1 Computer Languages 1

Get ready!

1 Before you read the passage, talk about these questions.

1 What does a programmer do?

2 How does translation affect computer function?

Chapter 3 **27**

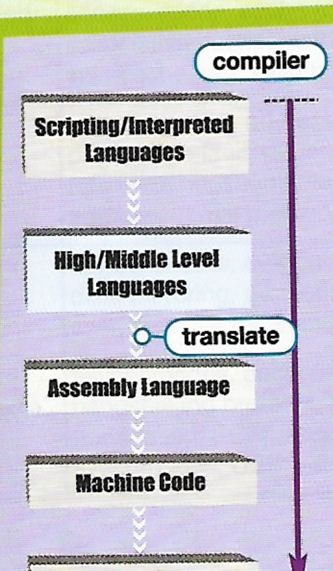
How computers process information

Computers are constantly processing large amounts of information. Operating a computer involves sending and receiving complex sets of instructions. Computers have their own language, called **machine language**. Machine language is made up of **binary digits** that are represented by the numbers 0 and 1. Every possible computer operation is encoded with different combinations of these two numbers.

However, **programmers** usually do not send commands in machine language. They write software in **human-readable programming languages**. This allows programmers to write software quickly and efficiently. These languages, like **C** and **Java**, are more compatible with the way humans think. However, computers still require instructions in machine language.

Systems software facilitates this communication within the computer. A **compiler** is a software component that **translates** human-readable language into an **assembly language**. This language is simpler than a human-readable language. But it still uses letters and words. The computer needs an **assembler** to turn those instructions into the binary translation.

For example the programmer might write the command "A + B." Then, a compiler converts it into an assembly language: "Add A,B." Finally, an assembler translates it into machine language: "1000110010100000." The computer uses these instructions to perform the command.



Reading

2 Read the textbook chapter. Then, choose the correct answers.

1 What is the main idea of the chapter?

- **A** how to write a computer program
- **B** recent changes in computer software
- **C** how computers send and receive information
- **D** a comparison of different programming theories

2 Which language does NOT need translation before a computer reads it?

- **A** assembly language
- **B** machine language
- **C** Java
- **D** human-readable programming language

3 What is true of binary digits?

- **A** They are also called assembly language.
- **B** They are most commonly used by programmers to write instructions.
- **C** They are not complex enough for most computer operations.
- **D** They are used to encode all computer functions.

Vocabulary

3 Match the words (1-8) with the definitions (A-H).

1 __ assembly language
2 __ Java
3 __ C
4 __ programmer
5 __ machine language
6 __ compiler
7 __ assembler
8 __ translate

A a program that converts complicated operations into simpler letters and words

B a program that changes written instructions into a binary translation

C a set of instructions written in numerical form

D a human-readable programming language that is object-oriented and simple

E to convert something from one form to another

F written instructions that have not been converted to a binary translation

G a person who writes and develops software

H a human-readable programming language that is focused on procedures

4 **Write a word that is similar in meaning to the underlined part.**

1 Computers can only understand commands written in a system that uses a combination of zeros and ones.
_ i _ _ r _ d _ g _ _ s

2 In order to write programs quickly and efficiently, we use words that are designed to send instructions to computers.
h _ _ a _ - r _ _ _ a b _ _ p _ o _ _ _ m m _ _ g _ a n _ u _ g _ s

3 Computers are built with a program that provides basic functions in order to facilitate operation.
s _ _ _ e _ s s _ f _ w _ _ e

5 **Listen and read the textbook chapter again. Why is human-readable programming language useful?**

Listening

6 **Listen to a conversation between a student and an instructor. Mark the following statements as true (T) or false (F).**

1 __ The man completed a Java assignment.

2 __ The woman recommends strategies for learning different languages.

3 __ According to the woman, having many languages helps engineers build faster computers.

7 **Listen again and complete the conversation.**

Student: I'm having a hard time 1 _______ _______ _______ straight. There are so many.

Instructor: There are quite a few. Which ones are you having trouble with?

Student: All of them. I think it's 2 _______ _______ _______ all these languages.

Instructor: I see where you're coming from. It's a lot to learn.

Student: But why are there 3 _______ _______ _______ languages?

Instructor: The first reason is that it makes 4 _______ _______ _______.

Student: 5 _______ _______ _______?

Instructor: The computer only understands 6 _______ _______ _______, or binary digits. It would take programmers a long time to write programs in binary format.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

I'm having a hard time understanding ...

There are quite a few ...

How does that help?

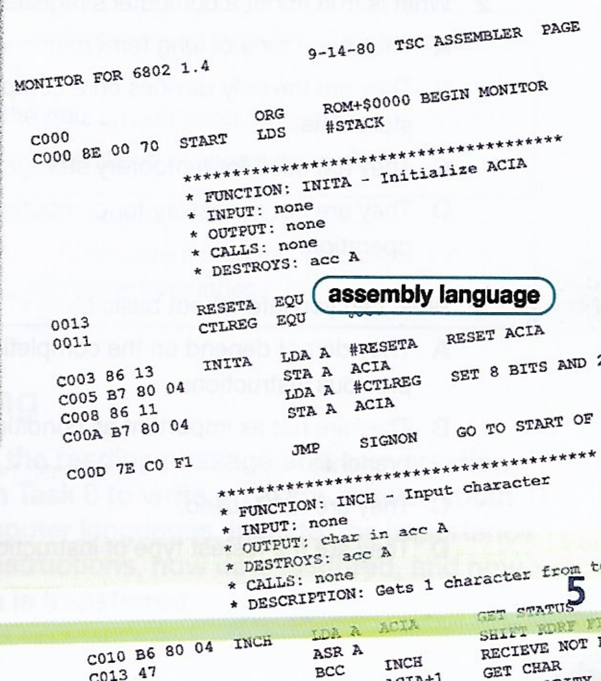
Student A: You are a student. Talk to Student B about:

- the different kinds of programming languages
- why you are confused
- the uses of different languages

Student B: You are an instructor. Talk to Student A about the uses of different programming languages.

Writing

9 **Use the reading passage and conversation from Task 8 to write a student's notes on programming languages. Include: at least two different programming languages, their functions, and their benefits.**

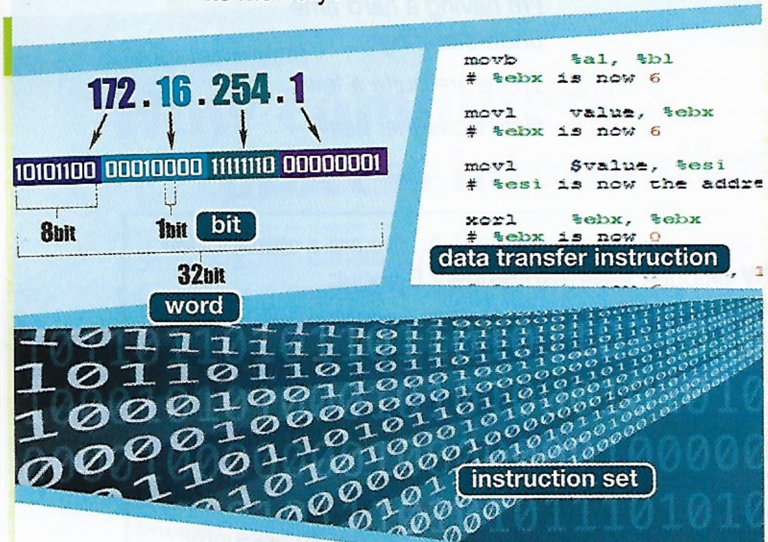


Get ready!

1 Before you read the passage, talk about these questions.

1 What determines a computer's actions?

2 How does a computer retrieve information from its memory?



Reading

2 Read the textbook chapter. Then, choose the correct answers.

1 What is the main idea of the article?
- **A** the history of computer languages
- **B** how a computer stores and transfers data for use
- **C** the benefits of different programming languages
- **D** a comparison of typical instructions that computers must execute

2 What is true about a computer's registers?
- **A** They are a type of long term memory.
- **B** They are the only devices on a computer that store data.
- **C** They are used for temporary storage.
- **D** They are not necessary for computer operation.

3 What can you infer about basic blocks?
- **A** They do not depend on the completion of previous instructions.
- **B** They are not as important as conditional branches.
- **C** They are rarely used.
- **D** They are the fastest type of instruction.

Chapter 4

How computers Process Information

A computer's function is to follow specific commands, or **instruction sets**. However, processing multiple commands can be time-consuming. The **stored-program concept** allows **instructions** to be efficiently stored in machine language.

Storing instructions in the machine's **register** allows information to be accessed more quickly. Registers are made of **bits**, or binary digits. Since bits are so small, they are typically used in groups. A **word** is the most commonly used grouping of bits. It is often made up of 32 or 64 bits, depending on the system. The speed at which **data** is accessed depends upon the available number of bits.

Information stored in the long-term memory of the computer must also be available. A **data transfer instruction** allows data to transfer from the memory to the registers. Then it becomes easily accessible and can be retrieved more quickly. The data must have a destination, or **address**, that is also sent by the data transfer instruction.

When data is put into the computer, various instructions are executed. A **basic block** is the most fundamental set of instructions. More complex sets include **conditional branches**. Unlike basic blocks, these can only execute after previous instructions are complete.

Vocabulary

3 Match the words (1-8) with the definitions (A-H).

1	__ stored-program concept	5	__ bit
2	__ basic block	6	__ register
3	__ data transfer instruction	7	__ word
4	__ conditional branch	8	__ instruction

- **A** a series of instructions that does not have branches
- **B** a command that is part of a computer language
- **C** an action that is only completed if other actions are completed first
- **D** the theory that instructions can be stored as numbers in the computer's memory
- **E** a part of the computer's hardware that temporarily stores instructions
- **F** an operation on a computer that moves data from one type of storage to another
- **G** a standard group of units of information
- **H** the smallest unit of information on a computer

4 Read the sentence pairs. Choose which word or phrase best fits each blank.

1 **instruction set / word**

A A(n) ________________ is a series of commands.

B A(n) ________________ can be part of a command.

2 **data / address**

A The information is transferred to a particular ______________.

B ______________ from the long term memory goes to a register for temporary use.

5 Listen and read the textbook chapter again. What is the importance of a data transfer instruction?

Listening

6 Listen to a conversation between an instructor and a student. Mark the following statements as true (T) or false (F).

1 __ The speakers discuss the woman's score on an exam.

2 __ According to the woman, instructions help to control the computer's hardware.

3 __ The man identifies stored-program concept incorrectly.

7 Listen again and complete the conversation.

Instructor: All right. Tell me what you know about instructions.

Student: Those are the computer's language. **1** _______ _______ _______ to the computer in order to control its hardware.

Instructor: Good. Next question. Why is the **2** _______ - _______ _______ _______?

Student: It's the idea that **3** _______ _______ _______ _______ in the computer's memory.

Instructor: Why is that necessary?

Student: It makes **4** _______ _______ _______, right?

Instructor: Correct. And how does the machine retrieve data **5** _______ _______ _______?

Student: Let's see. That requires a data transfer instruction. Then data moves **6** _______ - _______ _______ to the registers.

Speaking

8 With a partner, act out the roles below based on Task 7. Then, switch roles.

USE LANGUAGE SUCH AS:

Don't forget to ... / I'll remember ...

Please explain ...

Student A: You are an instructor. Talk to Student B about:

- the quiz on computer languages
- the definitions of important terms

Student B: You are a student. Talk to Student A about computer languages.

Writing

9 Use the reading passage and conversation from Task 8 to write an exam answer about computer languages. Include: the importance of instructions, how data is stored, and how data is transferred.

3 Arithmetic for Computers 1

Get ready!

1 Before you read the passage, talk about these questions.

1 What is the function of arithmetic in computer processes?

2 Why should programmers know both programming and machine languages?

Arithmetic in Computers

Bits determine the fundamental functions of computers. Each word contains a set number of bits, and each combination corresponds to a number. These numbers are represented in one of several **number bases.** Although we typically think in **base 10**, computers function best in **base 2**. Each number set is **subscripted** with a ten or a two to indicate to whether it is decimal or binary.

Computers use arithmetic to signal various functions. Computers must distinguish between positive and negative numbers in order to operate the hardware. A **signed number** refers to a number that has a negative or positive sign. An **unsigned number** does not have a sign, so it must be zero or a positive number. **Two's complement** is a representation of signed binary numbers that uses **leading 0's** and **leading 1's.** If the word has a leading 0, it is positive. If it has a leading 1, it is negative.

The hardware is programmed to test the **sign bit** for positivity or negativity. The sign bit is also the **most significant bit,** which is farthest to the left. The bit with the highest value is the digit to the right of the sign bit. The rightmost bit is the **least significant bit**, or the bit with the lowest value.

Reading

2 Read the textbook chapter. Then, choose the correct answers.

1 What is the main idea of the article?

- A benefits of different number systems
- B the way numbers are represented in programming
- C how to translate information between number systems
- D practical applications for computer arithmetic

2 Which of the following is NOT true of the two's complement representation?

- A It uses binary numbers.
- B It can contain a leading 0.
- C It features a sign bit.
- D It occurs in base 10.

3 What tells a computer whether a number is positive or negative?

- A sign bit
- B number base
- C least significant bit
- D subscript

Vocabulary

3 Match the words (1-8) with the definitions (A-H).

1 __ two's complement	5 __ base 10
2 __ sign bit	6 __ signed number
3 __ base 2	7 __ subscript
4 __ number base	8 __ unsigned number

- A the leading digit that is tested by the hardware to indicate whether a number is positive or negative
- B the representation of binary numbers using leading 0 and leading 1
- C a number that does not have a negative or a positive sign
- D to add a distinguishing number or character to a larger number or character
- E the indication of how many numbers are used in a certain system
- F a number system, also called the decimal system, that uses the numbers 1 through 10
- G a number that is either positive or negative
- H a number system, also called the binary system, that uses the numbers 0 and 1

4 **Read the sentence pairs. Choose which word or phrase best fits each blank.**

1 **leading 0 / leading 1**

A Since this word has a ________, it is negative.

B On the other hand, a ________ indicates that the number is positive.

2 **most significant bit / least significant bit**

A The ________ is also known as the sign bit.

B The bit farthest to the right is the ________.

