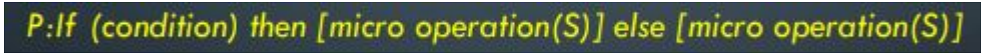
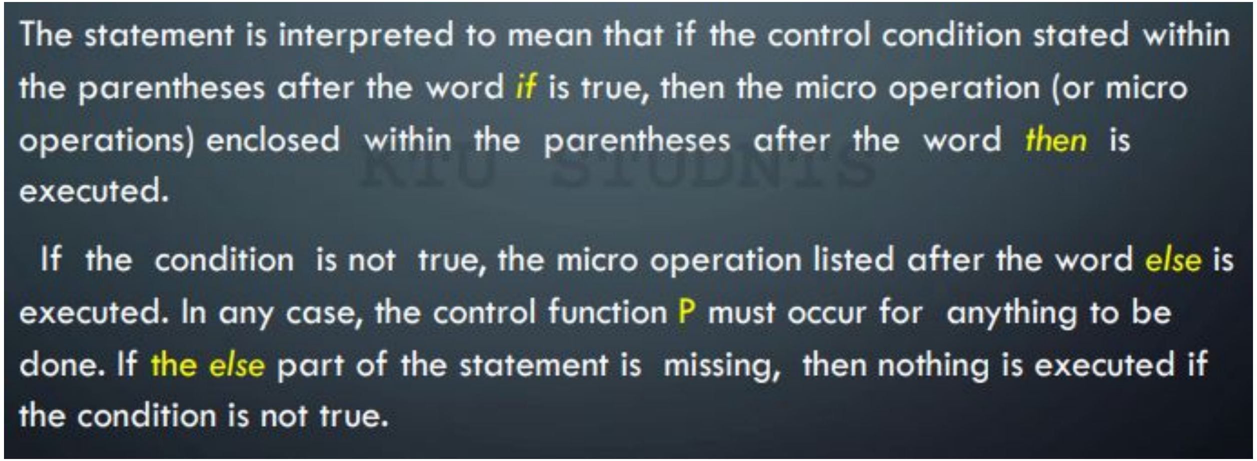
Write notes on conditional control statements.

- A conditional control statement is symbolized by an if-then-else statement in the following manner:

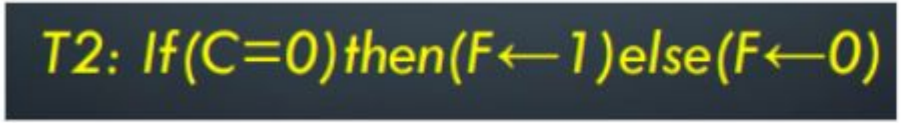


P:If (condition) then [micro operation(S)] else [micro operation(S)]



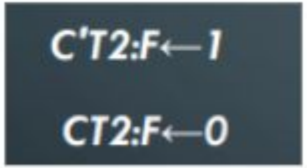
The statement is interpreted to mean that if the control condition stated within the parentheses after the word *if* is true, then the micro operation (or micro operations) enclosed within the parentheses after the word *then* is executed.

If the condition is not true, the micro operation listed after the word *else* is executed. In any case, the control function P must occur for anything to be done. If the *else* part of the statement is missing, then nothing is executed if the condition is not true.

Example: 

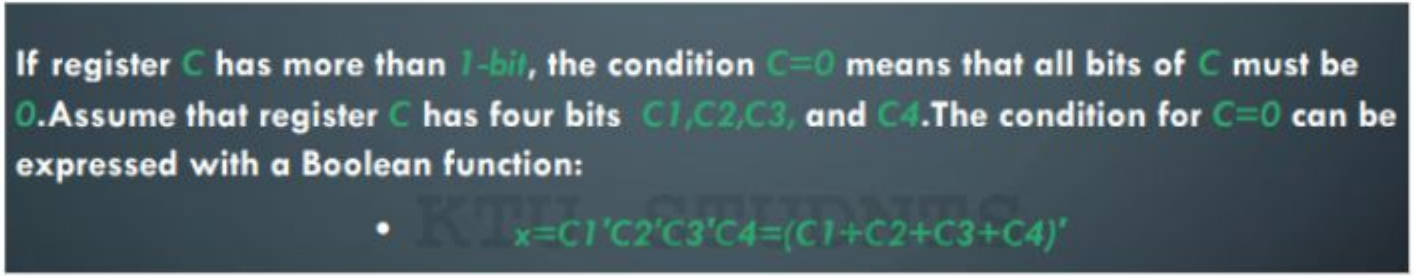
$T2: If(C=0) then (F \leftarrow 1) else (F \leftarrow 0)$

- F and C is assumed to be a 1 bit flipflop that can be set or cleared.
- Here, the statement is equivalent to the following two statements:



$C'T2: F \leftarrow 1$

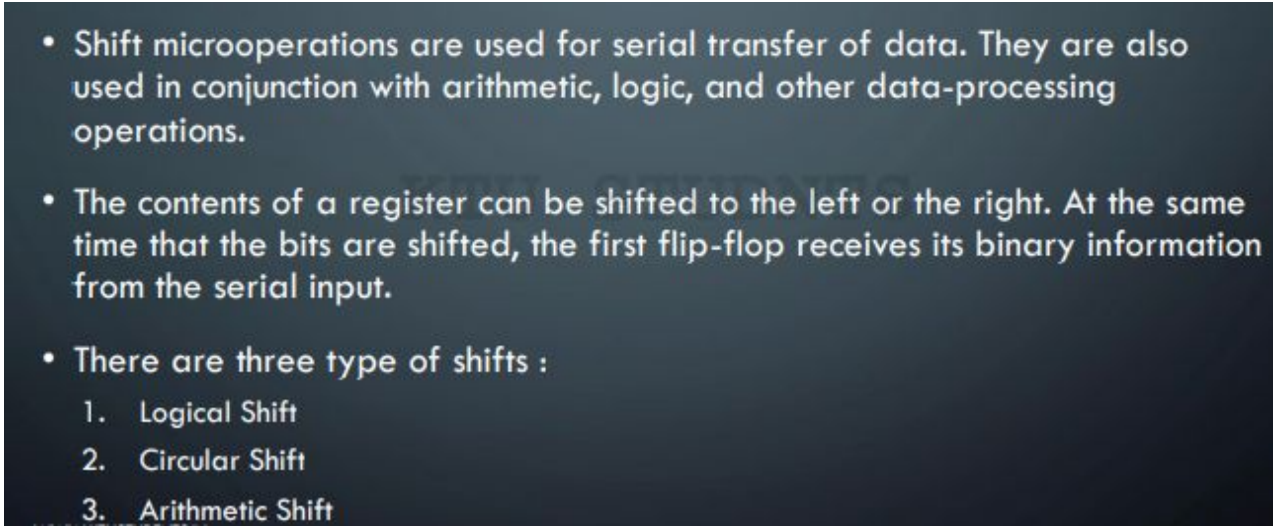
$CT2: F \leftarrow 0$



If register C has more than 1-bit, the condition C=0 means that all bits of C must be 0.Assume that register C has four bits C1,C2,C3, and C4.The condition for C=0 can be expressed with a Boolean function:

- $x = C1'C2'C3'C4 = (C1+C2+C3+C4)'$

Discuss shift and conditional control micro operations.



- Shift microoperations are used for serial transfer of data. They are also used in conjunction with arithmetic, logic, and other data-processing operations.

- The contents of a register can be shifted to the left or the right. At the same time that the bits are shifted, the first flip-flop receives its binary information from the serial input.

- There are three type of shifts :
  1. Logical Shift
  2. Circular Shift
  3. Arithmetic Shift

An 8-bit register A has one input x. The register operation is represented symbolically as P: A7 ← x, $A_i \leftarrow A_{i+1}$ i = 0,1,2,3 ... 6. What is the function of the register?

