CSE2312-002/003, Fall 2014, Homework 3

Paper Copy Due in Class: September 25, 2014 (at 2pm for 002, at 3:30pm for 003)

The following problems are from Chapter 2 of the ARM Edition of the Patterson and Hennessy textbook (available on Blackboard as PDFs under Course Materials). In addition to the resources available in Chapter 2, you may find Appendices B1 and B2 of the ARM Edition useful for these problems (also available as PDFs on Blackboard). Appendix A of the book may also be useful.

The problems below respectively correspond to the book problems 2.8 (all), 2.10 (all), 2.13 (all), 2.15 (all), 2.16 (all), and 2.18 (except 2.18.1).

Problems 2.15, 2.16, and 2.18 part b are all bonus and will be counted for extra credit, but there is no penalty for not completing them. However, part a for 2.15, 2.16, and 2.18 is required.

Exercise 2.8

The following problems deal with sign extension and overflow. Registers r0 and r1 hold the values as shown in the table below. You will be asked to perform an ARM operation on these registers and show the result.

a.	`r0=70000000sixteen, r1=0x0FFFFFFFsixteen`
b.	`r0=0x40000000sixteen, r1=0x40000000sixteen`

2.8.1 For the contents of registers r0 and r1 as specified above, what is the value of r4 for the following assembly code:

```
ADD r4, r0, r1
```

Is the result in r4 the desired result, or has there been overflow?

2.8.2 For the contents of registers r0 and r1 as specified above, what is the value of r0 for the following assembly code:

```
SUB r4, r0, r1
```

Is the result in r4 the desired result, or has there been overflow?

2.8.3 For the contents of registers r0 and r1 as specified above, what is the value of r4 for the following assembly code:

```
ADD r4, r0, r1
ADD r4, r0, r0
```

Is the result in r4 the desired result, or has there been overflow?

In the following problems, you will perform various ARM operations on a pair of registers, r0 and r1. Given the values of r0 and r1 in each of the questions below, state if there will be overflow.

a.	ADD r0, r0, r1
b.	SUB r0, r0, r1 SUB r0, r0, r1

2.8.4 Assume that register r0 = 0x70000000 and r1 = 0x10000000. For the table above, will there be overflow?

2.8.5 Assume that register r0 = 0x40000000 and r1 = 0x20000000. For the table above, will there be overflow?

2.8.6 Assume that register r0 = 0x8FFFFFFF and r1 = 0xD0000000. For the table above, will there be overflow?

Exercise 2.10

In the following problems, the data table contains bits that represent the opcode of an instruction. You will be asked to translate the entries into assembly code and determine what format of ARM instruction the bits represent.

a.	1010 1110 0000 1011 0000 0000 0000 0100two
b.	1000 1101 0000 1000 0000 0000 0100 0000two

2.10.1 For the binary entries above, what instruction do they represent?

2.10.2 What type (DP-type, DT-type) instruction do the binary entries above represent?

2.10.3 If the binary entries above were data bits, what number would they represent in hexadecimal?

In the following problems, the data table contains ARM instructions. You will be asked to translate the entries into the bits of the opcode and determine what the ARM instruction format is.

a.	ADD r0, r0, r5
b.	LDR r1, [r3, #4]

2.10.4 For the instructions above, show the hexadecimal representation of these instructions.

2.10.5 What type (DP-type, DT-type) instruction do the instructions above represent?

2.10.6 What is the hexadecimal representation of the opcode, Rd, and Rn fields in this instruction? For DP-type instruction, what is the hexadecimal representation of the Rd, I and operand2 fields?

Exercise 2.13

2.13 In the following problems, the data table contains the values for registers r3 and r4. You will be asked to perform several ARM logical operations on these registers.

a.	`r3 = 0x55555555, r4 = 0x12345678`
b.	`r3 = 0xBEADFEED, r4 = 0xDEADFADE`

2.13.1 For the lines above, what is the value of r5 for the following sequence of instructions:

```
ORR r5, r4, r3, LSL  #4
```

2.13.2 For the values in the table above, what is the value of r5 *(NOTE: there had been a typo asking for r2)* for the following sequence of instructions:

```
MVN r3, #1
AND r5, r3, r4, LSL #4
```

2.13.3 For the lines above, what is the value of r5 for the following sequence of instructions:

```
MOV r5, 0xFFEF
AND r5, r5, r3, LSR  #3
```

In the following exercise, the data table contains various ARM logical operations. You will be asked to find the result of these operations given values for registers r0 and r1.

a.	`ORR  r2, r1, r0, LSL #1`
b.	`AND  r2, r1, r0, LSR  #1`

2.13.4 Assume that r0 = 0x0000A5A5 and r1 = 00005A5A. What is the value of r2 after the two instructions in the table?

2.13.5 Assume that r0 = 0xA5A50000 and r1 = A5A50000. What is the value of r2 after the two instructions in the table?

2.13.6 Assume that r0 = 0xA5A5FFFF and r1 = A5A5FFFF. What is the value of r2 after the two instructions in the table?

Exercise 2.15

For these problems, the table holds some logical operations that are not included in the ARM instruction set. How can these instructions be implemented?

a.	`ANDN  r1, r2, r3    // bit-wise AND of r2, !r3`
b.	`XNOR  r1, r2, r3    // bit-wise exclusive-NOR`

2.15.1 The logical instructions above are not included in the ARM instruction set, but are described above. If the value of r2 is 0x00FFA5A5 and the value of r3 is 0xFFFF003C, what is the result in r1?

