

Course Outline (F2011)

COE / BME - 538: Microprocessor Systems

Instructors	<table border="0"> <tr> <td style="padding-right: 20px;">Name:</td> <td>Vadim Geurkov</td> <td>Lev Kirischian</td> </tr> <tr> <td>Office:</td> <td>ENG-430</td> <td>ENG-432</td> </tr> <tr> <td>Phone:</td> <td>(416) 979-5000 ext 6088</td> <td>(416) 979-5000 ext 6076</td> </tr> <tr> <td>Email:</td> <td>vgeurkov@ee.ryerson.ca</td> <td>lkirisch@ee.ryerson.ca</td> </tr> <tr> <td>Office hours:</td> <td>Tuesday, 3 to 4 pm</td> <td>Tuesday, 3 to 4 pm</td> </tr> </table>	Name:	Vadim Geurkov	Lev Kirischian	Office:	ENG-430	ENG-432	Phone:	(416) 979-5000 ext 6088	(416) 979-5000 ext 6076	Email:	vgeurkov@ee.ryerson.ca	lkirisch@ee.ryerson.ca	Office hours:	Tuesday, 3 to 4 pm	Tuesday, 3 to 4 pm
Name:	Vadim Geurkov	Lev Kirischian														
Office:	ENG-430	ENG-432														
Phone:	(416) 979-5000 ext 6088	(416) 979-5000 ext 6076														
Email:	vgeurkov@ee.ryerson.ca	lkirisch@ee.ryerson.ca														
Office hours:	Tuesday, 3 to 4 pm	Tuesday, 3 to 4 pm														
Prerequisite	All required second-year courses															
Course Website	http://www.ee.ryerson.ca/~courses/coe538															
Compulsory Texts	<ol style="list-style-type: none"> H.-W. Huang, <i>HCS12/9S12: An Introduction to Software and Hardware Interfacing</i>, Delmar Cengage Learning, 2010. K. Clowes, <i>Microprocessor Systems, Selected Course Notes</i>. 															
Reference Texts	<ol style="list-style-type: none"> F. Cady, <i>Microcontrollers and Microcomputers: Principles of Software and Hardware Engineering</i>, Oxford University Press, 2010. J. Valvano, <i>Introduction to Embedded Systems: Interfacing to the Freescale 9S12</i>, Cengage Learning, 2010. M. Mazidi, D. Causey and J. Mazidi, <i>HCS12 Microcontrollers and Embedded Systems</i>, Prentice Hall, 2009. 															
Calendar Description	<p>This course introduces students to small microprocessor-based systems, with an emphasis on embedded system hardware and software design. Topics will include microprocessor architecture and structure, with an overview of 8- and 16-bit systems; Assembly language programming and the use of high-level languages; basic input/output including parallel communications with and without handshaking and serial protocols; hardware and software protocols and timing; using interrupts and exceptions; overview of single-chip microprocessors and controllers with an emphasis on the Motorola HCS12; the internal structure and design of peripheral devices; memory system design and analysis; the use and structure of development tools such as (cross) assemblers and compilers, monitor programs, simulators, emulators, etc.</p>															
Learning Objectives	<p>Through the use of assembly language, by the end of the course students will become thoroughly familiar with the elements of microprocessor software and hardware. They will be able to:</p> <ol style="list-style-type: none"> Use technical knowledge including microprocessor architecture, microprocessor I/O interface and peripherals, assembly/C language programming and debugging methodology. Use design tools and related resources including microprocessor hardware and software, microprocessor peripherals, assemblers, C compilers, and monitor programs. (4a) Learn debugging techniques for microcontroller programs, including breakpoints, status 															

readouts, single-stepping, and crash dumps. Learn to debug hardware/software interaction problems. Learn good practices in structuring a microprocessor control program. Progress from simple programs to a moderately complex control program. Apply the programming principles including top-down programming, bottom-up programming, and functional programming to define an accurate programming problem statement. Recognize that good problem definition assists the program design process. (4b)

3. Describe differences between the various approaches that can be used to solve a microprocessor programming problem using assembly/C language. Select one specific approach to solve the problem. When the selected approach fails to solve the problem satisfactorily, analyze the cause of failure using standard assembly/C language programming and debugging methodologies. Based on the analysis, come up with new suggestions to improve the existing approach. Integrate the new suggestions into the existing design plan. Judge the completeness and quality of the generated solutions using standard assembly/C language programming and debugging methodologies. (4d)
4. Describe the iterative process of programming and debugging microprocessor programs using assembly/C language. Using debugging tools to generate information on the current state of an assembly/C language program that can be used to modify and improve the current program solution. Based on the generated information, examine and critique the current program solution to revise the solution as needed. Incorporate and integrate feedback from the teaching assistants and others and generate new knowledge about the programming problem. (4h)
5. Produce lab and project reports using appropriate format, grammar, and citation styles for technical and non-technical audiences. (7a)
6. Illustrate concepts including the structure of assembly/C language programs and obtain experimental results. (7d)

