

# WinTRiLOGI Helps

---

## 1. Introduction

- [Overview of WinTRiLOGI Software Package](#)
- [Ladder Logic Programming Tutorial](#)
- [Using WinTRiLOGI](#)
- [PLC & PC Hardware Setup and Configuration](#)

## 2. TRiLOGI's Main Menu

- [File Menu](#)
- [Edit Menu](#)
- [Controller Menu](#)
- [Simulate Menu](#)
- [Circuit Menu](#)

## 3. Ladder Logic Programmer's Reference

- **Ladder Logic Editor**
    - [Using TRiLOGI Ladder Logic Editor](#)
  - **Ladder Logic Language Reference**
    - [Ladder Logic Fundamentals: Contact & Coils](#)
    - [Special Bits](#)
    - [Special Functions](#)
    - [Using the Sequencers](#)
- 

## Context Sensitive Help Files

These files are displayed when you press <F1> keys at different parts of the WinTRiLOGI programs. These links are listed here for your quick reference.

### 1. TRiLOGI Helps

- [Ladder Circuit Edit Mode](#)
- [I/O Table Entry](#)
- [Simulation of PLC Program](#)
- [Full Screen Monitoring of Target PLC](#)

# Overview and WinTRiLOGI Installation Guide for x86 Compatible PCs running Windows 95, 98, NT or 2000

---

The WinTRiLOGI version 3.5 is developed specially for programming the new \*E10+ and H-series PLCs manufactured by Triangle Research International, It replaces the DOS version of TRiLOGI version 3.3 and 3.3E which are unable to run under the newer 32 bit Windows operating systems such as Windows 2000, NT or XP.

**\*Note:** Due to timing issue during communication, the older generation of E10 PLCs time out faster than Windows O/S is able to respond and hence the WinTRiLOGI program is unable to program them. Please continue to use the DOS TRiLOGI software to program these older generation of E10 PLCs, or upgrade to newer E10 plus PLCs.

## 1. Installing WinTRiLOGI

- a. You should install Java Run Time Environment (**JRE**) Version 1.3.1 on your PC **before** installing the WinTRiLOGI . First, double-click on the file "j2re1\_3\_1-win.exe" to install Java . Please follow all instructions provided by the Install Shield program and install it in the given default path: "[C:\Program Files\JavaSoft\JRE\1.3.1](#)". TRiLOGI needs to install some library files to the JRE default directory later, so it is advisable that you use the default path to avoid problems.
  - b. You may be asked to restart the computer after installing JRE. Follow the instructions so that JRE will be registered in the Windows Registry after restart.
  - c. After you have installed JRE 1.3.1, opens up the CD-ROM's "[x86-Windows](#)" folder and double-click on the "[SetupWTL3.exe](#)" to extract all WinTRiLOGI files into Drive C:.
  - d. All WinTRiLOGI Version 3.5 files will be installed in the following directory: "C:\TRiLOGI\WTL3". You normally would not need to go directly to this directory to run TRiLOGI. This is because during installation of TRiLOGI, a program Group folder "WinTRiLOGI 3.5" will be created in the Start Menu to provide short cuts to the WinTRiLOGI application.
- 

## 2. Using WinTRiLOGI

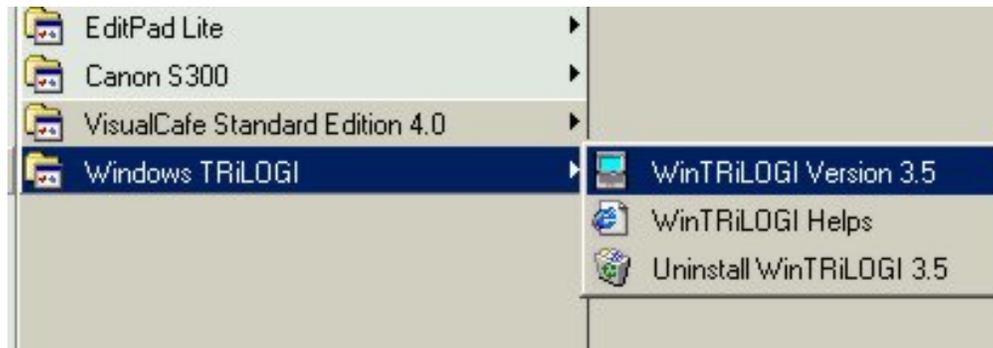
- [Starting The WinTRiLOGI Application](#)

## 3. PLC & PC Hardware Setup and Configuration.

- [Single PLC to One PC Running WinTRiLOGI](#)
- [Multiple PLCs to One PC Running WinTRiLOGI](#)

## Using WinTRiLOGI Software

1. If the WinTRiLOGI and JRE has been properly installed on your PC, you can just click on the "Start" button and select "WinTRiLOGI Version 3.x" from the "Windows TRiLOGI" group, as follow:

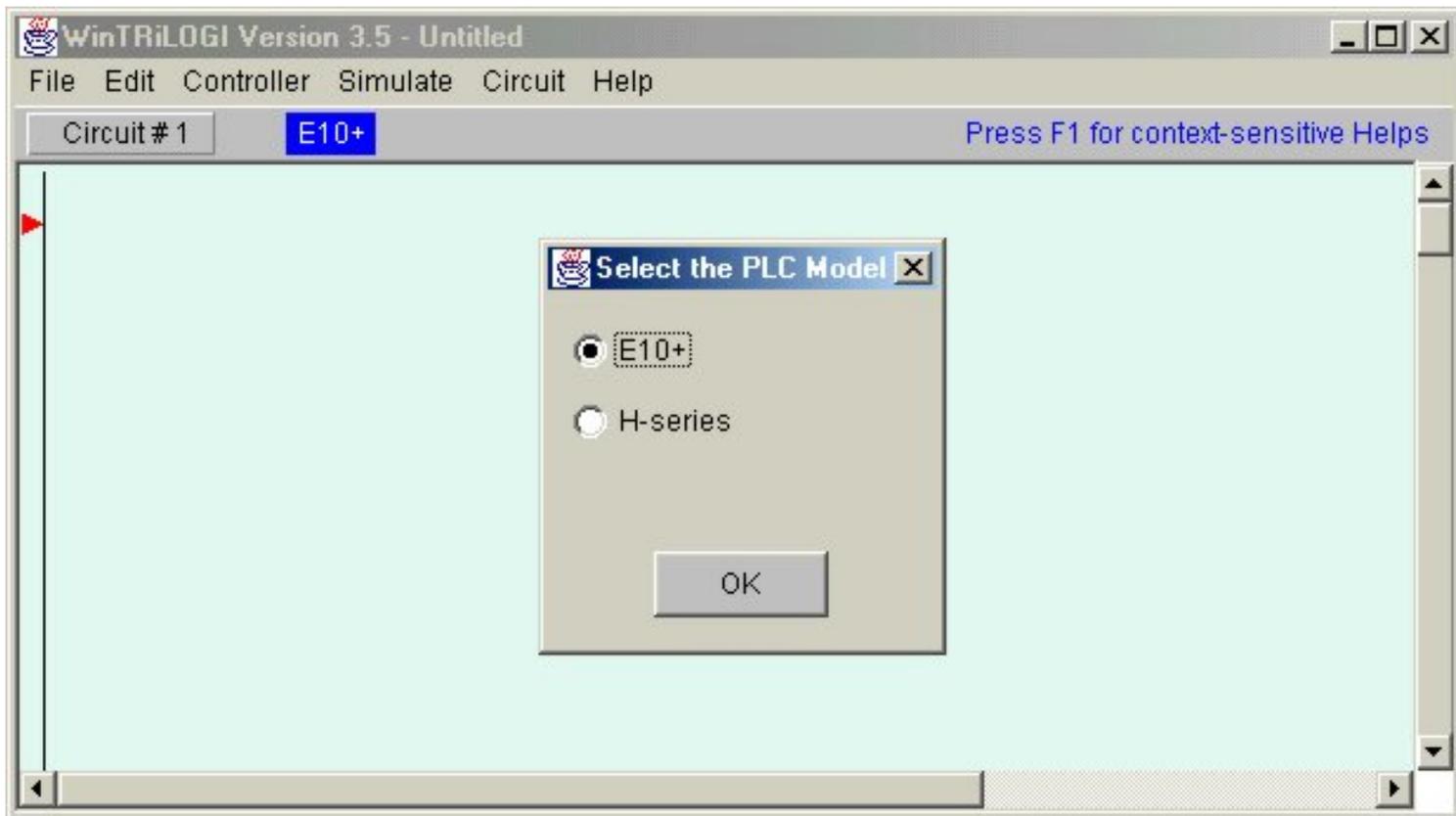


2. You can also open My Computer and open the folder: [C:\TRiLOGI\WTL3\](#), then double click on the file "**WTL35.jar**" to start TRiLOGI application. If JRE has been properly installed the WTL35.jar file will be recognized by Windows to represent executable Java program and it will run immediately.
3. The third alternative is to run the program from DOS command line: First, run the MS-DOS prompt and then navigate to the directory "[C:\TRiLOGI\WTL3](#)" (assuming that is the folder where you've installed TRiLOGI). At the directory, enter the following command line:

```
C:\TRiLOGI\WTL3> java -jar WTL35.jar
```

This method of starting TRiLOGI application has an advantage in that it opens the Java Console window which can be useful because system errors and exceptions are normally reported via the Java Console. This can give a clue to reason of failure.

4. When the WinTRiLOGI program first starts up, you will be asked to select the target PLC family. The selected model is clearly displayed as on the status bar as shown in the following diagram.



The model selection is important because it affects the way the program compiles the source code and the availability of special functions, special bits etc. If you have selected the wrong model you can restart the program and select the right model again.

---

## HELP!!!

When running WinTRiLOGI, you can get on-line help any time by pressing the <F1>. A Help window will open to show you the typical key/mouse actions. You can also click on the <More Help> button to get context-sensitive help loaded into your web-browser. It is assumed that you have Internet Explorer installed in the following directory:

[C:\Program Files\Internet Explorer\IEXPLORE.EXE](#)

However, If your PC does not come with this browser installed, then TRiLOGI Application will report problems opening the web-browser. If that is the case you'll need to use the "Notepad" program to manually edit the "config.tl5" file in the "[C:\TRiLOGI\WTL3\](#)" directory. Modify the first line:

**Browser Path=C:/Program Files/Internet Explorer/IEXPLORE.EXE**

to match the correct browser path info.

---

# PC & PLC Hardware Setup and Configuration.

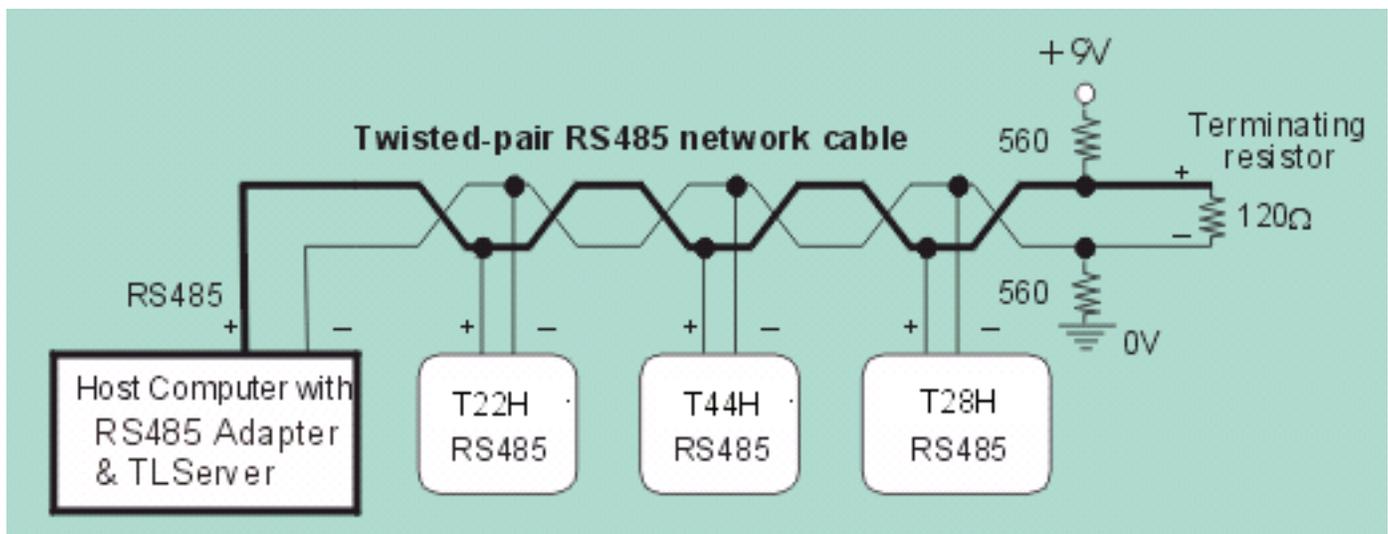
## 1. Single PLC to One PC Running WinTRILOGI

The simplest configuration will be when there is only one PLC and one PC involved. You simply connect the PLC's RS232 port to the any of the RS232 serial port (COM1: to COM4:) of a PC and run the WinTRILOGI software on it. If you use other than COM1: on your PC, you will need to click on the "controller > Serial Port Setup" to select the right COM port to match the communication port number.

## 2. Multiple PLCs to One PC Running WinTRILOGI

You can connect multiple H-series PLCs to a single PC by connecting every PLC's RS485 in a daisy-chain manner to the PC's RS232 port. You do need to purchase a **RS232-to-RS485 converter** (we recommend the [Auto485](#) adapter) to connect the PC's RS232 port to the RS485 network. Please refer to the PLC's User Manual for details on installation issues regarding electrical specifications and termination requirements when connecting the PLCs in an RS485 network.

The WinTRILOGI can access to any of the PLCs on the RS485 network just by specifying the ID address of the PLC concerned. Up to 32 Standard PLCs can be networked to a PC. If you replace the RS485 driver IC by a 1/8 power type you can link up to 256 PLCs to a single PC for programming and monitoring.

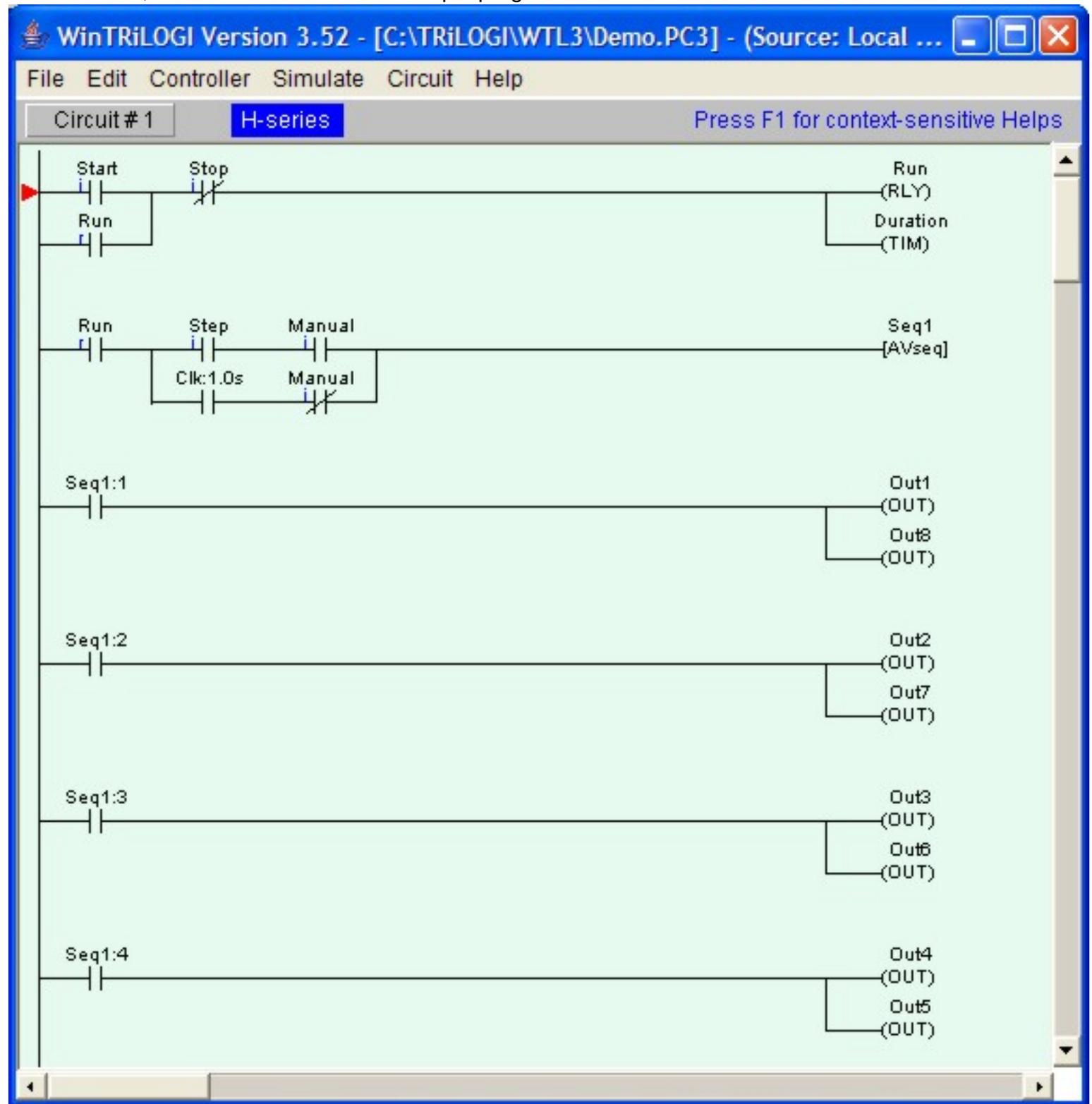


## Ladder Logic Programming Tutorial



### Your Assignment: Creating Your First Ladder Logic Program

In this tutorial, we would like to create a simple program as shown below:



Simply follow the steps below to create your first ladder logic circuit.

- Open pull-down "File" menu and select "New".
  - You should now be in the "Browse" mode of the logic editor. The vertical line on the left end of the screen is the "power" line. The cursor is at the position where you can key in your very first ladder logic.
- 

**Goto to Step 1**



## Ladder Logic Programming Tutorial STEP1



Before we commence the circuit creation, let us define the I/Os to be used for this program. The following I/Os are required:

**Inputs** : Start, Stop, Manual, Step

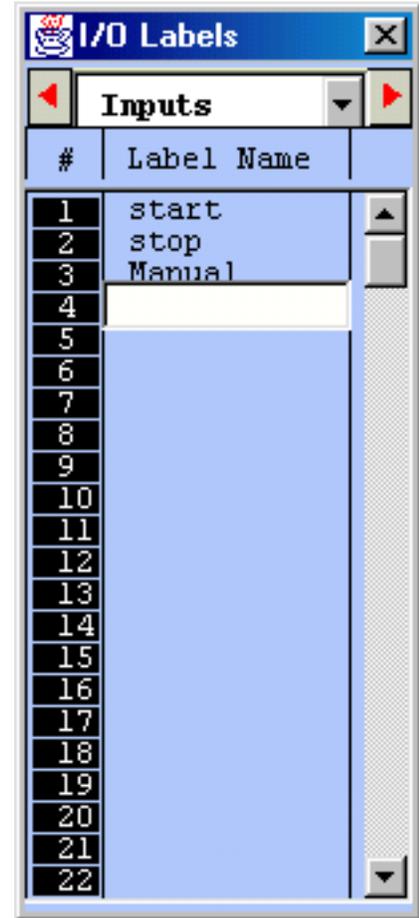
**Outputs** : Out1, Out2,.... Out8

**Relays** : Run

**Timers** : Duration

**Sequencer** : Seq1

1. Open up the I/O label editing Window by pressing <F2>. (Although you can also click on "Edit" menu and select the item "I/O Table" to achieve the same, we strongly recommend learning the hot key F2 as it is often much more convenient to use).
2. Scroll to the "Inputs" window by using the left/right cursor keys or by clicking on the red color left/right arrow buttons or simply select it from the choice box between the left/right arrow buttons.
3. Move the deep blue color highlight bar to Input #1 position by clicking on it. Click again to open up a text field for entering the name for Input #1.



4. Enter the name "Start" for Input #1. Press <Enter> key to accept the name. The text field will be closed and the name "Start" is now assigned to Input #1. If you made a mistake, simply press the "spacebar" or click on the input location again to edit it.
5. Press <Enter> key again and the highlight bar will be moved to Input #2.
6. Without using the mouse button, simply start typing the name "Stop" at Input #2. The text field will be automatically opened up at Input #2 for entry. Press <Enter> after typing in the name for "Stop" input.
7. Complete entry of the other two input label names "Manual" and "Step" as above. Note that if you enter more than 10 characters in the text field, only the first 10 characters are accepted. Also, white spaces between names are not acceptable and will be automatically converted to the underscore character ( '\_' ). e.g. If you enter the name: "M series PLC" for an I/O, it will be accepted as "M\_series\_P".
8. After entering label names for Inputs #1 to #4, move to the "Output" table by pressing the right cursor key or by clicking on the right arrow button. Enter all the output and relay label names in their respective I/O tables. We will discuss the "Timer" table in the next step.

### Important Notes

- a. You can **shift** the Items in the I/O table up or down or insert a new label between two adjacent, pre-defined labels. Simply press the <Ins> key or **Right-Click** the mouse button to pop up the "Shift I/O" menu which allows you to shift the selected I/O. However, please note that if you shift the I/O down, the last entry in the I/O table (e.g. Input #256) will be lost.
- b. WinTRiLOGI allows I/O label names of up to 10 characters. However, if you wish to keep compatibility with Version 3.x you should use no more than 8 characters to define the I/O names.

Back to Assignment



Go To STEP2

## Ladder Logic Programming Tutorial STEP 2



1. Timer table has an extra column "Set Value" located to the right of the "Label Name" column.
2. After you have entered the label name "Duration" for Timer #1, a text entry box is opened up at the "Set Value" location of Timer #1 for you to enter the SV for the timer. SV range is between 0 and 9999. Enter the value 1000 at this location.
3. For a normal timer with 0.1s time base, the value 1000 represents 100.0 seconds, which means that the "Duration" timer will time-out after 100.0 seconds.
4. We are now left to define the sequencer, "Seq1". The sequencer is an extremely useful device for implementing sequencing logic found in many automated equipment. TRiLOGI supports 8 sequencers of 32 steps each. Each sequencer requires a "Step counter" to keep track of the current step sequence.

#	Label Name	Set Value
1	Duration	1000
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		

5. The first 8 counters in the counter table double as the step counters for the 8 sequencers. These sequencers must be named "Seq1" to "Seq8" if they are to be used, i.e. Counter #1 to be named as "Seq1", Counter #2 as "Seq2", etc. However, any counter not used as sequencer may assume any other name (up to a maximum of 10 characters) if they are used as ordinary counters.

If you are at the "Timers" table, pressing the right cursor key again will bring up the "Counters" table. Enter the name: "Seq1" at the label column for Counter #1. Press <Enter> and the text entry field will be opened at the "Set Value" column. For now, let's enter a preset value of "4" for "Seq1".

6. We have now completed defining the I/Os, timers and counters. Press the <ESC> key to close the counter or other tables. Note that not all labels need to be defined before programming. You may create the label names any timer during circuit creation by pressing hotkeys <F2>.

