EMUL166-OCDS-XC16*x*-PC

Getting Started Guide

Version 2.0

Table of Contents

Features of the EMUL166-OCDS-XC16*x*-PC

The Nohau EMUL166-OCDS-XC16*x*-PC emulator supports the XC161 device, which has OCDS Level 1 support. The emulator provides loading of code, Run Control, Shadow Memory and Watch Points.

The emulator connects to the target board using a 16-pin Berg connector as described by Infineon (see Figure 1 for a picture of the EMUL166-OCDS pod). The connection to the host PC is with a 25-pin male to a 25-pin female cable plugged into the LPTx Port.

The user interface software is Nohau's Seehau C/C++ debugger. Seehau provides multiple views of the code; C/C++, C/C++ mixed with assembly and pure assembly. Seehau supports symbolic debugging for Keil and Altium.

The run control provides Source Step Into and Over, Assembly Step Into and Over, and Return Step. There is a facility to preset register values and view data in multiple formats and windows.

The Data Window displays in multiple formats including: ASCII, Binary, HEX, Graphs, Gauges and custom defined formats. The Data Windows have access to different memory areas including: Code, Data, Register and Shadow RAM. The Shadow RAM will reflect changes to Data Memory while the target is running.

Figure 1 EMUL166-PC/OCDS Target Connection

System Installation

The purpose of this document is to get your emulator up and running quickly. We have outlined 3 steps to accomplish this. The first step is to install the hardware, then install the software, and then run a program**.**

Install the Hardware

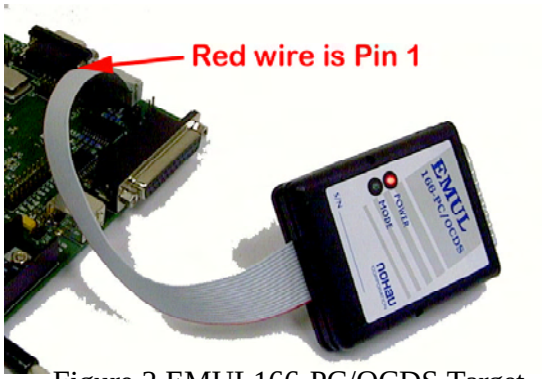


Figure 2 EMUL166-PC/OCDS Target Connection

To install the EMUL166-OCDS-XC16*x*-PC hardware plug the 16-pin Berg connector onto the OCDS connector on your target board, the red wire of the ribbon cable is pin 1 (see Figure 2).

The female end of the 25-pin to 25-pin cable plugs into the DB25 connector of the EMUL166-OCDS-PC and the male end plugs into the LPT port of your PC.

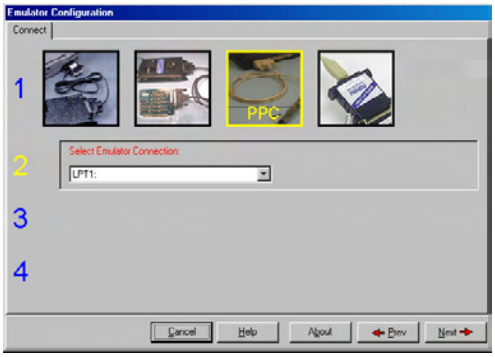


Figure 3 Connect

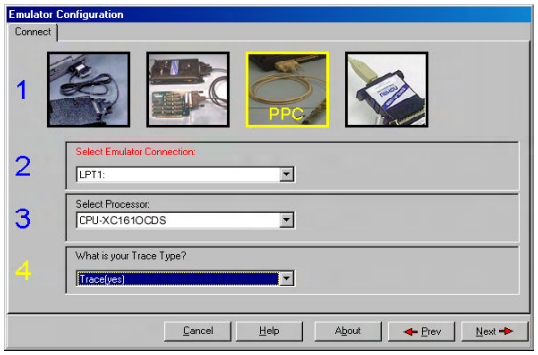


Figure 4 Connect

Install the Software

1) Place the Seehau CD into your CDROM drive. After autorun executes the Nohau Software Installer window will appear.
2) Click the button labeled "Install Seehau Interface for EMUL-16x".
3) This will start a standard Install Shield installation.
4) After the installation has completed click on Start ->Programs-> Seehau-16x and select *Configure Seehau.*
5) The Emulator Configuration Communications window appears as in Figure 3.

