EE445M Midterm Study Guide (Spring 2020):

Instructions:

- Open book and open notes.
- No calculators or any electronic devices (turn cell phones off).
- Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space (boxes) provided.
- *Anything outside the boxes will be ignored in grading.*
- For all questions, unless otherwise stated, find the most efficient (time, resources) solution.

Lecture notes: Lectures 1-6

Week	Notes	Topics
1	Lecture 1	Introduction - ARM architecture, instruction set, stack, µVision4 compiler, GPIO, timer, UART, device drivers
2	Lecture 2	Software design - Modular programming, call & data flow graphs, flowcharts, I/O synchronization Debugging - Lab environment, intrusiveness, monitor, output to scope, simulator
3	Lecture 3	RTOS - Multi-threading/-tasking, OS architecture OS kernel - Interrupt servicing, operating modes, context switching
4	Lecture 4	Threads - TCB, cooperative & preemptive multitasking, round-robin scheduler Thread communication & synchronization - Critical sections, reentrance, FIFO, mailbox
5	Lecture 5	Semaphores - Spinlock & blocking semaphores, monitors, deadlock Debugging - Testing, performance measures (response time, jitter, throughput)
6	Lecture 6	Scheduling - Real-time scheduling, priority scheduler, scheduling anomalies, fixed-rate scheduler

Book: Chapters 1-6 (except 3)

1. Computer Architecture

1.1. Introduction to Real-Time Operating Systems

1.1.1. Real-time operating systems

1.1.2. Embedded Systems

1.2. Computer Architecture

1.2.1. Computers, processors, and microcontrollers

1.2.2. Memory

1.3. Cortex-M Processor Architecture
    1.3.1. Registers
    1.3.2. Stack
    1.3.3. Operating modes
    1.3.4. Reset
    1.3.5. Clock system
1.4. Texas Instruments Cortex-M Microcontrollers
    1.4.1. Introduction to I/O
    1.4.2. Texas Instruments TM4C123 LaunchPad I/O pins
    ~~1.4.3. Texas Instruments TM4C1294 Connected LaunchPad I/O pins~~
    ~~1.4.4. Texas Instruments MSP432 LaunchPad I/O pins~~
    1.4.5. Interfacing to a LaunchPad
1.5. ARM Cortex-M Assembly Language
    1.5.1. Syntax
    1.5.2. Addressing modes and operands
    1.5.3. List of twelve instructions
    1.5.4. Accessing memory
    1.5.5. Functions
    1.5.6. ARM Cortex Microcontroller Software Interface Standard
    1.5.7. Conditional execution
    1.5.8. Stack usage
    ~~1.5.9. Floating-point math~~
    ~~1.5.10. Keil assembler directives~~
1.6. Pointers in C
    1.6.1. Pointers
    1.6.2. Arrays
    1.6.3. Linked lists
1.7. Linking Assembly to C
1.8. Macros in C
1.9. Function Pointers in C

2. Microcontroller Input/Output
2.1. Parallel I/O
    2.1.1. TM4C I/O programming
    ~~2.1.2. MSP432 I/O programming~~
2.2. Interrupts
    2.2.1. NVIC
    2.2.2. SysTick periodic interrupts
    2.2.3. Periodic timer interrupts
    2.2.4. Critical sections
    2.2.5. Executing periodic tasks
    2.2.6. Software interrupts
2.3. First in First Out (FIFO) Queues
2.4. Edge-triggered Interrupts
    2.4.1. Edge-triggered interrupts on the TM4C123
    ~~2.4.2. Edge-triggered Interrupts on the MSP432~~

2.5. UART Interface
    2.5.1. Transmitting in asynchronous mode
    2.5.2. Receiving in asynchronous mode
    2.5.3. Interrupt-driven UART on the TM4C123
    ~~2.5.4. Interrupt-driven UART on the MSP432~~
~~2.6. Synchronous Transmission and Receiving using the SSI~~
~~2.7. Input Capture or Input Edge Time Mode~~
    ~~2.7.1. Basic principles~~
    ~~2.7.2. Period measurement on the TM4C123~~
    ~~2.7.3. Period measurement on the MSP432~~
    ~~2.7.4. Pulse width measurement~~
    ~~2.7.5. Ultrasonic distance measurement~~
~~2.8. Pulse Width Modulation~~
    ~~2.8.1. Pulse width modulation on the TM4C123~~
    ~~2.8.2. Pulse width modulation on the MSP432~~
~~2.9. Analog Output~~
2.10. Analog Input
    2.10.1. ADC Parameters
    2.10.2. Internal ADC on TM4C
    ~~2.10.3. Internal ADC on MSP432~~
    ~~2.10.4. Central Limit Theorem~~
~~2.11. Board Support Package~~
2.12. Introduction to Debugging
    2.12.1. Overview of Debugging
    2.12.2. Functional Debugging
    2.12.3. Performance Debugging (FFT analysis)
    2.12.4. Debugging heartbeat
    2.12.5. Profiling

