



# ARITHMETIC MICRO-OPERATIONS

# AGENDA

1. What are Micro-operations and Arithmetic Micro-operations ?
2. Basic Arithmetic Micro-Operations .
3. Binary Adder
4. Adder-Subtractor
5. Binary incrementer
6. Binary decremener
7. Shift Micro-Operations (Only Arithmetic Perspective)

# WHAT ARE MICRO-OPERATIONS ?

A **microoperation** is an elementary operation performed with the data stored in registers. The microoperations encountered in digital computers are classified into four categories:

1. **Register transfer microoperations**- transfer one binary information from one register to other.
2. **Arithmetic microoperations**- perform arithmetic operations on numeric data stored in registers.
3. **Logic microoperations**- perform bit manipulation operations on non-numeric data stored in registers.
4. **Shift microoperations**-perform shift operations on data stored in registers.

The background is a dark blue gradient. In the corners, there are white line-art illustrations of circuit traces and nodes. Top-left: several lines with circular nodes. Top-right: a few lines with circular nodes. Bottom-left: a cluster of lines with circular nodes. Bottom-right: a few lines with circular nodes.

**ARITHMETIC MICROOPERATIONS-** PERFORM ARITHMETIC OPERATIONS ON  
NUMERIC DATA STORED IN REGISTERS.

# BASIC ARITHMETIC MICRO-OPERATIONS:

- addition

$$R3 \leftarrow R1 + R2$$

- subtraction

$$R6 \leftarrow R5 - R4$$

- increment

$$R1 \leftarrow R1 + 1$$

- decrement

$$R1 \leftarrow R1 - 1$$

- shift

$$R1 \leftarrow \text{ashr } R1 \ // \ R1 \leftarrow \text{ashl } R1$$

The background is a dark blue gradient. In the corners, there are white line-art illustrations of circuit boards or neural networks, with lines connecting to small circles.

# WHAT ABOUT MULTIPLICATIONS AND DIVISIONS ?

(MULTIPLICATIONS AND DIVISIONS ARE NOT CONSIDERED MICRO-OPERATIONS.)



**Multiplication** is implemented by a sequence of **adds and shifts**.

and

**Division** is implemented by a sequence of **subtract and shifts**.



# SOME ARITHMETIC MICRO-OPERATIONS

Symbolic Designation	Descriptions
$R3 \leftarrow R1 + R2$	Contents of R1 plus R2 transferred to R3
$R3 \leftarrow R1 - R2$	Contents of R1 minus R2 transferred to R3
$R2 \leftarrow R2'$	Complement contents of R2 (1's Complement)
$R2 \leftarrow R2' + 1$	2's Complement contents of R2 (negate)
$R3 \leftarrow R1 + R2' + 1$	R1 plus 2's Complement of R2
$R1 \leftarrow R1 + 1$	Increment content of R1 by 1
$R1 \leftarrow R1 - 1$	Decrement content of R1 by 1



The background is a dark blue-grey color. In the four corners, there are decorative white line art elements that resemble circuit traces or a stylized city skyline. These elements consist of thin lines connecting small circles, some of which are larger than others.

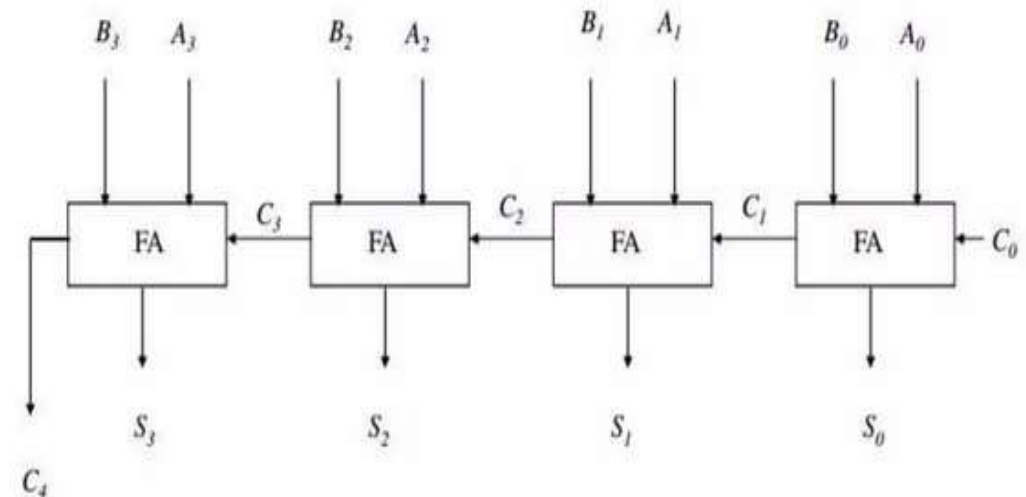
Half Adder is 2 bits and Full Adder is 3 bits.