2.15.2 The logical instructions above are not included in the ARM instruction set, but can be synthesized using one or more ARM assembly instructions. Provide a minimal set of ARM instructions that may be used in place of the instructions in the table above.

2.15.3 For your sequence of instructions in 2.15.2 show the bit-level representation of each instruction.

Various C-level logical statements are shown in the table below. In this exercise, you will be asked to evaluate the statements and implement these C statements using ARM assembly instructions.

a.	`A = B & C[0];`
b.	`A = A ? B : C[0];`

2.15.4 The table above shows different C statements that use logical operators. If the memory location at C[0] contains the integer value 0x00001234, and the initial integer value of A and B are 0x00000000 and 0x00002222, what is the result value of A?

2.15.5 For the C statements in the table above, write a minimal sequence of ARM assembly instructions that does the identical operation.

2.15.6 For your sequence of instructions in 2.15.5, show the bit-level representation of each instruction.

Exercise 2.16

For these problems, the table holds various binary values for register *r0*. Given the value of *r0*, you will be asked to evaluate the outcome of different branches.

a.	1010 1101 0001 0000 0000 0000 0000 0010two
b.	1111 1111 1111 1111 1111 1111 1111 1111two

2.16.1 Suppose that register *r0* contains a value from above and *r1* has the value

```
0011 1111 1111 1000 0000 0000 0000 0000two
```

What is the value of r2 after the following instructions?

```
              MOV r2, #0
              CMP r0, r1
              BGE ELSE
              B      DONE
 ELSE:        MOV r2, #2
 DONE:
```

2.16.2 Suppose that register *r0* contains a value from above and *r1* has the value

```
0011 1111 1111 1000 0000 0000 0000 0000two
```

What is the value of r2 after the following instructions?

```
              MOV r2, #0
              CMP r0, r1
              BLO ELSE
              B    DONE
ELSE:         MOV r2, #2
DONE:
```

2.16.3 Rewrite the above code using conditional instructions of ARM.

For these problems, the table holds various binary values for register *r0*. Given the value of *r0*, you will be asked to evaluate the outcome of different branches.

a.	0x00001000
b.	0x20001400

2.16.4 Suppose that register *r0* contains a value from above. What is the value of *r2* after the following instructions?

```
MOV r2, #0
```

```
         CMP r0, r0
         BLT    ELSE
         B    DONE
ELSE:    ADD r2, r2, #2
DONE:
```

2.16.5 Suppose that register *r0* contains a value from above. What is the value of *r2* after the following instructions?

```
         MOV r2, #0
         CMP r0, r0
         BHI   ELSE
         B    DONE
ELSE:    ADD r2, r2, #2
DONE:
```

Exercise 2.18 (except 2.18.1)

For these problems, the table holds some C code. You will be asked to evaluate these C code statements in ARM assembly code.

a.	`for(i=0;  i<10;  i++)`
	`    a += b;`

b.	`while  (a <  10){`
	`    D[a] = b + a;`
	`    a += 1;`
	`}`

2.18.2 For the table above, translate the C code to ARM assembly code. Use a minimum number of instructions. Assume that the value *a, b, i, j* are in registers *r0, r1, r2, r3*, respectively. Also, assume that register r4 holds the base address of the array *D*.

2.18.3 How many ARM instructions does it take to implement the C code? If the variables *a* and *b* are initialized to 10 and 1 and all elements of *D* are initially 0, what is the total number of ARM instructions that is executed to complete the loop?

For these problems, the table holds ARM assembly code fragments. You will be asked to evaluate each of the code fragments, familiarizing you with the different ARM branch instructions.

a.	MOV r1, #100 LOOP: LDR r3, [r2, #0] ADD r4, r4, r3 ADD r2, r2, #4 SUB r1, r1, #1 CMP r1, #0 BNE LOOP
b.	ADD r1, r2, #400 LOOP: LDR r3, [r2, #0] ADD r4, r4, r3 LDR r3, [r2, #0] ADD r4, r4, r3 ADD r2, r2, #8 CMP r1, r2 BNE LOOP

2.18.4 What is the total number of ARM instructions executed?

2.18.5 Translate the loops above into C. Assume that the C-level integer *i* is held in register *r1*, *r4* holds the C-level integer called result, and *r2* holds the base address of the integer *MemArray*.

2.18.6 Rewrite the loop in ARM assembly to reduce the number of ARM instructions executed.