Assignment 3

CMPT 215
Name: Yinsheng Dong
Student Number: 11148648
NSID: yid164
Due: Aug 10^{th}, 2017

August 11, 2017

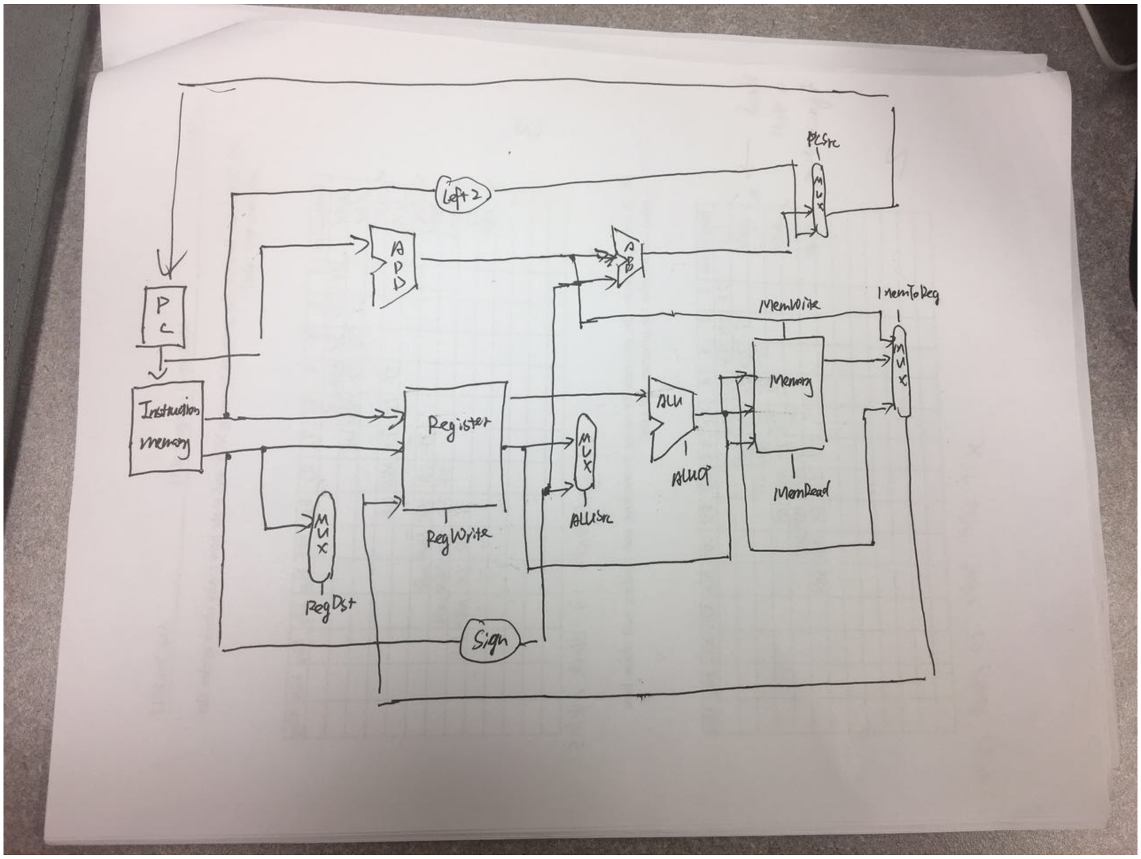
Total: 35
Please show all your work to get full marks

Problem 1.
(4 marks) Describe the modifications to the single clock cycle datapath that would be needed to implement the jal instruction and give the control signal setting that would be required for this instruction.

Solution:
Instruction type of jal: J
Operation of jal: R[31] = PC + 8; PC = JumpAddr
Modifications:
1. Need to change a 26-bits immediate so we need to create a 32-bit PC to store it and we also need a PCSrc multiplexor for handle other third input
2. Need a WriteData port of the register file will route the return address
3. Will need write back the address to register 31, so it has to be hard coded

Control signal setting:
PCSrc
RegDst
RegWrite
Mem-Read
Mem-Write
MemToReg
ALUSrc
ALUOp

