Number System

Download PDF

To access the updated handouts, please click on the following link:
https://yasirbhutta.github.io/computer-basics/docs/number-systems.html

Module 1: Bits

A bit is the smallest unit of information in a computer. It can be either 0 or 1. Bits are used to represent all of the data that is stored and processed on a computer, including text, images, audio, and video.

- What is a bit?
- How are bits represented?
- How are bits used?
- Patterns using Bits

Module 12: Boolean Operations

Boolean operations are used to manipulate bits. The two most common Boolean operations are AND and OR. The AND operation returns 1 if both bits are 1, and 0 otherwise. The OR operation returns 1 if either bit is 1, and 0 otherwise.

Module 13: Hexadecimal Notation

Hexadecimal notation is a way of representing numbers using 16 symbols instead of 10. The hexadecimal symbols are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F.

Module 14: Storing a Bit

Bits are stored in memory using electrical charges. A bit is stored as a high voltage if it is a 1, and a low voltage if it is a 0.

Module 1: bits

What is a bit?

- A bit is the smallest unit of information in a computer.
- It can be either 0 or 1.
- Bits are used to represent all of the data that is stored and processed on a computer, including text, images, audio, and video.

How are bits represented?

- Bits can be represented in a variety of ways, including electronically, optically, and magnetically.
- In most computers, bits are represented electronically using transistors.
- A transistor is a semiconductor device that can be switched to either a conducting or non-conducting state. When a transistor is conducting, it represents a 1 bit. When a transistor is non-conducting, it represents a 0 bit.

How are bits used?

Bits are used in a variety of ways in computers. For example, bits are used to:

Represent text characters: Each character in the English alphabet is represented by a unique sequence of bits. **Represent images:** Images are represented by a grid of pixels. Each pixel is represented by a combination of red, green, and blue bits. **Represent audio:** Audio is represented by a series of samples. Each sample is represented by a combination of bits. **Represent video:** Video is represented by a series of images. Each image is represented by a grid of pixels. Each pixel is represented by a combination of red, green, and blue bits. Examples of bits in use

Here are some examples of how bits are used in the real world:

- When you type a letter on your keyboard, the computer converts the letter to a sequence of bits. These bits are then sent to the monitor, which displays the letter on the screen.
- When you take a photo with your digital camera, the camera converts the image to a grid of pixels. Each pixel is represented by a combination of red, green, and blue bits. These bits are then stored on the camera's memory card.
- When you listen to a song on your MP3 player, the player converts the song to a series of samples. Each sample is represented by a combination of bits. These bits are then played through the player's speakers.
- When you watch a video on your computer, the computer converts the video to a series of images. Each image is represented by a grid of pixels. Each pixel is represented by a combination of red, green, and blue bits. These bits are then displayed on the computer's monitor.

Units of measurement for storage data

Sr. No.	Units	Abbr.	Description	Approximate Size
1	Bit		Computer works with binary digits in 0’s and 1’s form. A binary digit is called bit.	
2	Nibble		4 bit	
3	Byte		8 bit (Store single character)	1 character
4	Kilobyte	KB	1024 Bytes	1 page
5	Megabyte	MB	1024 KB (1 millon bytes)	1000 pages
6	Gigabyte	GB	1024 MB(1 billion bytes)	1 million pages
7	Terabyte	TB	1024 GB(1 trillion bytes)	1 billion pages

Patterns Using Bits

- A bit pattern is a sequence of bits, which are the smallest unit of data in computing. A bit can have two values: 0 or 1. Bit patterns are used to represent all kinds of data, including numbers, letters, symbols,

and images.

- Bit patterns can be of any length, but they are typically represented as strings of binary digits. For example, the bit pattern "01000001" represents the letter "A" in ASCII code. [^1]
- Bit patterns are used in a wide variety of computing applications. For example, they are used to store and transmit data, to encode and decode information, and to perform mathematical and logical operations.

