Final exam

VLSI Architecture Design Course

August 4, 1998

Electrical engineering Department

Student name:_____Student number_____

date:_____

This exam contains three questions.
The exam duration is 2:30 hours.
Please fill the answers ON THE EXAM forms. Additional support to your findings you should add in your exercise-book.

Good luck!

Question 1 (30%):

Consider a CPU (see figure 1) with 16 general purpose execution units, and on die unified (Instruction/Data) L1 cache. The CPU micro-architecture supports Out Of Order execution, and its internal to external frequency ratio is f1/f2=20.

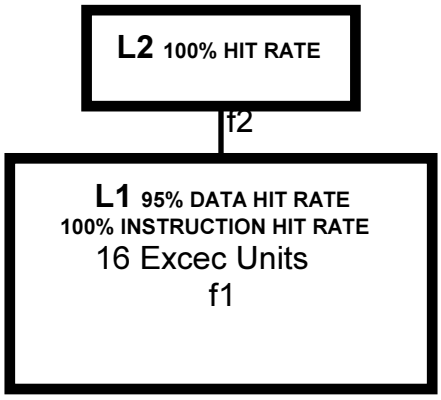

Figure 1.

Assumptions:

- The internal L1 cache hit rate is 95% for data and 100% for instructions. The access latency (for cache line) to L2 is 4 cycles, and the L2 hit rate is 100%.
- The program's average data access is 1 access per 3 instructions.

Note: You can make any reasonable assumptions regarding other CPU and application parameters (please state them).

You are asked to draw the CPU IPC (Instruction Per Cycle) vs Time curve based on the everage data you have.
You should identify and explain what are the reasons for the curve shape.

A. Identify the different curve segments and explain their cause ?

B. For each of the four curve segment please explain:

1 what are the parameters that impack the length of this segment?

2 What are the CPU microarchitecture (improvement) features that can shorten this interval?

C. Is this curve periodic, if so what is the period ?

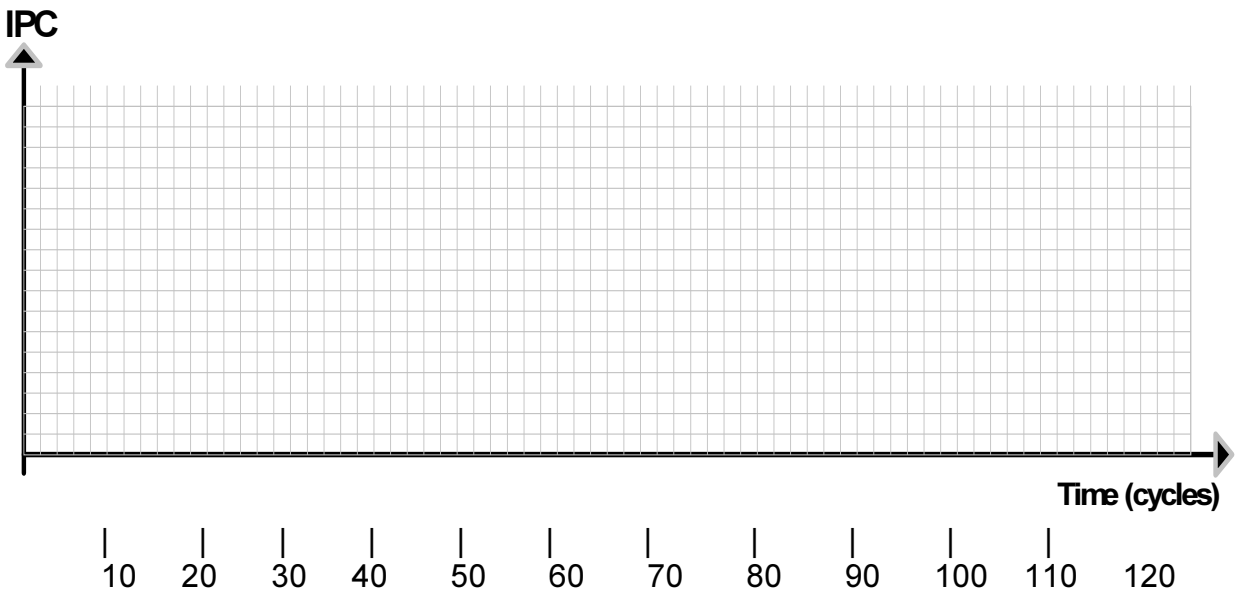

Figure 1 - CPU IPC vs Time curve

Question 2 (35%):

Branch Prediction

A.

