x	y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7
0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1

x	y	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1

① $F_0 = 0$ $F \leftarrow 0$ Clear

② $F_1 = xy$ $F \leftarrow A \wedge B$ AND

③ $F_2 = xy'$ $F \leftarrow A \wedge \bar{B}$

④ $f_3 = xy' + xy$

$= x(y' + y)$ $[0+1]$

$= x(1)$

$f_3 = x$ $F \leftarrow A$ Transfer A

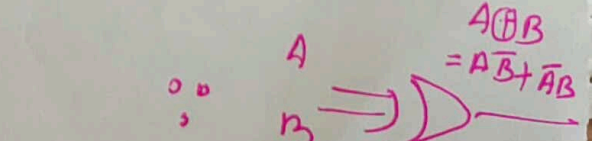
⑤

⑥ $f_5 = x'y + xy$

$= y(x' + x)$

$f_5 = y(1)$

⑦ $f_6 = x'y + xy'$ ∴



$x \oplus y$

Logic Microoperations

→ Logic microoperations specify binary operations for strings of bits stored in registers.

→ These operations consider each bit of the registers separately and treat them as binary variables.

Example:-

$$P_3 \leftarrow R \oplus R_1 \oplus R_2$$

R_1 : 1 0 1 0
R_2 : 1 1 0 0
---
0 1 1 0

R_1 after $P=1$]

This is according to truth table of XOR
But this will be carried out only when $P=1$
whereas P is a control function

