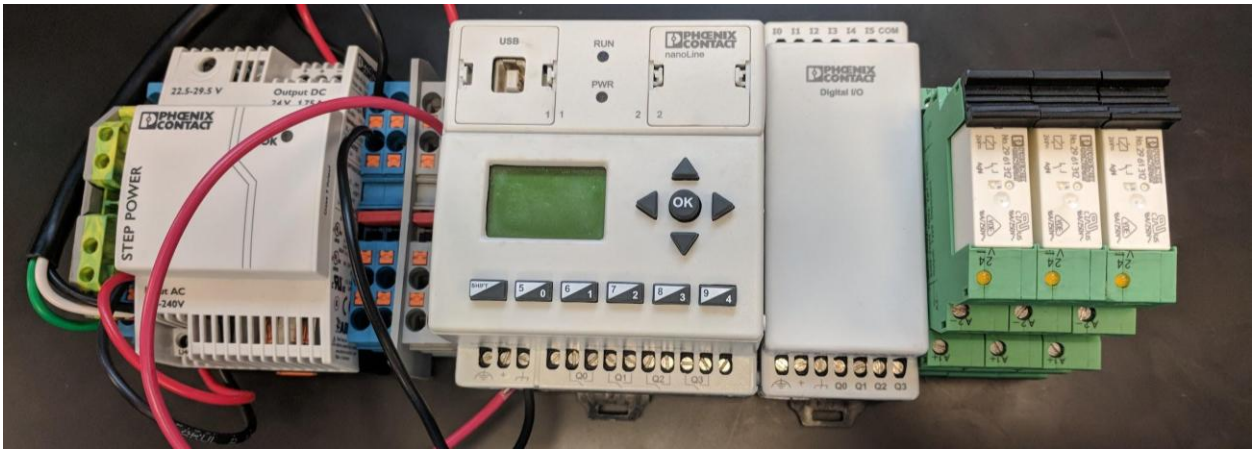
nanoLC Programmable Logic Controller (PLC) Manual

Including nanoNavigator Software

Kevin Carman & Clay Buxton



Datasheets

http://www.digikey.com/catalog/en/partgroup/nanoline-series/50362?mpart=2700453&vendor=277

https://www.digikey.com/catalog/en/partgroup/nanoline-series/71128

http://www.digikey.com/catalog/en/partgroup/phoenix-contact-sockets/19176

https://media.digikey.com/PDF/Data%20Sheets/Phoenix%20Contact%20PDFs/2868664.pdf

https://media.digikey.com/pdf/Data%20Sheets/Phoenix%20Contact%20PDFs/2967604.pdf

More info at: http://www.phoenixcontact.com/nanoline

Purchasing information

Main unit: $155.00

https://www.digikey.com/products/en?mpart=2700453&v=277

Main LCD: $35.00

https://www.digikey.com/product-detail/en/phoenix-contact/2701137/277-2640-ND/2528598

Main USB: $45.50 https://www.digikey.com/product-detail/en/phoenix-contact/2701195/277-2638-ND/2528596

Power Supply: $135

https://www.digikey.com/product-detail/en/phoenix-contact/2868664/277-1989-ND/2051161

I/O unit: $80.00

https://www.digikey.com/products/en?mpart=2701085&v=277

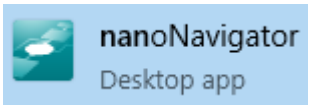
Relay: $41.00

https://www.digikey.com/product-detail/en/phoenix-contact/2967604/277-4921-ND/349737