A7 ← x

A6 ← A7

A5 ← A6

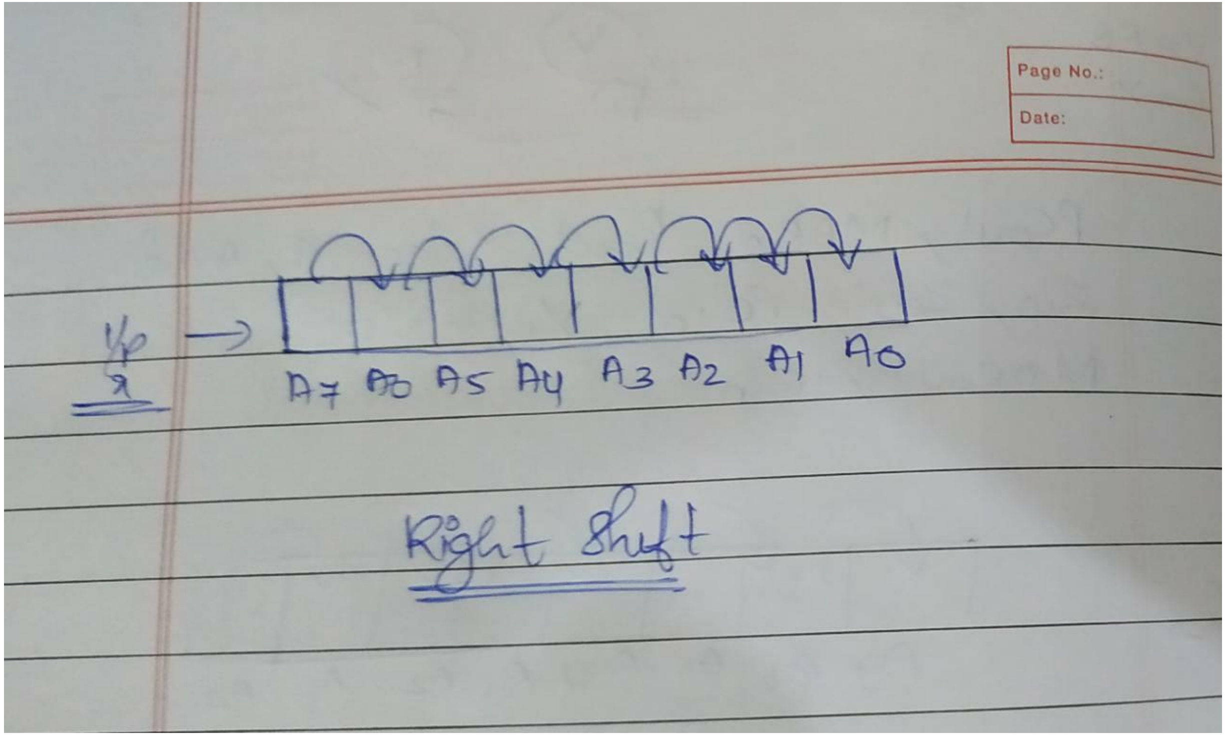
A4 ← A5

A3 ← A4

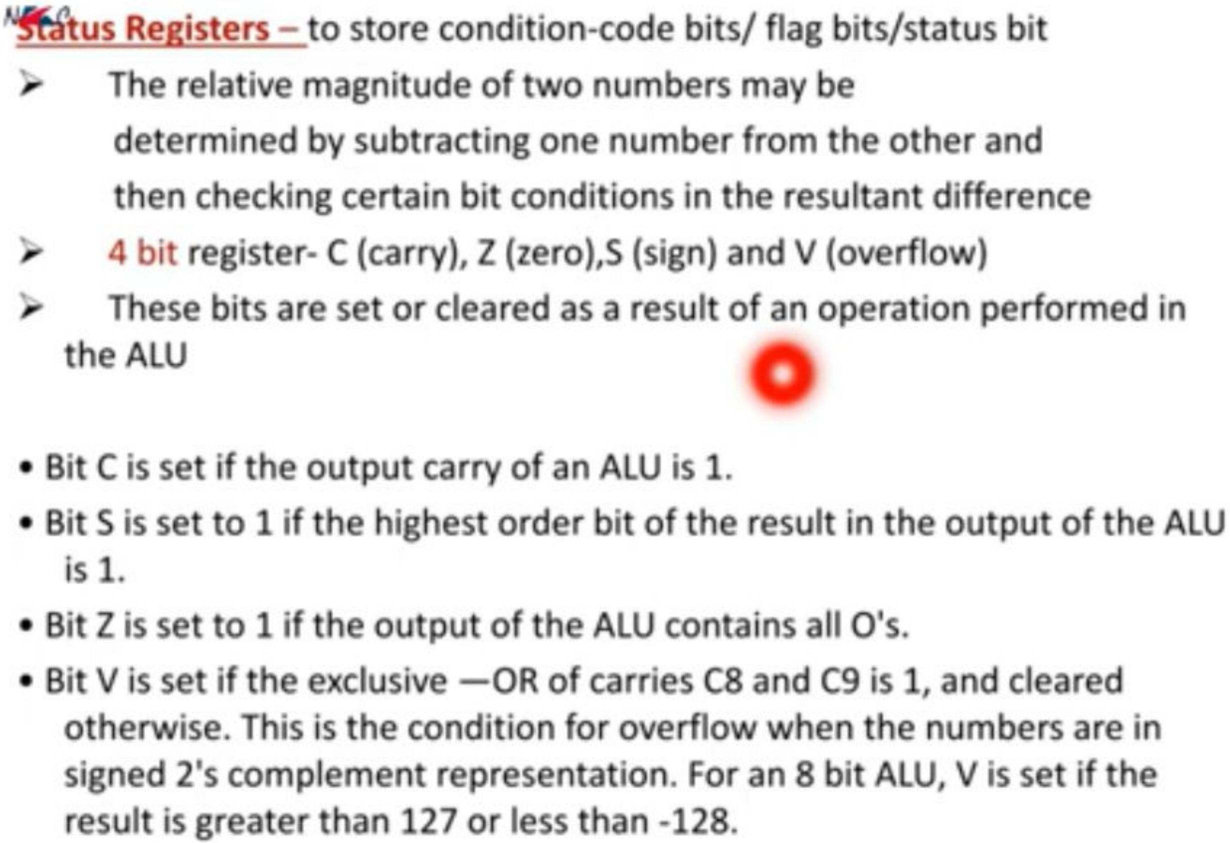
A2 ← A3

A1 ← A2

A0 ← A1



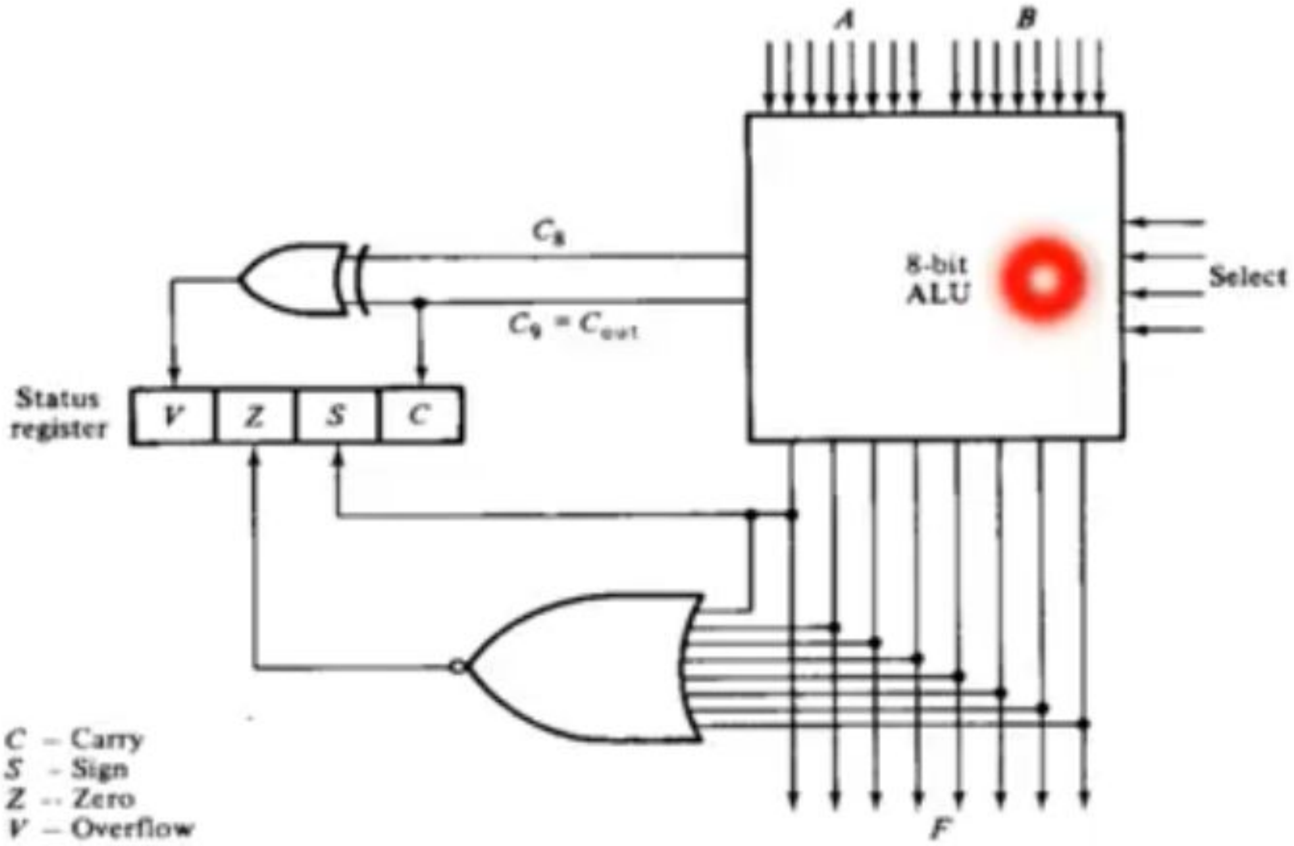
Explain the design of status register.



