Name-Last Name: _____ **Student ID:** _____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructor: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam	Exam Date: 1.6.2016
Duration: 120 minutes	

Questions	1	2	3	4	Total
Marks	20	40	20	20	100
Earned					

Q1. You are given the following MIPS assembly code for a procedure called proc1.

```
proc1:  add  $s1, $0, $0
        add  $v0, $0, $0
count:  beq  $a0, $0, done
        andi $s0, $a0, 0x1
        beq  $s0, $0, shift
        addi $s1, $s1, 1
shift:  srl  $a0, $a0, 1
        jal  proc1
done:   add  $v0, $v0, $s1
        jr   $ra
```

(a) proc1 function does not successfully return when it is called from the main function. Fix the code so that it successfully returns. (Hint: You should modify one of the instructions.) [5]

(b) Using the corrected code, write the return value ($v0) for given values of $a0. Write your answer in decimal. [10]

a0 | 0x000F | v0 | |

a0 | 0x0013 | v0 | |

(c) Describe in words what proc1 does. [5]

Q2. Following parameters are given for byte addressable memories:

Capacity of virtual memory	8 pages
Capacity of main memory	2 pages
Capacity of cache	4 words
TLB access time	1 ns
Cache access time	50 ns

Page size	4 words
Block size	1 word
Word size	4 bytes
TLB/Cache associativity	2-way
Replacement type	LRU

Suppose the page table, TLB, and cache have the following data in them.

Page Table

VPN	V	PPN
7		
6		
5	1	0
4		
3		
2	1	1
1		
0		

Cache

	Way 1				Way 0		
	U	V	Tag	Data	V	Tag	Data
Set 1	1	1	11	D	1	10	C
Set 0	1	1	01	B	1	00	A

TLB

	Way 1				Way 0		
	U	V	VPN	PPN	V	VPN	PPN
	1	1	010	1	1	101	0

Based on the above information, answer the following questions: [Each is 5 points.]

(a) How many bits do we need to address the virtual memory?

(b) How many bits do we need to address the main memory?

You are given the virtual address **0x24**.

(c) What is the physical address for this virtual address?

(d) Is this memory access is a hit or miss on the cache? Why? Show your work.

(e) If it is hit, which data is accessed? (A,B,C,D?) If not, where do we bring the new data?

(f) After the memory access 0x24, is there any changes necessary on the TLB or cache? If yes, show the changes made on TLB and/or cache.

Cache

		Way 1			Way 0		
	U	V	Tag	Data	V	Tag	Data
Set 1	1	1	11	D	1	10	C
Set 0	1	1	01	B	1	00	A

TLB

	Way 1			Way 0		
U	V	VPN	PPN	V	VPN	PPN
1	1	010	1	1	101	0

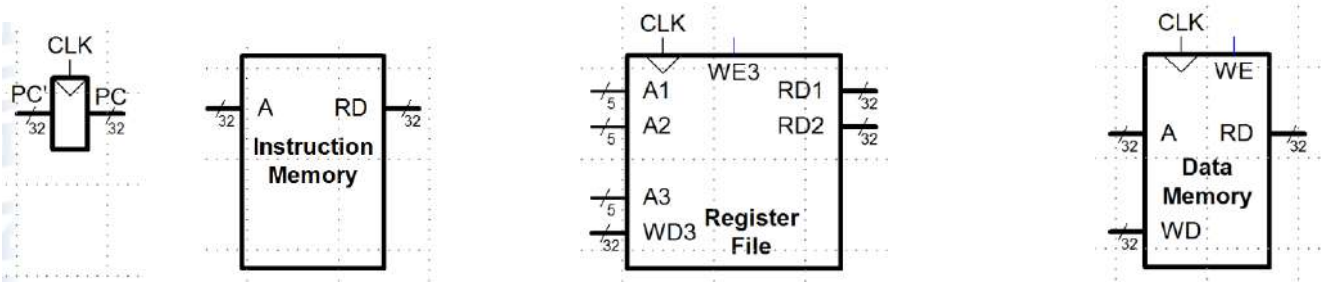
(g) What is the total access time of data?

(h) What is TLB and why is it used?

Q3. You have 32-bit Program Counter (PC), instruction memory, register file and a data memory.

(a) Give an example R-type instruction and write its machine code format, including the bit numbers. (You do not need to give the binary values. You can write the labels of each field of the format.) [5]

(b) Draw the single cycle microarchitecture for ONLY R-type instructions. You can use extra hardware blocks if necessary. You should clearly show the bit numbers. [15]



Q4. You are given the following MIPS code:

```
      addi $t0, $0, 2
loop: beq $t0, $0, done
      lw $s0, 0($a0)
      lw $s1, 4($a0)
      add $s0, $s0, $s1
      sw $s0, 0($a0)
      addi $t0, $t0, -1
      addi $a0, $a0, 4
      j loop
done:jr $ra
```

a) How many cycles does it take to execute this code on a single-cycle processor? [3]

b) How many cycles does it take to execute this code on a pipelined processor (with data hazard unit)? (If any NOPs necessary, write it on the above code.) [5]

c) Suggest a modification for the code (without adding or deleting any instructions) so that new code takes less time to execute. Explain your changes. [4]

d) In some cases, loop unrolling (writing the code sequentially instead of using loops) helps reduce the CPU time. Rewrite the above code without using any loops. You can use extra registers. Write your code in such a way that there is no stall or NOP penalty. Now, how many cycles does this code takes on pipelined processor? [8]

Name-Last Name: _____ **Student ID:** _____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructor: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam	Exam Date: 1.6.2016
Duration: 120 minutes	

Questions	1	2	3	4	Total
Marks	20	40	20	20	100
Earned					

Q1. You are given the following MIPS assembly code for a procedure called proc1.

```
proc1:  add  $s1, $0, $0
        add  $v0, $0, $0
count:  beq  $a0, $0, done
        andi $s0, $a0, 0x1
        beq  $s0, $0, shift
        addi $s1, $s1, 1
shift:  srl  $a0, $a0, 1
        jal  proc1
done:   add  $v0, $v0, $s1
        jr   $ra
```

(a) proc1 function does not successfully return when it is called from the main function. Fix the code so that it successfully returns. (Hint: You should modify one of the instructions.) [5]

When jal proc1 instruction executes, we loose the content of register ra, which is the return address to the main. We should change this instruction to "j count" so that the loop can executes again.

(b) Using the corrected code, write the return value ($v0) for given values of $a0. Write your answer in decimal. [10]

a0 | 0x000F | v0 | 4

a0 | 0x0013 | v0 | 3

(c) Describe in words what proc1 does. [5]

It counts the number of 1's in register a0.

Q2. Following parameters are given for byte addressable memories:

Capacity of virtual memory	8 pages
Capacity of main memory	2 pages
Capacity of cache	4 words
TLB access time	1 ns
Cache access time	50 ns

Page size	4 words
Block size	1 word
Word size	4 bytes
TLB/Cache associativity	2-way
Replacement type	LRU

Suppose the page table, TLB, and cache have the following data in them.

Page Table

VPN	V	PPN
7		
6		
5	1	0
4		
3		
2	1	1
1		
0		

Cache

	Way 1				Way 0		
	U	V	Tag	Data	V	Tag	Data
Set 1	1	1	11	D	1	10	C
Set 0	1	1	01	B	1	00	A

TLB

	Way 1				Way 0		
U	V	VPN	PPN	V	VPN	PPN	
1	1	010	1	1	101	0	

Based on the above information, answer the following questions: [Each is 5 points.]

(a) How many bits do we need to address the virtual memory? 7

The number of bytes in virtual memory = number of pages in VM x page size in words x word size in bytes = 8x4x4=128. Thus, we can use 7 bits to address 128 bytes.

(b) How many bits do we need to address the main memory? 5

The number of bytes in main memory = number of pages in MM x page size in words x word size in bytes = 2x4x4=32. Thus, we can use 5 bits to address 32 bytes.

You are given the virtual address **0x24**.

(c) What is the physical address for this virtual address?

Since the last two bits are byte offset, and the next two bits are page offset (4 words in a page), the remaning bits are used to translate virtual pages to physical pages.

VA= 0x24 is 010 0100 in binary. The physical page number for page 010 ise 1. Then, the physical address is 10100.

(d) Is this memory access is a hit or miss on the cache? Why? Show your work.

The physical address is 10100. The last two bits are byte offset. Since we have two sets in cache, the next bit is for set. We should look to set 1. The tag is 10, which is the tag in way 0. And the valid bit (V) is 1. Thus, it is a hit.

(e) If it is hit, which data is accessed? (A,B,C,D?) If not, where do we bring the new data?

C

(f) After the memory access 0x24, is there any changes necessary on the TLB or cache? If yes, show the changes made on TLB and/or cache.

Cache

		Way 1			Way 0		
	U	V	Tag	Data	V	Tag	Data
Set 1	1	1	11	D	1	10	C
Set 0	1	1	01	B	1	00	A

TLB

	Way 1			Way 0		
U	V	VPN	PPN	V	VPN	PPN
~~1~~0	1	010	1	1	101	0

In the TLB, Least Recently Used (LRU) data now is in way 0. Thus, U bit should be changed to 0.

(g) What is the total access time of data?

TLB is hit and cache is hit. TLB time + Cache time = 1ns+1ns=2ns

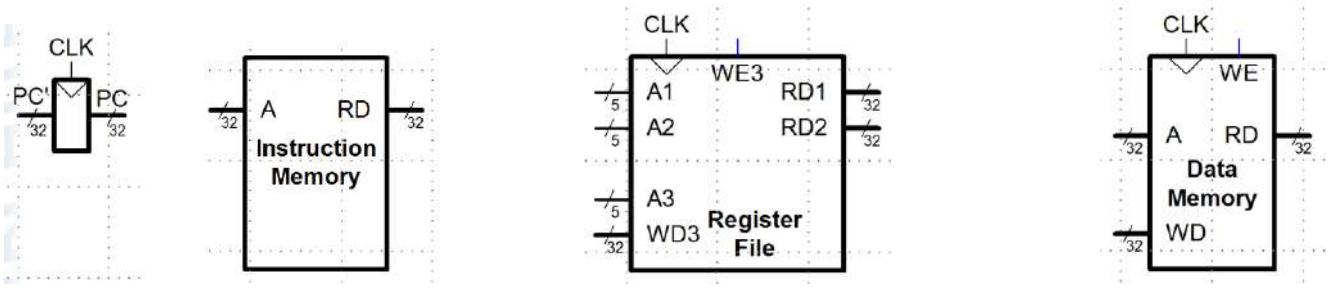
(h) What is TLB and why is it used?

TLB is the cache for the page table. It stores the page table entry for the recently accessed data. In this way, it reduces the number of main memory accesses.

Q3. You have 32-bit Program Counter (PC), instruction memory, register file and a data memory.

(a) Give an example R-type instruction and write its machine code format, including the bit numbers. (You do not need to give the binary values. You can write the labels of each field of the format.) [5]

(b) Draw the single cycle microarchitecture for ONLY R-type instructions. You can use extra hardware blocks if necessary. You should clearly show the bit numbers. [15]



Q4. You are given the following MIPS code:

```
                                  # of executions
      addi $t0, $0, 2             1
loop: beq $t0, $0, done           3
      lw $s0, 0($a0)              2
      lw $s1, 4($a0)              2
      add $s0, $s0, $s1           2
      sw $s0, 0($a0)              2
      addi $t0, $t0, -1           2
      addi $a0, $a0, 4            2
      j loop                      2
done:jr $ra                       1
```

a) How many cycles does it take to execute this code on a single-cycle processor? [3]

Total number of instructions fetched is 19. So, it takes 19 cycles.

b) How many cycles does it take to execute this code on a pipelined processor (with data hazard unit)? (If any NOPs necessary, write it on the above code.) [5]

```
                                  # of fetches
      addi $t0, $0, 2             1
loop: beq $t0, $0, done           3
      lw $s0, 0($a0)              3 (last fetch is flushed)
      lw $s1, 4($a0)              2
      NOP                         2
      add $s0, $s0, $s1           2
      sw $s0, 0($a0)              2
      addi $t0, $t0, -1           2
      addi $a0, $a0, 4            2
      j loop                      2
done:jr $ra                       3 (first three fetches are flushed)
```

Total number of fetches is 24. The last instruction needs 4 more cycles to finish. Then, the answers is 28 cycles.

c) Suggest a modification for the code (without adding or deleting any instructions) so that new code takes less time to execute. Explain your changes. [4]

# of fetches

```
        addi $t0, $0, 2        1
loop:   beq $t0, $0, done      3
        lw $s0, 0($a0)         3  (last fetch is flushed)
        lw $s1, 4($a0)         2
        addi $t0, $t0, -1      2
        add $s0, $s0, $s1      2
        sw $s0, 0($a0)         2
        addi $a0, $a0, 4       2
        j loop                 2
done:   jr $ra                 3  (first three fetches are flushed)
```

We move `addi $t0, $t0, -1` instruction in place of NOP. Then, we gain 2 cycles. The total number of cycles becomes 26.

d) In some cases, loop unrolling (writing the code sequentially instead of using loops) helps reduce the CPU time. Rewrite the above code without using any loops. You can use extra registers. Write your code in such a way that there is no stall or NOP penalty. Now, how many cycles does this code takes on pipelined processor? [8]

```
lw $s0, 0($a0)
lw $s1, 4($a0)
lw $s2, 8($a0)
add $s0, $s0, $s1
add $s1, $s1, $s2
sw $s0, 0($a0)
sw $s1, 4($a0)
```

Now, 7 instructions are fetched, plus 4 cc. The total execution time is 11 clock cycles.

Name-Last Name: _____ **Student ID:** _____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructors: Assoc. Prof. Dr. Suleyman TOSUN
Final Exam	Assist. Prof. Dr. Mehmet KOSEOGLU
Duration: 120 minutes	Exam Date: 01.06.2017

Questions	1	2	3	4	5	Total
Marks	20	20	20	20	20	100
Earned						

Q1. a) Write the values of the registers and the stack for the following MIPS program. The value of the stack pointer is initially sp=0x7FFFFFFC.

