25.4. THE BINARY NUMBER SYSTEM

The binary number system has only two symbols, namely 0 and 1. The base of the binary number system is therefore 2.

A binary digit, be it 0 or 1, is called a ***bit*** which is a contraction of the two words of binary digit. A binary number consists of a sequence of bits. The weight of each bit position is one power of 2 greater than the weight of the position to its right.

Binary number	1 0 1 1
Weights	$2^3\ 2^2 2^1\ 2^0$

Counting in the binary system is done in the same way as in the decimal number system. With decimal numbers, the counting is done first from 0 to 9. After that a single digit cannot represent a higher number. Then a second digit is added to their left and counting continues as 10, 11, 12,.... 20, 21, 22, ... and so on. Binary counting is done in a similar way. To represent 2 in binary system , we write 1 followed by 0. This gives 10 (read as one-zero, but not ten). Thus 10 is the binary equivalent of 2 of the decimal system. Similarly 3 can be represented as 11(read as one-one). After representing 3 , the binary digits are exhausted. For representing 4 in the binary system we have to use the second digit followed by two first digits, i.e 100(one-zero-zero). Table 1 shows binary count sequence from 0 to 15. It is a common practice to include zeros to the left of a bit for purposes of symmetry and convenience. 4-bit binary numbers are called ***nibbles.***

Table 1. Binary Count Sequence.

Decimal Number	*Binary Number*
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1

Decimal Number	Binary Number
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

25.5. BINARY ARITHMETIC

The binary arithmetic operations are performed in the same manner as in decimal system.

25.5.1. Binary Addition

Let us first look at the addition of two single-bit binary numbers. Let X and Y represent single-bit numbers.

X	Y	X + Y
0	0	$0+0=0$
1	0	$1+0=1$
0	1	$0+1=1$

We find here that adding 0 to a number does not change the number.

X	Y	X + Y
1	1	$1+1=10$, meaning $1+1=0$, carry 1

In this case, the sum is 0 with a *carry* of 1. This carry is similar to a decimal carry when the decimal sum exceeds the number 1.

When adding two binary numbers having an arbitrary number of bits, we add the bits in each column, one column at a time starting from the right most column. A carry bit generated in a given bit position will be added to the bits in the column to the left of it.

Example 25.1:

```
Carry bits :  1 1 1
            X=0 0 1 1
            Y=0 1 0 1
        ---------------
          X+Y=1 0 0 0
```

Binary numbers containing fractional parts are added in a similar way. A ***binary point*** separates the integer and fractional parts of a binary number.

Example 25.2:

Add $X=1010$ and $Y=0011.11$

We adjust the length of X by inserting 0s as necessary to make both X and Y have the same number of bits.

```
Carry bits :      1
            X=1 0 1 0 . 0 0
            Y=0 0 1 1 . 1 1
        ---------------------
          X+Y=1 1 0 1 . 1 1
```

Carries produced on the fractional side are added to the bits in the column to the left of it.

Example 25.3: Add 1011.01 to 1101.11

$$\begin{array}{rr} \text{Carry bits:} & 11111.1 \\ X = & 1011.010 \\ Y = & 1101.110 \\ \hline X+Y = & 10001.000 \\ \hline \end{array}$$

25.5.2. Binary Subtraction

If we subtract 0 from a number, its value does not change. Thus,

$$\begin{array}{ccc} X & Y & X-Y \\ 0 & 0 & 0-0=0 \\ 1 & 0 & 1-0=1 \end{array}$$

If we subtract 1 from a number, its value reduces by 1.

$$\begin{array}{ccc} X & Y & X-Y \\ 1 & 1 & 1-1=0 \end{array}$$

When 1 is to be subtracted from 0, we must *borrow* a 1 from the next most significant bit. The procedure is the same as that practised in decimal subtraction.

Example 25.4:

Borrowed 1

$$\begin{array}{rcccc} & & 0 & 10 & \\ X = & 1 & 0 & 1 & 1 \\ Y = & 0 & 1 & 0 & 1 \\ \hline X-Y = & 0 & 1 & 1 & 0 \\ \hline \end{array}$$

Note that 1 + 1 = 0, carry 1, and 0 − 1 = 1, borrowed 1

Example 25.5:

$$\begin{array}{rcccc} & 0 & 1 & 1 & 10 \\ X = & 1 & 0 & 0 & 0 \\ Y = & 0 & 0 & 1 & 1 \\ \hline X-Y = & 0 & 1 & 0 & 1 \\ \hline \end{array}$$

Bits in the fractional parts of binary numbers are subtracted in the same way as bits in the integer parts.

Example 25.6:

second borrow

first borrow

$$\begin{array}{rccccccc} & 0 & 1 & 1 & 10 & & & \\ & & & & 0 & & 10 & \\ X = & 1 & 0 & 0 & 1 & . & 0 & 1 \\ Y = & 0 & 0 & 1 & 1 & . & 1 & 0 \\ \hline X-Y = & 0 & 1 & 0 & 1 & . & 1 & 1 \\ \hline \end{array}$$

25.5.3. Binary Multiplication

Binary multiplication is performed in the same manner as with decimal numbers. The rules of multiplication are simple.

