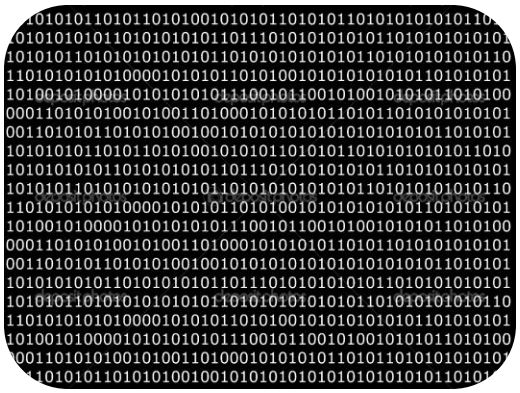
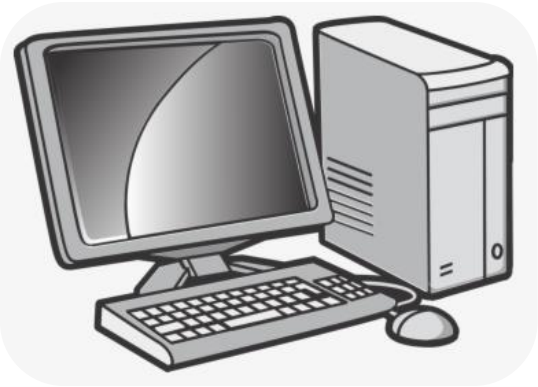
AMIRAJ

COLLEGE OF ENGINEERING & TECHNOLOGY

Unit-1

Computer Data Representation





Subject:- COA **Code:-3140707**	**Prepared by:** **Asst. Prof. S Y JOSHI** **(CSE Department,ACET)**	AMIRAJ COLLEGE OF ENGINEERING & TECHNOLOGY

Topics to be covered

- Basic computer data types
- Complements
- Fixed point representation
- Register Transfer and Micro-operations
- Floating point representation
- Register Transfer language
- Register Transfer
- Bus and Memory Transfers
- Arithmetic Micro-Operations
- Logic Micro-Operations
- Shift Micro-Operations
- Arithmetic logical shift unit.

Basic computer data types

- Binary information in digital computers is stored in memory or processor registers.
- The data types found in the registers of digital computers may be classified as being one of the following categories: (1) numbers used in arithmetic computations, (2) letters of the alphabet used in data processing, and (3) other discrete symbols used for specific purposes. All types of data, except binary numbers, are represented in computer registers in binary-coded form.
- A number system of base or radix *r* is a system of that uses distinct symbols for r digits
- The decimal number system in everyday use employs radix 10 system. The 10 symbols are 0, 1, 2, 3, 4, 5, 6,7, 8, and 9; highest number being *r-1*

Basic computer data types

- The binary number system uses the radix 2. The two digit symbols used are 0 and 1.
- The string of digits 101101 is interpreted to represent

$$1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 45$$

- Besides the decimal and binary number systems,
- Octal (radix 8)- 0,1,2,3,4,5,6,7 and
- Hexadecimal (radix 16) – 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
- Octal can be converted to decimal as follows