5 **Listen and read the textbook chapter again. What is the difference between base 10 and base 2?**

Listening

6 **Listen to a conversation between an instructor and a student. Mark the following statements as true (T) or false (F).**

1 __ The woman identifies the number bases incorrectly.

2 __ According to the man, positive and negative numbers can be difficult to identify.

3 __ According to the man, two's complement is a better system than signed numbers.

7 **Listen again and complete the conversation.**

Student: I don't have a good grasp of how 1 ________ ________ ________ ________ can be represented.

Instructor: There are several ways that they can be identified.

Student: I'm just 2 ________ ________ ________ about it.

Instructor: As you know, there are 3 ________ ________ ________ ________ numbers.

Student: Isn't there a problem with using signs to 4 ________ ________?

Instructor: Yes, there is. That's why we use the two's complement representation.

Student: Can you 5 ________ ________ ________ ________?

Instructor: Basically, it is a way of 6 ________ ________ ________ as binary digits. It uses leading 0s and leading 1s to indicate whether it is positive or negative.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

I'm confused about ...

There are several ...

Can you clarify that for me?

Student A: You are a student. Talk to Student B about:

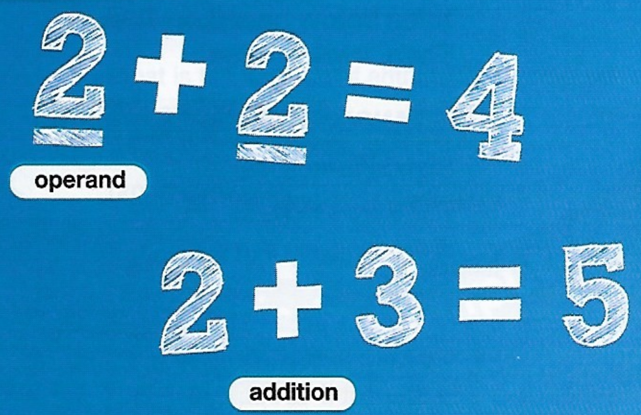
- computer arithmetic
- what you need explained
- how numbers are represented

Student B: You are an instructor. Talk to Student A about how numbers are represented.

Writing

9 **Use the reading passage and conversation from Task 8 to write a student's notes on computer arithmetic. Include: at least two different ways of identifying numbers, how they function, and which is more commonly used.**

Arithmetic in Computers: Part II



Computers perform arithmetic that is similar to the operations that we perform by hand. **Addition** is a basic operation. The sum of two **operands** is used to execute a specific instruction. In computing, addition is also used to perform **subtraction**. In common subtraction, sometimes a **value** from the next higher digit must be **borrowed**. This ensures that the **result** is a positive value. However, computer arithmetic simply adds a negative value to a positive value for the same result.

Many calculations require **carry-outs**. This number is taken from the right column to the left in order to complete an operation. **Multiplication** and **division** are related operations that computers perform to complete instructions. A **bit-wise shift** helps computers complete these operations more quickly.

Occasionally, a calculation will produce **overflow**. Overflow occurs when an operation produces more digits than the hardware can handle. Some computer programs **ignore** overflow, while others must **recognize** it. When overflow is detected, an **exception** or **interrupt** occurs. This suspends the current program until the issue is resolved. If programmed correctly, the computer jumps to a predetermined address to handle the exception. It can then resume its normal operations.

Get ready!

1 Before you read the passage, talk about these questions.

1 What are some common mathematical operations?

2 How is math used by computers?

Reading

2 Read the textbook chapter. Then, choose the correct answers.

1 What is the main idea of the article?

A instructions for completing mathematical operations

B a comparison of different mathematical operations

C how computers execute mathematical operations

D sample equations for different mathematical operations

2 Which of the following is NOT a possible result of overflow?

A An exception occurs.

B Hardware is damaged.

C The program ignores it.

D The occurrence is recognized.

3 What is true of bit-wise shifts?

A They help reduce overflow.

B They improve the efficiency of multiplication.

C They are used in subtraction.

D They are operations that add extra bits.

Vocabulary

3 Match the words (1-8) with the definitions (A-H).

1 __ overflow

2 __ bit-wise shift

3 __ exception

4 __ recognize

5 __ carry-out

6 __ value

7 __ operand

8 __ borrow

A an event that disrupts the execution of a program.

B a number that is used in a mathematical equation

C a condition that occurs when the result of a calculation is too large for the storage system of the computer

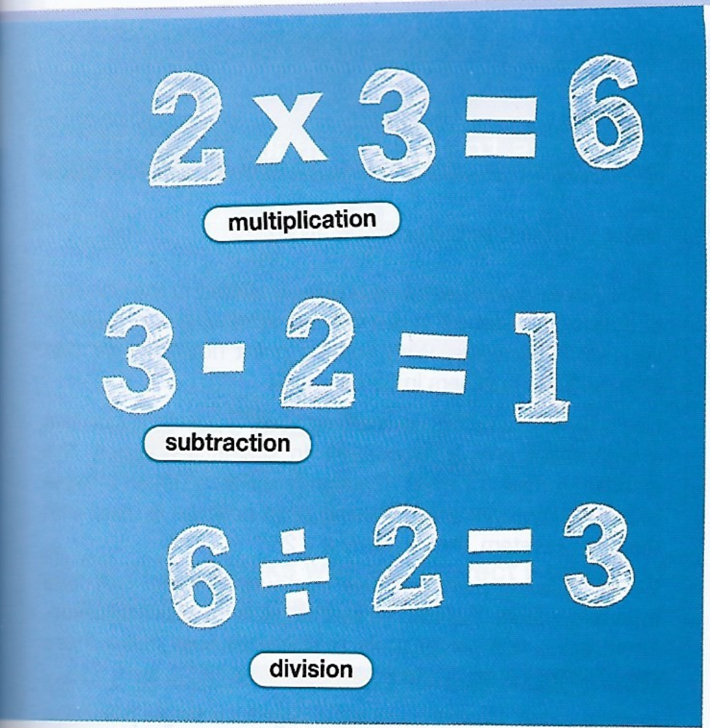
D to notice something

E to take a number, usually 10, from the next higher digit column

F a number, either positive or negative

G a number that is carried from the right column to the left in an equation

H an operation that moves the value of bits left or right



4 **Read the sentence pairs. Choose which word or phrase best fits each blank.**

1 **addition / subtraction**

A The process of ________ involves deducting one amount from another.

B The computer uses ________ to combine sums.

2 **multiplication / division**

A "Two times seven equals fourteen" is an example of ________.

B ________ is used to find out how many times two goes into four.

3 **result / interrupt**

A A(n) ________ is a temporary pause in the program.

B We see the ________ when all operations are complete.

5 🎧 **Listen and read the textbook chapter again. What does an exception do?**

Listening

6 🎧 **Listen to a conversation between two students. Mark the following statements as true (T) or false (F).**

1 __ The man identifies an error in the math.

2 __ The program will require minor adjustment.

3 __ The students find an interrupt in the program.

7 🎧 **Listen again and complete the conversation.**

Student 1: Hey, Annie. Did you have a chance to look **1** ________ ________ ________ ________?

Student 2: I was actually **2** ________ ________ ________ ________. I could use some help.

Student 1: Great. I'll give you a hand.

Student 2: In looking it over, I **3** ________ ________ ________.

Student 1: What **4** ________ ________ ________?

Student 2: There must be an error in the math somewhere. The computer isn't able to **5** ________ ________ ________.

Student 1: Well, the math looks good. I don't **6** ________ ________ ________ ________.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

Did you have a chance ...?

I noticed ...

I see what happened ...

Student A: You are a student. Talk to Student B about:

- a new computer program
- problems you noticed
- possible solutions

Student B: You are a student. Talk to Student A about problems with a new program.

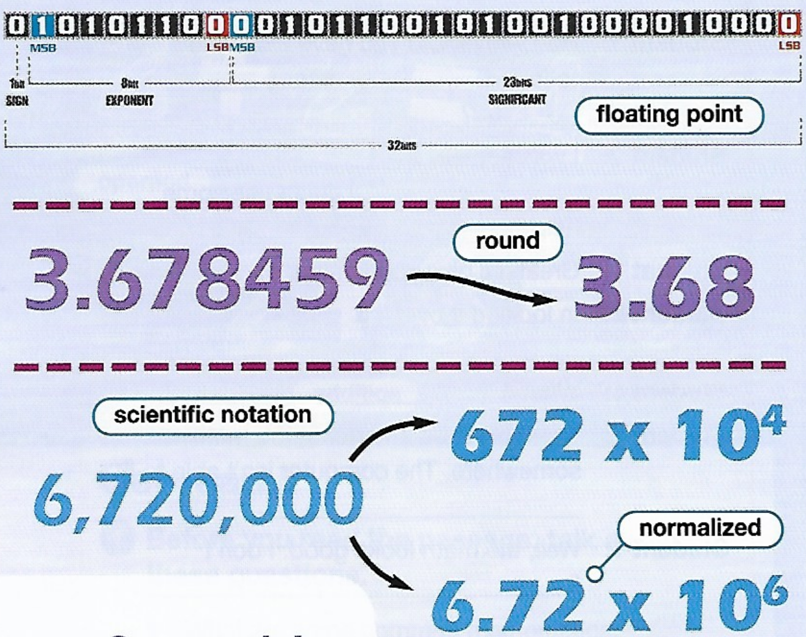
Writing

9 **Use the reading passage and conversation from Task 8 to write a student's review of a new computer program. Include: what arithmetic was used, an identification of the problem, and a possible cause of the problem.**

5 Arithmetic for Computers 3

HOME ABOUT US SERVICES CONTACT

computofaqs.com



Answers to Tech Industry Questions

Q: What is **floating point** notation?

A: Floating point notation is closely related to the concept of **scientific notation**. It is a way of expressing very large or very small numbers. With floating point notation, we can express large numbers in 32-bit words.

Like scientific notation, **normalized** floating point notation has a **significant** and an **exponent**. In scientific notation, the standard format is $a \times 10^b$. The significand a can be an **integer** or any real number. Since floating point notation is used with the binary system, the format is $a \times 2^b$.

We use floating point because computers cannot calculate **infinite** numbers. The closest we can get is an **approximation**. There are a number of tools available to help make these approximations **accurate**.

In some cases, the exponent is too large for a **single precision** word. **Double precision** minimizes occurrences of overflow and **underflow**. There are also a number of rounding tools that are commonly used. **Guard digits** allow for greater accuracy during intermediate addition. In some situations, a **sticky bit** is added before the number is finalized. The sticky bit ensures that numbers are **rounded** accurately. In ideal circumstances, the numbers are accurate within one-half **ULP**.

Get ready!

1 Before you read the passage, talk about these questions.

1 Why do computer engineers use floating point arithmetic?

2 What extra bits are used to make approximations more accurate?

Reading

2 Read the webpage. Then, choose the correct answers.

1 Why do engineers use approximations in computer arithmetic?

A Computers are not able to process infinite numbers.
B Single precision words are too small for some exponents.
C Double precision cannot completely eliminate underflow and overflow.
D Numbers must be accurate within one-half ulp.

2 What is the purpose of a sticky bit?

A to avoid underflow and overflow
B to implement more accurate rounding
C to join double precision words
D to round numbers to the nearest integer

3 Which is NOT mentioned in the passage?

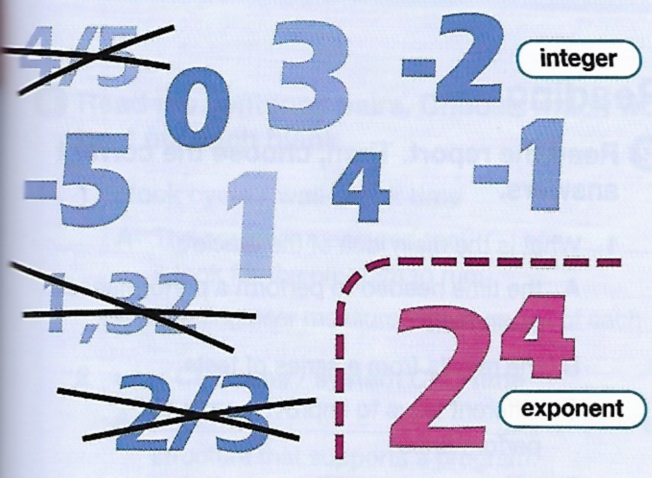
A Scientific notation has a significand and an exponent.
B Some tools increase accuracy in rounding.
C Floating point makes it easy to express large numbers.
D Approximations are usually not accurate.

Vocabulary

3 Match the words (1-8) with the definitions (A-H).

1 __ ulp
2 __ guard
3 __ integer
4 __ exponent
5 __ underflow
6 __ floating point
7 __ single precision
8 __ double precision

A a natural number, the negative of a natural number, or zero
B a number that indicates to what power another number is raised
C a situation in which a negative exponent is too large for a 32-bit word
D the expression of a value in a 32-bit word
E the expression of a value in two 32-bit words
F a type of computer arithmetic using a moveable binary point
G a measure of the margin of error in rounding
H an extra bit added to the right of the binary point



4 Fill in the blanks with the correct words and phrases from the word bank.

WORD BANK

round infinite sticky bit accurate scientific notation normalized approximation

1 The numbers we provided were only a rough ________________ .

2 A(n) ________________ is sometimes added to make rounding more accurate.

3 In ________________ numbers, there are no leading zeroes.

4 ________________ is a convenient way of writing very large and very small numbers.

5 The engineer decided to ________________ up to the nearest dollar on the project budget.

6 The error was caused by a calculation that was not ________________ .

7 Computers cannot compute or store ________________ values.

5 Listen and read the webpage again. How can engineers avoid situations in which the exponent is too large for a single precision word?

Listening

6 Listen to a conversation between an instructor and a student. Mark the following statements as true (T) or false (F).

1 __ The woman asks the man to explain scientific notation to the class.

2 __ The man confuses underflow and overflow.

3 __ The man got a high score on the floating point exam.

7 Listen again and complete the conversation.

Instructor: Todd, can you tell the class why we use **1** ______ ______ arithmetic?

Student: We use it so large numbers will fit in a 32-bit word.

Instructor: That's right. How about an **2** ______ that's too large for the exponent field? Do you remember what that's called?

Student: It's called **3** ______ .

Instructor: Close, but not quite. It's called **4** ______ .

Student: That's right, I always get those **5** ______ ______ . Underflow is when a negative exponent is too large.

Instructor: Correct. And what can we do to avoid overflow and underflow?

