

1

3

5

Copyright © 2016 L.S. Marshall, Systems and Computer Engineering, Carleton University revised September 5th, 2016

Course Objectives To introduce students to the principles and practice of software development for systems that are are characterized by one or more of the following terms: real-time, concurrent, event-driven, and embedded. Although a specific implementation technology will be used to provide hands-on programming experience, the goal is to present techniques that are applicable to a diverse range of applications, hardware/software components, programming languages and operating systems.

2

SYSC 3303 - Course Introduction

Lecture, Lab, and Office Hours Schedule Section A: Dr. Lynn Marshall, Room ME4230,

- 24 Lectures: Mon/Wed 4:05-5:25pm: ME4499: Wed Sept 7th to Wed Dec 7th (although Fri Dec 9th will follow a Monday schedule, the current plan is no lecture that day)
- 11 Labs Mon/Tue 9:35-11:25am: CB5109; Mon 1:35-3:25pm: AA508: Mon Sept 12th to Tue Dec 8th (we may use the time between 3:25 and 3:55pm on the Monday afternoon lab; no lab Tue Oct 11th; we may use lab and lecture time on Fri Dec 9th, if needed)
- You may use any SCE lab whenever it is not reserved
- Office Hours: Mon 1:35-3:25pm in AA508 and by appointment

SYSC 3303 - Course Introduction

•

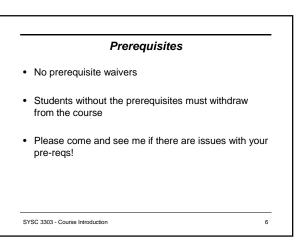
Lecture, Lab, and Office Hours Schedule • The "Term Calendar" document on the course web site has a summary of all lectures, labs, deliverables, etc., and should be studied closely!

Prerequisites

 Engineering students must have credit for: SYSC 2003

and

- SYSC 2004 (CSE / Comm Eng) or SYSC 2100 (SE)
- Computer Science students must have credit for COMP 2003 or COMP 2401 and
 - COMP 2002 or COMP 2402
- BIT and U of Ottawa: special requirements



Background Knowledge

- Experience has shown that students who do well in this course:
 - are competent at developing sequential, objectoriented programs in C++ or Java
 - have a solid understanding of fundamental abstract data types (e.g., bags, sets, lists, queues, stacks, maps) and elementary data structures (e.g., arrays, linked lists, trees)

7

9

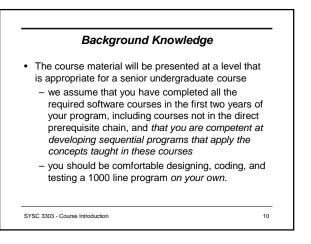
SYSC 3303 - Course Introduction

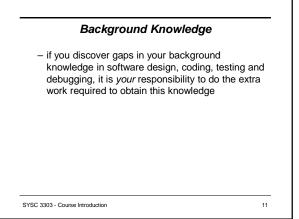
Background Knowledge - have a good understanding of the principles of computer system organization, as provided by SYSC 2001 or COMP 2003 - have some knowledge of concurrency, mutual exclusion and condition synchronization, as provided by: SYSC 2003 (background threads, interrupt service routines, shared buffers), or • SYSC 3001, COMP 3000 (concurrent processes/threads/tasks, semaphores) SYSC 3303 - Course Introduction 8

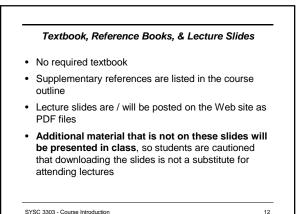
Background Knowledge

- are open to the idea that we can use pictorial modelling languages to reason about concurrent real-time programs as systems composed of interacting components (some concurrent, some not), at a level of abstraction above the coding details
 - · we assume that you are familiar with the UML for modelling small-scale sequential programs
 - · we'll introduce elements of the UML for modelling concurrent systems

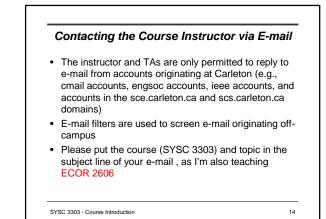
SYSC 3303 - Course Introduction











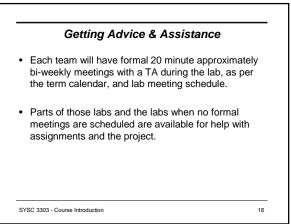
Getting Advice & Assistance Instructor office hours are given in this presentation and are posted on the course Web site - meetings at other times can sometimes be arranged Questions and comments of general interest (e.g., items related to the lectures, assignments & project) can be e-mailed to your instructor; however, these will normally be answered in the next class, not via email "Private" questions (e.g., PMC accommodations,

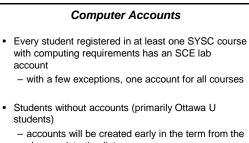
marks, illness, etc.) can be dealt with via e-mail or by booking an appointment

15

17

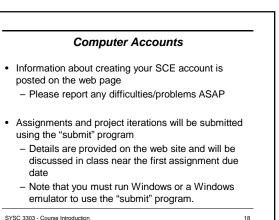
SYSC 3303 - Course Introduction

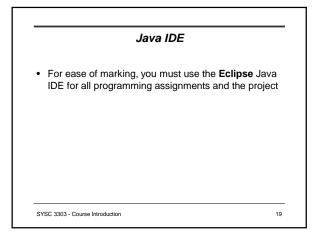


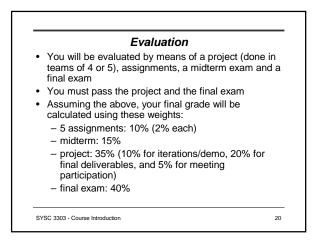


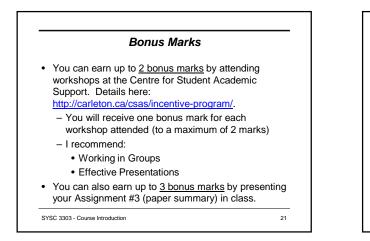
- class registration list - if you have not enrolled in this course, please do
- so ASAP, so that your account will be created

