CS/ECE 5780/6780 Lab #1
Spring 2010
Davis

LAB #1: Getting started with the CSM12C32 Module and PBMCUSLK Project Board

Lab write-up is due to your TA at the beginning of your next scheduled lab. Don't put this off to the last minute! In future labs, there will also be pre-lab work to complete before the star t of the next lab. **NO LATE LAB REPORTS WILL BE ACCEPTED**.

1 Objectives

- Test your kit to make sure it's working – note you'll perform this same test at the end of the semester prior to turning your kit in.
- Gain familiarity with lab tools and development board prototyping.
- Learn procedures for assembling and downloading programs to the CSM12C32 module using the CodeWarrior IDE.
- Learn how to interface with the CSM12C32 module standalone and with the PBMCUSLK project board.
- Write a simple C program to demostrate base proficiency.

2 Tasks

1. Download sample C-code to the CSM12C32 module using the project board.
2. Create and download a 4-bit counter to the CSM12C32.
3. Download C-code to the CSM12C32 in the stand-alone configuration.
4. Write, compile, and download your original code.

3 Microcontroller Kit

Each partnership will check out one microcontroller kit which includes:

- PBMCUSLK project board
- CSM12C32 module
- 9 VDC universal AC/DC adapter
- USB cable
- RS-232 serial cable
- Jumper wires kit

- MCU project development board CD
- 68HC12 development board CD
- Technical training 8-bit, 16-bit, and 32-bit processors CD
- CD containing the CodeWarrior IDE tools (claimed but not verified yet)
- Various documentation

Both parties must be present to checkout the kit and sign the loan agreement. The entire contents of the kit must be CLEAN, COMPLETE, AND IN WORKING CONDITION before the last day of class. You are responsible for up to $250 for loss or damage.

4 Other Useful Information

The following is available from the course website:

- Lab1 example code (Lab1Example.c)
- Serial monitor code (hcs12serialmon2r1.zip)
- Reference materials for the MC9S12C microcontroller.
- Reference materials for the CSM12C32 module.
- Reference materials for the MCU project board.
- Reference materials for CodeWarrior.

If you would like to work at home, you can download the “CodeWarrior Development Studio for HCS12X Microcontrollers Special Edition” from the Freescale web site. Note that this free version is limited to 32K, but it will be sufficient for this course. CodeWarrior is already installed on the lab’s computers, and a CD containing the software is in your lab kit.

5 Saving Your Work

Students should also bring a USB flash drive for saving work completed during the lab. Internet is available on lab machines for e-mail. **IMPORTANT: NO WORK SHOULD BE LEFT ON THE LAB MACHINES.** So make sure all your files are copied onto your thumb drive (or equivalent) and that your files are deleted from the lab machines before you leave. If you want to use your own PC then that’s your choice. Bottom line is that you don’t want somebody else using your code since given 2 copies of the same code, it’s hard to determine who copied. Plus it is in everybody’s best interest to do the work and learn how this stuff really works.

6 Procedures

Part 1: Download code and check that your lab kit works:

1. Create a new CodeWarrior project and replace the contents of main.c with the contents of Lab1Example.c (see tutorial at the end of this handout on how to create a project in CodeWarrior). The main.c file can be found under Sources.
2. Review the code and make sure you understand each of the instructions. Refer to the Motorola Data Sheet document for more detailed information.
3. Mount your CSM12C32 module to the J5 MCU Port on the Project Board, making sure that pin1 on the module matches up with pin1 on J5.
4. Connect your Project Board to your computer using the USB cable provided.
5. Make sure USER jumpers on your module have not been removed.
6. Compile and download your newly compiled program (see tutorial a the end of this handout on how to download over USB).
7. In the debugger, press the green run button, or click Run → Start.
8. Press SW1 and SW2 (on the module, not the dip switches on the Project Board) and note how LED1 and LED2 of the module board turn on and off. Also note how the LEDS (1-4) on the Project Board light up as well.
9. (Now press buttons PB1 and PB2 on the Project Board and note how they perform the same action as SW2 and SW1 on the module. The code in Lab1Example.c ties together SW1 and PB2, and SW2 and PB1, and electronically enables the four LEDs. Refer to pages 20 and 21 of the MCU Project Board Student Learning Kit User Guide document.
10. Check-off the working code with your TA.

Part 2: Construct, download, and run a simple code assignment: 4-bit Gray-Code counter:.

1. Create a lab directory on your lab PC. Use your own name for this directory. To minimize the risk of someone stealing your code, be sure to clean this director y of all your personal lab assignments and code at the end of the session. Remember to save your work on your own USB drive.
2. Modify the Lab1Example.c code to create a 4-bit Gray-Code counter that increments when SW1, SW2, PB1 or PB2 is pressed. For this program, the module will be plugged into the Project Board and use the available four LEDs. See Gray-Code tutorial at the end of this handout.
3. Remember to review MCU Project Board Student Learning Kit User Guide (pg. 20-21) and Application Module Student Learning Kit Users Guide (pg. 11-12) for more detailed information about the interconnectivity of the module and Project Board.
4. Check-off the working counter with your TA.

Download using serial monitor:

1. Download over USB the serial monitor to your module (see the tutorial at the end of this handout).
2. Disconnect the CSM12C32 module from the Project Board.
3. Plug the barrel connector, from the power supply, into the module.
4. Plug the serial cable from the computer into your module.
5. Create a new CodeWarrior project and replace the contents of main.c with the contents of Lab1Example.c.
6. Compile and download your newly compiled program (see tutorial at the end of this handout on how to download a program using the Serial Monitor).
7. If the program is not running, make sure to press the green run button, or click Run→Start.
8. Press SW1 and SW2 and make sure that the LEDS of the module turn on and off.
9. Check-off the working module with your TA.

