# Carleton University

## Department of Systems and Computer Engineering

***SYSC 2003*** ***Introduction to Real-Time Systems*** ***Winter 2013***

## Course Outline

**Instructor*:***

Professor Gabriel A. Wainer. Room 3216 VS

**Course Description and Objectives:**

Real-time systems are designed to interact with their environment in a time-critical manner. SYSC 2003 provides an introduction to the fundamental principles of these systems. It also presents different issues associated with using a single processor computer system to implement a real-time system. The objectives of this course are to:

- introduce the development of real-time embedded systems
- introduce the event-driven nature of real-time systems
- introduce hardware interrupts as the sources of asynchronous events
- introduce the concept of concurrency in a single processor computer system
- provide practical laboratory exercises that reinforce an event-driven perspective in the development of embedded real-time systems
- provide essential prerequisite material for further studies, including:
  - SYSC 3601 Microprocessor Systems
  - SYSC 4001 Operating Systems
  - SYSC 3033 Real-Time Concurrent Systems

This course builds on the concepts introduced in ECOR 1606 and SYSC 2001. The main differences between SYSC 2003 and the prerequisite courses are:

- the paradigm shift from a sequential mindset (ECOR 1606) to an event-driven mindset (SYSC 2003)
- programming embedded systems in an specialized development platform
- programming at the hardware/software interface
- the design and implementation issues associated with concurrency and event-driven systems

The course will provide practical experience in low-level programming in an embedded platform. This environment is based on the Motorola 68HC12 microcontroller, which will be expanded with a Project Board containing specialized input/output devices and interfaces with the environment. We will follow a development methodology based on developing incremental prototypes running in a simulated environment that will be later incorporated to the real environment running in the embedded platform.

The development will be done using Assembly language (miniIDE Cross Assembly) and ANSI C programming (ImageCraft ICC12 compiler for the 68HC12 with NoICE C-level debugger). The software will be executed using the simHC12 simulator for development and testing, and the embedded systems laboratory for final testing and execution.

The development experience is intended to illustrate the course material; however, producing expert assembly language programmers is not an objective. The concepts that will be covered are applicable to a wide range of computer systems, programming languages, and applications.

**Prerequisites:** **SYSC 2001**

**Textbook:** The HCS12 / 9S12: An Introduction to Software and Hardware Interfacing. H. Hwang. 2nd Edition. Delmar Cengage Learning

## Course Outline

**Web Site**
Course materials will be available on the SYSC-2003 Website: http://www.sce.carleton.ca/courses/sysc-2003

**Laboratory and Assignments:**

The laboratory is made up of five graded assignments. Assignments will be posted on the course webpage. Each assignment must be handed in on (or before) the due date and time. Late assignments will not be accepted without a valid medical certificate.

The computer lab is open seven days a week, whenever the building is open. You may use the lab at any time, except for those timeslots when the lab is reserved for other courses. Tutorial lab sessions are scheduled so that you may meet with your TA for assistance with assignments.

**Note that the probability that machines will be busy increases as each due date approaches.**

**Exams:**
There will be on closed book, no calculators, mid-term test. Arrangements for the mid-term test will be announced during the class. A closed book, no calculators, final exam will be held during the University's formal examination period.

*The Final Exam is for evaluation purposes only and will not be returned to the student.*

Students who miss an assignment or midterm must present a valid medical certificate to the instructor within a reasonable time after the deadline or midterm; otherwise, the student will receive a zero for that item.

**Medical Certificates:**
A medical certificate must adhere to the format required by the Registrar. The format is available as a PDF form through the Registrar's website http://www.carleton.ca/registrar/forms.htm. All medical certificates must be presented immediately upon return from the illness; they will not be accepted after the fact.

For assignments: Assignments are to be worked on throughout the week, not just the day they are due. If you miss an assignment, you will be given a mark based on your partial work if a valid medical certificate is given. If you cannot provide your partial solution, you will be given zero unless you have a valid medical certificate showing prolonged illness. If the midterm is missed, with a valid medical certificate, the Final exam mark will be given as the midterm mark.