Student: Well, sometimes we can use **6** ______ ______ .

Speaking

8 With a partner, act out the roles below based on Task 7. Then, switch roles.

USE LANGUAGE SUCH AS:

Can you tell us ...? / That's right.

Close, but not quite ...

Student A: You are an instructor. Talk to Student B about:

- floating point concepts
- what he or she knows about the subject
- what concepts will be on an upcoming exam

Student B: You are a student. Talk to Student A about floating point concepts.

Writing

9 Use the reading passage and conversation from Task 8 to write an essay on floating point concepts. Include: why computers use floating point arithmetic, how to ensure accurate calculations, and why computers use approximations instead of precise values.

6 Assessing Performance 1

Get ready!

1 Before you read the passage, talk about these questions.

1 What are some ways to measure computer performance?

2 Why are computer performance tests important?

AJC Computers

Computer Performance Report

Client: LewsTech industries
Report Date 8/24 **Report Time:** 1:17 pm

On Friday, we ran several routine tests on the central office computer. Since the last evaluation, a few users complained about slow processing speeds. We used a variety of **metrics** to measure system **performance**. This included several types of time measurements. The goal of the tests was to assess the system's **execution time** and **throughput**. Most metrics were normal, but we will need to perform a few more tests.

We first tested the central processing units for each computer. This involved both the **wall-clock time** and the total **CPU time**. Each CPU was functioning at an expected level. We also evaluated the processors' ability to run programs with minimal resources. Both **user CPU time** and **system CPU time** were tested. According to the results, all programs are running smoothly.

We also looked at the processor itself. We examined the speeds of the **clock cycles**. We found that the **clock rate** is slower than normal. Each cycle was measured to determine overall **CPI**, or clock cycles per instruction. Based on our results, the IT team will further investigate the cause of the decreased processing speed.

Reading

2 Read the report. Then, choose the correct answers.

1 What is the main idea of the article?
- **A** the time needed to perform a performance evaluation
- **B** the results from a series of tests
- **C** different ways to improve computer performance
- **D** a comparison of different testing methods

2 Which is NOT true of clock cycles?
- **A** They measure the speed of a processor.
- **B** They have different lengths.
- **C** They are used to determine CPI.
- **D** They increase program efficiency.

3 How will the IT team address the system issues?
- **A** overhaul the system
- **B** uninstall several programs
- **C** install new processors
- **D** investigate the problems further

Vocabulary

3 Match the words (1-7) with the definitions (A-G).

1 __ performance
2 __ CPI
3 __ execution time
4 __ clock rate
5 __ CPU time
6 __ metric
7 __ throughput

- **A** a measurement of a certain aspect of something's performance
- **B** the amount of work that something can do and the time it takes to accomplish it
- **C** the amount of work a computer can do in a specific amount of time
- **D** the number of clock cycles it takes to complete an instruction
- **E** the amount of time the central processing unit takes to complete a task
- **F** the rate of cycles per second a computer takes to perform
- **G** the time that elapses from the start of a task to the end

4 **Read the sentence pairs. Choose which word or phrase best fits each blank.**

1 **clock cycle / wall-clock time**

A The engineer measured the ______________________ it took for the program to run.

B The engineer measured the duration of each ____________.

2 **user CPU time / system CPU time**

A ______________________ runs the background structure that supports a program.

B The performance of the processor, while running programs, is measured by ______________________.

5 **Listen and read the report again. How is the speed of a processor measured?**

Listening

6 **Listen to a conversation between two engineers. Mark the following statements as true (T) or false (F).**

1 __ The test results showed strong CPI performance.

2 __ The users of the computers complained about slow performance.

3 __ According to the woman, the computer's throughput level was disappointing.

7 **Listen again and complete the conversation.**

Engineer 2: A few people complained that their computers were slow. So I used **1** ______________ ______ both the speed and capacity of the processor.

Engineer 1: What metrics did you use?

Engineer 2: I started with the central processing unit. I tested it for **2** ______________________.

Engineer 1: Good thinking. What did you find?

Engineer 2: Well, there's **3** ______________________ ______.

Engineer 1: What's the good news?

Engineer 2: The throughput is still fairly high. The computer still processes large amounts of information in a **4** ______ ______________.

Engineer 1: Well, that is good. We need the computers to **5** ______________.

Engineer 2: However, there were some **6** ______________ ______.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

Did you get ...?

There is good news ...

However, there is also ...

Student A: You are an engineer. Talk to Student B about:

- a test that he or she just ran
- what metrics were used
- the results of the test

Student B: You are an engineer. Talk to Student A about a computer performance test.

Writing

9 **Use the reading passage and conversation from Task 8 to write a computer performance report. Include: two metrics that were tested, the results of the test, and further recommendations.**

7 Assessing Performance 2

All About Computers: Performance Assessments

HOME | ABOUT US | SERVICES | CONTACT

Manufacturers like to talk about the **workload** their computers can handle. But those claims aren't always reliable. At All About Computers, we assess computer systems and evaluate manufacturers' claims. We use a variety of **benchmarks** to measure computer performance. We use real **applications** that you use every day and measure their performance in **MIPS**. We follow the **reproducibility** rule. We also use **SPEC CPU benchmarks** and the **SPEC ratio**. This is how we test the execution times of the machine's central processing unit. We take steps to ensure the best, most reliable results.

Our Process

Performance assessments require many calculations. The first is **arithmetic mean.** This compares the average execution time to the overall execution time. We also evaluate **weighting factors**. Programs with higher percentages of use have higher weighting factors. The weighting factor and execution times are used to calculate the **weighted arithmetic mean**. That yields the total performance of the workload.

Why We Do It

Some manufacturers claim that a new version of a product is significantly faster. And they'll increase the price. But there's a limit to how much faster a system can get. (See our explanation **diminishing returns**.) **Amdahl's law** is also used to find the maximum expected improvement to a system when only part of it is improved. That's why bringing in All About Computers is worth the investment.

Get ready!

1 Before you read the passage, talk about these questions.

1 What are some ways to evaluate the performance of a computer?

2 How is arithmetic used to evaluate computer performance?

Reading

2 Read the webpage. Then, choose the correct answers.

1 What is the main idea of the webpage?

A which manufacturers' machines have the best performance

B ways to improve a machine's performance

C a company's methods for evaluating a computer's performance

D how consumers can test computer performance at home

2 Which of the following is part of Amdahl's law?

A high percentages of program use

B a measure of the CPU

C decreasing performance over time

D changing one aspect to improve overall performance

3 How does the company assist its customers?

A testing manufacturers' statements

B increasing processing speed

C creating processing benchmarks

D comparing different weighting factors

Vocabulary

3 Match the words and phrases (1-8) with the definitions (A-H).

1 __ workload

2 __ MIPS

3 __ reproducibility

4 __ weighting factor

5 __ SPEC CPU benchmark

6 __ SPEC ratio

7 __ arithmetic mean

8 __ weighted arithmetic mean

A the ability to duplicate something

B the average of execution times compared with total execution time

C the percentage of usage that a program in a workload has

D a measurement of the execution speed of a program by the millions of instructions

E the sum of the weighting factors and execution times

F the measurement of the execution time of a computer compared to that of another

G a set of real programs that measure the performance of the central processing unit

H the set of programs that a computer runs on a daily basis

4 **Read the sentence pairs. Choose which word or phrase best fits each blank.**

1 **benchmarks / applications**

A ____________ are the programs that a computer executes every day.

B One way to evaluate computer performance is to use ____________ .

2 **Amdahl's law / diminishing returns**

A According to ____________, adjusting one element of a computer can help to find the maximum expected improvement to a whole system.

B According to ____________, increasing a production element can decrease production in the long run.

5 **Listen and read the webpage again. What is the function of benchmarks?**

Listening

6 **Listen to a conversation between an engineer and an intern. Mark the following statements as true (T) or false (F).**

1 __ The woman made an error when she tested the computer speed.

2 __ The manufacturer made illegal claims about the new computers.

3 __ The weighted arithmetic mean shows a small difference in speed.

7 **Listen again and complete the conversation.**

Engineer: Let's **1** ____________. I think you'll be surprised.

Intern: Really? Why is that?

Engineer: Well, this manufacturer **2** ____________ were fifteen percent faster, right?

Intern: Right. Is that not true?

Engineer: Not according to our tests. It failed to meet **3** ____________.

Intern: So they lied about their product? Isn't that illegal?

Engineer: They didn't lie, exactly. See, the **4** ____________. But only under certain conditions.

Intern: I'm not sure I get **5** ____________.

Engineer: Let me explain. They **6** ____________ with a light workload.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

Is that not true?

What do you mean?

I'm not sure ...

Student A: You are an engineer. Talk to Student B about:

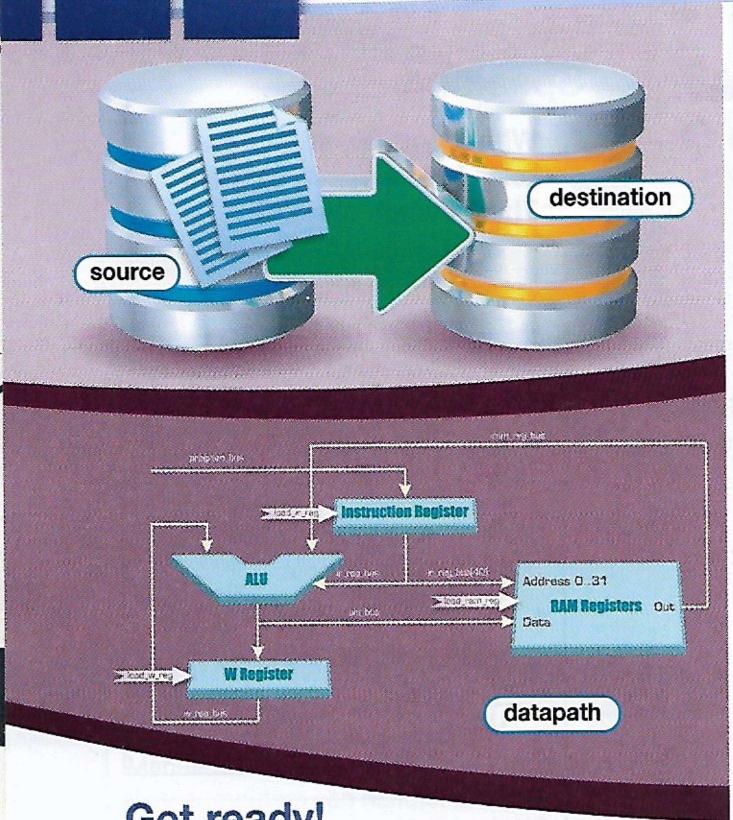
- a performance test
- how results are gathered
- challenges when assessing computer performance

Student B: You are an intern. Talk to Student A about a performance test.

Writing

9 **Use the reading passage and conversation from Task 8 to write an evaluation of a computer's performance. Include: types of measurements that were used.**

8 Datapaths and Control



Get ready!

1 Before you read the passage, talk about these questions.

1 What devices are used in datapaths?

2 What are the three instruction classes?

Reading

2 Read the textbook chapter. Then, choose the correct answers.

1 What is the chapter mostly about?

A the physical construction of a standard datapath

B the components and implementation of datapaths

C new advances in the hardware used in datapaths

D troubleshooting common datapath errors

2 Which is NOT a component of the datapath?

A ALU C memory

B multiplexer D PC

3 Which device stores the address of the next instruction?

A program counter

B instruction register

C data selector

D control unit

Computer Design

Datapaths

The term ***datapath*** refers to a series of devices that perform calculations. A **control** distributes program instructions to the datapath, memory, and I/O devices. We discussed control units previously in Chapters 1 and 3. Standard datapaths consist of a **PC** (program counter) and various small registers. Arithmetic logic units (**ALUs**) and simple **adders** perform basic arithmetic tasks. In some cases, there are multiple adders and ALUs performing calculations simultaneously. However, it is impractical to wire every possible I/O connection. Many datapaths solve this problem with a **multiplexer**, or **data selector.** Multiplexers transfer data from the correct input **source** to its **destination.**

The address of the current instruction is stored in an instruction register. Be careful not to confuse the instruction register with the PC. The PC stores the address of the next planned instruction, like a bookmark.

In order to understand **implementation**, we must understand the three **instruction classes:**

- **memory-reference** instructions read from memory or write data to memory
- **arithmetic-logical** instructions perform calculations
- **branch** instructions provide the PC with a new instruction address

Note that all three instruction classes make use of ALUs! Don't let the term 'arithmetic-logical' fool you.

Vocabulary

3 Match the words (1-8) with the definitions (A-H).

1 __ PC

2 __ ALU

3 __ adder

4 __ branch

5 __ datapath

6 __ multiplexer

7 __ implementation

8 __ arithmetic-logical

A an instruction that tells the datapath to perform mathematical operations

B a device that chooses from several inputs and sends to a single output

C a circuit that carries out arithmetic and logical operations

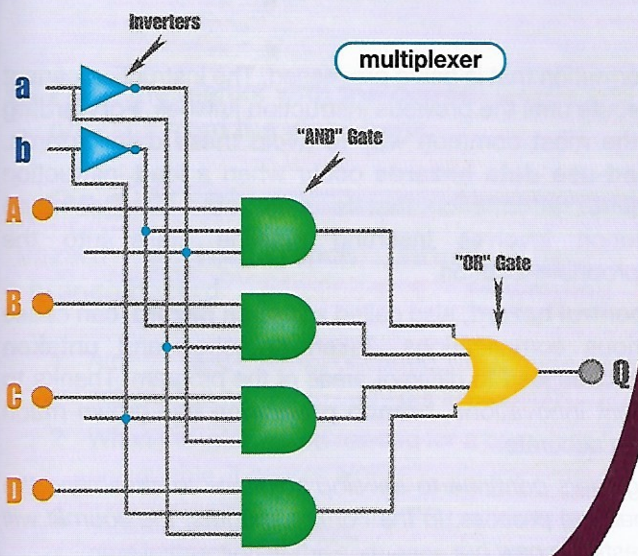
D a register that stores the address of the next instruction

E the process of carrying out a task in a certain way

F a series of units that perform data processing tasks

G an instruction that changes the instruction address in the PC

H a circuit that performs addition operations



4 Read the sentence pairs. Choose which word or phrase best fits each blank.

1 **control / instruction class**
 A The ____________ gives instructions to the datapath.
 B ____________ is a category for types of instructions.

