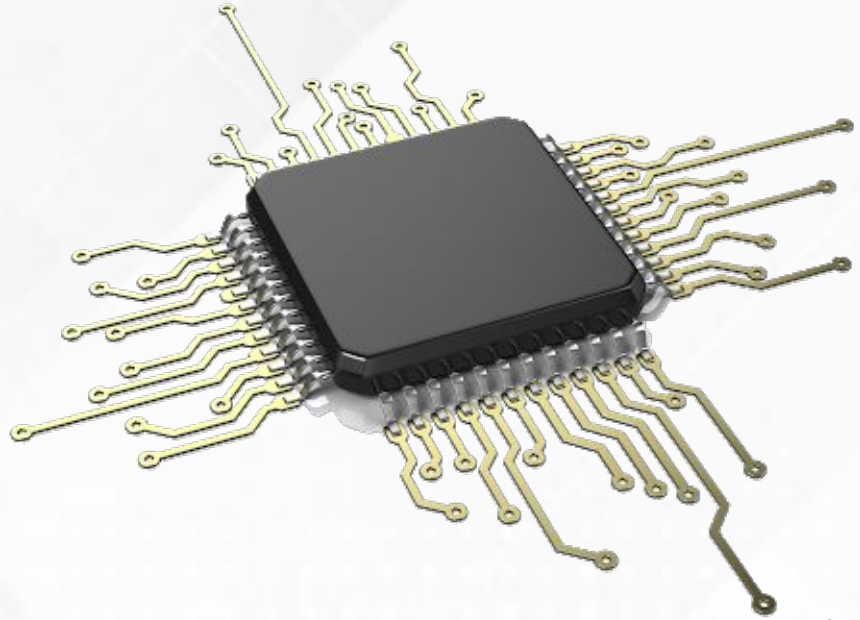
Computer Organization & Architecture
(COA)
GTU # 3140707

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Unit-1

Computer Data Representation & Register Transfer and Micro-operations



Prof. Krunal D. Vyas
Computer Engineering Department
Darshan Institute of Engineering & Technology,
Rajkot
krunal.vyas@darshan.ac.in
9601901005

Outline

- Basic computer data types & Complements
- Fixed point & Floating point representation
- Register transfer language
- Bus and Memory transfers (Three-State Bus Buffers, Memory Transfer)
- Arithmetic Micro-operations
- Logic Micro-operations
- Shift Micro-operations
- Arithmetic logical shift unit

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Basic computer data types & Complements

Section - 1

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Basic Computer Data Types & Complements

Basic Computer Data Types

- Decimal Number System
  - Radix(r) = 10, Number range = 0 - 9
- Binary Number System
  - Radix(r) = 2, Number range = 0 - 1
- Octal Number System
  - Radix(r) = 8, Number range = 0 - 7
- Hexadecimal Number System
  - Radix(r) = 16, Number range = 0 - 9 & A - F
- Binary Coded Decimal Numbers

Complements

- (r-1)'s complement
  - 9's complement
  - 1's complement
  - 7's complement
  - 15's complement
- r's complement
  - 10's complement
  - 2's complement
  - 8's complement
  - 16's complement

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Fixed point & Floating point representation

Section - 2

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Fixed point representation

- Integer Representation
  - Signed-magnitude representation
  - 1's complement representation
  - 2's complement representation
- Arithmetic Addition
  - Using 1's complement
  - Using 2's complement
- Arithmetic Subtraction
  - Using 1's complement
  - Using 2's complement

- Example: Represent (-14)
  - Signed-magnitude representation : 1 0001110
  - 1's complement representation : 1 1110001
  - 2's complement representation : 1 1110010

+ 6	00000110
+ 13	00001101
+ 19	00010011

+ 6	00000110
- 13	11110011
- 7	11111001

Note: negative number must initially be in 2's complement

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Floating point representation

- Two parts
  - *Mantissa* – Represents signed fixed-point number
  - *Exponent* – Represents position of the decimal point
- For example, the decimal number +6132.789 is represented in floating-point with a fraction and an exponent as follows:

- The value of the exponent indicates that the actual position of the decimal point is four positions to the right of the indicated decimal point in the fraction.
- Floating-point is always interpreted to represent a number in the following form: $m \times r^e$
- Here, *m* stands for *mantissa*, *e* stands for *exponent*, *r* means *radix*

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Floating point representation

- Normalization
  - A floating point number is said to be normalized if the most significant digit of the mantissa is nonzero.
  - For example, the decimal number 350 is normalized but 00035 is not.
- ANSI 32-bit floating point byte format

- *S = Sign of Mantissa, E = Exponent bits, M = Mantissa bits*
- Example, $13 = 1101 = 0.1101 \times 2^4$

  $= 00000100\ \ .11010000\ \ 00000000\ \ 00000000$
- Example, $-17 = -\ 10001 = -0.10001 \times 2^5$

  $= 10000101\ \ .10001000\ \ 00000000\ \ 00000000$

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Register transfer language

Section - 3

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

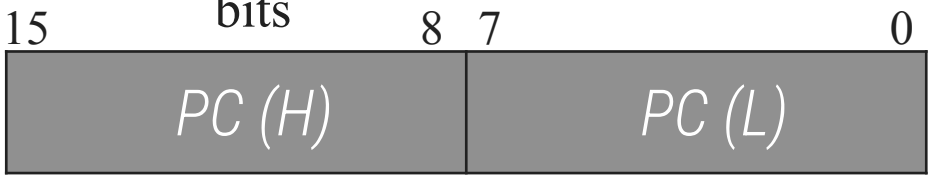
Register

- Computer Registers are designated by capital letters.
- For example,
  - MAR – Memory Address Register
  - PC – Program Counter
  - IR – Instruction Register
  - R1 – Processor Register

Register
R

Numbering of bits

Showing individual bits



Divided into two parts

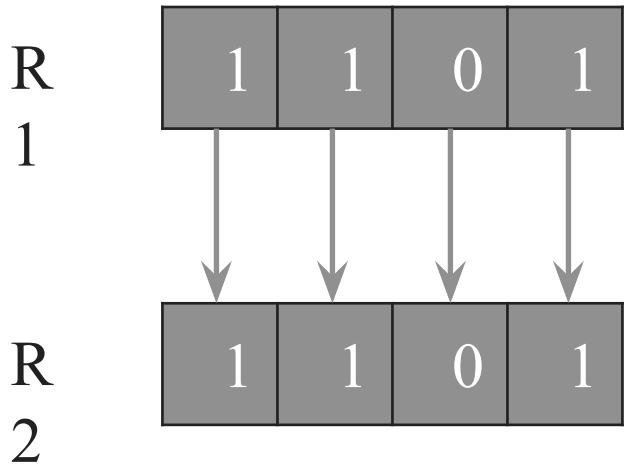
Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Register Transfer Language

- Information transfer from one register to another is designated in symbolic form by means of a replacement operator is known as Register Transfer.
- The statement

$$R2 \leftarrow R1$$

- denotes a transfer of the content of register *R*1 into register *R*2.



- The symbolic notation used to describe the microoperation transfers among registers is called a register transfer language.
- The term *"register transfer"* implies the availability of hardware logic circuits that can perform a stated microoperation and transfer the result of the operation to the same or another register.
- A register transfer language is a system for expressing in symbolic form the microoperation sequences among the registers of a digital module.

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

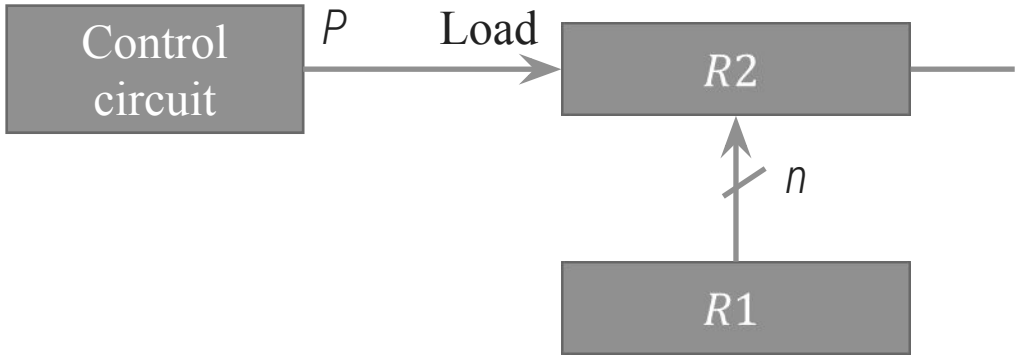
Register Transfer with Control Function

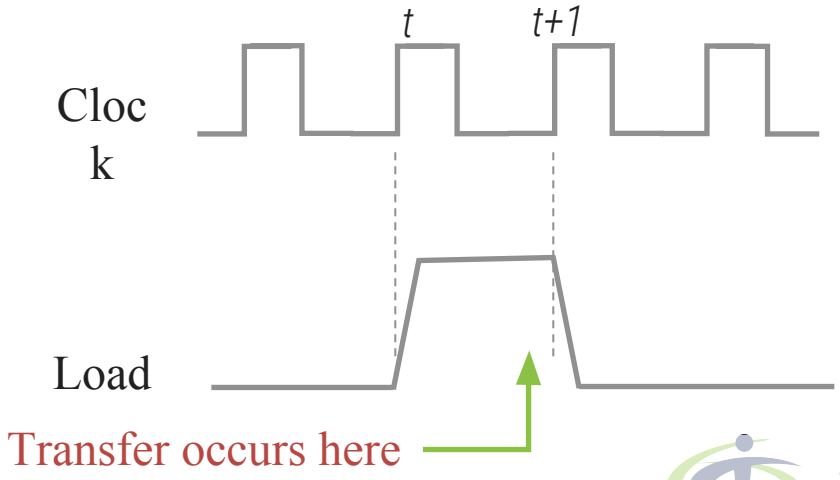
- Normally, we want the transfer to occur only under a predetermined control condition using if-then statement.