A full adder basic consist of 2 half adders and an OR gate.

# BINARY ADDER

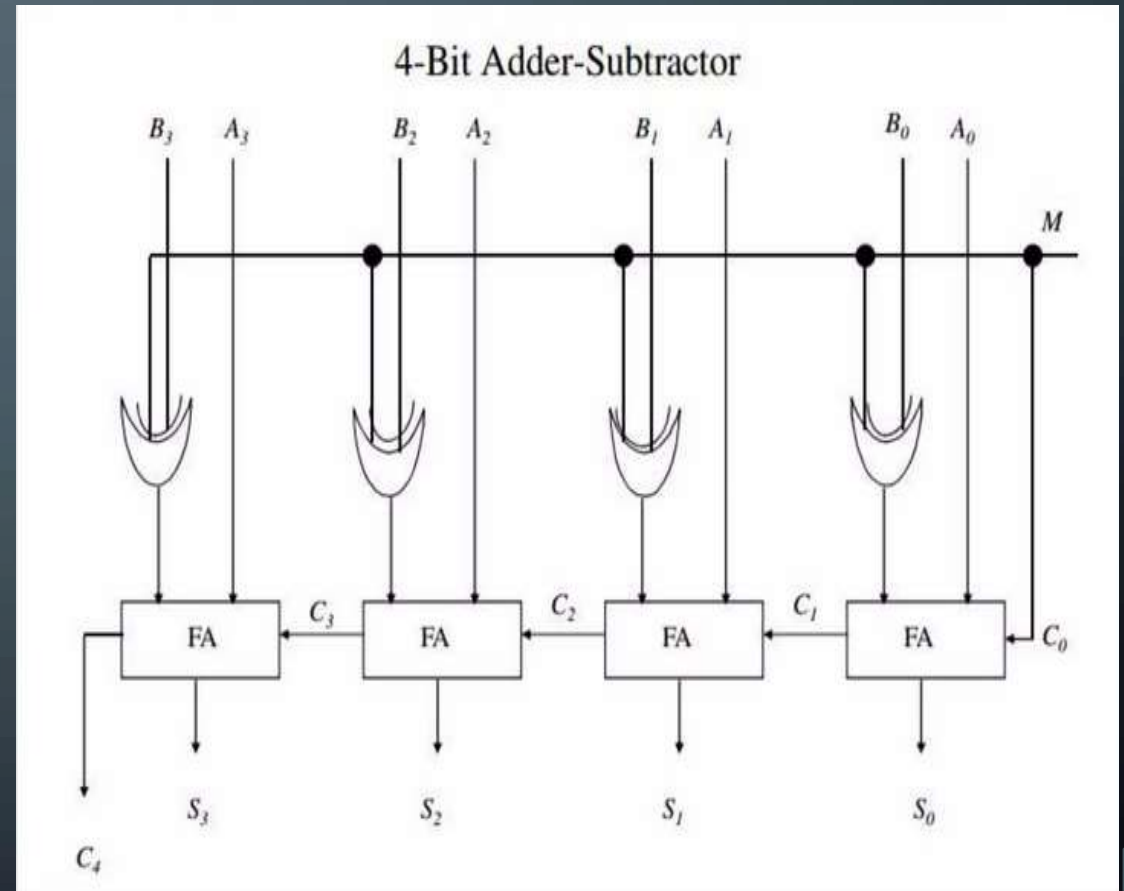
- We implement a binary adder with registers to **hold the data and a digital circuit to perform the addition**.
- The Binary adders is **constructed using full adders connected in cascade** so that the carry produced by one full adder becomes an input for the next.
- Adding two n-bit numbers requires n full adders.
- The n data bits for A and B might come from R1 and R2 respectively.

