

# Table Of Contents

## Introduction

### Chapter 1 – Installation

- Software and Hardware requirements
- Installing the PLC Detective
- Before Running The PLC Detective For The First Time
- Running the PLC Detective
- Switching focus between the PLC Detective and CNC11 Window
- Keyboard jogging and focus

### Chapter 2 – Overview Of The PLC Detective Window

- Switching between The Source Code, Logic and Watch Views
- Graphics and color conventions

### Chapter 3 – Using The Source Code View

#### Viewing Options and Source Code Navigation

- Font Selection
- Search
- Source Code display for logic in inactive stages

#### Tools for PLC Program Development and Debug

- A note about PLC program behavior
- F1 – Run/Pause
- F2 – Step Rung
- Breakpoints
- Mouse Over (Hover)
- Mapping Word Variables to allow Viewing of System Variables, Timers and D/A Values
- Viewing Source Code in Active and Inactive Stages

### Chapter 4 – Using The Logic Diagram View

#### Viewing and Run-time Options

- F1 – Start
- F3 – Switch to Time, Switch to Event
- F5, F6 & F7 – Zoom In, Zoom Out & Zoom Reset
- F8 – Rising and Falling Edge Types
- F11 – Save
- F12 – Load
- Pre and Post Trigger Settings

#### Capturing and Analyzing PLC Events

- Setting up a PLC capture
- Viewing and Analyzing capture results

### Chapter 5 – Using The Watch View

## Introduction – CENTROID PLC Detective

Centroid's new PLC Debug tools make writing and troubleshooting PLC programs faster and easier. These new tools are based on Centroid's already powerful PLC programming source language and enable system integrators to tackle larger and more complex retrofits.

Every System Integrator who has ever tried to create or modify a PLC program for a new or modified machine knows the frustration of having run some new PLC program, wondering "Whats it doing now?". Now Centroid's "PLC Detective" allows you to clearly see the true/false state of each instruction while the machine is running, highlighted in color!

Everyone who has ever tried to debug a long term intermittent problem knows the frustration of trying to figure out what led up to a fault condition. Now Centroid's " PLC Detective" solves this problem once and for all. Set your trap , and wait , when the error occurs, go look back in time at all the I/O's leading up to the problem. This is "game changing" software for techs.

To start PLC detective from the main screen of CNC12, press **CTRL+E**, this will bring up PLC detective, allowing you to see what is active and not active in the PLC program and some other useful information regarding the PLC.

## CENTROID PLC Detective Features List

### Live Source Code Display

- Display PLC source code live on screen, while the machine is running.
- See the state of inputs, outputs, and variables by name and I/O number.
- Set live “breakpoints” on the status (true or false) of PLC statements in source code lines.
- "Mouse over" word variables to see the actual values of any word variable in real-time.
- Map word variables to System Variables, Timers and A/D readings and then mouse over the mapped word variables to view real-time values.

### Logic Analyzer with Timing Diagram display

- Capture critical events before and after an error or fault.
- Save timing diagrams for later analysis and offline viewing.
- Set traps to catch long term intermittent problems.
- View the state of I/O's and variables leading up to a problem condition
- Movable cursors automatically measure and display elapsed time between PLC events

# Chapter 1: PLC Detective Installation

## Software Requirements

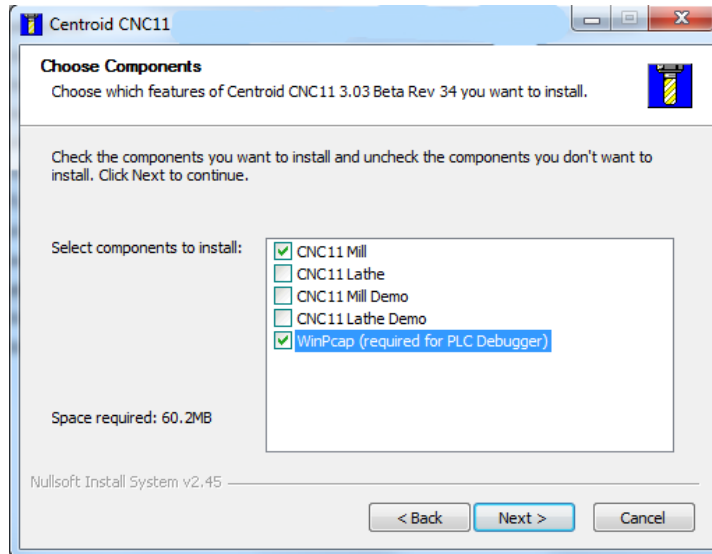
- Windows Systems: Windows 7 or higher, Centroid CNC11 v3.04 or higher