$$(736.4)_8$$

$$= 7 \times 8^2 + 3 \times 8^1 + 6 \times 8^0 + 4 \times 8^{-1}$$

$$= 7X64 + 3X8 + 6X1 + 4/8$$

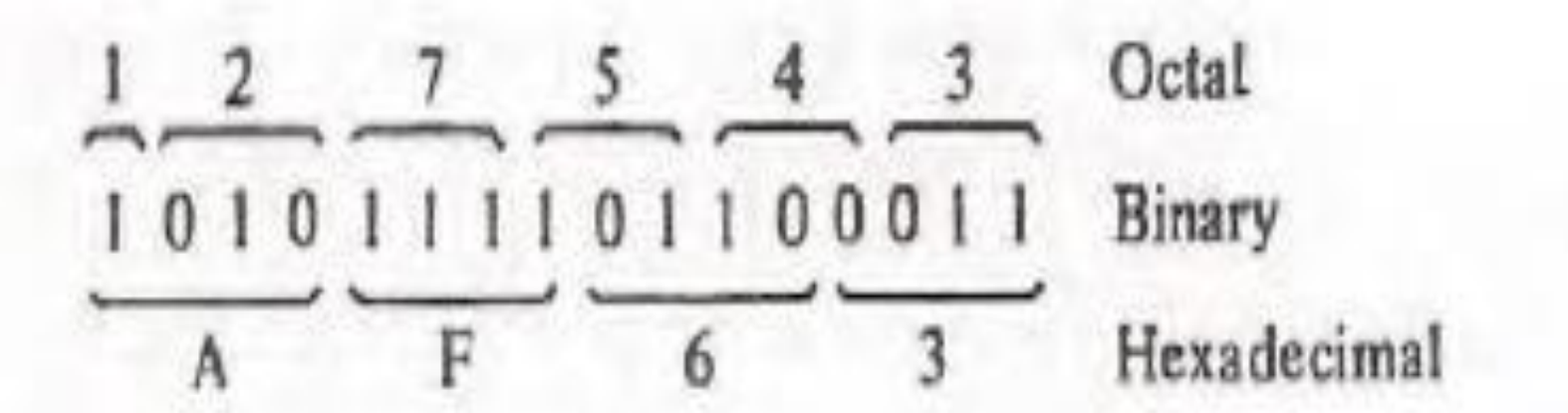
$$= (478.5)_{10}$$

Basic computer data types

- Hexadeciamal can be converted to decimal

$$(F3)_{16} = F \times 16 + 3$$
$$= 15 \times 16 + 3$$
$$= (243)_{10}$$

Octal and Hex can be obtained from Binary as shown below:



Complements

- A binary code is a group of n bits that assume upto 2^n distinct combinations of 0s and 1s.
- A BCD code is a binary coded decimal i.e. binary coding decimal numbers.
- ASCII (American Standard Code for Information Interchange),
- which uses seven bits to code 128 characters is standard alphanumeric character code.
- Complements are used in digital computers for simplifying the subtraction operation and for logical manipulation. There are two types of complements
- For each base r system: the r's complement and the (r - 1)'s complement

Complements

- For binary base 2 system: the 2's complement and the 1's complement
- For decimal base 10 system: the 10's complement and the 9's complement
- The 9's complement of 546700 is

  999999 - 546700 = 453299

- The 9's complement of 12389 is

  99999 - 12389 = 87610

- The 1's complement of a binary number is formed by Changing 1's into 0's and 0's into 1's.
- For example, the 1's complement of 1011001 is 0100110 and the 1's complement of 0001111 is 1110000.

Complements

- The 10's complement of the decimal 2389 is 7610 + 1 = 7611 and is obtained by adding 1 to the 9's complement value.
- The 2's complement of binary 101100 is 010011 + 1 = 010100 and is obtained by adding 1 to the 1's complement value.
- For example, the 1's complement of 1011001 is 0100110 and the 1's complement of 0001111 is 1110000.

- Solve:

Find the 9’s and 10’s complement of 246700.

Find the 1’s and 2’s complement of 1101100.

Fixed-Point Representation

- In computer binary systems it is customary to represent the sign with a bit placed in the leftmost position of the number.
- sign bit is equal to 0 for positive and to 1 for negative.
- a number may have a binary (or decimal) point.
- There are two ways of specifying the position of the binary point: by giving it a fixed position or by employing a floating-point representation.
- The fixed-point method assumes that the binary point is always. fixed in one position.
- The two positions most widely used are (1) a binary point in the extreme left of the register to make the stored number a fraction, and (2) a binary point in the extreme right of the register to make the stored number an integer. In either case, the binary point is not actually present.

Fixed-Point Representation

Integer Representation for signed numbers

- signed-magnitude representation 1 0001110
- signed-1's complement representation 1 1110001
- signed-2's complement representation 1 1110010

Floating Point Representation

- The floating-point representation uses a second register to store a number that designates the position of the decimal point in the first register.
- The floating-point representation of a number has two parts. The first part represents a signed, fixed-point number called the mantissa. The second part designates the position of the decimal (or binary) point and is called the exponent The fixed-point mantissa may be a fraction or an integer. For example,
- the decimal number +6132.789 is represented in floating-point with a fraction and an exponent as follows:

Fraction	*Exponent*
+0.6132789	*+04*

Floating Point Representation

- A floating-point binary number is represented in a similar manner except that it uses base 2 for the exponent.

