ECE 381: Microcontrollers

Timothy York

Web Site:
http://www.siue.edu/~tyork/classes/ece381

SOUTHERN ILLINOIS UNIVERSITY
EDWARDSVILLE

1st Class: What are we going to do?

- Necessary class business stuff (syllabus, schedule, office hours)
- Introduction to microcontrollers
- Intro. to the Cypress PSoC (Programmable System-on-a-Chip) which will be what we use for this class
  - Get set up in lab
  - Choose partners
  - Pick a station
  - Get a PSoC Dev Kit

What I Expect of You

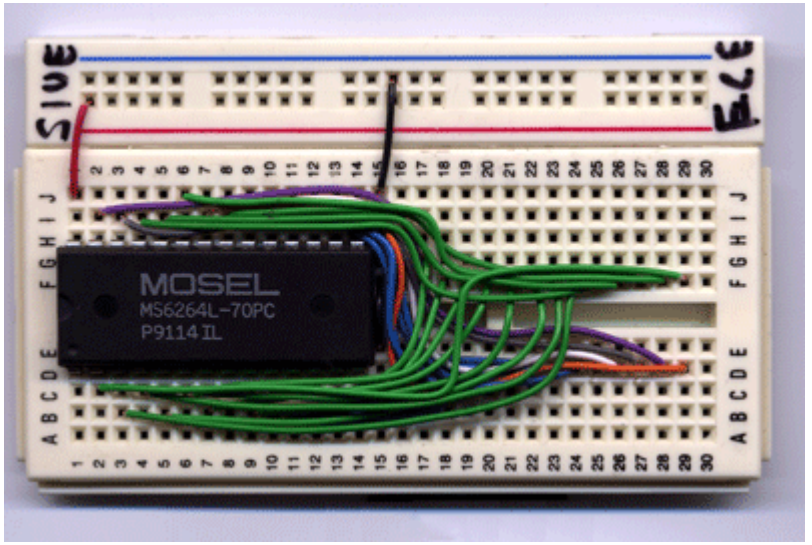
- Class syllabus: http://www.siue.edu/~tyork/classes/ece381/syllabus.pdf
- Highlights:
  - 50% of grade is labs, 30% exams, 15% final project, 5% class participation/other homework
  - Class Attendance is (Mostly) Mandatory!
    - Notify me if you can't make it (job, illness, life event, etc.)
    - Open lab sessions are not mandatory if you have already demoed
  - There will be readings, from both book and datasheets (Over course of semester, I hope you learn how to properly read a datasheet!)
  - Work in pairs on labs, but turn in separate reports (report format: http://www.siue.edu/~tyork/classes/ece381/labReportFormat.html)
- No set lab time!
  - Unlike other classes, labs will take longer than 3hr session, plan accordingly
  - Once I know more about parts kits, I'll know whether or not you get to keep the PSoC5 kit. Unitl then, leave it in lab.

ECE 381 Microcontrollers – Spring 2017 Schedule

Week	Lecture/Activity	Lab Assignment	In Lab	Submit
1	Overview / C Language Primer	Lab 1 – Introduction		
	General Purpose I/O	Lab 2 – Polling and the LCD	Lab 1 Demo	
2	Interrupts	Lab 3 – Interrupts and Rotary Encoders	Lab 2 Demo	
	PSoC5 Schematic Editor and Modules			Lab 2 Report
3	Lab 4 Overview	Lab 4 – Square Wave Generator	Lab 3 Demo	
	The RS232 Interface and Protocol			Lab 3 Report
4	Lab 5 Overview	Lab 5 – RS232 Interfacing	Lab 4 Demo	
	Class convenes in lab			Lab 4 Report
5	*Exam 1*			
	Lab 6 Overview	Lab 6 – I2C External RAM	Lab 5 Demo	
6	*Class convenes in lab*			Lab 5 Report
	USB Basics			
7	Lab 7 Overview	Lab 7 – Thermostat & Stepper Motors	Lab 6 Demo	
	Class convenes in lab			Lab 6 Report
8	PS/2 Keyboard Protocol / Bitbanging			
	Class convenes in lab		Lab 7 Demo	
9	Lab 8 Overview	Lab 8 – PS/2 Keyboard Interface		
	The PSoC5 Analog System			Lab 7 Report
10	Lab 9 Overview	Lab 9 – Wireless Temperature Sensor	Lab 8 Demo	
	SPI Protocol and Interfacing			
11	Lab 10 Overview	Lab 10 – SPI SRAM Interface	Lab 9 Demo	Lab 8 Report
	Exam 2			
12	The Final Project			
	ADCs & DACs	Lab 11 – Data Acquisition System	Lab 10 Demo	Proposal/9 Report
13	*Class convenes in lab*			
	Final Project Assignment			
14	*Class convenes in lab*	Begin Work on Final Project	Lab 11 Demo	
	Class convenes in lab			
15	*Class convenes in lab*			Lab 11 Report
	Class convenes in lab			
Finals			Project Demo	Project Report

