Question 1. (20 Points)

Given the following code, a 5-stage MIPS pipeline like the one we studied in class, and the following assumptions:

- You have a branch delay slot, and assume branch taken.
- The decision for branching is made in the ID stage.
- You have forwarding from the end of EX stage until the beginning of the WB stage, and no other forwarding.

```
1   loop: addi $1, $0, 0x0001
2         addi $2, $0, 0xF001
3         slt  $3, $2, $1
4         bne  $3, $0, N1
5         addi $1, $0, 0xA001
6         addi $2, $0, 0x0001
7   N1:   slt  $2, $1, $0
8         beq  $2, $0, loop
```

a) How many times do you enter the loop? Explain your answer.

It is an infinite loop as after the loop the proceeding instruction on line 4 "SLT $2, $1, $0" is never possible because $0 = 0 and it is not possible for $1 to be less than 0 without being a signed value. Therefore, $1 < $0 is never true forcing $2 to be 0 on each passthrough. Thus the following BEQ instruction is satisfied each time, branching back to the loop

b) Draw arrows on the code above to show ALL data dependencies. Circle the required register, and draw the arrow toward the place where that register's contents are required.

Question 2. (30 Points)

For a direct-mapped cache design with a 32-bit address, the following bits of the address are used to access the cache:

Tag	Index	Offset
31-12	11-4	3-0

Starting from power on, the following byte-addressed cache references are recorded in sequence of from left to right:

Address (decimal)					
1000	3100	2048	128	130	140

a) What is the cache block size (in words?) How many entries does the cache have? *(Show your work for full credit)*

block size = $2^{\#index-2} = 2^4 = 64$ words

#entries = $2^{\#index} = 2^8 = 256$ entries

b) How many blocks are replaced? What is the hit ratio? Show your work.

		V	TAG	INDEX	Hit/miss	
1000 →	11 11101000	1	10	111110	miss	- replace
3100 →	11000011100	1	3	000001	miss	- replace
2048 →	10000000000	1	2	000000	miss	- replace
128 →	10000000	1	0	001000	miss	- replace
130 →	10000010	1	0	001000	hit	
140 →	10001100	1	0	001000	hit	

hit ratio = #hits / total = 2/6

4 replacements

c) List the final state of the cache in the table below, with each valid entry represented as a record of <index,tag,data>

Index (decimal)	Tag (decimal)	Data (MEM[*addr*])
62	0	MEM[11000]
1	3	MEM[3100]
0	2	MEM[2048]
8	0	MEM[140]

Question 3. (30 Points)

Below questions refer to a clock cycle in which the processor fetches the following instruction word:

100011 10001 10000 0000000000100000 Load FP double

Assume that data memory is all zeros and that the processor's registers have the following values at the beginning of the cycle in which the above instruction word is fetched:

$0	$1	$2	$8	$9	$10	$15	$16	$17	$18	$PC
0x55	0x11	0x1020	0x3120	0x5	0x15	0x2000	0x16	0x5000	0x700	0x2000

Answer the following by referring to Figure 1 and information provided in Page 10, 11 and 12:

a) What is the output of the Sign-extend unit (32bits) and the Input 1 of MUX 2 (output of Jump Shift Left 2 unit (Jump address[31-0], found in top left)) for this instruction word?

Sign-extend = 0000 0000 0000 0000 0000 0000 0010 0000₂

MUX 2, Input 1 = 1000110000000000000100000₂

b) What are the values of the ALU control unit's inputs for this instruction?

LW → ALUOp = 00

funct = xxxxxx

ALU function = add

ALU control = 0010

c) What is the new PC address after this instruction is executed? Explain the path through which this value is determined.

PC_{new} = 2004

- 200 sent to Add 4
- Result sent to MUX 1 input 0
- MUX1 input 0 sent to MUX2 input 0
- MUX2 input 0 sent to PC

d) For each MUX (MUX 1 to MUX 5), show the values of its data output during the execution of this instruction and these register values.

MUX 1 → input 0 = 2004
MUX 2 → input 0 = 2004
MUX 3 → input 0 = inst[20-16] = 10000
MUX 4 → input 1 = sign extend inst[15-0] = 0000 0000 0000 0000 0000 0000 0000 0000
MUX 5 → input 0 = Read data @ 0x5020

e) For the ALU and the two add units, what are their data input values?

Add (PC+4)
- 2000
- 4

Add (Branch)
- PC+4
- sign extend [15-0] shift left 2 = 0000 0000 0000 0000 0000 0000 0000 0000

ALU
- 0x5000
- MUX 4 input 1 = sign extend [15-0] = 0000 0000 0000 0000 0000 0000 0000 0000

f) What are the values of all inputs (both data and control signals) for the "Registers" unit?

