EE 354 **Fall, 2013**
Simulation Notes

1. Create a new project in µVision4 using AT89C51CC03 processor.
2. Enter the c-code that you want to simulate. Your c-code should use the A/D converter. Here is some sample code which outputs the A/D converter to port 2.

```
//AtoDTest.c
#include <at89c51cc03.h>
//Takes the input from the A/D converter channel 0 and
// sends it to P2 and P3.  Assumes P2 has D/A triggered
// by bit P4.0.  P2 holds 8 MSBs.
void main (void)
   {unsigned char tmp;
    ADCF  = 0x01;         // P1.0 = ADC[0]
    ADCON = 0x20;         // Enable ADC Function
    ADCLK = 0x00;         // Prescaler to 0
    EA = 0;               //Turn off interrupts
    while(1)              // Loop Forever
     {ADCON &= 0xF8;      // Reset ADC Channel Select
      ADCON |= 0x00;      // Select ADC = Ch0
      ADCON |= 0x20;      // Use Standard mode
      ADCON |= 0x08;      // Start ADC Convert
      tmp = (ADCON & 0x10);    // Get done bit
      while(tmp != 0x10)       // Loop until complete
        tmp = (ADCON & 0x10);
      P2 = ADDH;               // Send 8 MSB to P2
      P3 = ADDL;
      P4_0 = 0;                // Low going pulse to D to A
      P4_0 = 1;                //  write line
      ADCON &= 0xEF;           // Clear ADEOC = 0
     }
   }
```

3. To simulate a sine wave on the A to D input

File → New

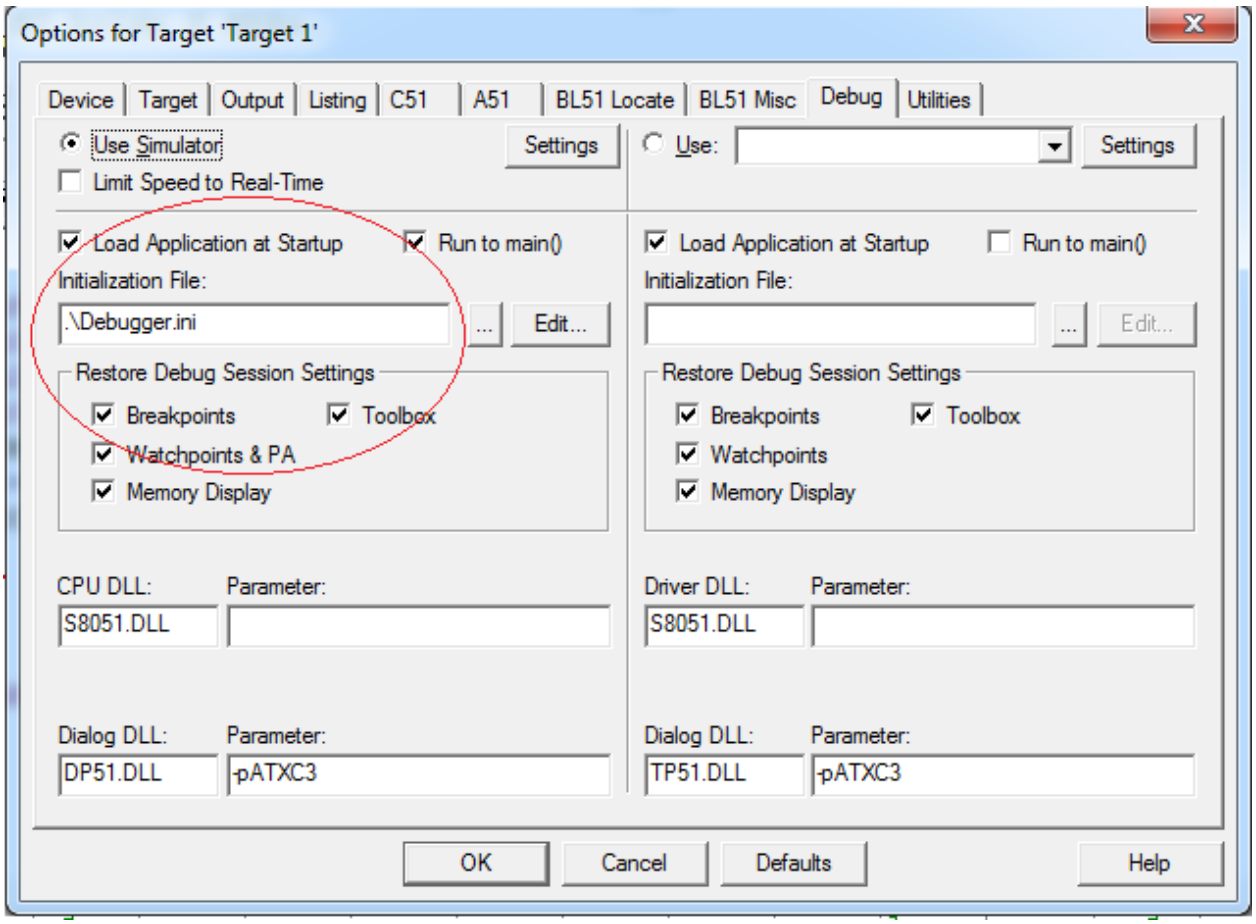
Enter the following simulation code

```
//
// Generate Sine Wave Signal on AD Channel 0
//
signal void ADC(void)
 {float amplitude;    // peak-to-peak voltage
  float frequency;    // output frequency in Hz
  float offset;       // voltage offset
  float duration;     // duration in Seconds
  float val;
  long  i, end;
  amplitude     = 2.3;
  offset    = 1.0;
  frequency = 100;  //Change this to whatever is needed
  duration  = 0.1;

  printf ("Sine Wave Signal on AD Channel 0.\n");

  end = (duration * 100000);
  for (i = 0 ; ; i++)
   {// Runs signal continuously
    val = (float)i*(0.00001);
    AIN0 = __sin(2*3.14159*frequency*val) + offset;
    swatch (0.00001);                // in 10 uSec increments
  }
}
```