Total: $491.50

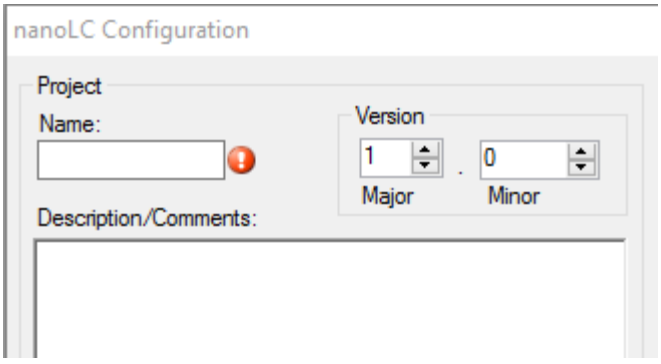
nanoNavigator

Setup

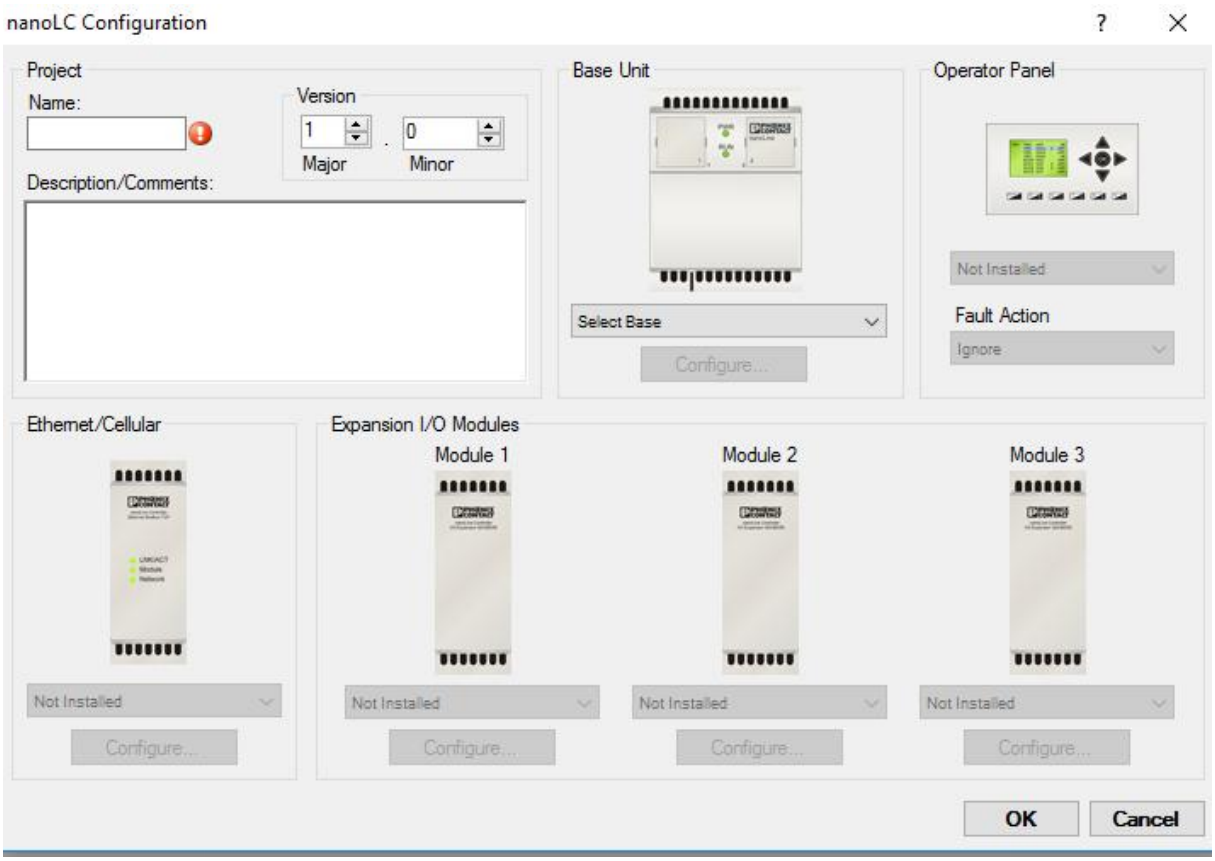


1. Launch the program

2. Run the Connection Wizard

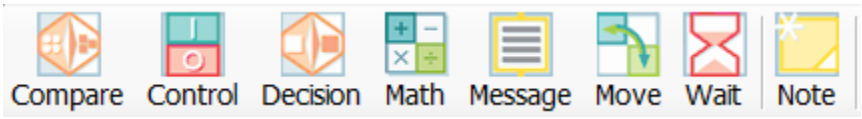


3. Choose a name for your project. Feel free to choose a version number and add a description to your project file.



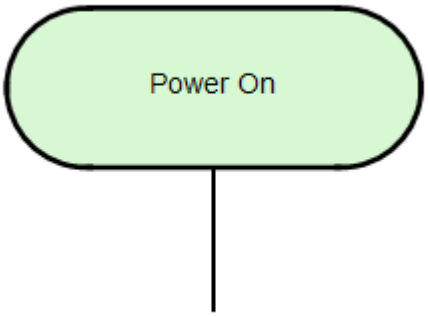
4. The connection wizard should have chosen the correct parts for you. If not, select the correct base unit and select that the operator panel is installed. If you are using any other modules, select the respective units as well.

5. Hit ‘OK’, then ‘Create Flow Chart’ to start your project.

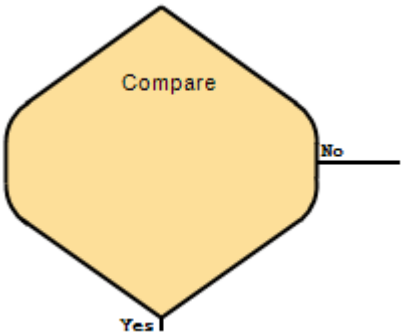


The picture above shows all of the function blocks you can use to create your logic, they will be explained in more depth as we go on.

