Wenzheng Kang JHU 605.204

7.

Memory Address	Binary representation
3CB0	1111 0010 1100 0000
3CB1	1111 0010 1100 0001
3CB2	1111 0010 1100 0010
3CB3	1111 0010 1100 0011
3CB4	1111 0010 1100 0100
3CB5	1111 0010 1100 0101
3CB6	1111 0010 1100 0110
3CB7	1111 0010 1100 0111

index bit size = $\log_2(4) = 2$ bits.

tag value = 12 bits.

Index	Tag	Tag
00	1111 0010 1100	1111 0010 1100
01	1111 0010 1100	1111 0010 1100
10	1111 0010 1100	1111 0010 1100
11	1111 0010 1100	1111 0010 1100

Given the list of memory address,

0X3CB4 (m), 0X3CB1 (m), 0X3CB7 (m), 0X3CB2 (m), 0X3CB6 (m), 0X3CB1 (H), 0X3CB5 (m), 0X3CB0 (m), 0X3CB6 (H), 0X3CB7 (H), 0X3CB3 (m), 0X3CB5 (H), 0X3CB0 (H),

∴ 5 cache hits intotal.

10.

10. beq $t0, $t1, 0x10001

I-format: 6 bits opcode | 5 bits rs | 5 bits rt | 16 bits immediate

4_{16} 8_{10} 9_{10} 0x10001

000100 01000 01001 10000000000000001

6 bits | 5 bits | 5 bits | 17 bits.

The immediate is 17 bits binary. So it's not possible to assemble beq $t0, $t1, 0x10001 into hex.

12.

5_{10}	16_{10}	19_{10}	2_{10}
5_{hex}	$s0	$s3	2_{10}

bne $s0, $s3, 2

00101 10000 10011 0000000000000010.

23. 0x29100028

00101001 00010 0000000000000101000.

opcode = 001010 = A I-type instruction

rs = 01000 = 8_{10} => $t0

rt = 10000 = 16_{10} => $s0

offset = 40_{10}

slti $s0, $t0, 40

24) 120400.28125

120400 = 11101011 00101 0000

28125 = 11011011011101

Combine two parts together, get the number in normalized format, we move the radix 16 spots to the left

so $1.1101\ 0110\ 1010\ 0000\ 1101\ 1011\ 1011\ 101 \times 2^{16}$

Add bias of 127 to 16 get $143 = 10001111$

We need 23 bits to represent the fraction.

Then we get the binary representation

0 10001111 1101 0110 1010 0000 0100100 = $47EB2824_{16}$

Sign bit

25) nor $s4, $t7, $s1 — R-type instr.

opcode = 0 20_{10} 15_{10} 17_{10} funtion code = $27_{16} = 100111$

Shamt

000000 10100 01111 10001 00000 100111

$\Rightarrow$ A3E209

30) $16385_{10} = 01000\ 0000\ 0000\ 001$

$-16385_{10} = 11000\ 0000\ 0000\ 001$ (sign magnitudes)

$= C001_{16}$

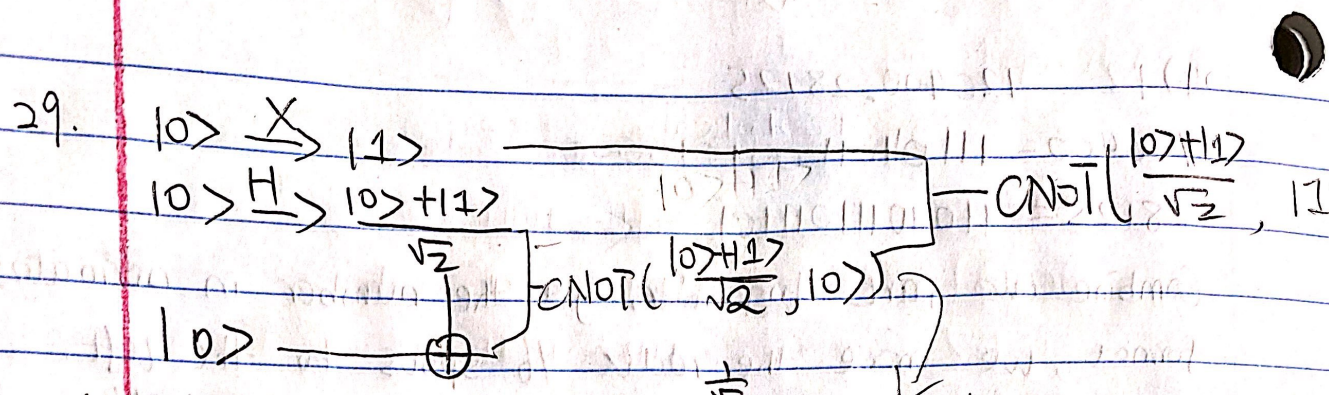
$19473_{10} = 1001\ 1000\ 0010\ 001$

flip each digit 0110 0111 1101 110

plus one + 1

0110 0111 1101 111

29.