2 **memory-reference / data selector**
 A A ____________ chooses the right input and sends it to its destination.
 B A ____________ instruction reads or writes information.

3 **source / destination**
 A The ____________ is where information comes from.
 B The ____________ is where the information is going.

5 Listen and read the textbook chapter again. Why do datapaths use a multiplexer?

Listening

6 Listen to a conversation between a student and an instructor. Mark the following statements as true (T) or false (F).

1 __ The woman is confused about the purpose of the control.
2 __ The woman correctly identifies the first step in datapath instructions.
3 __ The man recommends that the woman read the chapter on ALUs again.

7 Listen again and complete the conversation.

Student: Well, I know that the **1** ________ gives the instructions. But I don't really understand the data flow.
Instructor: Like you said, the control gives instructions to the **2** ________. Do you know what the first step is?
Student: No, I don't.
Instructor: **3** ________ ________ ________.
Student: Is it having the **4** ________ fetch the next instruction?
Instructor: That's right. See, you understand better than you think you do. From there, we usually have to follow a **5** ________ - ________ instruction.
Student: Okay. Then what?
Instructor: Well, all **6** ________ ________ use the ALU. So that's where the data goes next.

Speaking

8 With a partner, act out the roles below based on Task 7. Then, switch roles.

USE LANGUAGE SUCH AS:
The first step is that ...
From there ...
I thought ...

Student A: You are a student. Talk to Student B about:
- datapath implementation
- units involved in data processing
- concepts you are confused about

Student B: You are an instructor. Talk to Student A about datapath implementation.

Writing

9 Use the reading passage and conversation from Task 8 to write a teacher evaluation. Include: how an instructor helped you understand datapaths, what you were confused about, and what you learned.

Pipeline Hazards

Article from the Journal of Computer Programming and Engineering

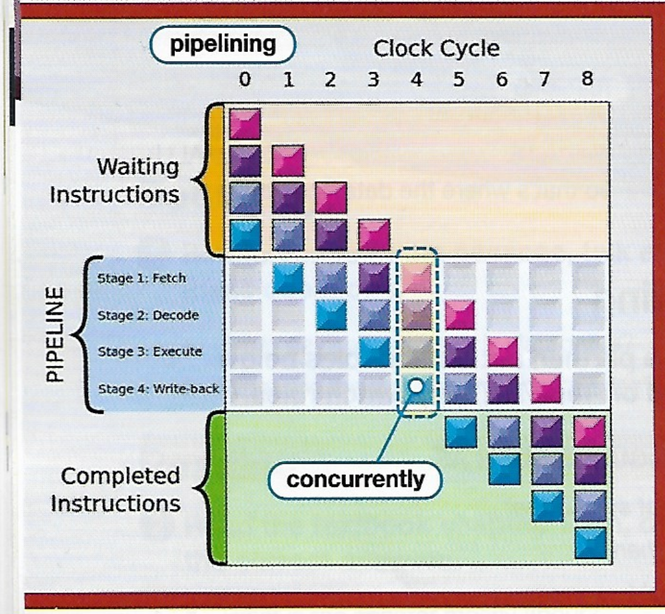
Pipelining is a standard technique for improving throughput. It works by **concurrently** operating all **stages** of an instruction set. Though it does not decrease **latency**, it dramatically reduces throughput time. However, concurrent operations create a number of potential **hazards** in the pipeline:

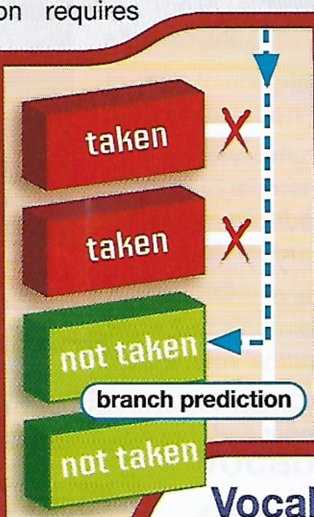
Structural hazards occur when the hardware is insufficient to accommodate all instructions. Unfortunately, it is impossible to predict what machines private users will own. **Pipeline stalls** can reduce the risk of structural hazards.

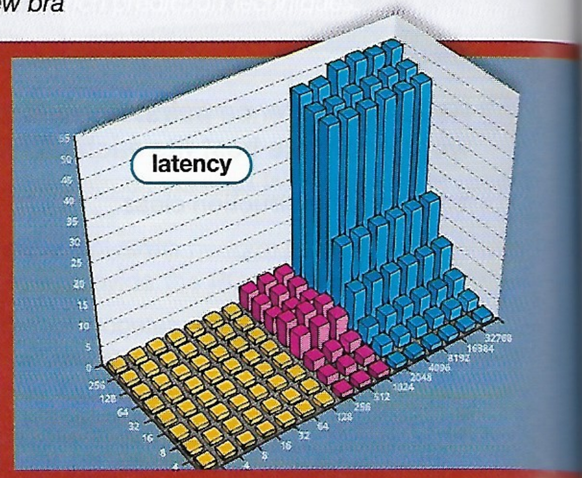
Data hazards occur when an instruction requires information that is being processed. The instruction cannot execute until the previous instruction finishes. **Forwarding** is the most common way to avoid these data hazards. **Load-use data hazards** occur when a load instruction requires information that is unavailable. One common solution involves inserting pipeline stalls into the appropriate location.

A **control hazard**, also called a **branch hazard**, can cause serious complications. Taken branches and **untaken branches** lead to different areas of the program. Thanks to recent innovations, **branch prediction** has grown much more accurate.

Engineers continue to develop solutions to streamline the pipelining process. In the coming months, the Journal will investigate new bra







Get ready!

1 Before you read the passage, talk about these questions.

1 What is the purpose of pipelining?
2 What types of hazards occur in pipelining?

Reading

2 Read the journal article. Then, mark the following statements as true (T) or false (F).

1 __ Pipelining dramatically reduces latency.
2 __ According to the article, forwarding is the most common way to avoid load-use data hazards.
3 __ Branch prediction helps prevent control hazards.

Vocabulary

3 Match the words (1-8) with the definitions (A-H).

1 __ stage
2 __ latency
3 __ pipelining
4 __ concurrently
5 __ data hazard
6 __ control hazard
7 __ branch prediction
8 __ load-use data hazard

A a specific task or action in an overall process
B a situation in which the data needed for an instruction is not available
C the time required to execute an individual instruction
D at the same time
E a situation in which the information needed for a branch is not available
F the act of guessing whether a branch will be taken
G a technique for implementing multiple instructions simultaneously
H a situation in which the data for a load instruction is not available

4 Fill in the blanks with the correct words and phrases from the word bank.

WORD BANK

hazard untaken branch structural hazard branch hazard pipeline stall forwarding

1 A(n) ____________ occurred when there were not enough adders to carry out instructions.
2 When the information needed for a branch is unavailable, it causes a(n) ____________.
3 In a(n) ____________, the PC proceeds to the next instruction in the sequence.
4 A(n) ____________ is a situation in which the planned instruction cannot be executed.
5 In order to avoid data hazards, most systems use ____________.
6 The engineers encountered a structural hazard, so they implemented a(n) ____________.

5 Listen and read the journal article again. When do computer programmers implement pipeline stalls?

Listening

6 Listen to a conversation between two computer engineers. Choose the correct answers.

1 What is the conversation mostly about?
- **A** a hazard in a data pipeline
- **B** a sudden increase in latency
- **C** a new pipelining approach
- **D** an error in forwarding

2 What will the man likely do next?
- **A** draw a diagram of a new pipeline
- **B** research control hazard solutions
- **C** implement an additional pipeline stall
- **D** use more effective branch prediction

7 Listen again and complete the conversation.

Engineer 1: Hey, April. It looks like we've got some kind of **1** ________ here.

Engineer 2: That's not good. Where did you find the problem?

Engineer 1: It's the program we worked on yesterday. We just implemented the **2** ________ and it's not working correctly.

Engineer 2: Well, that's not uncommon. Which **3** ________ failed to execute?

Engineer 1: The first problem is with this load instruction. It looks like a **4** ________ - ________ ________ ________.

Engineer 2: Did you add a **5** ________ ________?

Engineer 1: Yeah, but it didn't seem to change anything. That's why I'm confused.

Engineer 2: Hmm. May I **6** ________ ________ ________ ________?

Speaking

8 With a partner, act out the roles below based on Task 7. Then, switch roles.

USE LANGUAGE SUCH AS:

It looks like ...

It might have been ...

That makes sense.

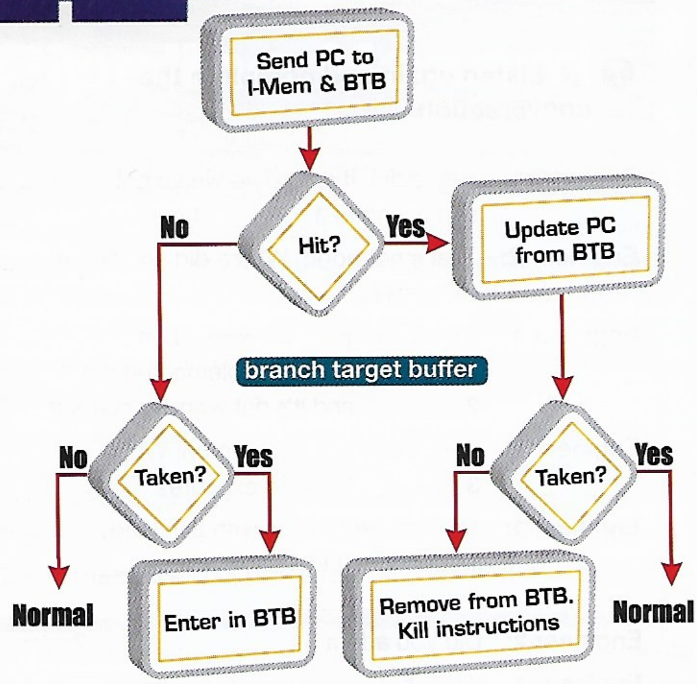
Student A: You are an engineer. Talk to Student B about:
- a hazard in a pipeline
- what type of hazard occurred
- how to resolve the problem

Student B: You are an engineer. Talk to Student A about a pipeline hazard.

Writing

9 Use the reading passage and conversation from Task 8 to write an error resolution report. Include: a type of hazard that occurred, what steps were taken to correct it, and whether or not the issue was resolved.

10 Pipelining 2



Pipelines (CONTINUED)

As indicated in previous chapters, the challenge of pipelining is avoiding hazards. In addition to **bubbles**, we can also use **NOPs**. In essence, a NOP (No Operation) is an instruction that does nothing. In this respect, it functions very much like a pipeline stall. However, constantly bubbling the pipeline does not ensure smooth execution of instructions. In the case of branch instructions, we need to use branch prediction.

Branch prediction allows us to guess whether a branch will be taken. When the prediction is wrong, we simply **flush instructions** and start over. But flushing instructions takes up valuable time. Fortunately, there are a number of advanced branch prediction methods.

Dynamic branch prediction involves looking up whether a branch was recently taken. This information is stored in a **branch history table**, or **branch prediction buffer**. A **correlating predictor** operates similarly, but also looks up global branch data. **Tournament branch predictors** are the most useful because they provide more options.

But even an advanced predictor will never be totally accurate. Keeping a NOP in the **branch delay slot** can help eliminate penalties. Another approach is to store the branch destination in a **branch target buffer**. This reduces the time needed to retrieve branch information.

Get ready!

1 Before you read the passage, talk about these questions.

1 What are some types of branch prediction?
2 What is the purpose of a NOP?

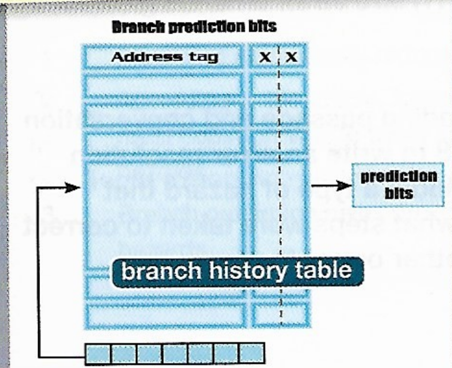
Reading

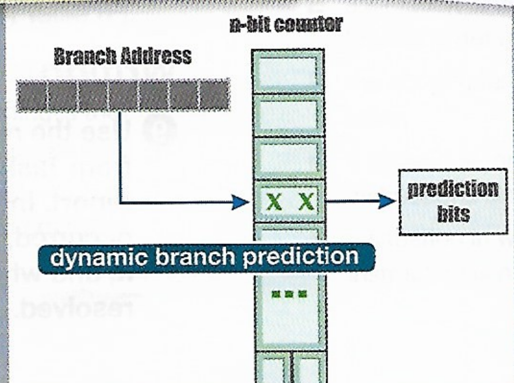
2 Read the textbook chapter. Then, choose the correct answers.

1 What is the chapter mostly about?
   A problems with early methods of branch prediction
   B ways to avoid flushing instructions from the pipeline
   C how branch prediction makes pipelining more efficient
   D the differences between NOPs and pipeline stalls

2 What is the function of a branch history table?
   A It looks up global data about recently taken branches.
   B It stores data about whether a branch was recently taken.
   C It records the destination of the next branch.
   D It calculates the accuracy of branch prediction.

3 Which idea is NOT mentioned in the passage?
   A Instructions are flushed from the pipeline when a prediction is wrong.
   B Tournament branch predictors are more versatile than other predictors.
   C Correlating predictors use local and global data about taken branches.
   D NOP instructions are executed in inactive stages of the pipeline.





Vocabulary

3 Match the words (1-7) with the definitions (A-G).

1 __ NOP
2 __ flush instructions
3 __ correlating predictor
4 __ branch target buffer
5 __ branch delay slot
6 __ branch history table
7 __ tournament branch predictor

A a cache that stores the next instruction for a taken branch
B a branch predictor that uses local and global data
C a space containing the first instruction that will be executed after a branch
D a small memory that records whether a branch was recently taken
E an instruction that does nothing
F a branch predictor that has multiple prediction types to choose from
G to discard all current instructions

4 Read the sentence and choose the correct word.

1 The **branch prediction buffer / correlating predictor** contains information about whether a branch was recently taken.
2 Some hazards can be resolved by inserting a **bubble / branch delay slot** into the pipeline.
3 **Branch target buffer / Dynamic branch prediction** uses information about recently taken branches.