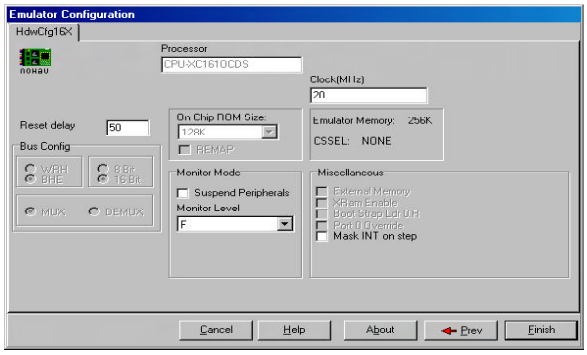


Figure 5 Emulator Configuration Screen

6) The first step is to select the interface type. For the EMUL166-XC16X-OCDS/PC emulator the PPC is selected. This is shown in Figure 4. Click the next button to proceed.
7) Select the CPU-XC16XOCDS. Click the next button to proceed.
8) Use the default setting of Trace (yes) by clicking the next button to proceed.
9) The Emulator Configuration HdwCfg16X tab will appear as in Figure 5. Proceed by clicking the Finish button.
10) The Start Emulator will appear. Click Yes to start the emulator.

Configure the Emulator

The XC161 controller will reset to different addresses depending upon the state of the EA pin at reset. If the EA pin is low (Eternal mode), than at reset the PC will contain address 000000H, which is the address of external memory. If the EA pin is high (Single Chip mode), then at reset the PC will contain the address C00000H, which is the address of the on-chip 128K FLASH.
NOTE: The current version of the emulator does support FLASH programming.

To use the current version of the emulator in Single Chip mode you can load your program into the on-chip SRAM. The ability of the Seehau debugger to preset register values makes this fairly simple to accomplish.

Register Values at Reset

The first register to preset is the PC. From the main menu bar select CONFIG->Emulator. When the dialog appears click on the MISC Setup tab. Check the Program Counter box and enter E00000, as shown in Figure 6.

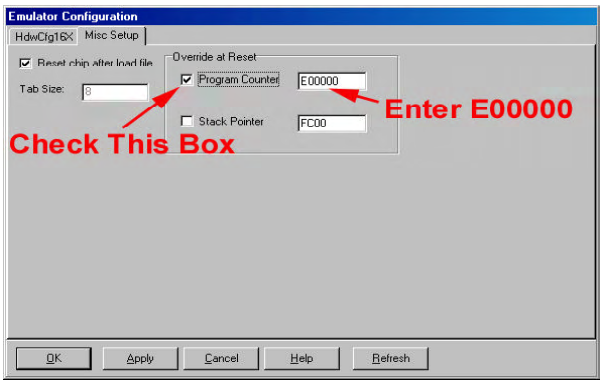


Figure 6 HdwCfg16x Tab

Configuring the Emulator For Your Target

The next register to set is the base address for the Vector Table. From the main menu bar select View->S F Regs. The window in Figure 7 appears. Click the + next to C166S-V2 CORE, after the list expands select *vecseg, this is shown highlighted in Figure 7.
Next click the right arrow under selected, then click add and select new window. A new register window will appear containing the *vecseg register. To set value at reset, right-click on *vecseg and select Change Attributes. The window in figure 8 now appears. Check the Enable Reset Value box and enter E0 for the value.

