VLSI Architecture Design Course (048853)
Final Exam
2 July 2000

Electrical engineering Department

Student name:_____**Student number**_____

This exam contains three questions.
The exam duration is 2:30 hours.
Please fill the answers ON THE EXAM forms.
If you need more space to write, please use the given empty exercise-book.

Good luck!

Q1	
Q2	
Q3	
Total	

Question 1 (34%)

You have recently been recruited by InDec Corp. to work on a second version of their "Alphium" processor. Alphium is a simple inorder processor with 5 pipeline stages. Its clock frequency is 600MHz and the chip power supply (Vcc) is 1.65V. Alphium achieves an IPC of 0.8 instructions per cycle on a specific benchmark. ($IPC = \frac{1}{CPI}$).

Your assignment is to evaluate the improvement in execution time, power and energy consumption for four proposed new versions of the Alphium processor. Here is a description of the alternatives:

- **Low Power Alphium (LPA):** This would be the same design as the original Alphium but it would be clocked at 400MHz to save power. The power supply (Vcc) would remain 1.65V and the achieved IPC would be 0.8 again.
- **Superscalar Alphium (SSA):** This would be a 4-way superscalar version of the Alphium. It would have the ability to issue up to 4 instruction per cycle. The power supply (Vcc) would remain 1.65V but the clock frequency would be reduced to 500MHz, due to the complexity of the issuing logic. The achieved IPC would be 2. Assume that the effective capacitance switched in the superscalar design would be 4 times that of the original.
- **Low Voltage Alphium (LVA):** This would be the same design as the original Alphium again but both the power supply and the clock frequency would be reduced. Power supply (Vcc) would be 1V and clock frequency would be 400MHz. The IPC would remain 0.8.
- **Low Voltage Superscalar Alphium (LVSSA):** This would be a 4-way superscalar version of with reduced power supply. The clock frequency would be 300MHZ, the power supply (Vcc) would be 1V and an IPC of 2 would be achieved. Assume that the effective capacitance switched in the superscalar design would be 4 times that of the original.

Here is some information that you may find useful:

Power: When we measure power for a system, we care about the maximum instantaneous power the system can consume. This is important as it determines the maximum current that the power supply must be able to supply to the system and the amount of heat that has to be removed from the system. In this question, the system refers to the processor only.

Energy: $E=C*Vcc^2$ is just the energy per transaction. This is not interesting. We care about the energy consumed from the power supply to execute a task (or perform some computation). Once the task is executed, the processor can be turned off and no further energy is needed. The energy per task determines how many tasks you can execute before the battery runs out.

Question 1 (cont)

A) [10 points] Complete the basic formulas that will allow you to compare the 5 alternatives to the original Alphium. You will need the formulas for execution time, power consumption, energy consumption for the benchmark, performance per power ratio for the benchmark and performance per energy ratio for the benchmark. (Hint: for your convenience, assume a given task has a predefined number of instructions - e.g., 10^6)

Execution Time =

Power =

Energy =

Performance per Power =

Performance per Energy =

Question 1 (cont)

B) [10 points] Fill in the two following tables. In each box write how the proposed new version compares to the original Alphium for that feature (e.g. $\frac{ExecTime_{new}}{ExecTime_{original}}, \frac{Power_{new}}{Power_{original}}$). Two fractional digits per entry are enough.

	IPC	Freq	Vdd	Relative ExecTime	Relative Power	Relative Energy
Alphium	0.8	600MHz	1.65V	1	1	1
LPA	0.8	400MHz	1.65V			
SSA	2.0	500MHz	1.65V			
LVA	0.8	400MHz	1.0V			
LVSSA	2.0	300MHz	1.0V			

	IPC	Freq	Vdd	Relative Performance/Power	Relative Performance/Energy
Alphium	0.8	600MHz	1.65V	1	1
LPA	0.8	400MHz	1.65V		
SSA	2.0	500MHz	1.65V		
LVA	0.8	400MHz	1.0V		
LVSSA	2.0	300MHz	1.0V		

Question 1 (cont)

C) [8 points] Circle the right answer (**true** or **false**), and explain your answers briefly:

1. Without any other changes, lowering the clock frequency of a processor leads to energy savings **True** **False**

2. Increasing performance always leads to energy wasting. **True** **False**

3. Lowering supply voltage can be combined with increasing performance. **True** **False**

4. Comparing processors using performance per Power is the same as using performance per Energy. **True** **False**

Question 1 (cont)

D) [6 points] For a **High end Workstation**, which of the above metrics (execution time, power, energy, performance/power, performance/energy) would you use to pick a processor? Why?

For a **Mobile computer**, which of the above metrics (execution time, power, energy, performance/power, performance/energy) would you use to pick a processor? Why?

For **Personal Digital Assistant (PDA)**, which of the above metrics (execution time, power, energy, performance/power, performance/energy) would you use to pick a processor? Why?

Question 2 (36%)

Branch Prediction

A) [8 points]

1. What is the minimal history length (in number of bits) required in a local predictor for a perfect prediction (once the predictor warms up) of each one of the following recurrent infinite sequences:

   010101 ... ___

   00110011 ... ___

   001001 ... ___

2. Find a pattern with a maximal length, which may be perfectly predicted by a local predictor with a history length of 3 bits (once the predictor warms up). Explain. (hint: you do not have to show the sequence!)

