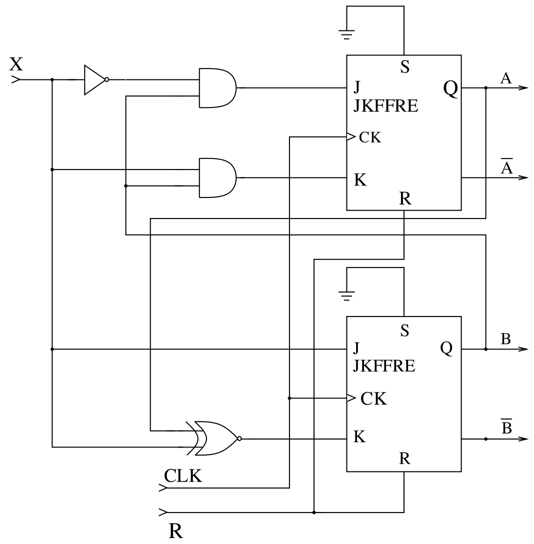
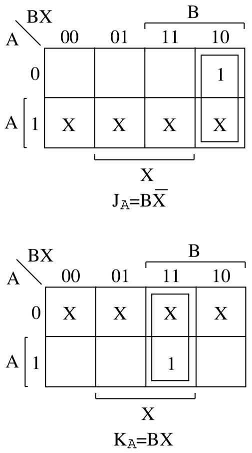
Designing with JK Flip-Flops (cont'd)

- Simplify the flip-flop input and output equations.

$$J_A = B\bar{X};\ K_A = BX;$$
$$J_B = X;\ K_B = AX + \bar{A}\bar{X}$$

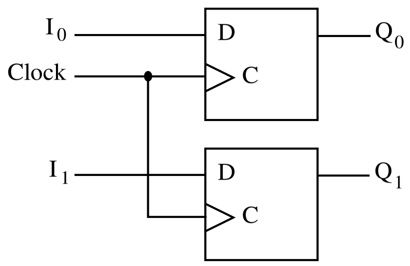
- Draw the logic diagram with flip-flops and combinational gates.





Register

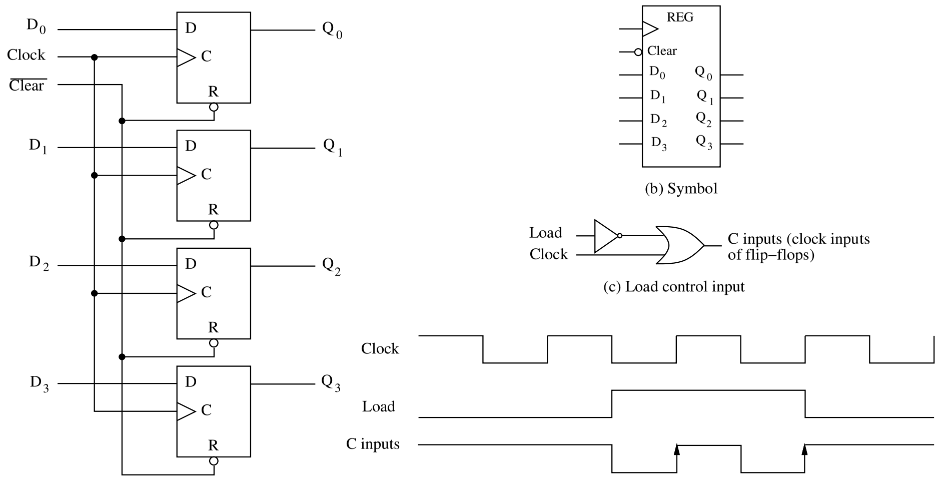
- A **register** is a collection of flip-flops with a common clock used primarily to hold (a vector of) binary information.
- Generally, registers are used to perform simple data storage and movement operations.
- A 2-bit register: # of states? # of inputs/outputs? next state? Moore or Mealy?