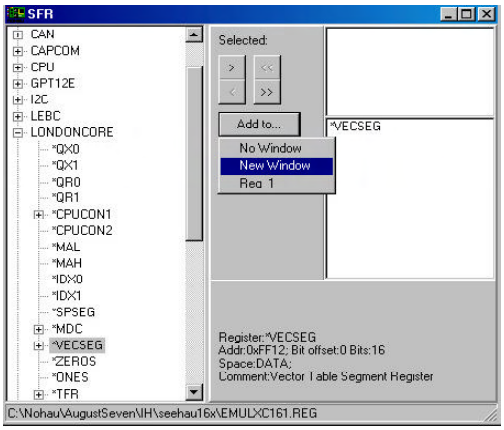


Figure 7 SFR Window

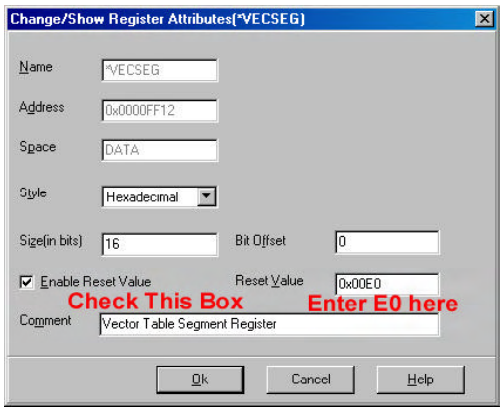


Figure 8 Change Attributes Window

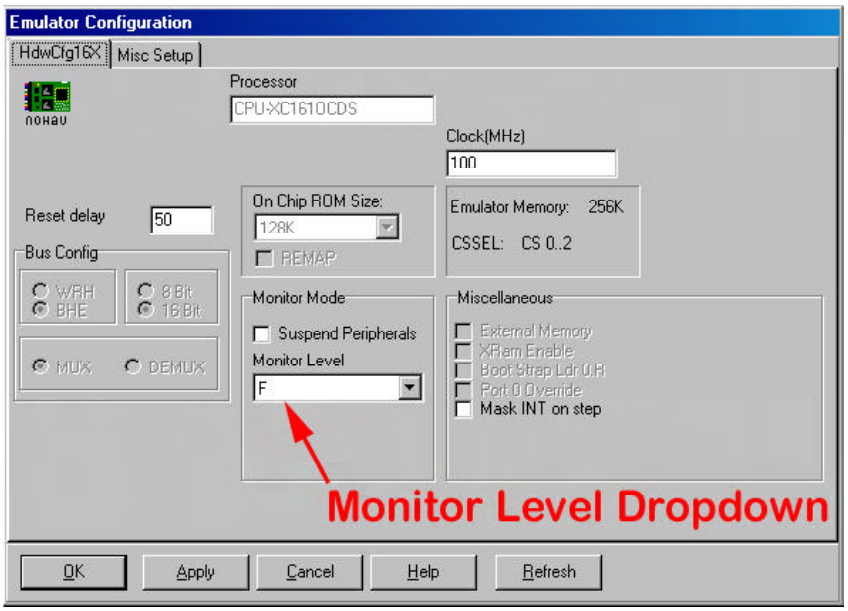
Save the Settings

In order to start the Seehau software with these settings you need to save the configuration. This is done by selecting CONFIG->Save Settings from the main menu bar. Use the *startup.bas*, which is the default startup file name.

Emulator Configuration Screen

The Emulator configuration screen is accessed from the main menu bar by selecting Config -> Emulator. The HdwCfg16X tab is used to configure the emulator hardware. The Misc Setup tab is used for reset options.

Figure 9 HdwCfg16X



The Reset Delay lengthens Reset Pulse by the time entered (in mS), the default is 50mS.

The Suspend Peripherals will stop interrupts at a break. The Interrupt enable in Debug Mode is available and set by the Level of the Monitor Interrupt in the Monitor Mode section as shown in Figure 9.

The Mask INT on step is used with Source Stepping to prevent stepping into interrupts. This is particularly useful when there is a frequently occurring interrupt in your system.

Emulator Demonstration Program

Now that we have configured the emulator we will demonstrate some of the features. We will show how to Load Code, Setup a Shadow Memory display in ASCII and use the Watch Points to break on a complex event.

Load and Run the Timer Demo Example Program

The next step would be to load a program. From the file menu select File->Load Code. When the file open dialog box appears, open the folder called XC16*x*, select the file *XC16x.abs* and click the OPEN button. After the program loads you can verify that the program has loaded by selecting File->Verify loaded code from the main menu bar. When the file dialog appears select the *XC16x.abs* file, and click open.

Next click the  Step Into button on the Seehau Toolbar. The program will run to the first line of the function main. Congratulations the emulator is installed and running!

Using Shadow Memory

The EMUL166-OCDS-XC16*x*-PC emulator provides Shadow RAM which can be used to observe changes to the memory while the program is executing. When the value changes it will be highlighted in red, there is an array, called show, in the *XC16x.abs* file that can be used to demonstrate this. In the Data Window (shown in Figure 10) replace the default 000000 with "show" in the address entry text box.

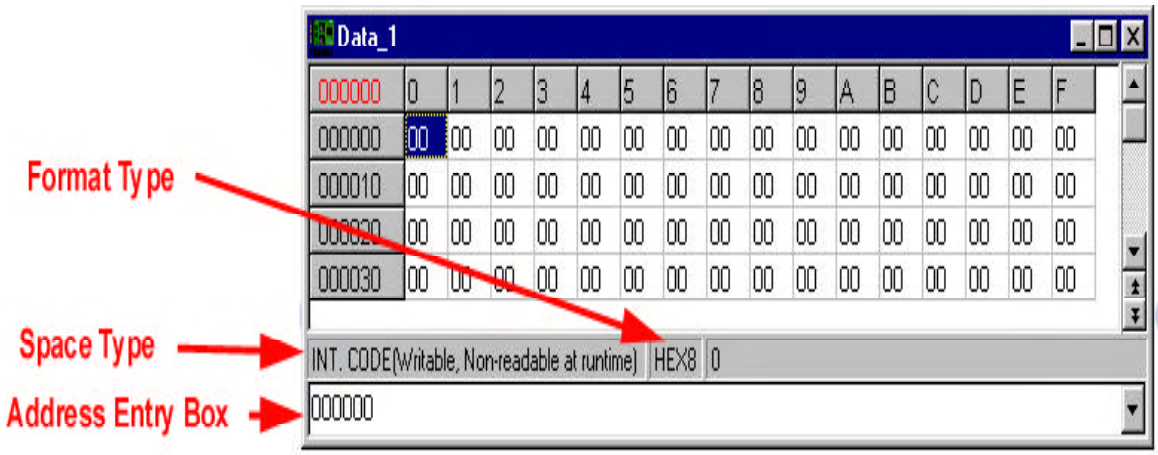


Figure 10 Data Window Shadow RAM

Next click on the Space type area of the status bar and select Shadow. Lastly click on the Format type area and select ASCII. Click the OK button. The Data Window should now have C010 as the starting address and display data in ASCII. To observe the Shadow Memory click on the 

icon on the Seehau main tool bar. The data should now highlight in red the values that are changing.

