Computer Organization

Fall 2016
Tues – Thurs D210
10:30 – 11:15, 1:00 = 2:15

INSTRUCTOR PARTICULARS

Instructor: Robert Trenary
Office: B250 CEAS
Office Hours: Tues – Thurs 3:30 – 4:30
Phone: 276-3127
E-MAIL trenary@wmich.edu

Teaching Assistant: Colin MacCreery
Office: B125,
Office Hours: M – W 1 – 2:15
E-Mail: colin.c.maccreery@wmich.edu

TEXT: There is no text for the course. Many course materials will be available at http://www.cs.wmich.edu/~trenary/cs2230.php. The following text can be used if you wish: MSP430 Microcontroller Basics: John H. Davies,Elsevier, Newnes. I do not follow it nor do I encourage you to own this. You may look at a copy by asking me, because I like to keep it around for occasional reference.
You will be required to purchase an MSP430 kit, for approximately $50. These will be assembled, supported, and sold through the Computer Club, who will come to class one time. After that, you will need to get them at the Club office (2225 Kohrman)

GRADED EVENTS:

- Programming assignments (30%) Generally there are no late assignments. You will be wise to plan to submit a completed program or documented evidence of progress. I may give some quizzes, which will count in this portion of the grade.
- Midterm Exams (30%) : The first of these is tentatively scheduled as Thursday of seventh week.
- Final Examination: (35%) As per finals week schedule. 11:30 is an 11:00 class.

COURSE GOALS:

Overview of Computer Organization down to the microprogramming level.
Understanding of basic system concepts and software.
MSP430 architecture and assembly language.
UNIX / C proficiency

Academic Integrity

The following text is recommended by the Faculty Senate for all course syllabi, and it includes links to relevant information. Please make sure you are familiar with the university guidelines for academic integrity.

Students are responsible for making themselves aware of and understanding the University policies and procedures that pertain to Academic Honesty. These policies include cheating, fabrication, falsification and forgery, multiple submission, plagiarism, complicity and computer misuse. The academic policies addressing Student Rights and Responsibilities can be found in the Undergraduate Catalog at http://catalog.wmich.edu/content.php?catoid=24&navoid=974 and the Graduate Catalog at http://catalog.wmich.edu/content.php?catoid=25&navoid=1030 If there is reason to believe you have been involved in academic dishonesty, you will be referred to the Office of Student Conduct. You will be given the opportunity to review the charge(s) and if you believe you are not responsible, you will have the opportunity for a hearing. You should *consult with your instructor if you are uncertain about an issue of academic honesty prior to the submission of an assignment or test. In addition, students are encouraged to access the Code of Honor, as well as resources and general academic policies on such issues as diversity, religious observance, and student disabilities:*

- *Office of Student Conduct www.wmich.edu/conduct*
- *Division of Student Affairs www.wmich.edu/students/diversity*
- *Registrar's Office www.wmich.edu/registrar and www.wmich.edu/registrar/policies/interfaith*
- *Disability Services for Students www.wmich.edu/disabilityservices.*

Students who take this class must be prepared to submit electronic copies of some or all assignments. The University expects that all students will be evaluated and graded on their own work. If you use language, data or ideas from other sources, published or unpublished, you must take care to acknowledge and properly cite those sources. Failure to do so constitutes plagiarism.

Throughout the term you will be required to solve problems. Solutions are to consist of YOUR OWN WORK! Learning can certainly take place through discussions regarding assignments, and you may ask for hints or ideas; however, when discussion is occurring, THE WRITING OF ASSIGNMENT SOLUTIONS SHOULD NOT TAKE PLACE DURING THAT DISCUSSION. You are to write your own solutions. Here are additional rules to follow, when writing your solutions that are to be submitted for a grade:

- Do not look at or copy material that someone else has written for the same or similar solution.
- Do not give your solution in part or whole, either in hardcopy or electronic form, to anyone else.
- Do not put yourself in a position of having access to another's files or work.
- Do not give another person access to your files or work.

