

This course concentrates on the problems encountered when attempting to use computers in real-time (RT) and embedded applications where the computer system must discern the state of the real world and react to it within stringent response-time constraints. Both design methodology and practical implementation techniques for RT systems are presented. Although some hardware will be involved, it should be noted that this course concentrates on real time software.

**Instructor:** Craig Scratchley

Email : Use your Canvas Inbox to reach me\*

Office hours : After lectures, and by appointment

**TAs:**

Yuhui Gao




Mohammad Soltanshah

Hossein Sheikhi Darani

**Lectures/Tutorials:**

	<a href="#">ENSC 351-D100 (6815)</a>	Embedded & RT System Software (Lecture)	77	Tu 12:30PM - 2:20PM	WMC3520	Sep 8, 2021- Dec 7, 2021
				Th 12:30PM - 2:20PM	AQ3181	Sep 8, 2021- Dec 7, 2021

**Lab Sessions:**

	<a href="#">ENSC 351-LA01 (6825)</a>	Embedded & RT System Software (Laboratory)	27	Tu 4:30PM - 8:20PM	ASB8800	Sep 8, 2021- Dec 7, 2021
	<a href="#">ENSC 351-LA02 (6826)</a>	Embedded & RT System Software (Laboratory)	32	We 4:30PM - 8:20PM	ASB8800	Sep 8, 2021- Dec 7, 2021
	<a href="#">ENSC 351-LA04 (6840)</a>	Embedded & RT System Software (Laboratory)	18	Fr 11:30AM - 3:20PM	**	Sep 8, 2021- Dec 7, 2021

**Piazza Discussion Board for the course:**

Course-related questions should be posted, usually publicly, to the Piazza discussion board available via Canvas. \*Messages of a technical nature sent directly to TAs and/or instructor may not be replied to.

## Topics:

### Intro to Embedded and Real-time System Software

- Examples
- Characteristics
- Collaboration Graph Notation
- Embedding real-time software

### “Standard of care” when designing real-time and embedded software

- to reduce legal, financial and safety risks.

### Multi-threaded Programming

- Single-threaded programming
- Multiple threads
  - Collaboration Graph Notation for threads
  - Joining threads
- Sharing data between threads in a process
  - Collaboration Graph Notation for data areas
- Synchronization
  - Mutexes
  - Condition Variables
  - Collaboration Graph Notation for synchronization
  - Barriers
- MultiCore Systems and Symmetric Multiprocessors (SMPs)
- Processes
- Shared Memory

### Testing Multithreaded Programs (time permitting)

- Developing Test Harnesses

### Scheduling

- Priorities
- Priority inversion
- Priority inheritance (technology permitting)
  - Complexity of priority inheritance
- Considering devices when scheduling
- Scheduling on multi-processors
- Rate-monotonic scheduling (time permitting)
- Earliest-deadline-first scheduling (time permitting)

### Clocks and Timers

- Timers
- Timeouts
- Collaboration Graph Notation for timeouts

### Intro to Atomic Operations

- Lock-free buffers
- Memory ordering

### Interrupts

- Interrupts and Interrupt Service Routines (ISRs)

- Collaboration Graph Notation for ISRs
- Interrupt Handling Functions (Handlers)
- Handlers on SMPs and need for spinlocks

## Device Drivers

**Textbooks:** Anthony Williams, C++ Concurrency in Action: Practical Multithreading, Manning Publications, 2012 or 2019 (Available from Amazon.ca or online at SFU library: [https://sfu-primo.hosted.exlibrisgroup.com/primo-explore/fulldisplay?docid=01SFUL\\_ALMA51189217430003611&context=L&vid=SFUL&lang=en\\_US&search\\_scope=default\\_scope&adaptor=Local%20Search%20Engine&tab=default\\_tab&query=any,contains,c%20%20%20concurrency%20in%20action&offset=0](https://sfu-primo.hosted.exlibrisgroup.com/primo-explore/fulldisplay?docid=01SFUL_ALMA51189217430003611&context=L&vid=SFUL&lang=en_US&search_scope=default_scope&adaptor=Local%20Search%20Engine&tab=default_tab&query=any,contains,c%20%20%20concurrency%20in%20action&offset=0))