Inspect/Watch Window

The Inspect/Watch Window is used to inspect or change variables in your program. The Inspect/Watch Window can be opened by selecting New -> Inspect/Watch from the main menu bar. The window in Figure 11 appears.

In the upper text box enter the name of the variable you want to inspect. In this case type “show” and press enter. The variable show will appear with a + sign to the left of it, indicating there are multiple elements to view. When you click the + sign the individual elements will be displayed. The element that we were observing with the Shadow RAM was element 7, this is highlighted as shown in Figure 11.

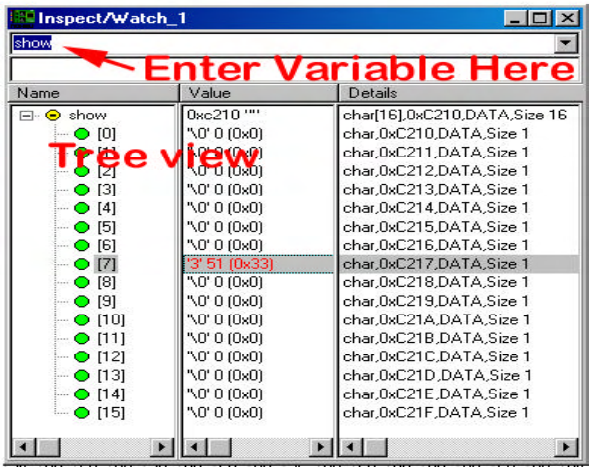


Figure 11 Inspect Window

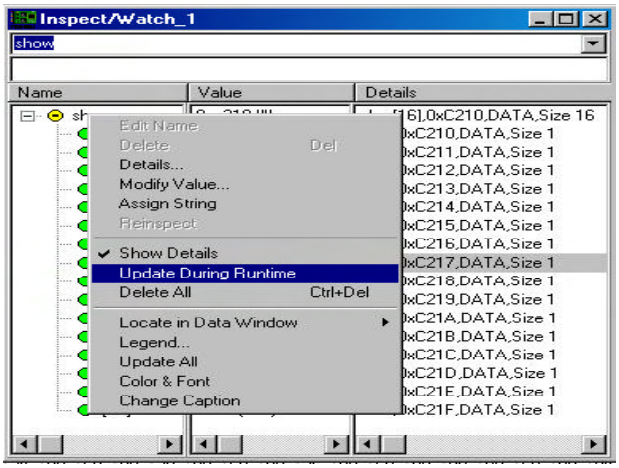


Figure 12 Update During Runtime

The Inspect Window can either show the values when the program is stopped at a Breakpoint or while the program is running, by using the Shadow RAM. To display the variable while the program is running right-click on the variable and select Update During Runtime (Figure 12).

To see show update while the program is running click on the 

icon and the values that are changing will be highlighted in red.

Watch Points

The EMUL-OCDS-XC16*x*-PC has Watch Points that can be used to break execution based on Instruction Execution, Data Accesses (Reads or Writes), Data Values and Task ID. The on-chip triggering facility can detect Instruction Pointer, Data Values, Data Reads or Data Writes inside or outside a range. The result of this can be AND'd or OR'd with a second comparator. This can be used to find a Data Value, Instruction Pointer, Read Address or TASK ID equal to a value or a value with a Mask.

Watch Point Logic

The Watch Point Logic has two modes of operation. The first mode in which the Equal comparator (DCMPx) is used for Equal comparisons is shown in Figure 1. The arrows point to the control on the Watch Point configuration tab that corresponds to that section of the block diagram.