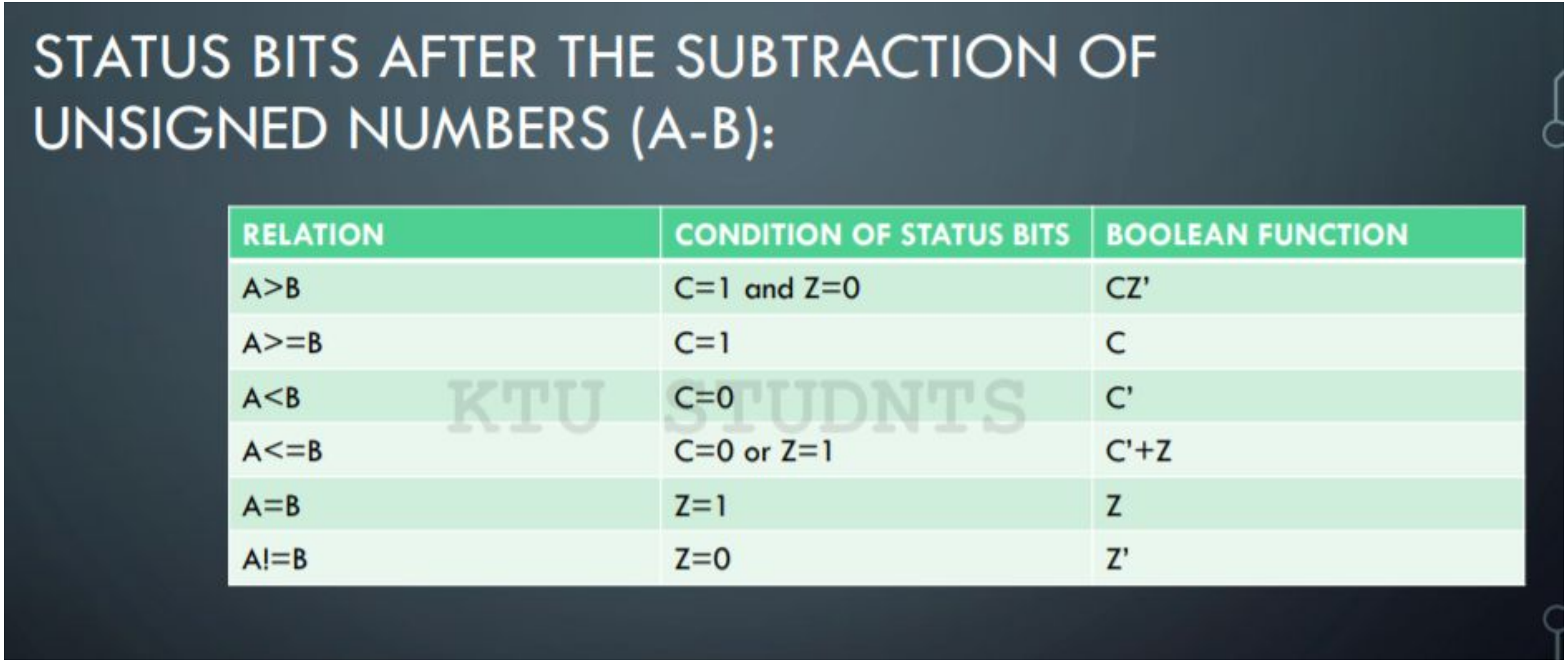
Status Registers – to store condition-code bits/ flag bits/status bit

- ➢ 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
- ➢ 4 bit register- 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

- Bit C is set if the output carry of an ALU is 1.
- Bit S is set to 1 if the highest order bit of the result in the output of the ALU is 1.
- Bit Z is set to 1 if the output of the ALU contains all O's.
- 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. For an 8 bit ALU, V is set if the result is greater than 127 or less than -128.



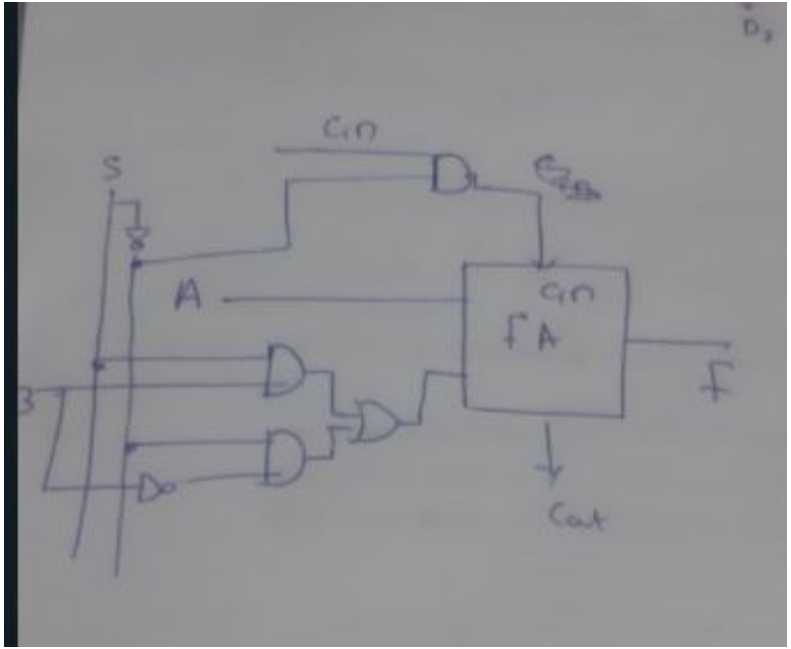
C – Carry
S – Sign
Z -- Zero
V – Overflow



STATUS BITS AFTER THE SUBTRACTION OF UNSIGNED NUMBERS (A-B):

RELATION	CONDITION OF STATUS BITS	BOOLEAN FUNCTION
A>B	C=1 and Z=0	CZ'
A>=B	C=1	C
A<B	C=0	C'
A<=B	C=0 or Z=1	C'+Z
A=B	Z=1	Z
A!=B	Z=0	Z'

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.