<https://learning.oreilly.com/library/view/c-concurrency-in/9781933988771/>

2019:

<https://learning.oreilly.com/library/view/c-concurrency-in/9781617294693/?ar>

R.J.A. Buhr, R.S. Casselman, Use Case Maps for Object-Oriented Systems, Prentice Hall, 1996 (use version changed by me – on Canvas)

Scott Chacon and Ben Straub, Pro Git (2<sup>nd</sup> edition), Apress, 2014 (available from <https://git-scm.com/book/en/v2>) (I may possibly switch to a subversion book.)

One or more articles related to “standard of care” and the legal/financial and safety risks of embedded software.

A good book on C++. There are also a few in the library, perhaps on reserve. This is good:

Stanley B. Lippman, Josée Lajoie, Barbara E. Moo, C++ Primer, Addison-Wesley, 2012 (available online from SFU Library: [https://sfu-primo.hosted.exlibrisgroup.com/primo-explore/fulldisplay?docid=01SFUL\\_ALMA51189075050003611&context=L&vid=SFUL&lang=en\\_US&search\\_scope=default\\_scope&adaptor=Local%20Search%20Engine&isFrbr=true&tab=default\\_tab&query=any,contains,c%20%20%20primer&sortby=date&facet=frbrgroupid,include,399635676&offset=0](https://sfu-primo.hosted.exlibrisgroup.com/primo-explore/fulldisplay?docid=01SFUL_ALMA51189075050003611&context=L&vid=SFUL&lang=en_US&search_scope=default_scope&adaptor=Local%20Search%20Engine&isFrbr=true&tab=default_tab&query=any,contains,c%20%20%20primer&sortby=date&facet=frbrgroupid,include,399635676&offset=0))

Bjarne Stroustrup, The C++ programming language, Pearson, 2013.

Other resources as referenced by the instructor.

**Prerequisites:** ENSC 254.

**Grading :**

(if we **can** have in-person exams – my preference)

Multipart project (could include one or more practical examinations) and quizzes 45%,

Online discussion on “standard of care” and legal/financial and safety risks of embedded software 5%,

**\* AND \***

Mid-term and Final exam and possible quizzes 50%.

**\*OR\***

Extra project (max) 15%,

Mid-term and Final exam and possible quizzes (min) 35%.

(if we **cannot** have in-person exams)

Multipart project (about 6 parts normally done with a partner) and possible quizzes 70%,

Practical exam and possible quizzes 25%,

Online discussion on “standard of care” and legal/financial and safety risks of embedded software 5%,

Extra project (max) 15% (typically lowers the 75% above)

**Academic Dishonesty:**

Academic Dishonesty will not be tolerated in this course. Please see the relevant SFU policy documents on the SFU website. Submissions of work by students may be evaluated to determine if they have been plagiarized.

As I have done in the past, expect that I will recommend a harsh penalty, including the assignment of a grade of “FD” (failed – academic dishonesty) for the course, if you engage in academic dishonesty. I have reported a number of students over the past years, for example. Additional information can be found in the handout for Assignment 1, and perhaps other assignments as may be required.

**Attendance Expectations:**

Each student is expected to attend all lectures/tutorials. Some lab sessions may be required, and it is assumed that you will attend lab sessions to get help. You are welcome to come to extra lab sessions if TAs have time for you. Some material for the practical exam and project that cannot be found in the textbook(s) for the course will be covered in the lectures. If you are unable to attend any lecture/tutorial, please notify me, the instructor, on Canvas as early as possible. Details on how to notify me may be discussed and updated in class but otherwise please use your Canvas Inbox.