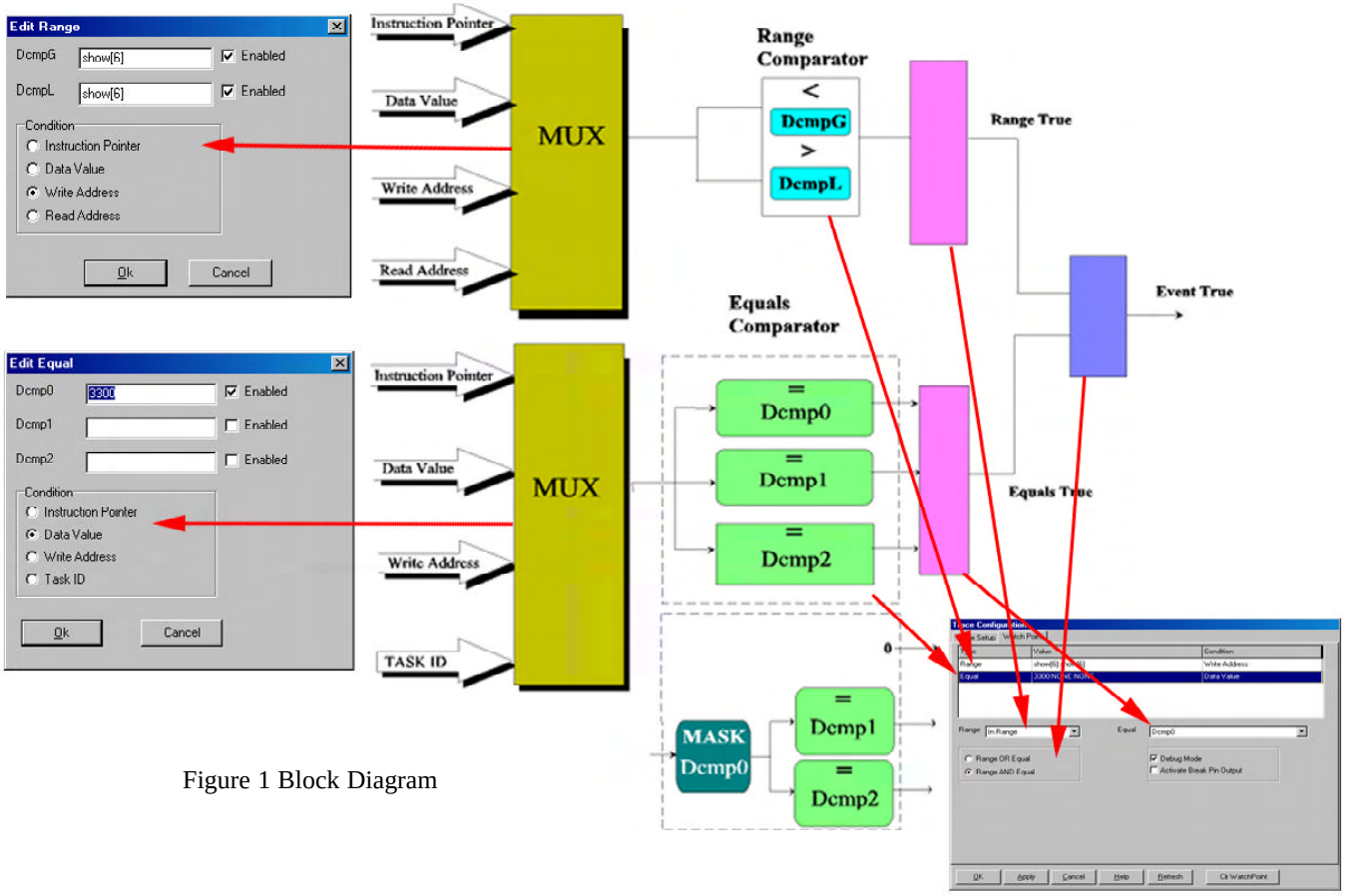


Figure 1 Block Diagram

Equal Comparisons

The Range MUX selects one of the following conditions for Range comparison: Instruction Pointer, Data Value, Read Address or Write Address. When in Range comparison is selected (see Figure 2) then the input must be greater than DcmpL and less then DcmpG. For example, if the condition is set to Instruction Pointer, DcmpG equals 0x1000, DcmpL equals 0x FF0, the comparison will be true, if the code executes at an address between 0xFF0 and 0x1000. When an out of Range comparison is selected for the same settings the code execution address has to be less than 0xFF0 and greater than 0x1000 for the comparison to be true.

To enter the values for the Range right-click on the range line as shown in Figure 2. The window in Figure 3 will appear. To select the condition click one of the condition radio buttons. Enter the values for the Range, DcmpG is the upper box and DcmpL is the lower box.