Address		Instructions			
0x00400000		lui $s0, 0x1000		s0=	
0x00400004		ori $s0, $s0, 0x0008		s0=	
0x00400008		lw $a0, -4($s0)		a0=	
0x0040000C		addi $a1, $s0, -8		a1=	
0x00400010		lw $s1, 4($s0)		s1=	
0x00400014		add $a2, $s1, $0		a2=	
0x00400018		jal Proc1			
0x0040001C		addi $s2, $v0, $0		s2=	
0x00400020					
0x00400024	Proc1:	addi $sp, $sp, -12		sp=	
0x00400028		sw $ra, 8($sp)		Write the stored values on stack!	
0x0040002C		sw $s0, 4($sp)			
0x00400030		sw $s1, 0($sp)			
0x00400034		addi $v0, $0, 0			
0x00400038	Loop:	beq $a0, $0, Done			
0x0040003C		lw $s0, 0($a1)			
0x00400040		slt $s1, $s0, $a2			
0x00400044		beq $s1, $0, Next			
0x00400048		addi $v0, $v0, 1			
0x0040004C	Next:	addi $a0, $a0, -1			
0x00400050		addi $a1, $a1, 4			
0x00400054		j Loop			
0x00400058	Done:	lw $ra, 8($sp)		ra=	
0x0040005C		lw $s0, 4($sp)		s0=	
0x00400060		lw $s1, 0($sp)		s1=	
0x00400064		addi $sp, $sp, 12		sp=	
0x00400068		jr $ra			

Address	Data
0x100000000	13
0x100000004	10
0x100000008	21
0x10000000C	15
0x100000010	7
0x100000014	16
0x100000018	11
0x10000001C	6
0x100000020	30
0x100000024	28

Address	Stack Data
0x7FFFFFFC	XXXXXXXX
0x7FFFFFF8	
0x7FFFFFF4	
0x7FFFFFF0	

b) Briefly describe what Proc1 function does.

Q2. Consider a virtual memory system that can address a total of 2^{32} bytes. You have only 8 MB of physical (main) memory. Assume that page size is 4 KB.

(a) How many bits is the physical address?

(b) What is the maximum number of virtual pages in the system?

(c) How many physical pages are in the system?

(d) How many bits are the virtual page numbers?

(e) How many bits are the physical page numbers?

(f) Assume that, in addition to the physical page number, each page table entry (each page table line) also contains some status information in the form of a valid bit (*V*) and a dirty bit (*D*). How many bytes long is each page table entry? (Round up to an integer number of bytes.)

(g) Sketch (draw) the layout of the page table. What is the total size of the page table in bytes? [4]

(h) Assume we have TLB with 4 entries. Sketch the TLB by clearly labeling all fields and indicating the number of bits for each field. Assume TLB uses LRU replacement. [4]

(i) What is the total size of the TLB in bits?

Q3. a) Suppose we have a five stage (IF, ID, EX, MEM, WB) pipeline architecture with **no hazard unit.** Assume register write and read can be done at the same clock cycle. You are given the following MIPS code that executes on this architecture. Fill the given table by writing corresponding stages for each clock cycle. You must add NOP instructions if necessary.

MIPS code:

add $s0, $s0, $s1

add $s2, $s3, $s4

addi $t0, $t0, 2

sub $s0, $s0, $s2

mul $s5, $s2, $t0

div $s6, $s0, $t0

add $s6, $s5, $s6

Clock Cycle Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
add $s0, $s0, $s1	IF	ID	EX	ME	WB																	

b) Suppose that EX stage takes different number of clock cycles for different instructions as given in the following table. Fill the given table for this pipeline architecture by inserting necessary NOP instructions.

Instruction type	Number of EX clock cycles
add, sub	2
mul, div	4

<u>MIPS code:</u>

add \$s0, \$s0, \$s1

add \$s2, \$s3, \$s4

addi \$t0, \$t0, 2

sub \$s0, \$s0, \$s2

mul \$s5, \$s2, \$t0

div \$s6, \$s0, \$t0

add \$s6, \$s5, \$s6

Clock Cycle Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
add \$s0, \$s0, \$s1	IF	ID	EX1	EX2	ME	WB																

Name-Last Name: _____ **Student ID:** _____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructors: Assoc. Prof. Dr. Suleyman TOSUN
Final Exam	Assist. Prof. Dr. Mehmet KOSEOGLU
Duration: 120 minutes	Exam Date: 01.06.2017

Questions	1	2	3	4	5	Total
Marks	20	20	20	20	20	100
Earned						

Q1. a) Write the values of the registers and the stack for the following MIPS program. The value of the stack pointer is initially sp=0x7FFFFFFC.

Address		Instructions			
0x00400000		lui $s0, 0x1000		s0=	0x10000000
0x00400004		ori $s0, $s0, 0x0008		s0=	0x10000008
0x00400008		lw $a0, -4($s0)		a0=	10
0x0040000C		addi $a1, $s0, -8		a1=	0x10000000
0x00400010		lw $s1, 4($s0)		s1=	15
0x00400014		add $a2, $s1, $0		a2=	15
0x00400018		jal Proc1			
0x0040001C		addi $s2, $v0, $0		s2=	5
0x00400020					
0x00400024	Proc1:	addi $sp, $sp, -12		sp=	0x7FFFFFF0
0x00400028		sw $ra, 8($sp)		Write the stored values on stack!	
0x0040002C		sw $s0, 4($sp)			
0x00400030		sw $s1, 0($sp)			
0x00400034		addi $v0, $0, 0			
0x00400038	Loop:	beq $a0, $0, Done			
0x0040003C		lw $s0, 0($a1)			
0x00400040		slt $s1, $s0, $a2			
0x00400044		beq $s1, $0, Next			
0x00400048		addi $v0, $v0, 1			
0x0040004C	Next:	addi $a0, $a0, 4			
0x00400050		j Loop			
0x00400054	Done:	lw $ra, 8($sp)		ra=	0x0040001C
0x00400058		lw $s0, 4($sp)		s0=	0x10000008
0x0040005C		lw $s1, 0($sp)		s1=	15
0x00400060		addi $sp, $sp, 12		sp=	0x7FFFFFFC
0x00400064		jr $ra			

Address	Data
0x100000000	13
0x100000004	10
0x100000008	21
0x10000000C	15
0x100000010	7
0x100000014	16
0x100000018	11
0x10000001C	6
0x100000020	30
0x100000024	28

Address	Stack Data
0x7FFFFFFC	XXXXXXXX
0x7FFFFFF8	0x0040001C
0x7FFFFFF4	0x10000008
0x7FFFFFF0	15

b) Briefly describe what Proc1 function does.

It counts the number of values in the array that are less than 15.

Q2. Consider a virtual memory system that can address a total of 2^{32} bytes. You have only 8 MB of physical (main) memory. Assume that page size is 4 KB.

(a) How many bits is the physical address?

8MB = 2^{23}B => 23 bits

(b) What is the maximum number of virtual pages in the system?
$2^{32}/2^{12}=2^{20}$

(c) How many physical pages are in the system?
$2^{23}/2^{12}=2^{11}$

(d) How many bits are the virtual page numbers?
20

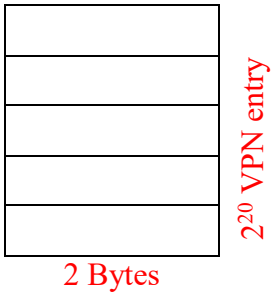
(e) How many bits are the physical page numbers?
11

(f) Assume that, in addition to the physical page number, each page table entry (each page table line) also contains some status information in the form of a valid bit (*V*) and a dirty bit (*D*). How many bytes long is each page table entry? (Round up to an integer number of bytes.)

D	V	PPN
1	1	11

Total of 13 bits. When we round up, we need 2 Bytes for each entry.

(g) Sketch (draw) the layout of the page table. What is the total size of the page table in bytes? [4]



Total size = 2.2^{20}Bytes=2MB

(h) Assume we have TLB with 4 entries. Sketch the TLB by clearly labeling all fields and indicating the number of bits for each field. Assume TLB uses LRU replacement. [4]

	Entry 3				Entry 2			Entry 1			Entry 0		
	U	V	VPN	PPN	V	VPN	PPN	V	VPN	PPN	V	VPN	PPN
Bits	2	1	20	11	1	20	11	1	20	11	1	20	11

(i) What is the total size of the TLB in bits?

130

Q3. a) Suppose we have a five stage (IF, ID, EX, MEM, WB) pipeline architecture with no hazard unit. Assume register write and read can be done at the same clock cycle. You are given the following MIPS code that executes on this architecture. Fill the given table by writing corresponding stages for each clock cycle. You must add NOP instructions if necessary.

MIPS code:

```
add $s0, $s0, $s1

add $s2, $s3, $s4

addi $t0, $t0, 2

sub $s0, $s0, $s2

mul $s5, $s2, $t0

div $s6, $s0, $t0

add $s6, $s5, $s6
```

Clock Cycle / Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
add $s0, $s0, $s1	IF	ID	EX	ME	WB																	
add $s2, $s3, $s4		IF	ID	EX	ME	WB																
addi $t0, $t0, 2			IF	ID	EX	ME	WB															
NOP				IF	ID	EX	ME	WB														
sub $s0, $s0, $s2					IF	ID	EX	ME	WB													
mul $s5, $s2, $t0						IF	ID	EX	ME	WB												
NOP							IF	ID	EX	ME	WB											
div $s6, $s0, $t0								IF	ID	EX	ME	WB										
NOP									IF	ID	EX	ME	WB									
NOP										IF	ID	EX	ME	WB								
add $s6, $s5, $s6											IF	ID	EX	ME	WB							

b) Suppose that EX stage takes different number of clock cycles for different instructions as given in the following table. Fill the given table for this pipeline architecture by inserting necessary NOP instructions.

Instruction type	Number of EX clock cycles
add, sub	2
mul, div	4

MIPS code:

add $s0, $s0, $s1

add $s2, $s3, $s4

addi $t0, $t0, 2

sub $s0, $s0, $s2

mul $s5, $s2, $t0

div $s6, $s0, $t0

add $s6, $s5, $s6

Clock Cycle / Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
add $s0, $s0, $s1	IF	ID	EX1	EX2	ME	WB																
add $s2, $s3, $s4		IF	ID	EX1	EX2	ME	WB															
addi $t0, $t0, 2			IF	ID	EX1	EX2	ME	WB														
NOP				IF	ID	EX	ME	WB														
NOP					IF	ID	EX	ME	WB													
sub $s0, $s0, $s2						IF	ID	EX1	EX2	ME	WB											
mul $s5, $s2, $t0							IF	ID	EX1	EX2	EX3	EX4	ME	WB								
NOP								IF	ID	EX	ME	WB										
NOP									IF	ID	EX	ME	WB									
div $s6, $s0, $t0										IF	ID	EX1	EX2	EX3	EX4	ME	WB					
NOP											IF	ID	EX	ME	WB							
NOP												IF	ID	EX	ME	WB						
NOP													IF	ID	EX	ME	WB					
NOP														IF	ID	EX	ME	WB				
NOP															IF	ID	EX	ME	WB			
add $s6, $s5, $s6																IF	ID	EX1	EX2	ME	WB	

Name-Last Name: _____ **Student ID:** _____

Section (Check one)

☐ **Section 1 (Wednesday)**

☐ **Section 2 (Friday)**

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructor: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam	Exam Date: 15.04.2016
Duration: 100 minutes	

Questions	**1**	**2**	**3**	**4**	**5**	**Total**
Marks	**20**	**20**	**20**	**20**	**20**	**100**
Earned						

Q1. Suppose we have an array A with 100 elements. The memory address of the first array element is 0xA1B2C3D4. **Write a MIPS function named *ordered*, which checks if the elements of the array A are in ascending order.** If they are, the function should return 1. Otherwise, it returns 0. Call the function *ordered* from the main function. And then, store the return value in register s0.
Note that you do not have to store the variables in the stack when function is called.

Important note: You are allowed to use only the instruction we have seen in our lectures. You cannot use any other pseudo instructions. If you use any, you will lose 5 points for each pseudo instruction.

Q2. You are given a MIPS program below.

a) Fill the given table for this program by entering the following:

- How many times does each instruction execute (fetched) in the program?
- What is the clock cycles of each instruction when executes on multi-cycle processor? Diagram of the multi-cycle processor is given in Question 5 if you need it
- What are the total clock cycles for each instruction and the whole program when executed on multi-cycle processor?

	Instruction	**How many times does an instruction execute?**	**Clock cycles of the instruction for multi-cycle processor**	**Total clock cycles for multi-cycle processors**
	addi $s0, $0, 2			
loop:	beq $s0, $0, done			
	srl $s0, $s0, 1			
	bne $s0, $0, else			
	lw $s5, 0($a0)			
	add $s5, $s5, $s0			
	sw $s5, 0($a0)			
else:	addi $s0, $s0,-1			
	j loop			
done:	jr $ra			
Total (Single-Cycle):			**Total (Multi-Cycle):**	

b) Show the formats of the instructions beq (opcode=4) and srl (funct=2). Then, write machine codes for these instructions in hexadecimal format. (s0=16)

Instruction	Hexadecimal Machine Code
beq $s0, $0, done	
srl $s0, $s0, 1	

Q3. We would like to add **blez** instruction to single cycle MIPS processor. blez instruction copies the branch target address (BTA, which is *PCBranch* in Figure 2) to the program counter (PC) if the value in register [rs] is less than or equal to zero. In other words, if [rs] ≤ 0, PC = BTA.

a) First, add an output signal to the ALU (given in Figure 1) and name this signal as LTEZ (Less Than Equal to Zero). LTEZ=1 when the Result is less than or equal to zero (i.e., Result ≤ 0). Otherwise, LTEZ=0. [5]

b) By using LTEZ output signal, show the necessary changes on data-path of single-cycle processors given in Figure 2 and explain your changes. [10]

c) Fill the control signals in Table I. [5]

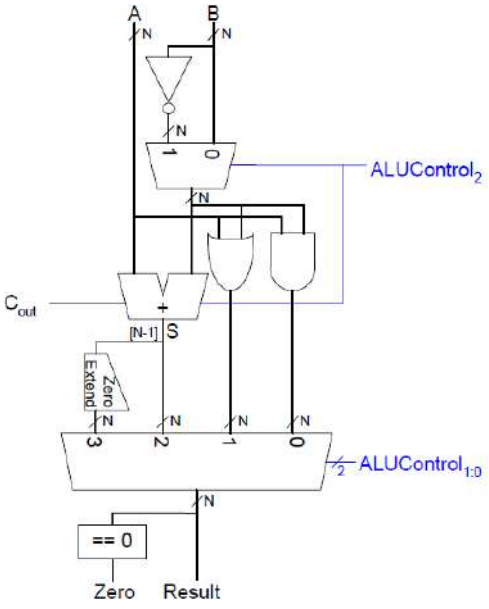


Figure 1: ALU

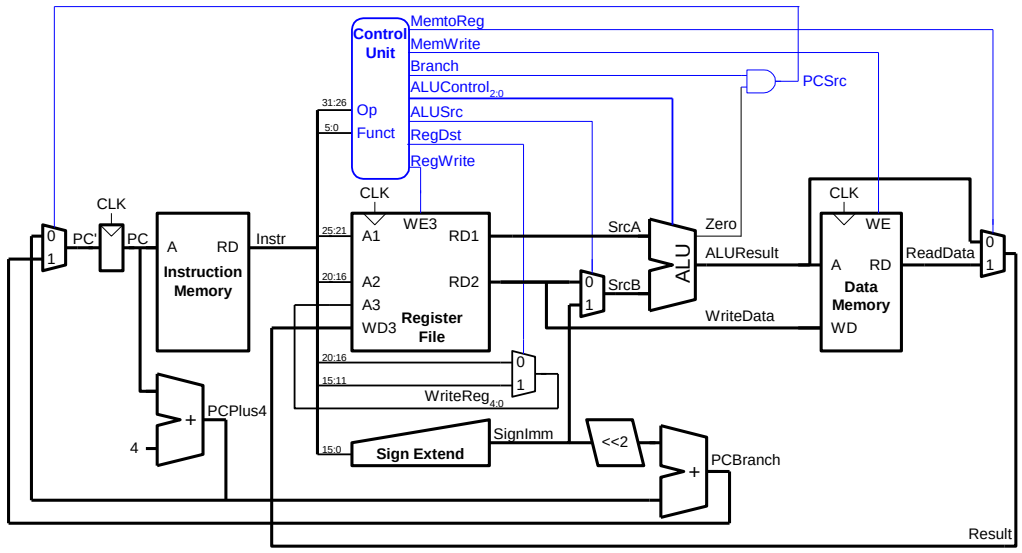


Figure 2: Single Cycle MIPS Processor

c) Tabel I: Control signals for **blez** instruction

Inst.	$Op_{31:26}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$
blez	000110							

Q4. **a)** In a MIPS architecture, indicate if the following instructions use sign extend logic or ALU. If an instruction uses ALU, what is the type of operation (add, subtract, and, or, …etc)? [10]

Instruction	Uses sign extend (YES or NO)	Uses ALU (YES or NO)	Type of ALU operation
lw $s0, 0($a0)			
sllv $s0, $s1, $s2			
sw $s0, 0($a0)			
jal target			
bne $s0, $s1, target			

b) Suppose following delays have been determined for the elements of the single-cycle processor. You can ignore the delays of other elements in the design.

