UNIVERSITY OF CALIFORNIA, SANTA CRUZ
BOARD OF STUDIES IN COMPUTER ENGINEERING

CMPE 118/L: INTRODUCTION TO MECHATRONICS

LABORATORY ASSIGNMENT NUMBER 0 FOR CMPE 118

Due by 6:00pm on Wednesday, January 13, 2010
Pre-Lab due by 6pm on Thursday, January 7, 2010

Purpose: This lab is intended to acquaint you with 'C' programming for embedded systems and soldering.

- Learning the ropes for the CodeWarrior programming environment
- How to write, debug and run a simple C program on an embedded micro-computer
- How to get simple character input from the user in C
- How to order samples from a semiconductor company
- How to solder simple components to printed circuit boards
- Start your lab notebook

Minimum Parts Required: A 'CockRoach 3000', a blank PCB with associated components, and a bound lab notebook.

References: Course Notes: The Events and Services Framework. State Machines

- http://www.instructables.com/id/How-to-Solder-Videos%3a-Why-is-soldering-difficult-s/
- http://blog.makezine.com/archive/2007/01/soldering_tutor_1.html
- http://www.elexp.com/t_solder.htm

Pre-Lab:

Complete the following exercises AFTER you have read through the lab assignment and BEFORE coming into the CMPE 118 Lab to complete the lab. See the course forum for more information on the lab notebook requirements.

☐ 0.1) Brainstorm with your design team to identify the essential 'Events' necessary for your desired behavior, determine how you will test for them in software and how you want to respond. Your Pre-Lab write-up should explain the results of your brainstorming session in the form of a state diagram and pseudo-code for the event checking routines.

For the purposes of the Pre-Lab, your “design team” should consist of you and 1 other person.

0.2) Familiarize yourself with soldering by reviewing the online references above.

In the Pre-Lab: Include your concepts from part 0.1

Part 1 – Exploring the MC9S12C32 and the CW12 Compiler

Reading: The Freescale CodeWarrior Quickstart manual (located in the Freescale folder on the Start Menu)

Assignment: Complete the Following Exercises:

☐ 1.1) Start the CodeWarrior (CW12) IDE (Integrated Development Environment) by clicking on the Windows ‘Start Menu’, selecting the ‘Programs’→‘Freescale CodeWarrior’→‘ CodeWarrior Development Studio for HCS12(X) V4.7’ option, then selecting the ‘CodeWarrior IDE’ shortcut. CW12 is a program that has provisions for creating and editing programs (*.C files), creating and editing project files (*.mcp files), compiling programs, and downloading finished programs to targets (in our case, the “target” is the MC9S12C32 board).

☐ 1.2) First, create a temporary directory to act as your “Home” directory from the root directory (for example C:\JON) Using the Windows Explorer, copy the provided file “C:\CMPE118\SRC\SESTEST.C” into your home directory. Open this file with CW12. You may wish to edit and save changes to this file, so you might want to rename it at this point, also saving this new version into your home directory.

☐ 1.3) Next, you will want to compile and test your program. In order to compile the source file shown in the editor window, you will first need to create a Project file for this program. From the menu bar, select "File → New". Give your project file a name and use your home directory for the location. Pick the "HC(S)12 New Project Wizard" and select "OK". The next screen asks you to pick your micro so select "MC9S12C32". The next screen asks you to choose your languages; we will be using "C". Choose "No" for using the "Processor Expert", "OSEKturbo", "PC-lint", and "Floating Point Format". Choose "Small" for the memory model. In the connections screen choose the "HCS12 Serial Monitor."

1.4) You will need to define all of the *.C source code files needed to build your project. You will need SESTEST.C (or whatever you are calling it now), so click on the "Project→Add" option and select your SESTEST.C file. In addition, you will need to add the files ADS12.C, PWMS12.C, ROACHLIB.C, SES.C, TIMERS12.C, and TERMIO.C located in C:\CMPE118\SRC. To build your project, all you need to do now is click the "Project→Make" option. If there are any errors or problems, CW12 will display them at this time. If there are errors with your code, you will need to correct them all before your code will function (of course), so you should go back to the Editor window with your source code, correct the errors that the Status window revealed, and then try re-building your project. Repeat as required to remove all errors and warnings. To enable debugging information for each source file enable the "Bug" checkbox option next each file listed in the project window.