Read Reg 1 = 10001
Read Reg 2 = 10000
Write Reg = 10000
Write data =
Reg Write = 1

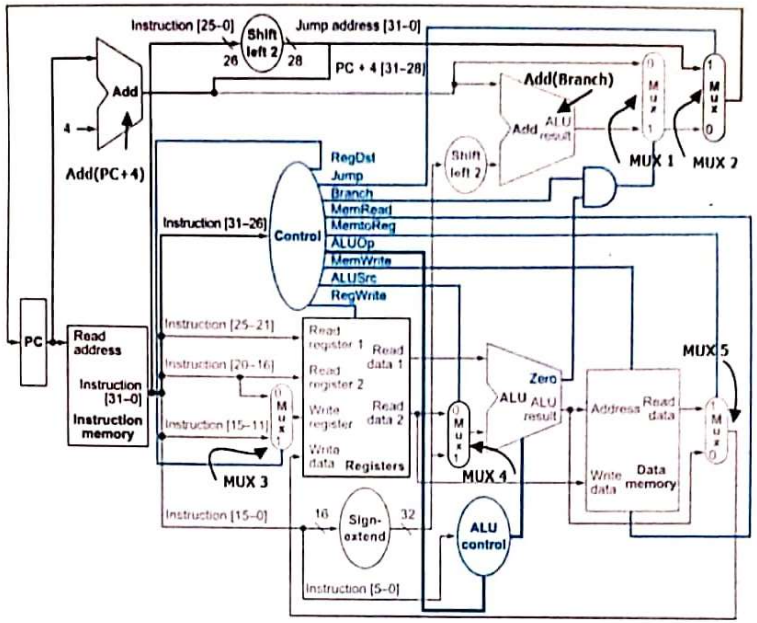

Figure 1. Single Cycle Implementation of MIPS Processor

opcode	ALUOp	Operation	funct	ALU function	ALU control
lw	00	load word	XXXXXX	add	0010
sw	00	store word	XXXXXX	add	0010
beq	01	branch equal	XXXXXX	subtract	0110
R-type	10	add	100000	add	0010
		subtract	100010	subtract	0110
		AND	100100	AND	0000
		OR	100101	OR	0001
		set-on-less-than	101010	set-on-less-than	0111

Table 1. ALU and ALUop Control Signals

Name	Fields						Comments
Field size	6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	All MIPS instructions are 32 bits long
R-format	op	rs	rt	rd	shamt	funct	Arithmetic instruction format
I-format	op	rs	rt	address/immediate			Transfer, branch, imm. format
J-format	op	target address					Jump instruction format

Table 2. MIPS Instruction format

Name	Register number	Usage	Preserved on call?
$zero	0	The constant value 0	n.a.
$v0–$v1	2–3	Values for results and expression evaluation	no
$a0–$a3	4–7	Arguments	no
$t0–$t7	8–15	Temporaries	no
$s0–$s7	16–23	Saved	yes
$t8–$t9	24–25	More temporaries	no
$gp	28	Global pointer	yes
$sp	29	Stack pointer	yes
$fp	30	Frame pointer	yes
$ra	31	Return address	yes

Table 3. MIPS Register Conventions

op(31:26)								
28-26 31-29	0(000)	1(001)	2(010)	3(011)	4(100)	5(101)	6(110)	7(111)
0(000)	R-format	bltz/gez	jump	jump & link	branch eq	branch ne	blez	bgtz
1(001)	add immediate	addiu	set less than imm.	set less than imm. unsigned	andi	ori	xori	load upper immediate
2(010)	TLB	FIPt						
3(011)								
4(100)	load byte	load half	lwl	load word	load byte unsigned	load half unsigned	lwr	
5(101)	store byte	store half	swl	store word			swr	
6(110)	load linked word	lwcl						
7(111)	store cond. word	swcl						

Table 4. MIPS Instruction Encoding

op(31:26)=01000 (TLB), rs(25:21)								
23-21 25-24	0(000)	1(001)	2(010)	3(011)	4(100)	5(101)	6(110)	7(111)
0(00)	mfc0		cfc0		mtc0		ctc0	
1(01)								
2(10)								
3(11)								

op(31:26)=000000 (R-format), funct(5:0)								
2-0 5-3	0(000)	1(001)	2(010)	3(011)	4(100)	5(101)	6(110)	7(111)
0(000)	shift left logical		shift right logical	sra	sllv		srlv	srav
1(001)	jump register	jalr			syscall	break		
2(010)	mfhi	mthi	mflo	mtlo				
3(011)	mult	multu	div	divu				
4(100)	add	addu	subtract	subu	and	or	xor	not or (nor)
5(101)			set l.t.	set l. t. unsigned				
6(110)								
7(111)								

Table 4. MIPS Instruction Encoding

Question 4. (20 Points)
Answer the following questions on cache and virtual memory:

a) Explain a scenario that the instruction cache (I-Cache) would have higher miss rate than the data cache (D-Cache).

If the data cache has a larger block size than the instruction cache then the I-Cache would have a higher miss rate

b) Instead of fetching only the desired data from the memory requested by the processor, the cache system always fetches one or a few blocks of data from the memory into the cache. Explain why the cache system prefers this configuration.

If multiple data sets are fetched to the cache each time then that will increase the chances of getting a hit when accessing the cache. It could also spatially grab extra data that is needed by other instructions.

c) Why no tag is required in Virtual Memory configuration?

For virtual memory cache data is mapped directly without a tag as it can be placed at any location within the cache.

d) Explain why the space on the disk (swap space) is reserved for the full virtual memory space of a process

The swap space is reserved to function as additional storage space for the virtual memory, additionally it provides organization of tasks into an file thus reducing the needed operations