Final Exam: 4/29 @ 2pm

BN-2N A-LIU, YI

BN-2S LIU, YU – Z

Exam Office hours: 4/11 12-1pm, 4/15 12-1pm

4/24 2-5pm

4/25 9am-1pm

or by appt.

Open book/notes (same as midterm)

No calculator

Topics:

- Number representation
- Assembly
  - endian, C to assembly, loops, arrays, subroutines
- Memory-mapped I/O
  - assembly, circuits, new device, serial transmission, UART, GPIO, polling, interrupts, ISRs
- CPUs
  - Single cycle, multicycle, control signals, datapath, new instruction, instruction encoding
- Memory/Storage
  - types of caches/organizations, memories, making big mems from small chips

Final exam is comprehensive - All material from lab/lecture is fair game

* Please fill out online course evaluation by April 13! *

1. Given the following cache and cache configuration:

   2-way set associative

   4 byte cache line ⇒ 2 bit offset

   32 byte cache ⇒ 32B/4B = 8 blks → 2 blks/set ⇒ 4 sets

   10-bit address

   Write back and LRU replacement

   Tag = 10 - 2 - 2 = 6

Tag	Set	Offset
6	2	2

You will be given a list of memory operations. **The accesses do not occur one after another. You should assume that before each question, the state of the cache is as given above.** In the space given, fill in the effects of the operation and give the state of the affected line *after* the cache operation.

	Block #	Tag (binary)	Data(3..0)	Valid	Dirty	MRU
Set Index = 0	0	010110	BB 33 14 A4	0	0	0
	1	010010	5C 50 37 49	0	0	1
Set Index = 1	0	100110	FF 59 B7 94	0	1	0
	1	100111	73 95 CB 11	1	0	1
Set Index = 2	0	111111	FF FF EE EE	1	0	0
	1	110011	19 AA BB CC	1	0	1
Set Index = 3	0	100100	A4 4A 55 66	1	1	0
	1	000011	12 34 56 78	1	0	1

(a) Operation: CPU writes 0xFF to 0x24E = 0b10010011110

Hit (H) or Miss (M):	Fetch from memory? No ☒ Yes ☐, from to

Cache State	Block #	Tag (binary)	Data	Valid	Dirty
Set Index = 3	0	100100	AA FF 55 66	1	1

	Block #	Tag (binary)	Data(3..0)	Valid	Dirty	MRU
Set Index = 0	0	010110	BB 33 14 A4	0	0	0
	1	010010	5C 50 37 49	0	0	1
Set Index = 1	0	100110	FF 59 B7 94	0	1	0
	1	100111	73 95 CB 11	1	0	1
Set Index = 2	0	111111	FF FF EE EE	1	0	0
	1	110011	19 AA BB CC	1	0	1
Set Index = 3	0	100100	A4 4A 55 66	1	1	0
	1	000011	12 34 56 78	1	0	1

(b) Operation: CPU reads 1 byte from 0x265 = 0b1001100101

Hit (H) or Miss (M):	Fetch from memory? No ☐ Yes ☒, from 0x264 to 0x267

Cache State	Block #	Tag (binary)	Data	Valid	Dirty
Set Index = 1	0	100110	mm mm mm mm	1	0

	Block #	Tag (binary)	Data(3..0)	Valid	Dirty	MRU
Set Index = 0	0	010110	BB 33 14 A4	0	0	0
	1	010010	5C 50 37 49	0	0	1
Set Index = 1	0	100110	FF 59 B7 94	0	1	0
	1	100111	73 95 CB 11	1	0	1
Set Index = 2	0	111111	FF FF EE EE	1	0	0
	1	110011	19 AA BB CC	1	0	1
Set Index = 3	0	100100	A4 4A 55 66	1	1	0
	1	000011	12 34 56 78	1	0	1