~~3. High Speed Input/Output~~

4. Thread Management
4.1. Introduction to RTOS
    4.1.1. Motivation
    4.1.2. Parallel, distributed and concurrent programming
    4.1.3. Introduction to threads
    4.1.4. States of a main thread
    4.1.5. Real-time systems
4.2. Thread Management
    4.2.1. Two types of threads
    4.2.2. Thread Control Block (TCB)
    4.2.3. Creation of threads
    4.2.4. Launching the OS
    4.2.5. Switching threads
    4.2.6. Profiling the OS
    4.2.7. Running the Scheduler in C

4.3. Periodic Tasks
4.4. Thread Suspend
4.5. Thread Sleeping
4.6. Thread Creation and Killing

5. Thread Communication/Synchronization

5.1. Spin-lock Semaphores
5.2. Spin-lock Semaphores with Cooperation
5.2. Blocking semaphores
    5.3.1. The need for blocking
    5.3.2. The blocked state
    5.3.3. Implementation
5.3. Thread Synchronization
    5.3.1. Resource sharing, nonreentrant code or mutual exclusion
    5.3.2. Condition variable
    5.3.3. Thread rendezvous
5.4. Producer-Consumer Problems
    5.4.1. Thread communication between two threads using a mailbox
    5.4.2. Producer/Consumer problem using a FIFO
    5.4.2. Little's Theorem
    5.4.3. FIFO implementation
    5.4.4. Three-semaphore FIFO implementation
    5.4.5. Two-semaphore FIFO implementation
    5.4.6. One-semaphore FIFO implementation
    ~~5.4.7. Kahn Process Networks~~
5.5. Debouncing a switch
    5.5.1. Approach to debouncing
    5.5.2. Debouncing a switch on TM4C123
    ~~5.5.3. Debouncing a switch on MSP432~~
5.6. Monitors
5.7. Path Expressions
5.8. Deadlocks

6. Thread Scheduling

6.1. Introduction to Scheduling
6.2. Cooperative Scheduler
6.3. Priority scheduler
    6.3.1. Implementation
    6.3.2. Starvation and aging
    6.3.3. Priority inversion and inheritance on Mars Pathfinder
    6.3.4. Running event threads as high priority main threads
6.4. Fixed Scheduling
6.5. Other Scheduling Algorithms
    6.5.1. Multi-level Feedback Queue
    6.5.2. Shortest Job First, SJF

Labs: Labs 1-3, important topics
Lab 1: Interrupts, Cortex M4 architecture, FIFO queues, UART, ADC
Lab 2: Real time OS, semaphores, critical sections, synchronization, communication
Lab 3: Debugging, priority, scheduling, blocking semaphores

Architecture (Chapters 1, 2 are a review of EE445L)
- Registers, buses, ports, stack,
- Interrupts NVIC, tail chain, late arriving,
  - SysTick, PendSV, edge triggered interrupts,
  - arm, enable, latency, jitter, hardware FIFO,
- ARM assembly code,
  - subroutine linkage (AAPCS),
  - parameter passing (registers and stack),
  - local variables (registers and stack),
  - interrupt linkage.
- Data flow graph, call graph, flowcharts.
- HAL, device driver.

Debugging
- intrusiveness,
- profile,
- dump,
- control, observability,
- coverage
- white box, black box

Data structures
- FIFO,
- statically allocated linked lists,
- dynamically allocated linked lists,

OS stuff
- latency, real- time, interrupt priority,
- reentrancy, critical sections, race condition,
- sleeping,
- scheduling
  - preemptive vs nonpreemptive=cooperative,
  - round robin,
  - priority,
  - rate monotonic,
  - earliest deadline first,
  - least slack-time first,
- semaphore implementation
  - spinlock,
  - cooperative spinlock,
  - blocking,

semaphore applications
OS Concepts
- kernel,
- bounded waiting,
- priority inversion, priority inheritance, aging, starvation,
- mutual exclusion,
- certification,
- hooks,
- slack time,
- lateness,
- rate monotonic scheduler, rate monotonic theorem,
- protection,
- CPU utilization,
- **semaphore application (study the book examples),**
- deadlocks
  - necessary conditions,
  - detection,
  - prevention,
  - avoidance,
- resource allocation graph, monitors.

See questions at the end of Chapters 4), 5) and 6).