4-Bit Binary Adder



# ADDER-SUBTRACTER

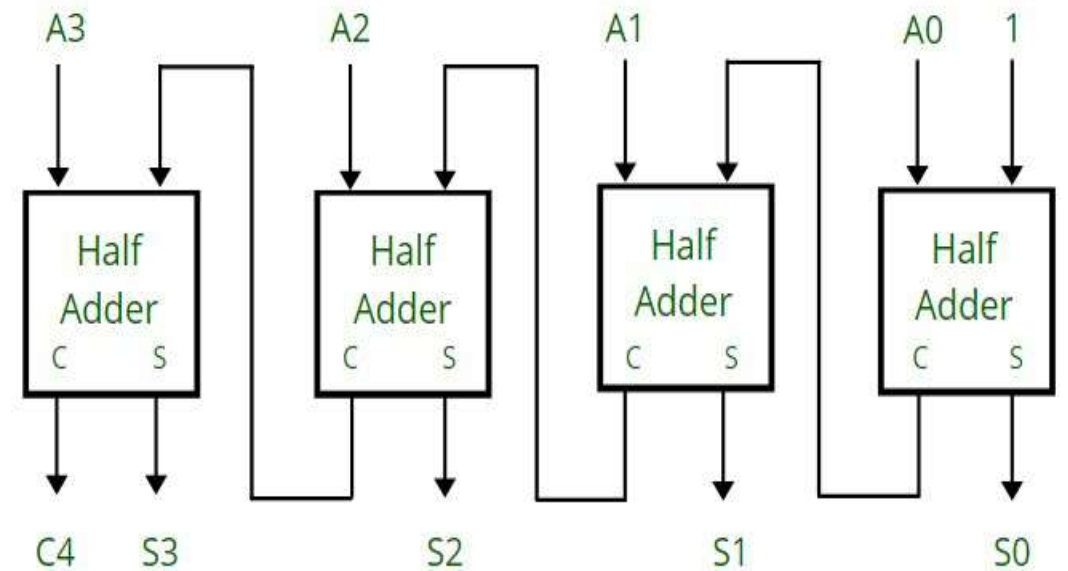
- Subtracting  $A - B$  is mostly easily done by adding  $B'$  to  $A$  and then adding 1.
- This makes it convenient to combine both addition and subtraction into one circuit, called an adder-subtractor.
- $M$  is the mode indicator or Control (CTR)
  - $M = 0$  indicates addition (  $B$  is left alone and  $C_0$  is 0.)
  - $M = 1$  indicates subtraction (  $B$  is complement and  $C_0$  is 1)



$$\text{XOR GATE} = A.M' + A' + M$$
$$M = 0 ; \text{XOR} = A$$
$$M = 1 ; \text{XOR} = A'$$

