Number Representation 1 – Decimal and Binary

The Binary System

The **Binary System** is a way of writing numbers using only the digits 0 and 1. This is the method used by the (**digital**) computer.

The system we use to write numbers is called **decimal** (or **denary**).

In the DECIMAL system fifty-two is written as 52.
In the BINARY system fifty-two is written as 110100.

Converting from Binary to Decimal

How can we convert numbers from the binary system to the decimal system?

Follow this example. We have 10110011. Note that this number is a sequence of 8 **bits**, so it is a **byte**. Add the weights as seen in the diagram. Add all **weights** that are associated with the bits that are equal to 1.

Weights	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**
Number	**1**	**0**	**0**	**1**	**0**	**1**	**1**	**0**

Add 128 + 16 + 4 + 2. This amounts to 150.

Sometimes we use **subscripts** to indicate whether a number is binary of decimal as shown here:

A binary number: 10010110_2
A decimal number: 150_{10}

The way we write numbers is called the **positional notation** method. This means that if for example you have the number 2343 the value represented by each digit depends on the position of the digit within the number so the leftmost two is representing 2000, the next digit, 3, represents 300, the 4 represents 40 and the rightmost digit, 3, represents just 3.

Exercise Binary to Decimal

Convert the following binary numbers to decimal.

a. 11001
b. 11011101
c. 1110010

Converting from Decimal to Binary

Suppose we want to convert the decimal number 149 to binary. One way we can do this is by repeatedly dividing the number by 2 and then taking the remainder digits to get the solution. An example is shown here:

2	149		
2	74	remainder	1
2	37	remainder	0
2	18	remainder	1
2	9	remainder	0
2	4	remainder	1
2	2	remainder	0
2	1	remainder	0
	0 Quotient after dividing by 2	remainder	1 Remainder after dividing by 2

After performing this process, you have to read the remainders FROM BOTTOM TO TOP. This gives us that the decimal number 149 is equal to the binary number 10010101.

Exercise Decimal to Binary

Convert the following decimal numbers to binary.

a. 108
b. 53
c. 74

Number Bases

Positional number systems depend on a special number called a **base**.

Note that:

$2725 = 2x1000 + 7x100 + 2x10 + 5.$

$$2725 = 2x\mathbf{10^3} + 7x\mathbf{10^2} + 2x\mathbf{10^1} + 5x\mathbf{10^0}.$$

The base number of the decimal system is 10.

Working on the same principles we can show that the base number of the binary system is 2.

Let us investigate the number 10010110_2.

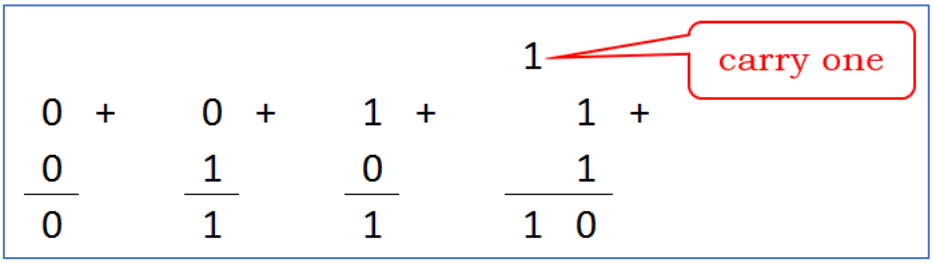
$10010110 = 1x128 + 0x64 + 0x32 + 1x16 + 0x8 + 1x4 + 1x2 + 0x1.$

$$10010110 = 1x\mathbf{2^7} + 0x\mathbf{2^6} + 0x\mathbf{2^5} + 1x\mathbf{2^4} + 0x\mathbf{2^3} + 1x\mathbf{2^2} + 1x\mathbf{2^1} + 0x\mathbf{2^0}.$$

Note also that the base number shows us how many different symbols are required to express a number i.e. in the decimal system we need 10 symbols (0, 1, 2, 3, 4, 5, 6, 7, 8 and 9) while in the binary system we need 2 (0 and 1).

