Data Storage

Slides derived from those available on the web site of the book:

Computer Science: An Overview, 11th Edition, by J. Glenn Brookshear

Data Storage

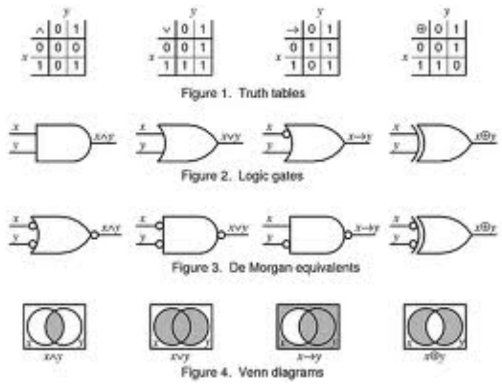
- Bits and Their Storage
- Main Memory
- Mass Storage
- Representing Information as Bit Patterns
- The Binary System
- Data Compression
- Communications Errors

Bits and Bit Patterns

- **Bit:** Binary Digit (0 or 1)
- Bit Patterns are used to represent information.
  - Numbers
  - Text characters
  - Images
  - Sound
  - And others

Boolean Operations

- **Boolean Operation:** An operation that manipulates one or more true/false values
- Specific operations
  - AND
  - OR
  - XOR (exclusive or)
  - NOT



The Boolean operations AND, OR, and XOR (exclusive or)

The AND operation

$$\begin{array}{lr} & 0 \\ \text{AND} & 0 \\ \hline & 0 \end{array} \qquad \begin{array}{lr} & 0 \\ \text{AND} & 1 \\ \hline & 0 \end{array} \qquad \begin{array}{lr} & 1 \\ \text{AND} & 0 \\ \hline & 0 \end{array} \qquad \begin{array}{lr} & 1 \\ \text{AND} & 1 \\ \hline & 1 \end{array}$$

The OR operation

$$\begin{array}{lr} & 0 \\ \text{OR} & 0 \\ \hline & 0 \end{array} \qquad \begin{array}{lr} & 0 \\ \text{OR} & 1 \\ \hline & 1 \end{array} \qquad \begin{array}{lr} & 1 \\ \text{OR} & 0 \\ \hline & 1 \end{array} \qquad \begin{array}{lr} & 1 \\ \text{OR} & 1 \\ \hline & 1 \end{array}$$

The XOR operation

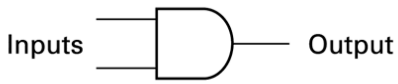
$$\begin{array}{lr} & 0 \\ \text{XOR} & 0 \\ \hline & 0 \end{array} \qquad \begin{array}{lr} & 0 \\ \text{XOR} & 1 \\ \hline & 1 \end{array} \qquad \begin{array}{lr} & 1 \\ \text{XOR} & 0 \\ \hline & 1 \end{array} \qquad \begin{array}{lr} & 1 \\ \text{XOR} & 1 \\ \hline & 0 \end{array}$$

Gates

- **Gate:** A device that computes a Boolean operation
  - Often implemented as (small) electronic circuits
  - Provide the building blocks from which computers are constructed
  - VLSI (Very Large Scale Integration)