Back to Assignment

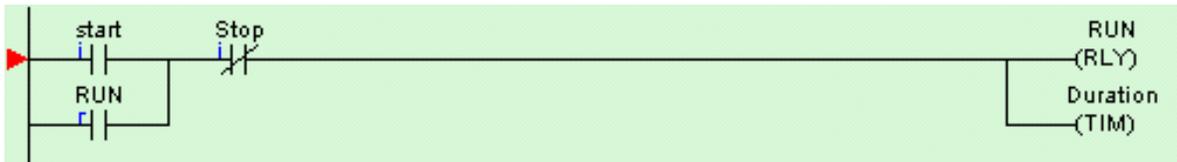


Go To STEP3

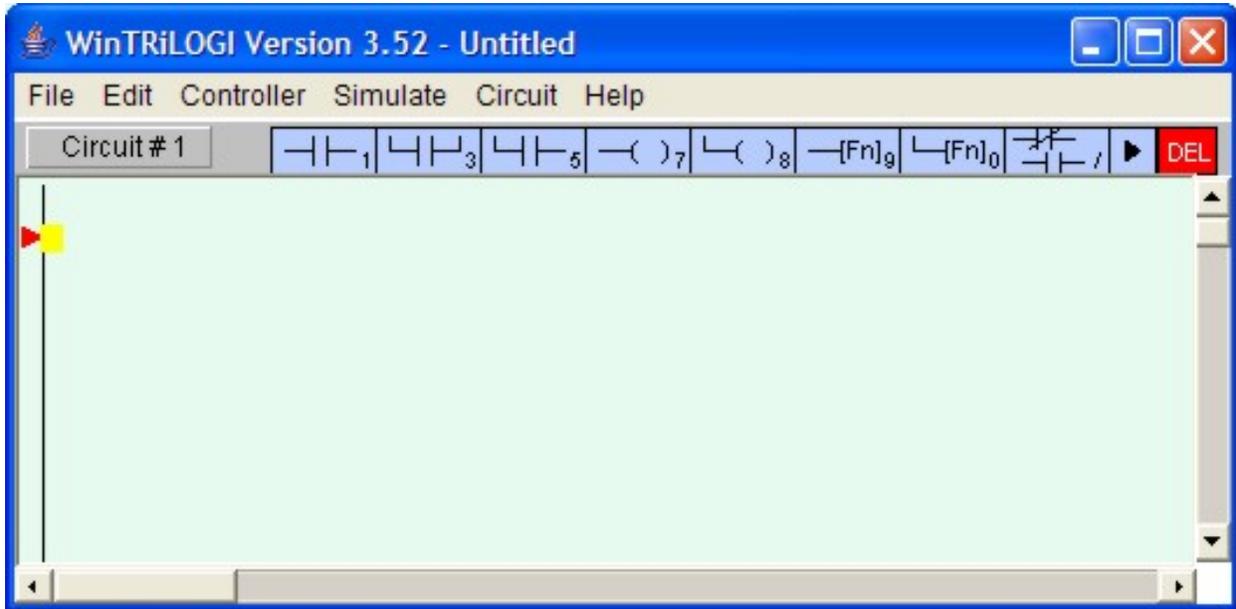
## Ladder Logic Programming Tutorial STEP 3



We are ready to create Circuit #1 as shown below:



1. With the circuit pointer (red color triangle) at Circuit #1, press the <Spacebar> to enter the "Ladder Edit" mode. You can also enter the circuit edit mode by double clicking at Circuit #1.



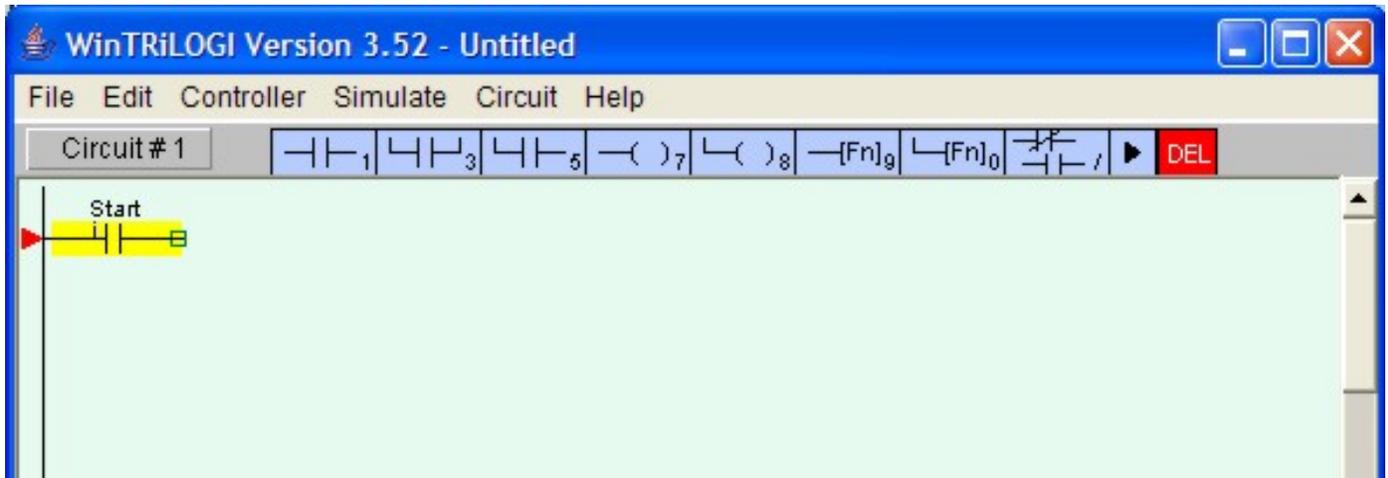
Once you enter the "Ladder Edit" mode, a row of ladder icons appear along the top of the main TRILOGI window just below the pull down menu. The following is a description of each item. A yellow color highlight bar, which you can move to select an element in the ladder circuit, will appear.

	<1> - Left click to insert a normally-open series contact. <2> - Right click to insert a normally-closed series contact.
	<3> - Left click to insert a N.O. parallel contact to highlighted element <4> - Right click to insert a N.C. parallel contact to highlighted element
	<5> - Left click to insert a N.O. parallel contact to enclose one or more elements. <6> - Right click to insert a N.C. parallel contact to enclose one or more elements.
	<7> - Insert a normal coil which may be an output, relay, timer or counter.
	<8> - Insert a parallel output coil (not an entire branch) to the current coil.
	<9> - Insert a special function coil which includes execution of CusFn
	<0> - Insert a parallel special function coil to the current coil.
	</> - Invert the element from N.O. to N.C. or from N.C. to N.O.
	Click to move the highlight bar to the right (same effect as pressing the right arrow key). This can be used to move cursor to a junction which cannot be selected by mouse click.
	<b>Double-click</b> to delete a highlighted element.

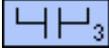
- Now insert the first element by left-clicking on the  icon. The icon will change to a bright yellow color to show you the element type that you are creating. At the same time, an I/O table should appear on the screen with a light beige-color background instead of the normal light blue background. The I/O table now acts like a pop-up menu for you to pick any of the pre-defined label names for this contact.

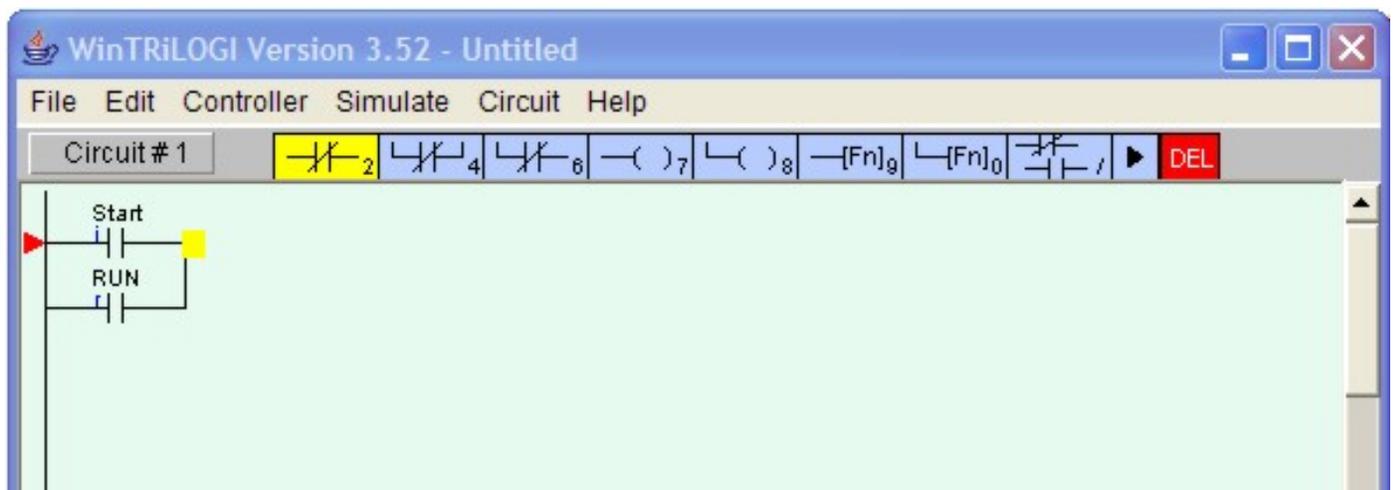
**Note:** Starting from TRiLOGI Version 3.52, if you pick any undefined I/O you will be prompted to enter the label name and what you entered will automatically be updated in the I/O table.

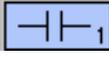
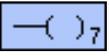
The contents in the table are not normally meant to be edited at this moment. Scroll to the "Input" table and click on the label name "Start" and a normally-open contact will be created at Circuit #1.



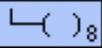
If you observe the highlight bar carefully, you will notice a dark green color square at the right end of the highlight. This indicates the insertion location where a series contact will be attached. You can change the insertion location to the left or the right of the highlight bar by pressing the <SHIFT> key once.

- Next, create the contact "RUN" which is parallel to the "Start" contact by left-clicking on the  icon. The I/O table will appear again. Scroll to the "Relay" table and select the "RUN" relay.
- To insert the normally-closed "Stop" contact in series with the "Start" and "Run" contacts, you need to move the highlight bar to the junction of the "Start" and "Run" contact. First click on the "Start" contact to select it. Then click on the  icon to move the highlight bar to the junction, as follow:



- Next, **right-click** on the  icon. It will change into yellow color normally-closed contact as shown in the above diagram. You are now inserting a normally-closed series contact at the location of the highlight bar. Pick the "Stop" label from the "Input" table to add the series contact.
- We will now connect a relay coil "Run" to the right of the "Stop" contact. Click on the  icon to insert the coil. Select "RUN" label from the "Relay" table. Remember that an input can never be used as a coil. Fortunately, TRiLOGI is smart enough not to call up the "Inputs" table when you are connecting a coil, to avoid unintentional errors.

Notice that the coil symbol ---(RLY) indicates that this is a relay coil, which is helpful in identifying the function of the coil. TRiLOGI automatically places the coil at the extreme right end of the screen and completes the connection with an extended wire.

7. Right below the relay coil is a parallel timer coil with label name "Duration". To create this coil, click on the  icon. This allows you to connect a parallel coil to the existing coil. The "I/O" table will pop up for selection again. Since we want to choose a timer, scroll to the "Timer" table and pick the first timer with the label "Duration" to complete the circuit.

**Press the <Enter> key once to complete Circuit #1**

Congratulation! You have just successfully created you very own ladder logic circuit. It is that simple!

---

Back To Step 2



Go To Step 4

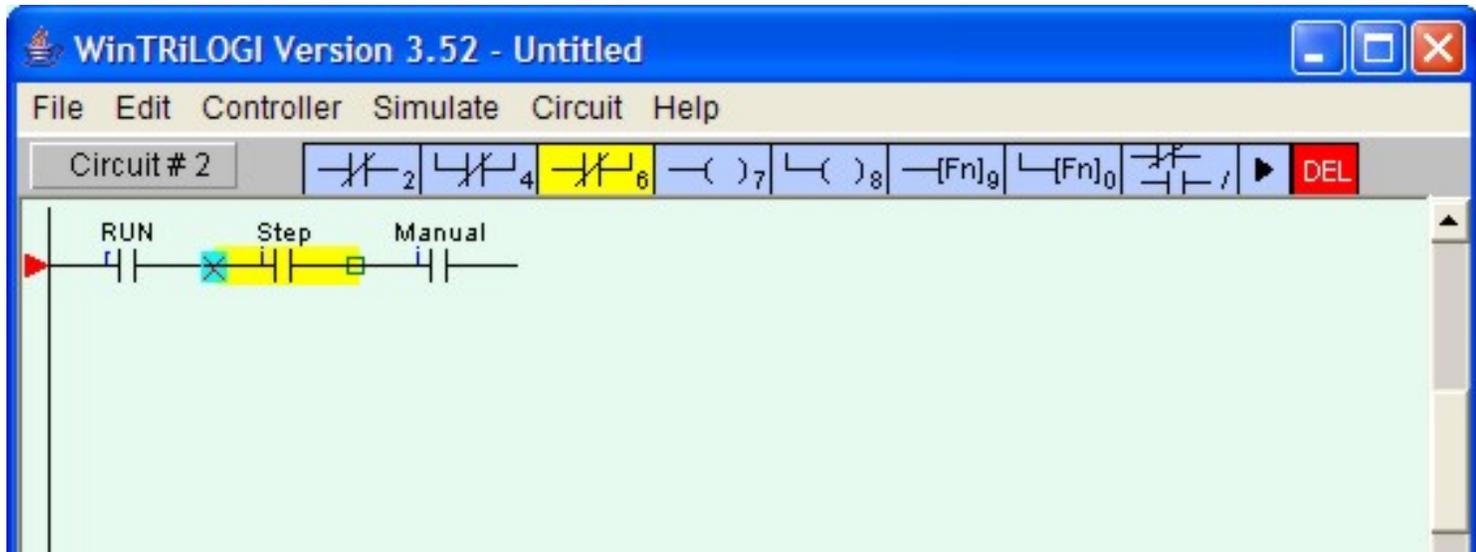
## Ladder Logic Programming Tutorial STEP 4

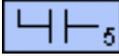
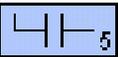


We will now create Circuit #2 as shown below.

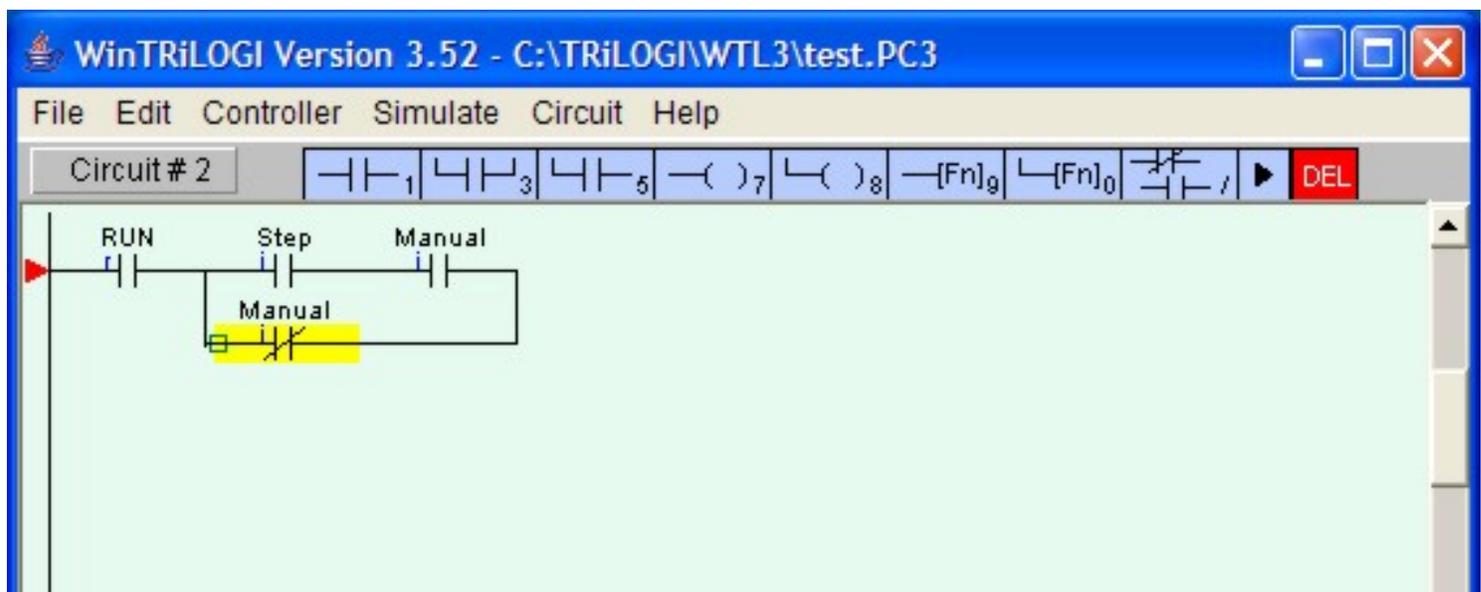


1. Follow the steps listed in STEP 3 to create the following circuit fragment:

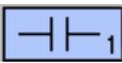


2. We want to enclose the two series contacts "Step" and "Manual" with a parallel branch that contains two elements. First, we will create the branch for the N.C. "Manual" contact.
3. Click on the element "Step" to highlight it. Then **right-click** on the  icon to create a N.C. parallel circuit that encloses both the "Step" and the "Manual" contacts. A cross will appear at the left hand end of the "Step" contact, indicating that this is the starting location of the parallel circuit. You should now click on the "Manual" contact to select the ending location for the parallel circuit. The yellow highlight bar will be positioned at "Manual" contact now.
4. You will notice that the  icon has now changed into a yellow color N.C. contact  with an opposite connection arm. You should now click on the  symbol to close the parallel branch. (One possible short-cut method is to double-click at the ending location to close the branch).

As usual an I/O table will be opened for you to select the I/O. For now, select the "Manual" label from the "input" table to create the following circuit:

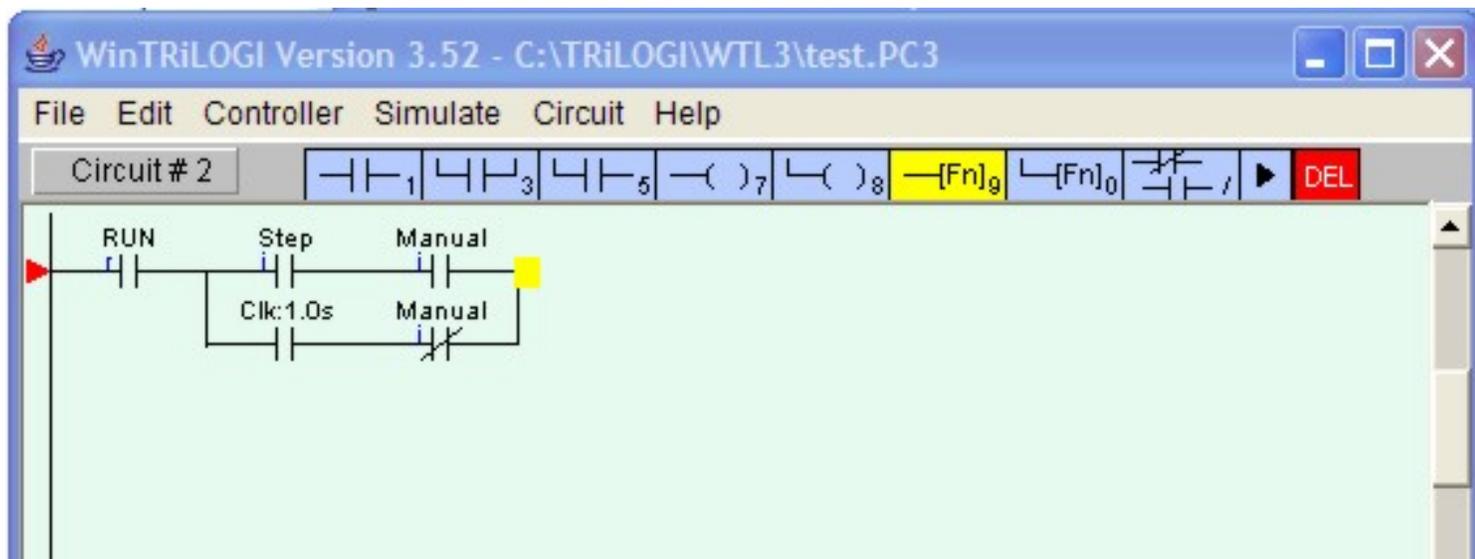


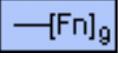
#	Label Name
1	SeqN:x
2	Normally ON
3	1st.Scan
4	0.01s Clock
5	0.02s Clock
6	-
7	0.1s Clock
8	0.2s Clock
9	-
10	1.0s Clock
11	1 min Clock
12	RTC Error

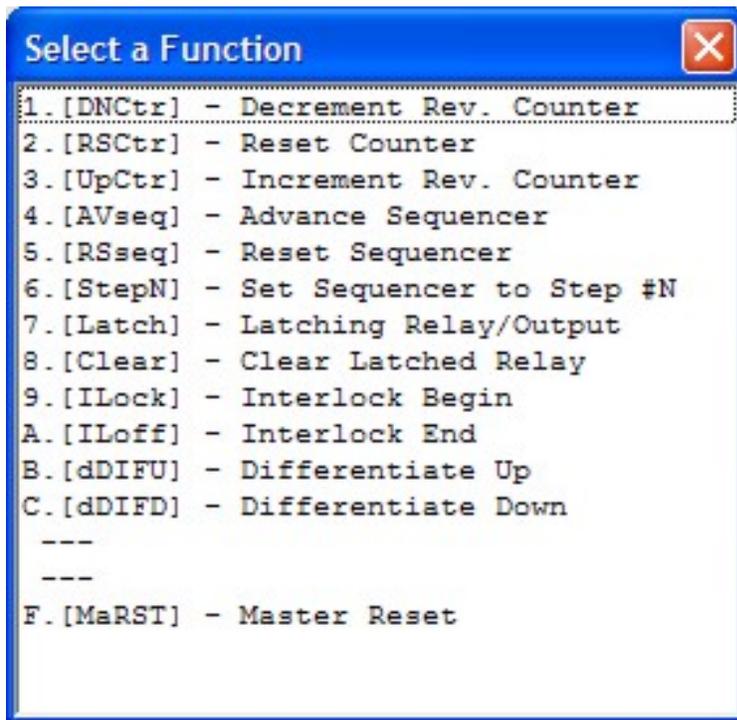
5. Next, we want to insert the special bit "Clk:1.0s" to the left of the "Manual" contact. Press the <SHIFT> key to move the insertion point to the left end of the highlight bar as shown above. Then left-click on the  icon to create a normally-open contact. Scroll the I/O table to the "Special Bits" table and select the item: "1.0s Clock". The parallel branch would have been completed by now.

**Note:** The "Special Bit" table comprises some clock pulses and some other special purpose bits. These include the 6 built-in clock pulses for H-series and 2 built-in clock pulses for E10 PLCs with periods ranging from 0.01s to 1 minute. Built-in clock pulses are useful if you need a time base to create, for example a "flashing light". A contact such as "Clk:0.1s" will automatically turn itself ON for 0.05s and then OFF for another 0.05s and then ON again, resulting in a series of clock pulses of period = 0.1 second.

6. Next, move the highlight bar to the right end junction of the parallel circuits as follow:



7. Now, click on the  icon to insert a special function coil. A popup menu will appear for you to select the desired special function. Click on the item "4.[AVseq]-Advance Sequencer" to insert the Advance Sequencer function [AVseq].



8. This function is one that will increment the step counter of Sequencer #1 each time its execution condition goes from OFF to ON.

**Again, remember to press the **<Enter>** key to complete Circuit #2**

Back to Step 3

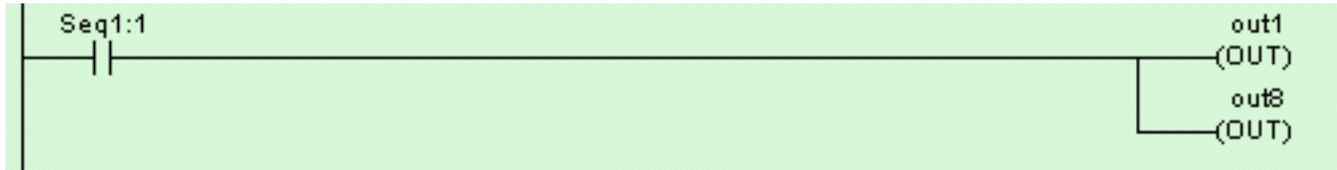


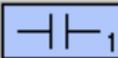
Go To Step 5

## Ladder Logic Programming Tutorial STEP 5



1. Circuits #3 to #6 are similar to one another. They make use of the Sequencer to turn on the Outputs 1 to 8 to create a pattern of "running lights" when executed. The label "Seq1:1" of the contact in Circuit #3 represents Step #1 of Sequencer 1. Remember that each sequencer can have up to 32 steps (Step #0 to 31), with each step individually accessible as a contact. A normally-open contact "Seq1:1" will be closed whenever the step counter of Sequencer 1 reaches number 1. Likewise a normally-closed contact "Seq5:20" will be opened when the step counter of Sequencer 5 reaches number 20.



2. To create the normally-open contact "Seq1:1", left-click on the  icon. When the I/O table pops up, scroll to the "Special Bit" table and select the item #1 "SeqN:x". When prompted to select a sequencer choose "Sequencer 1" and another dialog box will open up for you to enter the specific step number for this sequencer.
3. We have thus far been creating ladder circuits only by clicking on the ladder icons. A short-cut method of choosing elements to be created without using the mouse does exist. Observe the icon carefully and you will notice a small numeral at the lower right hand corner of each icon which correspond to the shortcut key. You may wish to try this short-cut for the remaining part of Circuit #3. Press the <7> key and the Output table will immediately pop up for selection of a coil. Pick "Out1" from the "Output" table and the "Out1" coil will be connected.
4. Circuits #4, 5 and 6 are very similar to Circuit #3 and you should have little problem creating them. Complete these circuits and we are ready for some interesting simulation exercises. When you have created all the circuits, press <Enter> key or <ESC> key at the last blank circuit to end "Ladder Edit" mode.
5. We can make our program more comprehensive to other users by utilizing the "**Comments**" feature of TRiLOGI. Open the "Circuit" menu and select "Insert Comment". A comment editor window will be opened up to allow you to add your comments to any part of the circuit. When you are done with your comments, just press <ESC> key or close the comment editor window and the comments you just entered will be inserted between the circuits. Each comment occupies a circuit position and there is no limit to the number of lines a comment circuit may have. (However, if you wish to keep data file compatibility with the old TRiLOGI Version 3.x you should limit the comment to no more than 4 lines per comment and each line should contain no more than 70 characters.)

