Christopher Martin, Jennifer Wolfe, Geoffrey Corey
April 22, 2013
CS 472 HW 3

Chapter 3 Homework Questions

(3.1) **Why is the program counter a *pointer* and not a *counter*?**

The program counter is used to access memory. The value held in the PC is treated as a memory address rather than an integer value.

(3.2) **Explain the function of the following registers in a CPU: PC, MAR, MBR, IR**

A. PC Program Counter points to the next instruction to be executed
B. MAR Memory Address Register contains the memory address currently be accessed.
C. MBR Memory Buffer Register hold data to be written to memory or read from it.
D. IR Instruction Register holds the fetched, currently executing instruction

(3.3) **For each of the following 6-bit operations, calculate the values of C, Z, V, N**

X	C	Z	V	N
A	0	0	0	0
B	1	1	0	0
C	0	0	0	0
D	1	0	0	0
E	0	0	0	1
F	1	0	0	1

(3.10) **Why does ARM provide a reverse subtract instruction?**

To negate a a value because ARM does not have an actual negation instruction.

(3.17) **ARM uses 12-bit literal. Compare and contrast the 8-bit format and 4-bit alignment vs straight 12-bit literal.**

Benefit: You can get more range out of a smaller bit value.
Con: You lose precision of that 32bit value due to its "compression"

(3.18) **Write one or more ARM instructions that will clear bits 20 to 25 inclusive in register r0. All other bits of r0 should remain unchanged.**

```
MOV         r1, #11111100000011111111111111111111
AND         r0, r0, r1
```

(3.19) **Swap contents of r0 and r1 without using any other registers or memory storage.**

```
EOR         r0, r0, r1
EOR         r1, r0, r1
EOR         r0, r0, r1
```

(3.25) **What is the binary encoding of the following instructions? A. STRB r1, [r2] B. LDR r3, [r4,r5] C. LDR r3,[r4],r5 D. LDR r3, [r4,#-6]**

A.

B.

C.

D.

(3.39) **Write ARM assembly that scans a null terminated string and copies the string from a source pointed to by r0 to a destination pointed to by r1**

```
Loop:
        LDR         r2, [r0], #1
        STR         r2, [r1], #1
        CMP         r2, #0          ; reached null, and stored in memory.
        BNE         Loop
```

(3.51) **Write ARM assembly that determines whether an odd length string is a palindrome or not. String is ASCII encoded, stored in memory. Pointer to beginning of string in r1, pointer to end of string in r2. On exit, r0 contains 0 if not palindrome, 1 if palindrome.**

```
pal:        MOV         r0, #0x0
again:        LDRB          r3, [r1]
        LDRB          r4, [r2]
        CMP         r3, r4
        BNE         notpal

        CMP         r1, r2
        BEQ        waspal
        ADD         r1, r1, #1
        SUB         r2, r2, #1
        B         again

waspal:         MOV         r0, #0x01
notpal:         MOV         pc, lr
```