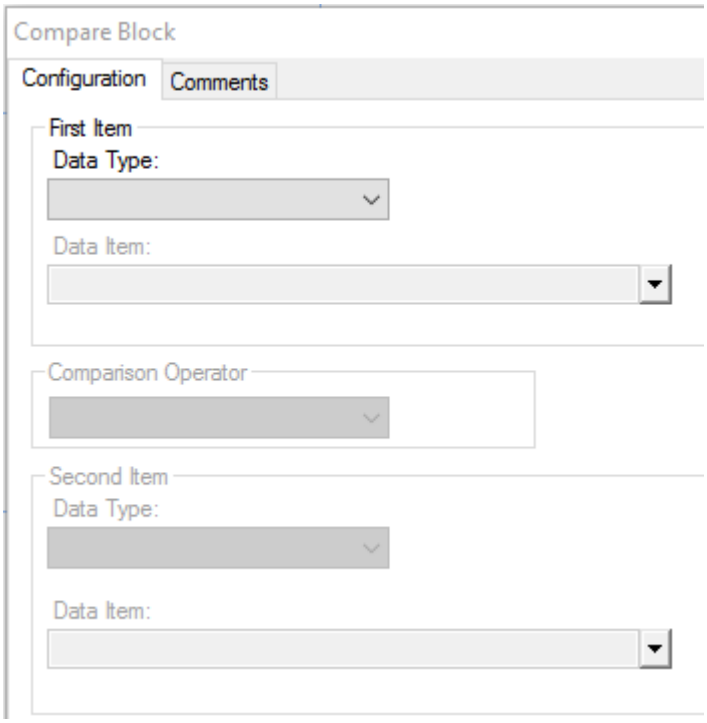
nanoNavigator allows for flowcharts to be 5 columns by x rows, where x can be any positive number. The small amount of columns is a major limitation that you’ll run into very quickly, but at least you can go down.



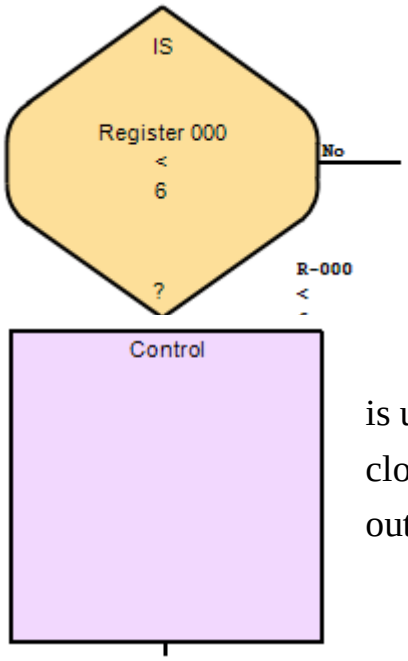
The first function block shown at the top left of your flowchart will always be the power on block. This block is a simple if-statement for if the power is on, then run the program.



Going through the list of available function blocks, the first one is the compare block. Obviously, it compares two items and outputs a signal to either yes or no depending on what the answer is. Place a block and double click on it to edit it's settings.

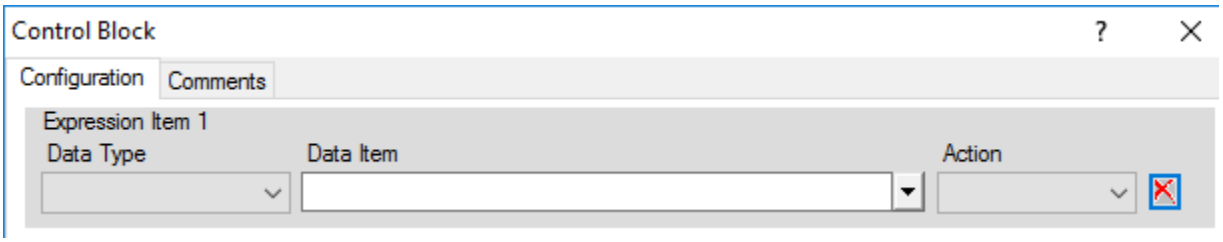


For this example, we want to see if register 000 is < 6. First, click on first item data type, and then click on register. Below that, choose register 000 as your data item. If the register is not initialized beforehand, then you will also need to choose a subtype. In this case, it will be an integer. Next, choose < as the comparison operator. Choose constant as the second data type and match its subtype with that of the registers. Lastly, choose 6 as the constant and hit OK.

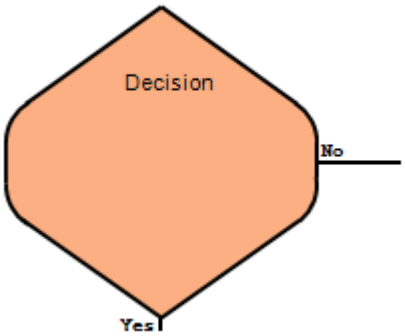


Your compare block should now look like this. You can double check your work by seeing that it says R-000 < 6 in the bottom right hand corner of the block.