7 Writeup

Due at the beginning of your next lab section (Feb. 3 or Feb. 5)

1. Printouts of your (well commented!) counter program.
2. Describe any problems you encountered.

8 Tutorials

A. Interfacing using the PB MCU Student Learning Kit (PBMCUSLK)

Creating a C-code project in CodeWarrior:

1. Start up the CodeWarrior IDE on one of the lab machines by finding the shortcut under Start → Programs → Freescale Codewarrior→ CW for HC12 v 4.5.
2. A new window will pop up, choose the HC(S)12 New Project Window option. Give the project a name and click OK. (If this window does not popup automatically, you can open it with File →New. . .
3. On page 2, select the model number of your module, MC9S12C32.

4. On the next page, select the C language and make sure the other options are deselected.
5. Click no on page 4 and 5.
6. Choose ANSI startup code on page 6.
7. On page 7 and 8, choose None and Small, respectively.
8. On page 9, select Full Chip Simulation, P&E Multilink/Cyclone Pro, and HSC12 Serial Monitor. These options allow you to create different builds of your program to be used for different interfacing options.
9. Finally, click finish and your project will be created.

Downloading over USB:

1. In your project window, make sure that P&E Multilink CyclonePro is selected. This option tells the compiler to build the program to debug on the Project Board.
2. To compile, click Project→Make, or press F7. Make sure you have no compiler errors.
3. To download, make sure the CSM12C32 module is plugged into the Project Board, and is receiving power from the USB port as indicated by the appropriate LEDs.
4. Click Project→Debug (F5)
5. A new program will open called True-Time Simulator & Real-time debugger. It will immediately try to download your compiled program to the board. If it warns you about erasing the flash, click OK.
6. After downloading, the program will display a number of windows showing the current status of your module. To run your program, simply click the green arrow, or click Run→Start. If you wish to step through the program, click the Step Over button.

Downloading the serial monitor to your module:

1. Visit the class website and download the serial monitor zip file (hcs12serialmon2r1.zip).
2. Extract the file on the hard drive and open the .mcp CodeWarrior project file.
3. Make sure the module is connected to the Project Board, and that the Project Board is plugged into the USB port.
4. Build the project by clicking Project→Make (F7).
5. Download the project over to your module by clicking Project→Debug (F5).
6. After it has downloaded over the program, you can disconnect the USB cable and detach the module. It is now ready as a stand-alone module using the serial port.

Downloading your program using the Serial Monitor :

1. Open your CodeWarrior project that you made before.
2. In the project window, select HCS12 Serial Monitor. This option tells the compiler to build the program to debug using the serial monitor.
3. To compile, click Project→Make (F7). Make sure you have no compiler errors.
4. To download, make sure the CSM12C32 module is plugged into the power adapter and that it is connected to the PC using the serial cable.
5. Click Project→Debug (F5)
6. A new program will open called True-Time Simulator & Real-time debugger. It will immediately - try to download your compiled program to the board. It may come up with a window called Monitor Setup. If it does, select the correct Com port that you wish to communicate over. Also, if it warns you about erasing the flash, click OK.
7. After downloading, the program will display a number of windows showing the current status of your module. To run your program, simply click the green arrow, or click Run→Start. If you wish to step through the program, click the Step Over button.
8. Note if you are debugging using the serial monitor, you can step through your code, but once you click Run, you cannot stop the program. You must reset it to regain control.
9. IMPORTANT: After you've downloaded your program onto your module, your module will continue to run your program each time it is reset. This will prevent you from downloading a new program onto it. To have your module boot back into the serial monitor, press and hold SW1 while pressing reset. If that fails, try pressing SW2 while pressing reset. Use this method if you ever receive a connection lost error while using CodeWarrior.

B. Gray Codes

You are all familiar with binary numbers in a normal encoding of integers:
00 = 0
01 = 1
10 = 2
11 = 3
etc.

A Gray code is a different encoding that has an important property that only one bit of the binary encoding changes between successive integer values. This property is also called Hamming distance = 1. This is an important property in many areas such as logic minimization, error correcting codes, etc. A gray code is also a reflected code – each reflection doubles the number of integer values

that are encoded.

For 2 integers (1 bit encoding):

Binary = integer

0 = 0
1 = 1

For 4 integers (2 bit encoding) – reflect the - bit encoding – e.g. flip the 1-bit encoding and append it to the previous 1-bit version and add a leading 0 to the first half and a 1 to the second half.

00 = 0
01 = 1
11 = 2
10 = 3

For 8 integers (3 bit encoding) – repeat the process again

000 = 0
001 = 1
011 = 2
010 = 3
110 = 4
111 = 5
101 = 6
100 = 7

Repeat again to get the 4-bit Gray-Code counter for your assignment.

0000 = 0	1100 = 8
0001 = 1	1101 = 9
0011 = 2	1111 = 10
0010 = 3	1110 = 11
0110 = 4	1010 = 12
0111 = 5	1011 = 13
0101 = 6	1001 = 14
0100 = 7	1000 = 15

Note that the next number after 15 is 0 and there is still only a single bit change in the Gray encoding. There are lots of ways to write this program: some are better than others. In this lab we just care that it functions correctly.