(c) Operation: CPU writes 0xFF to 0x338 = 0b110011|1000

Hit (H) or Miss (M):	Fetch from memory? No ☐ Yes ☐, from to

Cache State	Block #	Tag (binary)	Data	Valid	Dirty
Set Index = 2	1	110011	19 AA BB FF	1	1

2. Processor implementation

The new instruction takes the format of:

MAC R1 Imm2

Which implements K1 = K1 + R1 × 2^{Imm2} where Imm2 is an unsigned value. The functionality is as follows:

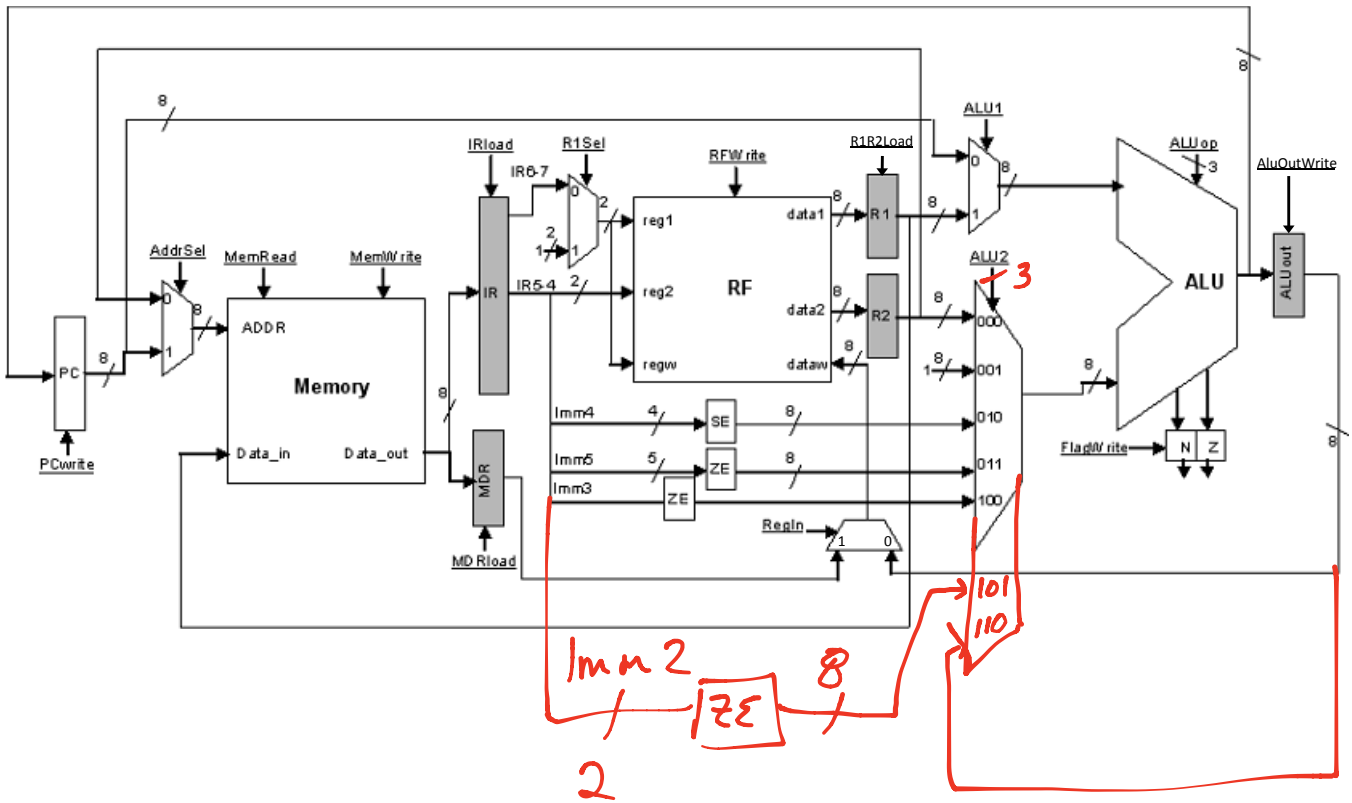
1. K1=K1+(R1<<Imm2)
2. PC=PC+1

The encoding is as follows:

7	6	5	4	3	2	1	0
R1		Imm2		1110			

This question has two parts: a) Fill in a table with the steps this instruction will perform each clock cycle, and b) indicate any modification to the datapath. On the next page, add any component you deem necessary (wires, multiplexers, registers, etc.).