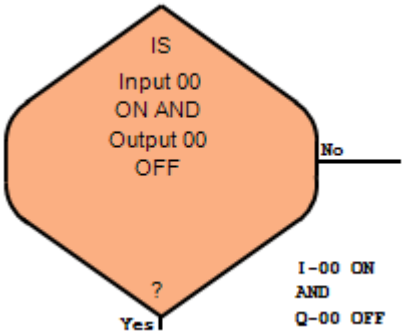
Next, we have the control function block. This block is used for changing binary data items or controlling clocks/counters. This block is often used for turning on/off outputs. Double click on the block to edit it's settings.



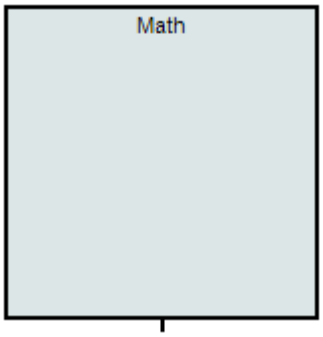
The data type can be seen on the left, which can be a flag/output/timer/counter. Let's increment timer/counter 00. To do this, select timer/counter as your data type. Next, select TC-00 as your data item. Finally, select incr as your action and press ok. Note: The control block can handle up to four controls per block.



The decision functional block acts as an if-statement in your flowchart. Note: It can only only handle up to two data items and the operator can only be AND/OR. Let's make an if-statement that checks if input 00 is ON, AND output 00 is OFF. As we've done for the previous blocks, select the appropriate terms and the output should look like the picture below.

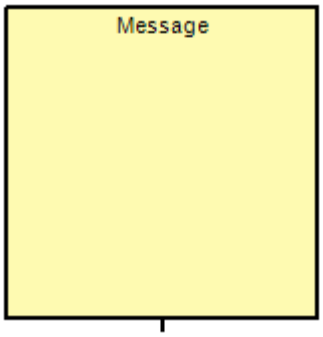


Note that again you can check your work in the bottom right hand corner of the block.



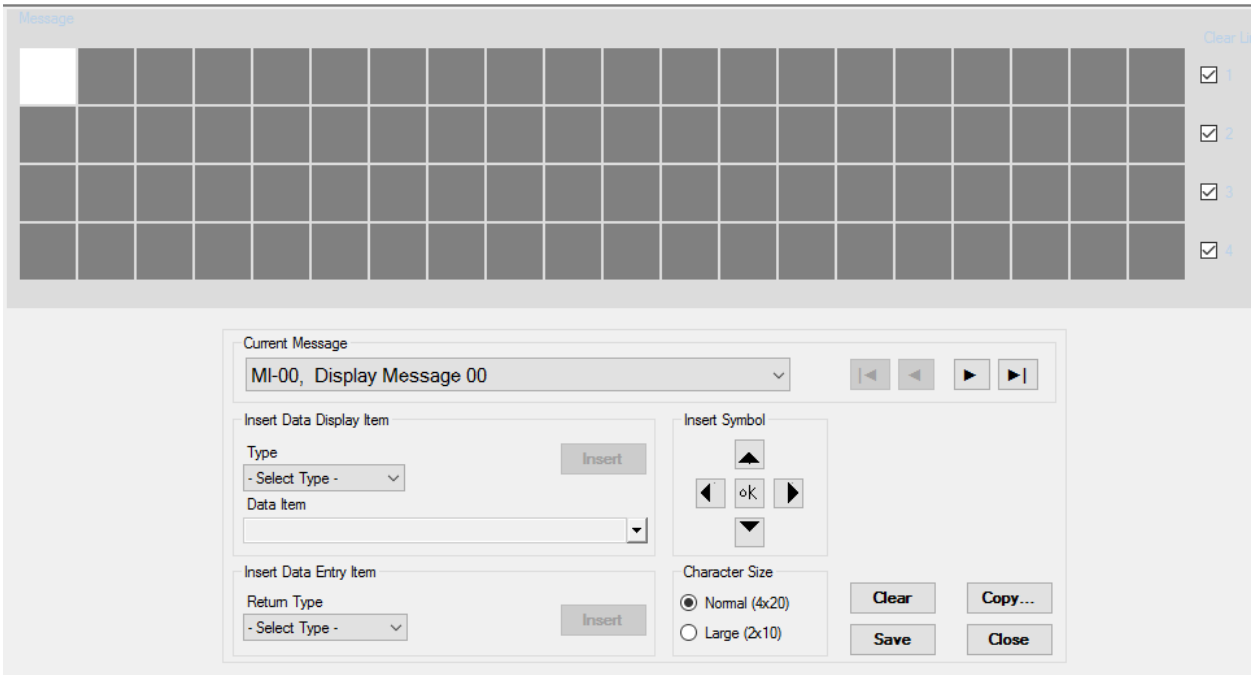
The math block, again obviously, is used for math. The math can be done with a number of data types, but it is primarily done with registers, either register-register or register-constant math. The data of the operation has to be stored somewhere as well, also usually a register.