S	A	B	Result	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	0	0
1	0	1	1	1
1	1	0	1	0
1	1	1	0	0

Describe processor organization with diagram using a) scratchpad memory b) Two port memory. (10)

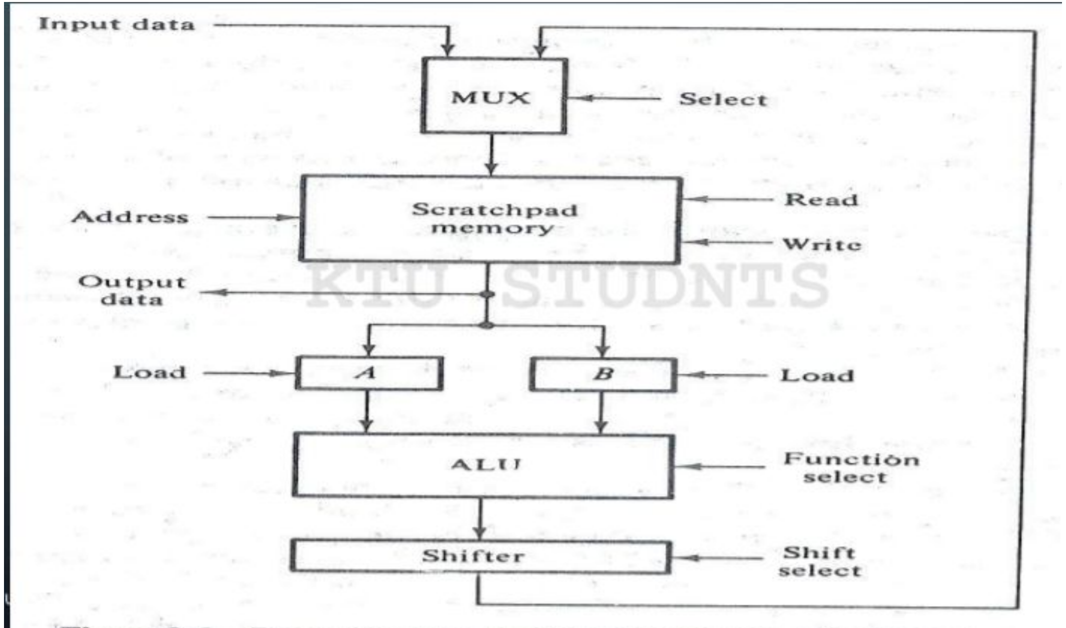


Figure 9-2 Processor unit employing a scratchpad memory

- R1 ← R2 + R3

T1: A ← M[010] read R2 to register A

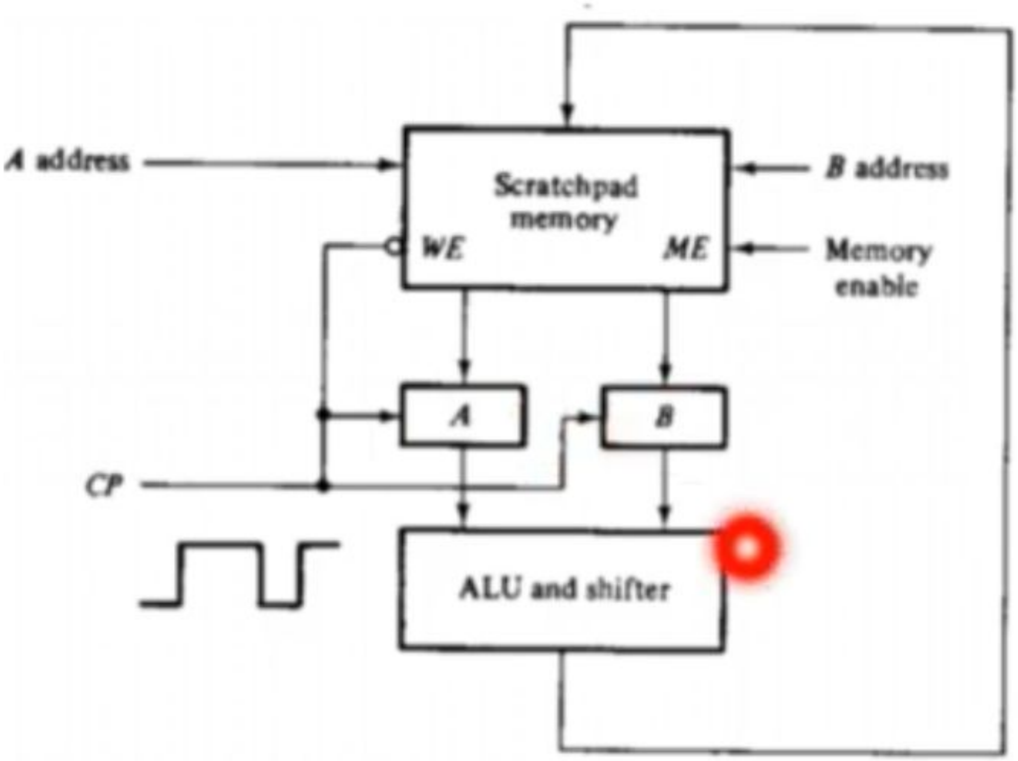
T2: B ← M[011] read R3 to register B

T3: M[001] ← A + B perform operation in ALU and transfer result to R1

Registers in a processor unit can be enclosed within a small memory unit

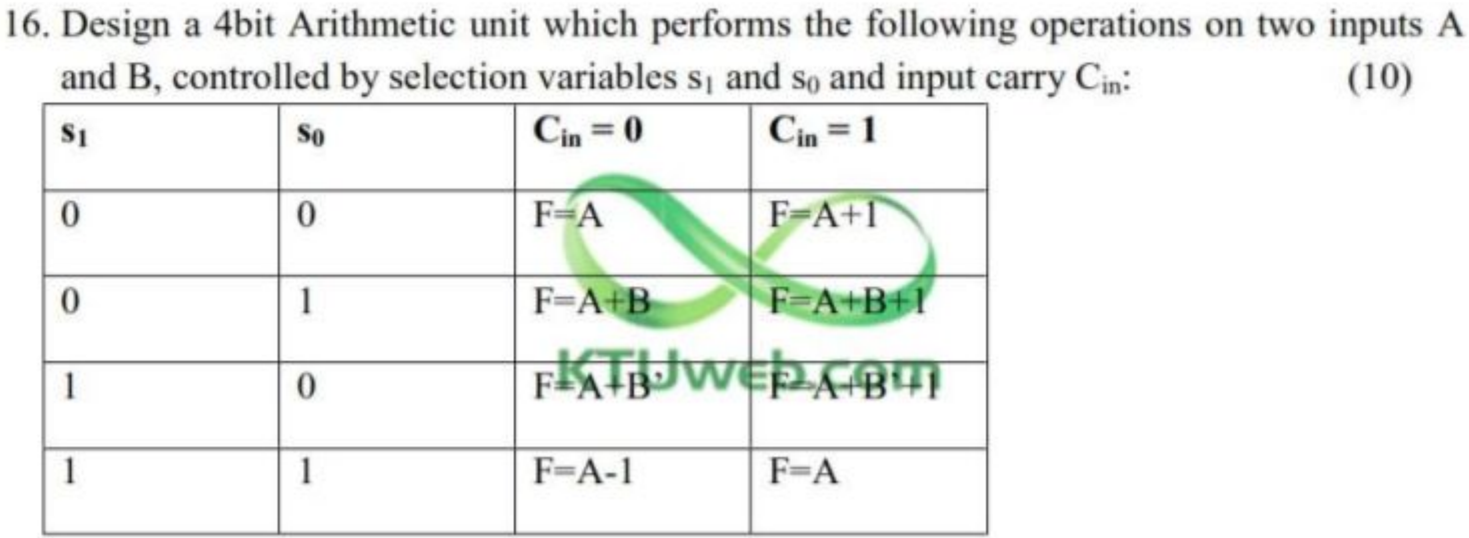
A small memory when included in a processor unit is scratchpad memory

