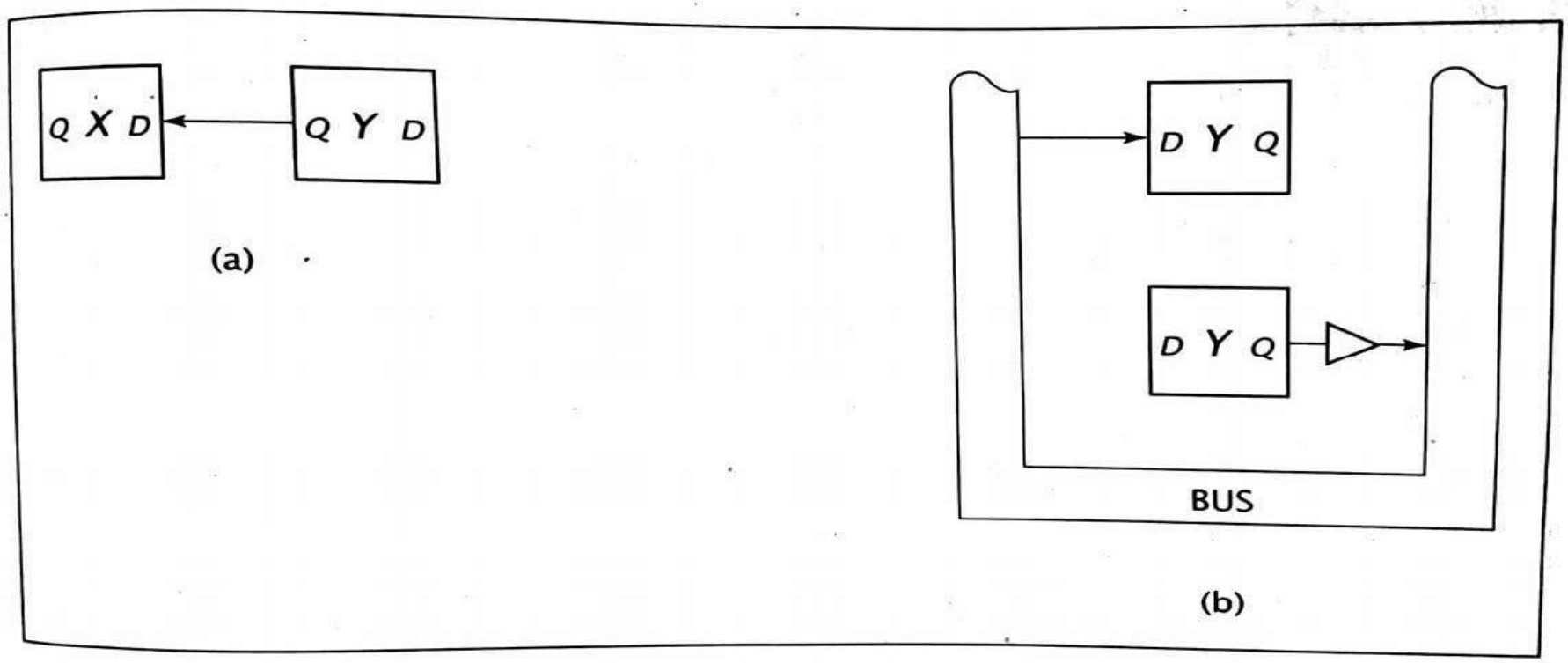
CHAPTER-3

RTL and HDL

Consider a digital system with two 1-bit registers, X and Y. The μop that copies the contents of register Y to register X can be expressed as X←Y.

Implementations of the micro-operation X←Y using (a) a direct connection and (b) a bus connection



Contd…

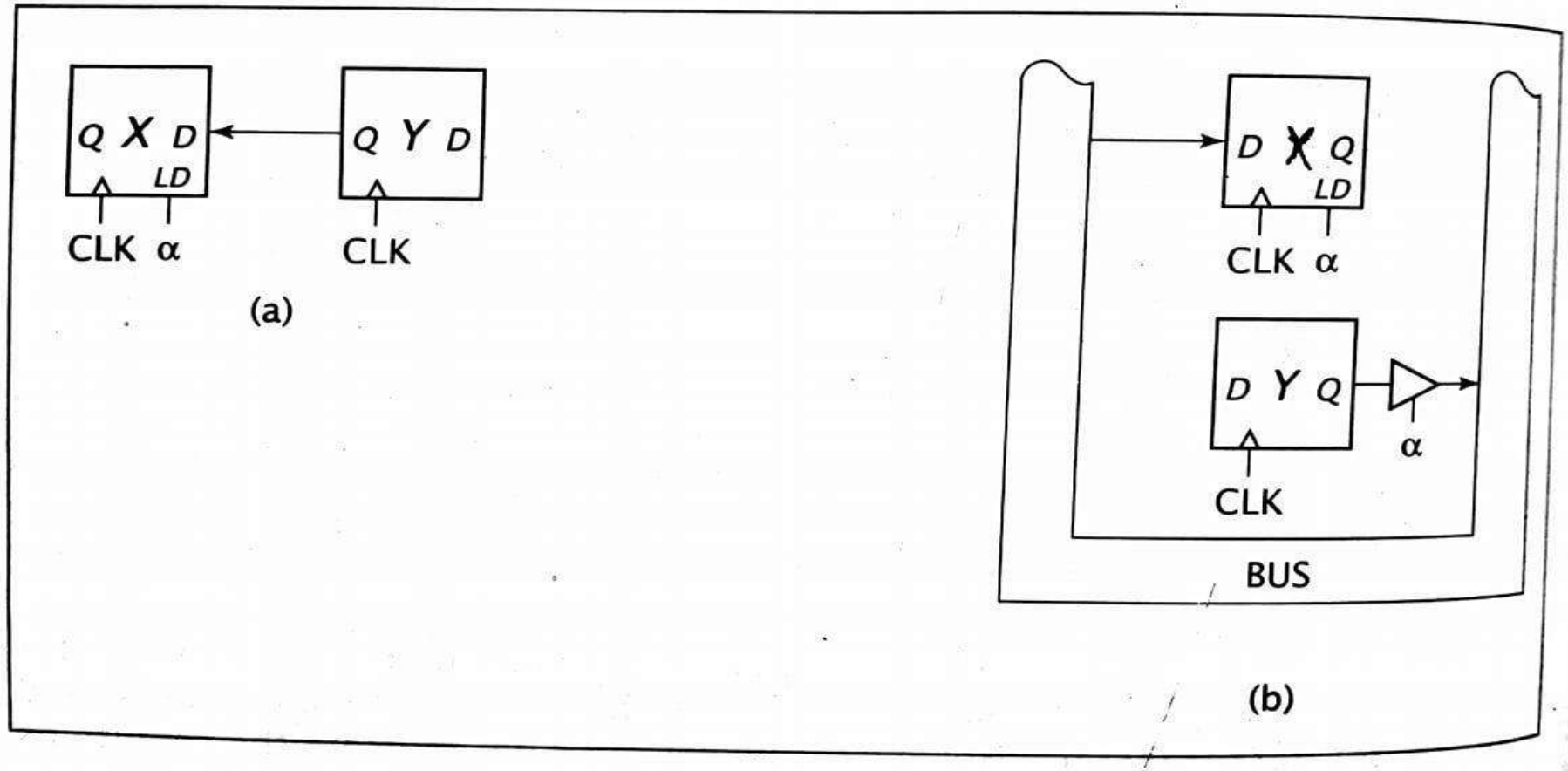
- Both designs provide a path for data to flow from register Y to register X, but neither specifies when X should load this data.
- Assume that the transfer should occur when control input α is high.

$$\alpha: X \leftarrow Y$$

- When all conditions to the left of the colon are asserted, the data transfers specified by the μops are performed.
- α is used to load register X and, in the bus-based implementation, to enable the tri-state buffer so that the contents of register Y are placed on the bus.

Contd...

