Computation I (5EIA0) Exam Training (CA) Answers 2021

Original authors SPIM: Tom van Nunen, Sjors van Riel, Thomas Lippens
Version 2021-2022: Esther Maas

Monday November 1, 2021

1 CA questions

Answer 1

Miss cycles per instruction:
I-cache: $0.08 \times 50 = 4.0$ cycles
D-cache: $0.4 \times 0.2 \times 50 = 4.0$ cycles
Actual CPI = CPI_base + Miss_cycles per data fetch + Miss_cycles per instruction = 4.0 + 4.0 + 4.0 = 12.0
Ideal processor is 12.0 / 4.0 = 3 times faster

The correct answer is: 3

Answer 2

Number of words in a data-block = $2^{(32-18-10-2)} = 2^2 = 4$ words.

Cache size (bits) = Ncache_blocks * associativity * block_size = 2^10 * 2 * (32*4 + 18 + 1 + 1) = 1024 * 4 * 148 = 606208 bits

The correct answer is: 606208

Answer 3

Tag size = N_address_bits - index - 2log(number_words_in_block) - 2log(number_bytes_in_word) = 32 - 12 - 2log(8) - 2log(4) = 32 − 12 − 3 - 2 = 15 bits

The correct answer is: 15

Answer 4

One sw instruction is one word. A block of 16 bytes can have 16/4 = 4 words.

The correct answer is: 4

Answer 5

Number of cache locations = 256 bytes / 4 bytes per block (1 word of 4 bytes each) = 64

The memory block address (not byte address, but block address) = 516 / 4 bytes = 129

Cache block number (location or entry) = 129 modulo 64 = 1

The correct answer is: 1

Answer 6

The minimum number of instructions per second (worst-case frequency) is determined by the critical timing path. For the 5-stage pipelined architecture this can be any of the 5-stages. From the given timings it is clear the ALU stage is the most critical, since its hardware blocks take highest amount of time.

It includes Mux,and ALU itself (1ns + 4ns).

Therefore, the stage requires 5 ns.

Therefore the minimum number of instructions per second (worst-case instruction execution frequency) f_max = 1/5ns = 200 MHz.

The correct answer is: 200

Question 7

The minimum number of instructions per second (worst-case frequency) is determined by the critical timing path. Usually a LW (load word) instruction triggers this critical path. The critical path of a LW requires the following time (all in ns):

6 (Instruction fetch) + 2 (Register read) + 6 (ALU address calculation) + 6 (Data read) + 2 (Register writeback) + 3 (3 Muxes in the path of LW) = 25 ns.

The Control blob operates in parallel and takes much less time (1 ns), so should not be taken into the computation.

Therefore the worst-case frequency $f_max = 1/25ns = 40MHz$.

The correct answer is: 40 MHz

Question 8

The correct answer is: Data and Control hazards

Question 9

The Addition instruction should wait in its RegisterRead stage for the computation of \$s0. The Subtraction instruction computes the value for \$s0.

When Addition gets into the RegisterRead stage, the Subtraction instruction just gets into the ALU stage.

Addition should wait for ALU (20ns) and RegisterWrite (20ns) stages of Subtraction, before \$s0 becomes available.

The correct answer is: 40

Answer 10

First, convert the instruction 0x08000001 into binary:
00001000000000000000000000000001

Its a J jump instruction, since the opcode is 000010.
We see that the instruction jumps to 26-bit word-address 00000000000000000000000001. To build up the full 32-bit address, we concatenate:
- in front: the first 4 bits from the current PC value
- at the end: 00 to get not word-address, but byte-address

The first 4 bits of the current PC value (0x00ffffff) are 0000 (just convert the hexadecimal 0x00ffffff into binary).

The resulting address is 0000 0000000000000000000000000100.
Convert to decimal = 4.

