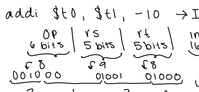
Homework 3

Tuesday, July 26, 2022 4:41 PM

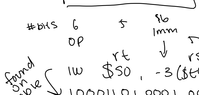
Q1

addi $t0, $t1, -10 → I-type



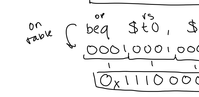
Explanation:
I type so use this template
use tables to find each value
→ convert to binary
→ convert to hex

0x2128FFF6



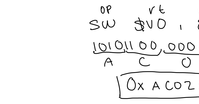
formed on table → convert to hex

0x8D10FFFD

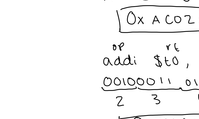


0x11100001

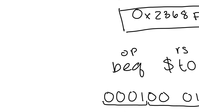
conditional I-type
label offset is imm value, 16 bits, convert to hex



0xAC020054

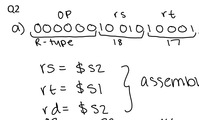


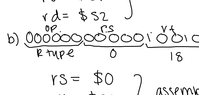
0x2368FFFC

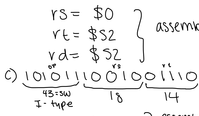


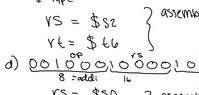
0x1110FFFC

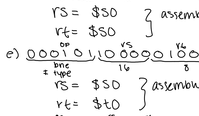
Q2

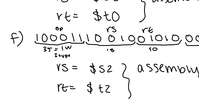












Q3

C code

```
Sum = 0;
for(i = 0; i>10; i++) {
    Sum = sum + LL[i];}
```

MIPS

```
# $s1 = i, $s2 = base address array L = 0x55552222
    lui   $s2, 0x5555          // load upper immediate
    ori   $s2, $s2, 0x2222
    addi  $s1, $0, 0
    addi  $t2, $0, 10
target: slt   $t0, $s1, $t2
    beq   $t0, $0, done
    sll   $t3, $s1, 2          // t3 = i*4 (byte offset)
    add   $t3, $t3, $s0        // address of LL[i]
    lw    $t4, 0($t3)
```

Q4

$t0 = c, $s0 = x, $s1 = y, $s2 = z

```
Switch (c) {
    case1: x = y + z;
           break;
    case2: x = y - z;
           break;
    default: x++;
           break;
}
```

```
    beq  $t0, $s0, case0  // case 0
    lw   $t1, 1           // load 1 to compare
    beq  $t0, $t1, case1  // case 1
    add  $s0, $0, 1       // default
case0: add  $s0, $s1, $s2
case1: sub  $s0, $s1, $s2
```

Q5

$t0 = random 32 bit int. $s0 = store count

(minimizing)

```
    lui  $t0, x
    ori  $t0, $t0, y      } loading the 32 bit int xy to reg $t0.
    addi $s0, $0, 0       } counter = 0
label1: beq  $t0, 0, label2
    and  $s1, $t0, 1
    beq  $s1, 0, label2
    add  $s0, $s0, 1
label2: srl  $t0, $t0, 1
    j    label1
end: ....
```

Q6

a) range is [0, 8) or (0 ≤ $s1 ≤ 7)

every shift left multiplies by 2

```
if ($s1 < 0) $t1 = 1
   else $t1 = 0
so if $t1 != 0 → done      So we want $t1 to equal 0.
   if $t1 == 0
   if ($s1 < 8) $t2 = 1    for $t1 = 0, $s1 must be greater than or equal to 0.
   else $t2 = 0
so if $t2 == 0 → done      → So we want $t2 = 1
   else add 1 + $s1        for $t2 = 1, $s1 must be less than 8
done...
```

b) because its unsigned so there are no negative numbers: no negative # means must be above 0. and

```
line1: if ($s1 < 8) $t1 = 1
       else $t1 = 0
line2: if $t1 = 0 → done
       else, continue
```

∴ we want $t1 in range [0, 8)

for example: if $s1 = 3

mips	meaning
slt iu $t1, $s1, 8	if (3 < 8) $t1 = 1
beq $t1, $0, done	if $t1 == 0 → done ✗
addi $s1, $s1, 1	else, continue ↓ increment $s1 by 1
	if (4 < 8) $t1 = 1
	& repeat until $s1 = 8, then $t1 == 0 → done