Chapter 6

Registers and Counters

6-1

Outline

- Registers
- Shift Registers
- Ripple Counters
- Synchronous Counters
- Other Counters

6-2

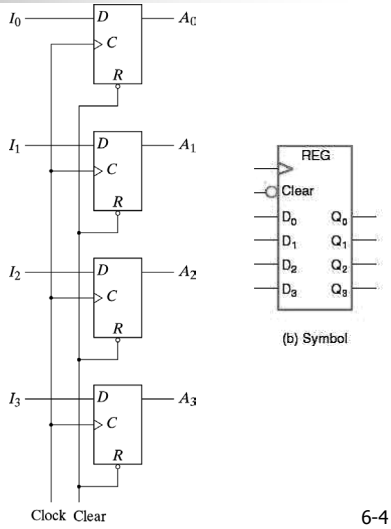
Registers and Counters

- Register:
  - A set of flip-flops, possibly with added combinational gates, that perform data-processing tasks
  - Store and manipulate information in a digital system
- Counter:
  - A register that goes through a predetermined sequence of states
    - A special type of register
  - Employed in circuits to sequence and control operations

6-3

The Simplest Register

- Consist of only flip-flops
- Triggered by common clock input
- The *Clear* input goes to the R (reset) input of all flip-flops
  - *Clear* = 0 → all flip-flops are reset ***asynchronously***
- The *Clear* input is useful for cleaning the registers to all 0's prior to its clocked operation
  - Must maintain at logic 1 during normal operations



6-4

Register with Parallel Load

- When *Load* = 1, all the bits of data inputs are transferred into the registers
  - Parallel loaded
- When *Load* =0, the outputs of the flip-flops are connected to their respective inputs
  - Keep no change

control data loading at the input side

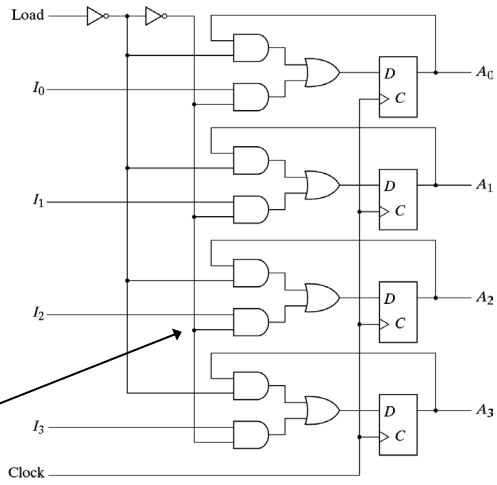


Fig. 6-2 4-Bit Register with Parallel Load

6-5

Outline

- Registers
- Shift Registers
- Ripple Counters
- Synchronous Counters
- Other Counters

6-6

The Simplest Shift Register

- Shift register: a register capable of shifting its binary information in one or both directions
- The simplest form: consist of only a chain of flip-flops in cascade

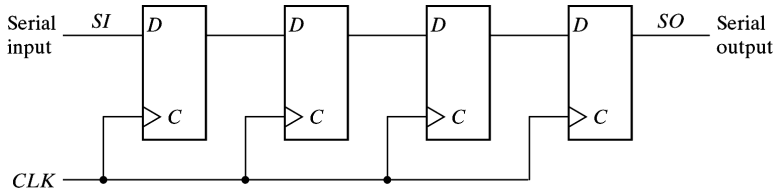
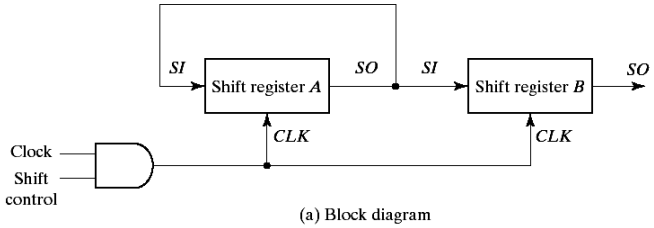


Fig. 6-3 4-Bit Shift Register

6-7

Serial Transfer

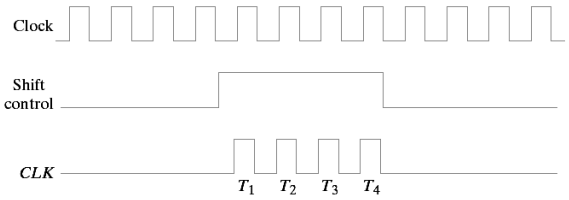
- Serial mode: information is transferred and manipulated one bit at a time
- Parallel mode: all the bits of the registers are transferred at the same time
- Shift control: determine when and how many times of the registers are shifted
- Register A is connected in circular mode in this circuit