# BINARY INCREMENTER

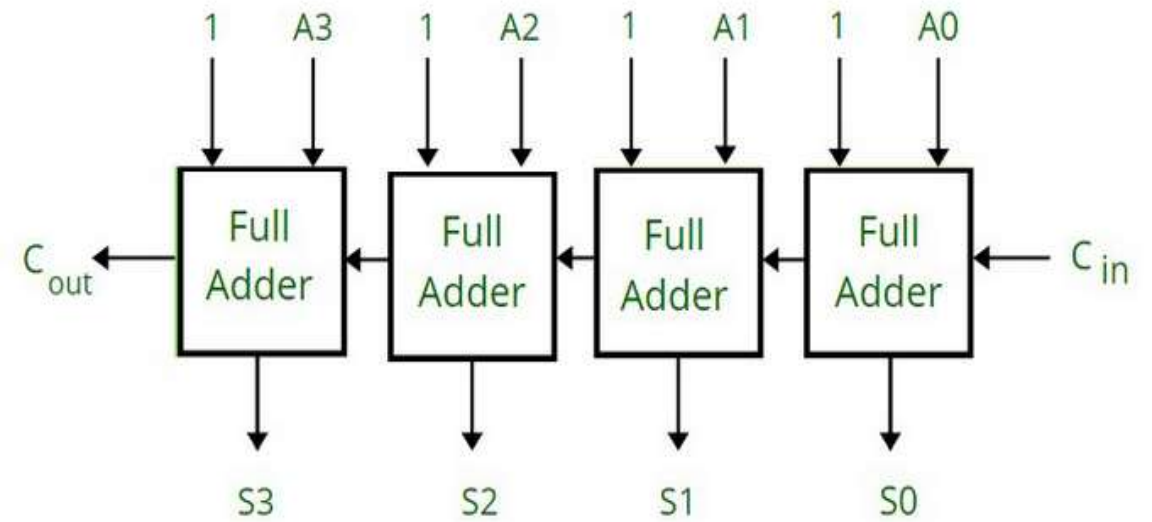
- The **binary incrementer adds 1 to the contents of a register**, e.g. , a register storing 0101 would have 0110 in it after being incremented.
- There are times when we want **incrementing done independent of a register**. We can accomplish this with **a series of cascading half-adders**.



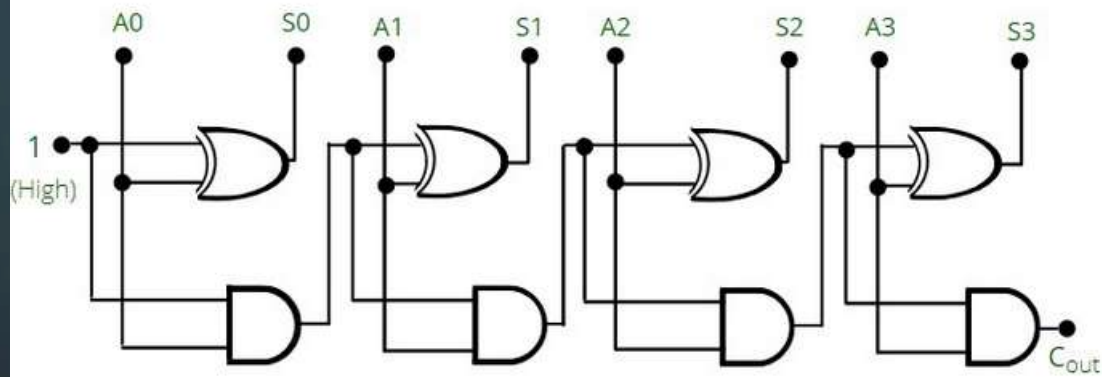
4- Bit Binary Incrementer

# BINARY DECREMENTER

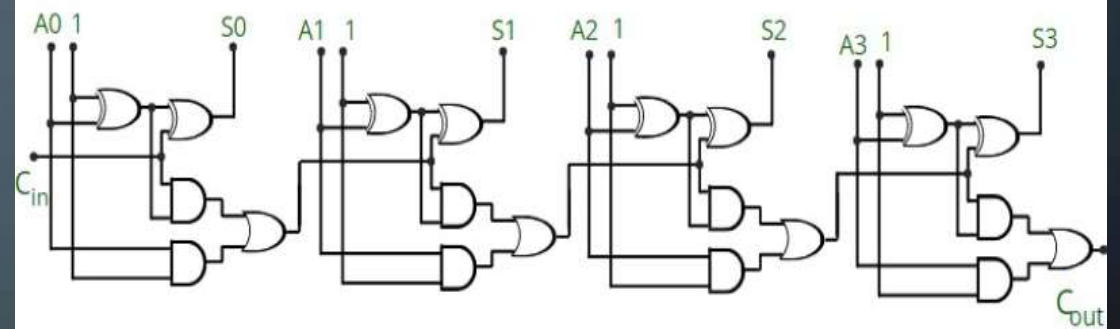
- The binary decrementer subtracts 1 to the contents of a register.
- For any n- bit binary decrementer, 'n' refers to the storage capacity of the register which needs to be decremented by 1. So we require 'n' number of full adders.



