IT5002 Computer Systems and Applications

2024/25 Semester II

Midterm Assessment

Question 1.

Choose ALL the items below that are not a part of the processor datapath or control covered in the lectures.

a. General purpose registers
b. Program Counter
c. +Main Memory
d. Arithmetic-Logic Unit
e. Decoder
f. +Keyboard

Question 2.

Fill in the blanks below:

$3440_X + 4042_X = 11522_X$

X = _____

Answer: 6

$3320_4 + 1332_4 =$ _____$_4$

Answer = _____$_4$

Answer: 11312_4

6432_8 - _____$_8 = 4533_8$

Answer: 1677_8

Convert the following to 32-bit IEEE-754 format. Express your answer in hexadecimal with no leadings 0's and leave out the 0x.

-137.625

Answer: 0x_____

Binary = $0b10001001.101 \times 2^0$

$$= 0b1.0001001101 \times 2^7$$

Encoding:

Mantissa = 00010011010000000000000

Exponent = 7 + 127 = 10000110

Total = 1 10000110 00010011010000000000000

= 1100 0011 0000 1001 1010 0000 0000 0000

= 0xC309A000

Question 3.

We have the following code in C:

```
do{
      x = x + 1;
} while(x < 10);
```

Which ONE of the following pieces of MIPS assembly best implements this C fragment? Assume that x is mapped to register $s0.

a.

```
L:    slti $t0, $s0, 10
      beq $t0, $zero, E:
      addi $s0, $s0, 1
      j L
E:    .. other code ..
```

b.

```
+L:   addi $s0, $s0, 1
      slti $t0, $s0, 10
      bne $t0, $zero, L
E:    … other code …
```

c.

```
L:    slti $t0, $s0, 10
      bne $t0, $zero, E
      addi $s0, $s0, 1
      j L
E:    … other code …
```

d.

```
L:    addi $s0, $s0, 1
      slti $t0, $s0, 10
      beq $t0, $zero, L
E:    … other code …
```

e. None of the above options a. to d. are correct.

Question 4.

We are given the following program that processes an integer array A. We assume all integers are 4 bytes long and that we are using byte addressing.

	addi $t0, $s0, 0	# This is at address 0x91F28
	addi $t1, $s0, 40	
	addi $s1, $zero, 0	
A:	slt $t2, $t0, $t1	
	beq $t2, $zero, C	# This beq
	lw $t2, 0($t0)	
	andi $t3, $t2, 1	
	beq $t3, $zero, B	
	add $s1, $s1, $t2	
B:	addi $t0, $t0, 4	
	j A	# This j
C:		# Other irrelevant code

A. What does this program do?

a. It adds all the elements of array A into $s1
b. +It adds all the odd elements of A into $s1 (e.g. 3, 7, 1, 15, etc)
c. It adds all the even elements of A into $s1 (e.g. 8, 2, 12, 22, etc)
d. It adds alternate elements of A into $s1 (e.g. A[0], A[2], A[4], etc)
e. None of the above options a. to d. are correct.

B. Which register contains the base address of A? Leave out the $.

Answer: $ _____

$s0

C. How many elements of A are processed by this code?

Answer: 10

D. Assume that each array element holds its own index, e.g. A[0]=0, A[1]=1, A[2]=2, A[3]=3, etc. How many instructions are executed by the code above?

Answer: 80

Outside loop: 3
In loop:
Odd numbers: 8
Even numbers: 7
Exit: 2
Total = $5 \times 7 + 5 \times 8 + 2 = 77$
Total = 80 instructions

E. What is the instruction encoding for the beq statement labelled “This beq”? Express your answer in hexadecimal with no leading 0’s and leave out the 0x.

Answer: 0x_____

opcode = 0x4 = 000100
$t2 = $10 = 01010
$zero = $0 = 00000
Offset = 6 instructions = 0000 0000 0000 0110
Total = 000100 01010 00000 0000 0000 0000 0110
= 0001 0001 0100 0000 0000 0000 0000 0110
= 0X 11400006

F. What is the instruction encoding for the j statement labelled “This j”? Express your answer in hexadecimal with no leading 0’s and leave out the 0x.

Answer: 0x_____

	addi $t0, $s0, 0	# This is at address 0x91F28
	addi $t1, $s0, 40	0x91F2C
	addi $s1, $zero, 0	0x91F30
A:	slt $t2, $t0, $t1	0x91F34

Address to jump to is 0x91F34

0000 0000 0000 1001 0001 1111 0011 0100

= 0000 0000 1001 0001 1111 0011 01

Opcode = 0x2 = 000010

Total = 000010 0000 0000 1001 0001 1111 0011 01

= 0000 1000 0000 0010 0100 0111 1100 1101

= 0X80247CD

Question 5.

Consider the following code being executed on the MIPS datapath and control that you learnt about in the lectures:

```
        addi $22, $zero, 5
        addi $23, $zero, 0
L:      sub $24, $25, $26
        add $24, $23, $21
        addi $23, $23, 1
        bne $22, $23, L          # This bne
```

We consider the bne instruction marked "# this bne". Fill in the control signal values, 32-bit signed extended immediate value, and other values stated below in hexadecimal, **after executing this bne statement, at the end of the first iteration.** For each answer leave out the 0x and all leading 0's. Note that register numbers in the program above are specified in decimal (e.g. $22 is decimal 22).