$0 \times 0 = 0$

$0 \times 1 = 0$

$1 \times 0 = 0$

$1 \times 1 = 1$

Multiplication of multibit binary numbers involves forming the partial products, shifting each successive partial product left one place, then adding all the partial products.

Example 25.7:

```
      X =     10101     multiplicand
    × Y =       101     multiplier
   ----------------
              10101  ⎫
             00000   ⎬  partial products
            10101    ⎭
            -------
  X × Y = 1101001       product
```

When multiplying binary numbers containing fractional parts, we set the binary point in the product using the same procedure followed for decimal multiplication.

Example 25.8:

```
  X =        10101.01
  ×            110.10
  --------------------
              0000000
             1010101
            0000000
           1010101
          1010101
  --------------------
        10001010.0010
```

Note that the partial products are written without binary points.

25.5.4. Binary Division

Binary division is performed in the same manner as with decimal numbers. However, binary division is much simpler than division with decimal numbers.

Example 25.9:

```
Dividend
Divisor   101) 1101001( 10101  Quotient
               101
               ---
                110
                101
                ---
                  101
                  101
                  ---
                    0
```

Example 25.10 :

```
100)11001(110.01
    100
    ---
     100
     100
     ---
      001
      000
      ---
       010
       000
       ---
        100
        100
        ---
          0
```

25.6. OCTAL NUMBERS

Digital circuits operate in binary but it is a cumbersome task to convert from decimal to binary. The octal number system is close to decimal, yet easy to convert to binary.

The octal number system has a base of 8, so it uses eight digits 0,1,2,3,4,5,6,7. The digits 8 and 9 do not belong to the octal number system. Because the base is a power of 2, $8 = 2^3$, it is easy to convert from octal to binary. Also, there are a fewer digits in any given octal number than in the corresponding binary number. So it is much easier to work with octal numbers.

25.6.1. Octal Arithmetic

The sum of two octal digits is the same as their decimal sum, provided the decimal sum is less han 8. If the sum is 8 or greater, we subtract 8 from it to obtain the octal digit. A carry 1 is generated vhen we correct the sum in the said manner.

Example 25.11:

$$\begin{array}{r} 4 \\ +2 \\ \hline 6 \\ \hline \end{array} \qquad \begin{array}{r} 6 \\ +7 \\ \hline 5\text{, carry 1} \\ \hline \end{array} \qquad \begin{array}{r} 5 \\ +3 \\ \hline 0\text{, carry 1} \\ \hline \end{array} \qquad \begin{array}{r} \text{carry: } 1\,1\; \\ X = 2\,5\,6 \\ Y = 4\,3\,7 \\ \hline X+Y = 7\,1\,5 \\ \hline \end{array}$$

Octal subtraction is done more easily using complements method, which we will learn in a latter ection. Octal multiplication is performed by converting decimal partial products to octal.

Example 25.12:

$$\begin{array}{rl} X = 15 & \\ \times Y = 24 & \\ \hline 64 & (20-16=4\text{, carry }2) \\ 32\; & (10-8=2\text{, carry }1) \\ \hline 404 & (6+2=8-8=0\text{, carry }1) \\ \hline \end{array}$$

5.7. HEXADECIMAL NUMBERS

The hexadecimal number system has a base of 16 and contains 16 symbols. The ten decimal mbols from 0 to 9 supplemented with 6 new symbols from A to F constitute the hexadecimal umber system.

Table 2.

Hexadecimal Digit	*Decimal Value*	*Hexadecimal Digit*	*Decimal Value*
0	0	9	9
1	1	A	10
2	2	B	11
3	3	C	12
4	4	D	13
5	5	E	14
6	6	F	15
7	7		
8	8		

Although hexadecimal number system is slightly difficult to interpret, it is easy to convert xadecimal numbers to binary and vice versa.

25.7.1. Hexadecimal Arithmetic

The sum of two hexadecimal digits is the same as their decimal sum, provided the decimal sum is less than 16. If the decimal sum is 16 or greater, we subtract 16 to obtain the hexadecimal sum and carry 1 to the column immediately left to it.

Example 25.13:

$$\begin{array}{cccccc}
 & & \text{Decimal} & & & \\
 & & \text{sum} & & & \\
C & \Rightarrow & 12 & \quad A & \quad E & \\
+3 & \Rightarrow & 3 & \quad +2 & \quad +D & \\
\hline
F & & 15 & \quad C & \quad B, & \text{carry } 1 \\
\hline
\end{array}$$

Hexadecimal subtraction is done using the complement method, which we learn in a latter section. Hexadecimal multiplication is performed by converting the decimal equivalents of partial products to hexadecimal.