Cycle	Instruction Steps
1	IR = Mem[PC] PC = PC + 1
2	R1 = RF[IR7..6] R2 = RF[IR5..4]
3	ALUout = R1 << Imm2 R1 = RF[1]
4	ALUout = R1 + ALUout
5	RF[1] = ALUout
6	
7	


PC
PCwrite
AddrSel
MemRead
MemWrite
ADDR
Memory
Data_in
Data_out
IRload
IR
IR6-7
IR5-4
R1Sel
RFWrite
reg1
reg2
regw
RF
data1
data2
dataw
R1
R2
R1R2Load
MDR
MDRload
Imm4
Imm5
Imm3
SE
ZE
RegIn
ALU1
ALU2
ALUop
ALU
ALUout
AluOutWrite
FlagWrite
N
Z
000
001
010
011
100
101
110
Imm2
ZE
8
2
3

3. Building Memories

How many 512 x 4-bit memory chips would be needed to build:

(a) A 2KByte memory?

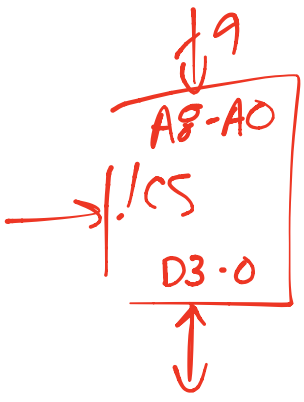
2048 x 8-bit 512 x 4 = 2048 entries > 8
x2 = 8 bit wide

