CS 220 | 328034 ID | Isaac Sullivan

BRADLEY University

CS 220 Computer Architecture

HW 09 – CPU Performance and Pipelining

Fall 2023

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS

Part 0: Reading

- **Chapter 13** - Instruction Sets: Characteristics and Functions
- **Chapter 16** - Processor Structure and Function
- **Chapter 17** - Reduced Instruction Set Computers
- **Chapter 18** - Instruction-Level Parallelism and Superscalar Processors

Part 1: Questions on Instruction Sets [40 pts]

Question 1: Machine instruction

Answer the following questions. **[10 pts]**

A. What are the typical elements of machine instruction?
B. What types of locations can hold source and destination operands?
C. If an instruction contains four addresses, what might be the purpose of each address?
D. What types of operands are typically in machine instruction sets?
E. Why is the transfer of control instructions needed?

A.	Operation code, source operand reference, result operand reference, and next instruction reference
B.	Registers and memory
C.	There will be two operands, one result, and the address of the next instruction in the addresses
D.	Addresses, numbers, characters, and logical data
E.	1. Computers need to be able to execute instructions more than once 2. The majority of programs require decision making of some kind 3. Splitting tasks up into simpler computations makes the computer process faster

Question 2

List and briefly describe **five** important **instruction set design issues. [10 pts]**

1.	Operation repertoire - How many and which operations to provide, and how complex operations should be
2.	Data types - The various types of data upon which operations are performed.
3.	Instruction format - Instruction length (in bits), number of addresses, size of various fields, and so on
4.	Registers - Number of processor registers that can be referenced by instructions, an
5.	Addressing: The mode or modes by which the address of an operand is specified

Question 3: On packed decimal

A. What is the relationship between the IRA character code and the packed decimal representation? **[5 pts]**

A	For the IRA bit pattern 011XXXX, the digits 0 through 9 are represented by their binary equivalents, 0000 through 1001, in the rightmost 4 bits. This is the same code as packed decimal.

B. For each of the following packed decimal numbers, show the decimal value. **[5 pts]**

B	
Packed decimal numbers	**Decimal value**
0111 - 0011 - 0000 - 1001	7309
0101 - 1000 - 0010	582

0100 - 1010 - 0110	4106

Question 4: Shifting

Convert the following hexadecimal numbers to their equivalents.

A. What is the difference between a logical shift and an arithmetic shift? **[5 pts]**

A	Logic shift is a shift of bits either left or right with vacant bits filled up with zeros. Arithmetic shift is a shift of bits either left or right with a sign bit preserved if possible.

B. Complete the table. **[5 pts]**

B					
Bit pattern	**value**	**Arithmetic left shift**	**value**	**Logical left shift**	**value**
0 - 0000	00000	0<-0000<-0	00000	0<-0000<-0	00000
0 - 0011	00011	0<-0011<-0	00110	0<-0110<-0	01100
0 - 1000	01000	0<-1000<-0	00000	0<-0000<-0	00000
1 - 1000	11000	1<-1000<-0	10000	1<-0000<-0	00000
1 - 1101	11101	1<-1101<-0	11010	1<-1010<-0	10100
1 - 1111	11111	1<-1111<-0	11110	1<-1110<-0	11100

Part 2: CPU Improvements [60 pts]

Question 1

Why is a two-stage instruction pipeline unlikely to cut the instruction cycle time in half, compared with the use of no pipeline? **[5 pts]**

The execution time will generally be longer than the fetch time. Execution will involve reading and storing operands and the performance of some operation. Thus, the fetch stage may have to wait for some time before it can empty its buffer.

Question 2

List and briefly explain 5 ways in which an instruction pipeline can deal with conditional branch instructions. **[5 pts]**

Multiple streams: to replicate the initial portions of the pipeline and allow the pipeline to fetch both instructions, making use of two streams.

Prefetch branch target - When a conditional branch is recognized, the target of the branch is prefetched, in addition to the instruction following the branch. This target is then saved until the branch instruction is executed. If the branch is taken, the target has already been prefetched.

Loop Buffer - A loop buffer is a small, very-high-speed memory maintained by the instruction fetch stage of the pipeline and containing the n most recently fetched instructions, in sequence.

Branch prediction - A prediction is made whether a conditional branch will be taken when executed, and subsequent instructions are fetched accordingly.

Delayed branch - It is possible to improve pipeline performance by automatically rearranging instructions within a program, so that branch instructions occur later than actually desired.

