Module 2

Register transfer logic: inter register transfer — arithmetic, logic and shift micro operations. Processor logic design: - processor organization — Arithmetic logic unit - design of arithmetic circuit - design of logic circuit - Design of arithmetic logic unit - status register —design of shifter - processor unit — design of accumulator.

1) Explain different type of micro-operations?

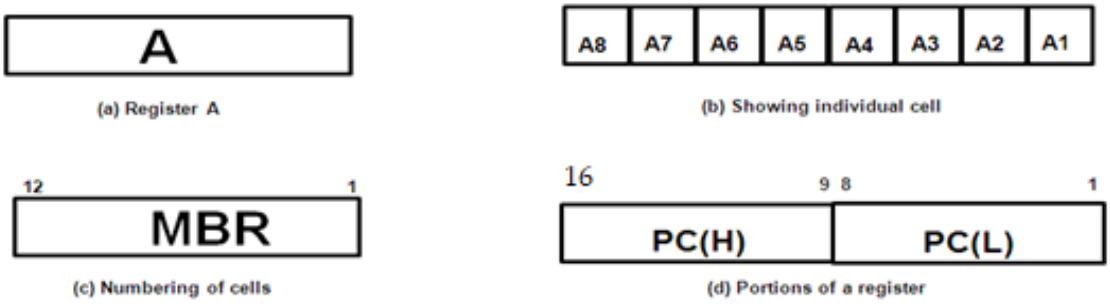
- Microoperations are the basic operations that can be performed by system on data stored in registers.
- Microoperations are used by processors to complete their tasks to transfer data from one place to another.

Types of Microoperations

1. Inter register transfer micro-operations
2. Arithmetic micro-operations
3. Logic micro-operations
4. Shift micro-operations

1)inter register transfer micro-operations

- These micro-operations do not change the information content when the binary information moves from one register to another.
- The registers in a digital system are designated by capital letters. Examples A, B, R1, R2 etc.
- The register that holds the address of the memory unit is called Memory Address Register (MAR). The cells of an n bit register are numbered in sequence from 1 to n, starting either from the left or from the right.
- A register can be represented in 4 ways which is shown in the figure.

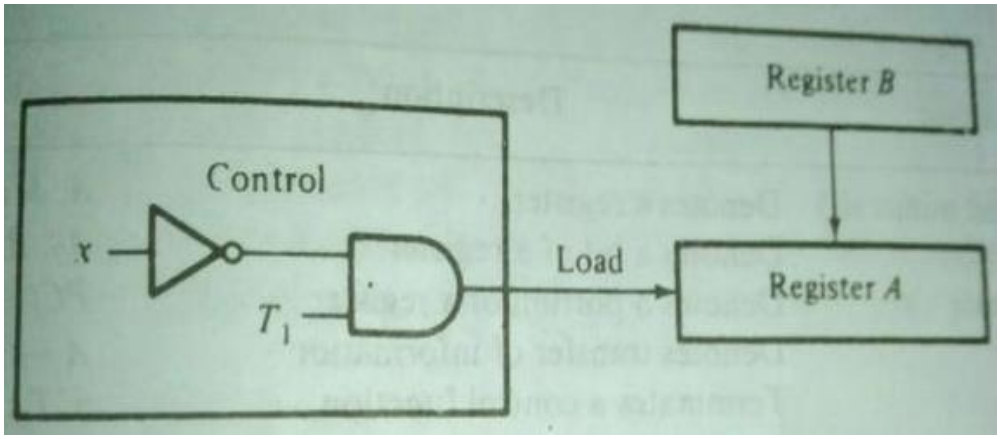


1. Figure (a) shows the most common way to represent a register is with a rectangular box in which name of the register is specified within the box.
2. In figure (b), it shows the individual cells of a register is assigned a letter with a subscript number and is marked from right to left.
3. The numbering of cells from right to left can be marked on top of the box as in figure (c).
4. A 16 bit register is partitioned into two parts, one high order part (H) consisting of eight high ordered cells and one lower order part (L) consisting of eight low ordered cells as in figure (d).

Conditional transfer

- The condition that determines when the transfer is to occur is called a control function. A control function is a Boolean function that can be equal to 0 or 1. The control function can be specified with the following statement:

  x'T1: A←B
- It means that the transfer operation be executed by the hardware only when the Boolean function x'T1=1.That is, variable x=0 and timing variable T1=1.
- Example: Hardware implementation of a controlled transfer x'T1: A←B is shown in figure. Here A←B transfers the contents of register B to register A. Contents of source register B do not change after the transfer.



- The basic symbols of the register transfer logic are listed in the following table.

Symbol	Description	Examples
Letters(and numerals)	Denotes a register	A,MBR,R2
Subscript	Denotes a bit of register	A_2,B_6
Parenthesis ( )	Denotes a portion of a register	PC(H),MBR(OP)
Arrow ←	Denotes transfer of information	A←B
Colon :	Terminates a control function	$x'T_1$:
Comma ,	Separates two micro operations	A←B,B←A
Square brackets [ ]	Specifies an address for memory transfer	MBR←M[MAR]

2.Arithmetic Micro operations

- The basic arithmetic micro-Operations are add, subtract, complement, and shift.
- Add operation- It can be specified by the statement: F←A + B That is, the contents of register ‘A’ is added to the contents of register B and the sum

is transferred to register F. Here we require 3 registers and one digital function that addition operation such as parallel adder.

- The other basic arithmetic operations the following table.

Symbolic designation	Description
$F \leftarrow A+B$	Contents of A plus B transferred to F
$F \leftarrow A-B$	Contents of A minus B transferred to F
$B \leftarrow \overline{B}$	Complement register B(1's complement)
$B \leftarrow \overline{B}+1$	Form the 2's complement of contents of register B
$F \leftarrow A+\overline{B}+1$	A plus 2's complement of B transferred to F
$A \leftarrow A+1$	Increment the contents of A by 1(count up)
$A \leftarrow A-1$	Decrement the contents of A by 1(count down)

- Arithmetic subtraction is most often implemented through complementation and addition. That is,

$F \leftarrow A+B'+1$

- Adding 1 to the 1's complement of B gives the 2's complement of B. When adding 2's complement of B to A it will results in minus operation (A-B).
- Increment and decrement micro-operations are implemented with an up counter and down counter.
- There must be a direct relationship between the statements written in a register transfer language and the registers and digital functions which are required for their implementation.

3. Logic Micro Operations

- Logic micro-operations specify binary operations for a string of bits stored in registers. These operations consider each bit in the registers separately and treat it as a binary variable.
- For example, consider the AND operation of A and B. Content of A is 1101 and content of B is 1010. Then,

  1101(A)

  1010(B)

  A AND B will be 1000 (it will contain in the register F)
- There are 16 different logic operations can be performed with two binary variables, as described in the below table.
- All 16 logic operations can be expressed in terms of the AND, OR and complement operations. Special symbols are adopted for these three micro operations.

Boolean Functions	Operator symbol	Name	Comments
$F_0=0$		Null	Binary constant 0
$F_1=xy$	x.y	AND	x and y
$F_2=xy'$	x/y	Inhibition	x but not y
$F_3=x$		Transfer	X
$F_4=x'y$	y/x	Inhibition	y but not x
$F_5=y$		Transfer	Y
$F_6=xy'+x'y$	$x \oplus y$	Exclusive-OR	x or y but not both
$F_7= x + y$	x + y	OR	x or y
$F_8= (x + y)'$	x↓y	NOR	Not-OR
$F_9= xy +x'y'$	x Oy	Equivalence	x equals y
$F_{10}=y'$	y'	Complement	Not y
$F_{11} = x + y'$	x⊂y	Implication	If y then x
$F_{12}= x'$	x'	Complement	Not x
$F_{13} = x' + y$	x⊃y	Implication	If x then y
$F_{14} = (xy)'$	x↑y	NAND	Not-AND
$F_{15} = 1$		Identity	Binary constant 1

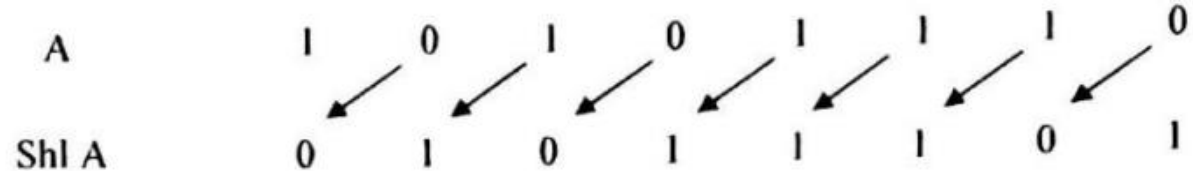
4.Shift Micro Operations

- Shift micro-operations transfer binary information between registers in serial computers. They are also used in parallel computers for arithmetic, logic and control operations
- Registers can be shifted to the left or right. There are no conventional symbols for shift. Here we are adopting the symbols shl and shr for shift left and shift right operations respectively.