Note: Numbers in parentheses (e.g. 4a) refer to the graduate attributes required by the Canadian Engineering Accreditation Board. For more information, see:

<http://www.ryerson.ca/feas/programs/qa/gradattributes.html>

http://www.engineerscanada.ca/e/pr_accreditation.cfm

Course Organization

3 hours lecture, 2 hours laboratory each week

Course Evaluation

Labs 1-7	- weeks 2-8	20%
Lab Project	- weeks 9-13	10%
Lab Project Report	- week 13	5%
Quiz	- week 4 (covers material up to end of week 3)	5%
Mid-Term Test	- week 8 (covers material up to end of week 6)	20%
Final Exam	- week 14 (covers material up to end of week 13)	40%

Examinations

Quiz, Mid-Term Test, Final Exam

Project

The lab assignments and the project (originally developed by [Prof. Peter Hiscocks](#)) involve a robot. The project is to program the *eebot* mobile robot with a navigation system that can find its way through a maze, reverse, and back its way out again. A possible variation on this is that the robot first learns the maze. Then it is started again at the beginning and should navigate the maze without errors. The project must be demonstrated during the Demonstration Weeks. The project report must be submitted on or before the end day of the semester. At the time of demonstration, students will also be required to submit the project source code electronically.

Course Content

The following table summarizes the lecture topics for the course. Numbers next to topics refer to the section numbers in *Huang* textbook (see course references) that cover the topic.

Topic	Sections	Hours	Topic, description
Introduction to the HCS12 Microcontroller	1.3~1.11	3	Introduction to COE 538 Scope and objectives Management <ul style="list-style-type: none"> - 1.3 Computer Hardware Organization - 1.5 Memory system Operation - 1.6 Program Execution - 1.8 The HCS12 CPU Registers - 1.9 HCS12 Addressing Modes - 1.11 A Sample of HCS12 Instructions
HCS12 Assembly Programming	2.2~2.10	3	<ul style="list-style-type: none"> - 2.2 Assembly Language Program Structure - 2.3 Assembly Directives - 2.5 Writing Programs to Do Arithmetic - 2.6 Program Loops - 2.7 Shift and Rotate Instructions - 2.8 Boolean Logic Instructions
HW/SW Development Tools for HCS12	3.2, 3.8		<ul style="list-style-type: none"> - 2.9 Bit Test and Manipulate Instruction - 2.10 Program Execution Time - 3.2 Development Tools for the HCS12 - 3.8 Using CodeWarrior
Advanced Assembly Programming	4.3~4.11	3	<ul style="list-style-type: none"> - 4.3 Stack - 4.4 What Is a Subroutine? - 4.6 The Stack Frame - 4.7 Mathematical Subroutines - 4.9 Subroutines for Creating Time Delay - 4.10 Introduction to Parallel I/O Port and Simple I/O Devices - 4.11 Simple I/O Devices
Advanced Parallel I/O	7.5~7.9		<ul style="list-style-type: none"> - 7.5 The HCS12 Parallel Ports - 7.7 Liquid Crystal Displays (LCDs) - 7.8 The HD4478U LCD Controller - 7.9 Interfacing Parallel Ports to a Keypad
Analog-to-Digital Converter	12.2~12.6	3	<ul style="list-style-type: none"> - 12.2 Basics of A/D Conversion - 12.3 The HCS12 A/D converter - 12.4 The Functioning of the ATD Module - 12.5 Procedure for Performing A/D Conversion
C Language Programming	5.3~5.11	3	<ul style="list-style-type: none"> - 5.3 Types, Operators, and Expressions - 5.4 Control Flow - 5.5 Input and Output - 5.6 Functions and Program Structure - 5.7 Pointers, Arrays, Structures, and Unions - 5.8 Writing C Programs to Perform Simple I/O - 5.11 Using the CodeWarrior IDE to Develop C Programs