Past exams:

Real time OS, semaphores, critical sections, synchronization, communication
Spring 2001,Quiz2, Question 2, Sleep primitive
Fall 2001,Quiz2, Question 4, Priority scheduler, deadlock
Spring 2002, Quiz1, Question 3, Dynamic thread allocation, thread Kill
Fall 2002, Quiz2, Question 2, application of semaphores
Fall 2002, Final, Question 4, use of semaphores
Fall 2002, Final, Bonus questions 1,2,6, assembly language used in OS programming
Fall 2003, Quiz1, Question 2, use of semaphores
Fall 2003, Quiz1, Question 3, changing the TCB
Fall 2003, Quiz1, Question 4, definition of time jitter
Fall 2003, Quiz1, Question 5, implementation of OS_Wait
Fall 2003, Final, Question 14, definitions of OS concepts/terms
Fall 2004, Quiz2, Question 2, Three thread rendezvous
Fall 2004, Quiz2, Question 3, Binary semaphore
Fall 2004, Final, Question 9, Path expression
Fall 2005, Quiz2, Question 4, Reader/writer problem
Fall 2005, Quiz2, Question 5, Cooperative thread scheduler
Fall 2006, Quiz2, Question 9, Fork
Fall 2006, Quiz2, Question 5, Resource allocation graph
Fall 2006, Final, Question 5, Exponential Queue or multi-level feedback queue scheduling
Spring 2008, Quiz2, Question 4, use of semaphores
Spring 2008, Final, Question 2, Effect of OS on time-jitter while sampling an ADC

Spring 2008, Final, Question 5, Critical section, design new instruction
Spring 2009, Quiz 2, Question 4, Critical section
Spring 2009, Quiz 2, Question 5, Fork and join
Spring 2009, Final, Question 5, kill threads that finish executing
Spring 2010, Quiz 1, Question 2, word bank
Spring 2010, Quiz 1, Question 4, alternate words for signal and wait
Spring 2010, Quiz 1, Question 5, what happens if an ISR calls OS_Wait
Spring 2010, Quiz 1, Question 6, implementing mutual exclusion
Spring 2010, Quiz 1, Question 7, application of semaphores
Spring 2011, Quiz 1, Question 4, definitions
Spring 2011, Quiz 1, Question 5, application of semaphores
Spring 2011, Quiz 1, Question 6, new implementation of semaphores
Spring 2011, Quiz 1, Question 7, priority scheduler (the 2011 class did horrible on this question because they parroted their lab solution without reading the question)
Spring 2010 Final, Question 5, definitions d, i, j
Spring 2011 Final, Question 8, bounded waiting
Spring 2011 Final, Question 9, real time OS, minimizing latency
Spring 2011 Final, Question 11, FIFO with semaphores
Spring 2011 Final, Question 12, implementing semaphores in a Dual core processor
Spring 2011 Final, Question 16, implementing a thread scheduler one a 16-core processor
Spring 2012 Quiz 1, Question 4, Two SPs.
Spring 2012 Quiz 1, Question 5, OS definitions.
Spring 2012 Quiz 1, Question 7, Monitor and deadlocks.
Spring 2012 Quiz 1, Question 8, OS_AddThread and OS_Kill.
Spring 2012 Quiz 1, Question 9, Use OS to debounce a switch.
Spring 2013 Quiz 1, Question 1, Priority.
Spring 2013 Quiz 1, Question 3, OS definitions.
Spring 2013 Quiz 1, Question 5, using semaphores.
Spring 2013 Quiz 1, Question 6, Assembly language thread switch.

General questions
Fall 2004, Quiz2, Question 4, Time-jitter
Fall 2004, Quiz2, Question 5, Definitions and a word bank
Fall 2005, Quiz2, Question 6, Time-jitter
Fall 2006, Final, Question 4, Critical section
Spring 2009, Quiz 2, Question 3, FIFO implementation
Spring 2011, Quiz 1, Question 1, time jitter
Spring 2011, Quiz 1, Question 2, reentrant, parameter passing, LR
Spring 2011, Quiz 1, Question 3, bit-banded I/O eliminates critical section, which registers are pushed on the stack during an interrupt context switch, what is LR during an ISR
Spring 2010 Final, Question 1, Cortex M3 interrupt context switch (answer for TM4C123)
Spring 2011 Final, Question 2, Cortex M3 interrupt context switch
Spring 2012 Quiz 1, Question 3, Harvard architecture.
Spring 2012 Quiz 1, Question 6, Reentrancy.
Spring 2013 Quiz 1, Question 2, Control and observability.
Spring 2013 Quiz 1, Question 4, Critical section