Current state	Next state $Q_1(t+1)\ Q_0(t+1)$				Output
	$I_1\ I_0$				
$Q_1\ Q_0$	00	01	10	11	$Q_1\ Q_0$
00	00	01	10	11	00
01	00	01	10	11	01
10	00	01	10	11	10
11	00	01	10	11	11

Registers

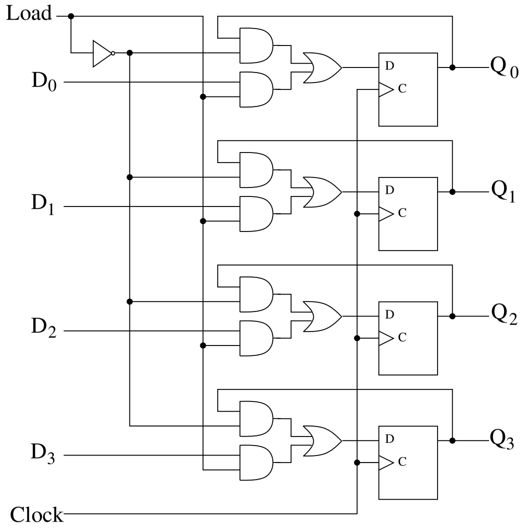
- How can we "selectively" store data in some registers and leave the others unchanged?
  - Use a flip-flop that can "hold" under some set of logic inputs: SR, JK, and T: "hold" state when inputs are "0".
  - "Gate" the input with a load control logic signal (**clock gating**).



(a) Logic diagram (b) Symbol (c) Load control input (d) Timing diagram

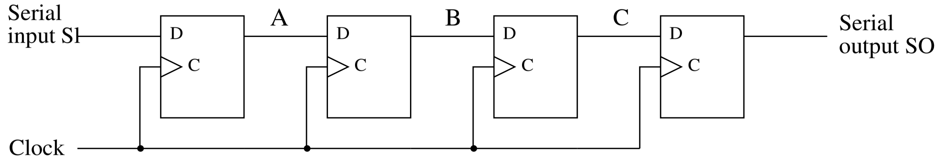
Registers with Load Control

- Clock gating might cause the **clock skew** problem.
- Better way: Run the clock continuously and use a load control to selectively change the register contents.
- $Load = 1$: Outputs are replaced by the current inputs; $Load = 0$: Data holds.

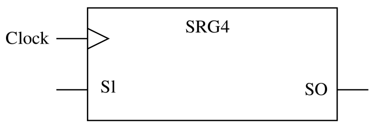


Shift Registers

- **Shift registers** are a special class of registers that can be used to store and manipulate data.
- The data input, SI, is often called "serial input" (shift-right input: shift a 1-bit input to the right at each clock).
- The data output, SO, is often called "serial output".
- Collectively, A, B, C, and SO are often called the "parallel outputs."



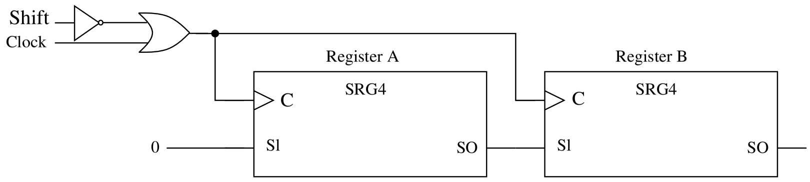
(a) Logic diagram

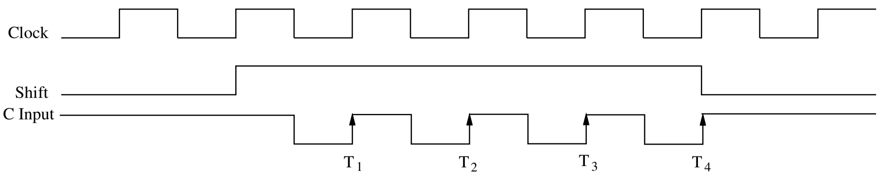


(b) Symbol

Serial Transfer

- $Shift = 1$: Enables the shift register; $Shift = 0$: Disables the shift register.

