IIT Hyderabad Study-abroad Program

EE 319K: Embedded Systems

Ramesh Yerraballi

Summer 2016

General Information

EE319K: Class Time	Meets Tue, Wed and Thu 10am-12noon
Lab Time	Tue, Wed(Checkout) and Thu 2-4pm
Classroom	**TBA**
Office	**TBA**
Contact	ramesh@mail.utexas.edu
Pre-requisites	EE 306, BME 303
TA	Kelsey Taylor Ball (kelseyball@utexas.edu)
Office Hrs	Instructor: Tue,Wed and Thu from 1-2pm or after class/by appointment TA: Lab Time on Tue/Thu is TA office hours.
Website	Canvas and http://users.ece.utexas.edu/~ryerraballi/sa2016/EE319K.html

Teaching Philosophy

This class will be taught using the methodology of a hybridized "flipped classroom". This means, each class session will require you to have read_material/watched_videos on the concepts that will be the focus of the class period. The class period will be used to clarify and expound on the reading material and "apply" these concepts and solve actual problems. The class lecture will be a hybrid of traditional teaching and flipped classroom approach.

Catalog Description

EE319K: Introduction to Embedded systems; machine language execution; assembly language programming; local variables; input/output synchronization; analog to digital conversion, digital to analog conversion; debugging; and interrupts.

Overview

EE319K will use a bottom-up educational approach. The overall educational objective is to allow students to discover how the computer interacts with its environment. It will provide hands-on experiences of how an embedded system could be used to solve EE problems. The focus will be understanding and analysis rather than design. The analog to digital converter (ADC) and digital to analog converter (DAC) are the chosen mechanism to bridge the CE and EE worlds. EE concepts include Ohms Law, LED voltage/current, resistance measurement, and stepper motor control. CE concepts include I/O device drivers, debugging, stacks, FIFO queues, local variables and interrupts. The hardware construction is performed on a breadboard and debugged using a multimeter (students learn to measure voltage and resistance). Software is developed in ARM Cortex-M (Thumb) assembly and C; most labs will be first

simulated and then run on the real board (Ti LaunchPad). Software debugging occurs during the simulation stage. Verification occurs in both stages.

Detailed Course Objectives

1. Understanding how the computer stores and manipulates data (characters, integers, and fixed-point numbers), the basic arithmetic and logical operations performed by the computer,
2. The understanding of embedded systems (a system with the computer hidden inside) using modular design and abstraction,
3. Assembly language programming: considering both function and style,
4. Understanding how the computer executes instructions (fetch opcode, fetch operand, read data, operate, and write data)
5. The use of a microcontroller (strategic use of RAM ROM and I/O) Microcontrollers typically have a little RAM and a lot of ROM. Globals, locals and the heap go in RAM. Constants and programs go in ROM.
6. Debugging and verification using a simulator and on the microcontroller (embedded systems typically do not have a print function) debugging using breakpoints, scanpoints, profiles, monitors, voltmeters, oscilloscopes, logic analyzers
7. How input/output actually happens (the students wire up analog and digital signals to the ARM and measure them with a voltmeter), synchronization, including switches, LEDs, LCDs, DACs, ADCs, and serial ports,
8. The implementation of an I/O driver (a set of programs that perform I/O)
9. Understanding, from an architecture standpoint, how local variables and parameters work (e.g., a space on the stack is dynamically created, the local variable is accessed using stack-pointer relative addressing, then the space is deallocated.)
10. Analog to digital conversion (ADC) e.g., the students interface a slide potentiometer to the ADC, and write software that measures the position of the slide, creating a display like “1.23 cm”
11. Interrupt synchronization, real-time ADC sampling (periodic timer interrupts), introduction to multithreaded programming
12. Simple motors (e.g., open and closed-loop stepper motor control)
13. Digital to analog conversion (DAC), used to make simple sounds
14. Design and implementation of elementary data structures, such as linked lists, stacks and queues.

After the successful conclusion of EE319K students should be able to understand the basic components of a computer, write assembly language programs that perform I/O functions and implement simple data structures, manipulate numbers in multiple formats, and understand how software uses global memory to store permanent information and the stack to store temporary information.

Text and Reference Materials

Jonathan W. Valvano, *Embedded Systems: Introduction to ARM® Cortex™-M Micro-*

controllers, Volume 1, 4th edition, 2013, ISBN: 978-1477508992. (available from Amazon). Older versions will be okay.