A pictorial representation of AND, OR, XOR, and NOT gates

AND

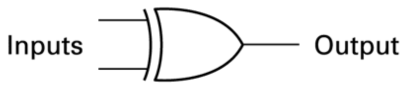


Inputs	Output
0 0	0
0 1	0
1 0	0
1 1	1

OR

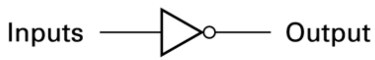
Inputs	Output
0 0	0
0 1	1
1 0	1
1 1	1

XOR



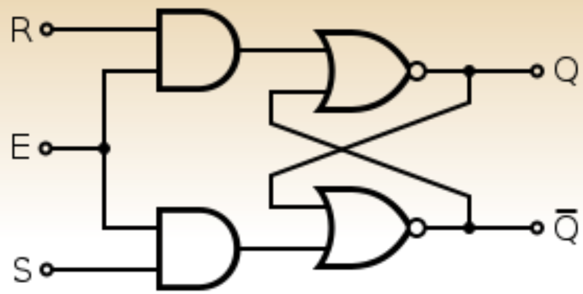
Inputs	Output
0 0	0
0 1	1
1 0	1
1 1	0

NOT

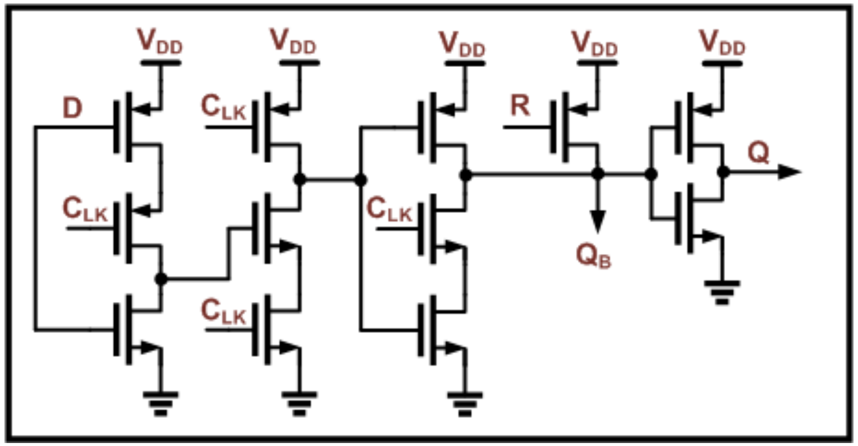


Inputs	Output
0	1
1	0

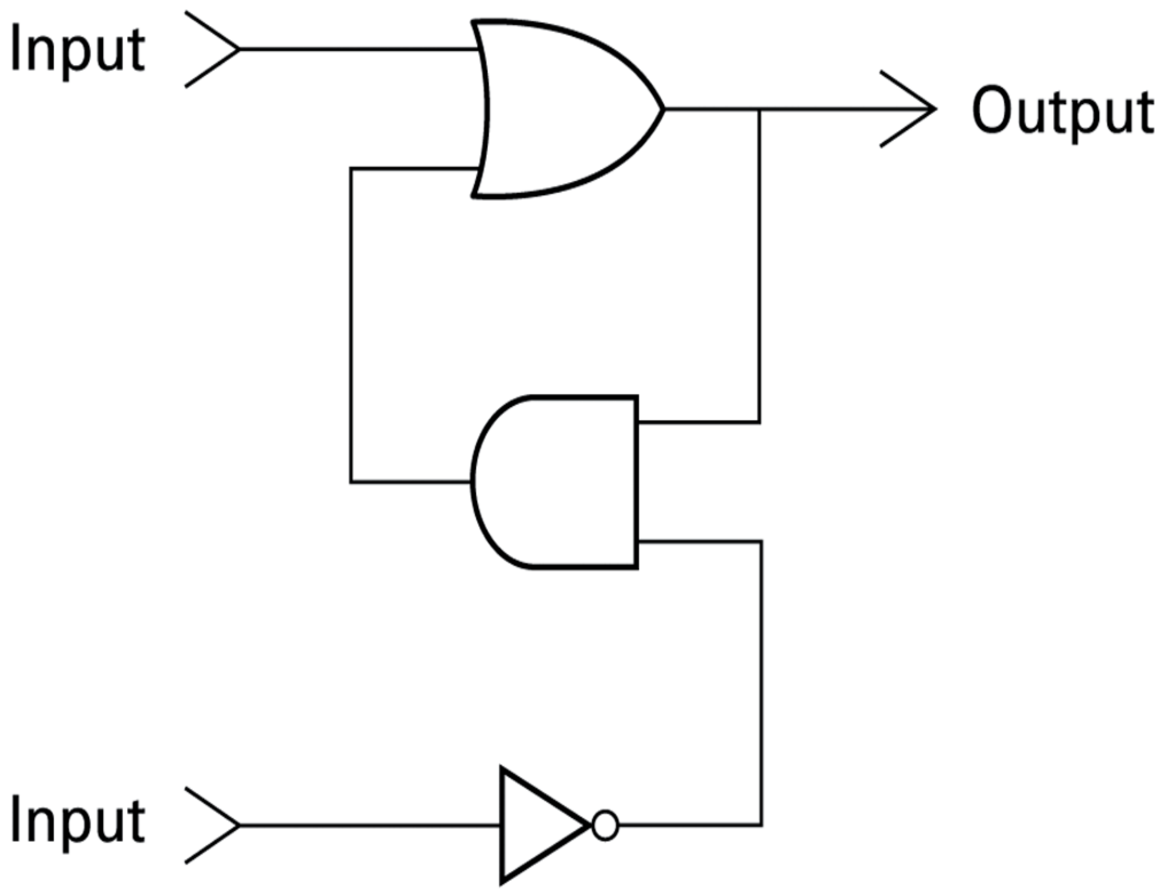
Flip-flops



- **Flip-flop:** A circuit built from gates that can store one bit.
  - One input line is used to set its stored value to 1
  - One input line is used to set its stored value to 0
  - While both input lines are 0, the most recently stored value is preserved

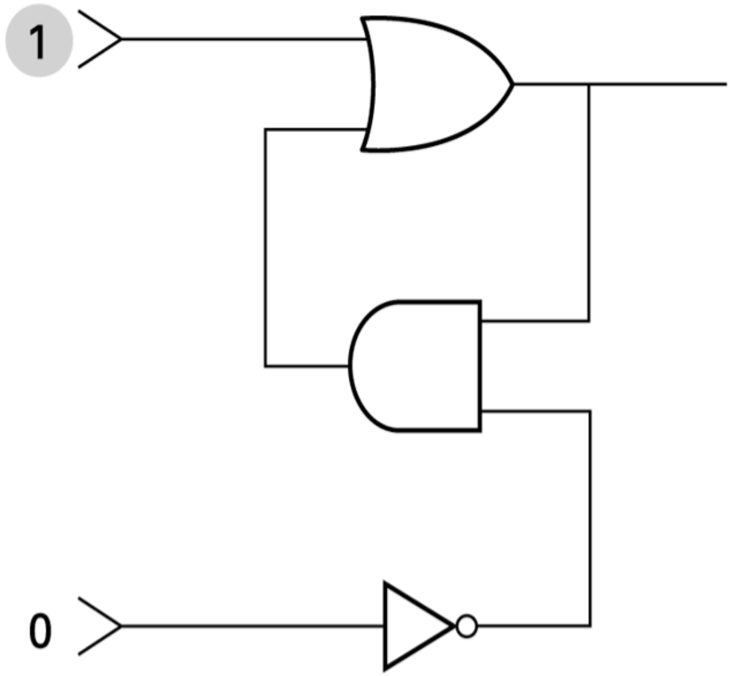


A simple flip-flop circuit



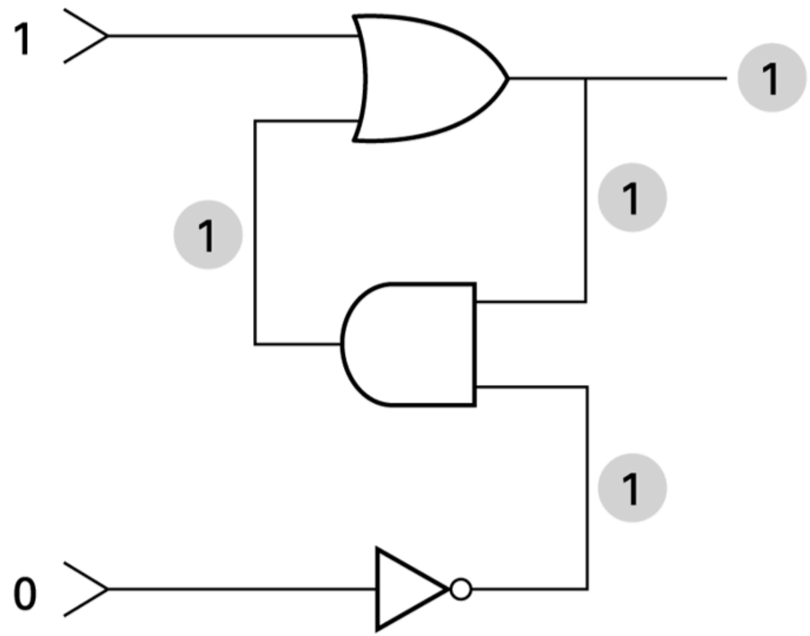
Setting the output of a flip-flop to 1

a. 1 is placed on the upper input.