Important University Policies

Each of these federal statutes, policies and statements affirms that we aspire to be a community of academics and professionals who value the diverse perspectives and experiences each individual brings to WMU. These differences contribute to our rich and vibrant environment, where we strive to broaden our understanding of complex issues and gain new insights of the world. Although our perspectives, ideas and opinions may vary, each person is valued and treated respectfully.

Please refer to the University's policy statement regarding Sexual and Gender-Based Harassment and Violence, Intimate Partner Violence, and Stalking Policy available from the following website:
www.wmich.edu/sexualmisconduct

The Faculty Senate Religious Observances Policy

The complete policy statement is available from the following website:
http://www.wmich.edu/facultysenate/downloads/MOA0702_religious_observances_final.pdf

WARNING – The University is a dangerous place where your ideas will be challenged and offensive statements will be heard. It is an equal opportunity forum for all. Mutual respect is the guiding principle.
Please see
wmich.edu/sites/default/files/attachments/u370/2016/Civility%20Stmt.7-27-16_0.pdf

The Civility statement affirms the value of each individual as a member of the university community. It further reminds us that ad hominem attacks directed towards the individual, rather than the position or idea, do not contribute to an environment allowing individuals to flourish.

Disability Services for Students (DSS)

Both in compliance with and in the spirit of the Americans with Disabilities Act (ADA), we would like to work with you if you have a disability that is relevant to your work in this class. If you have a documented disability and wish to discuss academic accommodations, please contact your instructor in a timely fashion with your DSS verification card. You may also contact the Office of Disability Services for Students if you need to register with the DSS office, 269-387-2116 (or at wmich.edu/disabiltyservices).

Thee and Thy Name

Western Michigan University recognizes that some students use first names other than their legal names to identify themselves. As an inclusive and diverse community, WMU allows students to use a preferred first name different than their legal name for certain purposes and records in the course of university business, communication, and education.

The legal name will continue to be used where required by law or university requirements. All student information will continue to be linked to both legal name and preferred name for the purposes of university records.

https://wmich.edu/registrar/policies/preferred-name

Rules for Active Shooter Situations *in Priority Order*

1) **Get Away ! Know your exits.**
2) **Barricade.**
3) **Attack.**
4) **Hide as a last resort.**

I am sorry to have to mention the above but … the ongoing march of the species does not always seem like progress.

Course Etiquette (Non-academic Requirements)

Please respect the learning opportunity for you classmates. Do not disrupt learning by arriving late, leaving early, or by being inattentive or disruptive in class. **Don't make or receive cell phone calls, text messages, and email**; turn off reminder alarms and your cell phone ringer. If you need to work for other classes or personal business, **don't do it in this class**. Take the time off if you need to do something more important than attending class.

If you think you may have difficulty meeting these or other academic requirements, please contact your instructor using e-mail, as soon as possible before or immediately after the problem occurs. Contact your instructor if you will miss or be late for a class meeting, if you have material submission problems, miss or anticipate missing course deadlines, or have personal problems that affect your work in this class.

FIRST SET OF GOALS:

You should make yourself responsible for the following skills:

1) Be able to convert between the following base systems: 2, 8, 16, 10. Generally, $?.?_? = ?.?_{10}$
2) Be able to add and subtract values represented in those bases.
2) Given a fixed number of bits and a representation scheme as in 2) be able to state the range of values which can be represented, in both a signed and unsigned representation.
3) Be able to write a simple C program (as opposed to C++). Know printf and write in C, and how to implement call by reference.
5) Simple navigation of Unix environment: edit files, log in remotely if necessary, script files, redirect input and output.

CS223 Helpful Hints

These comments are meant to help you work successfully in my course. Each instructor has style, practice, and idiosyncrasies. Here are some suggestions to deal with my version of CS223
R. Trenary

I) ***Ways in which you will learn in this course***.

A) Lecture Notes
The regular class meetings will be presented using chalkboard, verbal lecture. My handwriting is not wonderful, and the material presented in lecture will, as much as possible, be driven by interaction with students. This means that notes may not be immediately organized when you first create them in class. My advice is to ***Copy Your Notes Within 24 Hours***. They will then serve as a good resource, make you rethink the lecture, and prompt good questions.

