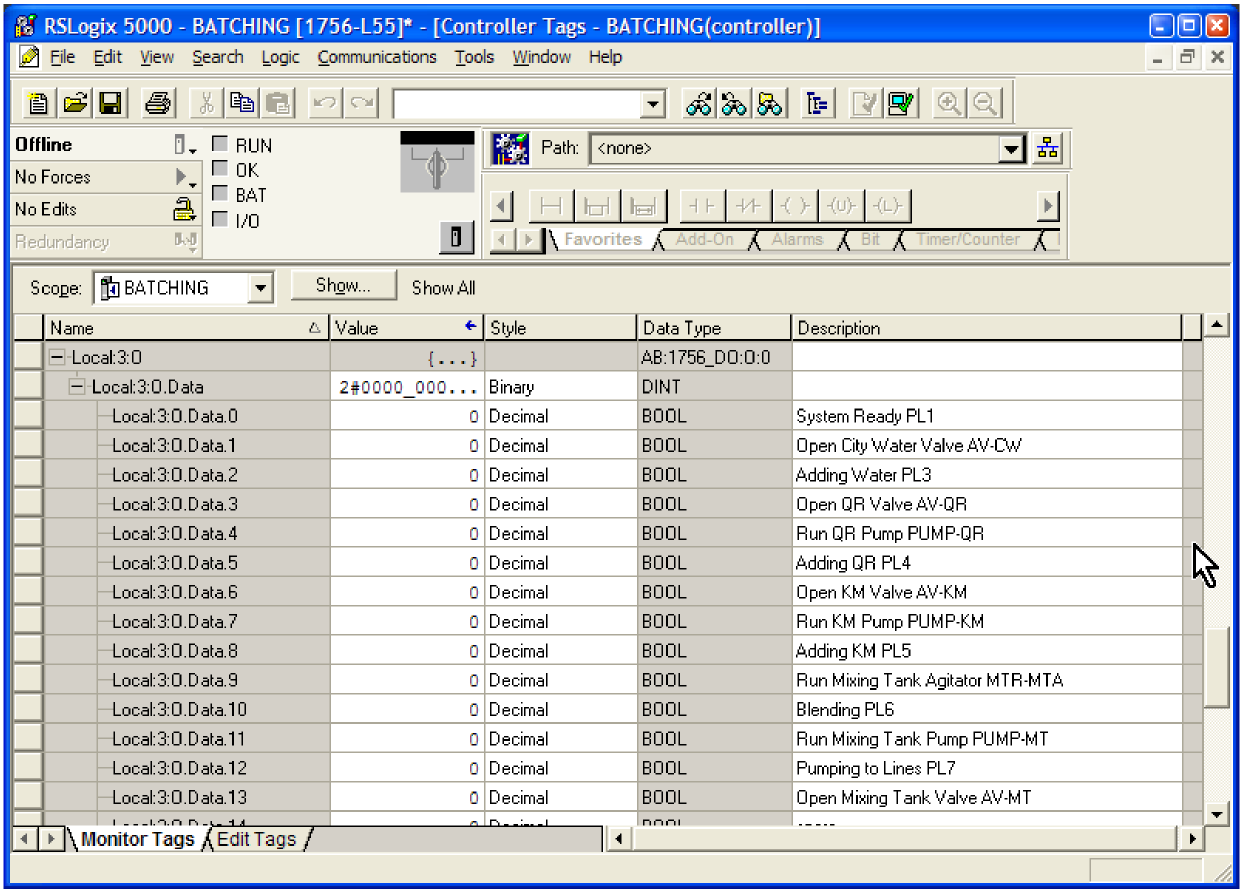
Name	Value	Style	Data Type	Description
Local:3:I.Data	2#0000_000...	Binary	DINT	
Local:3:I.Data.0	0	Decimal	BOOL	System Ready PL1
Local:3:I.Data.1	0	Decimal	BOOL	Open City Water Valve AV-CW
Local:3:I.Data.2	0	Decimal	BOOL	Adding Water PL3
Local:3:I.Data.3	0	Decimal	BOOL	Open QR Valve AV-QR
Local:3:I.Data.4	0	Decimal	BOOL	Run QR Pump PUMP-QR
Local:3:I.Data.5	0	Decimal	BOOL	Adding QR PL4
Local:3:I.Data.6	0	Decimal	BOOL	Open KM Valve AV-KM
Local:3:I.Data.7	0	Decimal	BOOL	Run KM Pump PUMP-KM
Local:3:I.Data.8	0	Decimal	BOOL	Adding KM PL5
Local:3:I.Data.9	0	Decimal	BOOL	Run Mixing Tank Agitator MTR-MTA
Local:3:I.Data.10	0	Decimal	BOOL	Blending PL6
Local:3:I.Data.11	0	Decimal	BOOL	Run Mixing Tank Pump PUMP-MT
Local:3:I.Data.12	0	Decimal	BOOL	Pumping to Lines PL7
Local:3:I.Data.13	0	Decimal	BOOL	Open Mixing Tank Valve AV-MT
Local:3:I.Data.14	0	Decimal	BOOL	spare
Local:3:I.Data.15	0	Decimal	BOOL	

I have hidden the Force Mask column and stretched the Description column to see the full description.

TIP

You can switch columns on or off in the Controller Tags window by selecting View > Toggle Column

Copy the descriptions from the spreadsheet for the output card.


RSLogix 5000 - BATCHING [1756-L55]* - [Controller Tags - BATCHING(controller)]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
Favorites
Add-On
Alarms
Bit
Timer/Counter
Show...
Show All
Monitor Tags
Edit Tags

Scope: BATCHING

Name	Value	Style	Data Type	Description
Local:3:0	{...}		AB:1756_DO:0:0	
Local:3:0.Data	2#0000_000...	Binary	DINT	
Local:3:0.Data.0	0	Decimal	BOOL	System Ready PL1
Local:3:0.Data.1	0	Decimal	BOOL	Open City Water Valve AV-CW
Local:3:0.Data.2	0	Decimal	BOOL	Adding Water PL3
Local:3:0.Data.3	0	Decimal	BOOL	Open QR Valve AV-QR
Local:3:0.Data.4	0	Decimal	BOOL	Run QR Pump PUMP-QR
Local:3:0.Data.5	0	Decimal	BOOL	Adding QR PL4
Local:3:0.Data.6	0	Decimal	BOOL	Open KM Valve AV-KM
Local:3:0.Data.7	0	Decimal	BOOL	Run KM Pump PUMP-KM
Local:3:0.Data.8	0	Decimal	BOOL	Adding KM PL5
Local:3:0.Data.9	0	Decimal	BOOL	Run Mixing Tank Agitator MTR-MTA
Local:3:0.Data.10	0	Decimal	BOOL	Blending PL6
Local:3:0.Data.11	0	Decimal	BOOL	Run Mixing Tank Pump PUMP-MT
Local:3:0.Data.12	0	Decimal	BOOL	Pumping to Lines PL7
Local:3:0.Data.13	0	Decimal	BOOL	Open Mixing Tank Valve AV-MT
Local:3:0.Data.14	0	Decimal	BOOL	[illegible]

This completes the descriptions for the I/O.

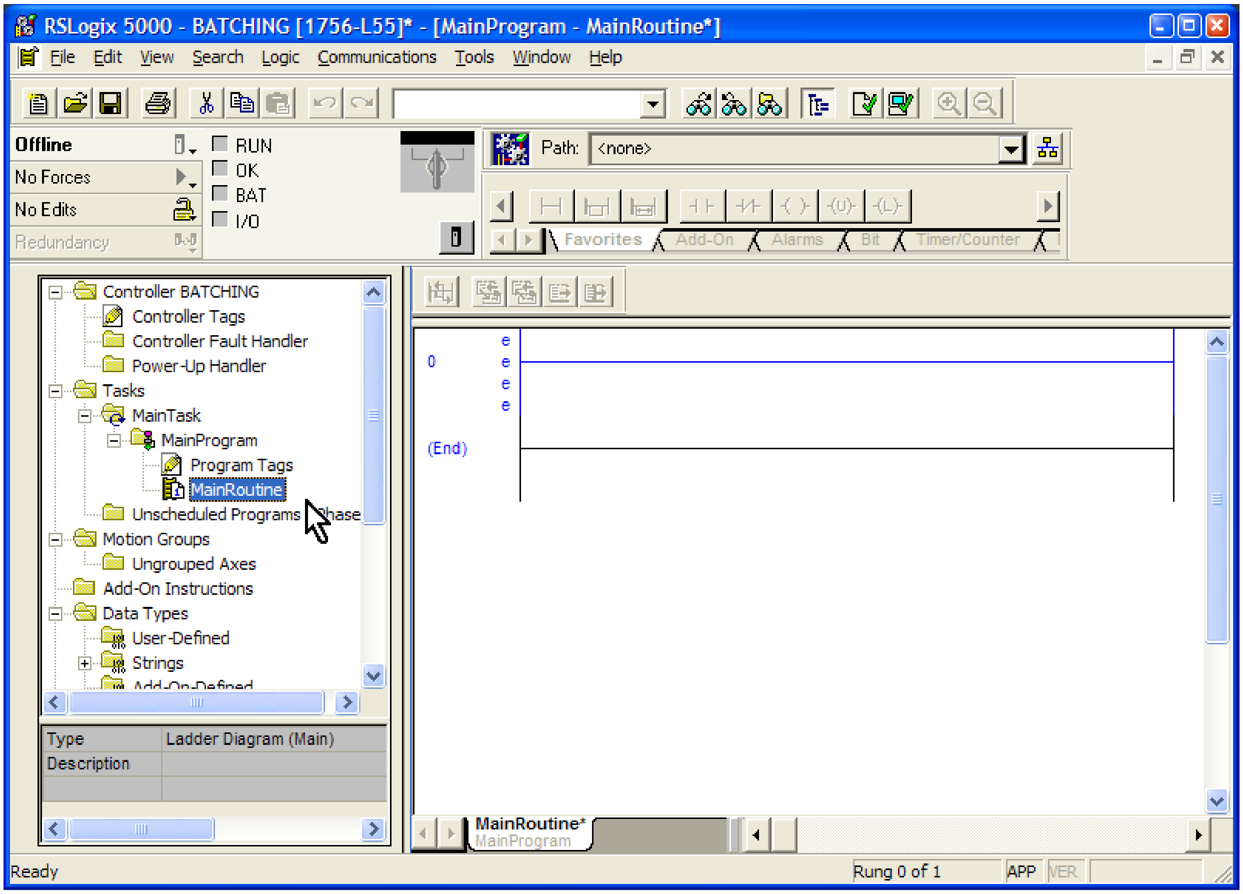
TIP It is a good idea to save your work frequently. This is done in RSLogix like it is in any other Windows program (CTRL-S, or File > Save or 💾).

Writing the Program

Ladder View

Open the Controller Organizer, expand the “Tasks” folder, and expand the “Main Program” folder.

Click on “Main Routine” and you should see this.



Setting Up An Overall Control Rung

Typically, a program will start with some kind of overall or master control rung. This rung will define a bit that must be on for the entire system to operate, and we include bits that we know must be true for the whole system to run.

In this project, we certainly want the E-Stop to be part of this logic. Our E-Stop (or, emergency stop) pushbutton switch is wired in such a way that the input must be on for the system to operate.

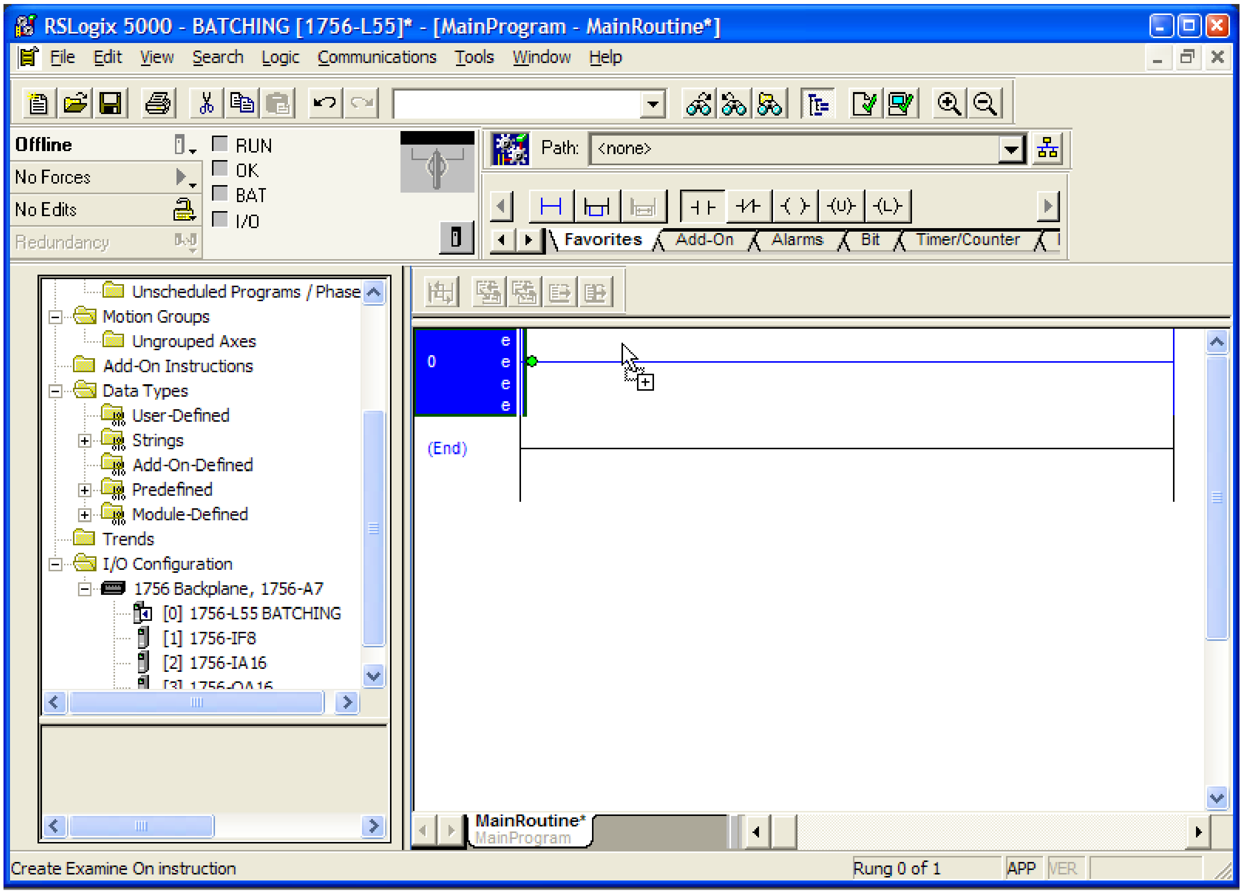
> TIP You can add a rung by right-clicking on the rung number and selecting “Add Rung” from the dropdown menu.
>
> You can also press CTRL-R.

We want to use the E-Stop input in this rung. Find the XIC (examine if closed) ⊣ ⊢ tool button in the User menu.

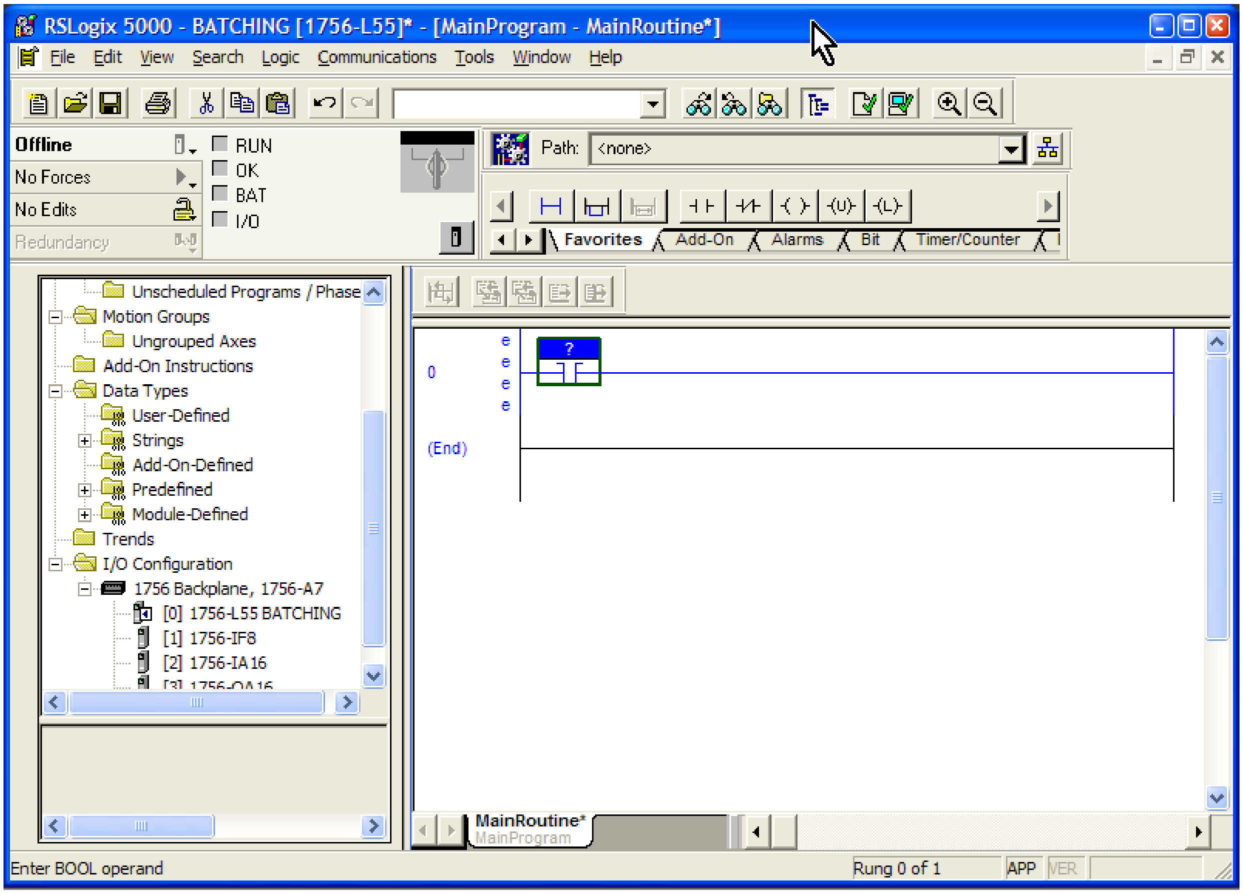
You can insert the instruction in a couple of ways. Simply clicking on the XIC icon will add it to the rung.

You can also click, hold and drag the tool to the spot in the rung where you want it inserted.

Click and drag it toward the new rung you just created. You will see that as you get near the rung, a green dot will appear. Green dots represent possible landing points for your instruction.



Release the mouse button and your screen should look like this.



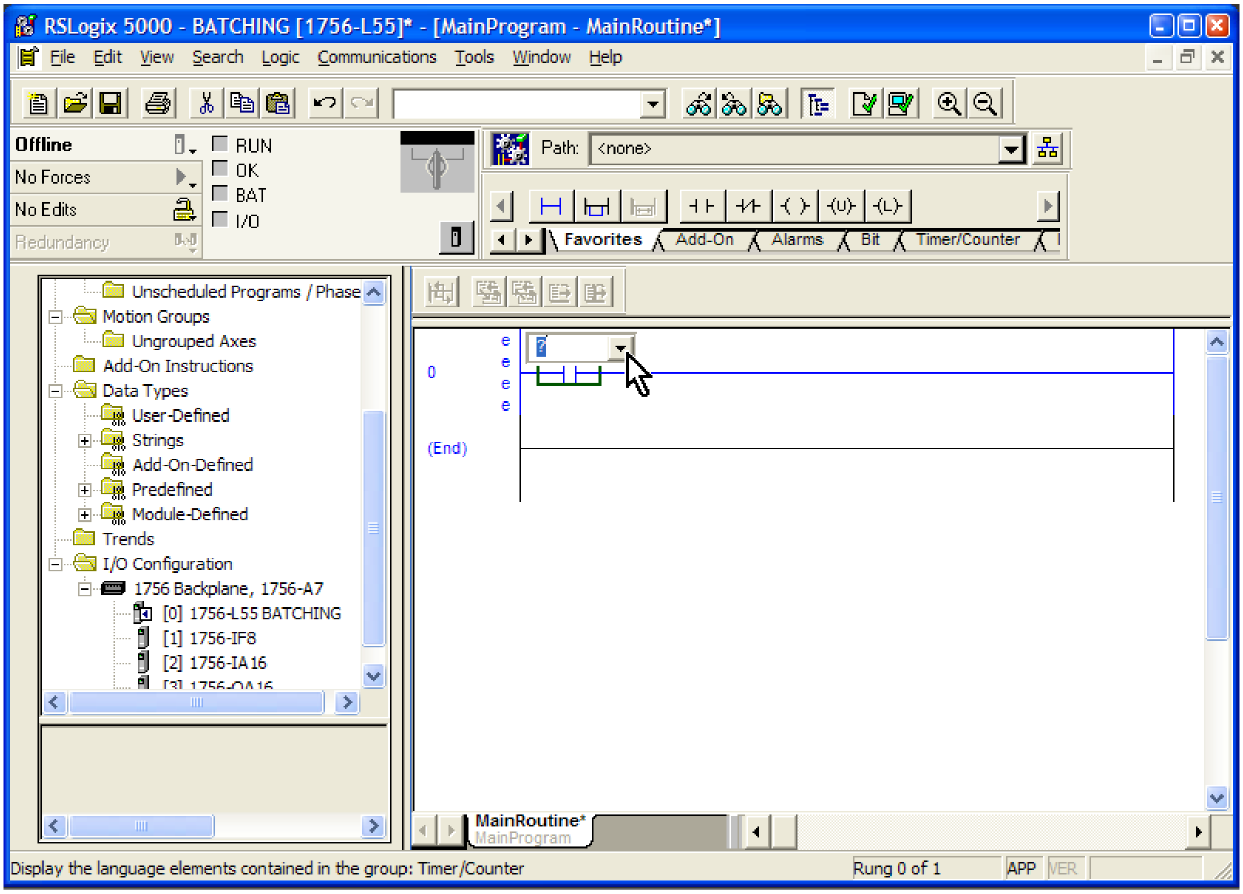
RSLogix 5000 - BATCHING [1756-L55]* - [MainProgram - MainRoutine*]

Path: <none>

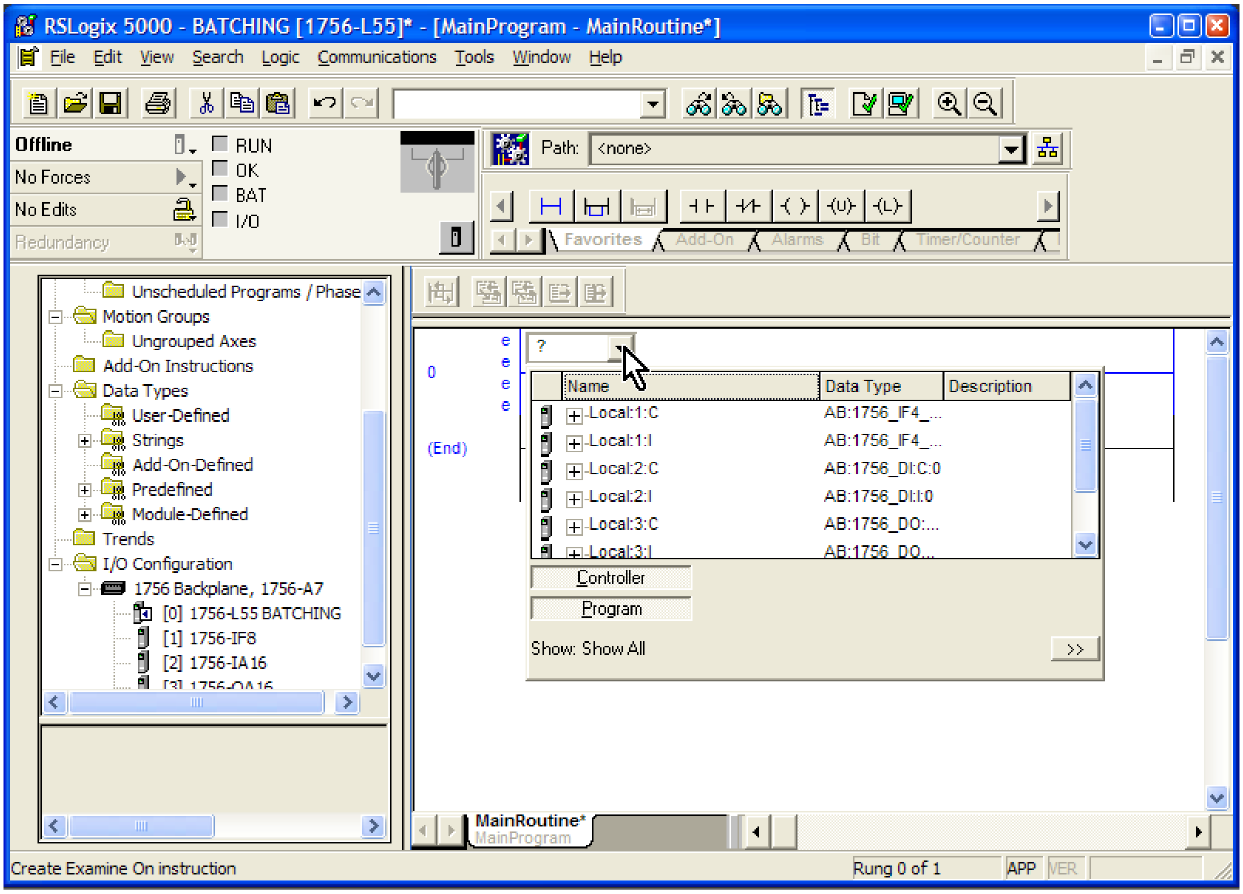
Enter BOOL operand

Rung 0 of 1

Press the enter key on your keyboard. A dropdown menu appears above the instruction.



Open the dropdown and your screen now looks like this.

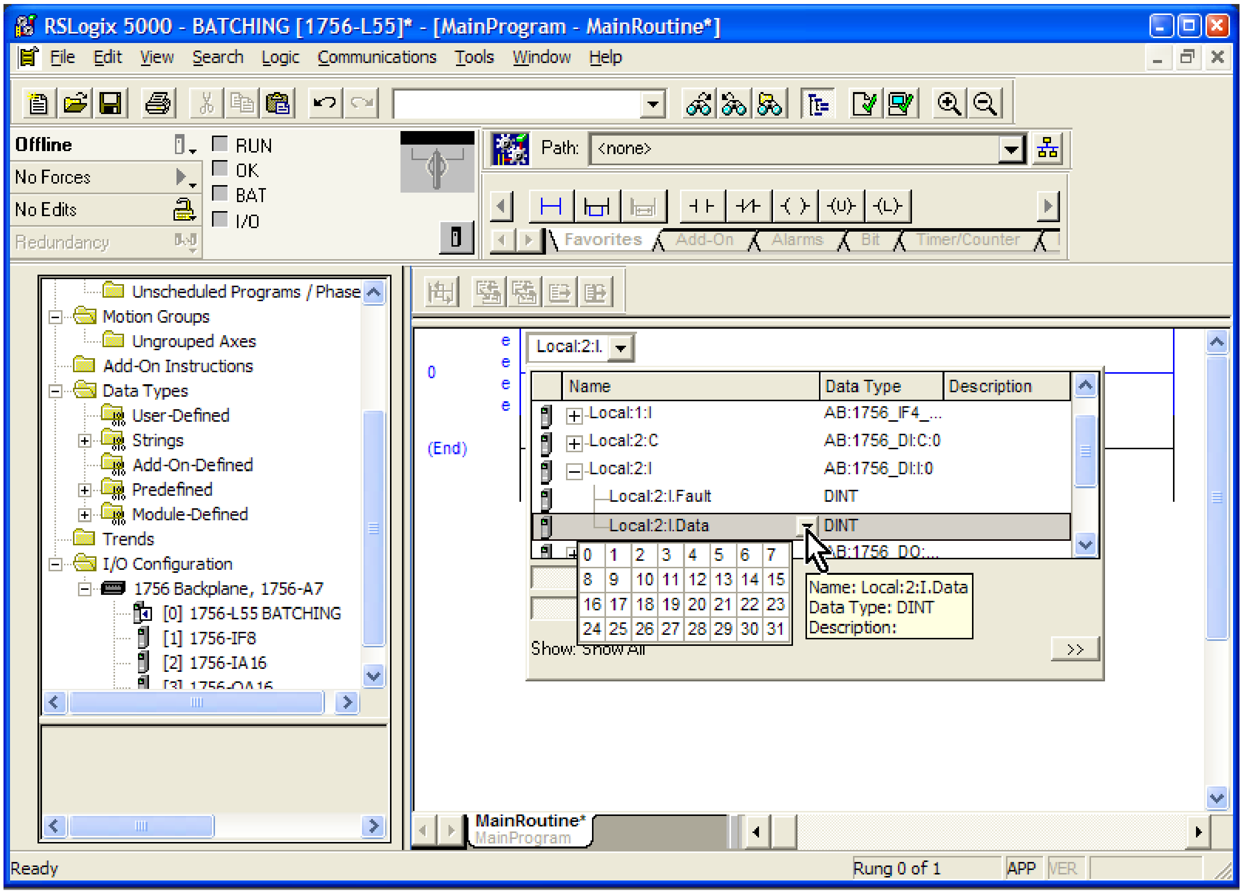


Name	Data Type	Description
Local:1:C	AB:1756_IF4_...	
Local:1:I	AB:1756_IF4_...	
Local:2:C	AB:1756_DI:C:0	
Local:2:I	AB:1756_DI:I:0	
Local:3:C	AB:1756_DO:...	
Local:3:I	AB:1756_DO...	

Notice that all the tag groups from our I/O are now showing.

From the I/O list, we see that the E-Stop switch is wired to the last position on the input card, giving it an address of Local:2:I.Data.15. We want to find that tag in this window.

Expand the tag group Local:2:I, then expand the tag group Local:2:I.Data.



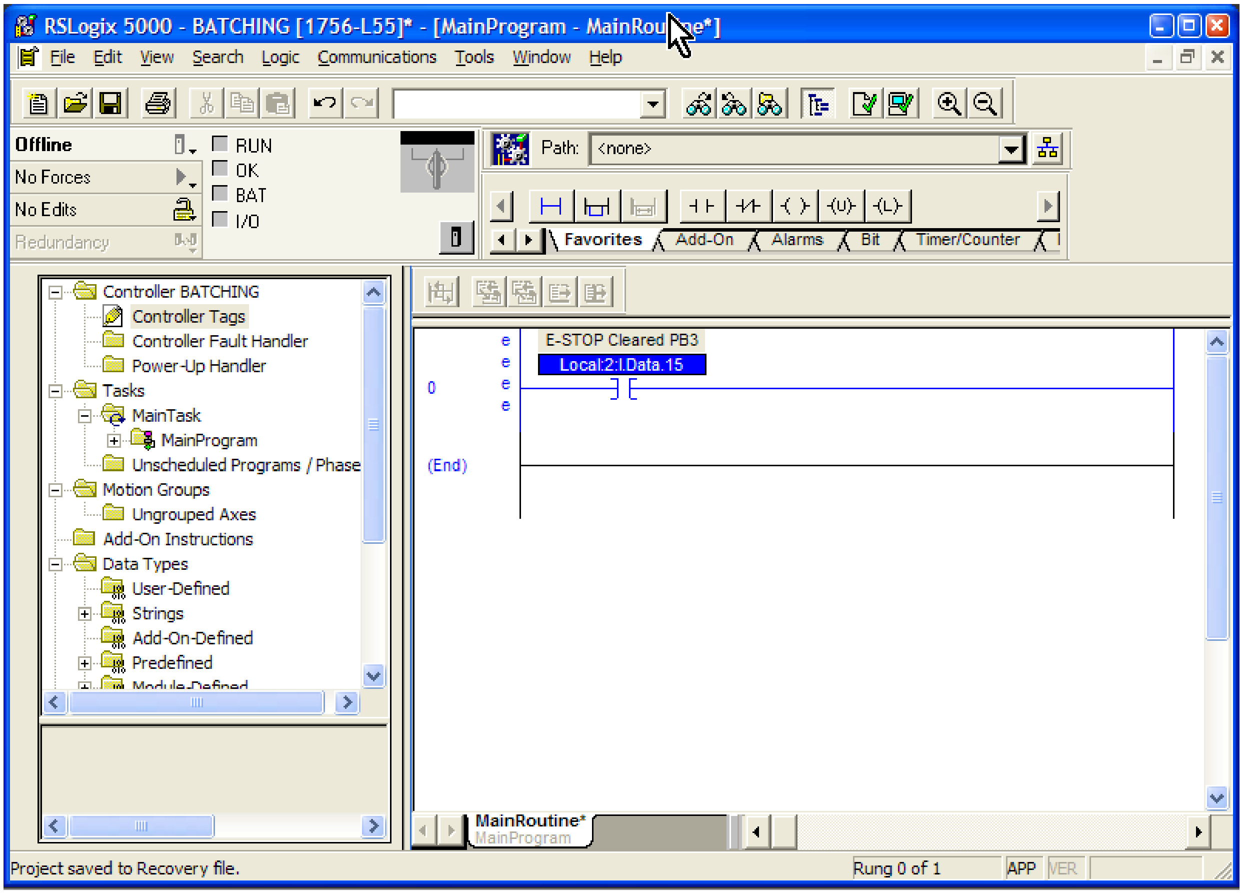
Local:2:I.

Name	Data Type	Description
Local:1:I	AB:1756_IF4_...	
Local:2:C	AB:1756_DI:C:0	
Local:2:I	AB:1756_DI:I:0	
Local:2:I.Fault	DINT	
Local:2:I.Data	DINT	
	AB:1756_DO:...	

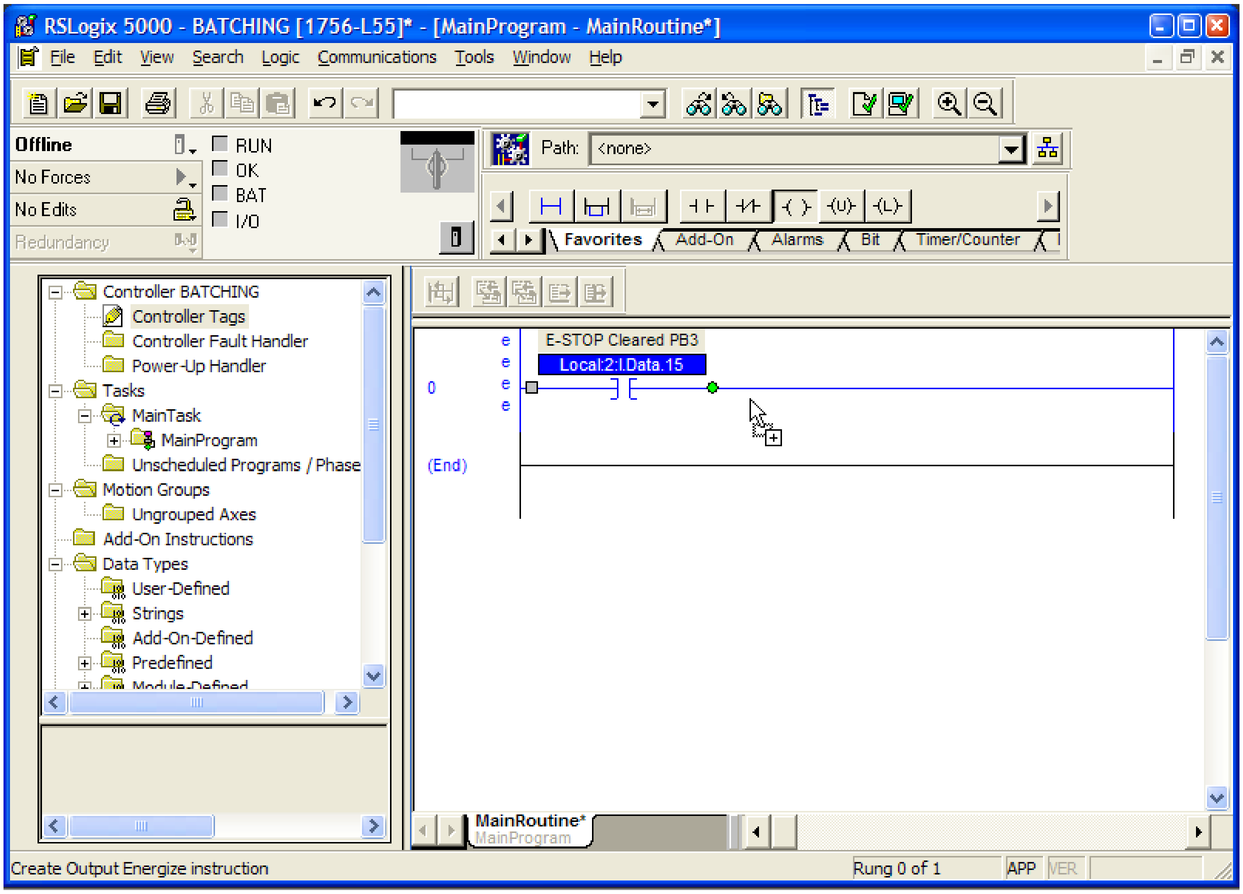
0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

Name: Local:2:I.Data
Data Type: DINT
Description:

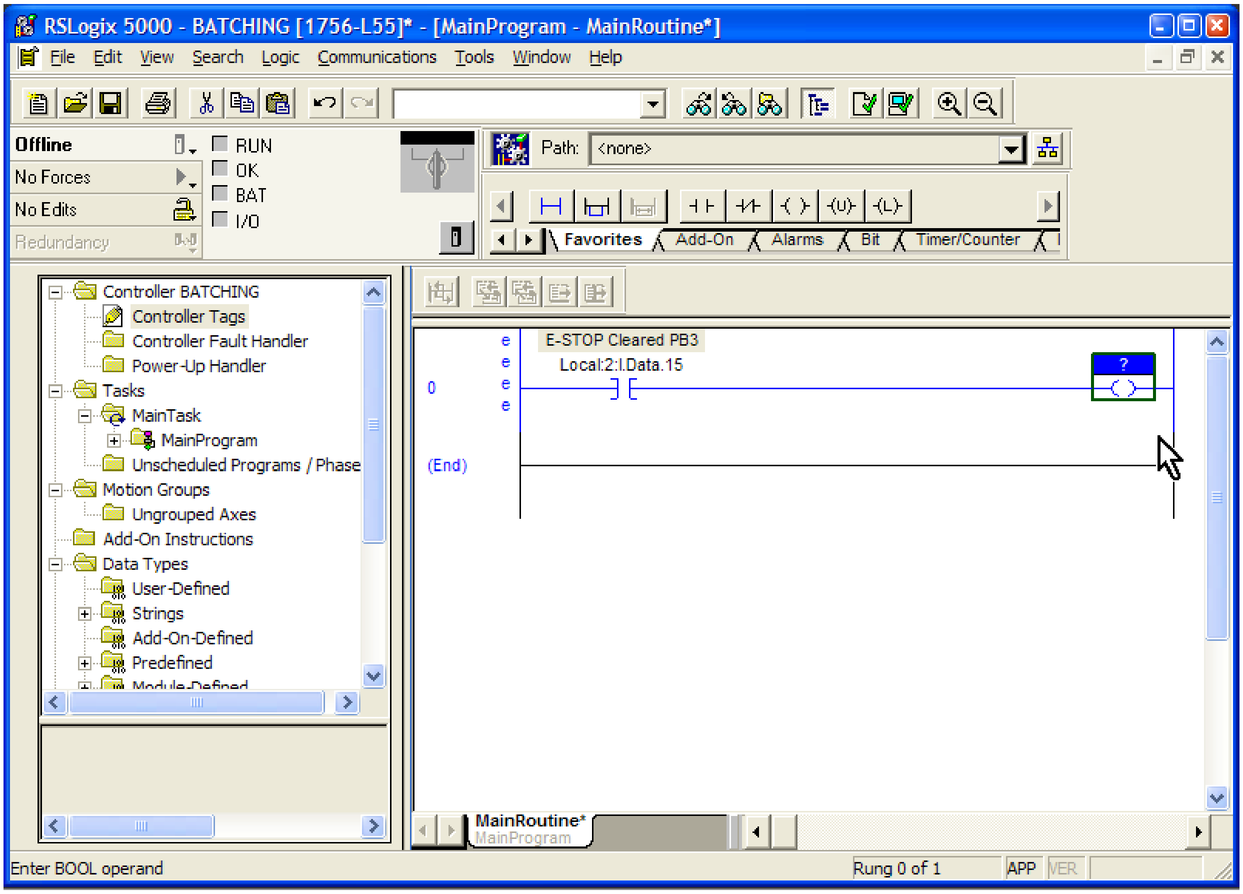
Click on bit 15 in the box to assign that address to the instruction.



Click and drag the OTE (output energize) tool button -( )- from the User menu down to the new rung. Place it on the marker at the right.



The screen looks like this.



RSLogix 5000 - BATCHING [1756-L55]* - [MainProgram - MainRoutine*]

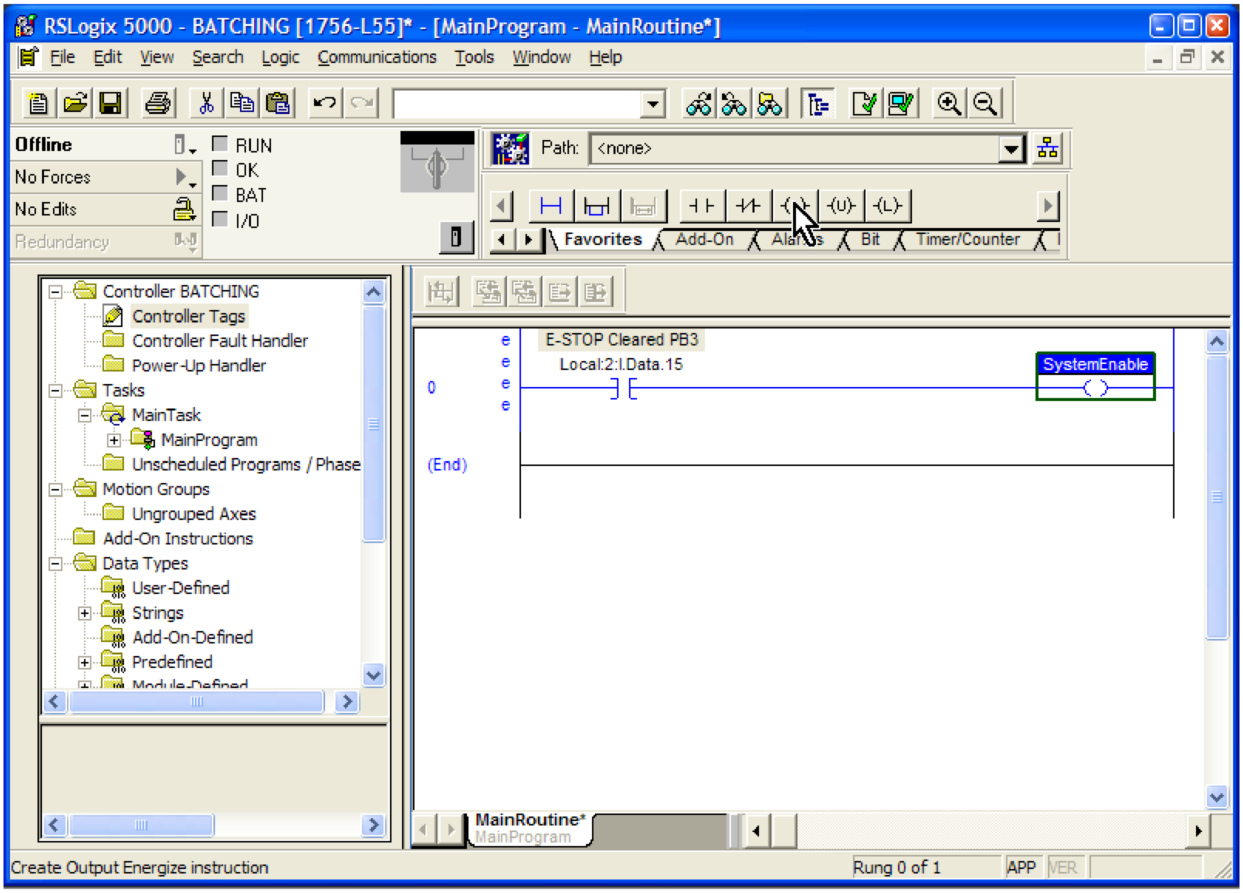
E-STOP Cleared PB3
Local:2:I.Data.15

Enter BOOL operand

Rung 0 of 1

PLC Programming with RSLogix 5000

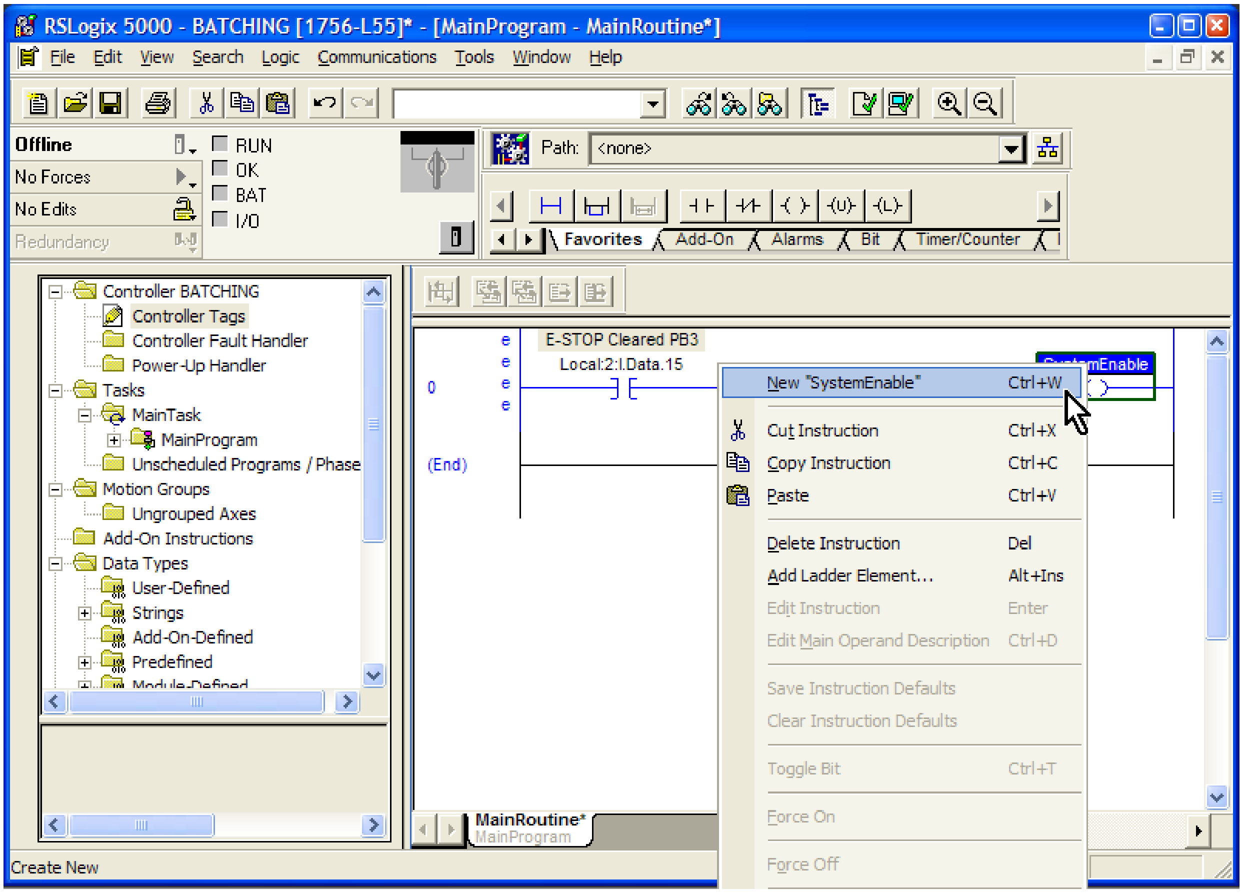
Click on the blue box (the tag name field) above the instruction. Type the phrase “SystemEnable” into the box and press enter.



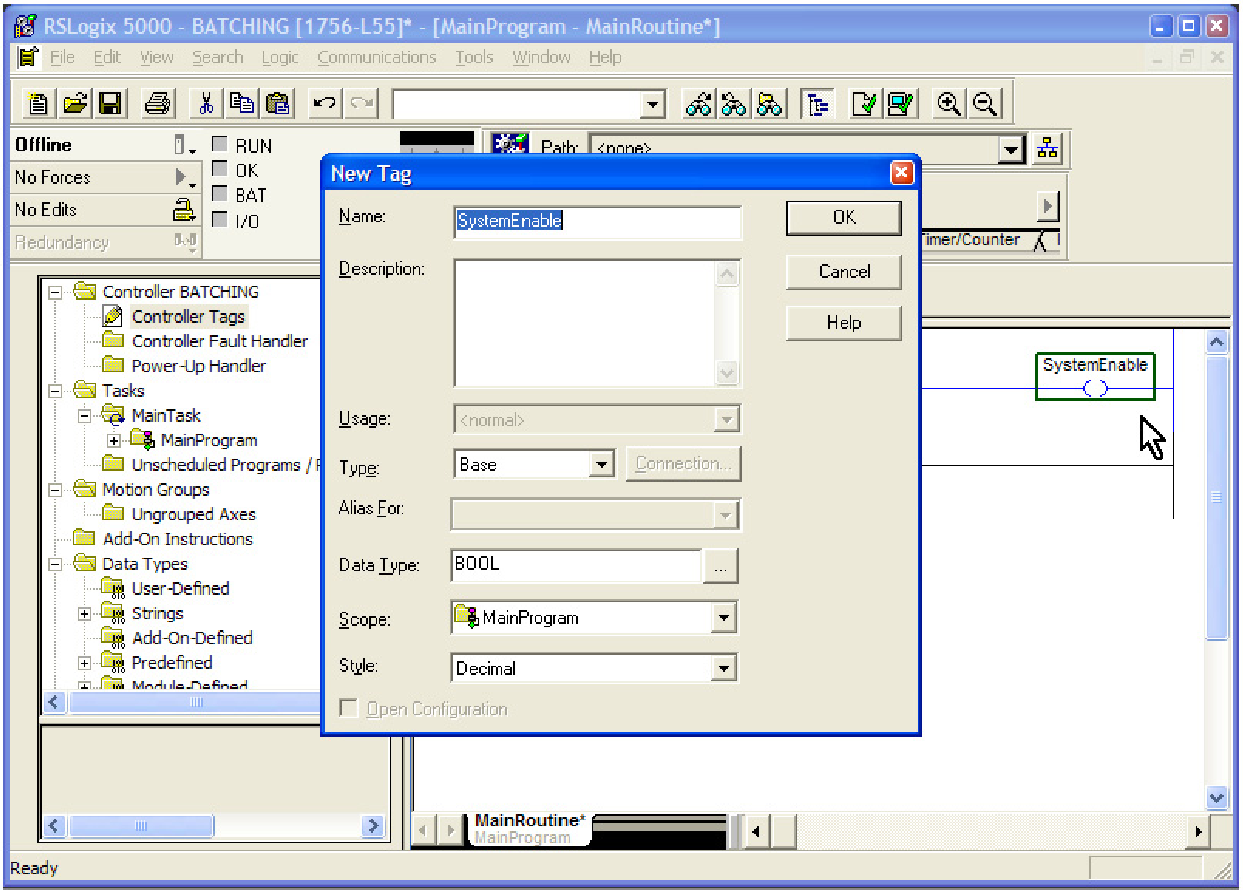
Since this is a tag name, there can be no spaces or special characters in the name.

This is a new tag for this program, so we have to define it. Right-click on the blue tag name field.

From the dropdown menu, choose “New System Enable”.



The “New Tag” dialog box appears.



New Tag

Name: SystemEnable

Description:

Usage: <normal>

Type: Base

Alias For:

Data Type: BOOL

Scope: MainProgram

Style: Decimal

□ Open Configuration

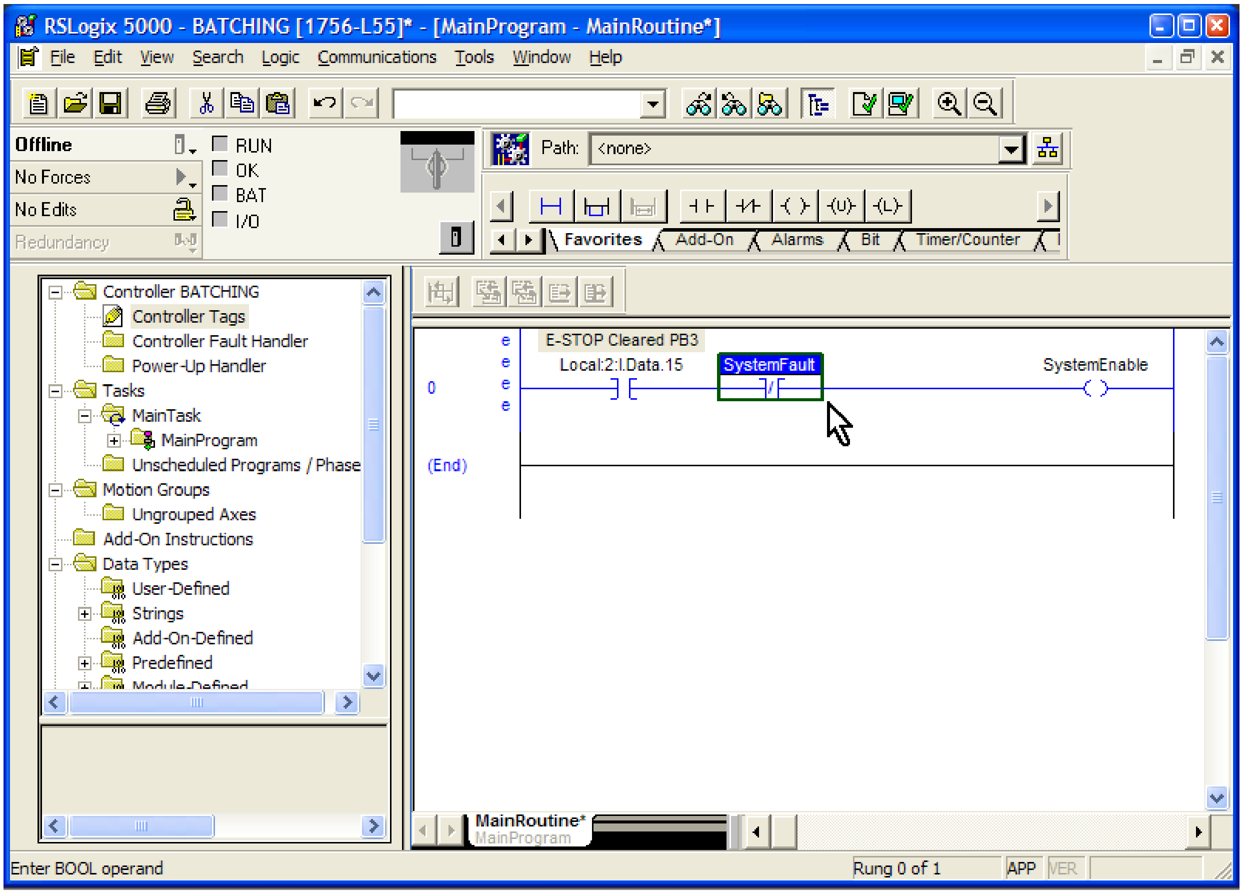
The tag we are defining here is simply a bit, so we can accept the default values in the dialog box. In fact, we really don’t need to add a description, as the tag name itself is pretty self-explanatory.

Click “OK”.

We know from the Project Scope that the system must stop if there is a fault. We are not sure of the details of all those faults yet, but we do know that we will summarize those faults somewhere in the program. It will result in a bit. We will use the address tag "SystemFault" for that bit. We also know that we want the "SystemEnable" to be on if we do *not* have a fault.

Bear with me here and it will make sense. Click and drag the XIO (examine if open) tool button ⊣/⊢ from the User menu down to the new rung. Place it just to the right of the E-stop input. Double-click on the tag filed and type "SystemFault" in the box.