- For example, the binary number +1001.11 is represented with an 8-bit fraction and 6-bit exponent as follows:

Fraction	Exponent
01001110	000100

Some other codes

- Gray Codes
- BCD 8421
- 2421
- Excess 3 codes
- Other alphanumeric codes like EBCDIC (Extended BCD Interchange Code)
- Error detecting/ Correcting codes-Parity
- Even parity odd parity

Register Transfer language

- A digital system is an interconnection of digital hardware modules that accomplish a specific information-processing task.
- The modules are constructed from registers, decoders, arithmetic elements, and control logic
- Digital modules are best defined by the registers they contain and the operations that are performed on the data stored in them.
- The operations executed on data stored in registers are called **micro-operations**.
- A **micro-operation** is an elementary operation performed on the information stored in one or more registers.

Register Transfer language

- The symbolic notation used to describe the micro-operation 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 micro-operation and transfer the result of the operation to the same or another register.
- Computer registers are designated by capital letters sometimes followed by numerals) to denote the function of the register.
- MAR Memory Address Register
- PC Program Counter

Register Transfer language

- **Register Transfer**

$$R2 \leftarrow R1$$

- denotes a transfer of the content of register R1 into register R2.
- In RTL, To show an if condition
- (where P is a control signal generated in the control section)

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

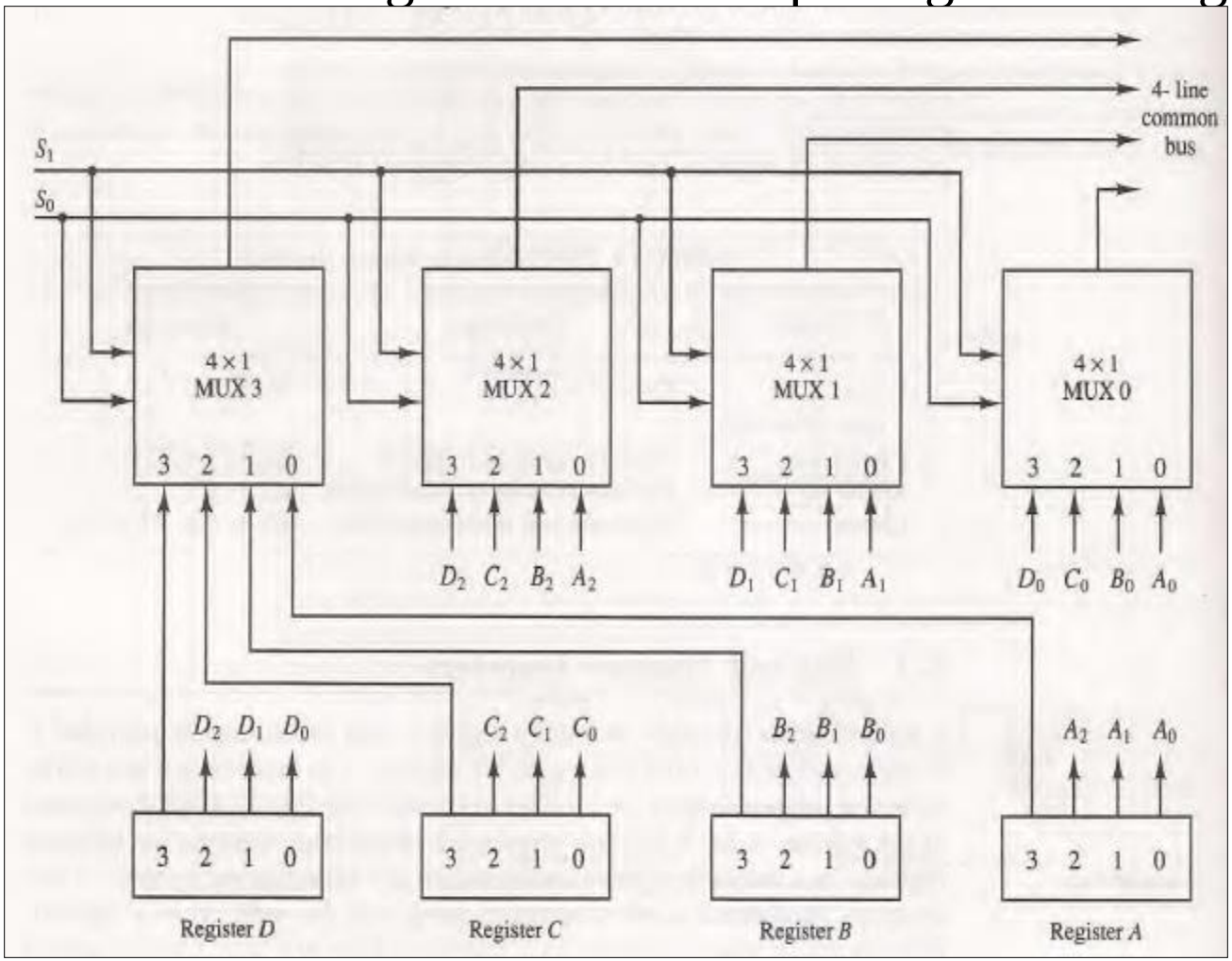
Can be shown in RTL as

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

- Generally a comma is used to separate two or more operations executed at the same time
- $T \ : \ R2 \leftarrow R1 \ , R1 \leftarrow R2$

Bus Transfers

- Common Bus System is a scheme for transferring information between registers in a multiple register configuration.

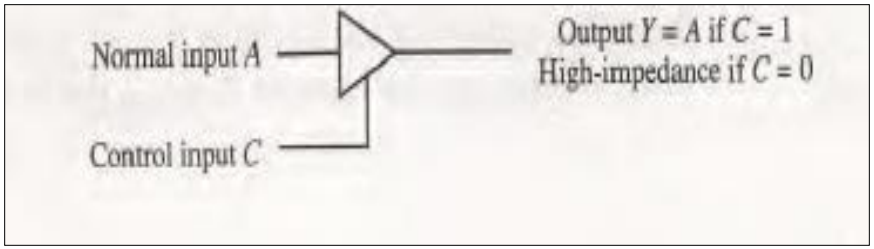


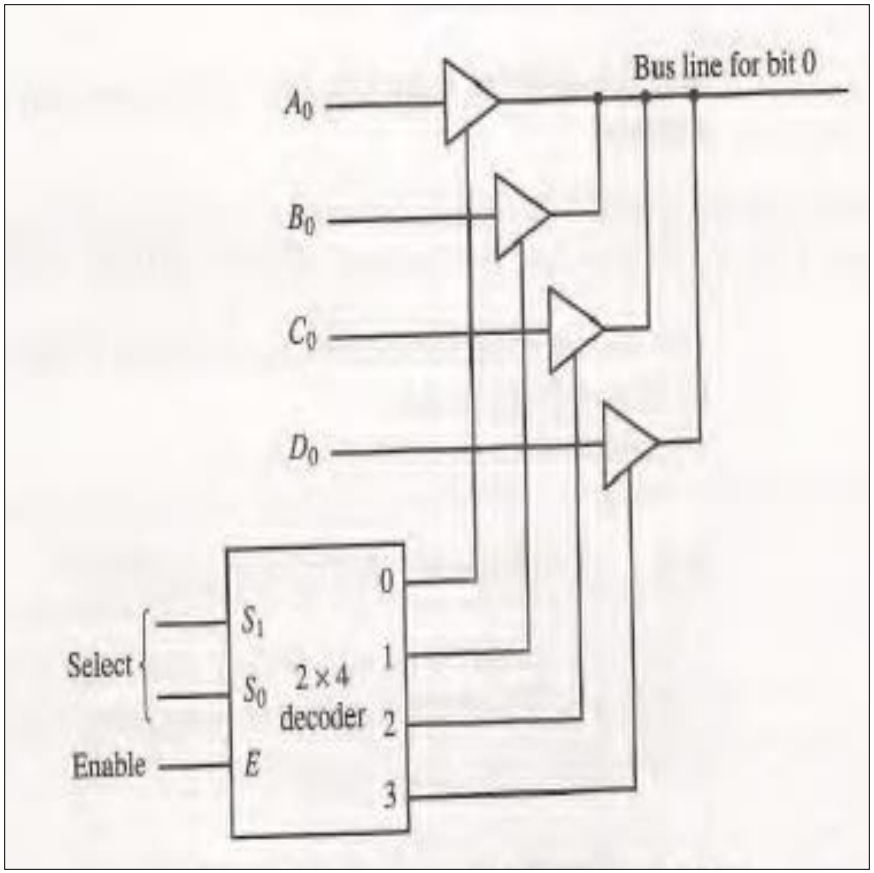
S_1	S_0	Register selected
0	0	A
0	1	B
1	0	C
1	1	D

A bus structure consists of common lines, can be constructed with the help of multiplexers

AMIRAJ COLLEGE OF ENGINEERING & TECHNOLOGY

Bus Transfers





- A bus system can also be constructed with three-state gates. 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 a high-impedance state. The high-impedance state behaves like an open circuit, which means that the output is disconnected and does not have a logic significance.