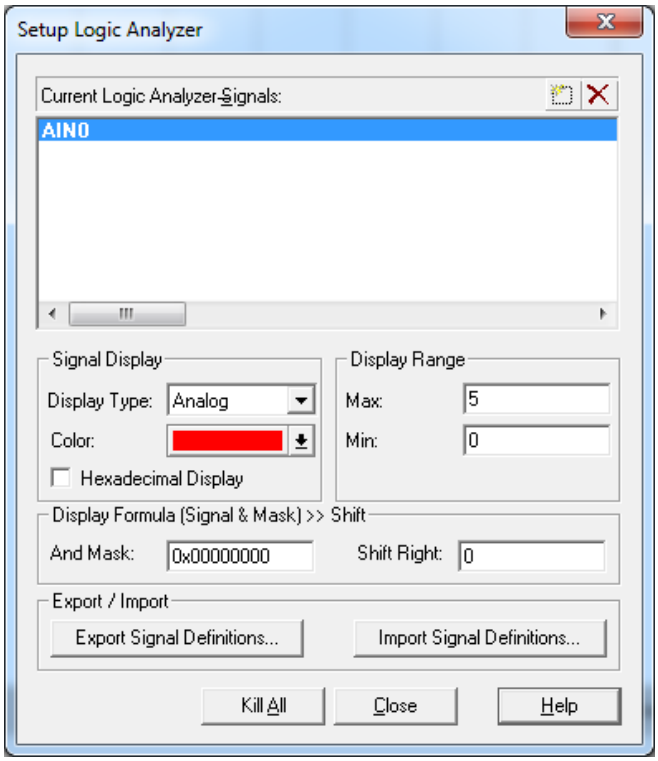
File → SaveAs Save this file as Debugger.ini in the project file.

4. Click on Project → Options for Target1 → Debugger tab

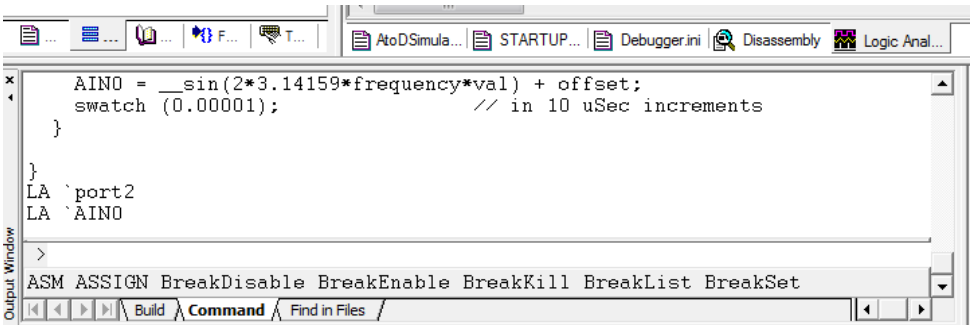
5. For the initialization entry click on the button with "…" on it and locate the Debugger.ini file you just saved.



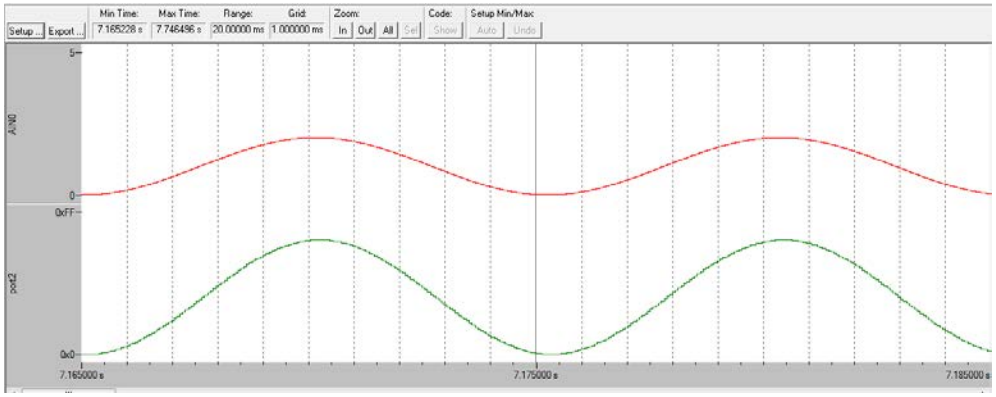
6. Under the options dialog you must also specify that you want a hex file produced if you are going to implement this program on your board.
7. Click OK and build project
8. *After* successful build run debugger Debug → Start/Stop Debug Session
9. In Debugger: View → analysis window → Logic analyzer Window
10. In Logic analyzer click on SetUp. Push Insert key and enter AIN0 as the signal name. The signal range should default to 0 to 5 volts. Your setup window should look like that shown below. You may also want to enter other signals. For example, if you want to see Port 2, push the Insert Key and enter port2.
11. Close the setup window.
12. At the bottom of the screen there should be a Output window. If not use the View menu to bring it up. The output window will look like that shown below. Type `ADC()` on the command line as shown and push return.
13. Run the simulation for a few seconds and then stop it. Look at your output in the logic analyzer simulation window.



Setup Window for Logic Analyzer



Output window for simulation. enter ADC() on the command line and push Enter



Logic simulator output window. ADC on top, port2 on bottom.