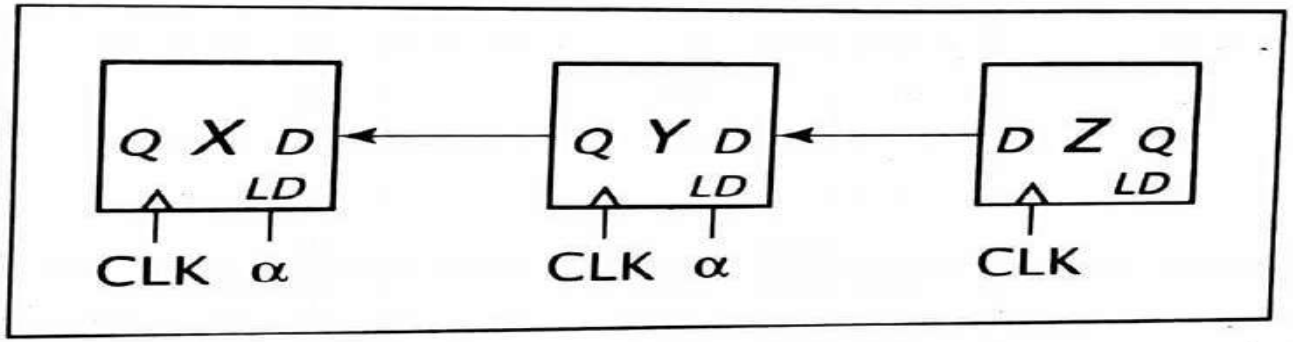
Implementations of the data transfer α: $X \leftarrow Y$ with control signals: (a) with direct path, and (b) using a bus



Contd…

- One way to improve system performance is to perform two or more µops simultaneously.
- The µops are separated by commas; the order in which they are written is unimportant because they are performed concurrently.
- Consider: α: X←Y, Y←Z or α: Y←Z, X←Y
  If X=0, Y=1 and Z=0 just before α becomes 1, these µops set X=1(the original value of Y) and Y=0.
- Note that a single bus cannot be used here because a bus can hold only one value at a time.
- When α=1, both Y and Z must travel on the data paths simultaneously.

Implementation of the data transfer α: X←Y, Y←Z

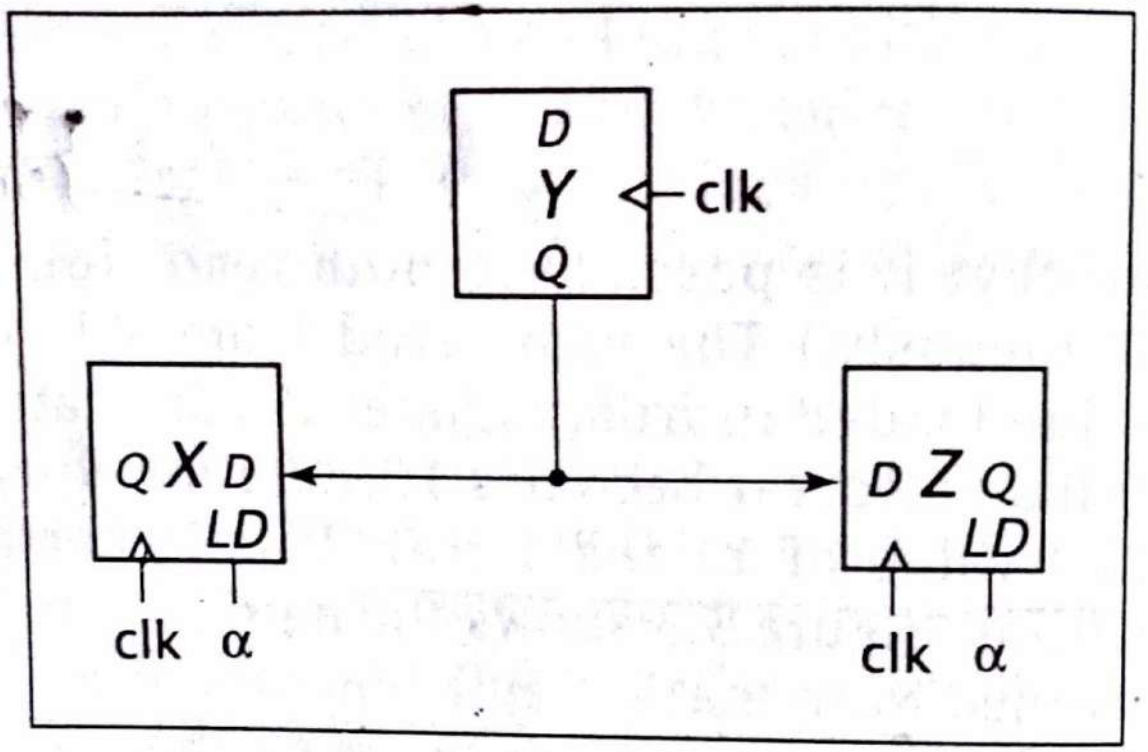


Consider the transfers that occur when α = 1.

α: X←Y, Z←Y

Register Y can be read by many other registers simultaneously; both micro operations can be performed concurrently.

Implementation of the data transfer α: X←Y, Z←Y



Contd…

- Sometimes it is necessary to move a constant value into a register, rather than data from another register.

  α: $X \leftarrow 0$

  β: $X \leftarrow 1$

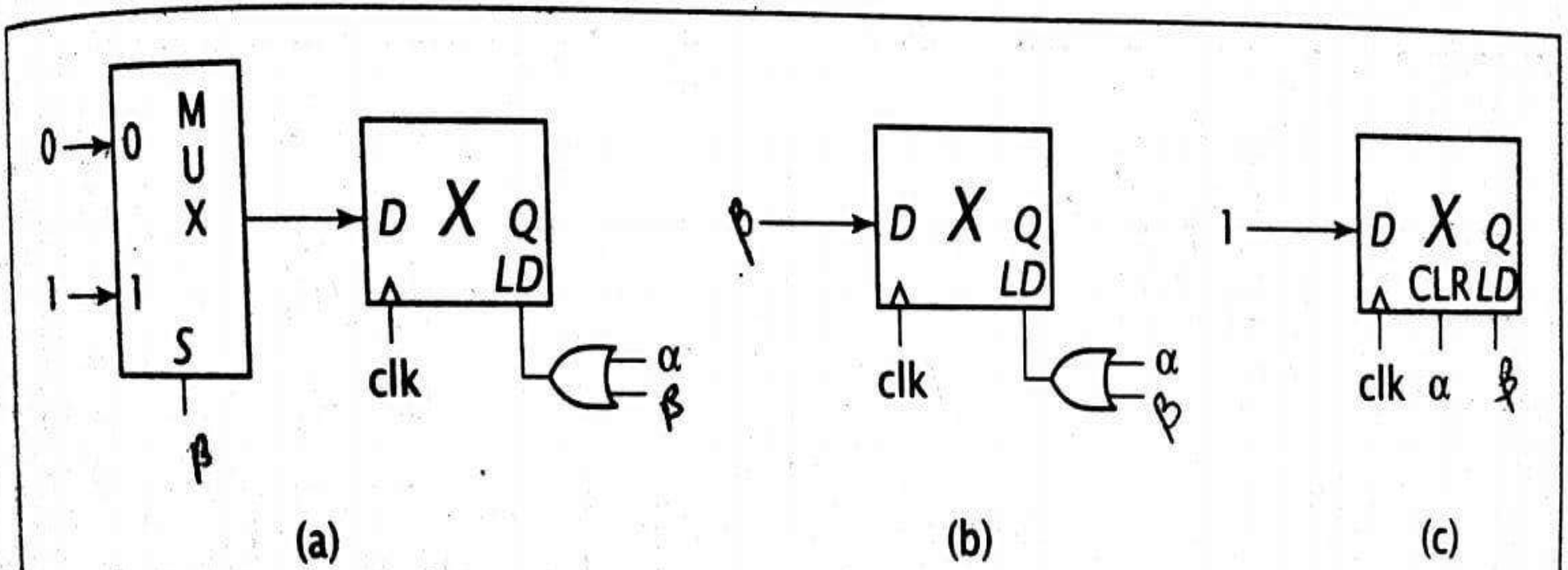
- The conditions can be modified so that they are mutually exclusive.

$\alpha\beta'$: $X \leftarrow 0$	α : $X \leftarrow 0$	$\alpha\beta'$: $X \leftarrow 0$
β : $X \leftarrow 1$	$\alpha'\beta$: $X \leftarrow 1$	$\alpha'\beta$: $X \leftarrow 1$

- In the first, X is set to 1 when both α and β are 1.
- In the second, X is set to 0.
- In the third set, neither condition is met and the value of X is not changed.