Memory Transfers

- The transfer of information from a memory word to the outside environment is called a read operation.
- The transfer of new information to be stored into the memory is called a write operation.
- A memory word will be symbolized by the letter M.
- Consider a memory unit that receives the address from a register, called the address register, symbolized by AR. The data are transferred to another register, called the data register, symbolized by DR.
- The read operation can be stated as follows:

  Read: DR <- M[AR]

- This causes a transfer of information into DR from the memory word M selected by the address in AR

Arithmetic Micro-Operations

- Arithmetic micro operations specify the addition subtraction or complementing of the string of bits stored in registers.

- Some of them are :

R3 ← R1 + R2	Contents of R1 plus R2 transferred to R3
R3 ← R1 - R2	Contents of R1 minus R2 transferred to R3
R2 ← R2’	Complement the contents of R2 (1's)
R2 ← R2’ + 1	2's complement the contents of R2 (negate)
R3 ← R1 + R2’ + 1	R1 plus the 2's complement of R2(subtraction)
R1 ← R1 + 1	Increment the contents of R1 by one
R1 ← R1 – 1	Decrement the contents of R1 by one

Logic Micro-Operations

- Logic micro operations specify binary operations for strings of bits stored in registers. There are 16 different logic operations that can be performed with two binary variables.

Boolean function	Microoperation	Name
$F_0 = 0$	$F \leftarrow 0$	Clear
$F_1 = xy$	$F \leftarrow A \wedge B$	AND
$F_2 = xy'$	$F \leftarrow A \wedge \overline{B}$	
$F_3 = x$	$F \leftarrow A$	Transfer A
$F_4 = x'y$	$F \leftarrow \overline{A} \wedge B$	
$F_5 = y$	$F \leftarrow B$	Transfer B
$F_6 = x \oplus y$	$F \leftarrow A \oplus B$	Exclusive-OR
$F_7 = x + y$	$F \leftarrow A \vee B$	OR
$F_8 = (x + y)'$	$F \leftarrow \overline{A \vee B}$	NOR
$F_9 = (x \oplus y)'$	$F \leftarrow \overline{A \oplus B}$	Exclusive-NOR
$F_{10} = y'$	$F \leftarrow \overline{B}$	Complement B
$F_{11} = x + y'$	$F \leftarrow A \vee \overline{B}$	
$F_{12} = x'$	$F \leftarrow \overline{A}$	Complement A
$F_{13} = x' + y$	$F \leftarrow \overline{A} \vee B$	
$F_{14} = (xy)'$	$F \leftarrow \overline{A \wedge B}$	NAND
$F_{15} = 1$	$F \leftarrow$ all 1's	Set to all 1's