## Hardware Requirements

- Mouse Required
- 2 GB Ram
- P4 Class CPU 2Ghz or higher

## Software Installation

- CNC12: PLC Detective comes preinstalled
- CNC11 v3.16+: PLC Detective comes preinstalled
- CNC11 v3.04 – v3.14: The PLC Detective is available for installation from the CNC11 installation menu. When installing CNC11 v3.04 – v3.14, make sure that WinPcap is checked:



## Chapter 1: PLC Detective Installation (Cont'd)

### Before Running The PLC Detective for The First Time

Before running the PLC Detective for the first time, and after any PLC source code changes, the PLC program needs to be compiled/recompiled. To compile a PLC program:

- 1.) Exit CNC11 software.
- 2.) Select the “Start” button in the lower left corner of the screen.
- 3.) Select “All Programs”->“Accessories”->“Command Prompt”
- 4.) At the command prompt, type: `cd \cncm`, then press **ENTER**.
- 5.) Type “`mpucomp source.src mpu.plc`” (without quotes) where `source.src` is the name of the PLC program you wish to compile, then press **ENTER**.

Example: To compile a PLC program called `dc3iob-basic.src` type:

```
mpucomp dc3iob-basic.src mpu.plc
```

- 6.) At the command prompt, type `exit`, then press **ENTER**.
- 7.) Start CNC11 software.

### Running The PLC Detective\*

In the `C:\cncm` folder and with CNC11 running, **double Left Click** `plcdebugger.exe` or `plcdetective.exe`. In CNC11 v3.08+, the PLC Detective can be started with **CTRL+E** provided you are at the main screen and not running a job.

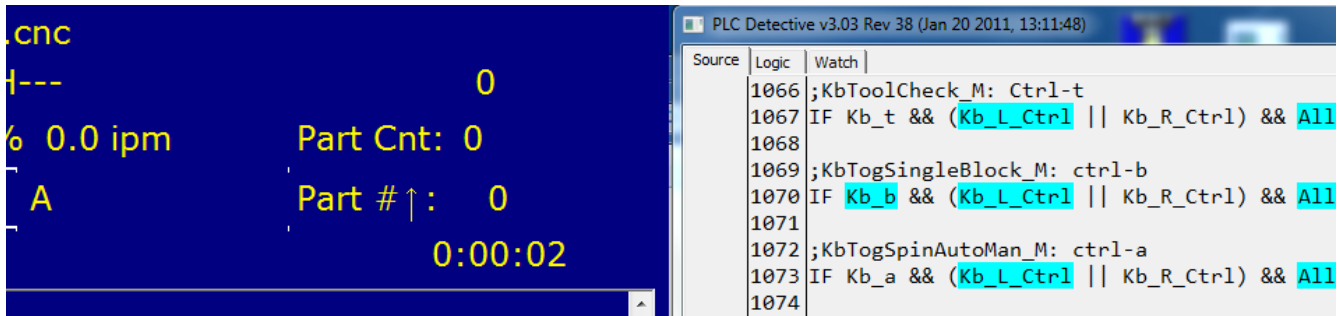
### Switching Focus Between The PLC Detective And CNC11 Window

If CNC11 and/or the PLC Detective window is running full screen, you will need to switch between windows by pressing alt-tab. Pressing alt-tab will cycle focus between all open windows. If only two windows are open, pressing alt-tab will toggle between the windows. To select from multiple windows, hold down the alt key and then press tab times until the desired window is selected and then release both keys.