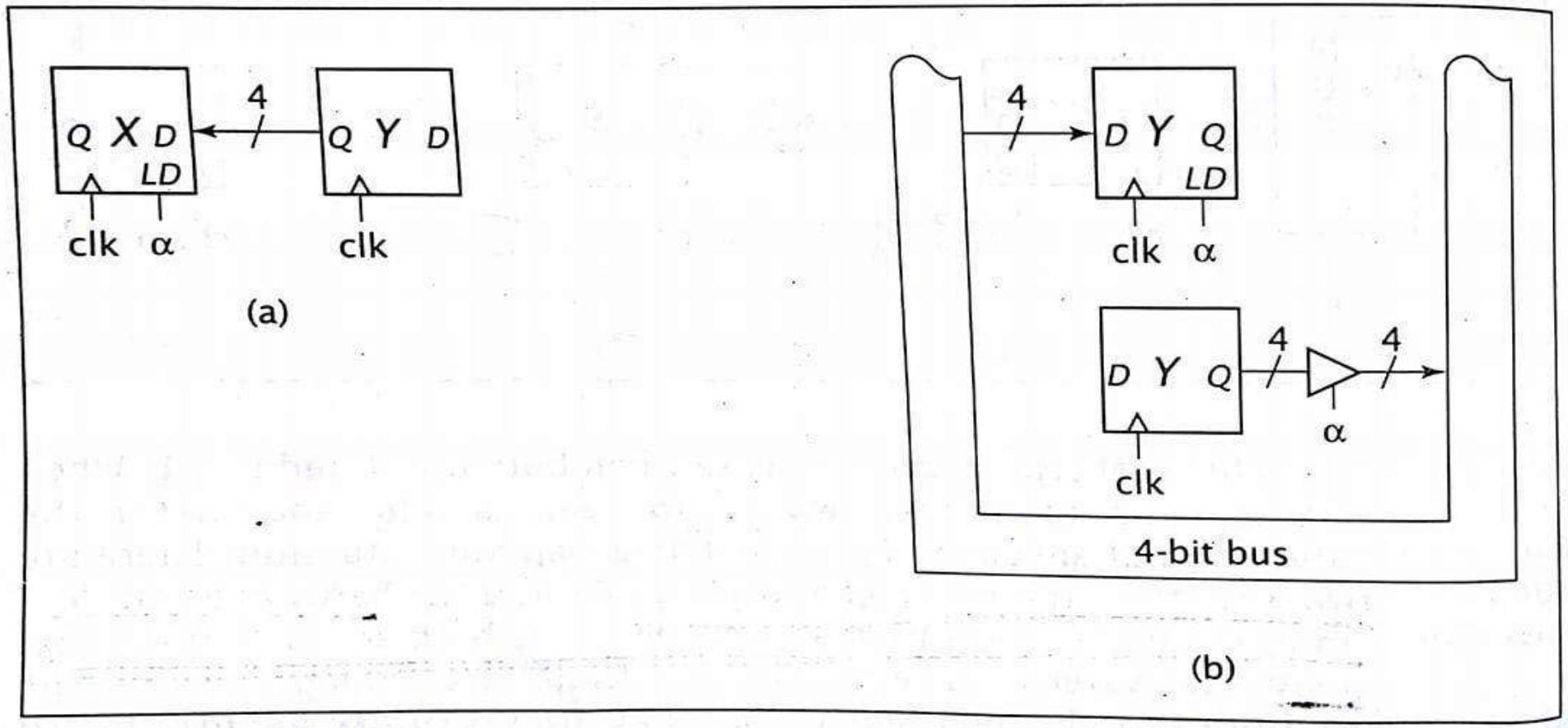
Contd…

Three implementations of the data transfers α: $X \leftarrow 0$ and β: $X \leftarrow 1$: (a) using a multiplexer to select the data input, (b) using β as the data input, and (c) using the CLR signal



Contd…

Implementations of the 4-bit data transfer α: $X \leftarrow Y$: (a) using a direct connection, and (b) using a bus



Arithmetic and logical + shift micro operations:

Arithmetic and logical micro-operations

Operation	Example
Add	$X \leftarrow X + Y$
Subtract	$X \leftarrow X - Y$ or $X \leftarrow X + Y' + 1$
Increment	$X \leftarrow X + 1$
Decrement	$X \leftarrow X - 1$
AND	$X \leftarrow X \wedge Y$ or $X \leftarrow XY$
OR	$X \leftarrow X \vee Y$
XOR	$X \leftarrow X \oplus Y$
NOT	$X \leftarrow /X$ or $X \leftarrow X'$

Shift micro-operations

Operation	Notation
Linear shift left	shl(X)
Linear shift right	shr(X)
Circular shift left	cil(X)
Circular shift right	cir(X)
Arithmetic shift left	ashl(X)
Arithmetic shift right	ashr(X)
Decimal shift left	dshl(X)
Decimal shift right	dshr(X)

Specification and implementation of simple systems:

- Assume that conditions j, o, h and n are mutually exclusive.