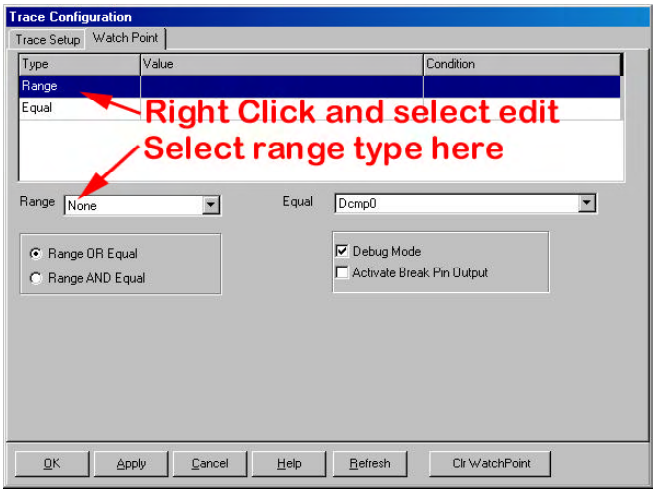


Figure 2 Range Type Selection

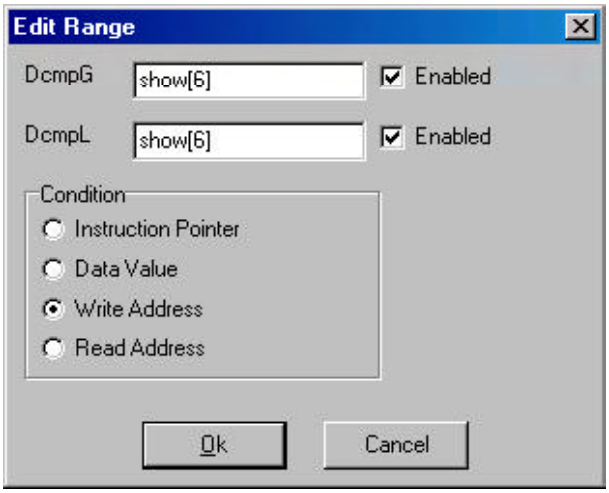


Figure 3 Edit Range Value

The Equal MUX selects one of the following conditions for the Equal comparator: Instruction Address, Data Value, Write Address or Task ID (used by some RTOS's to store the current task ID). This comparator has two modes, Equal and Mask.

When the Equal mode is used up to three values can be OR'd. The Equal mode is selected from the drop down list shown in Figure 4. Select one of the following items from the equal dropdown list: Dcmp0 for a single value, Dcmp0 OR Dcmp1 for two values or Dcmp0 OR Dcmp1 OR Dcmp2 for three values.

When the Mask mode is selected Dcmp0 is used for the Mask and Dcmp1 or Dcmp2 is used for the value. To select this mode select [Dcmp0 as Mask] Dcmp1 or [Dcmp0 as Mask] Dcmp1 OR Dcmp2. To enter the values right-click on the Equal line as shown in Figure 4.

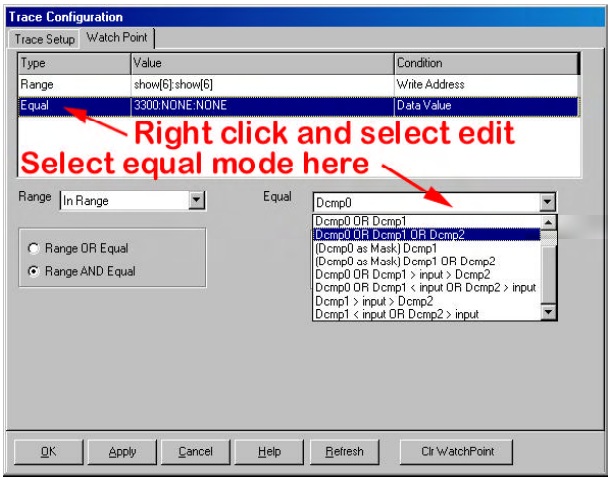


Figure 4 Equal Mode Selection

The result of the Range comparator is AND'd or OR'd with the result from the Equal comparator. When the result is true, it will cause the emulator to break if Debug Mode is selected, generate a Trigger OUT if Activate Break Output pin is selected or do both.

Range Comparisons

The second mode for the Watch Points uses the Equal comparator as a second Range comparator. The block diagram of this is shown is Figure 5.

The only difference is that the Equal comparator in this mode, is used as a second Range comparator. This section will describe the differences only.