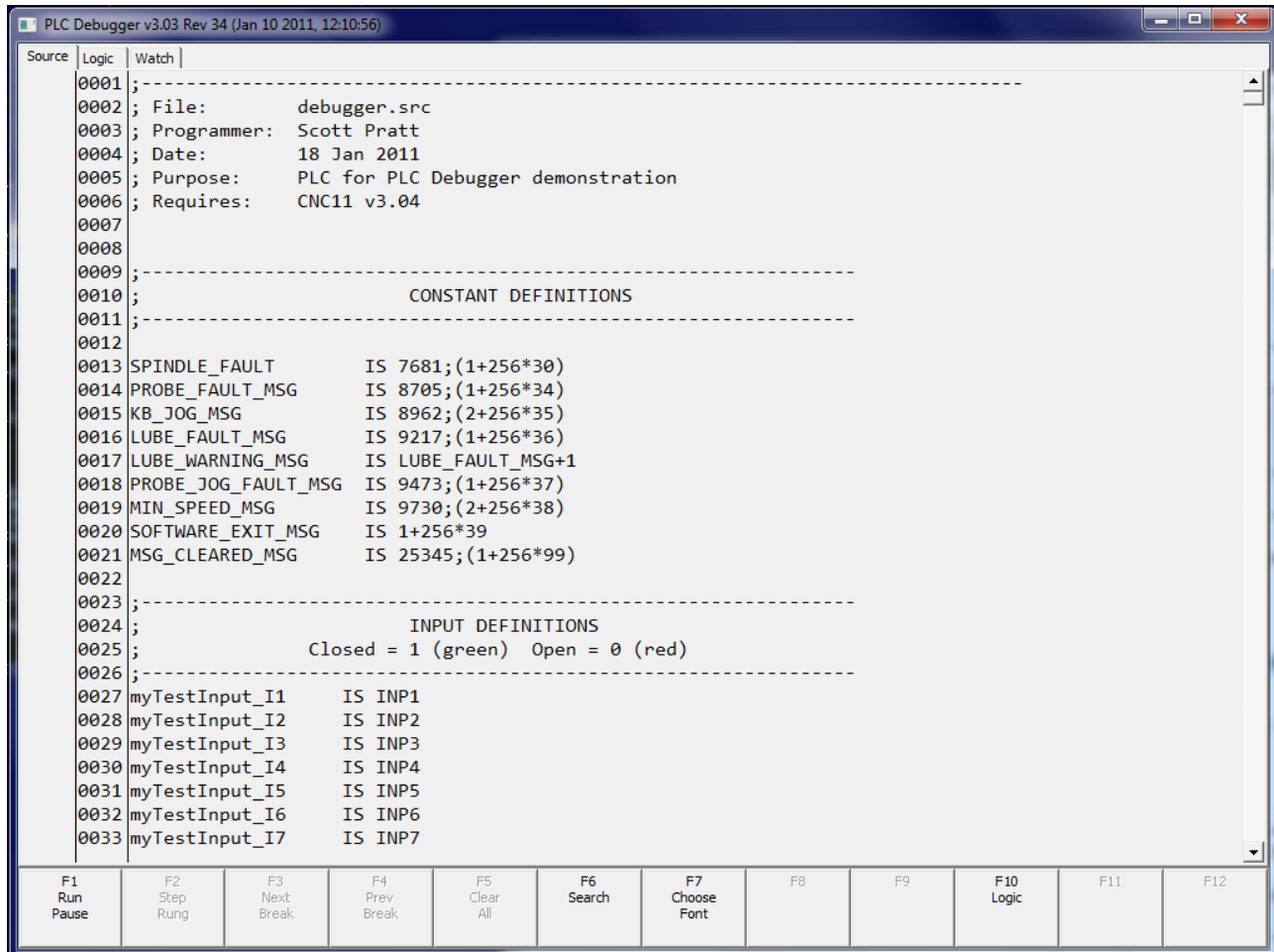


## Keyboard Jogging and Focus

Keep in mind that while focus is on the PLC Detective window, **no keyboard events are being sent to CNC11**. If PLC Detective is running in full screen, you won't be able to see keyboard events in PLC Detective, you'll need a bigger monitor and/or higher resolution settings so that you can have focus on CNC11 and still see the PLC Detective window in order to see keyboard events in PLC Detective.