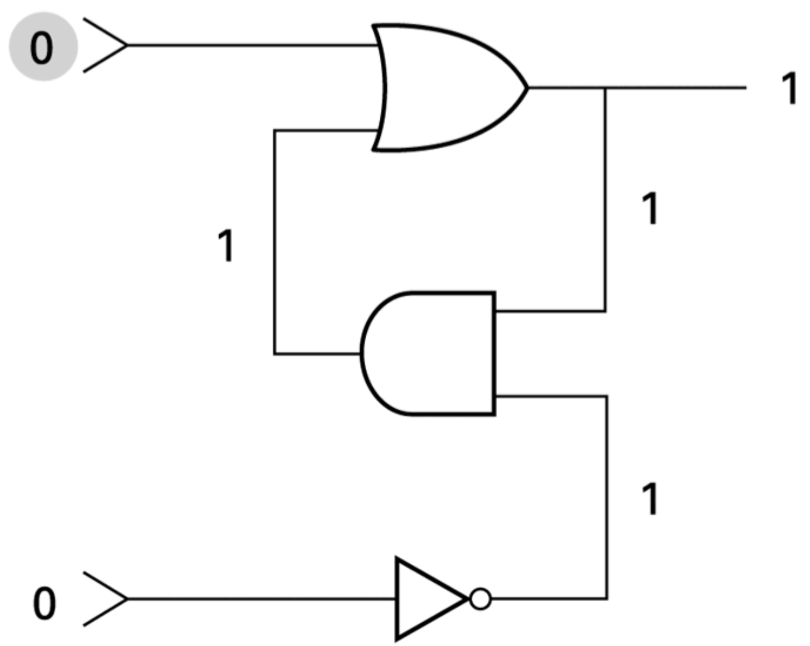
Setting the output of a flip-flop to 1 (continued)

b. This causes the output of the OR gate to be 1 and, in turn, the output of the AND gate to be 1.

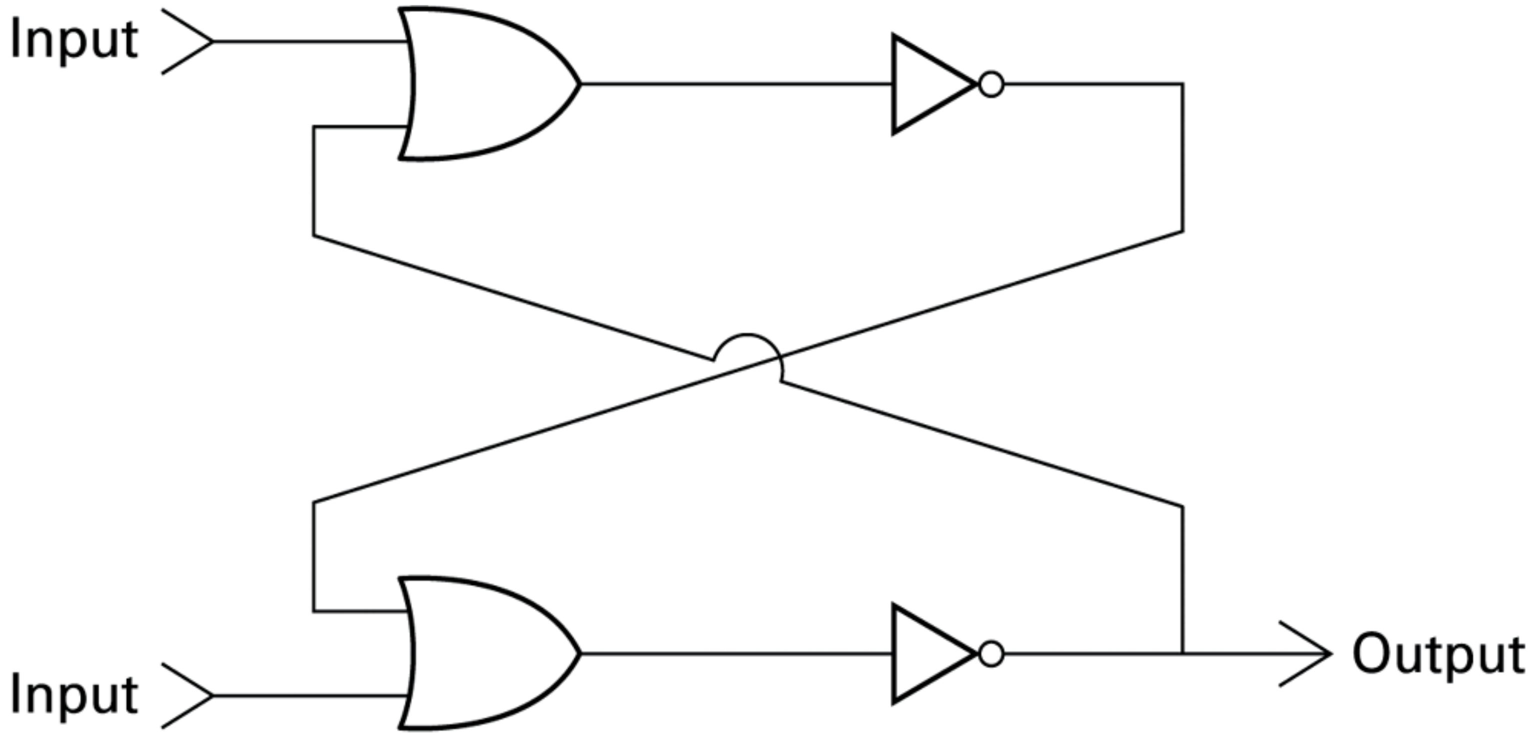


Setting the output of a flip-flop to 1 (continued)

c. The 1 from the AND gate keeps the OR gate from changing after the upper input returns to 0.



Another way of constructing a flip-flop



Hexadecimal Notation

- **Hexadecimal notation:** A shorthand notation for long bit patterns
  - Divides a pattern into groups of four bits each
  - Represents each group by a single symbol
- Example: 10100011 becomes A3

The hexadecimal coding system

Bit pattern	Hexadecimal representation
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

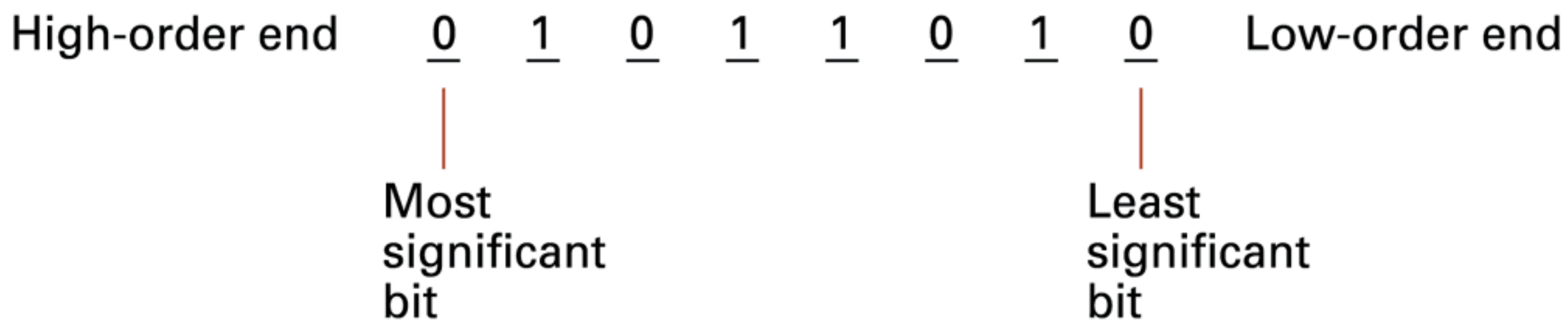
Data Storage

- Bits and Their Storage
- Main Memory
- Mass Storage
- Representing Information as Bit Patterns
- The Binary System
- Data Compression
- Communications Errors