(a) Block diagram

6-8

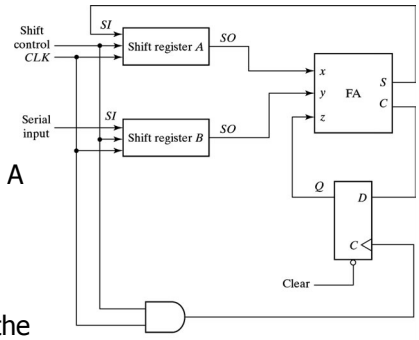
Serial Transfer Example



Timing Pulse	Shift Register A	Shift Register B
Initial value	1 0 1 1	0 0 1 0
After T1	1 1 0 1	1 0 0 1
After T2	1 1 1 0	1 1 0 0
After T3	0 1 1 1	0 1 1 0
After T4	1 0 1 1	1 0 1 1

Serial Adder

- The bits of two binary numbers are added one pair at a time through a single full adder (FA) circuit
- Initialization:
  - A = augend; B = addend
  - Carry flip-flop is cleared to 0
- For each clock pulse:
  - A new sum bit is transferred to A
  - A new carry is transferred to Q
  - Both registers are shifted right
- To add three or more numbers:
  - Shift in the next number from the serial input while B is shifted to the FA
  - A will accumulate their sum

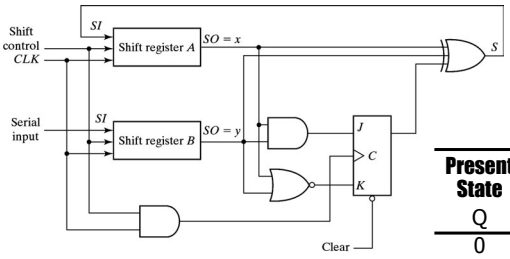


Serial v.s. Parallel

- Serial adders:
  - Use shift registers
  - A sequential circuit
  - Require only one FA and a carry flip-flop
  - Slower but require less equipment
- Parallel adders:
  - Use registers with parallel load for sum
  - Basically a pure combinational circuit
  - ***n*** FAs are required
  - Faster

6-11

Redesign the Serial Adder



$J_Q = xy$
$K_Q = x'y' = (x + y)'$
$S = x \oplus y \oplus Q$

Present State	Inputs		Next State	Outputs	F/F Inputs	
Q	X	Y	Q	S	J_Q	K_Q
0	0	0	0	0	0	X
0	0	1	0	1	0	X
0	1	0	0	1	0	X
0	1	1	1	0	1	X
1	0	0	0	1	X	1
1	0	1	1	0	X	0
1	1	0	1	0	X	0
1	1	1	1	1	X	0

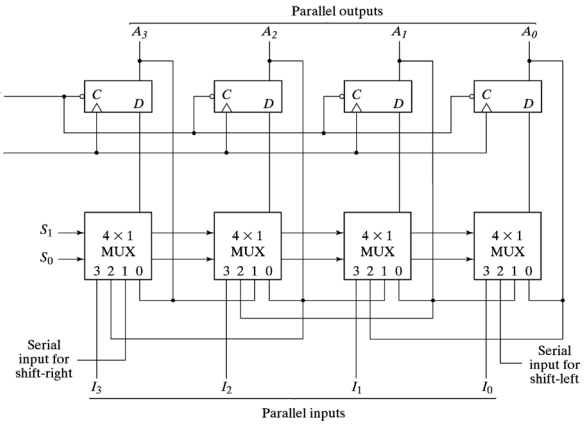
6-12

Universal Shift Register

- The most general shift register has the following capabilities:
  - A *clear* control to clear the register to 0
  - A *clock* input to synchronize the operations
  - A *shift-right (left)* control to enable the shift right (left) operation and the *serial input* and *output* lines associated with the shift right (left)
  - A *parallel-load* control to enable a parallel transfer and the *n* input lines associated with the parallel transfer
  - *n* parallel output lines
  - A control state that leaves the information in the register unchanged in the presence of the clock