Zyante, *Programming in C*, UT-EE319K edition. Available online:

Supplementary material:

- Reading assignments will be given from Patt's *Introduction to Computing Systems* (textbook for EE306)
- Manuals for the board, processor and tools used in class are linked from Prof. Valvano's EE319K site.
- Data sheets for most of the devices used in this class are also available as PDF files on Dr. Valvano's site.

Equipment

Board: Every group of two students will be given a Texas Instrument LaunchPad kit.

Tools: Every student should own their own voltmeter and their own wire strippers. You will also need Multimeter, see for example:

BG Micro:
http://www.bgmicro.com/MET1014.aspx

Jameco:
http://www.jameco.com/webapp/wcs/stores/servlet/Product_10001_10001_220812_-1

Harbor Freight, has 3 locations around Austin, usually sells voltmeters for less than $10:
http://www.harborfreight.com/7-function-digital-multimeter-90899.html

All EE319K students will need their own voltmeter and wire strippers.

Kit contents (1 per student):

- 1 Breadboard
- 1 Nokia 5110 LCD
- 1 7406, six open-collector drivers used for the LED interfaces
- 6 LEDs (20 mA, 2 red, 2 yellow, 2 green)
- 6 220 ohm 5%, 0.25 watt resistors
- 4 push-button switches
- Resistors: 1KΩ (1), 1.5KΩ (3), 12KΩ(3), 10KΩ(4)
- Wires
- 1 0.1 uF ceramic bypass cap, place across power and ground of the 7406
- 1 Stereo headphone jack 3pin 3.5mm
- 1 10KΩ slide pot, 15mm

Computer: We expect every EE319K student to have a personal laptop. In addition to being required to complete Exam 2, a personal laptop on which you develop your software and write your reports.

You will need to install Keil uVision for the ARM on your laptop.

Software

Keil uVision: Instructions on how to download and install the (free) tool chain for our board can be found at http://users.ece.utexas.edu/~valvano/Volume1/uvision/.

Legal Stuff

Scholastic dishonesty: "Faculty in the ECE Department are committed to detecting and responding to all instances of scholastic dishonesty and will pursue cases of scholastic dishonesty in accordance with university policy. Scholastic dishonesty, in all its forms, is a blight on our entire academic community. All parties in our community -- faculty, staff, and students --

are responsible for creating an environment that educates outstanding engineers, and this goal entails excellence in technical skills, self-giving citizenry, an ethical integrity. Industry wants engineers who are competent and fully trustworthy, and both qualities must be developed day by day throughout an entire lifetime. Scholastic dishonesty includes, but is not limited to, cheating, plagiarism, collusion, falsifying academic records, or any act designed to give an unfair academic advantage to the student. The fact that you are in this class as an engineering student is testament to your abilities. Penalties for scholastic dishonesty are severe and can include, but are not limited to, a written reprimand, a zero on the assignment/exam, re-taking the exam in question, an F in the course, or expulsion from the University. Don't jeopardize your career by an act of scholastic dishonesty. Details about academic integrity and what constitutes scholastic dishonesty can be found at the website for the UT Dean of Students Office and the General Information Catalog, Section 11-802."

You are encouraged to study together and to discuss information and concepts with other students. You can give "consulting" help to or receive "consulting" help from such students in oral form. However, this permissible cooperation should never involve one student having possession of a copy of all or part of work done by someone else, in the form of an email, an email attachment file, a portable storage device, or a hard copy. Copying of any part of a program is cheating without explicit reference to its source. We do enter lab assignments turned in by EE319K students through a plagiarism checker, comparing them to assignments of this and previous semesters. If we find two programs that are copied, there will be a substantial penalty to both students, e.g., failure in the course. Students who cheat on tests or in lab will fail. Prosecution of cases is very traumatic to both the student and instructor. It is appropriate to use software out of the book, class website as long as all copy-pasted software is explicitly referenced. Copy-pasting software from current or past EE319K students is scholastic dishonesty. Policies concerning the use of other people's software in this class:

- I strongly encourage you to study existing software.
- All applications and libraries must be legally obtained. E.g.,
  - You may use libraries that came when you bought a compiler.
  - You may use software obtained from the web.
  - You may copy and paste from the existing source code.
- You may use any existing source code that is clearly referenced and categorized:
  - original: completely written by you,
  - derived: fundamental approach is copied but it is your implementation,
  - modified: source code significantly edited to serve your purpose,
  - copied: source code includes minor modifications.

