

ELE538 Microprocessor Systems Laboratory Outline

Peter Hiscocks, Professor

Department of Electrical and Computer Engineering
Ryerson Polytechnic University

phiscock@ee.ryerson.ca

August 25, 2002

Contents

1 Learning Objectives	1
2 Overview	2
3 Surviving This Course	2
4 Care of the Equipment	3
5 Evaluation	3
6 References	4
7 Acknowledgements	5
8 Lab Topics and Schedule	6

1 Learning Objectives

The objectives of this laboratory are as follows:

- Through the use of assembly language, to become thoroughly familiar with common elements of microprocessor software and hardware.
- To learn debugging techniques for microcontroller programs, including breakpoints, status readouts, single-stepping, and crash dumps.
- To be able to relate electrical signals to software operation and vice-versa, in order to debug hardware-software interaction problems.
- To learn good practices in structuring a microprocessor control problem including the structure of the main loop and communication between routines.
- To progress from simple programs to a moderately complex control program.

2 Overview

The laboratory is based around a 68HC11 microcomputer development system, the MPP Board Version 1, together with a small mobile robot, the *eebot*.

The robot includes a complete microprocessor development system, the MPP Board Version 2, interfaced to the robot motors and sensors. The robot can be operated from an AC adaptor or from an internal 9.6 volt rechargeable battery.

The course consists of two basic phases. In the first phase, programming *eebot* begins with basic exercises such as reading the sensor array and controlling the speed and direction of two DC drive motors. In the second phase, the student will program the robot to follow a black line, detecting and making decisions at line branches. An obstacle course will be set up and robots will be required to detect and follow a black line that leads to an objective location.

Software modules developed in the first phase of the course will be re-used to form the basis of the guidance program in the second phase of the course. You will develop a library of software routines that can be used to construct a major program.

3 Surviving This Course

The 538 lab exercises are challenging and will require significant student effort. You will need to learn to use a number of computer tools and to understand the architecture and programming language of a moderately complex microprocessor.

In many cases, it is unavoidable that the lab exercise requires some knowledge of material which has not yet been covered in the lecture. In such cases, the lab supervisor may give a tutorial on that material in the lecture. The lab time is too short to spend time on a tutorial.

In order to provide the necessary background, the lab notes are extensive and contain tutorial material that must be read and understood to complete the lab exercises. In other words, the answer to '*Do I need to read all this stuff?*' is *Only if you would like your program to work.*

Each lab is typically 17 or so pages in length, so it is not the kind of thing that can be covered in a few minutes. In preparation for each lab, you should read over the lab material and use an editor to create any software that you will be testing and debugging. It is not an effective use of lab time to be typing in a computer program.

Lab time is primarily intended for getting help in debugging problems that you cannot solve on your own, and demonstrating programs that (finally!) work correctly.

The ELE538 lab is open for student use during regular school hours, and students are welcome to use the lab when a station is available. If the lab is in use by another section, check with the lab supervisor. However, there are usually no spare stations available during a scheduled lab section so you should be prepared to leave until the scheduled section is over.

Many of the exercises can be tested and debugged to some extent on the MPP Version 1 systems on the lab benches. You can at least assemble your source code in advance of a lab to ensure that any syntax errors are corrected.

There is also two *simulators* available for your use. The simulator is a program that runs on a PC, usually under the Windows operating system, and executes 68HC11 machine code. If you are debugging a routine that does not require access to the robot hardware, then a simulator can be very useful. Check with the lab supervisors on the location and availability of the simulators.

In summary

- Read the course outline (this document).

- Don't panic. There will be a flood of new terms and information that you will have to digest in a very short time, especially at the beginning of the course. However, everyone who puts effort into this course will pass.
- If something is unclear, ask your prof or lab supervisor - they know where the information can be found.
- Print out and read the *eebot Technical Description manual* which is in the course directory `~courses/ele538/labs/eebot-spec`. Notice especially the last section on testing the hardware from the Monitor program, which can be very helpful when something doesn't work according to plan.
- Print out and read each lab before the lab.
- Learn to read the `listing` file (suffix `.lst`) and how to use it with breakpoints to debug a computer program. Rely on systematic methods to find problems: trial and error will not work in systems this complicated.
- Get a copy of the textbook and use it to answer technical questions.
- Do the lab assignments. This is not only to get the lab marks, but also to prepare you for the exam in this course.

4 Care of the Equipment

Many courses that involve mobile robot exercises require students to construct their own robot. This is undoubtedly an educational process but because of time constraints it means that much of the focus is on the mechanical and electrical design and construction of the bot.

In order to focus on programming issues, we have decided to provide students with a 'ready to run' mobile robot, the *eebot*. The robot includes a complete microprocessor development system, interfaced to the robot motors and sensors. The robot can be operated from an AC adaptor or from an internal rechargeable battery.

The robots will be signed out to students at the beginning of each lab session, and then returned and signed in at the end of the lab session.

The robots themselves are delicate and we do not have many spares. It may take some time to obtain replacement parts for a damaged robot. Consequently, if there is repeated damage to the robots, there may be a point at which a robot is not available to you when it is needed.

If the robot drive motors are inadvertently started there is the possibility of driving the robot off the lab bench. When testing software, ensure that robot mainframe is lifted up slightly so that the wheels of the robot are not in contact with the desk.

Please treat the robot as a delicate device. If you notice anything that is defective on the robot, such as a broken wire, loose connector or broken mechanical part, please draw it to the attention of your lab supervisor.

Regardless of the appearance, the robot is not a toy. It is only to be run for the purpose of testing software that is part of the official lab curriculum. It is not suitable for racing or taking part in robot sumo wrestling competitions.