4-Bit Universal Shift Register

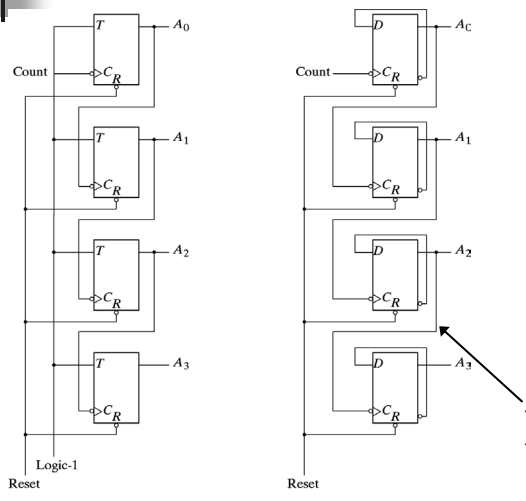
Mode Control		Register
S1	S0	Operation
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load



Outline

- Registers
- Shift Registers
- Ripple Counters
- Synchronous Counters
- Other Counters

Binary Ripple Counter



(a) With T flip-flops (b) With D flip-flops

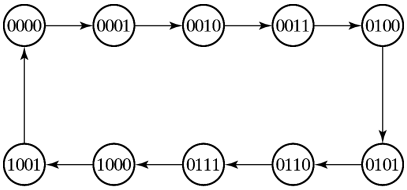
Binary Count Sequence

A3	A2	A1	A0
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

the output transition triggers the next flip-flop

BCD Ripple Counter

- The count will return to 0 after 9
- Q1: always complemented
- Q2: inverted when Q8 = 0 and Q1 = 1 → 0
- Q4: inverted when Q2 = 1 → 0
- Q8: when Q1 = 1 → 0
  if (Q2 = Q4 = 1) Q8 is inverted
  else Q8 = 0



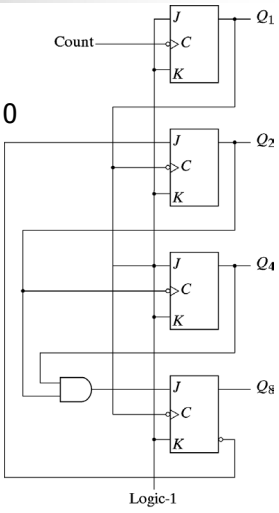


Fig. 6-9 State Diagram of a Decimal BCD-Counter

6-17

Three-Decade BCD Counter

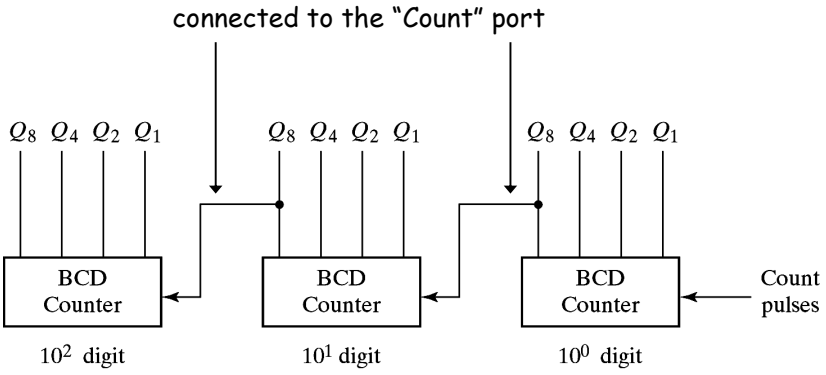


Fig. 6-11 Block Diagram of a Three-Decade Decimal BCD Counter

6-18

Outline

- Registers
- Shift Registers
- Ripple Counters
- Synchronous Counters
- Other Counters

