Lecture 4

Learning outcomes:

- *The Binary Number System.*

The Binary Number System

- The **binary number system** is a number system that uses only two digits: **0** and **1**.

- It is also called the **base-2** system because it has just two possible values for each digit.

- This system is fundamental to computer science and digital electronics because modern computers operate using binary data, which represents electrical signals as two states: off (0) and on (1).

Key Concepts of the Binary Number System:

❖ ***Digits and Place Value:***

➢ Each digit in a binary number is called a bit (binary digit).

➢ The place value of each bit in a binary number is a power of 2, starting from 2^0 on the right (just like in the decimal system, where each place value is a power of 10).

❖ ***Binary to Decimal Conversion:***

➢ To convert a binary number to decimal, multiply each bit by its corresponding power of 2 and sum the results.

➢ Example: Convert binary 1101 to decimal:

$1 \times 2^3 = 8$

$1 \times 2^2 = 4$

$0 \times 2^1 = 0$

$1 \times 2^0 = 1$

✓So, 1101 in binary is 8+4+0+1=13 in decimal.

❖ Decimal to Binary Conversion:

- To convert a decimal number to binary, repeatedly divide the decimal number by 2, recording the remainder each time (either 0 or 1).
- The binary equivalent is formed by reading the remainders from bottom to top.
- Example: Convert decimal 19 to binary:

19 ÷ 2 = 9 remainder 1

9 ÷ 2 = 4 remainder 1

4 ÷ 2 = 2 remainder 0

2 ÷ 2 = 1 remainder 0

1 ÷ 2 = 0 remainder 1

✓Reading the remainders from bottom to top, the binary representation of 19 is 10011.

Decimal vs Binary: Here are some equivalent values:

Decimal:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Binary:	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111

Decimal:	20	25	30	40	50	100	200	500
Binary:	10100	11001	11110	101000	110010	1100100	11001000	111110100

Binary Arithmetic:

- **Binary Addition:** Binary addition works like decimal addition, but the values carry over when the sum is 2 (in binary, 1 + 1 = 10). Here's how it works:

➢Binary Addition Rules:

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10 (carry 1)

- Example of Binary Addition: Let's add two binary numbers: 1011 (11 in decimal) and 1101 (13 in decimal).

1011

+

1101

✓11000 (Result)

Steps:

1. Add the rightmost bits: 1+1=10 (write down 0 and carry 1).
2. Next, add: 1+0+1(carry)=10 (write down 0 and carry 1).
3. Continue: 0+1+1(carry)=10 (write down 0 and carry 1).
4. Add: 1+1+1(carry)=11 (write down 1 and carry 1).
5. Write down the remaining carry of 1.

✓Final result: 11000 = 24

- **Binary Subtraction:** Binary subtraction follows similar rules as decimal subtraction, but borrowing occurs when subtracting 1 from 0. Borrowing in binary takes from the nearest 1 to the left, changing that bit to 0 and giving a 2 to the bit being borrowed from.

➢Binary Subtraction Rules:

0 - 0 = 0

1 - 0 = 1

1 - 1 = 0

0 - 1 = 1 (borrow 1)

- Example of Binary Subtraction: Subtract 1010 (10 in decimal) from 1101 (13 in decimal).

1101

-

1010

✓0011 (Result)

Steps:

1. Rightmost bits: 1−0=1.
2. Next bit: 0−1 requires borrowing. Borrow from the next 1(turn it into 0), making the bit 10, so 10−1=1.
3. Next bit: 0−0=0.
4. Leftmost bit: 1−1=0.

✓Final result: 0011 = 3

- **Another example:** Subtract 110010 (50 in decimal) from 1111 (15 in decimal).

110010

-

1111

100011

✓ Result is 100011 (35 in decimal)

Complement Methods in Binary Subtraction

- Complement methods are commonly used for binary subtraction because they allow subtraction to be transformed into addition, simplifying the hardware or algorithm needed for binary arithmetic.

- Binary complements are used to represent negative decimal numbers. A binary number can have different types of complements, but the 1's and 2's complements are the most common methods.

- By simply inverting the binary number, we can find the binary number's 1's complement. Adding 1 to the least significant bit and changing each bit from 0 to 1 we can find the 2's complement of any binary number.

Rule using 1s Complement:

- ✓***Step 1:*** Given numbers must be in the form X-Y with same digits. Extra 0 can be added at the beginning to make same digits.