  j : M ←A

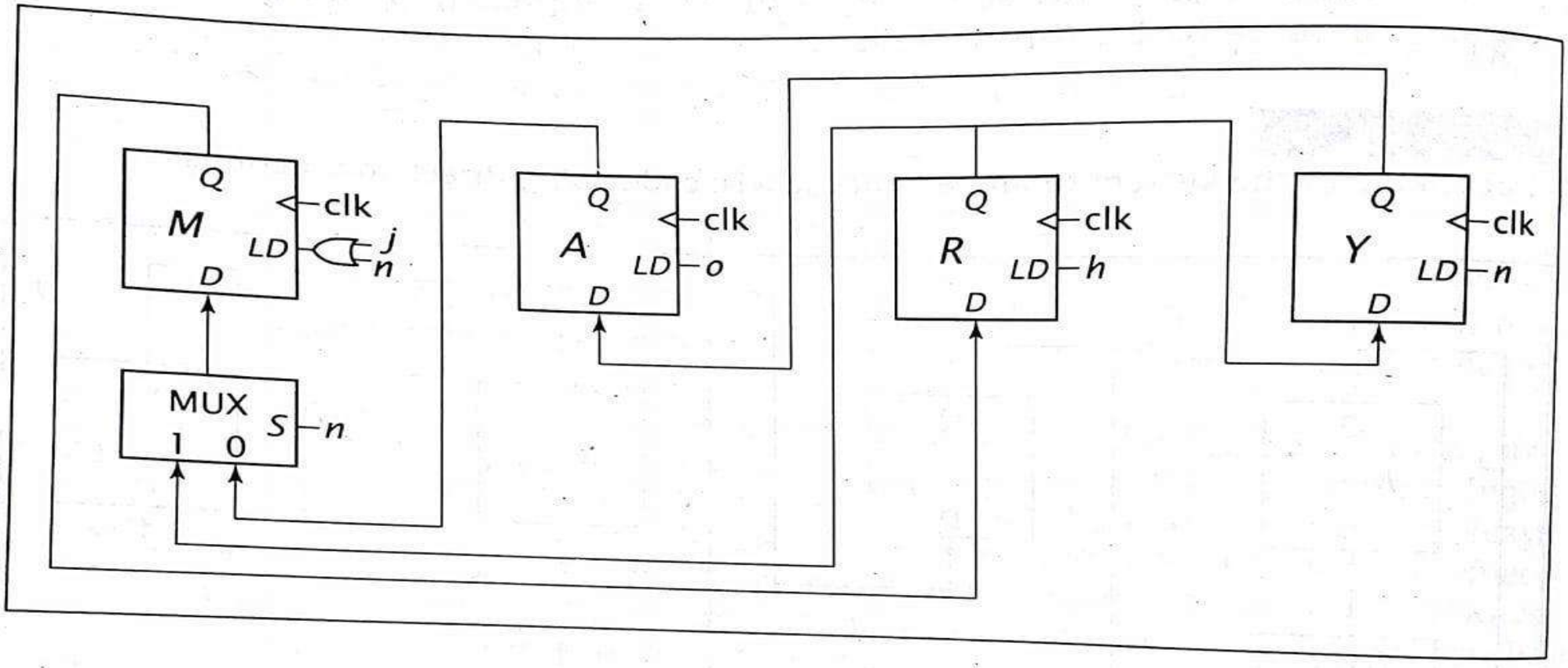
  o: A ←Y

  h: R ←M

  n: Y ←R, M ←R

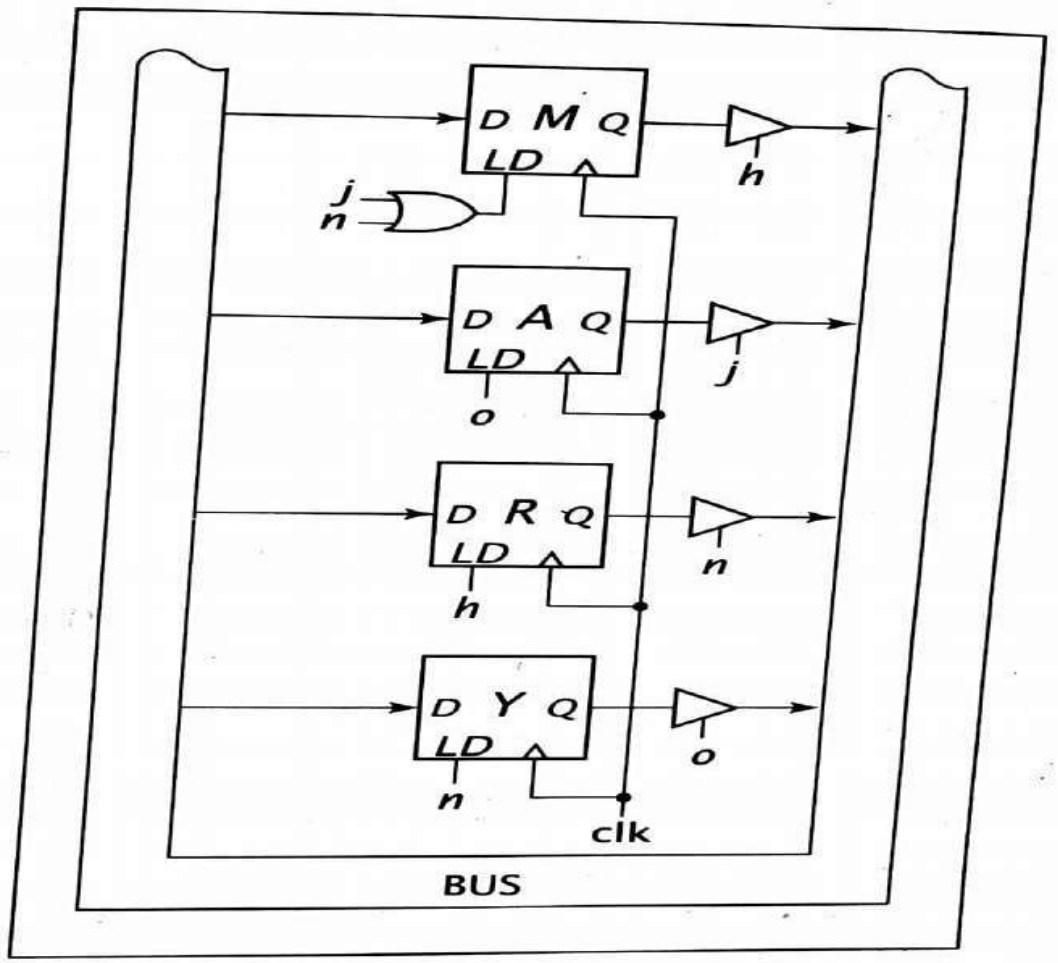
Contd...

Complete design of the system to implement the RTL code using direct connections



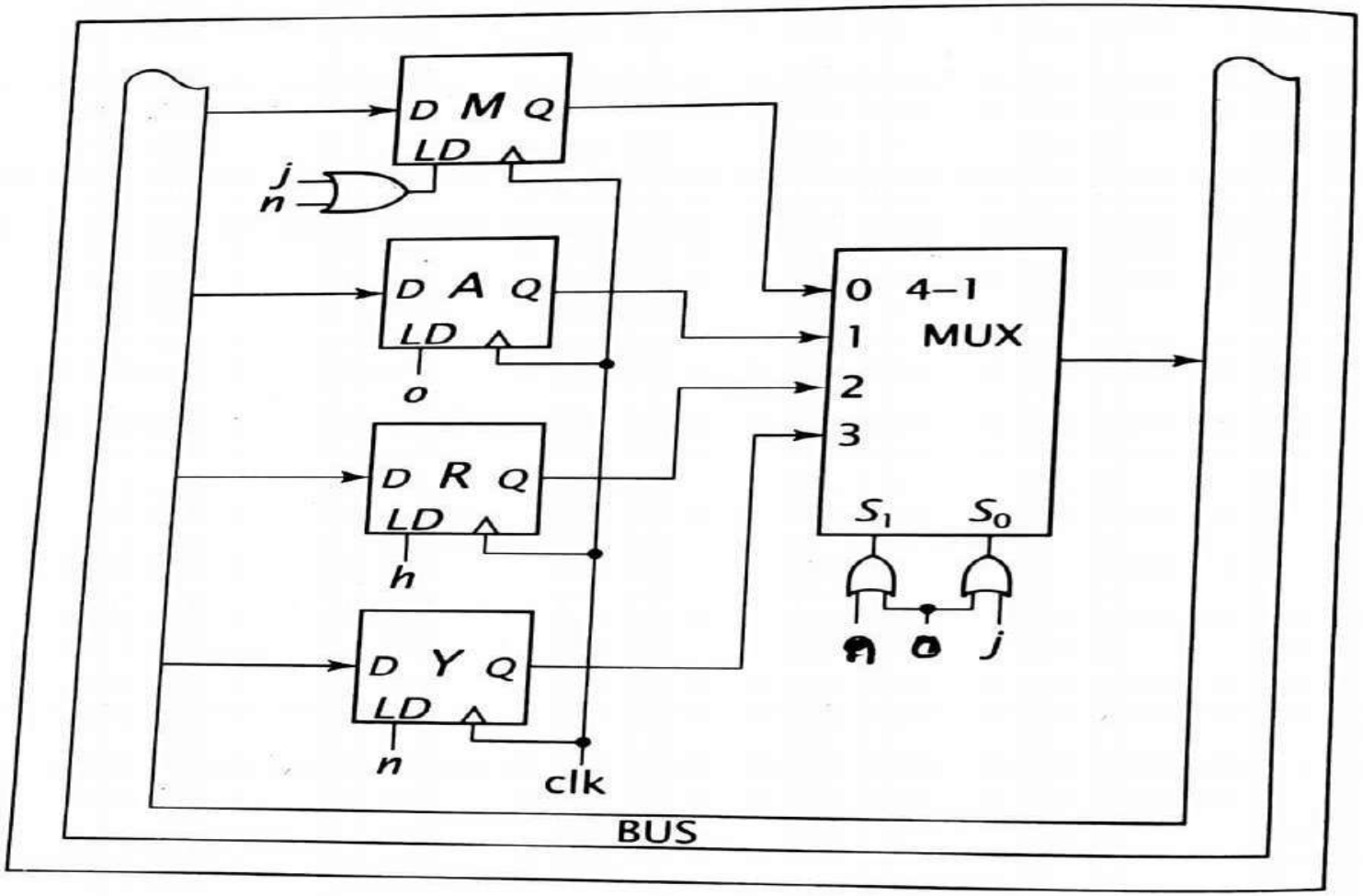
Contd…

Complete design of the system to implement the RTL code using a bus and tri-state buffers



Contd...

Complete design of the system to implement the RTL code using a bus and multiplexer