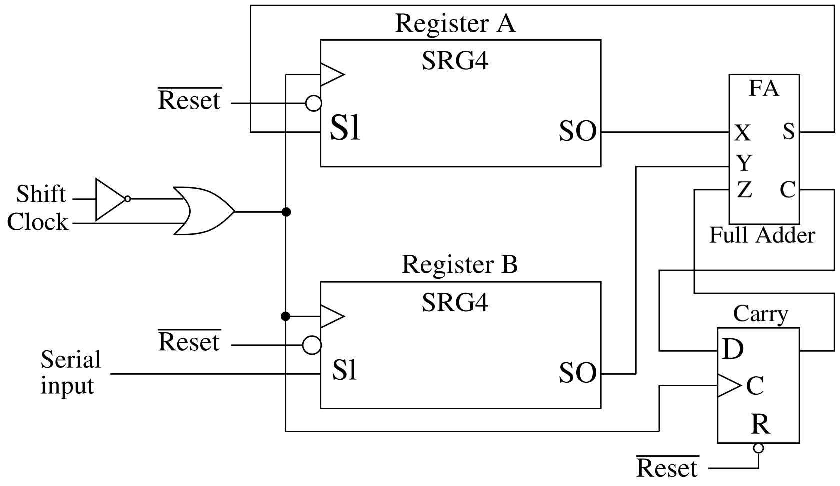




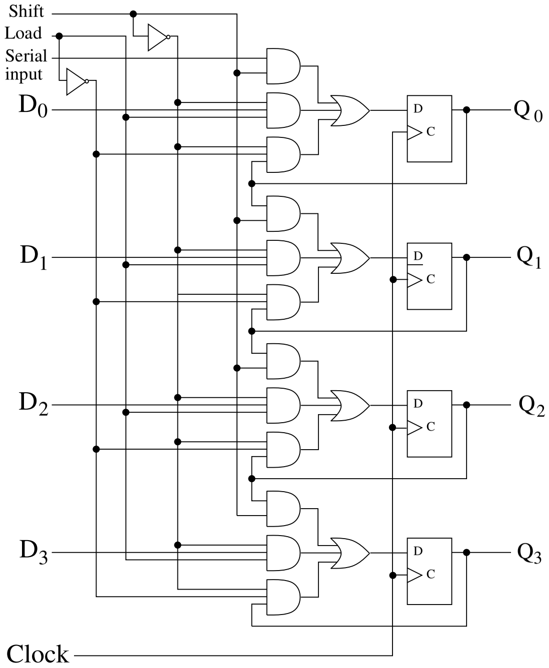
Timing pulse	Shift Register A				Shift Register B			
Initial value	1	0	1	1	0	0	1	0
After T_1	0	1	0	1	1	0	0	1
After T_2	0	0	1	0	1	1	0	0
After T_3	0	0	0	1	0	1	1	0
After T_4	0	0	0	0	1	0	1	1

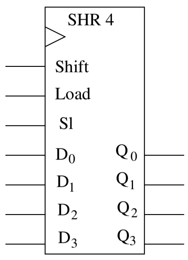
Serial Addition

- Add two numbers (held in registers A and B) serially, starting at the least significant bit.
- The serial adder is slower but smaller than the parallel (ripple-carry) adder.



Shift Register with Parallel Load





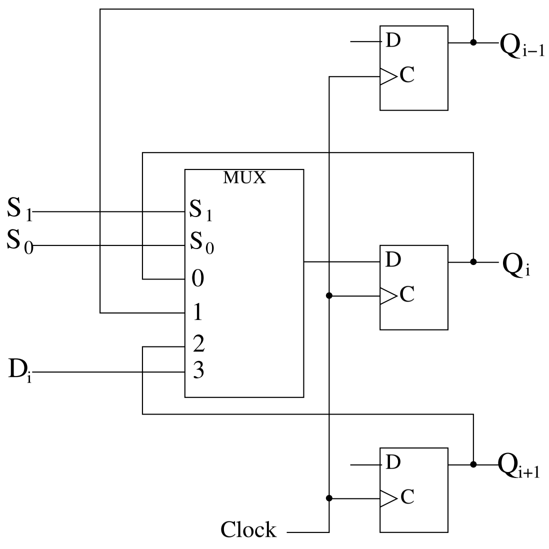
(b) Symbol

Shift	Load	Operation
0	0	No change
0	1	Load parallel data
1	X	Shift down from Q_0 to Q_3

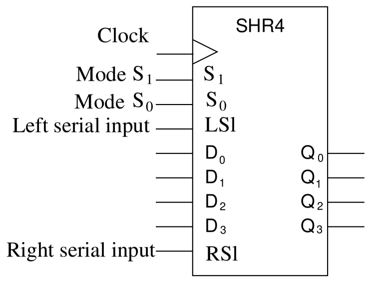
- Application: Load an n-bit data in parallel into a shift register and then transmit the data serially.