5 Listen and read the textbook chapter again. Why is a branch target buffer useful?

Listening

6 Listen to a conversation between an instructor and a student. Mark the following statements as true (T) or false (F).

1 __ The man used too many bubbles.
2 __ The woman suggests using dynamic branch prediction.
3 __ The woman advises the man to flush instructions.

7 Listen again and complete the conversation.

Instructor: I can see that you have **1** ______ inserted in all the right places. But you need to use better branch prediction.
Student: Okay. How do I do that? Should I use **2** ______ ______ ______?
Instructor: In this case, that's the best option. It'll look up information from the **3** ______ ______ ______.
Student: All right.
Instructor: But don't forget, you'll still have to **4** ______ ______ when the branch prediction is wrong.
Student: **5** ______ ______ ______ ______ by that?
Instructor: Well, the program might execute instructions based on a wrong prediction. So you have to **6** ______ ______ ______ those instructions.
Student: Oh, I see.

Speaking

8 With a partner, act out the roles below based on Task 7. Then, switch roles.

USE LANGUAGE SUCH AS:

How do I do that?
Don't forget to ... / What do you mean ...?

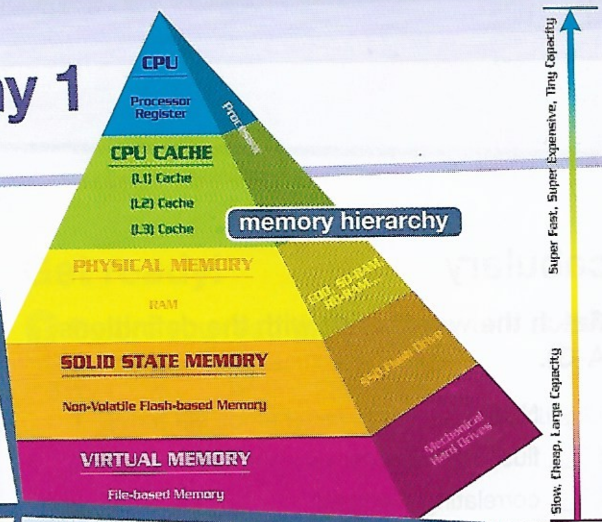
Student A: You are an instructor. Talk to Student B about:
- a pipelining assignment
- what problems he or she encountered
- your advice for resolving the problem

Student B: You are a student. Talk to Student A about a pipelining assignment.

Writing

9 Use the reading passage and conversation from Task 8 to write a student assessment. Include: a review of the student's pipelining assignment, problems he or she had with a pipeline, and how he or she resolved the problems.

11 Memory Hierarchy 1



Get ready!

1 Before you read the passage, talk about these questions.

1 What is a memory hierarchy?

2 What are the principles of temporal and spatial locality?

CompDIY — The do-it-yourself computer forum

Topic: Memory Hierarchy

Post

JackieN I'm currently building my first computer. Some of my friends told me to create a **memory hierarchy**. What is memory hierarchy, and how does it work?

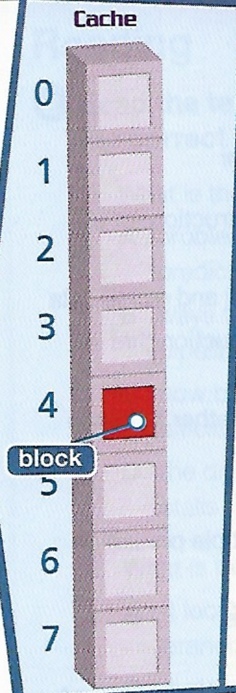
Total Replies: 2

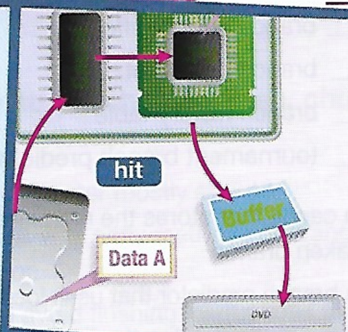
Craig32 A memory hierarchy is a way of arranging memory into multiple levels. The top level is SRAM, followed by layers of DRAM. The bottom level is your magnetic disk.

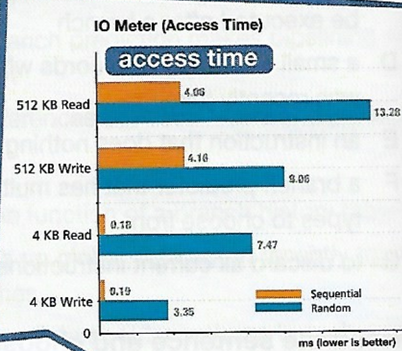
At any one time, we only use a small percentage of the memory. (This is the **principle of locality**.) So we put the data we're most likely to need in the cache. Programs will fill the cache based on **temporal locality** and **spatial locality**. That way, we can **reference** the data we need quickly. Without the memory hierarchy, the **access time** would be a lot longer.

Hope that answers your question. :)

TRalston1991 You should also look up memory accesses. A memory access is classified as either a **hit** or a miss. A miss occurs when the **block** you need isn't in the cache. You want your **hit rate** to be as high as possible. Make sure the **miss rate** isn't higher than the hit rate. **Miss penalties** can slow down your processor by a lot. You also want to keep your **hit time** down. Good luck!







Reading

2 Read the message board. Then, mark the following statements as true (T) or false (F).

1 __ The original post explains how a memory hierarchy works.

2 __ According to the message board, computer users want high hit rates.

3 __ The second reply corrects an error in the first reply.

Vocabulary

3 Match the words (1-8) with the definitions (A-H).

1 __ hit

2 __ block

3 __ hit rate

4 __ miss rate

5 __ reference

6 __ miss penalty

7 __ memory hierarchy

8 __ principle of locality

A a situation in which the requested data is present in the cache

B a concept that states that only a small amount of memory is used at one time

C the smallest unit of data that can exist in a level of memory

D the extra time required to retrieve data from lower levels of memory

E a system for organizing memory into multiple tiers

F the percentage of memory accesses found in the cache

G to open or recall something from its data location

H the percentage of memory accesses not found in the cache

4 **Read the sentence pairs. Choose which word or phrase best fits each blank.**

1 **access time / hit time**

A The __________________ is the time needed to determine if a block is in the cache.

B The __________________ is the time required to retrieve data from memory.

2 **temporal locality / spatial locality**

A We place recently used addresses in the cache based on __________________.

B Sequential addresses are in the cache based on ____________.

5 **Listen and read the message board again. According to the first reply, why is a memory hierarchy important?**

Listening

6 **Listen to a conversation between two engineers. Choose the correct answers.**

1 What is the conversation mostly about?

A the differences between types of localities
B a problem with the function of a program's memory
C an upcoming presentation about reducing access time
D recent improvements in hit and miss rates

2 What will the man likely do next?

A adjust the program to use more spatial locality
B install an extra cache in the memory hierarchy
C send a report to a colleague about the hit rate
D calculate the program's average hit time

7 **Listen again and complete the conversation.**

Eng. 1: I had a feeling that might be the case. Did the **1** _______ _______ go up, at least?

Eng. 2: Nope. The hit rate is **2** _______ _______ than it was before. The miss penalties are really slowing things down.

Eng. 1: Well, it sounds like we've got a problem there. Are we using **3** _______ _______ to fill the cache?

Eng. 2: Yeah, but in this case it doesn't seem to be a good choice.

Eng. 1: Well, let's improve the **4** _______ _______. Is that what you were thinking?

Eng. 2: Yes, that's exactly what I was thinking.

Eng. 1: With luck, that'll bring the **5** _______ _______ down. Aside from that, how do things look?

Eng. 2: It's hard to say at this point. The **6** _______ _______ is reasonable, though.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

How's the program ...?
It's even higher/lower than ...
It sounds like ...

Student A: You are an engineer. Talk to Student B about:

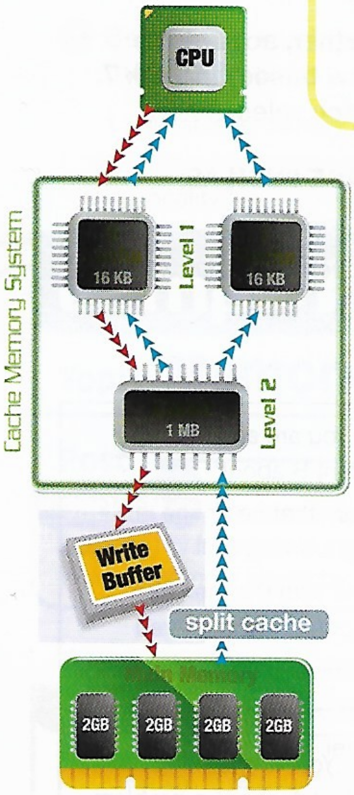
- a program that he or she is currently working on
- problems with the memory that he or she encountered
- how to solve the problem

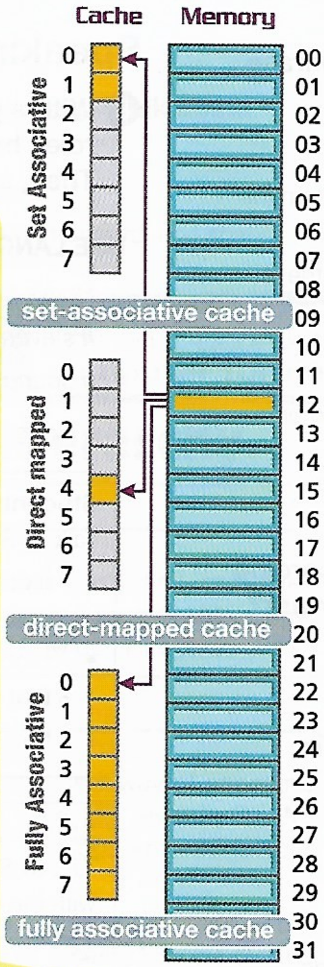
Student B: You are an engineer. Talk to Student A about a problem with the program's memory.

Writing

9 **Use the reading passage and conversation from Task 8 to write a post on a computer engineering forum. Include: a problem an engineer encountered, what measures were already taken, and what the results were.**

12 Memory Hierarchy 2





Cache (computing)

A **cache** is a small, fast memory unit that stores instructions and active program data. The cache allows the CPU to **access** relevant information quickly and efficiently. There are several different cache setup schemes, each useful in different scenarios.

A **direct-mapped cache** assigns each memory location to a specific cache location. In a **fully associative cache**, any block may be placed in any location. A **set-associative cache** is the middle ground between the two extremes. A set-associative cache assigns a set number of potential block locations. Blocks of data are identified by **tags**, and verified by a **valid bit**. The valid bit indicates whether or not the tag is current.

A **split cache** is a memory setup that utilizes two **parallel** caches. One cache only **handles** instructions, while the other handles data. While split caches increase cache bandwidth, they increase the rate of **cache misses.**

Memory hierarchies have various ways to keep data **consistent** between the cache and the main memory. One method is **write-through**, in which both are updated simultaneously. Write-through is effective, but slow. Some systems utilize a **write buffer** (a small **queue**) to streamline the process. Another solution is **write-back**. A write-back scheme updates the memory only after the cache entry is replaced with new information.

Get ready!

1 Before you read the passage, talk about these questions.

1 What are some different types of caches?

2 How can programmers ensure that the cache and the memory are consistent?

Reading

2 Read the encyclopedia entry. Then, mark the following statements as true (T) or false (F).

1 __ A tag identifies the valid bit as current or not current.

2 __ Cache misses are increased when a programmer uses a split cache setup.

3 __ In a write-back, the cache and memory are updated at the same time.

Vocabulary

3 Match the words (1-8) with the definitions (A-H).

1 __ tag

2 __ cache miss

3 __ split cache

4 __ write-back

5 __ write-through

6 __ direct-mapped cache

7 __ set-associative cache

8 __ fully associative cache

A a cache that assigns each block to a specific cache location

B a memory setup that uses two parallel caches

C a marker that identifies the contents of a block

D a cache in which any block can be placed in any location

E a process for updating the cache and the memory simultaneously

F a situation when the requested block is not in the cache

G a cache in which a block can be placed in a fixed number of locations

H a process for updating memory only when the cache block is replaced

4 **Read the sentence pairs. Choose which word or phrase best fits each blank.**

1 **queue / cache**

A A ______________ is the small, fast memory closest to the CPU.

B A ______________ is a series of blocks waiting to be processed.

2 **valid bit / write buffer**

A The ________________ helps to prevent processor stalls.

B The ________________ identifies a cache entry as current.

3 **handle / access**

A The control will ______________ information from the memory.

B The processor couldn't ______________ the request.

4 **consistent / parallel**

A The memory and the cache should be ______________ with one another.

B A split cache uses two ______________ caches.

5 **Listen and read the encyclopedia entry again. What does the valid bit do?**

Listening

6 **Listen to a conversation between two computer engineers. Choose the correct answers.**

1 What is the conversation mostly about?

A the implementation of a direct-mapped cache
B the challenges of working with a split cache
C a new development in the use of write buffers
D a problem with cache-memory consistency

2 What will the woman likely do next?

A reset the valid bits for the program
B install a fully-associative cache
C implement a write buffer with the program
D switch the program to a write-through scheme

7 **Listen again and complete the conversation.**

Engineer 1: Ray, I was hoping you could 1 ________ ________ ________ ________. I know you have a lot more experience with this than I do.

Engineer 2: Sure, Nell. Are you still having problems with that 2 ________?

Engineer 1: Yeah, I'm getting a lot of cache misses. And sometimes the processor can't 3 ________ ________ ________ at all.

Engineer 2: Let's have a look. Is this a 4 ________ ________ ________?

Engineer 1: No, it's a 5 ________ - ________ ________.

Engineer 2: Ah, I see. Okay, it looks like the cache and the memory aren't 6 ________.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

I was hoping you could give me a hand.

It looks like ...

Should I ...?

Student A: You are an engineer. Talk to Student B about:

- a problem with cache function
- the cause of the problem
- his or her recommended solution

Student B: You are an engineer. Talk to Student A about a problem with cache function.

