CS 2810: Midterm Review Worksheet

Spring 2013

1 Two's complement

See: `http://en.wikipedia.org/wiki/Two's_complement`

Consider a **6-bit** two's complement representation. Fill in the empty boxes in the following table. Addition and subtraction should be performed based on the rules for 6-bit, two's complement arithmetic.

Note: `TMin` is the smallest (most negative, farthest from zero, largest magnitude in the negative direction) possible value, and `TMax` is the largest (most positive, farthest from zero, largest magnitude in the positive direction) possible number.

Number	Decimal Representation	Binary Representation
Zero	0	
n/a	-1	
n/a	-7	
n/a	-31	
n/a	16	
n/a	20	
n/a	31	
n/a		`0 11001`
n/a		`1 10100`
`TMax` + `TMax`		
`TMin` - 5		
`TMax` + 1		

2 Floats

See: `http://en.wikipedia.org/wiki/IEEE_754-1985`

Consider the following 5-bit floating point representation based on the IEEE floating point format:

- There is a sign bit in the most significant bit.
- The next 3 bits are the exponent. The exponent bias is $2^{3-1} - 1 = 3$.
- The last bit is the fraction
- The representation encodes numbers of the form $V = (-1)^s \times M \times 2^E$, where M is the significand (fraction part) and E is the biased exponent.

The rules are like those in the IEEE standard (normalized, denormalized, representation of 0, infinity, and NaN). Fill in the table below. Here are the instructions for each field:

- **Binary**: The 8-bit binary representation.
- M: The value of the significand. This should be a number of the form x or $\frac{x}{y}$, where x is an integer, and y is an integral power of 2. Examples include 0, $\frac{3}{4}$. Note: this is the numerical value after accounting for implicit bits, the binary point, etc., not just the value stored in the relevant bits. It should **not** account for the exponent value. This is the value M that should fit into the formula given above to compute the overall value represented.
- E: The integer value of the exponent. Note: this is numerical value of the exponent after accounting for the bias, etc., not just the value stored in the relevant bits. This is the value E that should fit into the formula given above to compute the overall value represented.
- **Value**: The numeric value represented.

Note: you need not fill in entries marked with "—".

Description	**Binary**	M	E	**Value**
Minus zero				-0.0
Positive Infinity		—	—	$+\infty$
	`01101`			
Smallest number > 0				
Largest denormalized number				
One				1.0
$4.0 - 0.75$				
$2.0 + 3.0$				

3 x86 assembly language

To compile a C function and see its x86 assembly code:

```
gcc -m32 -Os -S myfile.c
```

x86 assembly language basics:

- http://en.wikibooks.org/wiki/X86_Assembly/GAS_Syntax
- http://en.wikibooks.org/wiki/X86_Assembly/Data_Transfer
- http://en.wikibooks.org/wiki/X86_Assembly/Control_Flow
- http://en.wikibooks.org/wiki/X86_Assembly/Logic
- http://en.wikibooks.org/wiki/X86_Assembly/Arithmetic

Consider the following C functions and assembly code:

```
int fun1(int *ap, int *bp) {
    int a = *ap;
    int b = *bp;
    return a + b;
}

int fun2(int *ap, int *bp) {
    int b = *bp;
    *bp += *ap;
    return b;
}

int fun3(int *ap, int *bp) {
    int a = *ap;
    *bp += *ap;
    return a;
}
```

```
pushl %ebp
movl %esp, %ebp
movl 8(%ebp), %edx
movl 12(%ebp), %eax
movl %ebp, %esp
movl (%edx), %edx
addl %edx, (%eax)
movl %edx, %eax
popl %ebp
ret
```

3.1 Identification

Which of the functions on the left compiled into the assembly code on the right?

Write the function name here: .....

3.2 Modification

Pick one of the other two functions and modify the assembly language to match it.

Write the name of the function you chose here: .....