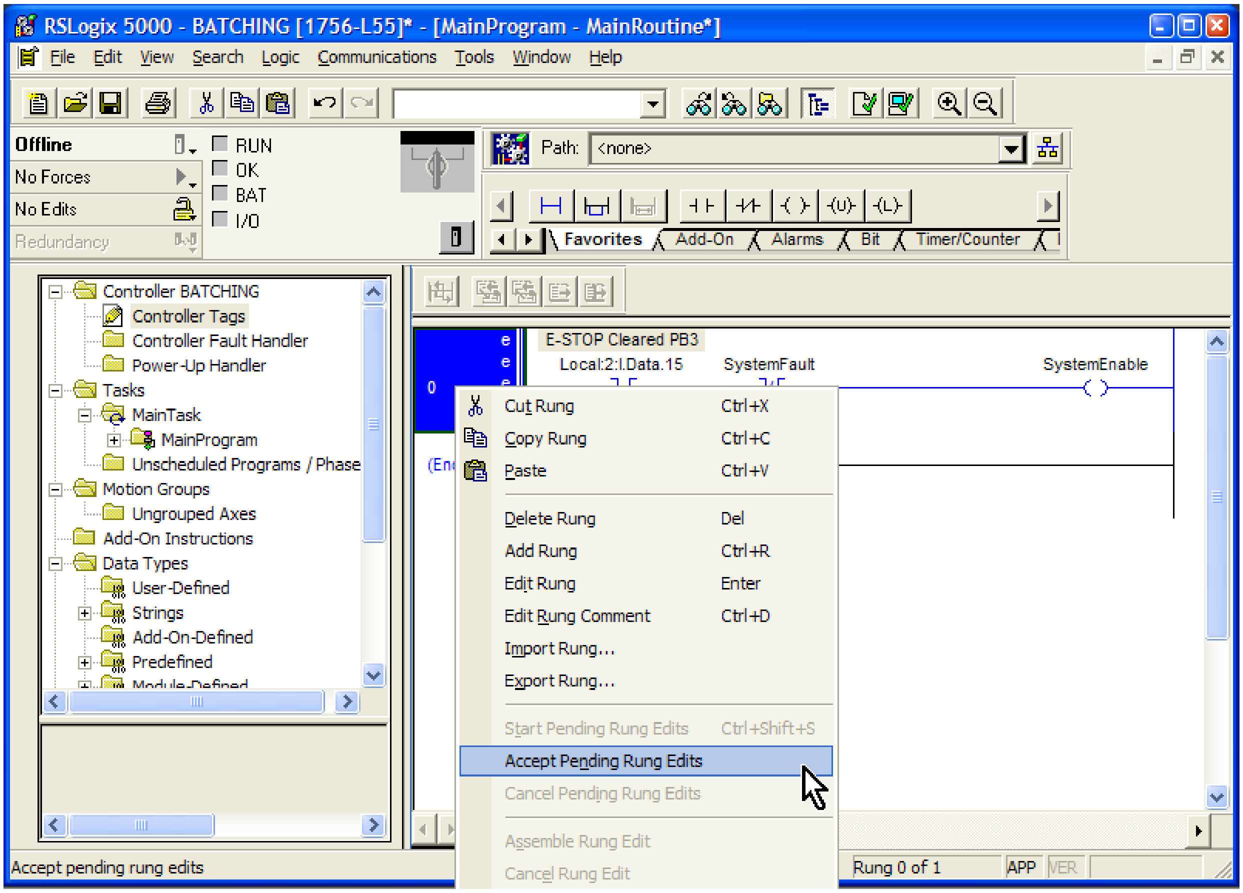
It should look like this:



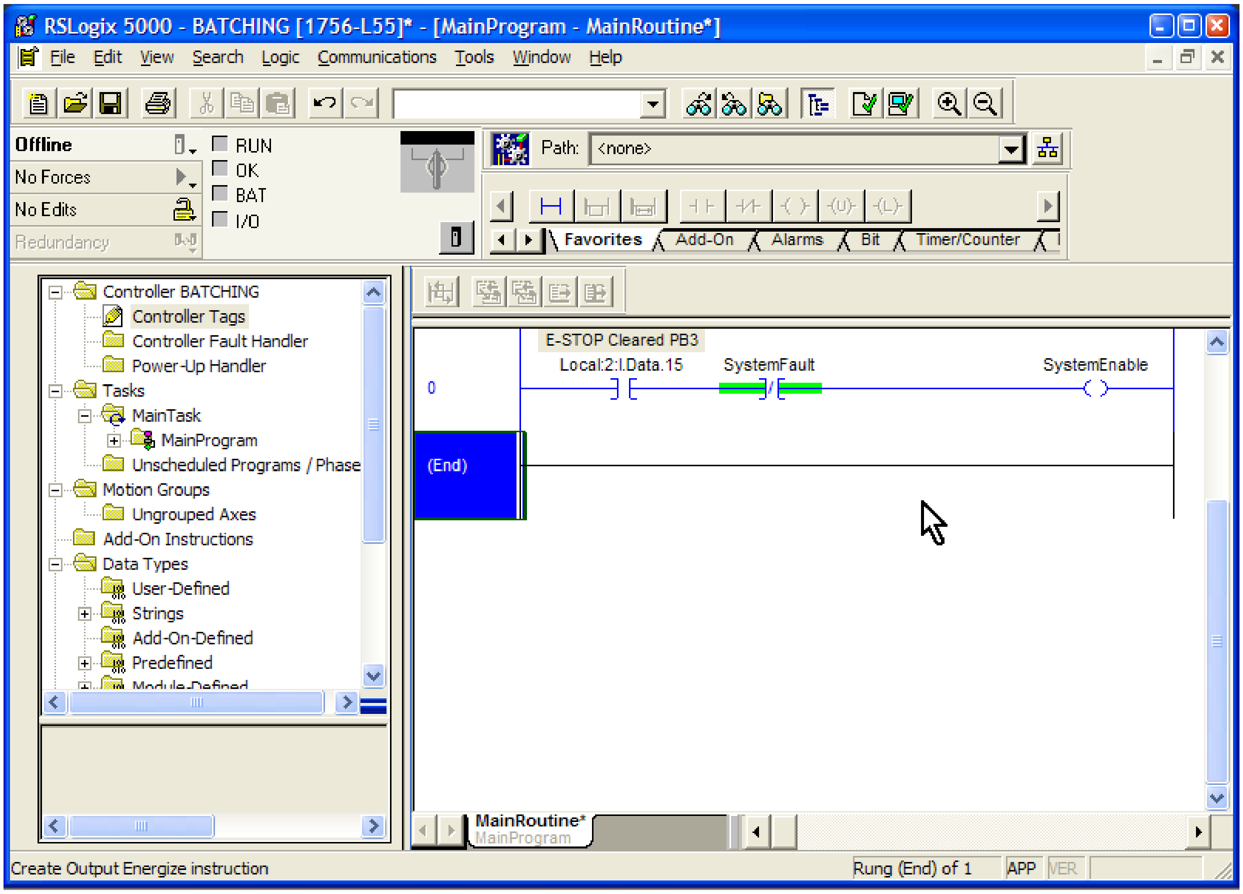
Right-click on the tag name and define the tag (you may also press CTRL-W to get to the "New Tag" dialog box). By the way, if you forget to define that tag, RSLogix 5000 will remind you when you accept the rung edits.

Let's see what we have. The logic of the rung works just like an electrical circuit. If the E-Stop is cleared and there is not a System Fault, the System Enable bit will be on. That is exactly what we want. We will work out the fault logic later.

We need to accept the current rung. Right-click on the rung number (0) and this dropdown appears.



Choose “Accept Pending Rung Edits”.



We have completed the first rung in the program.

You may wonder why the SystemFault instruction is highlighted green. This is because the value of the SystemFault tag is 0. Since the instruction is an XIC, or Examine If Closed, the instruction is true. Therefore, the instruction is highlighted.

You should note that these colors are highly configurable. In fact, I have seen many different color schemes. Just keep in mind that you may look at someone else’s laptop and find blue, for example, has been configured to highlight a bit that is true.

Starting a Batch Cycle

The Project Scope said that the operator may start a batch by pressing the “Start Batch” pushbutton on his console. Let’s start with that input.

Right-click on the last rung and choose “Add Rung”. Click and drag the XIC (examine if closed) tool button -| |- from the User menu to the left side of the new rung.

Click on the tag name field above the instruction and navigate through the tag groups until you find the input Local:2:I.Data.0. This is the “Start Batch PB1” pushbutton.

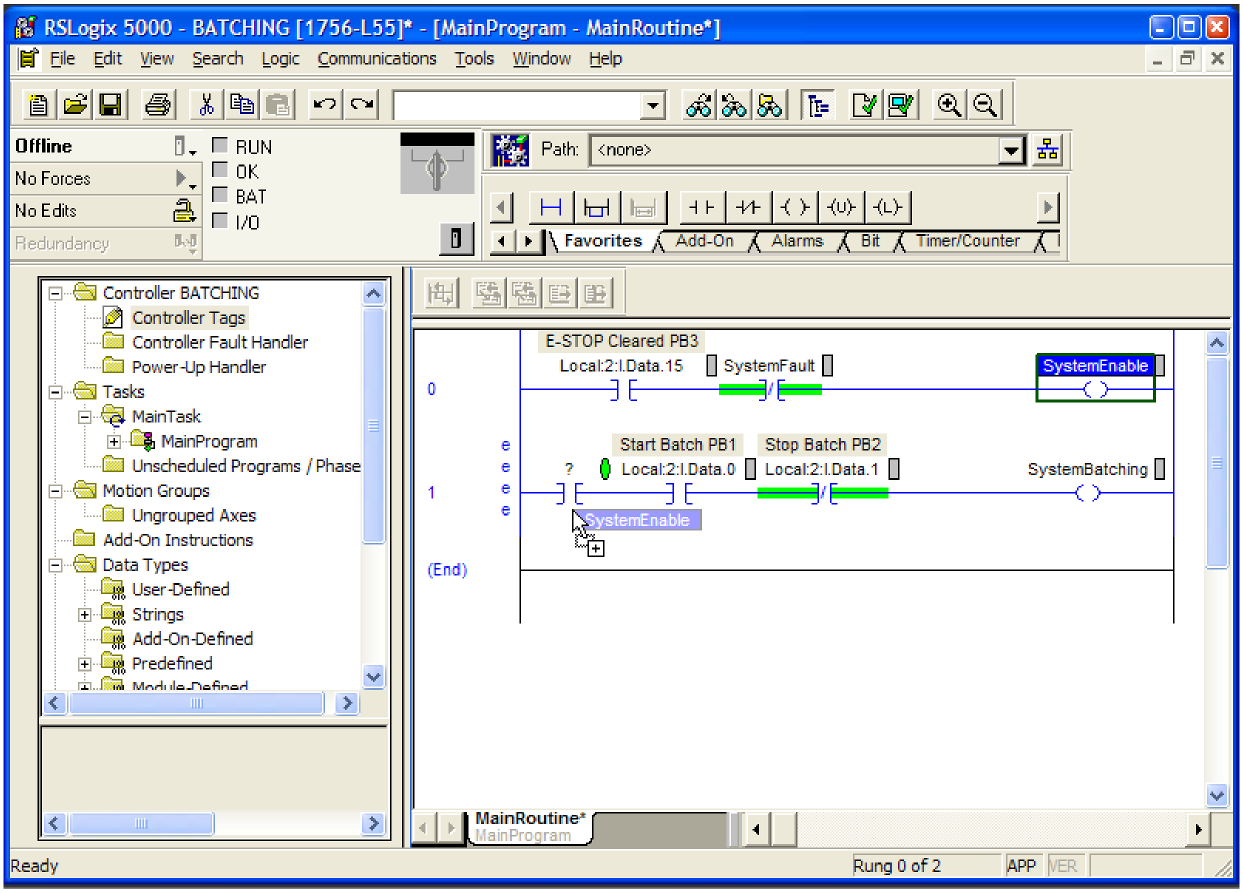
Click and drag the OTE (output energize) tool button -( )- from the User menu down to the new rung. Place it on the marker at the right. We are creating a new tag that indicates the system is currently batching. Label this tag “SystemBatching”. Right-click on the tag, select “New SystemBatching” and accept the defaults.

If the operator chooses, he may stop the batch. We will make use of the “Stop Batch” pushbutton. Click and drag the XIO (examine if open) tool button -|/|- from the User menu down to the new rung. Navigate through the tag groups until you find the input Local:2:I.Data.1. This is the “Stop Batch PB2” pushbutton.

We don’t want the operator to be able to start a batch if the System Enable bit is not on. We will add that by dragging the XIC (examine if closed) tool button -| |- to the left side of the new rung.

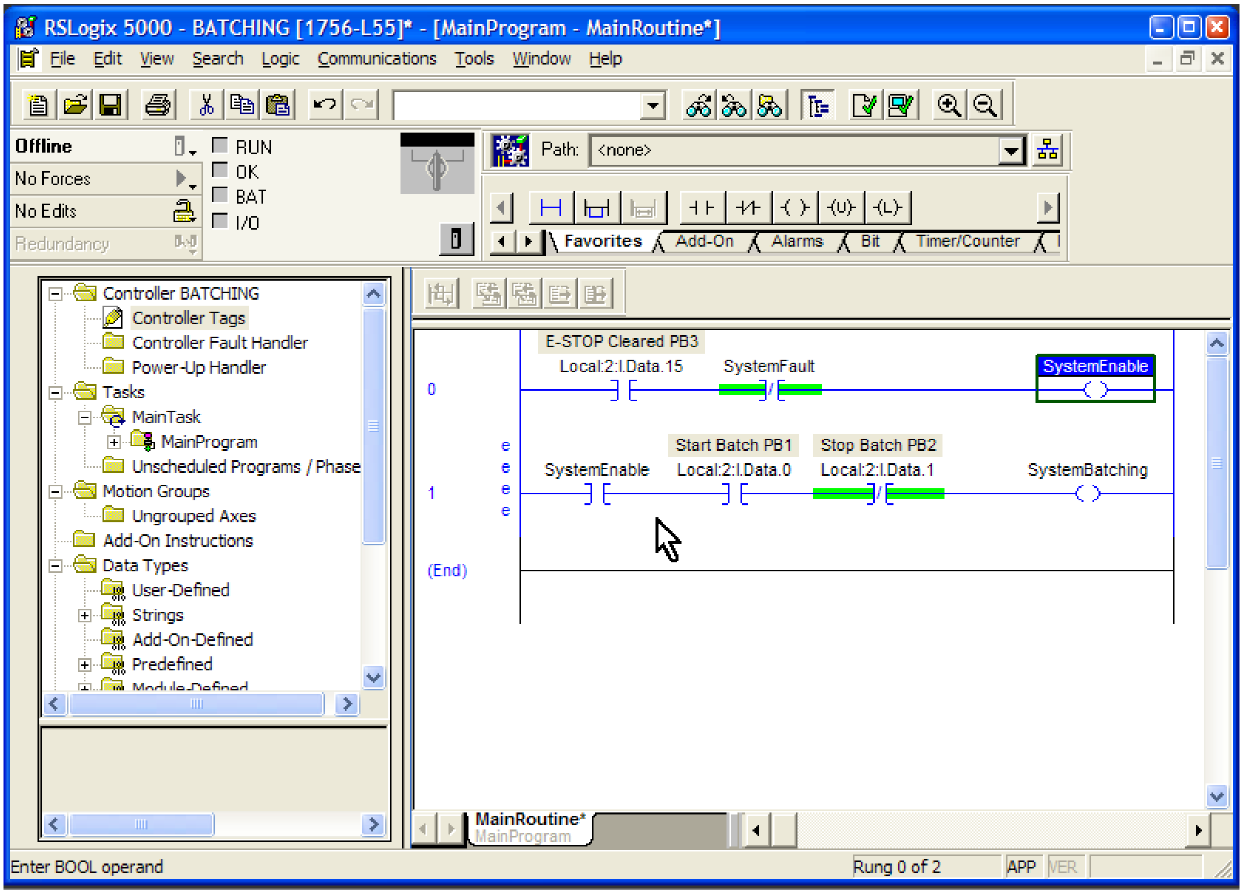
RSLogix 5000 lets us quickly assign a tag to this bit by dragging the bit name from another rung.

Click, hold and drag the tag name “SystemEnable” from the OTE in Rung 0. As you drag the tag name box, you will see gray rectangles appear, indicating that these are potential places to assign the tag. As the cursor gets nearer to a target, that target icon changes to a green oval.



Release the mouse button when the cursor is near the first instruction in the rung.

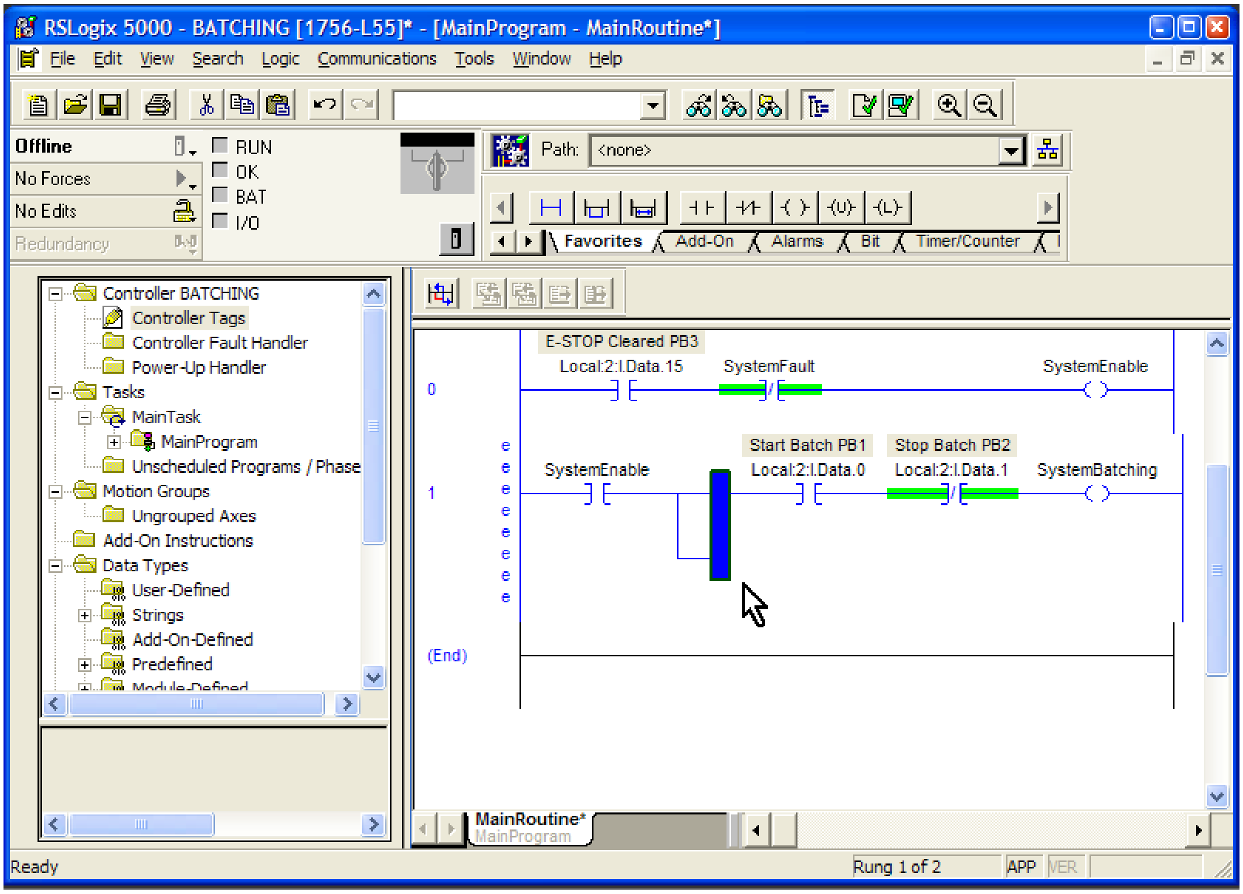
Here is what we have so far:



Enter BOOL operand

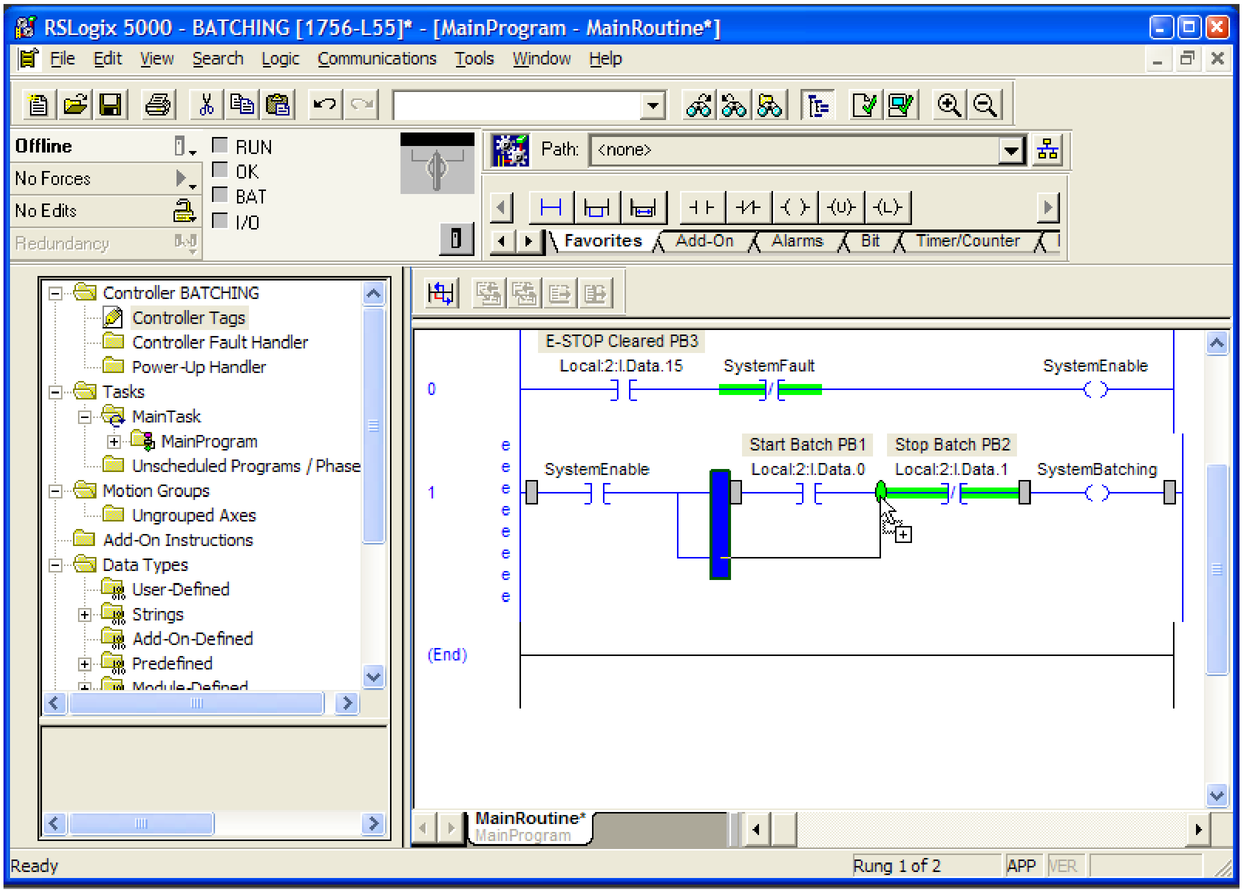
This rung we are creating will work much like a traditional motor starter circuit that uses a contact from the motor starter wired in parallel with the start button to hold in the coil. In the PLC, the “contact” is an XIO with the same tag as the “coil”, which is “SystemEnable”.

We need to “wire the contact” in parallel with the start button. We do this with a branch instruction. Drag the Branch tool button and place it on the marker between the System Enable bit and the Start Batch bit.

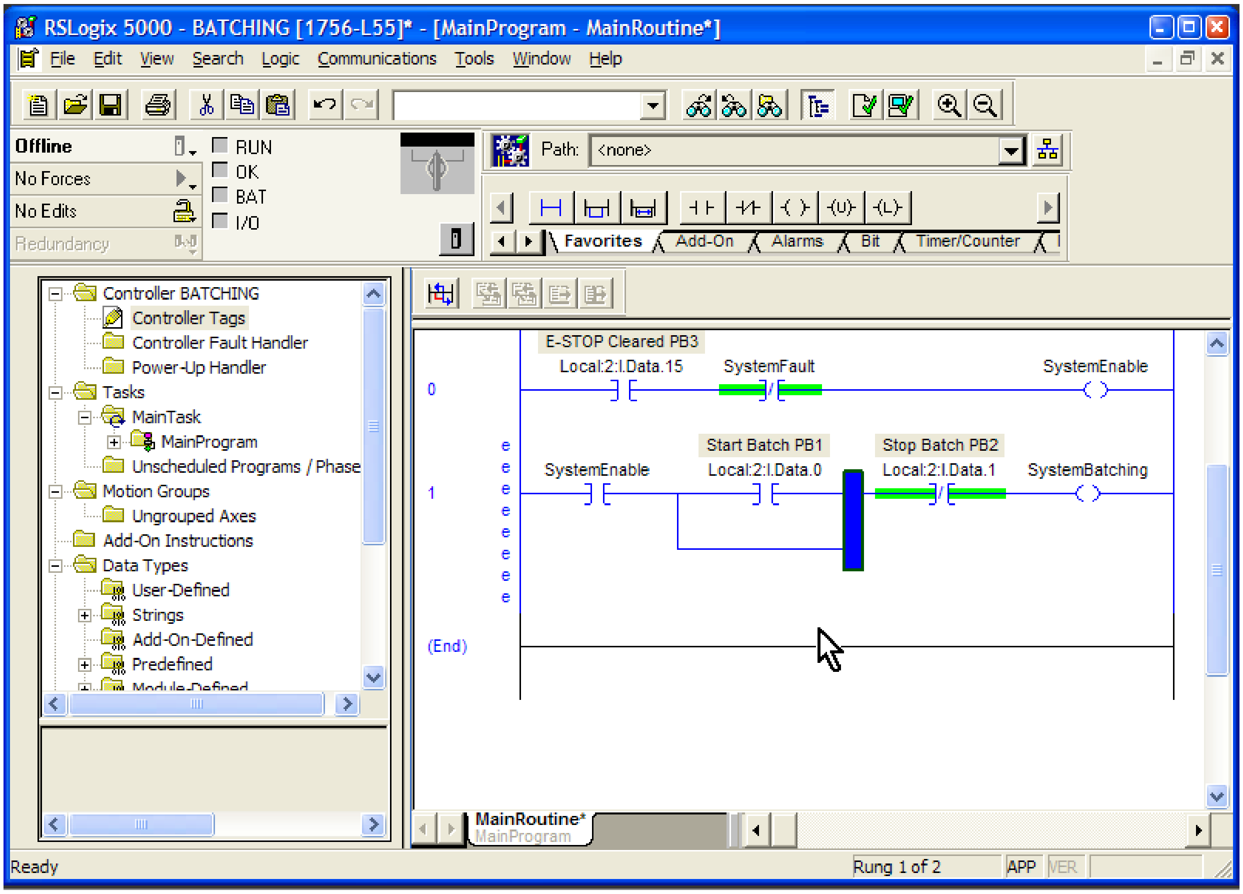


Click on the blue section of the branch and drag it to the target to the right of the Start Batch PB1 instruction.

As you are dragging, it looks like this.



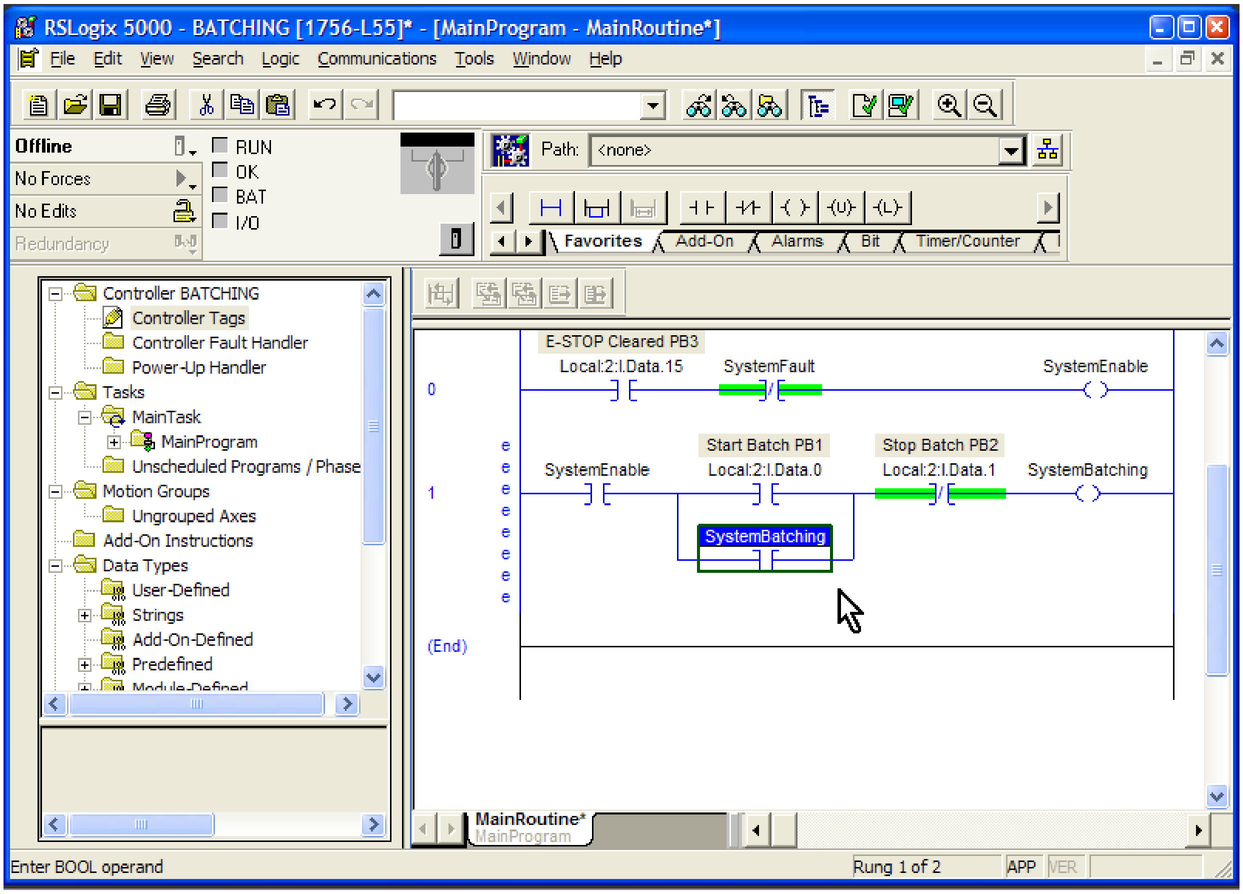
Release the mouse button and the branch will appear around the Start Batch bit.



Click and drag the XIC (examine if closed) tool button ┤├ from the User menu to the left side of the new branch.

We want this instruction to be tagged “SystemBatching”, just like the OTE in this rung.

There is a quick way to do that. Just click and drag the “SystemBatching” tag name from the OTE in this rung to the new instruction.



RSLogix 5000 - BATCHING [1756-L55]* - [MainProgram - MainRoutine*]

Path: <none>

Enter BOOL operand

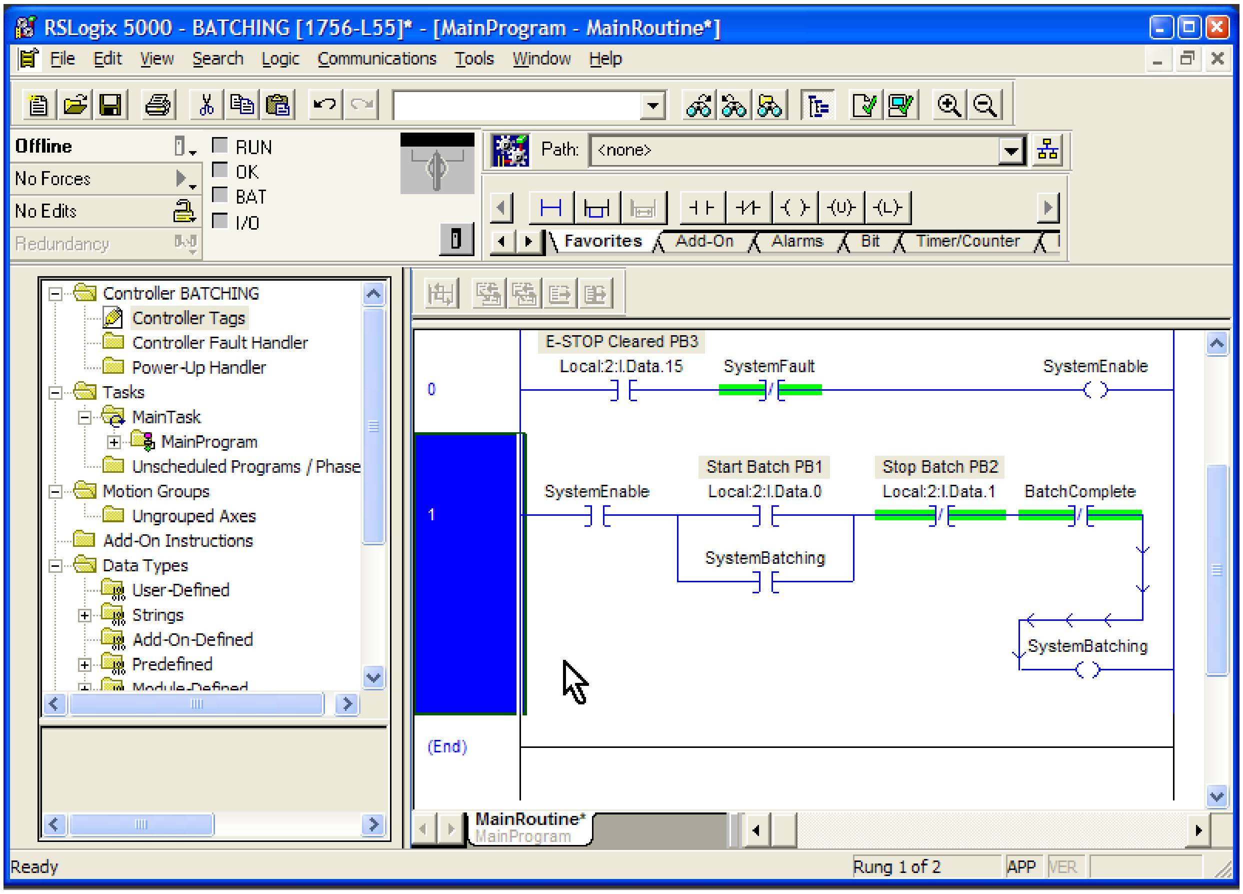
Rung 1 of 2

This is called a latching rung. If the SystemEnable bit is on, the SystemBatching bit can be latched by momentarily pressing the Start Batch pushbutton. The SystemBatching bit will stay on and the rung will remain latched until the Stop Batch pushbutton is pressed or the SystemEnable bit goes off.

Thinking ahead, though, we know that the system will stop the batch automatically after it has pumped all the finished product to the filling lines. We are not sure how we will know that yet, but we know we need a bit to unlatch the rung.

Click and drag the XIO (examine if open) tool button ┤/├ from the User menu down to the marker just in front of the SystemBatching OTE instruction. Type in the tag “BatchComplete” and define the tag (CTRL-W).

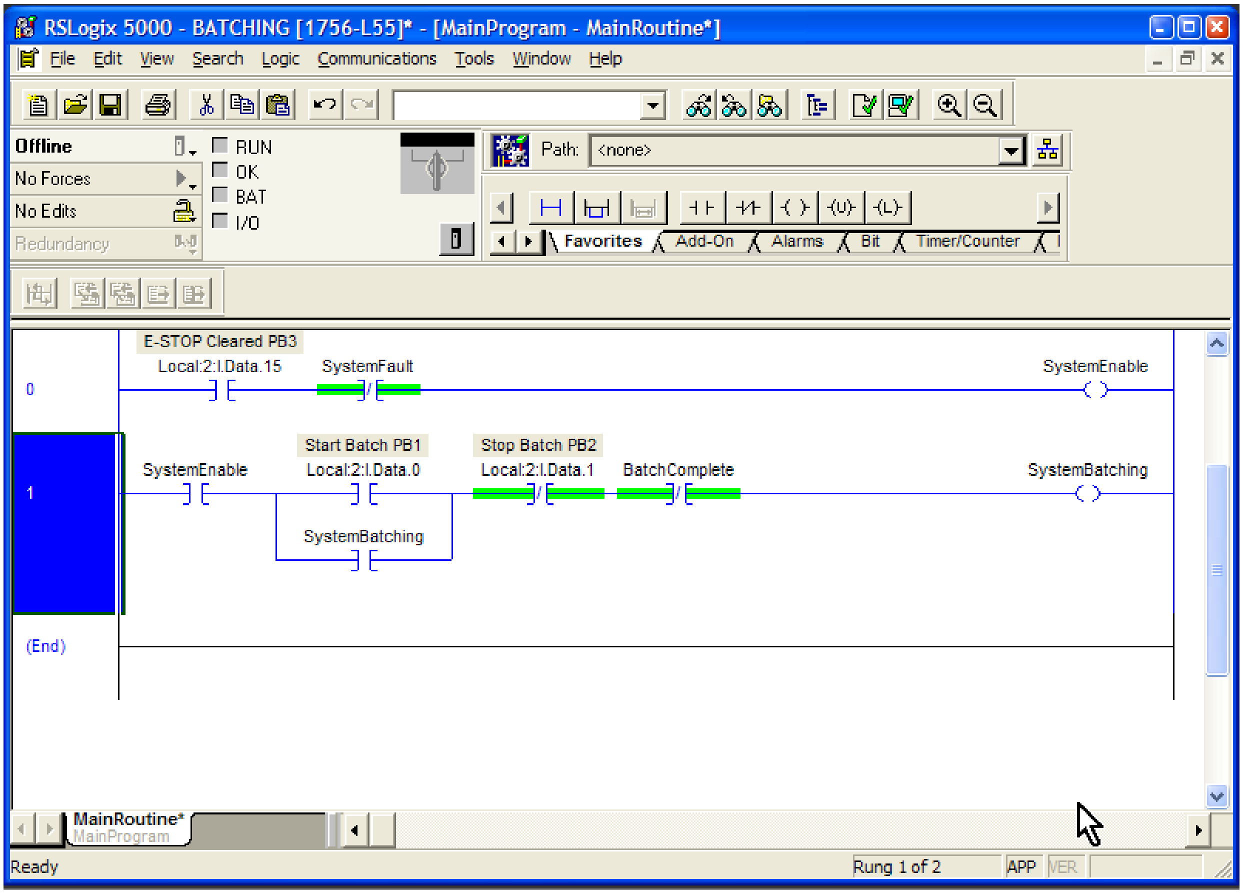
Right-click on the rung number and verify the rung. It should appear like this.



Notice how the rung has become too long to be contained on one line, so RSLogix is putting the OTE instruction below and re-routing the connecting line. It is accurate, but a little confusing.

You can get around this by a couple of ways. You can set the “Zoom” factor under the “View” menu to get the rung to appear on one line.

You can also hide the Controller Organizer by pressing “ALT-0”. That is what we will do here. We don’t have a real need to see the Controller Organizer right now, anyway.



Batching Steps

As you recall from the project scope, there are a number of steps needed to create the finished product. They are:

1. Adding City water
2. Adding chemical QR
3. Adding chemical KM
4. Blending the mixture with the agitator
5. Pumping the finished product to the filling lines

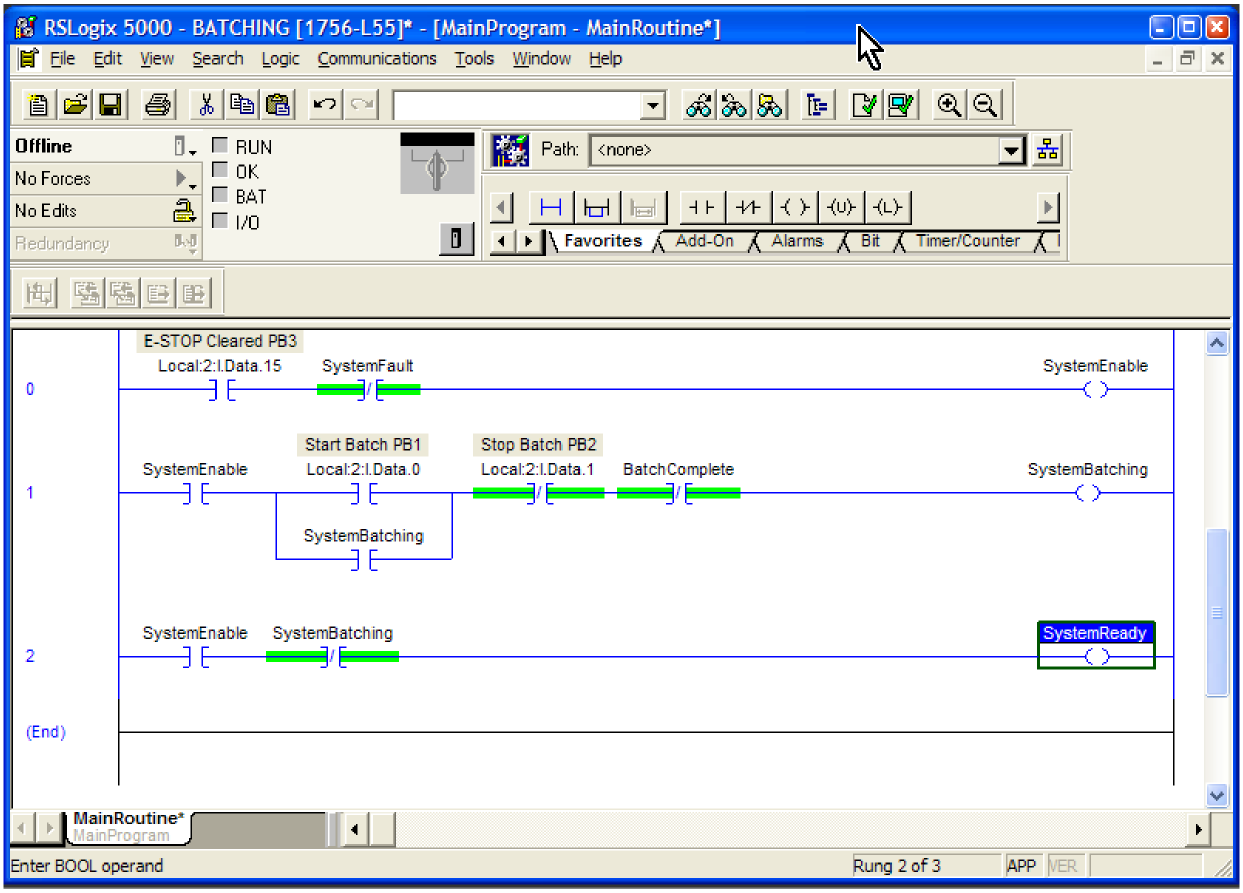
Step 1 – Adding City Water

We need to initiate Step 1. Before we do that, though, we need to add another permissive bit. We will tag that bit “SystemReady”.

We know that if the system is enabled but not currently batching, it is ready to begin a batch. We need insert a new rung and create a “SystemBatching” and “SystemReady” bit.

Add a new rung and program it as shown in Rung 2.

You can see how Rung 2 should look in the picture below. The “SystemReady” bit will be on when the system is enabled, but not batching.



To actually initiate the batch and hold the batch in Step 1, we are going to use the Output Latch (OTL) instruction. This instruction works in conjunction with the Output Unlatch (OTU) instruction. The instructions will work on the same bit address, but are typically found on different rungs.

The batch will be started when the operator pushes the Start Batch button. We will latch that bit and label it Step 1.

Insert a new rung at the bottom of the ladder. We need an XIC for the SystemReady bit and an XIC for the Start Batch pushbutton at the beginning of the new rung.

Click and drag the XIC (examine if closed) tool button ┤├ from the User menu to the left side of the new branch. Drag the “SystemBatching” tag from Rung 2 to the new instruction.

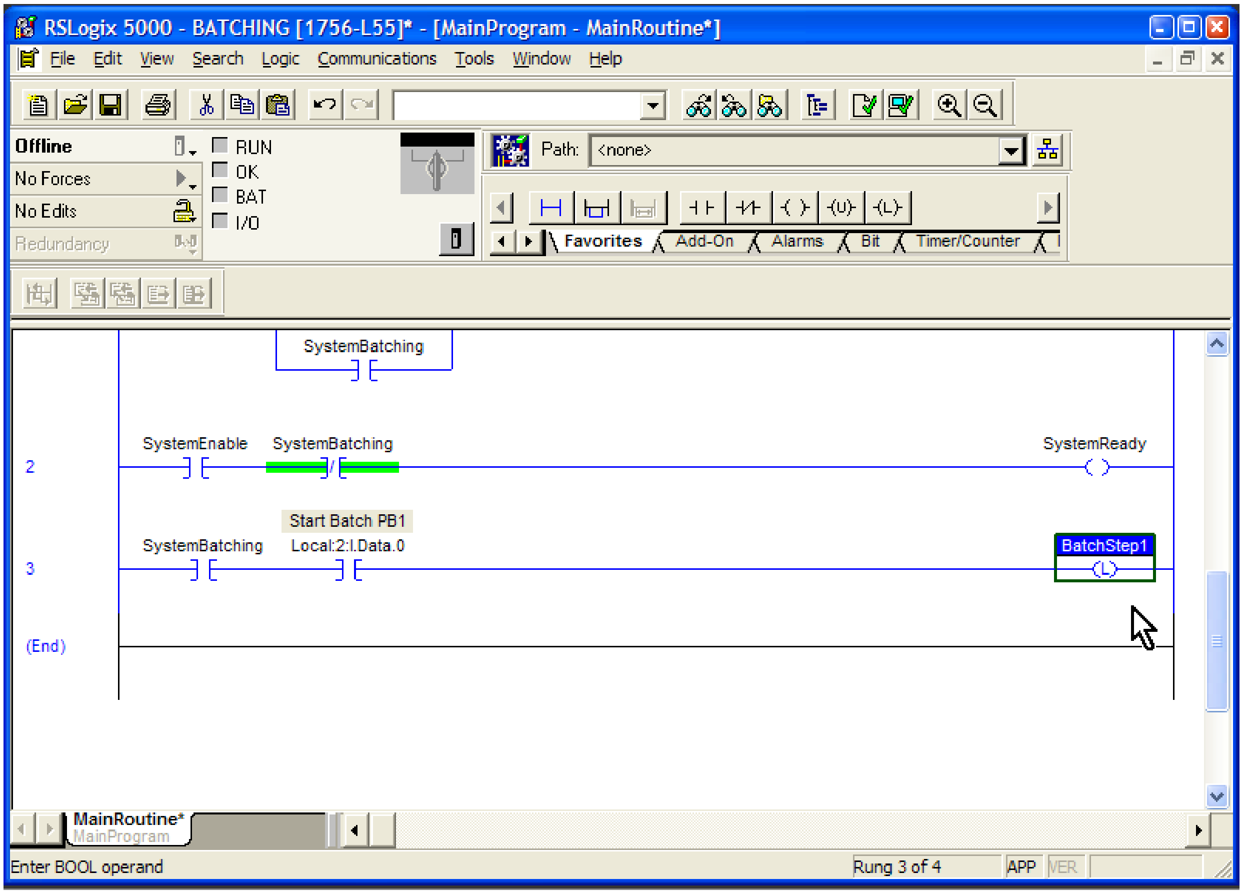
To save some typing, you can copy and paste instructions. Highlight the Start Batch instruction in Rung 1 by clicking on the XIC icon.

Press CTRL-C.

Click on the rung number for Rung 3.

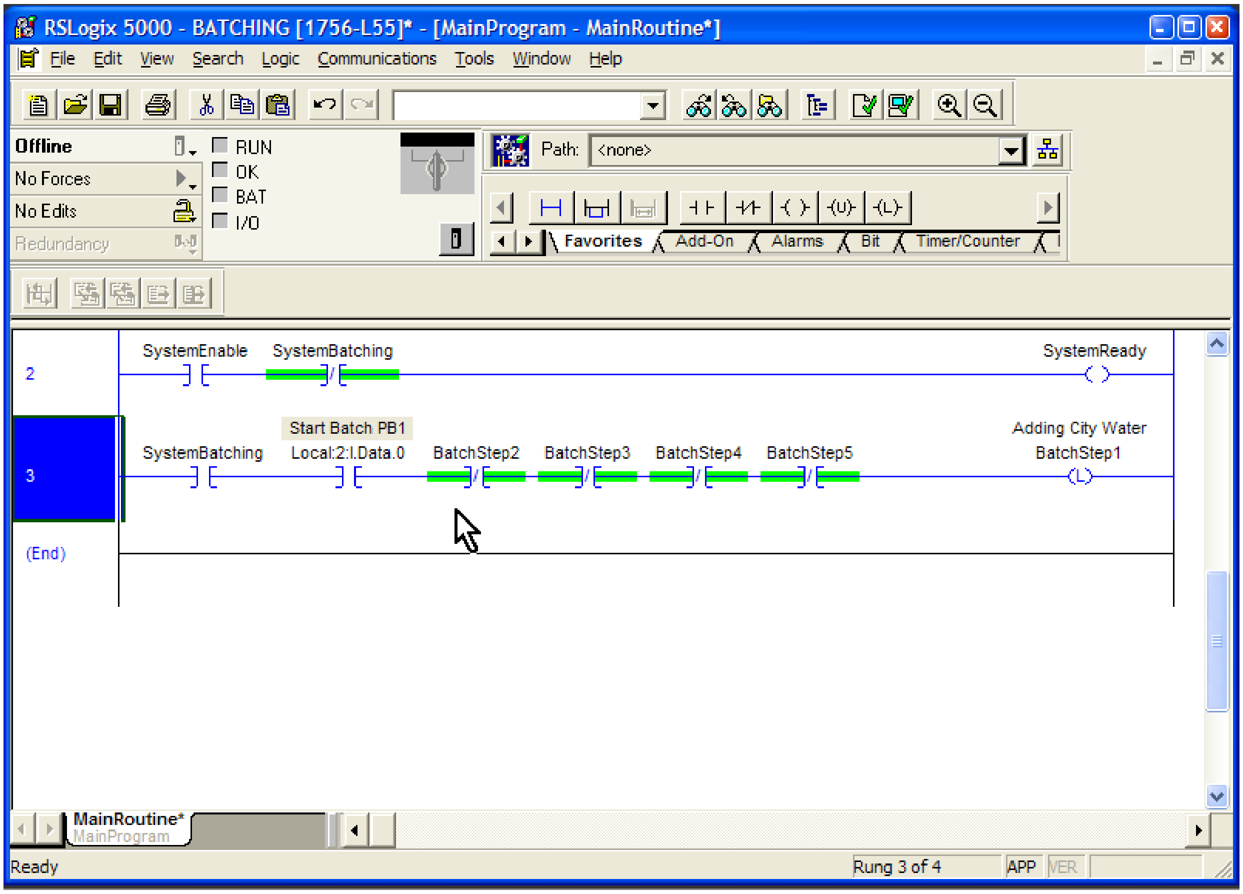
Press CTRL-V. The instruction is duplicated on Rung 3.

Click and drag the OTL (output latch) tool button -(L)- from the User menu to the right side of the new branch. Type in the tag name "BatchStep1" and define the new tag.



However, what if the button is pressed if the system is already batching and in another step? To prevent that from happening, we will make sure that the only way the system can enter Step 1 is if it is not in another step already.

Add a series of XIO instructions and tag them BatchStep2, BatchStep3, BatchStep4 and BatchStep5. Define all the new tags Type in the appropriate descriptors. Verify the rung.



You may wonder why we chose not to use the OTL output latch instruction in Rung 1. Many times, it is a matter of personal choice; sometimes a "traditional" latching rung is better than using an OTL. In Rung 1, we were able to keep all the logic affecting the SystemBatching bit on one rung. This makes it easier to read and a little more condensed. Some people view a traditional latch as a bit safer. It's your call, though.

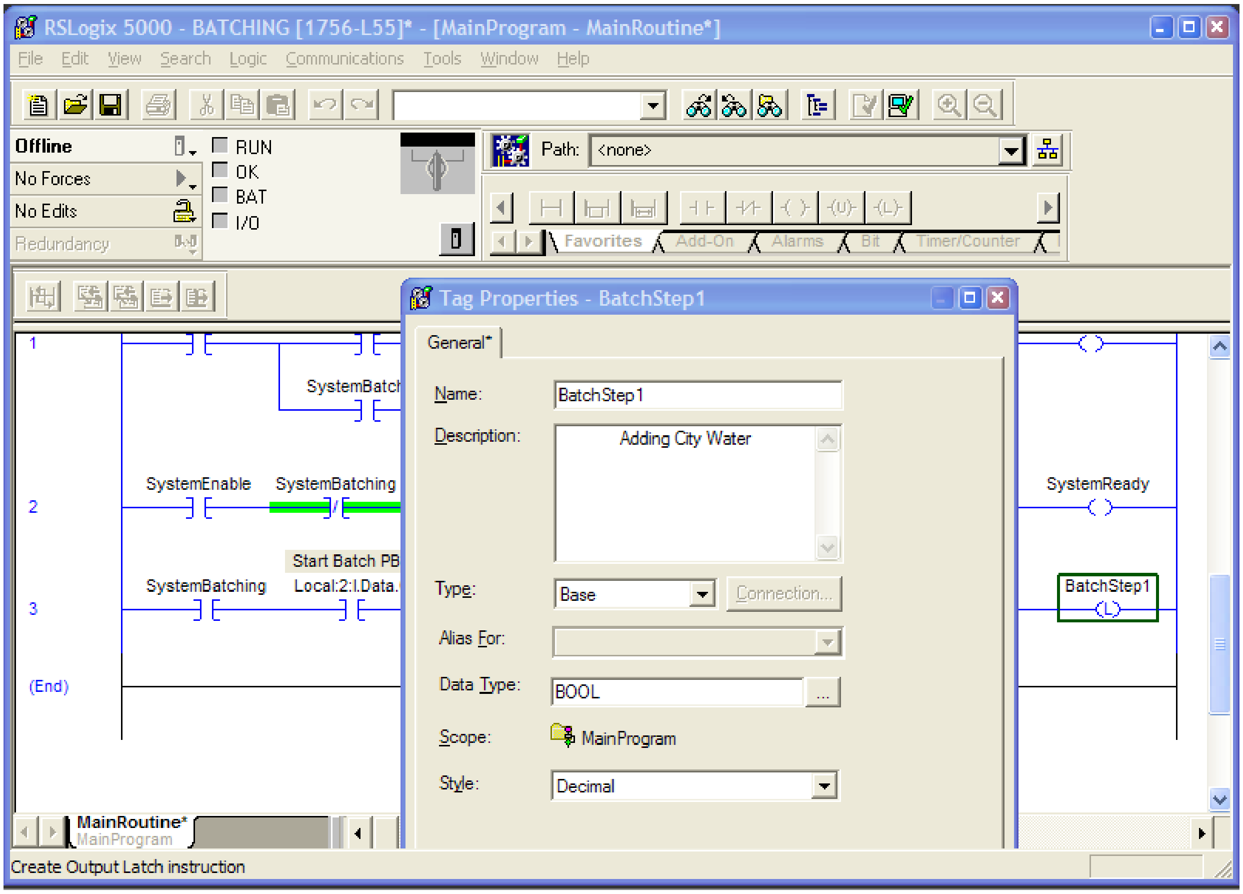
The Tag Database

In looking at Rung 3, we see that some descriptions are a bit lacking. The "SystemBatching" tag name explains what the bit does, but the "Batch Step" tag names don't tell us much. We need to add some descriptions.

You remember the actions for each Batch Step:

1. Adding City water
2. Adding chemical QR
3. Adding chemical KM
4. Blending the mixture with the agitator
5. Pumping the finished product to the filling lines.

We can right-click on the “BatchStep1” tag in the OTE and choose “Edit BatchStep1” properties. We can then add the text “Adding City Water” to the description box, as is shown below.