Element	Parameter	Delay (ps)
Register clock-to-Q	t_{pcq_PC}	30
Register setup	t_{setup}	20
Multiplexer	t_{mux}	25
ALU	t_{ALU}	200
Memory read	t_{mem}	250
Register file read	t_{RFread}	150
Register file setup	$t_{RFsetup}$	20

b1) What is the worst case delay for R-type instructions? Show how you determine this value. [5]

b2) What is the worst case delay for beq instruction? Show how you determine this value. [5]

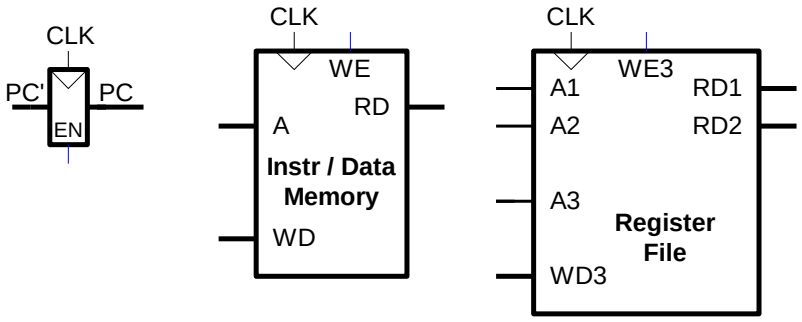
Q5. In the multi-cycle processor given in Figure 3, lw instruction takes 5 clock cycles.

1. Fetch: Fetch the instruction from instruction memory and store it into the instruction register.
2. Decode: Read the operands such as registers and immediate values.
3. Execute: Add operands (add rs and sign extended immediate value.)
4. Read memory: Read the data from memory and store it into data register.
5. Write result: Write the data from data register to destination register in register file.

Your goal is to ***design a microarchitecture for lw instruction that takes only 3 clock cycles***. You should merge cycles Fetch and Decode into FetchDec and merge Execute and Read memory into (ExecMem). Write result cycle will be the same.

a) Draw the data path using the 32-bit Program Counter (PC), instruction/data memory, register file, and additional hardware if necessary. You should clearly show the extra hardware and the bit numbers of each connection. Note that PC must be incremented in the first cycle. [10]

b) Draw the FSM (Finite State Machine) for new lw instruction. [10]



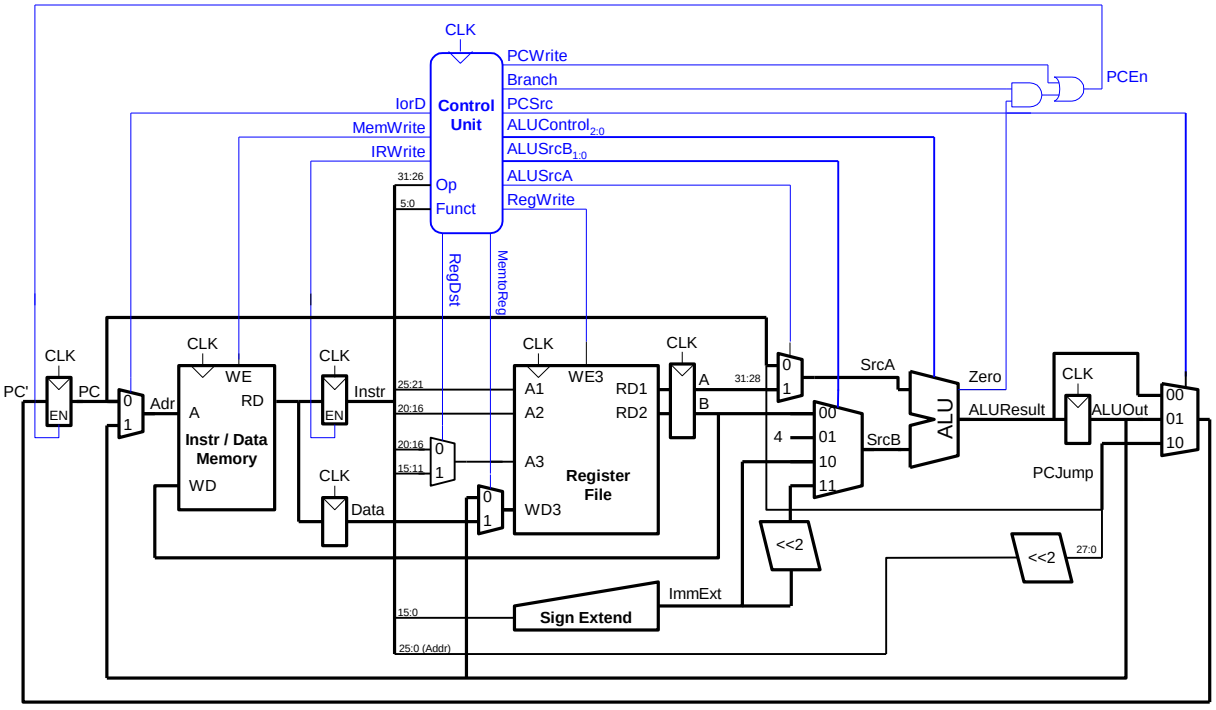


Figure 3:Multi cycle processor

Name-Last Name: _____ **Student ID:** _____

Section (Check one)

☐ **Section 1 (Wednesday)**

☐ **Section 2 (Friday)**

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructor: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam	Exam Date: 15.04.2016
Duration: 100 minutes	

Questions	**1**	**2**	**3**	**4**	**5**	**Total**
Marks	**20**	**20**	**20**	**20**	**20**	**100**
Earned						

Q1. Suppose we have an array A with 100 elements. The memory address of the first array element is 0xA1B2C3D4. **Write a MIPS function named *ordered*, which checks if the elements of the array A are in ascending order.** If they are, the function should return 1. Otherwise, it returns 0. Call the function *ordered* from the main function. And then, store the return value in register s0.
Note that you do not have to store the variables in the stack when function is called.

Important note: You are allowed to use only the instruction we have seen in our lectures. You cannot use any other pseudo instructions. If you use any, you will lose 5 points for each pseudo instruction.

```
Main:
            lui $a0, 0xA1B2
            ori $a0, $a0, 0xC3D4
            jal ordered
            add $s0, $v0, $0

ordered:    addi $t0, $0, 99      #t0=array_size-1 (we have 99 comparisons)
            addi $t1, $0, 0       #i=0
            addi $v0, $0, 1       #v0=1 means array is ordered.
      for:  beq $t0, $t1, done    #loop 99 times (99 comparisons)
            lw $t2, 0($a0)        #load the first element to t2
            lw $t3, 4($a0)        #load the next element to t3
            slt $t4, $t3, $t2     #t4=1 if t3<t2
            beq $t4, $0, else     #if t4=0 (means that t3 is not less than t2), go to next comparison
            addi $v0, $0, 0       #v0=0 (array is not ordered) since t3<t2
            jr $ra                #we do not need to check the array further. Return to main function.
      else: addi $t1, $t1, 1      #i=i+1
            addi $a0, $a0, 4      #point next array element
            j ordered             #go to loop to compare next two numbers
      done: jr $ra                # Go to main.  (If we reach this line, that means array is ordered. )
```

Q2. You are given a MIPS program below.

a) Fill the given table for this program by entering the following:
- How many times does each instruction execute (fetched) in the program?
- What is the clock cycles of each instruction when executes on multi-cycle processor? Diagram of the multi-cycle processor is given in Question 5 if you need it
- What are the total clock cycles for each instruction and the whole program when executed on multi-cycle processor?

	Instruction	**How many times does an instruction execute?**	**Clock cycles of the instruction for multi-cycle processor**	**Total clock cycles for multi-cycle processors**
	addi $s0, $0, 2	1	4	4
loop:	beq $s0, $0, done	2	3	6
	srl $s0, $s0, 1	1	4	4
	bne $s0, $0, else	1	3	3
	lw $s5, 0($a0)	0	5	0
	add $s5, $s5, $s0	0	4	0
	sw $s5, 0($a0)	0	4	0
else:	addi $s0, $s0,-1	1	4	4
	j loop	1	3	3
done:	jr $ra	1	3	3
Total (Single-Cycle):		**8**	**Total (Multi-Cycle):**	27

b) Show the formats of the instructions beq (opcode=4) and srl (funct=2). Then, write machine codes for these instructions in hexadecimal format. (s0=16)

beq $16, $0, done (It is an I-type instruction)

opcode	rs	rt	immediate
000100	00**000**	**1**0000	**0000** 0000 **0000** 0111
1 0	**1**	0	**0** 0 **0** 7

When we determine the value of immediate field, we count the number of instruction between the srl (next instruction after beq) and the instrcution at the done line (jr $ra). It is 7.

srl $16, $16, 1 (It is an R-type instruction.)

opcode	rs	rt	rd	shamt	funct
000000	00**000**	**1**0000	**1000**0	000**01**	**00**0010
0 0	**1**	0	**8** 0	**4**	2

Instruction	Hexadecimal Machine Code
beq $s0, $0, done	0x1010_0007
srl $s0, $s0, 1	0x0010_8042

Q3. We would like to add **blez** instruction to single cycle MIPS processor. blez instruction copies the branch target address (BTA, which is *PCBranch* in Figure 2) to the program counter (PC) if the value in register [rs] is less than or equal to zero. In other words, if [rs] ≤ 0, PC = BTA.

a) First, add an output signal to the ALU (given in Figure 1) and name this signal as LTEZ (Less Than Equal to Zero). LTEZ=1 when the Result is less than or equal to zero (i.e., Result ≤ 0). Otherwise, LTEZ=0. [5]

b) By using LTEZ output signal, show the necessary changes on data-path of single-cycle processors given in Figure 2 and explain your changes. [10]

c) Fill the control signals in Table I. [5]

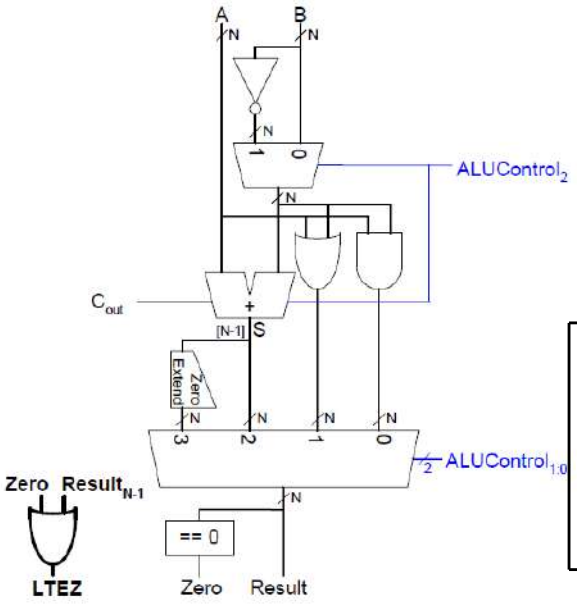


Figure 1: ALU

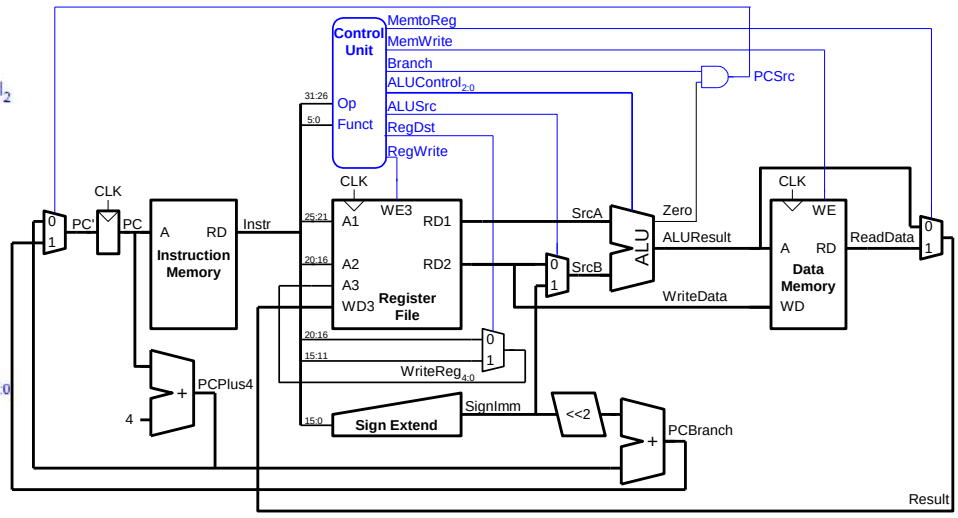
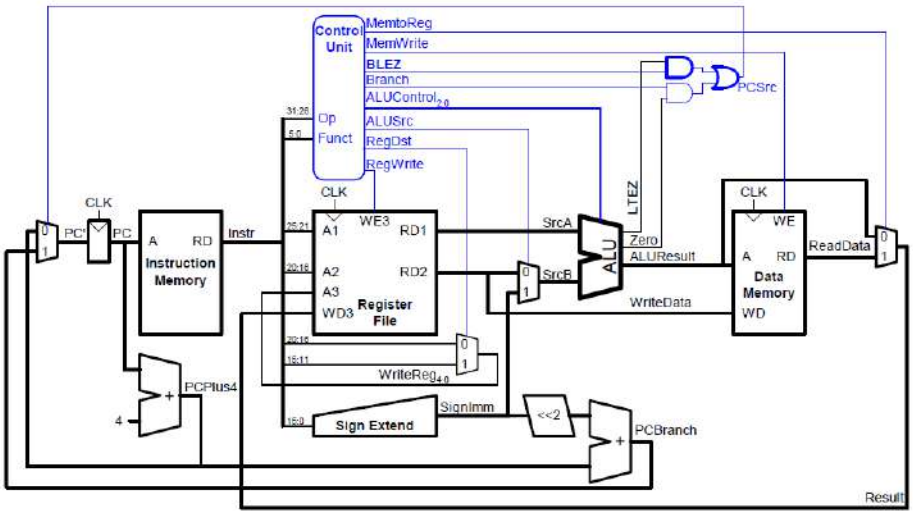


Figure 2: Single Cycle MIPS Processor

If $Result_{N-1}$ is 1, that means the result is negative and it is less than zero. If the Result is zero, the *zero* output will be 1. If we OR these two signals, we get LTEZ.