$$\text{If (P = 1) then } (R2 \leftarrow R1)$$

- where P is a control signal generated in the control section.
- A control function is a boolean variable that is equal to 1 or 0. The control function is included in the statement as follows:

$$P : R2 \leftarrow R1$$





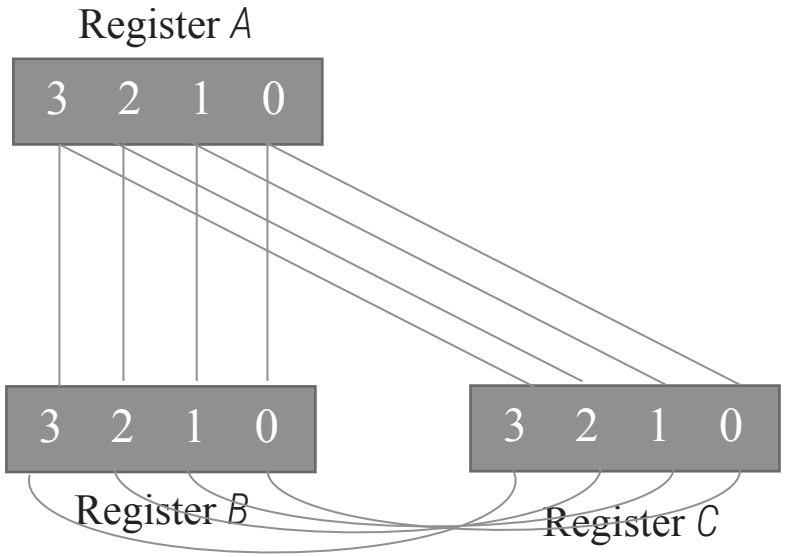
Bus and Memory transfers

Section - 4

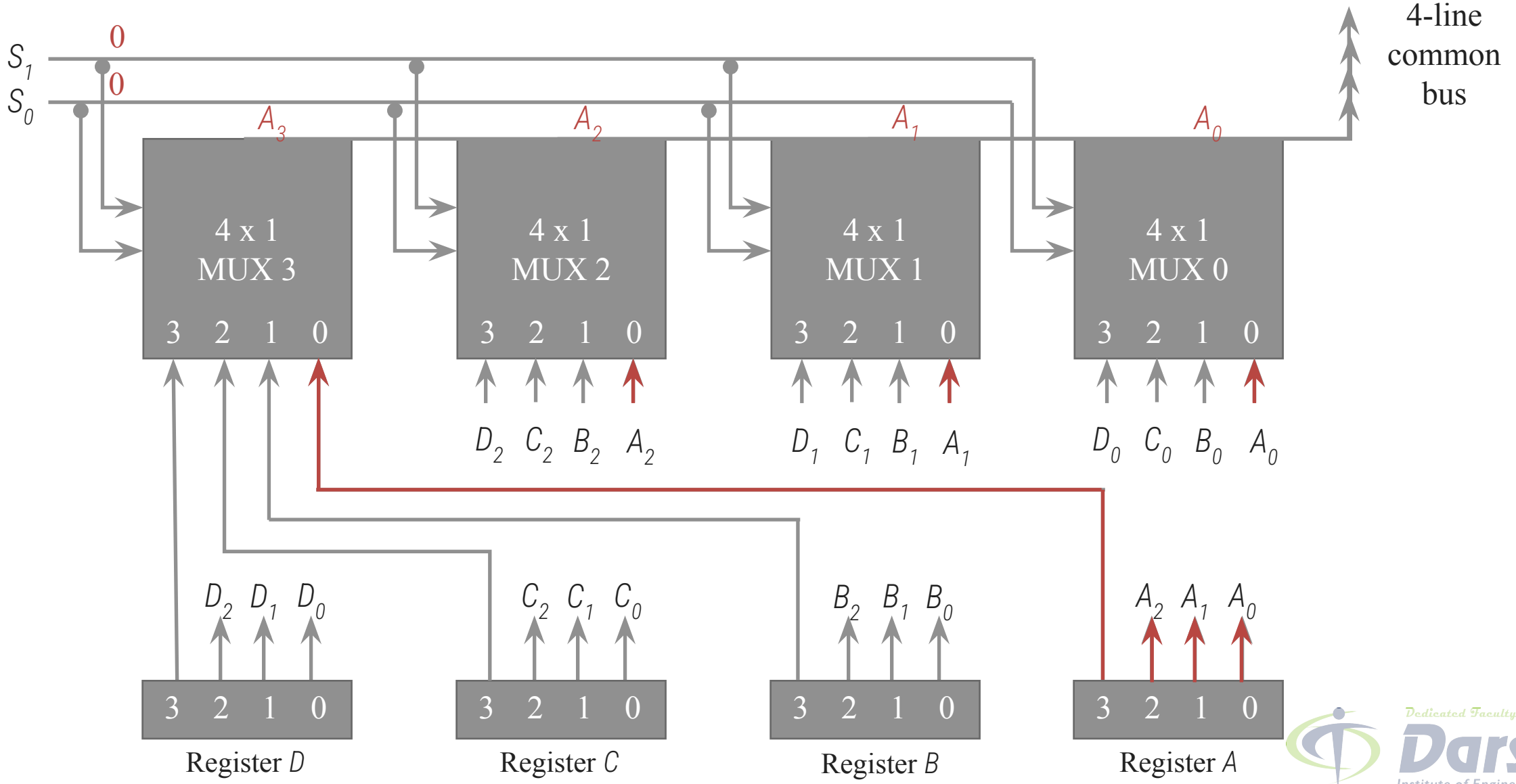
Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Common Bus System for 4 registers

- A typical digital computer has many registers, and paths must be provided to transfer information from one register to another.
- The number of wires will be excessive if separate lines are used between each register and all other registers in the system.
- A more efficient scheme for transferring information between registers in a multiple-register configuration is a common bus system.
- A bus structure consists of a set of common lines, one for each bit of a register, through which binary information is transferred one at a time.
- One way of constructing a common bus system is with multiplexers.
- The multiplexers select the source register whose binary information is then placed on the bus.



Common Bus System for 4 registers



Common Bus System for 4 registers

- The construction of a bus system for four registers is explained earlier.
- Each register has four bits, numbered 0 through 3.
- The bus consists of four 4 x 1 multiplexers each having four data inputs, 0 through 3, and two selection inputs, S_1 and S_0.
- The diagram shows that the bits in the same significant position in each register are connected to the data inputs of one multiplexer to form one line of the bus.
- The two selection lines S_1 and S_0 are connected to the selection inputs of all four multiplexers.
- The selection lines choose the four bits of one register and transfer them into the four-line common bus.
- When $S_1S_0 = 00$, the 0 data inputs of all four multiplexers are selected and applied to the outputs that form the bus.
- This causes the bus lines to receive the content of register A since the outputs of this register are connected to the 0 data inputs of the multiplexers.

Darshan
Institute of Engineering & Technology

Common Bus System for 4 registers

- Table shows the register that is selected by the bus for each of the four possible binary values of the selection lines.
- In general, a bus system will multiplex *k* registers of *n* bits each to produce an *n*-line common bus.
- The number of multiplexers needed to construct the bus is equal to *n*, the number of bits in each register.
- The size of each multiplexer must be *k x 1* since it multiplexes *k* data lines.
- For example, a common bus for eight registers of 16 bits requires

  Multiplexers - 16 of (8 x 1)