Main Memory Cells

- **Cell:** A unit of main memory (typically 8 bits which is one **byte**)
  - **Most significant bit:** the bit at the left (high-order) end of the conceptual row of bits in a memory cell
  - **Least significant bit:** the bit at the right (low-order) end of the conceptual row of bits in a memory cell

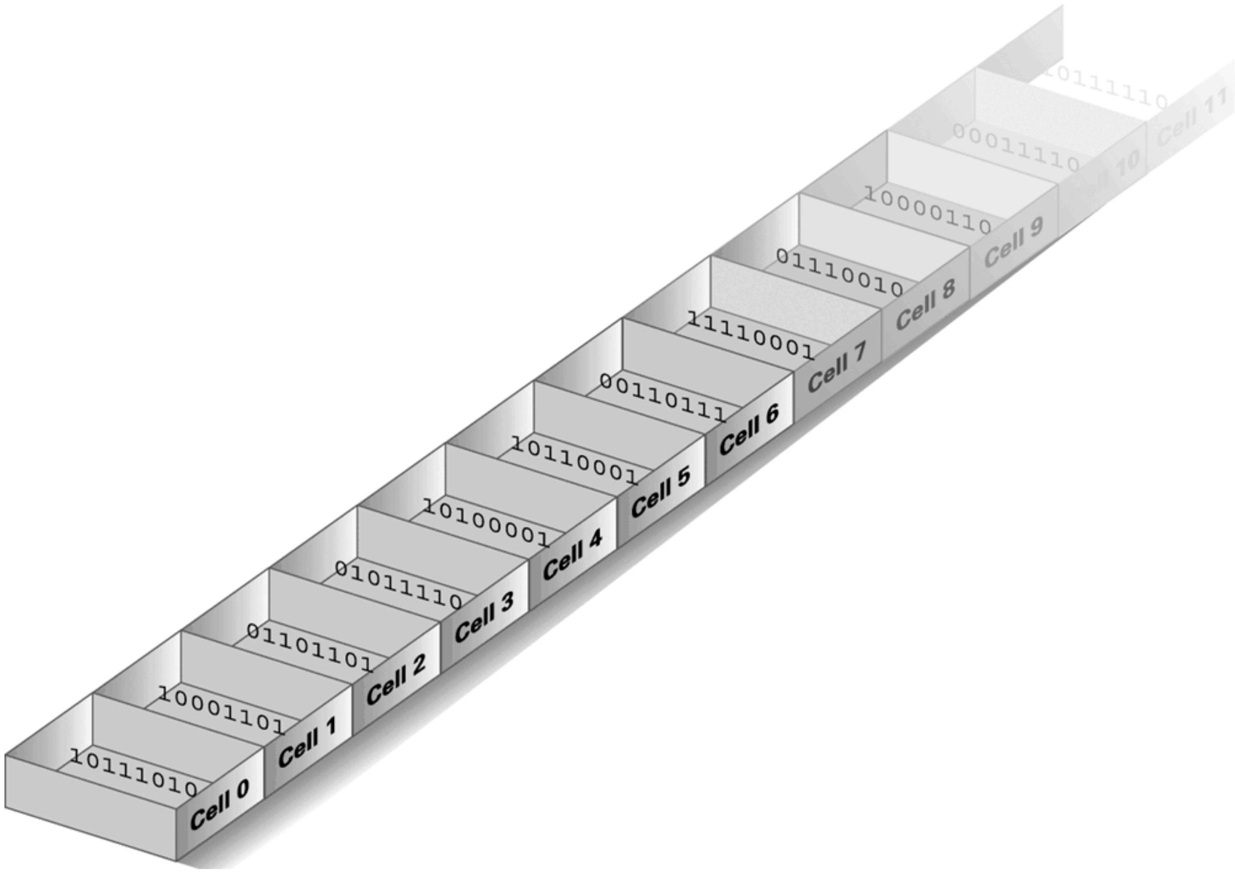
The organization of a byte-size memory cell



Main Memory Addresses

- **Address:** A “name” that uniquely identifies one cell in the computer’s main memory
  - The names are actually numbers.
  - These numbers are assigned consecutively starting at zero.
  - Numbering the cells in this manner associates an order with the memory cells.

Memory cells arranged by address



Memory Terminology

- **Random Access Memory (RAM):** Memory in which individual cells can be easily accessed in any order
- **Dynamic Memory (DRAM):** RAM composed of volatile memory

Measuring Memory Capacity

- **Kilobyte:** 2^{10} bytes = 1024 bytes
  - Example: 3 KB = 3 times1024 bytes

- **Megabyte:** 2^{20} bytes = 1,048,576 bytes
  - Example: 3 MB = 3 times 1,048,576 bytes

- **Gigabyte:** 2^{30} bytes = 1,073,741,824 bytes
  - Example: 3 GB = 3 times 1,073,741,824 bytes

Data Storage

- Bits and Their Storage
- Main Memory
- Mass Storage
- Representing Information as Bit Patterns
- The Binary System
- Data Compression
- Communications Errors

Mass Storage

- On-line versus off-line
- Typically larger than main memory
- Typically less volatile than main memory
- Typically slower than main memory

Mass Storage Systems

- Magnetic Systems
  - Disk
  - Tape
- Optical Systems
  - CD
  - DVD
- Flash Technology
  - Flash Drives
  - Secure Digital (SD) Memory Card

Data Storage

- Bits and Their Storage
- Main Memory
- Mass Storage
- Representing Information as Bit Patterns
- The Binary System
- Data Compression
- Communications Errors

Representing Text

- **Each character (letter, punctuation, etc.) is assigned a unique bit pattern.**
  - ASCII: Uses patterns of 7-bits to represent most symbols used in written English text

  (American Standard Code for Information Interchange)

  - ISO developed a number of 8 bit extensions to ASCII, each designed to accommodate a major language group

  (International Organization for Standardization)

  - Unicode: Uses patterns of 16-bits to represent the major symbols used in languages world wide

Appendix A: **Samples from ASCII**

Symbol	ASCII	Hex	Symbol	ASCII	Hex
A	01000001	41	0	00110000	30
B	01000010	42	1	00110001	31
C	01000011	43	2	00110010	32
a	01100001	61	#	00100011	23
b	01100010	62	.	00101110	2E
c	01100011	63	line feed	00001010	0A

The message “Hello.” in ASCII

01001000	01100101	01101100	01101100	01101111	00101110
H	**e**	**l**	**l**	**o**	**.**