We add BLEZ control signal to the controller. If BLEZ is 1 and LTEZ is 1, the PCSrc will be 1 and the BTA will be stored in PC. The contol for beq does not change and we can OR them together.
There is no other changes to the data path.

c) Tabel I: Control signals for **blez** instruction

Inst.	$Op_{31:26}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$	BLEZ
blez	000110	0	X	0	0	0	X	01	1

Q4. **a)** In a MIPS architecture, indicate if the following instructions use sign extend logic or ALU. If an instruction uses ALU, what is the type of operation (add, subtract, and, or, …etc)? [10]

Instruction	Uses sign extend (YES or NO)	Uses ALU (YES or NO)	Type of ALU operation
lw $s0, 0($a0)	YES	YES	ADD
sllv $s0, $s1, $s2	NO	YES	SHIFT
sw $s0, 0($a0)	YES	YES	ADD
jal target	NO	NO	---
bne $s0, $s1, target	YES	YES	SUBTRACT

b) Suppose following delays have been determined for the elements of the single-cycle processor. You can ignore the delays of other elements in the design.

Element	Parameter	Delay (ps)
Register clock-to-Q	t_{pcq_PC}	30
Register setup	t_{setup}	20
Multiplexer	t_{mux}	25
ALU	t_{ALU}	200
Memory read	t_{mem}	250
Register file read	t_{RFread}	150
Register file setup	$t_{RFsetup}$	20

b1) What is the worst case delay for R-type instructions? Show how you determine this value. [5]

$T_{R\text{-}Type} = t_{pcq\_PC} + t_{mem} + t_{RFread} + t_{mux} + t_{ALU} + t_{mux} + t_{RFsetup}$ =(30+250+150+25+200+25+20)ps=700ps

After reading the instrcution, register values are read and then the ALU operation takes place. Then, the result is written to regster file. The last mux is the mux in front of PC and T_setup is for writing to PC.

b2) What is the worst case delay for beq instruction? Show how you determine this value. [5]

$T_{beq} = t_{pcq\_PC} + t_{mem} + t_{RFread} + t_{mux} + t_{ALU} + t_{mux} + t_{setup}$ =(30+250+150+25+200+25+20)ps=700ps

After reading the instrcution from instruction memory, tow register values are read and they are subtracted. If the result is zero, then the BTA address is written to the PC. The last mux is the mux in front of PC and T_setup is for writing to PC. AND operation is ignored.

Q5.

Q5. In the multi-cycle processor given in Figure 3, lw instruction takes 5 clock cycles.

1. Fetch: Fetch the instruction from instruction memory and store it into the instruction register.
2. Decode: Read the operands such as registers and immediate values.
3. Execute: Add operands (add rs and sign extended immediate value.)
4. Read memory: Read the data from memory and store it into data register.
5. Write result: Write the data from data register to destination register in register file.

Your goal is to ***design a microarchitecture for lw instruction that takes only 3 clock cycles***. You should merge cycles Fetch and Decode into FetchDec and merge Execute and Read memory into (ExecMem). Write result cycle will be the same.

a) Draw the data path using the 32-bit Program Counter (PC), instruction/data memory, register file, and additional hardware if necessary. You should clearly show the extra hardware and the bit numbers of each connection. Note that PC must be incremented in the first cycle. [10]

b) Draw the FSM (Finite State Machine) for new lw instruction. [10]

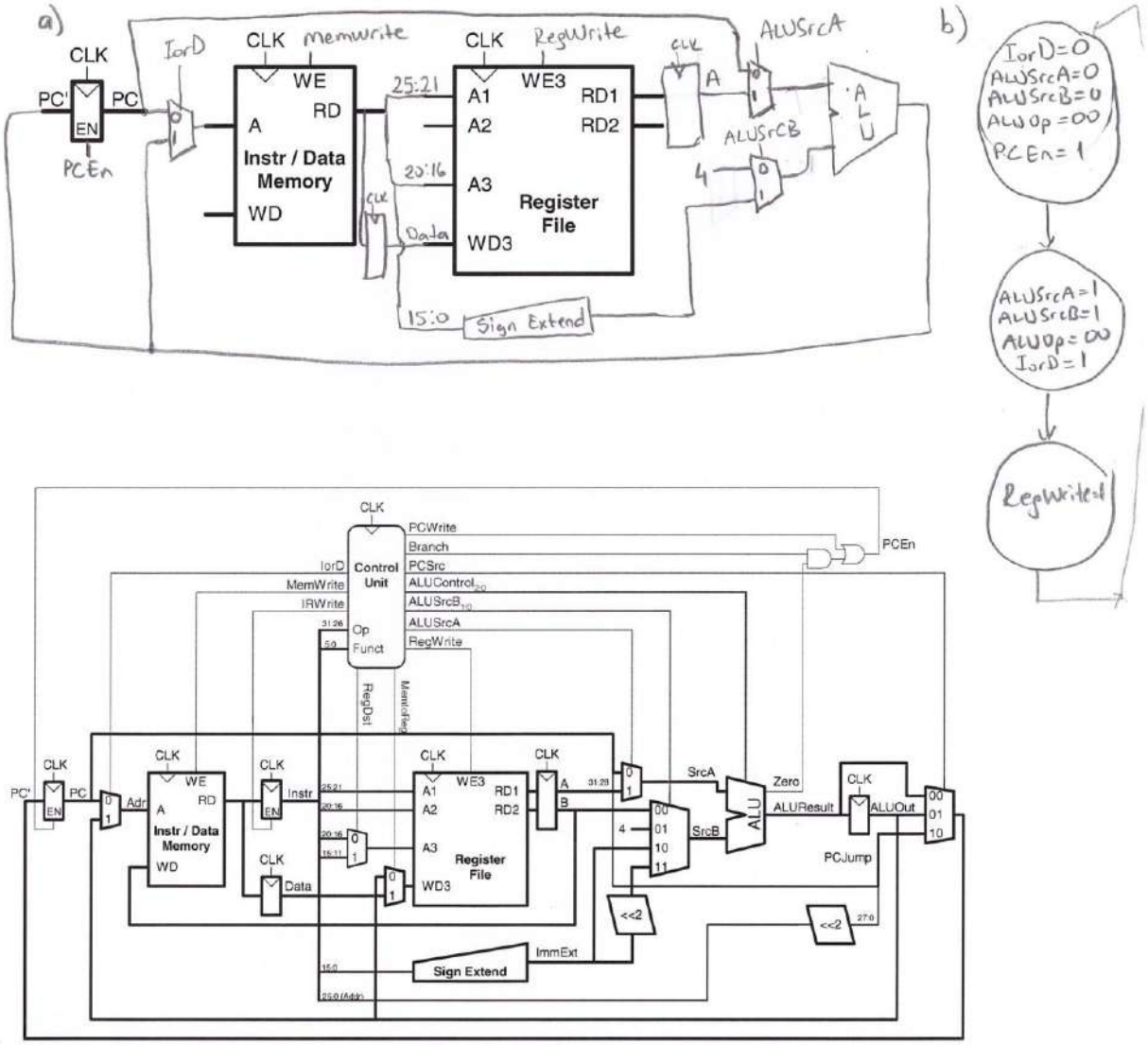


Figure 3:Multi cycle processor

Name-Last Name: _____ **Student ID:** _____

Section No (1,2,3?): ____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructors: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam	Assist. Prof. Dr. Mehmet KOSEOGLU
Duration: 120 minutes	Exam Date: 18.04.2017

Questions	**1**	**2**	**3**	**4**	**5**	**6**	**Total**
Marks	**20**	**20**	**15**	**10**	**20**	**15**	**100**
Earned							

Q1. a) Write the values of the registers after the following MIPS program finishes its execution.

```
        lui $s0, 0x1234
        ori $s0, $s0, 0x0335
        andi $s0, $s0, 0x000F
        sra $s1, $s0, 2
        or $s2, $s0, $s1
        slt $s3, $s1, $s2
        bne  $s1, $s3, else
        addi $s2, $s2, -1
else:   sll $s4, $s2, 2
        jr $ra
```

s0	**s1**	**s2**	**s3**	**s4**

b) For the given "*number*" value, what does function f1 do? Write output values (value in s0) for the given *number* values in the table.

```
main:   addi $a0, $0, number
        addi $sp, $sp, -4
        sw $ra, 0($sp)
        jal f1
        add $s0, $v0, $0
        lw $ra, 0($sp)
        addi $sp, $sp, 4
        jr $ra    #exit

f1:     addi $t0, $0, 0
        addi $v0, $0, 1
        bne $a0, $0, else
        jr $ra
else:   beq $a0, $t0, done
        addi $t0, $t0, 1
        mul $v0, $v0, $t0
        mflo $v0
        j else
done:   jr $ra
```

Write the description of f1 below:

Number	0	3	5
S0			

Q2. You have four instructions stored in the memory as given in the following table:

Instructions	Address	Instruction
Inst1	0x00400000	0x14100003
Inst2	0x00400004	0x012A4025
Inst3	0x00400008	0x2210FFFB
Inst4	0x0040000C	0x08100000
Inst5	0x00400010	---

a) Write the binary values for each instruction. Clearly show which bits corresponds to which field in the instruction format (opcode, rs, rt, rd.. etc?).

Instructions	Instruction format
0x14100003	
0x012A4025	
0x2210FFFB	
0x08100000	

b) Write down the corresponding MIPS assembly code below for each machine code.

Instructions	MIPS Code
Inst1	
Inst2	
Inst3	
Inst4	
Inst5	

Name	Register
$0	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

Instruction	Opcode
j	000010
jal	000011
beq	000100
bne	000101
addi	001000
slti	001010
andi	001100
ori	001101
xori	001110
lui	001111
lw	100011
sw	101011

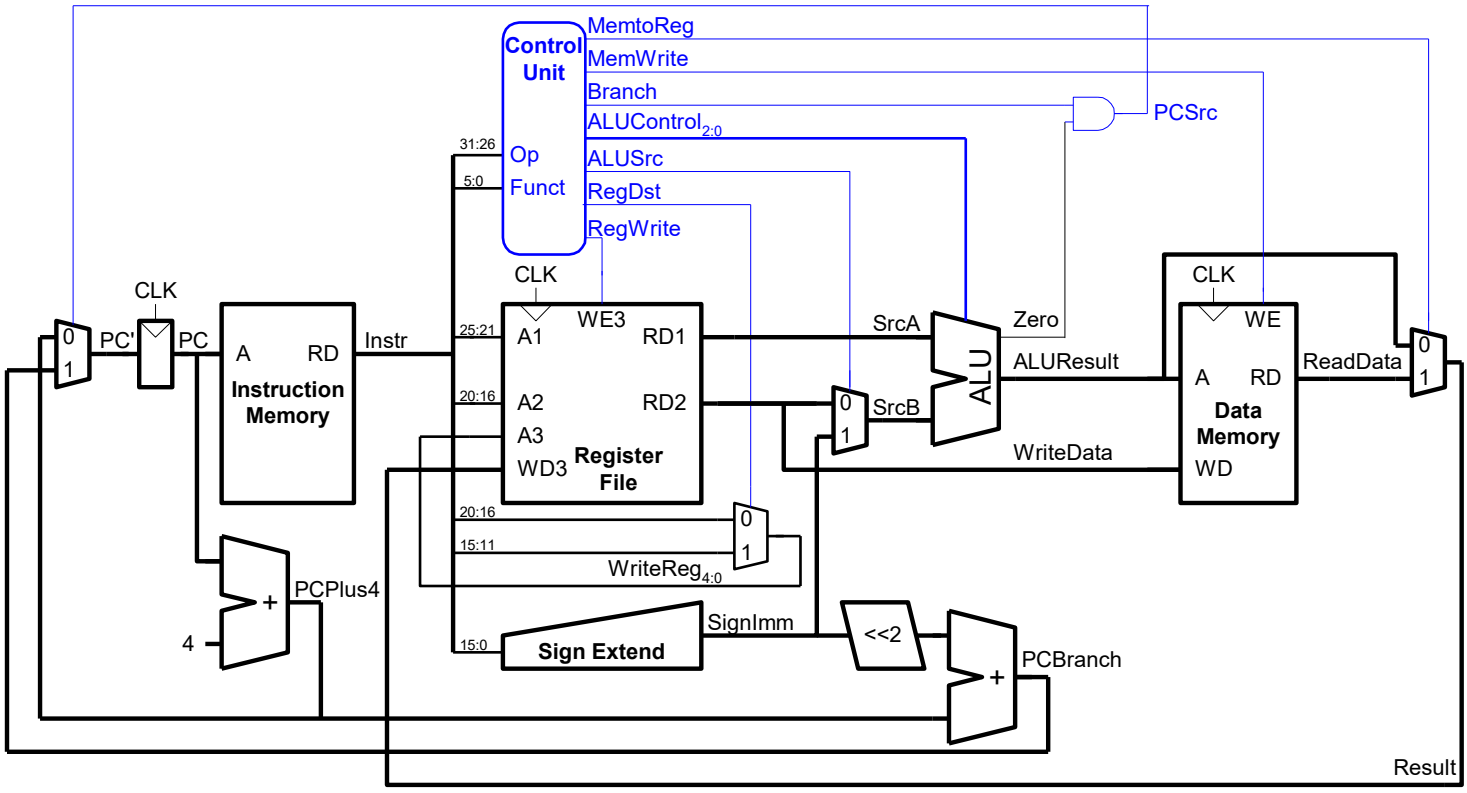
Instruction	Funct
sll	000000
srl	000010
sra	000011
jr	001000
div	011010
add	100000
sub	100010
and	100100
or	100101
xor	100110
nor	100111
slt	101011

Q3. MIPS architesture has some conditional branches and unconditional jumps. We list some of them below. For each instruction type, write the maximum number of instructions between the current program counter (PC) and the target instruction. You should also write the instruction type.

Instruction	Maximum number of instructions that we can jump over	Instruction type
J		
JR		
JAL		
BEQ		
BNE		

Q4. We would like to add R-type lw instruction (**lwr $rd, $rt($rs)**) to single cycle MIPS processor. The lwr instruction is similar to lw but it sums two registers (specified by $rs, $rt) to obtain the effective load address and uses the R-Type format. Loaded word from memory is written to register rd.

a) Show the necessary changes on data-path of single-cycle processors if any.
b) Fill the control signals in Table I.