A←shl A (1 bit shift to the left of register A)

B←shr B (1 bit shift to the right of register B)

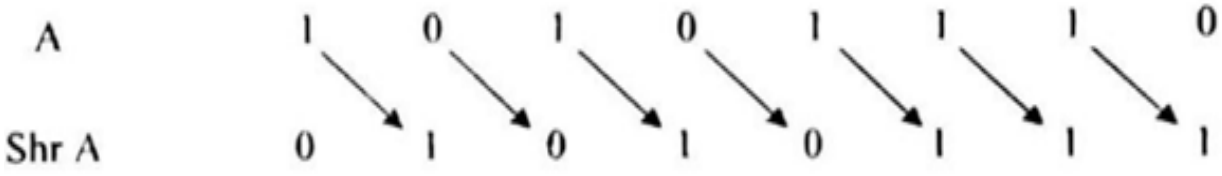
Example 1: Shl A



- Here when we are shifting to left the bit at position A0 is vacant. This gap can be filled by transferring the bit of position An in the original register A. This can be considered as a circular shift.

  A←Shl A, A0 ←An

Example 2: Shr A

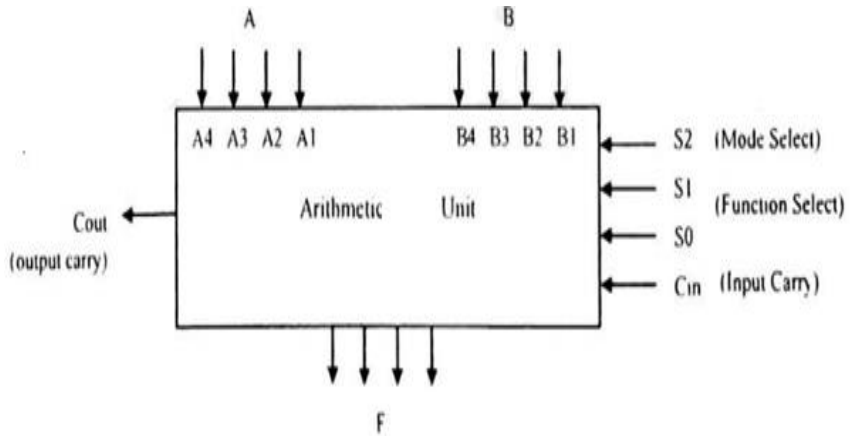


- When we are shifting the register content to the right, the bit at position An is vacant. This gap can be tilled by receiving the value of the 1 bit register E.

A←Shr A, An ←E

2) Design an arithmetic logic unit (ALU unit)?

- An ALU is a multioperation, combinational logic digital function. It can perform a set of basic arithmetic operations and a set of logic operations.
- The ALU has number of selection lines to select a particular operation in the unit. The selection lines are decoded within the ALU.

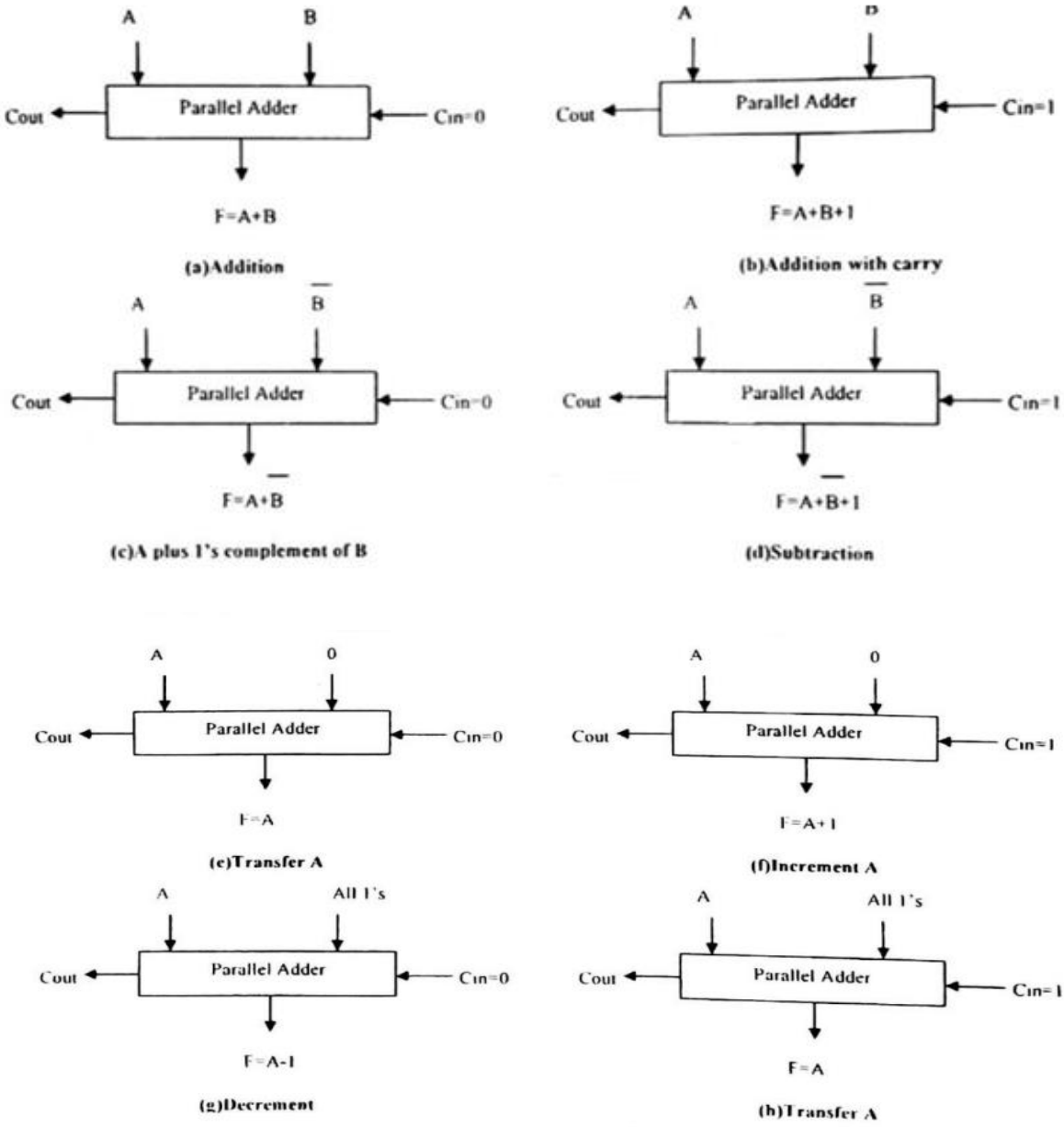


- The four data inputs from A are combined with the four inputs B to generate an operation at the F outputs.
- The mode select input S2 distinguishes between arithmetic and logic operations. The two function select inputs S1 and S0 specify the particular arithmetic or logic operation to be generated.
- With three selection variables it is possible to specify four arithmetic operations and four logical operations.
- The input and output carries have meaning only during an arithmetic operation. The design of ALU can be carried out in 3 stages:

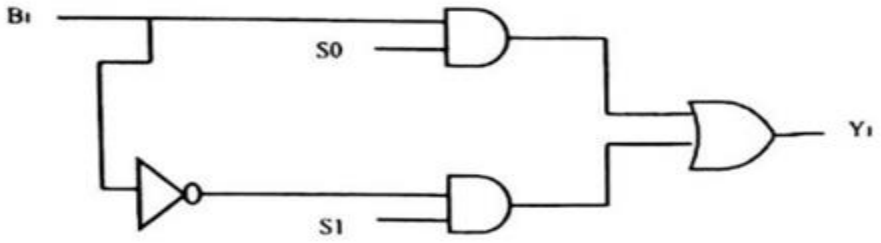
1. Design of Arithmetic section
2. Design of Logic section
3. Modification of arithmetic section so that it can perform both arithmetic and logic operations.

Q) Design arithmetic Circuit?

- The basic component of arithmetic section of an ALU is a parallel adder.
- Parallel adder is constructed with a number of full adder circuits connected in cascade. By combining the data inputs to the parallel adder, it is possible to obtain different types of arithmetic operations.
- The following figure demonstrates the arithmetic operations obtained when one set of inputs to the parallel adder is controlled externally. The no of bits in the parallel adder may be of any value.