Here is an example of multiplying register 000 by 20 and storing it in register 001.

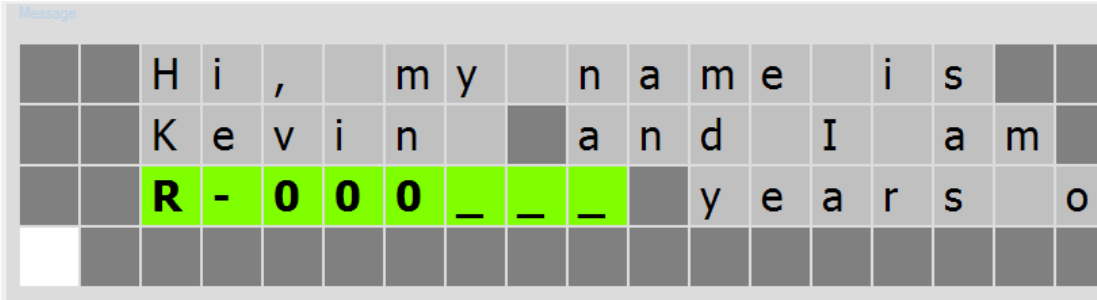


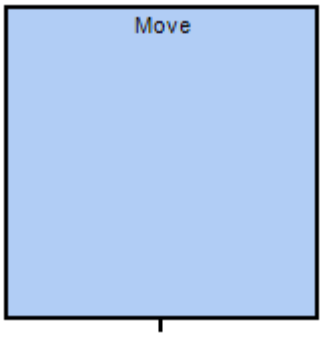
The message block is used to update the message on the screen of the nanoLC. Messages must be made in another menu and selected in the settings of the functional block.

Go to View->Messages in the menu bar.



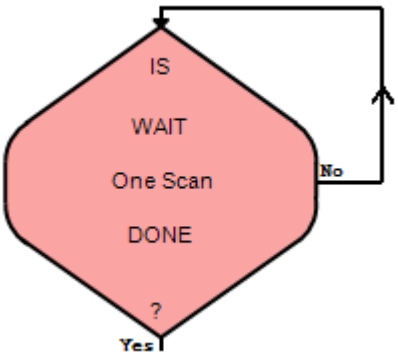
This is the main area where you edit messages for the nanoLC. The grid area is where the message can be typed. It will display on the nanoLC how it appears in this display. Current Message allows you to customize and store many different messages, they can be swapped when they are called from a message functional block. Insert Data Display Item allows you to show the contents of a variable on the screen. For example, if you select output 00, it will show whether it is on or off on the screen. **Don't forget to click save on your message before you close!**





The move block is used for moving the location of data. This is helpful for moving data into registers for storage.

This is a simple example moving the data from register 000 to register 001.



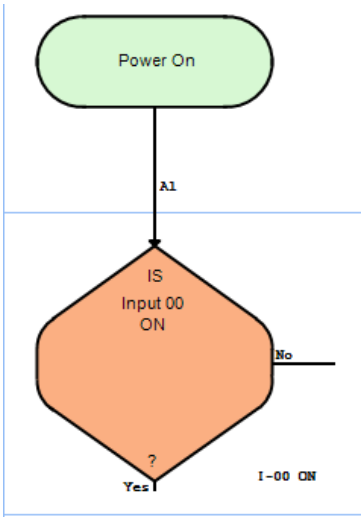
Lastly, the wait block is used for pauses in the flowchart. An example would be for turning on a light bulb for 5 seconds before shutting off. It defaults to one scan wait, but a scan is something less than a second long, so it is often necessary to change it to seconds.

Example Program

One good example video: https://www.youtube.com/watch?v=qXA2O47rqqw

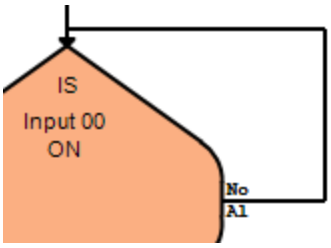
Let's make a simple program that turns on a light bulb for 5 seconds when the user pushes a button. We'll assume that everything is wired correctly to the nanoLC, and just focus on the program.

1. A decision block is needed to check if the button has been pressed or not.
2. Connect the power on to the decision block. This can be done by hovering over the power on, clicking in the circle, and then clicking on the northern edge of the decision block.