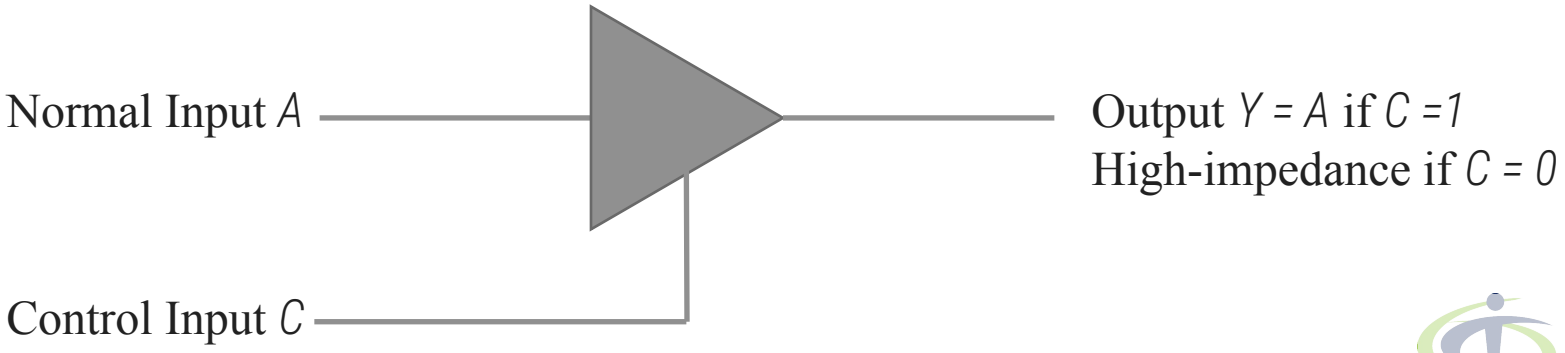
  Select Lines - 3

S_1	S_0	Register Selected
0	0	A
0	1	B
1	0	C
1	1	D

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

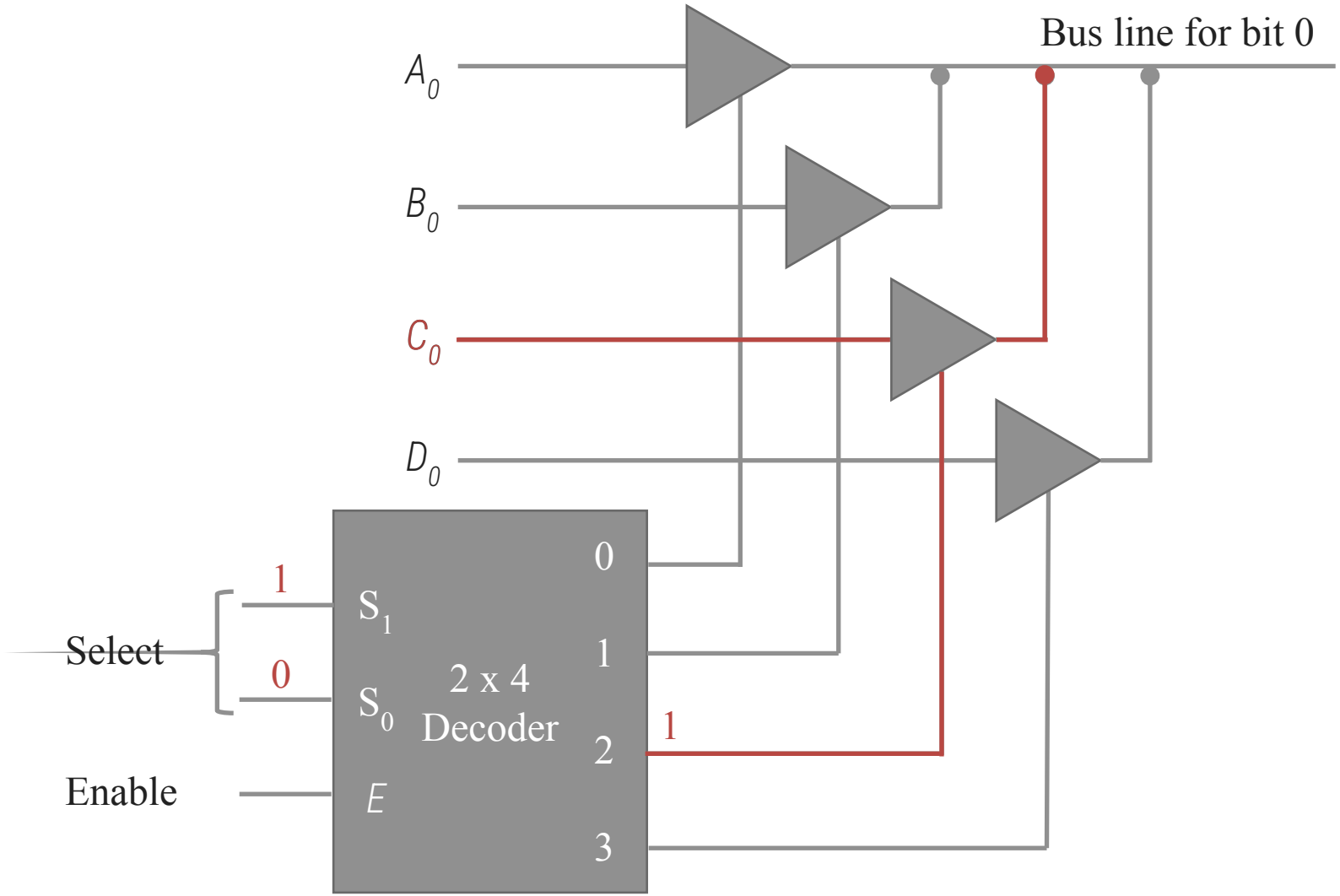
Tri-state Buffer (3 state Buffer)

- A three-state gate is a digital circuit that exhibits three states.
- Two of the states are signals equivalent to logic 1 and 0 as in a conventional gate.
- The third state is high impedance state which behaves like an open circuit, which means that the output is disconnected and does not have logic significance.
- The control input determines the output state. When the control input C is equal to 1, the output is enabled and the gate behaves like any conventional buffer, with the output equal to the normal input.
- When the control input C is 0, the output is disabled and the gate goes to a high-impedance state, regardless of the value in the normal input.



Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Common Bus System using Decoder and Tri-state Buffer



Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Common Bus System using Decoder and Tri-state Buffer

- The construction of a bus system with three-state buffers is demonstrated in previous figure.
- The outputs of four buffers are connected together to form a single bus line.
- The control inputs to the buffers determine which of the four normal inputs will communicate with the bus line.
- The connected buffers must be controlled so that only one three-state buffer has access to the bus line while all other buffers are maintained in a high impedance state.
- One way to ensure that no more than one control input is active at any given time is to use a decoder, as shown in the figure: Bus line with three state-buffers.
- When the enable input of the decoder is 0, all of its four outputs are 0, and the bus line is in a high-impedance state because all four buffers are disabled.
- When the enable input is active, one of the three-state buffers will be active, depending on the binary value in the select inputs of the decoder.

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Arithmetic Micro-operations

Section - 5

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Arithmetic Microoperations

- Arithmetic microoperations perform arithmetic operations on numeric data stored in registers.

Symbolic Designation	Description

Add Microoperation

$$R3 \leftarrow R1 + R2$$

Subtract Microoperation

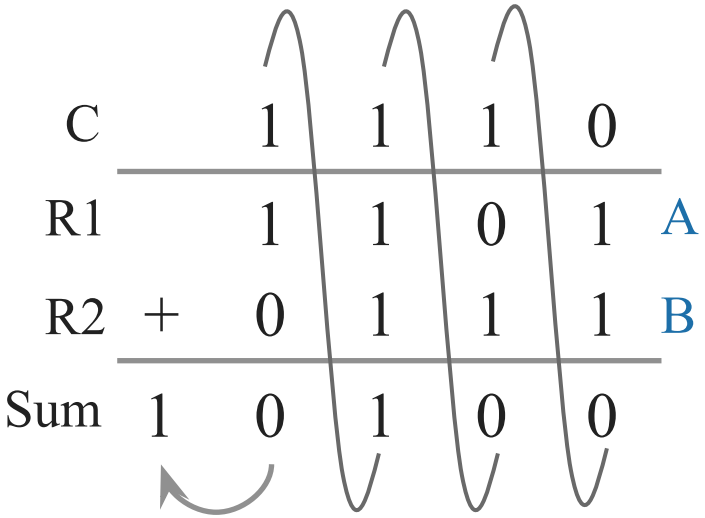
$$R3 \leftarrow R1 - R2$$

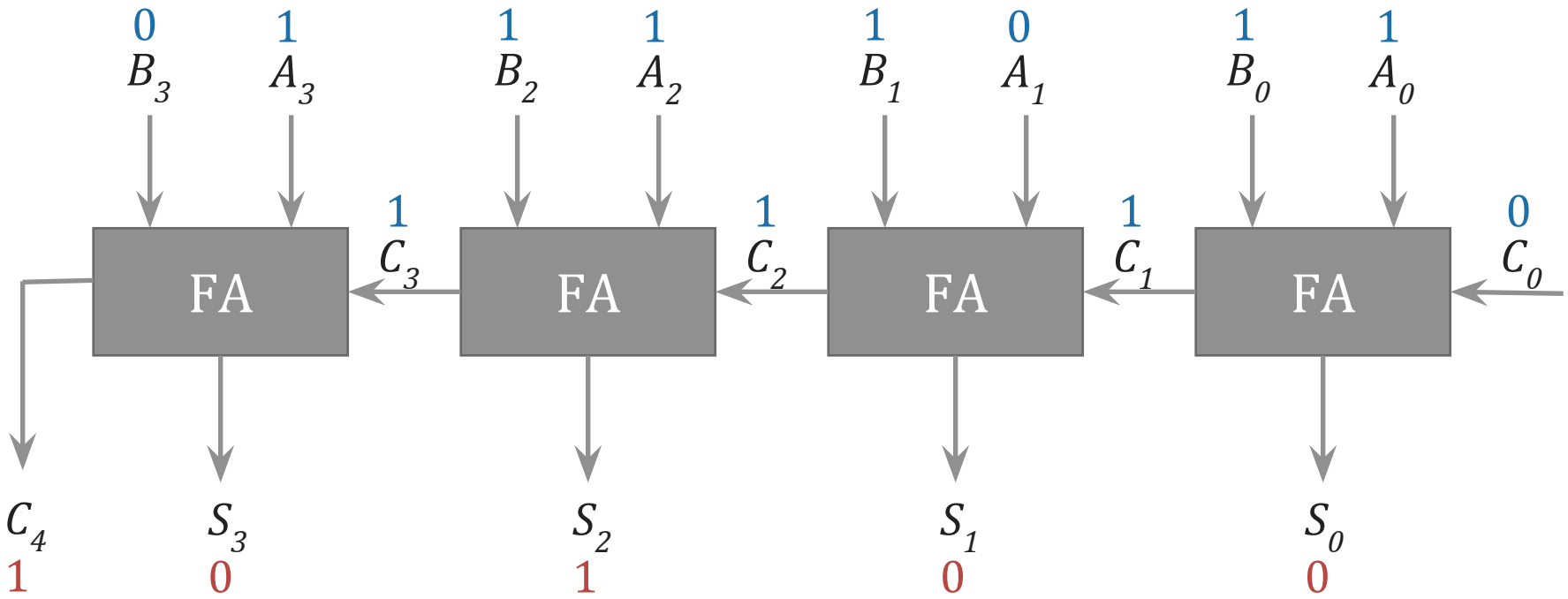
$$R3 \leftarrow R1 + \overline{R2} + 1$$

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4-bit Binary Adder

- The digital circuit that generates the arithmetic sum of two binary numbers of any length is called a binary adder.
- Example





Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

- The binary adder is constructed with full-adder circuits connected in cascade, with the output carry from one full-adder connected to the input carry of the next full-adder.
- The figure shows the interconnections of four full-adders (FA) to provide a 4-bit binary adder.
- The augends bits of A and the addend bits of B are designated by subscript numbers from right to left, with subscript 0 denoting the low-order bit.
- The carries are connected in a chain through the full-adders.
- The input carry to the binary adder is C0 and the output carry is C4.
- The S outputs of the full-adders generate the required sum bits.
- An n-bit binary adder requires n full-adders.
- The output carry from each full-adder is connected to the input carry of the next-high-order full-adder.
- The n data bits for the A inputs come from one register (such as R1), and the n data bits for the B inputs come from another register (such as R2). The sum can be transferred to a third register or to one of the source registers (R1 or R2), replacing its previous content.