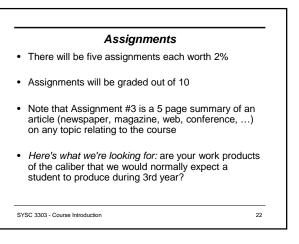
SYSC 3303 - Course Introduction

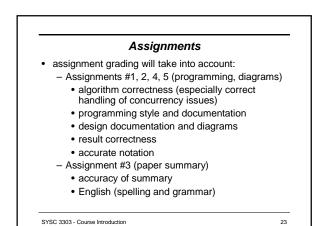




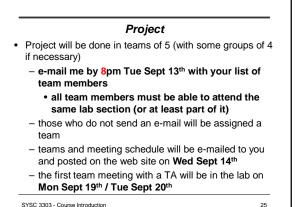






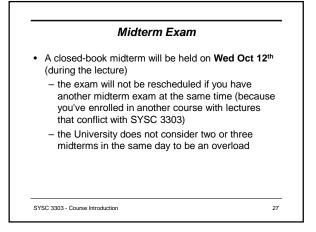


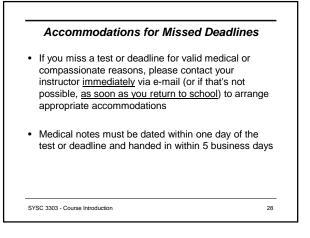
<section-header><list-item><list-item><list-item><list-item><list-item>



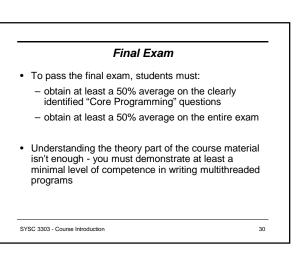
SYSC 3303 - Course Introduction

Project Students who refuse to join a team, or do not participate with their team-mates, will receive a project mark of 0 and a final grade of FND If it is apparent that not all team members participated equally, adjustments will be made to each team member's project mark SYSC 3303 - Course Introduction 26





Final Exam · A closed-book three-hour final exam will be held during the University's December examination period (December 10th to 22nd). With the exception of students who receive FND based on their refusal to participate in a project team, and those who did not write the midterm (or make-up essay), all students are eligible to write the final examination. Those who have less than 40% on the midterm and miss the final exam, will receive FND and thus be ineligible to apply to the Registrar's Office for deferral of the final examination SYSC 3303 - Course Introduction 29



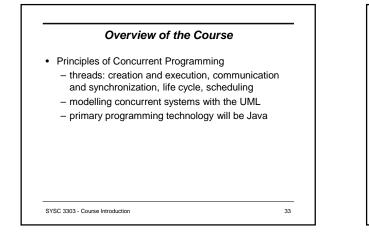
Final Exam

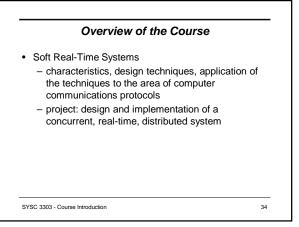
- The final exam is for evaluation purposes only and will not be returned to students
 - See the course outline for more details
- Deferred final exams: see the current Undergraduate Calendar, Academic Regulations of the University, Section 2.2, The Course Outline; Section 2.3, Standing in Courses/Grading System; and Section 2.5, Deferred Final Examinations

SYSC 3303 - Course Introduction

Accommodations The Faculty of Engineering requires students to have a conflict-free timetable, so requests to accommodate missed exams, assignment due dates, project milestones, etc. because of conflicts with other courses, jobs or vacation plans will not be considered · Students with disabilities - see the course outline, and the current Undergraduate Calendar, Academic Regulations of the University, Section 2.9 Students with religious obligations - see the current Undergraduate Calendar, Academic Regulations of the University, Section 2.10 32

SYSC 3303 - Course Introduction



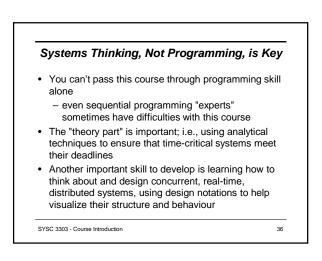


Overview of the Course · Advanced Topics - introduction to the theory of hard real-time systems and scheduling - very large scale software development and true multi-processing systems - other applications of real-time concurrent systems development - theory only, no hands-on programming

SYSC 3303 - Course Introduction

35

31



...But Don't Ignore the Programming Part

- Programming helps novice real-time system developers understand the run-time behaviour implied by the design diagrams
 - analogy: would you be be able to use the UML effectively to model large-scale, sequential OO programs if you hadn't first learned how to write OO programs in Java, C++, Smalltalk, etc.?
 - similarly, how can you model concurrent, eventdriven systems using abstract design notations if you don't have hands-on experience observing the run-time behaviour implied by the abstractions?

SYSC 3303 - Course Introduction

37

Programming Requires a Paradigm Shift

- The difficulties in developing a concurrent, real-time system arise because we can't code the desired overall system behaviour directly
 - behaviour emerges at run-time through the interaction of cooperating autonomous components
 - designing the components to work together is the challenge

38