The University Honor Code is "The core values of the University of Texas at Austin are learning, discovery, freedom, leadership, individual opportunity, and responsibility. Each member of the University is expected to uphold these values through integrity, honesty, trust, fairness, and respect toward peers and community." http://registrar.utexas.edu/catalogs/gi09-10/ch01/

Grading Criteria

Task	Date	Percentage
Homework Assignments (8)	Homeworks are due Tuesdays	10%
1st Test – Written	Thu: 7/7	15%
2nd Test – Programming	Thu: 7/21	20%
Final Exam – Oral	Thu: 8/11	25%
Programming Lab Assignments (9)	Due at respective lab times	30%

When programming labs are performed as a team (of two) only one solution must be turned in. All exams are closed book. Cutoff scores for the corresponding letter grades will not be

determined until after the final exam.

Partial Lecture/Reading Schedule

Date	Book *E-Book*	Lecture Schedule
Week 1	Ch1,2,3	Introduction Course administration; Embedded systems, development cycle; Flow charts, data flow and call graph Architecture – Arm Cortex-M3 architecture and execution; Memory allocation; Microcontroller Ports C Programming (C Primer) - Introduction to C; Structure of a C program; Logic and arithmetic expressions
		C Programming – Scope of variables in C (Local/Global);
		C Programming – Loops
Week 2	Ch 4	Modular programming - If-then, loops; Subroutines parameters and the Stack C Programming – Parameter Passing: Call-by-value and Call-by-reference. Functional Debugging Debugging - Debugging Keil uVision
Week 3	Ch 5	Pointers - Indexed addressing; Arrays; Strings Timers – Timers; Functional debugging. C Programming – Arrays, Indexing, Pointers.
Week 4	Ch 6	FSMs - Finite state machines (FSMs) C Programming - Structures and User-defined data types
		(Thursday) Exam1: Syllabus TBA
Week 5	Ch 9	Interrupts - interrupts and interrupt processing; SysTick interrupts DAC conversion - Digital to analog conversion (DAC); Sound generation C/Assembly Programming – AAPCS Convention
Week 6	Ch 7	**(Thursday) Exam 2: Syllabus TBA** LCD interface - LCD programming; Number conversions
Week 7	Ch 10	ADC conversion - Analog to digital conversion (ADC) Lab 8 design methods
		C programming – Data Structures
Week 8	Ch8	Serial I/O - universal asynchronous receiver transmitter (UART); UART programming and interrupts; Lab 9 Introduction
	Ch11	Thread communication - Producer-consumer problems; FIFO queue; C Programming – FIFO C Programming – Recursion
Week 9		**(Thursday) Final Exam: Syllabus – Comprehensive**

Lab Schedule

All Labs except lab 2 are in C. Labs are due on Wednesday of the week during afternoon lab time.

Date/Weight	Lab
Week 1/3%	**Lab1**: Alarm: I/O, parallel port, direction register and logical function.
Week 2/3%	**Lab2**: LED and Switch interfacing (simulated and board) (Assembly)
Week 3/3%	**Lab3**: Debugging techniques, one switch, one LED
Week 4/4%	**Lab4**: Traffic Light Controller
Week 5/4%	**Lab5**: Digital Piano using 4-bit DAC, Systick Interrupts
Week 6/3%	**Lab6**: LCD device driver, decimal fixed-point output
Week 7/3%	**Lab7**: Real-time Position Monitor, ADC
Week 8/3%	**Lab8**: Distributed DAS, Serial Port Interrupts, FIFO queue
Week 9/4%	**Lab9:** Design competition

Homework Schedule

Homeworks are due on Tuesdays at start of lab.

Due Date	Task
Week 2	Homework 1: C Programming Basics – Variables, Expressions, Conditionals
Week 3	Homework 2: Pointers and functions, call by value, call by reference
Week 4	Homework 3: Arrays, Indexing, Pointers
Week 5	Homework 4: Practice Exam 2
Week 6	Homework 5: Game Proposal
Week 7	Homework 6: Graphics on LCD
Week 8	Homework 7: Sound for Game
Week 9	Homework 8: Game Engine