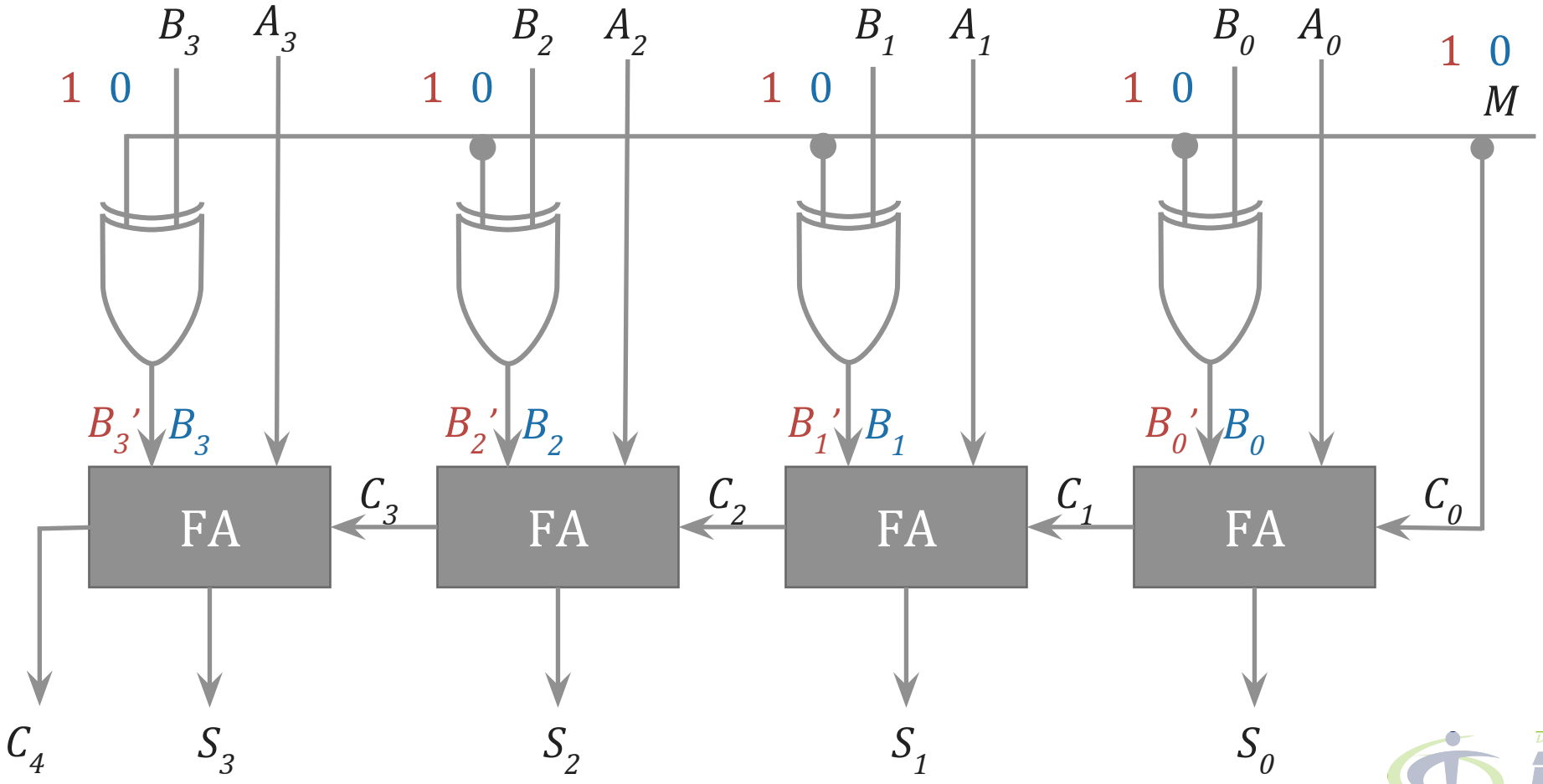
Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4-bit Binary Adder-Subtractor

when *M = 0* the circuit is an *Adder*
when *M = 1* the circuit becomes a *Subtractor*

When *M = 0*, we have
$C_0 = 0$ & $B \oplus 0 = B$

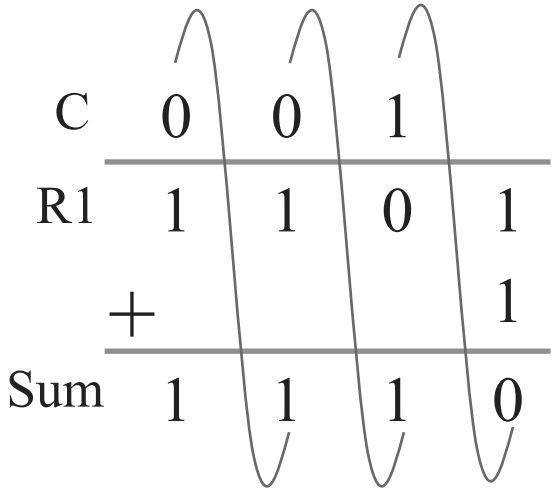
When *M = 1*, we have
$C_0 = 1$ & $B \oplus 1 = B'$

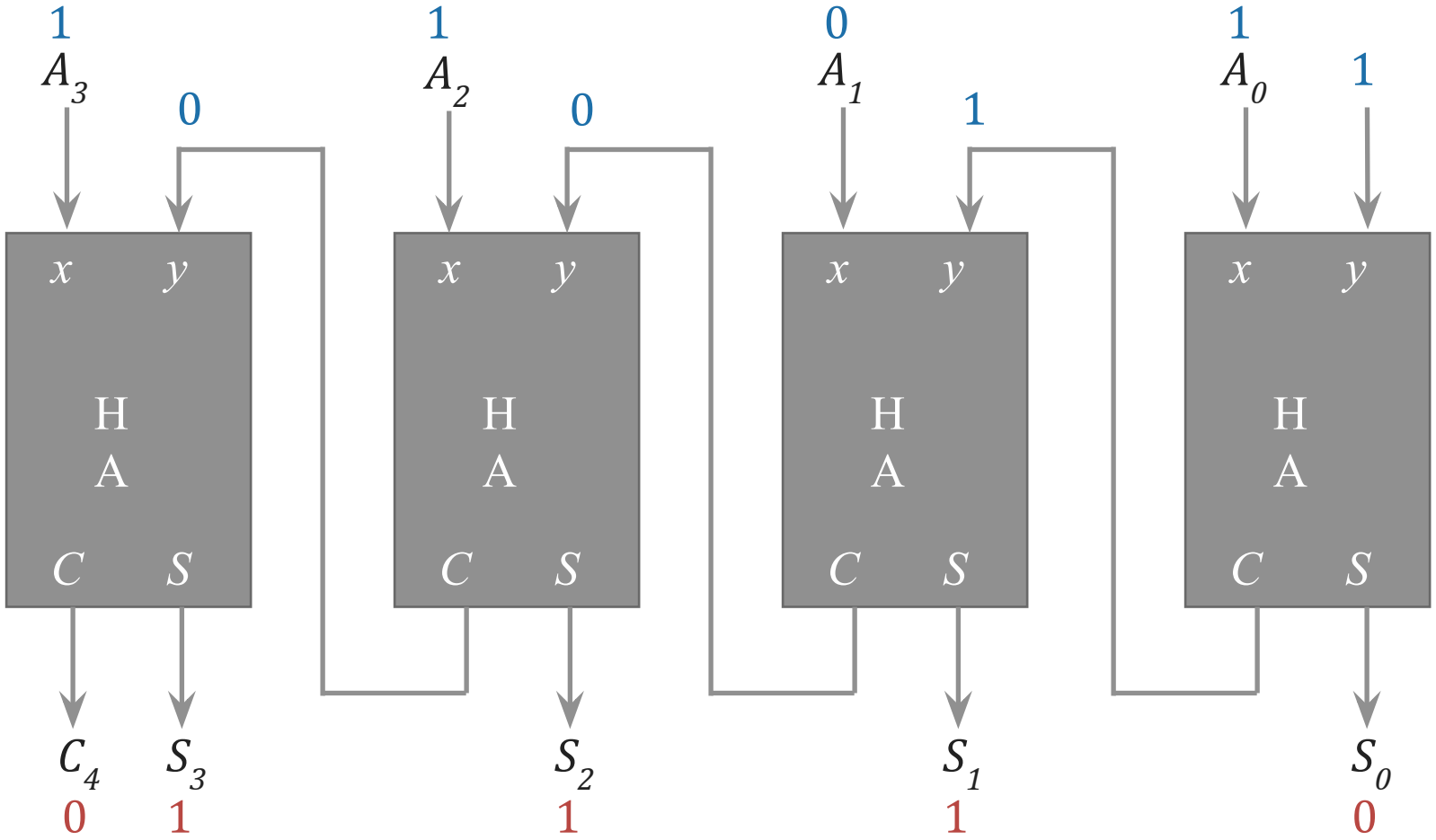


Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4-bit Binary Incrementer

- The increment microoperation adds one to a number in a register.