- ✓***Step 2:*** Calculate 1's complement of ‘Y ’

- ✓***Step 3:*** Add result of step 2 with ‘X’

- ✓***Step 4:*** If there is an extra bit, remove that extra bit and add on its remaining bits. If there is no extra bit, find 1's Complement of result in step 3 and add (-)ve sign.

Rule using 2s Complement:

- ✓***Step 1:*** Given numbers must be in the form X-Y with same digits. Extra 0 can be added at the beginning to make same digits.

- ✓***Step 2:*** Calculate 2's complement of 'Y'

- ✓***Step 3:*** Add result of step 2 with 'X'

- ✓***Step 4:*** If there is an extra bit, remove that and the remaining bits will be the answer. If there is no extra bit, find 2's Complement of result in step 3 and add (-)ve sign.

Examples

- Subtract (1010 which is 10 in decimal) from (1111 which is 15 in decimal) using 1's and 2's complement, (1010) – (1111).

- ***Using 1's Complement***

✓First calculating 1's complement of (1111) is (0000).

✓Adding (0000) with (1010), we get (0000) + (1010) = (1010).

✓Since, there is no extra bit i.e. 4 digits added with 4 digits and gives 4 digits result.

✓Calculating 1's complement of (1010) we get (0101) and putting (-)ve sign.

✓Hence, result is (**0101**) which is -5 in decimal.

- ***Using 2's Complement***

✓First calculating 2's complement of (1111) is (0000) + (1) = (0001).

✓Adding (0001) with (1010) we get, (0001) + (1010) = (1011).

✓Since, there is no extra bit i.e. 4 digits added with 4 digits and gives 4 digits result.

✓Calculating 2's complement of (1011) we get (0100) + (1) = (0101) and putting (-)ve sign.

✓Hence, result is (**0101**) which is -5 in decimal.

- Subtract (11 which is 3 in decimal) from (100 which is 4 in decimal) using 1's and 2's complement, (11) – (100).

- ***Using 1's Complement***

✓First calculating 1's complement of (100) is (011).

✓Adding (011) with (011), we get (011) + (011) = (110).

✓Since, there is no extra bit i.e. 3 digits added with 3 digits and gives 3 digits result.

✓Calculating 1's complement of (110) we get (001) and putting (-)ve sign.

✓Hence, result is -(001) which is -1 in decimal.

➢ ***Using 2's Complement***

✓First calculating 2's complement of (100) is (011) + (1) = (100).

✓Adding (100) with (011), we get (100) + (011) = (111).

✓Since, there is no extra bit i.e. 3 digits added with 3 digits and gives 3 digits result.

✓Calculating 2's complement of (111) we get (000) + (1) = (001) and putting (-)ve sign.

✓Hence, result is -(001) which is -1 in decimal.

- Subtract (1111 which is 15 in decimal) from (1010 which is 10 in decimal) using 1's and 2's complement, (1111) – (1010).

- ***Using 1's Complement***

✓First calculating 1's complement of (1010) is (0101).

✓Adding (0101) with (1111), we get (0101) + (1111) = (10100).

✓Since, there is an extra bit i.e. 4 digits added with 4 digits and gives 5 digits result.

✓Remove the extra bit and add on its remaining bits, (0100) + (1) = (0101).

✓Hence, result is (**0101**) which is 5 in decimal.

- ***Using 2's Complement***

✓First calculating 2's complement of (1010) is (0101) + (1) = (0110).

✓Adding (0110) with (1111) we get, (0110) + (1111) = (10101).

✓Since, there is an extra bit i.e. 4 digits added with 4 digits and gives 5 digits result.

✓Remove the extra bit and the remaining bits will be the answer (0101).

✓Hence, result is (**0101**) which is 5 in decimal.

- **Binary Multiplication:** Binary multiplication works similarly to decimal multiplication. The digits are multiplied, and binary addition is used to combine partial products. Multiplying by 0 always results in 0, and multiplying by 1 results in the other number.

