Homework 2 – Assembly Language Practice

Out: 2.8.21
Due: 2.16.21

1. [Instruction executions]
   For each instruction given below provide the following:
   i) Specify if the instruction legal.
   ii) If it is legal, evaluate it with the provided initial register values and give the value of $10 after each instruction execution, in HEX.

$1 = 0x0000D0A0	$5 = 0x00012BAD
$2 = 0x00000073	$6 = 0x0234EF01
$3 = 0xFFFFFF00	$7 = 0x00007B01
$4 = 0x00000FFF	$8 = 0x00000005

   a) add $10, $1, $2
   b) sub $10, $2, $1
   c) and $10, 0x00000345, 0x00000001
   d) lw $10, $8($7)
   e) add $10, $3, -10
   f) or $10, $3, 0x000000AB
   g) and $10, $5, 0x000000AB
   h) xor $10, $6, $5
   i) sll $10, $7, 3
   j) sra $10, $6, 2
   k) sra $10, $3, 1

2. [Memory Access]
   - Given the following memory map, for each instruction:
   i) Describe what it does
   ii) Write down the address of the memory operand
   iii) Write down what value is written to the destination register or memory location.
   iv) If the instruction is illegal or results in an error, say so.
   - Treat the instructions as a program, not individual instructions. Assume little endian.
   - Write the final content of the memory and registers.

LABEL	ADDRESS	INITIAL CONTENTS
a:	0	0xA0
	1	0xA1
b:	2	0xA2
	3	0xA3
c:	4	0xA4
	5	0xA5
	6	0xA6
	7	0xA7

d:	8	0xA8
	9	0xA9
	10	0xB0
	11	0xB1
e:	12	0xB2
f:	13	0xB3
g:	14	0xB4
h:	15	0xB5
i:	16	0xB6
	17	0xB7
	18	0xB8
	19	0xB9

```
LA   $t0,a
LW   $s0,d
SB   $t0,13($zero)
LA   $t3,f
LB   $t4,0($t3)
SW   $t0,i
LH   $t1,0($t3)
SH   $t1,g
LW   $t5,17($zero)
LBU  $t5,f
```

3. [Translating C code fragments to MIPS assembly]
   Given the following C declaration and code, write the appropriate MIPS assembly code. Use the appropriate loads and stores. You do not need to give the assembly language declarations.

```
main()
{
    int ia = 7;
    int ib = 0x23;
    int ic,id,ie,ig;

    ia = 0x1234;
    ib = ia;
    ic = ia + ib;
    id = ic | ib & 17;
    ie = ~ig;
    ig = (ia - ib) ^ (ic + id);
}
```

4. [Translating C to MIPS assembly – Conditionals]
   Rewrite the following C code in MIPS assembly language. Be sure to include all the code needed to access memory. You do not need to give the assembly language

declarations.

int a = 4;
int b = 30;
int c = 20;
int d = 10;

if (a == b) c = 33;
else if (b != c) a = 20;
else {
  if (a > b) b = 10;
  else if (c <= 10) c = 12;
  else a = 5;
}

5. [Memory and data references]
   a) What's the difference between LI, LA, and LW?
   b) Translate C to AL (you will need LI, LA, and LW!).
      You do not need to give the assembly language declarations.

      int VarA,VarB,VarC;
      *int *VarE, *VarF;*
      int VarD = 10;
      int ArrayA[100];

      VarA = 20;
      VarB = VarA;
      ArrayA[1] = VarB; // careful with this one (from previous year's mid-term)
      VarF = &VarE;

6. [Loops]
   In class, we talked about two methods of doing data transfers using for loops, (i) computing the A[i] and B[i] every iteration and (ii) using pointers which were incremented every iteration. In this problem, code data transfer with a loop using a different variation: Use A and B in the displacement slot and use a register to be the offset. For example, your memory operands for SW and LW could be A($t0) and B($t0), respectively. Demonstrate with array A containing 6 words (e.g. 1,2,3,4,5,6), that are copied to array B.

7. [Shifts and logicals]
   Give the instruction or instruction sequence to perform the following functions. Assume little endian.
   a) Negate $s0
   b) Take the two's complement of $s1
   c) Set bits 7, 14, 15, 26, and 29 in $s2
   d) Clear bits 2, 5, and 11 in $s3

e) Branch if and only if bits 1, 12, and 13 of $s4 are set
f) Shift $s5 to the right by 6 while preserving the sign
g) Move bits 5, 6, and 7 of $s6 to bits 0, 1, and 2 while clearing all other bits.