EE445M Final Exam Study Guide (Spring 2020):

Instructions:

- Open book, open notes and open web.
- No electronic devices other than laptop/PC (all cell phones off).
- You are allowed to access any resource on the internet, but no electronic communication other than with instructors.
- Unless otherwise stated, find the most efficient (time, resources) solution.

Lecture notes: Lectures 1-12
Skip Lecture 13, slides 2-9, no other OS

Labs: Labs 1-7, important topics
Lab 1: Interrupts, Cortex M architecture, FIFOs, serial port, ADC
Lab 2: Real time OS, semaphores, critical sections, synchronization, communication
Lab 3: Priority scheduling, blocking semaphores, debugging
Lab 4: File system, SPI, SD cards
Lab 5: Memory & process management, process loading & linking
Lab 6: Ethernet/Wifi, TCP/IP, and distributed systems
Lab 7: Applications and abstraction

Book: Chapters 1-9

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 LlaunchPad 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
3.1. The Need for Speed
3.2. High-Speed I/O Applications
3.3. General Approaches to High-Speed Interfaces
3.3.1. Hardware FIFO
3.3.2. Dual Port Memory
3.3.3. Bank-Switched Memory
3.4. Fundamental Approach to DMA
3.4.1. DMA Cycles
3.4.2. DMA Initiation
3.4.3. Burst versus Single Cycle DMA
3.4.4. Single Address versus Dual Address DMA
3.4.5. DMA programming on the TM4C123
3.5. Exercises

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

7. Memory and Process Management

- 7.1. Memory Management
  - 7.1.1. Use of the heap
  - 7.1.2. Simple fixed-size heap
  - 7.1.3. Memory manager: malloc and free

7.2. Process Management
7.3. Dynamic loading and linking
7.4. Exercises

8. File system management

8.1. Performance Metrics
8.1.1. Usage
8.1.2. Specifications
8.1.3. Fragmentation
8.2. File System Allocation
8.2.1. Contiguous allocation
8.2.2. Linked allocation
8.2.3. Indexed allocation
8.2.4. File allocation table (FAT)
8.3. Solid State Disk
8.3.1. Flash memory
8.3.2. Flash device driver
8.3.3. eDisk device driver
8.3.4. Secure digital card interface
8.4. Simple File System
8.4.1. Directory
8.4.2. Allocation
8.4.3. Free space management
8.5. Write-once File System
8.5.1. Usage
8.5.2. Allocation
8.5.3. Directory
8.5.4. Append
8.5.5. Free space management
8.6. Readers-Writers Problem
8.7. Exercises

9. Communication Systems

9.1. Fundamentals
9.1.1. The network
9.1.2. Physical Channel
~~9.1.3. Wireless Communication~~
~~9.1.4. Radio~~
9.2. Controller Area Network (CAN)
9.2.1. The Fundamentals of CAN
9.2.2. Texas Instruments TM4C CAN
9.3. Embedded Internet
9.3.1. Abstraction
9.3.2. Message Protocols
9.3.3. Ethernet Physical Layer
~~9.3.4. Ethernet on the TM4C1294/MSP432E401Y~~

9.4. Internet of Things
    9.4.1. Basic Concepts
    9.4.2. UDP and TCP Packets
    9.4.3. Web server
    ~~9.4.4. UDP communication over WiFi~~
    ~~9.4.5. Other CC3100 Applications~~
~~9.4. Bluetooth Fundamentals~~
    ~~9.4.1. Bluetooth Protocol Stack~~
    ~~9.4.2. Client-server Paradigm~~
~~9.5. CC2650 Solutions~~
    ~~9.5.1. CC2650 Microcontroller~~
    ~~9.5.2. Single-Chip Solution, CC2650 LaunchPad~~
~~9.6. Network Processor Interface (NPI)~~
    ~~9.6.1. Overview~~
    ~~9.6.2. Services and Characteristics~~
    ~~9.6.3. Advertising~~
    ~~9.6.4. Read and Write Indications~~
~~9.7. Application Layer Protocols for Embedded Systems~~
    ~~9.7.1. CoAP~~
    ~~9.7.2 MQTT~~
9.8. Exercises

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 2012 Final, Question 3, is a real-time scheduler possible?
Spring 2012 Final, Question 5, How does paging provide for security?
Spring 2012 Final, Question 6, Mailbox
Spring 2012 Final, Question 10, Wait_For_Events
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.
Spring 2013 Final, Question 3, How does MSP/PSP provide for security?
Spring 2013 Final, Question 6, FIFO with semaphores
Spring 2013 Final, Question 11, PendSV to implement cooperation

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 2012 Final, Question 1, Definitions
Spring 2013 Quiz 1, Question 2, Control and observability.
Spring 2013 Quiz 1, Question 4, Critical section
Spring 2013 Final, Question 1, Definitions
Spring 2013 Final, Question 7, Time-jitter
Spring 2014 Final, Question 7, Profiling

File System
Fall 2006, Quiz1, Question 2, Bit vector free space
Fall 2006, Quiz1, Question 3, File system
Spring 2008, Quiz1, Question 1, File translation table
Spring 2008, Quiz1, Question 2, Block size
Spring 2009, Quiz1, Question 1, Contiguous Allocation
Spring 2010, Quiz2, Question 4, File system logger
Spring 2010, Quiz2, Question 5, High reliability File system
Spring 2011, Quiz2, Question 5, Wear leveling disk system
Spring 2011, Quiz2, Question 6, Disk clustering
Spring 2011, Final, Question 7, FAT file system
Spring 2012, Quiz2, Question 4, File system in flash EEPROM
Spring 2012, Final, Question 8, Indexed file system
Spring 2013, Quiz2, Question 4, File system file table
Spring 2013, Final, Question 9, FAT file system
Spring 2014, Quiz2, Question 4, File system with an index table
Spring 2014, Final, Question 11, File system with FAT

CAN
Fall 2005, Final, Question 4, CAN bandwidth
Fall 2005, Final, Question 5, CAN latency (although the solution for this question is specific to the 9S12, it could be asked in general, or in specific for the STM32)
Fall 2006, Final, Question 3, CAN Id
Spring 2008, Final, Question 1, Noise
Spring 2008, Final, Question 7, Fifo queue
Spring 2009, Final, Question 1, General concepts, ACK
Spring 2011, Final, Question 6, Half duplex and differential drive
Spring 2012, Final, Question 7b), CAN error recovery
Spring 2014, Final, Question 3, purpose of the 120 ohm resistors
Spring 2014, Final, Question 6, Is CAN synchronous or asynchronous?
Spring 2014, Final, Question 12, CAN synchronization

DMA
Spring 2011 Final, Question 5, Cycle steal versus burst DMA
Spring 2014 Final, Question 8, Cycle steal versus burst DMA

~~**Fuzzy Logic**~~
~~Spring 2014 Final, Question 9, Fuzzy Logic math~~
~~Book questions 10.3 10.4, 10.5, and 10.8~~

~~**Motor & Control Questions**~~
~~Fall 2005, Final, Question 8, Design of a PID controller~~
~~Fall 2006, Final, Question 1, Tach interface~~
~~Fall 2006, Final, Question 2, Measure motor current~~
~~Fall 2006, Final, Question 6, Design of a PID controller~~
~~Spring 2008, Final, Question 3, Motor interface~~
~~Spring 2008, Final, Question 4, PWM and motor control~~
~~Spring 2009, Final, Question 3, Motor interface~~
~~Spring 2009, Final, Question 7, Measure motor current~~