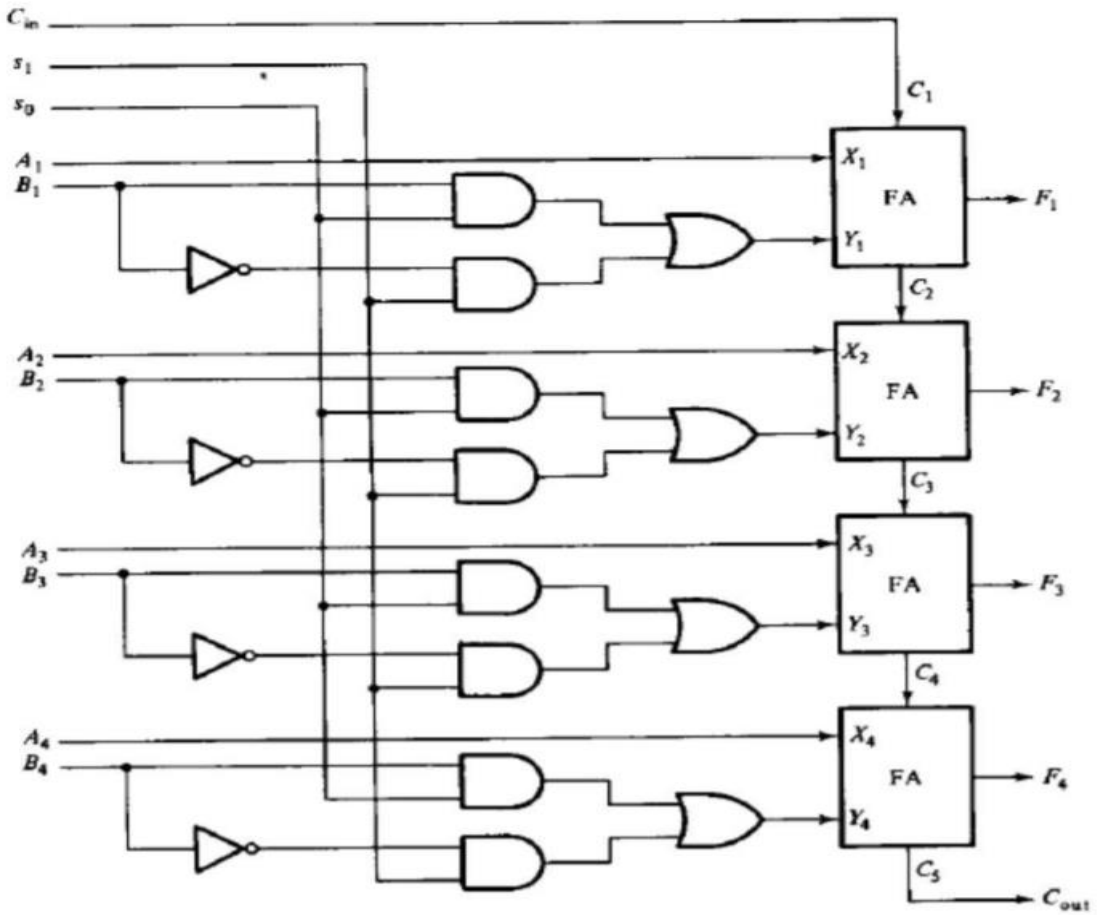


- By changing the B input and Cin, we get 8 operations. So the input B is applied in four different form by using following circuit. The circuit that controls input B to provide the function is called a true/complement, one/zero element. This circuit can be shown as follows.



S1	S0	Yi
0	0	0
0	1	Bi
1	0	$\overline{Bi}$
1	1	1

- The A input is applied directly to the 4-bit parallel adder and the B input is modified. The resultant arithmetic circuit is shown in below figure.

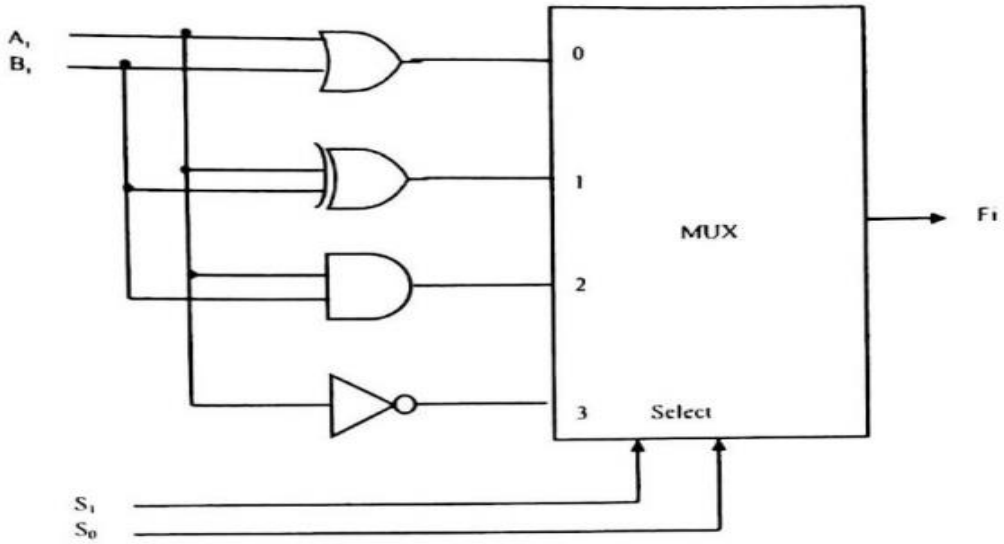


- The functional table for full adder circuit is shown in figure.

Function select			Y equals	Output equals	Function
S1	S0	Cin			
0	0	0	0	F=A	Transfer A
0	0	1	0	F=A+1	Increment A
0	1	0	B	F=A+B	Add B to A
0	1	1	B	F=A+B+1	Add B to A plus 1
1	0	0	B’	F=A+B’	Add 1’s complement of B to A
1	0	1	B’	F=A+B’+1	Add 2’s complement of B to A
1	1	0	All 1’s	F=A-1	Decrement A
1	1	1	All 1’s	F=A	Transfer A

Q) Design Logic circuit?

- The logic microoperation manipulates bits of operands separately and treat each bit as a binary variable. The 16-logic operation can be generated in one circuit and selected by means of four selection lines.

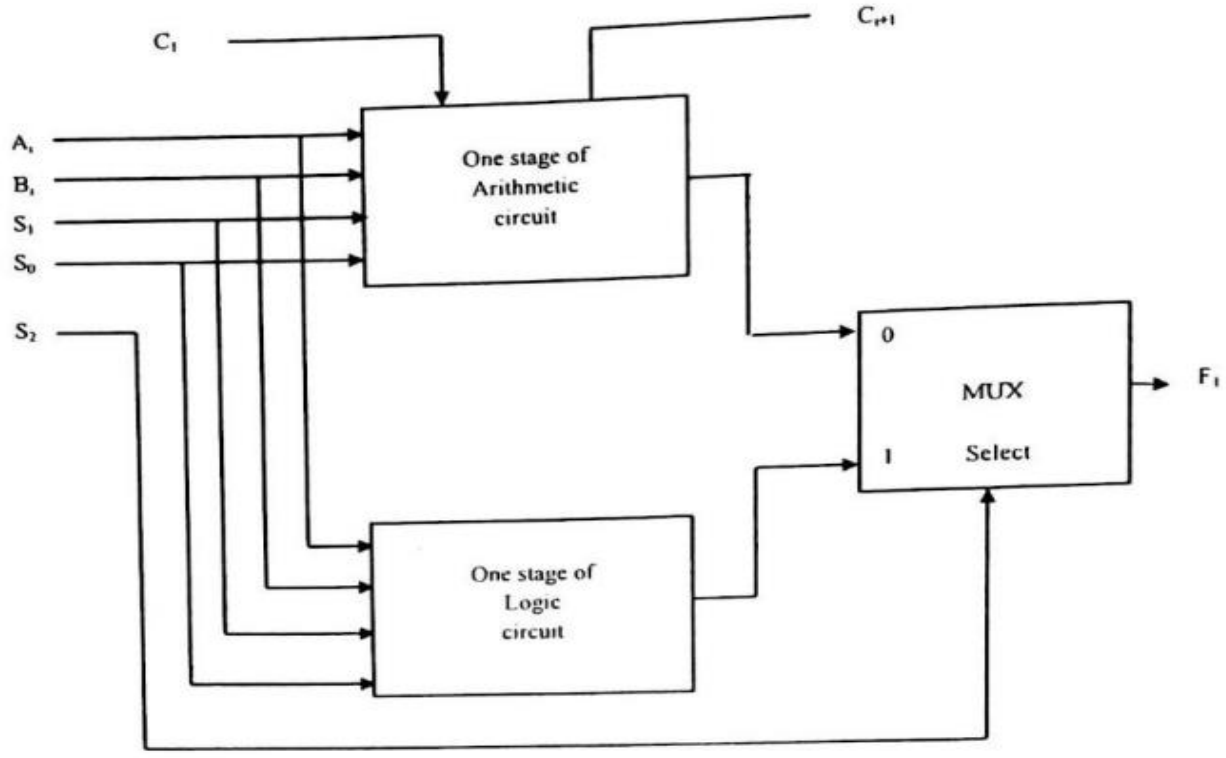


- All logic operations can be obtained by means of AND, OR and NOT operations.
- For three operations, we need two selection variables. But, with two selection variables, we can select four operations. So we can include one more operation XOR in our design.

- The above diagram shows design of a logic circuit. In this one typical stage designated by subscript i. The circuit must be repeated n times for an n bit logic circuit.
- The four gates generate the four logic operations OR, XOR, AND and NOT. The two selection variables in the multiplexer select one of the gates for the output.
- The function table lists the output logic generated as a function of two selection variables.