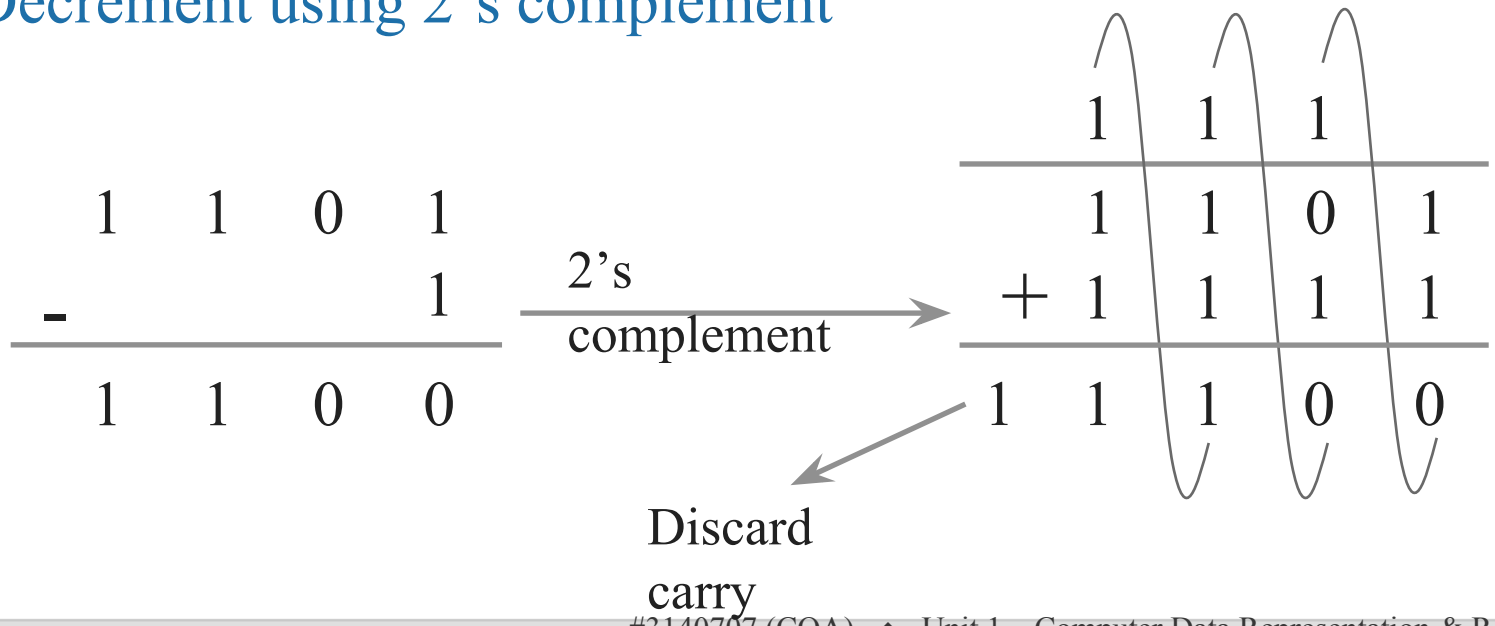
Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4-bit Arithmetic Circuit

- The arithmetic micro operations can be implemented in one composite arithmetic circuit.
- The basic component of an arithmetic circuit is the parallel adder.
- By controlling the data inputs to the adder, it is possible to obtain different types of arithmetic operations.
- The output of binary adder is calculated from arithmetic sum.

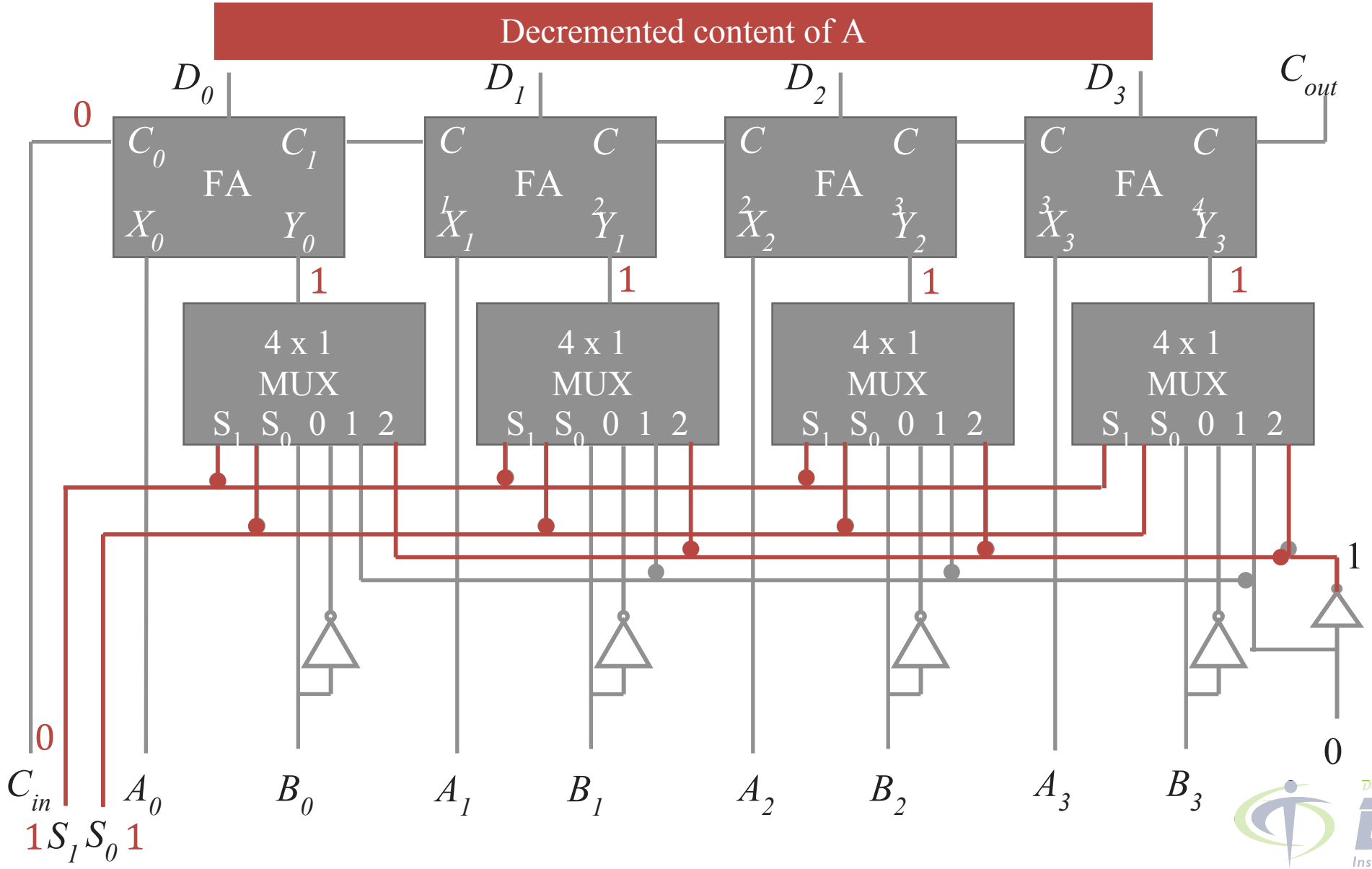
$$D=A+Y+C_{in}$$

Decrement using 2's complement



$$Decrement = A + 1111 + 0$$

4-bit Arithmetic Circuit



Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4-bit Arithmetic Circuit

Hardware implementation consists of:

- 4 full-adder circuits that constitute the 4-bit adder and four multiplexers for choosing different operations.
- There are two 4-bit inputs A and B.
- The four inputs from A go directly to the X inputs of the binary adder. Each of the four inputs from B is connected to the data inputs of the multiplexers. The multiplexer's data inputs also receive the complement of B.
- The other two data inputs are connected to logic-0 and logic-1.
  - Logic-0 is a fixed voltage value (0 volts for TTL integrated circuits)
  - Logic-1 signal can be generated through an inverter whose input is 0.
- The four multiplexers are controlled by two selection inputs, S_1 and S_0.
- The input carry C_{in} goes to the carry input of the FA in the least significant position. The other carries are connected from one stage to the next.
- 4-bit output $D_0 \ldots D_3$

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4-bit Arithmetic Circuit

- When $S_1S_0 = 00$
  - If $C_{in} = 0$ then $D = A + B$; Add
  - If $C_{in} = 1$ then $D = A + B + 1$; Add with carry
- When $S_1S_0 = 01$
  - If $C_{in} = 0$ then $D = A + \overline{B}$; Subtract with borrow
  - If $C_{in} = 1$ then $D = A + \overline{B} + 1$; A + 2's complement of B i.e. A - B
- When $S_1S_0 = 10$
  - Input B is neglected and all 0's are inserted to Y inputs

