Ayman Hajja. Tue Jul 25 2017 20:48:53 GMT-0400 (EDT)

****

1) Provide the type and assembly language instruction for the following binary value:

100011 01011 10100 0000000000010010
I-Format   lw $s4, 18($t3)

****

2) Provide the type and assembly language instruction for the following binary value:

000000 01110 01111 01010 00000 101010
R-Format   slt $t2, $t6, $t7

****

3) Provide the type and assembly language instruction for the following binary value:

101011 10111 01010 1111111111110100
I-Format   sw $t2, -12($s7)

****

4) Provide the type and assembly language instruction for the following binary value:

000000 01010 10110 10010 00000 100100
R-Format   and $s2, $t2, $s6

****

5) Provide the type and assembly language instruction for the following binary value:

001101 01101 10100 1111111111110101
I-Format   ori $s4, $t5, -11

****

6) Provide the type and hexadecimal representation of the following instruction:

lw $s6, -8($s3)
I-Format   0x8E 76 FF F8

****

7) Provide the type and hexadecimal representation of the following instruction:

addi $t1, $s2, 0
I-Format   0x22 49 00 00

****

8) Provide the type and hexadecimal representation of the following instruction:

add $t5, $t5, $t7
R-Format 0x01 AF 68 20

****

9) Provide the type and hexadecimal representation of the following instruction:

lw $s7, 0($t4)
I-Format 0x8D 97 00 00

****

10) Provide the type and hexadecimal representation of the following instruction:

slti $s3, $s7, -3
I-Format 0x2A F3 FF FD

****

11) Provide the type, assembly language instruction, and hexadecimal representation of the instruction described by the following MIPS fields:

op = 0, rs = 3, rt = 2, rd = 3, shamt = 0, funct = 34
R-Format sub $v1, $v1, $v0 0x00 62 18 22

****

12) Provide the type, assembly language instruction, and hexadecimal representation of the instruction described by the following MIPS fields:

op = 0x23, rs = 1, rt = 2, const = 0x4
I-Format lw $v0, 4($at) 0x8C 22 00 04

****

13) Write the "pure" MIPS assembly code that loads the 32-bit constant/immediate below into register $t1

0010 0000 0000 0001 0100 1001 0010 0100
lui $t1, 0x2001
ori $t1, $t1, 0x4924

****

14) Convert the following MIPS code to binary (or machine language):

```
100 beq $s0, $s1, IF      # branch if ( i == j )
104 addi $s1, $s1, -1     # j = j - 1
108 j L1          # jump over else
```

```
112 IF: addi $s0, $s0, 1     # i = i + 1
116 L1: add $s1, $s1, $s0   # j = j + i
```

Note here that the address of each line/word is specified in decimal. When you convert to machine language, everything should be in binary. Please add at least one space between the fields of every instruction.

```
000100 10000 10001 0000000000000010
001000 10001 10001 1111111111111111
000010 00000000000000000000011101
001000 10000 10000 0000000000000001
000000 10001 10000 10001 00000 100000
```

****