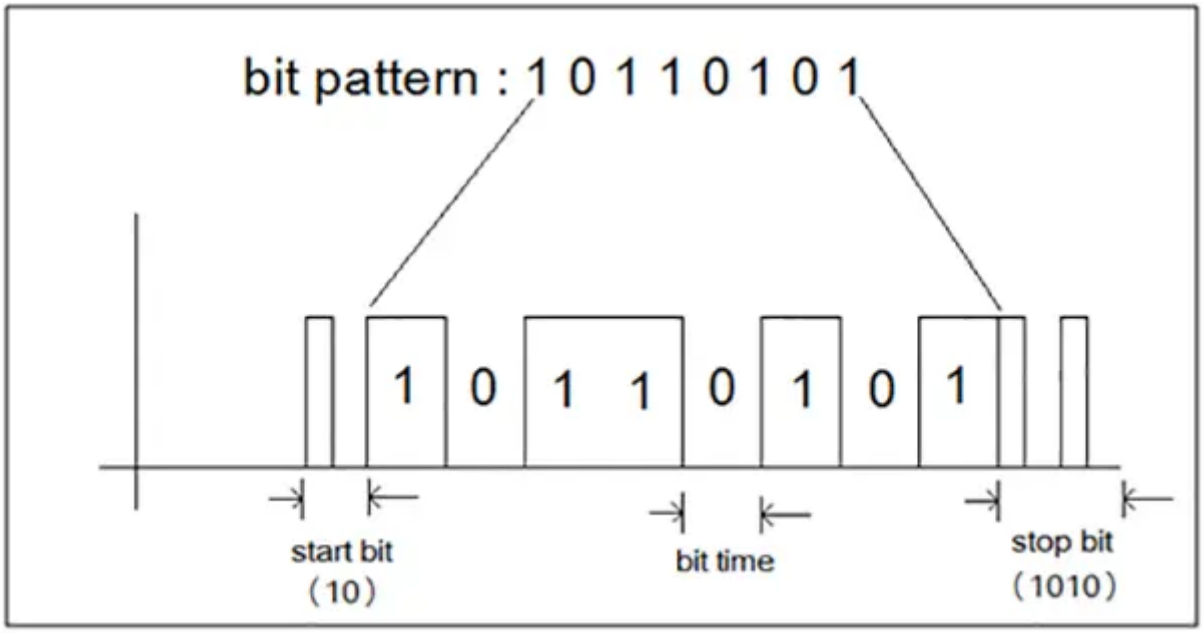

Image source: engineersgarage.com

Here are some examples of bit patterns:

- The bit pattern "00110001" represents the number 1 in ASCII code.
- The bit pattern "01001000 01100101 01101100 01101100 01101111" represents the word "Hello" in ASCII code.

Bit patterns are a powerful tool for representing and manipulating information in computing.

ASCII code

- ASCII stands for American Standard Code for Information Interchange. It is a character encoding system that uses a 7-bit or 8-bit binary code to represent characters.
- ASCII is used in most computers and electronic devices to represent text, numbers, and other symbols.

Each ASCII character is assigned a unique code. For example, the ASCII code for the letter "A" is 65. The ASCII code for the space character is 32.

For example, to convert the letter "A" to binary code, you would follow these steps:

- The ASCII code for the letter "A" is 65.
- Write the ASCII code in decimal form: 65
- Convert the decimal code to binary code: 01000001
- Pad the binary code with zeros on the left to make it 8 bits long: 01000001

The following table shows some common ASCII characters and their binary codes:

Character	ASCII Code	Binary Code
A	65	1000001
B	66	1000010
C	67	1000011
...	...	...
Z	90	1011010
0	48	110000
1	49	110001
2	50	110010
...	...	...
9	57	111001
space	32	100000
!	33	100001
"	34	100010
#	35	100011
...	...	...
~	126	1111110

click on the following link to view complete table: ASCII, decimal, hexadecimal, octal, and binary conversion table - IBM

Convert decimal code to binary code:

For example, to convert the decimal number 10 to binary code, we would do the following:

```
10 / 2 = 5 Remainder: 0 (bit #1)
5 / 2 = 2 Remainder: 1 (bit #2)
2 / 2 = 1 Remainder: 0 (bit #3)
1 / 2 = 0 Remainder: 1 (bit #4)
```

So, 10 (decimal) = 1010 (binary)

See also:

- ascii to binary converter
- How to convert decimal to binary - rapidtables.com
- Binary to decimal - rapidtables.com

Module 2: Boolean Operations

What are Boolean operations?**

- Boolean operations are logical operations that are used to manipulate bits.
- The two most common Boolean operations are AND and OR.
- The AND operation returns 1 if both bits are 1, and 0 otherwise.
- The OR operation returns 1 if either bit is 1, and 0 otherwise.

Boolean expressions

Boolean expressions are expressions that use Boolean operations to combine bits. Boolean expressions can be used to perform simple calculations on bits, such as checking if two bits are both equal to 1, or checking if a bit is equal to 0.

Truth tables

Truth tables are a way of representing the output of a Boolean expression for all possible combinations of inputs. Truth tables can be used to verify that a Boolean expression works as expected.

Example: Let's create a truth table for the expression A AND B:

A | B | A AND B 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1 In this example:

- When A = 0 and B = 0, A AND B evaluates to 0.
- When A = 1 and B = 1, A AND B evaluates to 1.

Example: Let's create a truth table for the expression A OR B:

A | B | A OR B 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1

In this example:

- When A = 0 and B = 0, A AND B evaluates to 0.
- When A = 0 and B = 1, A AND B evaluates to 1.

