Instructor: A. Cyrus Sabzevari

Email: asabzeva@gmu.edu

Office Hours: Monday 6:15 – 7:10 in the adjunct office (Engineering Building, room 3707 or 3708) or by appointment

Course Description:

This is a graduate level course in Distributed, Embedded and Real-Time Systems designed for real-time multiprocessing and distributed processing.

It discusses the theoretical and practical concepts in real-time systems with an emphasis on both hard real-time and soft real-time distributed multi-processing. Several operating systems (e.g. Linux, VxWorks), computer architectures and scheduling methods will be used to illustrate the concepts. The course provides students with hands-on experience in distributed real-time systems through projects which involve real-time software design, implantation and test.

Prerequisites:

- Knowledge of basic computer system organization and operating system concepts and multiprocessor computer architecture (ECE 511 Advanced Microprocessor Architecture).

- This class requires systems level design and analysis as well as computer programming. It also requires understanding and knowledge of basic multi-processor computer architecture. You need to program in C/C++ and Java (project and assignments). This course requires good software development and programming skills. Without good programming skills, students have traditionally done poorly.

Required Text:


Other Recommended Texts:


**Other recommended books and papers:**

**Time Triggered Protocol (TTP):**


**Controller Area Network:**


**Real-Time and Distributed Protocols and Design:**


**Distributed Systems and Operating Systems:**


**Linux and POSIX:**


Class Meeting Topics:

Overview:
Reading: lecture notes, Liu: chapters 1 and 2

Hard and Soft Real-Time Embedded Systems

Jobs and Processes
Release Times, Deadlines, Timing Constraints

Synchronization Mechanisms

Concept of Process
Concept of Thread
Mutual exclusion

Semaphores

Process Synchronization

Real-time Hardware
Reading: lecture notes, Laplante: chapters 1 and 2, Bertolotti & Manduchi: chapter 20

Processor Architecture

Memory Technologies

Pipelined Instruction Processing

Multi-Core Processors

CISC vs. RISC

I/O

DMA

Analog and Digital I/O

Distributed Real-Time Architectures

Control Area Network

Time Triggered Architecture

System-Level Design and Model of Real-Time Systems:

Reading: lecture notes, Liu: chapter 3

System Level Design
Incremental Design Process

Application Modeling

Processes and Resources

Periodic Task Model

Process Dependencies and Data Flow

Scheduling

**Distributed Hard Real-Time Systems with Time-Triggered Protocol and Controller Area Network:**

Reading: Liu sections 4.1, 5.1, 5.2, lecture notes, Paret: Part A.

Time-Triggered (Clock-Driven) vs. Event-Triggered (Priority-Driven)

Time-Driven Systems

Time Triggered Protocol (TTP)

Event-Driven Systems

Controller Area Network (CAN) Protocol

Sporadic Tasks Scheduling

Incremental Mapping and Scheduling and Strategies

**Approaches to Real-Time Scheduling, Time Driven Scheduling and Incremental Mapping, Event Driven Scheduling and Incremental mapping:**

Reading: lecture notes and Liu: chapter 4, Laplante: section 3.1, 3.2

Other recommended reading: Pop, Kopetz

Polled Loop and Delay

Cyclic Code Structure, Finite State Machines and Co-routines

Interrupts

Time Driven Scheduling

Event Driven Scheduling

Static Scheduling

Dynamic Schedules

Reconfiguration
Optimality of EDF and LST Scheduling Algorithms

Validating Priority Driven Systems

**Schedulability Analysis and Bus Access for Event-Driven Systems:**

Reading: lecture notes and Liu: chapter 5, 6.1-6.4

Other recommended reading: Pop, Kopetz

Fixed Priority

Dynamic Priority

Response Time Analysis

Schedulability Analysis for Distributed Systems

Bus Access Optimization

Mapping and Scheduling and Strategies

Sporadic Tasks Scheduling

Experimental Evaluation

**Introduction to POSIX Real-Time extension and Network Programming**

Reading: lecture notes

Other recommended reading: Gallmeister, Butenhof, Stevens.

Threads

Synchronization

Communication

Scheduling

Sockets

*Midterm exam – in class, closed books, closed notes, individual effort*

**Resource and Resource Access Control:**

Reading: lecture notes and Liu: 8.1-8.5, Laplante: 3.3

Contention and Resource Access Control

Non-preemptive Critical Sections

Priority Inheritance Protocol
Priority Ceiling Protocol

Memory Management Issues

**Operating Systems:**

Reading: lecture notes and Laplante: 3.1, 3.5

Other recommended reading: Bovet

Scheduling

System Services

Scheduling in Linux kernels

Scheduling in VxWorks 5.5

Soft vs. Hard Real-Time

Process Management

System Management

Clustering

**Real-Time Programming:**

Reading: Lecture notes and Laplante: chapter 4

Coding for Real-Time Software

Coding Standard for Real-Time Software

Procedural Languages

Modularity and Typing Issues

Parameter Passing and Dynamic Memory Allocation
Exception Handling
Programming Languages
Object Orient Programming
Design Patterns
Synchronizing Objects and Garbage Collection
Compiler Optimizations of Code

**Deadlock and Starvation, Distributed Mutual Exclusion, Theoretical Foundations of Distributed Systems:**

Reading: Lecture notes

Other recommended reading: Singhal, Bacon, Chow

Causes and handling of deadlock
Deadlock models

Resource models

Virtual Memory

Distributed Mutual Exclusion protocols

Limitations of a distributed system
Lamport’s logical clocks
Vector clocks
Causal ordering
Global state
Termination detection

Network Time Protocol (NTP)

**Final Exam – in class, closed books, closed notes, individual effort**

**Grading:**

There will be an in-class midterm and final. All exams are individual effort, closed book, closed notes, and no internet.
<table>
<thead>
<tr>
<th>Component</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>20%</td>
</tr>
<tr>
<td>Assignments</td>
<td>20%</td>
</tr>
<tr>
<td>Midterm exam</td>
<td>30%</td>
</tr>
<tr>
<td>Final exam</td>
<td>30%</td>
</tr>
</tbody>
</table>