☐ 1.4) When you are ready for the next step, plug the RS232 (serial) connector into the CockRoach connector (on top of the CockRoach). Make sure you have the battery power connected and switched to the "ON" position (the toggle switch next to the CockRoach wheel).

☐ 1.5) You are now ready to download your new code. To do this, you will need to open uBug12. This is found in the "Start → Programs → Technological Arts → uBug 12" menu. You should now see a terminal window with a command prompt at the bottom. Set SW2 on the MC9S12C32 to the BOOT position and reset the chip. In the uBug12 command prompt enter **CON 1** assuming you are using COM port 1. uBug12 should now display a short message. To erase any existing program use **fbulk**. Now, everything is ready for the new code to be downloaded. You need to tell uBug12 which file to download, so enter **fload ;b** and select your file, which should be named something like MONITOR.ABS.S19 (the *.S19 file is compiled code in the proper format for downloading) and located in the BIN directory of your project. Once your code is finished downloading, the uBug12 will inform you, and you should see a prompt. You are now ready to execute your code.

☐ 1.6) At this point, you are ready to try out your program. **Be sure that the CockRoach is up on the blocks so that when the motors start running it will not drive off of the bench**. When the download was complete, you should have gotten the message LOADED OKAY in the terminal window. To run your program, type in "go" at the prompt and press <ENTER>. This tells the MC9S12C32 to execute your program, which has been loaded into memory. If all has gone well, the CockRoach will begin executing your code.

☐ 1.7) Congratulations! You've just written and downloaded your first embedded program.

☐ 1.8) Next, spend some time exploring the library functions provided for you. Write some simple programs to turn the motors on & off, get a character from the keyboard, print a value to the screen, read the light sensor and test the bumper switches.

<u>In the report:</u> Include the listings from the programs that you wrote to explore the libraries. (Note that we encourage you to print this two to a page and double sided to save paper; you may also use the printer in the CMPE-118 for this).

Part 2 – A Mechanical CockRoach

Reading: Carryer, Kenny, Olhine (CKO) Ch. 5 (Make sure you understand Ch. 1-4)

Background: The behavior of many types of simple creatures can be described in terms of their response to outside stimulus. This will be your opportunity to give a mechanical creature an interesting behavior.

Assignment: This is your mission: you are to write a C program that will run on the MC9S12C32 and cause the CockRoach 3000 to behave like a real cockroach.

The fundamental behaviors are that they:
- run around when the lights are on
- hide in the dark
- won't get stuck in corners or against chair legs, etc.

☐ 2.1) Begin by writing some simple programs to explore the inputs and outputs to/from the 68HC12. Use the library functions provided to control the motors, read the bumper states, and read the light level.

☐ 2.2) Complete the detailed design and implementation of the program that you worked out in Part 0.

☐ 2.3) Write and test your program to imitate a cockroach. Be sure to develop and test incrementally to avoid a huge headache! Use typedef enum to name your states!

☐ 2.4) Get your functional roach checked off by the TA, Tutor, or Professor.

In the report: Include a complete description of the design and implementation of your program. This should include high-level descriptions (like a state diagram) as well as the PDL (Program Design Language) or Pseudo-Code and the final C code that resulted.

Part 3 – A 'Smarter' CockRoach

Reading: Nothing new.

Assignment: Now that you have the basic functions down pat, see if you can make it more interesting to watch by endowing the machine with a more complex behavior. Use your imagination, but as a suggestion I will offer the idea of searching for the dark by following some recognizable pattern.

Part 4 – PCB Assembly and Samples

Reading/Viewing:
- http://www.instructables.com/id/How-to-Solder-Videos%3a-Why-is-soldering-difficult-s/
- http://blog.makezine.com/archive/2007/01/soldering_tutor_1.html
- http://www.elexp.com/t_solder.htm
- http://www.sparkfun.com/commerce/tutorial_info.php?tutorials_id=59
- http://www.sparkfun.com/commerce/tutorial_info.php?tutorials_id=58

Assignment: Later in the class, everyone will need motor driver boards and analog voltage followers to protect the larger I/O boards. This assignment is to assemble and debug one of the supplied PCBs. Boards and parts will be distributed in lab after a brief soldering demo. Another useful resource for this class (and further in your engineering career) is free samples. We'll get you started here on ordering free samples and resupply some parts for the class at the same time.