  $D = A + 0 + C_{in}$
    - If $C_{in} = 0$ then $D = A$; Transfer A
    - If $C_{in} = 1$ then $D = A + 1$; Increment A
- When $S_1S_0 = 11$
  - Input B is neglected and all 1's are inserted to Y inputs

  $D = A - 1 + C_{in}$
    - If $C_{in} = 0$ then $D = A - 1$; 2's compliment
    - If $C_{in} = 1$ then $D = A$; Transfer A

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4-bit Arithmetic Circuit

- Arithmetic Circuit Function

S_1	S_0	C_{in}	Y	$D = A + Y + C_{in}$	Microoperation
0	0	0	B	$D = A + B$	Add
0	0	1	B	$D = A + B + 1$	Add with carry
0	1	0	B'	$D = A + B'$	Subtract with borrow
0	1	1	B'	$D = A + B' + 1$	Subtract
1	0	0	0	$D = A$	Transfer
1	0	1	0	$D = A + 1$	Increment A
1	1	0	1	$D = A - 1$	Decrement A
1	1	1	1	$D = A$	Transfer A

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Logic Micro-operations

Section - 6

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Logic Microoperations

- Logic micro operations specify binary operations for strings of bits stored in registers.
- These operations consider each bit of the register separately and treat them as binary variables.
- Example

$$P: R1 \leftarrow R1 \oplus R2$$

R1		1	0	1	0
R2	$\oplus$	1	1	0	0
R1 after $P = 1$		0	1	1	0

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

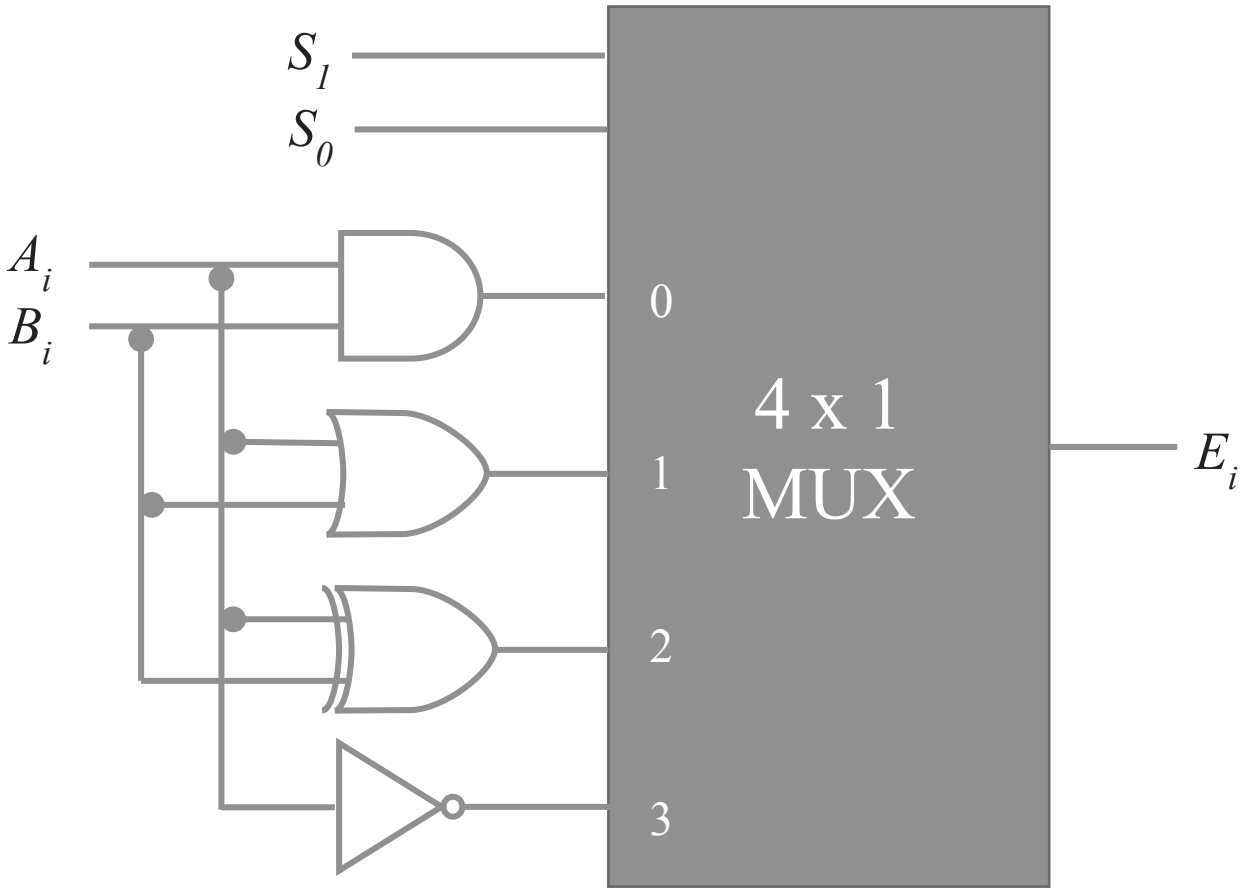
Logic Microoperations

Boolean Function	Microoperation	Name
		Clear
		AND
		Transfer A
		Transfer B
		Exclusive-OR
		OR

Boolean Function	Microoperation	Name
		NOR
		Exclusive-NOR
		Complement B
		Complement A
		NAND
		Set to all 1's

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Hardware Implementation of Logic Circuit



S_1	S_0	Output	Operation
0	0		AND
0	1		OR
1	0		XOR
1	1		Complement

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Applications of Logic Microoperations

1. Selective Set Operation

- The *selective-set* operation sets to 1 the bits in register A where there are corresponding 1's in register B.
- It does not affect bit positions that have 0's in B.
- The OR microoperation can be used to selectively set bits of a register.

1 0 1 0	*A* before
1 1 0 0	*B* (logic operand)
1 1 1 0	*A* after

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Applications of Logic Microoperations

2. Selective Complement Operation

- The *selective-complement* operation complements bits in register A where there are corresponding 1's in register B.
- It does not affect bit positions that have 0's in B.
- The exclusive - OR microoperation can be used to selectively set bits of a register.

1	0	1	0	*A* before
1	1	0	0	*B* (logic operand)
0	1	1	0	*A* after

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Applications of Logic Microoperations

3. Selective Clear Operation

- The *selective-clear* operation clears to 0 the bits in register A only where there are corresponding 1's in register B.
- It does not affect bit positions that have 0's in B.
- The corresponding logic microoperation is $A \leftarrow A \wedge B'$.

1 0 1 0	*A* before
1 1 0 0	*B* (logic operand)
0 0 1 0	*A* after

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Applications of Logic Microoperations

4. Mask Operation

- The *mask* operation is similar to the selective-clear operation except that the bits of register A are cleared only where there are corresponding 0's in register B.
- The mask operation is an AND microoperation.

1	0	1	0	*A* before
1	1	0	0	*B* (logic operand)
1	0	0	0	*A* after

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Applications of Logic Microoperations

5. Insert Operation

- The *insert* operation inserts a new value into a group of bits.
- This is done by first masking and then ORing them with required value.
- The mask operation is an AND microoperation and the insert operation is an OR microoperation.

Mask

A 0 1 1 0 1 0 1 0

B 0 0 0 0 1 1 1 1

A 0 0 0 0 1 0 1 0

Insert

A 0 0 0 0 1 0 1 0

B 1 0 0 1 0 0 0 0

A 1 0 0 1 1 0 1 0

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Applications of Logic Microoperations

6. Clear Operation

- The *clear* operation compares the words in register A and register B and produces an all 0's result if the two numbers are equal.
- This operation is achieved by an exclusive-OR microoperation.

$$
\begin{array}{cccc}
1 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 \\
\hline
0 & 0 & 0 & 0
\end{array}
\qquad
\begin{array}{l}
A \\
B \\
\\
A \leftarrow A \oplus B
\end{array}
$$

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

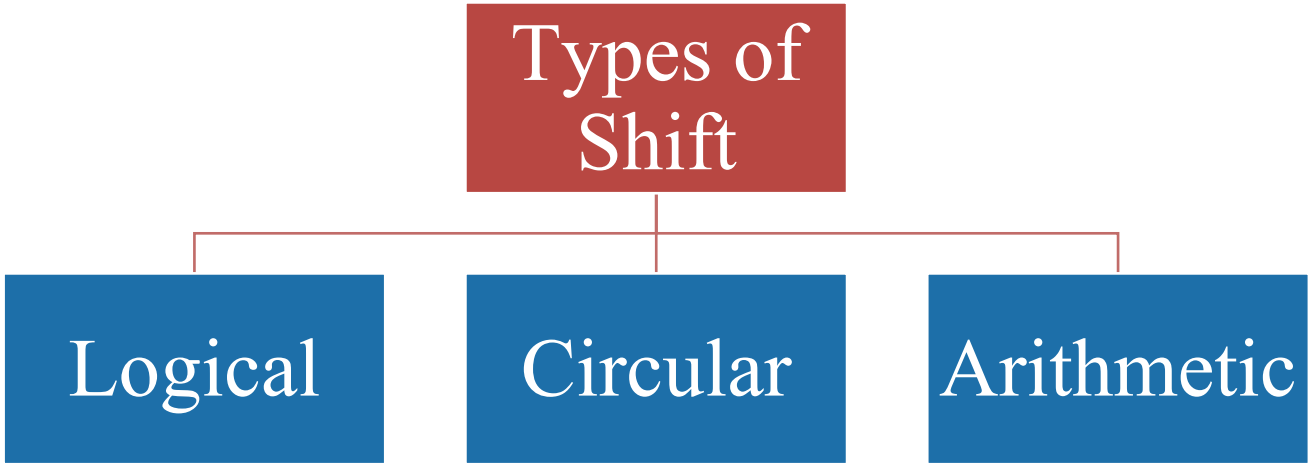
Shift Micro-operations

Section - 7

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Shift Microoperations

- Shift microoperations are used for serial transfer of data.
- Used in conjunction with arithmetic, logic and other data processing operations.
- The content of the register can be shifted to the left or the right.
- The first flip-flop receives its binary information from the serial input.
- The information transferred through the serial input determines the type of shift.