4- Bit Binary Decrementer



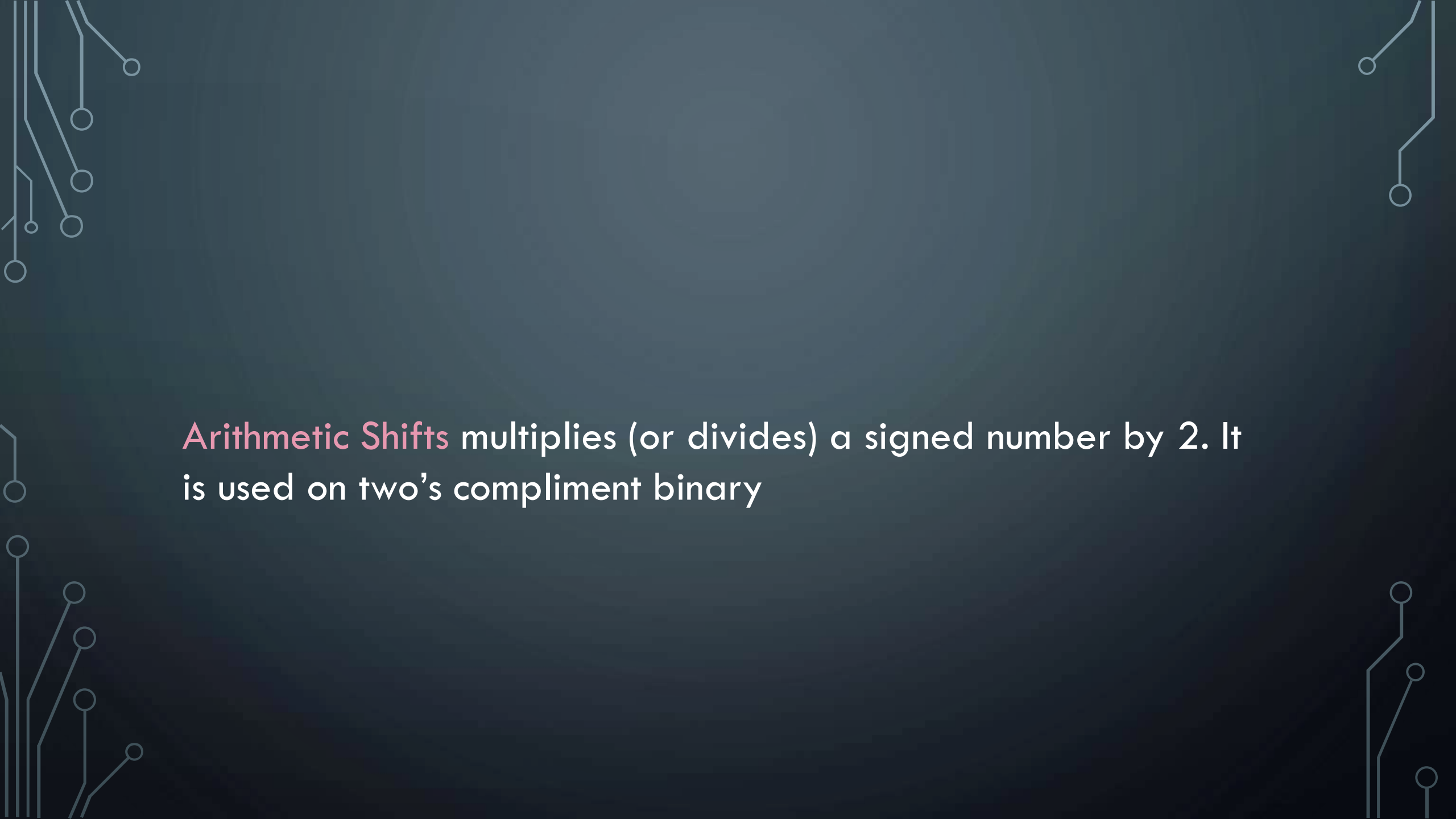
4- Bit Binary Incrementer (In detail)



4 Bit Binary Decrementer (In Detail)

# SHIFT MICRO-OPERATIONS

- Shift Micro-operations are used for serial transfer of data and are used in conjunction with arithmetic and logical micro-operations.
- The Register contents can be shifted to the left or to the right.
- There are three types of Shift Operations:
  - Logical Shifts
  - Arithmetic Shifts
  - Circular Shifts

The background is a dark blue gradient. In the corners, there are decorative white line art elements resembling circuit boards or neural networks, with lines and small circles.