A comment circuit may be moved around or deleted just like any other ladder circuits. If you wish to edit the comment, just double-click on it or press the <Spacebar> to open up the comment editor window. You can use the normal text editing keys such as left, right, up, down cursor keys, and <Ctrl-Left>, <Ctrl-Right>, <Del> and <Backspace> keys for editing the comment text.

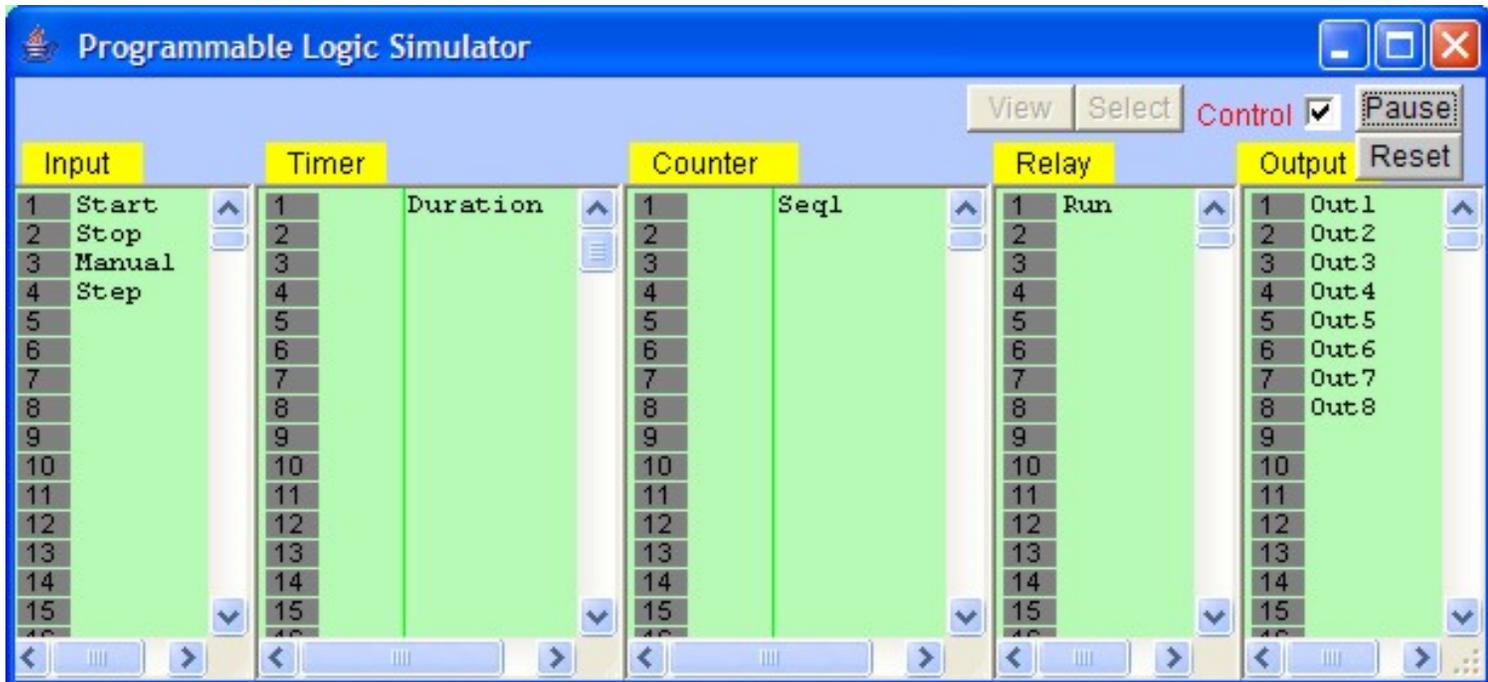
Back to Step 4



Goto Step 6



**The stage has been set and the show is ready!** Having completed the demo program, it is time to test if it works as intended using the built-in real-time programmable controller simulation engine. Open the "Simulate" pull-down menu and activate the command "Run (All I/O reset) - Ctrl+F9". TRiLOGI will immediately compile the ladder program and if no error is detected, it will instantly proceed to open up the "Programmable Logic Simulator" screen, as shown below:



1. If you have followed closely all the instructions during the creation of the demo program, you should not encounter any compilation error. However, if you do receive an error message, then please check your circuit against the picture shown in the [assignment](#) page, make all necessary corrections and then try again.
2. The simulator screen comprises 5 columns: Input, Timer, Counter/Sequencer, Relay, and Output. With the exception of the Relay table which contains up to 512 elements, and the Timer table which contains up to 128 timers, all other columns contain 256 elements each. Every column has its own vertical scroll bar. You can use the mouse to scroll each column independently to locate the desired I/O.
3. The label names for the inputs, outputs, relays, timers and counters defined earlier in the I/O tables automatically appear in their respective columns. To the left of each label name column is an "LED" lamp column which indicates the ON/OFF state of respective I/O. A red color lamp represents the ON state of an I/O, whereas a dark grey color lamp represents an OFF state. The I/O number is indicated in the middle of the lamp.

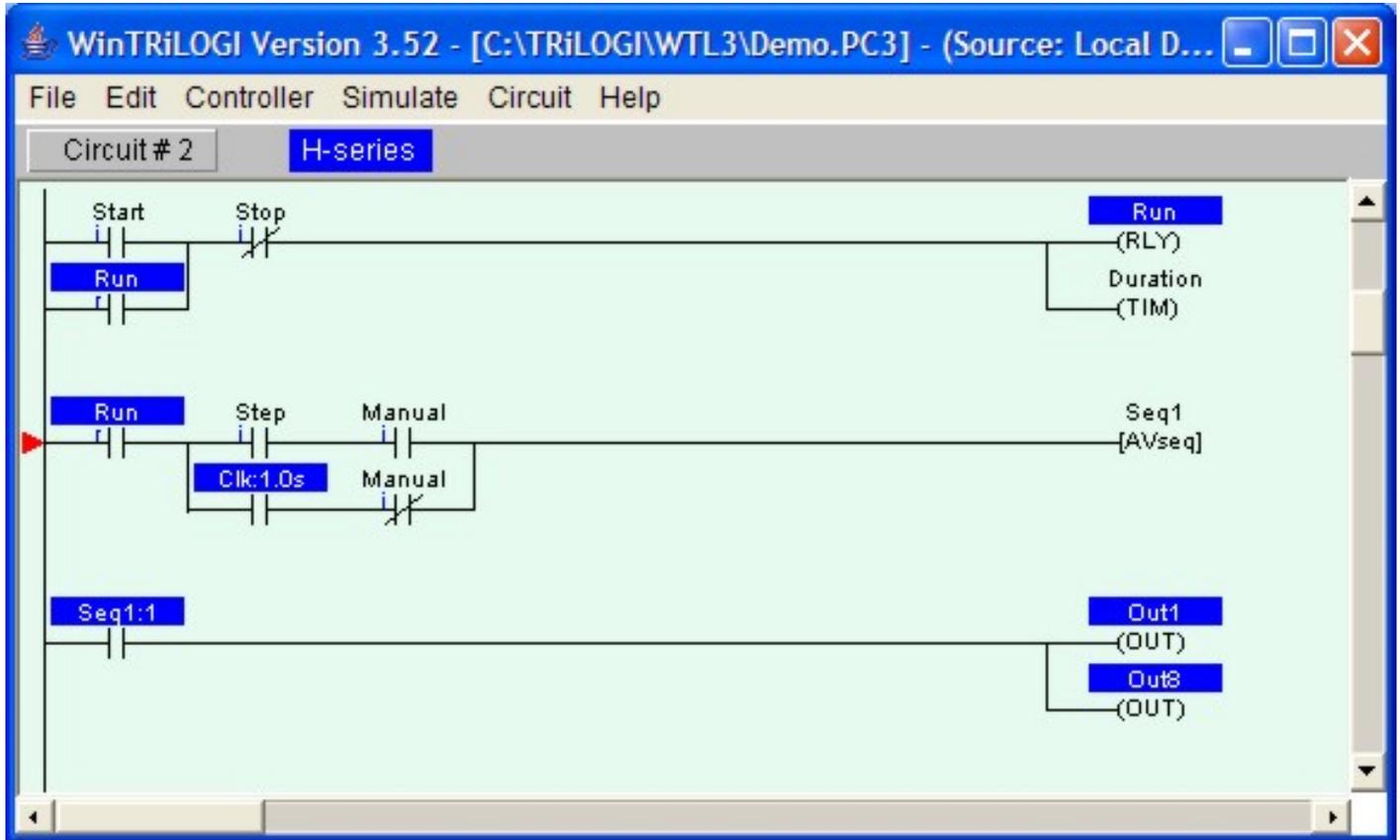
The simulator requires the use of the mouse to work properly so it is important to remember the mouse button actions as follow:

<b>Left Mouse Button</b>	Turn ON the I/O when pressed. Turn OFF when button is released.
<b>Right Mouse Button</b>	Toggle the I/O when pressed once. (i.e. OFF becomes ON and ON become OFF)

4. Our ladder program requires us to "push" the "Start" button momentarily. You can simulate this action by moving the mouse pointer to the "Start" label (or the LED lamp) and press the **LEFT** mouse button once and then release the button. The action starts!
5. At this time, notice that the relay "RUN" is latched ON and the timer "Duration" begins to count down from the value of 1000 every 0.1sec, and the Output #1-#8 are turning ON/OFF sequentially in a "running light" pattern. Sequencer "Seq1" in the "Ctr/Seq" column begins to count upward from 1 to 3 and then overflows to 0 and repeats continuously. For each step of the Sequencer, the corresponding Output will be turned ON. Our demo program will show a running light pattern starting from Outputs 1 & 8, then 2 & 7, 3 & 6 and 4 & 5 and then back to 1 & 8, 2 &

7.....

6. Now you should verify that the logic works as intended by observing the ladder diagram. You should notice that the "Run" labels in all circuits are highlighted as shown below:



7. The logic states of any I/O can be displayed on the ladder diagram directly. An Input, Output, Relay, Timer or Counter contact that is turned ON will have its label name highlighted in the ladder diagram. This feature helps greatly in debugging and understanding the logical relationship between each I/O. For example, from the above figure, we can see clearly that the "Self-latching" circuit for relay "Run" works as intended: when we first turn ON the "Start" input, "Run" will be energized and its contact which is parallel to "Start" will hold itself in the ON state, even if we subsequently turn OFF the "Start" input by releasing the button.
8. The timer coil "Duration", being connected in parallel to "Run" relay, will also be energized. However, its contact will only be closed after 100 seconds (when its present value counted to 0). To break the latched On "Run" relay, we must energize the "Stop" input momentarily to break the "power" flow. Try it now.
9. Let's restart the system by turning ON the "Start" input momentarily again. Next, we want to turn ON the "Manual" input. Move the mouse pointer to the "Manual" input and then press the **right mouse button**. "Manual" input will be "stuck at "ON" state even after you have released the right mouse button. Click on "Manual" button using the right mouse button again and it will be turned to OFF.
10. When "Manual" input has been turned ON, the running lights should stop. This is because the normally-closed contact of the "Manual" input in Circuit #2 is now turned OFF and the 1.0s clock pulse could not trigger the [AVseq] function anymore.
11. If you now left-click on the "Step" input, the running lights will move one step at a time in response to your mouse click. Observe the Seq1:x contact with respect to the counter value of Seq1 and the logic of this circuit become very clear instantly.
12. Observe that the timer "Duration" continues to count down every 0.1 second, and when it reaches 0, the "Duration" output coil label will be highlighted. You can use this timer to stop the program by connecting a N.C. "Duration" contact to Circuit #1. This is left as an exercise for you!

## Summary

We have completed this hands-on session and have successfully created a simple ladder program. We have also performed real time simulation to test the program's functionality. By now you would probably have a good appreciation of TRiLOGI's superb capability and ease of use and are ready to include TRiLOGI as an integral part of your programming needs.

---



Go Back to Step 5

# The File Menu

The File menu provides commands for the opening/saving of TRiLOGI files on the local harddisk.

## 1. New <Ctrl+N>

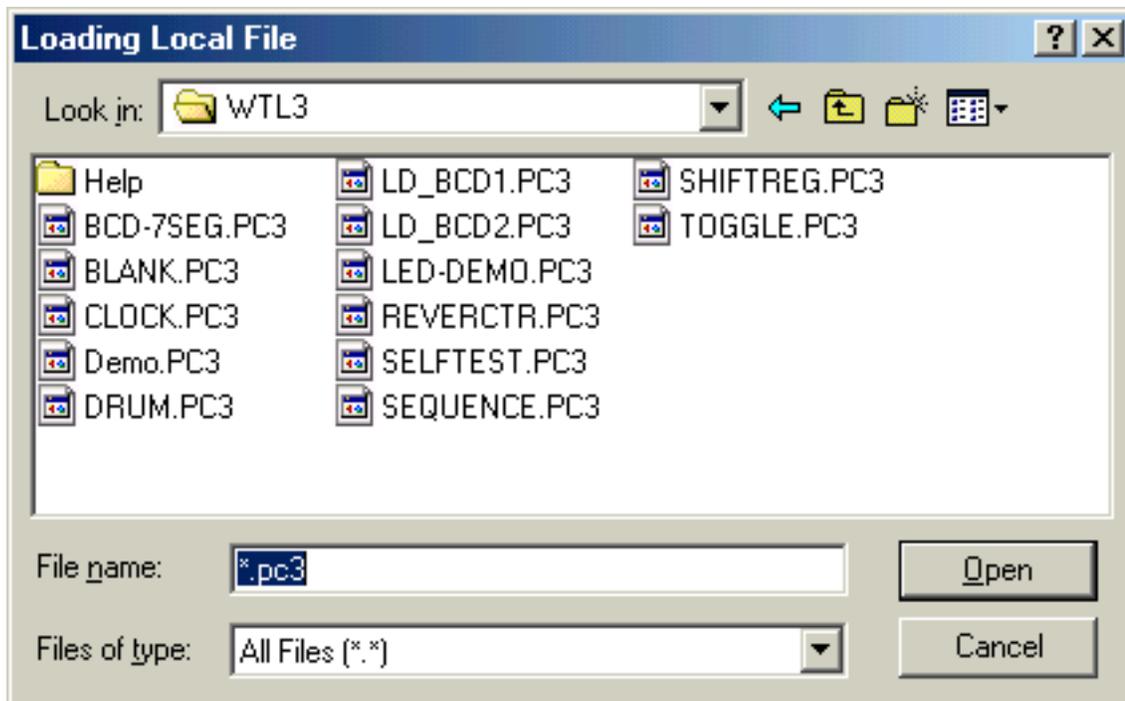
Activate this command when you want to create a new ladder logic program. All current ladder circuits will be cleared from the screen and the default filename is "Untitled".

## 2. Save <Ctrl+S>

This command updates the currently opened ladder logic program and all I/O tables to the disk. .

## 3. Open (Local Drive)

WinTRiLOGI 3.5 is able to open files created by DOS TRiLOGI Version 3.3E, 3.3 and 4.x as well as Internet TRiLOGI version 5.x. After you have selected the "**Open (Local Drive)**", you will be presented with the file open dialog box. The default extension of the files will depend on the PLC model selected, as follow:



For E10+ PLCs, the default extension is ".PE3", whereas for H-series PLC the default extension is ".PC3". If you have TRiLOGI files previously saved with a different extension then you will need to manually enter the extension in the "**File name:**" field. E.g. If you have previously created a ladder program using Internet TRiLOGI version 5.x and you wish to open it in WinTRiLOGI 3.5, then enter "\*.PC5" in the "File name:" and all files with the ".PC5" extension will become visible in the folder windows.

## 4. Save As (Local Drive)

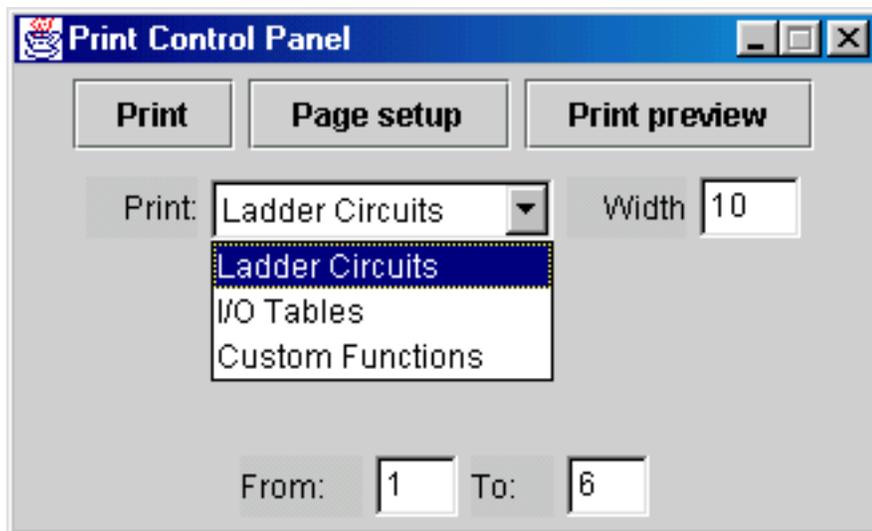
You can save the currently opened files to another file name or into another folder using this command.

**Caution!**

Please note that due to the enhanced capability of the WinTRiLOGI 3.5, such as the longer label name and unlimited number of comment lines, a file saved by WinTRiLOGI may become unreadable by the old DOS TRiLOGI program if you used these enhanced features. So do make a backup copy of your old ".PE3" or ".PC3" files created by the DOS program before editing them using the WinTRiLOGI program, just in case you prefer to go back to using the DOS program to continue with your work.

**4. Print**

When executed the following "Print Control Panel" will appear:



To print, first select the item from the choice box and define the range you wish to print and then click on the "Print" button. For "Ladder Circuits", the range indicates the circuit numbers. For "I/O Tables", the range indicates the I/O number (up to 256) and for "Custom Functions", the range is the function number. (**Note:** Although "Custom functions" are not supported on the E or H series PLCs, you can still use this print feature to print out Custom function created by DOS TRiLOGI version 4.x and Internet TRiLOGI version 5).

You can use the "Print preview" button to check the pagination of the printing on screen. You can select paper size and print orientation. etc. by clicking the "Page setup" button. Empty custom functions will be automatically skipped to save paper. When you select to print the "Ladder Circuits" a special "**Width**" textbox appears. This textbox is for you to enter the maximum number of series element that can be printed on the paper width. Changing this number affects the scaling of the ladder diagram when printed. The smallest number is 5 and largest number is 13. Use a smaller number if you wish to have a larger printout. However, please note that if your ladder program contains circuits with more elements than that indicated by the "Width" parameter the "out-of-page" part of those ladder circuits will not be printed.

**5. Exit**

Execute this command to exit orderly from the TRiLOGI program. You will be prompted to save the current file if the contents have been edited and the changes have not yet been saved.

# The Edit Menu

---

## 1. Abort Edit Circuit

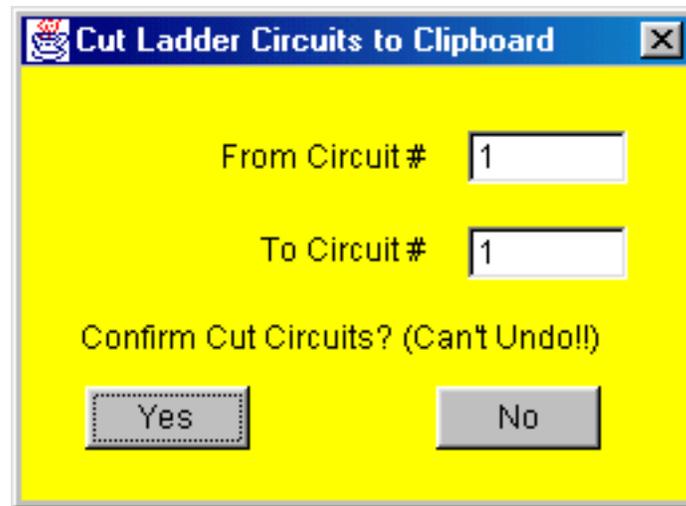
Changes made to the current ladder circuit can be aborted if you execute this command before pressing <Enter> to accept changes made to the current circuit. If changes have already been accepted by pressing the <Enter> key, then this command will have no effect. This command is useful if you wish to completely abandon changes you have made to a circuit without going through all the undo steps.

## 2. Undo <Ctrl+Z>

Undo the last changes made to a ladder circuit. TRiLOGI automatically stores the last 10 edited steps so you could execute undo several times to restore the circuit back to its original shape.

## 3. Cut Circuit - <Ctrl+X>

You can remove a number of circuits from the current ladder program and store them temporarily in the clipboard for pasting into another part of this ladder program or into another file altogether. In other words, it lets you move a block of circuits from one part of the ladder program to another part or into another file. Once you execute the "Cut Circuit" command, a prompt box as shown below will appear. You have to specify the range of the circuits you wish to cut and press the "Yes" button to remove them from the ladder program.



**Please note that you can't UNDO a Cut Circuit operation.**

## 4. Copy Circuit (Ctrl+C)

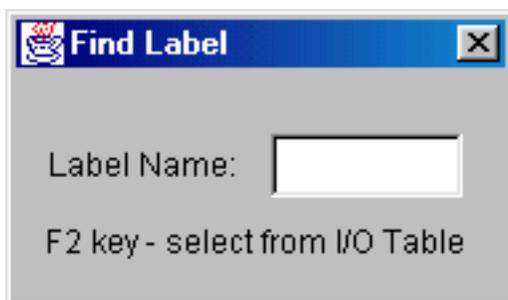
You can copy a block of circuits from the current ladder program and store them into the clipboard for pasting into another part of this ladder program or into another ladder program file altogether. The range dialog box similar to "Cut Circuit" will appear for you to enter the range of circuit to copy.

## 5. Paste Circuit <Ctrl+V>

When you execute this command, the block of ladder circuit which you "Cut" or "Copy" into the clipboard will be pasted just before the currently selected circuit. The current circuit number will be adjusted to reflect the change.

## 6. Find <Ctrl+F>

The Find command allows you to quickly locate a ladder logic circuit that contains a particular label name. This is useful for searching for the activity of a particular I/O in the program.



**Find Ladder element:** you can enter into the the text field a string that partially or fully matches the label name you wish to locate. You can also press the <F2> key to open up the I/O table and pick the label name from the I/O table.

## 7. Goto <Ctrl+G>

Use this command to move towards a specific circuit number. The "Goto" command is particularly useful if your program contains many circuits, and it is inconvenient to search for a particular circuit using the mouse or the cursor keys.