➢Binary Multiplication Rules

$0 \times 0 = 0$

$0 \times 1 = 0$

$1 \times 0 = 0$

$1 \times 1 = 1$

- Example of Binary Multiplication: Multiply 101 (5 in decimal) by 11 (3 in decimal).

```
    101
×
     11
--------
    101     (101 × 1)
+
  1010    (Shift left and multiply by 1)
--------
✓1111    (Result)
```

Steps:

1. Multiply 101 by the rightmost 1, giving 101.
2. Multiply 101 by the next 1, giving 101, but shift it one place to the left (1010).
3. Add the two results: 101+1010=1111 = 15

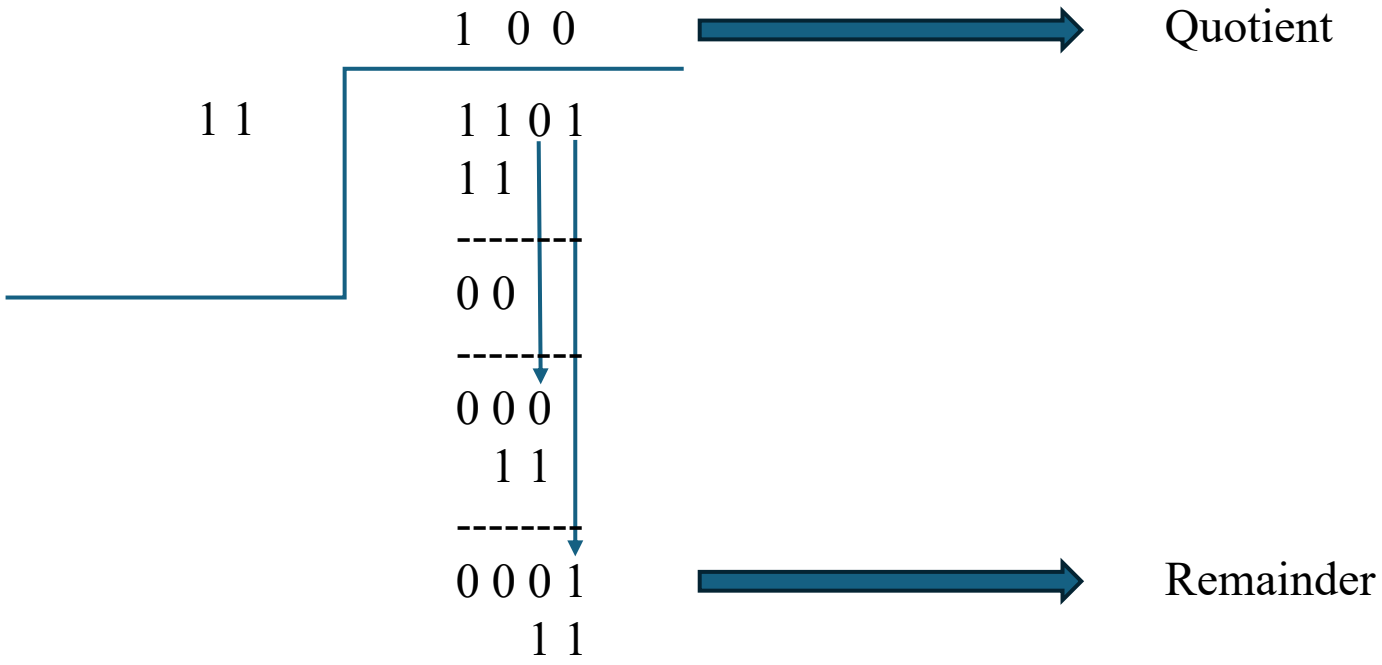
➢ **Another example:** Multiply 1111 (15 in decimal) by 1111 (15 in decimal).

```
1111
 ×
1111
-----------
      1111
     11110      +
    111100
   1111000
-----------
11100001
```

✓ Result is 11100001 (225 in decimal)

❑ **Binary Division:** Binary division is similar to long division in decimal. The divisor is repeatedly subtracted from the dividend, shifting the result and bringing down bits as needed.

➢ Example of Binary Division: Divide 1101 (13 in decimal) by 11 (3 in decimal).