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 ... ___
   001001 ... ___
   00110011 ... ___

2 Find a sequence with a maximal cycle length, which may be perfectly predicted by a local predictor with a history length of 3 bits (once the predictor warms up). Explain.

B. 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).

For each one of the bits of the sequence bellow (which was generated by a single branch), for each one of the predictors, mark 1 if the predictor is correct and 0 if it is wrong, For the Ch row mark BP or LP according to the chosen predictor and OP is the overall prediction (1/0).

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

C. Design a "loop predictor", which will be used to predict the outcome (Taken/Not Taken) of a new loop instructions:

   loop Ri, address ; if (Ri== 0) then PC = address else Ri- -.

Requirements:
The predictor should have a prediction rate of ~98%.
The predictor must use a minimal number of bits (every bit counts ! don't add bits to improve prediction rate beyond the requested accuracy).
The predictor should support up to 4 nested loop instructions.
Assume an address space of 32 bits and that the initial value of Ri > 0.

You should specify the role of each filed in your predictor, the number of bits in each field and the number of entries (Explain all your results and assumptions).

Question 3 (35%):

This question will confront the issue of a partially decoded instruction cache in a variable instruction length machine.
Figure 2 depicts machine A with a 5 stage pipe, and 4 cycle latency to the L2 cache (i.e., Instruction miss in L1 cache and hit in L2 cache, causes a 4 cycle penalty). Machine B has a shorter pipe, one pipe stage was reduced relative to machine A. The instruction length decode stage was removed, and the instruction length information is kept in the L1 instruction cache. On machine B, the length decode stage is between the instruction L1 cache and L2 cache, thus increasing the instruction L1 miss penalty to 5 cycles.