## Chapter 2: Overview Of The PLC Detective Window



Three views are available in the PLC Detective window – Source, Logic and Watch. To select a view, either click on the desired tab in the top left of the window or use the **F10** key to cycle the view. To navigate vertically within a view, use scroll bar along the right side of the window.

### Graphics and Color Conventions

Execution of PLC program is paused on this line if it is highlighted in yellow

If I/O bit is TRUE (logic "1") it is highlighted in Cyan – Input = closed, output and memory bit = SET

1000 If source code is grayed and line numbers are red, the currently displayed stage is inactive

An I/O bit highlighted in Cyan, surrounded by a red border is negated (!) off or open & TRUE.

## Chapter 3: Using The Source Code View

By far the most useful view for PLC development, testing and debugging is the Source Code View. With it, you can monitor the values of word values and watch the active state of every input, output and memory location in real-time as they are being used in the PLC program. You can as well as set “Breakpoints” to stop the PLC program before or after execution of certain PLC statement(s) or when a PLC event occurs or doesn't occur.

To get started, let's go over a few of the features that help make the source code view easier to read and find our way around.

### Viewing Options and Source Code Navigation

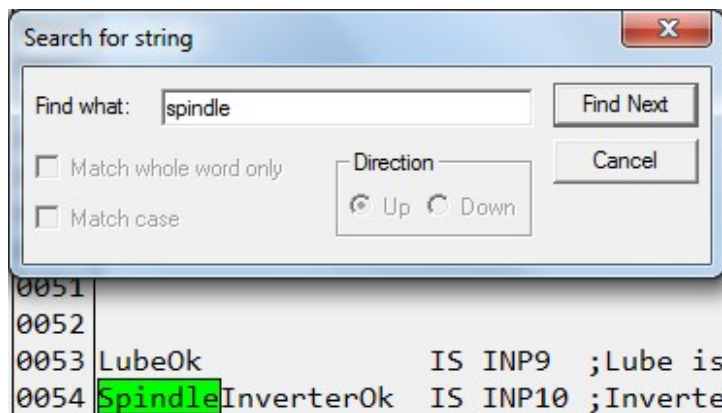
#### Font Type Selection, Size and Style

When selecting a font, we recommend using “regular”, 12pt, “Consolas” font. “Consolas” is a “mono-spaced” font which means that all the letters are spaced equally. A mono-spaced makes it much easier to read source code because the text between lines above and below each other remains aligned.

#### Search

Accessed by either pressing **F6 – Search**, or in CNC12 v5.0+ **CTRL+F**, this is without a doubt the easiest way to find that variable you are looking for. Search is case insensitive and will find words which include the string you enter -whether whole or partial to the word it occurs in.

Below is an example of the first instance found on a search for “spindle”:



**NOTE:** Starting in CNC12 v5.2 it's possible to search up and down. Any version of CNC11 or CNC12 before v5.2 can only search down. However, once it reaches the end of the source code it will wrap around to the top again and continue searching.

#### Go To Line

Added in CNC12 v5.2 pressing **CTRL+G** will allow you to specify the line you wish to jump to.

#### Source Code display for logic in inactive stages

All source code logic in an inactive stage is grayed out and the line numbers are red. Logic in an inactive stage is not acted on although the states and values of variables and I/O which are acted on in active stages will still be updated within the inactive view.

## Chapter 3: Using The Source Code View (Cont'd)

### A note about PLC program behavior

It is **important** to understand that any variables and memory bits on the current line (highlighted in yellow when paused) will not be updated until the program is allowed to continue **past** the current line. Another important factor to keep in mind is that, while memory and word variables will be updated immediately upon passing the line of code that acts on them, **Outputs are not updated until the end of the PLC program is reached, Inputs are updated only at the beginning of the PLC program.**