Topic	Sections	Hours	Topic, description
Interrupts, Clock Generation, and Operation Modes	6.2~6.7	3	<ul style="list-style-type: none"> - 6.2 Fundamental Concepts of Interrupts - 6.3 Resets - 6.4 HCS12 Exceptions - 6.6 Clock and Reset Generation Block - 6.7 Real-Time Interrupt - 6.11 HCS12 Operation Modes
Timer Functions	8.3~8.7	3	<ul style="list-style-type: none"> - 8.3 Standard Timer Module - 8.4 Timer Counter Register - 8.5 Input-Capture Function - 8.6 Output-Compare Function - 8.7 Pulse Accumulator
Timer Functions	8.8~8.11	3	<ul style="list-style-type: none"> - 8.8 Modulus Down counter - 8.10 Pulse-Width Modulation Function - 8.11 DC Motor Control
Serial Communication Interface	9.3~9.9	3	<ul style="list-style-type: none"> - 9.3 The RS-232 Standard - 9.4 The HCS12 SCI - 9.6 The SCI Operation - 9.8 Flow Control of USART in Asynchronous Mode - 9.9 Interfacing SCI with TIA-232
The SPI Function	10.2~10.8	3	<ul style="list-style-type: none"> - 10.2 Introduction to the SPI Function - 10.3 Registers Related to the SPI Subsystem - 10.4 SPI Operation - 10.5 SPI circuit connection - 10.6 Configuration / Data Transfer in SPI - 10.8 The 74HC595 Shift Register
Inter-Integrated Circuit (I ² C) Interface	11.2~11.6	3	<ul style="list-style-type: none"> - 11.2 The I²C Protocol - 11.3 An Overview of the HCS12 I²C Module - 11.4 Registers for I²C Operation - 11.5 Programming the I²C Module - 11.6 The Serial Real-Time Clock DS1307
Internal Memory Configuration and External Expansion	14.3~14.10	3	<ul style="list-style-type: none"> - 14.3 Internal Resource Remapping - 14.4 Expanded Memory Mapping - 14.5 On-Chip Flash Memory - 14.6 The On-Chip EEPROM Memory - 14.7 HCS12 External Memory Interface - 14.9 Memory Devices - 14.10 Example of External Memory Expansion for the HCS12
Review	1.3~14.10	3	Review and Catch-Up
			Final Examination

Laboratory/Tutorials

Lab/ Week	Title	Room
1/2	Using the Serial Monitor Program & Introduction to Assembly Language Programming	ENG-411
2/3	Programming the Liquid Crystal Display	---“---
3/4	Battery and Bumper Displays	---“---
4/5	Motor Control & Using the Hardware Timer	---“---
5/6	Robot Roaming Program	---“---
6/7	Wheel Counter Interrupt	---“---
7/8	The <i>eebot</i> Guider.	---“---
P/9~13	Project: Robot Guidance Challenge	---“---

Lab Management

Labs will be graded 10 marks maximum for the 1st lab, 20 marks for 2nd lab, 25 marks for 3rd lab, 20 marks for 4th lab, 30 marks for 5th lab, 20 marks for the 6th lab, and 15 marks for 7th lab to a maximum of 140 marks which will be scaled to 20% of the final mark. Credit for labs will be based on the quality of how well the project works (*demonstration*) and how well the student can *answer questions* about the lab. If answers to these questions are inadequate, the lab will be marked as 0, although the student will be given an opportunity to rectify his or her preparation. Partial marks may be assigned at the discretion of the instructor.

The Lab Project accounts for 15% of the final mark. The project must be demonstrated during the Demonstration Weeks. The project report must be submitted on or before the end day of the semester. It must include:

- A formal description of the work (at least 2 pages, no more than 5 pages)
- An appendix containing a hard copy of all source code (.asm or .c files)

The proper report description should address the following:

- Overall approach and description of performance
- Main design decisions
- Problems encountered and their solutions
- Recommendations: how you would continue the project to make it even better and how you would try to fix any remaining bugs.

At the time of demonstration, you will also be required to submit your source code electronically. (You will be told how to do this.)

The Project Evaluation will be done according to the following table:

Evaluation of	Item	Maximum mark
Lab Project (10%)	Basic functionality	4.5%
	Code quality	4.5%
	Extra functionality	1.0%
Lab Project Report (5%)	Report English quality	2.5%
	Report technical quality	2.5%

All the labs and the project are done individually. Each student must also keep a complete and continuous record in a binder of the year's lab activities.

Equipment should not be moved during the lab; if you believe equipment to be defective, report it to the lab instructor who will take care of the problem.

Important Notes

1. All of the required course specific written reports will be assessed not only on their technical or academic merit, but also on the communication skills of the author as exhibited through these reports.
2. To obtain a passing grade in the course, a student must obtain at least 50% in both the lab and theory portions of the course.
3. Laboratories are conducted using a Motorola HCS12-based microprocessor board and computer-aided design tools from Freescale: [Special Edition: CodeWarrior for HCS12\(X\) Microcontrollers \(Classic\)](#)