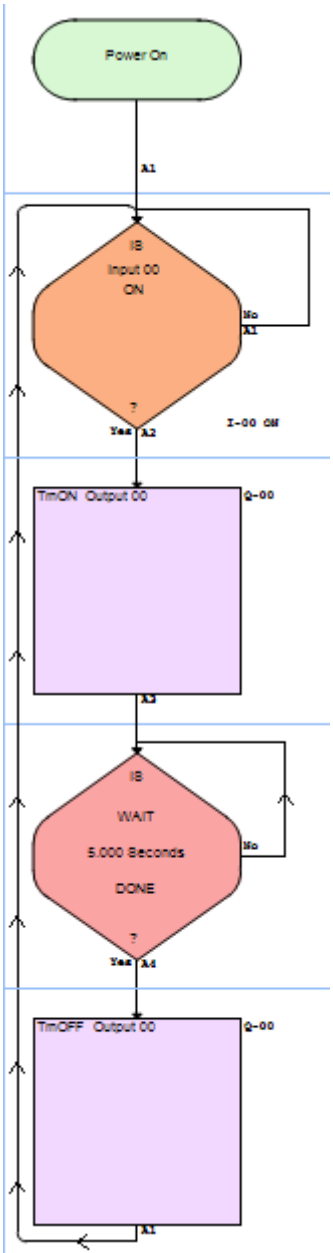


3. Since nothing happens when the button is not pressed, the No leg of the decision block can be connected back to the top of the flowchart.

4. The next block we need is a command block to turn the light on. This block should be configured to turn output 00 ON.
5. The Yes leg of the decision block can then be connected to the top of the control block.

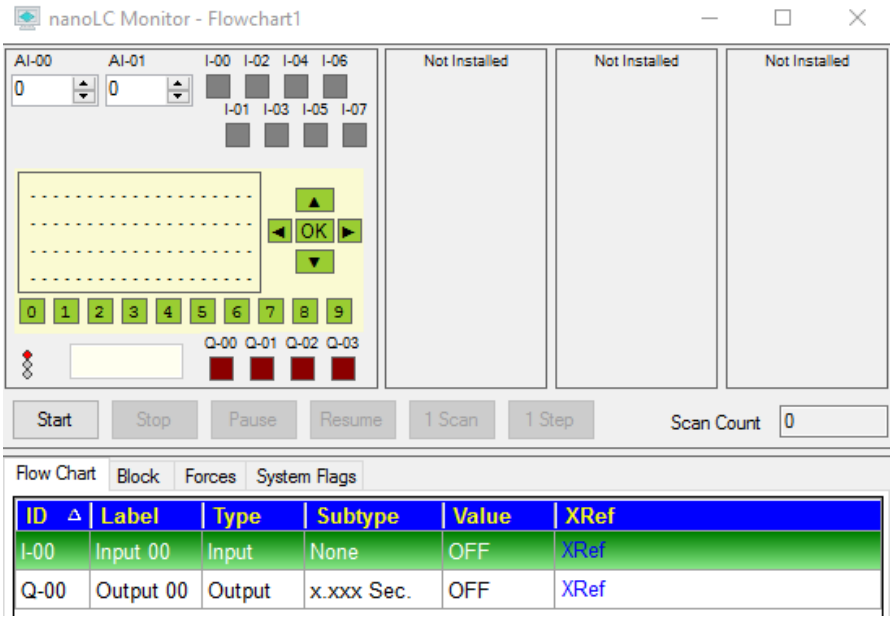


6. Once we turn the light on, we need to wait 5 seconds. So a wait block must be added and adjusted to wait for 5 seconds.
7. The output of the control block can then be connected to the wait block.
8. After 5 seconds is over, we need to turn the light off again. Another control block must be used to turn output 00 OFF. The wait block can be connected to this new control block.
9. Lastly, we need to connect the output of the control block back into the top of the flowchart, so that the program can run continuously. Below is the final flowchart we created.



Simulating

To simulate a project, go to nanoLC->Simulate Project in the menu bar.