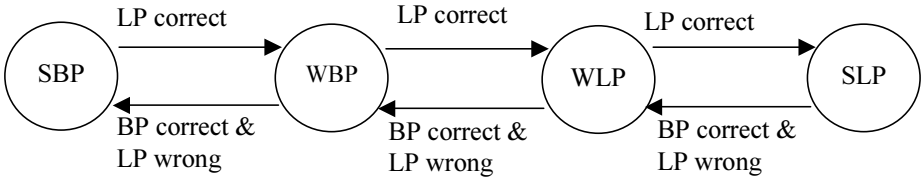
B) [20 points] The following three predictors are given:

1. A Bimodal Predictor (BP), made of an array of 2-bit saturating counters, initialized to weakly taken (2).

2. A Local Predictor (LP) with a history of length 3, were histories are initialized to 000, and each history points to an array of counter similar to that of predictor (BP).

3. A Chooser (Ch) which uses an array of counters similar to that of (BP) to choose between predictor (BP) and predictor (LP). The chooser counters are initialized to weakly point at predictor (BP). And it uses the following state machine:

Question 2 (Cont)



When both predictors are wrong the state doesn't change.

SBP = Strongly BP, takes Bimodal Prediction
WBP= Weakly BP, takes Bimodal Prediction

SLP = Strongly LP, Takes Local Prediction
WLP= Weakly BP, Takes Local Prediction

The sequence in the chooser table below belongs to a single branch.
You have to fill this table, for the BP and LP rows mark ***T*** if the predictor is correct and ***F*** if it is wrong.
For the Ch row mark SBP, WPB, WLP or SLP according to the chooser next state.
For the OP row mark the current overall prediction (1/0).
Use the Bimodal Predictor Log and Local Predictor Log tables to help you in the BP and LP predictions:

Chooser Table

		0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0
BP																						
LP																						
Ch	WBP																					
OP																						

Bimodal Predictor Log

		0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0
BP	WT																					
prediction																						

Local Predictor Log

History		0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0
000	WT																					
001	WT																					
010	WT																					
011	WT																					
100	WT																					
101	WT																					
110	WT																					
111	WT																					
Prediction																						

Question 2 (Cont)

C) [8 points] Some local predictors include 2 types of history - real history, and speculative history. The real history register tracks the branch behavior at retirement time of the branch. The speculative history register tracks the branch behavior at fetch time of the branch, based on the BTB prediction. On misprediction, in addition to flushing the pipe, the real history of every branch is copied to the speculative history. The following questions relate to the 2 types of history.

1. Which type of history would you use for predicting the branch at fetch time?

2. What is the benefit of having real history?

3. What is the benefit of having speculative history?

4. What implication does the pipe depth and Re-Order Buffer size have on the benefit of speculative history?

Question 3 (35%)

Out Of Order architecture.

A) [5 points] The Out of Order architecture has the following stages: Fetch, Issue & Rename, Execute, and Writeback. One of its strengths is that its ability to execute instructions in a different order than the programmer originally specified.

However, the Fetch and Issue stages of Out of Order the architecture are always handled in program order. Why?

B) [5 points] what is register renaming, why is it desirable, and how does it work?

Question 3 (Cont)

C) [5 points] Pipelined architectures have three different types of data hazards (data dependencies) with respect to Registers. Name and define them. For each type, give a short code sequence that illustrates the hazard and describe how an Out of Order architecture handles these hazard.

D) [5 points] why does an Out of Order architecture need branch Prediction?

Question 3 (Cont)

E. **[15 points]** The following program sums the elements of an array:

```
for (i=0;i<n;i++)
    sum += array[i];
```

```
        <Op>   <dst,src1,src2>
        LD     R3, sum
Loop:   LD     R2, array(R1)
        Add    R3,R3,R2
        Add    R1,R1,#1
        CMP    R1,#N            / N is loop limit
        JNE    Loop
        ST     [R5],R3
```

And consider an out of order machine with the following stages:

F = Fetch	Fetch 3 instructions per clock. Stalls when the decoder stalls.
D = Decode	Decode 3 instructions per clock. Stalls when renamer stalls.
N = Rename	Rename 3 instructions per clock. Stalls when there is no space in the ROB for the renamed instructions.
S = Schedule	Contains a ROB and RS that are practically infinite.
E = Execute	Contains 9 general execution units, and only one port for memory. Computed results can be consumed in the next cycle.
R = Retire	Retires up to 3 instructions -inorder- from the ROB.

The branch predictor of this machine predicts always taken on the JNE command in the code above.
The caches are Perfect - no cache misses.
Hint: there are enough execution units - they pose no limit, but only one load each cycle!

The following table contains the instruction flow inside this machine, you need to complete this table. **Assume N=3**:

- fill the instruction flow, use the letters: F D N S E R,
  Use "–" for stall, and X for a flush action.
- In case of branch misprediction, fill the instructions on the wrong path.
- For your help we added 2 columns after the instruction to help you track dependencies.

No.		Dst	Dep	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Notes
1	LD R1, Array			F	D	N	S	E	R															
2	LD R2, 0(R1)			F	D	N	S																	
3	Add R3,R3,R2			F	D	N	S																	
4	Add R1,R1,#4				F	D	N																	
5	CMP R1,#N				F	D	N																	
6	JNE Loop				F	D	N																	
7	LD R2, 0(R1)					F	D																	
8	Add R3,R3,R2					F	D																	
9	Add R1,R1,#4					F	D																	
10	CMP R1,#N						F																	
11	JNE Loop						F																	
12	LD R2, 0(R1)						F																	
13	Add R3,R3,R2																							
14	Add R1,R1,#4																							
15	CMP R1,#N																							
16	JNE Loop																							
17																								
18																								
19																								
20																								
21																								
22																								
23																								
24																								
25																								
26																								
27																								
28																								
29																								
30																								
31																								
32																								
33																								
34																								
35																								
36																								
37																								

Question 3 (Cont)

E) [bonus] what is the maximum number of ROB entries used in the previous program?

How a smaller ROB would affect the execution of the previous program?