B) Questions
I am quite annoying about this issue because I believe that the best students aggressively make certain that they are following the lecture. You can ask any question about "Anything At All In Time Or Space" and will find AAAITOS on the agenda at the beginning of every class. I have taught this class often enough to assure you that a student's question almost always touches on ideas that other students are wondering about. So ask early and often.

C) Independent/Small Programs ("Lab")
Computer Science and CS223 focus on using computers to solve problems. You are encouraged to get in the habit of testing ideas with small programs. This can be done to learn about statements in a language or to try to test with experiment questions which may arise.

II) ***Terms. Some phrases***

Nota Bene (latin) abbreviated **NB**. "Note Well". "This is important"
Molto Bene (Italian), **Muy Bien** (Spanish) loosely, "very good"
i.e. "that is"
e.g. "for example"
AAAITOS - "Anything At All In Time Or Space"

III) ***Grading***

The allocation of a mark ('A', 'BA' etc.) for the work in the course is done once – when the grade sheets are filled out at the end of the course. Until that time there is only data, generated from exams, programs etc. That data is processed into a single number by weighting (exam points are weighted most heavily). Those numbers are used to create a histogram. That histogram is then organized into groups ('clusters') which are then labeled with the traditional labels 'A', 'BA', etc.).
My goal will be to get a wide distribution of performance so that the clusters are well defined. I will exercise subjective judgment to label the data points that are not clearly in a group and to determine the label for a given group. Please keep any exam or program that has been given a grade in case there is any question about grades.

C Linking and Loading

C language topics: Source, object, executable. C specific things: printf, pass by ref, logicals, (shifts and bitwise logicals), write ? . gcc –c , gcc – o, Separate compilation of source code.

Compiling, Linking, Loading
The creation of computer programs involves the translation of the programmer's idea (the algorithm) and expressed in a programming language as a source file into a stored program executed on some computer. The transformations are called translation (compilation), linking/loading, and execution.
In our environment the source file for a C program is typically named as <something>.c. So a file might be created using a text editor to create a file called MySourceFile.c. This file is first translated using the gcc compiler; gcc is actually a script that invokes the whole process to ate the running program. If we want to take only the first step we can use the command

```
%	gcc –c MySourceFile.c
```

and we will create a file called an object file whose name is MySourceFile.o .
The object file is a file which can be linked together with other object files to create a program. It is important to notice that other object files may be drawn from source files written in differing languages. We will do this by having C main programs combined with Sparc assembly language functions, and vice versa.
Object files can be combined using gcc as in

```
%	gcc  MyCSourcefile.o MySparcSourcefile.o
```

where there are source files named MyCSourcefile.c and MySparcSourcefile.s (where Sparc assembly language files are named .s by convention). The result of the command above will be an executable which is, by default, called a.out. Then this executable can be run by merely invoking its name:

```
%	a.out
```

Note that errors can occur at each step (translation, loading, execution). Error messages can help reveal what stage of the process has caused the error. For example, a loading error typically has an error message from 'ld' the program that actually does the loading.

C Language
The C programming language is a subset of C++ with a few exceptions. We will use C in this course because it simplifies the programming environment. There are features of C/C++ which you may not have seen. The sections below will deal with some that we will need.

General Structure of C program
The text file for a C source program is similar to a C++ program. Typically there will be a statement

```
	#include <stdio.h>
```

and not the include used in C++ programs.
There should be prototypes for all functions called below main, and you are warned that main should be an int function.
In all functions the variables must be declared before any statements that use them. Thus, the C++ practice of declaring for loop indices in the for loop i.e. for (int I = 0; …   ) will not be allowed in C. The loop variable must be declared before any executable statement.