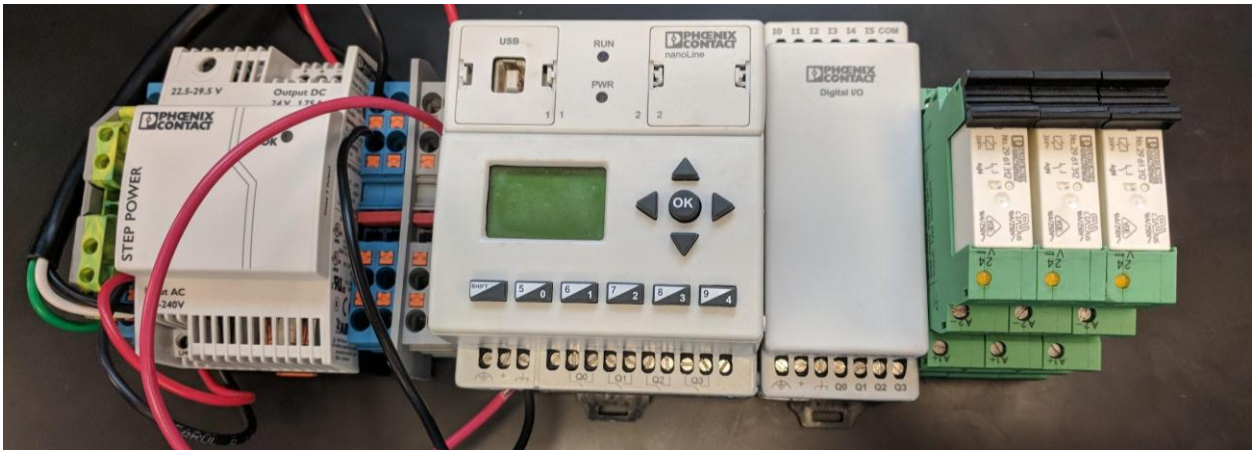


This simulation allows you to run the flowchart as it would be on the actual nanoLC. I-00/I-07 are the inputs and Q-00/Q-03 are the outputs. The flowchart section shows all the I/O used in the current flowchart. By pressing start, the flowchart will be simulated like it would be in real time. The scan count will increment to keep track of runtime. You can click on the I-00/I-07 inputs and they will function as a normal button press. To hold an input, right click. Your flowchart will also update as it progresses.

Downloading

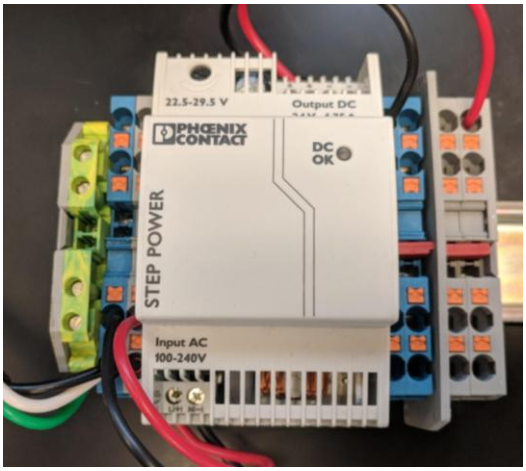
Connect the nanoLC to the computer and in the menu bar nanoLC->Download & Go. The nanoLC should now have the flowchart downloaded and running!

Hardware



The NanoLC has 4 main modules that are used the in early labs. They are the **Power Unit**, **Main Unit**, **Digital IO Unit,** and **Relays**

Power Unit



The power unit is the power supply for the nanoLC and its components. It takes in 120V power from the wall and turns it into the 24V that is used by the NanoLC and many of its modules. The Connection units (Blue and Gray terminals) are used to distribute the signal as shown in the chart below

The two left connection units are used to primarily handle 120V inputs, it is **not** recommended to change the 120V inputs into the unit. The power unit itself has 2, 24V inputs that are used to supply power to the main unit. They are generally connected into the as shown in the chart to provide power to the NanoLC and other modules if necessary.

Main Unit

The Main Unit houses the computational components of the NanoLC. This is where you can control the rest of the modules, and will upload your programs. The unit itself houses a control panel, inputs, and outputs. For some projects this and the power unit will be all that is necessary. Located on the top of the unit is the inputs, and a common ground. On the bottom of the unit is the power connectors, and the outputs.

Powering the Main Unit. The 24VFC input (V in) is a 24V input, GND is ground. FE is not used in the scope of the labs. Both inputs should be delivered from the Power unit. Reference the
Power Unit chart for how those
are used.

Inputs:

U0 - U1 are analog inputs.

COM is a common ground.

I1-I7 are digital inputs.

The inputs can be used for
various purposes in the
software. The Digital inputs
can **only be read at 24V,** interfacing with a lower voltage must be done using relays that give a 24V input to the Main module

Outputs: All of the outputs are digital outputs. The outputs on the NanoLC are relays that can be activated in the software. Each input has 2 terminals, the left terminal is the input voltage for the relay, the right is the output. When the output is turned on in software, the circuit is completed, and the NanoLC outputs the specified voltage.

