VLSI Architecture Design Course (048853)
Final Exam
June 25 2001

Electrical engineering Department

Student name:_____**Student number**_____

This exam contains TWO questions.
The exam duration is 2:30 hours.
Please fill the answers ON THE EXAM forms.

TAKE YOUR TIME, READ THE QUESTION THOUROUGHLY, UNDERSTAND THE CONTENT AND ONLY THAN START TO ANSWER

Good luck!

Q1	
Q2	
Total	

Question 1 (50%)

This question is related to memory accesses - Store and Load instructions.

Figure 1, shows the basic Producer (creates the value in a Register), *Store* the value into Memory. The Value is *load*ed from memory into a register and than being used by the Consumer.

You'll have to evaluate three memory Load/Store related concepts:
- **Write buffer**
- **Memory disambiguation**
- **Memory renaming**

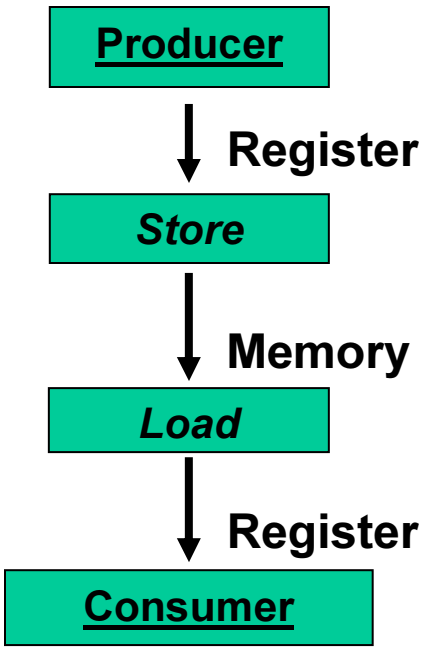


Figure 1

1A: Write Buffer

Write Buffer is a FIFO (First In First Out), serves as a buffer between the fast execution and the memory (caches). The Write buffer has N lines with a following structure:

N

Address	Data

Question 1A (15%):

a. What is the purpose of a write buffer?

_____

_____

_____

b. What are the attributes that impact the size (N lines) of the Write buffer?

1. _____
2. _____

c. When a Load instruction is executed, what actions related to the Write buffer should be performed?

_____

_____

d. Is the address Tag of the Write buffer Physical or Virtual? Why?

_____

_____

e. Bonus Question (5%): Describe the Load instruction execution process, for data that is non-aligned with Write Buffer boundary (non aligned access) :

1. _____
2. _____
3. _____
4. _____
5. _____

1B: Memory disambiguation

In an out of order machine, when a Load-after-Store instruction is executed, we would like to enable Out Of Order memory access (Load performed before Store). To enable the OOO access we should check first the identity of the addresses. A predicator to check this identity is built:

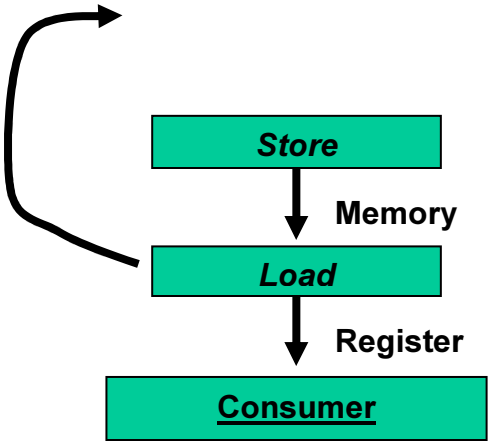


Figure 2

Question 1B (15%):

a. Mark with V those cases that Memory disambiguation process should be performed?

Store instruction \ Load instruction	Immediate	indirect
Immediate		
Indirect		

b. Suggest a simple structure of the prediction table for Memory Disambiguation (being used when scheduling loads).

_____

_____

_____

c. What is the process of Recovery (when your prediction was wrong)?

_____

_____

1C:Memory renaming

To cut Store Load latencies, the Producer can transfer the DATA directly (bypass) to the consumer (see Figure 3).

In this question, we will use the ROB (Re Order Buffer) as the structure that checks the Store Load address identity, and bypasses the value from the producer to the consumer.