Representing Numeric Values

- Binary notation: Uses bits to represent a number in base two
- Limitations of computer representations of numeric values
  - Overflow: occurs when a value is too big to be represented
  - Truncation: occurs when a value cannot be represented accurately

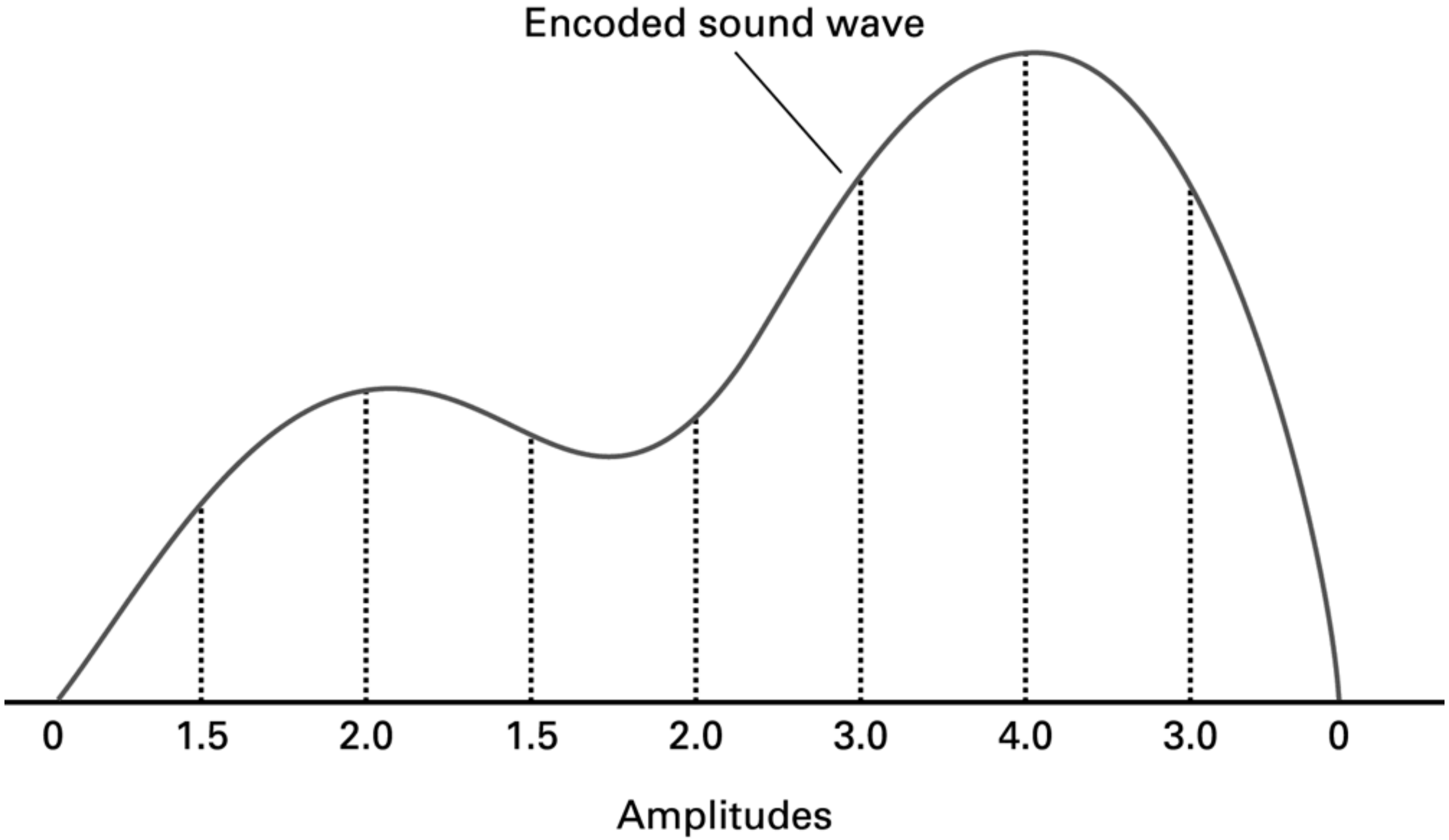
Representing Images

- Bit map techniques
  - Pixel: short for “picture element”
  - RGB
  - Luminance and chrominance
- Vector techniques
  - Scalable
  - TrueType and PostScript

Representing Sound

- Sampling techniques
  - Used for high quality recordings
  - Records actual audio
- MIDI
  - Used in music synthesizers
  - Records “musical score”

The sound wave represented by the sequence 0, 1.5, 2.0, 1.5, 2.0, 3.0, 4.0, 3.0, 0



Data Storage

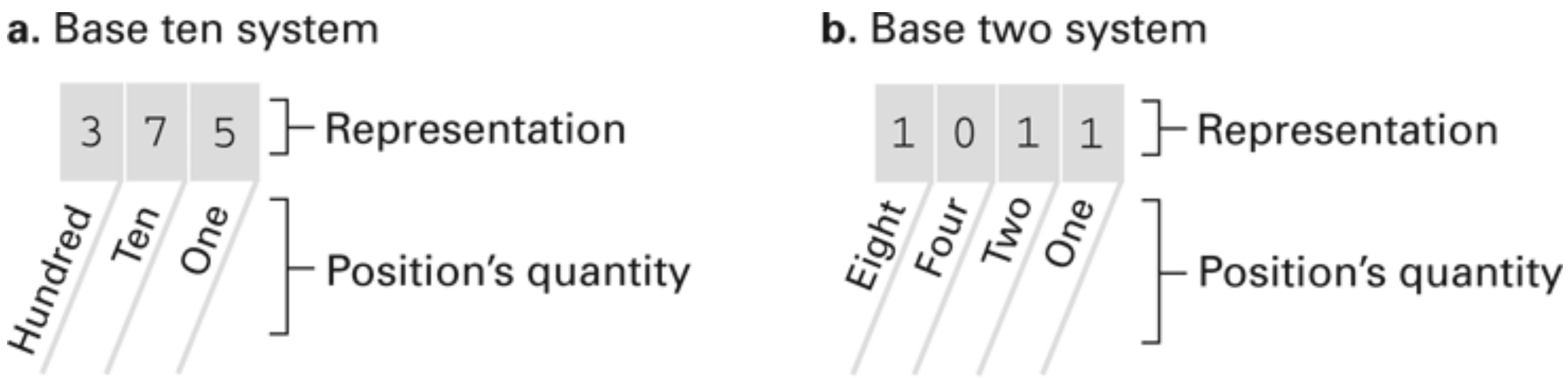
- Bits and Their Storage
- Main Memory
- Mass Storage
- Representing Information as Bit Patterns
- The Binary System
- Data Compression
- Communications Errors

