CS 2230 Study Guide

Exam is March 1st. You will have 75 minutes. No notes, calculators, friends. Please be prepared to arrive punctually with a writing utensil. I will supply all paper.

Caveat Emptor – If we have talked about it, and I forgot to put it here, it is still fair game. So help out during class.

Creating and Executing Programs

source, object, executable, translate, link/load, executable (what errors can occur at each stage ? examples?). High level language (platform independence), target machine, assembly language, Source files from different languages can be used to create a single executable, a.out, which must be invoked ./a.out (this is the default name).

> gdb – object and executable require the –g switch. Execute using executable as input to gdb ($gdb ./a.out). Control via b(reak), r(un), c(ontinue) and single stepping (nexti, stepi). Look at program state via p(rint) for registers ($r1) or x(amine) for memory. Format of output controlled via /<format> as in x/10c MEM for ten characters starting at MEM. q(uit) to quit. Breaks can be set by source code line number or assembly language label.

C – the language. There is a definite system, machine orientation.

cpp - #include, #define

C source code structure – directives, prototypes (difference between prototype and actual executable code ? Where are each for printf() ?), main, arcgc and argv command line arguments, and then functions.

gcc – various switches –mmcu, -o, -c, -S

printf() and format descriptors %u, %d, %x, %s, %c, %p and \n, \t. scanf() is a counterpart for printf(). Each format descriptor should correspond to a value provided via some expression (as opposed to single variable).

Scalar data types – int, char, float, with unsigned, signed , long etc modifiers. Any can be made a pointer with '*' before variable name e.g. unsigned long * V;

Compound data types – arrays. ***An array name is equivalent to a pointer (NOT QUITE, see: http://stackoverflow.com/a/10186799).*** A "string" is not a data type but is sequence of null terminated bytes treated as ascii characters by many string oriented functions such as strlen(), strncmp(), etc…

Standard and other Libraries -- is always loaded and contains a multitude of functions (printf(), atoi(), …) . Other libraries e.g. math must be explicitly mentioned. Thus math.h can be include BUT gcc –lm must also be invoked. Think about that.

On the fly variable declaration is not as free as in other languages e.g. for (int i=0,) ?

POINTERS and PASS BY REFERENCE – C passes only by value, thus requiring us to pass addresses to effect pass by reference, requiring an address ('pointer') data type. & <==> "address of". 'Dereferencing' a pointer gets us to the thing pointed at which can be either a left or right hand value. (X = X + 1 uses symbol X in two different ways. Right ?). A pointer variable initially points at nothing (is NULL ?). An array name is a pointer. A pointer can be treated as an array.

Bitwise operations (&, |, <<, >>,), equivalent operations in assembly and C such as:

C	Assembly
^=	xor
&= ~	bic (bit clear), and + not
|=	bis (bit set), or
=	mov

Basic Computer (System) Organization Concepts

Von Neumann (stored program), CPU, memory, registers, addresses, Fetch-Execute-Cycle, ALU and Control, program counter. The BOB (box of bits) can 'mean' many things: signed and unsigned integer (2s complement), ascii, and INSTRUCTION which can vary greatly by architecture. RISC (reduced instruction set computing) and CISC (complex instruction set computing) describe a spectrum of complexity which through time has risen and fallen. An N bit BOB can have 2 to the Nth bit patterns, 00 … 0 to 11 … 1 ranging in value when treated as base 2 numbers from 0 to (2 to the Nth) – 1.

2s Complement – definition, sign bit, sign extension, two additions in one operation (!?), N,Z,V,C ,

MSP430 Microprocessor

16 bit processor, 16 bit registers , 16 bit address space (byte and word addressable). The memory model includes areas for executable, ram, interrupt vectors, and I/O. There are various clocks which can be controlled and a watchdog timer which is often disabled. "Ports" are at particular addresses and we can configure pins for input / output depending on how we connect devices to the board. There is a built-in protocol that

allows us to flash and debug programs. Our executable can include a library (libemb) which allows communication with our host machine..

Assembly Language

Statements are either instructions (actual or emulated) or assembler directives. Statements can be labeled with <IDENTIFIER>: on margin. Instructions can be two operand, one operand, or control. Operands can be of various form depending on 'addressing mode'. There are 7 addressing modes including some which create constants on the fly (see Constant Generation registers).

Assembler directives can be used to affect object file (e.g. .global), create data (e.g. .word or .byte) and myriad other conveniences. Not all assembler directives occupy bytes in the stored program.

The includes and binaries for the MSP are configured on host in particular places and allow us to use symbols via include files for most locations. We can write in C and look at equivalent assembly in .s file. Our source code must be suffixed .S to allow #include for such symbols.

The registers are named R0 .. R15 but R0 … R3 have hardware assigned purpose (PC, SP, SR / CG , CG) and R4 is used for a 'frame pointer' by agreement. R15 .. R12 are used for parameter passing, also by convention.

Programs you have written – conversions, bit shifting, msp430 led lighting , and hand encoding instructions. Parameter passing on MSP430.

6) **(32 points)** Consider the following addition of two 8 bit values (represented in hex).

			Unsigned	Signed (2's complement)
	BC	<==>	b)	c)
+	28	<==>	d)	e)
	----			
a)			f)	g)

a) What 8 bit (hex) pattern will result from the addition ?

b) What **decimal unsigned** value does BC represent ?

c) What **decimal signed** value does BC represent (using 2's complement) ?

d) What **decimal unsigned** value does 28 represent ?

e) What **decimal signed** value does 28 represent ?

f) What **decimal unsigned** value does your answer in a) represent?

g) What **decimal signed** value does your answer in a) represent?

How will the flags N Z V and C be set on this addition ? N __ Z __ V __ C __