**Arithmetic Shifts** multiplies (or divides) a signed number by 2. It is used on two's complement binary



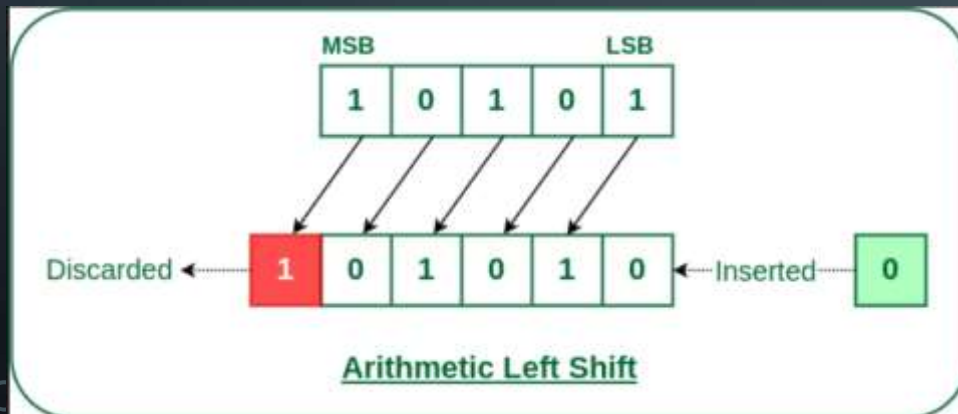
# SHIFT MICRO-OPERATIONS (ARITHMETIC)

Symbolic Designation	Descriptions
$R \leftarrow \text{ashl } R$	Arithmetic shift-left register R
$R \leftarrow \text{ashr } R$	Arithmetic shift-right register R

# ARITHMETIC SHIFT OPERATIONS

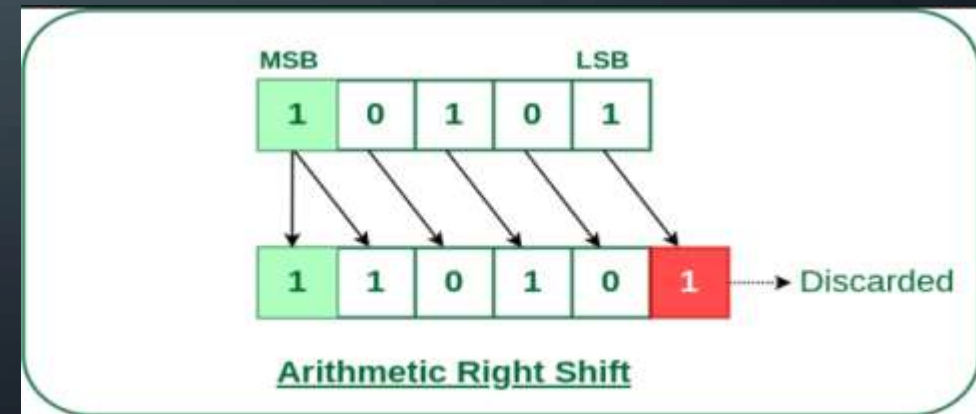
## ARITHMETIC LEFT SHIFT OPERATION

- In this shift, **each bit is moved to the left one by one**. The empty least significant bit (LSB) is filled with zero and the most significant bit (MSB) is rejected.



## ARITHMETIC RIGHT SHIFT OPERATION

- In this shift, **each bit is moved to the right one by one** and the least significant (LSB) bit is rejected and the empty most significant bit (MSB) is filled with the value of the previous MSB.





# PRAISE BE!!!

- SHAH MALAV

- SHAH MRUNAL