Lab Things That Might Be Useful

- Wire Strippers/Clippers
- Micro USB Cable



What is a Microcontroller?

- Book: “A processor, memory (ROM & RAM), and general purpose I/O integrated on a single chip.”
- Examples: PSoC (of course), TI MSP430, Parallax Propeller, Atmel AVR (Arduino), Freescale HCS12, Coldfire
- Keyword: “Controller”
  - Interface with external devices/circuits/components through GPIO, ADCs, various communicatoin buses
  - Do some simple processing
  - Control other devices/circuits/components through GPIO

Virtually Every µC Does These Things Too

- Interrupts
  - “Every X milliseconds, run Y code...”
  - “When Bit 0 of Port 1 goes high...”
- Precision Timers & Counters
  - “Master CLK at 24 MHz, divide by A, then divide by B”
  - Watchdog timers “If watchdog not reset, initiate recovery”
- Analog Inputs (ADCs)
- Pulse Width Modulation

Other µC Things

- Registers, lots of registers
  - Data registers, data direction registers, condition code registers, timer control registers, etc.
  - Setup/input/output involve reading and writing to registers
- Memory Map: Which data goes where
  - i.e. Registers (0x0000-0x00ff), SRAM (0x0100-0x0FFF), ROM/Flash (0x1000-0x1CFF), Vectors (0x1D00-0x1FFF)
- Internal registers
  - Accumulators
  - Program Counter, Inst. Reg., Stack Pointer, etc.

The PsoC5 (CY8C5888LTI-LP097)

- 32-bit ARM Cortex-M3 core plus DMA controller and digital filter processor, at up to 80 MHz

  256KB Flash, 32KB ECC Flash, 2KB EEPROM, 64KB RAM

- http://www.cypress.com/file/45906/download - Datasheet

Assembly Language

- Ultimately, all programs are composed in assembly

```
        ORG 0x0100      ;Start program at first address in ROM
        LDAA #0xFF      ;Load accumulator A with 0xFF (all 1's)
        STAA 0x01       ;Store in Data Direction register
                        ;(address = 0x01) for Port0 (1 makes bit an
                        ;output)
        CLRA            ;Clear Accumulator A

MAIN
        STAA   0x00     ;Store A to address 0x00 (Port0)
        INCA            ;Increment Accumulator A
        BRA MAIN        ;Always branch back to MAIN
```

Higher Level Language

- C is a high level language typically used for programming (We will be using C for the PSoC)

```
#include <uconAPI.h>    //Header file for microcontroller
                        //Defines PORT0 and PORT0_DDR

void main()
{
    PORT0_DDR = 255; //Set PORT0 to be all outputs
    PORT0 = 0;       //Initialize PORT0 to be 0


    while (1) {      //Main loop
        PORT0 += 1;  //Increment PORT0
    }
}
```

C Primer for ECE 381

- C is a typed language:
  - char, int, short, unsigned, float, etc. are types
  - Each defines a space in memory (data) of a defined size and an address (pointer)
  - PSoC Defines BYTE (unsigned char), WORD (unsigned int)
  - PSoC5 Sizes (NOTE: THESE ARE DIFFERENT FROM MOST STD C):
    - char (1 byte)
    - int (4 bytes)
    - unsigned (4 bytes)
    - float (4 bytes) – NOTE: The PSoC5 Does NOT have a hardware FP unit. All fp done in software → SLOW!
- C uses pointers
  - int * var vs. int var
  - Pointers are memory locations only, can POINT to variables or storage
  - int var1
    - &var1 is a pointer to the address of var1
    - *var1 is the value at the address of var1