Digital IO Unit

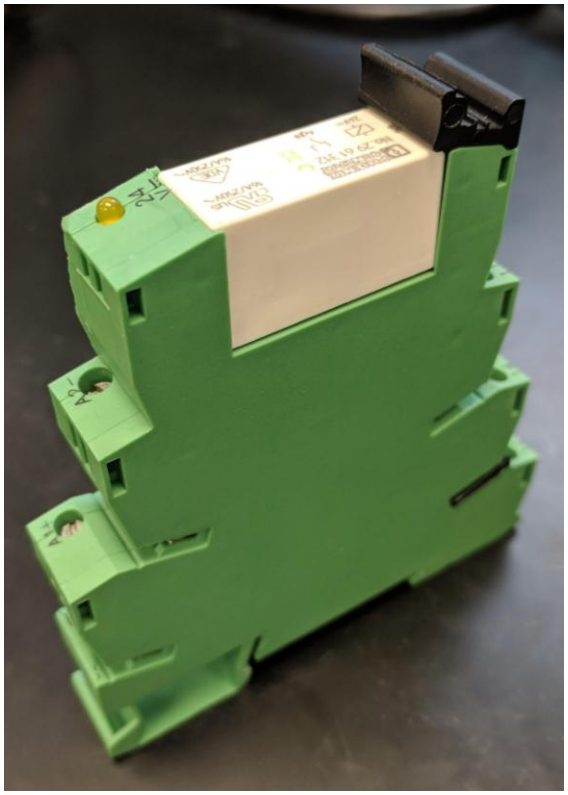
The Digital IO Unit is used to expand the input and output capabilities of the NanoLC. This module includes 4 *sinking* outputs, 6 digital inputs, and a common ground. This connects to the NanoLC using a serial port on the left side of the module that will plug into the Main Unit when attached to the structural bar. Once connected the inputs and outputs will be available in software

Powering the Unit: Similar to the Main Unit, 24V must be supplied to the IO Unit from the power unit in the same way the Main Unit is.

Inputs: 8 digital inputs that will be available at the end of the main unit inputs, if one IO module is used, I0-I5 on the IO module, will be referenced as I8-I13 in software as seen above

Outputs: 4 Sinking outputs. These outputs do not function the same as the outputs on the Main Unit. Sinking outputs when activated will create a ground. They do not output a voltage, but instead will complete a circuit. They are not outputs in the normal sense of the word, but output a control over the circuit.

Relays



The relays used with the NanoLC are 24V activated mechanical relays. The actual relay is the white box mounted in the top, while the green structure is only for mounting and connecting.

Inputs: The only input is a 24V signal that activated the relay

Outputs: The outputs complete a circuit based on if the relay is activated. If the 24V is not activated the circuit is connected between the bottom and the middle

terminals on the output side. If the circuit is activated the circuit is completed between the top and the middle terminals on the output side.

Interfacing with the nanoLC

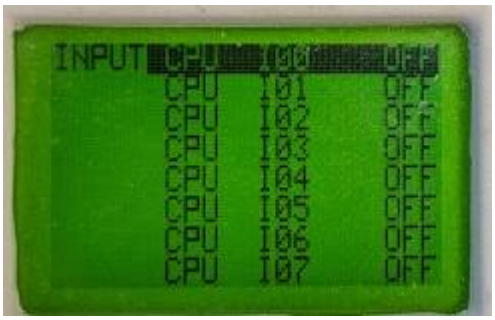
A message will display on the main screen of the nanoLC showing whether the project is running or not.

To change this, press SHIFT+OK to enter the settings.

In the settings menu, you can start/stop projects and check the status of the I/O. By pressing 1, you enter a menu that asks if you want to start or stop execution of the project.

To go back, press SHIFT+↑.

Back in the settings menu, by pressing 0, you enter the area where you can monitor all of your I/O.



By pressing 0, we enter a menu that shows the status of all of the inputs. In this picture, all of the inputs are currently OFF.

By pressing OK on one of the inputs, another menu shows the options to manually force them ON/OFF, which can be very useful for debugging and testing.

Safety

The nanoLC is an industrial microcontroller that is completely capable of manipulating high voltages. There a multiple safety precautions that needs to be followed when using the nanoLC.

1. Leave the NanoLC unplugged as long as possible

While using the nanoLC, turn the power off whenever possible. Leaving the nanoLC off when plugging things in, writing code, or any other development reduces the risk of damaging the nanoLC and the risk of getting shocked.

2. Check your wiring

Before testing your code on the nanoLC it's always best to look over your wiring. It's easy to wire thingsincorrectly, so checking your wires will reduce your risk of damaging the equipment.

3. Color code your wires

We have both red and black wires available in the lab. Use red for positive connections and black for negative connections. Color coding your wires makes it easier to visualize what your nanoLC is doing.

4. Test with lower voltage first

If you are planning on developing a circuit that uses high voltage it may be a smart idea to first test with a much lower voltage. In case something goes wrong using a lower voltage will decrease the chance of damaging the nanoLC and decrease the chance of someone getting shocked.

5. Simulate your code first

Even though you have the ability to upload and run your code at the same time it isn't always wise to do so. To make sure everything works it's wise to simulate your code first.

6. Make sure there are no live wires

Live wires can result in a painful shock. Make sure you don't leave any wires only connected at one end. Also be sure there are no wire fibers exposed. The wire we use for the nanoLCs is frayed and can easily have exposed strands. If you notice any exposed strands be sure to run the wire through a terminal block first. This will reduce the risk of touching a frayed strand.

Following these safety guidelines will keep you safe while working with the nanoLCs.