RSLogix 5000 - BATCHING [1756-L55]* - [MainProgram - MainRoutine*]

Tag Properties - BatchStep1

General*

Name: BatchStep1

Description: Adding City Water

Type: Base

Alias For:

Data Type: BOOL

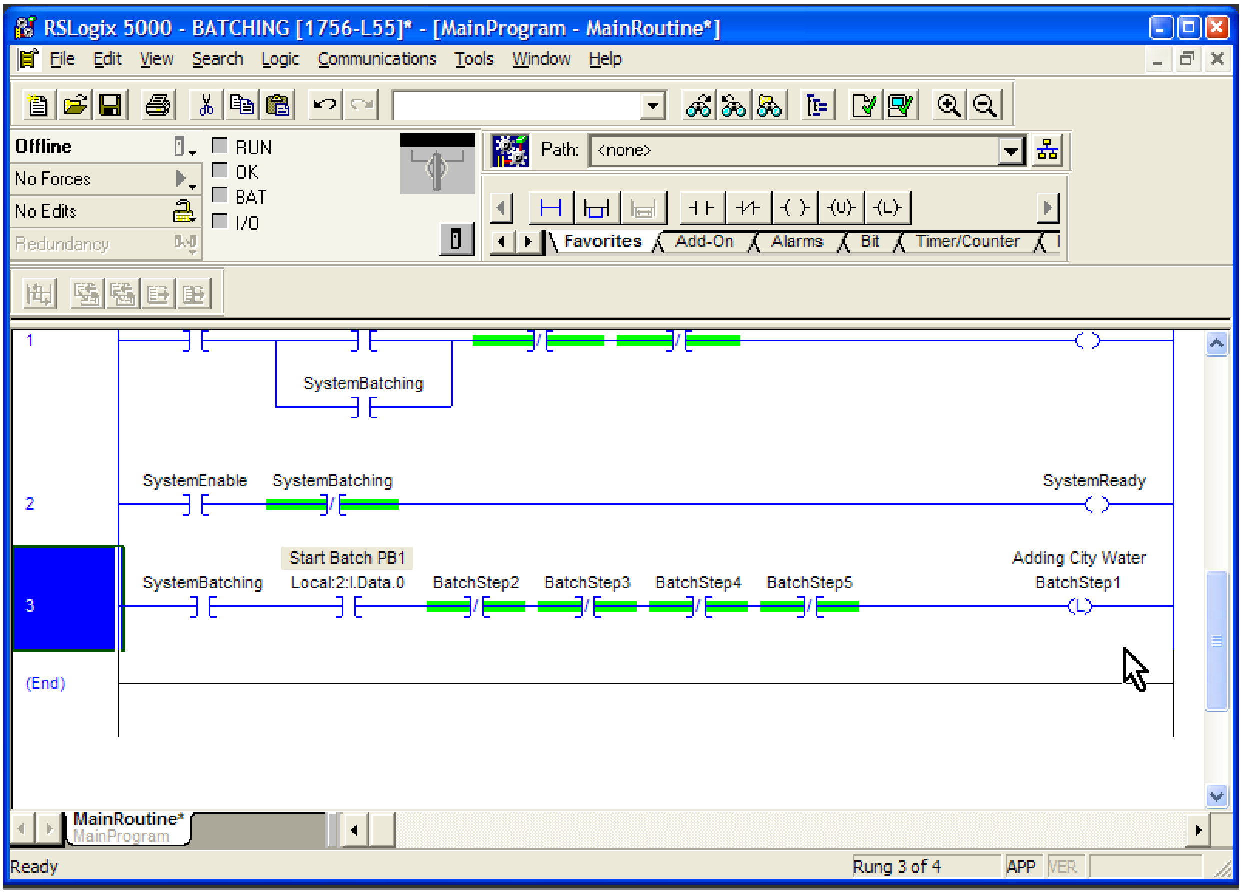
Scope: MainProgram

Style: Decimal

Create Output Latch instruction

71

The result is this:



Since we have three more tags to define, let's take a look at the tag database.

First, though, we need to know that RSLogix 5000 has different categories for tags. Rockwell calls this attribute of a tag **Data Scope**.

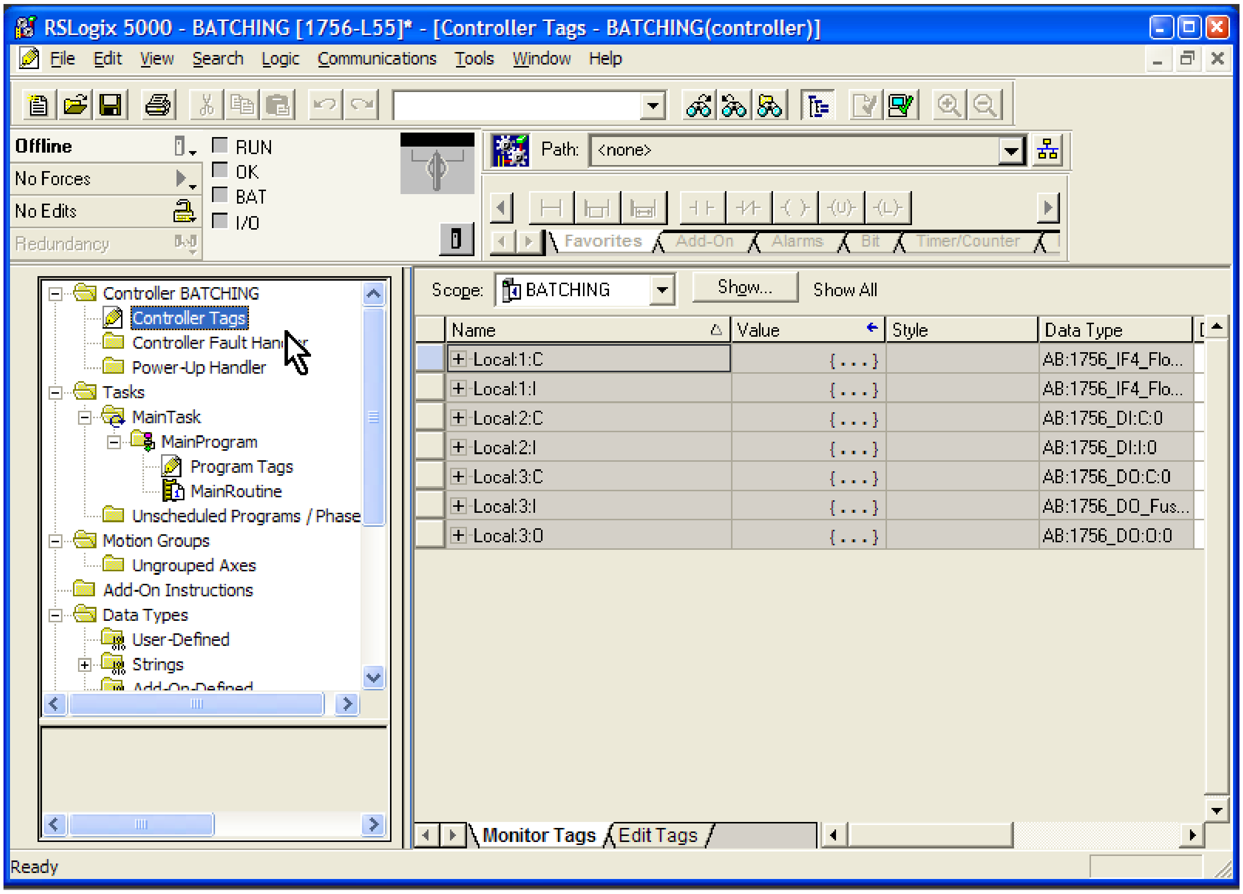
There are **controller** tags, or global tags, that can be used by all the tasks and programs in the PLC.

There are **program** tags, or local tags, that can be used only by an individual program. When tags are created as we have done, this is the default.

Let's take a look at the tags we have so far in the Tag Monitor.

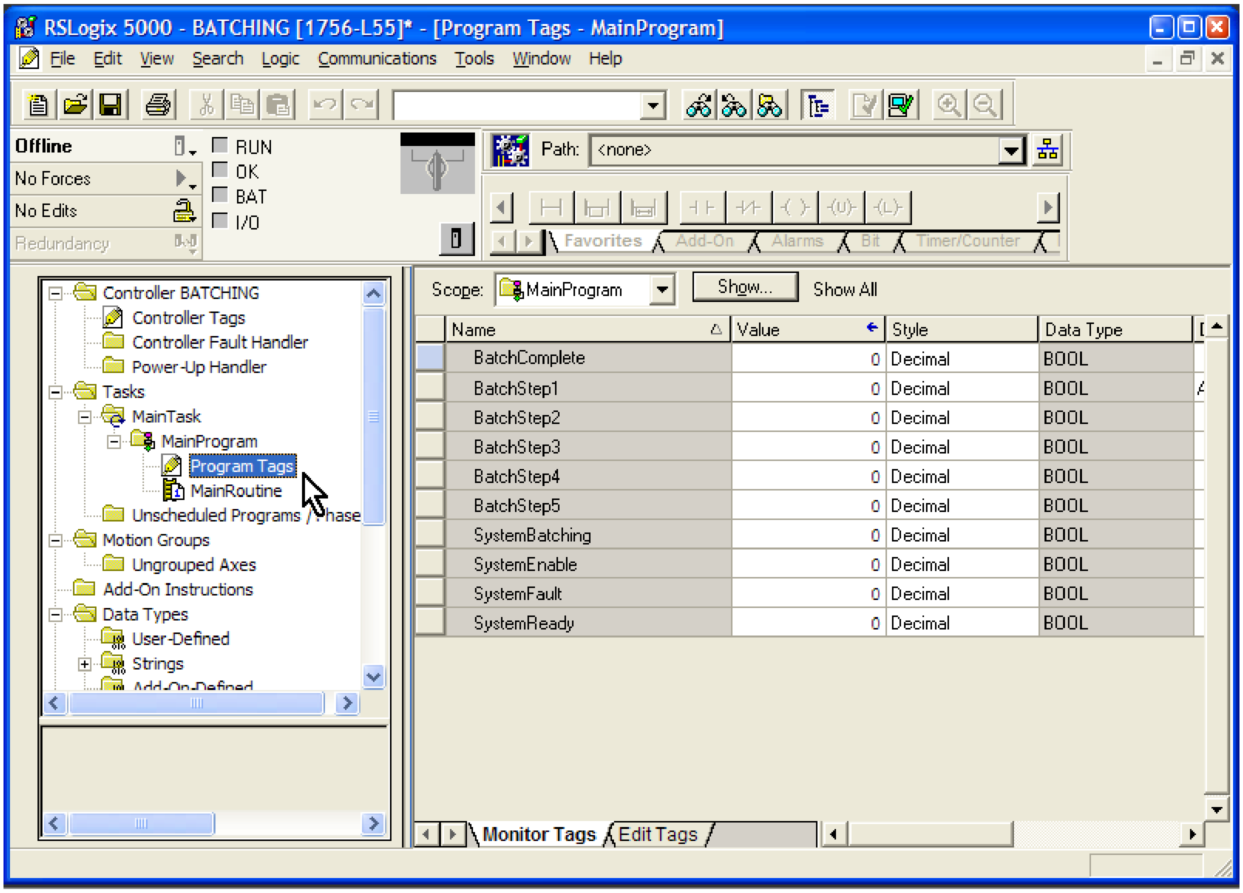
Press ALT-0 so the Controller Organizer comes back into view.

Double-click on “Controller Tags” under Controller BATCHING. These are the tag groups that were assigned when we added our I/O.



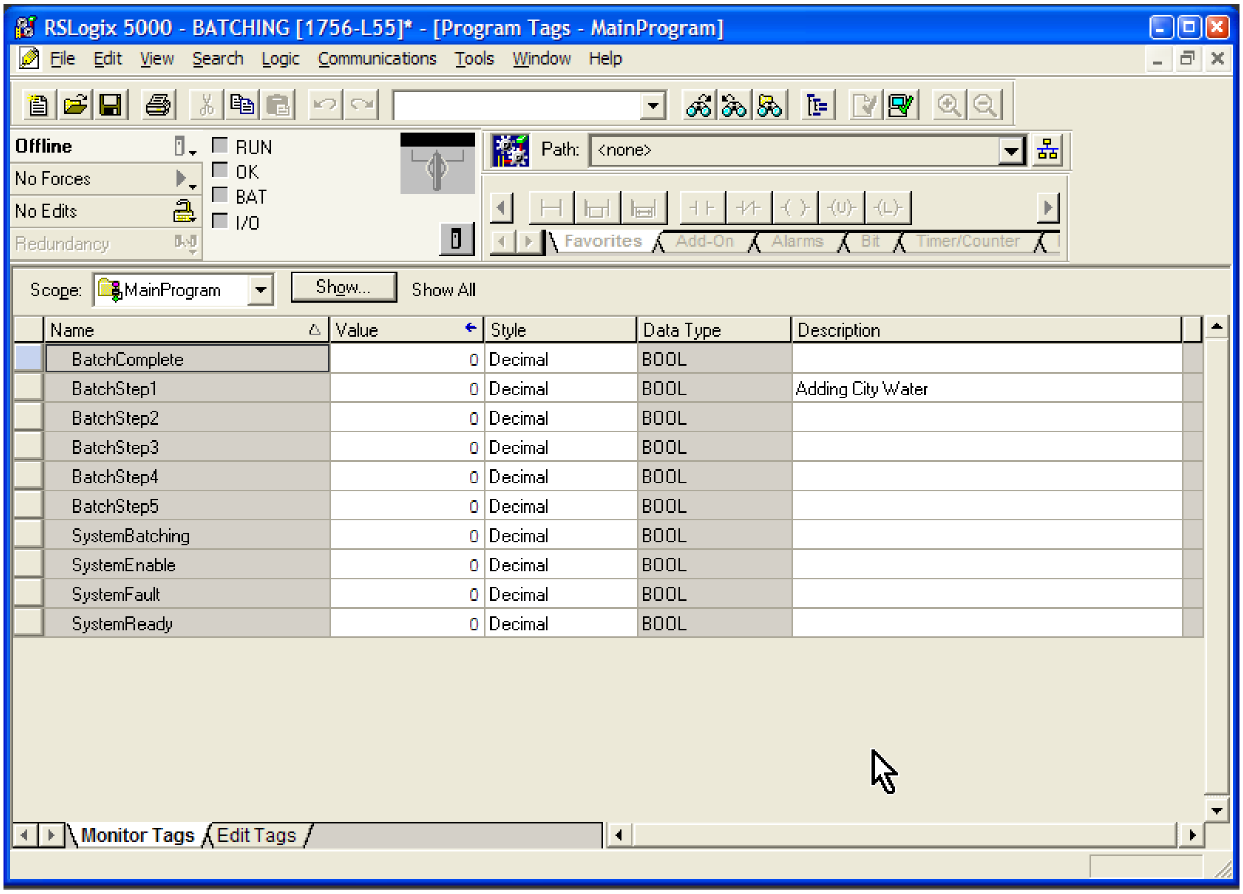
Name	Value	Style	Data Type
+ Local:1:C	{...}		AB:1756_IF4_Flo...
+ Local:1:I	{...}		AB:1756_IF4_Flo...
+ Local:2:C	{...}		AB:1756_DI:C:0
+ Local:2:I	{...}		AB:1756_DI:I:0
+ Local:3:C	{...}		AB:1756_DO:C:0
+ Local:3:I	{...}		AB:1756_DO_Fus...
+ Local:3:O	{...}		AB:1756_DO:O:0

Double-click on “Program Tags” under Tasks > Main Program. Here are the tags we created as we wrote the program.



Name	Value	Style	Data Type
BatchComplete	0	Decimal	BOOL
BatchStep1	0	Decimal	BOOL
BatchStep2	0	Decimal	BOOL
BatchStep3	0	Decimal	BOOL
BatchStep4	0	Decimal	BOOL
BatchStep5	0	Decimal	BOOL
SystemBatching	0	Decimal	BOOL
SystemEnable	0	Decimal	BOOL
SystemFault	0	Decimal	BOOL
SystemReady	0	Decimal	BOOL

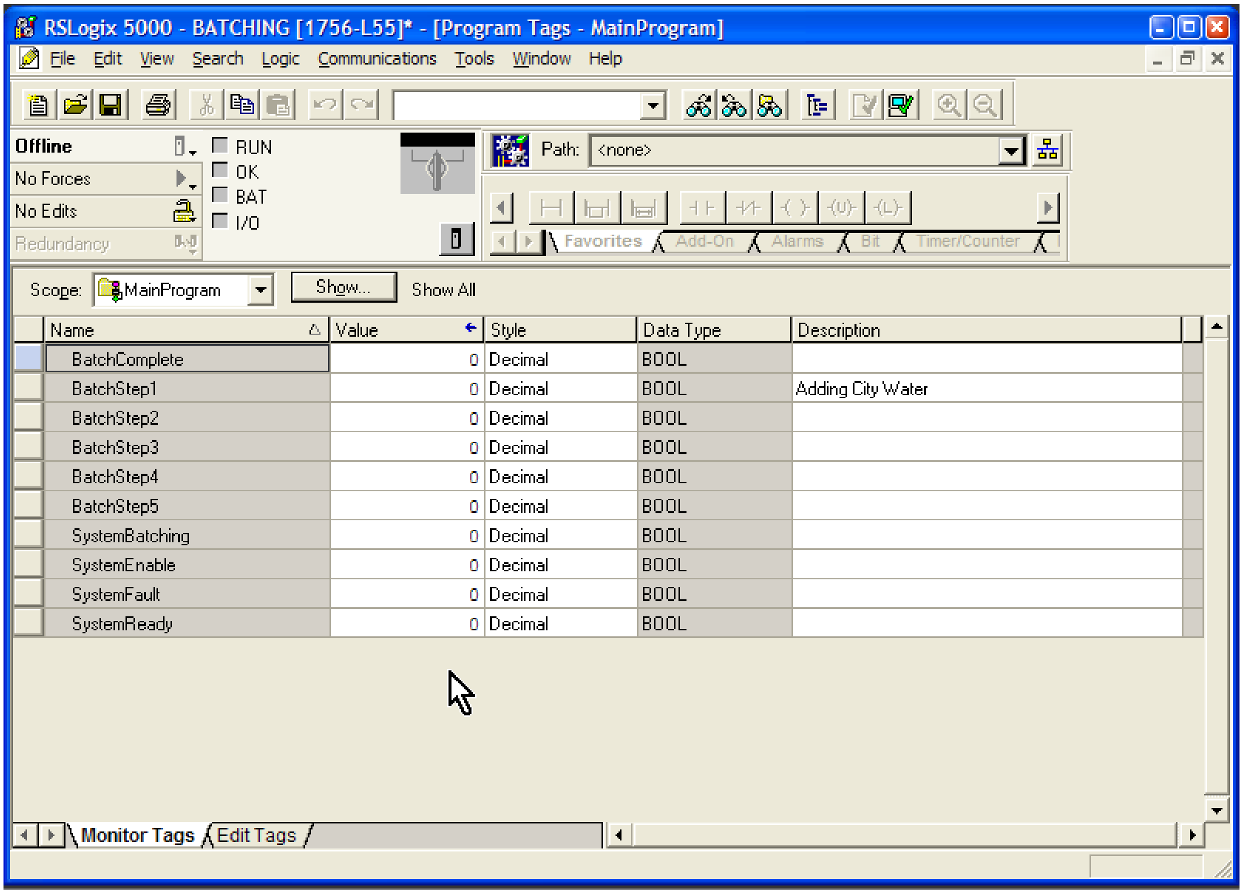
Hide the controller organizer (ALT-0) and we can see the descriptions.



Name	Value	Style	Data Type	Description
BatchComplete	0	Decimal	BOOL	
BatchStep1	0	Decimal	BOOL	Adding City Water
BatchStep2	0	Decimal	BOOL	
BatchStep3	0	Decimal	BOOL	
BatchStep4	0	Decimal	BOOL	
BatchStep5	0	Decimal	BOOL	
SystemBatching	0	Decimal	BOOL	
SystemEnable	0	Decimal	BOOL	
SystemFault	0	Decimal	BOOL	
SystemReady	0	Decimal	BOOL	

Let's add the rest of the batch step descriptions.

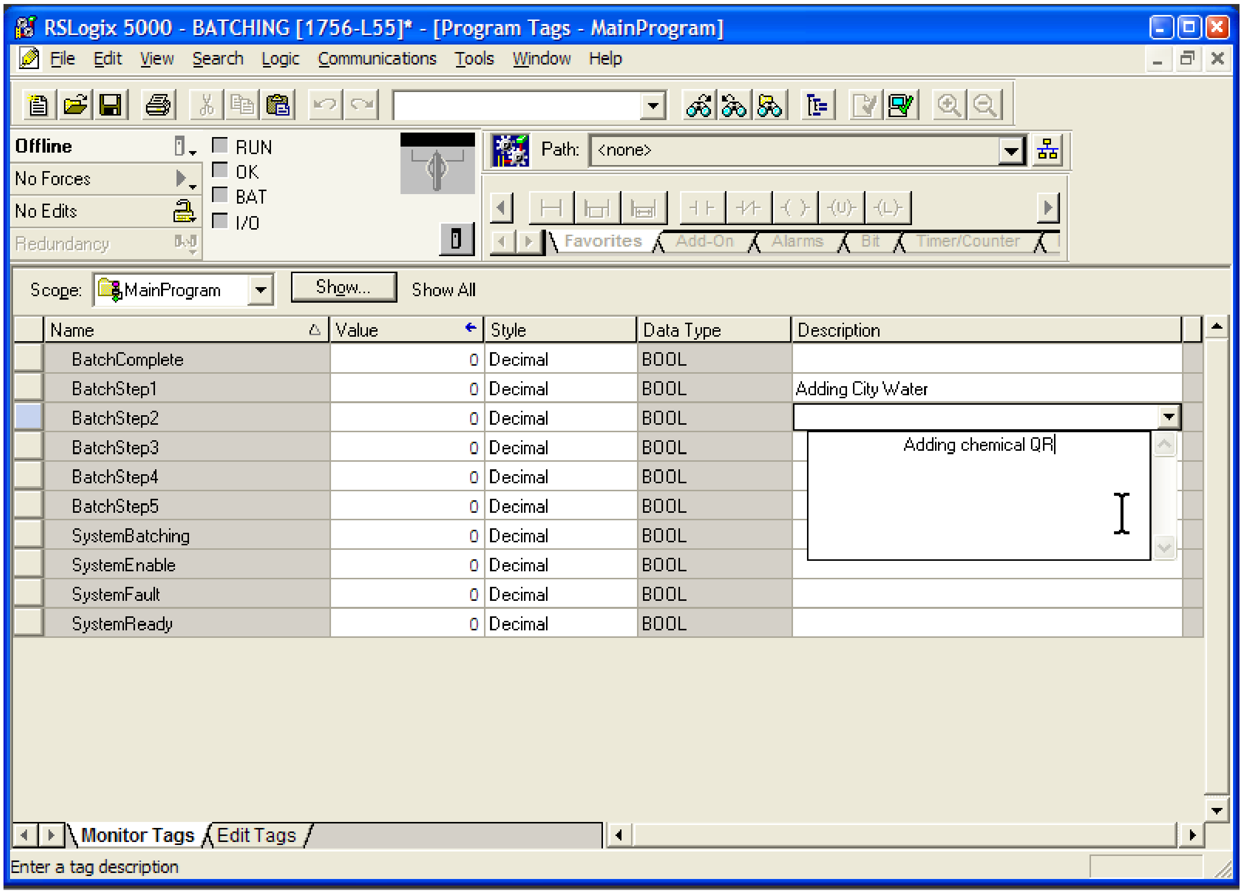
Click on the "Edit Tags" tab at the bottom.



Name	Value	Style	Data Type	Description
BatchComplete	0	Decimal	BOOL	
BatchStep1	0	Decimal	BOOL	Adding City Water
BatchStep2	0	Decimal	BOOL	
BatchStep3	0	Decimal	BOOL	
BatchStep4	0	Decimal	BOOL	
BatchStep5	0	Decimal	BOOL	
SystemBatching	0	Decimal	BOOL	
SystemEnable	0	Decimal	BOOL	
SystemFault	0	Decimal	BOOL	
SystemReady	0	Decimal	BOOL	

You'll notice that some of the fields in the table are different.

Select the “Description” field for BatchStep2 and type in the description.



RSLogix 5000 - BATCHING [1756-L55]* - [Program Tags - MainProgram]

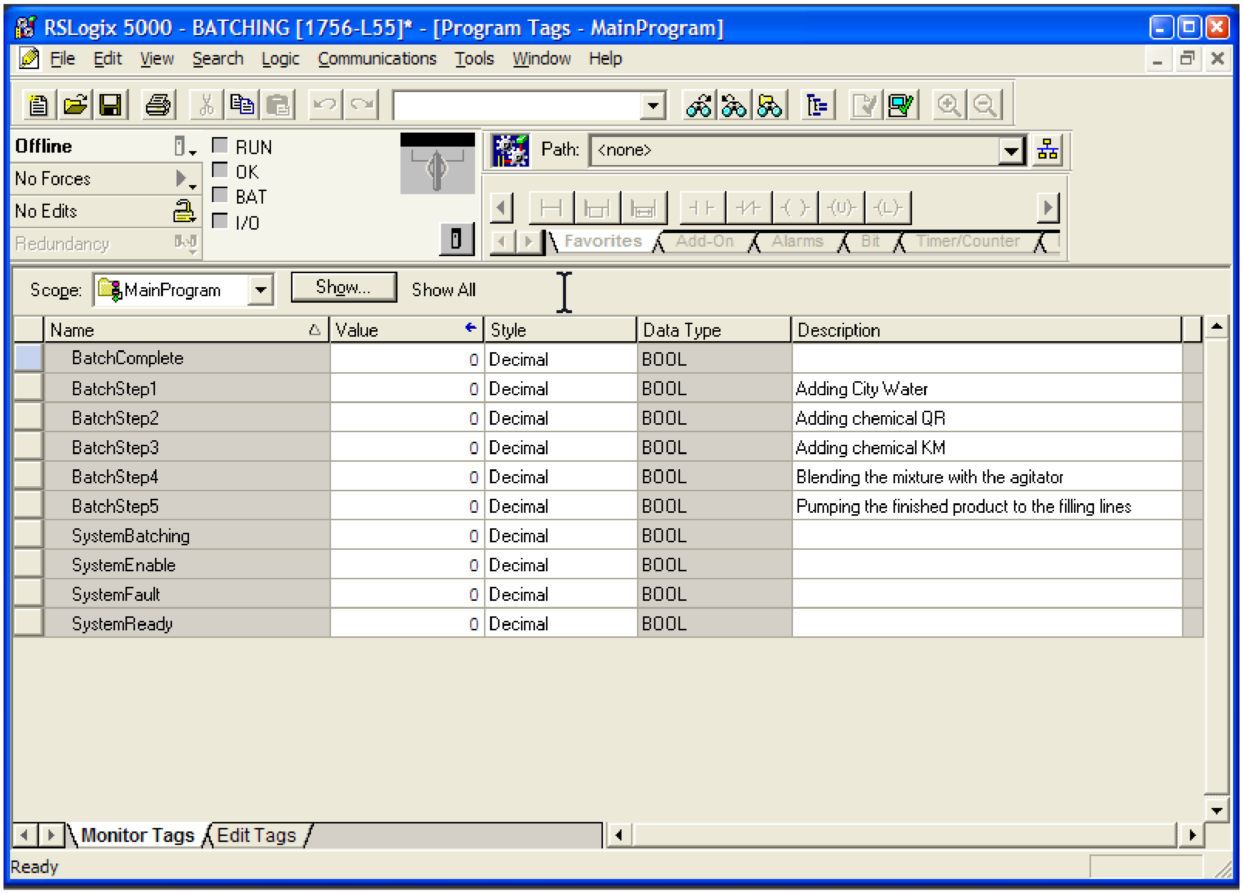
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
Favorites Add-On Alarms Bit Timer/Counter
Show...
Show All
Monitor Tags
Edit Tags

Scope: MainProgram

Name	Value	Style	Data Type	Description
BatchComplete	0	Decimal	BOOL	
BatchStep1	0	Decimal	BOOL	Adding City Water
BatchStep2	0	Decimal	BOOL	Adding chemical QR
BatchStep3	0	Decimal	BOOL	
BatchStep4	0	Decimal	BOOL	
BatchStep5	0	Decimal	BOOL	
SystemBatching	0	Decimal	BOOL	
SystemEnable	0	Decimal	BOOL	
SystemFault	0	Decimal	BOOL	
SystemReady	0	Decimal	BOOL	

Enter a tag description

Click on the description field for each batch step and add the respective description.


RSLogix 5000 - BATCHING [1756-L55]* - [Program Tags - MainProgram]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
Favorites Add-On Alarms Bit Timer/Counter
Show...
Show All
Monitor Tags
Edit Tags
Ready

Scope: MainProgram

Name	Value	Style	Data Type	Description
BatchComplete	0	Decimal	BOOL	
BatchStep1	0	Decimal	BOOL	Adding City Water
BatchStep2	0	Decimal	BOOL	Adding chemical QR
BatchStep3	0	Decimal	BOOL	Adding chemical KM
BatchStep4	0	Decimal	BOOL	Blending the mixture with the agitator
BatchStep5	0	Decimal	BOOL	Pumping the finished product to the filling lines
SystemBatching	0	Decimal	BOOL	
SystemEnable	0	Decimal	BOOL	
SystemFault	0	Decimal	BOOL	
SystemReady	0	Decimal	BOOL	

One note, though; use care when you are editing tags in this table. Many times, there is no undo available.

Analog Inputs

Before we continue with Step 1, we need to scale the analog cards for the Mixing Tank Scales and the Ultrasonic Level Sensor.

There are a couple of ways to scale a value we get from an analog input in RSLogix 5000. We can scale the value within the program, or we can do the scaling right in the card.

There are advantages and disadvantages to both methods. Scaling in the card is simple and straightforward, but once the card is configured, the configuration cannot be changed unless the PLC is taken offline. This is not good for processes that must run continuously.

Scaling in the program is more difficult, but adjustments to the scaling algorithm can be made while the PLC is still running.

In our case, as in many batching applications, the process does not have to run continuously. For example, when the mixing tank is full, we could shut down the PLC and manually run the pump that empties the tank. This “window” would give us more than enough time to re-configure the analog card.

To configure our 1756-IF8 analog input card, we need to know what signal type we have coming from our sensors, the range of the signals we get from our sensors and the engineering units for each sensor.

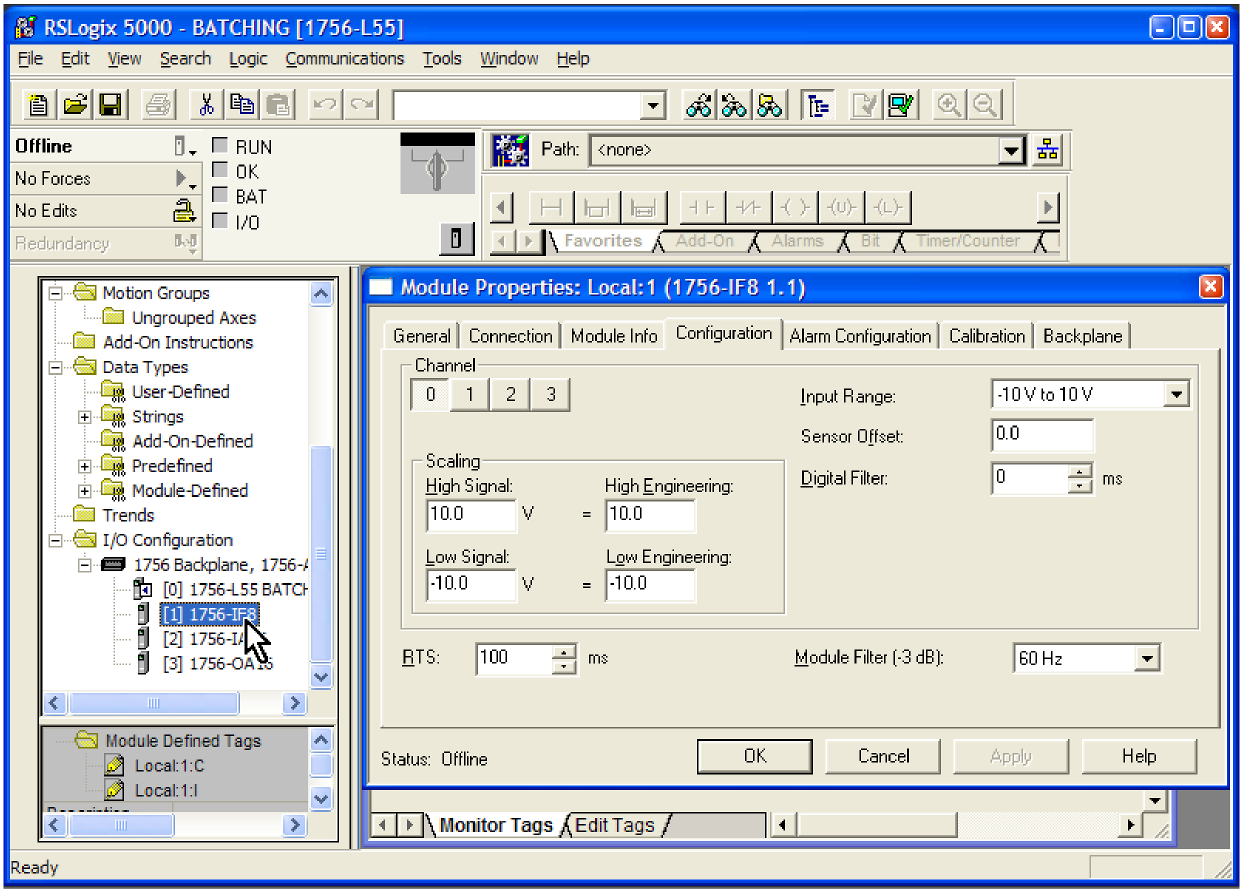
Let’s start with the scales.

Setting up the Analog Input Card to Calculate Tank Weight

Scales are for these types of applications usually consist of a standalone unit that is calibrated by the manufacturer of the scales. The unit usually has a display that shows the actual weight and an output that can be fed to a PLC. Let’s assume that the output of our unit has been calibrated for 0-10 VDC. Zero volts equals 0 pounds, and 10 volts equals 2000 pounds.

Now we know the signal type is DC voltage, the range is 0-10 and the engineering units are pounds.

Right-click on the 1756-IF8 card in the Controller Organizer and choose “Properties”. Click on the “Configuration” tab and you will see this.



Module Properties: Local:1 (1756-IF8 1.1)

Channel: 0

Input Range: -10 V to 10 V

Sensor Offset: 0.0

Digital Filter: 0 ms

Scaling

High Signal: 10.0 V = High Engineering: 10.0

Low Signal: -10.0 V = Low Engineering: -10.0

RTS: 100 ms

Module Filter (-3 dB): 60 Hz

Status: Offline

You’ll see that Channel 0, which is our Scales channel, is selected.

Click on the dropdown menu for “Input Range” and select “0V to 10V”.

Change the “Low Signal” field to 0.

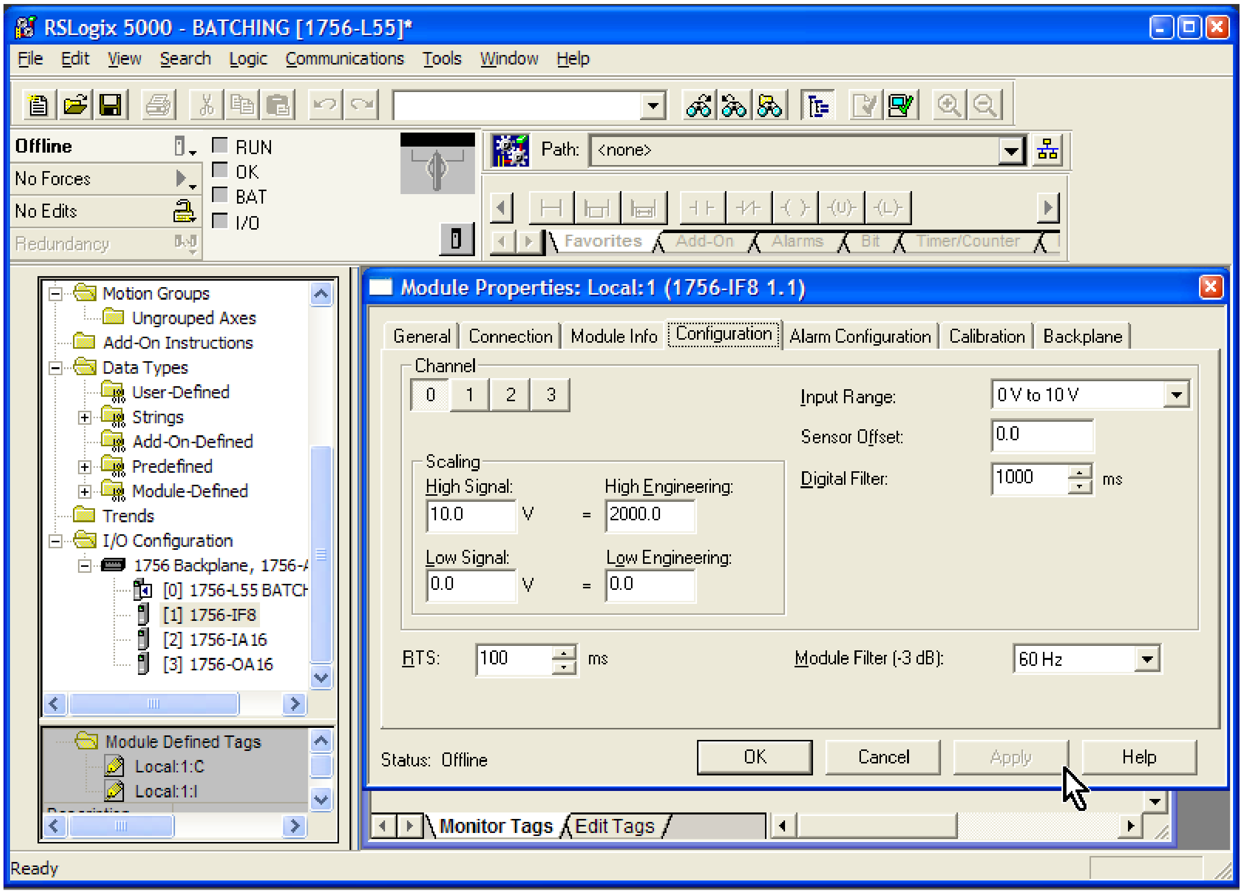
Change the “High Engineering” field to 2000.

Change the “Low Engineering” field to 0.

That is all we really need to do. However, we are going to take advantage of the fact that there is a filter available. This filter smoothes input transitions.

Set the “Digital Filter” field to 1000 ms.

Click “Apply” and we are done with the scales.



Input Range: 0 V to 10 V

Sensor Offset: 0.0

Digital Filter: 1000 ms

High Signal: 10.0 V = High Engineering: 2000.0

Low Signal: 0.0 V = Low Engineering: 0.0

RTS: 100 ms

Module Filter (-3 dB): 60 Hz

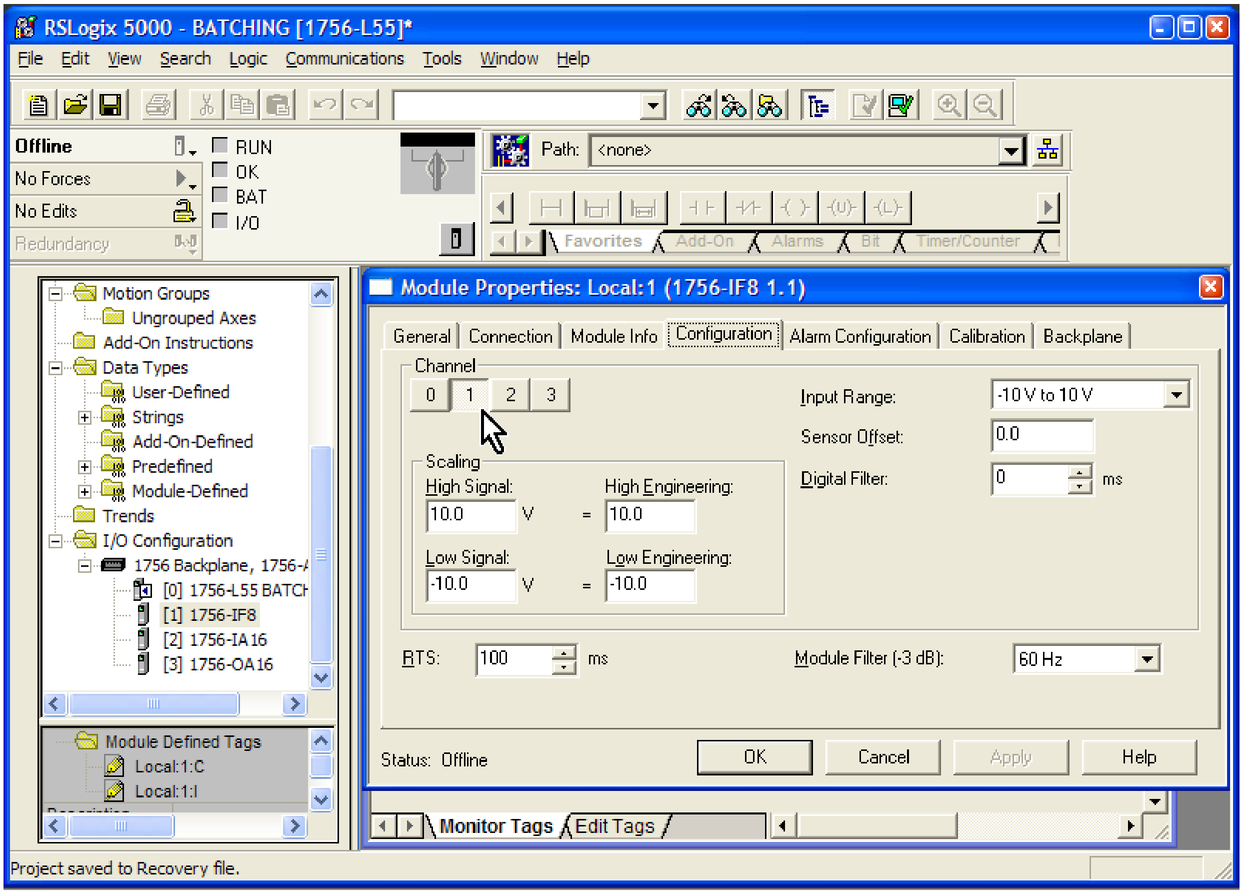
Status: Offline

Setting up the Analog Input Card to Calculate Tank Level

Like the scales, the ultrasonic level sensor is calibrated to a 0 to 10VDC signal. Zero volts indicate the tank is empty. Ten volts indicates the tank is at its “full “ mark, hopefully a safe distance from its actual maximum capacity.

The engineering unit used for the tank level is percentage.

Select channel 1 on the card.



RSLogix 5000 - BATCHING [1756-L55]*

Module Properties: Local:1 (1756-IF8 1.1)

Channel: 1

Scaling

High Signal: 10.0 V = High Engineering: 10.0

Low Signal: -10.0 V = Low Engineering: -10.0

Input Range: -10 V to 10 V

Sensor Offset: 0.0

Digital Filter: 0 ms

RTS: 100 ms

Module Filter (-3 dB): 60 Hz

Status: Offline

Project saved to Recovery file.

Click on the dropdown menu for “Input Range” and select “0V to 10V”.

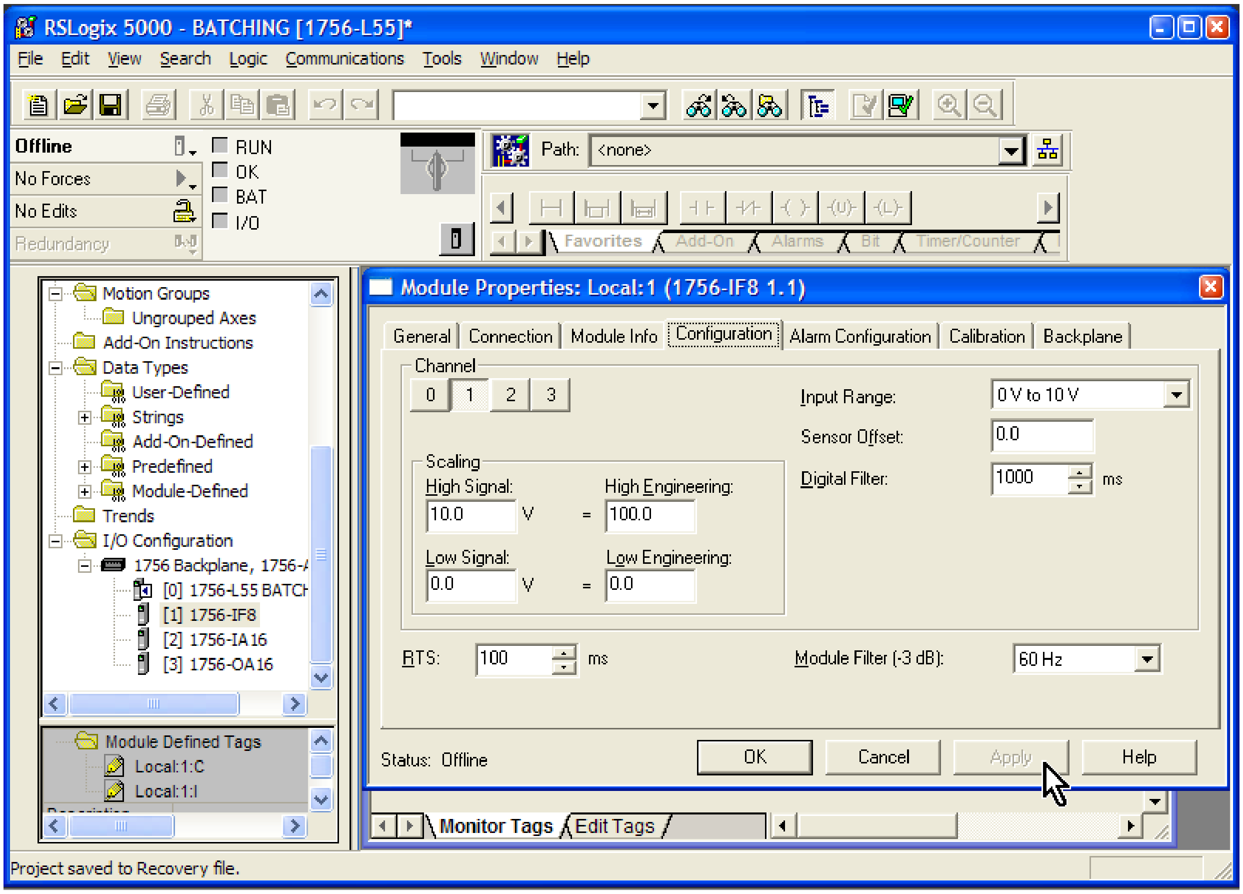
Change the “Low Signal” field to 0.

Change the “High Engineering” field to 100.

Change the “Low Engineering” field to 0.

Again, that is all we really need to do, but we will set the “Digital Filter” field to 1000 ms.

Click “Apply”.



Module Properties: Local:1 (1756-IF8 1.1)

Channel: 0 [1] 2 3

Input Range: 0 V to 10 V

Sensor Offset: 0.0

Digital Filter: 1000 ms

Scaling

High Signal: 10.0 V = High Engineering: 100.0

Low Signal: 0.0 V = Low Engineering: 0.0

RTS: 100 ms

Module Filter (-3 dB): 60 Hz

Status: Offline

Project saved to Recovery file.

Back to Batching – Step 1

Double-click on the “MainRoutine” in the Controller Organizer, and then hide the organizer.

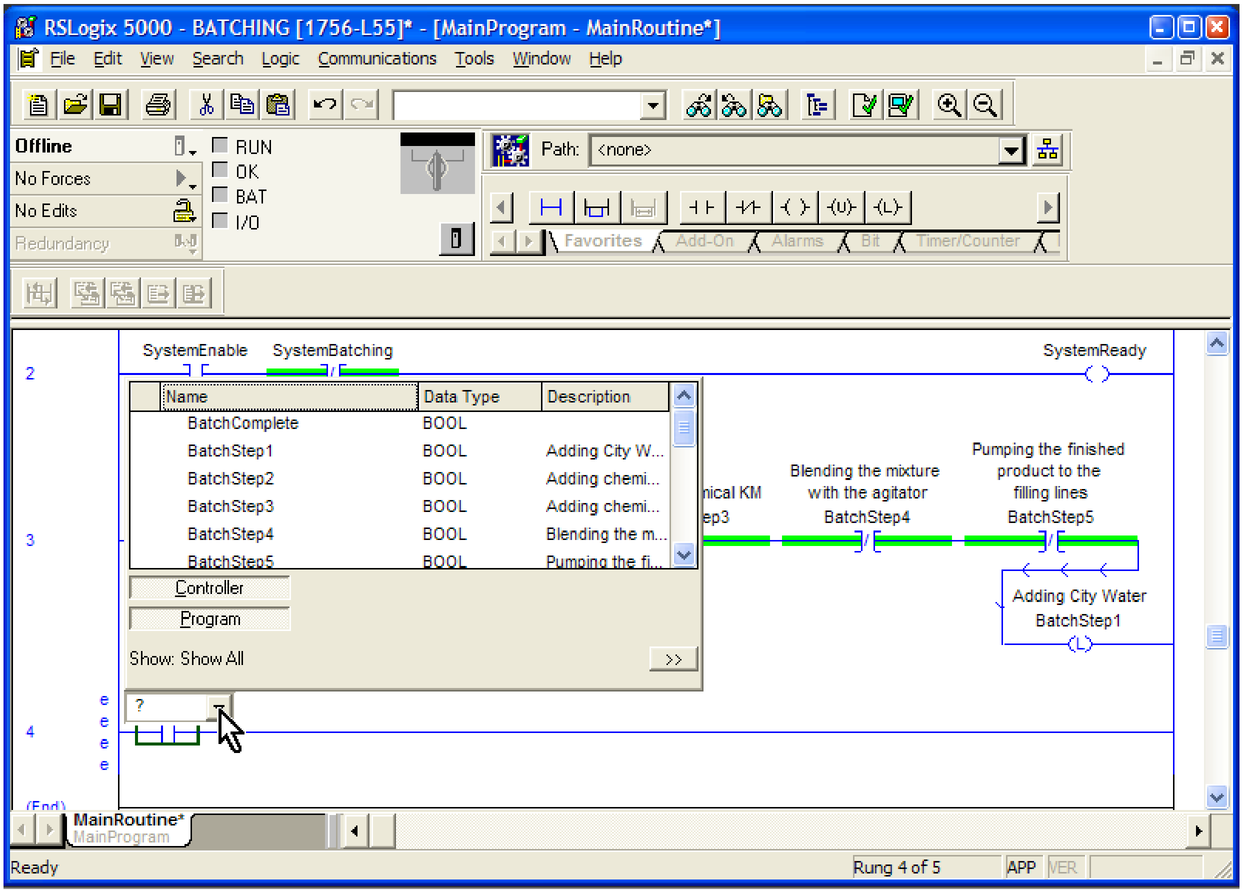
Scroll down and add a new rung at the bottom of the ladder.

We will open the city water valve in this rung, so we want to make sure that it is still safe and desirable to open the valve. That is, make sure that there are no faults, the E-Stop button has not been pressed and the Stop Batch pushbutton has not been pressed. Using an XIC with the SystemBatching bit will confirm all of that.

Insert an XIC. We could drag the tag “SystemBatching” from Rung 3, but let’s add it by using the dropdown tag menu on the instruction.

Double-click on the tag name field above the instruction. At this point, you could type the tag name and press enter to assign the tag to this instruction.

Let's use the dropdown, though. Click on the arrow in the dropdown header and you see this.


RSLogix 5000 - BATCHING [1756-L55]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
Favorites
Add-On
Alarms
Bit
Timer/Counter
SystemEnable
SystemBatching
SystemReady
Blending the mixture with the agitator BatchStep4
Pumping the finished product to the filling lines BatchStep5
Adding City Water BatchStep1
Controller
Program
Show: Show All
>>
?
MainRoutine*
MainProgram
Ready
Rung 4 of 5
APP
VER

Name	Data Type	Description
BatchComplete	BOOL	
BatchStep1	BOOL	Adding City W...
BatchStep2	BOOL	Adding chemi...
BatchStep3	BOOL	Adding chemi...
BatchStep4	BOOL	Blending the m...
BatchStep5	BOOL	Pumping the fi...

Scroll down until you find the "SystemBatching" tag. Select it and press enter.

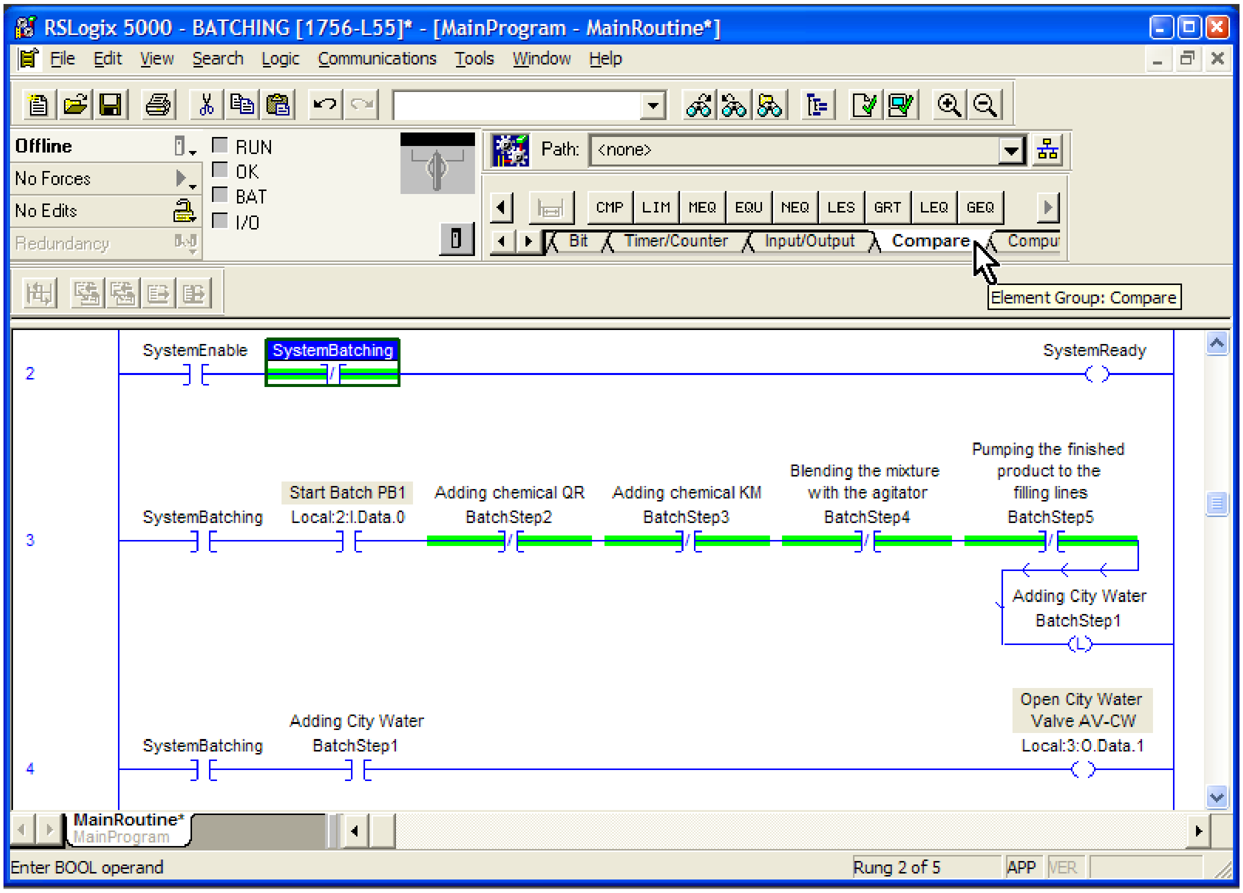
RSLogix gives you a number of ways to assign a tag name. Use whatever is easiest for the particular situation.

Since we only want to open this valve and add city water in Step 1, insert an XIC with the tag "BatchStep1".