Modulo 6 Counter

State table for the modulo 6 counter

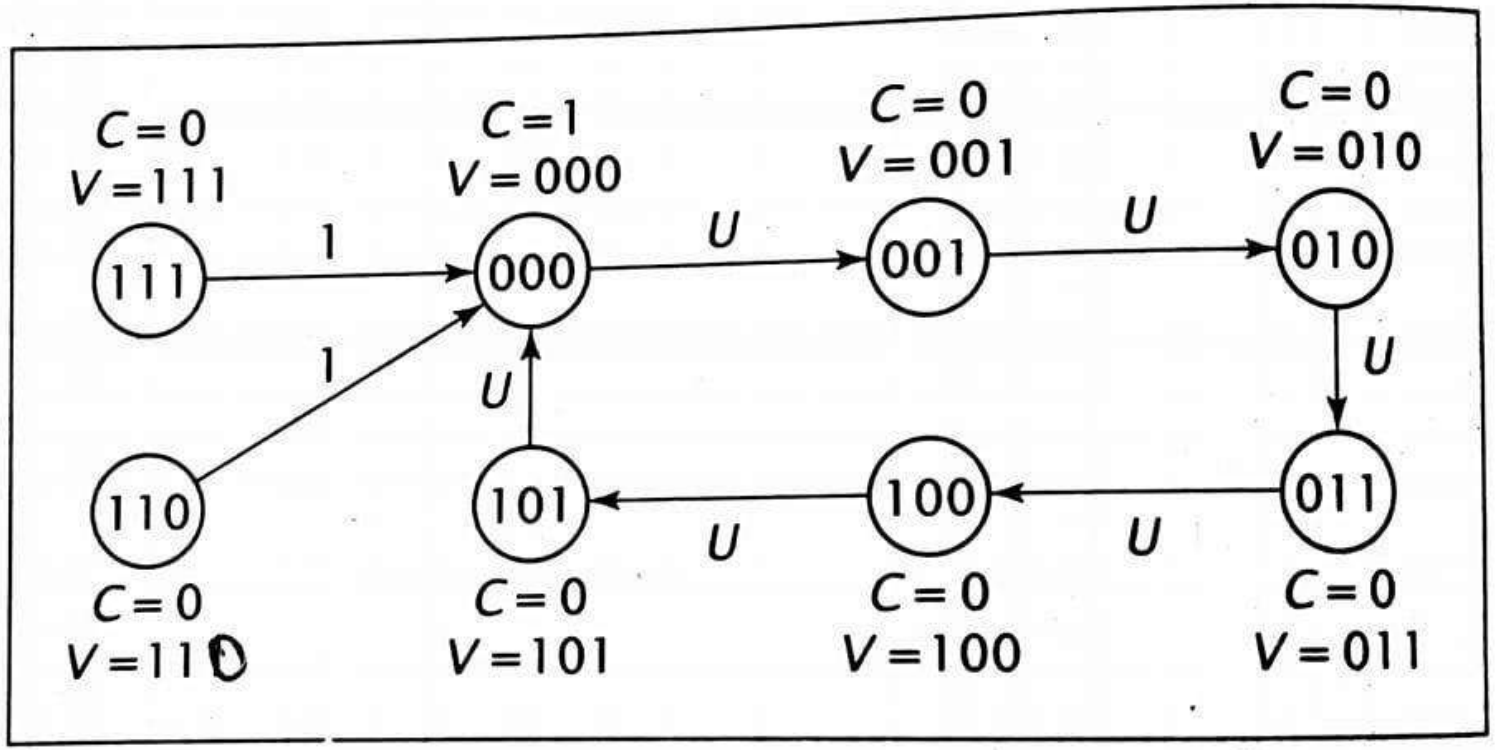
Present State	U	Next State	C	$V_2V_1V_0$
S_0	0	S_0	1	000
S_0	1	S_1	0	001
S_1	0	S_1	0	001
S_1	1	S_2	0	010
S_2	0	S_2	0	010
S_2	1	S_3	0	011
S_3	0	S_3	0	011
S_3	1	S_4	0	100
S_4	0	S_4	0	100
S_4	1	S_5	0	101
S_5	0	S_5	0	101
S_5	1	S_0	1	000
S_6	X	S_0	1	000
S_7	X	S_0	1	000

The behavior of the modulo 6 counter can be expressed by :

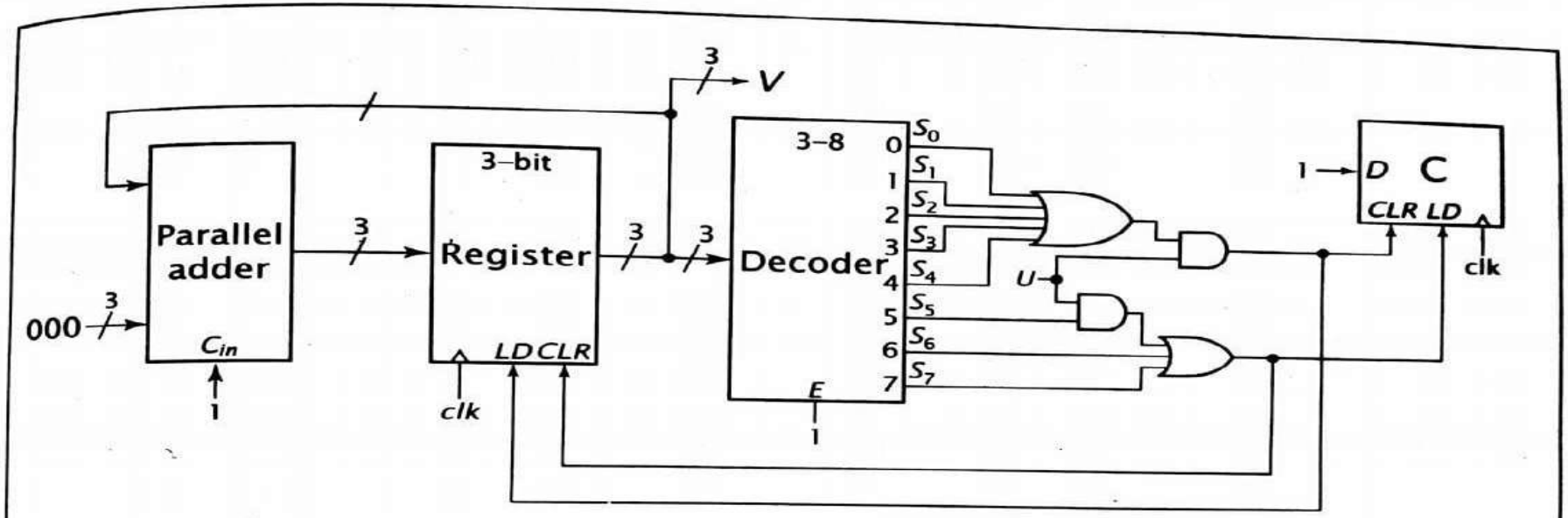
$$(S_0+S_1+S_2+S_3+S_4)U \quad : V \leftarrow V+1,\ C \leftarrow 0$$

$$S_5U+S_6+S_7 \quad : V \leftarrow 0,\ C \leftarrow 1$$

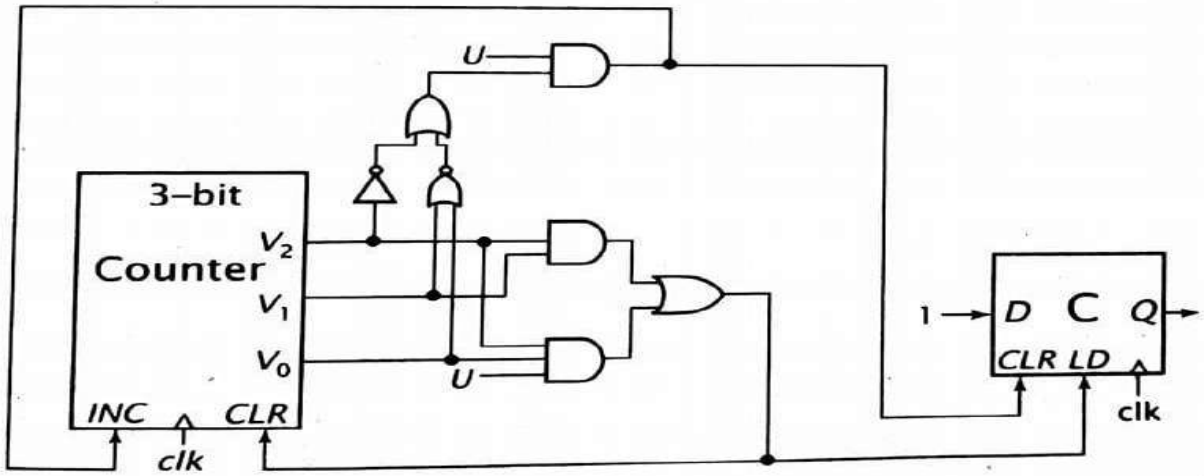
State diagram for the modulo 6 counter



Two implementations of the RTL code for the modulo 6 counter: (a) using a register, and (b) using a counter

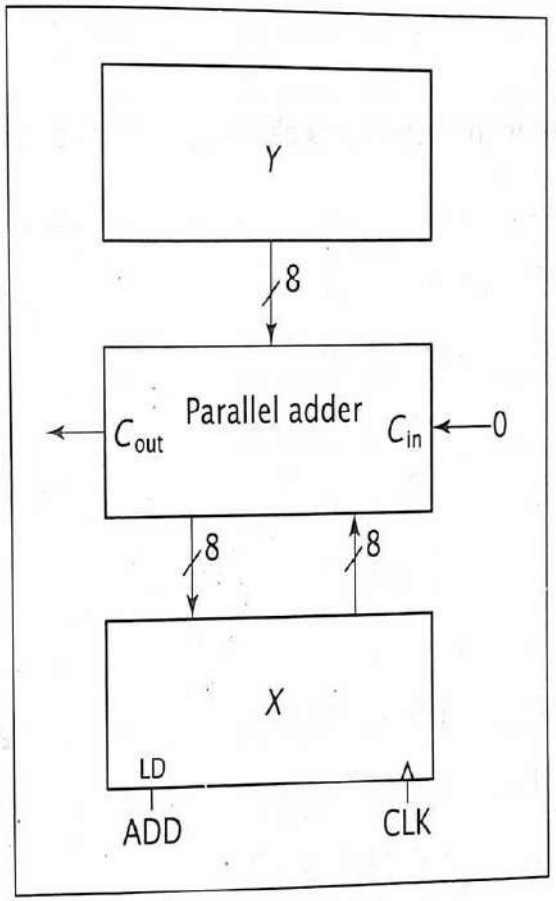


(a)