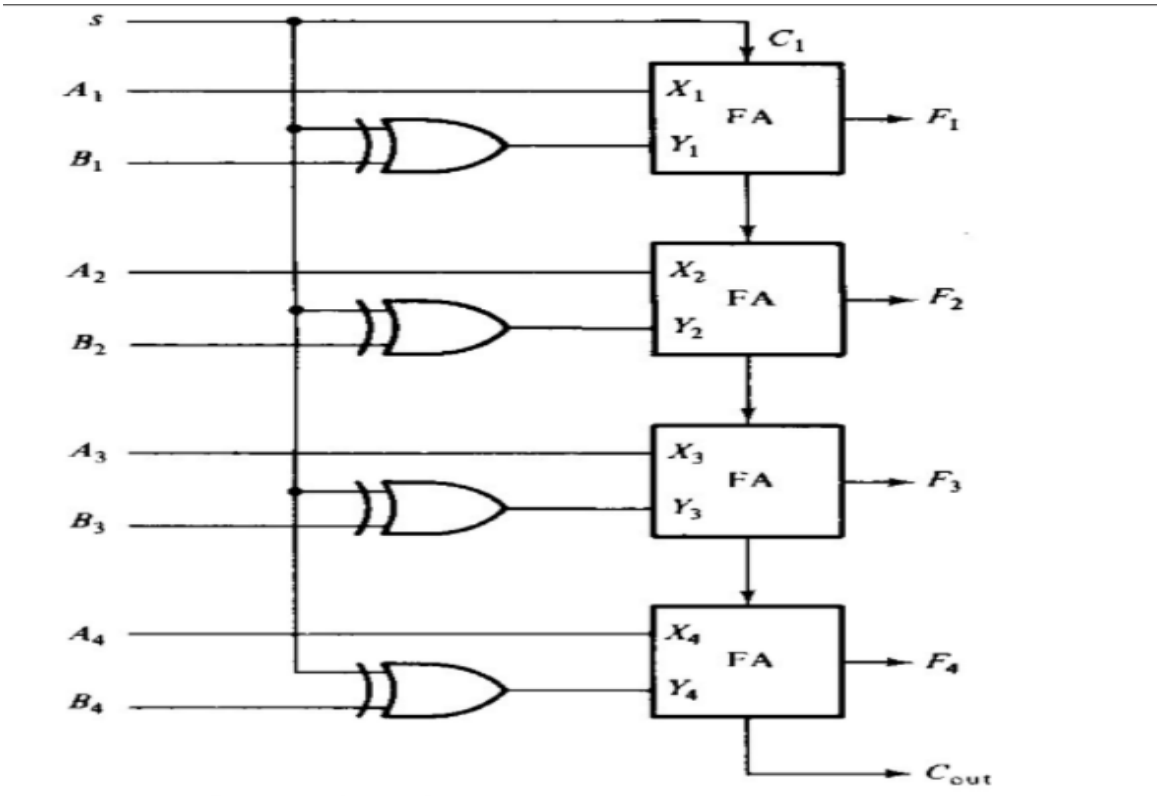
S1	S0	Output	Operation
0	0	Fi=Ai+Bi	OR
0	1	Fi=Ai ⊕ Bi	XOR
1	0	Fi=AiBi	AND
1	1	Fi=Ai’	NOT

Final design of ALU unit by combining arithmetic circuit and logic circuit



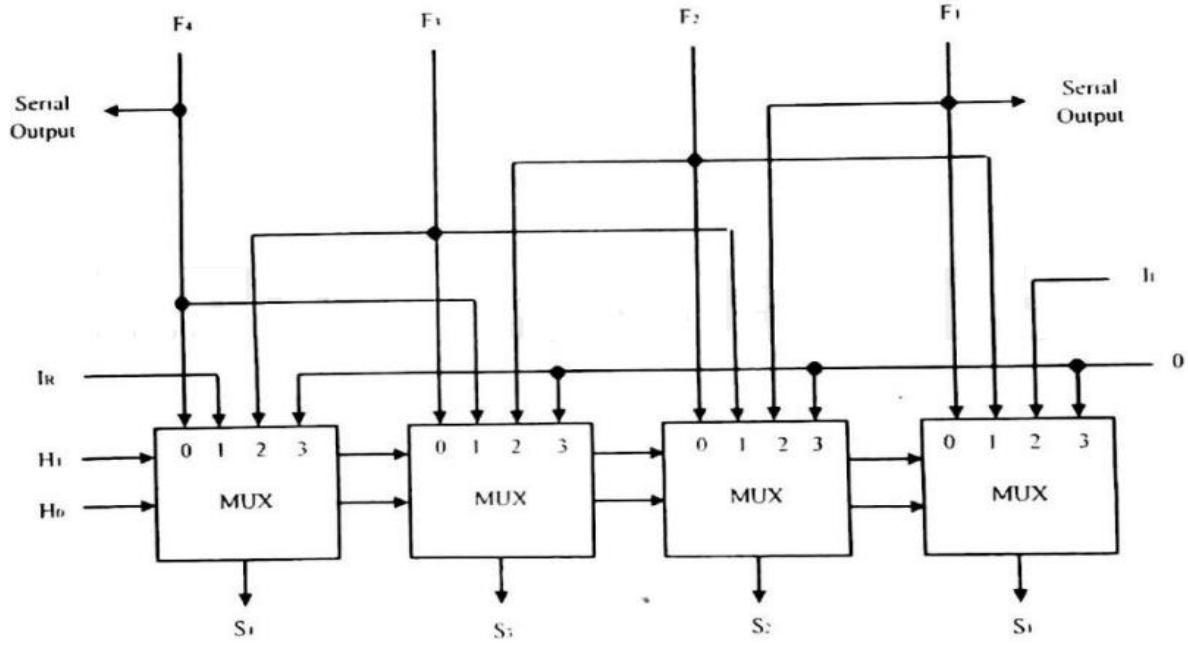
- The logic circuit can be combined with the arithmetic circuit to produce one arithmetic logic unit.
- Selection variables S1 and S0 can be made common to both sections provided we are using a third selection variable S2 to differentiate between the two. This configuration is illustrated in the following figure.
- The outputs of the logic and arithmetic circuits in each stage go through a multiplexer with selection variable S2. When S2=0, the arithmetic output is selected, but when S2= 1, the logic output is selected.

Q) Design an adder or subtractor circuit with one selection variable 's' and two inputs A and B. When s=0 the circuit performs A+B. When s=1, the circuit performs A-B by taking 2's complement of B?



Q) Design of Combinational Logic Shifter?

- The shift unit attached to the processor transfers the output of the ALU onto the output bus. Shifter may function in 4 different ways.
    1. The shifter may transfer the information directly without a shift.
    2. The shifter may shift the information to the right.
    3. The shifter may shift the information to the left.
    4. In some cases no transfer is made from ALU to the output bus.
- A shifter is a bi-directional shift-register with parallel load. The information from ALU can be transferred to the register in parallel and then shifted to the right or left.
- A combinational logic shifter can be constructed with multiplexers. The following figure will show the same.



- In this configuration, a clock pulse is needed for the transfer to the shift register, and another pulse is needed for the shift.
- Another clock pulse may also in need of when information is passed from shift register to destination register.

- The number of clock pulses may reduce if the shifter is implemented with a combinational circuit. In such cases, only one clock pulse is required to transfer from source register to destination register.
- In a combinational logic shifter, the signals from the ALU to the output bus propagate through gates without the need for clock pulse.
- Shifter operation can be selected by two variables H1H0.
    1. If H1 H0=00, no shift is executed and the signals from F go directly to the S lines.
    2. If H1 H0=01, shift right is executed
    3. If H1 H0=10, shift left is executed.
    4. If H1 H0=11, no operation
- The following summarizes the operation of a shifter.