Pass By Reference Via Pointers
The parameter passing in C has only one mode: pass by value. Recall that C++ has both pass by value and pass by reference. This distinction governs whether the argument which is passed can be modified in the function (pass by reference allows such changes). Since C has only a pass by value mode and thus there needs to be a way to effect a pass by reference. This is done by using pointer variables. The pervasive use of pointers in C is a result of this necessity.
An example of the use of pointers to cause pass by reference is shown below:

```
#include <stdio.h>
void swap(int *, int *); /* a function to swap two variables */
int main()
{int x,y;
  x=6;y=7;
  printf("before the swap x=%d y=%d\n",x,y);
	swap(&x,&y);
  printf("after the swap=%d y=%d\n",x,y);
)
void swap(int *x, int *y)
```

```
{int temp;
 temp = *x;
  *x = *y;
  *y = temp;
}
```

The printf statements are examples of output in C since we do not have cout from C++. We will use printf for output. See below for its particulars. The example above is the only reasonable way we can cause a swap function to be written in C, since pass by value is the default parameter mode.

printf
The printf function is used to create output. It is a function which has parameters. The first of these is called a *format string* and is a string variable. Embedded within the format string are *format descriptors* (e.g. %d) and *escape characters* (e.g. \n). The format string specifies the output, literally, except for the format descriptors which describe the way in which values are to be output. The values which are output are the other parameters to the printf function. In the example above they are the values of x and y. The format descriptor %d says to output the value as a signed decimal. Other format descriptors include %u (unsigned decimal), %x (hexadecimal), %s (string), %f (float). The escape character \n is equivalent to newline in C++.

Thus the output above is

```
before the swap x=6 y=7
after the swap x=7 y=6
```

To understand the way in which format descriptors work consider the output from the printf statement

```
        printf("%x %u %d ", -1,-1,-1);
```

Note that the expressions which are interpreted via the format descriptors do not need to be variables.
The output above is

```
  0xffffffff   32     -1
```

The reasons for this output require us to consider a number of specifics about the Sparc machines on which you run your programs.

[To Be Continued]

Conversions

From \ To	2	8	10	16
2 (Binary)		Group 3 Bits, Map to Octal Digit (Right to Left)	Formula 1	Group 4 Bits, Map to Hex Digit (Right to Left)
8 (Octal)	Map each octal digit to 3 bits		Formula 1	8 □2□
10 (Decimal)	Division Alg	Division Alg		Division Alg
16 (Hex)	Map each hex digit to 4 bits	16 □2□8	Formula 1	

I) **Place Value Systems** represent a number V as a sequence of digits $d_n\, d_{n-1} \ldots d_2\, d_1\, d_0 .\, d_{-1}\, d_{-2} \ldots$ where a base **b,** a positive integer, is understood, and is used to compute the value associated with the digit. The values are powers of b which correspond to the subscripts above. $b^0 = 1$ and $b^{-n} = 1/b^n$. The value V of a place value numeral can be computed as

This equation is called Formula 1.
Note that the digits can be 0 and the numeral can have arbitrary length. But we tend to suppress leading and trailing 0's in place value numerals. So we write 4,196.25 and not 0000004196.250000000. (The comma has no mathematical meaning). Notice further that the digits are limited by the base to the range 0, …, b-1 (be sure you see why this is true). Note also that when our base is greater than 10, we need to invent new names for digits. So hex numbers require symbols 0,1,…9,A,B,C,D,E,F to represent the 16 digits.

II) Given N binary digits, there are 2^N possible numerals whose values range from 0 to 2^N-1 . There is a natural 1-1 correspondence between the number of digits in a base = 2^N, e.g. $16=2^4$, and N digit binary patterns. Thus every hex digit naturally corresponds to 4 digit binary numeral. This correspondence allows more compact description of bit patterns using hex notation.

III) When a value V is divided by a base b, the remainder represents the digit d_0 . Successive applications of this fact will yield the digits d_0, d_1, d_2 …. until the quotient becomes 0. After that point the digits corresponding to leading 0.'s in the numeral V represented in base b as $d_n\, d_{n-1} \ldots d_2\, d_1\, d_0$. This process is termed here the 'division algorithm'.

IV) When a value V, which is less than 1, and represented as $d_{-1} * b^{-1} + d_{-2} * b^{-2} \ldots + d_n * b^n \ldots$ is multiplied by the base b, the resulting value will = $d_{-1} * b^0 + d_{-2} * b^{-1} \ldots + d_n * b^{n+1}$. The net effect of successive applications of this rule is that a numeral less than 1 can be converted to an arbitrary base. More simply, this lets you convert a decimal to an arbitrary base.