Bidirectional Shift Register

- By feeding a 4-input multiplexer to each D flip-flop in a shift register, we can implement a device which **shifts right, shifts left, holds, parallel loads**.



(a) Logic diagram of one typical stage



(b)Symbol

Mode control S_1	S_0	Register Operation
0	0	No change
0	1	Shift down
1	0	Shift up
1	1	Parallel load

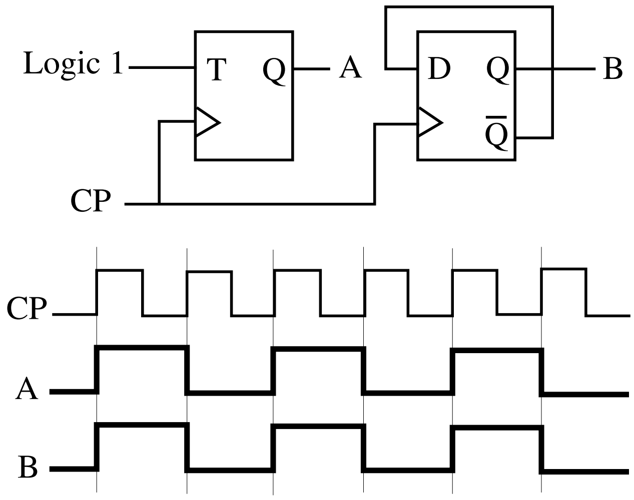
Counter

- **Counter:** A register that goes through a prescribed sequence of states upon the application of input pulses.

- An n-bit **binary counter** consists of n flip-flops and can count in binary from 0 through $2^n - 1$.

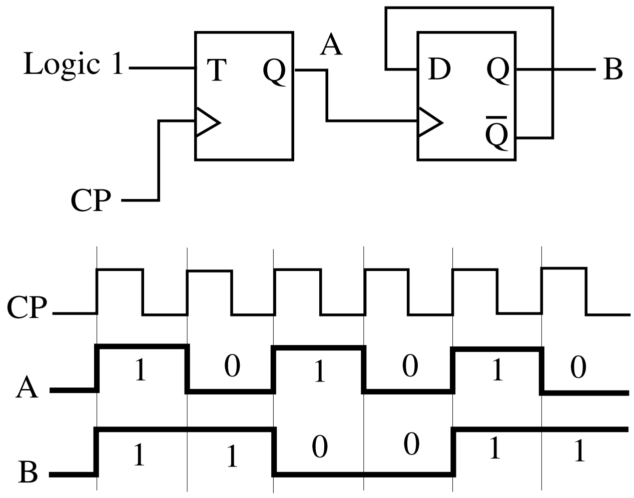
- Two types of counters
  - **Ripple counters:** The flip-flop output transition serves as a source for triggering other flip-flops.
  - **Synchronous counters:** The Clk inputs of all flip-flops receive the common clock pulse, and the change of state is determined from the present state of the counter.

A “Divide-by-2” Counter



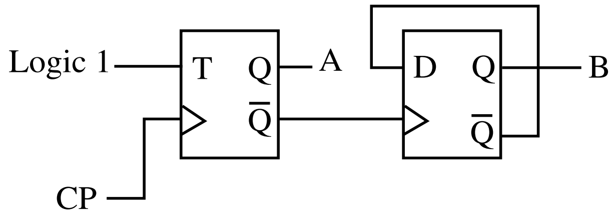
- The frequency of the output waveform is half the clock CP (“divide-by-2”).