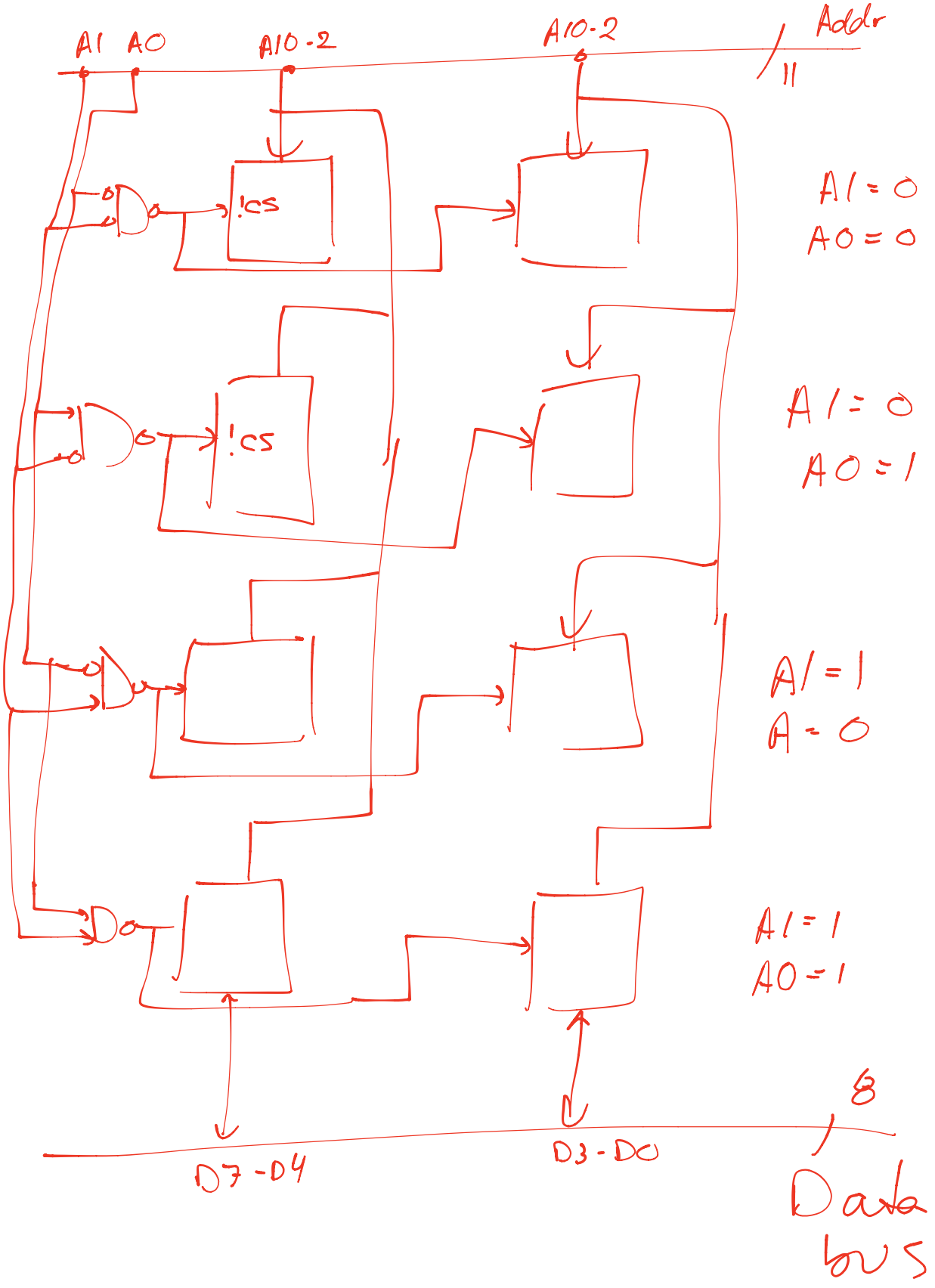
(b) A 32KByte memory?

32K x 8-bit ⇒ $2^{15}/2^9$ ⇒ $2^6 = 64$ x 2 ⇒ 128

(c) A 4K x 12 bit memory?

Draw the design for a 2K x 8 bit memory built using 512 x 4 bit memory chips




A1 A0
A10-2
A10-2
Addr
11
!cs
!cs
A1 = 0
A0 = 0
A1 = 0
A0 = 1
A1 = 1
A = 0
A1 = 1
A0 = 1
D7-D4
D3-D0
8
Data
bus

4. Assembly

Assuming that initially all registers contain zero, show using eight hexadecimal digits the final value for each register after executing all the instructions of the following program. The final value for r16 is given. "foo" is a function that abides by the NIOS II calling conventions and is properly behaved. The code is placed starting from address 0x00001000, that is main = 0x00001000. If you cannot determine the value of a register write UNKNOWN and explain why.

```
      .data
xyz:      .word bobos, 0x2, 0x3, 0x4
bobos:    .byte 0x10, 0x20, 0x30, 0x40, 0x50, 0x60, 0x70

     .text
     .global main
main:
     movia     r16, xyz

     ldw       r17, 0(r16)

     ldw       r18, 0(r17)

     addi      r19, r0, 0x123F

     addi      r20, r0, -2

     and       r21, r20, r19

     or        r22, r20, r19

     addi      r10, r0, 10

     call      foo

     movia     r8, bobos

     beq       r8, r17, dodo
koko:
     srli      r9, r16, 8
     br        done
dodo:
     slli      r9, r16, 4
done:
```

r16 = 0x0fab1000	r21 =
r17 =	r22 =
r18 =	r8 =
r19 =	r9 =
r20 =	r31 =
r10 =	