For single-bit binary values, fill "X" if the value is a "don't care", i.e. can be either 0 or 1. In cases of "don't care", if you fill in 0 or 1 you will get the answer wrong.

Immediate = -4 = -(0000 0000 0000 0100)

= 1111 1111 1111 1100_{2s}

= 0xFFFC

RegDst: 0x X

RegWrite: 0x 0

Read Register 1 (RR1): 0x 16

Read Register 2 (RR2): 0x 17

Read Data 1 (RD1): 0x 5

Read Data 2 (RD2): 0x 1

32-bit sign-extended Immediate (leave a space between each byte):

0x FF FF FF FF FF FF FF FC

ALUSrc: 0x 0

ALUcontrol: 0x 6

ALU result: 0x 4

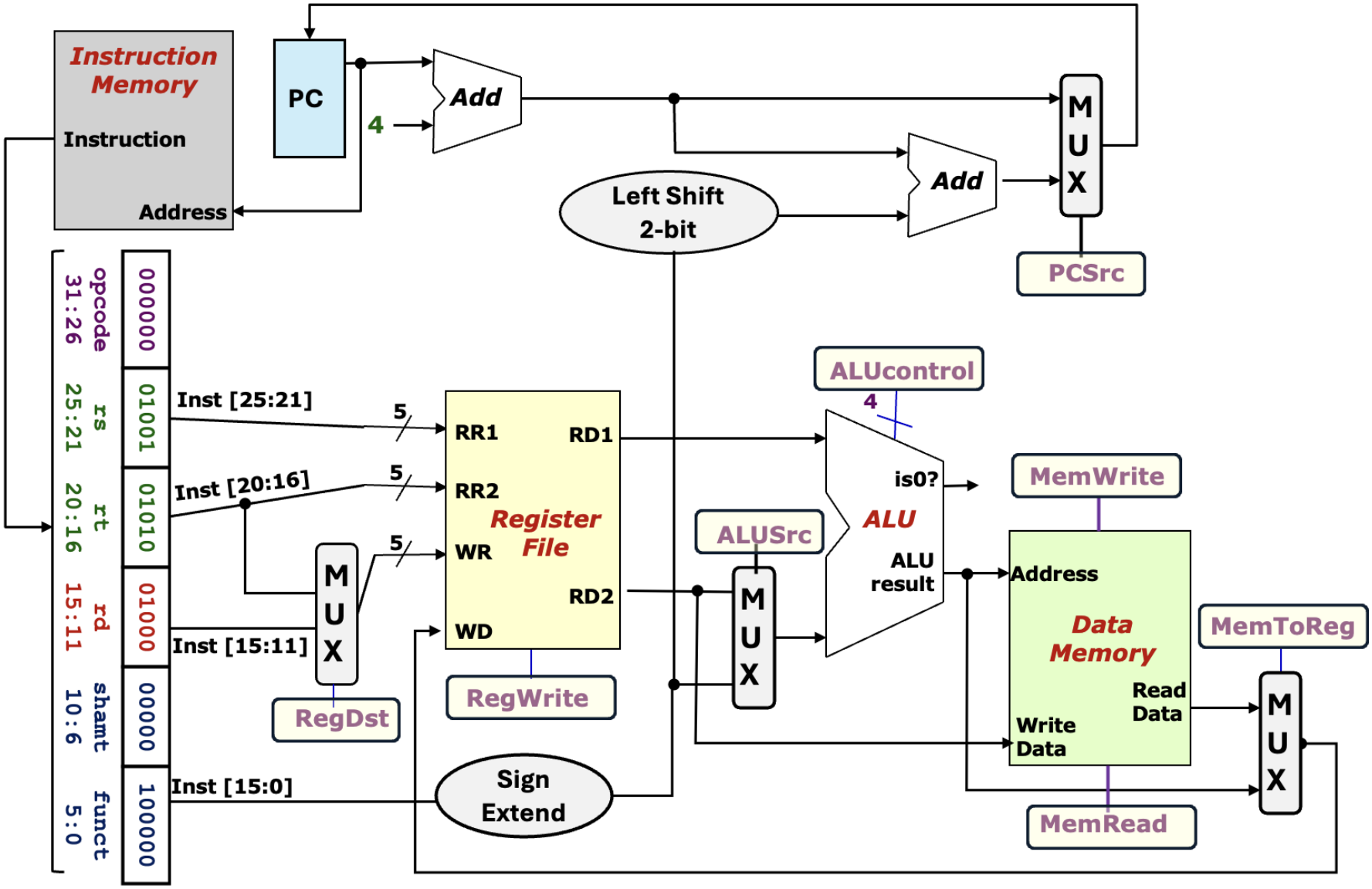
Is0: 0x 0

MemRead: 0x 0

MemWrite: 0x 0

PCSrc: 0x 1

MemToReg: 0x X


Instruction Memory
Instruction
Address
PC
4
Add
Add
Left Shift 2-bit
M U X
PCSrc
opcode 31:26
rs 25:21
rt 20:16
rd 15:11
shamt 10:6
funct 5:0
000000
01001
01010
01000
00000
100000
Inst [25:21]
Inst [20:16]
Inst [15:11]
Inst [15:0]
5
5
5
RR1
RR2
WR
WD
RD1
RD2
Register File
RegWrite
RegDst
ALUcontrol
4
ALUSrc
is0?
ALU
ALU result
MemWrite
Address
Data Memory
Read Data
Write Data
MemRead
MemToReg
Sign Extend