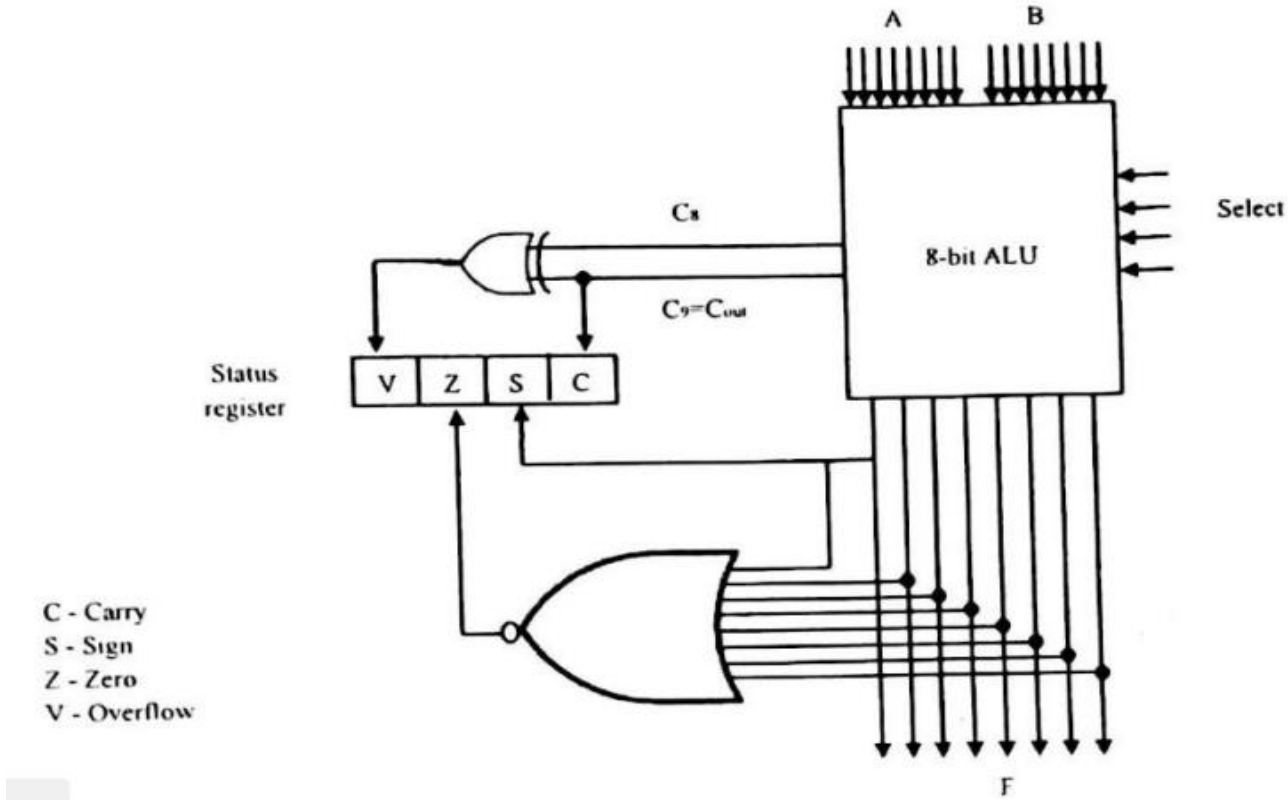
H1	H0	Operation	Function
0	0	S←F	Transfer F to S (no shift)
0	1	S←shr F	Shift right F into S
1	0	S←shl F	Shift left F into S
1	1	S←0	Transfer 0's into S

- The above diagram of combinational-logic shifter shows only four stages of the shifter. The shifter must consist of n stages in a system with n parallel lines.
- Inputs I_R and I_L serve as serial inputs for the last and first stages to fill the gap which must occur during shift right and shift left operations respectively.
- A selection variable H2 may use for specifying what goes into I_R and I_L.

Q) Design 4-bit Status Register?

- The relative magnitude of two numbers may be determined by subtracting one number from the other and then checking certain bit conditions in the resultant difference. This status bit conditions are stored in a status register.

- The checking conditions may vary for signed and unsigned numbers. Status register is a 4-bit register. The four bits are C (carry), Z (zero), S (sign) and V (overflow). These bits are set or cleared as a result of an operation performed in the ALU.
    1. **Bit C** is set if the output carry of an ALU is 1. It is cleared if the output carry is 0.
    2. **Bit S** is set to 1 if the highest order bit of the result in the output of the ALU is 1. If the highest order bit is 0, this bit is cleared.
    3. **Bit Z** is set to 1 if the output of the ALU contains all 0's. That is if the result is zero Z bit is 1, and if the result is nonzero Z bit is 0.
    4. **Bit V** is set if the exclusive OR of carries C8 and C9 is 1, and cleared otherwise. This is the condition for overflow when the numbers are in Signed 2's complement representation.
- The following figure shows the block diagram of an 8-bit ALU with a 4-bit status register.



- After an ALU operation, status bits can be checked to determine the relationship that exist between the values of A and B. If bit V is set after the addition two signed numbers, it indicates an overflow condition.
- If Z is set after an exclusive OR operation, it indicates that A=B. A single bit in A can be checked to determine if it is 0 or 1 by masking all bits except the bit in question and then checking the Z status bit.
- The following table lists the various conditions for determining the relative magnitudes of A and B by checking the status bit when the operation is performed on unsigned binary numbers.

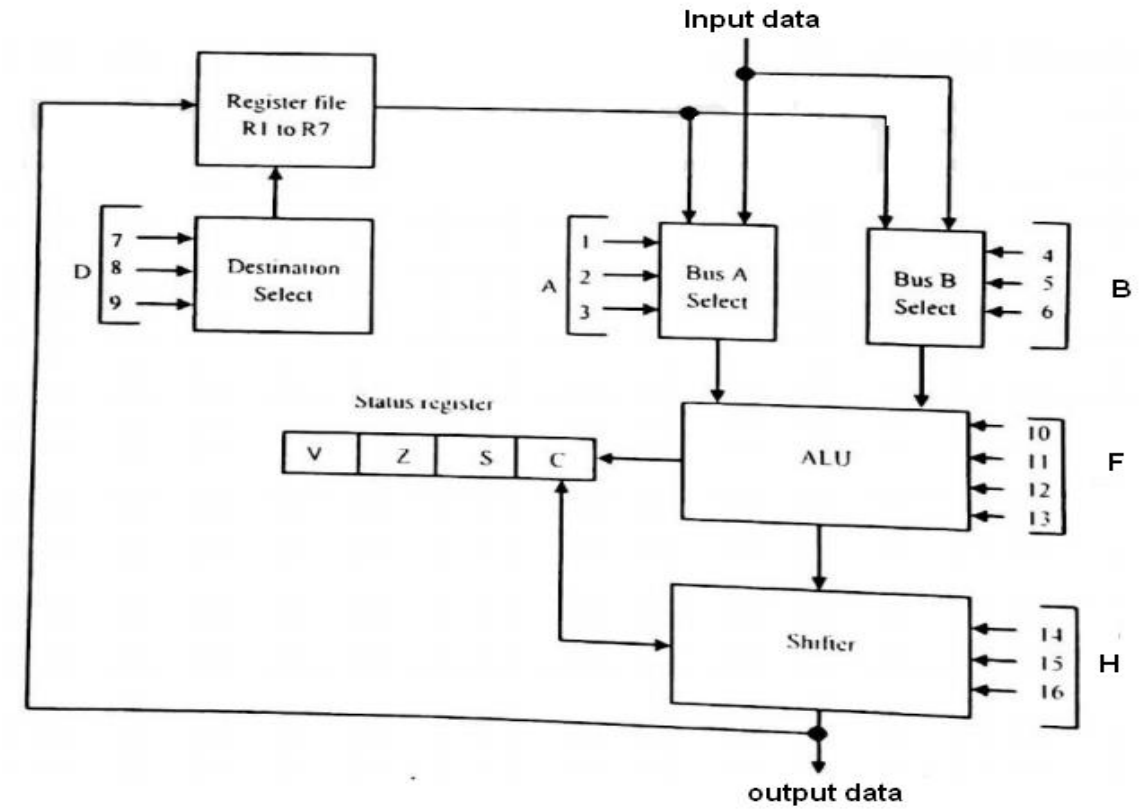
Relation	**Condition of Status bits**	**Boolean function**
A<B	C=1	C
A<=B	C=1or Z=1	C+Z
A>B	C=0	C’
A>=B	C=0 or Z=1	C’+Z
A=B	Z=1	Z
A≠ B	Z=0	Z’

- The following table lists the various conditions for determining the relative magnitudes of A and B by checking the status bits when the operation is performed on signed binary numbers.

Relation	**Condition of Status bits**	**Boolean function**
A>B	Z=0 and (S=0,V=0 or S=1,V=1)	Z’(S ⊙V)
A>=B	Z=1 or (S=0,V=0 or S=1,V=1)	(S ⊙V)
A<B	S=1,V=0 or S=0,V=1	(S ⊕ V)
A<=B	S=1,V=0 or S=0,V=1or Z=1	(S ⊕ V)+Z
A=B	Z=1	Z
A≠ B	Z=0	Z’

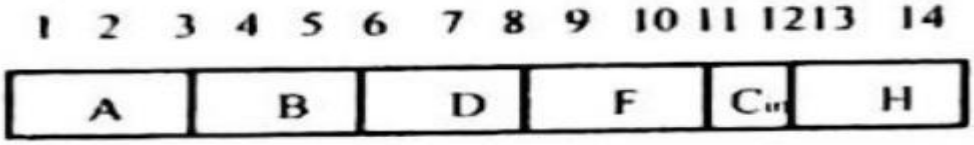
Q) Draw and explain the block diagram of Processor Unit?

- The micro-operations within the processor during a given clock cycle can be determined by the selection variables. The selection variables control the buses, the ALU, the shifter, and the destination register.
- A block diagram for the processing unit is shown in the below figure.



- It consists of seven registers (R1 to R7) and a status register. The output of the seven registers goes through two multiplexers to select the inputs to the ALU.
- If the input is giving from any external source it can also accepting by the same multiplexers. The output from the ALU goes through a shifter and then to a set of external output terminals.
- It is also possible to transfer the content from shifter to any one of the registers.

- There are 16 selection variables in the unit. It can be specified by a control word. The 16 bit control word is shown in fig.