There are 16 various Microoperations which can be carried out on to the register

x	y	1
1	0	1

x	y'
1	0

Date _____
Page _____

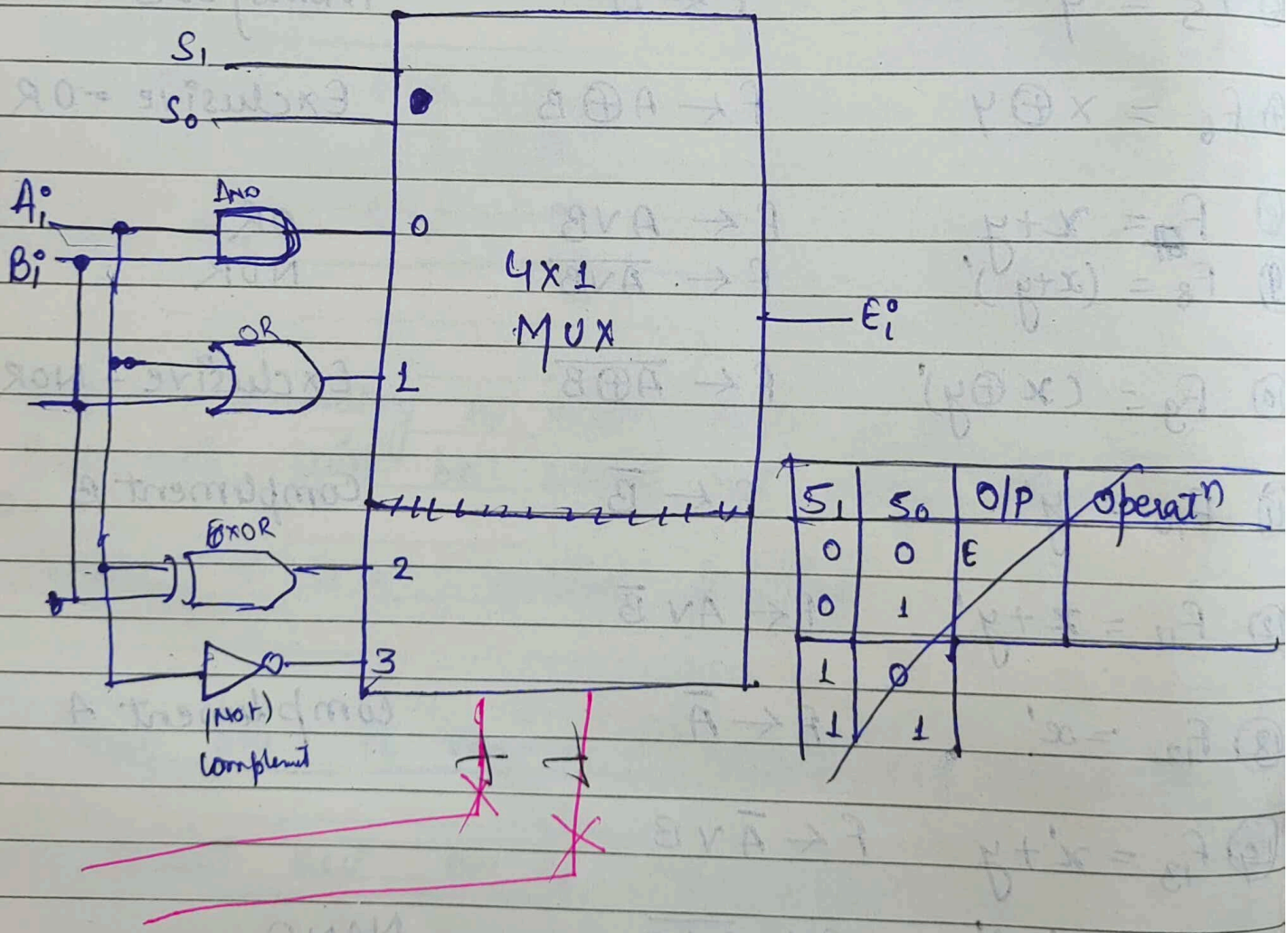
	Boolean function	Microoperation	Name
①	$F_0 = 0$	$F \leftarrow 0$	Clear
②	$F_1 = xy$	$F \leftarrow A \wedge B$	AND
③	$F_2 = xy'$	$F \leftarrow A \wedge \bar{B}$	
④	$F_3 = x$	$F \leftarrow A$	Transfer A
⑤	$F_4 = x'y$	$F \leftarrow \bar{A} \wedge B$	
⑥	$F_5 = y$	$F \leftarrow B$	Transfer B
⑦	$F_6 = x \oplus y$	$F \leftarrow A \oplus B$	Exclusive-OR
⑧	$F_7 = x + y$	$F \leftarrow A \vee B$	OR
⑨	$F_8 = (x+y)'$	$F \leftarrow \overline{A \vee B}$	NOR
⑩	$F_9 = (x \oplus y)'$	$F \leftarrow \overline{A \oplus B}$	Exclusive-NOR
⑪	$F_{10} = y'$	$F \leftarrow \bar{B}$	Complement B
⑫	$F_{11} = x + y'$	$F \leftarrow A \vee \bar{B}$	
⑬	$F_{12} = x'$	$F \leftarrow \bar{A}$	Complement A
⑭	$F_{13} = x' + y$	$F \leftarrow \bar{A} \vee B$	
⑮	$F_{14} = (xy)'$	$F \leftarrow \overline{A \wedge B}$	NAND
⑯	$F_{15} = 1$	$F \leftarrow$ all 1's	Set to all 1's

Basic Hardware Implementation of logic circuit

Where in we'd be considering only 4 operations i.e.,

1. AND
2. OR
3. XOR
4. Complement

For implementation let us take or consider a multiplexer of size 4X1



S_1	S_0	O/P	Operat^n
0	0	E	
0	1		
1	0		
1	1		

S_1	S_0	Output	Operation
0	0	$E = A \wedge B$	AND
0	1	$E = A \vee B$	OR
1	0	$E = A \oplus B$	XOR
1	1	$E = \bar{A}$	Complement

→ This How logic ckt is design
→ Overheare only one state is define

X ~~Application of logic Microoperation~~

MSb: Most Significant bit
LSb: Least Significant bit

Shift Micro operation

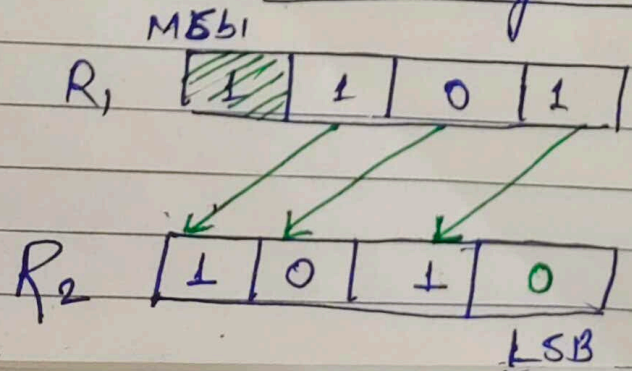
- Shift Microoperation is used for serial transfer of data.
- Used in conjunction with arithmetic, logic and other data processing operations.
- The content of the register can be shifted to the left or the right.
- The first flip-flop receives its binary information from the serial input. (either onto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

Types of shift

(1) logical shift

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

Shl - logical shift left.