## 8. I/O Table <F2>

Open up the [I/O Table](#) for defining label names for the PLC's I/O. For detailed explanation of I/O tables, please click on the following link: [I/O Definition Table](#)

## 9. View I/O Type on Ladder <F3>

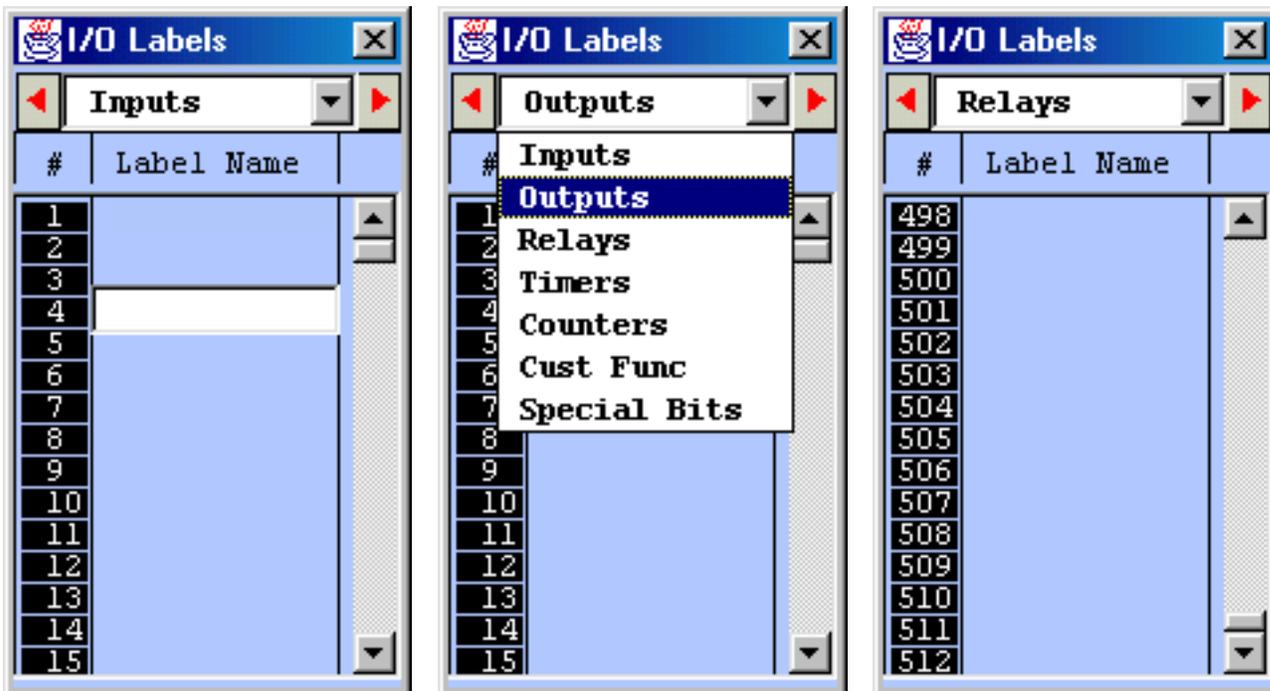
Toggle between display or no display of the I/O type for ladder logic contacts on the screen. All ladder logic contact symbols are normally identified by their label names. However, you can also choose to display an optional small literal to indicate the I/O types. e.g. i=input, o=output, r=relay, t= timer and c=counter. When TRiLOGI first starts, the display is enabled but you have the option of turning it off if you find it distracting.

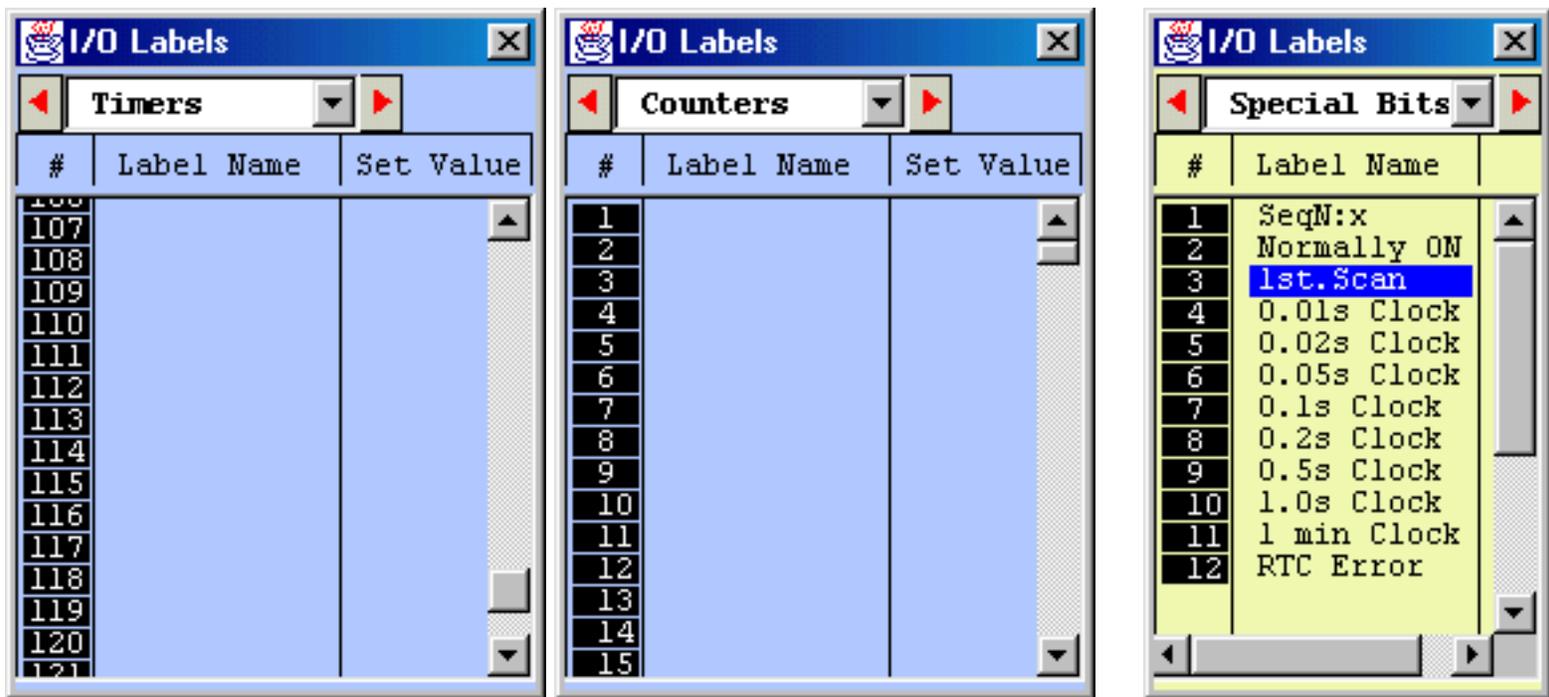
# I/O Definition Tables

Unlike many ladder logic editors which are "numeric-centric" - meaning all ladder components are referred to by their I/O numbers, TRiLOGI programs are "labelname-centric" because TRiLOGI constructs the ladder logic strictly based on the use of label names and the compiler generates the correct I/O representation based on the labelnames defined in the I/O table. The advantages are that ladder programs constructed from label names are far easier to understand and remember, and you can rearrange the I/O location for a certain label without changing the program at all (e.g. move a load to another output driver).

You can open the I/O table by pressing the <F2> key. It is good to remember this short cut key since you will be using it very often. The first time you press <F2> the input table will be opened. You can then scroll to other tables using one of the three methods:

- Using the left/right cursor keys.
- Click on the word "Inputs" - a choice box will open for you to select the I/O table you wish to scroll to.
- Click on the two buttons beside the "Inputs" choice box to scroll left or right from table to table.





## Editing Label Names

To create or edit labelname for an I/O, simply click on the I/O number in the opened table and a text entry box will appear for you to enter/edit a label name. Once you have finished editing, press the <Enter> key to close the box and the name will appear on the I/O table. You can also use the keyboard to move the blue color highlight bar to an I/O location and press the <Space> bar to edit the label.

## Editing Set Values (Timers and Counters only)

Timer and Counter Tables each has an extra "Set Value" column. If you define a new timer/counter label the Set Value field for this newly defined timer/counter will be opened for entry. Otherwise you can manually open this field for entry by clicking on it or by pressing the <End> key. Only numeric values between 0 and 9999 should be entered. If you enter an illegal character, the text field will not be closed when you press <Enter>.

## Label Name Restrictions

You can enter up to 10 characters per label name. Only alphabets and numbers can be used. No space is allowed between characters. If you enter more than 10 characters only the first 10 characters will be recorded when you press the <Enter> key to close the text entry field. If you enter a name that contains illegal characters, they will automatically be converted to underscore characters.

## Important Notes

- You can **shift** the Items in the I/O table up or down or insert a new label between two adjacent, pre-defined labels. Simply press the <Ins> key or **Right-Click** the mouse button to pop up the "Shift I/O" menu which allows you to shift the selected I/O. However, please note that if you shift the I/O down, the last entry in the I/O table (e.g. Input #256) will be lost.
- WinTRiLOGI allows I/O label names of up to **10** characters. However, if you wish to keep compatibility with Version 3.x, you should use no more than **8** characters to define the I/O names.

# The Controller Menu

---

All commands in this menu are for communication with the PLCs via the PC's RS232 serial communication port. Most PCs have one or more RS232 communication port that appear as a male DB9 socket. If your PC does not have a serial port, we suggest you purchase a add on serial card on PCMCIA or PCI bus format. Although some USB to RS232 converter may also be used (one USB to RS232 converter that we tested successfully is made by BAFO model BF810), we did receive reports that some users experienced difficulties communicating with the PLC when using a USB to RS232 converter.

## 1. Select Controller <Ctrl-I>

The WinTRiLOGI program allows you to use an RS485 adapter such as the "Auto485" to selectively program or monitor any networkable PLC (such as the H-series) that are linked to a twisted pair RS485 network. Each PLC must be defined with a unique ID. You can use this command to select which PLC you wish to program/monitor by entering the ID of the PLC of interest in the "Current ID" field (in hex format from 00 to FF).

Note that if there is only a **single** PLC connected to the PC via RS232 or RS485 and you have no idea what its ID is, you can click on the "Detect ID" button to check the PLC's ID automatically. The "Detect ID" command uses the point-to-point hostlink command "IR\*" to retrieve the PLC's ID. You should not click on the "Detect ID" button if more than one PLC is connected to the PC since all the PLCs will response to the "IR\*" command and the PC will receive a garbled response string from the network.

## 2. Serial Port Setup

You can change the serial port selection as well as communication parameters using this command. For more details explanation [please click here](#).

## 3. On-Line Monitoring <Ctrl+M>

See [On-Line Monitoring](#) help document for details.

## 4. Program Transfer to PLC <Ctrl+T>

You can use this command to transfer your TRiLOGI ladder program into the PLC. You will be prompted to confirm your action to prevent accidentally affecting the target PLC. The ladder program must be compiled successfully before the program transfer process can take place. The progress of the transfer process will be clearly shown on the program transfer dialog box.

## 5. Open Matching Source File

You can use this command to query the connected PLC for the filename of the last TRiLOGI program transferred to it and it will attempt to match it to a file stored in the installation directory. If the file is found, it will be opened. Otherwise it will report that the file is not found. Note that this command only opens the source file based on file name matching. It does not verify whether the file has been modified. It is the user's responsibility to ensure that the file stored in the PC is the same one that has been compiled and transferred to the PLC.

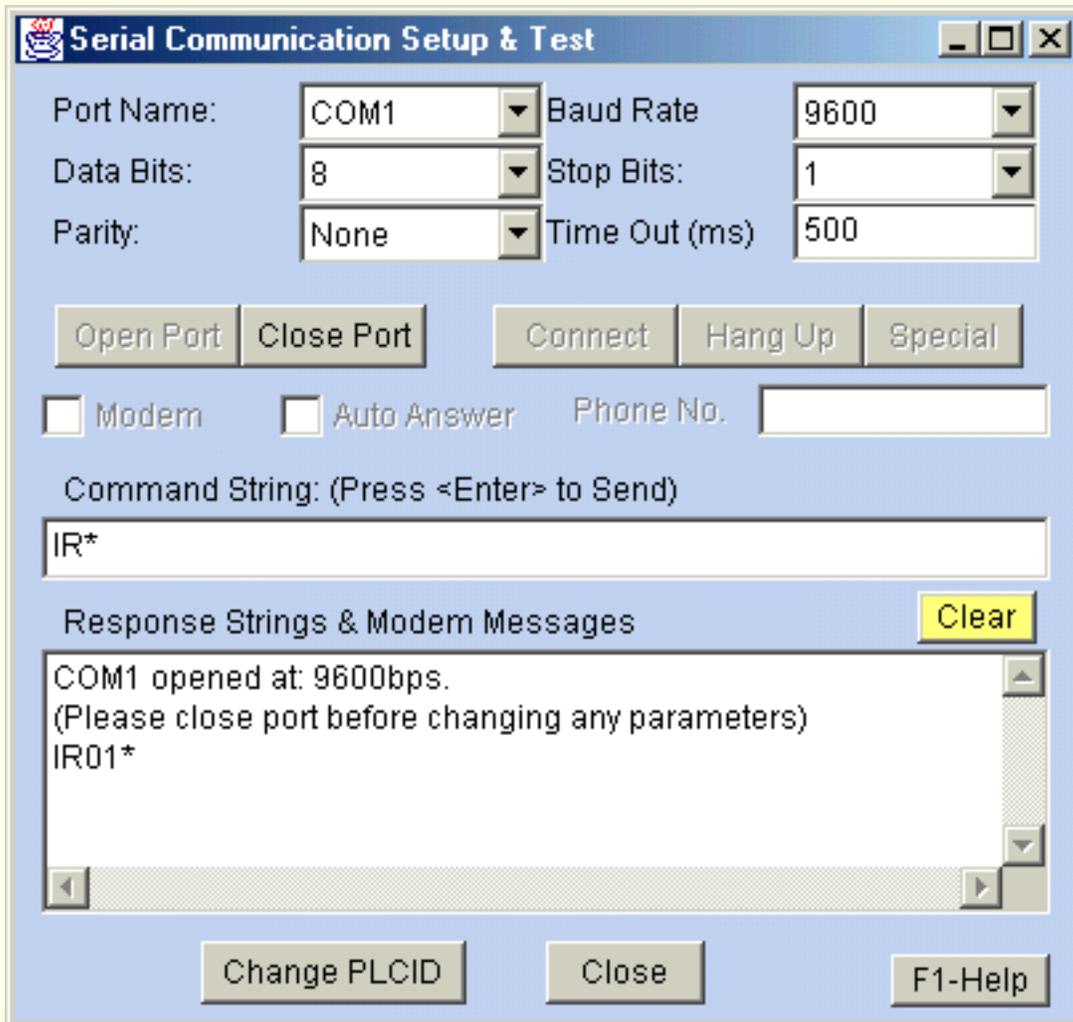
Note that if you have created a new file (i.e. the file name is "Untitled" ) and then attempt to perform on-line monitoring, this command will be called automatically to try to open a file that matches the PLC. The command is also invoked when you select a PLC with a different ID either from the "Controller" menu or from within the "Full-Screen Monitoring" window.

## 6. Get PLC's Hardware Info

You can find out the PLC's model number, the maximum of input, outputs, relays, timers and

counters supported on this PLC as well as the total amount of program memory available. The same info will be displayed when you try to transfer a program to the PLC.

## Setting Up Serial Communication Port



This dialog box allows you to configure the serial port of the host computer to match the setting on the PLC for proper communication. Most of the items here are self-explanatory. If you have more than one PLC connected to the host computer via RS485, all the PLCs must have the same serial port settings. The **Open Port** button allows you to test whether the communication port is

available to WinTRiLOGI. You can also click the **Close Port** button to temporarily relinquish the port to other applications (such as the TLServer in Internet TRiLOGI Version 5). Note that you will need to close an opened port before you can change its parameter.

The (Command String) text entry field allows you to test communication with the PLC using its native or MODBUS ASCII protocols. Note that both point-to-point and multi-point host link commands are accepted here. If you enter a string here and press <Enter>, the ASCII string will be sent to the PLC connected to the serial port and the response string will be displayed in the bottom text box. If the comm port is not yet opened this command will automatically open it.

If you have only **one** PLC connected to your computer, then you can test the communication now using the following command string:

```
Command String : IR*
Response String: IR01*
```

The response string tells you that the ID address of this single PLC is 01. You can then try other

host link commands using this ID address. (e.g. @01RI0000\* to query the states of inputs #1 to #8) If you have more than one PLC connected you should not use the "IR\*" since all connected PLCs will try to respond simultaneously, thus resulting in a garbage return string.

To change the ID of a PLC, e.g., from 01 to 05, you can send the command string "@01IW0500\*" to the PLC. There is also a new "Change PLCID" button which does this for you automatically. You can click on the "Detect ID" button to check the current ID and then the "Change ID" button to write the new ID to the PLC.

---

## Changing Communication Settings

WinTRiLOGI program automatically set the default comm port settings according to the PLC model selected (for E10+ and H-series PLC it is **9600 bps, 8 data bits, 1 stop bit, no parity**) . Most likely you may want to leave the comm port settings at their default values: Some reasons for changing the comm port settings may be due to the need to change the PLC's serial port to lower values (e.g. for communication via radio or 2400bps modem).

## Modem Support

**1. Dial Modem:** WinTRiLOGI incorporates support for dialing a modem connected to the PC's com port. This is useful if the PLC has to be located at a remote location yet still has access to the public telephone line or to a cellular phone. You can then connect the PLC to a standard analog modem such as the US Robotic 33.6Kbps or Hayes Acura smart modem. WinTRiLOGI can then dial the phone number of the remote modem and make a connection. Once a connection is established, you can monitor/control or transfer program to the PLC just like it is via the serial port directly.

### Notes:

- Due to the time delay for modulation/demodulation process as well as transmission delay, two-way communications via modem tends to be noticeably slower than connection made by direct cable connection. This is quite normal and does not indicate a problem with the communication setup.
- The PC's modem must be able to emulate the COM port of the PC in order for the WinTRiLOGI modem function to work. Some of the newer computers employ "win modem" or "soft modem" which may only work with Windows' dial-up networking. These kinds of modems are implemented in software and they do not necessarily emulate a standard PC COM port properly. They also demand quite a bit of CPU horsepower to process the communication properly. Therefore these type of modems may not work too well with the WinTRiLOGI. If your built-in soft modem does not work properly with WinTRiLOGI, you should get an external PCMCIA card modem and these are quite inexpensive nowadays and they will work much better with the WinTRiLOGI modem support function.

To setup WinTRiLOGI to dial a modem, first close the active COM port by clicking on the "Close Port" button. Select the COM port where the modem is connected to. (you can find out the which COM the modem is connected to by checking the "Control Panel -> Modems -> Properties") Click to select the "Modem" checkbox. You will then be able to enter a telephone number to dial. The 3 buttons: "Connect", "Hang Up" and "Special" become enabled when you select the "Modem" mode. Note that the "Baud Rate" field now becomes the "DTE speed" which specify the line rate between the PC and the modem (this has nothing to do with the actual baud rate between the modems which will be automatically negotiated based on the quality of connection). Normally you should leave the DTE speed set to the highest value (115200) unless your modem manufacturer specifies otherwise. The PLC can be operating at a different baud rate from the PC to modem-line-rate because of the modulation/dimoduation action of the modem.

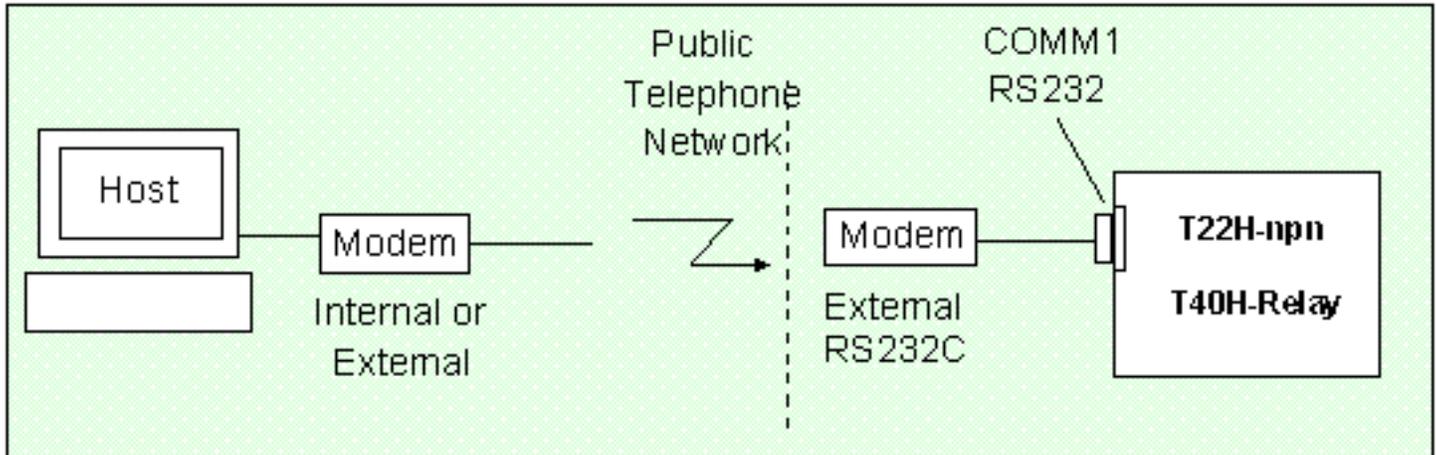
**\* Important:** The PLC-to-modem connection must be properly prepared before you can use

WinTRiLOGI to connect to the PLC's modem. [Click here to read more details about the PLC-to-Modem Communication Setup.](#)

Once you have entered a proper phone number, click on the "Connect" button to start dialing the modem (make sure that the "Auto Answer" check box is not checked). If the remote modem is busy or does not answer the call you will see the corresponding error messages in the response box. Click on the "Hang Up" button anytime to abort the dialing operation.

## PLC-to- Modem Communication Setup

A remotely located H-series PLC can be connected to a host PC via public-switch telephone network (PSTN), radio or cellular phone network. This can be accomplished by using two analog modems, one connected to the PLC's RS232 serial port, and another modem connected to the remote host PC as follow:

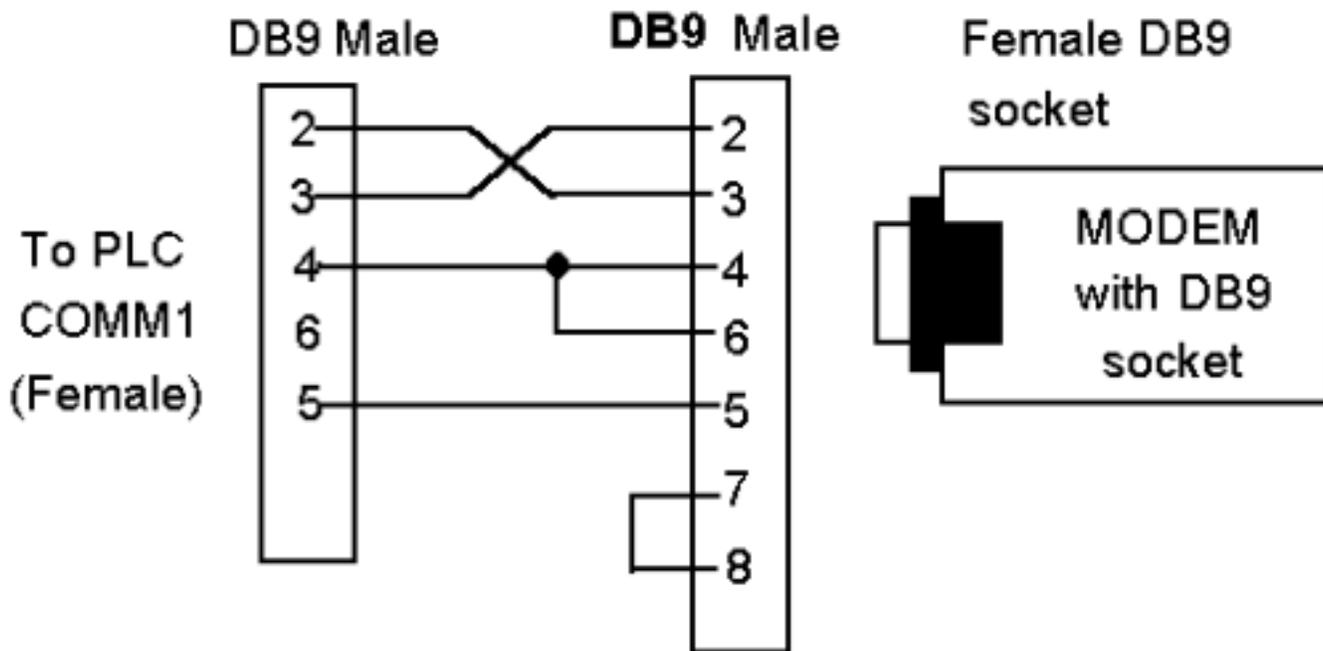
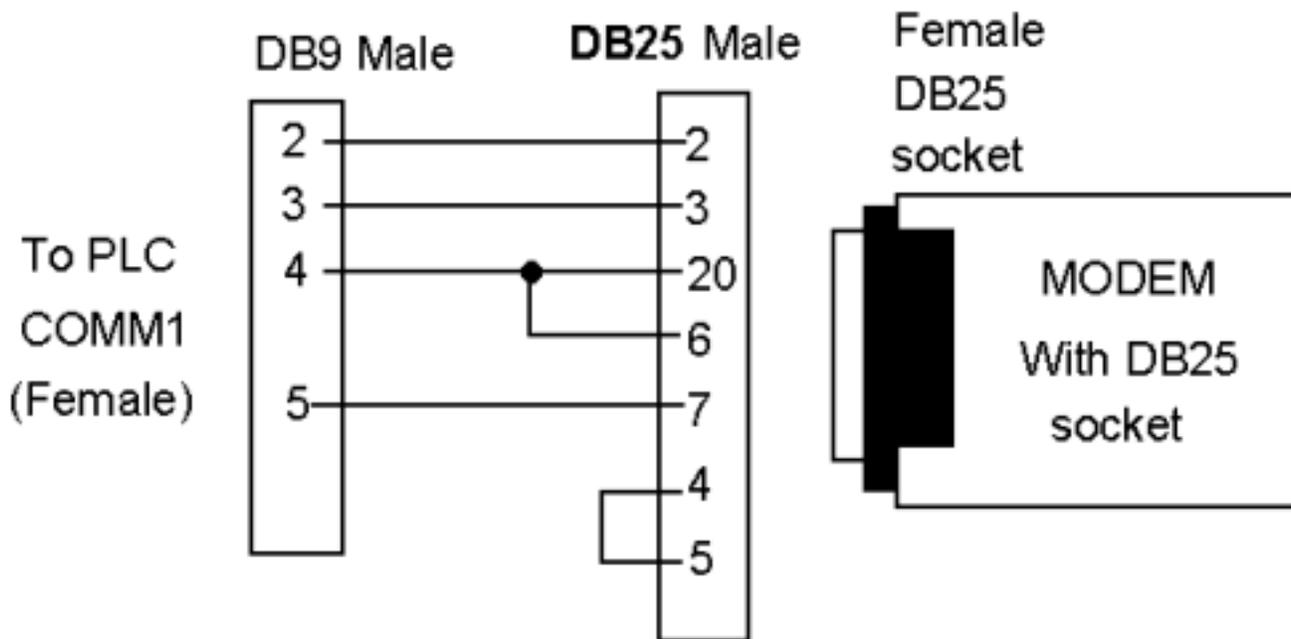


There are a some technical issues that need to be handled carefully in order to successfully implement the modem-linked host communications as described in the following sections.