Scratch Pad Memory with 2 Ports



Overcomes delay

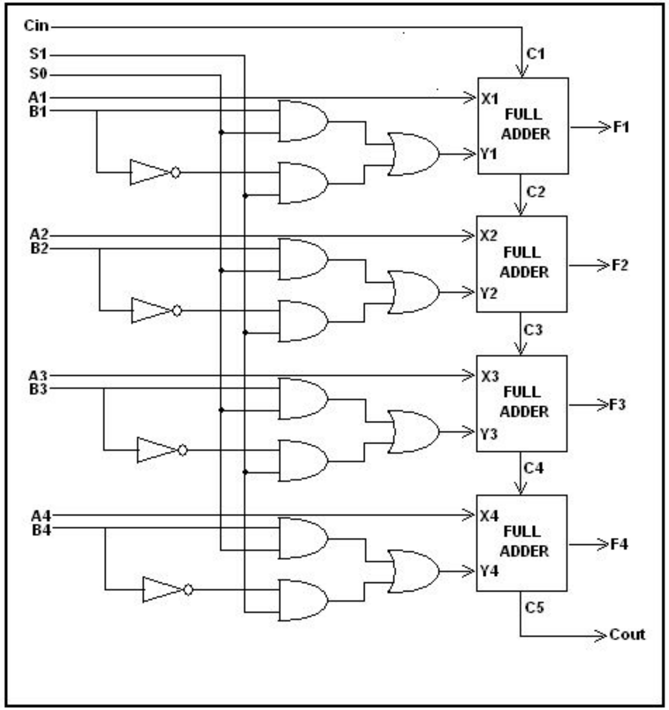
- 2-port memory has two address lines to select two words of memory simultaneously
- A and B are latches
- CP =1 , WE=1 :- A and B are open and reads data from memory.
- CP=0 , WE =0:- A and B are closed, if ME is enabled result of microoperation written into B address
- R1←R2+R3 can be done in one clock pulse.
- B address is fixed as destination address.



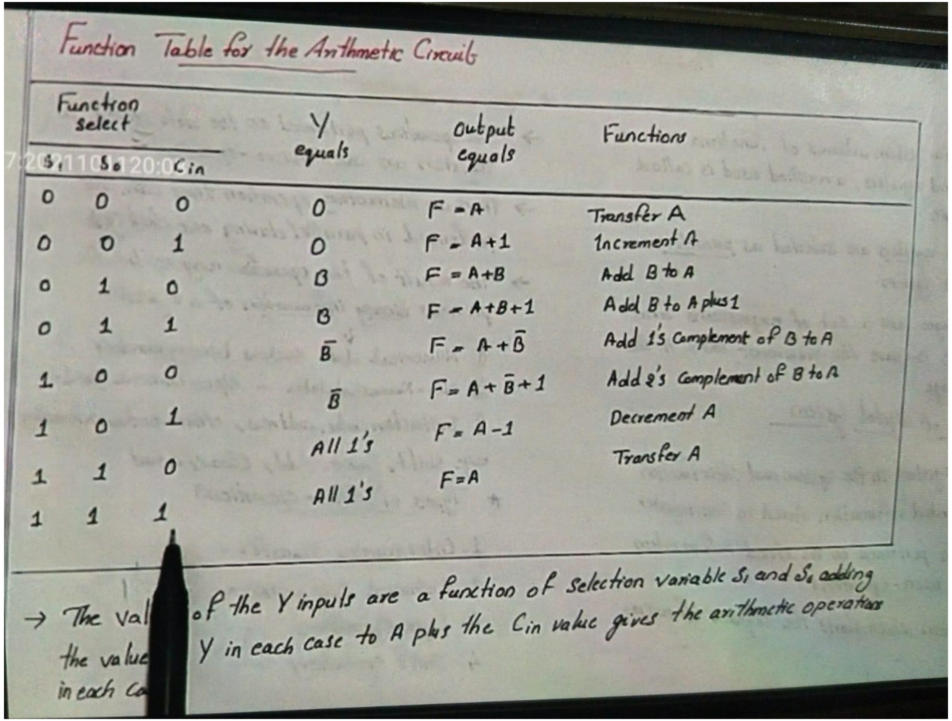
16. Design a 4bit Arithmetic unit which performs the following operations on two inputs A and B, controlled by selection variables s_1 and s_0 and input carry C_{in}: (10)

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

Answer next slide



Arithmetic Circuit



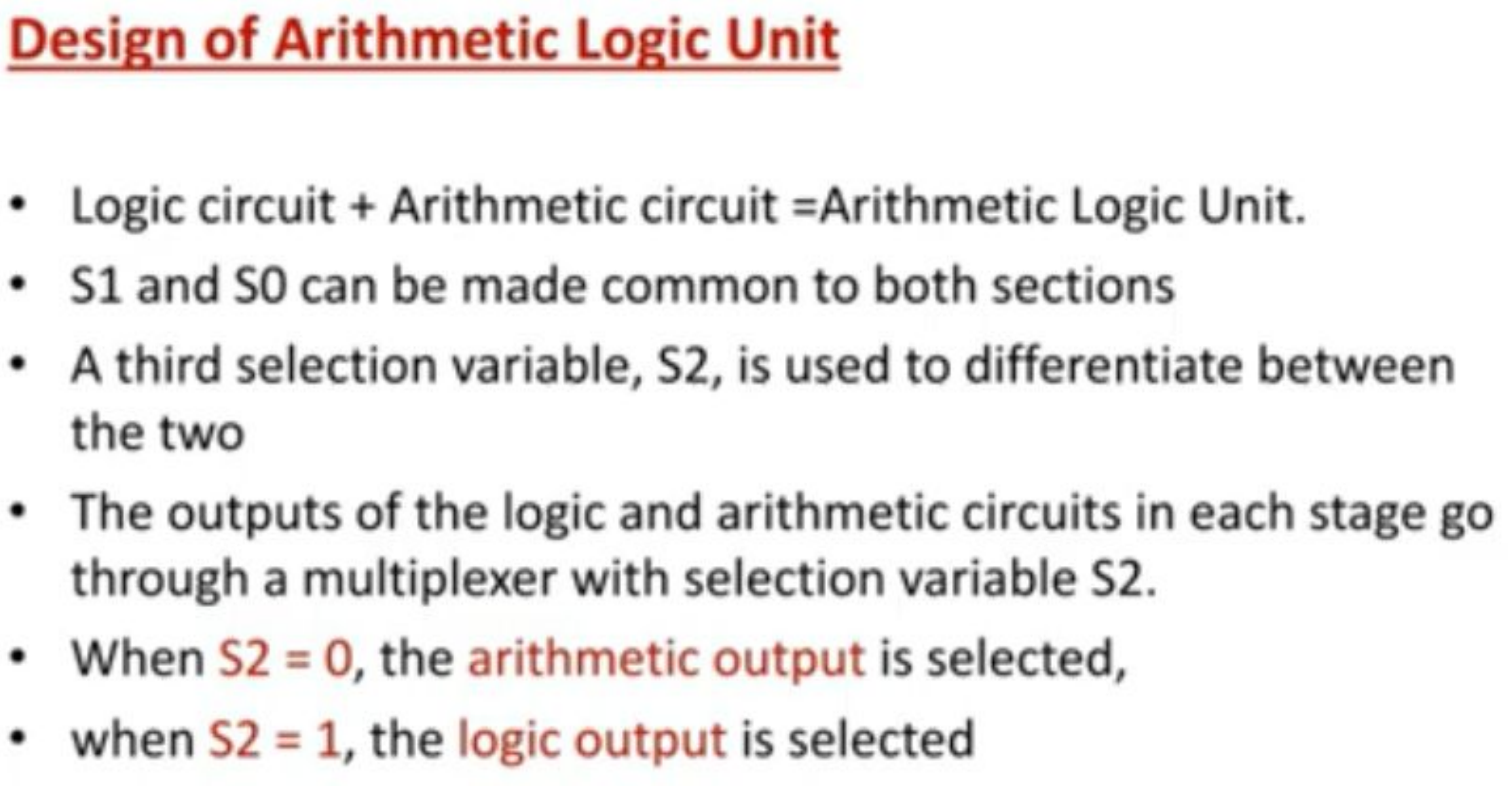
Function Table for the Arithmetic Circuit

Function select			Y equals	Output equals	Functions
S_1	S_0	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	$\bar{B}$	F = A + $\bar{B}$	Add 1's Complement of B to A
1	0	1	$\bar{B}$	F = A + $\bar{B}$ + 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

→ The val of the Y inputs are a function of selection variable S_1 and S_0 adding the value Y in each case to A plus the Cin value gives the arithmetic operations in each ca

If 10 marks write all these otherwise just make th combined figure shown in the below slides

How to construct an ALU, which perform basic arithmetic and logic operations?