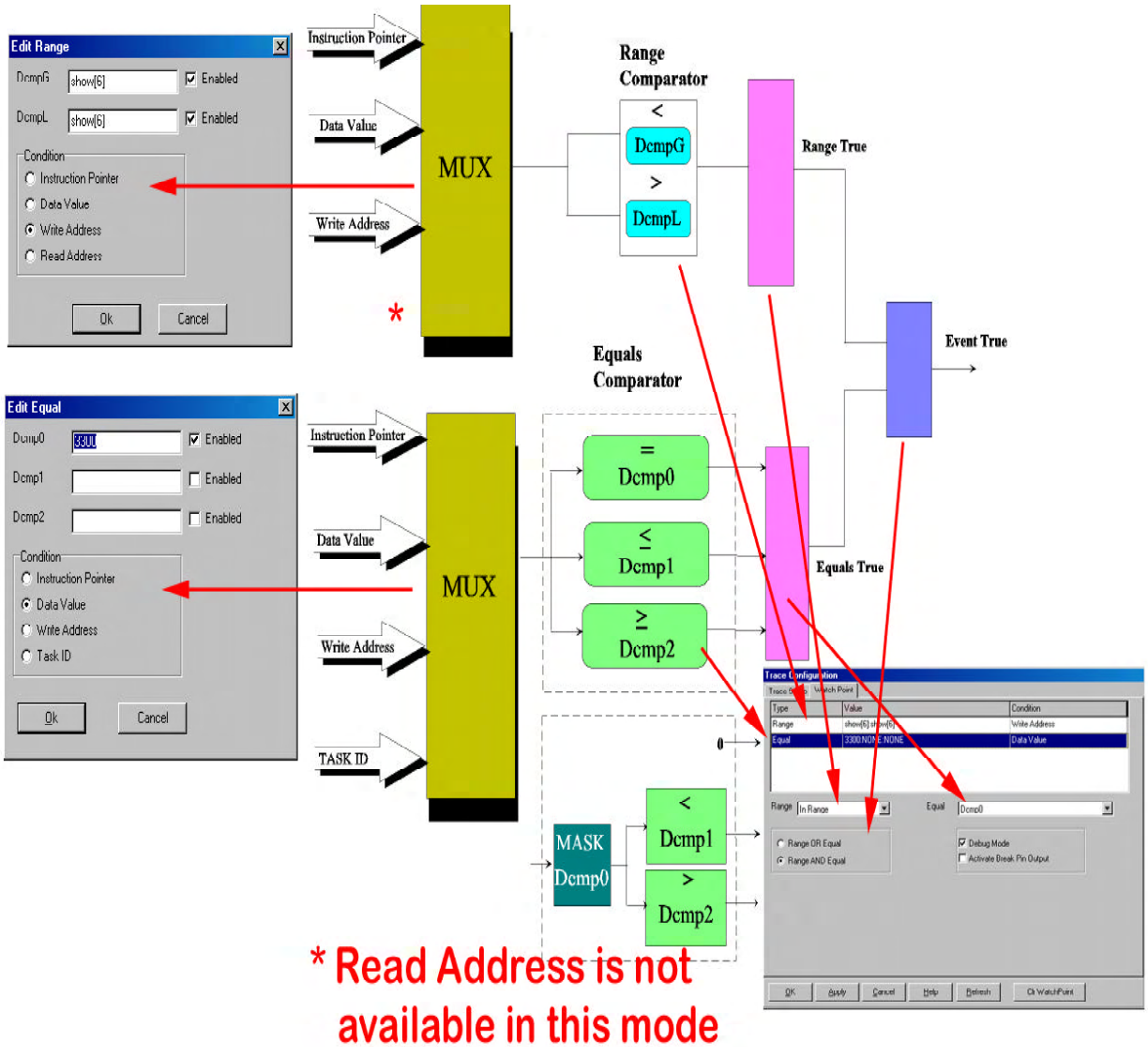


Figure 5 Block Diagram Range mode

Chapter 2 Watchpoints

The Range mode is selected from the drop down list (shown in Figure 6) by selecting one of the following: Dcmp0 OR Dcmp1 > input > Dcmp2, Dcmp0 OR Dcmp1 < input OR Dcmp2 > input, Dcmp1 > input > Dcmp2, Dcmp1 < input OR Dcmp2 > input.

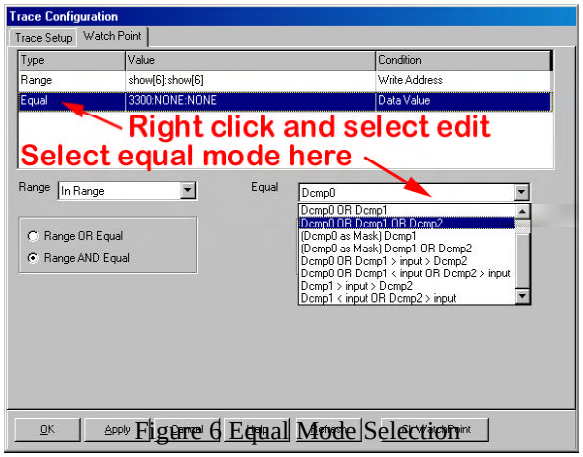


Figure 6 Equal Mode Selection

The Dcmp0 OR Dcmp1 > input > Dcmp2 is true if the value in Dcmp0 equals the input or is inside a range. If we use 0x700 for Dcmp0, 0x1000 for Dcmp1, 0xFF0 for Dcmp2 and Instruction pointer for the condition, the condition will be true if code execution is inside the range of 0xFF0 to 0x1000 or equal to 0x700.

The Dcmp1 > input < Dcmp2 is used to detect an inside range condition. The value in Dcmp2 must be less than the value in Dcmp1. If we use 0xFF0 for Dcmp2, 0x1000 for Dcmp1 and Instruction pointer for the condition, the condition will be true if code execution is inside the range of 0xFF0 to 0x1000.

