Lab2

In this lab, you will access some of the general purpose I/O ports using some of the given examples provided by TI. You will also begin using the Bitscope oscilloscope.

1) Do not connect up the DSP board to the computer at this time.
2) When you receive the Bitscope oscilloscope from the instructor, insert the CD into the CD/ROM drive and perform the setup. Don’t hook the hardware to the computer through the USB port at this time. The setup is reasonably straightforward, and a ‘BitScope DSO’ icon should show up on your desktop. Open the Bitscope program, and you should see a screen like the following:

3) Click on the ‘SETUP’ button on the top right of the screen. You may see something like this on the screen:
3) Once the Bitscope software is installed and the setup of the USB is completed, restart your computer. The Bitscope should be ready for use. Details on the software can be found at: [http://www.bitscope.com/software/dso/guide/1.3/](http://www.bitscope.com/software/dso/guide/1.3/).

4) Your CD that was used to install the CCS software also contains a number of examples for use. Insert the CD into the CD-ROM drive. Click on ‘BROWSE CONTENTS’. Go into folder ‘drive’\Spectrum Digital\F2833x_\HeaderFiles_FromTI\ sprc530, where ‘drive’ is the drive for the CD-ROM. Follow the setup instructions in file ‘setup_DSP2833x_v102’. After the folder ‘tides’ is installed on your computer, you may wish to create a duplicate folder in your ‘My Projects’ folder for actual use, because you may want to modify the content of some of the examples provided.

5) With CCS running, open up the project in the ‘tides’ folder (the instructor is not sure why all the folders are necessary):

   \tides\c28\DSP2833x\v102\DSP2833x_examples\gpio_toggle\Example_2833xGpioToggle.pjt

6) Ensure that the DSP board is powered up before connecting to the computer using the USB cable.

7) Connect to the DSP board using Debug/Connect. Then build and load the project onto the DSP board. Run the program, and using an oscilloscope probe, monitor some of the
GPIO pins (for example, put your ground at P7-20, and your probe at P8-9). Refer to the eZdsp board layout in the technical reference guide to help you.

8) Halt the program, and increase the variable ‘i’ in the code by a factor of 10. Then go through the steps of compiling, building, loading, and running the program again. Can you see the effect of increasing ‘i’ by a factor of 10?

9) Halt the program. Now we’re going to look at two of the variables in ‘real time’. First, go into your ‘Example_2833xGpioToggle.c’ program (within the CCS Source Code Window) and place variables ‘GpioDataRegs.GPADAT.all’ and ‘GpioDataRegs.GPBDAT.all’ within the Watch Window. Ensure they are shown as hex numbers by highlighting the row of the variable in question and clicking on the word ‘Radix’ at the top of the last column. Under Debug, select ‘Real-time Mode’. You may get prompted with a question. Click on ‘Yes’. Then place your cursor anywhere within the Watch Window and right mouse click. Select the option ‘Continuous Refresh’.

10) Now, when you run the program, you should see the two variables update continuously. Near the top of the code, EXAMPLE1, EXAMPLE2 and EXAMPLE3 are defined. You will likely be running EXAMPLE1, because it is set to 1. Rerun the code with the other two examples. Do you get the same answer for the two variables placed in the Watch Window? Is this what you would expect?

11) Go farther down in the code and look at function Gpio_example1. Note what your two variables are being set to display. Is this what you are seeing in your Watch Window? If not, then there must be something wrong with the program provided by TI.

*Your task is to fix the code to make the program output the proper data to the register(s) (and hence to the GPIO ports in question).*

Refer to the Data Manual to help you (the instructor is being vague on purpose here to force you to look at the manual). This will be a good opportunity for you to learn how the code has been developed. Try single-stepping through the program, and/or set up breakpoints in the code to help you (if you right click on a line of code, you can toggle a hardware or software breakpoint).

12) If you complete this task and have time left over, go ahead and run the example ‘Example_2833xEPwmDeadBand’ in the ‘epwm_deadband’ folder. Use the GPIO pin locations in step 7 above to see the output (and know why that is one of the pins that will provide you with your PWM output).

13) You have now used some of the programs provided by TI to access the GPIO pins of the eZdsp board, and you have investigated the construction of one program to understand how the code has been developed based on the hardware construction of the DSP board. This ends Lab 2. Be sure the DSP board is disconnected from the computer prior to removing power.