### 1. Modem Connection

**Modem1:** The host PC may use any internal or external modem that can communicate at 2400 bps or faster. Connect the modem to the PC as instructed in the modem's manual and connect the phone line to the phone jack on the back of the modem marked "WALL" or "Line". The WinTRiLOGI program already includes built-in support for dialing a modem connected to one of its COM port.

**Modem2:** The modem to be attached to the PLC (modem2) must be an external modem with an RS232 connection port. Since modem are DCE type device, they most likely come with a female type DB25 or DB9 socket meant for plugging into the PC's RS232 port. Since the PLC's host link port is also a female DB9, we need to construct a DB9-male-to-DB25-male cable or DB9-male-to-DB9-male cable to link the PLC to the modem, as follow:



## 2. Communication Speed

When communicating via modems, there are two different definitions of communication speeds that you should be aware of:

The "DTE Speed" or "line rate" is the serial communication speed between the modem and the device connected to its RS232 port. Most modems can automatically detect the RS232 speed of the device and can assume any speed from 1200, 2400 all the way to 115,200 bps. The first ASCII character they receive from the host will determine the DTE speed that the modem will use to communicate with the host device.

The "modem-to-modem communication speed" is what you read on the modem specifications, such as 33.6Kbps, 56Kbps etc. When two modems are connected,

they automatically negotiate for the best speed to communicate between the two of them based on the quality of the phone connection and the maximum speed that both modems are able to achieve. We usually have no control of what speed they choose to communicate. But one thing is for sure, which is that the modem-to-modem speed is always lower than the DTE speed.

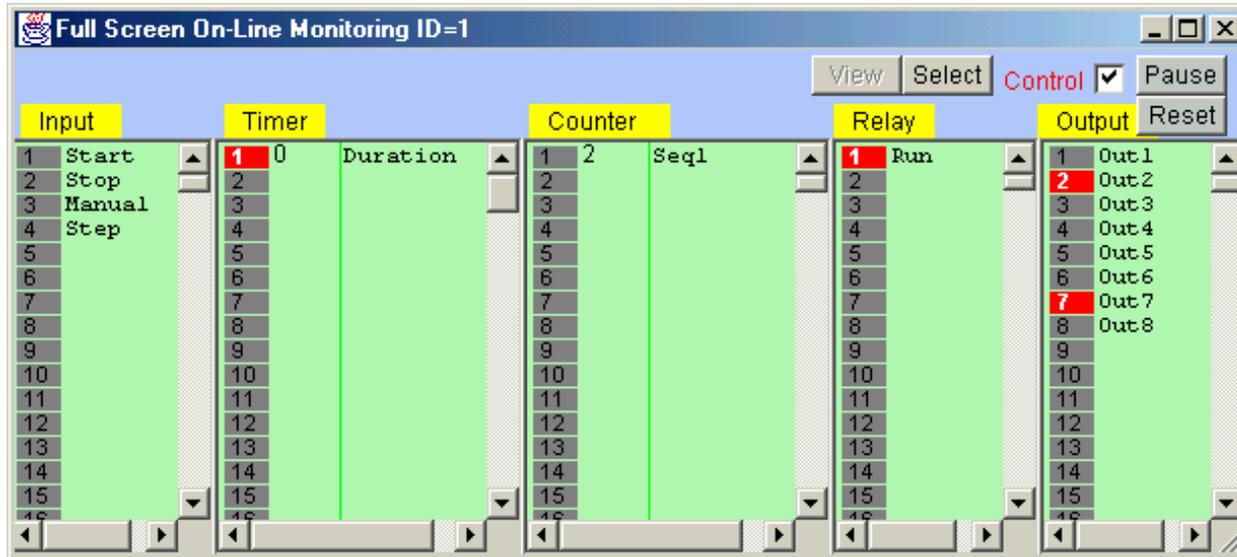
Since the default communication baud rate of the H-series PLC's RS232 serial port is 9,600 bps, and since the H-series PLC are unable to initiate communication with the modem, you should therefore use a PC and a HyperTerminal or any other terminal emulation program configured to 9600bps to configure the modem before connecting it to the H-series PLC.

Firstly, you can send a string such as "ATZ" to the modem to reset its speed into 9600bps. Next, you will need to configure this modem to power up in "Auto Answer" mode. This is usually done by sending the command ATSO=1 to the modem.

Most modems allow you to save the current settings to their non-volatile memory so that when a modem is next powered up it will assume all the saved settings. For US Robotic the command is "AT&W". Other modem may use different AT command string to save the settings into non-volatile memory. Once the modem has been configured to power up with a "line rate" of 9600bps and in Auto Answer mode it can be connected to unattended H-series PLC at a remote location.

# Full-Screen Monitoring

After you have successfully established communication with the PLC, a "Full-Screen Monitoring" window will be opened for you to monitor and control your PLC. The ID address of the PLC selected for on-line monitoring will be shown on the window's title, as shown below:



The Full Screen Monitoring screen comprises 5 columns: Input, Timer, Counter/Sequencer, Relay, and Output. With the exception of the Relay table which contains up to 512 elements, all other columns contain 256 elements each and each one has a vertical scroll bar. You can use the mouse to scroll each column independently to locate the desired I/O.

The label names for the inputs, outputs, relays, timers and counters defined earlier in the I/O tables automatically appear in their respective columns. To the left of each label name column is an "LED" lamp column which indicates the ON/OFF state of respective i/os. A red color lamp represents the ON state of an i/o, whereas a dark grey color lamp represents an OFF state. The I/O number is indicated in the middle of the lamp.

The monitoring screen requires the use of the mouse to work properly so it is important to remember the mouse button actions as follow:

<b>Left Mouse Button</b>	Turn ON the PLC's I/O when pressed. Turn OFF the PLC's I/O when button is released.
<b>Right Mouse Button</b>	<b>Toggle</b> the PLC's I/O when pressed once. (i.e. OFF becomes ON and ON becomes OFF)

There is a check box **Control**  near the upper right hand corner of the on-line simulator. This check box must be checked before you can force set/reset the PLC's I/O. When you run the "Full Screen Monitoring" the first time, this box is unchecked to avoid accidentally changing the state of a PLC's I/O.

### Important Notes

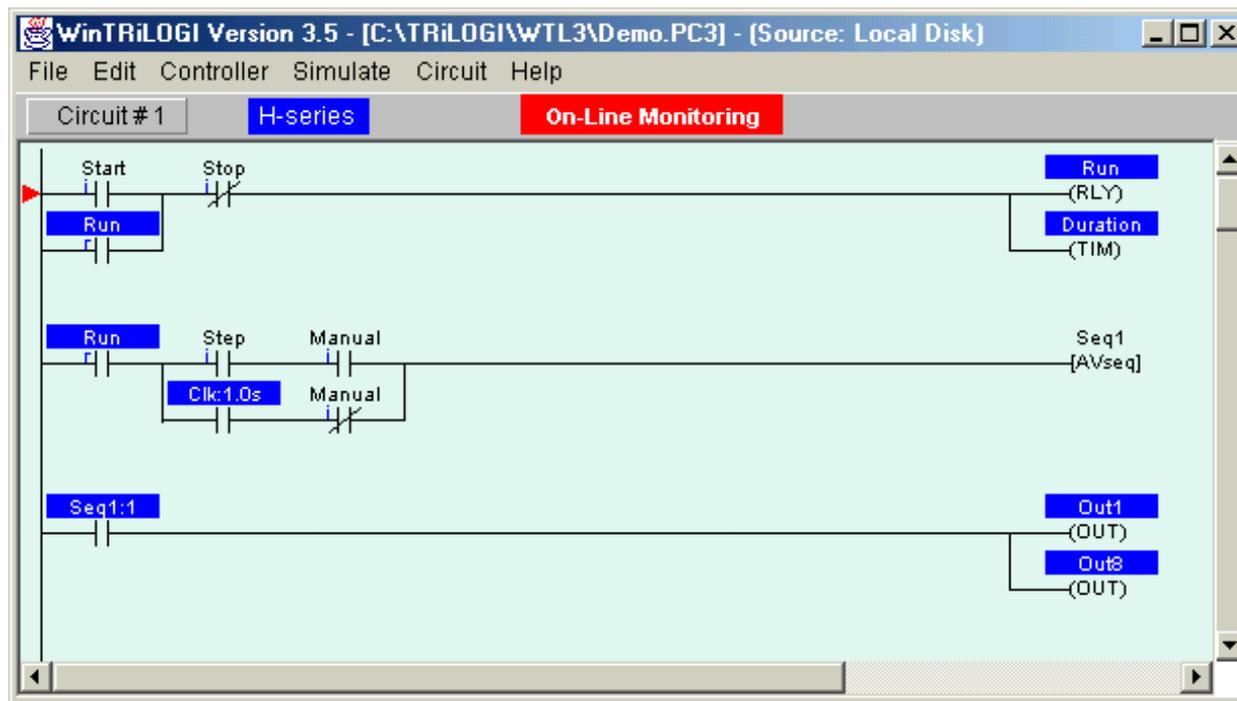
- If you right click on the "Input" field, the selected PLC input will be changed only for 1 scan time. After that the PLC's I/O update process will automatically change the input bit back to its actual physical input state. It is not possible to permanently force the PLC's input to a different logic state even if the PLC is paused. As such, you will not be able to see the changes being reflected on the screen when you click on an "Input".

### View, Select, Pause, Reset Buttons

Pause	Halt the PLC. The "Pause" lamp of the PLC should be lighted up after you click the pause button. This button is disabled when the "Control" check box is not selected.
Reset	Reset All I/Os and data in the PLC. (same as <Ctrl-R>). This button is disabled when the "Control" check box is not selected.
Select	Allows you to select a PLC for monitoring by specifying the ID address. Up to a maximum of 256 PLCs may be connected to one PC and WinTRILOGI allows you to select any one of the PLCs for on-line monitoring and/or programming. When clicked, the password dialog box will be opened with only the ID field available for you to enter the new ID value.

### Monitoring PLC's I/O Status on Ladder Diagram

- During on-line monitoring, **the logic states of any i/o of the PLC is also displayed on the ladder diagram.** An input, output, relay, timer or counter that is turned ON will have its label name highlighted in the ladder diagram. This feature helps greatly in debugging and understanding the logical relationship between each i/o. For example, from the above figure, we can see clearly how the "Self-latching" circuit for relay "Run" works. When we first turn ON the "Start" input, "Run" will be energized and its contact, which is parallel to "Start", will hold itself in the ON state, even if we subsequently turn OFF the "Start" input.



- Note that whether the highlight is turned is based strictly on the logic state of an element. You will have to interpret whether the contact is opened or closed by examining if it is a normally-open (N.O.) or a normally-closed (N.C.) contact. A highlighted N.C. contact means that the contact is opened, whereas a highlighted N.O. contact means that the contact is closed.

# The Simulate Menu

---

WinTRiLOGI allows you to perform almost 100% simulation of your PLC's program off-line on your PC. This is a great tool for testing a program quickly before a machine has been manufactured. It is also a wonderful tool for all new PLC programmers to practice their ladder logic programming skill without the need to purchase a PLC test station.

WinTRiLOGI automatically compiles a ladder program before activating the simulator. If an error is found during compilation, the error will be highlighted where it occurs and the type of error is clearly reported so that you can make a quick correction.

## 1. Run (All I/O Reset) <Ctrl+F9>

This should be the option to use when you first begin to test your TRiLOGI program. When executed, all I/O bits (including inputs) are cleared to OFF state, all integer data are set to 0 and all string data are set to empty string. Then the "[Programmable Logic Simulator](#)" window will open for you to conduct the simulation test of your TRiLOGI ladder program.

## 2. Run (reset Except i/p) <Ctrl+F8>

Very often you may wish to keep the input settings "as is" when you reset the simulator. This situation is quite realistic in the sense that when a PLC is powered-on, some of its inputs may already be in the ON state. (e.g. sensors may detect the end positions of a cylinder rod, etc). This command allows you to preserve the logic states of all "Inputs" while resetting all other data. Note that the <Ctrl-F8> key also works in the "[Simulator](#)" screen so that at any time you can reset the simulator without affecting the logic states of the inputs.

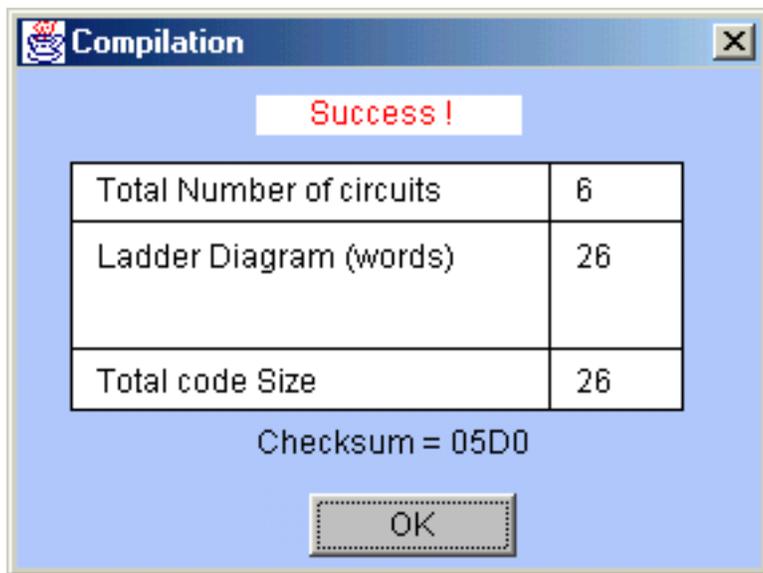
## 3. Continue Run (no reset) <F9>

Use this command to continue simulating the program since you last closed the simulator. All previous data are kept intact and will be used by the simulator to continue execution of the ladder program. If you have made changes to the ladder program, then the whole program will be recompiled before running.

Note that first scan pulse (1st.Scan) will not be activated when this command is executed since this is supposedly a continuation from the previous simulation run. This command can be useful if you have discovered a simple bug in your software during simulation, you can fix it immediately and test the effect of the change on the simulator instantly without restarting the entire simulation session from the beginning again.

## 4. Compile Only <F8>

Allows you to compile the TRiLOGI file only in order to view the compilation statistics:



## 5. Reset All I/Os <Ctrl-R>

Clears all I/Os in the simulation engine without invoking the simulator. Since all I/Os whose logic states are turned ON in the simulator will also be shown as highlight on the ladder diagram, this offers a way to clear the I/Os if it hinders your viewing of the ladder program.

# Programmable Logic Simulator

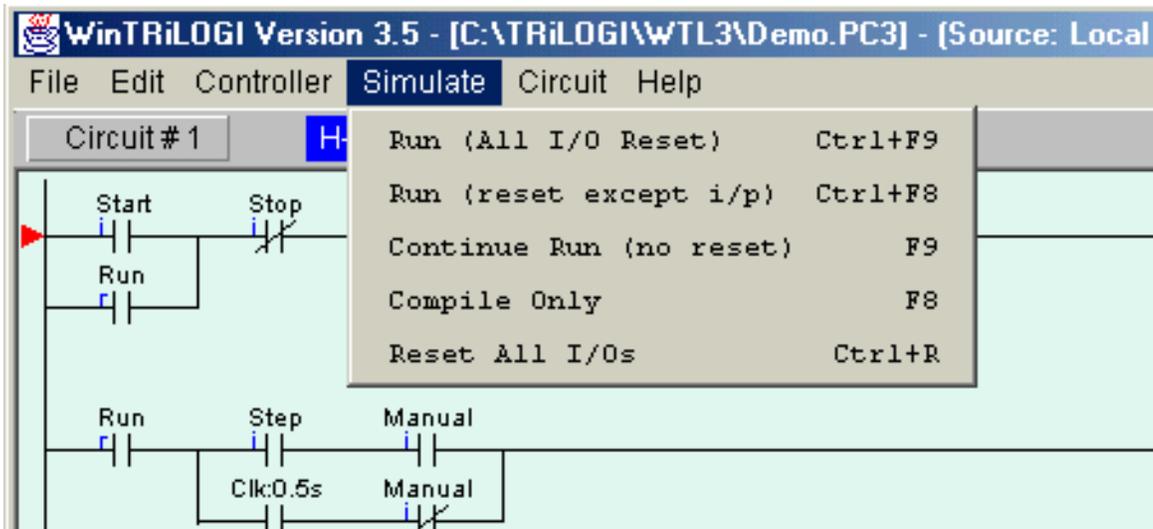
The beauty of the WinTRiLOGI is that you can test run your program **off-line** directly on the same PC that runs TRiLOGI, or connect to a real PLC located to all its real-time data. The experiences are almost identical to each other. They use the same visual feedback method about the logic states of I/O and internal data. You can test run the WinTRiLOGI program offline thanks to its built-in simulator engine which is in effect a "soft PLC" that executes all the commands that the actual "hard" PLC understands.

The simulator screen and the view variable screens are shared by both the simulator and the [on-line monitoring mode](#) and will be described in this document.

## 1. Run Simulation

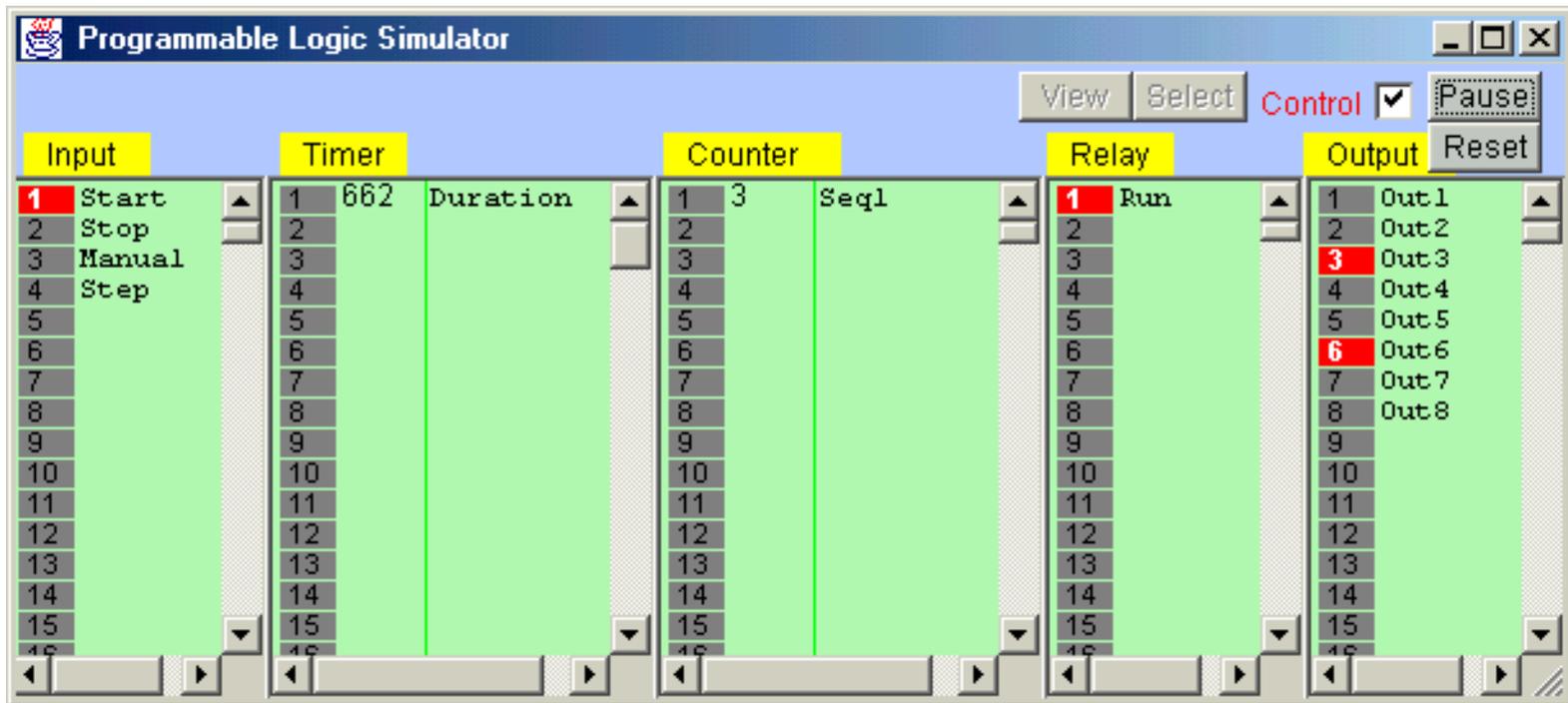
All WinTRiLOGI programs can be almost fully simulated on your PC. You invoke the Simulator by clicking on the "Simulate" Menu and select one of the three Run options:

- Run (All I/O Reset) - Clear all inputs, outputs, relays, timers and counter bits to logic OFF, clear all internal memory variables to zero and all strings to empty string before running the simulator..
- Run (reset except i/p) - The same as the above, but the input logic state will be kept. This is useful if you have simulated the program halfway and then close the simulator. You may wish to maintain the input logic states of the control scenario while resetting the rest of the simulator's data.
- Continue Run (no reset) - allows you to continue a previously aborted simulation session..



## 2. The Simulator Screen

When you run the simulator, TRiLOGI will immediately compile the ladder program and if no error is detected, it will instantly proceed to open up the "Programmable Logic Simulator" screen, as shown below:



The simulator screen comprises 5 columns: Input, Timer, Counter/Sequencer, Relay, and Output. With the exception of the Relay table which contains up to 512 elements and the timer table which contain up to 128 elements, all other columns contain 256 elements each and has a vertical scroll bar. You can use the mouse to scroll each column independently to locate the desired I/O.

The label names for the inputs, outputs, relays, timers and counters defined earlier in the I/O tables automatically appear in their respective columns. To the left of each label name column is an "LED" lamp column which indicates the ON/OFF state of the respective I/O.. A red color lamp represents the ON state of an I/O, whereas a dark grey color lamp represents an OFF state. The I/O number is indicated in the middle of the lamp.