Design of Arithmetic Logic Unit

- Logic circuit + Arithmetic circuit =Arithmetic Logic Unit.
- S1 and S0 can be made common to both sections
- A third selection variable, S2, is used to differentiate between the two
- 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,
- when S2 = 1, the logic output is selected

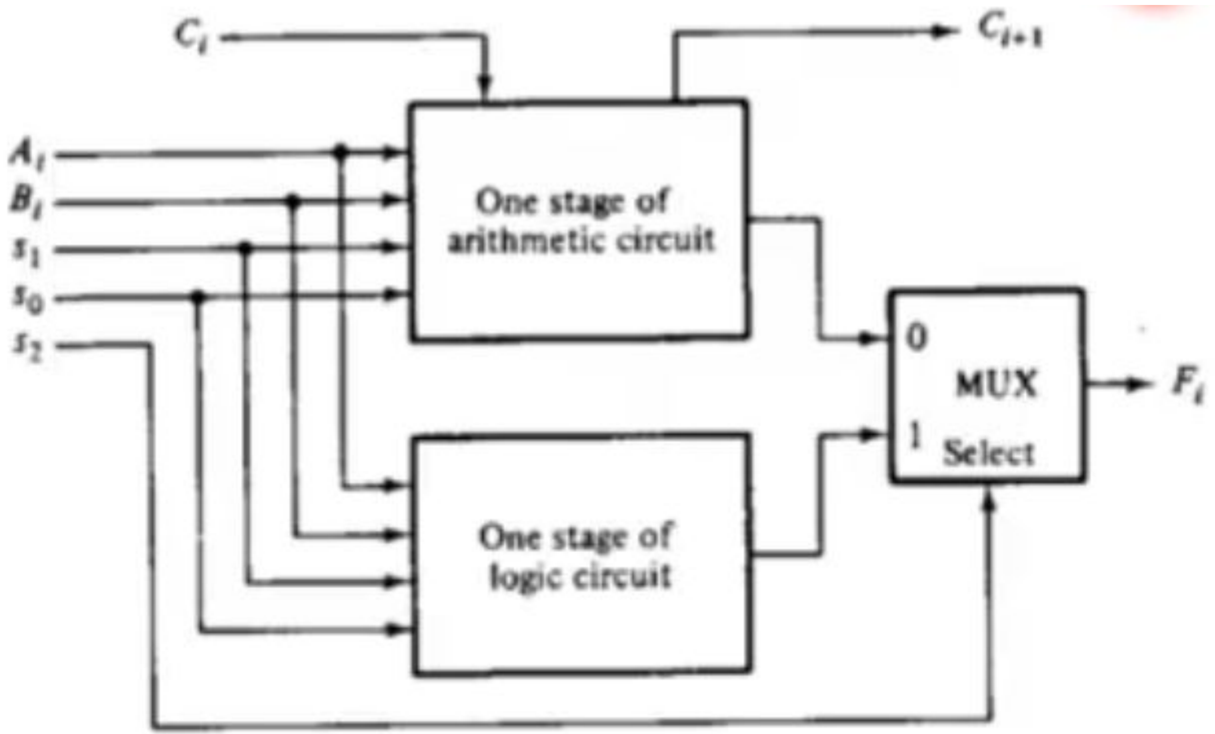
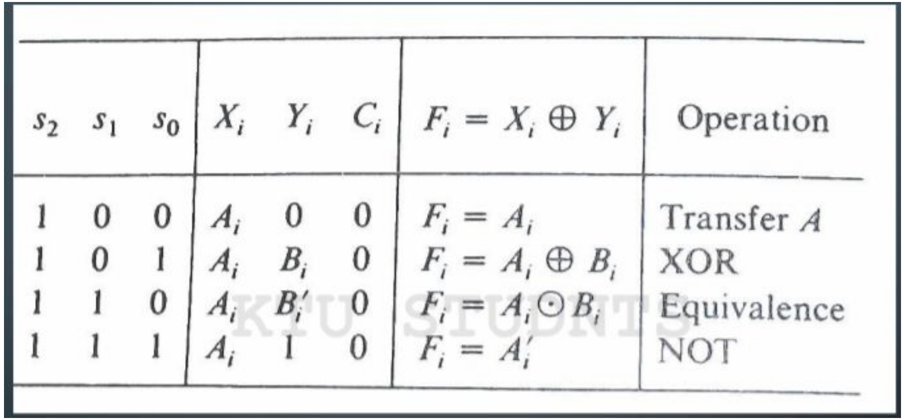


Figure 9-12 Combining logic and arithmetic circuits



s_2	s_1	s_0	X_i	Y_i	C_i	$F_i = X_i \oplus Y_i$	Operation
1	0	0	A_i	0	0	$F_i = A_i$	Transfer A
1	0	1	A_i	B_i	0	$F_i = A_i \oplus B_i$	XOR
1	1	0	A_i	B_i'	0	$F_i = A_i \odot B_i$	Equivalence
1	1	1	A_i	1	0	$F_i = A_i'$	NOT

When S2=1

Xi = Ai + S2 S1’ S0’Bi + S2 S1 S0’ Bi’

Yi = S0 Bi + S1 Bi’

Zi = S2’ Ci

When S2 = 0, the three functions reduce to:

Xi = Ai

Yi = S0 Bi + S1 Bi’

Zi = Ci

which are the function for the arithmetic circuit.