The Binary System

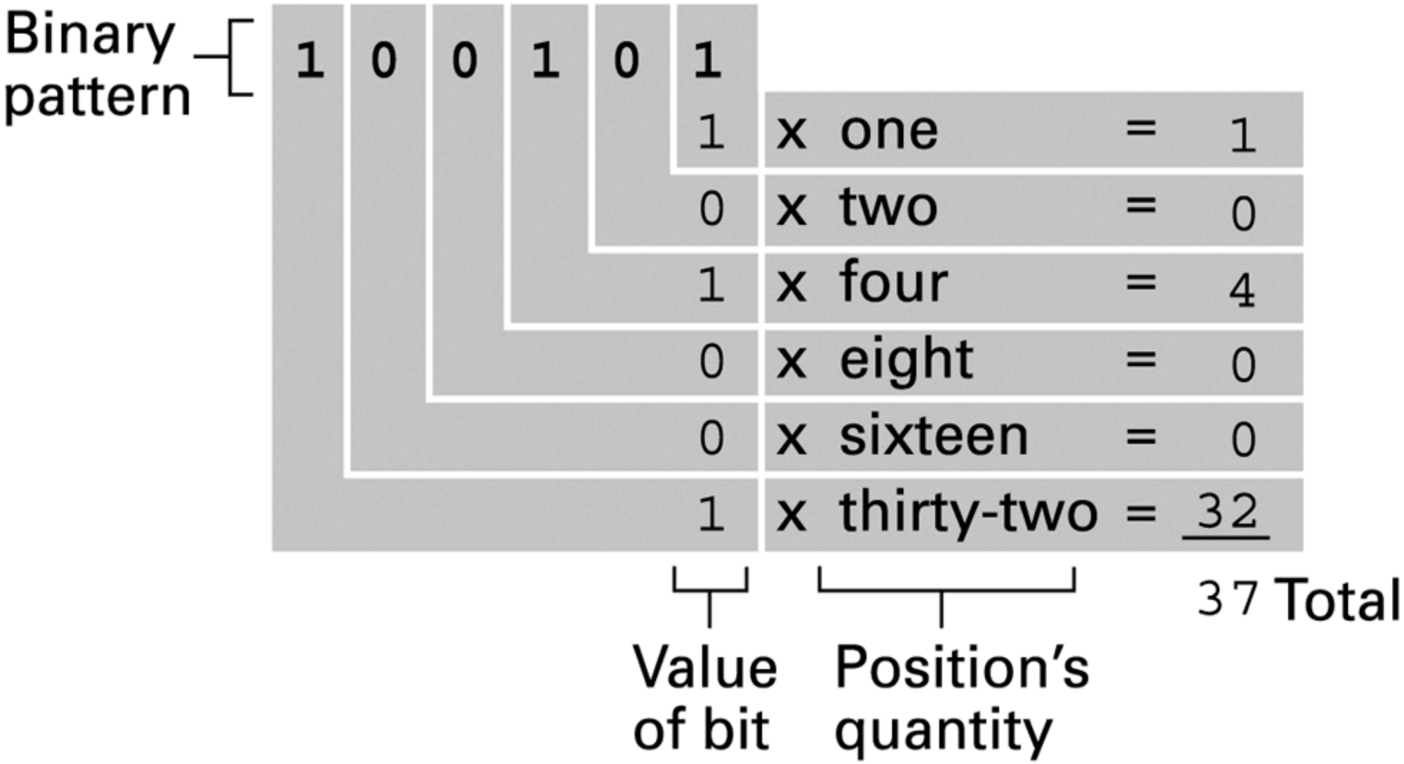
The traditional decimal system is based on powers of ten.

The Binary system is based on powers of two.

The base ten and binary systems



Decoding the binary representation 100101



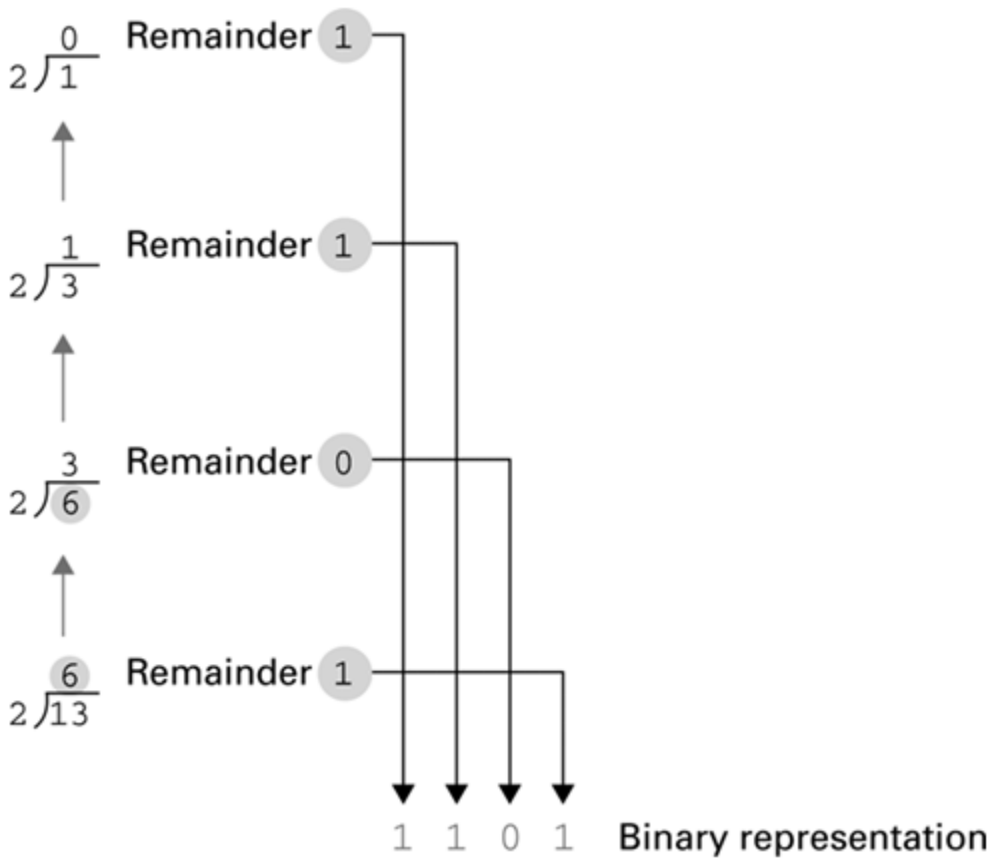
An algorithm for finding the binary representation of a positive integer

Step 1. Divide the value by two and record the remainder.

Step 2. As long as the quotient obtained is not zero, continue to divide the newest quotient by two and record the remainder.

Step 3. Now that a quotient of zero has been obtained, the binary representation of the original value consists of the remainders listed from right to left in the order they were recorded.