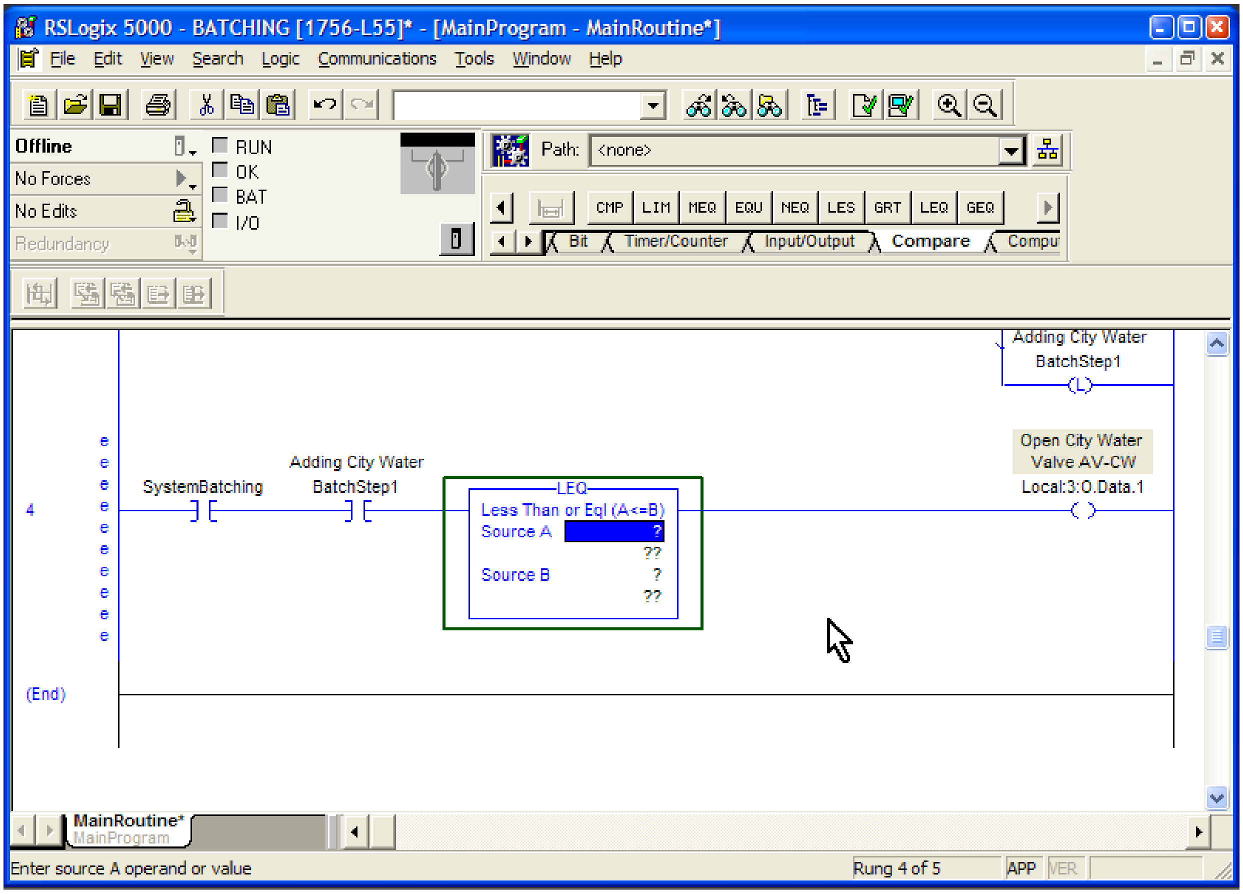
Insert an OTE for the city water valve (Local:3:O.Data.1).

Remember that we want to put 1275 lbs. of water in the Mixing Tank. We will use the LEQ (Less than or Equal To) instruction to accomplish that.

Scroll to the right in the Language Element toolbar until you see the “Compare” tab. Click on it.



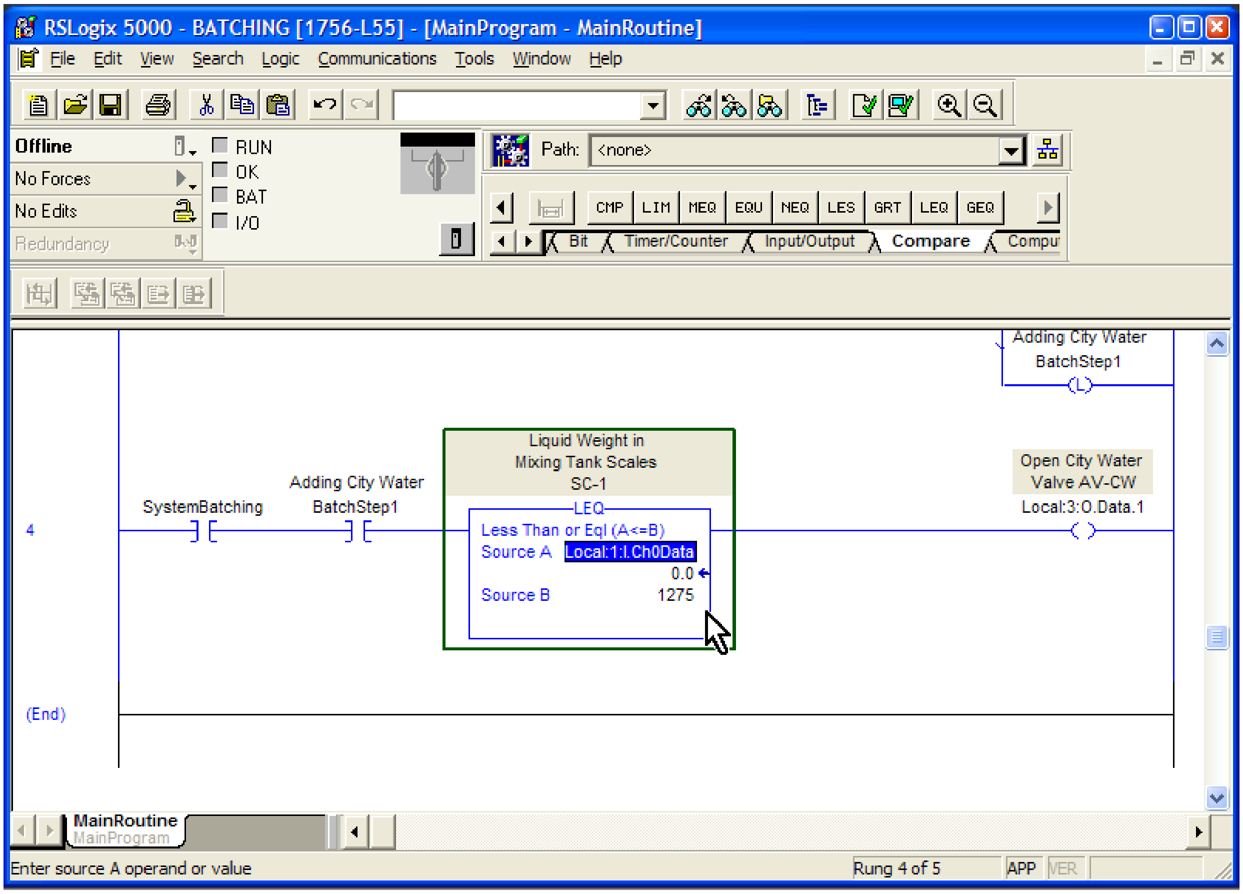
Click and drag the LEQ tool button to the marker just to the left of the OTE.



Source A in the instruction is the Tank Weight. Click on the blue tag field and scroll through the tags until you find Local:1:I.Ch0Data (Liquid Weight in Mixing Tank Scales). Press enter.

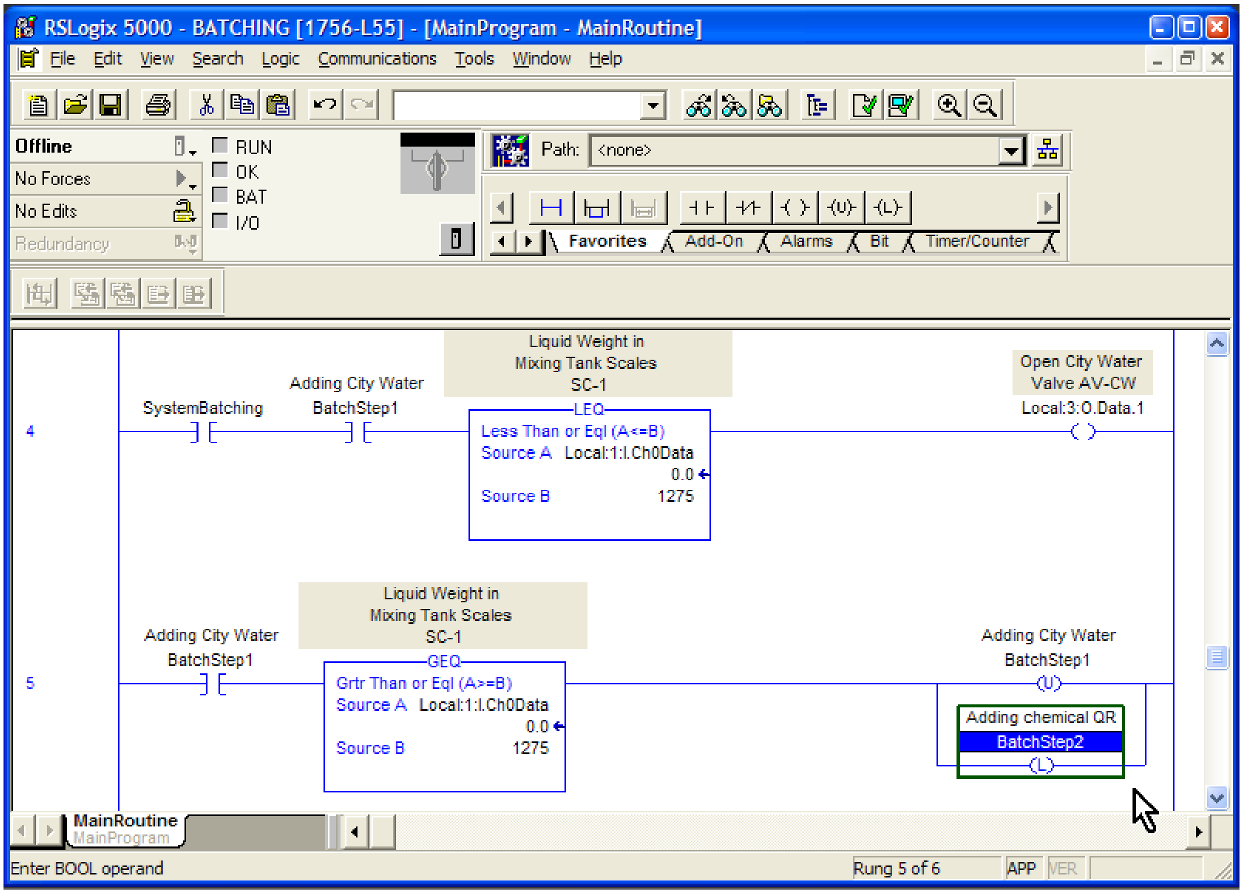
Source B is our setpoint, which as you recall from the Project Scope is 1275 lbs. Enter 1275 for Source B.

The LEQ instruction will remain true as long as the tank weight does not exceed 1275 lbs.



After the correct amount of city water has been added, we need to proceed to Step 2.

We will make use of the fact that we know the system is currently in Step 1, but the Mixing Tank has enough water (1275 lbs) to go to the next step.



We will use the OTU (Output Unlatch) instruction to turn off the bit we latched in Rung 3.

Rung 5 works like this:

The XIC instruction “BatchStep1” is on. The Mixing Tank weight has reached the setpoint of 1275, so the GEQ (Greater Than or Equal To) instruction is also true. As a result, the bit “BatchStep1” is unlatched (turned off) and “BatchStep2” is latched (turned on).

Take a moment to make sure you understand how as the weight in the tank rises past 1275, the City Water valve is turned off and the system transitions to Step 2.

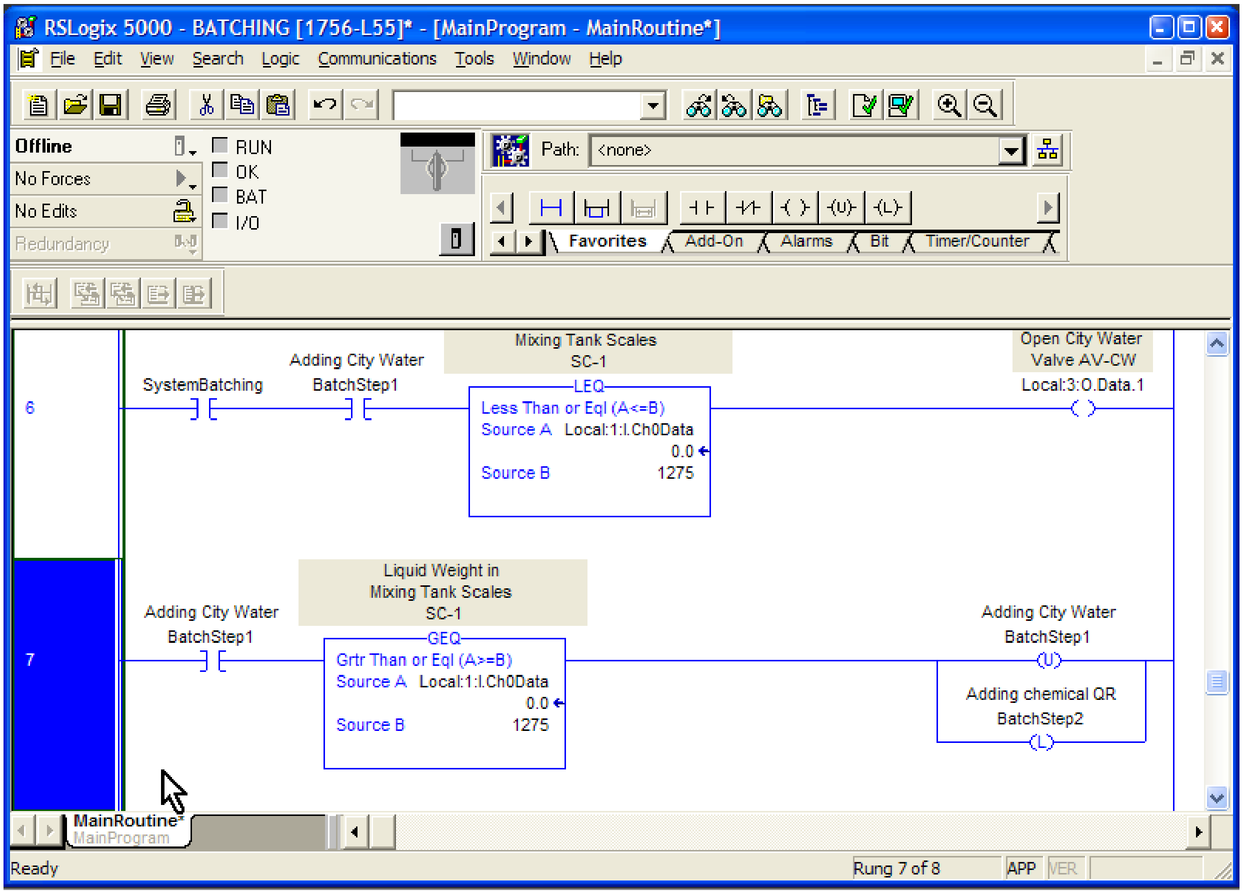
Step 2 – Adding Chemical KM

Step 2 will be similar to Step 1, so rather than creating new rungs from scratch, we are going to copy and paste the rungs from Step 1.

Click on the rung number for Rung 4. Hold the SHIFT key and click on the rung number for Rung 5. A green bar surrounds both rung number blocks.

Press CTRL-C.

Press CTRL-V. You now have new rungs, 6 and 7.



We will start with rung 6.

Change “BatchStep1” to “BatchStep2” (in this instance, just double-click on the tag name and change the “1” to “2” and press enter).

We will be looking for a setpoint of 1275 + 390, since there is already 1275 lbs. of water in the Mixing Tank and we need to add 390 lbs. of QR. Change Source B in the LEQ instruction to 1665.

Refer to your I/O list spreadsheet and note that the QR automatic valve is addressed as Local:3:O.Data.3. Assign that to the OTE.

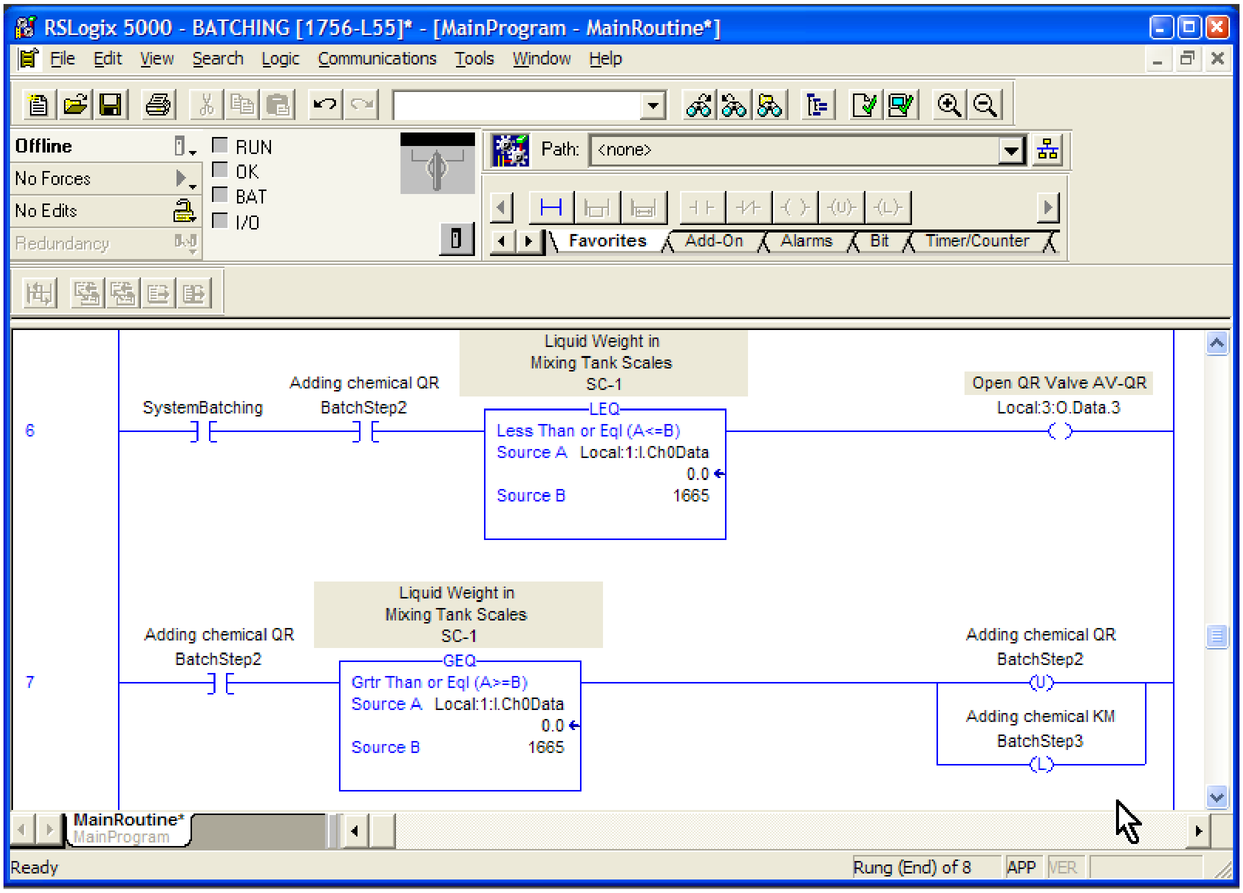
In Rung 7, change “BatchStep1” to “BatchStep2”.

Change Source B in the GEQ instruction to 1665.

Change the OTU (Output Unlatch) instruction from “BatchStep1” to “BatchStep2”.

Change the OTL (Output Latch) instruction from “BatchStep2” to “BatchStep3”.

It should look like this.



RSLogix 5000 - BATCHING [1756-L55]* - [MainProgram - MainRoutine*]

Rung 6: SystemBatching, Adding chemical QR BatchStep2, LEQ Less Than or Eql (A<=B) — Liquid Weight in Mixing Tank Scales SC-1 — Source A Local:1:I.Ch0Data 0.0, Source B 1665 — Open QR Valve AV-QR Local:3:O.Data.3

Rung 7: Adding chemical QR BatchStep2, GEQ Grtr Than or Eql (A>=B) — Liquid Weight in Mixing Tank Scales SC-1 — Source A Local:1:I.Ch0Data 0.0, Source B 1665 — Adding chemical QR BatchStep2 (U), Adding chemical KM BatchStep3 (L)

There is a major process difference between adding City Water and adding QR – the QR chemical needs to be pumped. This means that not only do we have to open a valve, but we also have to turn on a pump.

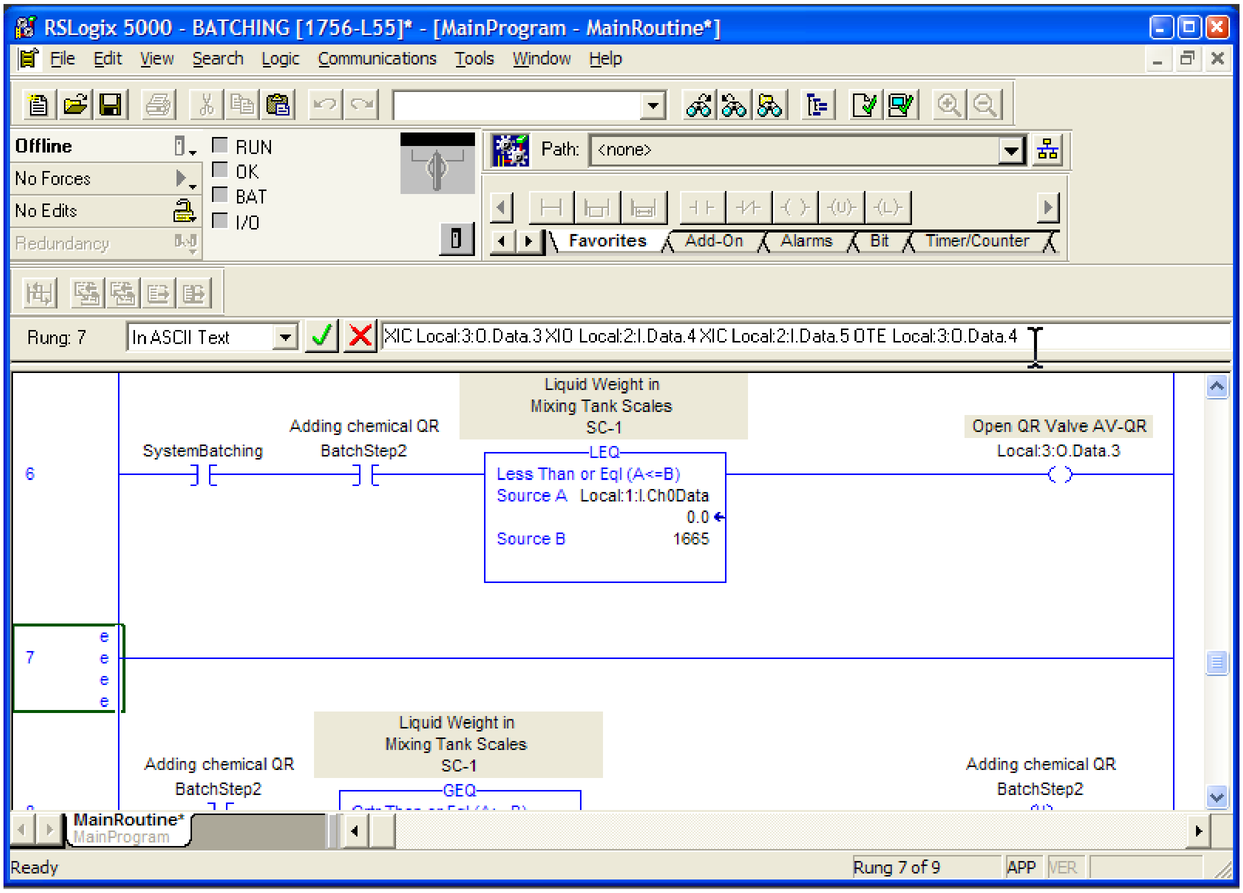
First, we want to make sure the valve is being instructed to open, so we will use an XIC from pump AV-QR in the logic. Next, we will wait to turn on the pump until the valve is verified to be open by limit switch LS-QR2. As a failsafe, we will look at limit switch LS-QR1 to make sure it is *not* indicating the valve is closed.

Right-click on Rung 6 and choose “Add Rung”.

So that you can learn a bit more about ASCII editing, we will construct the entire rung from the ASCII command line.

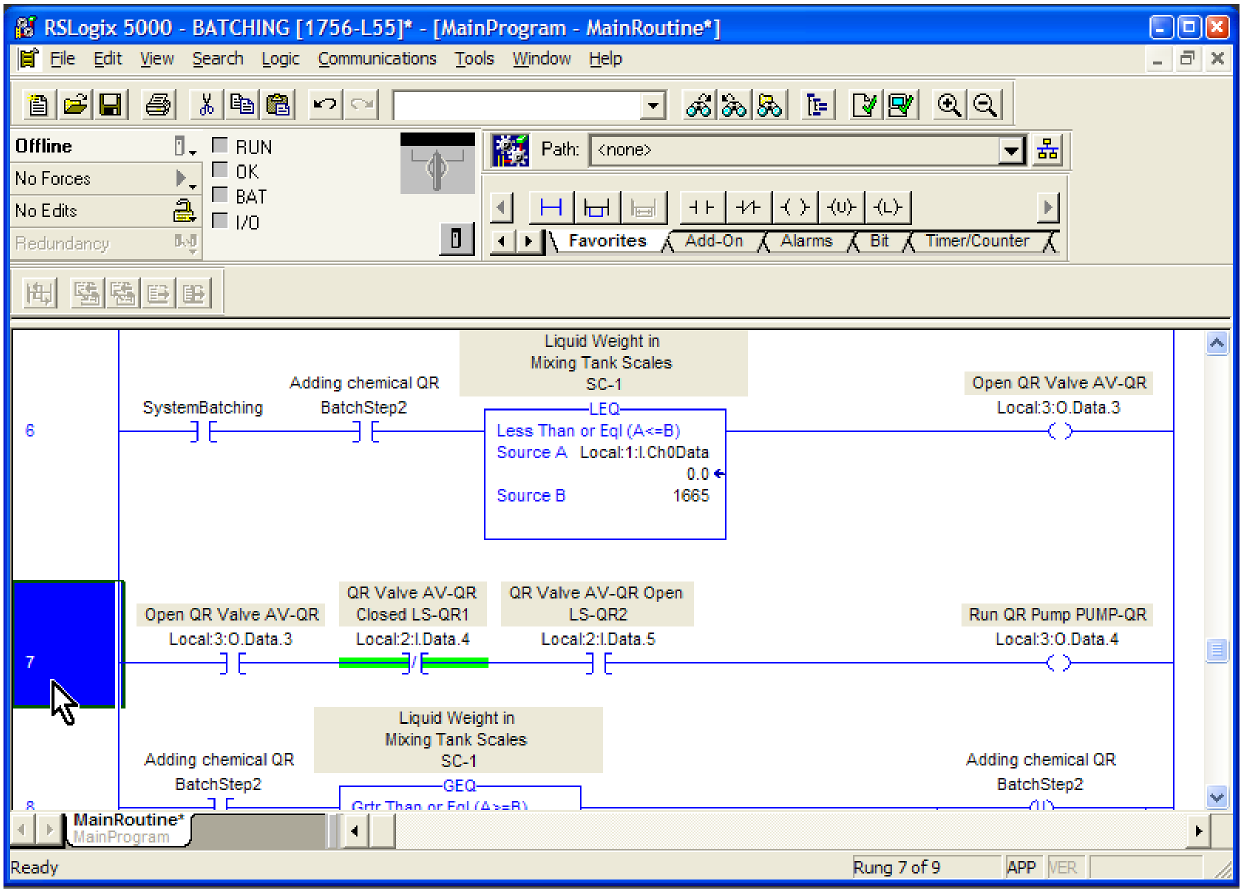
Double-click on Rung 7. The ASCII string input box appears. Type in the following string:

XIC Local:3:O.Data.3 XIO Local:2:I.Data.4 XIC Local:2:I.Data.5 OTE Local:3:O.Data.4



Rung: 7   In ASCII Text   XIC Local:3:O.Data.3 XIO Local:2:I.Data.4 XIC Local:2:I.Data.5 OTE Local:3:O.Data.4

Press enter and the instructions appear.



OK, maybe that wasn't the best application for the ASCII editor, but at least now you know it can be done.

Step 3 – Adding Chemical KM

Step 3 will have the same logic as Step 2, so will cut and paste to create new rungs.

Click on Rung 6, hold down the SHIFT key and click on Rung 8 to select the new rungs. Press CTRL-C. Click on the last rung and press CTRL-V.

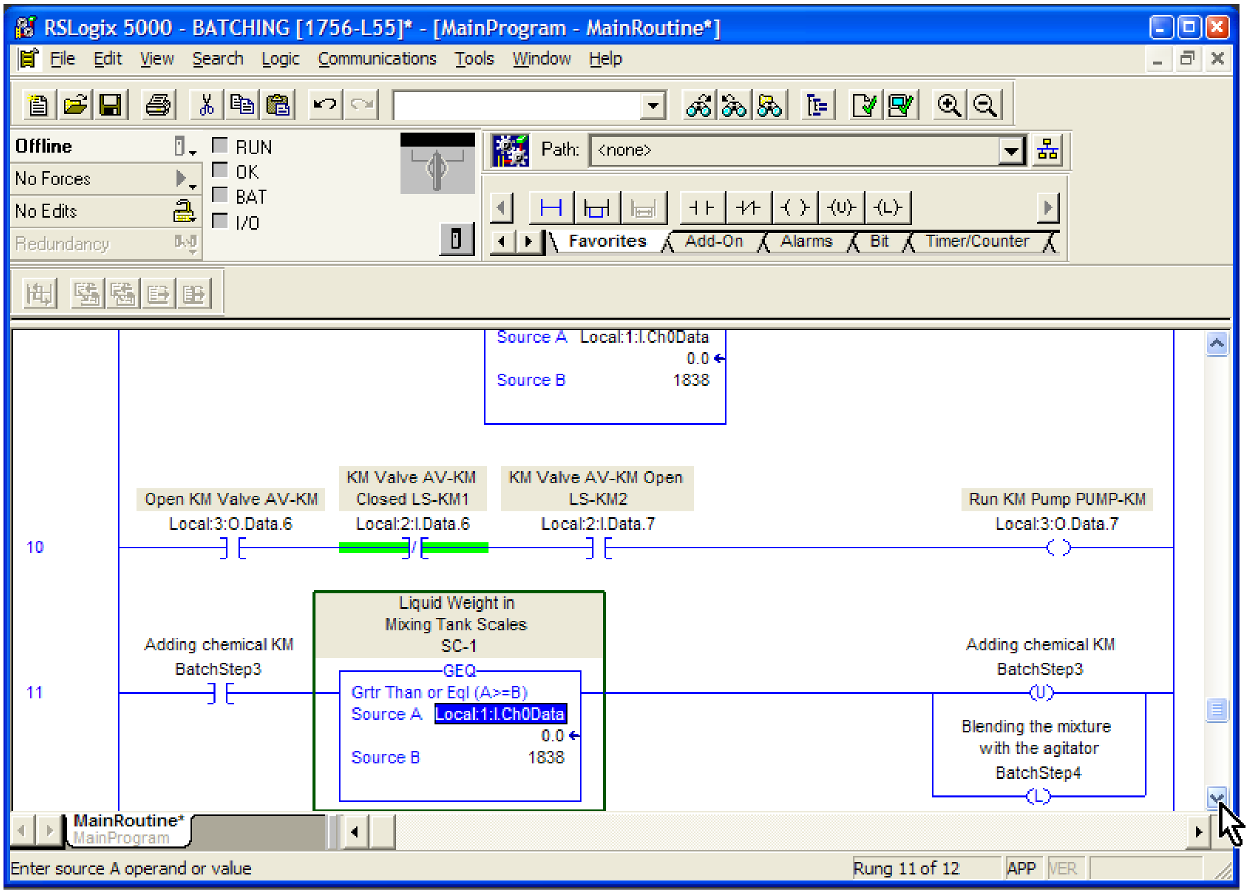
Change all the tag names for XIO and XIC instructions to match the new step.

We will be looking for a setpoint of 1665 + 173, since there is already 1665 lbs. of water in the Mixing Tank and we need to add 173 lbs. of KM. Change Source B in the LEQ instruction to 1838.

Change Source B in the GEQ instruction to 1838.

Change the latch and unlatch outputs to increment the system to Step 4.

It should look like this after you have made the changes and verified the rungs.



Enter source A operand or value

Step 4 – Blending

After all the ingredients are in the Mixing Tank, we have to run the Agitator for 3 minutes.

We will set up a timer to run for 3 minutes. When the timer is done, we will increment the system to Step 5. That will turn off the Agitator.

Start by inserting a new rung at the bottom.

Insert an XIC instruction with the address of B3:0/2.

Insert an XIC instruction with the address B3:0/13.

Click the “Timer/Counter” tab on the Language Element menu. Click and drag the TON tool button down to the right side of the new rung.

This type of timer is called a Timer On Delay. As soon as the instructions preceding it are true, it will begin timing. The Enable bit (EN) will turn on. After it reaches its Preset value, the Done bit (DN) will turn on.

First, we have to assign a tag name to the timer. Double-click on the “Timer” field in the instruction. Type “AgitatorRunTime” for the tag name.

Define the new timer tag name, as you cannot assign presets until you do this.

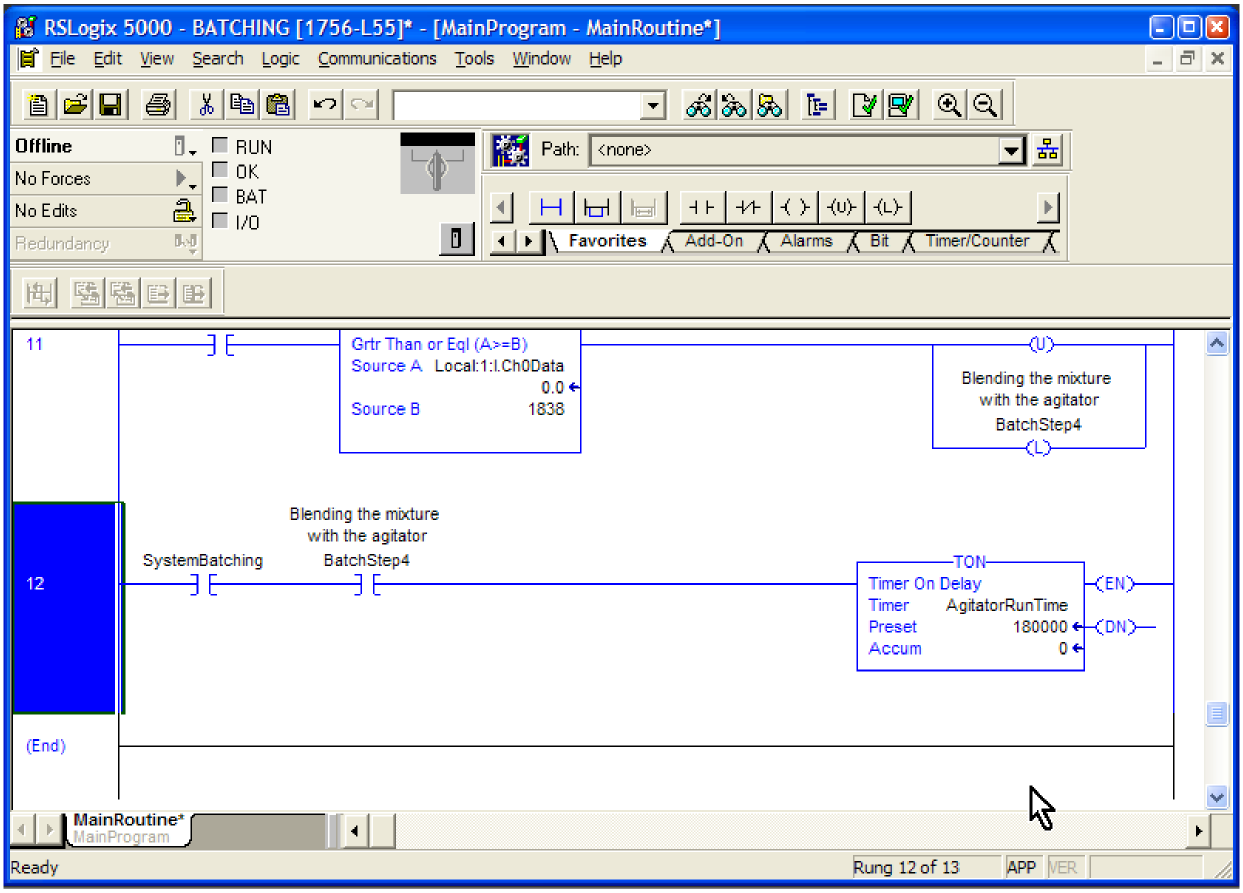
The time base for the TON instruction is always 1 msec.

We want to time for 3 minutes, or 180 seconds, so we will enter a preset of 180000.

Double-click on the Preset field and type 180000. Press enter.

Leave the accumulated value at 0, as this will be changed as the timer begins timing.

The completed rung looks like this.



11
Grtr Than or Eql (A>=B)
Source A Local:1:I.Ch0Data
0.0
Source B 1838
Blending the mixture with the agitator BatchStep4 (U) (L)

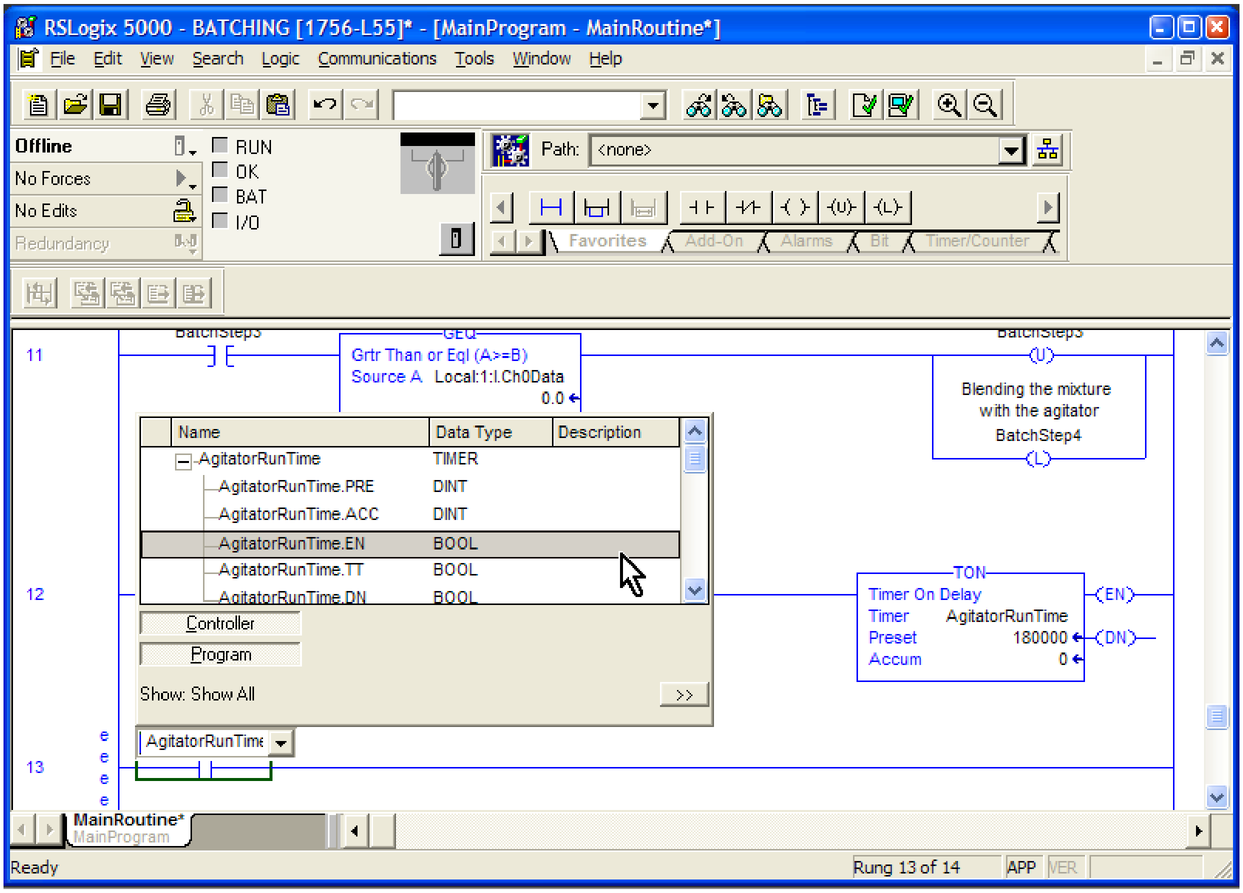
12
SystemBatching
Blending the mixture with the agitator BatchStep4
TON
Timer On Delay
Timer AgitatorRunTime
Preset 180000
Accum 0
(EN)
(DN)

(End)

To run the Agitator, insert a new rung at the bottom.

Click on the “User” tab of the tool button menu and drag an XIC instruction to the first marker of the new rung. We will use the Enable bit (EN) of the timer for the tag.

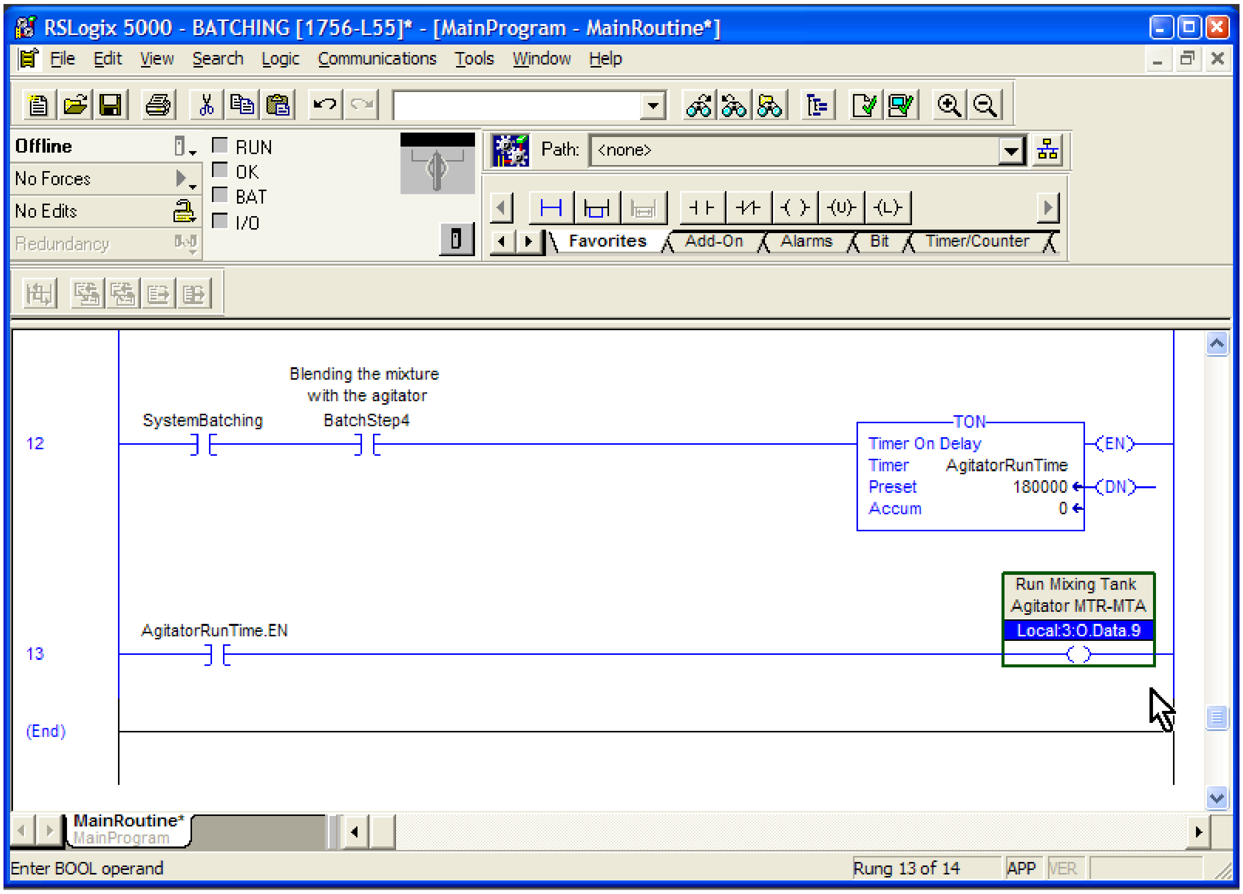
To assign a timer's enable bit, double-click on the tag name field for the new XIC instruction and navigate to the EN (enable) bit of the timer.



Name	Data Type	Description
AgitatorRunTime	TIMER	
AgitatorRunTime.PRE	DINT	
AgitatorRunTime.ACC	DINT	
AgitatorRunTime.EN	BOOL	
AgitatorRunTime.TT	BOOL	
AgitatorRunTime.DN	BOOL	

Show: Show All

Add an OTE instruction for the Mixing Tank Agitator.



The enable bit of a timer is on only when the timer is enabled. In Rung 12, for example, if the SystemBatching bit turns off, or the Batch Step 4 bit turns off, the timer will no longer be enabled. Consequently, its enable bit, AgitatorRunTime.EN will turn off.

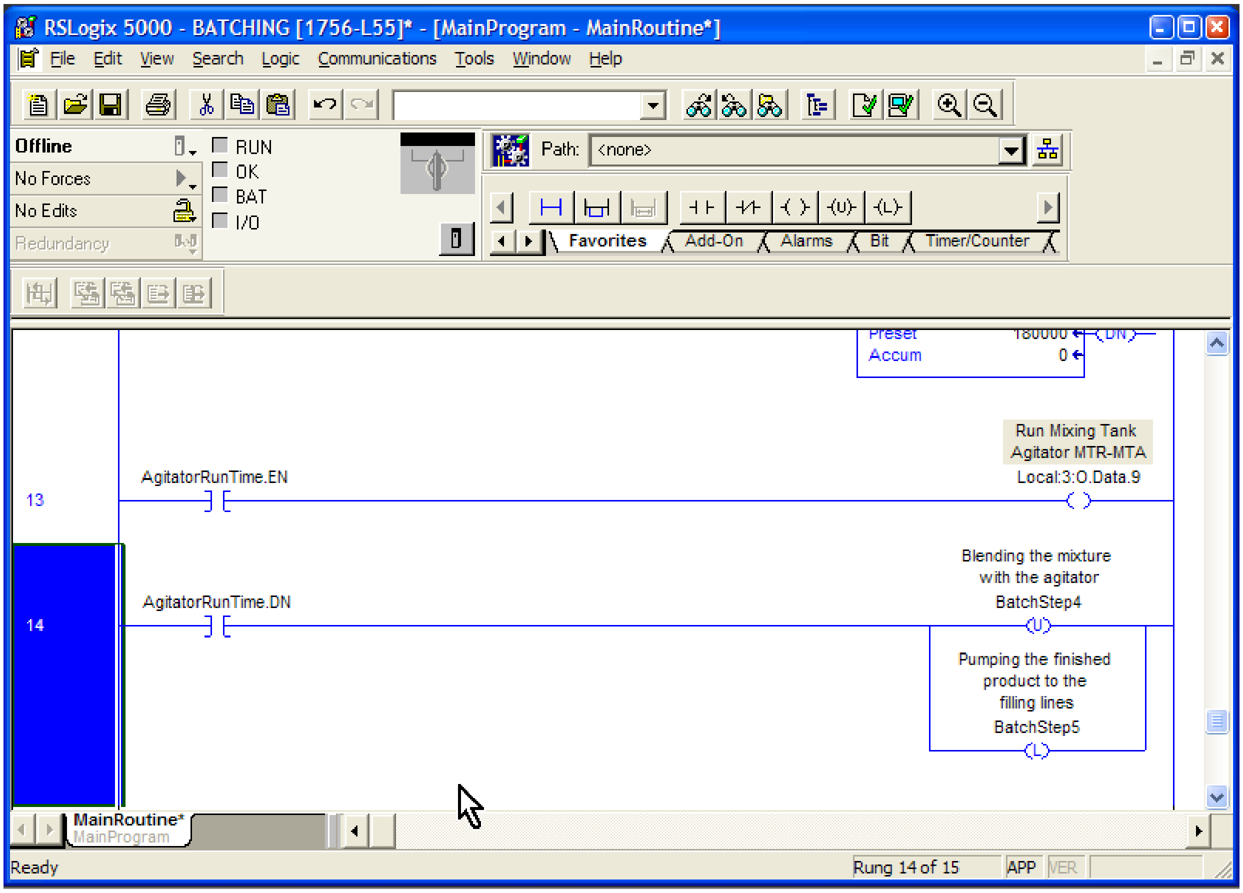
We will turn off the timer by incrementing the system to the next step when the timer is done.

Insert a new rung at the bottom and drag an XIC to the first marker. We will assign the Done bit (DN) of the timer to this instruction. It is formatted just like the Enable bit, except you use .DN as the suffix. Tag this instruction as AgitatorRunTime.DN.

We will use the same method to increment the system to Step 5 that we have used previously. Drag an OTU instruction to the last marker in the rung. Tag the instruction to be Step 4, which is BatchStep4.

Insert a new branch around the OTU by dragging a Branch tool button to the marker in front of the output. Grab the right leg of the branch and move it to the marker after the output.

Insert an OTL instruction on the bottom of the branch and tag it as BatchStep5.


RSLogix 5000 - BATCHING [1756-L55]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
Favorites
Add-On
Alarms
Bit
Timer/Counter
Preset 180000
Accum 0
DN
Run Mixing Tank Agitator MTR-MTA
Local:3:O.Data.9
13
AgitatorRunTime.EN
Blending the mixture with the agitator
BatchStep4
U
14
AgitatorRunTime.DN
Pumping the finished product to the filling lines
BatchStep5
L
MainRoutine*
MainProgram
Ready
Rung 14 of 15
APP
VER

Now, when the timer is done and AgitatorRunTime.DN turns on, it will unlatch Step 4 and start Step 5.

Step 5 – Pump to Filling Lines

Since this step involves a pump, it will be similar to Step 3. Let’s save ourselves some typing and copy all of Step 3.

Click on Rung 9. Hold down the SHIFT key and click on Rung 11.

Press CTRL-C.

Click on the last rung. Press CTRL-V.

In Rung 15, change BatchStep4 to BatchStep5.

Change the tag of the output instruction to Local:3:O.Data.13.

We want to pump until the Mixing tank is empty.

We will use the Ultrasonic Level Sensor to determine if the tank has liquid in it. We will use a GEQ instruction to accomplish this.

Double-click on the “LEQ” text in the LEQ instruction. Type GEQ (for Greater than or equal to) in that space. Change Source A of the instruction to Local:1:I.Ch1Data. This is the address of the scaled level sensor.

We may be tempted to put a value of 0 into Source B, but that could be risky. First, pumping the tank dry might be hard on the pump. Second, because of drift in the level sensor, we might never get a reading of zero in some instances.

The Process engineers have told us that emptying the tank to 3% is desirable. Put a 3 in for the value of Source B.

Change Rung 16 to show the correct valve, limit switches and pump addresses.

Since this is the last step in the batching process, we will use Rung 17 to increment out of Step 5 and complete the batching cycle.

Change the first XIC in Rung 17 to BatchStep5.

Copy the GEQ instruction from Rung 16. Click on the instruction, press CTRL-C and click on the GEQ instruction in Rung 18. Press CTRL-V.

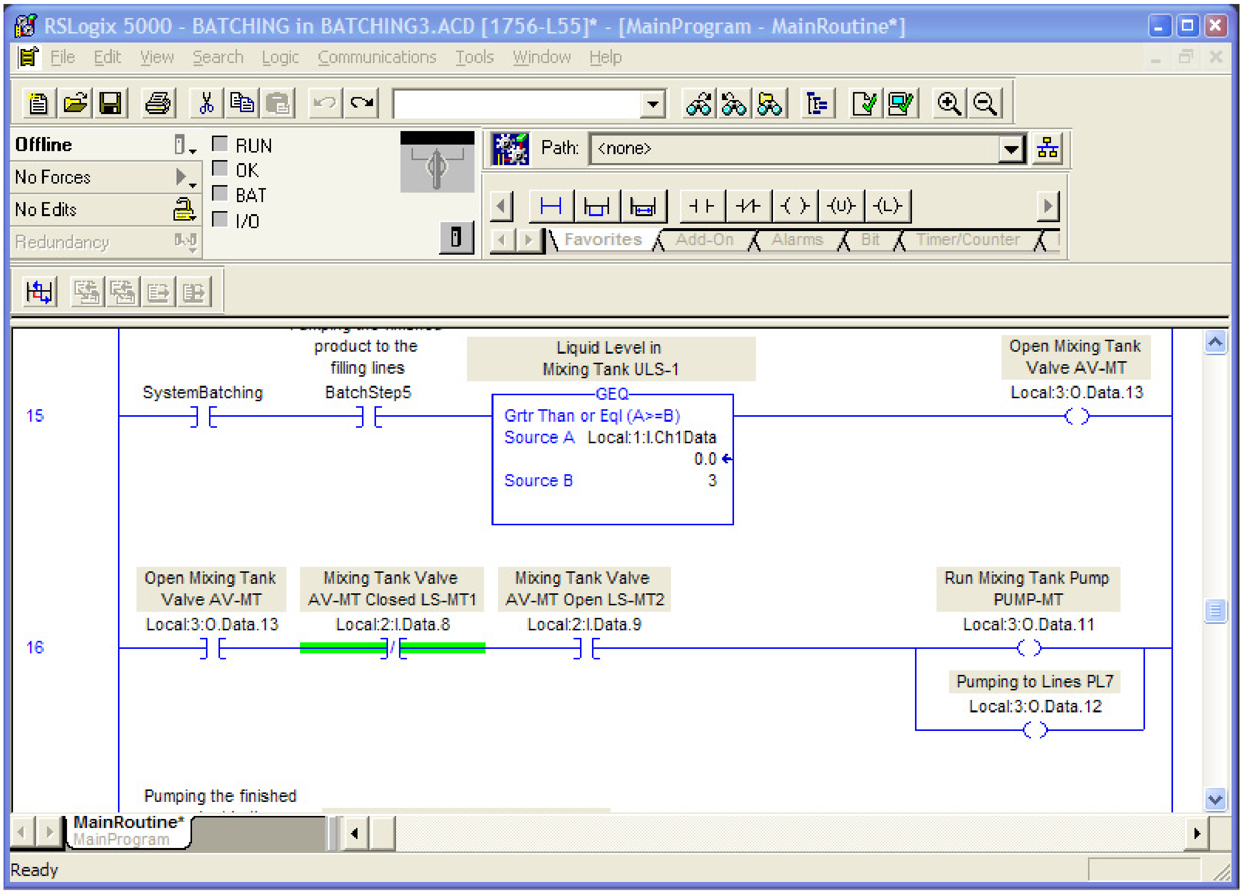
Delete the other GEQ from Rung 17 (the one that references the tank weight).

Change the remaining GEQ in Rung 17 to a LEQ.

Change the OTU instruction’s tag from BatchStep3 to BatchStep5.

Right-click on the OTL (L) instruction in the bottom branch. Choose “Change Instruction Type” and make it an OTE. Change the tag to BatchComplete. This is the “batch complete” bit.

The screen should look like this.



When the system enters Step 5, the Mixing Tank will be nearly full. The GEQ instruction will be true and open the Mixing Tank valve.

When the valve has been opened and verified by the limit switches, the pump will run.

When the level in the tank drops to 3%, the valve will close and the Mixing Tank pump will stop.

Simultaneously, the BatchStep5 bit will be unlatched and the BatchComplete OTE instruction will be turned on. It will trip the latch in Rung 1 and turn off the SystemBatching bit.

The system returns to its idle state and is ready to begin a new batch.