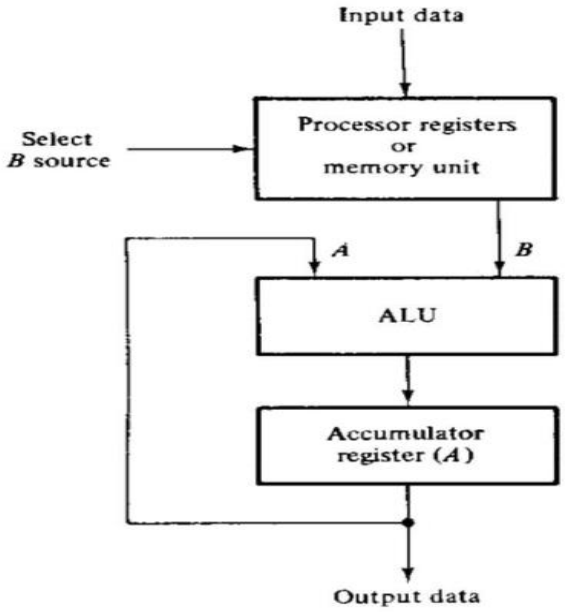


- The control word is partitioned into six fields, with each field is designated by a letter name. All the fields, except Cin, have a code of 3 bits.
    1. A selects the input register for the left side of ALU.
    2. B selects the input register for the right side of ALU.
    3. D selects the destination register.
    4. F and Cin bits together selects the function for ALU.
    5. H selects the type of shift in shifter unit.
- The table given below will gives you the functions of control variables for the processor.

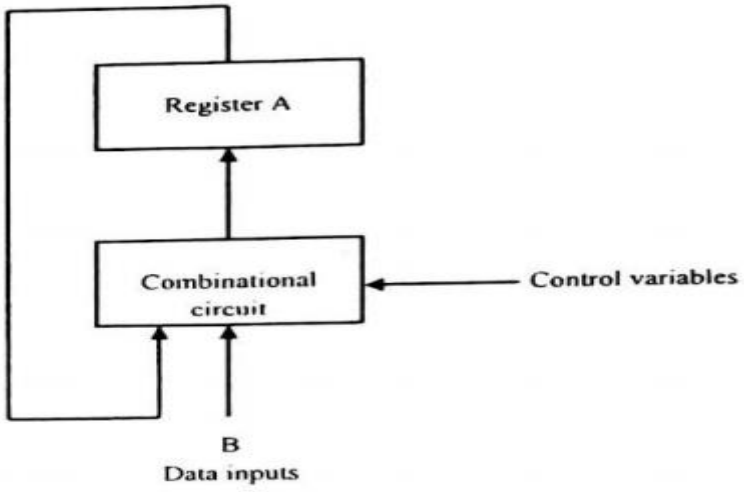
Binary code	**Function of selection variable**					
	A	**B**	**D**	**F with Cin=0**	**F with Cin=1**	**H**
000	**Input data**	**Input data**	**None**	A,C←0	A+1	No shift
001	R1	R1	R1	A+B	A+B+1	Shift right
010	R2	R2	R2	A-B-1	A-B	Shift left
011	R3	R3	R3	A-1	A,C←1	0's to o/p bus
100	R4	R4	R4	AVB	…….	……
101	R5	R5	R5	A⊕B	…….	Calculate right with C
101	R6	R6	R6	A ∧ B	……	…….
111	R7	R7	R7	A'	……..	Calculate left with C

Q) Design of Accumulator?

- Accumulator register is a multipurpose register capable of performing many micro-operations. The organization of a processor unit with an accumulator register is shown below.



- The ALU associated with the register maybe constructed as a combinational circuit. In this configuration, the accumulator register with parallel load is connected to the ALU.
- The block diagram of the accumulator that forms as sequential circuit is shown below:



- Accumulator can also perform data processing operations. Total of nine operations are considered here for the design of accumulator circuit.
- These operations are described below.

Control variable	F with $C_{in}=0$	F with $C_{in}=1$
P1	$A \leftarrow A+B$	Add
P2	$A \leftarrow \underline{0}$	Clear
P3	$A \leftarrow \overline{A}$	Complement
P4	$A \leftarrow A \wedge B$	AND
P5	$A \leftarrow A \vee B$	OR
P6	$A \leftarrow A \oplus B$	Exclusive-OR
P7	A←shr A	Shift-right
P8	A←shl A	Shift-left
P9	A←A+1	Increment
Z bit	If(A=0)then(Z=1)	Check for zero

Design Procedure

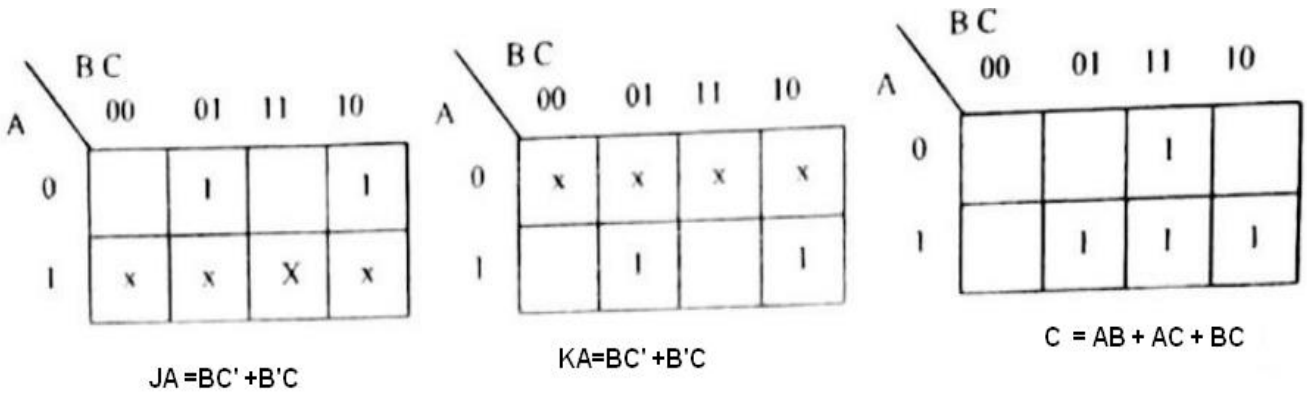
1. **Add B to A** (P1)
    - Add micro-operation is initiated when control variable P, is 1. To perform addition operation, accumulator can use a parallel adder composed of full adders. The full adder in each stage 1 will accept the input and and a previous carry bit C_1.
    - Sum bit is transferred to flip flop A_i, and output carries C_{i+1} is transferred to the next stage as input carry of that stage.
    - The state table of a full adder, when considered as a sequential circuit is shown below.

Present State	Input		Next State	Flip-flop inputs		Output
A_i	B_i	C_i	A_i	JA_i	KA_i	C_{i+1}
0	0	0	0	0	X	0
0	0	1	1	1	X	0
0	1	0	1	1	X	0
0	1	1	0	0	X	1
1	0	0	1	X	0	0
1	0	1	0	X	1	1
1	1	0	0	X	1	1
1	1	1	1	X	0	1

- The excitation input for the JK flip flop is shown below for reference.

Present state	Next State	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

- According to these values the above flip flop inputs are set. The flip flop input functions and the Boolean functions for the output are simplified in the maps as shown in fig.



These two equations should affect the flip flop only when PI is enabled. Therefore, they should be ANDed with control variable P, Then the equation becomes.

$$JA_i = B1C1'P1 + B1'C1P1$$

$$KA_i = B1C1'P1 + B1'C1P1$$

$$C_{i+1} = A1B1 + A1C1 + B1C1$$

2. Clear (P2)

- Control variable P_2 clears all flip flops in register A. To cause this transition in a JK flip flop. we need only apply control satiable P2 to the K input of the flip flop.
- The J input will he assumed to be 0 if nothing is applied on it. The input functions can be written as.

$$JA_i = 0$$

$$KA_i = P2$$

3. COMPLEMENT(P3)

- To cause this transition in a JK flip-flop we need to apply p3 to both J and K inputs.

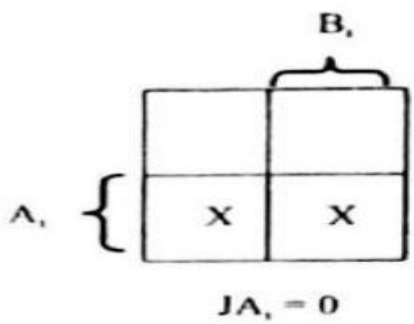
$$JA_i = P3 \qquad KA_i = P3$$

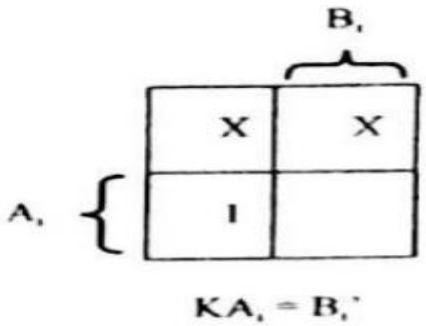
2. AND (P4)

- This micro operation is initiated with control variable P4.This operation performs the logic AND operation between A_i and B_i and transfers the result to A.
- The excitation table for this operation is as shown below.