Tabel I: Control signals for **lwr** instruction

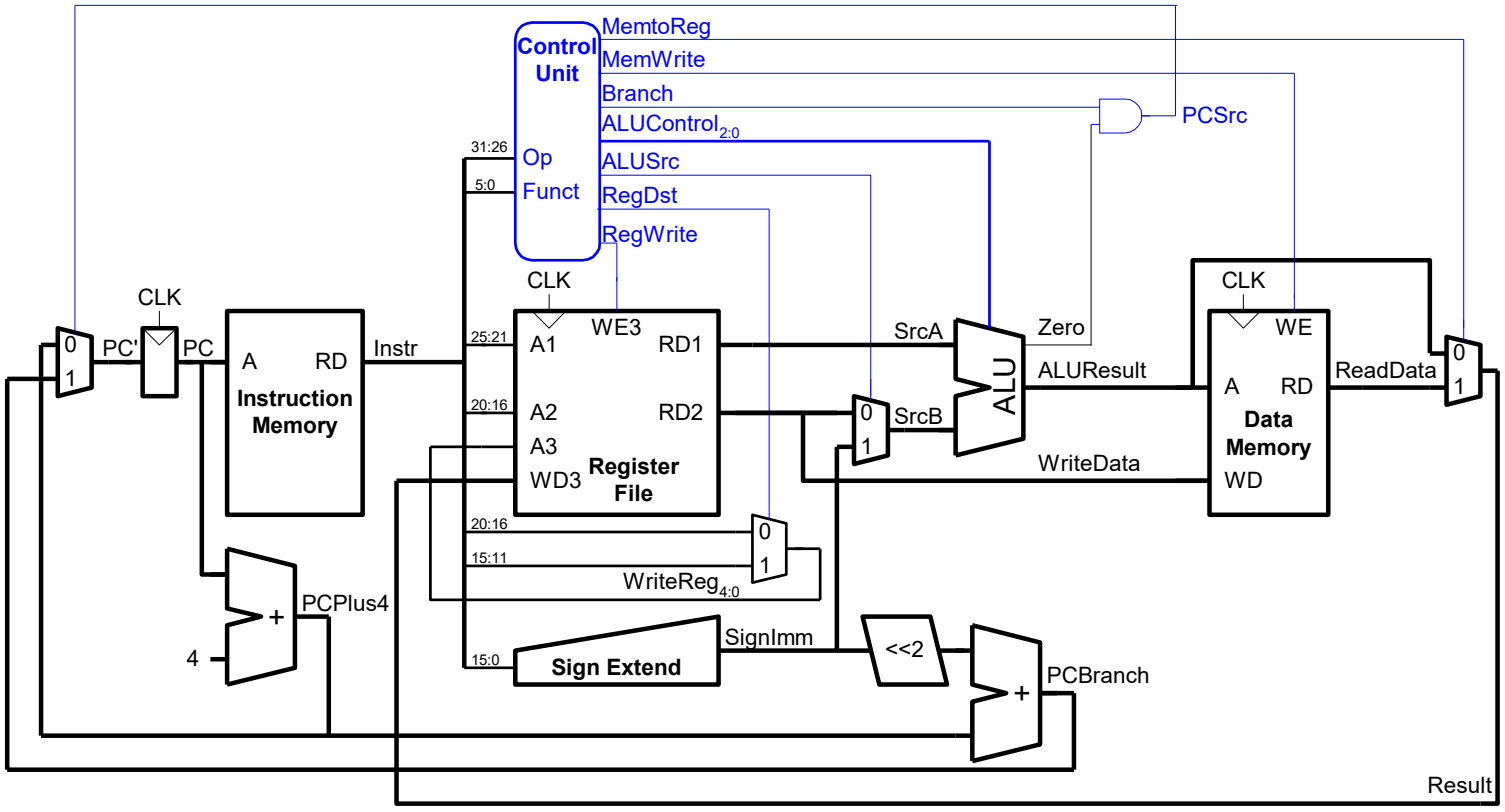
Inst.	Op31:26	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	ALUOp1:0
lwr	XXXXX							

ALUOp1:0	Operation
00	Addition
01	Subtraction
10	Look at funct.
11	Not used

Q5. We would like to also add R-type sw instruction (**swr $rd, $rt($rs)**) to single cycle MIPS processor. The swr instruction is similar to sw but it sums two registers (specified by $rs, $rt) to obtain the effective load address and uses the R-Type format. Then, it writes the word in rd register to the memory address.

a) Show the necessary changes on data-path of single-cycle processors if any. Be aware that register file has only two read ports! Briefly explain your changes.

b) Fill the control signals in Table II.



Tabel II: Control signals for **swr** instruction

Inst.	Op31:26	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	ALUOp1:0
swr	XXXXX							

Q6. For the multi-cycle processor given in Figure 3, how many clock cycles does it take to execute following MIPS program? Write how many times each instruction is fetched, clock cycle for each execution and total cycles for each instruction in the table.

	How many times is it fetched?	Number of clock cycles for each execution	Total execution times for each instruction
main: addi $a0, $0, ***3***			
addi $sp, $sp, -4			
sw $ra, 0($sp)			
jal f1			
add $s0, $v0, $0			
lw $ra, 0($sp)			
addi $sp, $sp, 4			
jr $ra			
f1: addi $t0, $0, 0			
addi $v0, $0, 1			
bne $a0, $0, else			
jr $ra			
else: beq $a0, $t0, done			
addi $t0, $t0, 1			
mult $v0, $v0, $t0 #R-type			
mflo $v0 #R-type			
j else			
done:jr $ra			
Total:			

What is the CPI for this program?

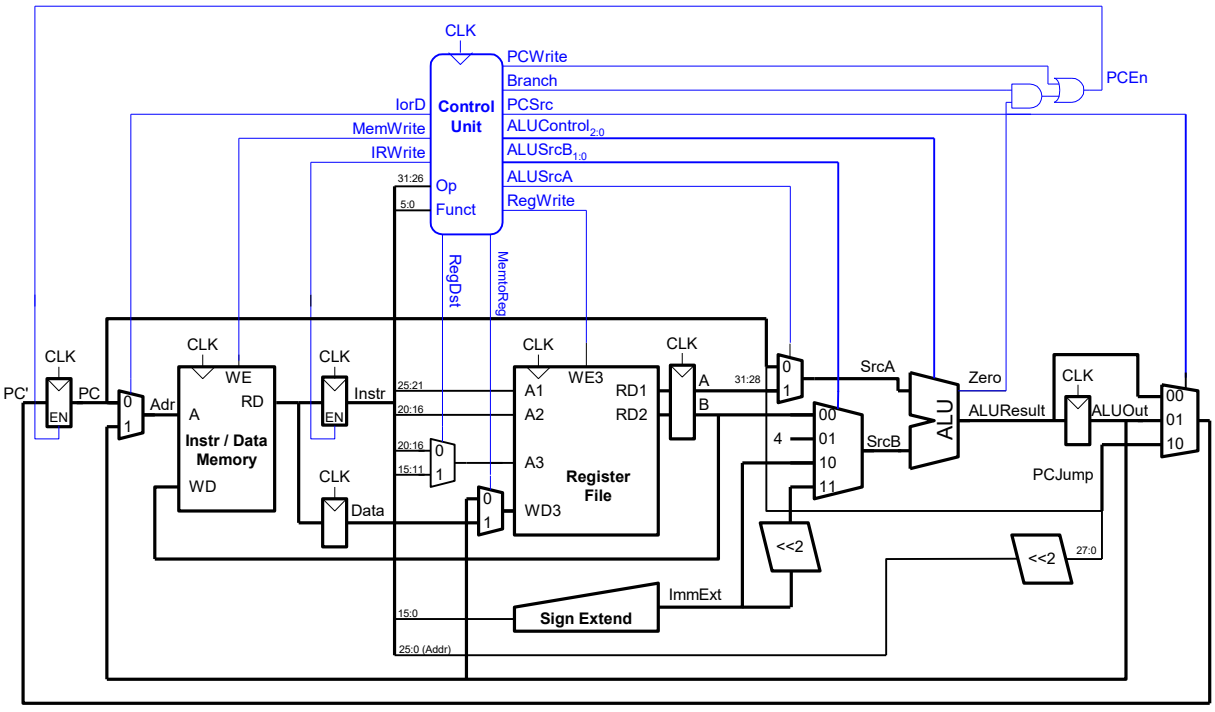


Figure 3:Multi cycle processor

Name-Last Name: _____ **Student ID:** _____

Section No (1,2,3?): ____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructors: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam	Assist. Prof. Dr. Mehmet KOSEOGLU
Duration: 120 minutes	Exam Date: 18.04.2017

Questions	1	2	3	4	4	5	Total
Marks	20	20	15	10	20	15	100
Earned							

Q1. a) Write the values of the registers after the following MIPS program finishes its execution.

		Register (1 point each)
	lui $s0, 0x1234	s0=0x12340000
	ori $s0, $s0, 0x0335	s0=0x12340335
	andi $s0, $s0, 0x000F	s0=5
	sra $s1, $s0, 2	s1=1
	or $s2, $s0, $s1	s2=5
	slt $s3, $s1, $s2	s3=1
	bne $s1, $s3, else	s1=1
	addi $s2, $s2, -1	s2=4
else:	sll $s4, $s2, 2	s4=16
	jr $ra	

b) For the given "*number*" value, what does function f1 do? Write output values (value in s0) for the given *number* values in the table.

```
main: addi $a0, $0, number
      addi $sp, $sp, -4
      sw $ra, 0($sp)
      jal f1
      add $s0, $v0, $0
      lw $ra, 0($sp)
      addi $sp, $sp, 4
      jr $ra   #exit

f1:   addi $t0, $0, 0
      addi $v0, $0, 1
      bne $a0, $0, else
      jr $ra
else: beq $a0, $t0, done
      addi $t0, $t0, 1
      mul $v0, $v0, $t0
      mflo $v0
      j else
done: jr $ra
```

Write the description of f1 below:

F1 calculates the factorial of given number. [5points]

Number	0	3	5
S0 [2 points each]	1	6	120

Q2. You have four instructions stored in the memory as given in the following table:

Instructions	Address	Instruction
Inst1	0x00400000	0x14100003
Inst2	0x00400004	0x012A4025
Inst3	0x00400008	0x2210FFFB
Inst4	0x0040000C	0x08100000
Inst5	0x00400010	---

a) Write the binary values for each instruction. Clearly show which bits corresponds to which field in the instruction format (opcode, rs, rt, rd.. etc?).

Instructions	Instruction format
0x14100003	0001 0100 0001 0000 0000 0000 0000 0011
0x012A4025	0000 0001 0010 1010 0100 0000 0010 0101
0x2210FFFB	0010 0010 0001 0000 1111 1111 1111 1011
0x08100000	0000 1000 0001 0000 0000 0000 0000 0000

b) Write down the corresponding MIPS assembly code below for each machine code.

Instructions	MIPS Code [5 points(3+2) each]
Inst1	Label: bne $s0, $0, Done
Inst2	or $t0, $t1, $t2
Inst3	addi $s0, $s0, -5
Inst4	j Label
Inst5	Done:

Name	Register
$0	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

Instruction	Opcode
j	000010
jal	000011
beq	000100
bne	000101
addi	001000
slti	001010
andi	001100
ori	001101
xori	001110
lui	001111
lw	100011
sw	101011

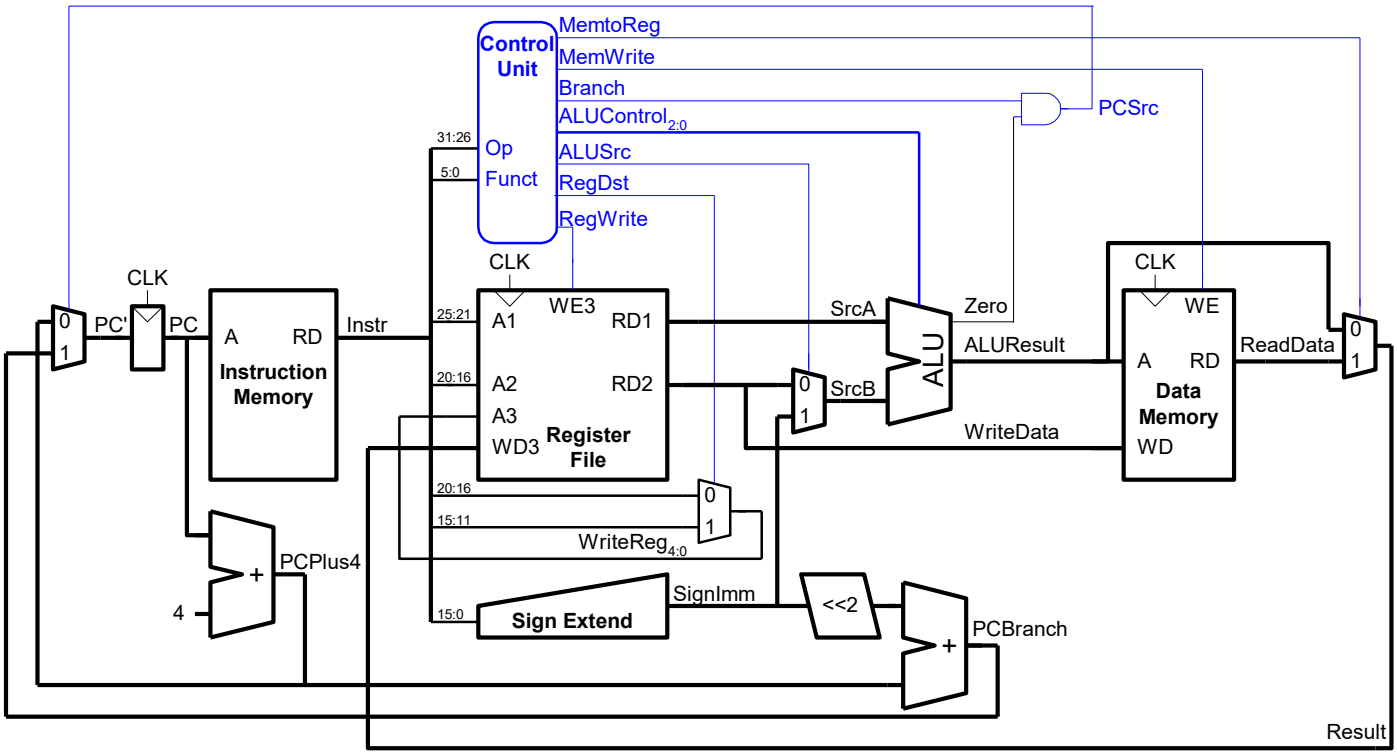
Instruction	Funct
sll	000000
srl	000010
sra	000011
jr	001000
div	011010
add	100000
sub	100010
and	100100
or	100101
xor	100110
nor	100111
slt	101011

Q3. MIPS architesture has some conditional branches and unconditional jumps. We list some of them below. For each instruction type, write the maximum number of instructions between the current program counter (PC) and the target instruction. You should also write the instruction type.

