Common bus system for 4 registers

Table shows the register that is selected by the bus for each of the four possible binary value of the selection lines

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

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

$R3 \leftarrow R1 \oplus R2$

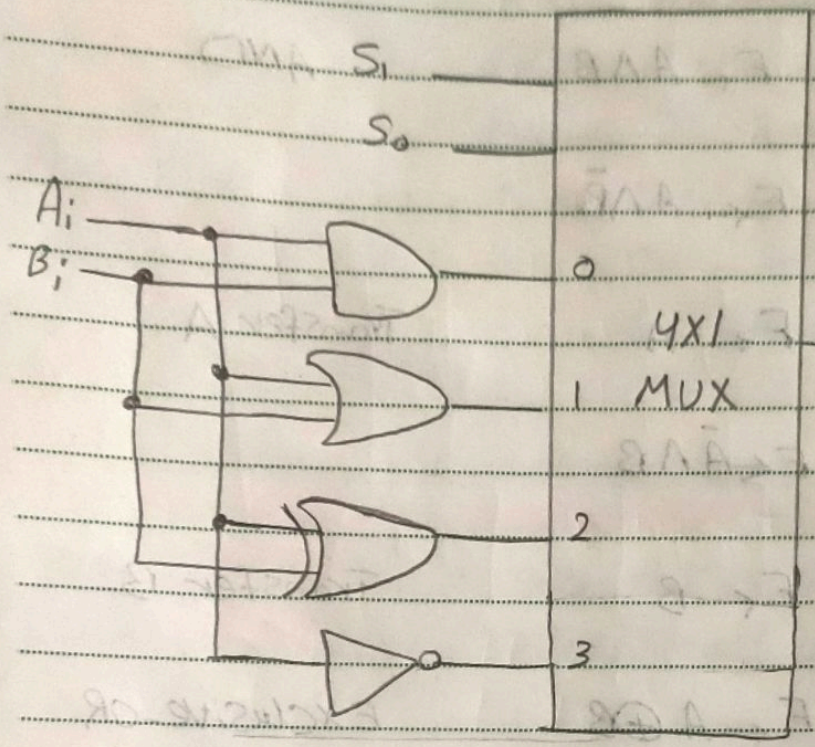
R1	1	0	1	0
R2 $\oplus$	1	1	0	0
R1 after R=1	0	1	1	0

متشابهين 0 ، متكاملين 1

Date: Page:

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

Hardware Imp of logic circuit



S_1	S_2	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

Applications of logic Microoperations. 10 applications

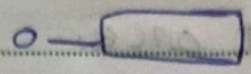
A, B are registers, B is bit data that will be used to modify the contents of A

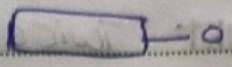
selective-set	$A \leftarrow A + B$	Clear	$A \leftarrow A \oplus B$
selective-complement	$A \leftarrow A \oplus B$	Insert	$A \leftarrow (A \cdot B) + C$
selective-clear	$A \leftarrow A \cdot B'$	Compare	$A \leftarrow A \oplus B$
Mask (delete)	$A \leftarrow A \cdot B$		

Date: Page:

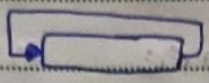
Shift Micro operations

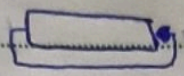
① Logical Shift

Right →  يتم إضافة صفرًا

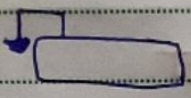
left → 

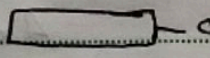
② Circular Shift

Right →  يتم أخذ الرقم الموضوع بالطرف المراد

left →  التشفيت لها ووضعها بالطرف الآخر

③ Arithmetic shift

Right →  - يتم تكرار الرقم بالطرف العاكس المراد التشفيت لها.

left →  - يتم وضع صفرًا.

ex:- R = 0 1 0 1

1- logical shift Right ⇒ 0 0 1 0

" " left ⇒ 1 0 1 0

2- Circular shift Right ⇒ 1 0 1 0

" " left ⇒ 1 0 1 0

3- Arithmetic shift Right ⇒ 0 0 1 0

" " left ⇒ 1 0 1 0

Date: Page:

ex: R1 = 11 00 1110 , Then:

- Arithmetic shift right once: 1111 0011
- ~ ~ right twice 1111 0011
- ~ ~ left once 1001 1100
- ~ ~ ~~right~~ left twice 0011 1000
- logical ~ right once 0110 0111
- ~ ~ left ~ 1001 1100
- Circular ~ right ~ 0110 0111
- ~ ~ left ~ 1001 1101

---

uses of logic — selective set

A_t 1100 — $A \leftarrow (A + B)$

$+ B$ 1010

A_{t+1} 1110 → OR

---

selective complement — $A \leftarrow (A \oplus B)$

A_t 1100 — 0 المتغايرين

$\oplus B$ 1010 — 1 المتغيرين

A_{t+1} 0110 → XOR

---

selective clear — $A \leftarrow (A \cdot B')$

A_t 1100 — B: 1010

$\cdot B'$ 0101 — B': 0101

A_{t+1} 0100 → $A \wedge \bar{B}$

MASK operation — $A \leftarrow (A \cdot B)$

A_t 1100

$\cdot B$ 1010

A_{t+1} 1000 →

Date: Page:

Clear operation

A_t 1100

B 1010

A_{t+1} 0110

$A \leftarrow A \oplus B$

0 متشابهين

1 متنافرين

Insert operation

→ A Mask operation to → Clear the desired bit Positions.

→ An OR ~ to → introduce the new bits into the desired Position

1101	1000	1011	0001	A(original)
1111	1111	1111	0000	MASK
1101	1000	1011	0000	A(Intermediate)
0000	0000	0000	1010	Added bits
1101	1000	1011	1010	A(Desired)

Show The structure of Arithmetic logic Shift unit page 22 part 2