In Screenshot #1 below, the program is paused at a breakpoint on line #1771. Note that even though the condition for the statement on line #1771 is TRUE, (myTestInput\_I2 is closed) the action(s) performed by that statement will not be complete until stepping or moving beyond the statement that sets it.

```
1770
1771 IF myTestInput_I2 THEN SET myTestOutput_O1, myTestWord_W4 = 100, SET myTestMem_M1
1772
1773 IF !myTestOutput_O2 THEN RST myTestMem_M1, myTestWord_W4 = 0
1774
1775 IF !myTestInput_I2 THEN RST myTestOutput_O1
```

Screenshot #1

In Screenshot #2 it can be seen that, while myTestMem\_M1 and myTestWord\_W4 have been updated, the output -myTestOutput\_O1- is still off and will remain so until the end of the PLC program is reached.

```
1771 IF myTestInput_I2 THEN SET myTestOutput_O1, myTestWord_W4 = 100, SET myTestMem_M1
1772
1773 IF !myTestOutput_O2 THEN RST myTestMem_M1, myTestWord_W4 = 0
1774
1775 IF !myTestInput_I2 THEN RST myTestOutput_O1
```

Screenshot #2

This can lead to unexpected results since, as Screenshot #3 shows, myTestMem\_M1 and myTestWord\_W4 are immediately reset as a result of myTestOutput\_O1 not being set until the end of the PLC program is reached.

```
1771 IF myTestInput_I2 THEN SET myTestOutput_O1, myTestWord_W4 = 100, SET myTestMem_M1
1772
1773 IF !myTestOutput_O2 THEN RST myTestMem_M1, myTestWord_W4 = 0
1774
1775 IF !myTestInput_I2 THEN RST myTestOutput_O1
```

Screenshot #3

# Chapter 3: Using The Source Code View (Cont'd)

## Tools for PLC Program Development and Debugging

### Run/Pause

Pressing or clicking on the **F1 – Run/Pause** button toggles the PLC program between a “running” state and a “paused” state. When toggling to a paused state, the program will stop on the next appropriate Breakpoint that the PLC program encounters. If no Breakpoint(s) have been set, it will stop at the first line of the PLC program. For further details on Breakpoints, please see the “Breakpoints” section of the manual found later in this chapter.

Please keep in mind that, when a PLC program is in a paused state, none of the I/O or variables are being updated and as such you will see no changes of states or values in either the PLC Detective screens or in the real-time I/O display in CNC11.

### Step Rung

Pressing or clicking on the **F2 – Step Rung** button “steps” through the PLC program one statement at a time. The **F2 – Step Rung** button is available only when the PLC program is in the paused state.

### Breakpoints

Breakpoints are used to pause execution of the PLC program at specific location(s) and can be an invaluable tool in developing method of Breakpoints can only be set on PLC program lines that begin with “IF”.

There are three types of Breakpoints:

- 1.) **Always Break** (Blue) – Pauses execution of the PLC program at the selected statement on every pass. Double left click to SET, double left click again to clear.
- 2.) **Break On TRUE** (Green) – Pauses execution of the PLC program only when the selected statement is TRUE. Double right click to cycle from TRUE->FALSE->clear
- 3.) **Break On FALSE** (Red) - Pauses execution of the PLC program only when the selected statement is FALSE. Double right click to cycle from TRUE->FALSE->clear

```
1771 IF myTestInput_I2 THEN SET myTestOutput_01, myTestWord_W4 = 100, SET myTestMem_M1
1772
1773 IF !myTestOutput_02 THEN RST myTestMem_M1, myTestWord_W4 = 0
1774
1775 IF !myTestInput_I2 THEN RST myTestOutput_01
```

**NOTE: ALL BREAKPOINTS IN INACTIVE STAGES ARE IGNORED**

### Mouse Over

“Mousing Over” or “Hovering” over a word variable will cause a tool tip to pop up displaying the current value of the word variable. Note: Once the value is displayed, the value will not be updated unless the mouse is moved.

```
1772
1773 IF !myTestOutput_02 THEN RST myTestMem_M1, myTestWord_W4 = 0
```

100

## Chapter 3: Using The Source Code View (Cont'd)

Mapping Word Variables to allow Viewing of System Variables, Timers and A/D Values

The PLC Detective allows you to view all variable and I/O data that is sent up to the PC but it does not have the ability to directly view the values for Timers, System Variables and A (A/D) input values. Fortunately, there is a work around that provides this functionality.