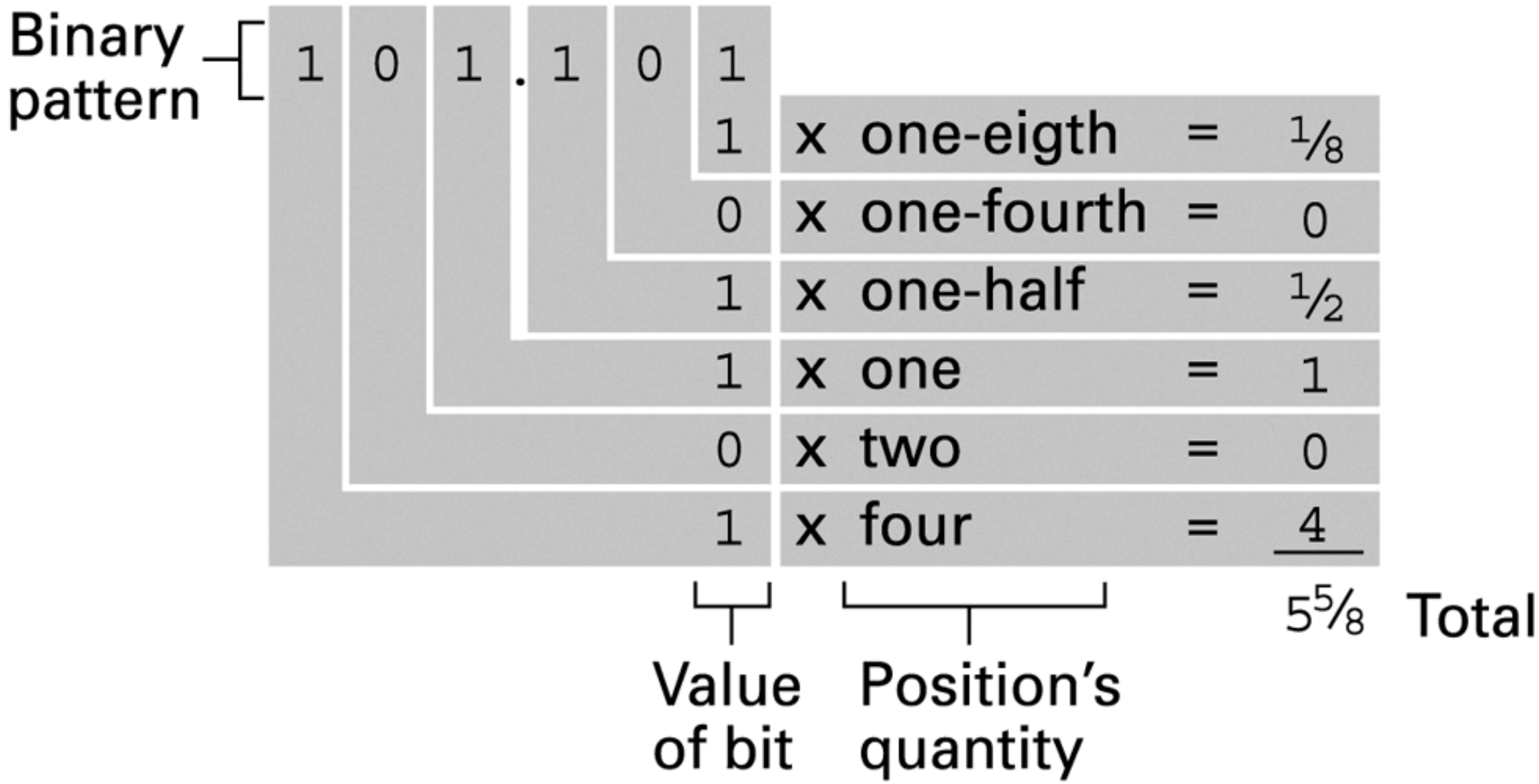
Applying the algorithm to obtain the binary representation of thirteen



The binary addition facts

$$\begin{array}{r} 0 \\ +\,0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 1 \\ +\,0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 0 \\ +\,1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +\,1 \\ \hline 10 \end{array}$$

Decoding the binary representation 101.101



Storing Integers

- **Two's complement notation:** The most popular means of representing integer values
- **Excess notation:** Another means of representing integer values
- Both can suffer from overflow errors.