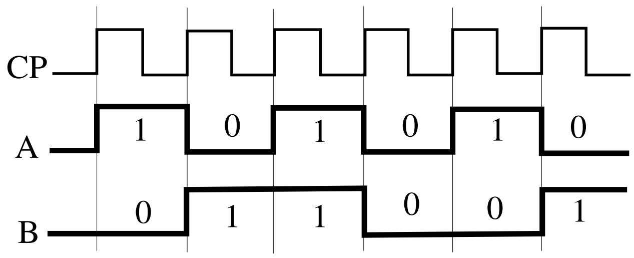
Down Ripple Counter



- B “divides” the frequency in A by 2 (doubles the period).
- Count (B, A) base 2: 3 $\longrightarrow$ 2 $\longrightarrow$ 1 $\longrightarrow$ 0 $\longrightarrow$ 3 $\longrightarrow$ 2 $\longrightarrow$ ....

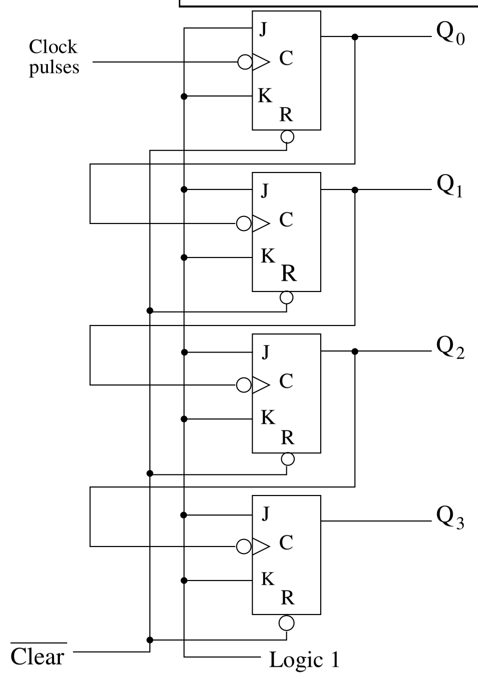
Up Ripple Counter





- B "divides" the frequency in A by 2 (doubles the period).
- Count (B, A) base 2: $0 \longrightarrow 1 \longrightarrow 2 \longrightarrow 3 \longrightarrow 0 \longrightarrow 1 \longrightarrow ....$

4-Bit Up Ripple Counter



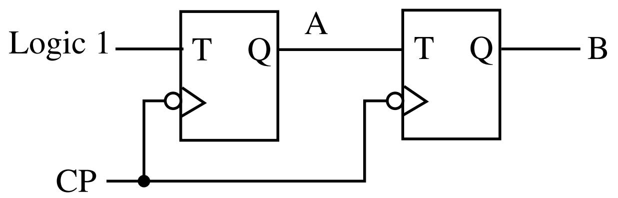
Upward Couting Sequence

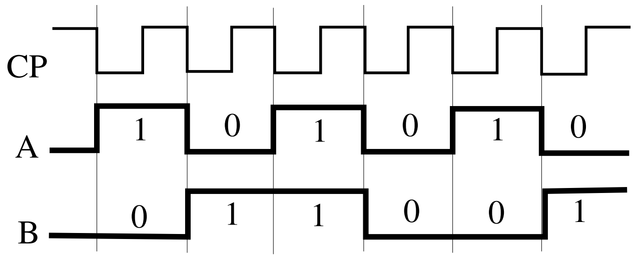
Q_3	Q_2	Q_1	Q_0
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

- Each edge sensitive transition causes a change in the next flip-flop's state.

Synchronous Counter

- The common clock pulse triggers all flip-flops simultaneously.
- A 2-bit synchronous up counter made with two T flip-flops.



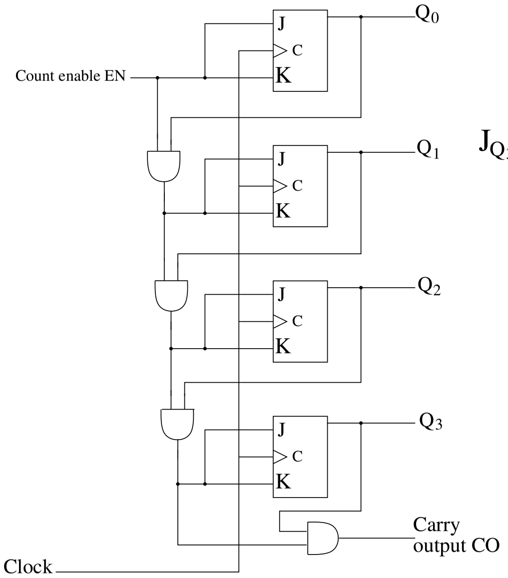


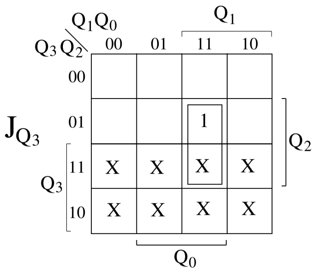
Synchronous Binary Counter with JK Flip-Flops