By slightly modifying your source code to map a word variable to the Timer, System Variable or A/D input that you want to monitor, you can view the current value(s) for these data types. This concept is illustrated in the examples that follow.

**NOTE: This example assumes I/O mapping consistent with use of an ALLIN1DC.**

### Declaring Word Variable(s)

Like any other variable, a word variable must be declared before it can be used. The following examples will declare 3 word variables, `valueOfOneSecTimer_W1` which will be mapped to a Timer declared as `oneSecondTimer_T1`, `analogInput_W2` which will be mapped to an analog input bit, and `encoderPosition_DW1` which will be mapped to a System Variable that holds the encoder position for the sixth encoder input.

```
valueOfOneSecTimer_W1 IS W1 ;Word variable to be mapped to  
                           oneSecondTimer_T1
```

```
analogInput_W2 IS W2 ;Word variable to be mapped to the analog input.
```

```
encoderPosition_DW1 IS DW1 ;Word variable to be mapped to  
                           SV_MPU11_ABS_POS_5
```

### Mapping Word Variables (MainStage of PLC program)

```
IF TRUE THEN valueOfOneSecTimer_W1 = oneSecondTimer_T1
```

```
IF TRUE THEN BTW analogInput_W2 analogInputBit0 12
```

```
IF TRUE THEN encoderPosition_DW1 = SV_MPU11_ABS_POS_5
```

## Chapter 4: Using The Logic View

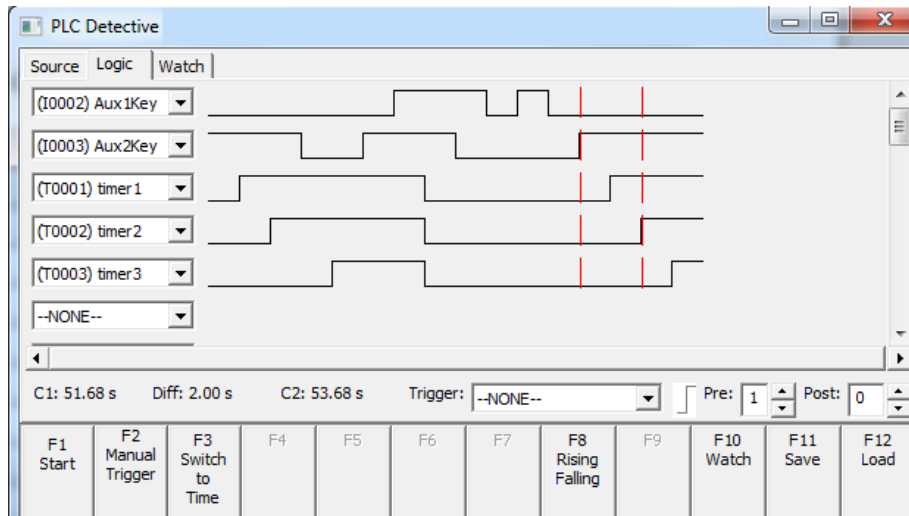


Figure 2. The Logic View.

The Logic View is used to record and analyze PLC event history. Up to 32 different PLC bits can be selected from the drop-down boxes on the left of the window. Only bit type entities and only PLC bits which have been defined in the PLC program are available to monitor. Though a Timer is not a “bit” type, the status of a Timer (TRUE or FALSE) is and thus can be monitored.

### Viewing and Run-time Options

**F1 – Start:** Starts automatic PLC Capture if a Trigger has been defined. When a capture is in progress, the button turns red, when the trigger event occurs, the button turns yellow and when the specified number of “Post” triggers events have elapsed, the button turns back to gray. Captures can be stopped at any time by pressing the F1 or F2 button at anytime.

**F2 – Manual Trigger:** Starts the PLC capture immediately turning the Manual Trigger button red. The capture runs until the Manual Trigger button is pressed again.

**F3 – Switch to Time, Switch to Event:** Switches between time based display and event based display. Default time base is 20 milliseconds.