Present State	**Input**	**Next State**	**Flip-flop Inputs**	
A_i	**B_i**	**A_i**	**JA_i**	**KA_i**
0	0	0	0	X
0	1	0	0	X
1	0	0	X	1
1	1	1	X	0

- The next state of A_i will be 1 only when the present state of A_i and data input B_i, is 1. The flip flop input functions can be simplified with the maps and the equations can be written as:





$$JA_i = 0$$

$$KA_i = B_i'$$

- By including the control variable p4, the equation can be rewritten as:

$$JA_i = 0$$

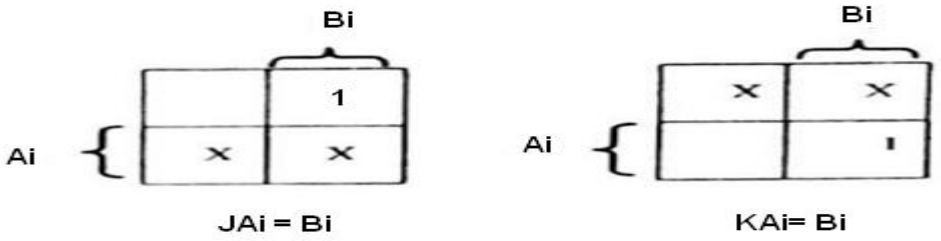
$$KA_i = B_i'P4$$

4. OR (P5)

- Control variable P5, initiates the logic OR operation between A_i and B_i The result is transferred to A_i.
- The excitation table for this operation is as shown below.

Present State	**Input**	**Next State**	**Flip-flop Inputs**	
A_i	**B_i**	**A_i**	**JA_i**	**KA_i**
0	0	0	0	X
0	1	1	1	X
1	0	1	X	0
1	1	1	X	0

- The simplified equations in the maps dictate that the J input be enabled when Bi =1. When Bi=0, the present state and next state of Ai are the same.
- When Bi=1, the J input is enabled and the next state of Ai, becomes 1. Input functions for the OR micro operation are:



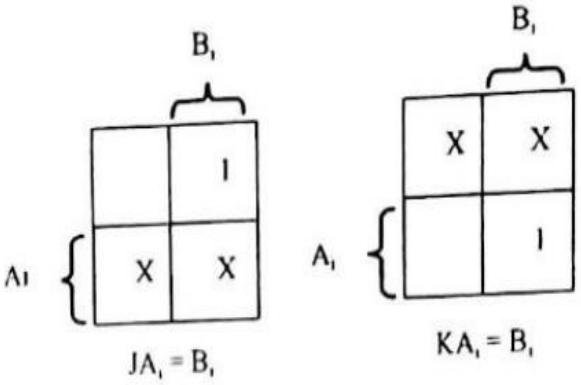
$$JA_i = B_i P5$$

$$KA_i = 0$$

6. Exclusive-OR (P6)

- Control variable P6 initiates the logic Exclusive-OR operation between A_i and B_i. The result is transferred to A_i.
- The excitation table and map simplification is as shown below.

Present State	Input	Next State	Flip-flop Inputs	
A_i	B_i	A_i	JA_i	KA_i
0	0	0	0	X
0	1	1	1	X
1	0	1	X	0
1	1	0	X	1



- The flip flop input functions are written as:

$$JA_i = B_i P6$$

$$KA_i = B_i P6$$

7. Shift-right (P7)

- Control variable P7 initiates the shift operation of Ai register one bit to the right. That is the value of flip flop A_{i+1} is transferred to flip flop A_i.
- The flip flop input functions can be written as:

$$JA_i = A_{i+1}\, P7$$

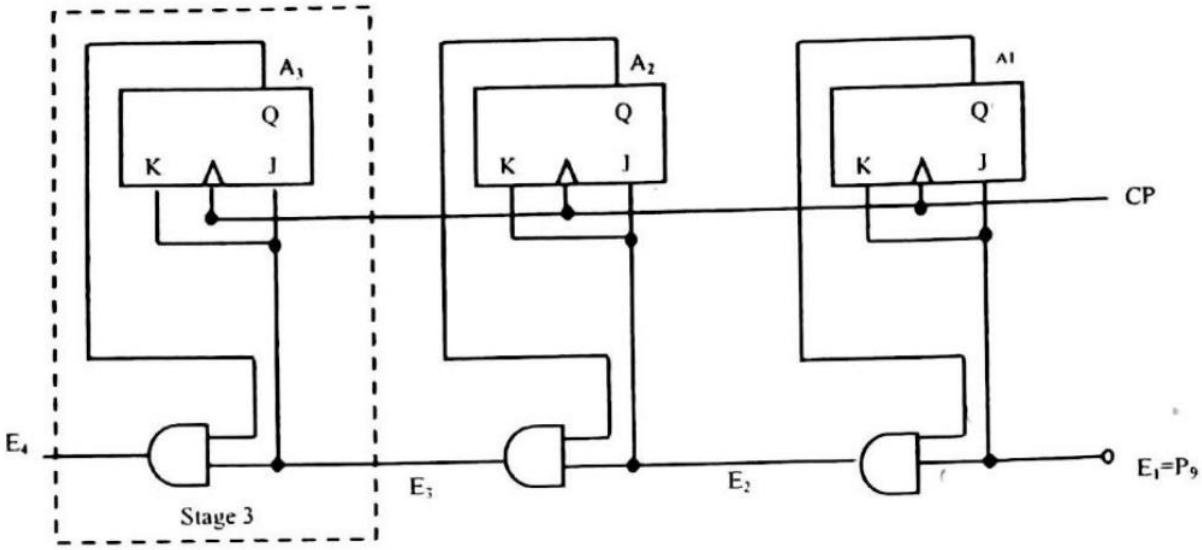
$$KA_i = A_{i+1}\, P7$$

8. Shift-left (P8)

- Control variable P8 initiates the shift operation of A_i register one bit to the left. That is, the value of flip flop A_{i-1} is transferred to flip flop A_i .
- The flip flop input functions can be written as:
  - $JA_i = A_{i-1}\, P8$
  - $KA_i = A_{i-1}\, P8$

9. Increment (P9)

- These operations increment the content of A register by one. The register behaves likes a synchronous binary counter with P9 enabling the count.
- A 3 bit synchronous counter is shown in the following figure.



- The Boolean function for a typical stage can be written as:

$$JA_i = E_i$$

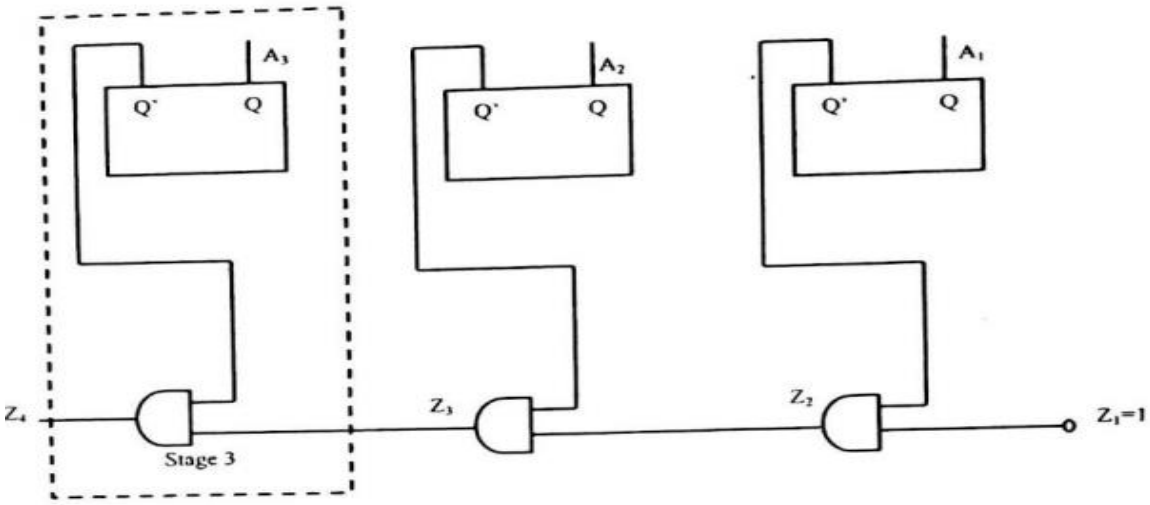
$$KA_i = E_i$$

$$E_{i+1} = E_i A_i \qquad i = 1,2,\ldots,n$$

$$E_1 = P_9$$

10. Check for Zero (Z)

- Variable Z is an output from the accumulator. This variable can be used to indicate a zero content in the A register.
- All the flip flops in the accumulator is cleared Z variable will be set to 1. When a flip flop is cleared, its complement output Q' is equal to 1.
- The following figure shows the first three stages of the accumulator that checks for zero content.