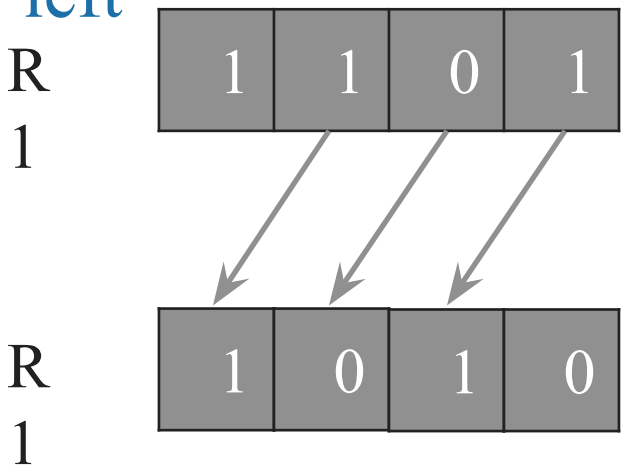
Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Types of Shift

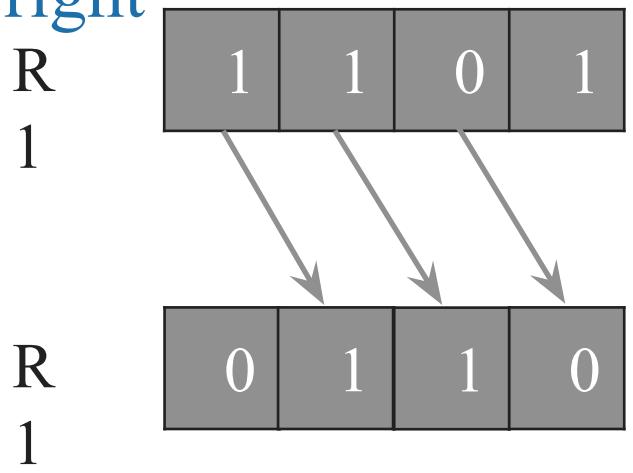
1. Logical Shift

- A *logical shift* is one that transfers 0 through the serial input.

shl - logical shift left



shr - logical shift right



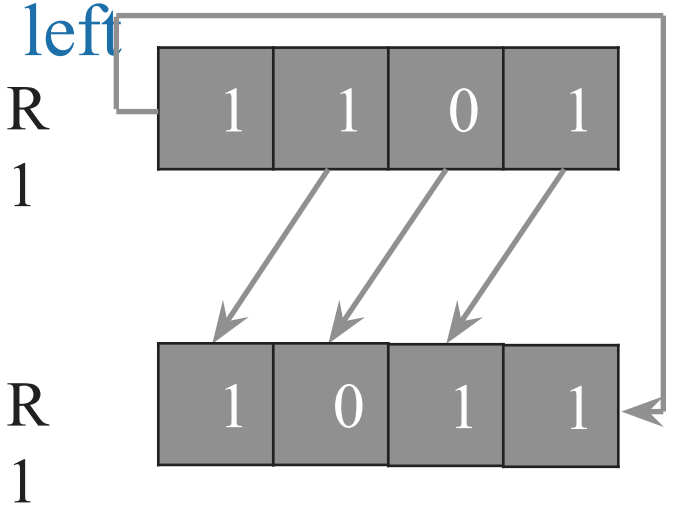
Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Types of Shift

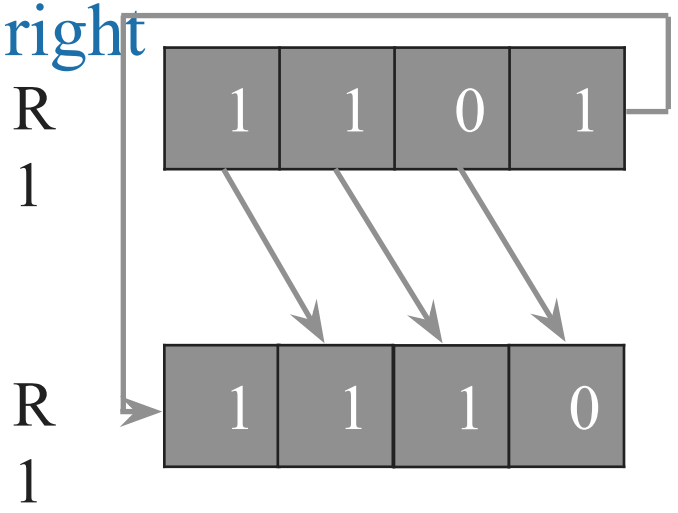
2. Circular Shift

- A *circular shift* (also known as a rotate operation) circulates the bits of the register around the two ends without loss of information.
- This is accomplished by connecting the serial output of the shift register to its serial input.

cil - circular shift left



cir - circular shift right



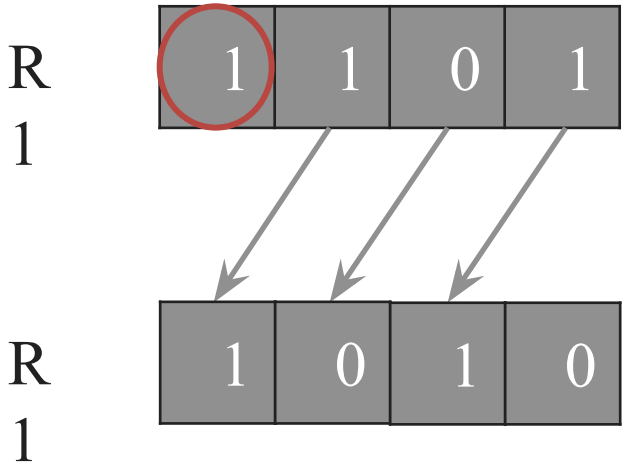
Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Types of Shift

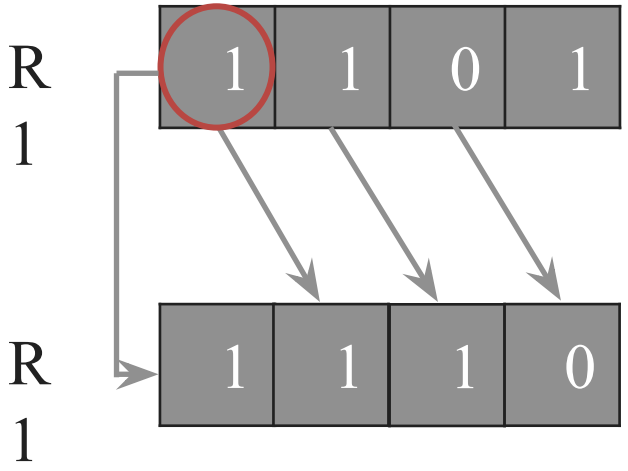
3. Arithmetic Shift

- An *arithmetic shift* is a micro-operation that shifts a signed binary number to the left or right.
- An arithmetic shift-left multiplies a signed binary number by 2.
- An arithmetic shift-right divides the number by 2.