**F5, F6 & F7 – Zoom In, Zoom Out & Zoom Reset:** Zooms in, out & resets zoom level to default.

**F8 – Rising and Falling Edge:** Begins capture on either rising or falling edge of trigger event.

**F11 – Save:** Saves currently selected PLC bits and trigger settings for re-use. It will also save any PLC event data captured at the same time.

**F12 – Load:** Loads previously saved capture settings and captured event data.

**Pre and Post Trigger Settings:** The Pre and Post trigger settings define the number of events that you wish to capture before and after the defined trigger event. When in a time based view, the values in the spinners represent the amount of time, in 20 millisecond intervals, to be captured before and after the trigger event. When in event based view, the values represent the number of PLC bit edge transitions (rising or falling as selected ) before and after the trigger event.

## Chapter 4: Using The Logic View (Cont'd)

### Setting up a PLC Capture

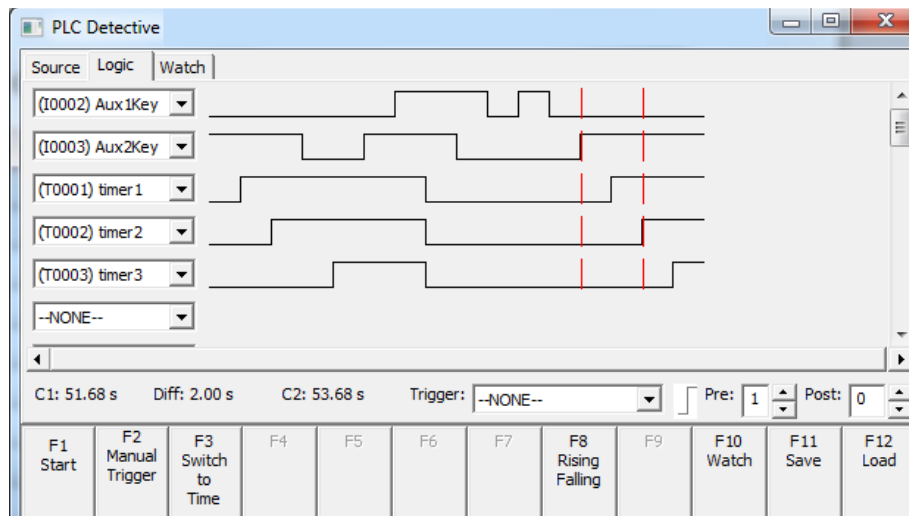
To setup a PLC:

1. Select the PLC Bits you wish to monitor from the drop-down boxes on the left of the screen.
2. Choose the number of Pre-Trigger and Post-Trigger events from the spinner boxes in the bottom right of the screen
3. Select the PLC bit you wish to trigger on in the “Trigger” drop-down box.
4. Choose Time or Event Based view by pressing **F3** – Switch to Time/Switch to Event
5. Choose Rising or Falling edge to trigger data collection with **F8**.
6. Press **F11** – **Save** if you wish to save these settings to be used later.
7. Press **F1** – **Start**. (or **F2** – **Manual Trigger** if you prefer)

Note that in Event View if the first event to occur is a Rising edge event, the event will not be recorded, meaning if that were the trigger event it will NOT start data collection. In Event view it's recommend to either use Falling edge, or to be sure at least one event occurs before the trigger if using Rising edge.

### Using The Cursors to Measure Time Between PLC Events and Total Elapsed Time

Left or Right clicking near a leading or falling edge of a PLC bit will place a cursor at that location. The “C1” and “C2” cursor indicators display the time (since the start of the capture) relating to where they are positioned. They are shown on the screen using red dashed lines, and can be re-positioned using a left mouse click for C1 and a right mouse click for C2. The “Diff” indicator is the absolute difference between the times.