Adding Binary Numbers

When adding, every 2 is **carried** to the next (left) column as a 1. So:



The following example shows two binary numbers being added.

Carry							
First number	0	0	1	1	1	1	0
Second number	1	0	0	1	0	1	1
Result of addition							1

Step 1:
0 + 1 gives 1

Carry					1		
First number	0	0	1	1	1	1	0
Second number	1	0	0	1	0	1	1
Result of addition						0	1

Step 2:
1 + 1 gives 0 carry 1

Carry				1	1		
First number	0	0	1	1	1	1	0
Second number	1	0	0	1	0	1	1
Result of addition					0	0	1

Step 3:
1 + 0 + 1 gives 0
carry 1

Carry			1	1	1		
First number	0	0	1	1	1	1	0
Second number	1	0	0	1	0	1	1
Result of addition				1	0	0	1

Step 4:
1 + 1 + 1 gives 1
carry 1

Proceeding in this way we get:

Carry		1	1	1	1		
First number	0	0	1	1	1	1	0
Second number	1	0	0	1	0	1	1
Result of addition	1	1	0	1	0	0	1

30

75

105

Exercise in Addition

Add the following binary numbers. Then check your result by using decimal numbers.

a. 1010001 and 0101110
b. 1011011 and 1011100

Subtracting Binary Numbers

For subtraction we use the idea of **borrowing**. This is required when the number to be subtracted is bigger than the number to be subtracted from. When we borrow one from the left column, this is valued as 2 when borrowed. The following diagram shows a subtraction of two binary numbers.

Borrow							
First number	1	0	1	1	0	1	1
Second number	0	1	0	1	1	1	0
Result of subtraction							1

Step 1:
1 – 0 gives 1

Borrow							
First number	1	0	1	1	0	1	1
Second number	0	1	0	1	1	1	0
Result of subtraction						0	1

Step 2:
1 – 1 gives 0

Borrow					2		
First number	1	0	1	1	0	1	1
Second number	0	1	0	1	1	1	0
Result of subtraction					1	0	1

Step 3:
0 - 1 means that we have to borrow

Proceeding in this way we get

Borrow		2		2	2		
First number	1	0	1	1	0	1	1
Second number	0	1	0	1	1	1	0
Result of subtraction	0	1	0	1	1	0	1

Exercise in Subtraction

Perform the following binary subtractions.

(a) 10001101 – 00101100
(b) 10000101 – 01010111

Registers

A **register** is a place to hold binary numbers. Registers normally are 16, 32, or 64-bits wide. Note that the number of bits is always a power of two.

Shifts

If we add a zero at the right end of a decimal number this would multiply the number by 10. If we do the same operation on a binary number the number would be multiplied by two. For example, 780_{10} is ten times as much as 78_{10} and 110_2 is twice as much as 11_2.

Suppose you have an 8-bit register holding the number 1010 and you shift the number to the left by one bit, then by another bit etc. After each shift the number is doubled. This is shown in the diagram below.

	128	64	32	16	8	4	2	1
1	0	0	0	0	1	0	1	0
2	0	0	0	1	0	1	0	0
3	0	0	1	0	1	0	0	0
4	0	1	0	1	0	0	0	0
5	1	0	1	0	0	0	0	0

In line 1 (no shift) the number is equal to ten (note the weights in red). Line 2 shows a left shift of 1 bit. The number is 20. After the fourth shift the number is 160. Another shift won't work because the number will be two high to be held in the register. This is called **overflow**.

Exercise on Shifts

(a) The number 47_{10} is stored in an 8-bit register. What will its value become if the number is shifted two bits to the left?

(b) The number 152_{10} is stored in an 8-bit register. What will its value become if the number is shifted three bits to the right?