Faults

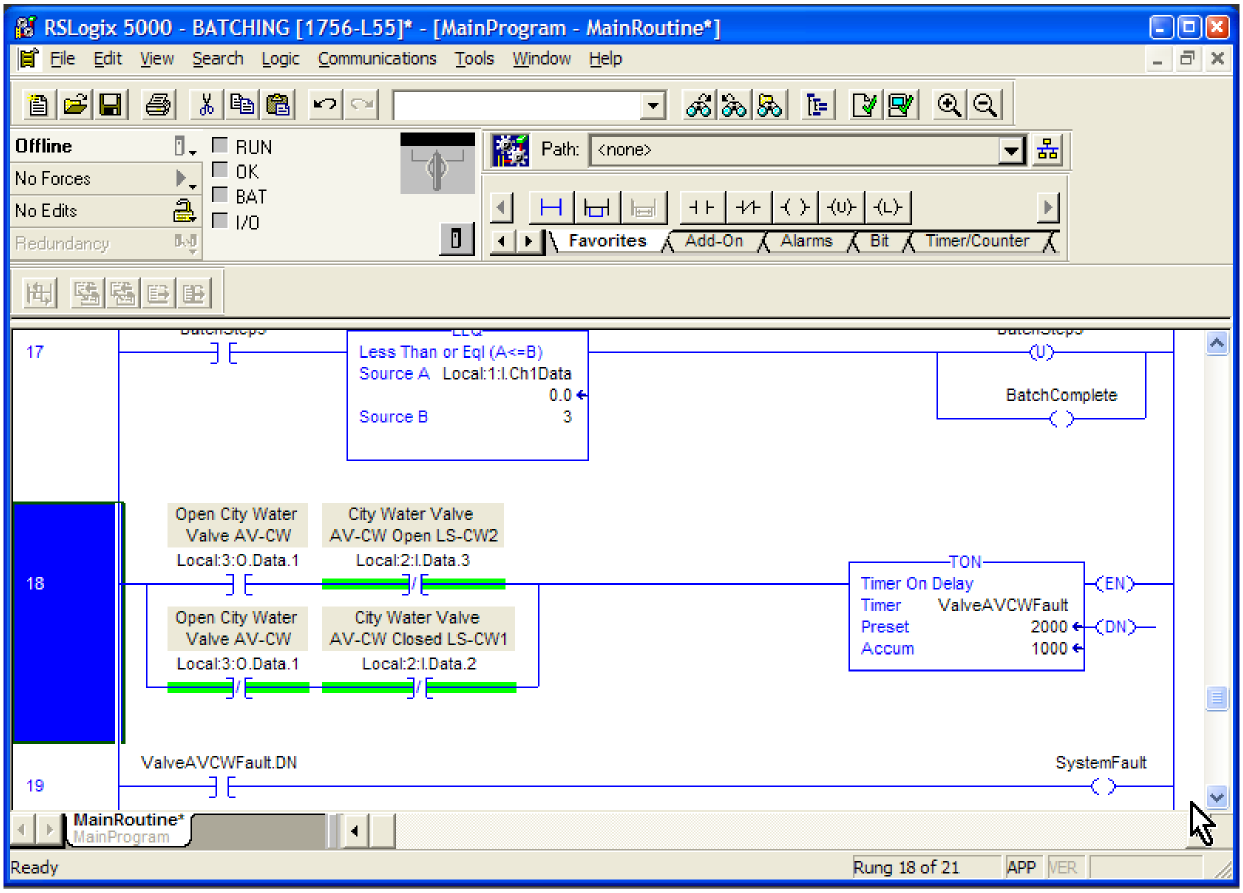
The program may be complete in an operational sense, but we need to add the logic to detect faults.

Valve Position Faults

We were told in the Project Scope that the valve position must be verified by the limit switches within 2 seconds.

We will use an individual timer for each valve to accomplish this.

Type in the logic as shown below.



We must accommodate the two conditions that indicate a fault. The first is that the valve was told to open, but did not open. The second fault condition is that the valve was told to close, but did not close.

Remember that the limit switches on the valves are electrically *normally open* and will be physically held closed by the valve when it is in position.

When the valve is open, we should get a signal to the input Local:2:I.Data.3 when limit switch LS-CW2 is activated by the valve.

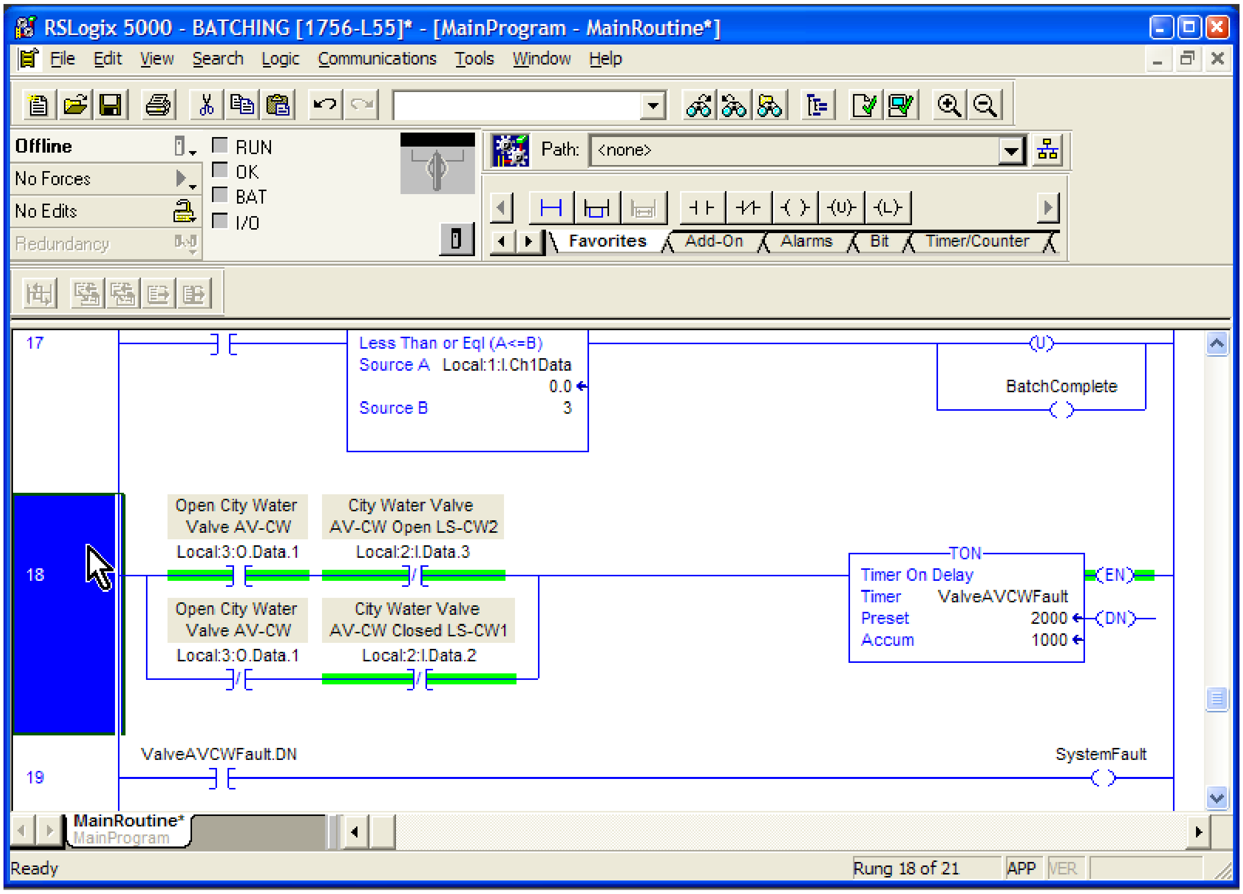
When the valve is closed, we should get a signal to the input Local:2:I.Data.2 when limit switch LS-CW1 is activated by the valve.

We use the XIO (examine if open) instruction. We want to know if we do *not* get a signal from the limit switches when we should. An XIO instruction used on an input will always be true if there is no signal on the input.

Let's look at how it handles the first fault condition (the valve was told to open, but did not open).

The City Water valve output is on but the limit switch LS-CW2 is electrically open. That means the XIO instruction for LS-CW2 in Rung 18 is true. This creates an undesirable condition and the timer will begin timing.

The screenshot below shows what the state of the instructions would be after 1 second of the valve being instructed to open, but LS-CW2 not yet being activated.



Timer ValveAVCWFault
Preset 2000
Accum 1000

You can see that the City Water valve is being told to open, as Local:3:O.Data.1 is highlighted. You can also see that LS-CW2 says that the City Water valve is *not* open, as it is also highlighted. The timer has begun timing and has reached, or accumulated, 1 second.

If the timer reaches 2 seconds, the timer done bit ValveAVCWFault.DN will come on and turn on the SystemFault bit in Rung 19.

Since we used the SystemFault bit in Rung 0, this will cause the SystemEnable bit to turn off.

Consequently, Rungs 1 and 2 will become false and the system will stop batching.

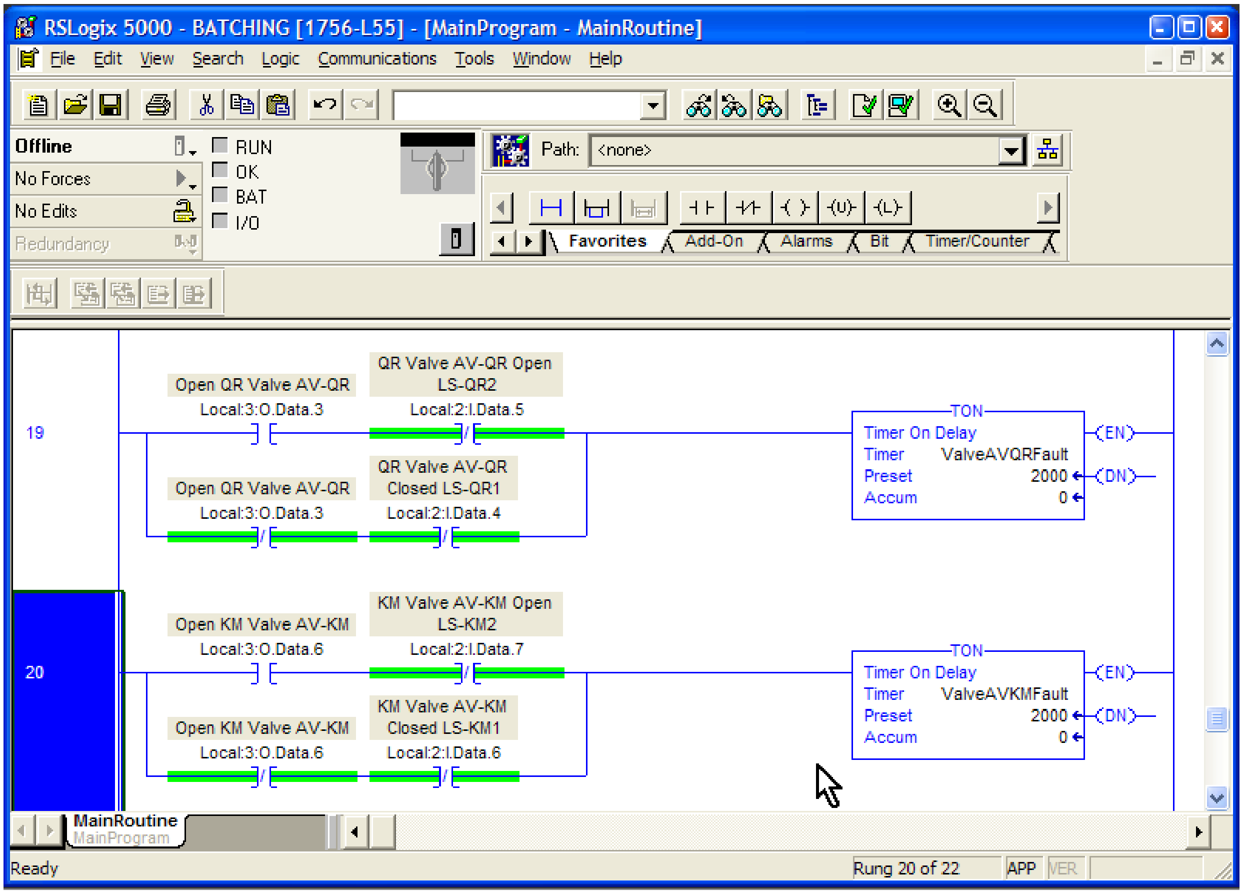
Let’s look at the other condition; that is, the valve was told to close, but did not close.

That is accomplished in the branch in Rung 18. If the valve output is not on but the limit switch is not activated, then the timer will begin and generate a fault.

Sometimes it gets confusing when you are working with the XIO instruction. The logic is inverted and you have to flip some things around in your head.

The logic we have used here to detect a valve fault is fairly standard and its use is widespread. The nice part about this logic is that it is pretty easy to troubleshoot. You can physically look at the valve to see if it is open or closed. Then, you can look at the input you are receiving from the limit switch. If you are getting a fault when you should not, then just invert the instruction for the limit switch. Change it from an XIO to an XIC, or vice versa.

Let’s create the fault timers for the remaining valves. This is the screenshot for the QR valve and KM valve fault logic.

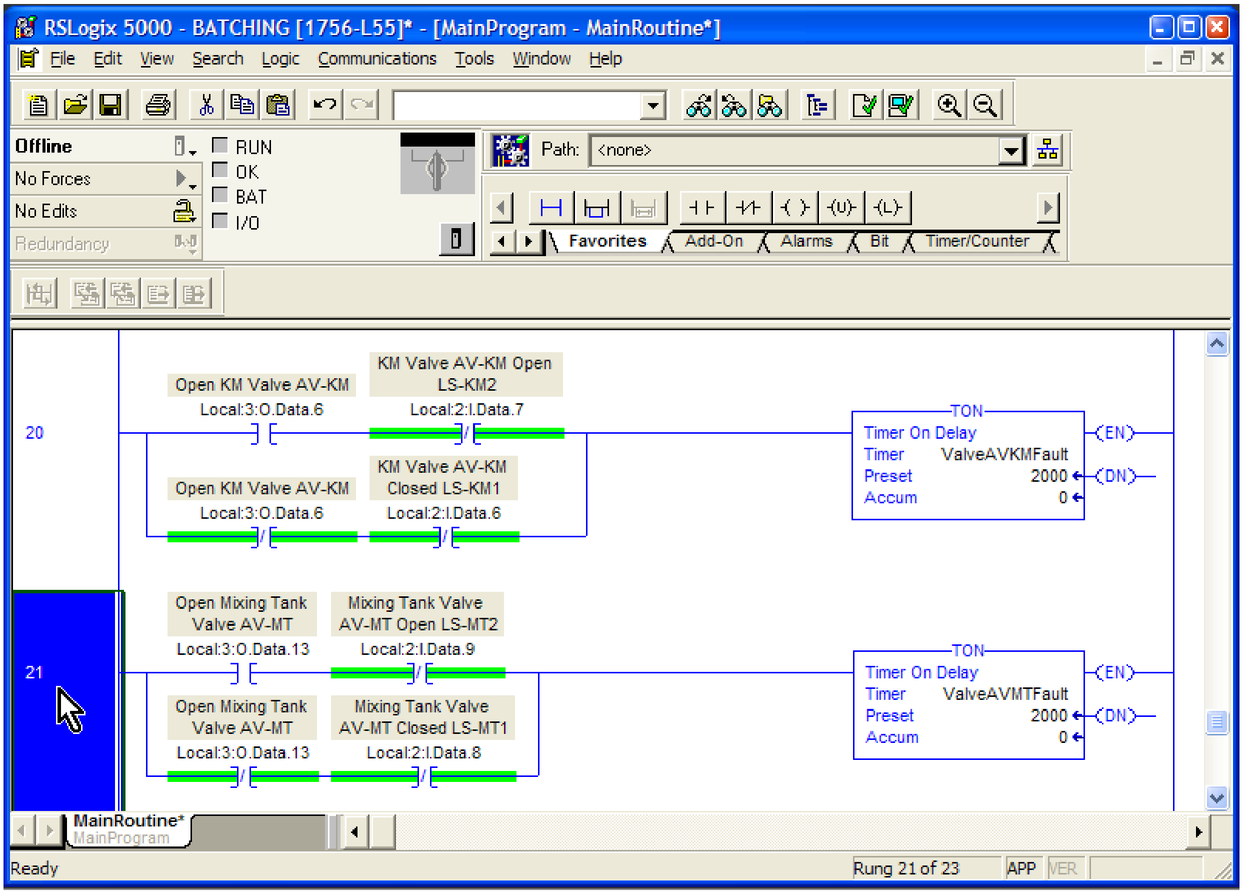

RSLogix 5000 - BATCHING [1756-L55] - [MainProgram - MainRoutine]

Rung 19:
Open QR Valve AV-QR Local:3:O.Data.3] [— QR Valve AV-QR Open LS-QR2 Local:2:I.Data.5]/[
Branch: Open QR Valve AV-QR Local:3:O.Data.3]/[— QR Valve AV-QR Closed LS-QR1 Local:2:I.Data.4]/[
TON Timer On Delay, Timer ValveAVQRFault, Preset 2000, Accum 0 (EN) (DN)

Rung 20:
Open KM Valve AV-KM Local:3:O.Data.6] [— KM Valve AV-KM Open LS-KM2 Local:2:I.Data.7]/[
Branch: Open KM Valve AV-KM Local:3:O.Data.6]/[— KM Valve AV-KM Closed LS-KM1 Local:2:I.Data.6]/[
TON Timer On Delay, Timer ValveAVKMFault, Preset 2000, Accum 0 (EN) (DN)

Ready — Rung 20 of 22 — APP — VER

This is the screenshot for the Mixing Tank valve fault logic.



20

Open KM Valve AV-KM
Local:3:O.Data.6

KM Valve AV-KM Open LS-KM2
Local:2:I.Data.7

Open KM Valve AV-KM
Local:3:O.Data.6

KM Valve AV-KM Closed LS-KM1
Local:2:I.Data.6

TON
Timer On Delay
Timer ValveAVKMFault
Preset 2000
Accum 0
(EN)
(DN)

21

Open Mixing Tank Valve AV-MT
Local:3:O.Data.13

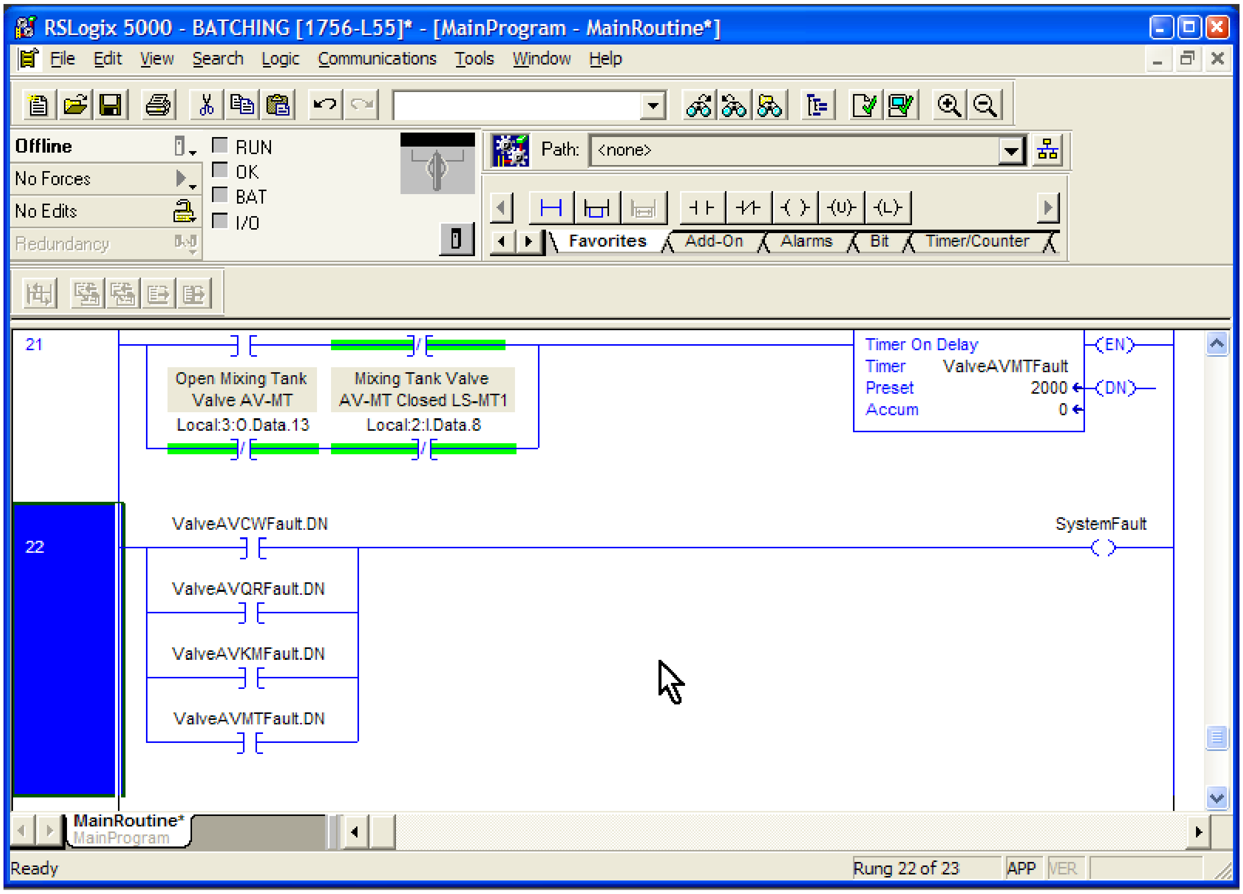
Mixing Tank Valve AV-MT Open LS-MT2
Local:2:I.Data.9

Open Mixing Tank Valve AV-MT
Local:3:O.Data.13

Mixing Tank Valve AV-MT Closed LS-MT1
Local:2:I.Data.8

TON
Timer On Delay
Timer ValveAVMTFault
Preset 2000
Accum 0
(EN)
(DN)

Notice how the System Fault rung has grown.


RSLogix 5000 - BATCHING [1756-L55]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
□ RUN
□ OK
□ BAT
□ I/O
Path: <none>
Favorites
Add-On
Alarms
Bit
Timer/Counter
MainRoutine*
MainProgram

Rung 21:

Tag	Description	Address
XIC	Open Mixing Tank Valve AV-MT	Local:3:O.Data.13
XIO	Mixing Tank Valve AV-MT Closed LS-MT1	Local:2:I.Data.8

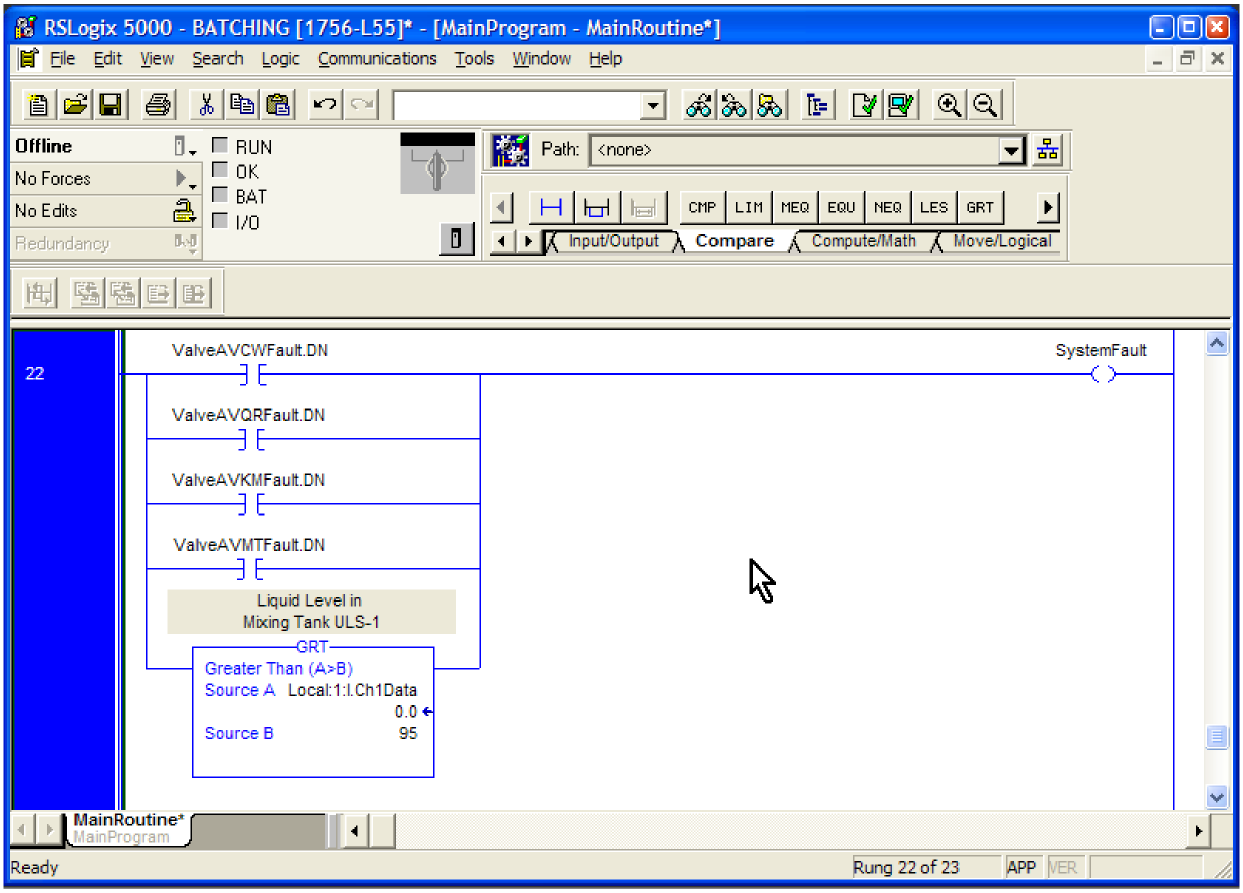
Timer On Delay
Timer ValveAVMTFault
Preset 2000
Accum 0
(EN)
(DN)

Rung 22:

ValveAVCWFault.DN
ValveAVQRFault.DN
ValveAVKMFault.DN
ValveAVMTFault.DN
SystemFault

Ready
Rung 22 of 23
APP
VER

To complete our fault logic, we need to add the Liquid High Level alarm. We can achieve this with one GRT instruction in the System Fault rung. It is agreed that if the liquid level reaches 95% of the tank's capacity, it will fault.

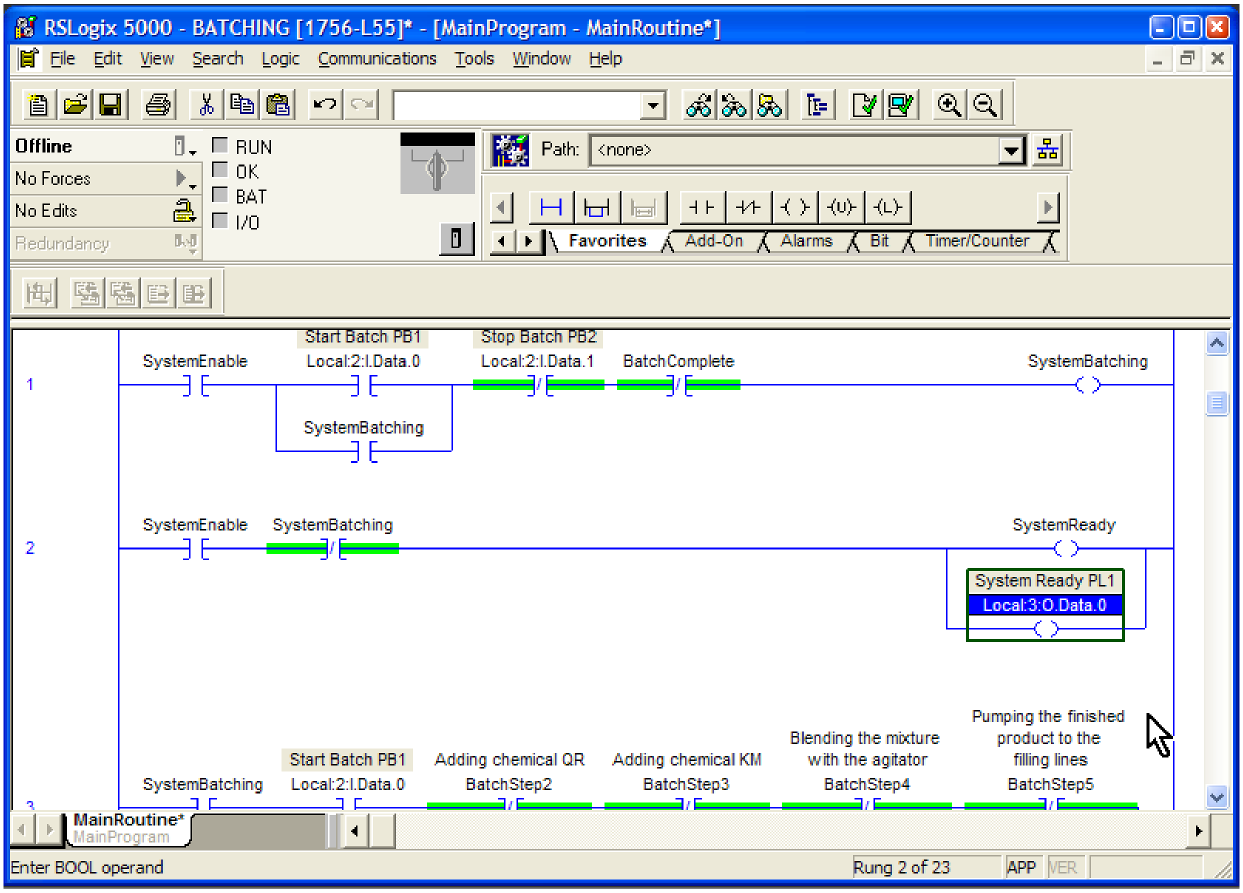

RSLogix 5000 - BATCHING [1756-L55]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
CMP LIM MEQ EQU NEQ LES GRT
Input/Output Compare Compute/Math Move/Logical
22
ValveAVCWFault.DN
SystemFault
ValveAVQRFault.DN
ValveAVKMFault.DN
ValveAVMTFault.DN
Liquid Level in Mixing Tank ULS-1
GRT
Greater Than (A>B)
Source A Local:1:I.Ch1Data
0.0
Source B 95
MainRoutine*
MainProgram
Ready
Rung 22 of 23
APP
VER

Console Status Indicators – Pilot Lights

I have held off putting the logic in for the Pilot Lights in an effort to keep the text as simple as possible. Now that you understand how the system operates, we can go back through the program and add the pilot lights.

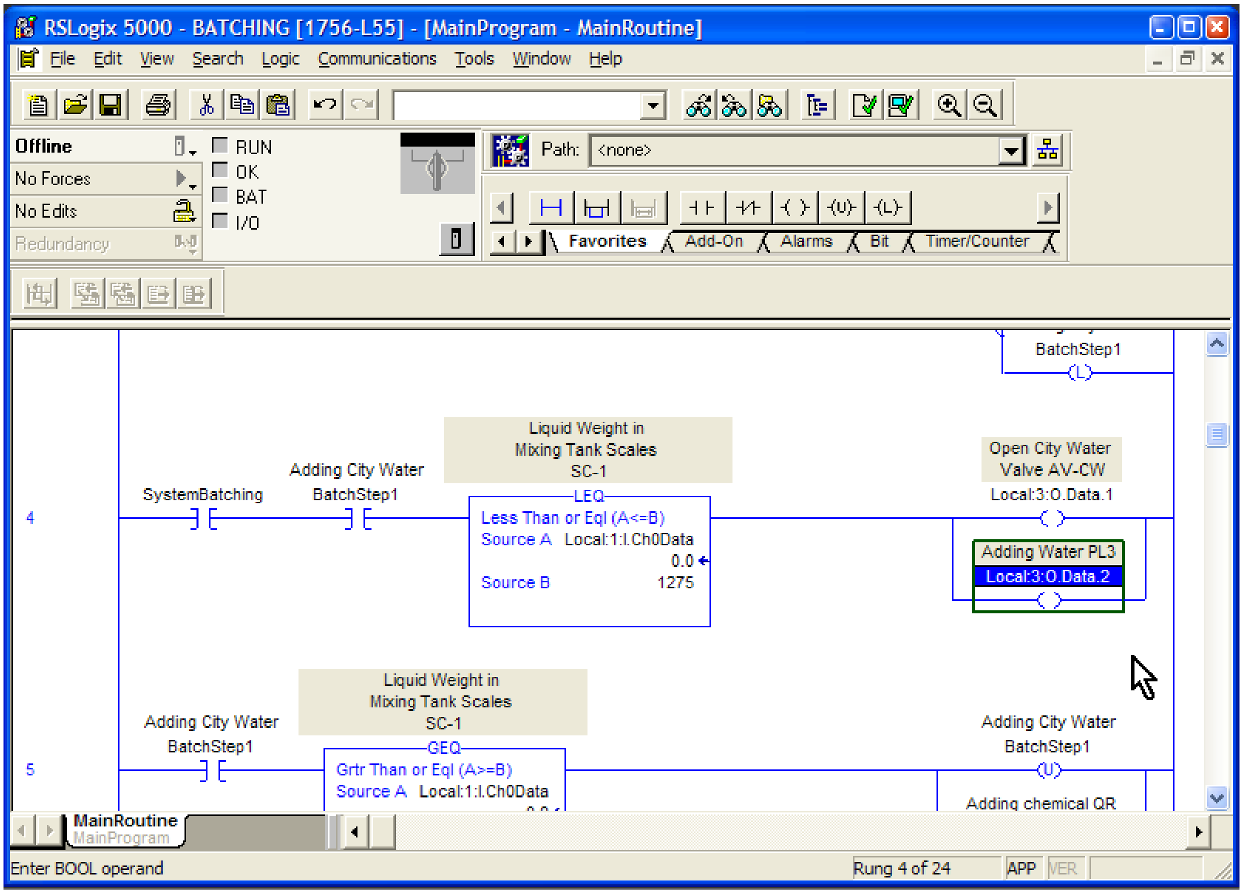
The first is the System Ready PL1 pilot light. Find Rung 2 in the logic.

Insert a branch around the SystemReady output. Put an OTE instruction on the bottom rung of the new branch and tag it Local:3:O.Data.0. With this logic, the light will come on when the SystemReady bit is on.

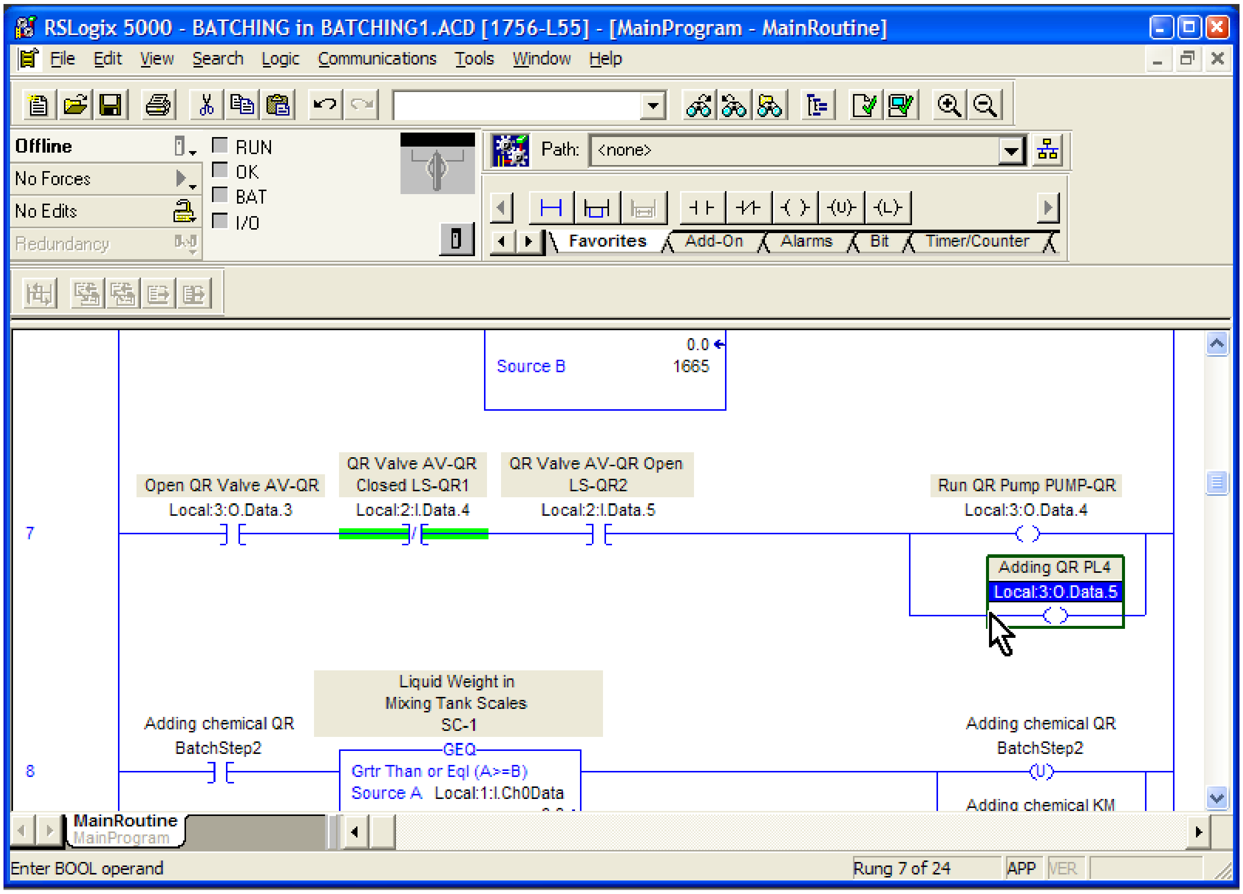


We will add the remaining pilot lights in a similar fashion.

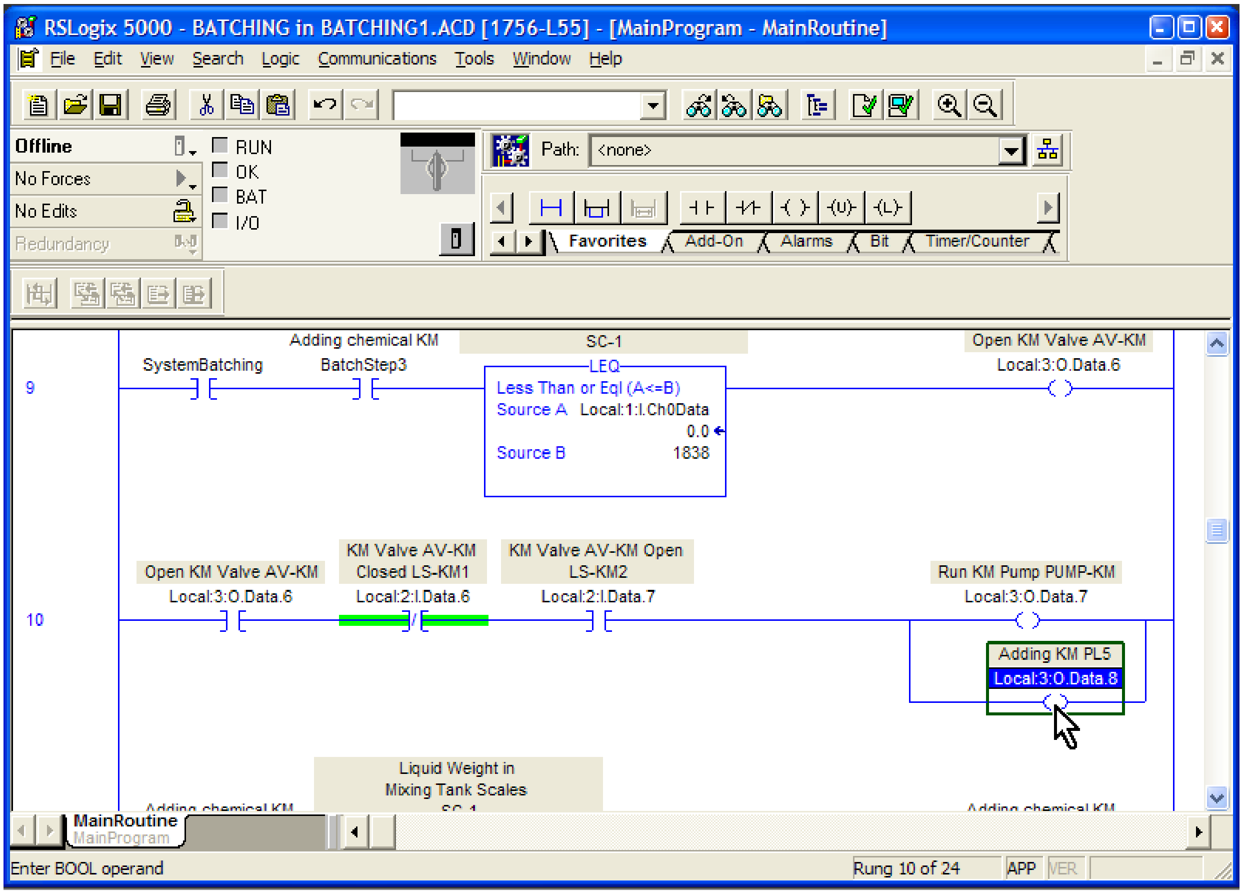
Adding water


RSLogix 5000 - BATCHING [1756-L55] - [MainProgram - MainRoutine]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
Favorites
Add-On
Alarms
Bit
Timer/Counter
BatchStep1
Liquid Weight in Mixing Tank Scales SC-1
Open City Water Valve AV-CW
Local:3:O.Data.1
4
SystemBatching
Adding City Water BatchStep1
LEQ
Less Than or Eql (A<=B)
Source A Local:1:I.Ch0Data
0.0
Source B 1275
Adding Water PL3
Local:3:O.Data.2
5
Adding City Water BatchStep1
Liquid Weight in Mixing Tank Scales SC-1
GEQ
Grtr Than or Eql (A>=B)
Source A Local:1:I.Ch0Data
Adding City Water BatchStep1
Adding chemical QR
MainRoutine
MainProgram
Enter BOOL operand
Rung 4 of 24
APP
VER

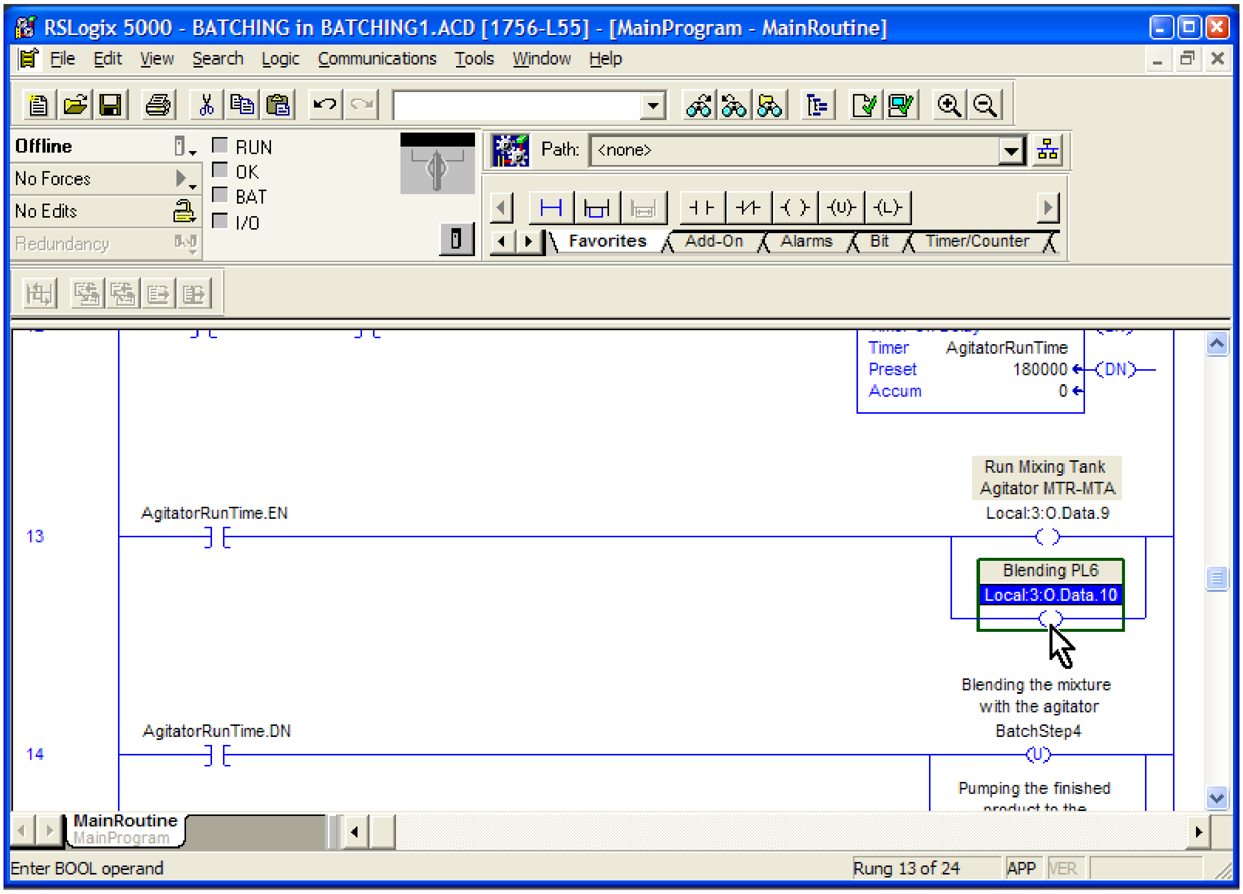
Adding QR


RSLogix 5000 - BATCHING in BATCHING1.ACD [1756-L55] - [MainProgram - MainRoutine]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
Favorites
Add-On
Alarms
Bit
Timer/Counter
Source B
0.0
1665
7
Open QR Valve AV-QR
Local:3:O.Data.3
QR Valve AV-QR Closed LS-QR1
Local:2:I.Data.4
QR Valve AV-QR Open LS-QR2
Local:2:I.Data.5
Run QR Pump PUMP-QR
Local:3:O.Data.4
Adding QR PL4
Local:3:O.Data.5
8
Adding chemical QR
BatchStep2
Liquid Weight in Mixing Tank Scales SC-1
GEQ
Grtr Than or Eql (A>=B)
Source A Local:1:I.Ch0Data
Adding chemical QR
BatchStep2
Adding chemical KM
MainRoutine
MainProgram
Enter BOOL operand
Rung 7 of 24
APP
VER

Adding KM


RSLogix 5000 - BATCHING in BATCHING1.ACD [1756-L55] - [MainProgram - MainRoutine]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
Favorites
Add-On
Alarms
Bit
Timer/Counter
9
SystemBatching
Adding chemical KM
BatchStep3
SC-1
LEQ
Less Than or Eql (A<=B)
Source A Local:1:I.Ch0Data
0.0
Source B 1838
Open KM Valve AV-KM
Local:3:O.Data.6
10
Open KM Valve AV-KM
Local:3:O.Data.6
KM Valve AV-KM
Closed LS-KM1
Local:2:I.Data.6
KM Valve AV-KM Open
LS-KM2
Local:2:I.Data.7
Run KM Pump PUMP-KM
Local:3:O.Data.7
Adding KM PL5
Local:3:O.Data.8
Liquid Weight in
Mixing Tank Scales
SC-1
Adding chemical KM
Adding chemical KM
MainRoutine
MainProgram
Enter BOOL operand
Rung 10 of 24
APP
VER

Blending



RSLogix 5000 - BATCHING in BATCHING1.ACD [1756-L55] - [MainProgram - MainRoutine]

Timer AgitatorRunTime
Preset 180000
Accum 0
(DN)

13 AgitatorRunTime.EN

Run Mixing Tank Agitator MTR-MTA
Local:3:O.Data.9

Blending PL6
Local:3:O.Data.10

14 AgitatorRunTime.DN

Blending the mixture with the agitator
BatchStep4
(U)

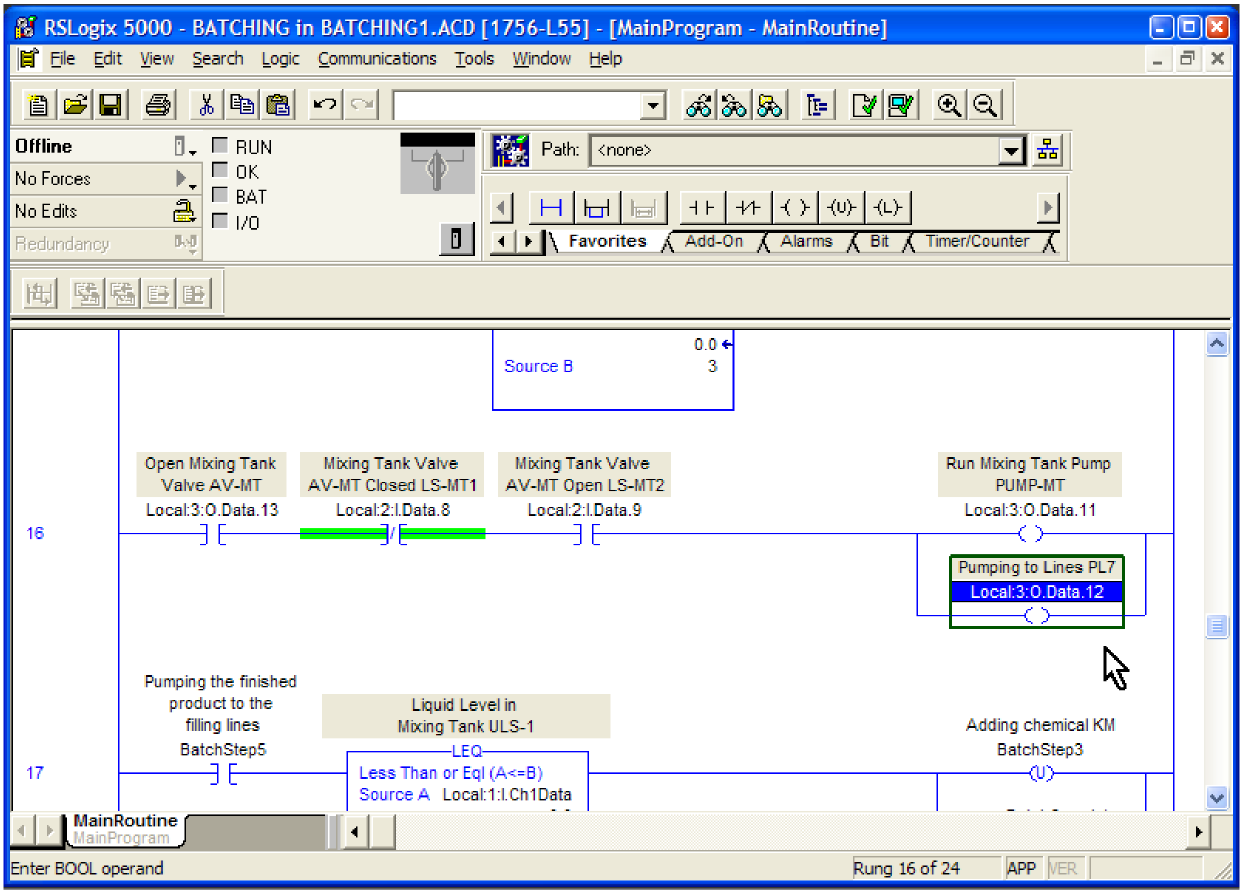
Pumping the finished product to the

Enter BOOL operand

Rung 13 of 24

APP VER

Pumping to Lines

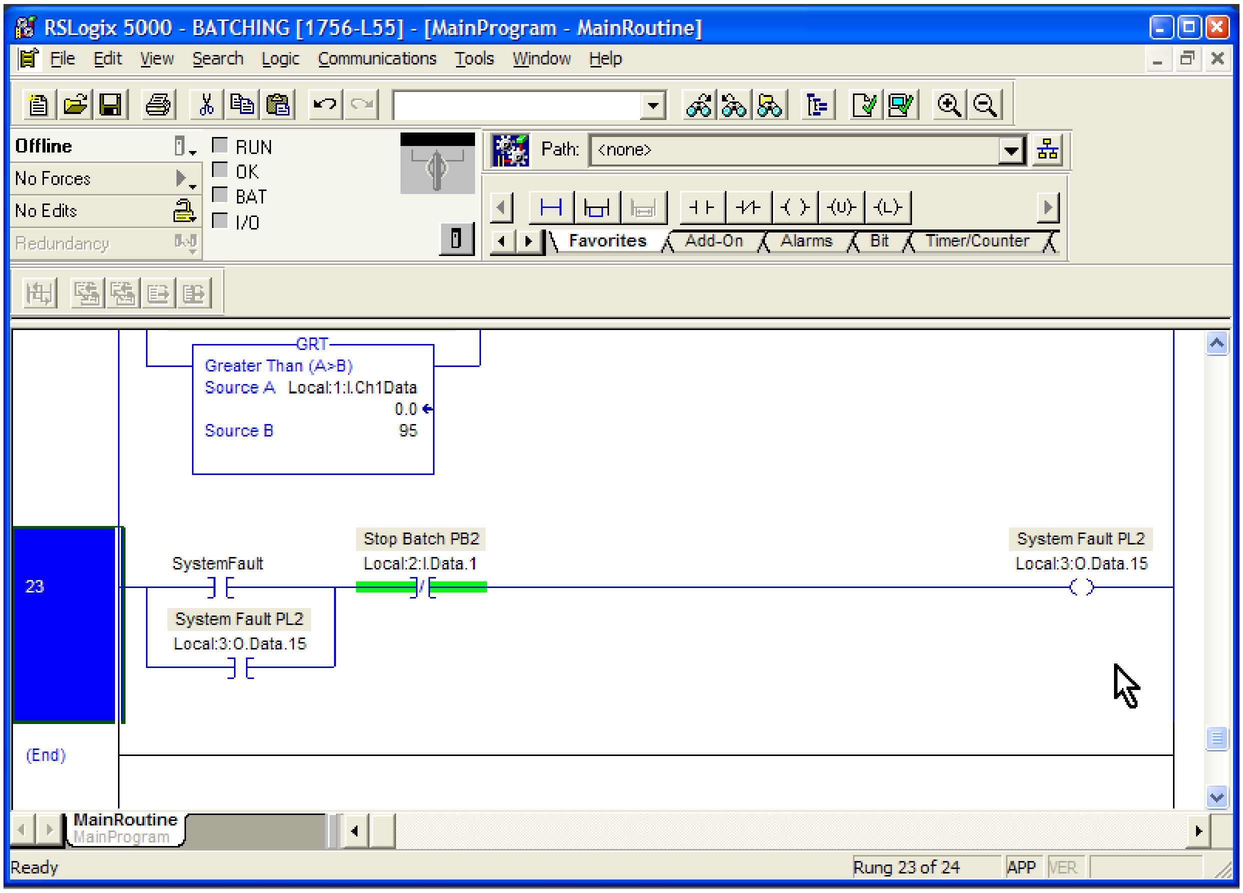


System Fault

We need to add a little logic to the System Fault pilot light. This light needs to stay on after it detects a fault; otherwise, the system will stop, the light may go out and the operator won't know there was a fault.

We are using the STOP BATCH pushbutton to reset the fault light.

Add the rung as shown below.



Adding Rung Comments

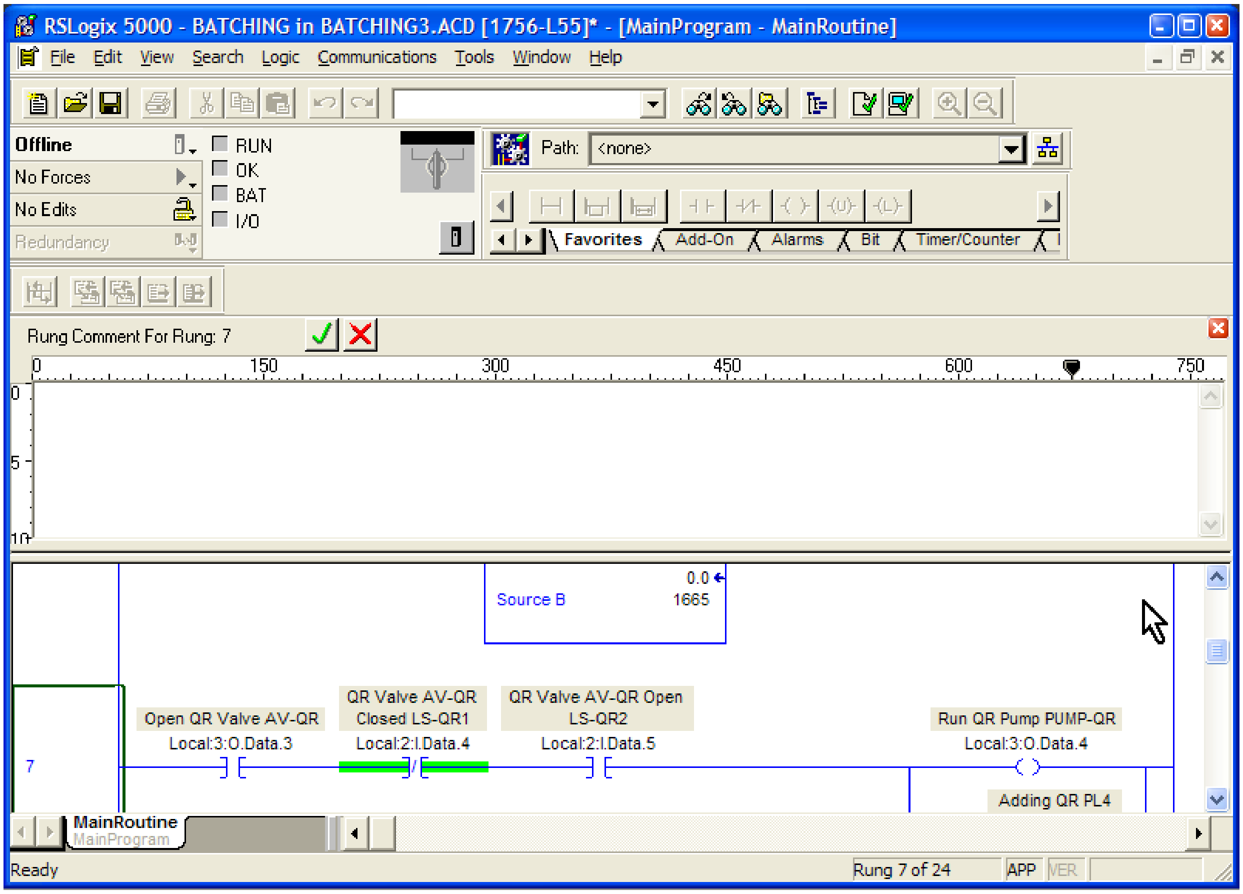
A good rung comment explains in plain language what the rung is supposed to do. These are very valuable. First, it is a way to double-check your own work. I have been surprised at the number of times I have found a mistake in my logic as I was writing the rung comments!