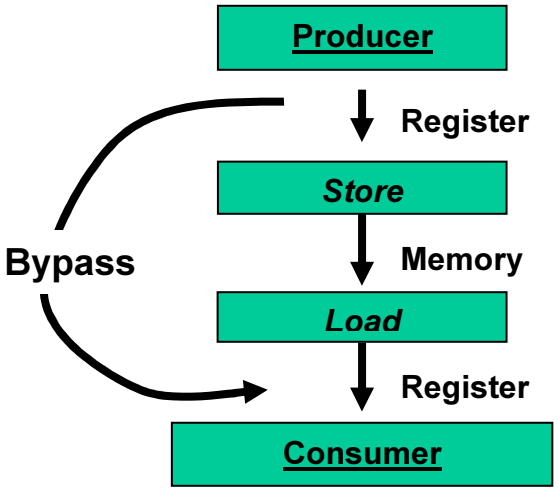


Figure 3

Question 1C (20%):
The following sequence of instructions are in the ROB:

n Add R3, R4 →R4
n+1 Store R4, M1 →M1
n+2 Sub R5, R7 →R7
n+4 Add R8, R6 →R6
n+5 Load M1, R1 →R1
n+6 Add R1, R5 →R5

a. For a wide OOO machine, what are the sequential execution steps that should be perform?

1. _____

2. _____

3. _____

4. _____

b. A wide OOO machine has a Memory Renaming mechanism.
The Memory Renaming mechanism does:

- **Identifies Load/Store pairs with the same addresses in the ROB,**
- **Identifies the "Producer" (Source) of the Store (R4 in our case) and the "Consumer" (Destination) of the Load (R1 in our case).**

Consider that Producer/Consumer Source and Destination pointers are provided by the Memory Renaming mechanism (known ahead of time), What are the sequential execution steps that should be perform in the above machine?

1. _____
2. _____
3. _____
4. _____
5. _____

c. Explain the differences between the two cases (OOO without and with Memory renaming):

_____

_____

_____

d. Bonus question (5%): In case of a Recovery. What are the process steps we have to take?

1. _____
2. _____
3. _____

Question 2: Branch Prediction (50%)

Question 2A (10 %)
In all 2 level predictors we use either local History or global history

a. What the purpose of using history counters?

_____

_____

b. **What can we learn from the information found in each History type?**

Local History: _____

_____

Global History: _____

_____

c. **When is it better to use global history?**

_____

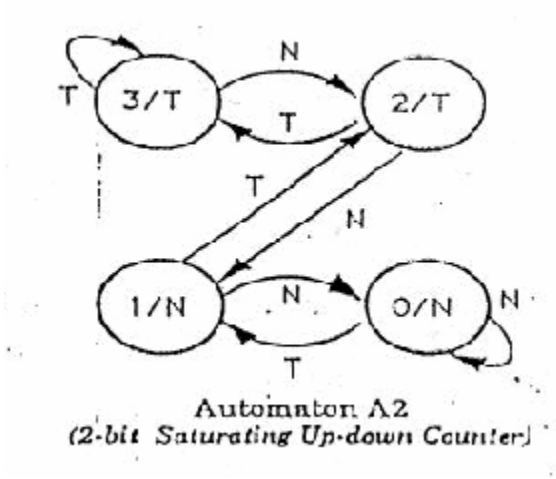
_____

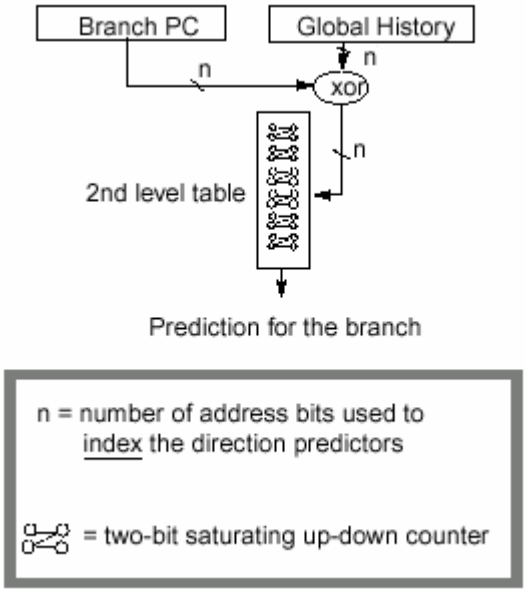
d. **When is it better to use local history?**

_____

_____

Question 2B (8%)
Consider the following Gshare predictor:

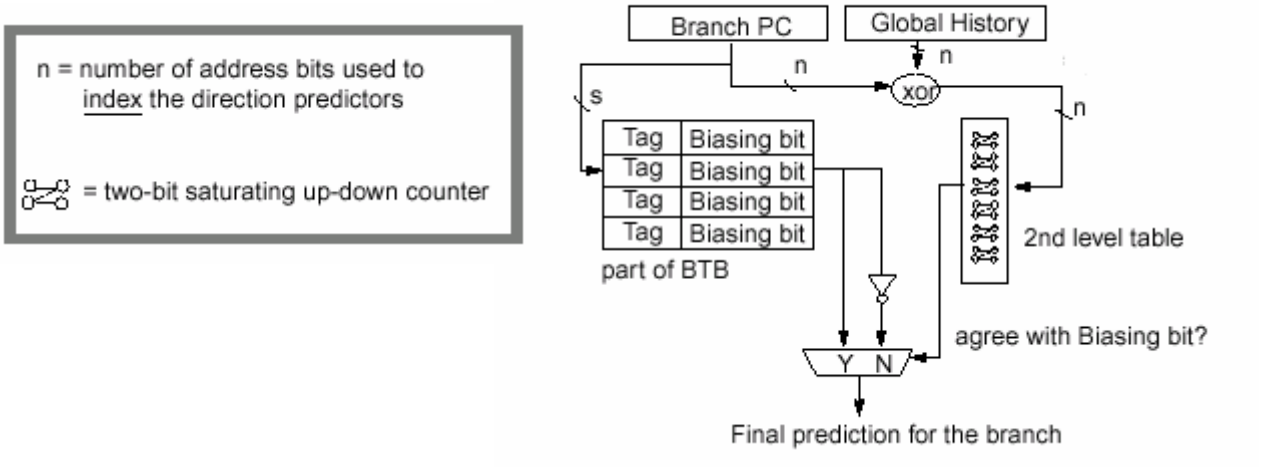




What is the benefit of using the xor function between the global history counter and the Branch PC (compare it with a predictor that uses the global history counter only)?

Question 2C (8%)

Consider the following Agree predictor:



Explain how does this predictor reduce negative branch interference?

Question 2D (15%)

The following code generates 3 branches:

```
1 For (i=0; i< N;i++){
2    If(X){
3      Op(x,Y);
4     }
5    If(Y){
6      Op(x,y)
7
8     }
9 }
```

Lets call the branches in lines 1,2 and 5 as branch I,X and Y respectively.
The sequence in the tables below belongs to these three branches.
You have to fill the following tables, for the Gshare and Agree predictors.

TABLE A

First two rows	**The branches and their true results.**
GH row	**Fill with the global history when accessing these branches in binary format. Use a history length of (n=3).**
PC row	**The lower n=3 bits of the program counter for the specific branch. These bits are used for the xor operation.**
Index row	**Fill with the index calculated for the 2nd level table.**

TABLE B: Gshare predictor table

Index column	**The index into the 2nd level table**
2nd column	**The initial state in each entry in the 2nd level table**
Next state	**Fill the state table with the next state for each branch. Use the following coding:** **ST** = Strongly taken , **WT** = Weakly Taken , **WNT**= Weakly Not Taken, **SNT**= Strongly Not Taken
Prediction row	**Fill the predictor prediction (0/1).**
Result row	**Fill with a T for a correct prediction and F for a wrong one**

TABLE C: Agree predictor table

Index column	**The index into the 2nd level table**
2nd column	**The initial state in each entry in the 2nd level table**
Next state	**Fill the state table with the next state for each branch. Use the following coding:** **SA** = Strongly Agree , **WT** = Weakly Agree , **WNT**= Weakly Not Agree , **SNT**= Strongly Not Agree
Bias Bit Row	**The corresponding bias bit for each branch.**
Prediction row	**Fill the predictor prediction (0/1).**
Result row	**Fill with a T for a correct prediction and F for a wrong one**

Hint: first fill the *prediction row* then the *result row* then the *state table*.

TABLE A

Branch		I	X	Y	I	X	Y	I	X	Y	I	X	Y	I	X	Y	I	X	Y	I	X	Y	I	X	Y
True Result		0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1
GH	000	000	000	001																					
PC		010	010	010	010	010	010	010	010	010	010	010	010	010	010	010	010	010	010	010	010	010	010	010	010
Entry		010	010																						

index		**TABLE B: Gshare Predictor table**																							
000	WNT																								
001	WNT																								
010	WNT	SNT																							
011	WNT																								
100	WNT																								
101	WNT																								
110	WNT																								
111	WNT																								
Prediction		0																							
Result		T																							

index		**TABLE C: Agree predictor table**																							
000	WA																								
001	WA																								
010	WA	SA																							
011	WA																								
100	WA																								
101	WA																								
110	WA																								
111	WA																								
Prediction		0																							
Bias Bit		0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0
Result		T																							

Question 2E (9%)

1) **In our example, which predictor is better Gshare or Agree predictor? Explain.**

_____

_____

_____

2) **What will be your answer to (1) if the bias bit for the branch X was 0 ?**

_____

_____

_____

3) **What will be your answer to (1) if the bias bit for the branch Y was 1 ?**

_____

_____

_____