Instruction	Maximum number of instructions that we can jump over (2 points each)	Instruction type (1 point each)
J	2^{26}	J
JR	2^{30}	R
JAL	2^{26}	J
BEQ	$2^{15}+1$	I
BNE	$2^{15}+1$	I

Q4. We would like to add R-type lw instruction (**lwr $rd, $rt($rs)**) to single cycle MIPS processor. The lwr instruction is similar to lw but it sums two registers (specified by $rs, $rt) to obtain the effective load address and uses the R-Type format. Loaded word from memory is written to register rd.

a) Show the necessary changes on data-path of single-cycle processors if any.
b) Fill the control signals in Table I.



There is no need to change data-path.

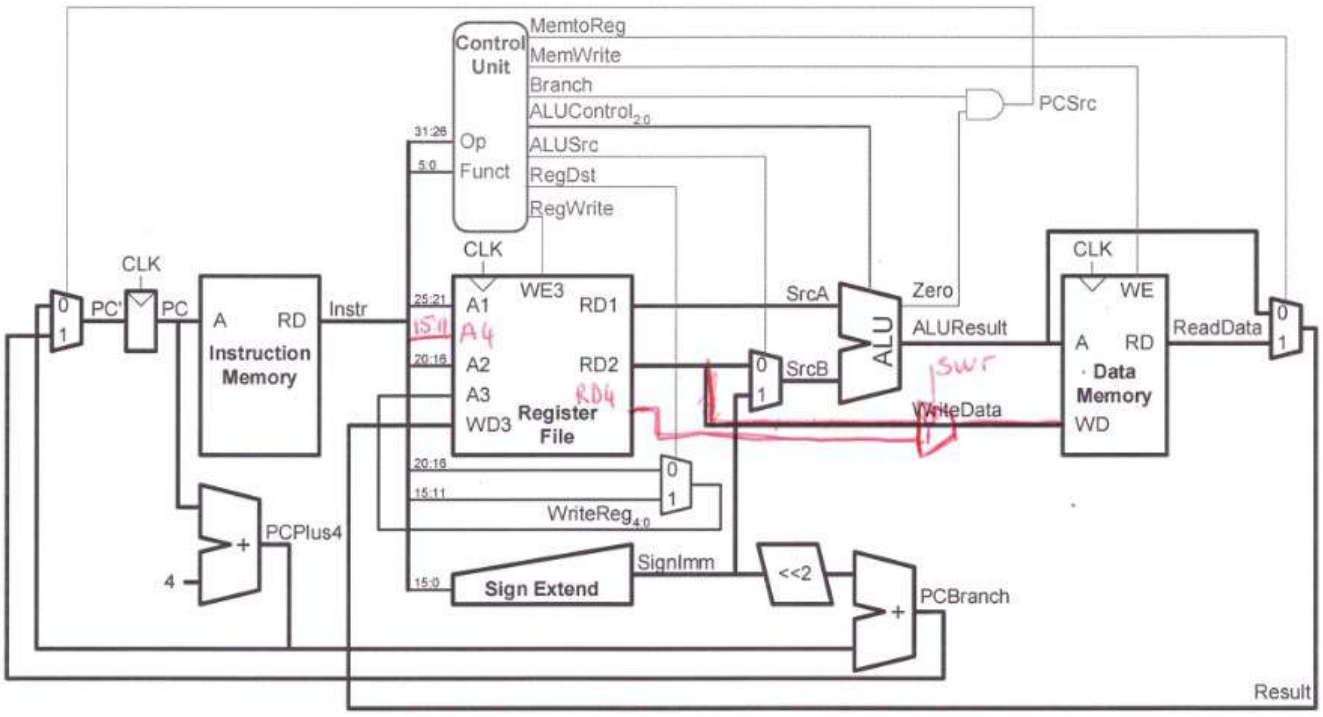
Tabel I: Control signals for **lwr** instruction

Inst.	$Op_{31:26}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$
lwr	XXXXX	1	1	0	0	0	1	00

Q5. We would like to also add R-type sw instruction (**swr $rd, $rt($rs)**) to single cycle MIPS processor. The swr instruction is similar to sw but it sums two registers (specified by $rs, $rt) to obtain the effective load address and uses the R-Type format. Then, it writes the word in rd register to the memory address.

a) Show the necessary changes on data-path of single-cycle processors if any. Be aware that register file has only two read ports!

b) Fill the control signals in Table II.



We add another read port to be able to read rd register. We add multiplexer in front of WD port of data memory and we select rd by setting srw select line of multiplexer. Control signals are below.

Tabel II: Control signals for **swr** instruction

Inst.	$Op_{31:26}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$	swr
swr	XXXX	0	X	0	0	1	X	10 (or 00)	1

Q6. For the multi-cycle processor given in Figure 3, how many clock cycles does it take to execute following MIPS program? Write how many times each instruction is fetched, clock cycle for each execution and total cycles for each instruction in the table.

	How many times is it fetched? [5 points]	Number of clock cycles for each execution [5 points]	Total execution times for each instruction [2 points]
main: addi $a0, $0, ***3***	1	4	4
addi $sp, $sp, -4	1	4	4
sw $ra, 0($sp)	1	4	4
jal f1	1	3	3
add $s0, $v0, $0	1	4	4
lw $ra, 0($sp)	1	5	5
addi $sp, $sp, 4	1	4	4
jr $ra	1	3	3
f1: addi $t0, $0, 0	1	4	4
addi $v0, $0, 1	1	4	4
bne $a0, $0, else	1	3	3
jr $ra	0	3	0
else: beq $a0, $t0, done	4	3	12
addi $t0, $t0, 1	3	4	12
mult $v0, $v0, $t0 #R-type	3	4	12
mflo $v0 #R-type	3	4	12
j else	3	3	9
done:jr $ra	1	3	3
Total:	28		102

What is the CPI for this program? CPI=102/28 [3 points]

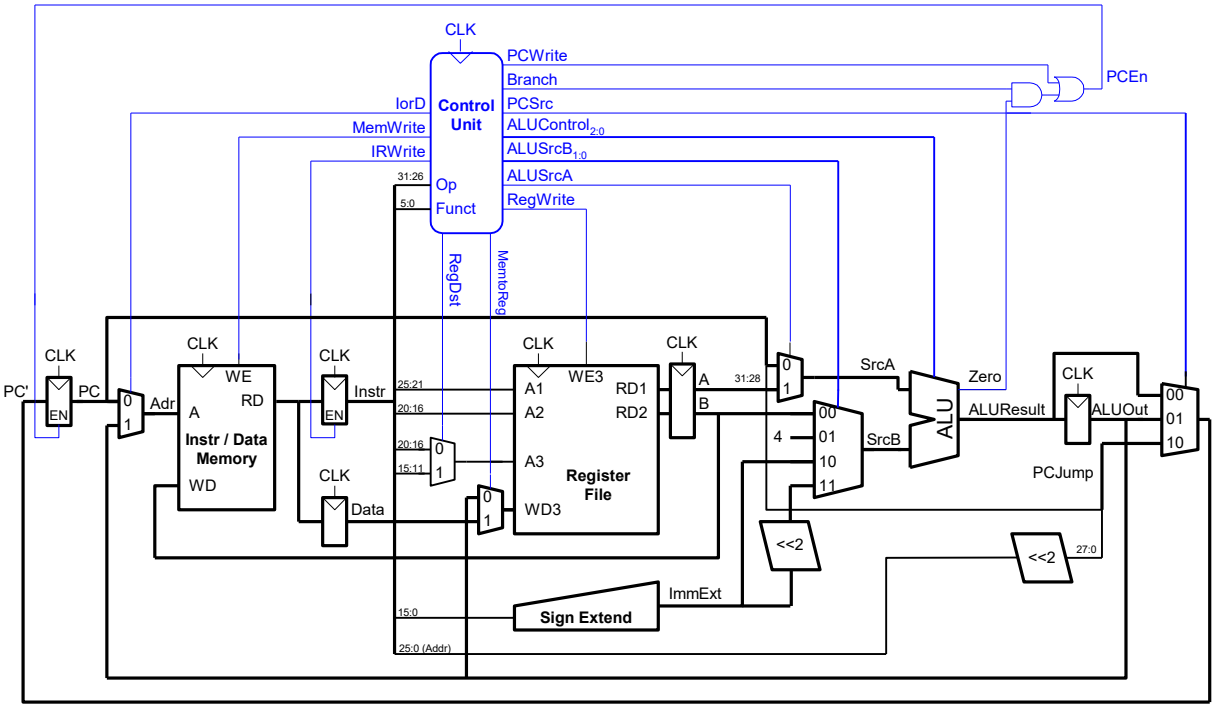


Figure 3:Multi cycle processor

Name-Last Name: _____ **Student ID:** _____

Section No (1,2,3?): ____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructors: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam	Assist. Prof. Dr. Mehmet KOSEOGLU
Duration: 120 minutes	Exam Date: 18.04.2017

Questions	1	2	3	4	5	Total
Marks	20	20	15	10	20	100
Earned						

Q1. a) Write the values of the registers after the following MIPS program finishes its execution.

```
        lui $s0, 0x1234
        ori $s0, $s0, 0x0335
        andi $s0, $s0, 0x000F
        sra $s1, $s0, 2
        or $s2, $s0, $s1
        slt $s3, $s1, $s2
        bne  $s1, $s3, else
        addi $s2, $s2, -1
else:   sll $s4, $s2, 2
        jr $ra
```

s0	s1	s2	s3	s4

b) For the given "*number*" value, what does function f1 do? Write output values (value in s0) for the given *number* values in the table.

```
main:   addi $a0, $0, number
        addi $sp, $sp, -4
        sw $ra, 0($sp)
        jal f1
        add $s0, $v0, $0
        lw $ra, 0($sp)
        addi $sp, $sp, 4
        jr $ra    #exit

f1:     addi $t0, $0, 0
        addi $v0, $0, 1
        bne $a0, $0, else
        jr $ra
else:   beq $a0, $t0, done
        addi $t0, $t0, 1
        mul $v0, $v0, $t0
        mflo $v0
        j else
done:   jr $ra
```

Write the description of f1 below:

Number	0	3	5
S0			

Q2. You have four instructions stored in the memory as given in the following table:

Instructions	Address	Instruction
Inst1	0x00400000	0x14100003
Inst2	0x00400004	0x012A4025
Inst3	0x00400008	0x2210FFFB
Inst4	0x0040000C	0x08100000
Inst5	0x00400010	---

a) Write the binary values for each instruction. Clearly show which bits corresponds to which field in the instruction format (opcode, rs, rt, rd.. etc?).

Instructions	Instruction format
0x14100003	
0x012A4025	
0x2210FFFB	
0x08100000	

b) Write down the corresponding MIPS assembly code below for each machine code.

Instructions	MIPS Code
Inst1	
Inst2	
Inst3	
Inst4	
Inst5	

Name	Register
$0	0
$at	1
$v0-$v1	2-3
$a0-$a3	4-7
$t0-$t7	8-15
$s0-$s7	16-23
$t8-$t9	24-25
$k0-$k1	26-27
$gp	28
$sp	29
$fp	30
$ra	31

Instruction	Opcode
j	000010
jal	000011
beq	000100
bne	000101
addi	001000
slti	001010
andi	001100
ori	001101
xori	001110
lui	001111
lw	100011
sw	101011

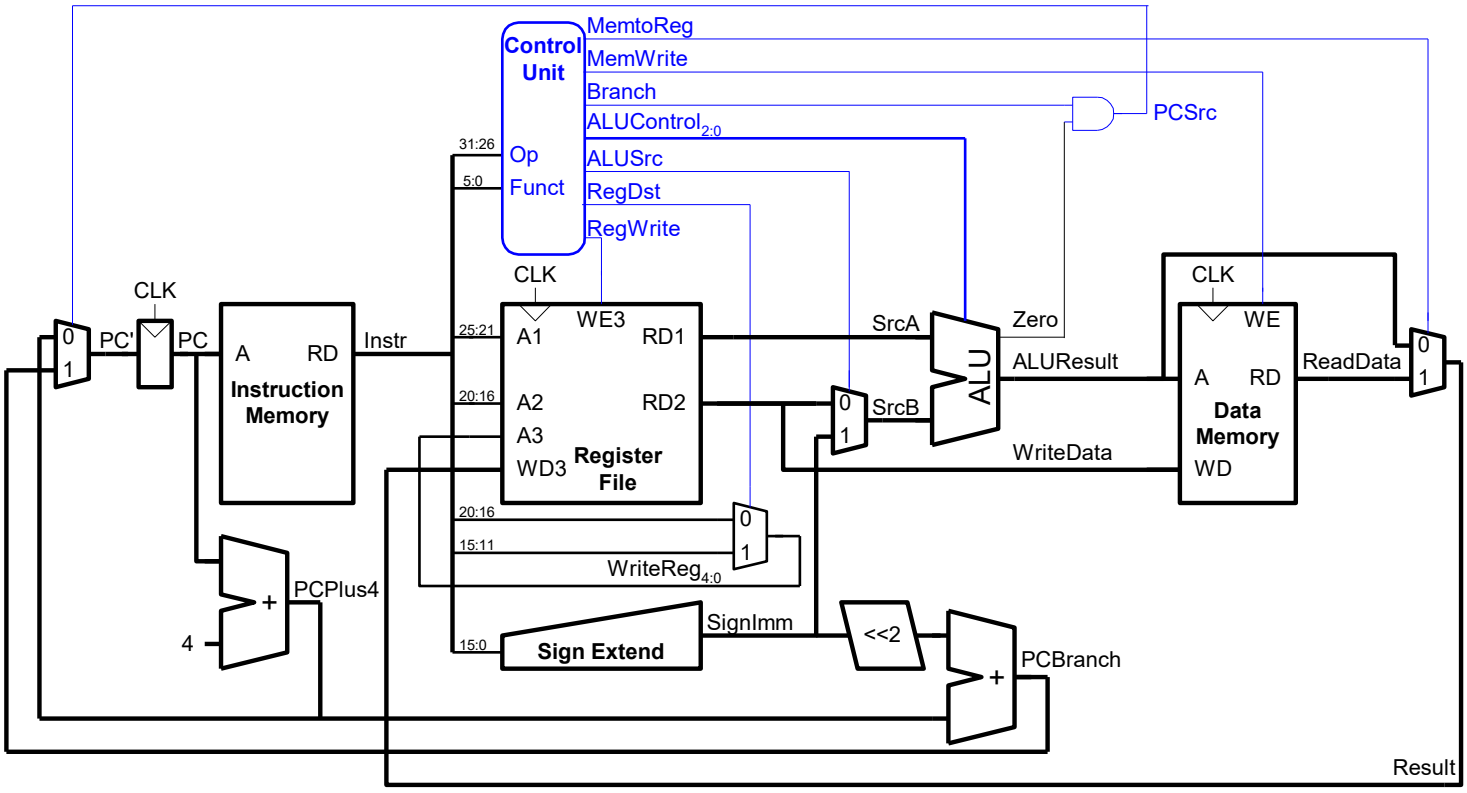
Instruction	Funct
sll	000000
srl	000010
sra	000011
jr	001000
div	011010
add	100000
sub	100010
and	100100
or	100101
xor	100110
nor	100111
slt	101011

Q3. MIPS architesture has some conditional branches and unconditional jumps. We list some of them below. For each instruction type, write the maximum number of instructions between the current program counter (PC) and the target instruction. You should also write the instruction type.