$$\frac{|0\rangle+|1\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}}\left(\begin{bmatrix}1\\0\end{bmatrix}+\begin{bmatrix}0\\1\end{bmatrix}\right)=\begin{bmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{bmatrix}$$

$$|0\rangle \otimes \frac{|0\rangle+|1\rangle}{\sqrt{2}} = \begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{bmatrix}=\begin{bmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\\ 0\\ 0\end{bmatrix}$$

$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\\ 0\\ 0\end{bmatrix}=\begin{bmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\\ 0\\ 0\end{bmatrix}=\frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\0\\0\end{bmatrix}=\frac{|00\rangle+|01\rangle}{\sqrt{2}}$$

similarly, $|1\rangle \otimes \frac{|00\rangle+|01\rangle}{\sqrt{2}} = \begin{bmatrix}0\\1\end{bmatrix}\otimes\begin{bmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{bmatrix}=\begin{bmatrix}0\\0\\ \frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{bmatrix}$

$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}0\\0\\ \frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{bmatrix}=\begin{bmatrix}0\\0\\ \frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{bmatrix}=\frac{1}{\sqrt{2}}\begin{bmatrix}0\\0\\1\\1\end{bmatrix}=\frac{|10\rangle+|11\rangle}{\sqrt{2}}$$

The possibility of each classical bits 00, 01 is 50%

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Q20.asm

```
1  .data
2  in_msg: .asciiz "Type your integer here: "
3  str1: .asciiz "There were "
4  zero: .asciiz " zeros.\n"
5  one: .asciiz " ones."
6
7  arr: .word 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1
8
9    .text
10 la $a0, in_msg
11 li $v0, 4
12 syscall
13
14 li $v0, 5
15 move $t0, $v0
16 syscall
17
18 addi $t0, $zero, 0 # set i = 0, loop count
19 addi $s1, $zero, 108 # set bound
20 addi $s2, $zero, 0 # zero count
21 addi $s3, $zero, 0 # one count
22 addi $s4, $zero, 1
23
24 Loop:
25         slt $t1, $t0, $s1 # i < 27
26         beq $t1, $zero, ENDLOOP # exit when i < 27 is false
27
28         # a) get arr[i]
29         lw $t2, arr($t0)
30
31         # b) increment $s2 if zero, increment $s3if one
32         slti $t5, $t2, 1 #if arr[i] < 1, arr[i] is zero
33         beq $t5, $0, Else #if arr[i] < 1 not true, arr[i] is one
34         addi $s2, $s2, 1 # increment zero counter
35         j Endif
```

Line: 59 Column: 1 Show Line Numbers

Mars Messages Run I/O

Clear

```
Type your integer here: 104837249
There were 14 zeros.
There were 13 ones.
-- program is finished running (dropped off bottom) --
```

Name
$zero
$at
$v0
$v1
$a0
$a1
$a2
$a3
$t0
$t1
$t2
$t3
$t4
$t5
$t6
$t7
$s0
$s1
$s2
$s3
$s4
$s5
$s6
$s7
$t8
$t9
$k0
$k1
$gp
$sp
$fp
$ra
pc
hi
lo

File Edit Run Settings Tools Help

Run speed at max (no interaction)

Edit Execute

Q20.asm

```
37          Else:
38                   addi $s3, $s3, 1 # increment one counter
39
40          Endif:
41
42                  addi $t0, $t0, 4
43                  j Loop
44
45  ENDLOOP:
46
47    # c) print summary
48  la $a0, str1
49  li $v0, 4
50  syscall
51
52  li $v0, 1
53  move $a0, $s2
54  syscall
55
56  la $a0, zero
57  li $v0, 4
58  syscall
59
60  la $a0, str1
61  li $v0, 4
62  syscall
63
64  li $v0, 1
65  move $a0, $s3
66  syscall
67
68  la $a0, one
69  li $v0, 4
70  syscall
71
```

Line: 71 Column: 1 Show Line Numbers

Mars Messages Run I/O

Clear

```
Type your integer here: 104837249
There were 14 zeros.
There were 13 ones.
-- program is finished running (dropped off bottom) --
```