JON CRAWFORD

CDA3101
Assignment 5
Pipelining Exercises

Objectives. Learn how instructions are pipelined and branches are predicted.

Solve the following problems. You must show your work in order to receive full credit.
(1) Fill in the traditional pipeline diagram that shows in which cycle each of the instructions in problem 1 will be in each stage of the 5-stage MIPS pipeline discussed in class. (25 points)

cycle	1	2	3	4	5	6	7	8	9	10
or $t4,$t2,$t4	IF	ID	EX	MEM	WB					
lw $t3,0($t4)		IF	ID	EX	MEM	WB				
add $t5,$t3,$t2			IF	ID	'S'	EX	MEM	WB		
sw $t3,0($t5)				IF	'S'	ID	EX	MEM	WB	

(2) Consider a code sequence consisting of 100 consecutive `lw` instructions. Except for the first instruction, each instruction uses the register that was loaded in the previous instruction as shown below. What would be the total number of cycles required for this sequence of instructions to execute given the pipelined datapath of Figure 4.65 on page 325 of your text? (25 points)

```
lw $t2,0($t1)
lw $t2,0($t2)
...
```

HAS DATA FORWARDING
n = 100

C1	C2	C3	C4	C5	C6
IF	ID	EX	MEM	WB	

OR 5 STAGES + (n-1)
5 + (100-1) = 104

```
    IF   ID   EX   MEM  WB
         IF   ID   EX   MEM  WB
```
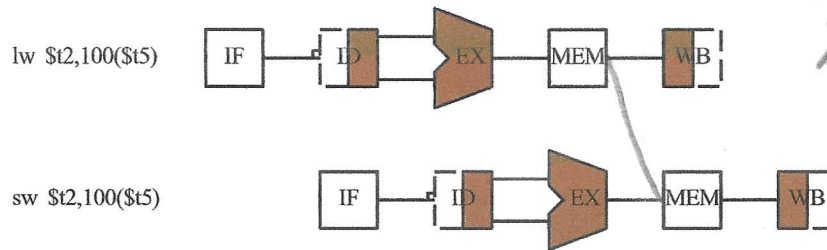
AFTER 4 THE PIPELINE IS FULL,

SO 4 + N OR 104.

WITHOUT FORWARDING THERE WOULD
BE 2 STALLS ADDED TO ALL 99 ROWS.

(3)   Consider an instruction sequence used for a memory-to-memory copy.

```
lw    $t2,100($t5)
sw    $t2,200($t6)
```

The value in $t2 to be stored into memory by the sw is actually not needed until *MEM* stage of the MIPS pipeline. Below is a drawing showing the two instructions being pipelined. Modify this figure in a manner similar to Figure 4.29 on page 279 of your text to show how the value could be forwarded to avoid a stall.

THE LOAD VALUE IS NOT
AVAILABLE UNTIL MEM/WB
AND IS NOT NEEDED UNTIL
EX/MEM

lw $t2,100($t5)   IF — ID — EX — MEM — WB

sw $t2,100($t5)   IF — ID — EX — MEM — WB

NEED A MUX IN
FRONT OF DM TO
TAKE THIS VALUE
AND A CONTROL UNIT
TO PICK IT, WE CAN
CALL IT "FORWARD DM"

Rewrite the following stall formula so this code sequence won't stall. (25 points)
NEED TO STOP IT FROM GOING BACK ALL THE WAY FOR MORE

```
if (ID/EX.MemRead AND
    (ID/EX.RegisterRt == IF/ID.RegisterRs OR
     ID/EX.RegisterRt == IF/ID.RegisterRt) AND
    ID/EX.RegisterRt != 0)
  stall the pipeline
```

IF (ID/EX.MEM READ AND IF/ID   AND
(ID/EX.REGISTER RT== IF/ID.REGISTER RS OR
ID/EX.REGISTER RT == IF/ID.REGISTER RT) AND
ID/EX.REGISTER RT !=0)
STALL THE PIPELINE

(4)   Fill in the following branch prediction information for the table below for the various types of branch predictors for the predictions of a single branch in an application. Assume the initial state of the 1-bit predictor is not taken and the initial state of the 2-bit predictor is the bottom right state in Figure 4.63. (25 points)

branch result	1-bit predictor		2-bit predictor	
	prediction	correct?	prediction	correct?
taken	0	No	01	No
taken	1	YES	10	YES
not taken	1	No	11	No
taken	0	No	10	YES
taken	1	YES	11	YES

I KNOW SO   *I THINK SO*
11 → 10
00   *01*
NO!   *IT COULD HAPPEN*