Examples of Boolean expressions

Here are some examples of Boolean expressions:

- `A AND B`: This expression returns 1 if both bits A and B are equal to 1, and 0 otherwise.
- `A OR B`: This expression returns 1 if either bit A or bit B is equal to 1, and 0 otherwise.
- `NOT A`: This expression returns the opposite of bit A.
- `(A AND B) OR C`: This expression returns 1 if either A AND B is equal to 1, or C is equal to 1, and 0 otherwise.

Applications of Boolean operations

Boolean operations are used in a variety of applications, including:

- Digital logic: Boolean operations are used to design and implement digital logic circuits.

- Computer programming: Boolean operations are used to write computer programs.
- Databases: Boolean operations are used to perform complex queries on databases.
- Search engines: Boolean operations are used to refine search results.

Module 3: Hexadecimal Notation

What is hexadecimal notation?

Hexadecimal notation is a way of representing numbers using 16 symbols instead of 10. The hexadecimal symbols are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F.

Why use hexadecimal notation?

Hexadecimal notation is a more compact way of representing binary numbers. For example, the binary number 1010101010101010 can be represented as the hexadecimal number 0xAAAA.

How to convert between hexadecimal and binary

To convert a hexadecimal number to binary, simply replace each hexadecimal digit with its corresponding binary equivalent. For example, to convert the hexadecimal number 0xAAAA to binary, we would replace the A digits with the binary number 1010, and we would replace the F digits with the binary number 1111. This would give us the binary number 1010101010101010.

To convert a binary number to hexadecimal, simply group the binary digits into groups of four. Then, replace each group of four binary digits with its corresponding hexadecimal digit. For example, to convert the binary number 1010101010101010 to hexadecimal, we would group the binary digits into groups of four:

```
1010 1010 1010 1010
```

Then, we would replace each group of four binary digits with its corresponding hexadecimal digit:

```
AAAA
```

This would give us the hexadecimal number 0xAAAA.

Another example, to convert hexadecimal number `1E` to binary:

1. Write down the hex number: We have 1E.
2. Convert hex to decimal: The decimal equivalent of 1E is 30 (since 1E represents 16 + 14 = 30). Represent each digit in binary:
3. 1 in binary is 0001.
4. E in decimal is 14, which in binary is 1110.
5. Combine the binary representations: 1E in binary is 00011110

Applications of hexadecimal notation

Hexadecimal notation is used in a variety of applications, including:

- **Computer programming:** Hexadecimal notation is used to represent machine code and memory addresses.
- **Electronics:** Hexadecimal notation is used to represent hexadecimal numbers in electronic circuits.
- **Networking:** Hexadecimal notation is used to represent MAC addresses and IP addresses.
- **Debugging:** Hexadecimal notation is used to debug computer programs and electronic circuits.

Module 14: Storing a Bit

How are bits stored in memory?

Bits are stored in memory using electrical charges. A bit is stored as a high voltage if it is a 1, and a low voltage if it is a 0.

Types of memory

There are two main types of memory in a computer: RAM and ROM. RAM stands for random-access memory. RAM is used to store data that is currently being used by the computer. ROM stands for read-only memory. ROM is used to store data that is permanent, such as the computer's BIOS.

RAM

RAM is made up of millions or even billions of tiny transistors. Each transistor can store a single bit of data. The transistors are arranged in a grid, and each transistor is addressed by a unique address.

When the computer needs to access a bit of data in RAM, it sends the address of the bit to the memory controller. The memory controller then sends a signal to the transistor at that address. The transistor then switches to the appropriate state (high voltage or low voltage) to represent the bit of data.

ROM

ROM is made up of a series of interconnected cells. Each cell can store a single bit of data. The cells are arranged in a grid, and each cell is addressed by a unique address.

When the computer needs to access a bit of data in ROM, it sends the address of the bit to the memory controller. The memory controller then sends a signal to the cell at that address. The cell then reads the bit of data and sends it back to the memory controller.

Review Questions

Excercises

1. Research the ASCII code table and find the bit patterns for the letters "G", "o", "o", "d".
2. Decode the following binary messages: "01000111 01110010 01100001 01110100 01101001 01110100 01110101 01100100 01100101".
3. Write down the binary representation of the numbers 0 to 15.
4. Convert the following decimal numbers to binary: 25, 47, 100, and 128.
5. Write down the ASCII code for the letters "A" to "Z".
6. Write down the ASCII code for the numbers 0 to 9.

7. Write down the ASCII code for the following symbols: !, @, #, and $.

References

[^1] ASCII, decimal, hexadecimal, octal, and binary conversion table - IBM