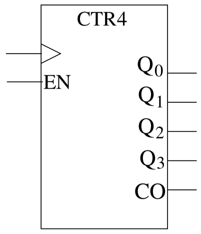
Present state				Next state				Flip–flop inputs							
Q_3	Q_2	Q_1	Q_0	Q_3	Q_2	Q_1	Q_0	J_{Q3}	K_{Q3}	J_{Q2}	K_{Q2}	J_{Q1}	K_{Q1}	J_{Q0}	K_{Q0}
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	X	0	0	X	0	X	1	X
1	0	0	1	1	0	1	0	X	0	0	X	1	X	X	1
1	0	1	0	1	0	1	1	X	0	0	X	X	0	1	X
1	0	1	1	1	1	0	0	X	0	1	X	X	1	X	1
1	1	0	0	1	1	0	1	X	0	X	0	0	X	1	X
1	1	0	1	1	1	1	0	X	0	X	0	1	X	X	1
1	1	1	0	1	1	1	1	X	0	X	0	X	0	1	X
1	1	1	1	0	0	0	0	X	1	X	1	X	1	X	1

- 16 states from 0000 to 1111.
- $J_{Q0} = K_{Q0} = 1$; $J_{Q1} = K_{Q1} = Q_0$; $J_{Q2} = K_{Q2} = Q_0Q_1$; $J_{Q3} = K_{Q3} = Q_0Q_1Q_2$.
- In general, $J_{Qi} = K_{Qi} = Q_0Q_1 \ldots Q_{i-1}$.

JK Up Binary Counter







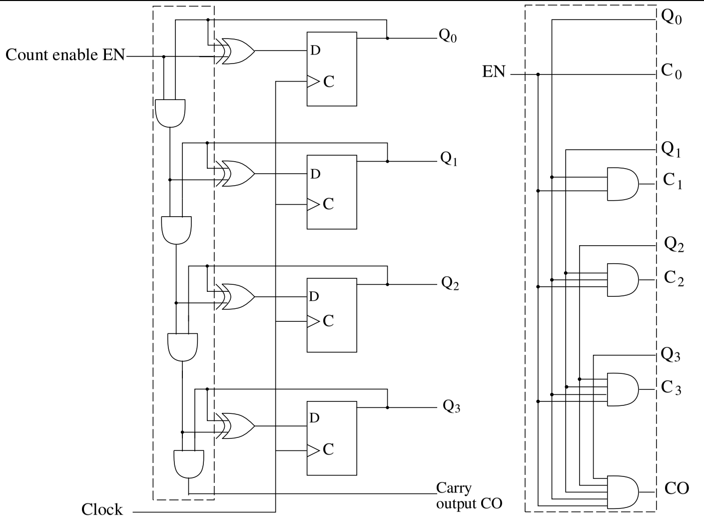
- With EN signal: $J_{Q0} = K_{Q0} = EN$; $J_{Q1} = K_{Q1} = EN \cdot Q_0$; $J_{Q2} = K_{Q2} = EN \cdot Q_0 \cdot Q_1$; $J_{Q3} = K_{Q3} = EN \cdot Q_0 \cdot Q_1 \cdot Q_2$.
- The carry output CO can be used to extend the counter to more stages.