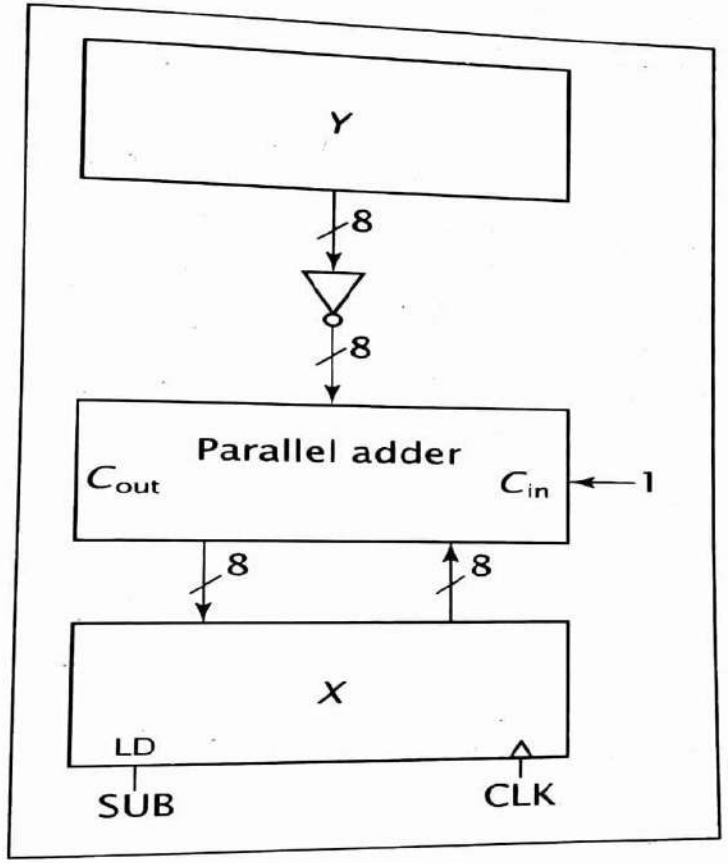


(b)

Implementation of the micro-operation $X \leftarrow X + Y$



Implementation of the micro-operation $X \leftarrow X - Y$



1. Write valid RTL statements that realize the following transitions. All registers are 1-bit wide.
   a. IF α = 1 THEN copy X to W and copy Z to Y
   b. IF α = 1 THEN copy X to W; otherwise copy Z to Y
   c. IF α = 0 THEN copy X to W
2. show the hardware to implement the RTL statements developed for qsn 1.

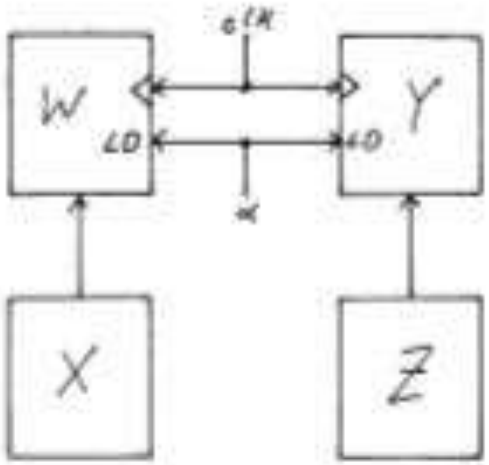
1. a) α: $W \square X$, $Y \square Z$

   b) α: $W \square X$

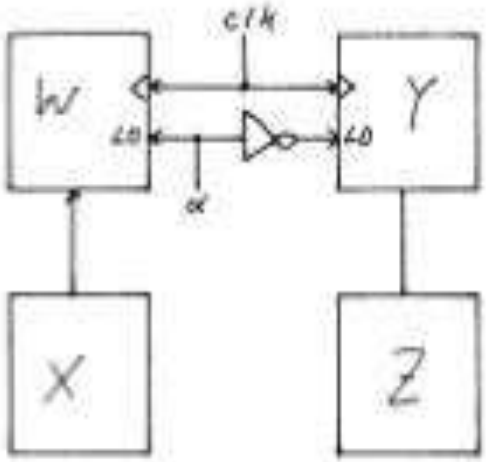
      α′: $Y \square Z$

   c) α′: $W \square X$

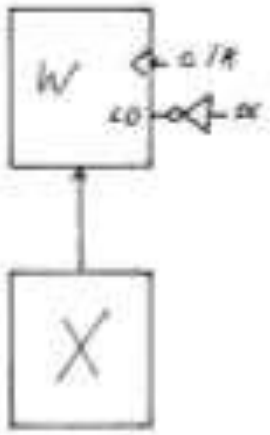
2. a)



b)



c)



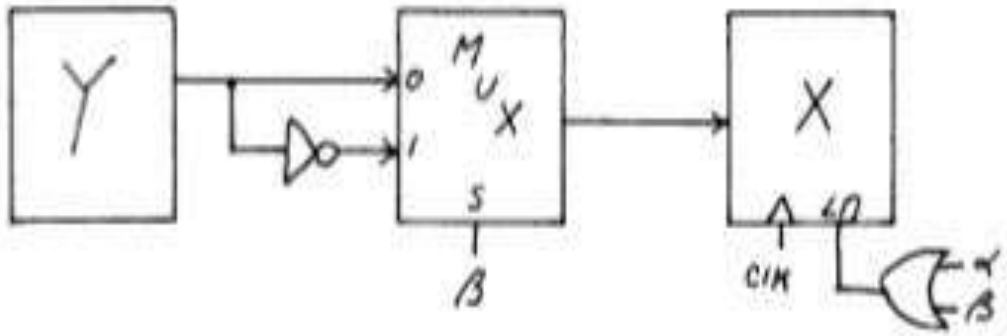
3. Write the RTL code for the following transitions. All registers are 1-bit wide. Signals α and β are never equal to 1 simultaneously.

   a. IF α = 1 THEN copy Y to X
      IF β = 1 THEN copy Y' to X
   b. IF α = 1 THEN set X to 0
      IF β = 1 THEN set X to X‘

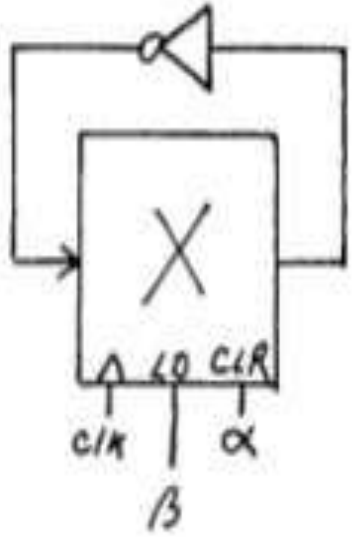
4. show the hardware to implement the RTL statements developed for problem 3.