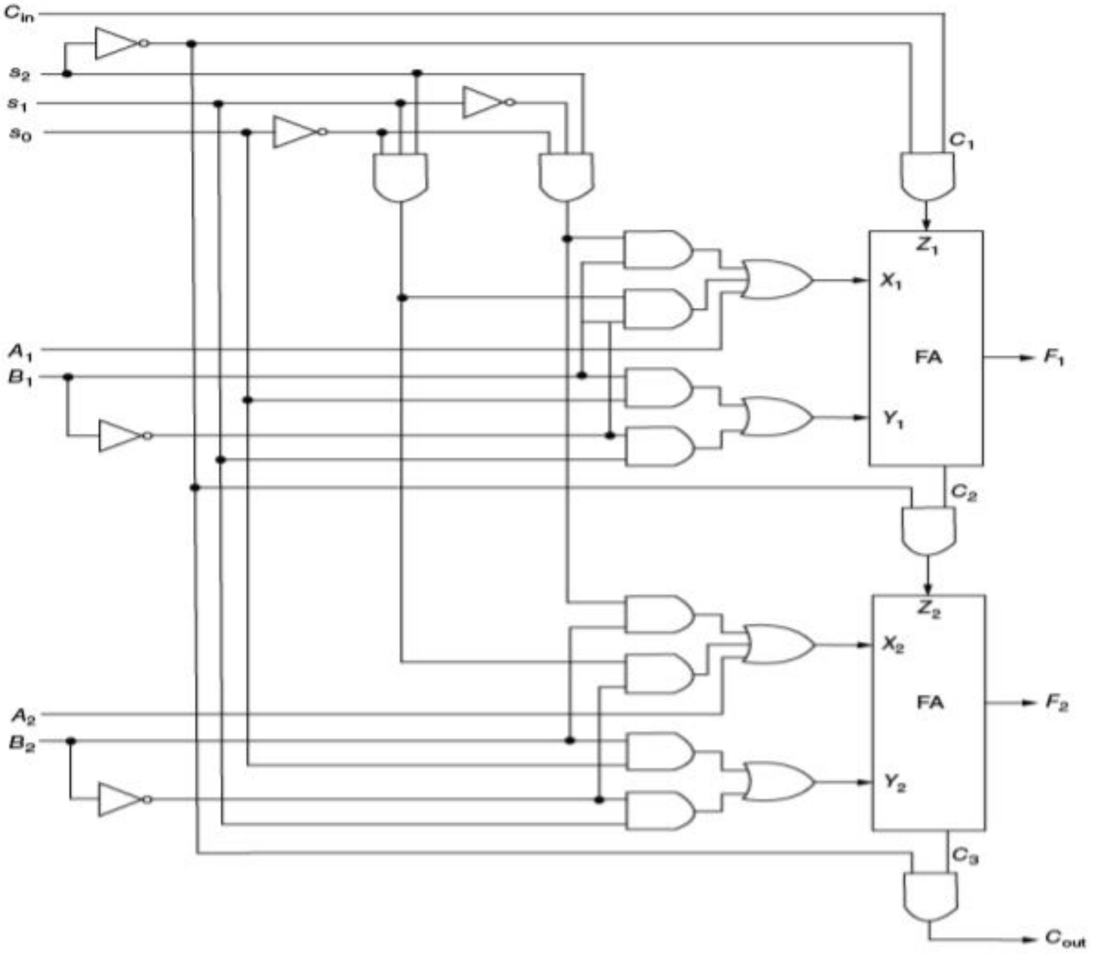
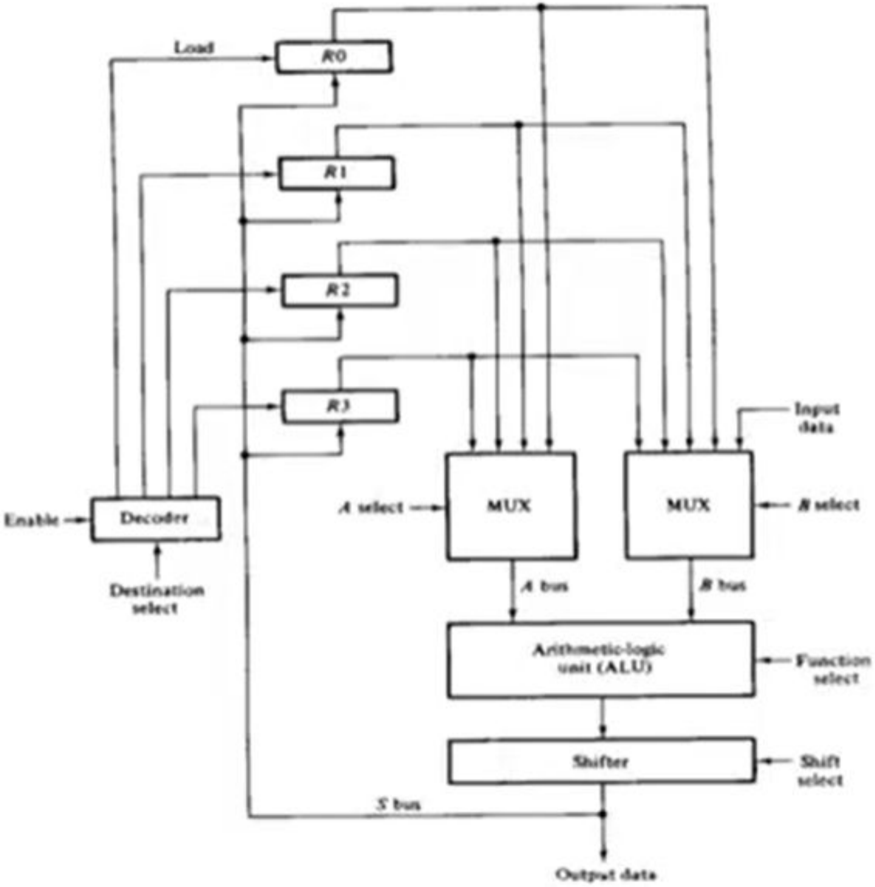
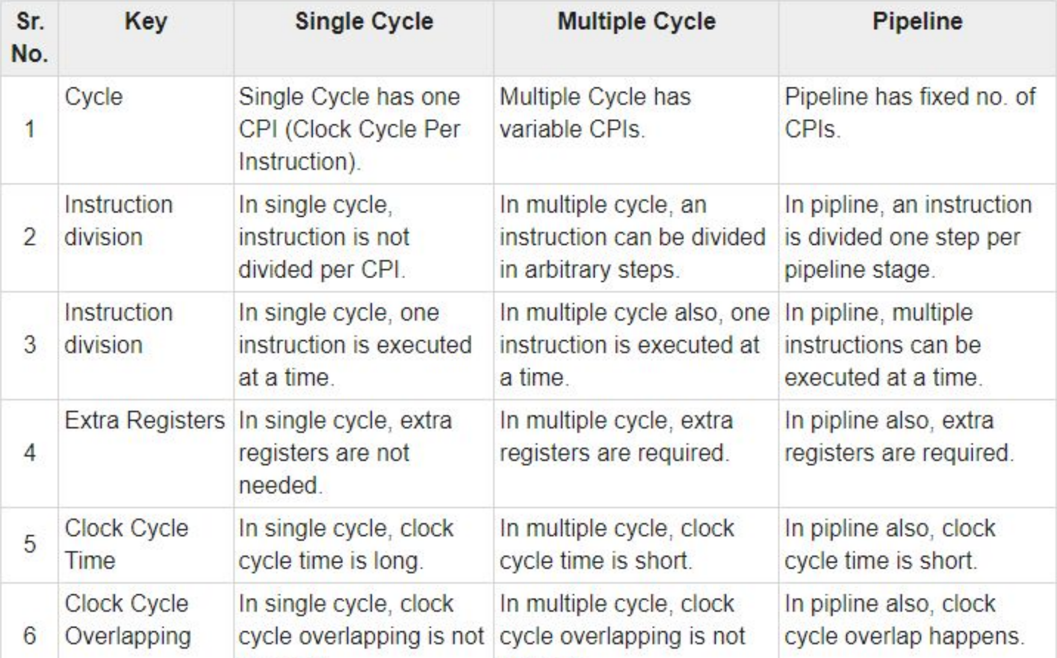


Figure 9-13 Logic diagram of arithmetic logic unit (ALU)

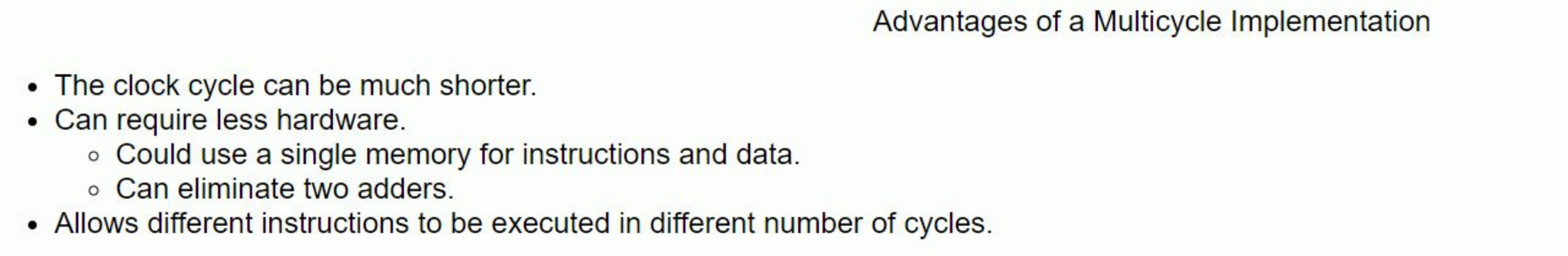
Describe about creation of data path in computer architecture.



Explain the advantage of using multi-cycle data path instead of single-cycle data path.