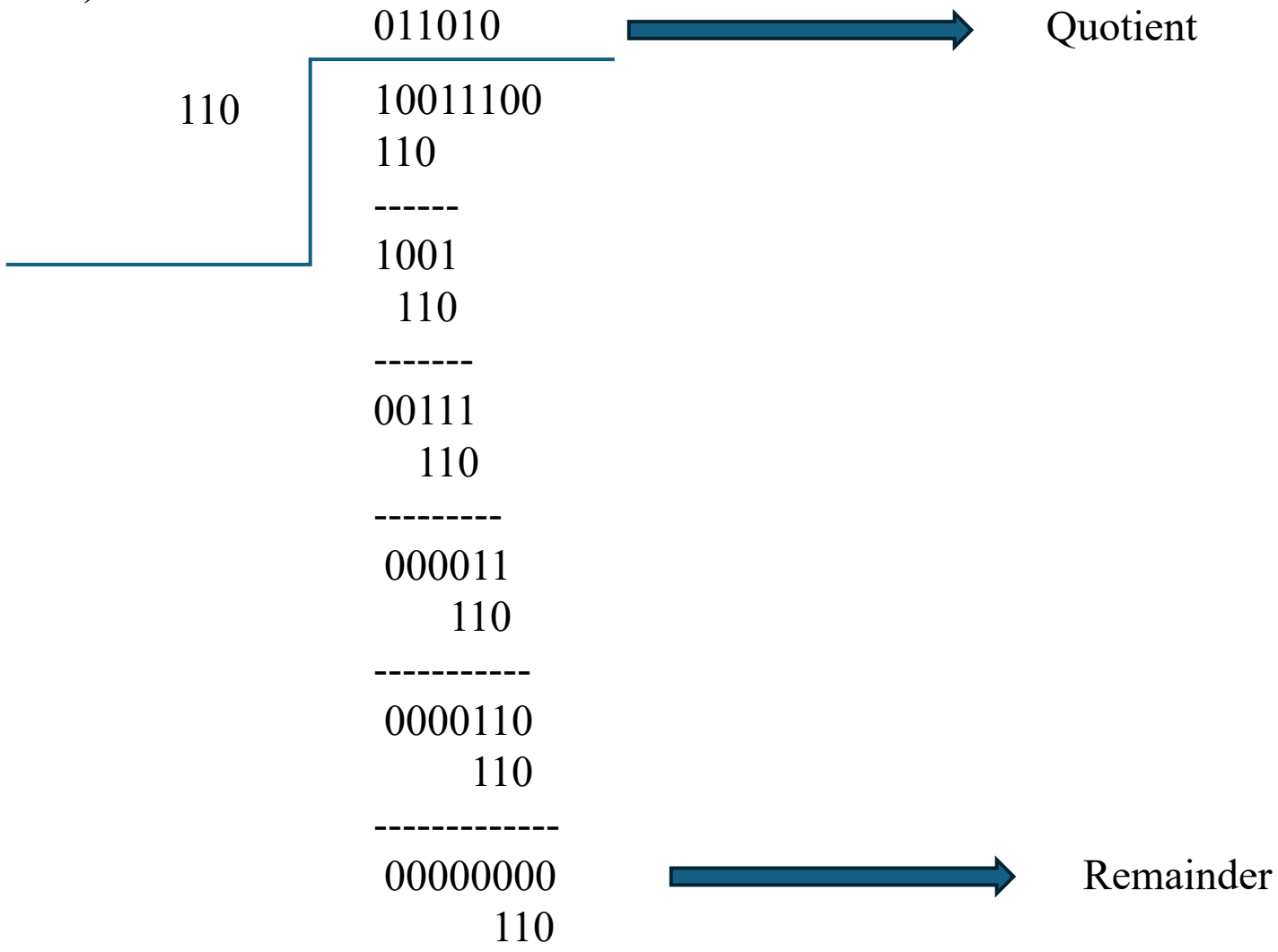


Steps:

1. Start with the leftmost bits of the dividend: 11 (this is equal to the divisor).
2. Write 1 as the first digit of the quotient (1) and subtract 11 from 11 to get 00.
3. Bring down the next bit, making it 000.
4. Since 000 is less than the divisor, write 0 in the quotient (10) and bring down the next bit to get 0001.
5. Since 0001 is less than the divisor, write 0 in the quotient (100).
6. The final quotient is 100 and the remainder is 0001(which equals 4 in decimal, with a remainder of 1).

❑ **Another example:** Divide 10011100 (156 in decimal) by 110 (6 in decimal).

✓ Result is 11010 (26 in decimal)



The importance of binary code

The binary number system is the base of all computing systems and operations. It enables devices to store, access and manipulate all types of information directed to and from the CPU or memory. This makes it possible to develop applications that enable users to do the following:

- view websites;
- create and update documents;
- play games;
- view streaming video and other kinds of graphical information;
- access software; and
- perform calculations and data analyses.

Applications of the Binary System:

- ***Digital Data:*** All data in a computer, whether text, images, audio, or video, is represented in binary.
- ***Computer Memory:*** Binary is used to represent memory addresses and data values.
- ***Logic Gates:*** In digital circuits, the binary system is used to design logic gates (AND, OR, NOT, etc.), which are the building blocks of CPUs and other digital components.
- ***Binary Encoding:*** Data is often encoded in binary formats like American Standard Code for Information Interchange (ASCII) (for text).