Writing

9 **Use the reading passage and conversation from Task 8 to write an email to a senior engineer. Include: the original problem, the measures taken to solve it, and the resolution to the problem.**

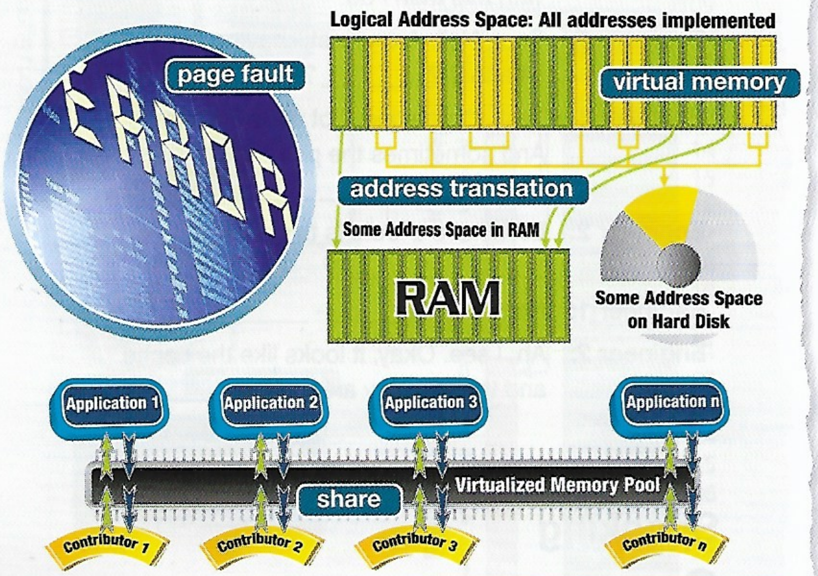
13 Virtual Memory

Get ready!

1 Before you read the passage, talk about these questions.

1 Why do programmers use virtual memory?

2 How does address translation work?



Unit 5.3 Virtual Memory & Paging

What is virtual memory?

Virtual memory allows multiple programs to **share** memory safely and effectively.

Remember: Like caches, virtual memory operates on the principle of locality.

In order to keep programs isolated, each program receives its own **address space**. Virtual memory translates the **virtual addresses** into the real or **physical addresses**. **Address translation** provides **protection** from interference by other programs.

An individual block of virtual memory is referred to as a **page**. To locate a page, the processor references the program's **page table**. This index of address translations is different for every program. The OS creates a **swap space** to store all pages for a program. This data structure is often included in the page table. Some processors use a **TLB** (translation-lookaside buffer) to streamline memory access by avoiding the page table.

*The alternative to paging is **segmentation**, which we will discuss in Unit 5.4.*

In virtual memory, a miss is known as a **page fault**. To avoid costly page faults, we must replace pages effectively. An **LRU** (least recently used) **replacement scheme** is the most widely used method. Most machines use a **reference bit** to calculate LRU more accurately. While LRU is not the most accurate replacement scheme, it is efficient.

Reading

2 Read the textbook chapter. Then, mark the following statements as true (T) or false (F).

1 What is the passage mostly about?

A troubleshooting problems with virtual memory

B the purposes of different virtual memory elements

C a comparison of types of virtual memory

D how to improve virtual memory on older machines

2 According to the passage, what is NOT true of page tables?

A They contain a list of address translations.

B They may contain an index of the swap space.

C They keep ongoing records of page faults.

D They are unique to a particular program.

3 How can programmers minimize page faults?

A implement LRU replacement

B reference the TLB instead of the page table

C create a well-defined swap space

D update the page table regularly

Vocabulary

3 Match the words (1-9) with the definitions (A-I).

1 __ TLB

2 __ share

3 __ protection

4 __ swap space

5 __ reference bit

6 __ segmentation

7 __ address space

8 __ physical address

9 __ LRU replacement scheme

A a field that indicates whether a page was recently accessed

B a memory location for a specific program

C an area of the disk set aside for virtual pages of a process

D to allow others to use something at the same time

E a variable-size address mapping setup

F a cache containing recent address translations

G the process of ensuring processes cannot interfere with each other

H a memory address within the main memory

I a method for changing out blocks or pages in a cache based on use

4 **Write a word or phrase that is similar in meaning to the underlined part.**

1 The basis of virtual memory is the process of converting a virtual address to a physical address.
_ d _ _ e _ s t _ _ n s _ _ _ i _ n]

2 In order to find a page, we reference the index of virtual and physical addresses. _ _ g _ _ a b _ _

3 The act of using main memory as a cache makes programming easier. v _ _ _ u a _ _ _ _ o r _

4 The processor found the block of virtual memory in the main memory. _ _ _ e

5 The address that the program sees is not the actual memory location. _ i r _ u _ l _ _ d r _ s _

6 A situation in which the requested page is not in the memory comes with a high penalty. p _ g _ _ a u _ _

5 **Listen and read the textbook chapter again. How are programs kept isolated?**

Listening

6 **Listen to a conversation between two students. Mark the following statements as true (T) or false (F).**

1 __ The students are reviewing the results of a recent test.
2 __ Processors access virtual addresses from the page table.
3 __ LRU replacement schemes use reference bits.

7 **Listen again and complete the conversation.**

Student 1: I don't think so. Professor Brown said it would only cover **1** ________.

Student 2: That's good. Will you quiz me on these **2** ________ ________ terms?

Student 1: Sure. What is the process of converting a virtual address to a **3** ________ ________?

Student 2: That's **4** ________ ________.

Student 1: Right. Okay, I have a question. What exactly is a **5** ________ ________?

Student 2: **6** ________ ________ ________ the address that the program sees. It corresponds to the physical address in the memory.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

Do you think ...?
I have a question ...
That refers to ...

Student A: You are a student. Talk to Student B about:

- an upcoming exam
- what concepts will be on the exam
- what concepts you are confused about

Student B: You are a student. Talk to Student A about an upcoming exam.

Writing

9 **Use the reading passage and conversation from Task 8 to write an email from a student to an instructor. Include: the concepts that will be on the upcoming test, what the student has studied, and what concepts are still unclear.**

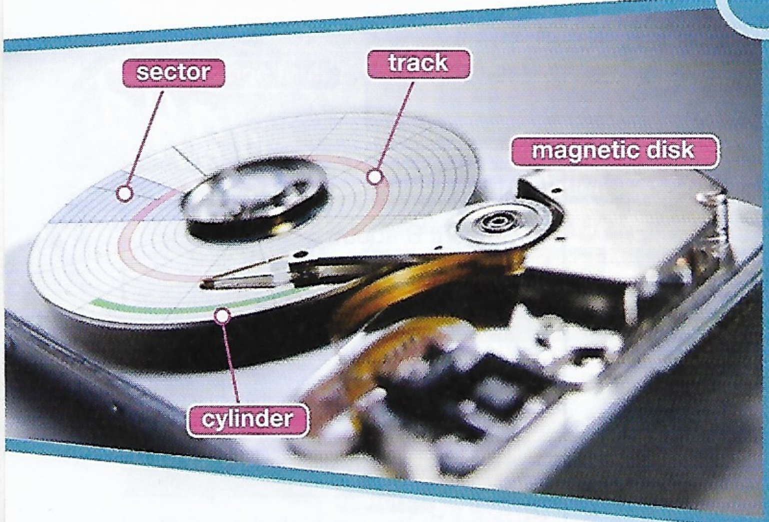
14 Disk Storage

Get ready!

1 Before you read the passage, talk about these questions.

1 How do magnetic disks organize data?

2 What kind of redundancy schemes are there for magnetic disks?



Reading

2 Read the journal article. Then, mark the following statements as true (T) or false (F).

1 What is the main idea of the article?

- **A** changes in disk storage methods over the years
- **B** the advantages of magnetic disk storage methods
- **C** challenges of using magnetic disk storage for secondary memory
- **D** ways to prevent disk storage failures

2 Which is NOT true of RAID configuration?

- **A** It is an efficient alternative to striping.
- **B** It sometimes requires hot swapping.
- **C** It uses standby spares to replace failed disks.
- **D** Its disks can be organized into protection groups.

3 Why is mirroring so expensive?

- **A** It requires the organization of additional protection groups.
- **B** It makes hot swapping necessary when disks fail.
- **C** It requires a duplicate disk for every data disk.
- **D** It is usually combined with the use of standby spares.

Excerpts from:

Magnetic Disk Storage and RAID Configurations

by ***Dr. Gerald Hart, Ph.D***
Article from the ***International Journal of Computer Hardware and Engineering***

Despite advancement in SSDs, **magnetic disks** are still the standard for secondary memory. With fast **seek times** and low **rotational latency**, disk storage is highly efficient.

One of the advantages of magnetic disk storage is its data organization. The disk is divided into **tracks**, and tracks are divided into **sectors**. Some older machines also reference **cylinders**. A **seek** positions the read/write head over the correct track or cylinder. Most magnetic disks have a dedicated **disk controller** to improve performance. Magnetic disks will remain useful as long as **controller time** remains low.

Redundancy schemes for magnetic disks are called **RAIDs** (redundant arrays of independent disks). RAID configurations are largely responsible for the practicality of magnetic disks. RAID 1, known as **mirroring**, is the most expensive RAID configuration. Mirroring requires a check disk for every active data disk. Other RAID configurations arrange data disks into **protection groups** to minimize hardware requirements. **Striping**, though referred to as RAID 0, has no actual redundancy.

No matter how efficient the RAID configuration, disks will fail and need replacement. While RAIDs usually prevent system failures, **hot swapping** is a risky process. In order to avoid shutting down the system, some machines use **standby spares**. The standby spares remain inactive until a primary disk fails.

Vocabulary

3 Match the words (1-7) with the definitions (A-G).

1 __ seek
2 __ track
3 __ cylinder
4 __ striping
5 __ mirroring
6 __ magnetic disk
7 __ rotational latency

- **A** all tracks that are underneath the read/write head
- **B** a type of nonvolatile memory that records data to rotating platters
- **C** the time required to move the correct sector under the read/write head
- **D** the process of distributing sequential blocks to separate disks
- **E** a single concentric circle on the surface of a disk
- **F** the process of recording identical data to two disks
- **G** the act of moving the read/write heads over the right track

4 **Read the sentence pairs. Choose which word or phrase best fits each blank.**

1 **sector / seek time**

A As disk technology advances, __________ decreases.

B Most magnetic disks can find the requested __________ quickly.

2 **RAID / disk controller**

A __________ is a method for increasing performance and reliability.

B A __________ handles instructions and operations for the disk.

3 **protection group / controller time**

A The engineers arranged redundancy with three disks to a __________.

B A high __________ can slow down the processor considerably.

4 **hot swapping / standby spare**

A A __________ remains inactive until a data disk fails.

B __________ places high demands on the system during replacement.

5 **Listen and read the journal article again. What is the advantage of using standby spares?**

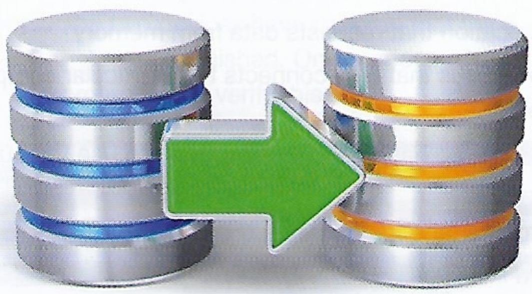
Listening

6 **Listen to a conversation between two computer engineers. Mark the following statements as true (T) or false (F).**

1 __ The engineers are deciding on a RAID scheme.

2 __ The woman would prefer to use mirroring.

3 __ The project will use standby spares instead of hot swapping.



7 **Listen again and complete the conversation.**

Engineer 1: Yeah, that's right. We know we'll be using **1** _________ _________. But we need to decide on the level of redundancy.

Engineer 2: Right. So we have to decide what **2** _______ scheme to use?

Engineer 1: Yes. What are your thoughts?

Engineer 2: Well, I think we should use **3** _______. It's the most reliable.

Engineer 1: **4** _______ _______. I don't think we can justify the cost of mirroring.

Engineer 2: But isn't it in budget? I **5** _______ _______ the budget proposal just a few minutes ago.

Engineer 1: You're forgetting about the **6** _______ _______. Part of that budget is needed for spare disks.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

We need to decide ...

I disagree ...

You're forgetting ...

Student A: You are an engineer. Talk to Student B about:

- disk storage for a new project
- what redundancy scheme to use
- why another scheme is not practical

Student B: You are an engineer. Talk to Student A about disk storage for a new project.

Writing

9 **Use the reading passage and conversation from Task 8 to write a report to a senior engineer. Include: the status of the new project, what disk configuration you plan to use, and why you chose that configuration.**

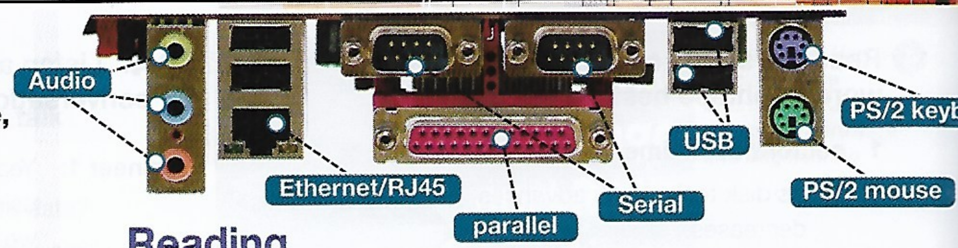
15 Buses

Get ready!

1 Before you read the passage, talk about these questions.

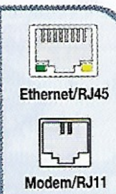
1 What are some different types of buses?

2 What is the difference between synchronous and asynchronous buses?



Buses (computing)

This is an article about computer interfaces.
For the transportation method, see Bus (vehicle).



In computing, a **bus** is an interface between different devices and subsystems. Buses are classified as either **serial buses** or **parallel buses**. The two types of buses transmit data differently. Some buses may be either parallel or serial. For example, an **SCSI** bus is typically parallel, but its protocol is sometimes implemented with serial buses.

Bus Transactions

A bus transaction begins with a request. It may contain several communications. Bus transactions can be grouped into two categories: **read transactions** and **write transactions**. The specifics of the transaction depend on the devices using the bus. Some devices, for instance, can only accommodate read transactions.

Types of Buses

Processor-memory buses are short, fast buses optimized for processor-memory communications. Despite their high speeds, they are only used to connect two devices. I/O buses are designed to connect many different peripherals and internal devices. I/O buses usually communicate with memory using a **backplane bus**.