The simulator requires the use of the **mouse** to work properly so it is important to remember the mouse button actions as follow:

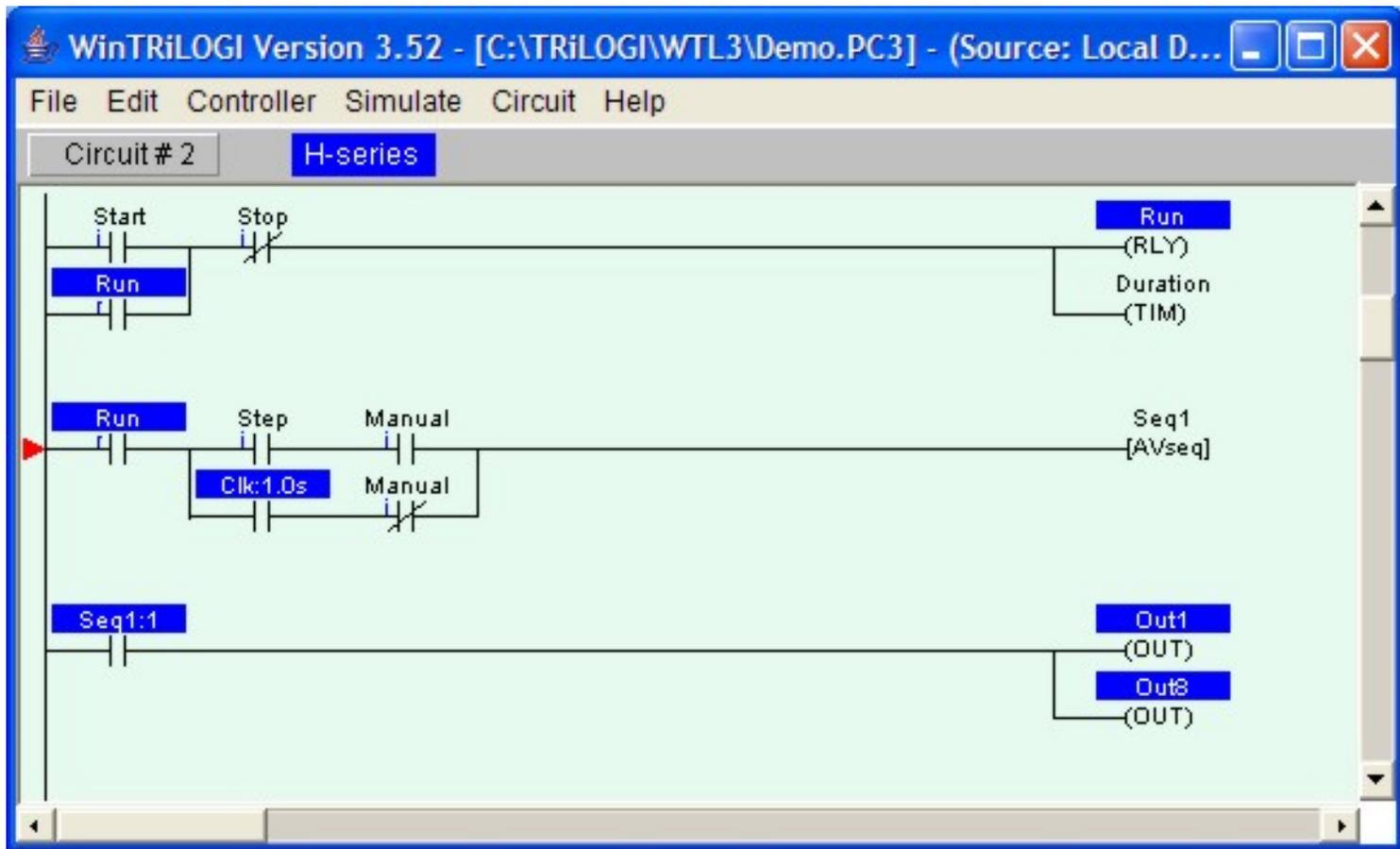
<b>Left Mouse Button</b>	Turn ON the I/O when pressed. Turn OFF when button is released.
<b>Right Mouse Button</b>	<b>Toggle</b> the I/O when pressed once. (i.e. OFF becomes ON and ON become OFF)

There is a check box with the name "**Control**" near the upper right hand corner of the simulator. This check box must be "checked" before you can force set/reset the I/O within the simulator. When running the simulator this box is normally checked. But when you run the "Full Screen Monitoring", this box is normally unchecked to avoid accidentally changing the state of a PLC's I/O.

### 3. Select, Pause, Reset Buttons

<b>Pause</b>	Halt the Simulator (including clock pulses)
<b>Reset</b>	Reset all I/Os and data in the simulator engine. (same as <Ctrl-F9>) (Press <Ctrl-F8> to reset all I/Os except the inputs)
<b>Select</b>	Not available during Simulation. See " <a href="#">On-line Monitoring</a> " for description of this button.

## 4. Displaying I/O Status on Ladder Diagram



**The logic states of any I/O can be observed on the ladder diagram directly.** An input, output, relay, timer or counter that is turned ON will have its label name highlighted in the ladder diagram. This feature helps greatly in debugging and understanding the logical relationship between each I/O. For example, from the above figure, we can see clearly how the "Self-latching" circuit for relay "Run" works. When we first turn ON the "Start" input, "Run" will be energized and its contact which is parallel to "Start" will hold itself in the ON state, even if we subsequently turn OFF the "Start" input.

Note that whether the highlight is turned is based strictly on the logic state of an element. You will have to interpret whether the contact is opened or closed by examining if it is a normally-open (N.O.) or a normally-closed (N.C.) contact. A highlighted N.C. contact means that the contact is opened, whereas a highlighted N.O. contact means that the contact is closed.

At any time you can reset all the I/Os so that they will not appear highlighted in the ladder program by pressing <Ctrl-R>.

# The Circuit Menu

---

## 1. Insert Comments

Comments are specific remarks used by a programmer to explain various characteristics of a program segment and are ignored by the compiler. WinTRiLOGI allows comments to be freely inserted between circuits. Execute this command and the Comment Editor will be opened. The comment editor allows you to enter any text you like that best describe the working of the circuit. All standard text editing keys, including cut and paste are applicable to the Comment Editor. When you have finished editing the comment, press <ESC> key to close it.

Once a comment has been created, it is assigned a circuit number and is treated like any other circuits. You can edit it by pressing the <spacebar> when you are in Browse mode, alternatively, you can move it around, copy it to another destination or delete it entirely using commands in the "Circuit" menu.

## 2. Insert Circuits

This command enables you to insert a new circuit just before the currently selected circuit. The current circuit number will be increased by one while the new circuit will assume the current circuit number. You will be placed in the [circuit editing mode](#) for immediate circuit creation.

## 3. Move Circuit

You can rearrange the order of the circuits by using this command. Select the circuit you wish to move and execute the "Move Circuit" command, then select a destination circuit location and press <Enter>. The selected circuit will be moved to the new location before the destination circuit.

Note that if you wish to move a block of circuits to a new location, you may find it more productive to use the "Cut Circuit" and "Paste Circuit" commands in the "Edit" menu.

## 4. Append Circuit

Execute this to add a new circuit to the ladder logic program. This new addition will be positioned immediately after the last circuit in the entire program.

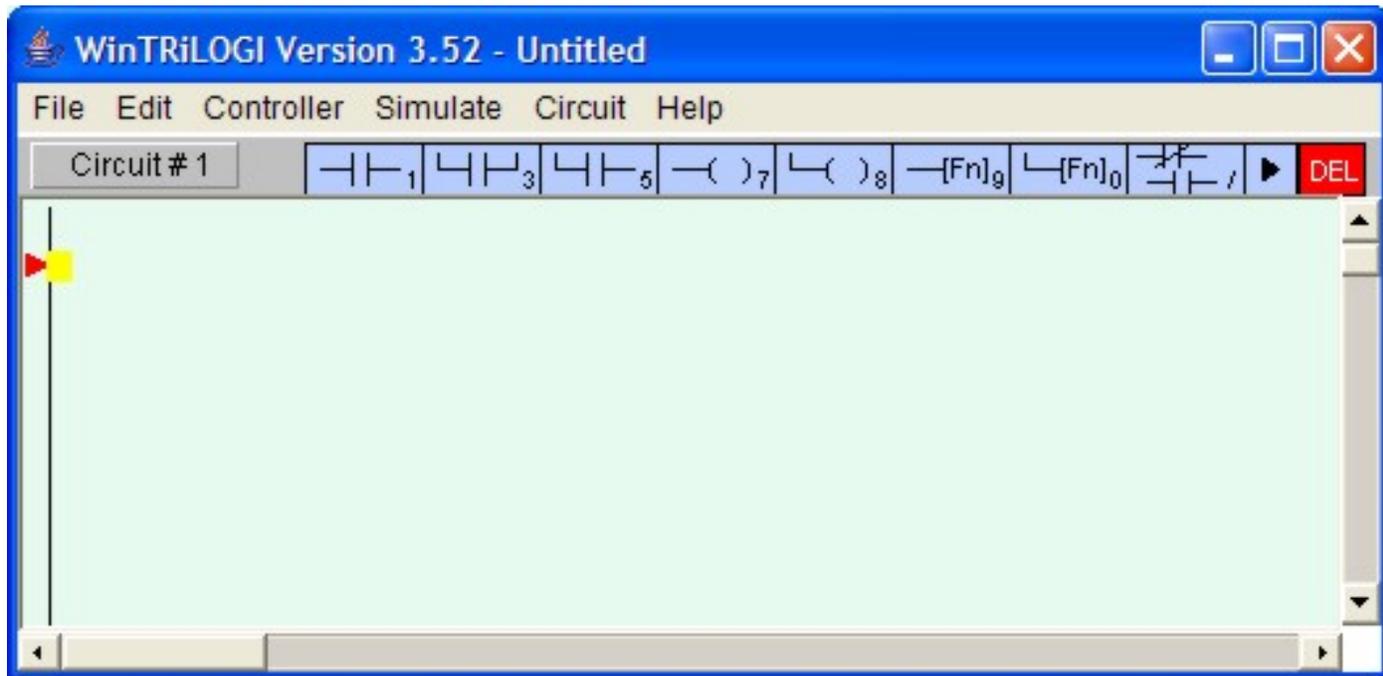
## 5. Delete Circuit

This command allows you to delete the one or more circuits. You will be prompted to enter the range of circuits that you wish to delete. **Please note that you can't UNDO a delete circuit operation.**

# TRiLOGI Ladder Logic Editor

## The Circuit Editing Mode

TRiLOGI comes with a smart editor which allows you to insert or delete a single element within a circuit easily. The editor interprets your circuit immediately upon entry and prevents you from creating illegal circuit connections. The functions of various keys in the circuit editing mode are detailed below. You know that you are in the circuit editing mode when a row of ladder logic icons appears along the upper status line next to the circuit number and a yellow color highlight bar appears and you can move it to select an element in the ladder circuit, as shown below:

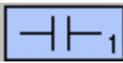
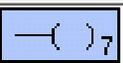


### Mouse Actions

**Left Click** - When you click on an element using a the left mouse button, the element is selected and highlighted by the yellow color highlight bar.

**Right Click** - When you click on an element using the right mouse button, you are allowed to directly edit the label name of the element. This can be a convenient feature if you need to change one or two characters in the name only.

**Insert Ladder Element** - You create the ladder circuit element simply by moving the mouse pointer to the icon and pressing either the left or the right mouse button to insert a ladder logic element to the currently highted element. The following is a description of the functions of each icon. A yellow color highlight bar will appear which you can move to select an element in the ladder circuit.

	<1> - Left click to insert a normally-open series contact. <2> - Right click to insert a normally-closed series contact.
	<3> - Left click to insert a N.O. parallel contact to highlighted element <4> - Right click to insert a N.C. parallel contact to highlighted element
	<5> - Left click to insert a N.O. parallel contact to enclose one or more elements. <6> - Right click to insert a N.C. parallel contact to enclose one or more elements.
	<7> - Insert a normal coil which may be an output, relay, timer or counter.
	<8> - Insert a parallel output coil (not an entire branch) to the current coil.

	<9> - Insert a special function coil which includes execution of CusFn
	<0> - Insert a parallel special function coil to the current coil.
	</> - Invert the element from N.O. to N.C. or from N.C. to N.O.
	Click to move the highlight bar to the right (same effect as pressing the right arrow key). This can be used to move the cursor to a junction which cannot be selected by mouse click.
	<b>Double-click</b> to delete a highlighted element. This acts as a safety against mistake.

When you click on an icon, for example, the . The icon will change to bright yellow color to show you the element type that you are creating. At the same time, an I/O table should appear on the screen with a light beige-color background. The I/O table acts like a pop-up menu for you to pick any of the pre-defined label name for this contact. This saves you a lot of typing and at the same time eliminates typo errors that could result in a compilation failure. You should spend a few minutes to follow the "[Ladder Logic Programming Tutorial](#)" on the steps needed to create a ladder program.

As mentioned previously, the ladder editor is intelligent and will only accept an action that can result in the creation of a correct ladder element. Otherwise it will simply beep and ignore the command.

### UNDO Circuit Editing

If you have wrongly inserted or deleted an element and wish to undo the mistake, you can either select "Undo" from the "Edit" menu or press <Ctrl-Z> key to undo the last step. The undo buffer stores the last 10 editing steps. You can also choose to abort all the operations on the current circuit by selecting "Abort Edit Circuit" to abort all changes made to the current circuit.

## Create Ladder Circuit Using The Keyboard

Users of existing TRiLOGI version 3.x or 4.x who are familiar with creating ladder programs using the keyboard will be delighted to know that they can still create their ladder programs using the keyboard. The keyboard actions are described below:

### Left/Right/Up/Down cursor keys

The cursor keys are for moving the highlight bar from one element to another in their four respective directions. You can only move in a direction which will end up with an element.

### <ESC>

Press <ESC> key to end the circuit editing mode and return to the browse mode of the logic editor.

### <Enter>

When you are done with editing the current circuit, hit <Enter> to proceed to the next circuit.

### <SHIFT> or <TAB>

If you observe the highlight bar carefully, you will notice a dark green color square at the right end of the highlight. This indicates the insertion location where a series contact will be attached. You can toggle the insertion location between the left or the right of the highlight bar by pressing the <SHIFT> key once. (Prior to WinTRiLOGI version 3.52 only the <TAB> key can be used. However, JRE 1.4.2 does not allow <TAB> key to be used in this manner, hence <SHIFT> key is now added to replace the <TAB> key).

The position of the cursor has no effect when you connect a parallel contact to the highlighted element. The left terminal of the element will always be connected to the left side of the parallel branch.

### **<0> to <9> , </> & <E> keys**

Pressing the key <0> to <9> and </> is equivalent to clicking on the icon shown in the table. The equivalent keyboard number is shown as a small numeral at the lower right corner of the icon. The </> key is the quickest way of converting a normally-open contact to a normally-closed one (and vice versa).

Pressing the <E> key when a contact or coil is selected allows you to edit the label name directly. Note that it is the user's responsibility to ensure that the label is valid.

# WinTRiLOGI Ladder Logic Editor

WinTRiLOGI's ladder logic editor window lies between the main menu bar along the top of the screen and the help message line along the bottom of the screen. The cursor will appear in the window whenever you are in the logic editor. The ladder logic editor comprises two modes: the [browse mode](#) and the [circuit editing mode](#). We shall explain the operation of both modes

## I. The Browse Mode

You are normally in the browser mode when you start up the program. The browse mode allows you to manipulate the whole ladder logic circuit as a single entity: you can view any circuit, make copies of it, move it to another location or delete it entirely. Each complete ladder logic "circuit" is given a circuit number. You should see a small red color marker showing you the currently selected circuit. The circuit number of the selected circuit is shown on the upper status line as "Circuit # xxx".

### Mouse Actions

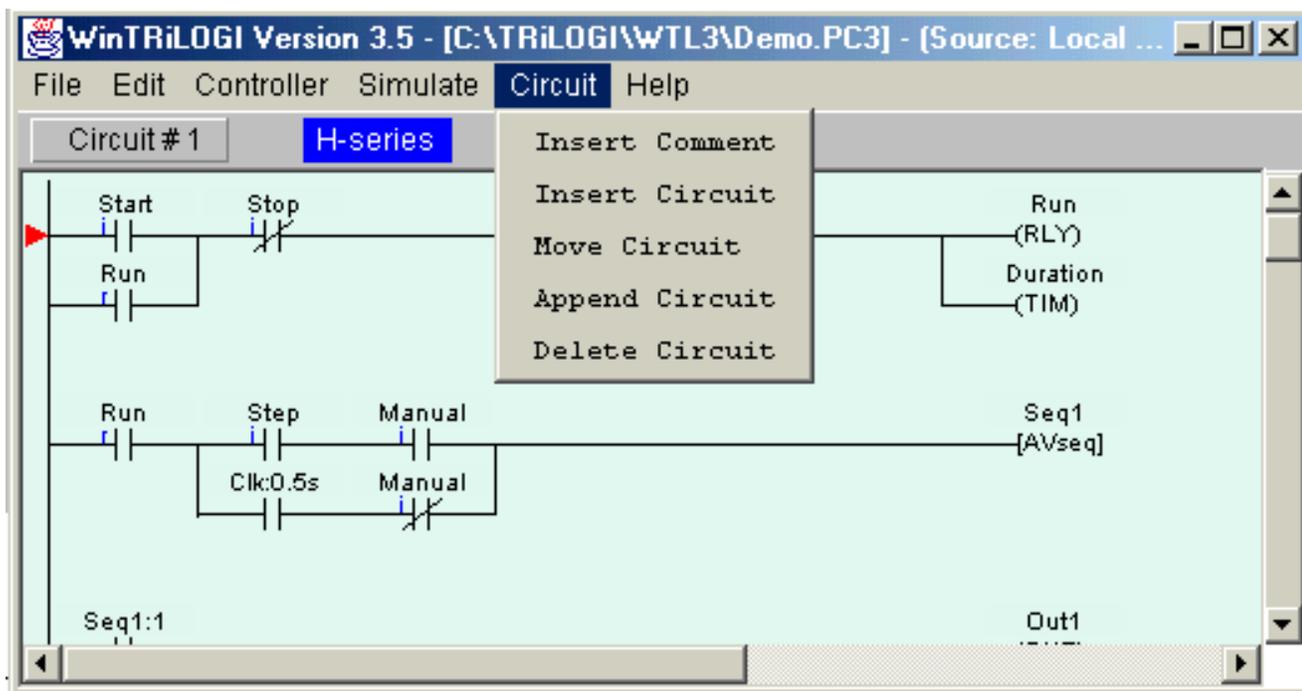
Since WinTRiLOGI Version 3.5 runs under windowing environment, all usual mouse action applies. You can grab the vertical scroll bar to scroll to your desired circuit and click on it to select it. Double click on a circuit enters the Circuit-Editing Mode which will be described later.

### Keyboard Actions

The functions of various keys in the browse mode are explained below:

1. **<Spacebar>** - Allows you to enter circuit editing mode for the currently selected circuit. If the selected circuit is a comment circuit, the comment editor will be opened automatically.
2. **<F1>** - Activates the help function to display on-line help.
3. **<F2>** - Opens the I/O Table to create the I/O Label Names
4. **<F3>** - Turns ON/OFF display of the I/O type for ladder logic contacts on the screen. All ladder logic contact symbols are normally identified by their label names. However, you can also display an optional small literal to indicate the I/O types. e.g. i=input, o=output, r= relay, t= timer and c=counter.
5. **<F5>** - Refreshes the display. If for some reason the screen is garbled by incomplete circuit display, you can just press the <F5> key to redraw the screen.
6. **<F8>** - Compiles the TRiLOGI program to show the compilation statistics.
7. **<F9>** - Runs the simulator without resetting any I/O.
8. **<Ctrl-F9>** - Resets all I/Os and then runs the simulator.
9. **<Ctrl-F8>** - Resets all I/Os except inputs and then runs the simulator.
10. **<Up>/<Dn><PgUp> and <PgDn> keys** - Use the up/down cursor keys to move the marker to other circuits and the "Circuit #" display at the upper status line will simultaneously reflect the change. If you attempt to venture beyond the screen, the logic editor screen will scroll. The <PgUp> and <PgDn> keys can be used to scroll one page at a time.

## II. The Circuit Menu



### Insert Comments

Comments are specific remarks used by a programmer to explain various characteristics of a program segment and are ignored by the compiler. WinTRiLOGI Version 3.5 allows comments to be freely inserted between circuits. Execute this command and the Comment Editor will be opened. The comment editor allows you to enter any text you like that best describe the working of the circuit. All standard text editing keys, including cut and paste are applicable to the Comment Editor. When you have finished editing the comments, press <ESC> key to close it.

Once a comment has been created, it is assigned a circuit number and is treated like any other circuits. You can edit it by pressing the <spacebar> when you are in Browse mode, alternatively, you can move it around, copy it to another destination or delete it entirely using commands in the "Circuit" menu.

### Insert Circuit

This command enables you to insert a new circuit just before the currently selected circuit. The current circuit number will be increased by one while the new circuit will assume the current circuit number. You will be placed in the circuit editing mode for immediate circuit creation.

### Move Circuit

You can rearrange the order of the circuits by using this command. Select the circuit you wish to move and execute the "Move Circuit" command, then select a destination circuit location and press <Enter>. The selected circuit will be moved to the new location before the destination circuit.

Note that if you wish to move a block of circuits to a new location, you may find it more productive to use the "Cut Circuit" and "Paste Circuit" commands in the "Edit" menu.

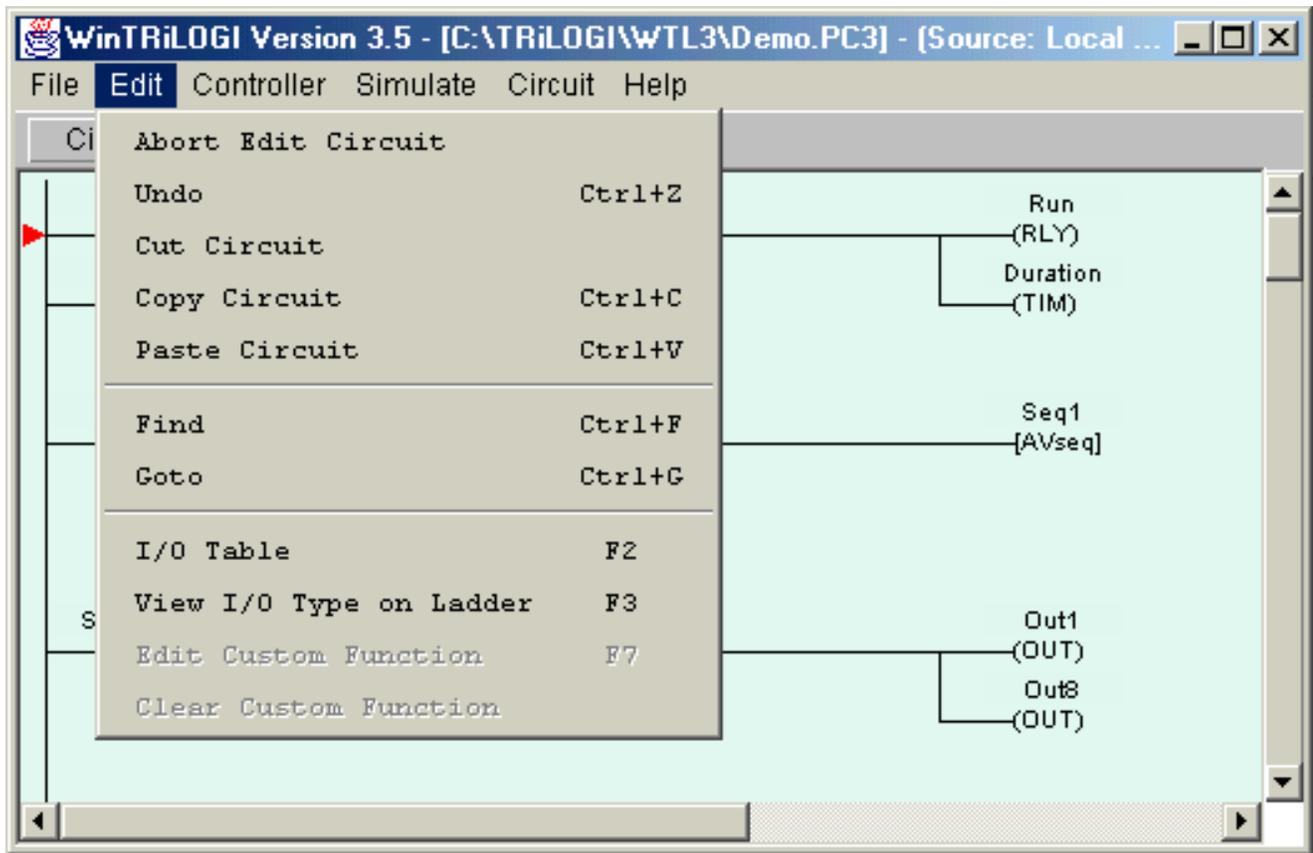
### Append Circuit

Execute this to add a new circuit to the ladder logic program. This new addition will be positioned immediately after the last circuit in the entire program.

### Delete Circuit

This command allows you to delete the one or more circuits. You will be prompted to enter the range of circuits that you wish to delete. **Please note that you can't UNDO a delete circuit operation.**

### III. The Edit Menu



#### **Cut Circuit**

You can remove a number of circuits from the current ladder program and store them temporarily in the clipboard for pasting into another part of the present ladder program or into another file altogether. In other words, it lets you move a block of circuits from one part of the ladder program to another part or into another file. **Please note that you can't UNDO a Cut Circuit operation.** However, if you do make a mistake you can always paste it back in its original position.

#### **Copy Circuit (Ctrl-C)**

You can copy a block of circuits from the current ladder program and store them into the clipboard for pasting into another part of the present ladder program or into another ladder program file altogether.

#### **Paste Circuit (Ctrl-V)**

When you execute this command, the block of ladder circuit which you "Cut" or "Copy" into the clipboard will be pasted just before the currently selected circuit. The current circuit number will be adjusted to reflect the change.