The datapath is in below

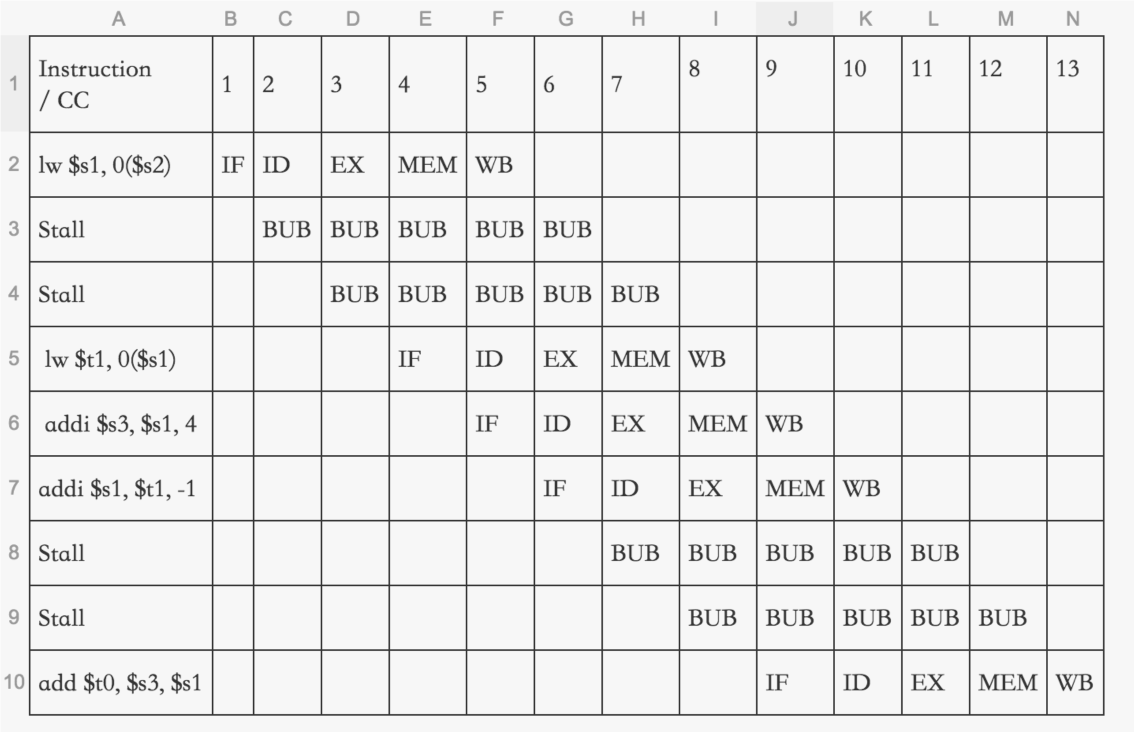


Problem 2.

(4 marks) Determine the clock cycle at which each of the instructions in the sequence given below would be completed, assuming the 5-stage pipeline without forwarding, and numbering the clock cycle at which the first of the instructions is fetched as clock cycle 1. Assume that if one instruction reads a register during the same clock cycle as another instruction is writing it, the new value will be read. Do not reorder the instructions. Instructions are fetched and executed exactly in the order given below, with the pipeline stalling if necessary.

```
lw $s1,0($s2)
lw $t1,0($s1)
addi $s3,$s1,4
addi $s1,$t1,-1
```