Instruction	Maximum number of instructions that we can jump over	Instruction type
J		
JR		
JAL		
BEQ		
BNE		

Q4. We would like to add R-type lw instruction (**lwr $rd, $rt($rs)**) to single cycle MIPS processor. The lwr instruction is similar to lw but it sums two registers (specified by $rs, $rt) to obtain the effective load address and uses the R-Type format. Loaded word from memory is written to register rd.

a) Show the necessary changes on data-path of single-cycle processors if any.
b) Fill the control signals in Table I.



Tabel I: Control signals for **lwr** instruction

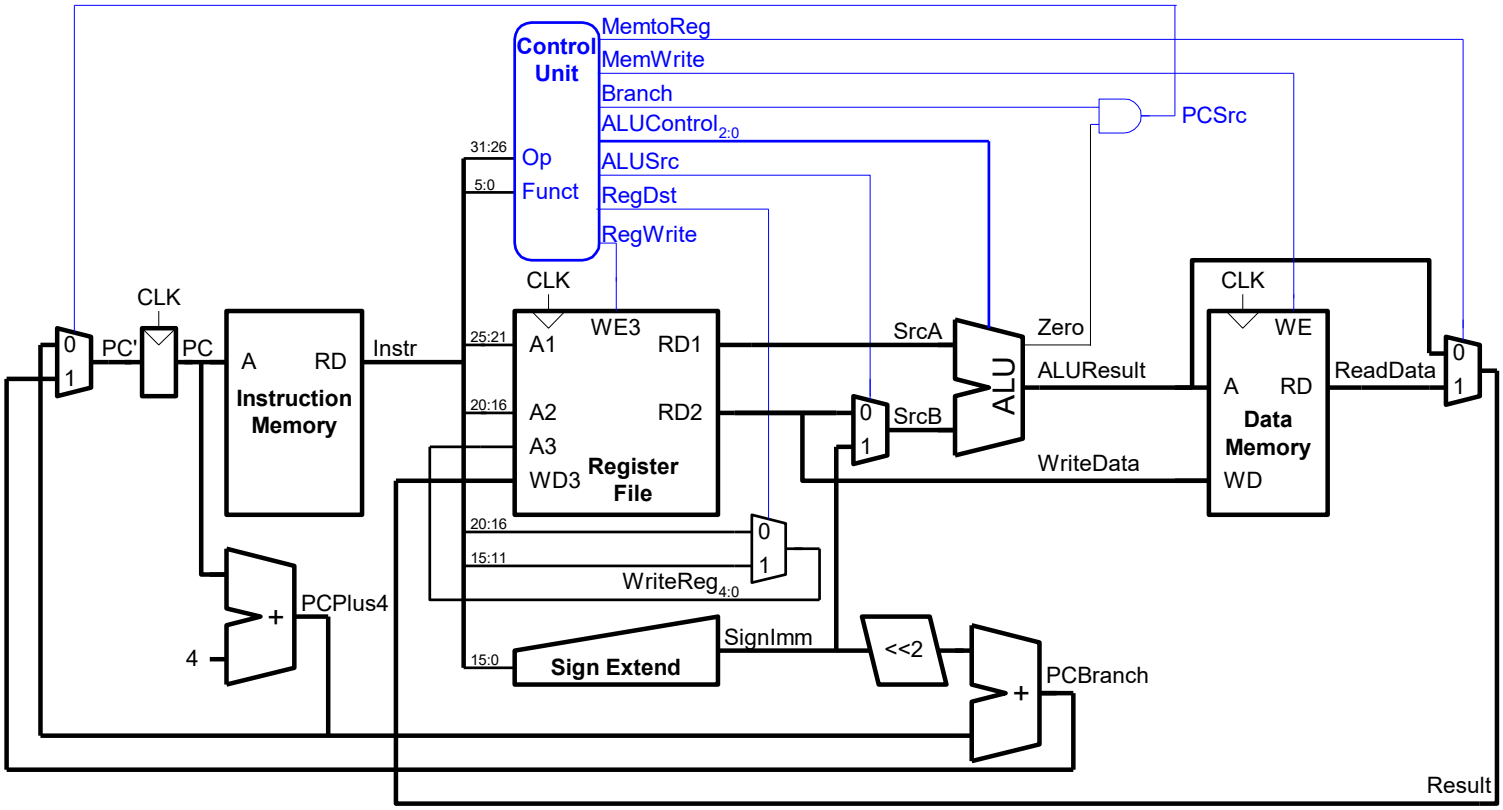
Inst.	$Op_{31:26}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$
lwr	XXXXX							

$ALUOp_{1:0}$	Operation
00	Addition
01	Subtraction
10	Look at funct.
11	Not used

Q5. We would like to also add R-type sw instruction (**swr $rd, $rt($rs)**) to single cycle MIPS processor. The swr instruction is similar to sw but it sums two registers (specified by $rs, $rt) to obtain the effective load address and uses the R-Type format. Then, it writes the word in rd register to the memory address.

a) Show the necessary changes on data-path of single-cycle processors if any. Be aware that register file has only two read ports! Briefly explain your changes.

b) Fill the control signals in Table II.



Tabel II: Control signals for **swr** instruction

Inst.	$Op_{31:26}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$
swr	XXXXX							

Name-Last Name: _____ **Student ID:** _____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructors: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam	
Duration: 100 minutes	Exam Date: 27.03.2018

Questions	1	2	3	4	Total
Marks	30	20	30	20	100
Earned					

Q1. You are given the part of the MIPS code and addresses for each instruction. You also have the Opcode and Funct values given in decimal.

Address		Instruction	Opcode/Funct
0x00400000		lui $16, 0x1000	15/.
0x00400004		jal Function	3/.
.....		……..	
.....		……..	
0x0040001C	Function	addi $16, $16, 4	8/.
0x00400020		lw $17, 0($16)	35/.
0x00400024		jr $31	?/8

a) [20 points] Write the machine code for jal and jr instructions.

	Binary	Hexadecimal
jal		
jr		

b) [10 points] Write content of following registers in hexadecimal after the program executes.

$16	
$31	

Q2. Write the MIPS instruction(s) for the following pseudoinstructions.

	Description	Pseudoinstruction	MIPS Instruction(s)
a	[s0]=not [s1]	`not $s0, $s1`	
b	Load **16-bit immediate** to $s0	`li $s0, 0x0005`	
c	**Branch** unconditionally (not jump)	`b label`	
d	Multiply $s1 and $s2, put result into 32-bit register $s0	`mul $s0, $s1, $s2`	

Q3. Write a MIPS code that counts the number of equal neighbors in an array. In another words, convert the following C code to the MIPS code. Suppose the base address of array A is 0x12348000.

```
int A[10];
int count=0;
for(i=0; i<9; i=i+1){
     if(A[i]==A[i+1])
          count=count+1;
}
```

Q4. You are given following MIPS code:

a) [8 points] In the following MIPS function, there are two mistakes on the instructions or their usage. Show them and write their correct version.

```
          addi $s0, $0, A
          addi $s1, $0, B
          addi $s2, $0, 0
while:    slt $t0, $s0, $s1
          bne $t0, $0, done
          sub $s0, $s0, $s1
          addi $s2, $s2, 1
          jr while
done:     add $v0, $s2, 0
          addi $v1, $s0, 0
```

	Wrong instruction	Corrected instruction
1		
2		

b) [8 points] Write the return values of the function ($v0 and $v1) for the following A and B values:

A=5, B=2	
v0=	v1=
A=9, B=3	
v0=	v1=

c) [4 points] What does this function do? What does it return in $v0 and $v1?

Name-Last Name: _____ **Student ID:** _____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructors: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam	
Duration: 100 minutes	Exam Date: 27.03.2018

Questions	1	2	3	4	Total
Marks	30	20	30	20	100
Earned					

Q1. You are given the part of the MIPS code and addresses for each instruction. You also have the Opcode and Funct values given in decimal.

Address		Instruction	Opcode/Funct
0x00400000		lui $16, 0x1000	15/.
0x00400004		jal Function	3/.
.....		……..	
.....		……..	
0x0040001C	Function	addi $16, $16, 4	8/.
0x00400020		lw $17, 0($16)	35/.
0x00400024		jr $31	?/8

a) [20 points] Write the machine code for jal and jr instructions.

	Binary						Hexadecimal
	~~0000~~ 0000 0100 0000 0000 0000 0001 11~~00~~						0x0040001C
jal	000011	00 0001 0000 0000 0000 0000 0111					0x 0C10 0007
jr	000000	11111	00000	00000	00000	001000	0x 03E0 0008
	opcode	Rs (31)	Rt (0)	Rd (0)	shamt	funct	

b) [10 points] Write content of following registers in hexadecimal after the program executes.

$16	0x1000 0004 (After lui, s0=0x10000000. After addi, s0=0x1000000+4=0x10000004
$31	0x 00400008 (The address after jal instruction)

Q2. Write the MIPS instruction(s) for the following pseudoinstructions.

	Description	Pseudoinstruction	MIPS Instruction(s)
a	[s0]=not [s1]	`not $s0, $s1`	nor $s0, $s1, $0
b	Load **16-bit immediate** to $s0	`li $s0, 0x0005`	addi $s0, $0, 0x0005 (ori $s0, $0, 0x0005)
c	**Branch** unconditionally (not jump)	`b label`	beq $0, $0, label
d	Multiply $s1 and $s2, put result into 32-bit register $s0	`mul $s0, $s1, $s2`	mult $s1, $s2 mflo $s0

Q3. Write a MIPS code that counts the number of equal neighbors in an array. In another words, convert the following C code to the MIPS code. Suppose the base address of array A is 0x12348000.

```
int A[10];
int count=0;
for(i=0; i<9; i=i+1){
     if(A[i]==A[i+1])
          count=count+1;
}
```

```
lui $t0, 0x1234
ori $t0, $t0,  0x8000
addi $s0, $0, 0              #count
addi $s1, $0, 0              #i
addi $s2, $0, 9

loop:  slt $t1, $s1, $s2
       beq $t1, $0, done

       sll $t2, $s1, 2       #t2=i*4
       add $t2, $t0, $t2

       lw $s3, 0($t2)
       lw $s4, 4($t2)

       bne $s3, $s4, not_eq
       addi $s0, $s0, 1      # count++

not_eq:addi $s1, $s1, 1      #i++
       j loop

done: ...
```

Q4. You are given following MIPS code:

a) [8 points] In the following MIPS function, there are two mistakes on the instructions or their usage. Show them and write their correct version.

```
          addi $s0, $0, A
          addi $s1, $0, B
          addi $s2, $0, 0
while:    slt $t0, $s0, $s1
          bne $t0, $0, done
          sub $s0, $s0, $s1
          addi $s2, $s2, 1
          jr while
done:     add $v0, $s2, 0
          addi $v1, $s0, 0
```

	Wrong instruction	Corrected instruction
1	`jr while`	`j while`
2	`add $v0, $s2, 0`	`add $v0, $s2, $0` `addi $v0, $s2, 0`

b) [8 points] Write the return values of the function ($v0 and $v1) for the following A and B values:

A=5, B=2	
v0= 2	v1= 1
A=9, B=3	
v0= 3	v1= 0

c) [4 points] What does this function do? What does it return in $v0 and $v1?

Divides A to B (A/B). v0 and v1 hold the quotient (result) and remainder of the division.

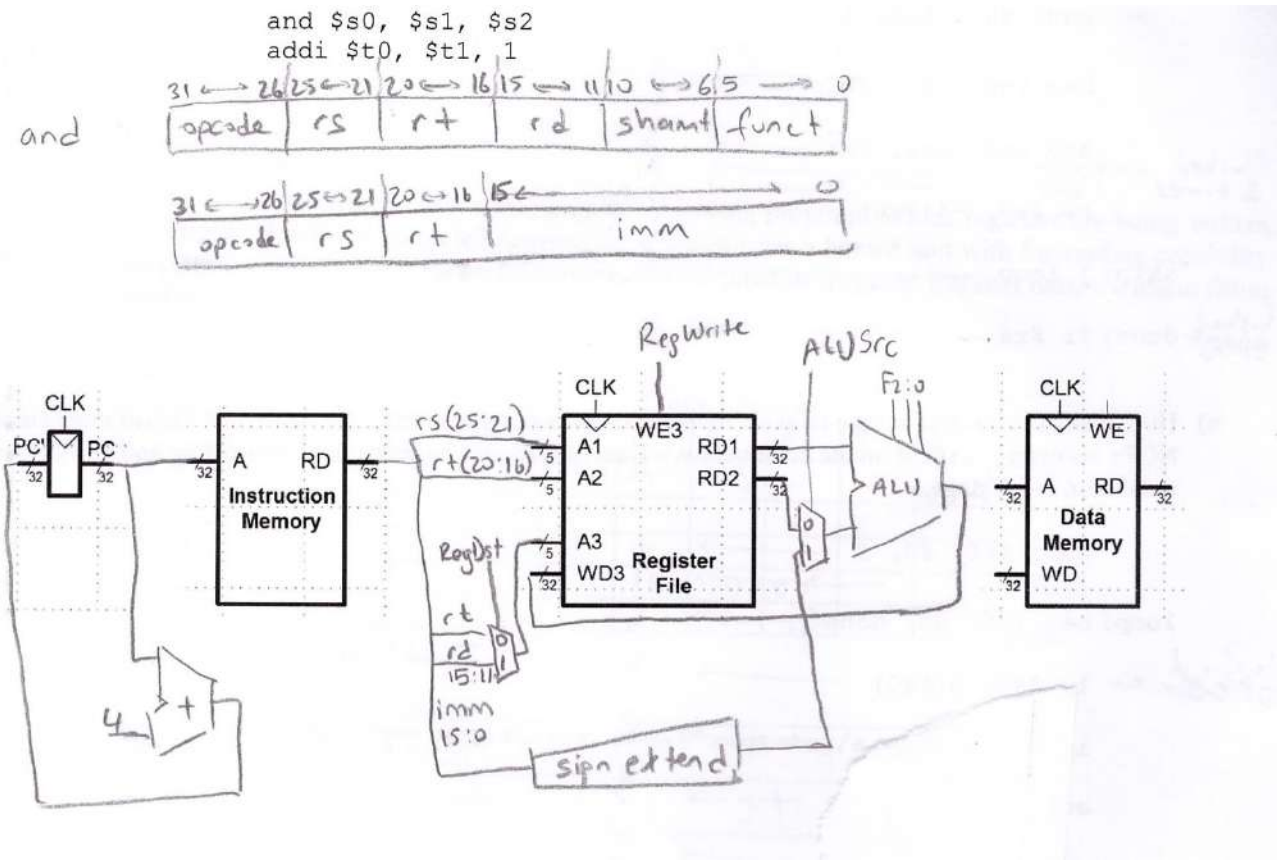
Name-Last Name: _____ **Student ID:** _____

Hacettepe University	Computer Engineering Department
BBM234 Computer Organization	Instructors: Assoc. Prof. Dr. Suleyman TOSUN
Midterm Exam 2	
Duration: 100 minutes	Exam Date: 03.05.2018

Questions	1	2	3	4	Total
Marks	30	30	20	20	100
Earned					

Q1. You have 32-bit Program Counter (PC), instruction memory, register file and a data memory.