- Each stage generates a variable Z_{i+1} by ANDing the complement output of A_i to an input variable Z_i. In this way, a chain of AND gates through all stages will indicate if all flips ate cleared.
- The Boolean function for a typical stage can be expressed as:

$$\mathbf{Z_{i+1} = Z_i A_i \quad i=1,2,3\ldots\ldots\ldots\ldots\ldots\ldots n}$$

$$\mathbf{Z_1 = 1}$$

$$\mathbf{Z_{n+1} = Z}$$

One stage of Accumulator

- Combining all the input functions for the J and K inputs flip flop A_1 produces a composite set of input Boolean functions for a typical stage.

$$JA_i = B_i C_i' P_i + B_i' C_i P_i + P_3 + B_i P_5 + B_i P_6 + A_{i+1} P_7 + A_{i-1} P_8 + E_i$$

$$KA_i = B_i C_i' P_i + B_i' C_i P_i + P_2 + P_3 + B_i' P_4 + B_i P_6 + A'_{i+1} P_7 + A'_{i-1} P_8 + E_i$$

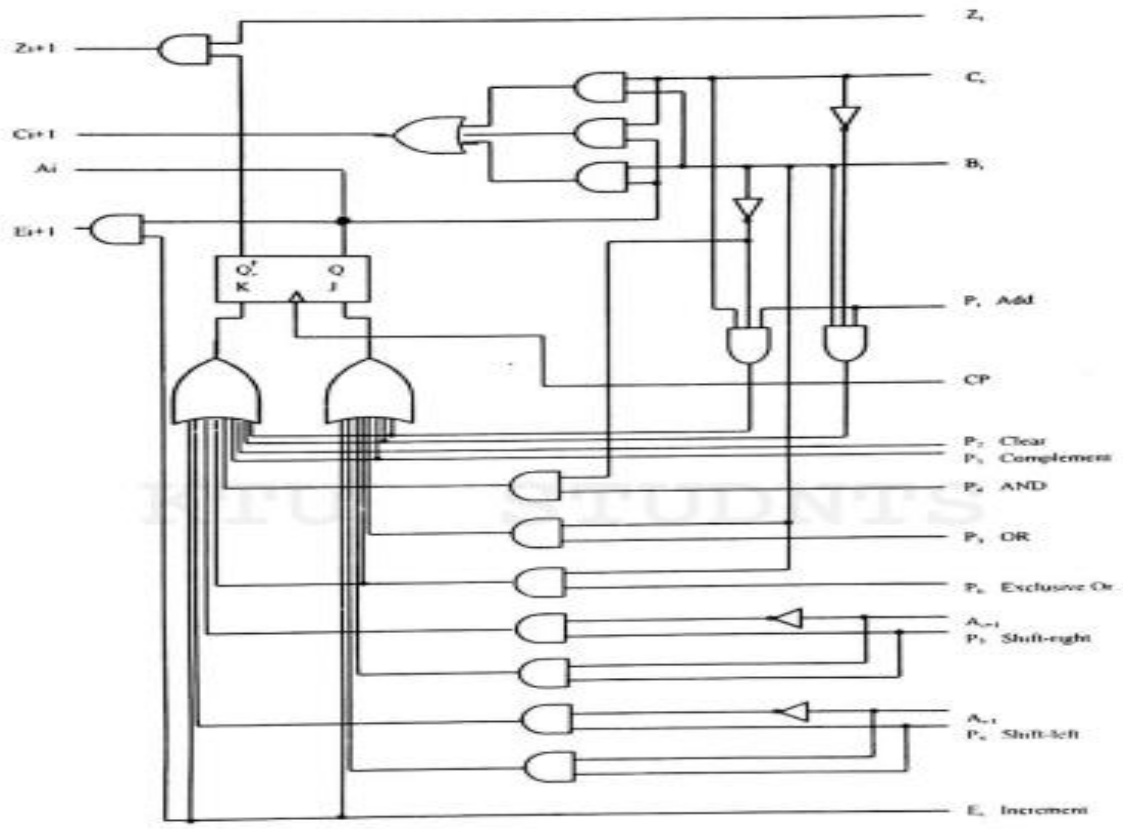
- Each stage in the accumulator must produce the carry for the next stage.

$$C_{i+1} = A_i B_i + A_i C_i + B_i C_i$$

$$E_{i+1} = E_i A_i$$

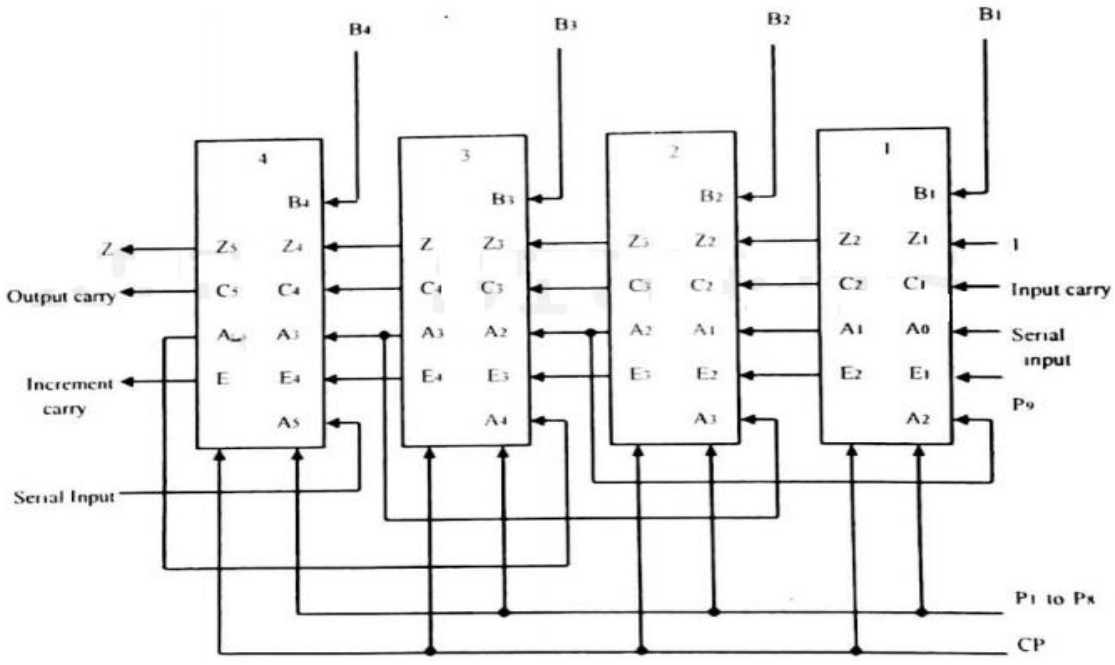
$$Z_{i+1} = Z_i A_i'$$

- The logic diagram for one typical stage of the Accumulator is shown below:



Complete Accumulator

- For a complete accumulator there will be n stages like this. The inputs and outputs of each stage can he connected in cascade to form a complete accumulator.



PREVIOUS YEAR UNIVERSITY QUESTION

1. Describe processor organization with diagram using a) scratchpad memory b) two port memory.
2. Design a 4bit Arithmetic unit which performs the following operations on two inputs A and B, controlled by selection variables s_1 and s_0 and input carry C_{in}

s_1	s_0	$C_{in} = 0$	$C_{in} = 1$
0	0	F=A	F=A+1
0	1	F=A+B	F=A+B+1
1	0	F=A+B'	F=A+B'+1
1	1	F=A-1	F=A

3. Write notes on status register.
4. Design a bus system for interconnecting four n bit registers
5. Design a 4bit combinational logic shifter
6. Design an adder/subtractor circuit with one selection variable S and two inputs A and B. When S=0, the circuit performs A+B and when S=1 it performs A-B by taking 2's complement of B,
7. Describe the different ways in which a general-purpose processor unit can be organized.
8. Write the register transfer logic format for a conditional control statement. Give an example and explain the same.
9. Mention the advantages of using a scratch pad memory. Draw the diagram of a processor that employs a scratch pad memory and explain the same.
10. Design a 4-bit combinational logic shifter with 2 control signals H_1 and H_0 that performs the following operations (bit values given in parenthesis are the values of control variables H_1 and H_0 respectively):- No shift (00), Shift-right (01), Shift left (10), Transfer 0's to S(11).
11. Draw and explain the block diagram for a 4-bit complete accumulator

12. Discuss about condition code bits in a 4 bit status register
13. Discuss shift and conditional control micro operations.
14. An 8-bit register A has one input x. The register operation is represented symbolically as P: $A_7 \leftarrow x$, $A_i \leftarrow A_{i+1}$ i = 0,1,2,3... 6. What is the function of the register?
15. Draw the block diagram for the hardware that implements the following statement

    x + yz: AR ← AR + BR where AR and BR are two n-bit registers and x, y, and z are control variables. Include the logic gates for the control function. (The symbol + designates an OR operation in a control or Boolean function and an arithmetic plus in a micro operation.)
16. Explain the design of status register.
17. Give a simple design for generating status bits for a 8-bit ALU.
18. Draw a labelled block diagram of a processor unit with seven registers R1 to R7 a status register , ALU with 3-selection variables and C_{in} and shifter with 3 selection variables.