NYU | TANDON SCHOOL OF ENGINEERING

Examination Book

A student using or receiving unauthorized assistance during an examination, as from notes or other students, is in violation of academic regulations and is subject to academic discipline, including forfeiture of credit for the course, probation and dismissal from NYU Tandon School of Engineering.

Name AMRUTHA PATIL

Subject CSA I.D. No. AP7982

Instructor _____

Exam. Seat No. _____ Section _____

Date _____ Grade _____

NEW YORK UNIVERSITY

1. Given that g is register x5.
h is register x6.

1.(i). ~~BLT X6, X5 CONDI~~ BLT X5, X6 CONDI

ADDI X5, X5, 1

CONDI : ADDI X6, X6, -1

1.(ii). BGE X5, X6, CONDI

LD X6, X0

CONDI : LD X5, X0

2. Given two registers are X5 and X6.
The best algorithm to swap 2 two registers without using a 3rd register is:
~~tet~~

X5 := X5 XOR X6 (XOR the registers & store in X5)
X6 := X6 XOR X5 (XOR the registers & store in X6)
X5 := X5 XOR X6 (XOR the registers & store in X5)

The RISC-V coole:

```
XOR X5, X6
XOR X6, X5
XOR X5, X6
```

3.

3.1. addi is an I type instruction
Hence, it will take up read, two registers with read, and one register is written.
Therefore
Energy spent = imem + 2x Iregister read + register write
= 140 pJ + 2x 70pJ + 60 pJ
= 340 pJ

3.2. ld is a R type instruction.
Hence, energy spent is similar to I type plus another 140pJ.
Therefore,
Energy spent = 140 pJ + 2x 70 pJ + 60pJ + 140 pJ
= 480 pJ.

3.3. beq will take up i-mem and 2 registers.
Therefore,
Energy spent = 140pJ + 2x 70 pJ.
= 280 pJ

4.

4.1.

- Data forwarding is required to resolve this hazard. The second add instruction will require the value of X1. Hence data needs to be forwarded from first add to second add after completion of the ALU.
- We need to use stalls here as well. NOP needs to be inserted after the first add. Since the first add has 2 ALU cycles the second add needs to wait for its completion to access the value of X1.
- Therefore, data forwarding and NOP is required to resolve this hazard.
- ADD X1, X2, X3
  NOP
  ADD X5, X4, X1

4.

4.2.

	CC0	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8
ADD X1, X2, X3	IF	ID	ALU	ALU	MEM	WB			
ADD X5, X4, X1			IF	ID	ALU	ALU	MEM	WB	

→ NOP required here.

Therefore, after adding NOP and utilizing data forwarding we have:

~~ADD X1, X2, X3~~
~~ADD X5, X4, X1~~
CC0

	CC0	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8
ADD X1, X2, X3	IF	ID	ALU	ALU	MEM	WB			
NOP									
ADD X5, X4, X1			IF	ID	ALU	ALU	MEM	WB	

5. The loop will run for 5 times until the value of so is 0

Hence the total cycles of this program is given by:

$4 + 5 \times (3 + 4 + 3) + 3 = 57$ clock cycles

Also, the no. of instructions executed is given by

$1 + (3 * 5) + 1 = 17$

Therefore, $CPI = \frac{57}{17} = 3.35$

6.

	1	2	3	4	5	6	7	8	9
xor	IF	ID	EX	MEM	WB				
addi		IF	ID	EX	MEM	WB			
lw			IF	ID	EX	MEM	WB		
sw				IF	ID	EX	MEM	WB	
or					IF	ID	EX	MEM	WB

At 5th clock cycle:

- At 5th clock cycle the xor instruction completes its execution and writes the value into S1.
- The addi instruction completes its execution & places the result into memory.
- The lw instruction performs execution.
- The sw instruction is decoded
- The or instruction fetches the registers. S0 and S1.
- S1 is being written.

S0, S1, S4 are being read.

The hazard detection unit:

→ need to take care of both read and write on S1.

→ stalls need to installed for proper execution.

7. Give 5 stage pipeline is :

7.1. or x13, x12, x11
ld x10, 0(x13) Ex to 1st RAW
ld x11, 8(x13) Ex to 2nd RAW
add x12, x10, x11 MEM to 1st RAW & MEM to 2nd RAW
subi x13, x12, 16 Ex to 1st RAW.

The data hazards & their resolution :

- Ex to 1st RAW hazard occurs when value of x13 is accessed by ld instruction when it is still in use by EX of or instruction. It can be resolved with 2 NOPs.
- Ex to 2nd RAW hazard occurs when value of x11 is accessed by 2nd ld instruction when it is still in use by EX of or instruction. It can be resolved with 2 NOPs.
- MEM to 1st and MEM to 2nd occurs when add instruction tries to access x11 and x10 respectively. It can be resolved with 2. NOPs.
- Ex to 1st RAW hazard occurs when value of x12 is accessed by subi instruction ~~when~~ while it is still in use by EX of add instruction.

```
OR x13, x12, x11
NOP
NOP
ld x10, 0(x13)
ld x11, 8(x13)
NOP
NOP
add x12, x10, x11
NOP
NOP
subi x13, x12, 16
```

7.2.

		1	2	3	4	5	6	7	8	9	10
1	OR	IF	ID	EX	MEM	WB					
2	ld		IF	ID	EX	MEM	WB				
3	ld		.	IF	ID	EX	MEM	WB			
4	NOP	(data load use hazard so make use of NOP)									
5	add					IF	ID	EX	MEM	WB	
6	subi						IF	ID	EX	MEM	WB

① ForwardA = X ForwardB = X
No instruction.

② ForwardA = X ForwardB = X
No instruction.

③ ForwardA = 0 ForwardB = 0
both registers of OR instr

④ ForwardA = 2 ForwardB = 0
RS1 in 1st ld from EX/MEM

⑤ ForwardA = 1 ForwardB = 0
RS1 in 2nd ld from MEM/WB

⑥ ForwardA = X ForwardB = X
No instruction.

⑦ ForwardA = 0 ForwardB = 1
RS2 in add instruction from MEM/WB, no forwarding for first operand.

⑧ ForwardA = 1 ForwardB = 0
RS1 of subi instruction from EX/MEM.

8.

	Instruction / Program	CPI	Circuit Complexity
A	it will decrease. → The new instruction will complete add in one instruction. which could have originally taken 2 instructions.	it has no effect. → the ALU will complete 3 adds in same instruction so no change in the CPI.	it will increase. → higher complexity of architecture is required to ~~add~~ enable one more operand for add.
B	it has no effect → same number of instructions hence no change in instructions.	it will increase → same ALU has to perform multiple operations hence stalls may be needed.	it will decrease. → the adder for PC can be eleminated. Hence, circuit can be simpler.
C	it has no effect → most instructions will remain same. in some cases it might decrease when using GH	it ~~will~~ in has no effect → the instruction will perform same hence no change in CPI.	it will increase. → complex circuit is required for this increase in register file.