☐ 4.1) Start assembly only after becoming comfortable soldering and after examining a completed board of the same type. Remember to only solder sockets, not the full ICs.

4.2) If you are feeling extra ambitious, you can do some surface mount solder paste and skillet to get a feel for how to do this. If you plan to do some surface mount fabrication with PCBs in the future, this is a very good technology to get a handle on. Basically, you can cut your production time for a fairly complex design to well below a week, and including parts which are very challenging to hand solder.

☐ 4.3) Show your completed board to the TA, Tutor, or Professor for check-off.

☐ 4.3) Go to the Texas Instruments website (www.TI.com) and make an account for yourself, order some parts as free samples (for instance op-amps and comparators might be useful later). For a single supply op-amp, navigate to the part page by following this link: http://focus.ti.com/docs/prod/folders/print/tl3414a.html and clicking on the samples button that corresponds to the line TL3414A in the PDIP package (you can get them to send you up to 10). You will need to set up an account (free). Parts should be sent to your own mailing addresses (they will send it FedEx), if you cannot do that, have it sent to: Computer Engineering, 1156 High Street – SOE3, Santa Cruz, CA 95064-1077. Look around TI's website for any parts you think might be useful later on. Motor drivers, filter chips, and rail to rail op-amps are always useful. You will notice that most of the parts they sample are surface mount, and these can be an issue to prototype with. We'll talk about ways of dealing with this in class.

4.4) Navigate to the Allegro site and request samples for the 16-lead SOIC of the 3949 Motor Driver and the A3982 Stepper Driver in 24-lead SOIC: http://www.allegromicro.com/en/. Again, take a look around and see if there are other things they sample that you are interested in having in your parts bins.

4.5) For the report, include a printout of the parts that were sampled, and when they come in, please bring them to the 118 TAs or Professor (only the motor drivers, the rest you keep for your lab kits).

Notes on writing your code

As you may have noticed from studying the example code provided, there are a number of things that you must do in order to have access to the libraries provided for programming the 'CockRoach 3000'. In particular, notice the header files that need to be included:

```
#include <mc9s12c32.h>
#include <roachlib.h>
#include <ses.h>
#include <timers12.h>
#include <stdio.h>
```

These header files are located in `C:\CMPE118\INCLUDE` (except for mc9s12c32.h) and the compiler is aware of them through the `LIBRARYPATH` environmental variable.

`mc9s12c32.h` brings in a set of definitions and prototypes that are used throughout the CMPE118 labs, it should always be included in your programs.

`roachlib.h` is the header file for a simple set of input and output routines provided for use on the CockRoach 3000. This gives you access to functions that control the motors, read the state of the bumpers, and read the state of the light sensor.

`ses.h` is the header file for the Software Events and Services library. This library provides access to a pre-defined and convenient implementation of the Events and Services Framework.

`timers12.h` is the header file for the Timer library. This library provides access to several timers and several useful functions associated with initializing, setting, and checking them.

Use the `typdef enum` construct for code readability and understanding. `Enum` is short for *enumerated constants*, and is used as follows: `typdef enum colors {red, blue, green, brown};` `colors` is now a variable type (just like char or int), and has only the declared types (red, blue, green, brown) as valid types. This is especially useful for state machines, where the state variable should have well named enums (reversing, driving_forward, turning_right, stopped, etc.).

Lab #0

Time Summary

Be sure to turn this in with your lab report

This information is being gathered solely to produce statistical information to help improve the lab assignments.

Pre-Lab	Preparing Outside of the lab ____________	In the lab working this part__________
Part 1	Preparing Outside of the lab ____________	In the lab working this part__________
Part 2	Preparing Outside of the lab ____________	In the lab working this part__________
Part 3	Preparing Outside of the lab ____________	In the lab working this part__________
Part 4	Preparing Outside of the lab ____________	In the lab working this part__________
Report	Preparing the Lab Report ______________	