**Marking Scheme:**

To pass the course, a student must (1) pass the final examination (D- or better) **and** (2) obtain an overall passing average (assignments plus midterms plus final exam). For students who pass the final exam, the final grade will be calculated as follows:

| | |
|---|---|
| Assignments: | 25% (5% each) |
| Mid-term test: | 24% |
| Final exam: | 51% |

**- Students with disabilities requiring academic accommodations** in this course must register with the Paul Menton Centre for Students with Disabilities (PMC) for a formal evaluation of disability-related needs. Registered PMC students are required to contact the PMC, 613-520-6608, every term to ensure that Prof. Wainer receives your Letter of Accommodation, no later than two weeks before the first assignment is due or the first in-class test/midterm requiring accommodations. If you only require accommodations for your formally scheduled exam(s) in this course, please submit your request for accommodations to PMC before the official deadline.

**Plagiarism**:
Plagiarism (copying and handing in for credit someone else's work) is a serious instructional offense that will not be tolerated. Please refer to the section on instructional offenses in the Undergraduate Calendar for additional information.

Students are encouraged to discuss design issues when working on assignments; however, they are expected to write their own programs individually. There is a fine line between cooperating with your colleagues (discussing problems and ideas) and copying program code (plagiarism). Students are warned that the assignments form a very important part of this course – doing the assignments (by oneself) is by far the best way of learning the material. In this context, it should be noted that copying assignments is a self-defeating exercise. Any student who resorts to copying is not likely to do well on the mid-terms or final exam.

**Health and Safety**
Every student should have a copy of our Health and Safety Manual. An electronic version of the manual can be found at:

http://www.sce.carleton.ca/courses/health-and-safety.pdf

**Weekly schedule (tentative)**
1 – Course objectives, organization and administration; Lab information.
Introduction: What is an embedded system? What is a real time system?
2- Microprocessors vs. Microcontrollers. Programmer's Model of the 68HC12: Instruction Set, Memory, I/O. Introduction to the SimHC12 emulator.
Programming the 68HC12in assembly language. Instruction Set: format, addressing modes. Basic operations: data transfer, arithmetic, overflow.
3 – Assembler program definition : Directives. Basic operations (cont.): multiple-byte operations (carry condition register). Introduction to the MiniIDE.
Multiplication and Division. Control flow: conditions, jumps. Basic operations: shift and Boolean logic operations.
4 – Program execution time. Hardware and software development tools. Assembly 2-Pass Process, Linking, Locating, Cross-Assembly, Simulation, Execution.
Array manipulation. Strings. Structures.
5 – The stack. Introduction to subroutines. Subroutine definition policies. The stack frame. Parameter passing: by value, by reference.
6 – Introduction to ANSI C language programming using the ICC12 C compiler.
Mixing C and Assembly. Basic concepts of Input/Output. I/O Addressing. I/O synchronization. Overview of the 68HC12 Parallel Port.
7 – Interfacing with simple Input/Output devices: LEDs. Port Expansion on the Axiom Board. 7-segment display.Keypad with software debouncing techniques.
8 – Midterm 1 Review.

LCD displays. Stepper Motors.Interrupts: fundamental concepts. Interrupts, Resets, Exceptions. Interrupt Processing: masking, priorities.

9 – Developing interrupt software. Interrupt vector. Key Wakeup functions. Interrupt-driven keypad. Introduction to Interrupt Programming Issues. Atomic ASM instructions, randomness of interrupt occurrence.

10 – Interrupt Programming in C. Timer Functions. Real-time interrupt. Timer counter register. Output compare function.

11 –Input capture function. Pulse Accumulator. PWM function. DC motor control. The AD converter. Interfacing with a D/A converter. A/D programming: Polled, Interrupt driven.

12 – Real-Time Programming Issues: Reentrant Programming. Concurrency, critical sections, shared variables, mutual exclusion. Buffering : FIFO, FIFO Dynamics, Double Buffering The Serial Subsystems. Asynchronous Serial I/O, SCI. Introduction of work-done versus work-ready interrupts.
Process Model Introduction to real-time kernels. Multithreaded Preemptive Scheduler. Semaphores : Spin-Lock versus Blocking.

13 –Review and wrap up.

**Exams and Evaluations**

22/1 – Assignment 1
5/2 – Assignment 2
26/2 – Assignment 3
5/3 – **Midterm**
19/3 – Assignment 4
2/4 – Assignment 5