Second, it helps a person who is unfamiliar with the program quickly learn how the program is supposed to work.

Finally, it adds value to your finished program. A well-documented program is worth more money to a client (or your company) than a program that is not documented.

They can even get you out of a jam. If you have to troubleshoot a system 6 months after you first programmed it, I can guarantee that you will not remember every reason for every line of code you wrote. If you have written code in some other language before, you certainly can understand the value of documenting your work.

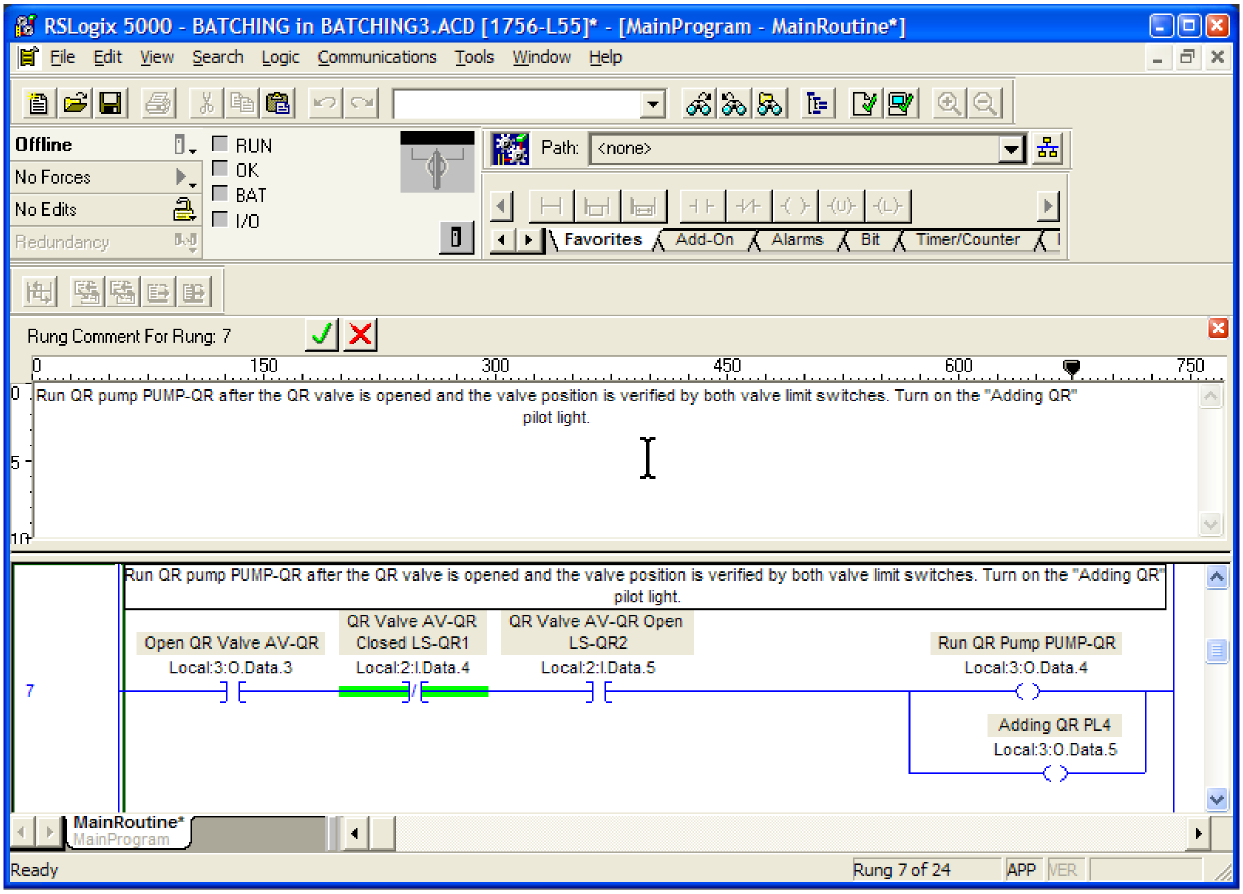
Let’s use a rung from our program as an example. Right-click on the rung number 7 and choose “Edit Rung Comment”. The rung comment box appears above the rung.



Type this in the Rung Comment field:

Run QR pump PUMP-QR after the QR valve is opened and the valve position is verified by both valve limit switches. Turn on the "Adding QR" pilot light.

Notice that you don't have to put details such as PLC tags in the rung comment – just document the concept of the rung.



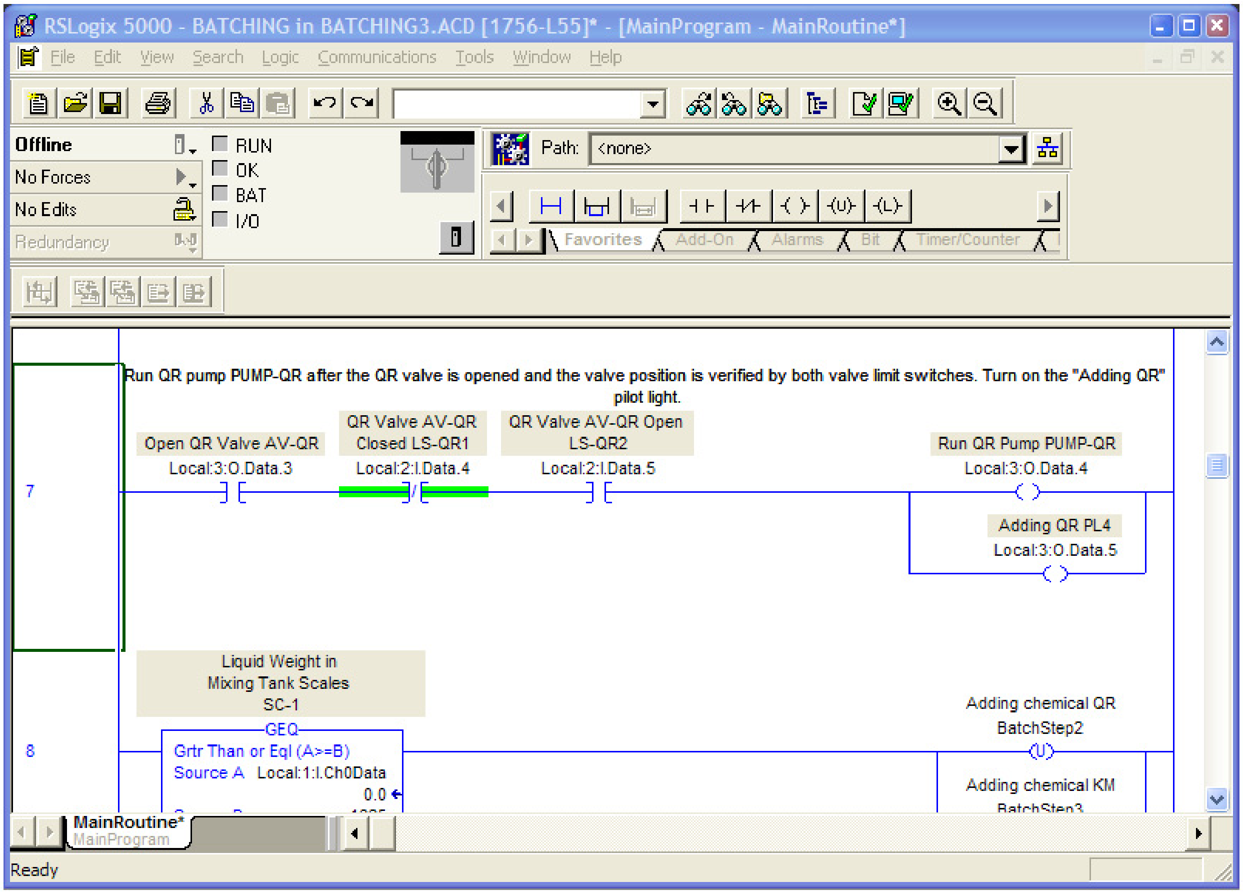
RSLogix 5000 - BATCHING in BATCHING3.ACD [1756-L55]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
Offline
No Forces
No Edits
Redundancy
RUN
OK
BAT
I/O
Path: <none>
Favorites Add-On Alarms Bit Timer/Counter
Rung Comment For Rung: 7
MainRoutine*
MainProgram
Ready
Rung 7 of 24
APP
VER

Run QR pump PUMP-QR after the QR valve is opened and the valve position is verified by both valve limit switches. Turn on the "Adding QR" pilot light.

Rung 7:

Instruction	Description	Tag
XIC	Open QR Valve AV-QR	Local:3:O.Data.3
XIO	QR Valve AV-QR Closed LS-QR1	Local:2:I.Data.4
XIC	QR Valve AV-QR Open LS-QR2	Local:2:I.Data.5
OTE	Run QR Pump PUMP-QR	Local:3:O.Data.4
OTE	Adding QR PL4	Local:3:O.Data.5

Click anywhere outside of the comment box and the comment appears above the rung.



RSLogix 5000 - BATCHING in BATCHING3.ACD [1756-L55]* - [MainProgram - MainRoutine*]

Run QR pump PUMP-QR after the QR valve is opened and the valve position is verified by both valve limit switches. Turn on the "Adding QR" pilot light.

Connecting To The PLC And Going Online

You have written your program and now you are ready to download, or send, the Batching program to the computer.

First, backup up your original file and put it in a safe place.

Connecting to a ControlLogix processor is done through Ethernet, the same way you connect a PC across a network.

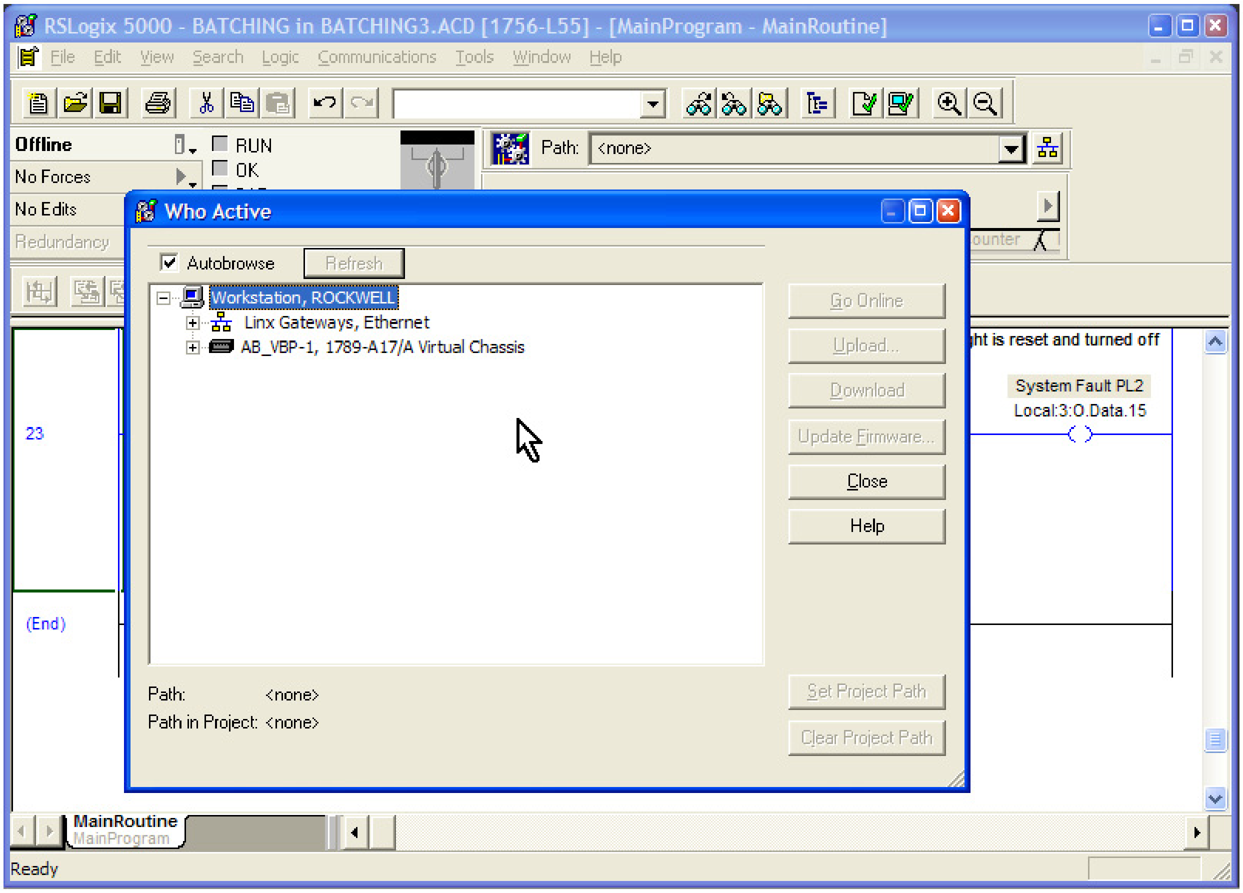
WARNING

There are dangerous voltages present on terminals of the PLC. Follow all warnings and instructions from the Allen-Bradley manual for connecting power to the PLC. If you are not familiar with hazards and the potential dangers of these voltages **STOP RIGHT NOW**. Consult a trained professional who is able to assist you.

Typically, most ControlLogix PLCs are connected through an Ethernet switch. You simply plug your computer into a open port on the switch.

Click on Communications > Who Active

A window similar to this will appear:



Path: <none>
Path in Project: <none>

Navigate to the PLC to which you would like to connect. When you are successful, the “Upload” and the “Download” buttons on the right will become available; they will no longer be grayed out.

You will then be able to download your program by following the prompts.

Let me say that I have greatly oversimplified this process. To be frank, this is where many people have problems. Sometimes connecting to a PLC is as easy as finding an Internet site in your browser. Other times, it can be an incredibly frustrating experience, due to no fault of your own.

Do what you can to prepare yourself by asking others about connecting to this particular PLC, making sure your computer can connect to a similar PLC, and so on.

Anybody who has ever connected to a PLC has had problems; they will understand your position.

There was one instance where the firmware in a particular PLC was not compatible with the version of RSLogix I was running in my laptop. It took a call to Rockwell's Tech Support to sort out the problem. They were very helpful, and it wasn't long before I was connected

TIP Tip: In Allen-Bradley PLC vernacular, **upload** means **get** the program from the PLC and load it in RSLogix on your computer.

Download means **send** the program from RSLogix on your computer to the PLC.

RSLogix Emulate 5000

Rockwell offers a nifty way to simulate going online. You can test your program at your desk, without being connected to a PLC. You can toggle virtual I/O points, change values in the virtual processor and see the results real time, just like you would if you were connected to the PLC.

The software that allows you to do this is called RSLogix Emulate 5000. If you don't have it installed on your computer, I highly recommend that you do so.

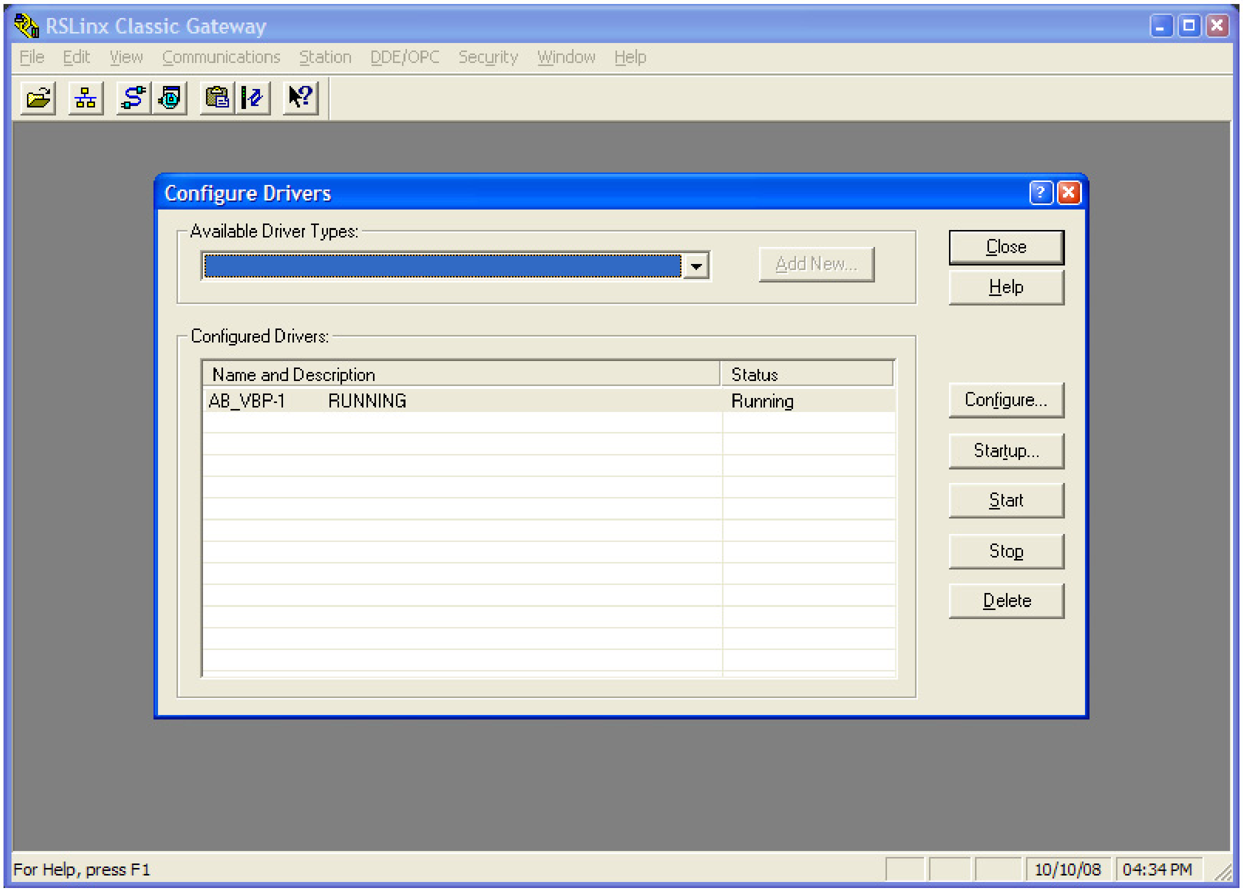
RSLinx

Before you can configure the emulator, you must install the emulator driver in RSLinx. To configure RSLinx, click Start > All Programs > Rockwell Software > RSLinx > RSLinx Classic

Click Communications > Configure Drivers

Click the dropdown to add a new driver. Choose "Virtual Backplane"

Your screen should look like this.



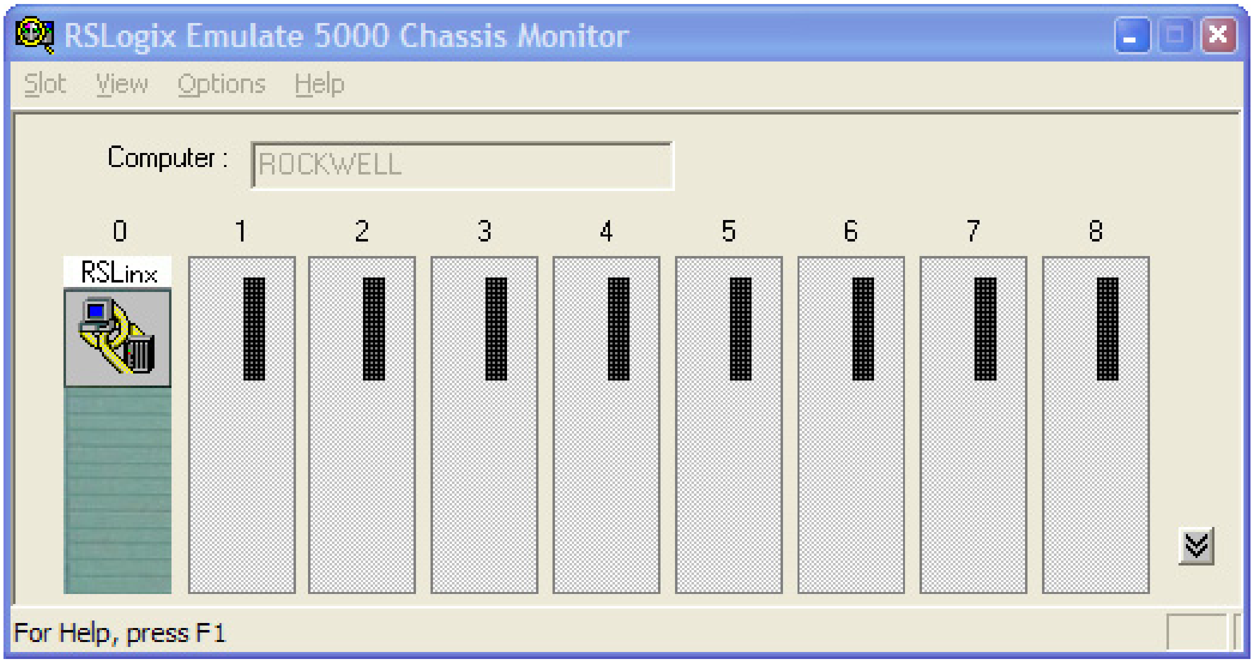
Name and Description	Status
AB_VBP-1 RUNNING	Running

Click “Close”.

Emulator

To run the emulator, click Start > All Programs > Rockwell Software > RSLogix Emulate 5000

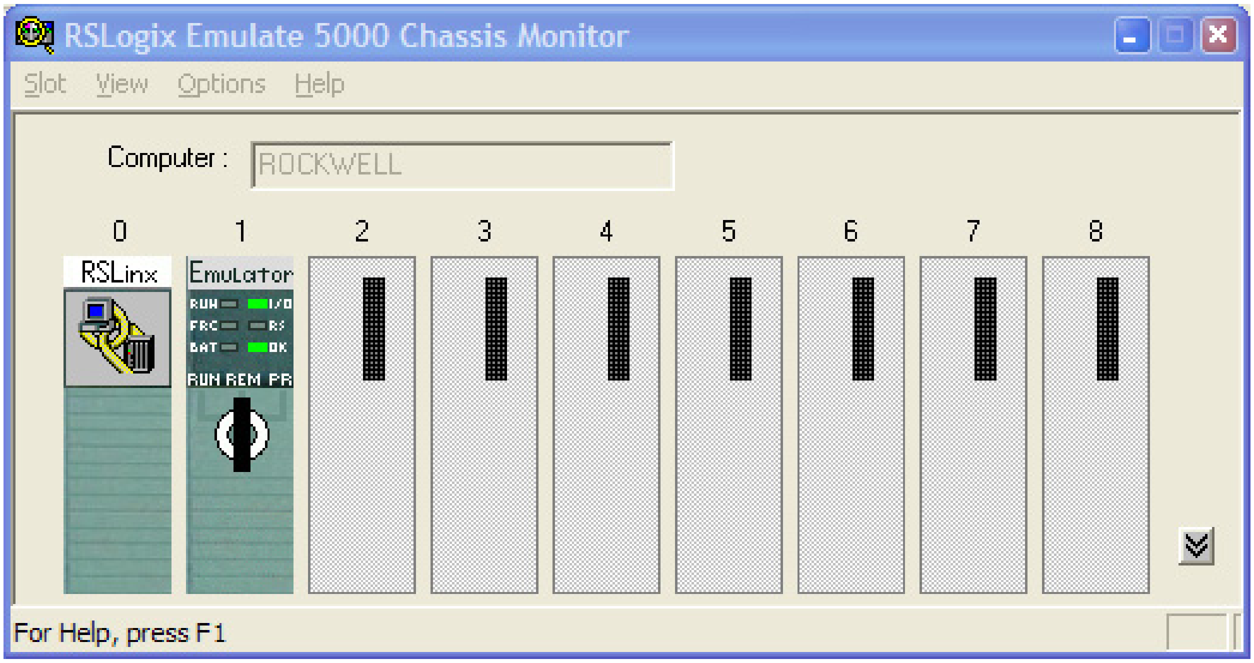
You will see this.



Computer : ROCKWELL

Choose Slot > Create Module > Emulator RSLogix Emulate 5000 Controller

Follow the prompts and this should appear.



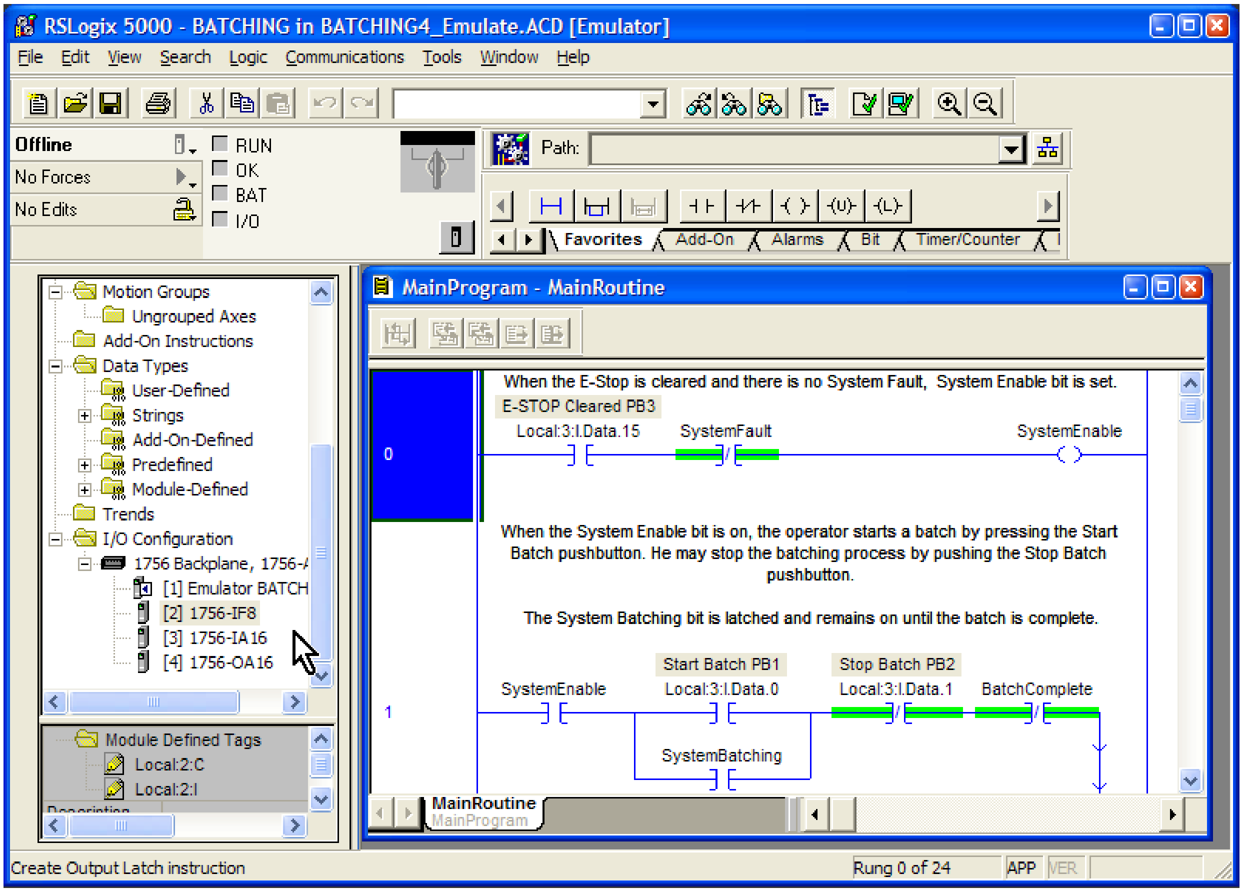
Change the Controller Type in the Batching Program

Before you can emulate a ladder logic project, you must configure the project's controller type to use the emulator. At this point, I make a copy of my working file as a precaution, as we need to alter the I/O Configuration to get the emulator to work.

The emulator needs to have the processor in Slot 1. You may try to run the emulator without changing the I/O, but frankly, I have always had to re-configure the I/O to get things to work.

Click Edit > Controller Properties. The Controller Properties window opens. Click the “Change Controller” button and select “Emulator RSLogix Emulate 5000 Controller”. Close the window.

Starting with the module in Slot 3, right-click on each module in the controller organizer and bump the slot number up by one, until the configuration is as shown below:



When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

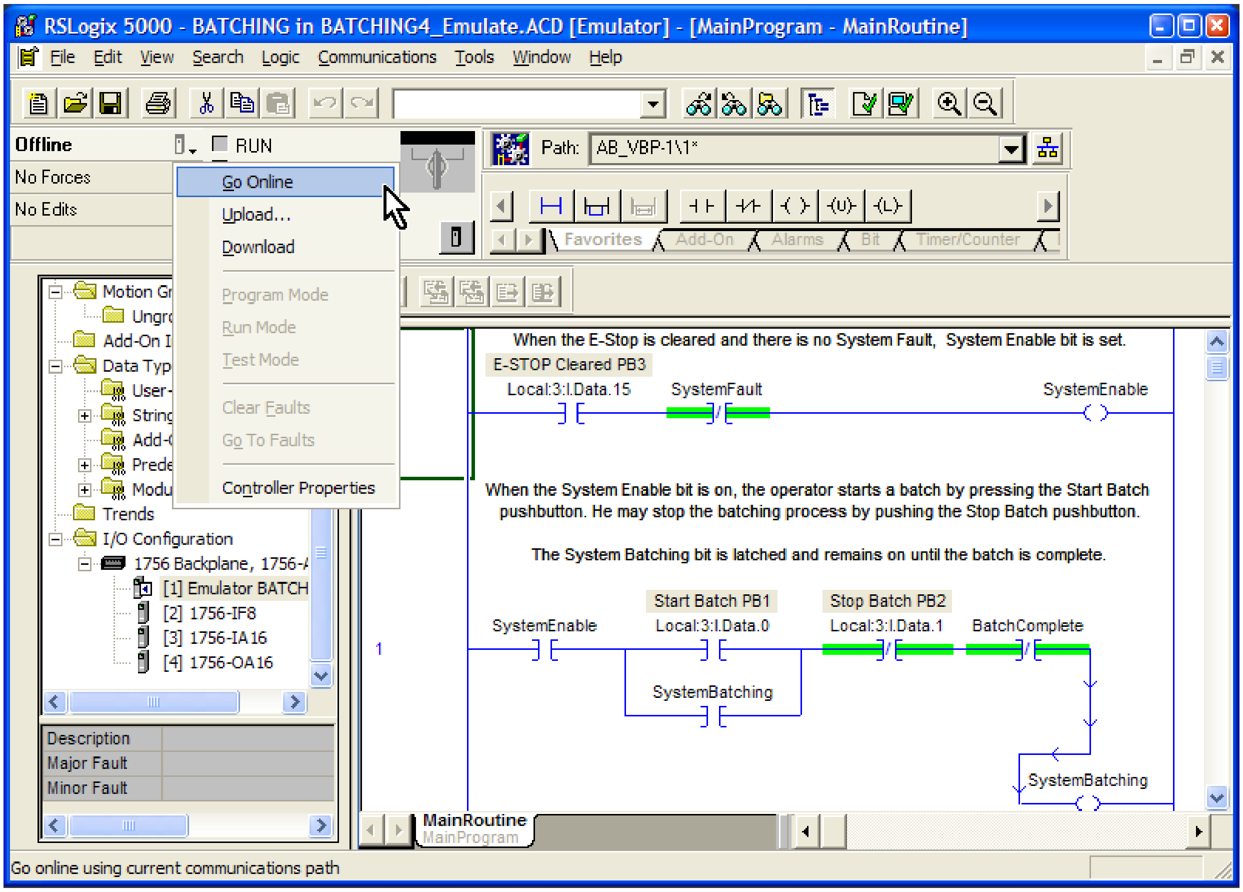
The System Batching bit is latched and remains on until the batch is complete.

Click Communications > Who Active

Expand the tree until your emulator is shown. Click the “Download” button to send the file to the emulator.

The next step is to go online.

Find the mode drop-down menu in the upper left. It currently says “Offline”.



When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

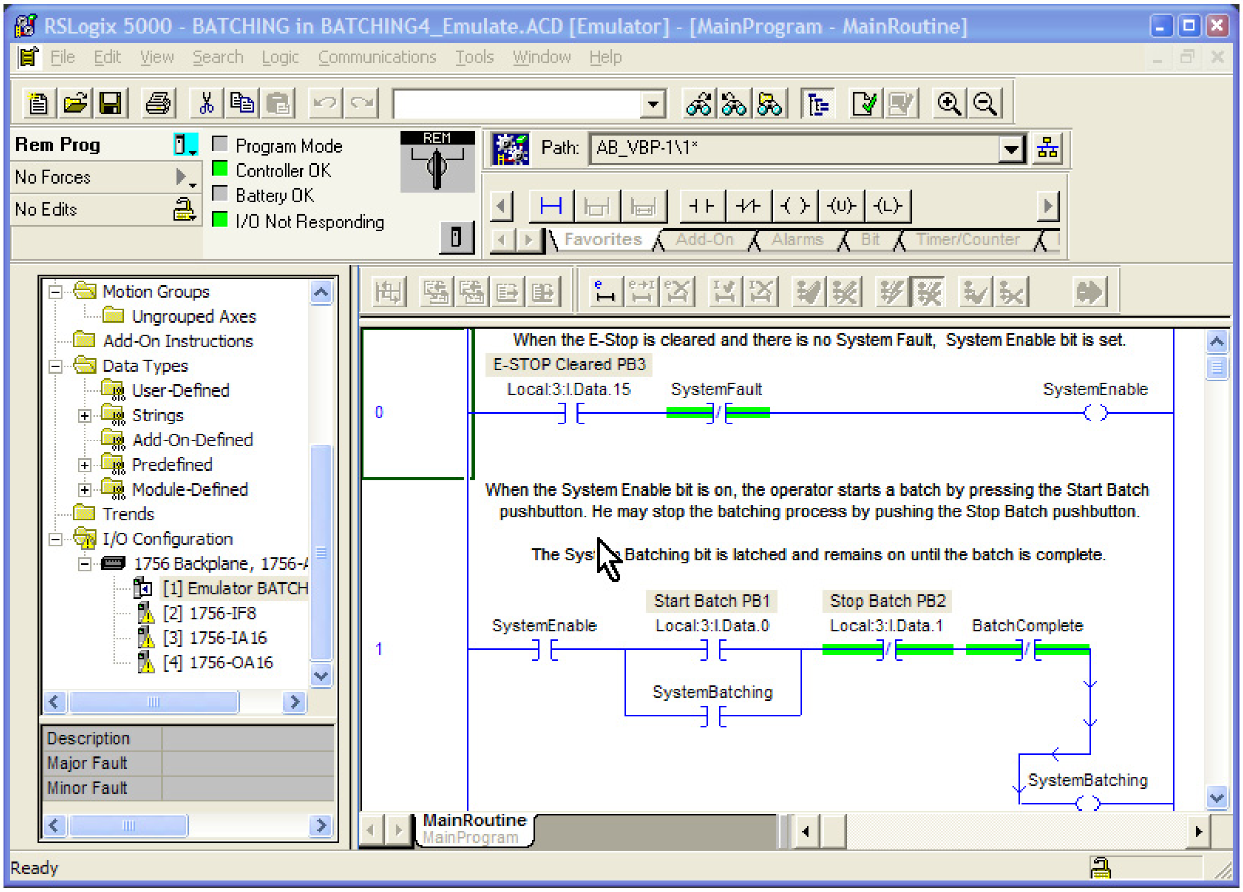
When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

The System Batching bit is latched and remains on until the batch is complete.

Go online using current communications path

From the dropdown menu, choose “Go Online”.

The processor is now in Remote Program mode.

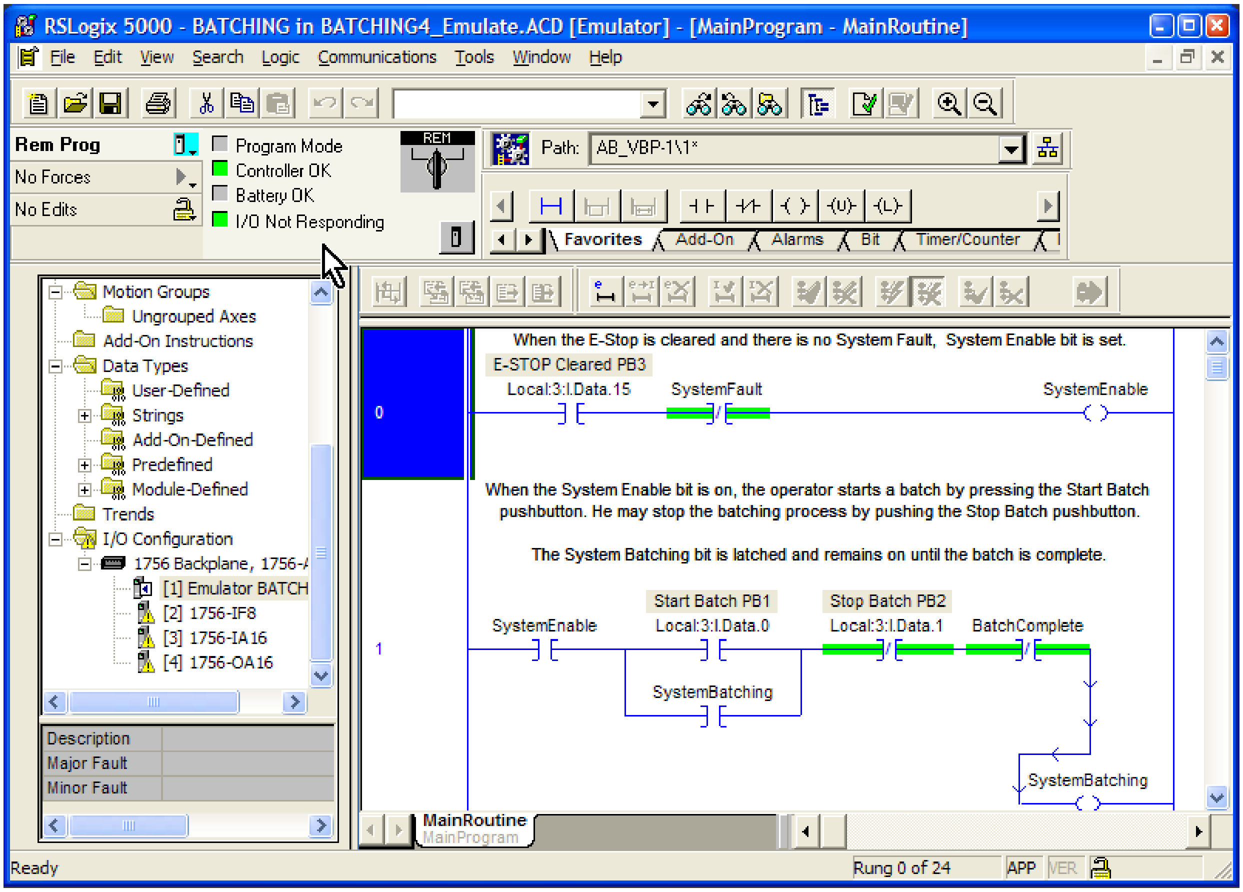


When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

The Sys Batching bit is latched and remains on until the batch is complete.

You see that the I/O is not responding as indicated by the green light in the status area. Also, there are fault symbols showing in the I/O Configuration area. That is OK for now.



When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

The System Batching bit is latched and remains on until the batch is complete.

Switch the processor to Run Mode with the dropdown menu (just to the left of where it currently says "Rem Prog").

Please note that if you were connected to a PLC that is operating a machine, you would take much greater care when switching to Run Mode.

A ControlLogix PLC has a key switch with three positions: RUN (run mode), REM (remote mode, meaning that the mode can be selected from RSLogix) and PROG (program mode).

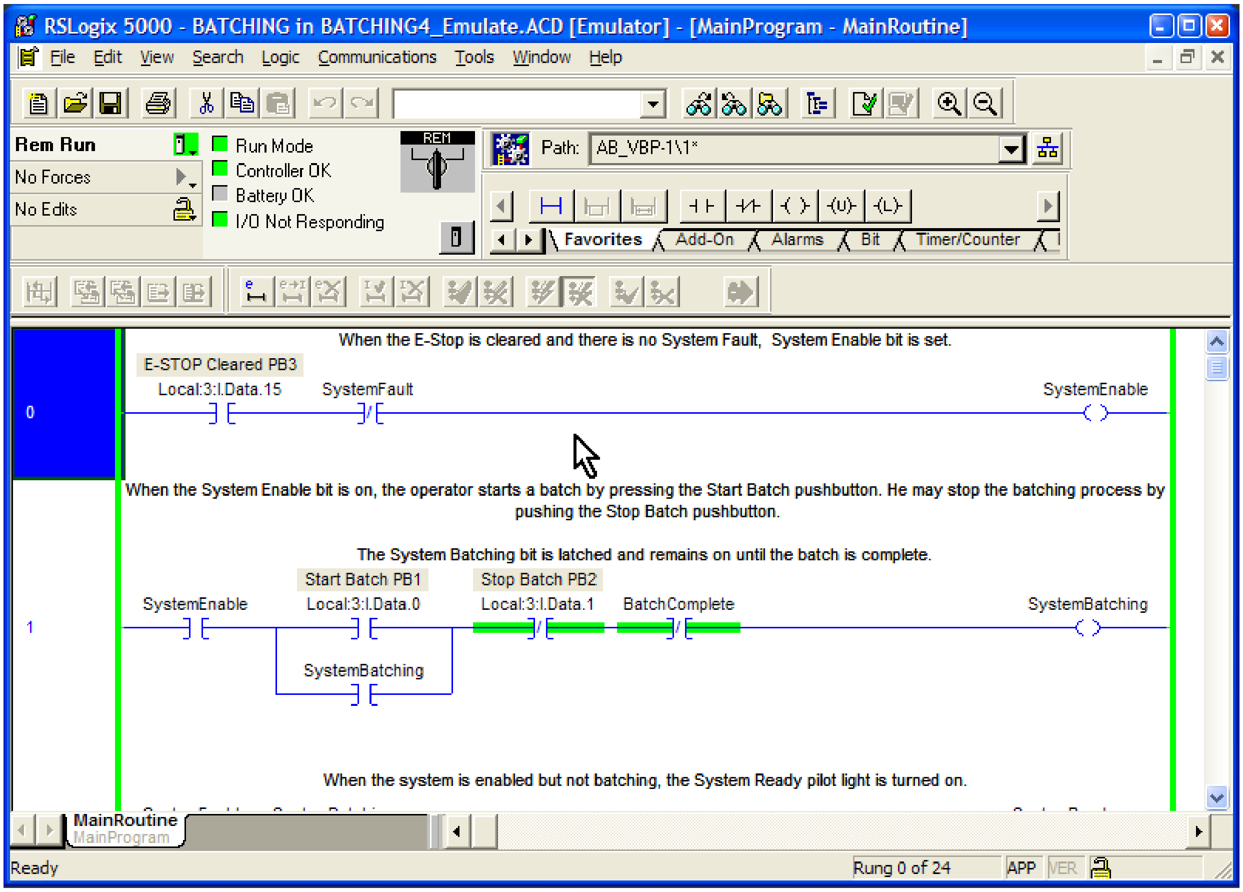
The three modes work this way:□□

Turning the key switch to RUN puts the processor in RUN mode. RSLogix can only monitor the program and data. No downloads or changes to the program can be mode.□□

REM allows RSLogix to put the PLC in Remote Run mode or Remote Program mode. Most often, the key switch is left permanently in this position.□□

PROG lets RSLogix upload or download programs, change data, etc. However, RSLogix cannot put the PLC in Run mode with the switch in this position.

Hide the Controller Organizer and your screen looks like this.



When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

The System Batching bit is latched and remains on until the batch is complete.

When the system is enabled but not batching, the System Ready pilot light is turned on.

We could have chosen "Test Mode". This is a special mode that allows the program to run, but disables all outputs. This would be a good practice if we were on an actual start-up and personnel and equipment safety was an issue.

You can see that the “E-STOP Cleared PB3” bit is off, making the XIC instruction off as indicated by the lack of a green highlight on the instruction.

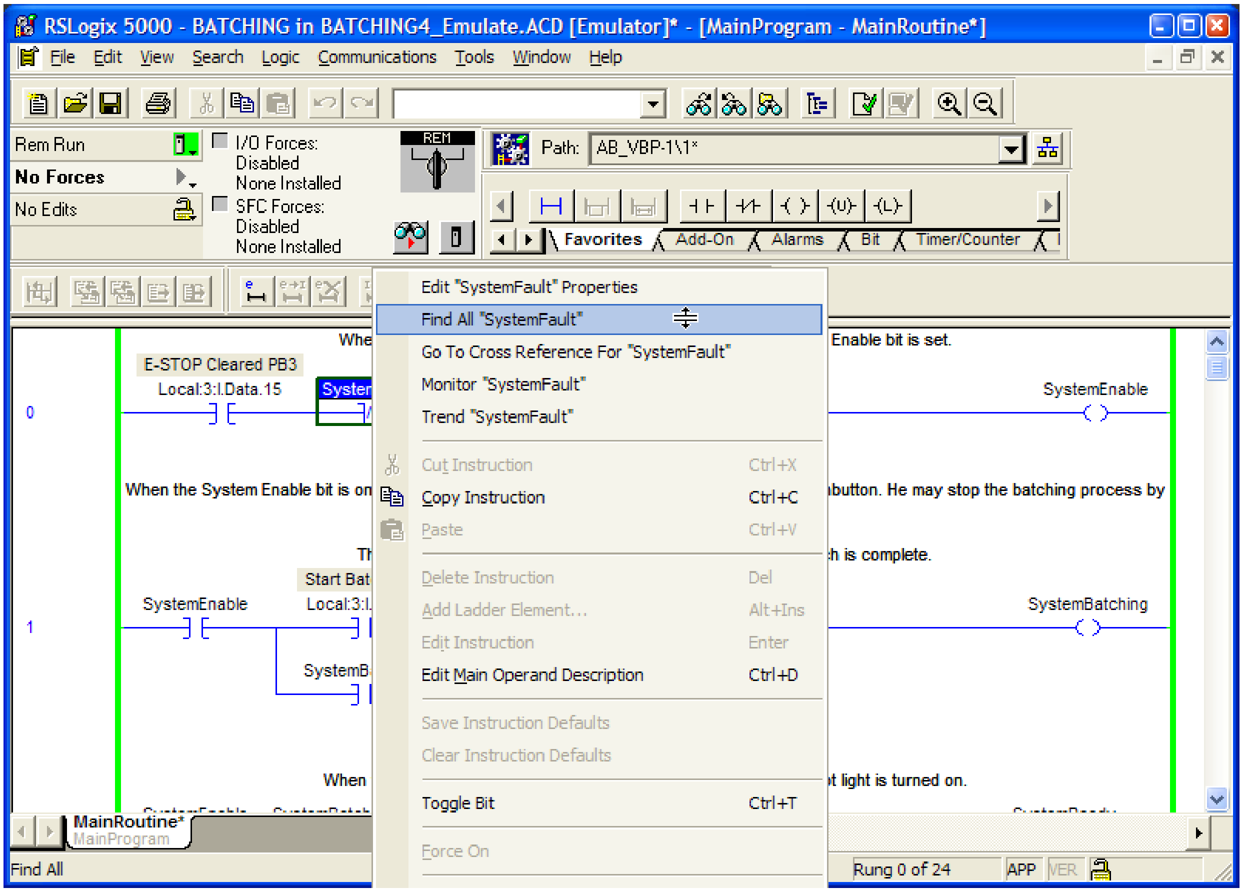
It seems we have a system fault. We know that the “SystemFault” bit is ***on***, since the ***XIO*** instruction in Rung 0 is ***false***. Again, a true state is indicated by the green highlight on the instruction.

Remember:
The XIC (normally open) instruction tells the PLC to look at a bit, and if the bit is ON, then the instruction is true.

The XIO (normally closed) instruction tells the PLC to look at a bit, and if the bit is OFF, then the instruction is true.

Let’s find out why we have a system fault.

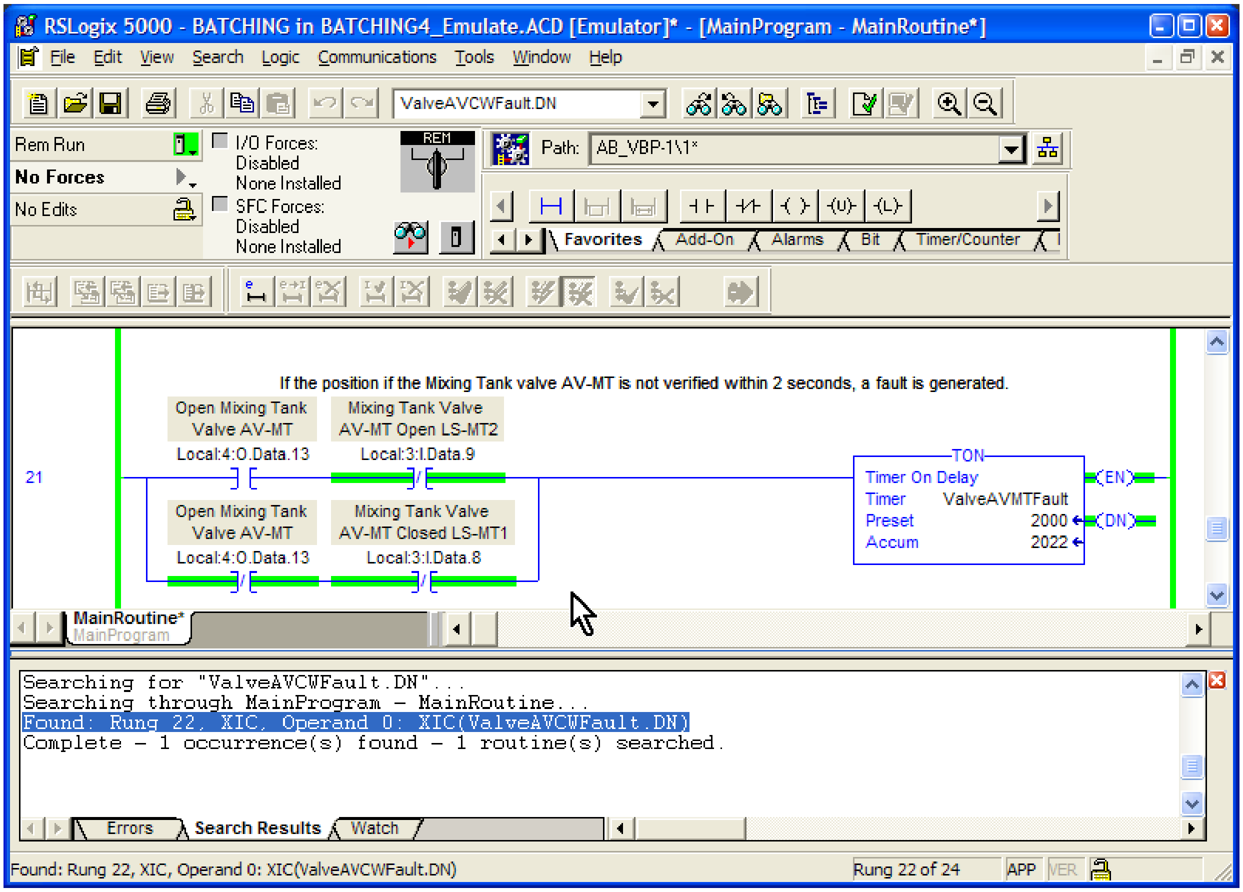
Right-click on the “SystemFault” tag name in Rung 0. Choose Find All “System Fault”


RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter
Edit "SystemFault" Properties
Find All "SystemFault"
Go To Cross Reference For "SystemFault"
Monitor "SystemFault"
Trend "SystemFault"
Cut Instruction Ctrl+X
Copy Instruction Ctrl+C
Paste Ctrl+V
Delete Instruction Del
Add Ladder Element... Alt+Ins
Edit Instruction Enter
Edit Main Operand Description Ctrl+D
Save Instruction Defaults
Clear Instruction Defaults
Toggle Bit Ctrl+T
Force On
0
E-STOP Cleared PB3
Local:3:I.Data.15
Syster
Whe
Enable bit is set.
SystemEnable
When the System Enable bit is on
ıbutton. He may stop the batching process by
Th
ɦ is complete.
1
SystemEnable
Start Bat
Local:3:I.
SystemB
SystemBatching
When
ıt light is turned on.
MainRoutine*
MainProgram
Find All
Rung 0 of 24
APP VER

A new window appears at the bottom, showing links to all the occurrences of “SystemFault”. Click on the link that contains the OTE.

We can see that the valve fault timers have timed out, turned on their respective done bits (DN) and caused a system fault.

Scroll up a rung and you will see that the output for mixing tank valve AV-MT on Local:4:O.Data.13 is not energized. In the lower branch of Rung 21 you can see that the input Local:I.Data.8 from the mixing tank valve AV-MT closed limit switch is not energized.


RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
ValveAVCWFault.DN
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter
21
Open Mixing Tank Valve AV-MT
Local:4:O.Data.13
Mixing Tank Valve AV-MT Open LS-MT2
Local:3:I.Data.9
Open Mixing Tank Valve AV-MT
Local:4:O.Data.13
Mixing Tank Valve AV-MT Closed LS-MT1
Local:3:I.Data.8
TON
Timer On Delay
Timer ValveAVMTFault
Preset 2000
Accum 2022
EN
DN
MainRoutine*
MainProgram
Errors
Search Results
Watch
Rung 22 of 24
APP
VER

If the position if the Mixing Tank valve AV-MT is not verified within 2 seconds, a fault is generated.

```
Searching for "ValveAVCWFault.DN"...
Searching through MainProgram - MainRoutine...
Found: Rung 22, XIC, Operand 0: XIC(ValveAVCWFault.DN)
Complete - 1 occurrence(s) found - 1 routine(s) searched.
```

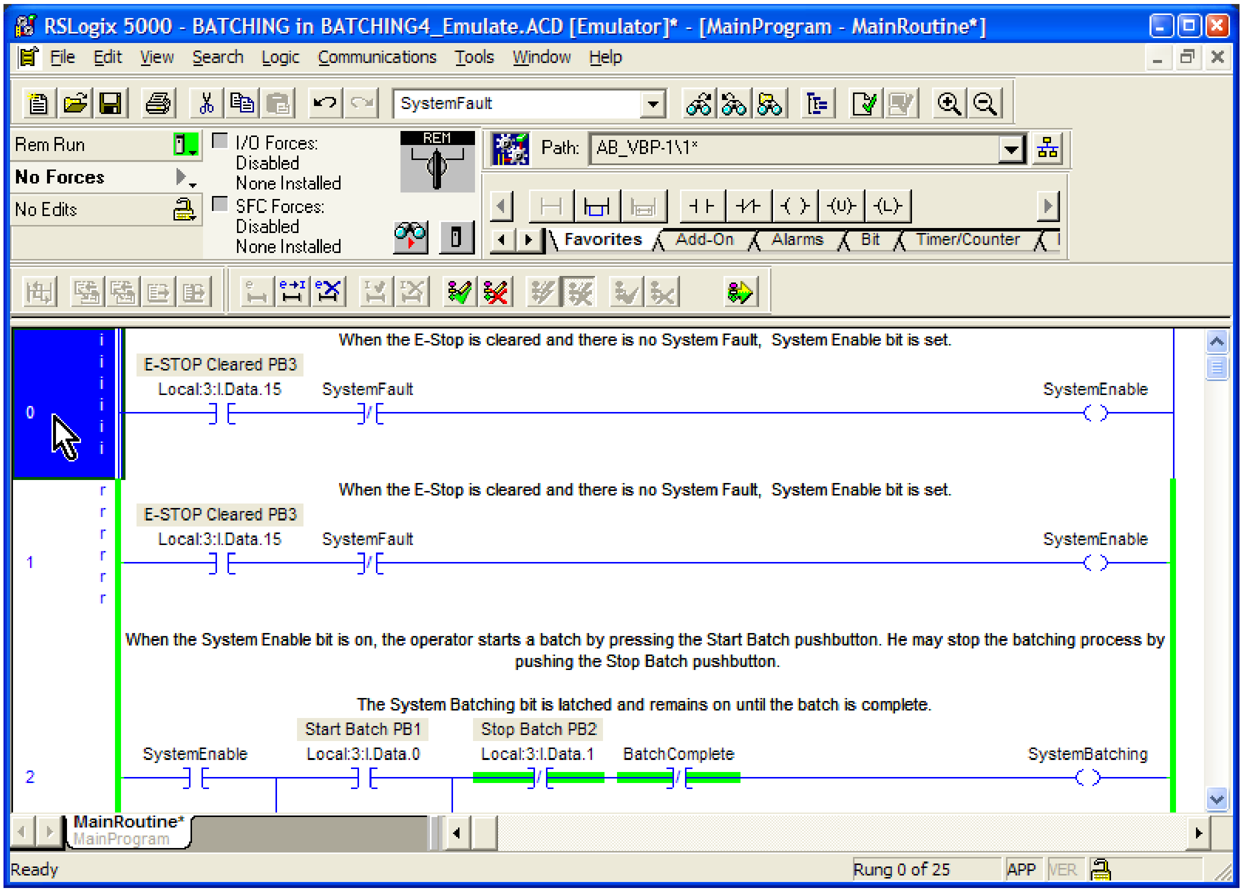
Found: Rung 22, XIC, Operand 0: XIC(ValveAVCWFault.DN)

Normally, of course, if we were telling the valve to close, we would get a signal from the limit switch on the input.