Bus Communications

Bus communications are either **synchronous** or **asynchronous**. Synchronous buses are highly efficient. Devices connected to a synchronous bus must use the same clock rate.

Asynchronous buses use a **handshaking protocol** to coordinate data transmission. **FireWire** and **USB** 2.0 are common examples of asynchronous clocking. Some asynchronous buses use a **split transaction protocol** to increase effective bandwidth.

Are you an expert in this subject? You can help by expanding this article.

Reading

2 Read the online encyclopedia article. Then, mark the following statements as true (T) or false (F).

1 What is the purpose of the article?
- A to explain the history of buses in computers
- B to compare buses made by two different companies
- C to define some of the most common types of buses
- D to give instructions for troubleshooting bus errors

2 According to the article, what is NOT true of buses?
- A Parallel buses are more common than serial buses.
- B Buses perform read and write transactions.
- C Handshaking protocols are used in asynchronous buses.
- D Processor-memory buses only connect two devices.

3 What is the advantage of synchronous buses?
- A They are highly efficient.
- B They use a handshaking protocol.
- C They are designed to connect multiple peripherals.
- D They use a split-transaction protocol.

Vocabulary

3 Match the words (1-9) with the definitions (A-I).

1 __ USB
2 __ FireWire
3 __ backplane bus
4 __ bus transaction
5 __ read transaction
6 __ write transaction
7 __ handshaking protocol
8 __ processor-memory bus
9 __ split-transaction protocol

A a communication that records data to memory
B a system in which both devices agree when to move to the next step
C a bus that connects processors, memory, and I/O devices
D a communication that requests data from memory
E a high-speed bus that only connects two particular computer components
F a system that can handle multiple requests to use the bus at one time
G a standard interface for high-speed communications
H a standard interface that is ideal for peripheral devices
I a series of communications that begins with a request

4 **Read the sentence pairs. Choose which word or phrase best fits each blank.**

1 **bus / SCSI**

A A(n) ____________ is a communication link between devices.

B The disk drives in the PC use ____________ connections.

2 **parallel bus / serial bus**

A A ____________ sends data one bit at a time.

B A ____________ sends multiple bits at a time.

3 **synchronous bus / asynchronous bus**

A A(n) ____________ uses a handshaking protocol.

B A(n) ____________ times communications with an internal clock.

5 **Listen and read the online encyclopedia article again. What is the difference between an I/O bus and a processor-memory bus?**

Listening

6 **Listen to a conversation between an intern and a computer engineer. Mark the following statements as true (T) or false (F).**

1 __ The man incorrectly identifies the purpose of asynchronous buses.

2 __ USB is a type of asynchronous bus.

3 __ The woman gives examples of parallel buses.

7 **Listen again and complete the conversation.**

Engineer: Okay. Synchronous buses are what we use for 1 _______ - _______ _______.

Intern: Right. And when do we use 2 _______ _______?

Engineer: Well, asynchronous buses are useful for a wider variety of purposes. 3 _______ is a good example.

Intern: So would 4 _______ also be an asynchronous bus?

Engineer: *Yes. And that also uses a* 5 _______ _______.

Intern: *I can't remember how a handshaking protocol works.*

Engineer: The two devices have to agree that the 6 _______ _______ is finished. One purpose of a handshaking protocol is to verify this.

Speaking

8 **With a partner, act out the roles below based on Task 7. Then, switch roles.**

USE LANGUAGE SUCH AS:

Do you have ...?

What do you need?

Let me get this straight ...

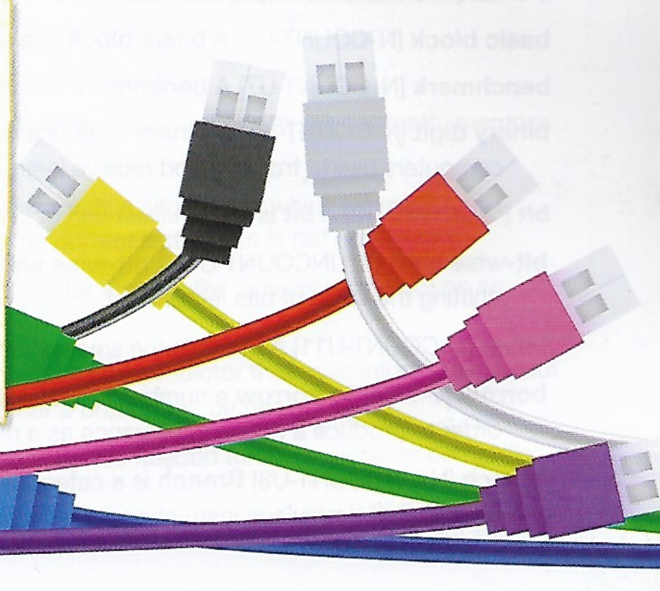
Student A: You are an intern. Talk to Student B about:

- bus communications
- the differences between types of buses
- when particular buses are used

Student B: You are an engineer. Talk to Student A about bus communications.

Writing

9 **Use the reading passage and conversation from Task 8 to write an email to a supervising engineer. Include: topics from a previous conversation, concepts that are clear, and concepts that are still unclear.**



Glossary

access [V-T-U12] To **access** something is to locate it and make it available for use.

access time [N-COUNT-U11] An **access time** is the amount of time required to obtain information from a computer's memory.

accurate [ADJ-U5] If a calculation is **accurate**, it is correct and exact.

adder [N-COUNT-U8] An **adder** is a digital circuit that carries out addition operations.

addition [N-COUNT-U4] **Addition** is the process of finding the sum of two or more numbers.

address [N-COUNT-U2] An **address** is the location of specific information within the computer's memory.

address space [N-COUNT-U13] An **address space** is a designated list of memory locations that are available only to a specific program.

address translation [N-UNCOUNT-U13] **Address translation** is the process by which a virtual address is redirected to a physical address.

ALU [ABBREV-U8] An **ALU** (arithmetic logic unit) is a type of digital circuit that carries out arithmetic and logical operations.

Amdahl's law [N-UNCOUNT-U7] **Amdahl's law** is an equation that determines the maximum overall improvement to a system if only one aspect of the system is changed.

application [N-COUNT-U7] An **application** is a real program that is part of a benchmark.

approximation [N-COUNT-U5] An **approximation** is a useful representation of a number that is not exact, but comes as close as possible under the circumstances.

arithmetic mean [N-COUNT-U7] An **arithmetic mean** is the average of execution times compared with total execution time.

arithmetic-logical [N-UNCOUNT-U8] **Arithmetic-logical** is a category of instruction that tells the CPU to carry out mathematical and logical operations.

assembler [N-COUNT-U1] An **assembler** is a program that changes written instructions into the binary translation.

assembly language [N-COUNT-U1] An **assembly language** is a form of written instructions for a computer that is simpler than high-level or human-readable programming languages but has not been converted to a binary translation.

asynchronous [ADJ-U15] **Asynchronous** is a bus that does not have a clock and instead relies on a handshaking protocol to time transactions.

backplane bus [N-COUNT-U15] A **backplane bus** is a single bus that connects processors, memory, and I/O devices.

base 10 [N-UNCOUNT-U3] **Base 10** is a number system, also called the decimal system, that uses the digits 0 through 9.

base 2 [N-UNCOUNT-U3] **Base 2** is a number system, also called the binary system, that uses the digits 0 and 1.

basic block [N-COUNT-U2] A **basic block** is a series of instructions that do not have branches.

benchmark [N-COUNT-U7] A **benchmark** is a workload that measures computer performance.

binary digit [N-COUNT-U1] A **binary digit** is a number, represented by either 0 or 1, that makes up the language that computers use to transmit and receive instructions.

bit [N-COUNT-U2] A **bit** is the smallest unit of information on a computer.

bit-wise shift [N-UNCOUNT-U4] A **bit-wise shift** is an operation that performs multiplication and division quickly by shifting the value of bits left or right.

block [N-COUNT-U11] A **block** is the smallest unit of information that can be present or absent within a level of memory.

borrow [V-T-U4] To **borrow** a number in subtraction is to take a number, usually 10, from the next higher digit column in order to produce a positive difference as a result.

branch [N-UNCOUNT-U8] **Branch** is a category of instruction that alters the next instruction stored in the PC based on the result of a previous instruction.

branch delay slot [N-COUNT-U10] A **branch delay slot** is an instruction slot that comes after a delayed branch instruction and contains an instruction that does not have any effect on the branch.

branch hazard [N-COUNT-U9] A **branch hazard**, also called a control hazard, is a situation in which a branch instruction is dependent on information that is not available yet, and the correct instruction is not carried out.

branch history table [N-COUNT-U10] A **branch history table**, also called a branch prediction buffer, is a small memory that records whether or not a branch was recently taken.

branch prediction [N-UNCOUNT-U9] **Branch prediction** is the act of guessing whether a branch will be taken or untaken in order to avoid branch hazards.

branch prediction buffer [N-COUNT-U10] A **branch prediction buffer**, also called a branch history table, is a small memory that records whether or not a branch was recently taken.

branch target buffer [N-COUNT-U10] A **branch target buffer** is a cache memory that stores the necessary destination instructions for a branch.

bubble [N-COUNT-U10] A **bubble**, also called a pipeline stall, is an intentional delay implemented to resolve hazards.

bus [N-COUNT-U15] A **bus** is a communication link that is shared between multiple devices.

bus transaction [N-COUNT-U15] A **bus transaction** is a series of bus communications that always contains a request and may or may not contain a response.

C [ABBREV-U1] **C** is a human-readable programming language that is focused on procedures and used for general purposes.

cache [N-COUNT-U12] A **cache** is a small memory that contains the data most likely to be requested.

cache miss [N-COUNT-U12] A **cache miss** is a situation in which a cache request cannot be completed because the requested data is not in the cache.

carry-out [N-COUNT-U4] A **carry-out** is a number that is carried from the right column to the left in a mathematical equation that is needed to get the final result of the operation.

clock cycle [N-COUNT-U6] A **clock cycle** is an interval of time that is used to measure the performance of a computer processor.

clock rate [N-COUNT-U6] A **clock rate** is the number of cycles per second a computer runs at.

compiler [N-COUNT-U1] A **compiler** is a computer program that converts complicated operations into simple computer instructions.

concurrently [ADV-U9] If two or more things are happening **concurrently**, they are happening at the same time.

conditional branch [N-COUNT-U2] A **conditional branch** is an instruction that is only completed if certain conditions are first met.

consistent [ADJ-U12] If two sources are **consistent**, they contain the same information.

control [N-COUNT-U8] A **control** is the part of a computer processor that delivers instructions to the datapath, memory, and other devices.

control hazard [N-COUNT-U9] A **control hazard**, also called a branch hazard, is a situation in which a branch instruction is dependent on information that is not available yet, and the correct instruction is not carried out.

controller time [N-UNCOUNT-U14] The **controller time** is the time required for a controller to receive and act on its instructions.

correlating predictor [N-COUNT-U10] A **correlating predictor** is a type of branch predictor that uses information about recently taken branches on the local and global scale to predict whether a branch will be taken.

CPI [ABBREV-U6] **CPI** (clock cycles per instructions) is the number of clock cycles needed for a computer to complete an instruction.

Glossary

CPU time [N-UNCOUNT-U6] **CPU time** is the amount of time the central processing unit (CPU) of a computer takes to complete a task.

cylinder [N-COUNT-U14] A **cylinder** is all of the tracks that are underneath a magnetic disk's read/write heads at any given time.

data [N-UNCOUNT-U2] **Data** is information stored in a computer.

data hazard [N-COUNT-U9] A **data hazard** is a situation in which a pipeline stalls because the data needed for an instruction is still being processed.

data selector [N-COUNT-U8] A **data selector**, also called a multiplexer, is a device that chooses one of several input signals and routes it to a single available output line.

data transfer instruction [N-COUNT-U2] A **data transfer instruction** is an operation on a computer that allows data to be transferred from memory to registers.

datapath [N-COUNT-U8] A **datapath** is a series of units that are involved in data processing operations.

destination [N-COUNT-U8] A **destination** is the location to which information is sent.

diminishing returns [N-UNCOUNT-U7] **Diminishing returns** is the principle that performance or production will decrease when a production factor is increased too much.

direct-mapped cache [N-COUNT-U12] A **direct-mapped cache** is a cache in which individual memory locations are assigned a specific location in the cache.

disk controller [N-COUNT-U14] A **disk controller** is a device that handles the physical operations of a magnetic disk and the transfer of data from disk to memory.

division [N-COUNT-U4] **Division** is the process of splitting a quantity into a particular number of equal parts.

double precision [N-UNCOUNT-U5] **Double precision** is the expression of a floating point value in two 32-bit words in order to avoid overflow and underflow.

dynamic branch prediction [N-UNCOUNT-U10] **Dynamic branch prediction** is the process of predicting whether or not a branch will be taken by finding out if the branch was taken the last time the instruction was executed.

exception [N-COUNT-U4] An **exception**, also called an interrupt, is an event that disrupts the execution of a program.

execution time [N-UNCOUNT-U6] **Execution time** is the time that elapses from the start of a task to the end.

exponent [N-COUNT-U5] An **exponent** is a number that indicates how many times a quantity is multiplied by itself.