#### **Find (Ctrl-F)**

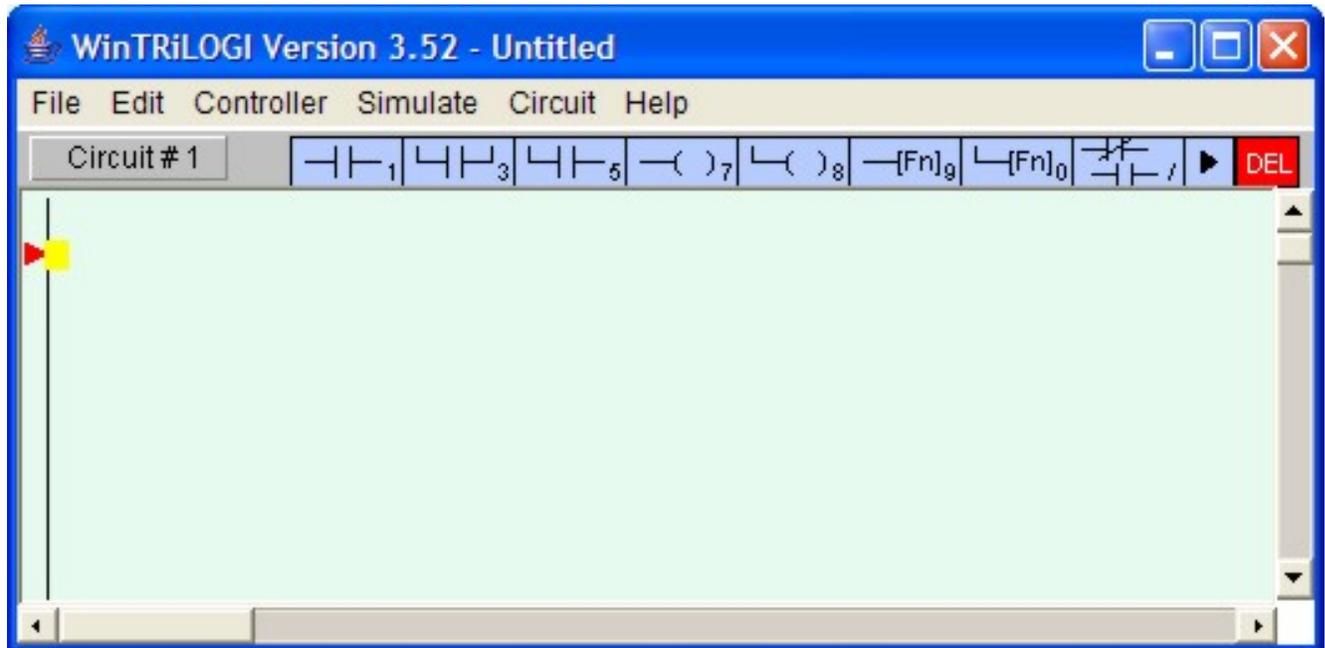
The Find command allows you to quickly locate a ladder logic circuit that contains a particular label name.

#### **Goto (Ctrl-G)**

Use this command to move towards a specific circuit number. The "Goto" command is particularly useful if your program contains many circuits, and it is inconvenient to search for a particular circuit using the mouse or the cursor keys.

### IV. The Circuit Editing Mode

TRiLOGI comes with a smart editor which allows you to insert or delete a single element within a circuit easily. The editor interprets your circuit immediately upon entry and prevents you from creating illegal circuit connections. The functions of various keys in the circuit editing mode are detailed below. You know that you are in the circuit editing mode when a row of ladder logic icons appears along the upper status line next to the circuit number and a yellow color highlight bar appears and you can move it to select an element in the ladder circuit, as shown below:



### Mouse Actions

**Left Click** - When you click on an element using a the left mouse button, the element is selected and highlighted by the yellow color highlight bar.

**Right Click** - When you click on an element using the right mouse button, you are allowed to directly edit the label name of the element. This can be a convenient feature if you need to change one or two characters in the name only.

**Insert Ladder Element** - You create the ladder circuit element simply by moving the mouse pointer to the icon and pressing either the left or the right mouse button to insert a ladder logic element to the currently highted element. The following is a description of the functions of each icon. A yellow color highlight bar will appear which you can move to select an element in the ladder circuit.

	<1> - Left click to insert a normally-open series contact. <2> - Right click to insert a normally-closed series contact.
	<3> - Left click to insert a N.O. parallel contact to highlighted element <4> - Right click to insert a N.C. parallel contact to highlighted element
	<5> - Left click to insert a N.O. parallel contact to enclose one or more elements. <6> - Right click to insert a N.C. parallel contact to enclose one or more elements.
	<7> - Insert a normal coil which may be an output, relay, timer or counter.
	<8> - Insert a parallel output coil (not an entire branch) to the current coil.
	<9> - Insert a special function coil which includes execution of CusFn
	<0> - Insert a parallel special function coil to the current coil.
	</> - Invert the element from N.O. to N.C. or from N.C. to N.O.

	Click to move the highlight bar to the right (same effect as pressing the right arrow key). This can be used to move the cursor to a junction which cannot be selected by mouse click.
	<b>Double-click</b> to delete a highlighted element. This acts as a safety against mistake.

When you click on an icon, for example, the . The icon will change to bright yellow color to show you the element type that you are creating. At the same time, an I/O table should appear on the screen with a light beige-color background. The I/O table acts like a pop-up menu for you to pick any of the pre-defined label name for this contact. This saves you a lot of typing and at the same time eliminates typo errors that could result in a compilation failure. You should spend a few minutes to follow the "[Ladder Logic Programming Tutorial](#)" on the steps needed to create a ladder program.

As mentioned previously, the ladder editor is intelligent and will only accept an action that can result in the creation of a correct ladder element. Otherwise it will simply beep and ignore the command.

### UNDO Circuit Editing

If you have wrongly inserted or deleted an element and wish to undo the mistake, you can either select "Undo" from the "Edit" menu or press <Ctrl-Z> key to undo the last step. The undo buffer stores the last 10 editing steps. You can also choose to abort all the operations on the current circuit by selecting "Abort Edit Circuit" to abort all changes made to the current circuit.

## Create Ladder Circuit Using The Keyboard

Users of DOS version of TRiLOGI version 3.x or 4.x who are familiar with creating ladder programs using the keyboard will be delighted to know that they can still create their ladder programs using the keyboard. The keyboard actions are described below:

### Left/Right/Up/Down cursor keys

The cursor keys are for moving the highlight bar from one element to another in their four respective directions. You can only move in a direction which will end up with an element.

### <ESC>

Press <ESC> key to end the circuit editing mode and return to the browse mode of the logic editor.

### <Enter>

When you are done with editing the current circuit, hit <Enter> to proceed to the next circuit.

### <SHIFT> (or <TAB>)

If you observe the highlight bar carefully, you will notice a dark green color square at the right end of the highlight. This indicates the insertion location where a series contact will be attached. You can toggle the insertion location between the left or the right of the highlight bar by pressing the <SHIFT> key once.

The position of the cursor has no effect when you connect a parallel contact to the highlighted element. The left terminal of the element will always be connected to the left side of the parallel branch.

### <0> to <9> , </> & <E> keys

Pressing the key <0> to <9> and </> is equivalent to clicking on the icon shown in the table.

The equivalent keyboard number is shown as a small numeral at the lower right corner of the icon. The </> key is the quickest way of converting a normally-open contact to a normally-closed one (and vice versa).

Pressing the <E> key when a contact or coil is selected allows you to edit the label name directly. Note that it is the user's responsibility to ensure that the label is valid.

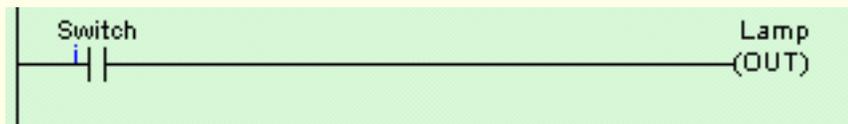
# Ladder Logic Fundamentals: Contacts, Coils, Timers and Counters

## Contacts

Ladder logic programs mimic the electrical circuit diagrams used for wiring control systems in the electrical industry. The basic purpose of an electrical control system is to determine whether a load should be turned ON or turned OFF, under what circumstances and when it should happen. To understand a ladder program, just remember the concept of current flow - a load is turned ON when the current can flow to it and is turned OFF when the current could not flow to it.

The fundamental element of a ladder diagram is a "Contact". A contact has only two states: open or closed. An open contact breaks the current flow whereas a closed contact allows current to flow through it to the next element. The simplest contact is an On/OFF switch which requires external force (e.g. the human hand) to activate it. Limit switches are those small switches that are placed at certain location so that when a mechanical device moves towards it, the contact will be closed and when the device moves away from it, the contact will be open.

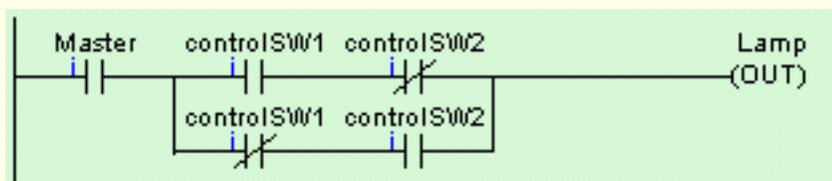
If a contact is connected to a load and the contact is closed, the load will be turned ON. This simple concept can be illustrated by the most basic ladder diagram as follow:



The vertical line on the left is the "Power" line, current must flow through the "Switch" contact in order to turn ON the load "Lamp". (In fact, there should be a second vertical line on the right end of the ladder diagram to provide a return path for the current flow, but this is omitted to simplify the circuit diagram). Now, if instead of wiring the switch to the lamp directly as suggested in the above diagram, you could connect the switch to the PLC's **input** and connect the lamp to the PLC's **output**, and then write the above ladder program to perform the same job. Of course it makes little sense to use a PLC if that is all you want to do. We will see how a PLC can simplify wiring shortly.

**Note:** The contact "Switch" shown in the above diagram is termed a **Normally-open (N.O.) contact**.

Now, let's say if there are 3 switches that must work together to control the lamp. A Master switch must be ON, and one of the two control switches "controlsw1" and "controlsw2" must be ON while the other must be OFF in order to turn ON the lamp (think of two-way switches in your house and you will get the idea). We can wire all 3 switches to 3 inputs of the PLC and the lamp to the output of the PLC. We can write the following ladder program to perform this task:



A contact with a "/" across its body is a **Normally-Closed (N.C.) contact**. What it means is that the ladder program is using the "inverse" of the logic state of the input to interpret the diagram.

Hence in the above ladder diagram, if "Master" and "controlSW1" are turned ON but "controlSW2" is turned OFF, the lamp will be turned ON since the inverse logic state of an OFF state "controlSW2" is true. Think of an imaginary current flowing through the "Master" contact, then through the "controlSW1" and finally through the normally-closed "controlSW2" contact to turn ON the lamp.

On the other hand, if "controlSW1" is OFF but "controlSW2" is ON, the Lamp is also turned ON because the current could flow via "Master" and then through the lower parallel branch via N.C. "controlSW1" and the N.O. "controlSW2".

**Note:** As you can see, although the switch "controlSW1" is connected to only 1 physical input to the PLC, but it appears twice in the ladder diagram. If you actually try to connect physical wires to implement the above circuits, both "controlSW1" and "controlSW2" will have to be of multiple poles type. But if you use a PLC, then these two switches only need to be of single-pole type since there is only one physical connection which is to the input terminal of the PLC. But in the ladder diagram the same contact may appear as many times as you wish as if it has unlimited number of poles.

The above example may be simple but it illustrates the basic concept of logical AND and OR very clearly. "controlSW1" and "controlSW2" are connected in series and both must be TRUE for the outcome to be TRUE. Hence, this is a logical AND connection. On the other hand, either one of the two parallel branches may be used to conduct current and hence this is a logical OR connection.

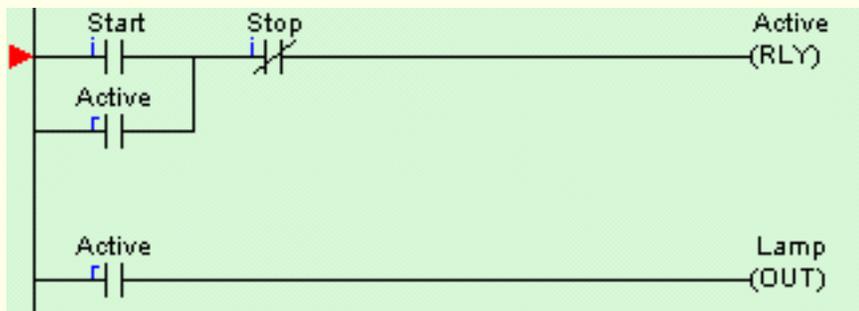
Once you understand this fundamental principle of interpreting a ladder diagram, everything should become clearer and simpler.

## Relay Coils

A contact can also be activated by the presence of an electrical current. This makes it possible for a control system to control the turning ON or OFF of a large load by using electrical current to activate a switch that can conduct high current. The simplest form of this type of contact is a relay.

In traditional electromagnetic relay, a coil of wire is wound around an iron core which turns it into an electromagnet. When current passes through the "**coil**" the magnet is "**energized**" and the force is used to either close a contact (that makes it a normally-open contact, closed only when energized) or open it (that will be a normally-closed contact since it is closed when not energized).

Ladder Logic programming language borrows some of those terms used to describe the electromagnetic relay for its own use. You connect a relay coil to the right end of the ladder diagram just like an output, as follow:



In a PLC, there are hundreds of internal "relays" which are supposed to behave like the typical electromagnetic relay. Unlike an output (e.g. the "Lamp" output) which has a physical connection out of the PLC, when an internal relay is turned ON, it is said to be "energized" but you will not see any changes in the PLC's physical I/Os. The logic state is kept internally in the PLC. The **contact** of the relay can then be used in the ladder diagram for turning ON or OFF of other relays or outputs. A relay contact in the ladder diagram can be Normally-Open or Normally-Closed and there is no limit to the number of contacts a relay can have.

## Out Coils

A PLC **output** is really just an internal relay with a physical connection that can supply electrical power to control an external load. Thus, like a relay, an output can also have unlimited number of contacts that can be used in the ladder program.

## Timer Coils

A timer is a special kind of relay that, when its coil is energized, must **wait for a fixed length of time** before closing its contact. The waiting time is dependent on the "Set Value" (SV) of the timer. Once the delay time is up, the timer's N.O. contacts will be closed for as long as its coil remains energized. When the coil is de-energized (i.e. turned OFF), all the timer's N.O. contacts will be opened immediately.

However, if the coil is de-energized before the delay time is up, the timer will be reset and its contact will never be closed. When a last aborted timer is re-energized, the delay timing will restart afresh using the SV of the timer and **not** continue from the last aborted timing operation.

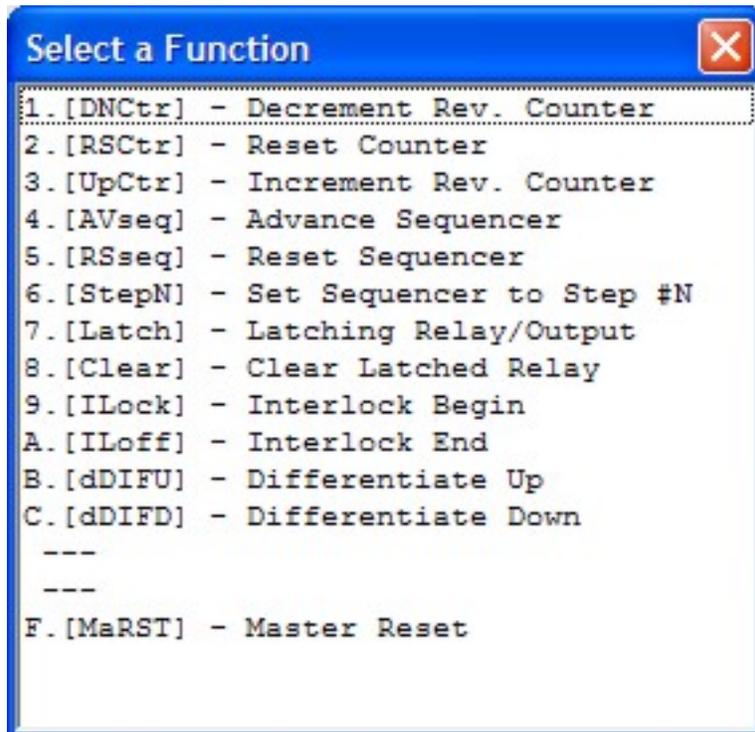
## Counter Coils

A counter is also a special kind of relay that has a programmable Set Value (SV). When a counter coil is energized for the first time after a reset, it will load the value of SV-1 into its count register. From there on, every time the counter coil is energized from OFF to ON, the counter decrement its count register value by 1. Note that the coil must go through OFF to ON cycle in order to decrement the counter. If the coil remain energized all the time, the counter will not decrement. Hence counter is suitable for counting the number of cycles an operation has gone through.

When the count register hits zero, all the counter's N.O. contacts will be turned ON. These counter contacts will remain ON regardless of whether the counter's coil is energized or not. To turn OFF these contacts, you have to reset the counter using a special counter reset function [\[RSctr\]](#).

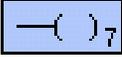
# Special Functions

During [ladder circuit editing](#), when you click on the  or  icon to create a special function coil, a special function menu will pop up as shown below:



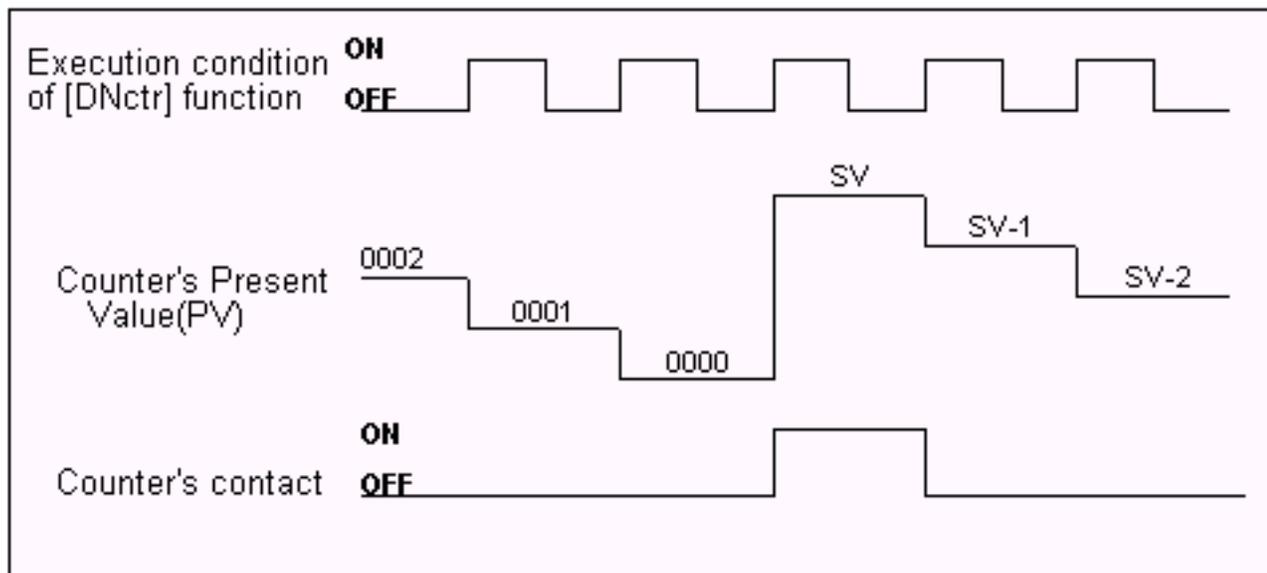
## 1. Reversible Counter Functions: [DNctr], [Upctr] and [RSctr]

The [DNctr], [UPctr] and [RSctr] functions work together to implement reversible counter functions on any of the 128 counters supported by TRiLOGI.

The ordinary down-counter (created by clicking on the  icon) essentially decrements the counter value by 1 from the "Set Value" (SV) and will stop when its count becomes zero. Unlike the ordinary down-counter, a reversible counter is a circular counter which changes the counter present value (PV) between 0 and the SV. When you try to increment the counter past the "Set Value", it will **overflow** to become '0'. Likewise if you try to decrement the counter beyond '0', it will **underflow** to become the "Set Value".

All three counter functions [DNctr], [UPctr] and [RSctr] can operate on the same counter (i.e. assigned to the same counter label) on different circuits. Although these circuits may be located anywhere within the ladder program, it is recommended that the two or three functions which operate on the same counter be grouped together in the following order: DNctr], [Upctr] and [RSctr]. Note that **NOT** all three functions need to be used to implement the reversible counter.

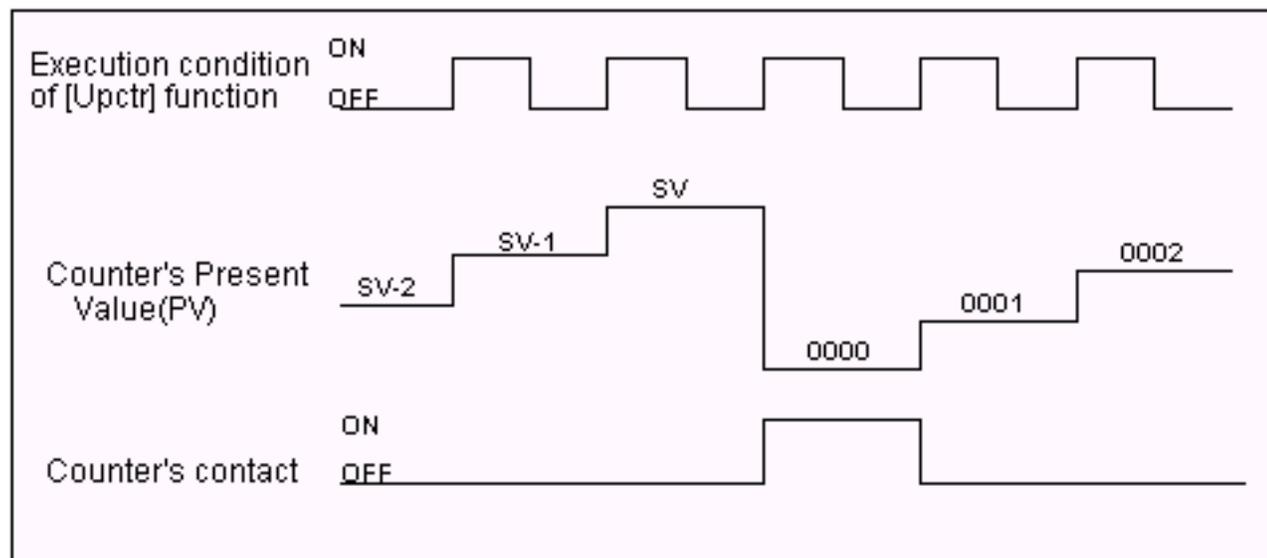
### Decrement Counter [DNctr]



Each time when the execution condition of a [DNctr] function changes from OFF to ON, the present value of the designated counter is changed as follow:

- a. If the counter's present value (PV) is inactive, load the counter register with the "Set Value" (SV, defined in the Counter table) **minus 1**.
- b. If the counter's present value (PV) is already '0', then load the counter's PV with the SV defined in the counter table and turn on the counter's contact (also known as the completion flag).
- c. Otherwise, decrement the counter PV register by 1.

### Increment Counter [Upctr]



Each time when the execution condition of an [Upctr] function changes from OFF to ON, the present value of the designated counter is affected as follow:

- a. If the counter is inactive, load the counter register with the number '0001'.
- b. If the counter's present value (PV) is equal to the Set Value (SV, defined in the Counter table), load the counter register with number '0000' and turn on the counter's contact (also known as the completion flag).

- c. Otherwise, increment the counter PV register by 1.

### **Reset Counter [RSctr]**

When the execution condition of this function changes from OFF to ON, the counter will reset to inactive state. This function is used to reset both a reversible counter and an ordinary down-counter coil.

## **2. Sequencer Functions: [AVseq], [RSseq] and [StepN]**

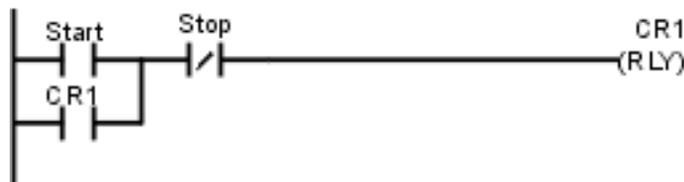
Please refer to the documentation on [Using TRiLOGI Sequencers](#)

## **3. Latch Relay Function [Latch]**

Latching relay is convenient for keeping the status of an execution condition even if the condition is subsequently removed. The program elements that are assigned as Latching Relays will remain ON once they are energized. Only Relays and Outputs may be assigned as Latching Relays.