We are going to have to temporarily disable the system fault bit so that we can test the remainder of the program. We will do this by putting a “jumper” across the “SystemFault” instruction in Rung 0.

Close the search window at the bottom of the screen.

Scroll to Rung 0 and double-click to the left of the rung. Your screen looks like this.



RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]

Rem Run
No Forces
No Edits

I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed

Path: AB_VBP-1\1*

0 i i i i i i

When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

1 r r r r r r

When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

2

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

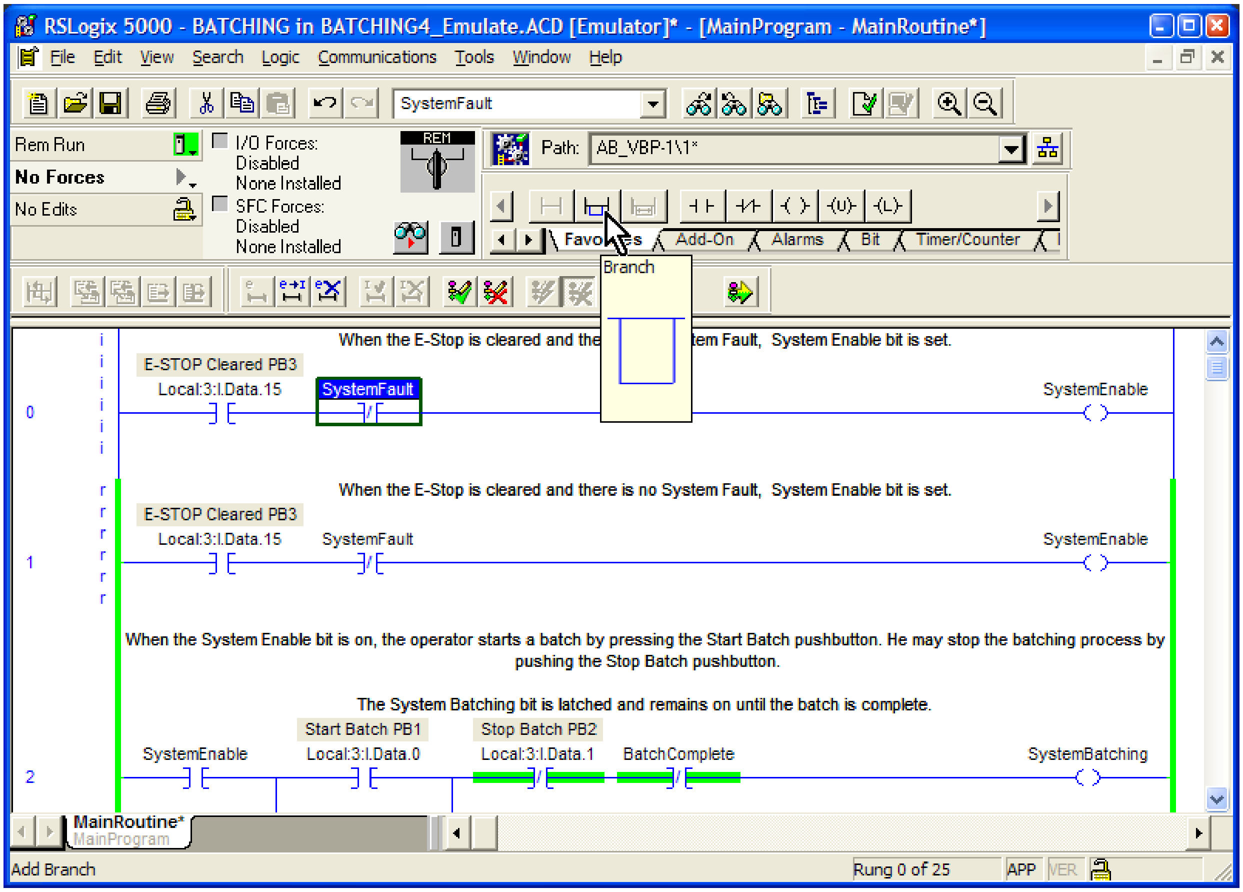
The System Batching bit is latched and remains on until the batch is complete.

Ready

Rung 0 of 25

The rung you are editing is a copy of Rung 0 and has the letter “i” to the left. The rung shown below has the letter “r” to the left, indicating that the PLC is still running this rung.

Find the “Branch” icon in the toolbar.



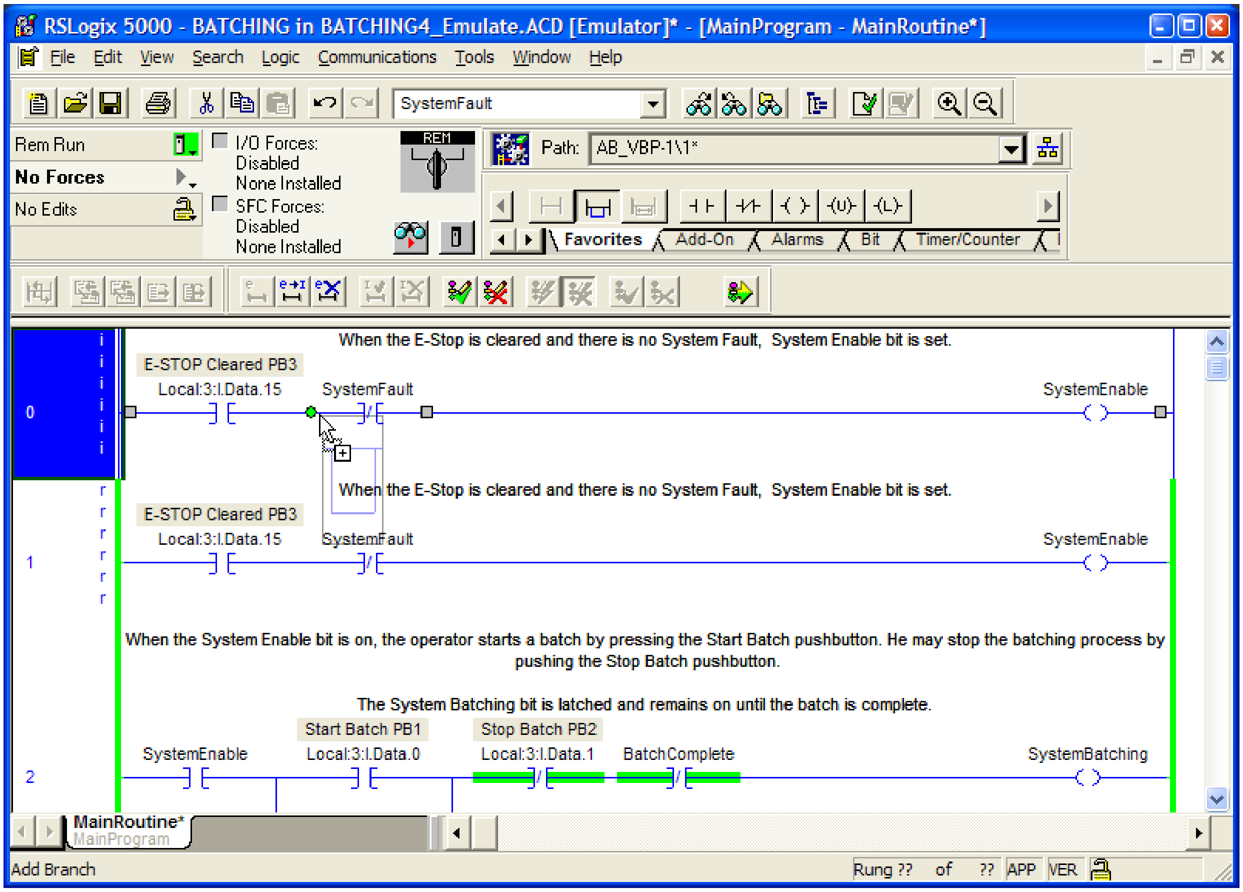
When the E-Stop is cleared and the tem Fault, System Enable bit is set.

When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

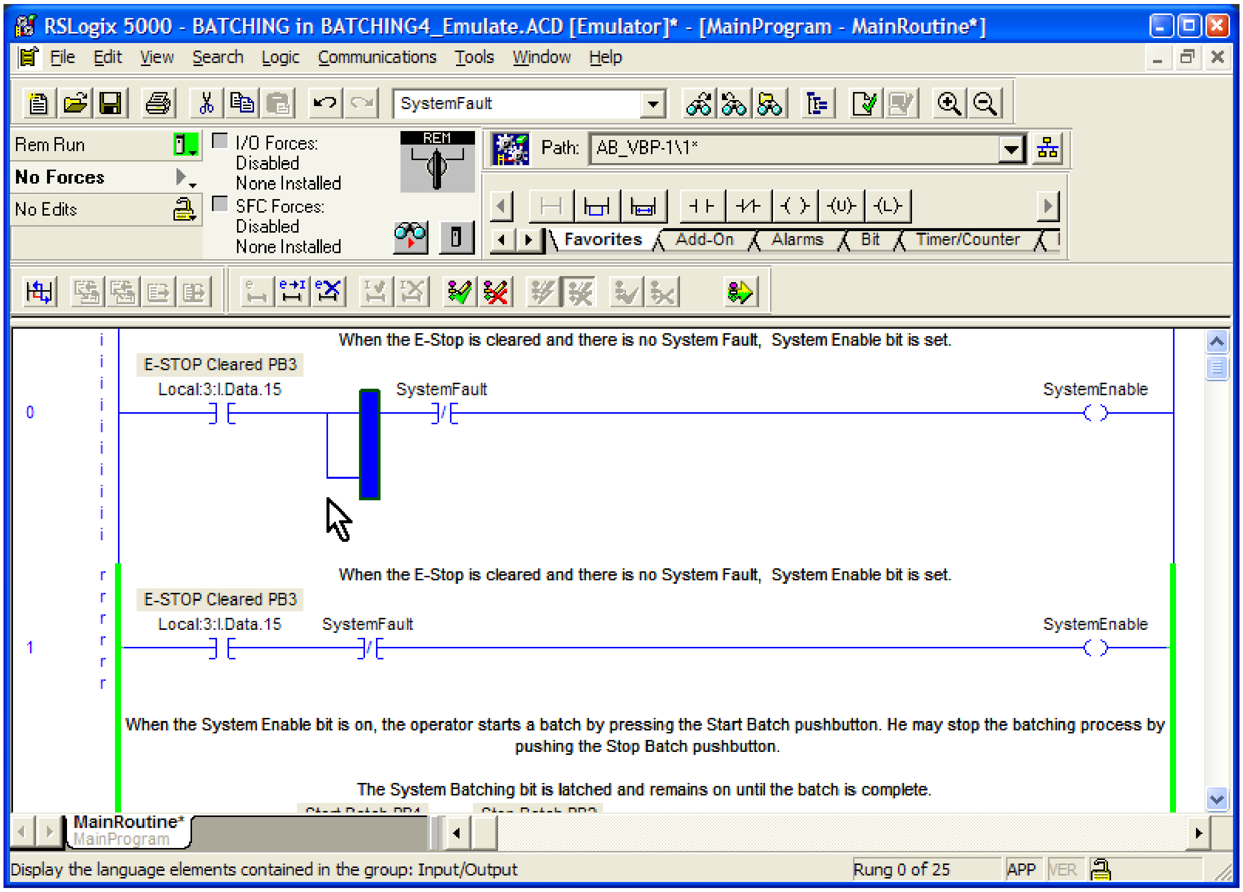
The System Batching bit is latched and remains on until the batch is complete.

Click and drag the branch to Rung 0.


RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
SystemFault
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter
When the E-Stop is cleared and there is no System Fault, System Enable bit is set.
E-STOP Cleared PB3
Local:3:I.Data.15
SystemFault
SystemEnable
0
1
When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.
The System Batching bit is latched and remains on until the batch is complete.
Start Batch PB1
Stop Batch PB2
Local:3:I.Data.0
Local:3:I.Data.1
BatchComplete
SystemBatching
2
MainRoutine*
MainProgram
Add Branch
Rung ?? of ?? APP VER

Small squares on the rung indicate potential landing spots for the branch. As you near a spot, the icon changes to a green circle. When the cursor is near the landing spot in front of the “SystemFault” bit, release the mouse button.

Your screen looks like this.



When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

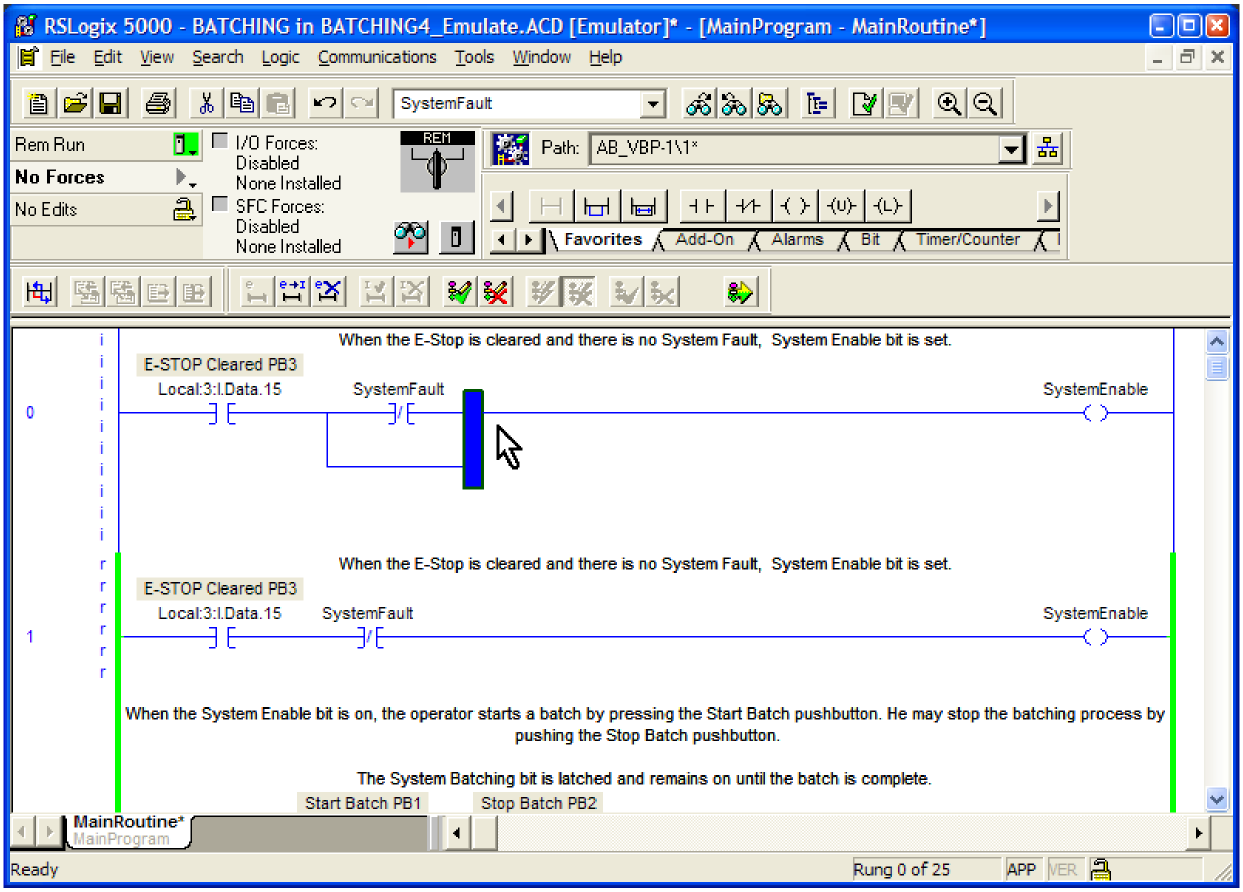
When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

The System Batching bit is latched and remains on until the batch is complete.

Display the language elements contained in the group: Input/Output

Click and drag the right part of the branch and place it to the right of the "SystemFault" instruction.



RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
SystemFault
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter

0 When the E-Stop is cleared and there is no System Fault, System Enable bit is set.
E-STOP Cleared PB3
Local:3:I.Data.15
SystemFault
SystemEnable

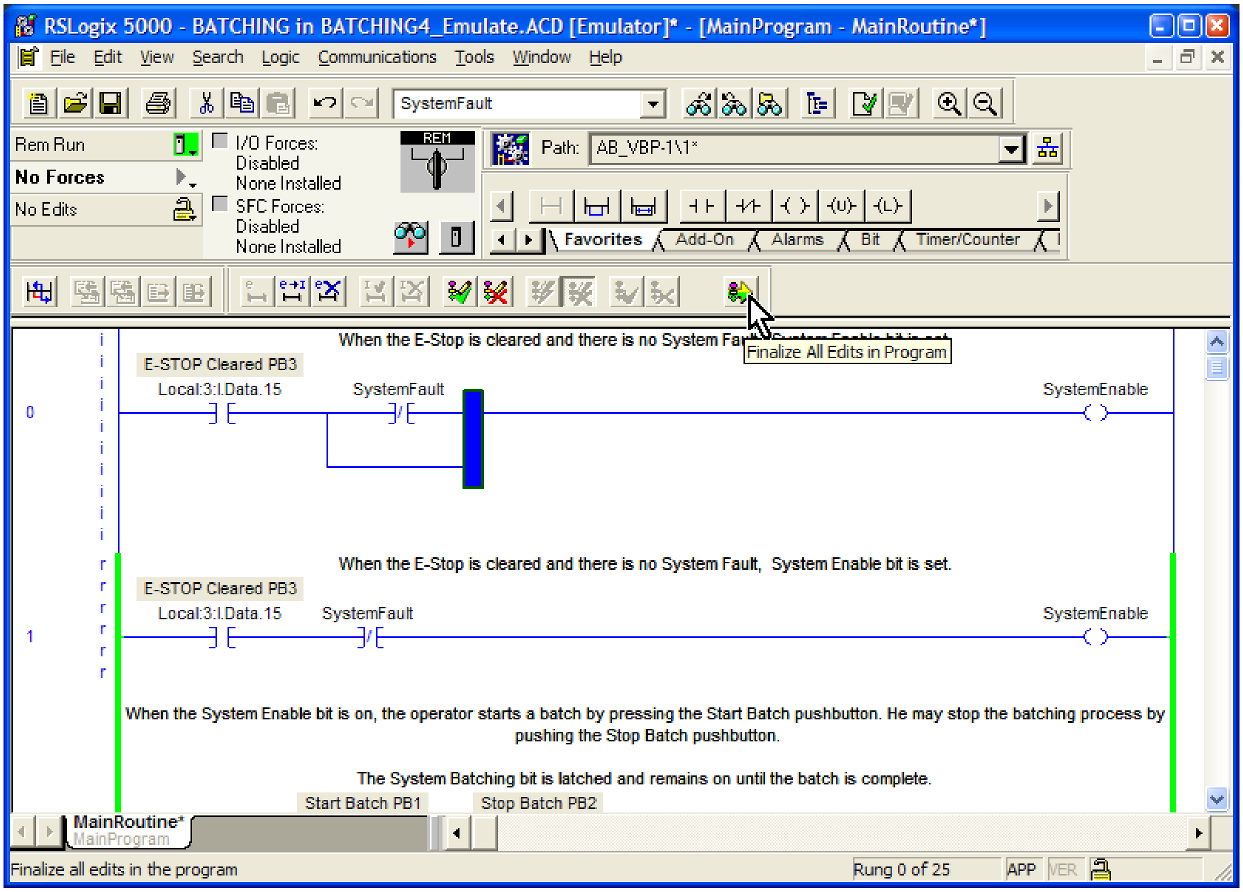
1 When the E-Stop is cleared and there is no System Fault, System Enable bit is set.
E-STOP Cleared PB3
Local:3:I.Data.15
SystemFault
SystemEnable

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

The System Batching bit is latched and remains on until the batch is complete.

Start Batch PB1
Stop Batch PB2

Find the “Finalize Edits” button in the toolbar.


RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
SystemFault
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites
Add-On
Alarms
Bit
Timer/Counter
Finalize All Edits in Program
MainRoutine*
MainProgram
Rung 0 of 25
APP
VER

When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

0 E-STOP Cleared PB3 Local:3:I.Data.15 — SystemFault — SystemEnable

When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

1 E-STOP Cleared PB3 Local:3:I.Data.15 — SystemFault — SystemEnable

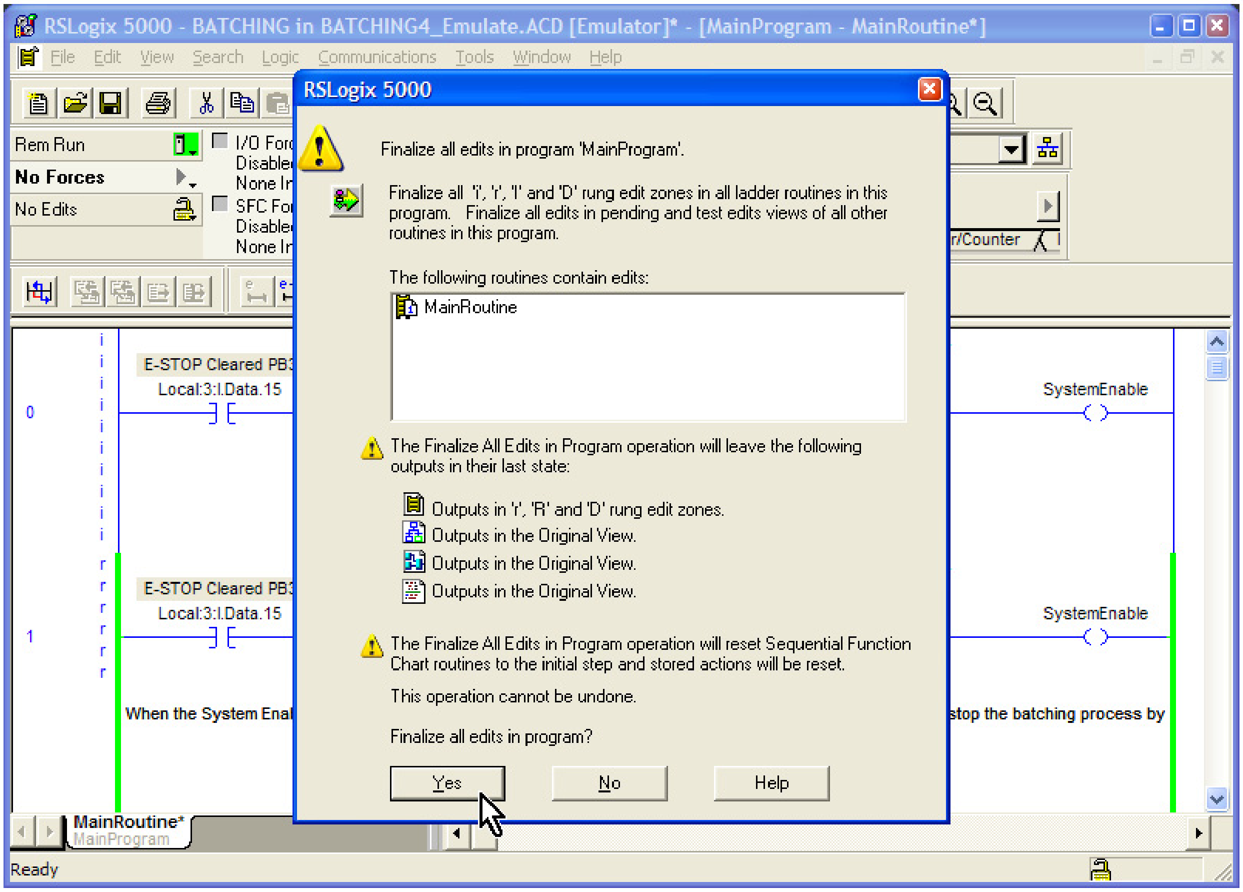
When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

The System Batching bit is latched and remains on until the batch is complete.

Start Batch PB1 Stop Batch PB2

Finalize all edits in the program

Click it and you will see this dialog box.


RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
Rem Run
No Forces
No Edits
I/O Forc
Disable
None Ir
SFC For
Disable
None Ir
r/Counter
E-STOP Cleared PB3
Local:3:I.Data.15
SystemEnable
0
1
When the System Enal
stop the batching process by
MainRoutine*
MainProgram
Ready

RSLogix 5000

Finalize all edits in program 'MainProgram'.

Finalize all 'i', 'r', 'I' and 'D' rung edit zones in all ladder routines in this program. Finalize all edits in pending and test edits views of all other routines in this program.

The following routines contain edits:

MainRoutine

The Finalize All Edits in Program operation will leave the following outputs in their last state:

- Outputs in 'r', 'R' and 'D' rung edit zones.
- Outputs in the Original View.
- Outputs in the Original View.
- Outputs in the Original View.

The Finalize All Edits in Program operation will reset Sequential Function Chart routines to the initial step and stored actions will be reset.

This operation cannot be undone.

Finalize all edits in program?

This is warning you that, among other things, the change cannot be undone (without re-editing the rung, at least – there is no "undo".

Click the "Yes" button.

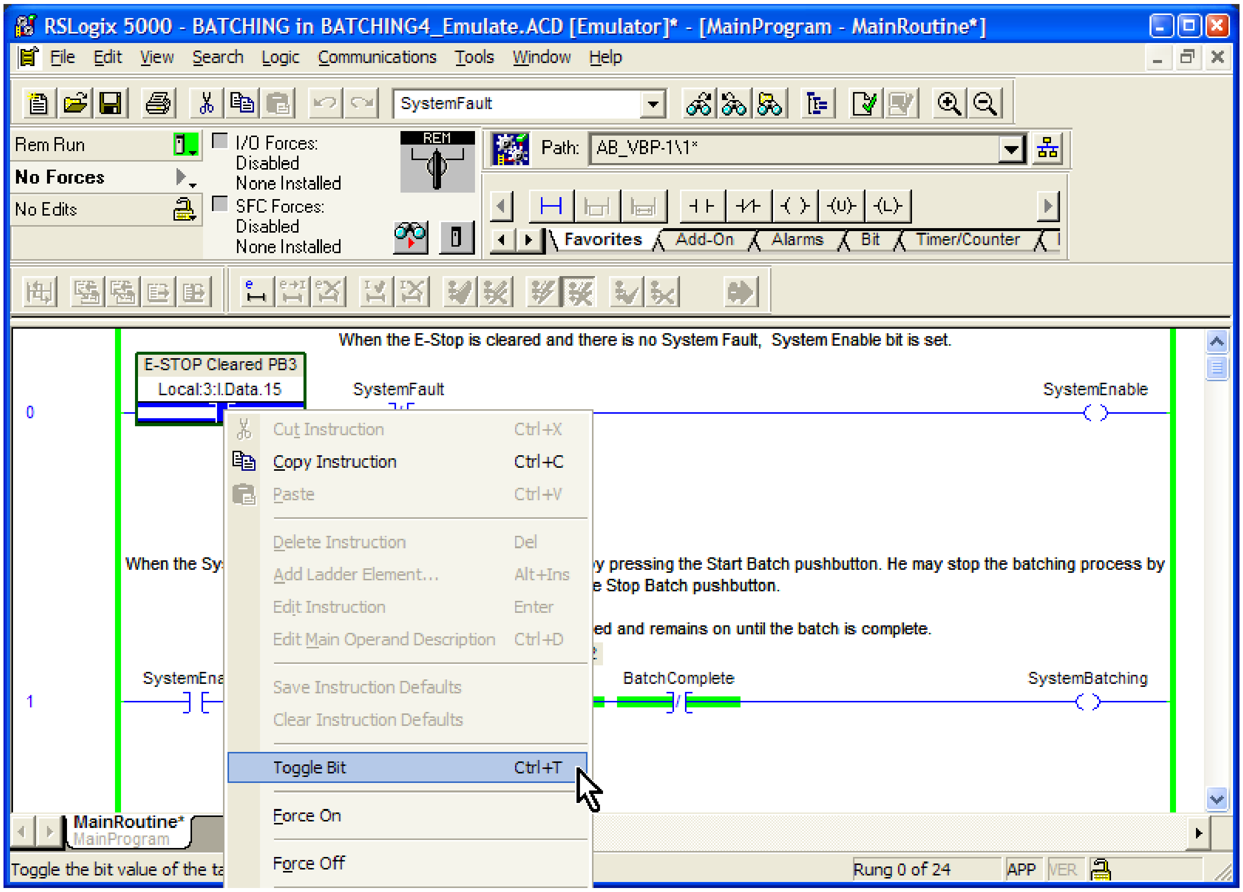
You see that the letters to the left of the rungs are gone. The PLC is now processing Rung 0 with the new logic.

However, the “SystemEnable” bit is still off. That is because we are not getting a signal from the “E-Stop Cleared PB3”.

RSLogix allows you to toggle the state of a bit. It will remain toggled unless logic, or voltage on an input, forces it to another state.

Since there is no voltage on the “E-Stop Cleared PB3 Local:3:I.Data.15” input, we can toggle it.

Right-click on the instruction.



RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]

Path: AB_VBP-1\1*

When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the Sy… …y pressing the Start Batch pushbutton. He may stop the batching process by …e Stop Batch pushbutton.

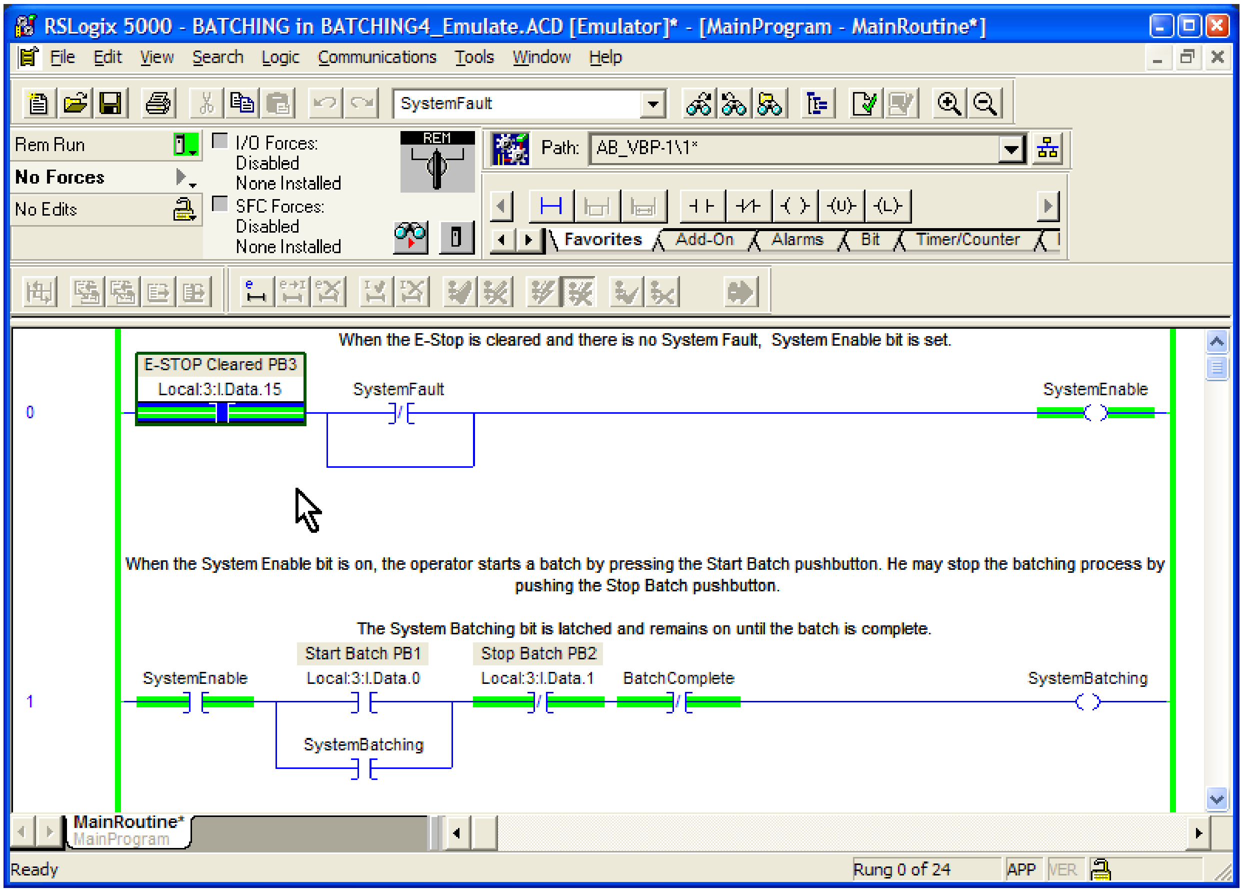
…ed and remains on until the batch is complete.

Toggle the bit value of the ta…

Rung 0 of 24

Choose “Toggle Bit”.

Now, the E-stop input instruction is true, as indicated by the green highlight. Also, the “SystemEnable” OTE is true. The “SystemEnable” XIO instruction in Rung 1 is also true. Since the all the preceding instructions are true, the “SystemEnable” OTE is true.



RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]

When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

The System Batching bit is latched and remains on until the batch is complete.

Ready   Rung 0 of 24

The system is now enabled.

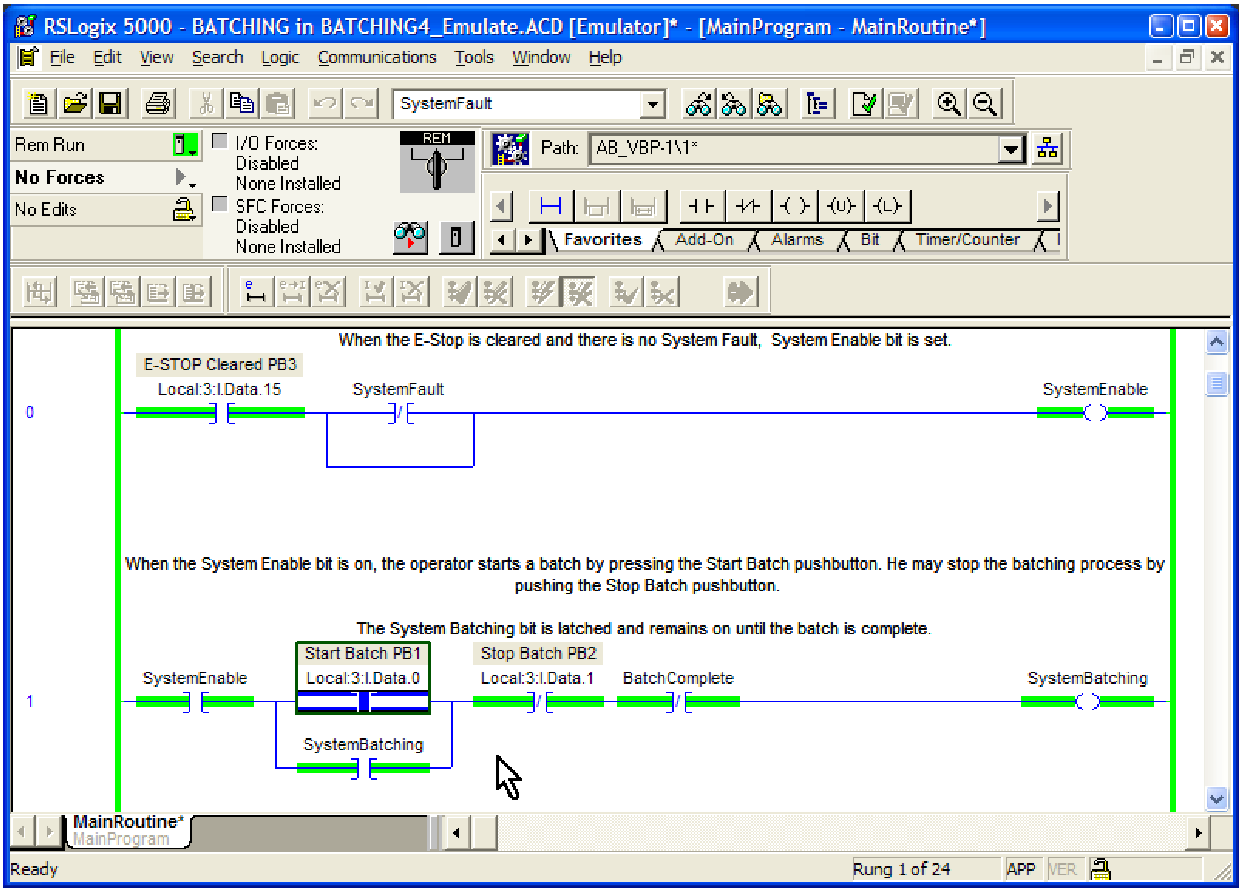
Take a moment to think of the equivalent hardwired relay circuit that would generate the same function as Rung 0.

Imagine that we have a pushbutton switch, labeled E-STOP Cleared PB3, and two relays, SystemFault and SystemEnable. We have wired a normally open contact of E-STOP Cleared PB3 in series to a normally closed contact of SystemFault and to the coil of SystemEnable.

We are pushing the button on switch E-STOP Cleared PB3, so the normally open contact is closed. The coil for relay SystemFault is energized, so the normally closed contacts of SystemFault are open. However, we have placed a jumper across this set of contacts. This energizes the coil of the SystemEnable relay.

Don’t forget to remove any temporary jumpers like this after you are done testing.

Let’s start a batch by toggling the “Start Batch PB1” in Rung 1. You see that the “SystemBatching” bit becomes true and the latch is now active. Toggle it again to the off state to simulate how the pushbutton actually works.



When the E-Stop is cleared and there is no System Fault, System Enable bit is set.

When the System Enable bit is on, the operator starts a batch by pressing the Start Batch pushbutton. He may stop the batching process by pushing the Stop Batch pushbutton.

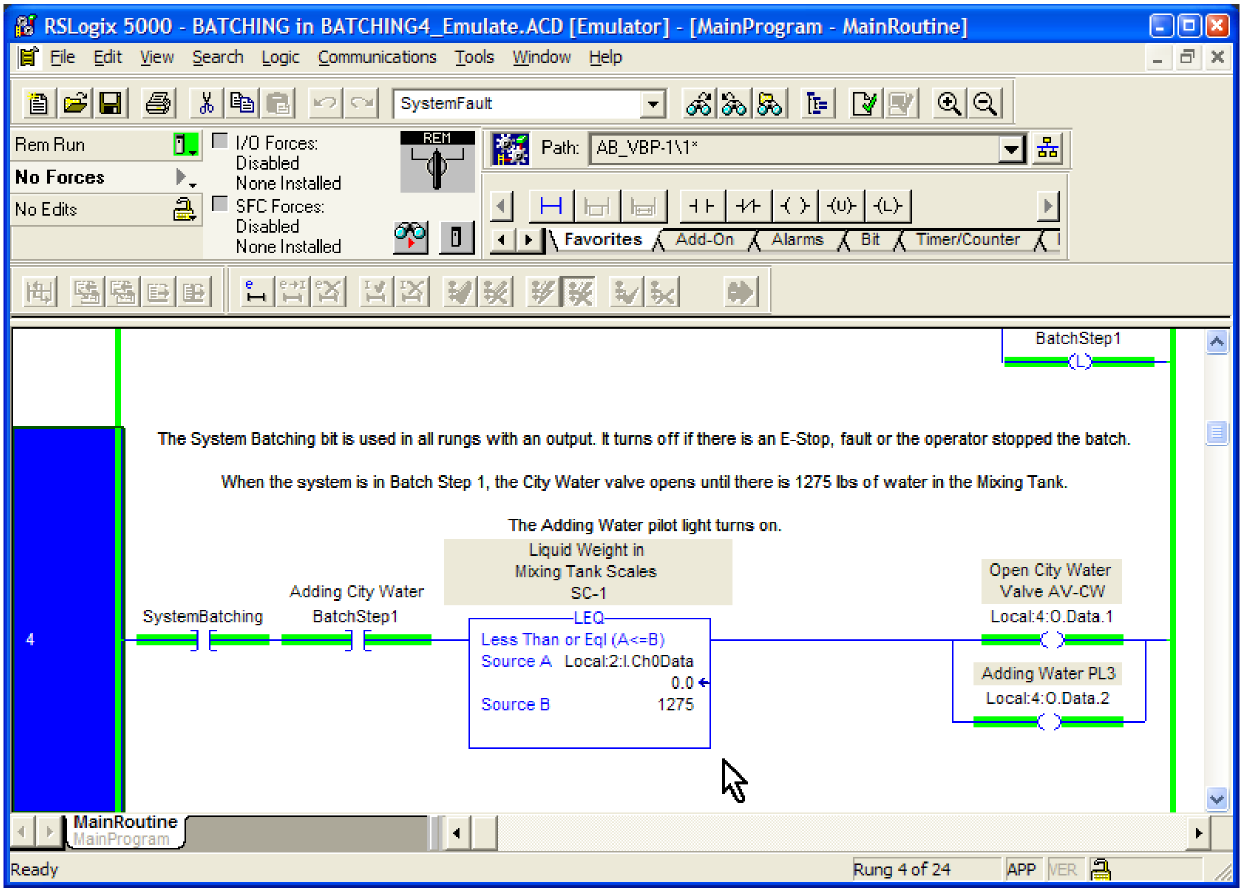
The System Batching bit is latched and remains on until the batch is complete.

In Rung 1, we can see that the system is batching as shown by the OTE instruction used with “SystemBatching”.

Look at Rung 4. The first two instructions look right, as the system is batching and we are in Batch Step 1. Both of these are highlighted.

The outputs are also true.

However, the LEQ (less than or equal to) is not highlighted; Source A is 0, which is certainly less than the value of 1665 in Source B.



RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator] - [MainProgram - MainRoutine]

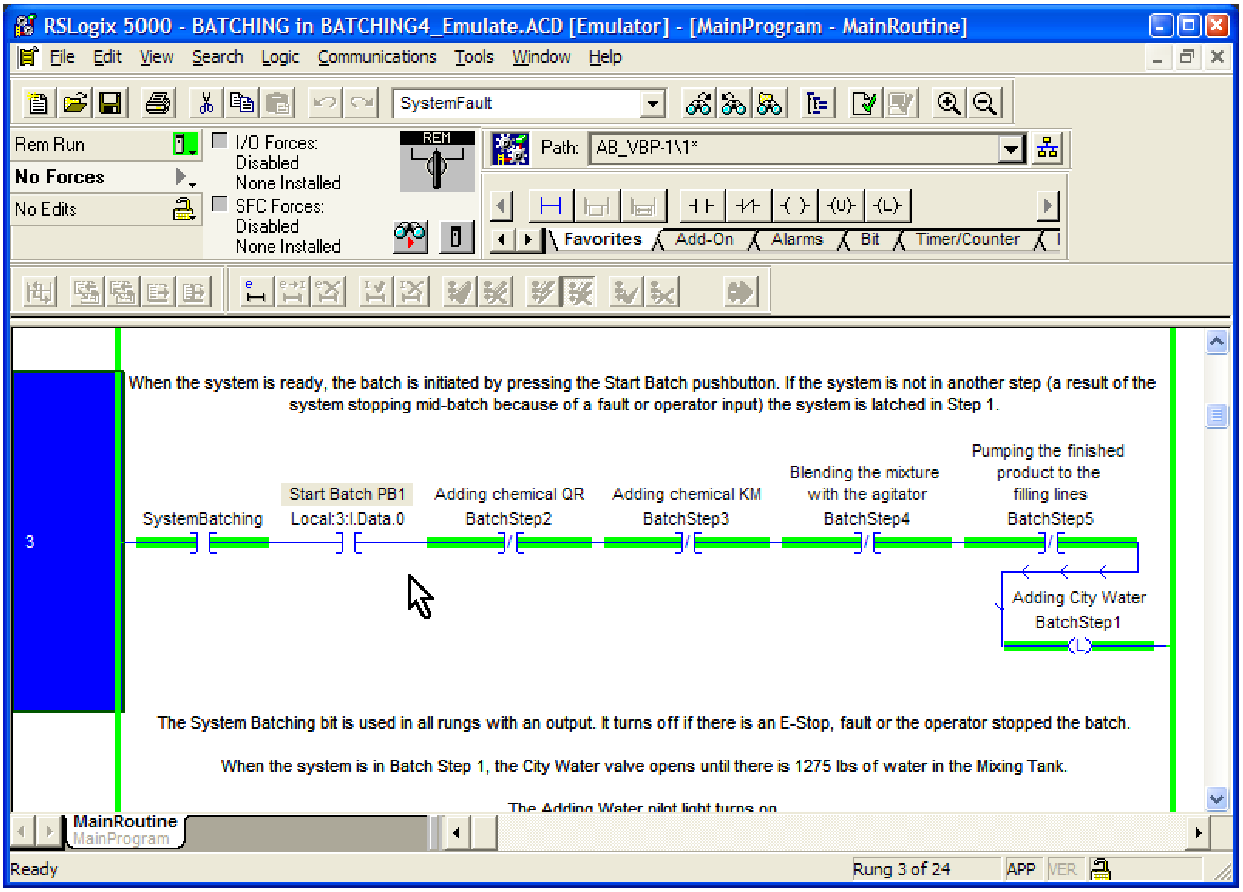
The System Batching bit is used in all rungs with an output. It turns off if there is an E-Stop, fault or the operator stopped the batch.

When the system is in Batch Step 1, the City Water valve opens until there is 1275 lbs of water in the Mixing Tank.

The Adding Water pilot light turns on.

RSLogix does not highlight some instructions even though they are true, such as an LEQ. You have to mentally compare the numbers yourself to see if the instruction is true.

In Rung 3, the latch “Adding City Water BatchStep1” is turned on. Even though the start pushbutton has been released, the latch instruction will stay on.



When the system is ready, the batch is initiated by pressing the Start Batch pushbutton. If the system is not in another step (a result of the system stopping mid-batch because of a fault or operator input) the system is latched in Step 1.

The System Batching bit is used in all rungs with an output. It turns off if there is an E-Stop, fault or the operator stopped the batch.

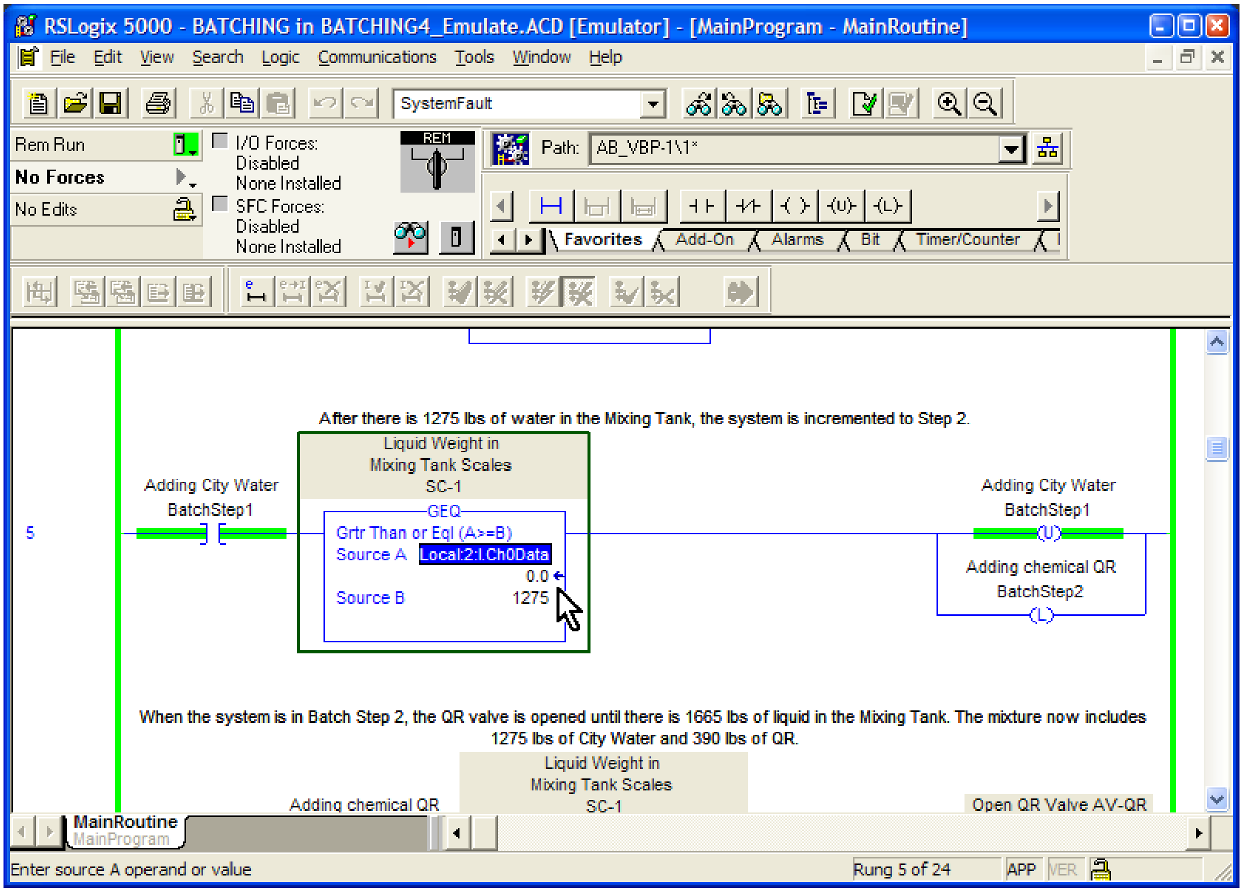
When the system is in Batch Step 1, the City Water valve opens until there is 1275 lbs of water in the Mixing Tank.

The Adding Water pilot light turns on

RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator] - [MainProgram - MainRoutine]
File Edit View Search Logic Communications Tools Window Help
SystemFault
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter
3
SystemBatching
Start Batch PB1
Local:3:I.Data.0
Adding chemical QR
BatchStep2
Adding chemical KM
BatchStep3
Blending the mixture with the agitator
BatchStep4
Pumping the finished product to the filling lines
BatchStep5
Adding City Water
BatchStep1
L
MainRoutine
MainProgram
Ready
Rung 3 of 24
APP
VER

The latch instruction and will remain on until the corresponding unlatch instruction is true.

Scroll to Rung 5. The liquid weight still shows 0.



RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator] - [MainProgram - MainRoutine]
File Edit View Search Logic Communications Tools Window Help
SystemFault
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter

After there is 1275 lbs of water in the Mixing Tank, the system is incremented to Step 2.

5

Adding City Water
BatchStep1

Liquid Weight in
Mixing Tank Scales
SC-1

GEQ
Grtr Than or Eql (A>=B)
Source A Local:2:I.Ch0Data
0.0
Source B 1275

Adding City Water
BatchStep1
(U)

Adding chemical QR
BatchStep2
(L)

When the system is in Batch Step 2, the QR valve is opened until there is 1665 lbs of liquid in the Mixing Tank. The mixture now includes 1275 lbs of City Water and 390 lbs of QR.

Adding chemical QR

Liquid Weight in
Mixing Tank Scales
SC-1

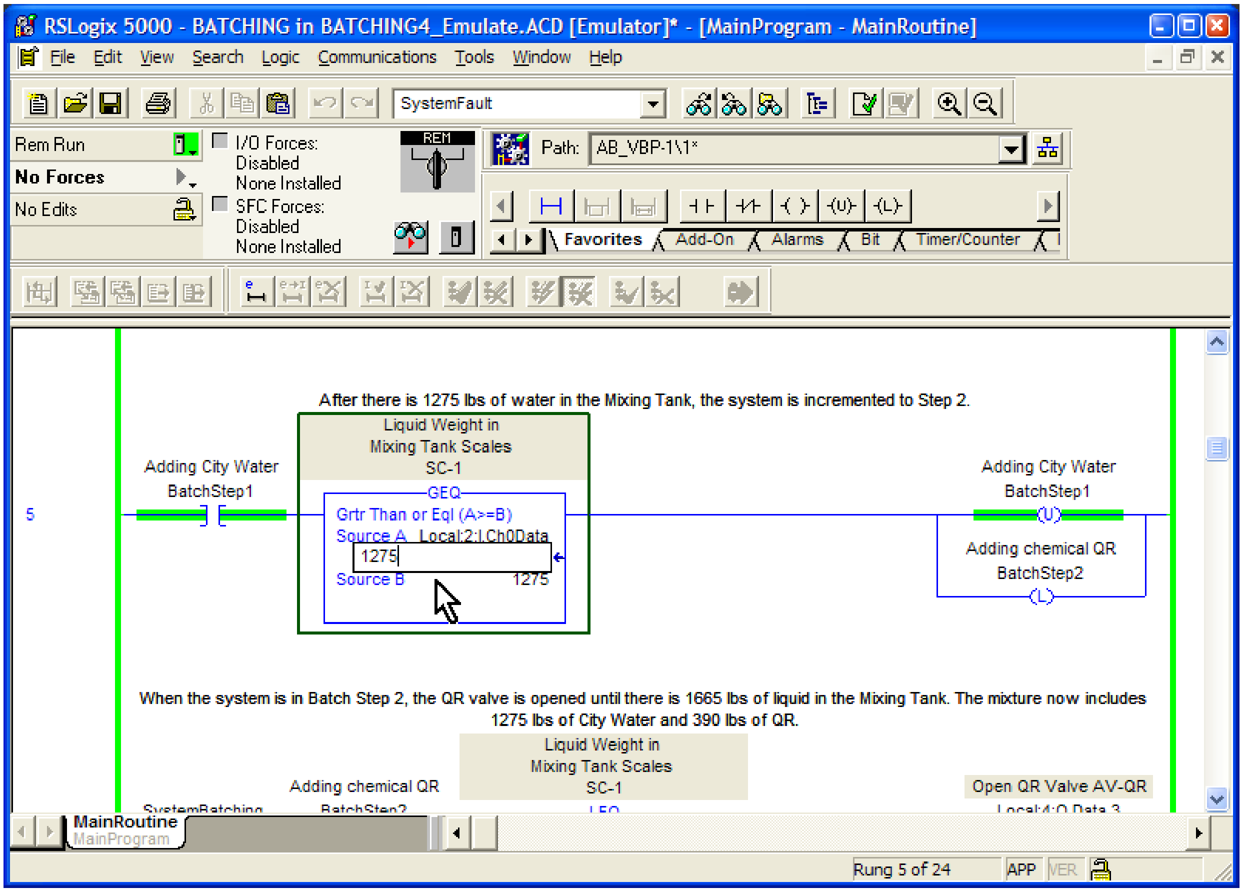
Open QR Valve AV-QR

Enter source A operand or value

Rung 5 of 24

We can simulate adding water to the tank.

Double-click on the value for Source A in the GEQ instruction in Rung 5. Change it to 1275.



RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine]

After there is 1275 lbs of water in the Mixing Tank, the system is incremented to Step 2.

When the system is in Batch Step 2, the QR valve is opened until there is 1665 lbs of liquid in the Mixing Tank. The mixture now includes 1275 lbs of City Water and 390 lbs of QR.