Two's complement notation systems

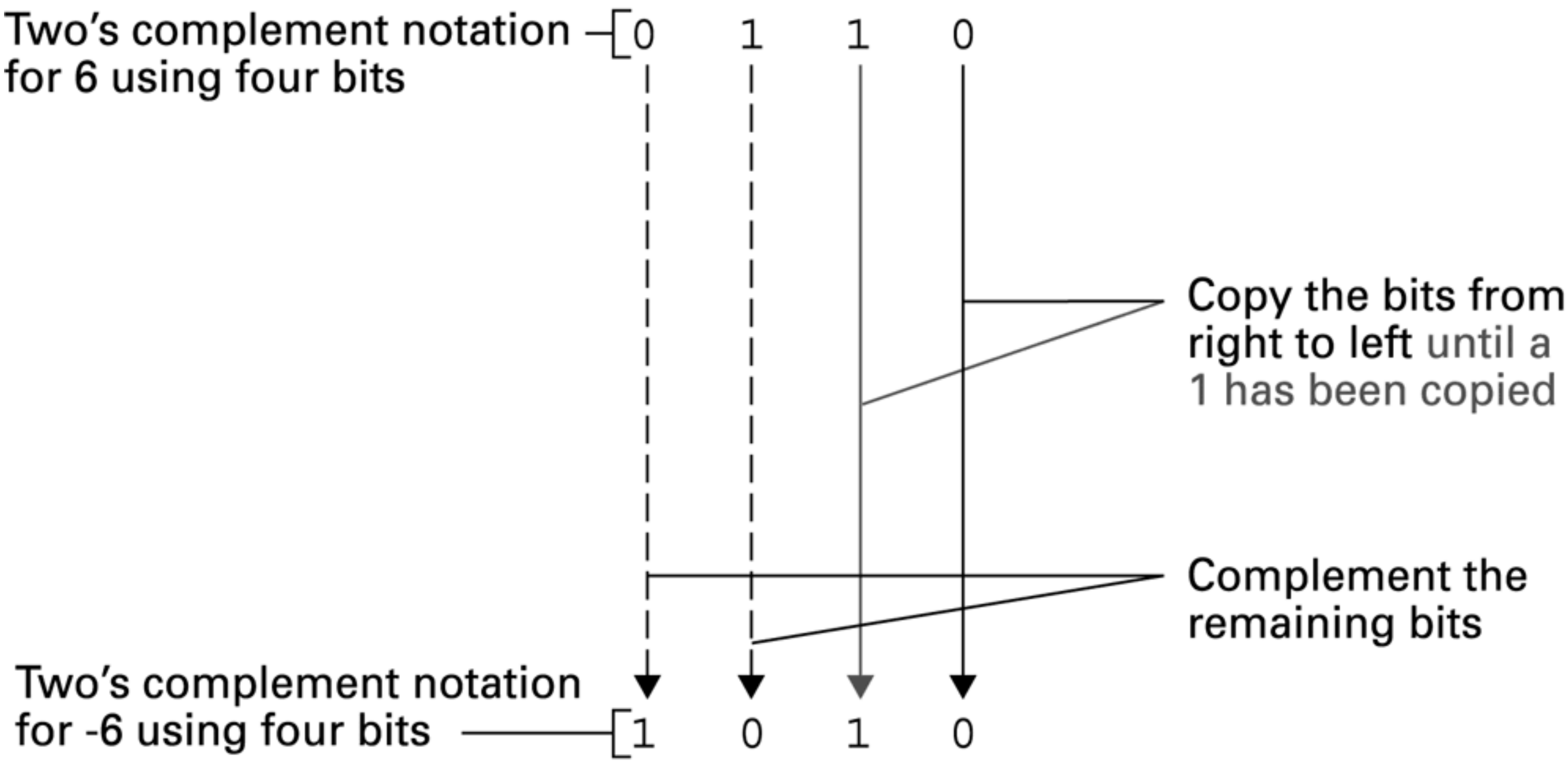
a. Using patterns of length three

Bit pattern	Value represented
011	3
010	2
001	1
000	0
111	−1
110	−2
101	−3
100	−4

b. Using patterns of length four

Bit pattern	Value represented
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	−1
1110	−2
1101	−3
1100	−4
1011	−5
1010	−6
1001	−7
1000	−8

Coding the value -6 in two's complement notation using four bits



Addition problems converted to two's complement notation

Problem in base ten		Problem in two's complement		Answer in base ten
3 + 2	→	0011 + 0010 = 0101	→	5
−3 + −2	→	1101 + 1110 = 1011	→	−5
7 + −5	→	0111 + 1011 = 0010	→	2

Data Storage

- Bits and Their Storage
- Main Memory
- Mass Storage
- Representing Information as Bit Patterns
- The Binary System
- Data Compression
- Communications Errors

Data Compression

- Lossy versus lossless
- Run-length encoding [250 ones, followed by 200 zeros, followed by 80 ones]
- Frequency-dependent encoding (e.g.Huffman codes) [More frequent use of item less length for the code]
- Relative encoding [Record only the differences]
- Dictionary encoding (Includes adaptive dictionary encoding such as LZW encoding. [Lempel-Ziv-Welsh])

Lempel-Ziv-Welsh (example)

Encoding message: xyx xyx xyx xyx

1:x

2:y

3:"Space"

4:xyx (Do not need to be sent to decoder)

Encoded Message: 121343434

Decoding: 121343434 by (1:x,2:y,3:"Space")

xyx (add 4 to dictionary)xyx xyx xyx

Compressing Images

- GIF [Graphic Interchange Format]: Good for cartoons
- JPEG [Joint Photographic Experts Group]: Good for photographs
- TIFF [Tagged Image File Format]: Good for image archiving

Compressing Audio and Video

- MPEG [Motion Picture Experts Group]
  - High definition television broadcast
  - Video conferencing
- MP3 [MPEG Layer 3]
  - Temporal masking
  - Frequency masking

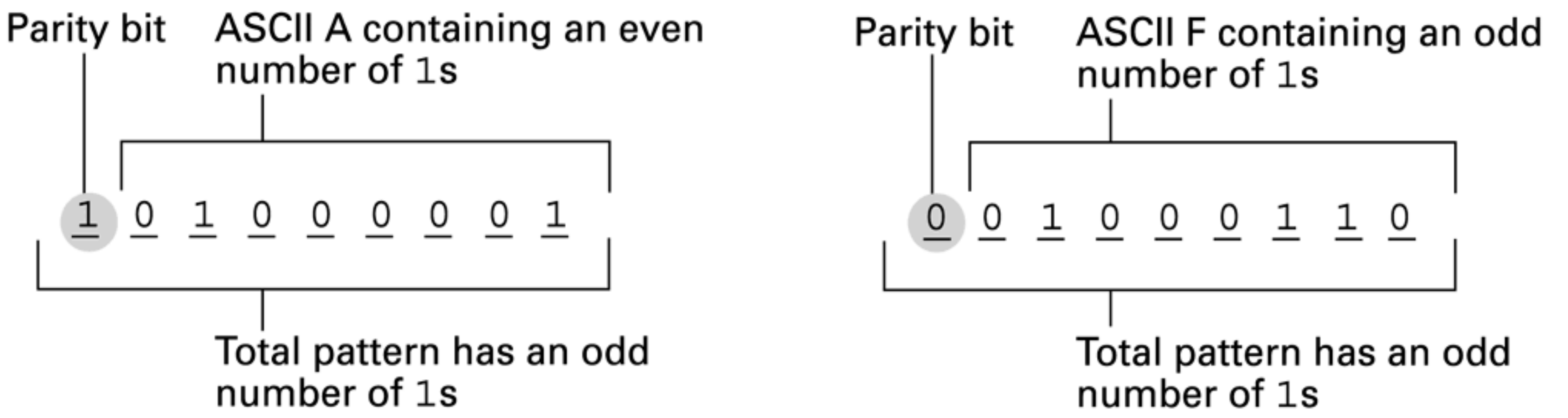
Data Storage

- Bits and Their Storage
- Main Memory
- Mass Storage
- Representing Information as Bit Patterns
- The Binary System
- Data Compression
- Communications Errors

Communication Errors

- Parity bits (even versus odd)
- Checkbytes (To detect burst of bit errors)
- Error correcting codes

The ASCII codes for the letters A and F adjusted for odd parity



An error-correcting code

Symbol	Code
A	000000
B	001111
C	010011
D	011100
E	100110
F	101001
G	110101
H	111010

Decoding the pattern 010100 using the code in previous slide (Hamming Distance)

Character	Code	Pattern received	Distance between received pattern and code	
A	0 0 0 0 0 0	0 **1** 0 **1** 0 0	2	
B	0 0 1 1 1 1	0 **1** **0** 1 **0** **0**	4	
C	0 1 0 0 1 1	0 1 0 **1** **0** **0**	3	
D	0 1 1 1 0 0	0 1 **0** 1 0 0	**1**	Smallest distance
E	1 0 0 1 1 0	**0** **1** 0 1 **0** 0	3	
F	1 0 1 0 0 1	**0** **1** **0** **1** 0 **0**	5	
G	1 1 0 1 0 1	**0** 1 0 1 0 **0**	2	
H	1 1 1 0 1 0	**0** 1 **0** **1** **0** 0	4	