On selecting [Latch] function, you can use the left/right cursor keys or click on the left/right arrow keys to move between the Relay and Output tables. The selected relay or output will now be assigned as a Latching Relay. You will be able to see the label name of the program element above the [Latch] symbol in the ladder diagram.

Although latch-relay can be used in place of self-latching (Seal) circuits, a latch-relay in an interlock section will not be cleared when the interlock occurs. Only a self-latching circuit as shown in the following will be cleared in an interlock section:



## **4. Clear Relay Function [Clear]**

To de-energize a program element that has been latched by the [Latch] function, it is necessary to use [Clear] function. On selecting [Clear], choose the output or relay to be de-energized. When the execution condition for that circuit is ON, the designated output or relay will be reset. In the ladder diagram, the program element label name will be shown above the [Clear] symbol.

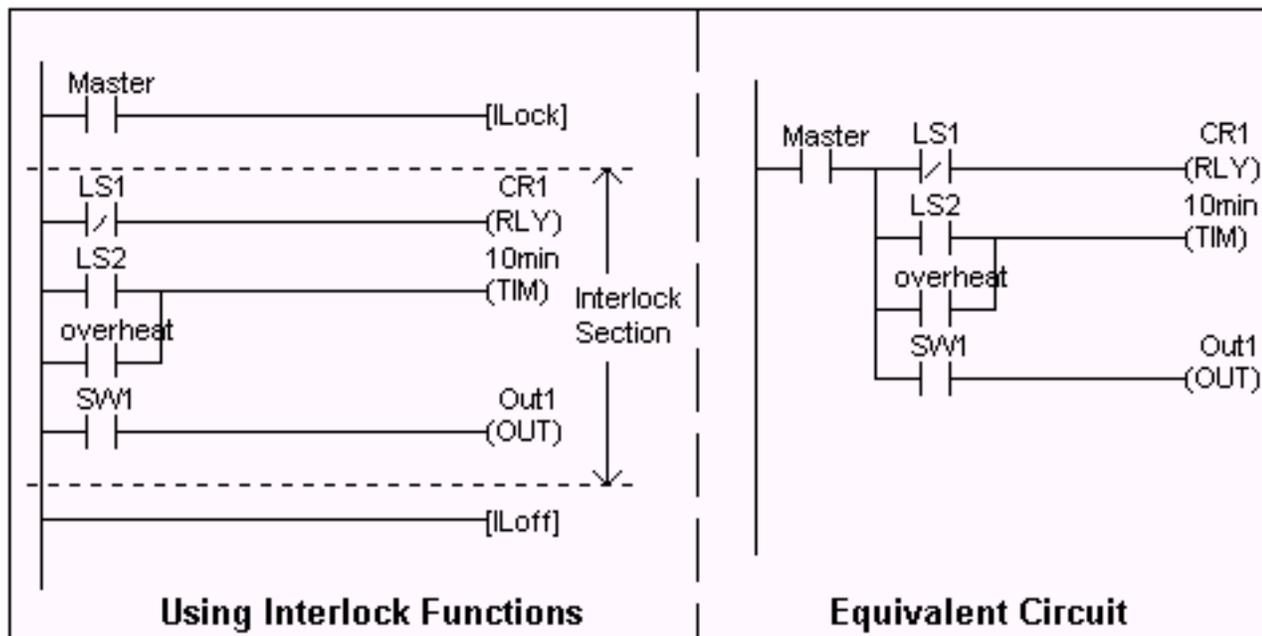
If the execution condition for [Latch] and [Clear] functions are both ON at the same time, then the effect of the designated bit depends on the relative locations of these two functions. Remember that an output or relay bit energized by [Latch] will remain ON until it is turned OFF by [Clear]. It is recommended that [Clear] circuit be placed just after the [Latch] circuit for the same output or relay controlled by these two functions. This ensures that [Clear] function has higher priority over [Latch] function, which is normally so in hardware latch-relay or other industrial PLCs.

## **5. Interlock [ILock]**

The "Interlock" [ILock] and "Interlock Off" [ILoff] functions work together to control an entire section of ladder circuits. If the execution condition of an [ILock] function is ON, the program will be executed as normal. If the execution condition of [ILock] is OFF, the program elements between the [ILock] and [ILoff] will behave as follow:

- . all output coils are turned OFF.
- b. all timers are reset to inactive.
- c. all counters retain their present values.
- d. Latched relays by [Latch] function are not affected.
- e. [dDIFU] and [dDIFD] functions are not executed.
- f. all other functions are not executed.

An Interlock section is equivalent to a master control relay controlling a number of sub-branches as follow:



Note that [LOff] is the only function that does not need to be energized by other program elements. When you use one or more [ILock] functions, there must be at least one [LOff] function before the end of the program. Otherwise the compiler will warn you for the missing [LOff]. The logic simulator always clears the Interlock at the end of the scan if you omit the [LOff] function.

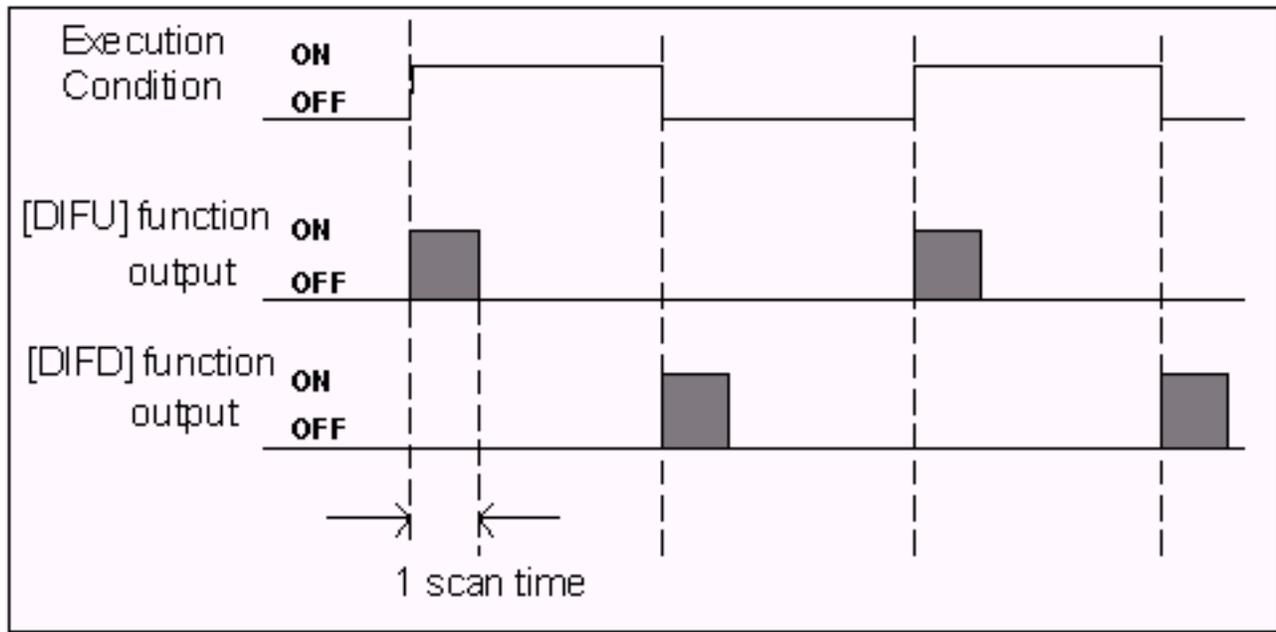
You can program a second or third level Interlock within an Interlock section using a few [ILock] functions. However, you only need to program one [LOff] function for the outermost Interlock section, i.e. [LOff] need not be a matching pair for an [ILock] function.

## 5. Differentiate Up and Down [d DIFU] and [d DIFD]

When the execution condition for [dDIFU] goes from OFF to ON, the designated output or relay will be turned ON for one scan time only. After that it will be turned OFF. This means that the function generates a single pulse for one scan time in response to the rising-edge of its execution condition. When its execution condition goes from ON to OFF nothing happens to the output or relay that it controls.

On the other hand, when the execution condition for [dDIFD] goes from ON to OFF, the designated output or relay will be turned ON for one scan time only. After that it will be turned OFF. This means that the function generates a single pulse for one scan time in response to the

trailing-edge of its execution condition. When its execution condition goes from ON to OFF, nothing happens to the output or relay that it controls.



## 6. Master Reset

An ON condition to this function clears all mailbox inputs, outputs, relays, timers and counter bits to OFF, resets all timers counters/sequencers to inactive state, and clears all latched relay bits. All integer variables will be cleared to zeros and all string variables will be assigned to empty string.

# Using TRiLOGI Sequencers

A sequencer is a highly convenient feature for programming machines or processes which operate in fixed sequences. These machines operate in fixed, clearly distinguishable step-by-step order, starting from an initial step and progressing to the final step and then restart from the initial step again. At any moment, there must be a "step counter" to keep track of the current step number. Every step of the sequence must be accessible and can be used to trigger some action, such as turning on a motor or solenoid valve, etc.

As an example, a simple Pick-and-Place machine that can pick up a component from point 'A' to point 'B' may operate as follow:

Step #	Action
0	Wait for "Start" signal
1	Forward arm at point A
2	Close gripper
3	Retract arm at point A
4	Move arm to point B
5	Forward arm at point B
6	Open gripper
7	Retract arm at point B
8	Move arm to point A

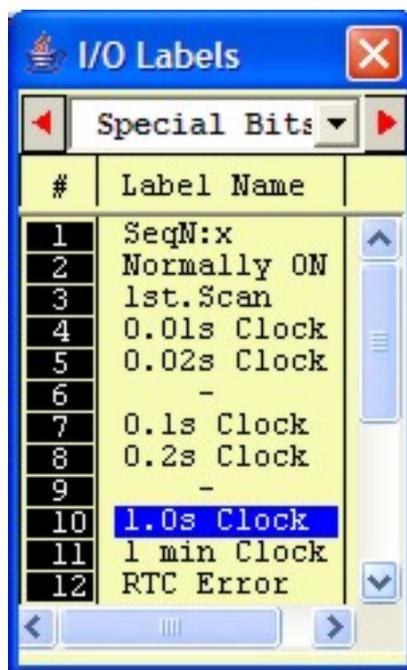
WinTRiLOGI supports **eight** sequencers of 32 steps each. Each sequencer uses one of the first eight counters (Counter #1 to Counter #8) as its step counter. Any one or all of the first eight counters can be used as sequencers "Seq1" to "Seq8".

To use a sequencer, first define the sequencer name in the Counter table by pressing the <F2> key and scroll to the Counter Table. Any counter to be used as sequencer can only assume label names "Seq1" to "Seq8" corresponding to the counter numbers. For e.g. if Sequencer #5 is to be used, Counter #5 must be defined as "Seq5". Next, enter the last step number for the program sequence in the "Value" column of the table.

Construct a circuit that uses the special function "Advance Sequencer" [AVSeq]. The first time the execution condition for the [AVseq] function goes from OFF to ON, the designated sequencer will go from inactive to step 1. Subsequent change of the sequencer's execution condition from OFF to ON will advance (increment) the sequencer by one step. This operation is actually identical to the [UPctr] instruction.

The upper limit of the step counter is determined by the "Set Value" (SV) defined in the Counter table. When the SV is reached, the next advancement of sequencer will cause it to overflow to step 0. At this time, the sequencer's contact will turn ON until the next increment of the sequencer. This contact can be used to indicate that a program has completed one cycle and is ready for a new cycle.

Accessing individual steps of the sequencer is extremely simple when programming with WinTRiLOGI. Simply create a "contact" (NC or NO) in [ladder edit mode](#). When the I/O window pops up for you to pick a label, scroll to the "Special Bits" table as follow:



The "Special Bits" table is located after the "Counters" table and before the "Inputs" table. Then click on the "SeqN:x" item to insert a sequencer bit. You will be prompted to select a sequencer from a pop-up menu. Choose the desired sequencer (1 to 8) and another dialog box will open up for you to enter the specific step number for this sequencer.

Each step of the sequencer can be programmed as a contact on the ladder diagram as "SeqN:X" where N = Sequencers # 1 to 8. X = Steps # 0 - 31.

e.g. Seq2:4 = Step #4 of Sequencer 2.

Seq5:25 = Step #25 of Sequencer 5.

Although a sequencer may go beyond Step 31 if you define a larger SV for it, only the first 32 steps can be used as contacts to the ladder logic. Hence it is necessary to limit the maximum step number to not more than 31.

## Special Sequencer Functions

Quite a few of the ladder logic special functions are related to the use of the sequencer. These are described below:

### **Advance Sequencer - [AVseq]**

Increment the sequencer's step counter by one until it overflows. This function is the identical to (and hence interchangeable with) the [\[UpCtr\]](#) function.

### **Resetting Sequencer - [RSseq]**

The sequencer can also be reset to become inactive by the [RSseq] function at any time. Note that a sequencer that is inactive is not the same as sequencer at Step 0, as the former does not activate the SeqN:0 contact. To set the sequencer to step 0, use the [StepN] function described next.

## Setting Sequencer to Step N - [StepN]

In certain applications it may be more convenient to be able to set the sequencer to a known step asynchronously. This function will set the selected sequencer to step #N, regardless of its current step number or logic state. The ability to jump steps is a very powerful feature of the sequencers.

## Reversing a Sequencer

Although not available as a unique special function, a Sequencer may be stepped backward (by decrementing its step-counter) using the [DNctr] command on the counter that has been defined as a sequencer. This is useful for creating a reversible sequencer or for replacing a reversible "drum" controller.

## Other Applications

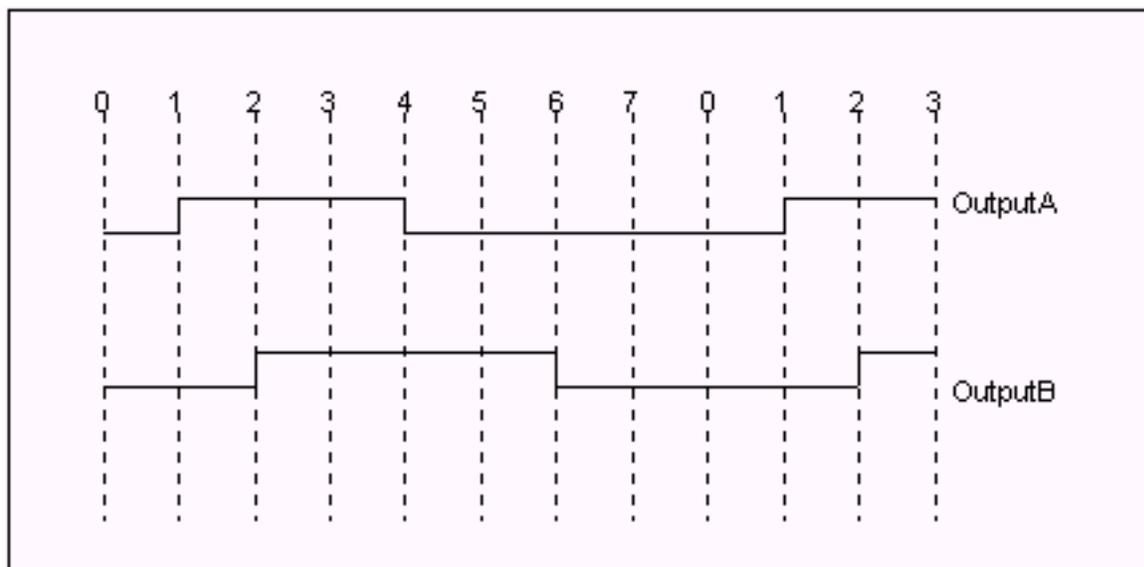
### a. Driving Stepper Motor

A sequencer may be used to drive a stepper motor directly. A two-phase stepper motor can be driven by four transistor outputs of the controller directly (for small motors with phase current < 0.5A) or via solid-state relays. The stepper motor can be driven using a sequencer that cycles through Step#0 to Step#3 (full-step mode) or Step#0 through Step#7 (half-step mode). Each step of the sequencer is used to energize different phases of the stepper motor. A clock source is needed to drive the stepper motor through its stepping sequence. The stepping rate is determined by the frequency (which is equal to 1/period) of the clock source.

Clock pulses with periods in multiples of 0.01 second can be generated easily using the "Clk:.01s" bit and an [Upctr] function. For e.g., to generate a clock source of period = 0.05s, use "Clk.01s" to feed to an [Upctr] counter with Set Value = 4. The counter's contact (completion flag) will be turned ON once every 5 counts (0,1,2,3,4), which is equivalent to a 0.05 sec. clock source.

### b. Replacing a Drum Controller

A drum controller can be replaced easily by a sequencer if the timing of the drum's outputs can be divided into discrete steps. Assuming a drum controls two outputs with the timing diagram shown in the following figure:



This can be replaced by an 8-step sequencer. Step 1 (e.g "Seq1:1") turns ON and

latch Output A using [Latch] function, Step 2 turns ON and latch Output B, Step 4 turns OFF Output A using the [Clear] function, and Step 6 turns OFF Output B. All other steps (3,5,7,0) have no connection.

### Program Example

Assume that we wish to create a running light pattern which turns on the LED of Outputs 1 to 4 one at a time every second in the following order: LED1, LED2, LED3, LED4, LED4, LED3, LED2, LED1, all LED OFF and then restart the cycle again. This can be easily accomplished with the program shown in Figure 6.9.

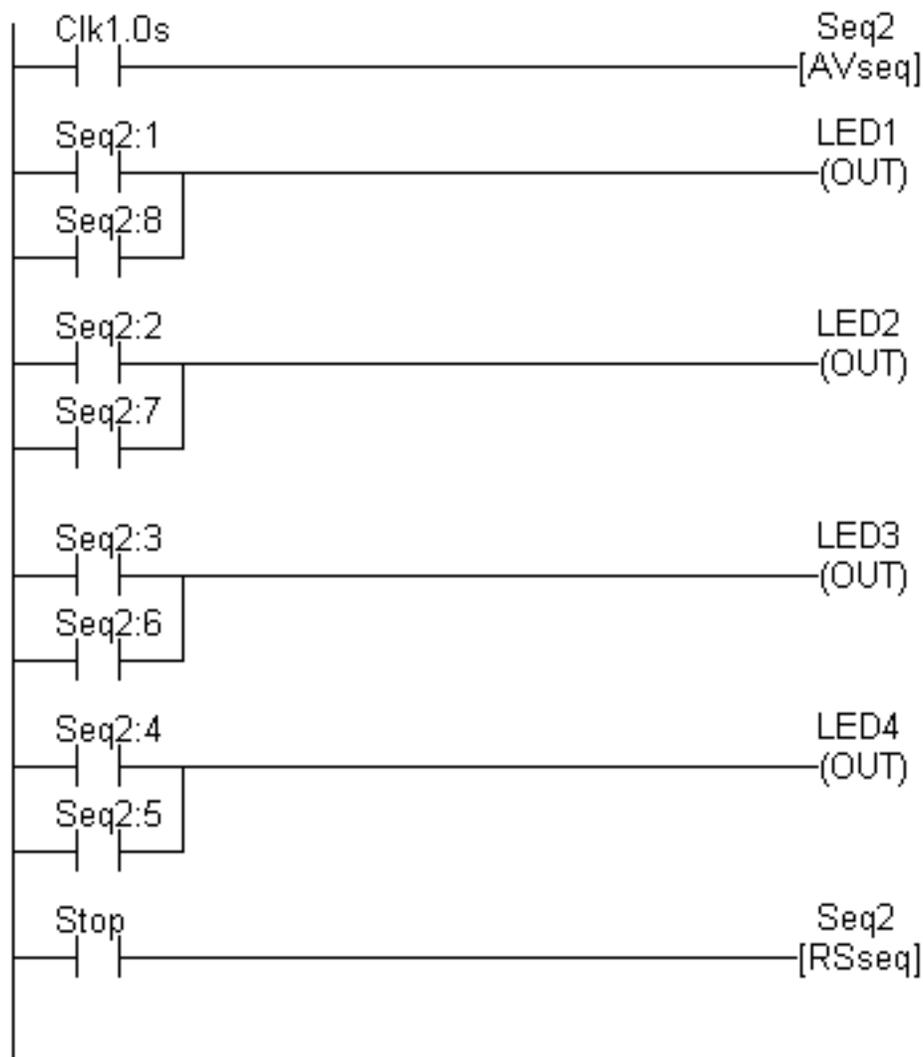
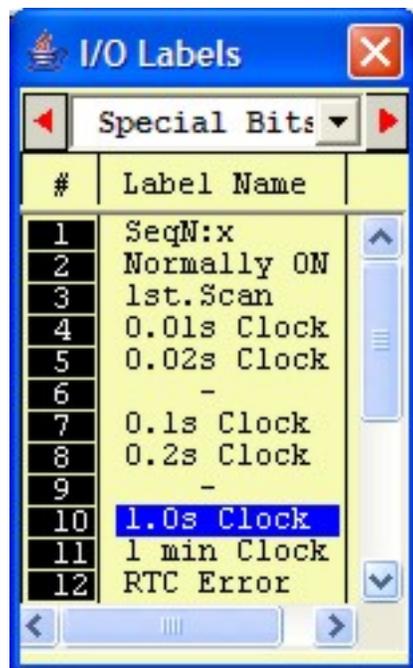


Figure 6.9

The 1.0s clock pulse bit will advance (increment) Sequencer #2 by one step every second. Sequencer 2 should be defined with Set Value = 8. Each step of the sequencer is used as a normally open contact to turn on the desired LED for the step. A "Stop" input resets the sequencer asynchronously. When the sequencer counts to eight, it will become Step 0. Since none of the LED is turned ON by Step 0, all LEDs will be OFF.

## Special Bits

TRiLOGI contains a number of special purpose bits that are useful for certain applications. These include 8 clock pulses ranging from periods of 0.01 second to 1 minute, a "Normally-ON" flag and a "First Scan Pulse", etc. To use any of these bits, enter the [ladder editor](#) and create a "contact"; when the I/O table pops up, scroll the windows until a "Special Bits" menu pops up. *This menu is located after the "Counter Table" and before the "Input" table.* as shown below:



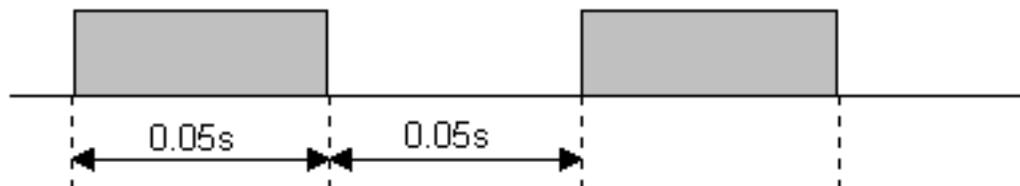
### 1. Clock pulse bits

The H-series PLC supports the following clock pulses:

Clock Pulse Period	Ladder Symbol
0.01 second	Clk:.01s
0.02 second	Clk:.02s
0.1 second	Clk:0.1s
0.2 second	Clk:0.2s
1.0 second	Clk:1.0s
1 minute	Clk:1min

A clock pulse bit is ON for the first half of the rated period, then OFF for the second half. Duty cycles for these clock pulse bits are therefore 50%, as follow:

#### Clk:0.1s (0.1 second Clock Pulse)



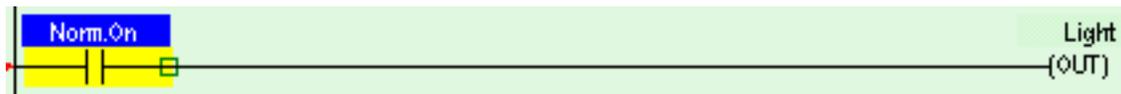
The clock pulse bits are often used with counter instructions to create timers. Additionally, they can be used as timing source for "Flasher" circuit. A reversible counter can also work with a clock pulse bit to create secondary clock pulses of periods that are multiples of the basic clock pulse rate.

## 2. SeqN:X

These are special "[Sequencer](#)" contacts which are activated only when the step counter of a Sequencer N reaches step #X. E.g. a Normally Open contact Seq2:6 is closed only when Sequencer #2 reaches Step #6. At any other step, this contact is opened. [Click this link for detailed explanation and working examples on how to use a Sequencer.](#)

## 3. Normally ON Flag - Norm.ON

You can make use of this flag if you need to keep something permanently ON regardless of any input conditions. This is because with the exception of Interlock Off function ———[Iloff], a coil or a special function is not allowed to connect directly to the power line (the vertical line on the left end of the ladder diagram). If you need to permanently enable a coil, consider using the "Normally-ON" bit from the "Special Bits" menu, as follow:



## 4. First Scan Pulse - 1st.Scan

This special bit will only be turned ON in the very first scan time of the ladder program. After that it will be permanently turned OFF. This is useful if you need to initialize certain conditions at the beginning. When the program is transferred to the PLC, this bit will only be ON when the PLC is first powered up or after it has been reset.