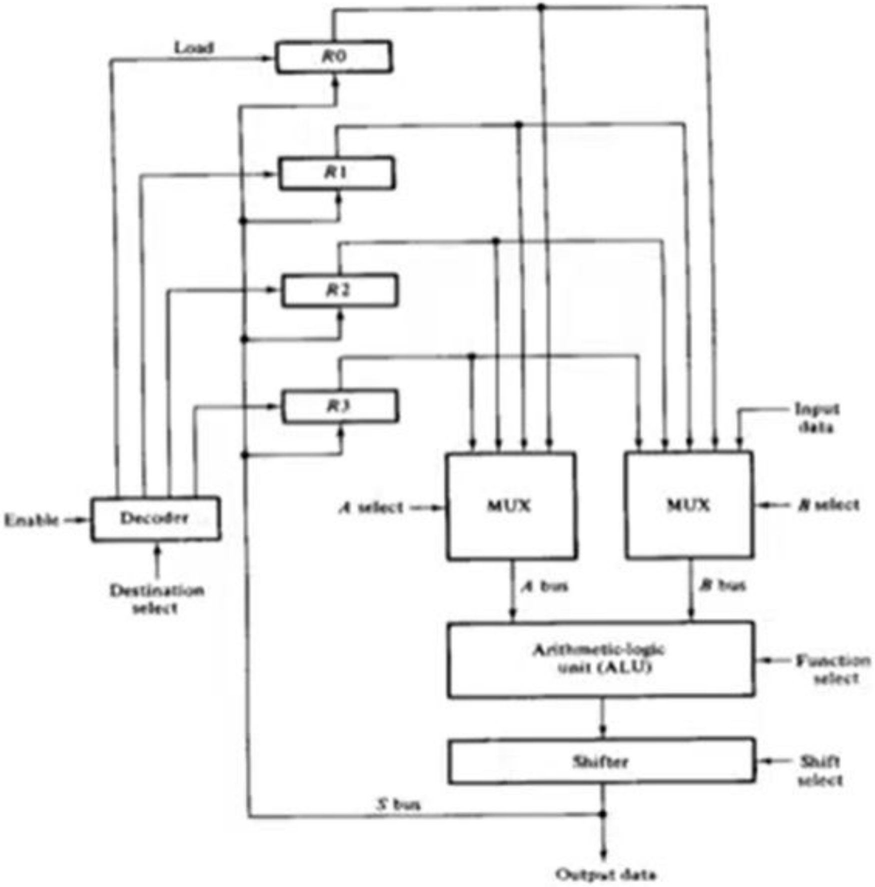
Sr. No.	Key	Single Cycle	Multiple Cycle	Pipeline
1	Cycle	Single Cycle has one CPI (Clock Cycle Per Instruction).	Multiple Cycle has variable CPIs.	Pipeline has fixed no. of CPIs.
2	Instruction division	In single cycle, instruction is not divided per CPI.	In multiple cycle, an instruction can be divided in arbitrary steps.	In pipline, an instruction is divided one step per pipeline stage.
3	Instruction division	In single cycle, one instruction is executed at a time.	In multiple cycle also, one instruction is executed at a time.	In pipline, multiple instructions can be executed at a time.
4	Extra Registers	In single cycle, extra registers are not needed.	In multiple cycle, extra registers are required.	In pipline also, extra registers are required.
5	Clock Cycle Time	In single cycle, clock cycle time is long.	In multiple cycle, clock cycle time is short.	In pipline also, clock cycle time is short.
6	Clock Cycle Overlapping	In single cycle, clock cycle overlapping is not	In multiple cycle, clock cycle overlapping is not	In pipline also, clock cycle overlap happens.



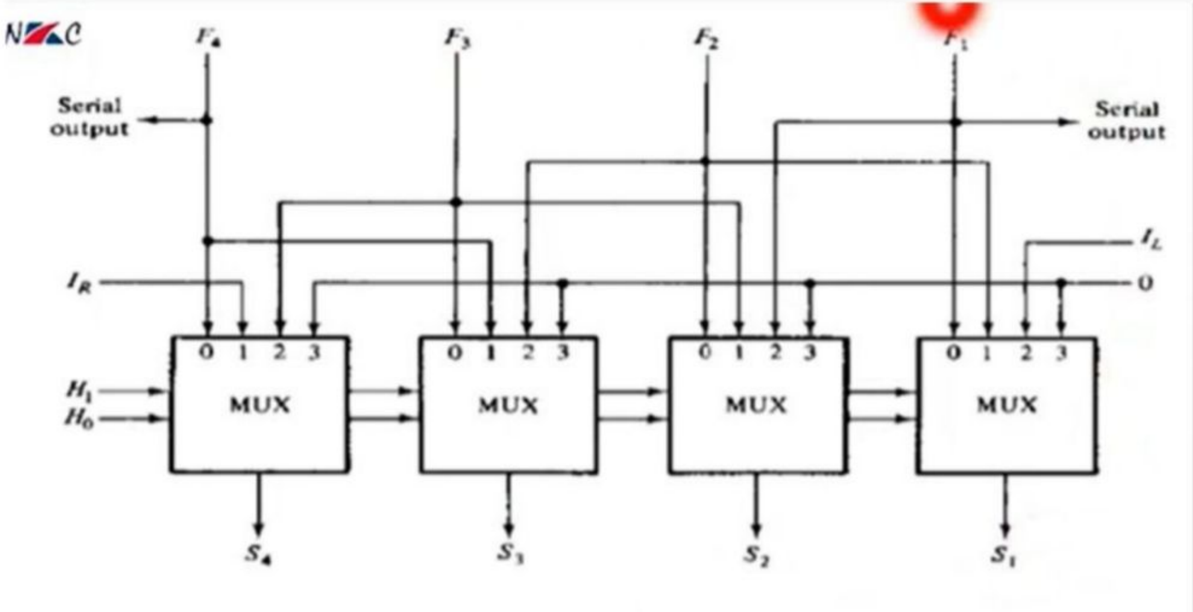
Advantages of a Multicycle Implementation

- The clock cycle can be much shorter.
- Can require less hardware.
    - Could use a single memory for instructions and data.
    - Can eliminate two adders.
- Allows different instructions to be executed in different number of cycles.

Design a bus system for interconnecting four n bit registers



Design a 4bit combinational logic shifter



H_1	H_0	Operation	Function
0	0	$S \leftarrow F$	Transfer F to S (no shift)
0	1	$S \leftarrow$ shr F	Shift-right F into S
1	0	$S \leftarrow$ shl F	Shift-left F into S
1	1	$S \leftarrow 0$	Transfer 0's into S

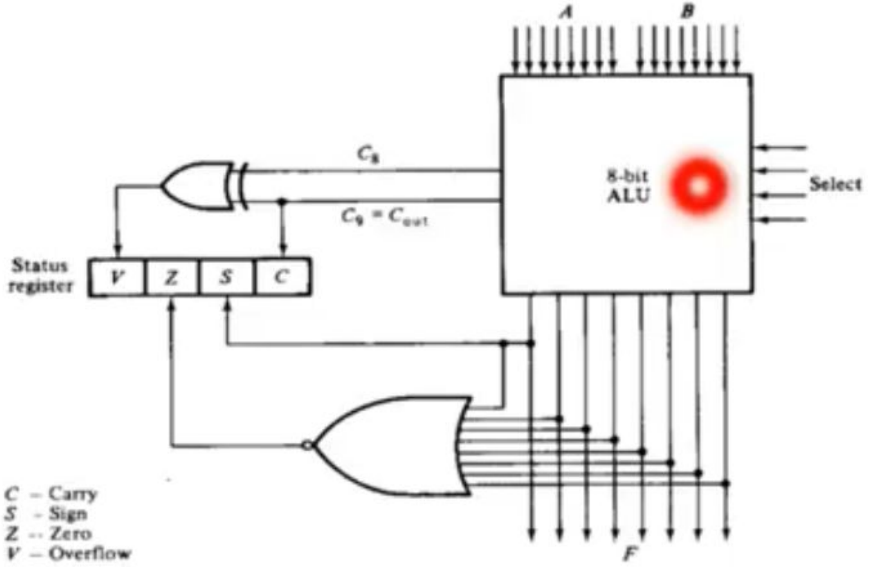
Register R6 is used in a program to point to the top of a stack containing 32-bit numbers. Write a sequence of instructions using the Index, Autoincrement, and Autodecrement addressing modes to perform each of the following tasks:

(*a*) Pop the top two items off the stack, add them, then push the result onto the stack.

(*b*) Copy the fifth item from the top into register R3.

For each case, assume that the stack contains ten or more elements.

With a neat diagram construct a 32-bit ALU.


A
B
8-bit ALU
Select
C_8
$C_9 = C_{out}$
Status register
V
Z
S
C
F
C – Carry
S – Sign
Z – Zero
V – Overflow