3. a) α: $X \square Y$
   β: $X \square Y'$
   b) α: $X \square 0$
   β: $X \square X'$

4. a)



b)

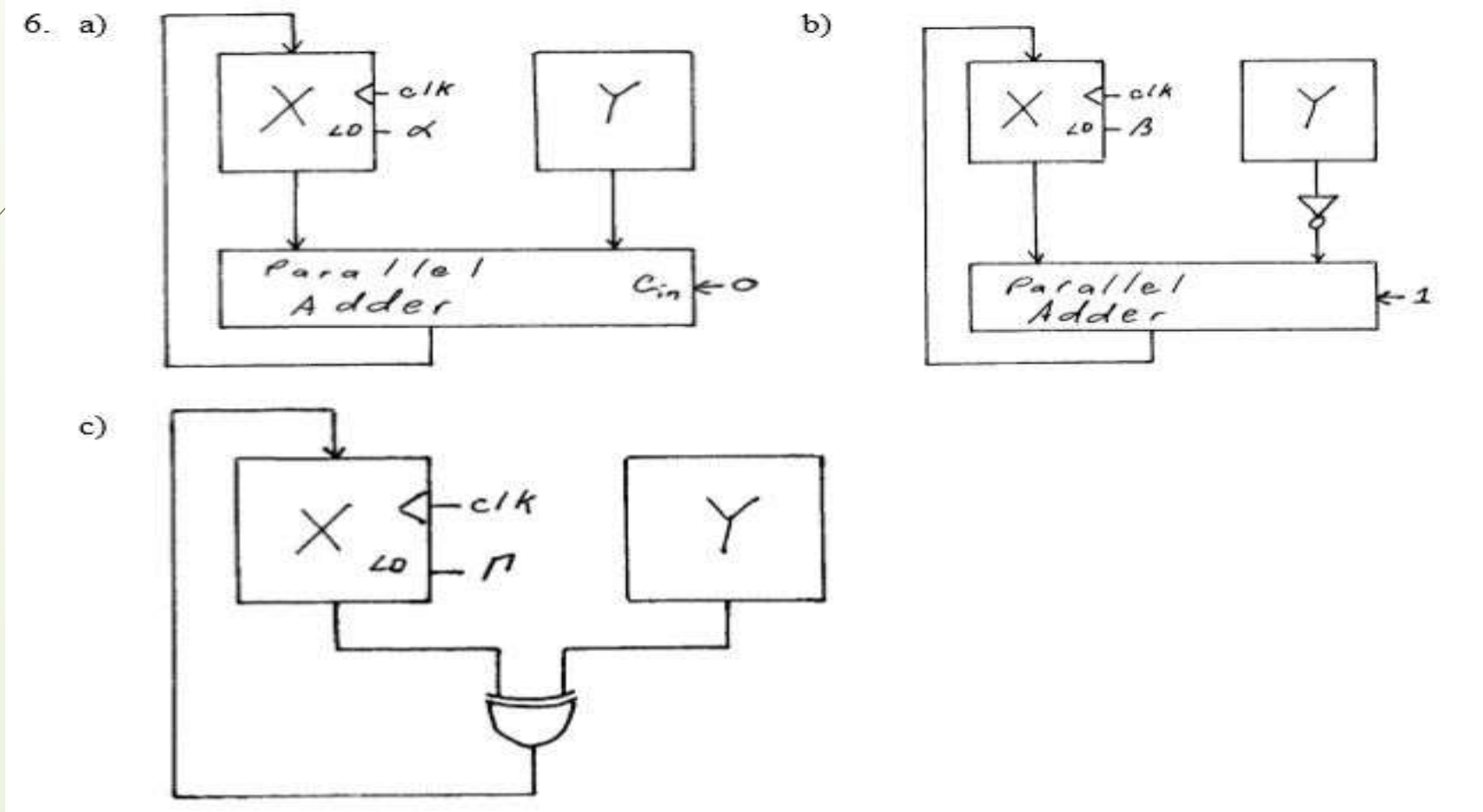


6. Show the hardware to implement the following RTL code.

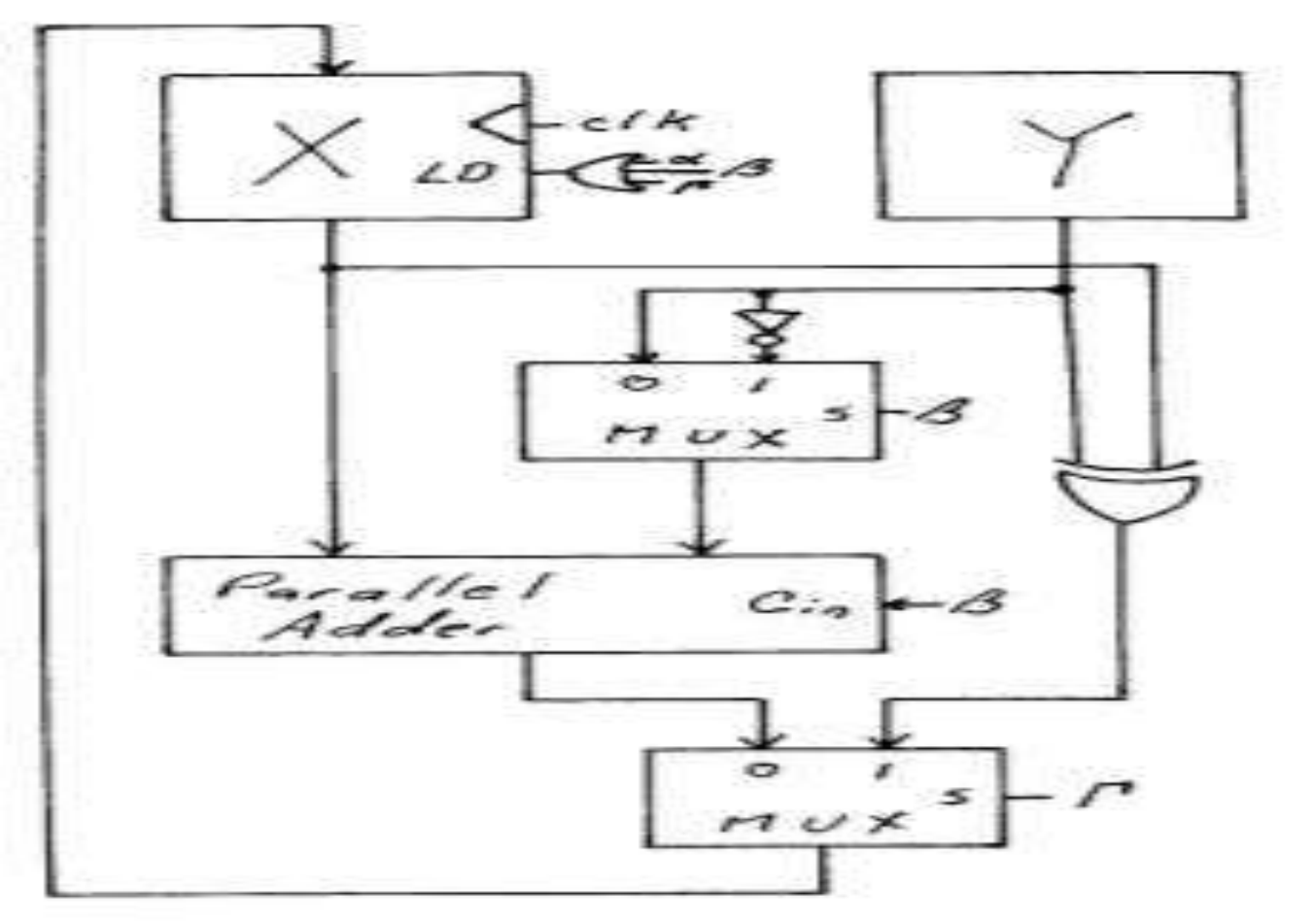
a. $\alpha : X \leftarrow X + Y$

b. $\beta : X \leftarrow X + Y' + 1$

c. $\gamma : X \leftarrow X \boxed{+} Y$

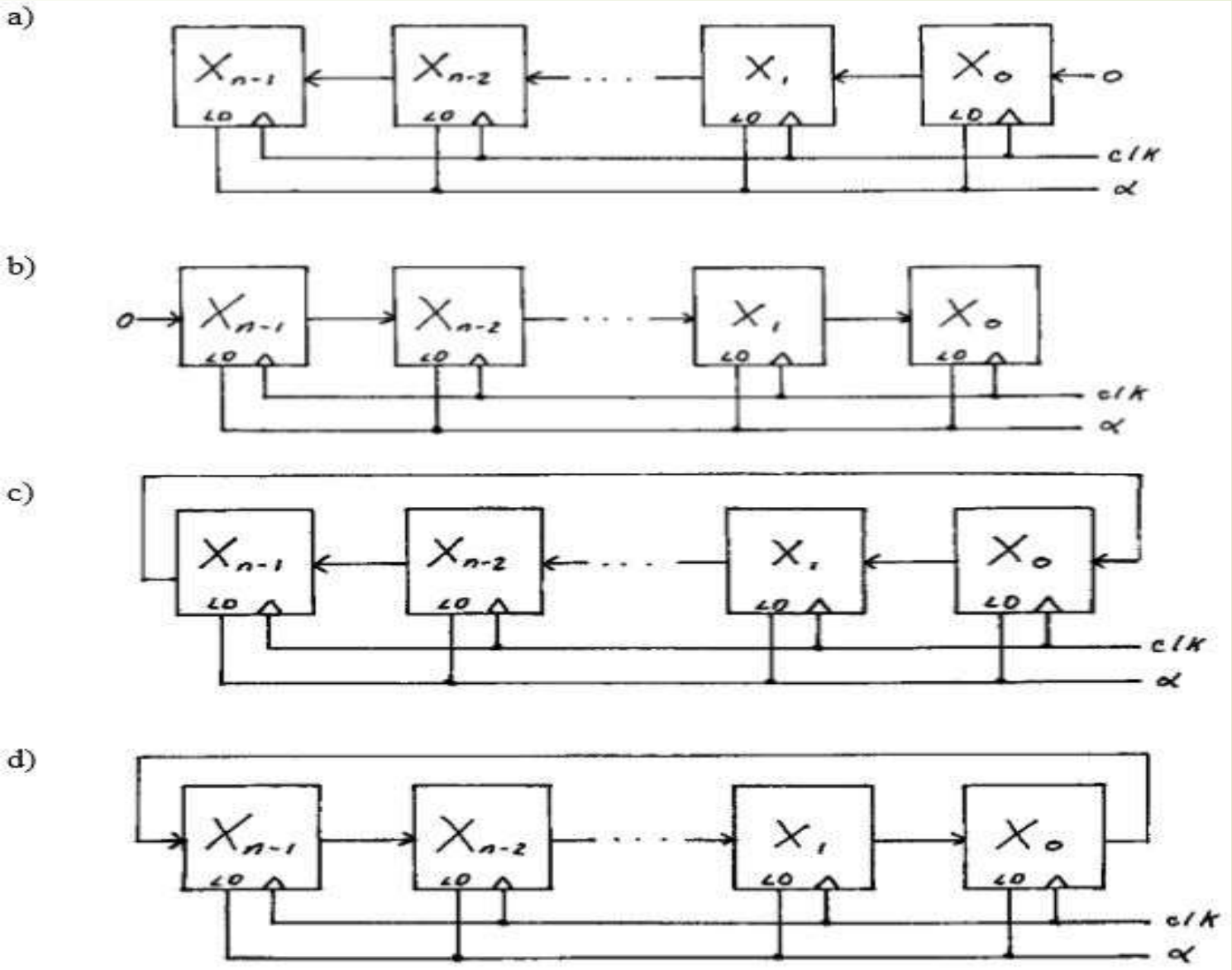


Contd...

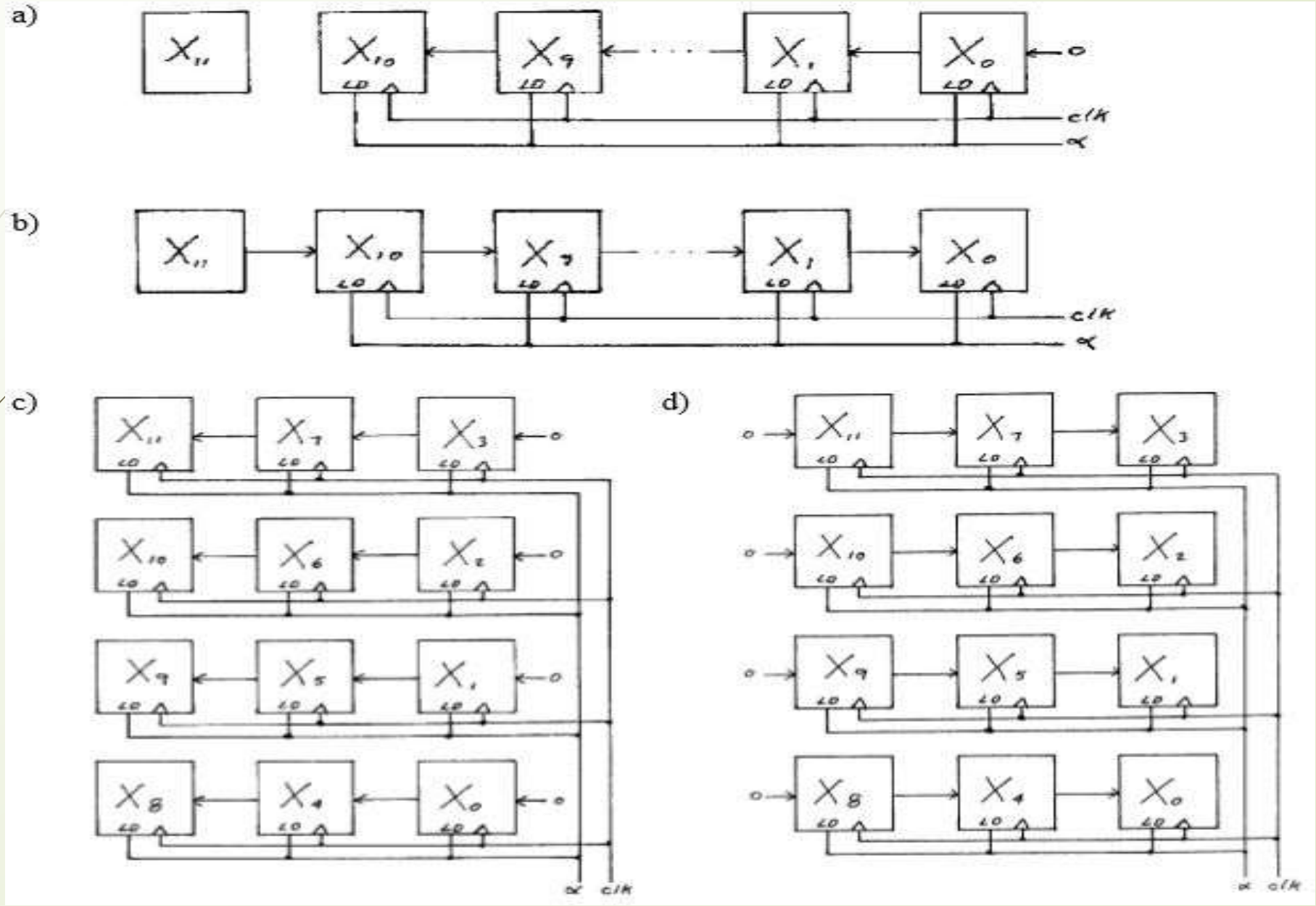


Q. Show the hardware to implement the following micro-operations. X consists of four D flip-flops. Each micro-operation occurs when α = 1.

a. shl(X) b. shr(X) c. cil(X) d. cil(X)



Q. Show the hardware to implement the following micro-operations. X consists of 12 flip-flops. Each micro-operation occurs when α =1.



VHDL(Very high speed integrated circuit(VHSIC)Hardware Description Language)

- VHDL was developed to provide a standard for designing digital systems.
- VHDL specifies a formal syntax. The designer creates a design file using that syntax just as a programmer writes a C-program.
- The designer then synthesizes the design using only design package that can accept VHDL files. This is equivalent to the programmer compiling the C-program. It checks for errors in syntax and declaration, but not in logic.
- Finally, the designer debugs the design using simulation tools.

Advantages of VHDL:

- PORTABILITY:
  Just as a valid C-program can be compiled by any compliant C compiler, a VHDL design can be synthesized by any design system that supports VHDL.
- DEVICE INDEPENDENT:
  The VHDL file is device independent. The same file can be used to implement the design on a custom ICs, on ASICs or any PLD that is capable of containing the design.
- SIMULATION:
  VHDL designs can be simulated by the design system, allowing the designer to verify the design performance before committing it to hardware.
- SYSTEM SPECIFICATION:
  The designer can design the system using a high level of abstraction , such as a finite state machine, down to a low level digital logic implementation.

Disadvantages of VHDL:

- VHDL source code often becomes long and difficult to follow, especially at a low level of abstraction.
- It is often less intuitive than a block diagram or RTL description of same system.
- Different design tools may produce different valid design for the same system, especially for high level of abstraction, providing no details about the implementation.

VHDL syntax:

VHDL design code has 3 primary sections:

i. Library declarations
ii. Entity section and
iii. Architecture section

- Library declaration:

  => It consists of statements that specify libraries to be accessed and modules of these libraries to be used.(such as “include” in C)

  => The most commonly used library is called IEEE. In this library, std_logic_1164 is used most often, which specifies i/p and o/p declarations for the designer to use(such as “stdio.h” in C).

```
Library IEEE;
use IEEE.std_logic_1164.all;
```

Contd…

- In the second section, the entity section, the designer specifies the name of the design and its inputs and outputs.
- The designer does not specify the logic that uses and drives these signals here; that is done in the architecture section.