資訊工程學系 計算機結構

姓名： 學號：

1. (9%) What functions should an integrated instruction fetch unit integrate? Please explain the reasons.

   A. Integrated branch prediction

   i. branch predictor is part of instruction fetch unit and is constantly predicting branches

   B. Instruction prefetch

   i. Instruction fetch units prefetch to deliver multiple instruct. per clock, integrating it with branch prediction

   C. Instruction memory access and buffering Fetching multiple instructions per cycle:

   i. May require accessing multiple cache blocks (prefetch to hide cost of crossing cache blocks)

   ii. Provides buffering, acting as on-demand unit to provide instructions to issue stage as needed and in quantity needed

2. (8%) Please describe data hazards between which two instructions.

   L.D F0, 10(R1)
   ADD.D F0, F4, F6
   SUB.D F10, F0, F8
   MUL.D F2, F8, F10
   DIV.D F8, F2, F6
   S.D F2, 0(R1)

RAW:

L.D and SUB.D
ADD.D and SUB.D
SUB.D and MUL.D
MUL.D and DIV.D
MUL.D and S.D

WAW:

L.D and ADD.D

WAR:

MUL.D and DIV.D
SUB.D and DIV.D

3. (9%) There are several types of limits to the gains that can be achieved by loop unrolling. Please illustrate three types and explain them in detail.
   - **A. Decrease in amount of overhead amortized with each extra unrolling**
   - **B. Growth in code size**
     - **i. For larger loops, concern it increases the instruction cache miss rate**
   - **C. Compiler**
   - **D. Register pressure: potential shortfall in registers created by aggressive unrolling and scheduling**

4. (6%) Please explain why Dynamic Branch Prediction is better than Static Branch Prediction.

The prediction will depend on the behavior of the branch at run time and the branch changes its own behavior during execution.

5. (6%) Please explain the difference between Basic Branch Prediction Buffer and Correlating Branch Predictors.

- **The previous schemes use only the recent behavior of a signal branch to predict the future behavior of the branch.**
- **Correlating Branch Predicator**
  - **Branch predictors that use the behavior of other branches to make a predication.**

6. (6%) Please explain why an exception is **imprecise**.
   - A. if the processor state when an exception is raised does not look exactly as if the instructions were executed sequentially in strict program order.
     - i. The pipeline may have *already completed* instructions that are *later* in program order than the instruction causing the exception.
     - ii. The pipeline may have *not yet completed* some instructions that are *earlier* in program order than the instruction causing the exception.

7. (6%) Please explain Temporal Locality and Spatial Locality.

Temporal Locality (Locality in Time): If an item is referenced, it will tend to be referenced again soon (e.g., loops, reuse)

Spatial Locality (Locality in Space): If an item is referenced, items whose addresses are close by tend to be referenced soon (e.g., straight-line code, array access)

8. (9%) For the original VLIW model, there were technical and logicastical problems that make the approach less efficient. Please illustrate three problems and explain them in detail.

A. Increase in code size

i. generating enough operations in a straight-line code fragment requires ambitiously unrolling loops

ii. **whenever VLIW instructions are not full, unused functional units translate to wasted bits in instruction encoding**

B. **Operated in lock-step; no hazard detection HW**

i. A stall in any functional unit pipeline caused entire processor to stall, since all functional units must be kept synchronized

ii. Compiler might prediction function units, but caches hard to predict

C. Binary code compatibility

i. **Pure VLIW ＼ different numbers of functional units and unit latencies require different versions of the code**

9. (8%) Assume that there are 16K bits in the Correlating Branch Prediction Buffer. Please derive the numbers of entries in (0,1), (3,1), (2,2), and (12,2) predicators.

A. (0,1): 16K entries

B. (3,1): 2K entries

C. (2,2): 2K entries

D. (12,2): 2 entry

10. (4%) Please explain why Branch Target Buffers need to keep Branch target information.

Branch target calculation is costly and stalls the instruction fetch.

11. (6%) What techniques can decrease the CPI to less that one? Please provide three kinds of techniques.

Dynamic Scheduling
Multiple Issue
Speculation

12. (8%) In Hardware-Based Speculation design, a specific reorder buffer is needed. Please describe the purpose of the buffer as detail as possible.
    - **A. In Tomasulo's algorithm, once an instruction writes its result, any subsequently issued instructions will find result in the register file**
    - **B. With speculation, the register file is not updated until the instruction commits**
        - **i. (we know definitively that the instruction should execute)**
    - **C. Thus, the ROB supplies operands in interval between completion of instruction execution and instruction commit**
        - **i. ROB is a source of operands for instructions, just as reservation stations (RS) provide operands in Tomasulo's algorithm**
        - **ii. ROB extends architectured registers like RS**

13. (20%) Please write the contents of Instruction Status, Reservation Stations, and Register result status at clock cycle 10.

Instruction status:

Instruction		j	k	Issue	Exec Comp	Write Result
LD	F6	34+	R2	1	2	3
LD	F2	45+	R3	2	3	4
MULTD	F0	F2	F4	3	10	
SUBD	F8	F6	F2	4	6	7
DIVD	F10	F0	F6	5		
ADDD	F6	F8	F2	6	9	10

	Busy	Address
Load1	No	
Load2	No	
Load3	No	

Reservation Stations:

Time	Name	Busy	Op	S1 Vj	S2 Vk	RS Qj	RS Qk
	Add1	No					
	Add2	No					
	Add3	No					
0	Mult1	Yes	MULTD	R(F2)	R(F4)		
	Mult2	Yes	DIVD		R(F6)	Mult1	

Register result status:

Clock		F0	F2	F4	F6	F8	F10	F12	...	F30
10	*FU*	Mult1					Mult2			

Note：FP multiply (6 EX cycles)、FP add (2 EX cycles)、and FP divide (12 EX cycles)、load (1 EX cycles)