The correct answer is: 4

Answer 11

One of the efficient conversions:

```
lw $s1, 0($s4)
lw $s2, 4($s5)
lw $t0, 12($s6)
sub $s0, $s1, $s2
sub $s0, $s0, $t0
```

In total: 5 assembly instructions

The correct answer is: 5

Answer 12

I-format instruction
Opcode of beq: 000100
$t0 register number is 9: 01000
$t2 register number is 10: 01010
Offset is -4 - PC = -4 - 1 = -5. Binary value in two's complement: 1111111111111011

The correct answer is: 00010001000010101111111111111011

Question 13

```
main:
    move $22,$0      # This initializes $22 to zero
    li $21, 50       # initialize register 21 − addition operand (interval) to 50
    li $v0, 5        # Set $v0 to 5, this tells syscall
                       to read an integer from the console
    syscall
    move $23,$v0     # $23 is a register used as a counter (here you enter 4)
loop:
    beq $0,$23,quit # if the counter is zero then quit
    add $22,$22,$21 # $22 = $22 + $21
    addi $23,$23,−1 # $23 = $23 − 1 (update counter)
    j loop
quit:
    move $a0, $22    # Load the resulting value to print into $a0
    li $v0, 1        # Set $v0 to 1, this tells syscall to
                       print the integer specified by $a0
    syscall          # Now print the integer
```

As a result, four iterations, four times adding 50 = 200

The correct answer is: 200

Question 14

```
    .text
main:
    li $v0, 5            # read number into $v0
    syscall              # make the syscall read_int
    move $t0, $v0        # move the number read (10) into $t0
    li $v0, 5            # read second number into $v0
    syscall              # make the syscall read_int
    move $t1, $v0        # move the number read (10) into $t1
    sub $t2, $t0, $t1    # 10−10 = 0
    move $a0, $t2        # move the number 0 to print into $a0
    li $v0, 1            # load syscall print_int into $v0
    syscall              #
    li $v0, 10           # syscall code 10 is for exit
    syscall
```

The correct answer is: The program will print 0 to console

Question 15

The correct answer is: **Instruction Memory, Mux, Register Read, Sign-extend, Mux, ALU, Data Memory, Mux, Register Write**

Question 16

The correct answer is: **Instruction Memory, Mux, Register Read, Mux, ALU, Mux, Register Write**

Question 17

The correct answer is **c**:

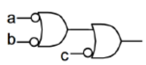

Figure 1:

Question 18

The correct answer is **OR gate or XOR gate.**

2 Spim

In the SPIM programming exercises the following SPIM directives are used:

`.text`	Start text segment.
`.data`	Start data segment.
`.asciiz str`	Store the string in data memory
`.word 1,40,...,n`	Store `n` words (32 bit) into data memory.i
`.space n`	Reserve `n` bytes in data memory.

This are the SPIM System Calls:

Table 1: SPIM System Calls

Function	Syscall (value in $v0)	Arguments	Return value
Print integer	1	$a0: integer	
Print string	4	$a0: string	
Read integer	5		$v0: integer
Exit	10		

Printing Negative Numbers

```
        .text
main:   la $s1, N
        lw $s1, 0($s1)
        la $s2, A
        mul $s3, $s1, 4
        add $s3, $s2, $s3
loop:   bge $s2, $s3, done
        lw $t0, 0($s2)
        bge $t0, $zero, skip
        move $a0, $t0
        li $v0, 1
        syscall
skip:   addi $s2, $s2, 4
        j loop
done:   li $v0, 10
        syscall

        .data
A:      .word 79, -13, -90, 80, 12, 0, -4, 23, -45, 7
N:      .word 10
```

Calculating Sum of Absolute Values

```
        .text
main:   move $0, $zero
        la $s1, N
        lw $s1, 0($s1)
        la $s2, A
        mul $s3, $s1, 4
        add $s3, $s2, $s3
loop:   bge $s2, $s3, done
        lw $t0, 0($s2)
        bge $t0, $zero, skip
        sub $t0, $zero, $t0
skip:   add $s0, $s0, $t0
        addi $s2, $s2, 4
        j loop
done:   move $a0, $s0
        li $v0, 1
        syscall
        li $v0, 10
        syscall

        .data
A:      .word 79, -13, -90, 80, 12, 0, -4, 23, -45, 7
N:      .word 10
```