Source A Local:2:I.Ch0Data 1275

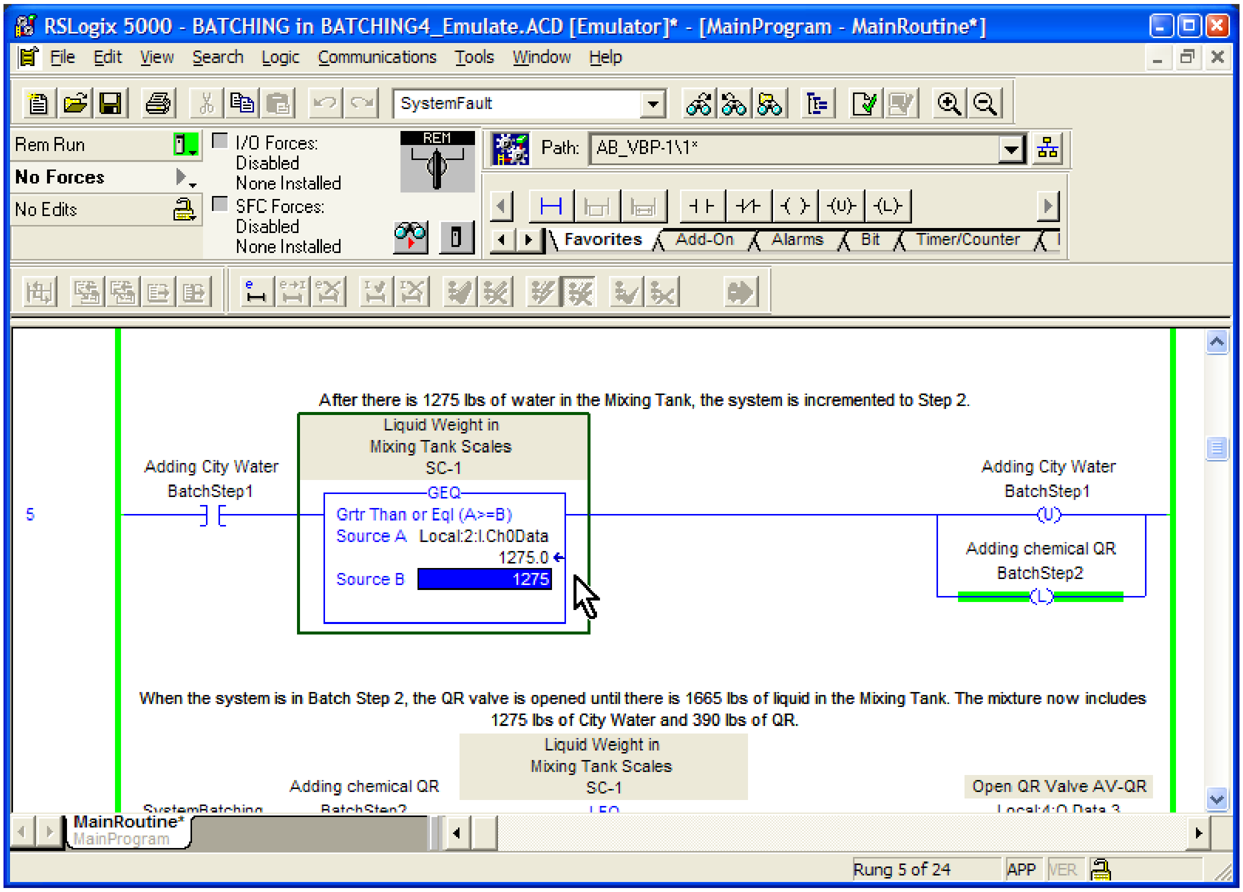
Source B 1275

Rung 5 of 24

Press enter.

The program has incremented to the next step.

You see that the "Adding City Water BatchStep1" output is turned off. The bit "Adding Chemical QR BatchStep2" is turned on.


RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
SystemFault
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter
MainRoutine* MainProgram
Rung 5 of 24
APP VER

After there is 1275 lbs of water in the Mixing Tank, the system is incremented to Step 2.

```
5   Adding City Water      Liquid Weight in             Adding City Water
    BatchStep1             Mixing Tank Scales           BatchStep1
                           SC-1                         -(U)-
                           GEQ
                           Grtr Than or Eql (A>=B)      Adding chemical QR
                           Source A  Local:2:I.Ch0Data  BatchStep2
                                     1275.0             -(L)-
                           Source B  1275
```

When the system is in Batch Step 2, the QR valve is opened until there is 1665 lbs of liquid in the Mixing Tank. The mixture now includes 1275 lbs of City Water and 390 lbs of QR.

```
    SystemBatching   Adding chemical QR   Liquid Weight in      Open QR Valve AV-QR
                     BatchStep2           Mixing Tank Scales    Local:4:O.Data.3
                                          SC-1
                                          LEQ
```

During your testing, you will continue this sequence until the specified weight for each ingredient is reached.

As the ingredients are added, the tank weight rises. When all the ingredients have been added, the batch is blended and sent to the filling lines.

In a real-world start-up scenario, you would verify each step of the process and confirm that the program functions as specified in the original Project Scope.

Editing in Run Mode

As happens many times, though, there is a change in the project procedure.

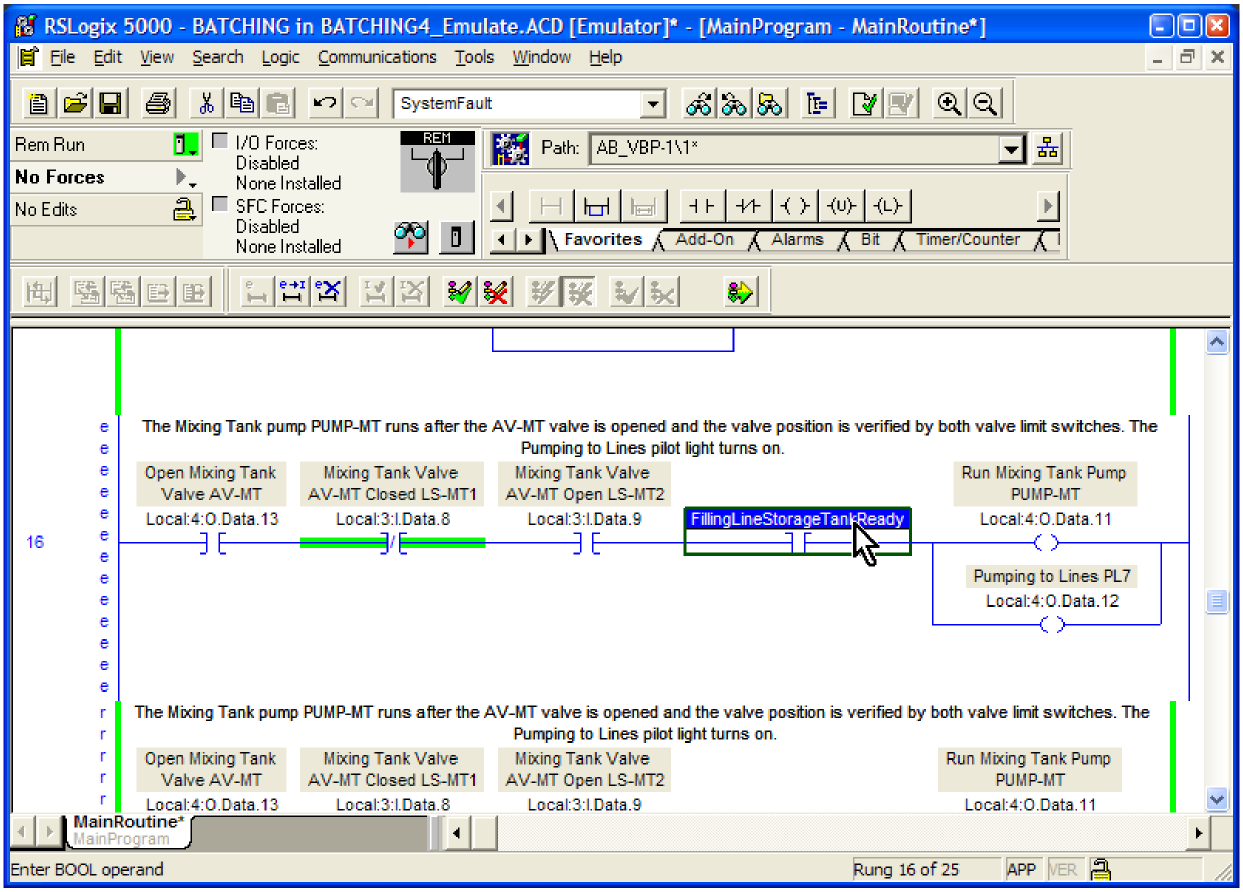
There is a storage tank on the filling lines that Mixing Tank Pump PUMP-MT feeds. The Process Engineer indicates that there may be a possibility of supplying too much finished product to this tank and that it could overflow. He decides the storage tank should have a high level switch mounted in it. If the level in the storage tank gets too high, he wants you to disable the Mixing Tank pump.

We still have spare inputs available. Let's use Local:3:O.Data.14. It is decided that the high level switch will be wired in the failsafe mode; that is, the level switch will be closed until a high level is reach. When wired in that fashion, if the wiring to the switch fails, you will not receive a signal from the storage tank pump and the PLC will stop the Mixing Tank Pump. The system will not run until the problem is corrected.

Scroll to Rung 16. Double-click on the rung number. A column of "i's" will appear. The rung is now open for editing.

Click and drag the XIC tool button to the first marker after Local:3:I.Data.9. When the marker turns green, release the mouse button.

Press enter. You may also click on the tag field. Type the tag name “FillingLineStorageTankReady” and press enter.


RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
SystemFault
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter
16
MainRoutine*
MainProgram
Rung 16 of 25
APP
VER

The Mixing Tank pump PUMP-MT runs after the AV-MT valve is opened and the valve position is verified by both valve limit switches. The Pumping to Lines pilot light turns on.

Open Mixing Tank Valve AV-MT Local:4:O.Data.13

Mixing Tank Valve AV-MT Closed LS-MT1 Local:3:I.Data.8

Mixing Tank Valve AV-MT Open LS-MT2 Local:3:I.Data.9

FillingLineStorageTankReady

Run Mixing Tank Pump PUMP-MT Local:4:O.Data.11

Pumping to Lines PL7 Local:4:O.Data.12

The Mixing Tank pump PUMP-MT runs after the AV-MT valve is opened and the valve position is verified by both valve limit switches. The Pumping to Lines pilot light turns on.

Open Mixing Tank Valve AV-MT Local:4:O.Data.13

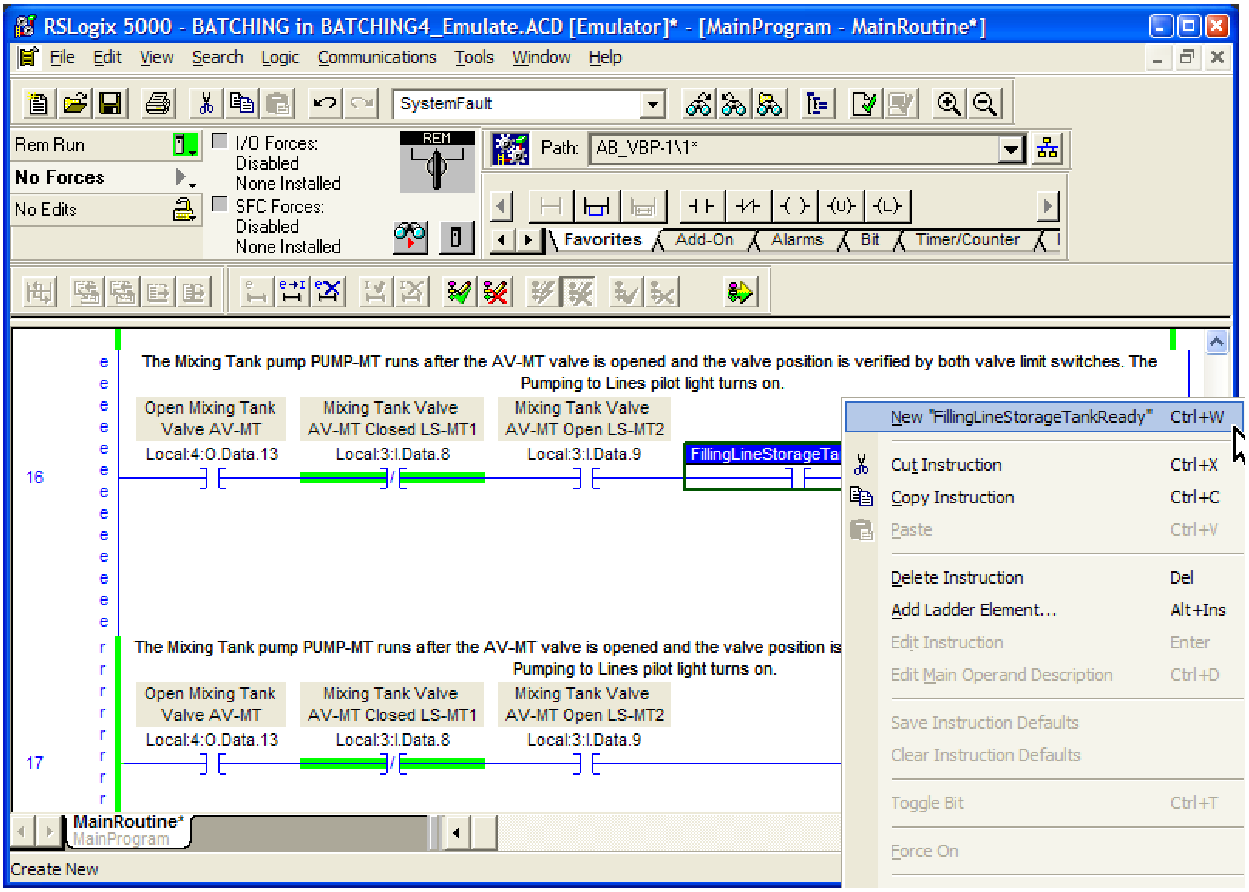
Mixing Tank Valve AV-MT Closed LS-MT1 Local:3:I.Data.8

Mixing Tank Valve AV-MT Open LS-MT2 Local:3:I.Data.9

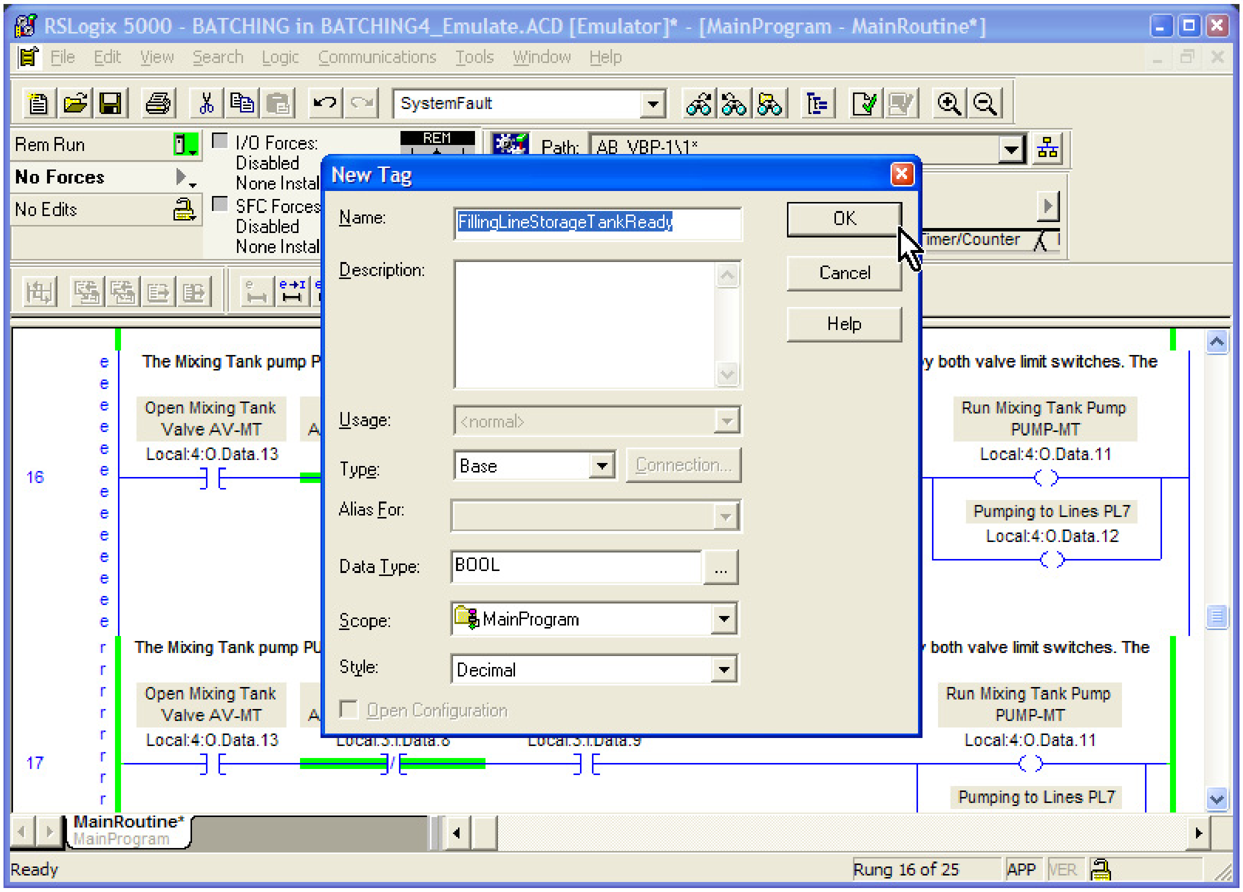
Run Mixing Tank Pump PUMP-MT Local:4:O.Data.11

Enter BOOL operand

Right-click on the new tag and choose New FillingLineStorageTankReady.


RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
SystemFault
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter
The Mixing Tank pump PUMP-MT runs after the AV-MT valve is opened and the valve position is verified by both valve limit switches. The Pumping to Lines pilot light turns on.
16
Open Mixing Tank Valve AV-MT
Local:4:O.Data.13
Mixing Tank Valve AV-MT Closed LS-MT1
Local:3:I.Data.8
Mixing Tank Valve AV-MT Open LS-MT2
Local:3:I.Data.9
FillingLineStorageTa
17
New "FillingLineStorageTankReady" Ctrl+W
Cut Instruction Ctrl+X
Copy Instruction Ctrl+C
Paste Ctrl+V
Delete Instruction Del
Add Ladder Element... Alt+Ins
Edit Instruction Enter
Edit Main Operand Description Ctrl+D
Save Instruction Defaults
Clear Instruction Defaults
Toggle Bit Ctrl+T
Force On
MainRoutine* MainProgram
Create New

A dialog for the new tag opens. In this case, we can accept the defaults and click the “OK” button.



RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]

New Tag

Name: FillingLineStorageTankReady

Description:

Usage: <normal>

Type: Base

Alias For:

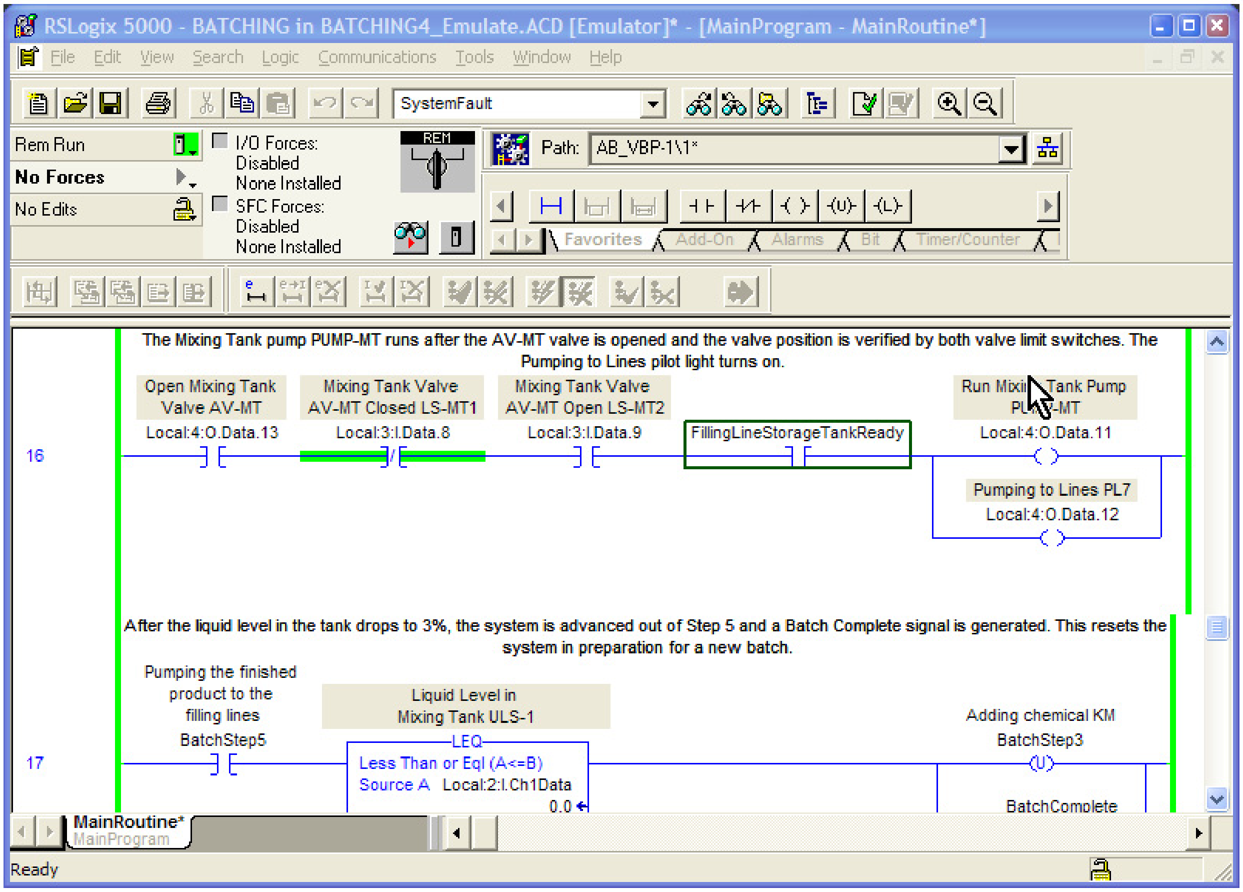
Data Type: BOOL

Scope: MainProgram

Style: Decimal

□ Open Configuration

Click the “Finalize All Edits in Program”.


RSLogix 5000 - BATCHING in BATCHING4_Emulate.ACD [Emulator]* - [MainProgram - MainRoutine*]
File Edit View Search Logic Communications Tools Window Help
SystemFault
Rem Run
No Forces
No Edits
I/O Forces: Disabled None Installed
SFC Forces: Disabled None Installed
REM
Path: AB_VBP-1\1*
Favorites Add-On Alarms Bit Timer/Counter
16
Open Mixing Tank Valve AV-MT
Local:4:O.Data.13
Mixing Tank Valve AV-MT Closed LS-MT1
Local:3:I.Data.8
Mixing Tank Valve AV-MT Open LS-MT2
Local:3:I.Data.9
FillingLineStorageTankReady
Run Mixing Tank Pump PUMP-MT
Local:4:O.Data.11
Pumping to Lines PL7
Local:4:O.Data.12
17
Pumping the finished product to the filling lines
BatchStep5
Liquid Level in Mixing Tank ULS-1
LEQ
Less Than or Eql (A<=B)
Source A Local:2:I.Ch1Data
0.0
Adding chemical KM
BatchStep3
U
BatchComplete
MainRoutine* MainProgram
Ready

The Mixing Tank pump PUMP-MT runs after the AV-MT valve is opened and the valve position is verified by both valve limit switches. The Pumping to Lines pilot light turns on.

After the liquid level in the tank drops to 3%, the system is advanced out of Step 5 and a Batch Complete signal is generated. This resets the system in preparation for a new batch.

Your online edit is now complete. The Mixing Tank pump will not run unless it receives a signal from the Filling Line Storage tank indicating the tank is ready to receive finished product.

If you want to cancel your current edits, click the tool to the immediate right of the “Accept Pending Program Edits” tool. This is called “Cancel Pending Program Edits”.

Run Mode on the Plant Floor

After you have connected to your PLC and downloaded your program, you are ready to begin your testing. However, before you go into Run Mode, **make sure that the E-stop circuit works properly**. I have not included wiring diagrams in this book, but most codes state that all control power is removed if the E-Stop button is pressed.

NFPA 79 Electrical Standard for Industrial Machinery states the following:

> “**9.2.2* Stop Functions.** The three categories of stop functions shall be as follows:
>
> (1) Category 0 is an uncontrolled stop by immediately removing power to the machine actuators . . .
>
> **9.2.5.4.1 Emergency Stop.** Emergency stop functions provided in accordance with 9.2.5.3 shall be designed to be initiated by a single human action.
>
> **9.2.5.4.1.1** In addition to the requirements for stop, the emergency stop shall have the following requirements:
>
> (1) It shall override all other functions and operations in all modes.
>
> (2) Power to the machine actuators, which causes a hazardous condition(s), shall be removed as quickly as possible without creating other hazards . . .
>
> (3) Reset of an emergency stop circuit shall not initiate a restart.”
>
> (National Fire Protection Association, 79-22, 79-23)

Make sure you follow your company’s start-up procedures as you begin testing the system.

⚠ **Test your E-Stop circuit thoroughly**, even if you are working on a system that has been running for months or years. Be ready to hit the E-Stop button if you see a hazardous situation developing.

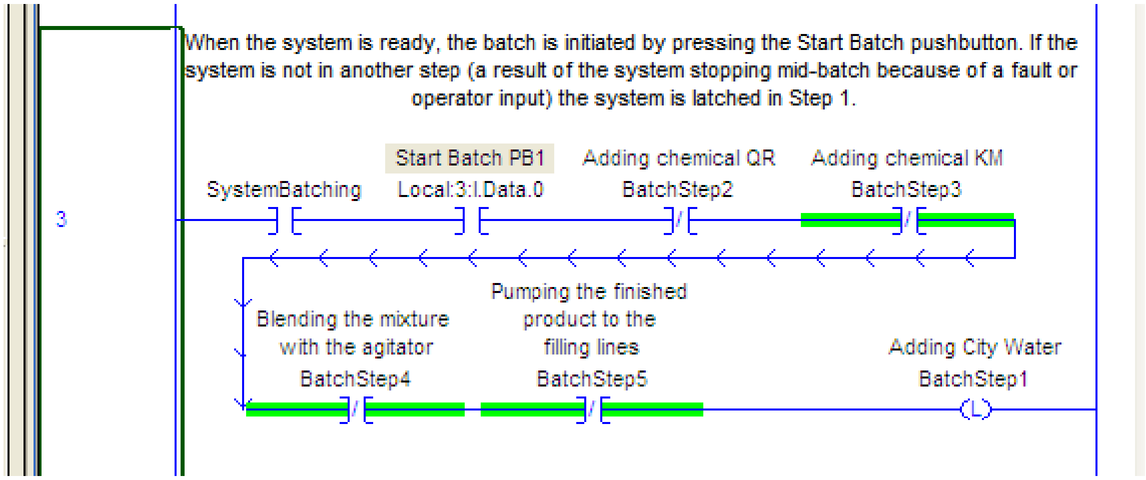
Add-On Instructions & Function Block Diagram Programming

A function block routine consists of a series of sheets. You can navigate to any sheet in the routine, and add and remove sheets as you wish. The function block editor displays only one sheet at a time.

Function blocks are intended to provide a graphical programming interface, like those used in many high-end DCS interfaces.

An Add-On Instruction (AOI) is a custom routine built with function blocks. You can write your own "function" and use it as often as you like.

To show an example of a function block diagram, let's use some ladder logic from our program. We can see that this rung simply performs the Boolean AND function by putting the necessary instructions in series with the output, as shown below.



When the system is ready, the batch is initiated by pressing the Start Batch pushbutton. If the system is not in another step (a result of the system stopping mid-batch because of a fault or operator input) the system is latched in Step 1.

AOIs

Before we go too much further, I need to tell you that there are differing opinions on the use of function block diagrams and add-on instructions in RSLogix 5000. Most programmers do not use them, as they are difficult to troubleshoot and only add value in some situations.

In versions 16.03 and below of RSLogix, you cannot change the programming in an add-on instruction while the system is in run mode. This, as you know, could be a major problem. Until Rockwell addresses this, it will continue to be a deterrent to using AOIs.

If, though, you have a programming scenario where the exact same logic must be repeated many times, it might make sense to create your own AOI.

Take as an example the way we monitored the valves in our program for faults. The fault detection logic is the same for all four valves. We could have developed an AOI that we could “plug in” to each valve.

One of the factors that determine whether ladder logic or an AOI should be used is the number of occurrences of the repeating logic. In our program, we have only 4 valves. It would not be worth the time to write and debug an AOI if we were going to use it only 4 times.

If we had 15 valves, then it would probably be worth the effort. That way, if we decided to change the valve fault detection logic later, we need only change the ladder logic in one place.

In summary, the advantages of AOIs are this:

- AOIs use repeatable, tested code that simplifies programming.

- AOIs make it easy to change the repeating code (if the PLC can be taken offline).

The disadvantages are:

- The ladder logic that comprises an AOI cannot be edited online.

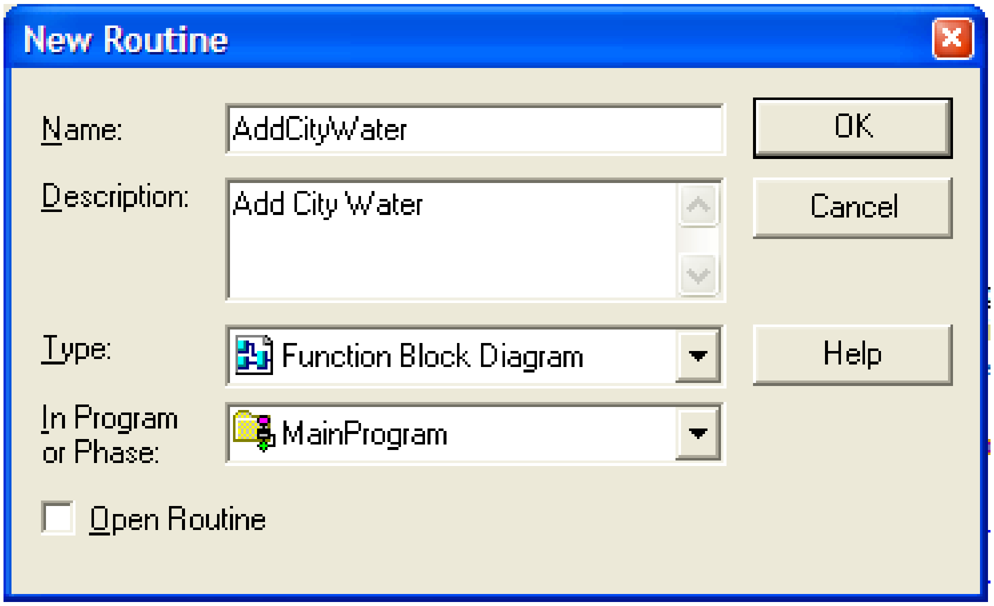
- Function Block Diagrams can become very complex and difficult to follow and troubleshoot in RSLogix 5000.

- Revision history of an AOI may be difficult to track. You may revise an AOI for a machine that is going to Client G that is different in function, but carries the same name, as an AOI that went to Client B a few months ago.

Use your discretion. Having said that, let’s create the function block diagram that could replace Rung 3.

Creating a Function Block Diagram

In the Controller Organizer, right-click on “Main Program” and choose “New Routine”. Fill in the fields as shown below.



New Routine

Name: AddCityWater

Description: Add City Water

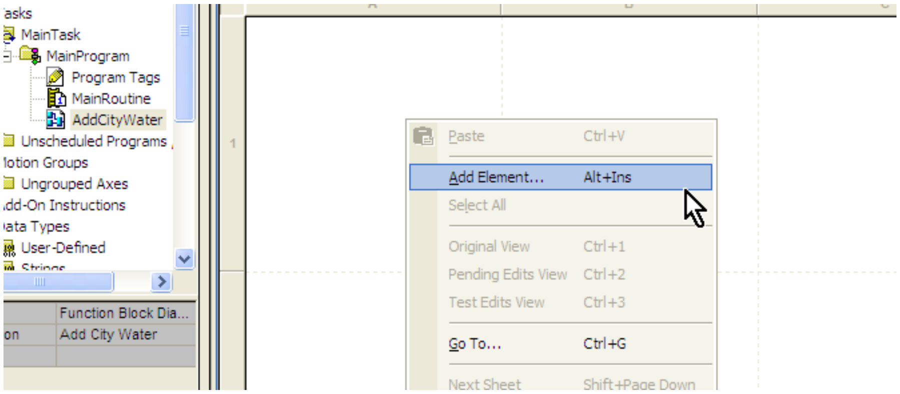
Type: Function Block Diagram

In Program or Phase: MainProgram

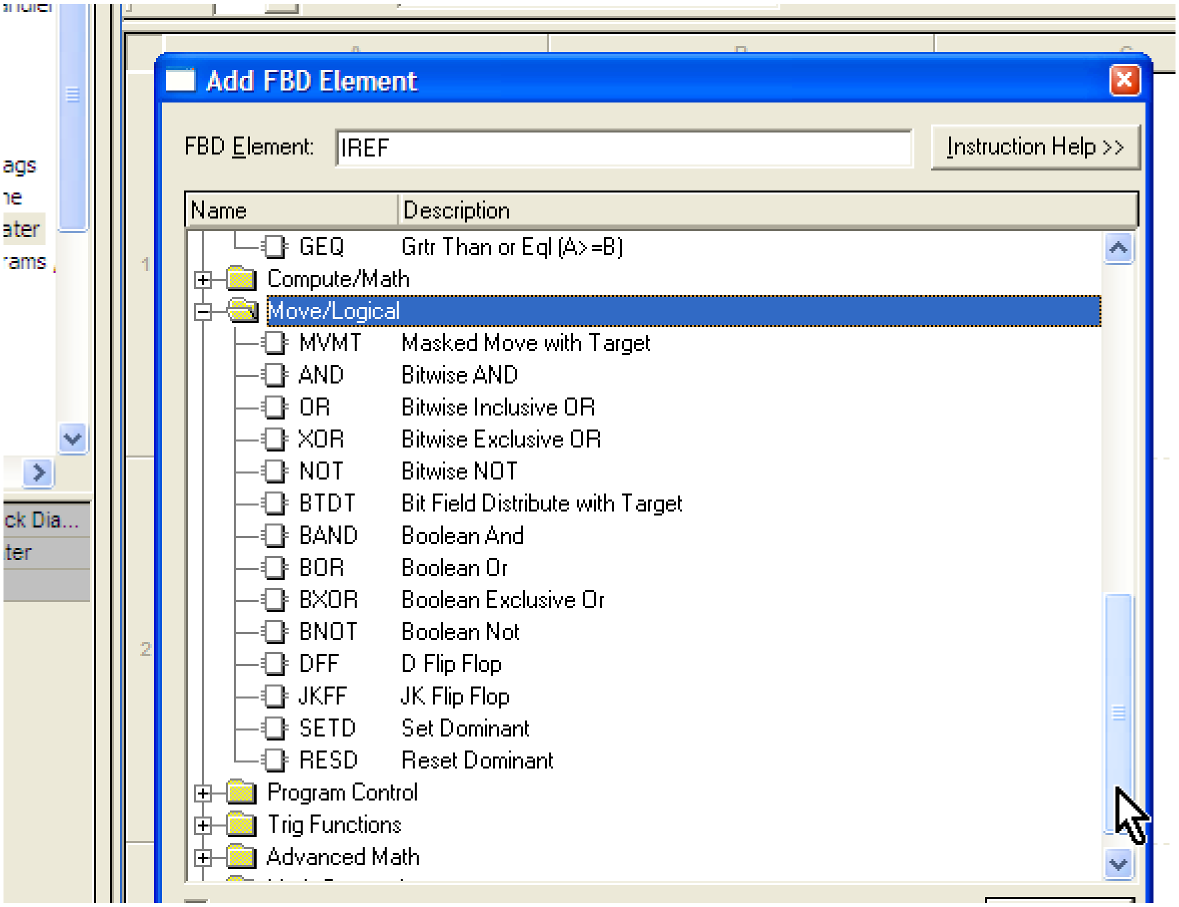
□ Open Routine

Click “OK”.

The “AddCityWater” routine is now shown as a Function Block diagram in the Controller Organizer. Double-click on it. Right-click on the sheet.



Choose “Add Element”. Expand the “Move/Logical” section and choose “BAND Boolean And”.



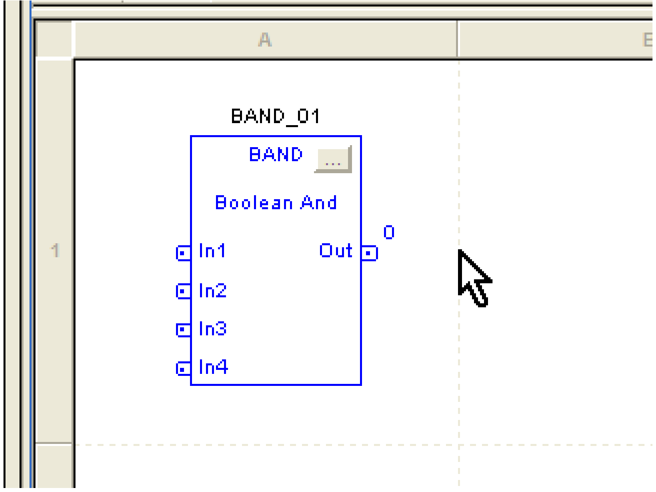
Add FBD Element

FBD Element: IREF

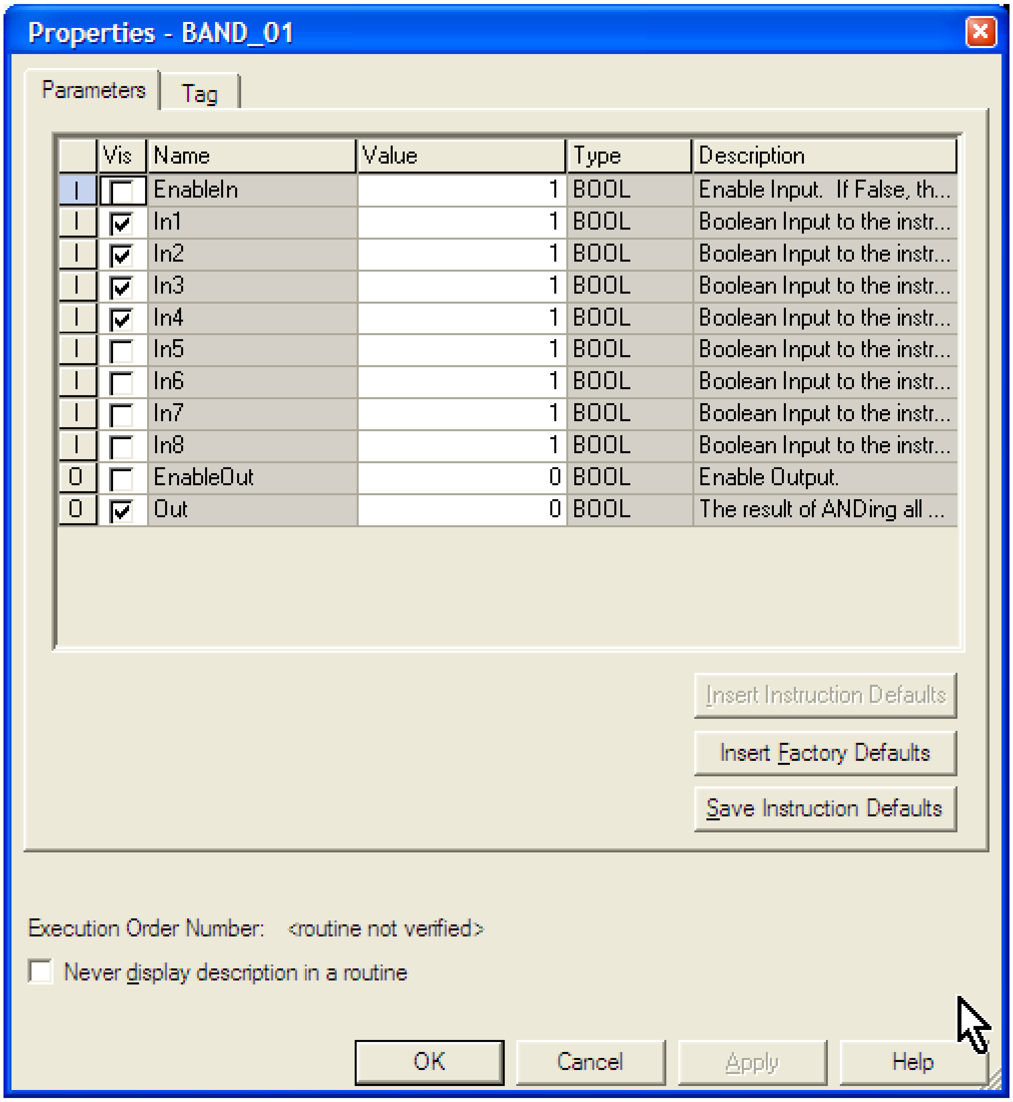
Instruction Help >>

Name	Description
GEQ	Grtr Than or Eql (A>=B)
⊞ Compute/Math	
⊟ Move/Logical	
MVMT	Masked Move with Target
AND	Bitwise AND
OR	Bitwise Inclusive OR
XOR	Bitwise Exclusive OR
NOT	Bitwise NOT
BTDT	Bit Field Distribute with Target
BAND	Boolean And
BOR	Boolean Or
BXOR	Boolean Exclusive Or
BNOT	Boolean Not
DFF	D Flip Flop
JKFF	JK Flip Flop
SETD	Set Dominant
RESD	Reset Dominant
⊞ Program Control	
⊞ Trig Functions	
⊞ Advanced Math	

The BAND function appears.



Click the "…" button in the function. The configuration screen appears.



Properties - BAND_01

Parameters | Tag

	Vis	Name	Value	Type	Description
I	□	EnableIn	1	BOOL	Enable Input. If False, th...
I	☑	In1	1	BOOL	Boolean Input to the instr...
I	☑	In2	1	BOOL	Boolean Input to the instr...
I	☑	In3	1	BOOL	Boolean Input to the instr...
I	☑	In4	1	BOOL	Boolean Input to the instr...
I	□	In5	1	BOOL	Boolean Input to the instr...
I	□	In6	1	BOOL	Boolean Input to the instr...
I	□	In7	1	BOOL	Boolean Input to the instr...
I	□	In8	1	BOOL	Boolean Input to the instr...
O	□	EnableOut	0	BOOL	Enable Output.
O	☑	Out	0	BOOL	The result of ANDing all ...

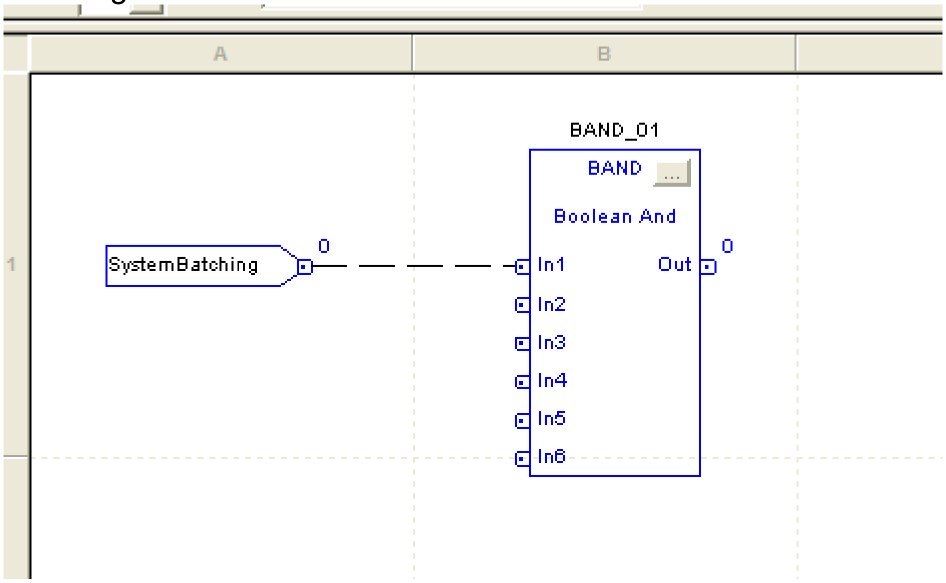
Execution Order Number: <routine not verified>

□ Never display description in a routine

Since there are 6 "input" instructions in the rung we are trying replace, check the "Vis" boxes for In5 and In6. Click "OK".

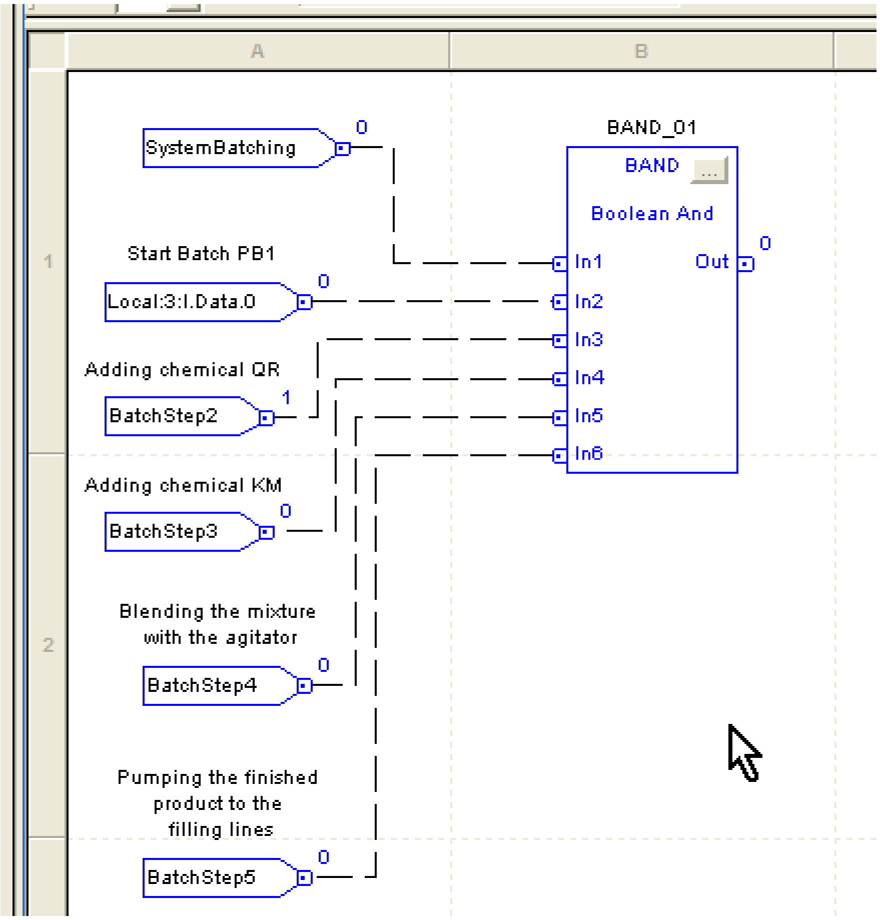
Click the drag the whole function n=block to the right. Right-click on a blank part of the sheet and choose "Add Element". Select "Input Reference". Click on the input block to access the tag list. Choose "SystemBatching", which is the first bit in Rung 3. Click and drag a line from the right side "tip" of the input reference to In1 on the BAND.

The diagram now looks like this.



Add the remaining inputs. Before you connect the lines, notice that some of the inputs need to be inverted to match the logic of Rung 3. Change the “Value” field in the BAND’s configuration to be 0 when the input is on.

The function block diagram looks like this.



We have not addressed the output, or the fact that the output must be latched, but you should now be familiar enough with function block programming to understand the benefits and drawbacks.

A Final Note About Our Program

You may have noticed that we ignored some aspects of the system that perhaps should have been addressed.

For example, in a real world system, you would probably have to make sure that the supply of ingredients QR and KM are available. If not, the pumps could run dry, and this is probably undesirable.

Another nice feature of the program would be to make sure the weight in the tank is changing if we are adding an ingredient. If not, this certainly indicates some kind of problem and the system should probably be shut down.

I have intentionally left out features like this for simplicity's sake. I don't want to overwhelm you with too many "what ifs".

The bottom line is that we fulfilled the Project Scope. *This is the single most important aspect of creating a successful program.*

Feel free to ponder enhancements to this batching logic. Perhaps you can use those ideas on your first real-world batching program.

How Do I . . . ?

Here is an FAQ section to address the most asked questions regarding RSLogix.

How do I . . . ?

Turn off Rung Comments

Tools > Options > Ladder Editor > Display *uncheck* Show Rung Comments

Change The Screen Font

Tools > Options > Ladder Editor > Fonts

Find An Instruction In The Program When You Only Know Some Of The Descriptor Or Rung Comment

Press CTRL-F. Make sure the appropriate right boxes are checked. Type the text in the "Find What:" box. Click "Find Next" or "Find All".

Open Another Program, But Still Keep My Existing Program On Screen

Just start RSLogix again. You can open a number of RSLogix windows simultaneously.

Force An Output On A Module To Come On

Right-click on the output in the ladder logic. Choose "Force On".

In the Forces Disabled/Enabled drop down, choose "I/O Forces > Enable All Forces". **This will force on the output and turn on the device that is connected to the output, no matter what the logic in the rung says.** The corresponding LED on the output module will come on. *Use this with caution.*

Force An Input On A Module To Come On

This differs from forcing on an output in that you can’t force electricity to become present at an input, but the PLC will think there is.

Right-click on the input in the ladder logic. Choose “Force On”. Unlike forcing an output, the corresponding LED in the input module will not come on.

Get Help On A Specific Instruction

Choose Help > Instruction Help

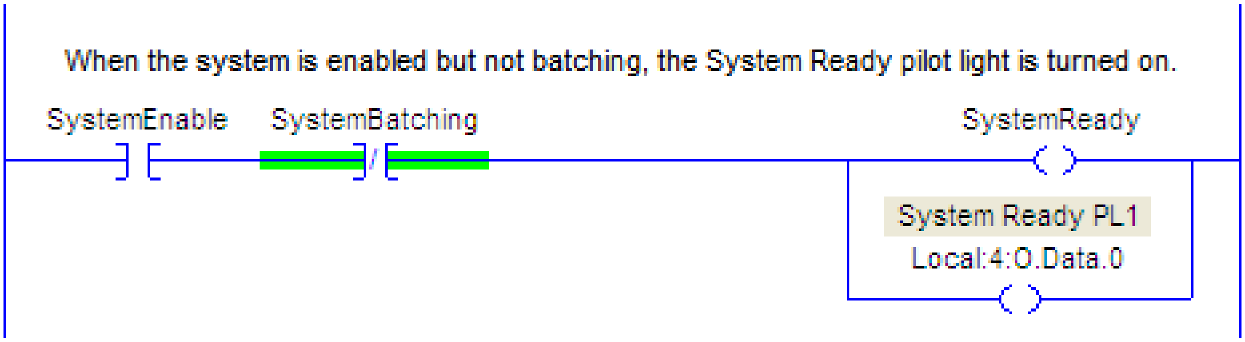
Disconnect From The PLC After All The Programming Changes Are Made

Choose File > Save (save the data tables). Disconnect the interface cable.

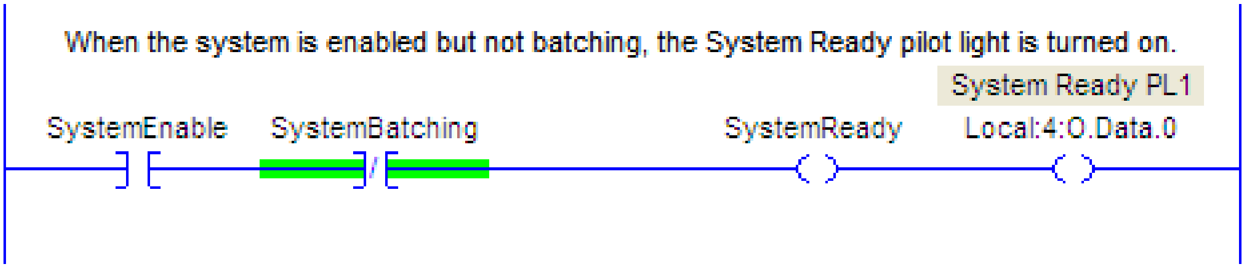
Tips, Shortcuts and Warnings

RSLogix 5000 Allows The Use Of “Serial” Logic That Does Not Conform To Traditional, Electrical Ladder Logic.

For example, both of the rungs shown below are valid in RSLogix 5000.



When the system is enabled but not batching, the System Ready pilot light is turned on.



When the system is enabled but not batching, the System Ready pilot light is turned on.

Clearly, the second version would not work if wired that way in an equivalent electrical circuit. It would not be allowed in RSLogix *500*, either.

The main advantage, in my opinion, to writing the code as it is shown in the second version is that you can get more instructions on the screen, and that involves less scrolling. And, the logic is slightly different; if something turns off the “SystemReady” bit somewhere lese in the program, PL1 would not come on.

The main disadvantage, in my experience, is that the second version will drive electricians and maintenance people crazy, if they are not familiar with RSLogix 5000. Their managers will most likely request that you re-write the rung in “traditional” ladder logic.

The Date Attribute Of The File Is Updated Every Time You Go Online Even If You Don't Save The Program.
Keep this in mind as you track your revisions.

Use Search And Replace With Caution.
Try to avoid the "Replace All" button unless you are absolutely, positively sure that things will come out right. There is no "Undo".

It is much safer to use the "Find Next" and "Replace" buttons. That way you can evaluate each change you make.

Don't Fault The Processor In Run Mode.
Make sure you don't ask the PLC to do anything like dividing by zero. The PLC will fault and the machine will stop. This can be very embarrassing.

Backup Your Files Frequently.
The easiest way is to "rev" your filename every so often, at least once every 30 minutes. Choose File > Save as and put a number in the filename.

Sometimes, files are corrupted through no fault of your own. You can always go back to the previous version so that you have not lost all of your work.

Disable A Rung With An AFI.
If you want to temporarily disable a rung, use the AFI instruction in the first position of the rung.

Don't Turn Off Power To The PLC OR Your Computer During An Online Edit.
There is a good chance this will corrupt your current file.

Keep The Original Program In A Very Safe Place.

If you need to make some changes on an existing program, upload the program from the PLC and store it on a floppy or a CD. If things go awry with your editing, you can always put the program back the way it was.

Keep Track Of What You Are Doing If You Modify An Existing Program.

People will want to know.

Rung Comments Are Extremely Important.

Not only will they explain the operation of the program to someone else, they will remind you of why you programmed the logic the way you did.

Use Passwords Carefully.

If you use one, don’t forget it . . . but if you do, your Rockwell rep can show you how to get around it.

Make Sure Your Program Is Programmed Properly For A System Power-Up.

There must be no machine motion until the operator initiates it.

Set Up A Printout.

Choose File > Print Options. This shows you everything you can print. If you are not careful, though, you may get hundreds of pages, many of which you don’t need. Use the “Print Preview” button to make sure you are getting only what you want.

Use CTRL-G To Goto Places Real Fast.

In the ladder editor, CTRL-G brings up a dialog box. Enter in a rung number or text to go to the first occurrence of the search string.

When You Are Done, Store Your Program To The EEPROM In The PLC.
It is a good idea to save your ladder file to the EEPROM. The onboard battery will keep your program in the PLC's RAM, but if the battery fails, the EEPROM will hold the program.

Use The Built-In Automatic Program Backup
Choose Tools > Options... and set your Project Files Search Path. Make sure AutoSave is enabled. RSLogix will automatically save your file every few minutes (the default is 10 minutes).

Write Your Rung Comments In The Present Tense.
You can then cut and paste them into a document to make an operator's manual.

Write Your Rung Comments In A Standalone Word Processor (such as Word)
Spell check them, then cut and paste them into RSLogix.

Conclusion

I hope that you have found the information in this book useful. Some of the concepts we have covered may seem confusing at first, but with time and effort, you will be able to program a PLC to do whatever you want it to do.

I wish you the best of luck in your endeavors.

THE FOLLOWING ARE TRADEMARKS OF ROCKWELL AUTOMATION, INC.

Allen-Bradley®
ControlLogix™
CompactLogix™
MicroLogix™
PanelView™
RSLinx®
RSLogix™
RSLogix™ 5000
RSView32®
SLC™ 500

THE AUTHOR OR THE PUBLISHER OF THIS BOOK IS IN NO WAY AFFILIATED WITH ROCKWELL AUTOMATION, INC.

Disclaimer

THE AUTHOR INTENDS THIS DOCUMENT TO BE ADVISORY ONLY. ITS USE IN INDUSTRY OR TRADE IS STRICTLY VOLUNTARY.

THIS DOCUMENT IS PROVIDED BY THE VENDOR “AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE VENDOR OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS DOCUMENT, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Courtesy of Rockwell Automation, Inc.