Up Binary Counter with D Flip-Flops

Present state				Next state				Flip−flop inputs			
Q_3	Q_2	Q_1	Q_0	Q_3	Q_2	Q_1	Q_0	D_{Q3}	D_{Q2}	D_{Q1}	D_{Q0}
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	0
0	0	1	0	0	0	1	1	0	0	1	1
0	0	1	1	0	1	0	0	0	1	0	0
0	1	0	0	0	1	0	1	0	1	0	1
0	1	0	1	0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	1	0	1	1	1
0	1	1	1	1	0	0	0	1	0	0	0
1	0	0	0	1	0	0	1	1	0	0	1
1	0	0	1	1	0	1	0	1	0	1	0
1	0	1	0	1	0	1	1	1	0	1	1
1	0	1	1	1	1	0	0	1	1	0	0
1	1	0	0	1	1	0	1	1	1	0	1
1	1	0	1	1	1	1	0	1	1	1	0
1	1	1	0	1	1	1	1	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0

- 16 states from 0000 to 1111.
- With EN signal: $D_{Q0} = Q_0 \oplus EN$; $D_{Q1} = Q_1 \oplus (Q_0 \cdot EN)$; $D_{Q2} = Q_2 \oplus (Q_0 \cdot Q_1 \cdot EN)$; $D_{Q3} = Q_3 \oplus (Q_0 \cdot Q_1 \cdot Q_2 \cdot EN)$.
- In general, $D_{Qi} = Qi \oplus (Q_0 \cdot Q_1 \cdot \ldots \cdot Q_{i-1} \cdot EN)$.

Serial and Parallel D Binary Counter



(a) Serial gating (b) Parallel gating

- $D_{Q0} = Q_0 \oplus EN$; $D_{Q1} = Q_1 \oplus (Q_0 \cdot EN)$;
  $D_{Q2} = Q_2 \oplus (Q_0 \cdot Q_1 \cdot EN)$; $D_{Q3} = Q_3 \oplus (Q_0 \cdot Q_1 \cdot Q_2 \cdot EN)$.
- **Serial counter:** A chain of (AND) gates is used to provide information to each stage about the state of prior stages.
- **Parallel counter:** Information are provided to all stages in parallel.

Up-Down Binary Counter

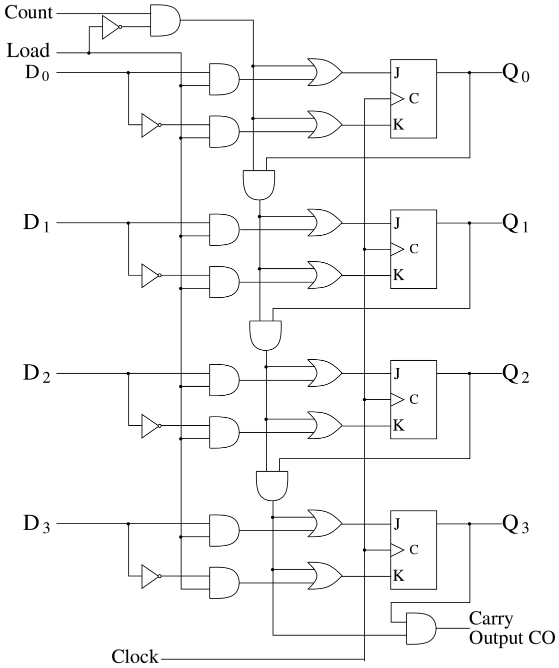
- $S = 1$: Up-counter; $S = 0$: Down-counter.
- $EN = 1$: Normal up- or down-counter; $EN = 0$: Disable both counters.
- T-type flip-flop equations ($T = J = K$) for a 4-bit, up-down binary counter:

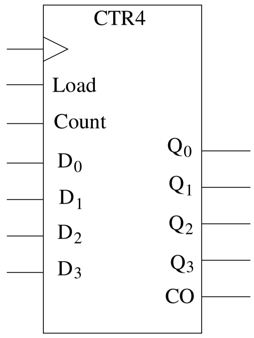
$$\begin{aligned}
T_{A0} &= EN \\
T_{A1} &= Q_0 \cdot S \cdot EN + \bar{Q}_0 \cdot \bar{S} \cdot EN \\
T_{A2} &= Q_0 \cdot Q_1 \cdot S \cdot EN + \bar{Q}_0 \cdot \bar{Q}_1 \cdot \bar{S} \cdot EN \\
T_{A3} &= Q_0 \cdot Q_1 \cdot Q_3 \cdot S \cdot EN + \bar{Q}_0 \cdot \bar{Q}_1 \cdot \bar{Q}_3 \cdot \bar{S} \cdot EN
\end{aligned}$$