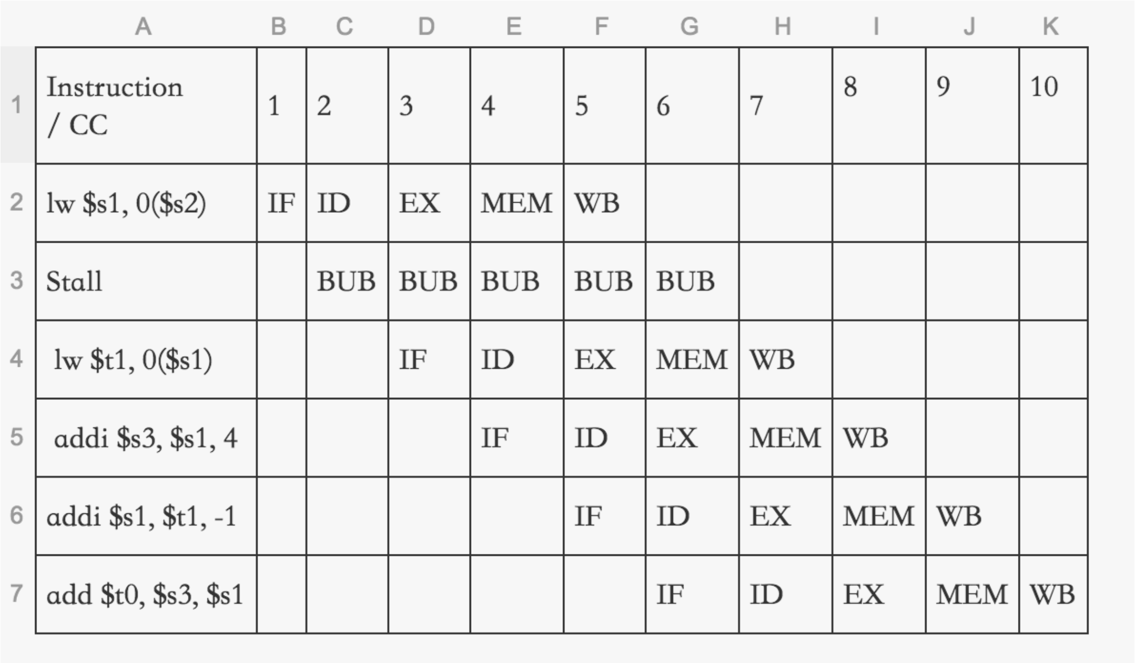
```
add    $t0 , $s3 , $s1
```



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Instruction / CC	1	2	3	4	5	6	7	8	9	10	11	12	13
2	lw $s1, 0($s2)	IF	ID	EX	MEM	WB								
3	Stall		BUB	BUB	BUB	BUB	BUB							
4	Stall			BUB	BUB	BUB	BUB	BUB						
5	lw $t1, 0($s1)				IF	ID	EX	MEM	WB					
6	addi $s3, $s1, 4					IF	ID	EX	MEM	WB				
7	addi $s1, $t1, -1						IF	ID	EX	MEM	WB			
8	Stall							BUB	BUB	BUB	BUB	BUB		
9	Stall								BUB	BUB	BUB	BUB	BUB	
10	add $t0, $s3, $s1									IF	ID	EX	MEM	WB

Solution:
The total clock cycle should be 13

Problem 3.
(4 marks) Repeat problem 2 but now assume forwarding.



	A	B	C	D	E	F	G	H	I	J	K
1	Instruction / CC	1	2	3	4	5	6	7	8	9	10
2	lw $s1, 0($s2)	IF	ID	EX	MEM	WB					
3	Stall		BUB	BUB	BUB	BUB	BUB				
4	lw $t1, 0($s1)			IF	ID	EX	MEM	WB			
5	addi $s3, $s1, 4				IF	ID	EX	MEM	WB		
6	addi $s1, $t1, -1					IF	ID	EX	MEM	WB	
7	add $t0, $s3, $s1						IF	ID	EX	MEM	WB

Solution:
The total clock cycle in the forwarding should be 10

Problem 4.
(9 marks) Consider the following code fragments:

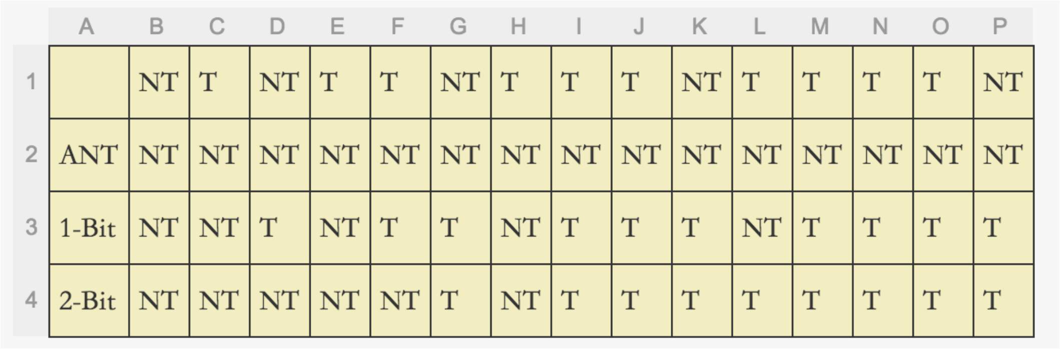
```
                li    $s1,1
                li    $s3,6
                add   $s4,$zero,$zero
outer_loop:     add   $s2,$zero,$zero
inner_loop:     addi  $s2,$s2,1
                mul   $t0,$s2,$s1
                add   $s4,$s4,St0
inner_test      bne   $s2,$s1,inner_loop
                addi  $s1,$s1,1
                bne   $s1,$s3,outer_loop
```

This code simple runs the inner loop $i+1$ times each time, with i starting at 1, going until 5, meaning the **inner_test** will be executed $1+2+3+4+5 = 15$ times. Give the branch prediction accuracy for **inner_test** for the following schemes:

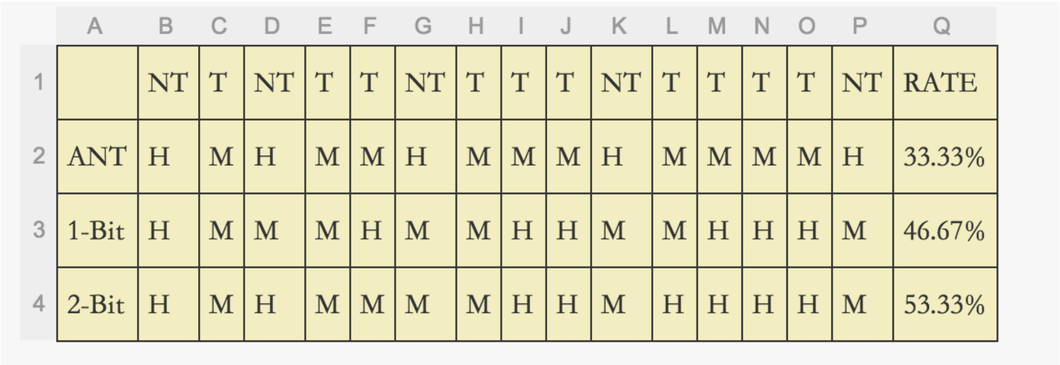
i Branch prediction of always **Not Taken**

ii 1-bit branch prediction of starting at **Not Taken**

iii 2-bit branch prediction of starting at **Strongly Not Taken**



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1		NT	T	NT	T	T	NT	T	T	T	NT	T	T	T	T	NT
2	ANT	NT	NT	NT	NT	NT	NT	NT	NT	NT	NT	NT	NT	NT	NT	NT
3	1-Bit	NT	NT	T	NT	T	T	NT	T	T	T	NT	T	T	T	T
4	2-Bit	NT	NT	NT	NT	NT	T	NT	T	T	T	T	T	T	T	T



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1		NT	T	NT	T	T	NT	T	T	T	NT	T	T	T	T	NT	RATE
2	ANT	H	M	H	M	M	H	M	M	M	H	M	M	M	M	H	33.33%
3	1-Bit	H	M	M	M	H	M	M	H	H	M	M	H	H	H	M	46.67%
4	2-Bit	H	M	H	M	M	M	M	H	H	M	H	H	H	H	M	53.33%

Solution:
The Always Not Taken, 1-bit branch predication and 2-bit branch predication are all in above tables, so finally we have got the rate of accuracy for inner_test:
Always Not Taken: 33.33%
1-Bit: 46.67%
2-Bit: 53.33%

Problem 5.
(4 marks) Suppose that a program does read operations on the following memory addresses **96, 508**. Give the number of the memory block that each of these addresses belongs to, for each of the following memory block sizes. Remember the above addresses are byte addressed.

i block size of one word (4 bytes)

Solution:
Number of memory block = address/block size
For Address 96: 96/4 = 24
For Address 508: 508/4=127

ii block size of four words (16 bytes)

Solution:
For Address 96: 96/16 = 6
For Address 508: 508/16=31

Problem 6.
(8 marks) Give the position (or set) in the cache that would be checked on each of the read operations of the above question, for each of the following caches.

i Direct-mapped cache with total capacity of 16 one-word blocks

Solution:
cache position = (block number) mod (number of block)
for address 96: 24 mod 16 = 8
for address 508: 127 mod 16 = 15

ii Direct-mapped cache with total capacity of 4 four-word blocks

Solution:
for address 96: 6 mod 4 = 2
for address 508: 31 mod 4 = 3

iii 4-way set-associative cache with total capacity of 16 one-word blocks

Solution:
total capacity (C): 16
set-associative (E): 4
set (M): 4
for address 96: 24 mod 4 = 0
for address 508: 127 mod 4 = 3

iv 2-way set-associative cache with total capacity of 4 four-word blocks

Solution:
total capacity (C): 4
set-associative (E): 2
set (M): 2
for address 96: 6 mod 2 = 0
for address 508: 31 mod 2 = 1

Problem 7.
(2 marks) Consider a computer system in which a physical page number is 24 bits, a virtual page number is 52 bits, and a virtual address is 64 bits. What is the maximum amount of physical memory, in GiB, that this system could have?

Solution:
$physical\ memory\ (bytes) = (virtual\ address \div virtual\ page\ number) \times physical\ page\ number$
$= (2^{64} \div 2^{52}) \times 2^{24}$
$= 2^{36}$
to convert to GiB:
$2^{36} \div 2^{30}$
$= 2^{6}$
$=$ 64 GiB

Bonus:

Problem 8.
(5 marks) Given two threads that share memory:

```
int a = 0;
int b = 4;
int c = 0;
int d = 0;
int z = 0
```

```
Thread 1:                       Thread 0:
 c = a+b;                       d = a+b;
 c = 0;                         z = a+d;
 a = d+b;                       a = c+d;

 end
```

What is the values of each variable at the end of the program? What is the technical term for what is going on? What are two ways to keep mutual exclusion?

Solution:
Final values for each variable:
a = 8
b = 4
c = 0
d = 8
z = 12

This technical term called structure hazard
Two ways to keep mutual exclusion:
1. Instruction level parallelism
2. Resource scheduling