# SYSC-2003 Introduction to Real-Time Systems

# Lab Tutorial: Using NoICE Debugger

NoICE is a remote debugger that can be used with our Axiom Boards. When you use NoICE, in truth you are running two programs. One resides on the host PC and the other on the target board. The program that runs on the target board waits for instructions from the PC program. The PC program will send the target a user program to load into memory, and will allow the user to insert breakpoints into the program to aid in debugging.

The NoICE target program acts as a layer between the HC12 and the user program. It fetches instructions into its own memory space and then allows the HC12 to execute them there. This explains how NoICE can insert a breakpoint into a program without shifting line addresses and user memory space around. This type of program is usually called a monitor.

## Getting NoICE

NoICE is installed in the undergraduate labs in Rooms MC4233, and possibly other rooms, check the course web site for details. Although a demo version is available, NoICE is not useful without an actual target board.

## Using NoICE

There are two main ways to use NoICE: S19 files, and DBG files. The S19 files provide only assembly level debugging, while the DBG files produced by ICC12 can be used to debug at the C code level, which is very desirable. MiniIDE and eGNU will only produce S19 files. ICC12 will produce both types of file.

### Debugging with S19 Files

First, the board must always be turned on before NoICE is loaded. After NoICE is loaded, you should see a memory disassembly starting at 0x0000. If you don't, close NoICE, reset your board, then open NoICE again. Next, select File->Load and then find your S19 file. Once it has downloaded, you will have to set the program counter to the start of your program by clicking on PC and typing in that address. If you are unsure of the address, check the listing file. You should note that most numbers are given in hexadecimal in NoICE, and all hexadecimal numbers are prefixed with 0x not $. After setting the program counter, select View->Source at PC to make the disassembly start at the start of your program.

As an aside, it is interesting to see what instructions NoICE disassembles because they may not exactly match what you have written in your source file.

You may want to load your listing file into the bottom pane under the view tab. You can do this by selecting File->View (shortcut key is F4). Browse through your listing file to find line addresses of lines you wish to breakpoint. Right-click on a line in the disassembly and select Set Breakpoint on Line. Execute your program by clicking on the Go button. When your program hits a breakpoint, control will be returned to the disassembly window, NoICE then allows you to step through your program using the buttons in the top row.

The view tab will display any text file. The watch tab allows you to add watches on specific memory locations. Whenever your program hits a breakpoint, you can inspect the value of your watches by looking here. You can also view sections of memory by using the memory dump function under the memory tab in the bottom pane, just enter a start address into the box and click read; new or changed values will appear in red. The output tab functions as a terminal emulator and allows you to see the output of any serial I/O your program may be sending. You may also to type directly into it, which will send any typed characters to your board.

This leads into a potential programming problem. NoICE uses the SCI0 channel to communicate with the target board. If a user's program also uses this channel, for whatever purpose, the two programs may interfere with each other. This will usually occur whenever NoICE tries to reassert control over the board. When NoICE does this, it may change registers and variables that are part of, or dependent on the SCI0 channel. Users debugging must be very careful to make sure the data they are looking at is their program's data, and not data from NoICE. Another aspect of this problem is that if the user's program is buggy, or if it interferes with something that NoICE needs to operate, NoICE may lose control and breakpoints will be missed. This may happen when programming SCI0 interrupts.

**Debugging with DBG Files (Source Level Debugging)**

ICC12 allows much easier debugging by producing DBG files for use with NoICE. These files contain the program's source code within them along with line numbers and variable names and locations. This allows NoICE to debug at the source-level.

To begin, you must set the options in ICC to generate NOT S19 files, but DBG files.
In ICC, under the project, select "Options"
Under Options, select the "Compiler" tab
For Output Format, select "S19 with ASM/Source Level Debugging"

Next, when loading the file into NOICE, you must be careful to load the DBG file, not the S19 file. In the Load's dialog box, change "Files of type" from S19 files to "Imagecraft DBG files".

As before, after loading the DBG file, the PC counter is automatically set. Next, the user should select View->Select Source File… (the Q button). A dialog will pop up asking which file you want to debug. Select the DBG file with the main() function within. You will now see the line numbers for every C statement.

Additionally, if you now select View->Mixed source/disassembly, NoICE will show both the C and the corresponding assembly instructions that comprise each C statement.

Right clicking on a variable will allow you to set a watch on that variable. Adding a watch is made easier because in the Add Watch dialog, you can now click on the button beside the Address Expression line and select from a list of symbols generated from the DBG file. As you use the program, you will see this List Symbols button in various places to assist your debugging. You can also now add watches simply by clicking the List Symbols button just below the menu bar on the far right, and then double click on any symbol to add a watch.

The different step functions have more meaning when using source-level debugging.
Step over means to execute the program until immediately after the function the PC is pointing at has returned.
Step into delves into the function the PC is currently pointing at.
Step PC means to execute the next assembly instruction and then stop. This will activate mixed source/disassembly mode.

NoICE has many more features, but these are the basics. If you need help, try using the online help feature in NoICE.