## Chapter 5: Using The Watch View

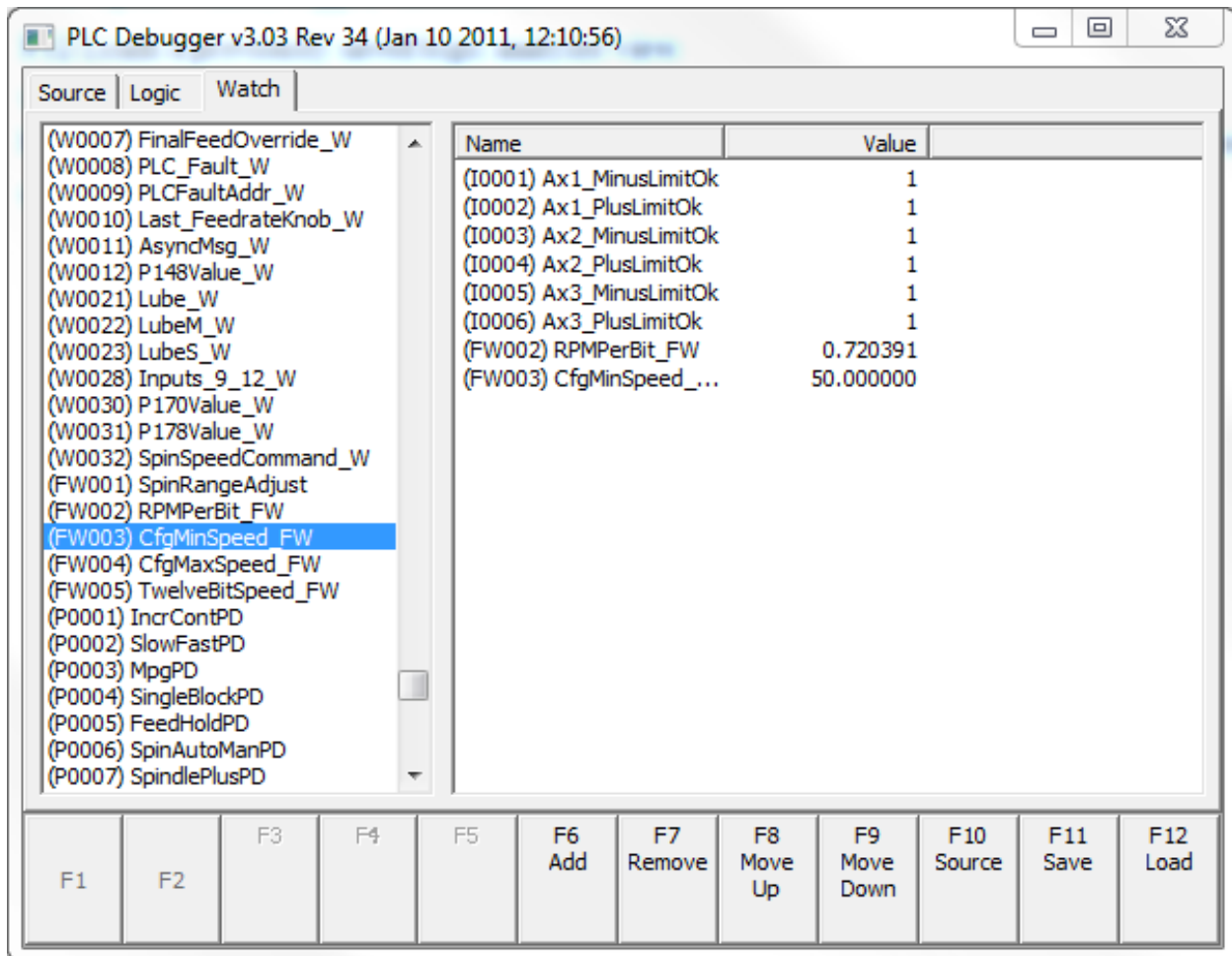


Figure 3. Watch view.

- The watch view can be used to customize the PLC bits/words that one wants to view.
- **F6 – Add** the selected PLC bit/word from the left hand side list to the right side view. A **double Left Click** can also be used.
- **F7 – Remove** the selected bit from the watch list. **ENTER** or **double Left Click** also works.
- **F8 – Move Up** the selected watched bit in the list.
- **F9 – Move Down** the selected watched bit in the list.
- **F10 – Source** code view.
- **F11 – Save** the watched bits to a file.
- **F12 – Load** the watched bits from a file.