5 Evaluation

The marking scheme for ELE538 is as follows:

Total lab work	35%
Mid-term exam	20%
Final examination	45%

You must obtain at least 18% in the lab work **and** 33% in the combined lecture examinations to pass the course. (ie, you must pass *both* lab and lecture.)

Plagiarism

You are welcome and encouraged to discuss the lab assignments with other students, especially when you are stuck on some point. However, direct copying of other people's assignments or some part of an assignment, or providing material for other people to copy directly, is *plagiarism* and constitutes academic misconduct.

Students will submit a copy of their lab work at the time they demonstrate the work. Once the assignment is submitted, it may not be resubmitted or changed¹. An automated computer program will check for similarities between student assignments and will notify your lab supervisor in the event that a significant portion of your work is similar to that of someone else.

If the assignments are deemed by the lab supervisor to be partial or full copies of each other, the students involved will receive zero for that assignment and a note of the plagiarism will be entered into the lab records. At the discretion of the lab supervisor, there may also be a note of academic misconduct added to the student academic record. On a second occurrence, a note of the plagiarism will be entered into the lab records, a note of academic misconduct will be added to the student academic record and the Department Chair will be advised. The standard procedures for Academic Misconduct will then apply: you can read about it in the Ryerson Calendar.

Notice that there is no distinction between students providing material for copying and students using copied material.

We are strict about plagiarism because plagiarizers do not learn the material and it puts students who do not cheat at a disadvantage. Be advised that two students were suspended and one required to repeat the course in the 1999-2000 academic year because of plagiarism in this course.

6 References

bf eebot Technical Description

Peter Hiscocks, 2002

A complete technical description of the *eebot* mobile robot used in this course.

M68HC11 Reference Manual

Motorola Document M68HC11RM/AD REV 3, 1991

The authoritative source of information about the 68HC11 microprocessor.

68HC11 Microcontroller, Construction and Technical Manual

Peter Hiscocks, 2001

Technical information on the MPP Board, 68HC11 Microprocessor Development System

Information on programming and interfacing the MPP Board used at Ryerson and elsewhere. Available from Active Electronics in

M68HC11: An Introduction, Software and Hardware Interfacing

Han-Way Huang

Delmar Thompson, 2001

The course text. A basic text on the 68HC11 microprocessor.

¹Except in special circumstances, which will require the manual intervention of your lab supervisor.

7 Acknowledgements

Special thanks to Jim Koch for his help and supervision in getting the robot together, Jason Naughton for his help with software and Nripendra Malhotra for robot construction. Professor Ken Clowes provided a pre-processor which compensates for some of the deficiencies of the assembler. He has also written a simulator for the 68HC11.

8 Lab Topics and Schedule

Week	Microprocessor Topic	Laboratory Topic	Assignment
1	Introduction to microprocessor program development.	<ul style="list-style-type: none"> • Setting up the development environment • Using vt6811 to connect to the Buffalo Monitor • The HELP command • Memory Dump, Memory Modify • Memory-mapped registers (Devices) • Writing a small Assembly Language Program • Breakpoints and Debugging 	Write and demonstrate an assembly language program to multiply two integers
2	Useful Techniques for the Toolkit Library.	<ul style="list-style-type: none"> • Software Delay • Bit Mask Operations • Converting Hex to ASCII • Programming the Liquid Crystal Display 	Write an ASCII string to the LCDisplay.
3	Displaying Battery Voltage and Bumper Switch Status.	<ul style="list-style-type: none"> • Read the A/D Converter • Fixed point math • Binary number conversion to ASCII string • Display routine architecture and mechanics • Read bumper switches, threshold, display. 	Read the Frob knob or battery voltage and display on the LCD. Read the bumpers and indicate on the LCD if actuated.
4.	Hardware Timer and Interrupt	<ul style="list-style-type: none"> • Hardware timer mechanism • Overflow flag mechanism • Interrupt driven counting • Timed delay 	Demonstrate a hardware-timed delay.
5.	Robot Roaming Program	<ul style="list-style-type: none"> • Program Structure • Conditional branches • Motor on-off and direction control. • The <i>alive</i> indicator 	Demonstrate a program that controls the robot so that it roams with timed turns. A collision causes a timed reverse and turning.

Week	Microprocessor Topic	Laboratory Topic	Assignment
6.	Interrupts. Structure of interrupt software.	<ul style="list-style-type: none"> • Using the wheel-counters. • Counting wheel rotations as a distance measurement. • Displaying the counts. 	LCDisplay program monitors wheel rotation counts. Routine to execute specified wheel rotation count.
7.	Guider Scan. Reading inputs into status registers.	Reading and displaying the guider sensor values. Setting <i>light</i> and <i>dark</i> thresholds in EEPROM.	Display basic guider operation with symbol display on the LCDisplay: eg, <>LDDL. Module to scan guider sensors.
8.	Robot Guidance 1: Structural diagram of the robot software.	Planning. Structure of the Main Loop, Inputs, Outputs, Interrupt Routines. Passing information from interrupts to main routine in mailboxes.	Program design document: Pseudocode for the main loop. Headers for various routines.
9.	Robot Guidance 2: Main Guidance Routine.	Construction of the routine linking the guider and the motor driver.	Static display of motor speed control with guider inputs and knob setting speed.
10.	Robot Guidance 3: System integration	Integration of display, sensor and motor speed control routines into main loop. Enabling/disabling line following.	Motor guidance control integrated with bumper switch emergency stop and status displays on the LCDisplay.
11.	Debugging Week	Testing and troubleshooting the robot operation.	List of bugs found.
12.	Demonstration Week	Each student demonstrates their robot following a course through various hazards to a final objective. Silly awards are presented.	Final robot control program.