CSE2312-001 (Fall 2020)
Homework #2

Notes:

- With this homework, we move from abstract processor concepts to the ARM processor as used on the RPi when running Rasbian operating system.
- All numbers are in base-10 unless otherwise noted.
- If part of a problem is not solvable, explain why in the answer area.
- Print out this form and handwrite your answers in the spaces below.
- Place the hw2_prob5.s file and the scanned answers to problems 1, 2, 3, and 4 in a single zip file with name lastname_hw2.zip, where lastname is your last name as listed in MyMav.
- Submit the single zip file to Canvas before 11:59:00pm on October 1, 2020.

1. Assuming R0 contains 0x40000000, R1 contains 0x12345678, and R2 contains 0x00000002, write the contents of the memory locations below after the STR instruction writes to memory, assuming that each operation is independent. If a number not known, mark the blank with an "X".

a. STR R1, [R0]; assuming little-endian convention:

   Value at address 0x40000000 is _____

   Value at address 0x40000001 is _____

   Value at address 0x40000002 is _____

   Value at address 0x40000003 is _____

b. STRH R1, [R0]; assuming little-endian convention:

   Value at address 0x40000000 is _____

   Value at address 0x40000001 is _____

   Value at address 0x40000002 is _____

   Value at address 0x40000003 is _____

c. STRB R1, [R0]; assuming little-endian convention:

   Value at address 0x40000000 is _____

   Value at address 0x40000001 is _____

   Value at address 0x40000002 is _____

   Value at address 0x40000003 is _____

d. STR R1, [R0]; assuming big-endian convention:

   Value at address 0x40000000 is _____

   Value at address 0x40000001 is _____

   Value at address 0x40000002 is _____

   Value at address 0x40000003 is _____

e. STRH R1, [R0]; assuming big-endian convention:

   Value at address 0x40000000 is _____

   Value at address 0x40000001 is _____

   Value at address 0x40000002 is _____

   Value at address 0x40000003 is _____

f. STRB R1, [R0]; assuming big-endian convention:

   Value at address 0x40000000 is _____

   Value at address 0x40000001 is _____

   Value at address 0x40000002 is _____

   Value at address 0x40000003 is _____

g. STRB R1, [R0, R2]; assuming big-endian convention:

   Value at address 0x40000000 is _____

   Value at address 0x40000001 is _____

   Value at address 0x40000002 is _____

   Value at address 0x40000003 is _____

h. STRH R1, [R0, R2]; assuming big-endian convention:

   Value at address 0x40000000 is _____

   Value at address 0x40000001 is _____

   Value at address 0x40000002 is _____

   Value at address 0x40000003 is _____

2. Assuming the memory locations contain the data below.

Address	Data
0x50000000	0x12
0x50000001	0x57
0x50000002	0x33
0x50000003	0x8A
0x50000004	0x9A
0x50000005	0xC0

What is the value of R0 (all 32-bits in hex) after each of the following instructions executes assuming little-endian convention:

a. LDR R0, [R1] assuming R1 = 0x50000000

b. LDRH R0, [R1, R2] assuming R1 = 0x50000000 and R2 = 4

c. LDRSH R0, [R1] assuming R1 = 0x50000002

d. LDRB R0, [R1] assuming R1 = 0x50000005

e. LDRSB R0, [R1] assuming R1 = 0x50000005

f. LDRSB R0, [R1, R2] assuming R1 = 0x50000000 and R2 = 3

3. For each of these C functions, specify the ARM7 register(s) in which each argument is passed and result is returned.

a. uint32_t fn4(uint16_t a, uint32_t b, int8_t c, uint32_t d)

   a is passed in: _____

   b is passed in: _____

   c is passed in: _____

   d is passed in: _____

   the result is returned in: _____

b. uint64_t fn2(uint64_t a, uint64_t b)

   a is passed in: _____

   b is passed in: _____

   the result is returned in: _____

4. Determine the value of the 12-bit value stored as for operand2 for the 32bit immediate value of the following instructions, where m = n ROR 2s as described in the lectures:

a. MOV R0, #0x8300000

s = _____ n = _____

b. MOV R0, #135168

s = _____ n = _____

c. Show the encoding with MVN to move a value of -23 to R0.

s = _____ n = _____

5. Write assembly functions that implement the following C functions:

a. uint64_t addU32_U64(uint32_t x, uint32_t y) // returns x+y

b. int64_t addS64(int64_t x, int64_t y) // returns x+y

c. int32_t convertS8ToS32(int8_t x) // converts 8-bit signed value to 32-bits signed

d. int32_t convertU16ToS32(uint16_t x) // converts 16-bit unsigned value to 32-bits signed

e. int16_t maxS16(int16_t x, int16_t y) // returns the maximum of x, y

f. uint32_t maxU32(uint32_t x, uint32_t y) // returns the maximum of x, y

g. bool isGreaterThanU16(uint16_t x, uint16_t y) // returns 1 if x>y, 0 else

h. bool isGreaterThanS16(int16_t x, int16_t y) // returns 1 if x>y, 0 else

i. int32_t shiftRightS32 (int32_t x, uint8_t p) // returns x >> p = x*2^(-p) for p = 0..31

j. uint16_t shiftU16(uint16_t x, int8_t p) // return x*2^p for p = -31..31

k. bool isEqualU16(uint16_t x, uint16_t y) // returns 1 if x=y, 0 if x!=y

l. bool isStrEqual(const char* str1, const char* str2) // returns 1 if the strings are equivalent, 0 otherwise

m. void strCat(char* strTo, const char* strFrom) // concatenates strFrom to the end of strTo (make sure that strTo contains enough room for strFrom and strTo to prevent a seg fault)

All of the functions above should be present in a single file named hw2_prob5.s with functions callable from a C program. You do not need to submit the C files.