C Primer for ECE 381

- Beware of where variables declared!
- PSoC expects all variables declared before executable code!

```
#include <uconAPI.h>    //Header file for microcontroller
                        //Defines PORT0 and PORT0_DDR

int globalVar;

void main()
{
    int localVar;

    PORT0_DDR = 255; //Set PORT0 to be all outputs
    PORT0 = 0;       //Initialize PORT0 to be 0

    while (1) {      //Main loop
        PORT0 += 1;  //Increment PORT0
    }
}
```

Global Space (Before Functions): Accessible to All

Local Space (In Functions): Only accessible to function

SOUTHERN ILLINOIS UNIVERSITY EDWARDSVILLE

C Primer for ECE 381

- Pass by value vs. Pass by reference

```
#include <uconAPI.h>    //Header file for microcontroller

void func1(int * var) {
    *var++;
}

int func2(int var) {
    return var + 1;
}

void main()
{
    int localVar = 0;
    int retVar;

    func1(&localVar);
    retVar = func2(localVar);
    ...
```

By reference (directly modifies variable value)

By value (makes local copy)

localVar = 1 after this

localVar still = 1 after this,
retVar = 2

SOUTHERN ILLINOIS UNIVERSITY EDWARDSVILLE

C Primer for ECE 381

- Arrays are contiguous blocks of memory with same type
  - char str1[16] ← 16 consecutive chars in memory
  - int var[8] ← 8 consecutive ints in memory
- Access individual elements through [] notation
  - str1[0] = 'H';
  - str1[10] = 'i';
  - var[2] = var[1] + var[0];
- Array name is really a pointer to first location
  - str1[10] equivalent to address of str1 + 10 bytes of memory
  - &str1[10] gives memory address of 10th element (str1[10] gives data)
  - Pass arrays to function by reference only
    - func(str1) only works if func declared as: void func(char * array)
    - Doesn't make a local copy since passed by reference

C Primer for ECE 381

- Strings are arrays of characters
- Last element of string is special '\0' or NULL character, denotes end of string
  - ie. char str[16] = {'H','e','l','l','o','\0'}
  - printf(str) → Hello

C Primer for ECE 381

- C has if, else if, else conditional statements
  - ```
    if (var == 1) {
        do something for var == 1
    }
    else if (var == 2) {
        do something for var == 2
    }
    else {
        do something for var not 1 or 2
    }
    ```
- C also has switch-case statement for conditionals
  - ```
    switch(var) {
        case 1:
        do something for var==1
        break;
        case 2:
        do something for var==2
        break;
        default:
        do something for var not 1 or 2
        break;
    }
    ```

Cases must be terminated by breaks

C Primer for ECE 381

- Loops can be done with *while, do…while,* and *for*
  - while (var < 10) {
    var++;
    }
  - do {
    var++;
    } while(var < 10);
  - for (var = 0; var < 10; var++){
    var1 = var;
    }

C Primer for ECE 381

- Every program has to have a main() function
  - This is essentially the code that gets executed when the uC gets turned on (sometimes after some backend setup)
  - For uCs, there is always an infinite loop somewhere in main
- Other functions can be declared in a .h file
  - Just the prototype
  - Code goes in a .c file of same name with same name as prototype
  - This way, compiler only has to recompile if changes to .c file or .h file
  - These .c & .h files can be exported easily to other projects

Assignments

- Reading:
  - Cady, Ch. 2
  - CY8C5888LTI-LP097 Data Sheet (labs/files/PSoC Technical Documents/psoc5_CY8C5888LTI-LP097.pdf)
  - PSoC5 Architecture Technical Reference Manual (labs/files/PSoC Technical Documents/PSoC5LP_Architecture_TRM.pdf)
  - PSoC5 Prototyping Kit Guide (/labs/files/PSoC Technical Documents/CY8CKIT_059 PSoC 5LP Protoyping Kit Guide.pdf)
- Lab 1: Introduction
  - 1a: Step-by-step instructions to blink the onboard LED
  - 1b: Write own simple program to make Port1 a counter