Question 3

Briefly define the following terms: **[10 pts]**

True data dependency	Can fetch and decode second instruction in parallel with first but can NOT execute second instruction until first is finished
Procedural dependency	Can not execute instructions after a branch in parallel with instructions before a branch
Resource conflicts	Two or more instructions waiting for the same resource
Output dependency	The Case when variable content is updated by an instruction and then the variable content is updated by another instruction following it
Anti Dependency	The case when a variable content is used by an instruction and then the variable content is updated in another instruction following it

QUESTION 4: PARALLELISM, SUPERSCALAR, SUPER-PIPELINING

A. What is the essential characteristic of the superscalar approach to processor design? **[5 pts]**

Can execute multiple independent instructions at the same time

B. Compare and contrast **[5 pts]**

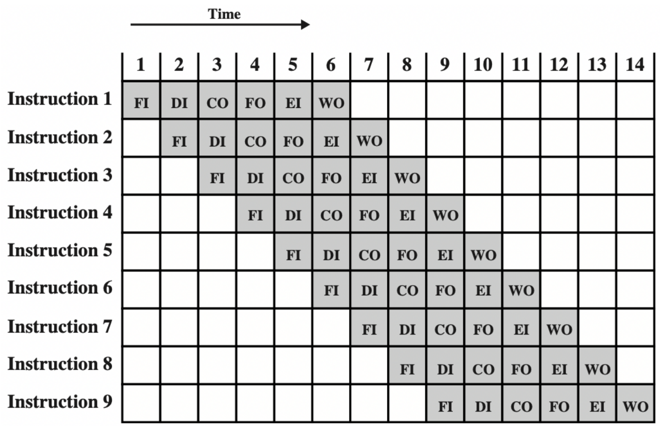
Super-scalar	Super-pipelining	Parallelism
common instructions—integer and floating-point arithmetic, loads, stores, and conditional branches—can be initiated simultaneously and executed independently	functions performed in each stage can be split into two non-overlapping parts, and each can execute in half a clock cycle	achieved by enabling multiple instructions to be at different stages of the pipeline at one time

Question 5: Instruction PIPELINING

Assume a pipeline with four stages: **[10 pts]**

- FI: fetch instruction
- DA: decode instruction and calculate addresses
- FO: fetch operand
- EX: execute

Draw a diagram similar to Figure 16. 10 (shown as below) for a sequence of 7 instructions to show how many time units are now needed.

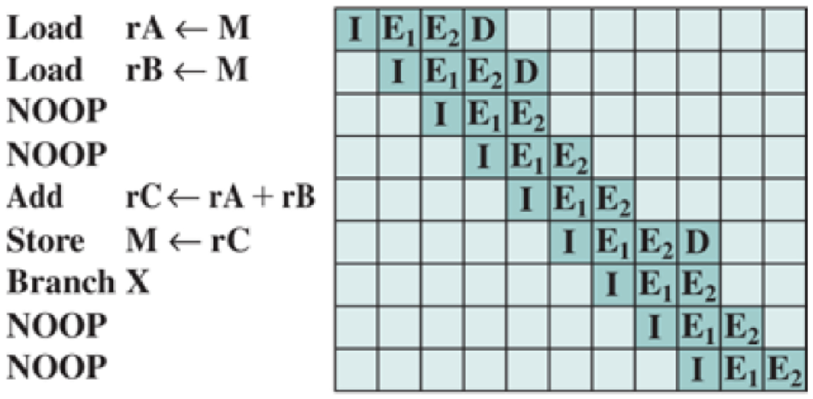

Time

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Instruction 1	FI	DI	CO	FO	EI	WO								
Instruction 2		FI	DI	CO	FO	EI	WO							
Instruction 3			FI	DI	CO	FO	EI	WO						
Instruction 4				FI	DI	CO	FO	EI	WO					
Instruction 5					FI	DI	CO	FO	EI	WO				
Instruction 6						FI	DI	CO	FO	EI	WO			
Instruction 7							FI	DI	CO	FO	EI	WO		
Instruction 8								FI	DI	CO	FO	EI	WO	
Instruction 9									FI	DI	CO	FO	EI	WO

#	1	2	3	4	5	6	7	8	9	10
I1	FI	DA	FO	EX						
I2		FI	DA	FO	EX					
I3			FI	DA	FO	EX				
I4				FI	DA	FO				
I5					FI	DA				
I6						FI				
I7							FI	DA	FO	EX

Question 6

Reorganize the code sequence in Figure below to reduce the number of NOOPs **[10 pts]**



Load rA <- M
Load rB <- M
NOOP
Branch X
Add rC <- rA + rB
Store M <- rC

Question 7 [10pts]

Consider the following assembly language program: **[10 pts]**

```
I1: Move R3, R7         /R3 ← (R7)/
I2: Load R8, (R3)       /R8 ← Memory (R3)/
I3: Add R3, R3, 4       /R3 ← (R3) + 4/
I4: Load R9, (R3)       /R9 ← Memory (R3)/
I5: BLE R8, R9, L3      /Branch if (R9) > (R8)/
```

This program includes WAW, RAW, and WAR dependencies. Show these.

WAW: R3(I1, I3)
RAW: R3(I1, I2), R3(I1, I3), R3(I3, I4), R8(I2, I5), R9(I4, I5)
WAR: R3(I2, I3)