- Carry outputs for the next stage:

$$\begin{aligned}
C_{up} &= Q_0 \cdot Q_1 \cdot Q_2 \cdot Q_3 \cdot S \cdot EN. \\
C_{down} &= \bar{Q}_0 \cdot \bar{Q}_1 \cdot \bar{Q}_2 \cdot \bar{Q}_3 \cdot \bar{S} \cdot EN.
\end{aligned}$$

Binary Counter with Parallel Load



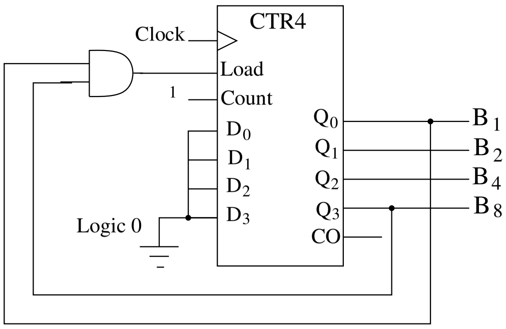


(b) Symbol

Load	Count	Operation
0	0	No change
0	1	Count next binary state
1	X	Load inputs

BCD Counter

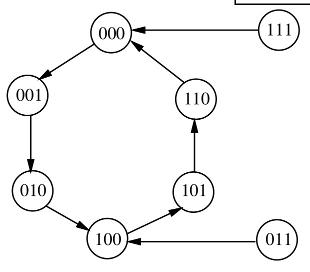
Present State				Next State				Output	Flip–Flop Inputs			
B_8	B_4	B_2	B_1	B_8	B_4	B_2	B_1	Y	T_{B_8}	T_{B_4}	T_{B_2}	T_{B_1}
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	0	1
0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	0	1
1	0	0	1	0	0	0	0	1	1	0	0	1



Load	Count	Operation
0	0	No change
0	1	Count next binary state
1	X	Load inputs

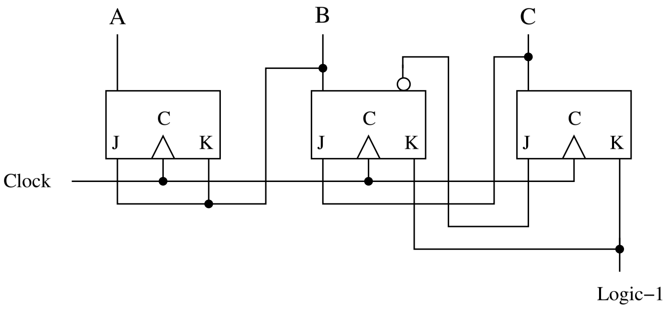
- 10 states: 0000 → 0001 → . . . → 1001 → 0000 . . .
- $T_{B1} = 1$; $T_{B2} = B_1\bar{B_8}$; $T_{B4} = B_1B_2$; $T_{B8} = B_1B_8 + B_1B_2B_4$; $Y = B_1B_8$.

Arbitrary Count Sequence



State diagram

Present State A	B	C	Next State A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	1	0	0	1	X	X	1	0	X
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	0	0	0	X	1	X	1	0	X



- Unused states: 011, 111.
- $J_A = B$; $K_A = B$; $J_B = C$; $K_B = 1$; $J_C = \bar{B}$; $K_C = 1$.

Memory

- **Memory:** A collection of storage cells together with the necessary circuits to transfer information to and from them.
- **Memory data:** A collection of binary digits to be stored into or accessed from memory cells. Typical data elements are:
  - **Bit:** A single binary digit.
  - **Byte:** A collection of 8 bits accessed together.
  - **Word:** A collection of binary bits whose size is a typical unit of access for the memory. Typically a power of two multiple of bytes (e.g., 2 bytes, 4 bytes, etc).
- **Memory address:** A collection of binary digits interpreted as a number that refers to a basic memory storage element.
- **Memory organization:** The structure of memory in terms of how data are accessed.
  - Organized as an indexed **array of words.** Index $\longleftrightarrow$ Memory address.
  - E.g., Intel 8080 used a 16-bit address to address 65,536 (2^{16}) bytes. (Intel Pentium II: 32-bit address)