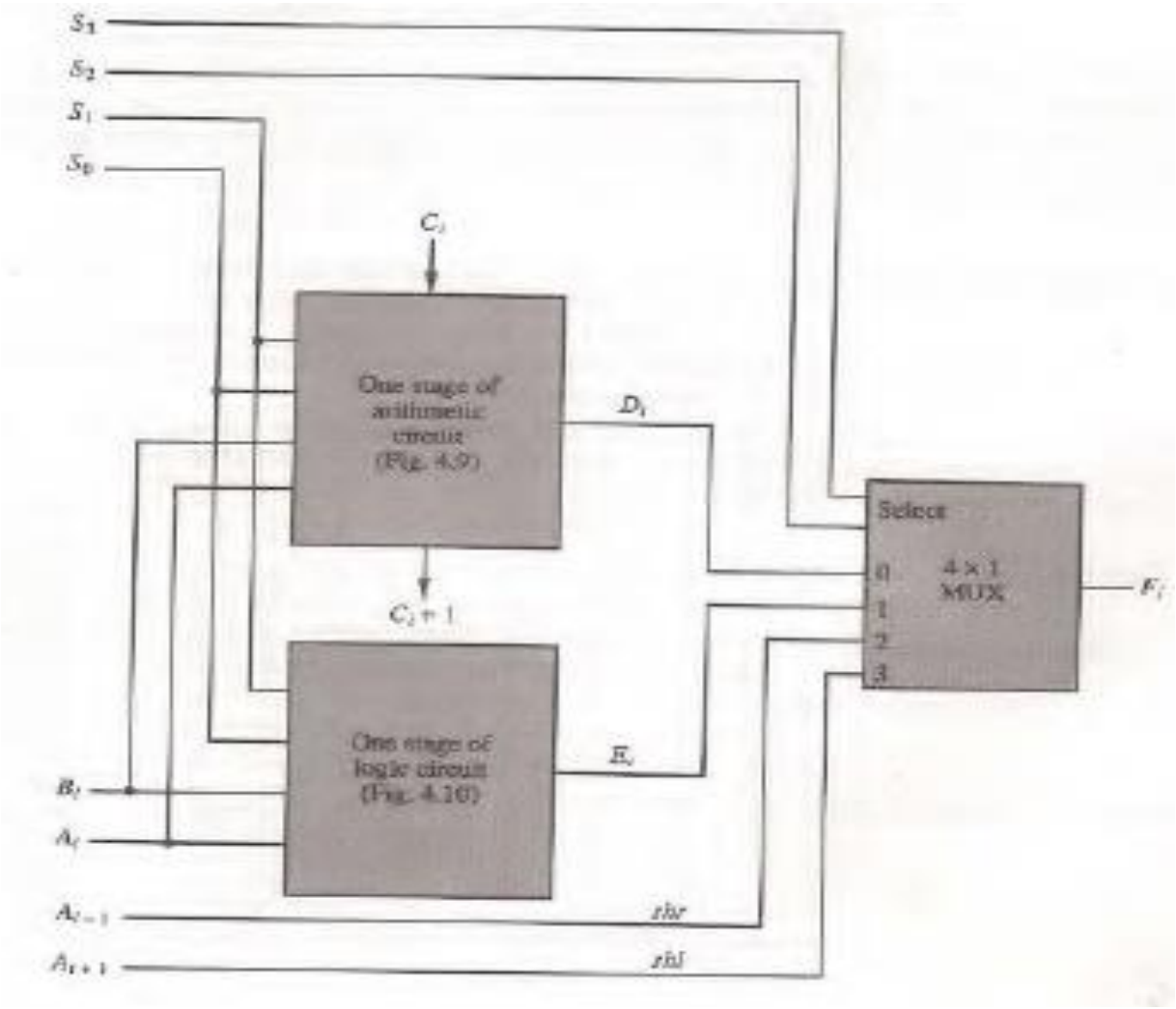
Shift Micro-Operations

- Shift micro operations are used for serial transfer of data
- also used with arithmetic, logic, and other data-processing operations.

Symbolic designation	**Description**
R ← *shl* R	Shift-left register R
R ← *shr* R	Shift-right register R
R ← *cil* R	Circular shift-left register R
R ← *cir* R	Circular shift-right register R
R ← a*shl* R	Arithmetic shift-left R
R ← a*shr* R	Arithmetic shift-right R

Arithmetic Logic Shift Unit

- Computer systems employ a number of storage registers connected to a common operational unit called an arithmetic logic unit, abbreviated **ALU.**



AMIRAJ COLLEGE OF ENGINEERING & TECHNOLOGY

Arithmetic Logic Shift Unit

- Table lists the 14 operations of the ALU.
- The first 8 are arithmetic and next four logic, last two are shift

Operation select						
S_3	S_2	S_1	S_0	C_{in}	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 carry
0	0	1	0	0	$F = A + \overline{B}$	Subtract with borrow
0	0	1	0	1	$F = A + \overline{B} + 1$	Subtraction
0	0	1	1	0	$F = A - 1$	Decrement A
0	0	1	1	1	$F = A$	Transfer A
0	1	0	0	x	$F = A \wedge B$	AND
0	1	0	1	x	$F = A \vee B$	OR
0	1	1	0	x	$F = A \oplus B$	XOR
0	1	1	1	x	$F = \overline{A}$	Complement A
1	0	x	x	x	$F =$ shr A	Shift right A into F
1	1	x	x	x	$F =$ shl A	Shift left A into F

References

- **Images , descriptive Tables , from Computer System Architecture, Morris Mano, 3rd edition Prentice Hall**

- **Note: These pdf/ppt notes are for purpose of teaching aids to classroom/online sessions study, and in no case imply for GTU syllabus or GTU exam. For GTU syllabus or exam related preparation, one may, however will need to attend college/online lectures and refer books given by GTU in their official syllabus.**