A. Propose a structure for the instruction L1 cache of machine B, with minimal area impact.

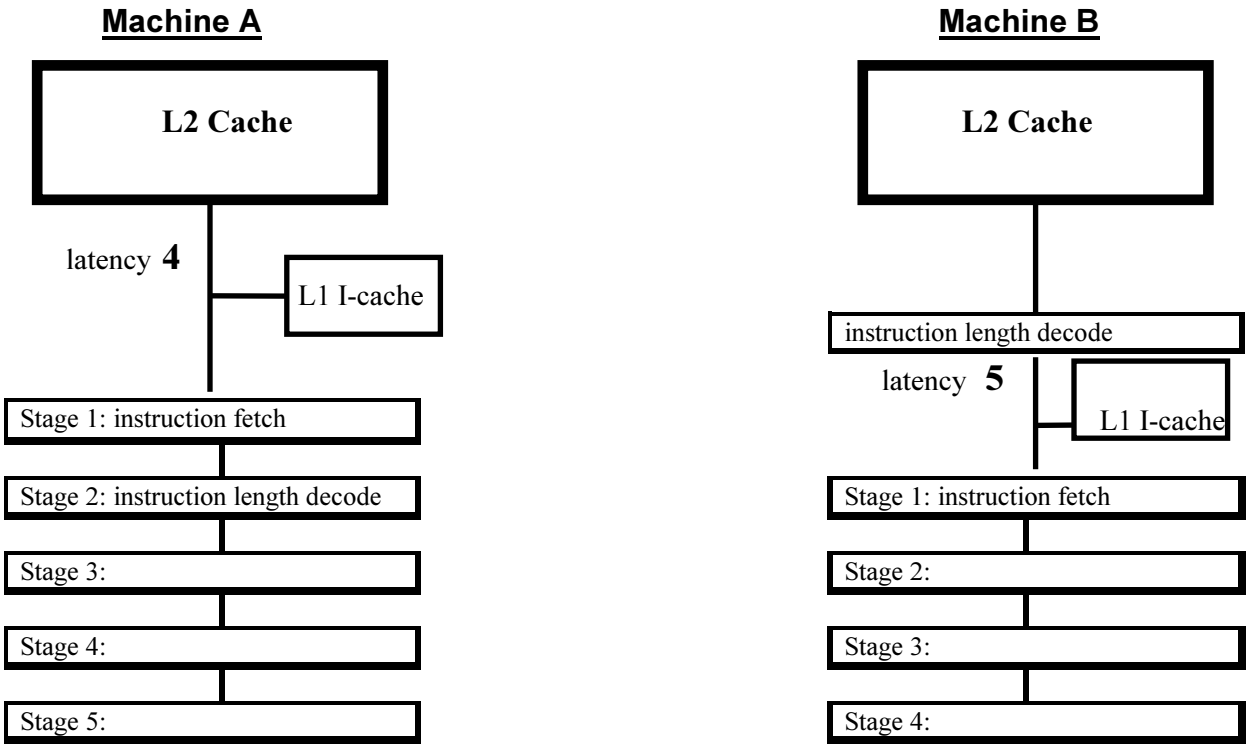

Figure 2 - Machine A has a length decode pipe stage, machine B has a partially decoded cache.
The following code section demonstrates a problem with machine B's pipeline for variable length Instruction Set Architectures (ISA). Assume value of R1 is 1 before this code begins. Assume a 16 byte instruction cache line size.

Address		Instruction	Comments
1F7h	Loop:	ADD R1, R1, #1	Increment R1 by 1
1F9h		AND R1, R1, #00000001	Mask R1 with the value 1
1FEh		JZ R1, Even	If R1=0 jump to label Even
200h		ADD R3, R3, #1	Increment R3 by 1
202h		JMP Odd	Jump to label Odd
204h	Even:	SUB R2, R2, #1	Decrement R2 by 1
206h	Odd:	JNZ R2, Loop	If R2!=0 jump to label Loop

Figure 3

B. Describe the problem of a length decoded instruction cache for variable length ISAs, as reflected by the above loop?

C. How many fetch requests are required in order to fetch all the instructions in the code above? Note the address of each instruction, and the address 200h begins on a new cache line.

Machine C (Figure 4) has a Trace cache instead of a partially decoded cache. The trace cache lines can contain up to 3 instructions per line. A trace has only one entry point - the beginning of the trace. Every backward taken branch ends the current trace (i.e. the instruction after a backward taken branch begins a new trace).

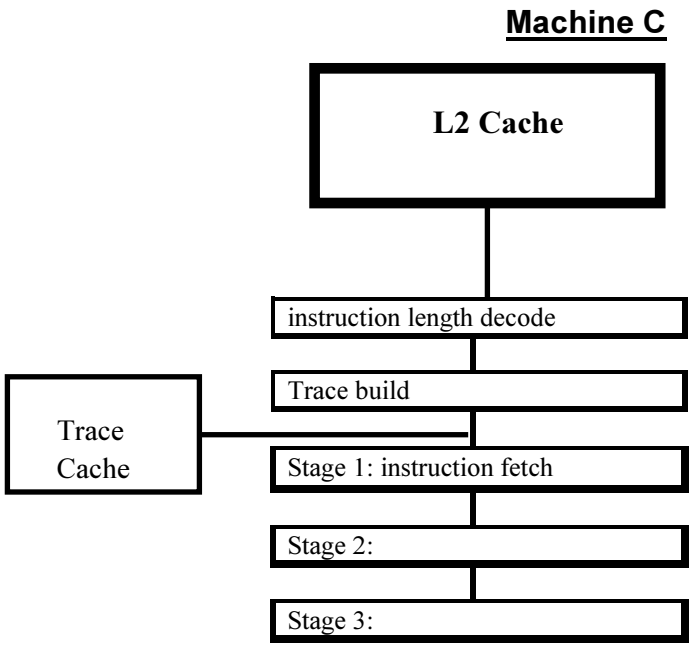

Figure 4 - Machine C has a Trace Cache

D. How would the code above (Figure 3) be mapped into the Trace Cache assuming the loop runs 10 times? Write the instructions into the TC lines below. Note the lines in which a new trace begins. Use as many lines that you need.

set	Trace head indication	Trace Address Tag	Instruction 0	Instruction 1	Instruction 2
0					
1					
2					
3					
4					
5					

Figure 5 - Trace Cache