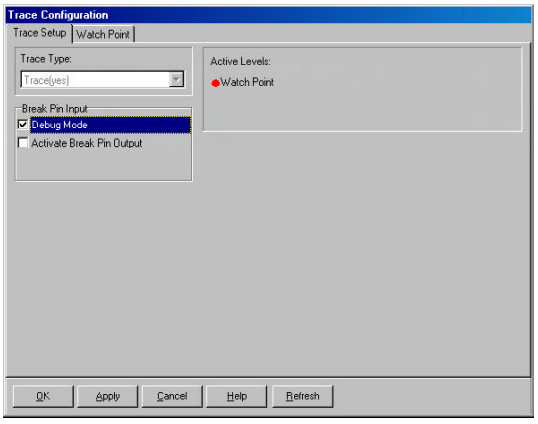
To enter the values right-click on the Equal line as shown in Figure 6. There is also a Break Input pin that can be used to break emulation from an external instrument such as a DSO.

How To Use Watch Points

In this section we will have some practical uses for the Watch Points. The examples will demonstrate how to use the Equal and Mask modes.

Using the Watch Points - Real Examples

The configuration for the Watch Point facility is found on the main menu by selecting CONFIG->Trace. The window in Figure 7 appears. This window has two tabs. The Trace Setup tab is used to configure the Trigger and Break Input. The second tab, Watch Point, is used to configure the Watch Points. One use for Watch Points would be to break when a variable equals a particular value. To demonstrate this we will use the `XC16`*x* example program. This program is a clock that updates every second. The clock's values are stored in an array called show used to demonstrate the Shadow RAM feature. The sixth and seventh elements of this array contain the seconds. Follow the configuration steps in the next section "How To Use Watch Points".



Variable Equals a Value Watch Point Example

To break when the lower digit of the second's field equals 3, begin by selecting Config -> Trace from the main menu bar. Then click the Watch Point tab, shown in Figure 8. Right-click on the Range line and select edit, the window in Figure 9 appears. In the DcmpG entry box type show[6], for the sixth element of the array. As you enter the text it will be echoed in the DcmpL entry box, since this expects a range. Click the radio button next Write Address. Click the OK button.

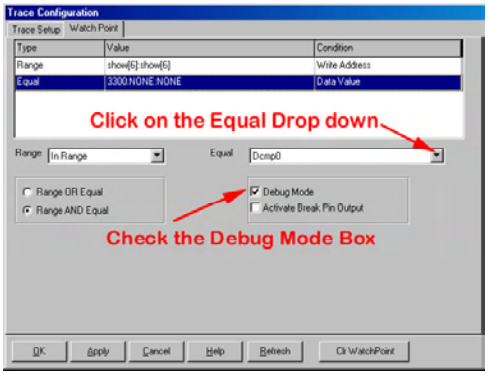


Figure 8 Watch Point Configuration

Right-click on the Equals line and select edit. In the Dcmp0 entry box type 0x3300. Click the radio button next to Data Value. Click the OK button (see Figure 9).

On the Watch Point tab click on the equal drop down and select Dcmp0. Check the Debug Mode box and click the OK button.

On the Seehau Toolbar click the 

icon. The program should stop when show[7] equals 3. To verify this look in the data location C017, it should contain 3.

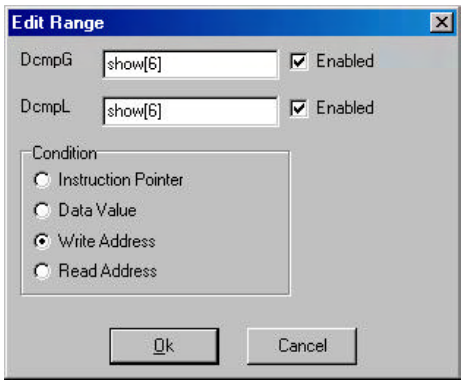


Figure 9 Edit Range Value

Variables Equal a Value Range Watch Point Example

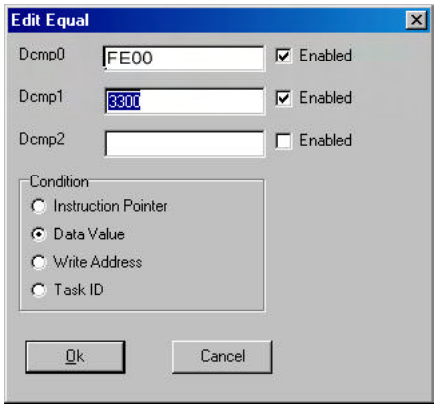


Figure 10 Edit Equal

To Break when seconds are odd (for example 31, 33) we will use the Mask in Equal comparator to detect bit 0 equal to 1. In the Watch Point window, click on the Equal dropdown arrow, select (use Dcmp0 as Mask) OR Dcmp1 (see Figure 8).
Right-click on the Equal line and select Edit. In the Edit Equal window (see Figure10), enter FE00 in Dcmp0 and 0x3300in Dcmp1. Click the OK button.

On the Seehau Toolbar click the 

icon. The program should stop when show[7] is odd. To verify this address C017 in the Data window should have a value of 3.