ashl - arithmetic shift left

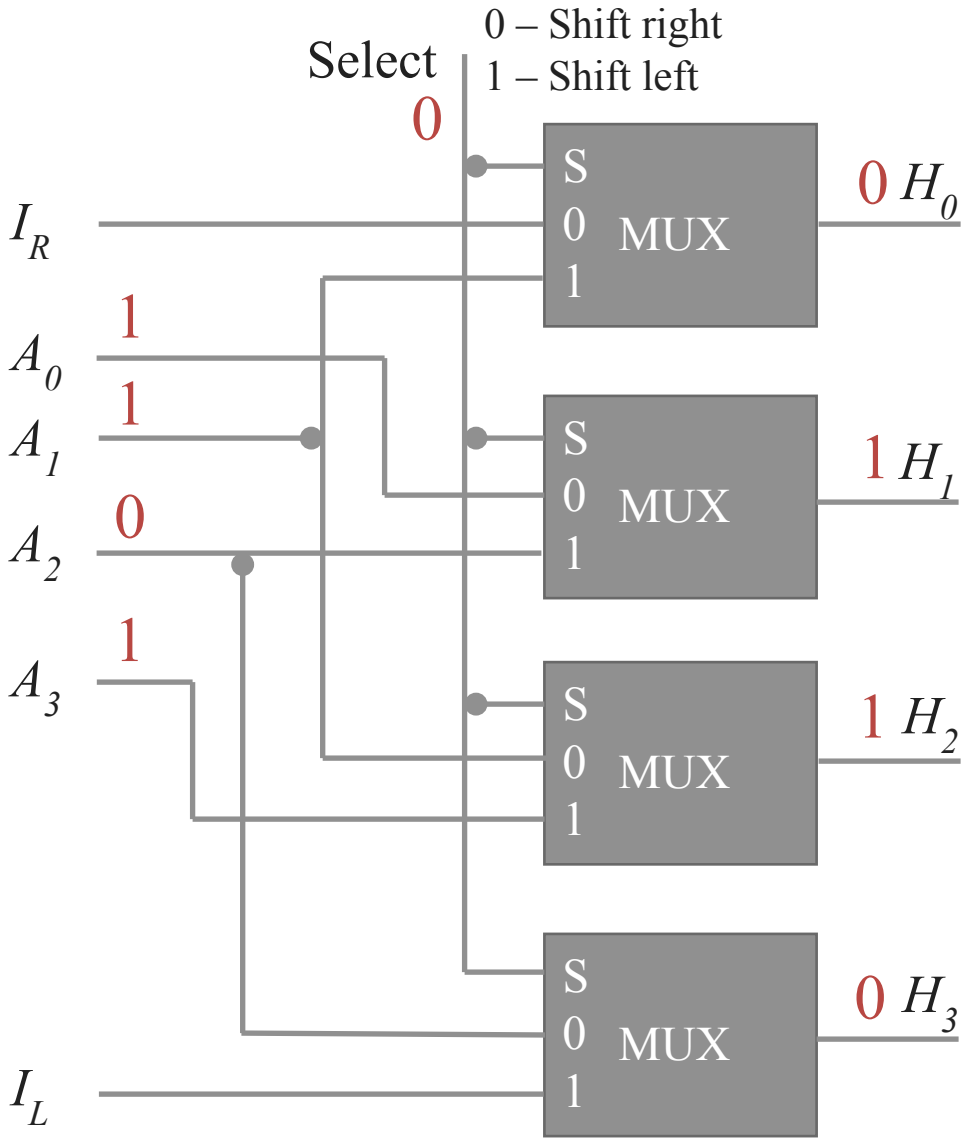


ashr - arithmetic shift right



Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4 - bit Combinational Circuit Shifter



S	H_0	H_1	H_2	H_3
0	I_R	A_0	A_1	A_2
1	A_1	A_2	A_3	I_L

4 - bit Combinational Circuit Shifter

- The 4-bit shifter has four data inputs, A_0 through A_3 and four data outputs, H_0 through H_3.
- There are two serial inputs, one for shift left (I_L) and the other for shift right (I_L).
- When the selection input S = 0, the input data are shifted right (down in the diagram).
- When S = 1, the input data are shifted left (up in the diagram).
- The two serial inputs can be controlled by another multiplexer to provide the three possible types of shifts.

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Arithmetic logical shift unit

Section - 8

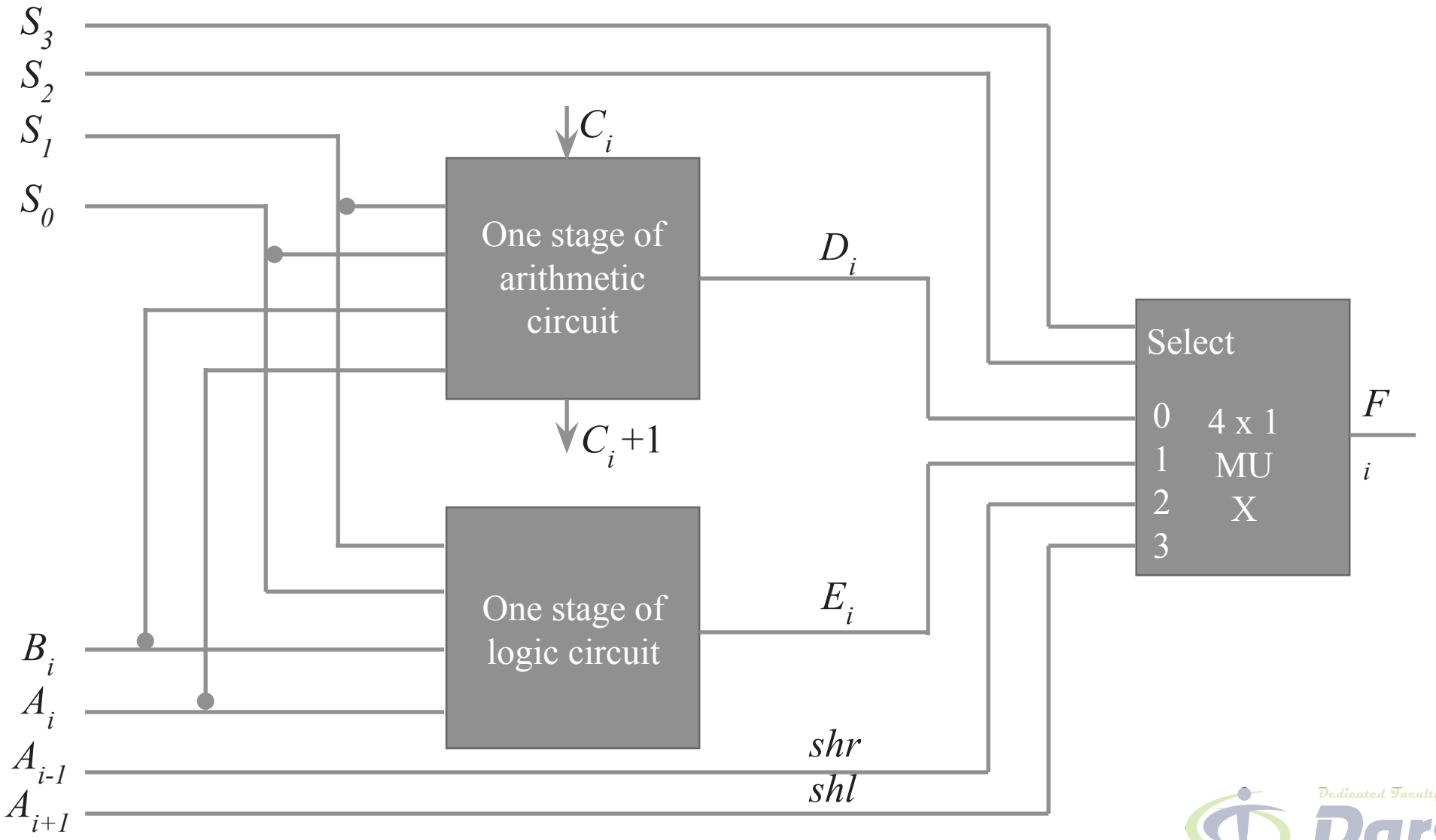
Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4 - bit Arithmetic Logic Shift Unit

- Instead of having individual registers performing the micro operations directly, computer systems employ a number of storage registers connected to a common operational unit called an arithmetic logic unit, abbreviated ALU.
- To perform a microoperation, the contents of specified registers are placed in the inputs of the common ALU.
- The ALU performs an operation and the result of the operation is then transferred to a destination register.
- The arithmetic, logic, and shift circuits introduced in previous sections can be combined into one ALU with common selection variables.

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4 - bit Arithmetic Logic Shift Unit



Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

4 - bit Arithmetic Logic Shift Unit

S_3	S_2	S_1	S_0	C_i	Operation	Function
0	0	0	0	0	$F = A$	Transfer A
0	0	0	0	1	$F = A + 1$	Increment A
0	0	0	1	0	$F = A + B$	Addition
0	0	0	1	1	$F = A + B + 1$	Add with
0	0	1	0	0	$F = A + B'$	Subtract with borrow
0	0	1	0	1	$F = A + B' + 1$	Subtraction
0	0	1	1	0	$F = A - 1$	Decrement

S_3	S_2	S_1	S_0	C_i	Operatio	Function
0	0	1	1	1	$F = A$	Transfer A
0	1	0	0	x		AND
0	1	0	1	x		OR
0	1	1	0	x	$F = A \oplus B$	XOR
0	1	1	1	x	$F = A'$	Complement A
1	0	x	x	x	$F =$ shr A	Shift right A into
1	1	x	x	x	$F =$ shl A	Shift left A into F

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Questions asked in GTU exam

1. What do you mean by register transfer? Explain in detail. Also discuss three-state bus buffer.
2. List and explain types of shift operations on accumulator.
3. Define RTL. Explain how register transfer takes place in basic computer system
4. What is multiplexing? Explain the multiplexing of control signals in ALU.
5. Explain how complement number system is useful in computer system. Discuss any one complement number system with example.
6. Draw the block diagram of 4-bit arithmetic circuit and explain it in detail.
7. Explain shift micro operations and Draw neat and clean diagram for 4-bit combinational circuit shifter.
8. Explain hardware implementation of common bus system using three state buffers. Mention assumptions if required.
9. Explain 4-bit adder-subtractor with diagram.
10. Explain floating point representation.
11. What is a Digital Computer System? Explain the role of binary number system in it.
12. Design a digital circuit for 4-bit binary adder.

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology

Questions asked in GTU exam

13. Represent $(8620)_{10}$ in (1) binary (2) Excess-3 code and (3) 2421 code.
14. Explain selective set, selective complement and selective clear.
15. How negative integer number represented in memory? Explain with suitable example.
16. Explain Micro operation.
17. What does this mean: R2 ← R1?
18. What does this mean: T0: R4 ← R0?
19. What is a Bus?
20. What is an ALU?
21. Represent the following conditional control statement(s) by two register transfer statements with control function. If (P=1) then (R1 ← R2) else if (Q=1) then (R1 ← R3)
22. State true or false: In binary number system, B - A is equivalent to B + A' + 1.
23. Draw a diagram of 4-bit binary incrementer and explain it briefly.

Dedicated Faculty, Committed Education
Darshan
Institute of Engineering & Technology