FireWire [N-UNCOUNT-U15] **FireWire** is a standard serial bus interface that is optimized for high-speed communications.

floating point [N-UNCOUNT-U5] **Floating point** is a kind of computer arithmetic that uses a variable binary point.

flush instructions [V PHRASE-U10] To **flush instructions** is to discard all current instructions from a pipeline, usually done in when an unexpected branching event occurs.

forwarding [N-UNCOUNT-U9] **Forwarding** is a process that avoids data hazards by retrieving missing data from internal buffers before it is available in registers or memory.

fully associative cache [N-COUNT-U12] A **fully associative cache** is a cache in which any block can be placed in any location within the cache.

guard digit [N-COUNT-U5] A **guard digit** is an extra bit to the right of the binary point that allows for more accurate rounding.

handle [V-T-U12] To **handle** a task is to perform the necessary actions to complete it.

handshaking protocol [N-COUNT-U15] A **handshaking protocol** is a system of coordination for asynchronous buses in which devices only proceed to the next step of the process after both have agreed that the current step is finished.

hazard [N-COUNT-U9] A **hazard** is a pipelining situation in which the next instruction cannot be executed in the next CPU clock cycle.

human-readable programming language [N-COUNT-U1] A **human-readable programming language** is a computer language that is compatible with the way people think and is used by programmers to write instructions for a computer.

hit [N-COUNT-U11] A **hit** is a situation in which requested data is present in a block in the upper level of a memory hierarchy.

hit rate [N-COUNT-U11] A **hit rate** is the percentage of memory accesses found on the upper level of a memory hierarchy, usually expressed as a fraction.

hit time [N-COUNT-U11] A **hit time** is the amount of time needed to access a level of the memory and determine whether the requested data is present in that level.

hot swapping [N-UNCOUNT-U14] **Hot swapping** is the act of replacing a hardware device while the rest of the machine is still running.

ignore [V-T-U4] To **ignore** something is to intentionally disregard it.

implementation [N-COUNT-U8] **Implementation** is the process of carrying out a task in a certain way.

infinite [ADJ-U5] If a number is **infinite**, it has no limitations on its value.

instruction [N-COUNT-U2] **Instructions** are the words that make up computer language.

instruction class [N-COUNT-U8] An **instruction class** is the general category under which an instruction falls.

instruction set [N-COUNT-U2] An **instruction set** is a specific set of words that prompts a computer to perform an action.

integer [N-COUNT-U5] An **integer** is a natural number, the negative of a natural number, or zero.

interrupt [N-COUNT-U4] An **interrupt**, also called an exception, is an event that disrupts the execution of a program.

Java [N-UNCOUNT-U1] **Java** is a human-readable programming language that is similar to C but modified to be object-oriented and simpler.

latency [N-UNCOUNT-U9] **Latency** is the time required to execute an individual instruction.

leading 0 [N-COUNT-U3] A **leading 0** is a digit at the beginning of a signed binary number that indicates it is positive.

leading 1 [N-COUNT-U3] A **leading 1** is a digit at the beginning of a signed binary number that indicates it is negative.

least significant bit [N-COUNT-U3] A **least significant bit** is a binary digit that is farthest to the right in a word.

load-use data hazard [N-COUNT-U9] A **load-use data hazard** is a situation that occurs when a load instruction requests data that is not available yet.

LRU replacement scheme [N-COUNT-U13] An **LRU** (least recently used) **replacement scheme** is a method for replacing blocks in the cache that involves removing the block that has been unused for the longest amount of time.

machine language [N-COUNT-U1] A **machine language** is a set of instructions written in numerical form.

magnetic disk [N-COUNT-U14] A **magnetic disk** is a form of nonvolatile memory that records data to multiple rotating magnetic platters.

memory hierarchy [N-COUNT-U11] A **memory hierarchy** is a system for organizing memory in which multiple tiers of memory are used, with each level increasing in size and access time relative to the distance from the CPU.

memory-reference [N-UNCOUNT-U8] **Memory-reference** is a category of instruction that tells the CPU to either retrieve data from or store data to memory.

metric [N-COUNT-U6] A **metric** is a measurement of a certain aspect of something's performance.

MIPS [ABBREV-U7] **MIPS** (million instructions per second) are a measurement of the execution speed of a program by the millions of instructions that are executed every second.

mirroring [N-UNCOUNT-U14] **Mirroring** is the process of recording identical data to both a primary disk and a redundant disk to increase data availability.

miss penalty [N-COUNT-U11] A **miss penalty** is the amount of time required to locate and transfer a block from a lower level of a memory hierarchy to an upper level, including the time needed to send the data to the processor.

Glossary

miss rate [N-COUNT-U11] A **miss rate** is the percentage of memory accesses not found on the upper level of a memory hierarchy, usually expressed as a fraction and calculated as 1 minus the hit rate.

most significant bit [N-COUNT-U3] The **most significant bit** is the binary digit that is farthest to the left in the word.

multiplexer [N-COUNT-U8] A **multiplexer** (MUX), also called a data selector, is a device that chooses one of several input signals and routes it to a single available output line.

multiplication [N-COUNT-U4] **Multiplication** is the process of adding a quantity to itself a particular number of times.

NOP [ABBREV-U10] A **NOP** (No Operation) is an instruction that has no effect and is used to avoid hazards and provide instructions to unused stages of the pipeline.

normalized [ADJ-U5] If a number in scientific notation is **normalized**, it does not have a leading zero.

number base [N-COUNT-U3] A **number base** is the indication of how many digits or numerals are used in a certain system.

operand [N-COUNT-U4] An **operand** is the number that is used in a mathematical equation.

overflow [N-UNCOUNT-U4] **Overflow** is a condition that occurs when the result of a calculation is too large for the storage system of the computer.

page [N-COUNT-U13] A **page** is a fixed-size block of virtual memory.

page fault [N-COUNT-U13] A **page fault** is an occurrence in which a requested page is not present in the main memory.

page table [N-COUNT-U13] A **page table** is a table stored in the memory that contains a list of virtual addresses and the corresponding physical addresses.

parallel [ADJ-U12] If two things are **parallel**, they correspond to each other or exist side-by-side.

parallel bus [N-COUNT-U15] A **parallel bus** is a bus that sends two sets of data simultaneously on parallel wires.

PC [ABBREV-U8] A **PC** (program counter) is a small register that keeps track of progress through program instructions by storing the address of the next instruction.

performance [N-UNCOUNT-U6] **Performance** is the amount of work something can do and the time needed to accomplish it.

physical address [N-COUNT-U13] A **physical address** is a memory address within the main memory.

pipeline stall [N-COUNT-U9] A **pipeline stall**, also called a bubble, is an intentional delay implemented to resolve hazards.

pipelining [N-UNCOUNT-U9] **Pipelining** is a technique for implementing instructions in which multiple instructions are executed simultaneously.

principle of locality [N-UNCOUNT-U11] The **principle of locality** is a concept that states that programs only use a small percentage of the available memory address space at any one time.

processor-memory bus [N-COUNT-U15] A **processor-memory bus** is a short, high speed bus that is optimized to connect processors to memory.

programmer [N-COUNT-U1] A **programmer** is a person who writes and develops software and programs for computers.

protection [N-COUNT-U13] **Protection** is a series of measures taken to ensure that the different processes that share a device cannot interfere with each other.

protection group [N-COUNT-U14] A **protection group** is a collection of disks that share the same redundant or check disk.

queue [N-COUNT-U12] A **queue** is a series of objects or blocks of information that are processed in sequential order.

RAID [ABBREV-U14] **RAID** (redundant arrays of independent disks) is a way of organizing disk space by using several small, independent disks as opposed to a smaller number of large disks to improve reliability and performance.

read transaction [N-COUNT-U15] A **read transaction** is a bus transaction that retrieves data from memory.

recognize [V-T-U4] To **recognize** something is to notice or acknowledge it.

reference [V-T-U11] To **reference** something is to open or recall it from its data location.

reference bit [N-COUNT-U13] A **reference bit** is a field in the cache that indicates whether or not a block of memory has been accessed recently.

register [N-COUNT-U2] A **register** is a part of the computer's hardware that temporarily stores instructions sent to the computer, allowing instructions to be accessed more quickly.

reproducibility [N-COUNT-U7] **Reproducibility** is the ability to duplicate something.

result [N-COUNT-U4] A **result** is the final product or answer after a process is complete.

rotational latency [N-COUNT-U14] **Rotational latency** is the amount of time required for a correct sector to rotate under the read/write head after it is positioned over the right track.

round [V-T-U5] To **round** a number is to express it as a number that is as close as possible, but only as accurate as is useful.

scientific notation [N-COUNT-U5] **Scientific notation** is a way of writing a number with only one digit to the left of the decimal point, multiplied by ten raised to an exponent. For example, 4,000 = 4.0 x 103.

SCSI [ABBREV-U15] The **SCSI** (small computer system interface) is a set of standards for communication between computers and peripheral devices.

sector [N-COUNT-U14] A **sector** is the smallest unit of a track that can contain data, and is usually 512 bytes in size.

seek [N-COUNT-U14] A **seek** is the act of physically moving a read/write head over the correct track on a disk.

seek time [N-COUNT-U14] A **seek time** is the amount of time required to move a read/write head into the correct position.

segmentation [N-COUNT-U13] **Segmentation** is a variable-size address mapping setup in which the address consists of a segment number and a segment offset.

serial bus [N-COUNT-U15] A **serial bus** is a bus that sends data one bit at a time.

set-associative cache [N-COUNT-U12] A **set-associative cache** is a cache that has a set number of locations in which any particular block can be placed.

share [V-T-U13] To **share** something is to allow others to use or experience it.

sign bit [N-COUNT-U3] A **sign bit** is the leading bit that is tested by computer hardware to indicate whether the number is positive or negative.

signed number [N-COUNT-U3] A **signed number** is a number that is marked as either positive or negative.

significand [N-COUNT-U5] A **significand** is part of a number in scientific notation or a floating point number consisting of its significant digits.

single precision [N-UNCOUNT-U5] **Single precision** is the expression of a floating point value in one 32-bit word.

source [N-COUNT-U8] A **source** is the location from which information originates.

spatial locality [N-COUNT-U11] **Spatial locality** is the principle that indicates that when a data location is referenced, addresses near it will likely be referenced soon.

SPEC CPU benchmark [N-COUNT-U7] A **SPEC CPU benchmark** is a set of real programs that measures the performance of the central processing unit.

SPEC ratio [N-COUNT-U7] A **SPEC ratio** is the measurement of the execution time of one computer compared to the execution time of another computer.

split cache [N-COUNT-U12] A **split cache** is a memory hierarchy in which a level of memory consists of two parallel caches, one for instructions and one for data.

split transaction protocol [N-COUNT-U15] A **split-transaction protocol** is a system that allows other requesters to access the bus while a previous requester is waiting for data to be sent.

Glossary

stage [N-COUNT-U9] A **stage** is one specific task or action in an overall process.

standby spare [N-COUNT-U14] A **standby spare** is a hardware device, usually a magnetic disk, that is already installed in the system but remains inactive unless the primary disk fails.

sticky bit [N-COUNT-U5] A **sticky bit** is an extra bit used in rounding whenever there is a number other than zero to the right of the round bit.

stored-program concept [N-COUNT-U2] The **stored-program concept** is a computing theory that states that instructions can be stored as numbers in the computer's memory.

striping [N-UNCOUNT-U14] **Striping** is the process of distributing sequential blocks of data onto separate disks with no redundancy.

structural hazard [N-COUNT-U9] A **structural hazard** is a situation in which hardware cannot accommodate the combination of instructions that are supposed to execute in a given time period.

subscript [V-T-U3] To **subscript** something is to add a distinguishing number or character to it.

subtraction [N-COUNT-U4] **Subtraction** is the process of deducting the amount of one number from the amount of another.

swap space [N-COUNT-U13] A **swap space** is an area of a disk that is designated for the virtual pages of a process.

synchronous [ADJ-U15] **Synchronous** is a bus that contains a clock and performs transactions relative to the clock.

system CPU time [N-COUNT-U6] A **system CPU time** is the amount of time a computer processor spends running the support system of a program.

systems software [N-UNCOUNT-U1] **Systems software** is a type of computer program, such as a compiler or assembler, that enables computer functions.

tag [N-COUNT-U12] A **tag** is a field in a memory hierarchy that identifies the contents of a block.

temporal locality [N-COUNT-U11] **Temporal locality** is the principle that indicates that when a data location is recently referenced, it will likely be referenced again soon.

throughput [N-COUNT-U6] **Throughput** is the amount of work a computer can do in a specific amount of time.

TLB [ABBREV-U13] A **TLB** (translation-lookaside buffer) is a cache that keeps record of recently used address translations in order to reduce use of the page table.

tournament branch predictor [N-COUNT-U10] A **tournament branch predictor** is an elaborate branch predictor that has multiple prediction types from which the program can choose.

track [N-COUNT-U14] A **track** is a single concentric circle on the recording surface of a magnetic disk.

translate [V-T-U1] To **translate** something is to convert it from one form to another.

two's complement [N-UNCOUNT-U3] **Two's complement** is a system of signed binary numbers using leading 0 and leading 1.

ULP [ABBREV-U5] A **ULP** (unit of least precision) is the measure of the degree of error in rounding.

underflow [N-COUNT-U5] **Underflow** is an occurrence in which a negative exponent is too large to fit in the exponent field of a 32-bit word.

unsigned number [N-COUNT-U3] An **unsigned number** is a number that does not have a negative or a positive sign, so it can only represent zero or a positive number.

untaken branch [N-COUNT-U9] An **untaken branch** is a branch instruction that yields to the next sequential instruction rather than routing to a new instruction address.

USB [ABBREV-U15] A **USB** (universal serial bus) is a standard serial bus interface that is ideal for lower-performance peripheral devices.

user CPU time [N-COUNT-U6] A **user CPU time** is the amount of time a computer processor spends running a program.

valid bit [N-COUNT-U12] A **valid bit** is a field in a memory hierarchy that is set when the block contains a valid address.

value [N-COUNT-U4] A **value** is a number, and can be either positive or negative.

virtual address [N-COUNT-U13] A **virtual address** is an address that matches up to a virtual memory space and redirects to a physical address when the memory is requested.

virtual memory [N-UNCOUNT-U13] **Virtual memory** is a data storage technique that uses the main memory as a cache for a secondary memory storage system.

wall-clock time [N-UNCOUNT-U6] **Wall-clock time** is the most commonly accepted notion of the passage of time, consisting of measurements in minutes and seconds.

weighted arithmetic mean [N-COUNT-U7] A **weighted arithmetic mean** is a sum of weighting factors and execution times that is used to evaluate the performance of the workload.

weighting factor [N-COUNT-U7] A **weighting factor** is the percentage of usage that a program in a workload has.

word [N-COUNT-U2] A **word** is a group of 32 bits.

workload [N-COUNT-U7] A **workload** is a set of programs that a computer runs on a daily basis.

write buffer [N-COUNT-U12] A **write buffer** is a queue that holds data that is already written to the cache but is waiting to be written to memory.

write transaction [N-COUNT-U15] A **write transaction** is a bus transaction that records data to memory.

write-back [N-COUNT-U12] **Write-back** is a process in which new data is only written to the cache, and is written to the memory when the block in the cache is replaced.

write-through [N-COUNT-U12] **Write-through** is a process in which the cache and the memory are updated at the same time to ensure they are consistent with each other.