Draw a single-cycle processor that can execute ONLY the following type of instructions. Define necessary control signals and write their values in the table. Note that you do not need some of the control signals in the table. Write NA to them if they are not used in your microarchitecture.



Tabel I: Control signals for **and** and **addi**

Inst.	$Op_{31:26}$	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	ALUOp
and	100100	1	1	0	NA	NA	NA	10 (or 1)
addi	001000	1	0	1	NA	NA	NA	00 (or 0)

Q2. You are given the following MIPS code. In the program, register `$a0` initially has the address of first array element. Array has the following data: **[5,5,8,8].**

a) How many cycles does it take to execute this code on a 5-stage pipelined processor without data hazard unit (no forwarding and no early branch resolution)? If any NOPs necessary, write it on the code below. Assume processor can flush wrong-fetched instructions when the branch or jump is resolved.
Hint: First, insert NOPs if necessary and then determine how many times each instruction is fetched.

	Instructions	Number of fetches	Explanation
	addi $t0, $0, 3	1	
	NOP	1	Data hazard on t0
	NOP	1	Data hazard on t0
loop:	beq $t0, $0, done	4	Loop iterates 3 times. Plus the last check.
	lw $s0, 0($a0)	4	Flushes at last fetch
	lw $s1, 4($a0)	4	Flushes at last fetch
	addi $t0, $t0, -1	4	Flushes at last fetch
	addi $a0, $a0, 4	3	
	beq $s0, $s1, skip	3	
	add $s0, $s0, $s1	3	Flushed twice.
	NOP	3	Data hazard on s0. Flushed twice.
	NOP	3	Data hazard on s0. Flushed twice.
	sw $s0, -4($a0)	1	
	skip: j loop	3	
done:	jr $ra	4	Flushed three times.
	Total fetches:	42	

Jr takes 4 more cc to finish its execution. Total cc= 42+4 =46 cc

b) How many cycles does it take to execute this code on a pipelined processor with data hazard unit? (If any NOPs necessary, write it on the code below.). Data hazard unit includes data forwarding and early branch resolution in ID stage.

	Instructions	Number of fetches	Explanation
	addi $t0, $0, 3	1	
	NOP	1	Data hazard on t0
loop:	beq $t0, $0, done	4	Loop iterates 3 times. Plus the last check.
	lw $s0, 0($a0)	4	Flushes at last fetch
	lw $s1, 4($a0)	3	
	addi $t0, $t0, -1	3	
	addi $a0, $a0, 4	3	
	beq $s0, $s1, skip	3	
	add $s0, $s0, $s1	3	Flushed twice.
	sw $s0, -4($a0)	1	
	skip: j loop	3	
done:	jr $ra	4	Flushed three times.
	Total fetches:	33	

Jr takes 4 more cc to finish its execution. Total cc= 33+4 =37 cc

Q3. a) The pipelined 5-stage MIPS processor is running the following program. Which registers are being written, and which are being read in the fifth cycle? Assume the processor has a hazard unit with forwarding capability. Show the stages (IF, ID, EX, M, WB) for each instruction in the pipeline diagram. Register names without filling the diagram will not be accepted.

	Clock Cycles								
Instructions	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
add $s0, $t0, $t1	IF	ID	EX	M	WB				
sub $s1, $t2, $t3		IF	ID	EX	M	WB			
and $s2, $s0, $s1			IF	ID	EX	M	WB		
or $s3, $t4, $t5				IF	ID	EX	M	WB	
slt $s4, $s2, $s3					IF	ID	EX	M	WB

Written register/s	**Read register/s**
s0 (first ins. Writes to s0 in WB stage)	t4 and t5 (or reads at ID stage)

b) The pipelined 5-stage MIPS processor is running the following program. Which registers are being written, and which are being read in the fifth cycle? Assume the processor has a hazard unit with forwarding capability. Show the stages (IF, ID, EX, M, WB) for each instruction in the pipeline diagram. Register names without filling the diagram will not be accepted.

	Clock Cycles								
Instructions	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
sw $t5, 72($t0)	IF	ID	EX	M	WB				
addi $s1, $s2, 5		IF	ID	EX	M	WB			
sub $t0, $t1, $t2			IF	ID	EX	M	WB		
lw $t3, 15($s1)				IF	ID	EX	M	WB	
or $t2, $s4, $s5					IF	ID	EX	M	WB

Written register/s	**Read register/s**
None (sw writes to memory in M stage. Nothing is written in the registers in WB stage)	s1 (lw reads s1 at ID stage to calculate the memory address)

Q4. You are given the following MIPS code running on a **single-cycle MIPS processor**. **Assume MemtoReg control signal has *stuck-at-zero* fault**, meaning that signal is always zero.

```
        lw $s0, 0($0)
        lw $s1, 4($0)
        addi $s3, $0, 5
        beq $s0, $s1, done
        sll $s3, $s3, 2
        addi $s3, $s3, 1
Done:   add $v0, $s3, $0
```

Memory

Address	Data
0x0000000C	45
0x00000008	35
0x00000004	25
0x00000000	15

a) In the following table, write down which instructions will malfunction. In other words, which instructions will not execute correctly? Write the registers that will be written by these instructions, their written values, and their expected values.

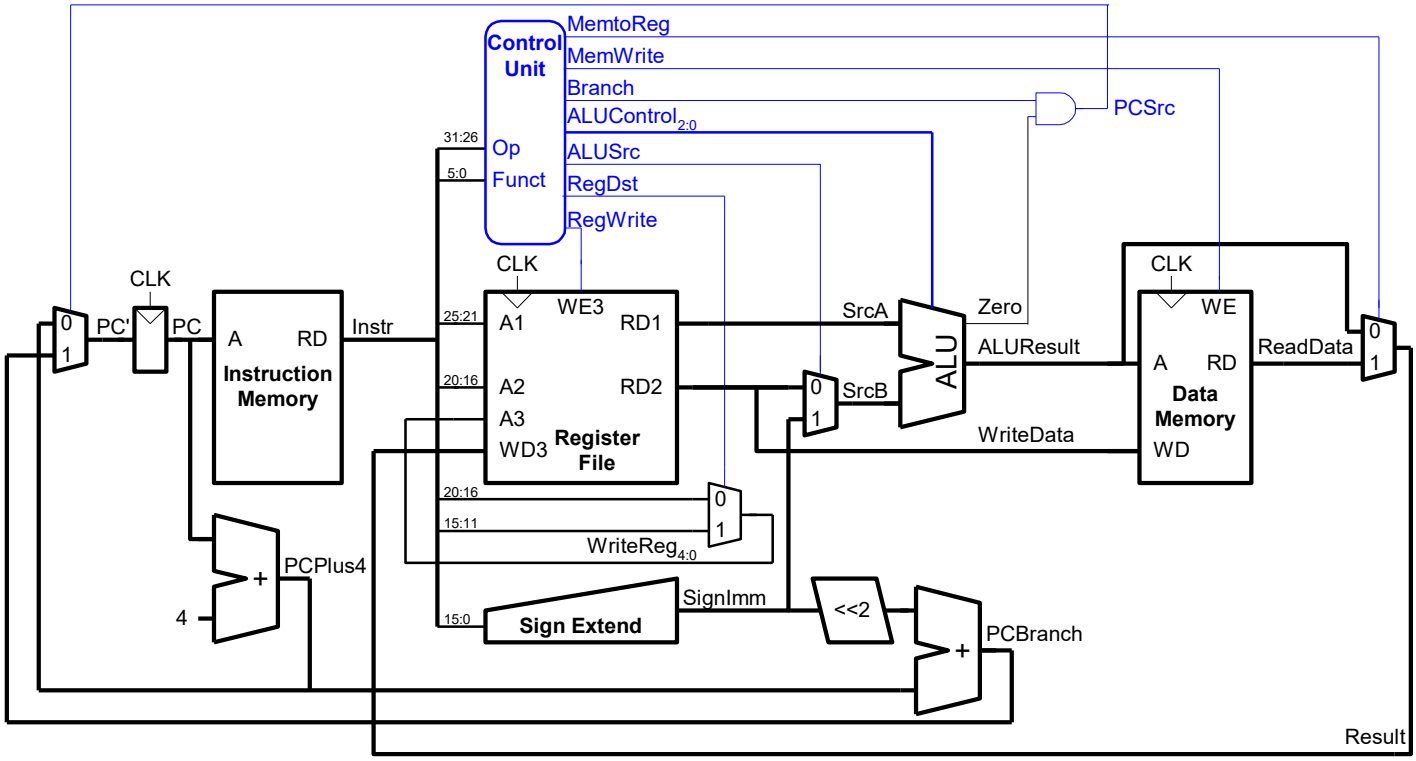
Malfunctioning instructions	**Registers written**	**Written value**	**Expected value**
lw $s0, 0($0)	s0	0	15
lw $s1, 0($0)	s1	4	25

Since lw cannot write the value from memory to register file, it writes the output of ALU, which is the calculated address.

b)

What value is returned by this function in v0 register?	**Is the returned result expected/correct value?**	**If no, what is the correct value?**
21	YES	-

Beq compares s1 and s2. We were expecting 15 and 25 in them, which are not equal. However, the written values, 0 and 4, are also not equal. Therefore, the code executes as expected although lw instructions malfunction.

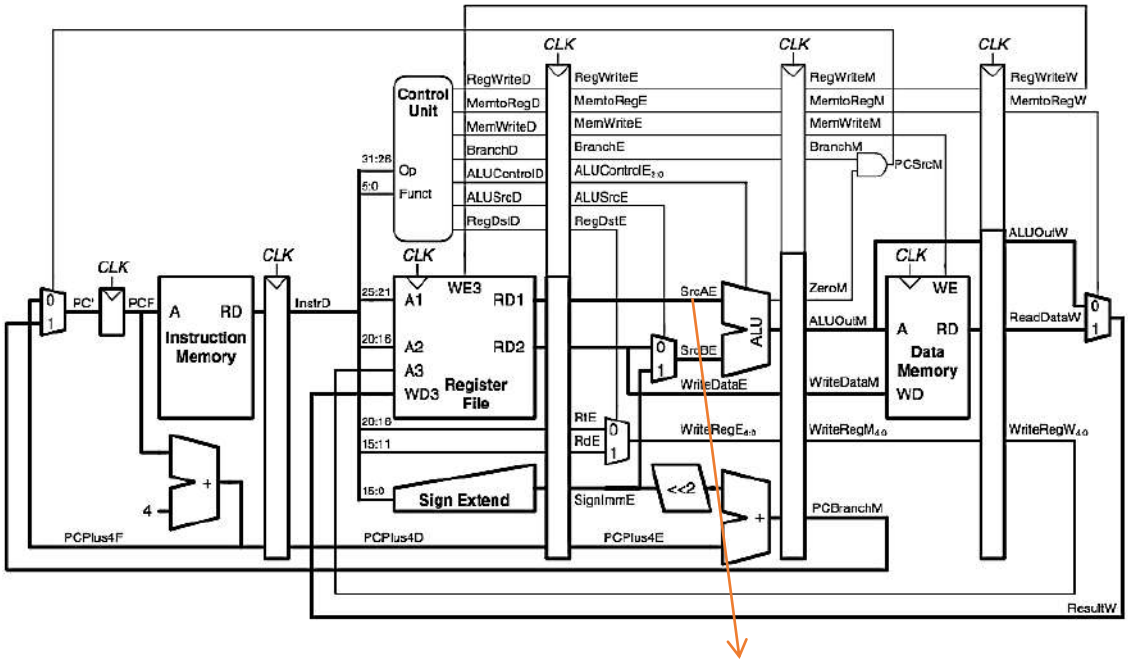


Single-cycle MIPS processor

IMPORTANT: SUBMIT TOGETHER WITH PART-I OF THE FINAL EXAM!

Name-Last Name: _____ **Student ID:** _____

Questions	4	5
Marks	20	20
Earned		



SrcAE=0x0000 0005

Q4. a) The MIPS program below is executed on the pipelined architecture given above. At a time instant, we observe that SrcAE=0x0000 0005 as shown on the figure. At the same moment, write down the instructions in the following phases:

Fetch	Decode	Execute	Memory	Writeback
`and $s5,$t2,$t3`	`sub $s4,$t3,$t4`	`add $s3,$t1,$t2`	`lw $s2,40($0)`	`addi $t4,$0,5`

b) At the same moment, what are the values of the following signals?

Signal	Value
InstrD	`sub $s4,$t3,$t4`
SrcBE	6
ALUOutM	40
MemWriteM	0
WriteRegW	t4

MIPS program:

```
addi $t1, $0, 5
addi $t2, $0, 6
addi $t3, $0, 4
addi $t4, $0, 5
lw   $s2, 40($0)
add $s3, $t1, $t2
sub $s4, $t3, $t4
and $s5, $t2, $t3
sw   $s6, 20($t1)
or   $s7, $t3, $t4
```

Q4. A cache has the following parameters: *b*, block size given in numbers of words; *S*, number of sets; *N*, number of ways; and *A*, number of address bits. The following repeating sequence of lw addresses (given in hexadecimal) are retrieved in a loop which is executed **three times**.

40 44 48 4C 70 74 78 7C 80 84 88 8C

Assuming least recently used (LRU) replacement for associative caches, determine the miss rate at each loop if the sequence is input to the following caches. Cache is empty at the beginning.

a) direct mapped cache,
S = 16, b = 1 word

Data	
7C	Set 15
78	Set 14
74	Set 13
70	Set 12
	Set 11
	Set 10
	Set 9
	Set 8
	Set 7
	Set 6
	Set 5
	Set 4
4C 8C	Set 3
48 88	Set 2
44 84	Set 1
40 80	Set 0

	No of hits	No of misses
1st loop	0	12
2nd loop	4	8
3rd loop	4	8

b) 2-way associative cache
S=8, *b*=1 word

Way 1 Data	Way 0 Data	
	7C	Set 7
	78	Set 6
	74	Set 5
	70	Set 4
8C	4C	Set 3
88	48	Set 2
84	44	Set 1
80	40	Set 0

	No of hits	No of misses
1st loop	0	12
2nd loop	12	0
3rd loop	12	0

c) direct mapped cache,
S=8, b=2 words

Data	Data	
7C	78	Set 7
74	70	Set 6
		Set 5
		Set 4
		Set 3
		Set 2
4C 8C	48 88	Set 1
44 84	40 80	Set 0

	No of hits	No of misses
1st loop	6	6
2nd loop	8	4
3rd loop	8	4