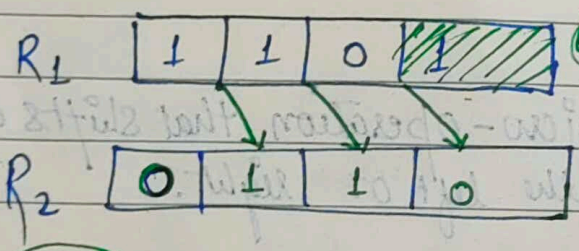
Now the Que arises for the 2 position what about MSb 1 and LSb position

Note:

→ So, the MSb would be discarded

→ And, through as per the defination of logical shift 0 would be transferred to through serial input over here, which would zero (0).

Shr - logical shift right



As per the defination of logical shift 0 would be transferred through serial input over here, which would be zero (0).

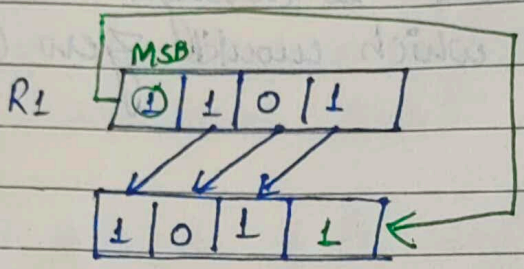
Slit variation to this logical shift right instead of providing zero to it as a serial input we are having second type of shift that is a circular shift which perform the rotation operation.

② Circular shift :-

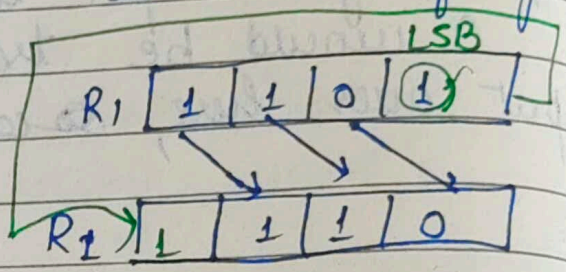
- A circular shift (also known as a rotate operation) circulates the bits of the register around the two ends without loss of information.

• This is accomplished by connecting the serial output of the shift register to its serial input

cil - circular shift left



cir - circular shift right



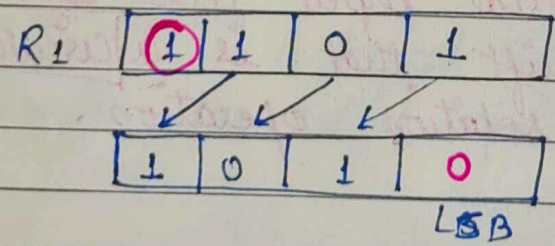
③ Arithmetic shift

• An arithmetic shift is a micro-operation that shifts a signed binary number to the left or right.
(+ve & -ve info)

• An arithmetic shift-left multiplies a signed binary number by 2.

• An arithmetic shift-right divides the number by 2.

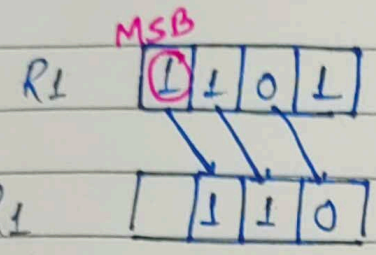
ashl - arithmetic shift left



Her MSB signifies sign means 1 means a -ve no.

Here LSB would be zero

ashr- arithmetic shift right



In arithmetic shift right the content of MSB would be copied directly to the MSB itself there won't be change in that because it is shift right operation

Note

Hence, shift left is our normal logical shift left but in arithmetic shift right we need to check for the MSB. MSB should not change. because it is a sign and sign should not change.

HARDWARE IMPLEMENTION OF THIS PARTICULAR SHIFTER OF CIRCUIT

4-bit combinational circuit shifter