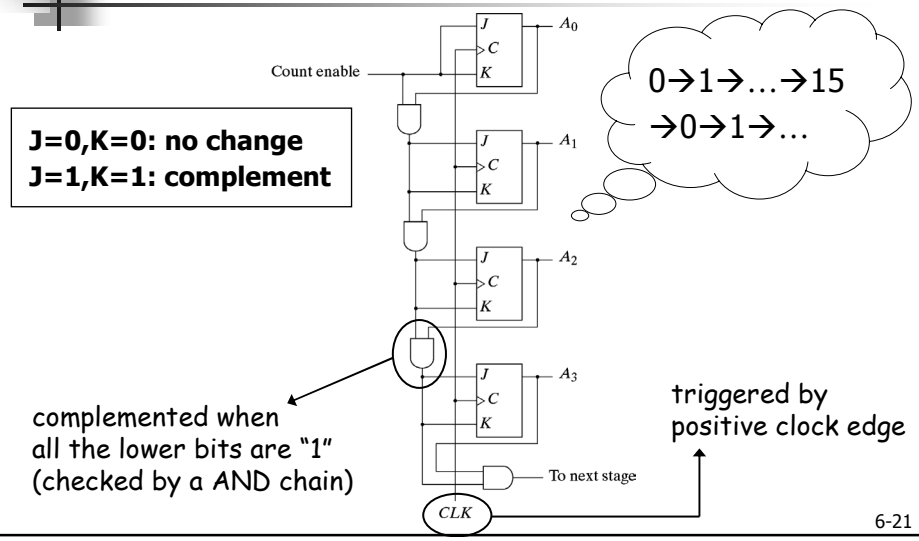
6-19

Ripple v.s. Synchronous

- Ripple counters:
  - Flip-flops are triggered by the outputs of another flip-flops
    - Triggering source may not the same for each flip-flop
  - The flip-flops are changed serially
- Synchronous counters:
  - Flip-flops are triggered by common clock pulses
    - Triggering sources are the same for all flip-flops
  - All operations are performed simultaneously

6-20

Binary Counter



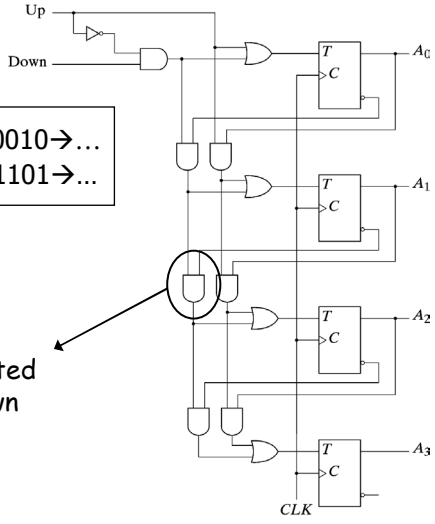
J=0,K=0: no change
J=1,K=1: complement

complemented when
all the lower bits are "1"
(checked by a AND chain)

triggered by
positive clock edge

6-21

Up-Down Binary Counter



Up: 0000→0001→0010→…
Down: 1111→1110→1101→…

take the complemented
values for count-down
calculation

6-22

Design a BCD Counter

- Go through normal sequential circuit design procedure

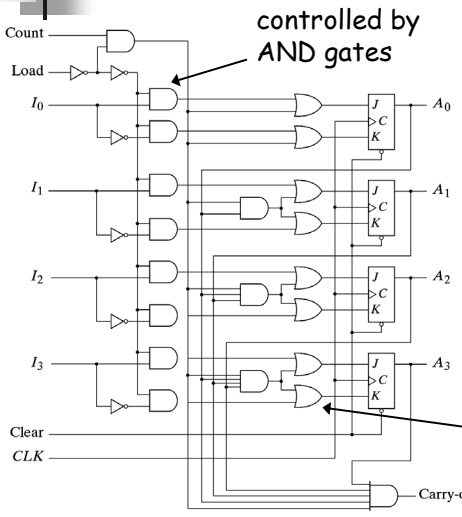
Present State				Next State				Output	Flip-Flop Inputs			
Q_8	Q_4	Q_2	Q_1	Q_8	Q_4	Q_2	Q_1	y	TQ_8	TQ_4	TQ_2	TQ_1
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	1	0	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	0	1
0	0	1	1	0	1	0	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	0	1
0	1	1	1	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

$T_{Q1} = 1$ $\quad T_{Q4} = Q_2Q_1$ $\quad y = Q_8Q_1$

$T_{Q2} = Q_8'Q_1$ $\quad T_{Q8} = Q_8Q_1 + Q_4Q_2Q_1$

6-23

Binary Counter with Parallel Load



controlled by AND gates

Clear	CLK	Load	Count	Function
0	X	X	X	Clear to 0
1	↑	1	X	Load inputs
1	↑	0	1	Count next
1	↑	0	0	No change

asynchronous input

merged by OR gates

set when count = 1111

6-24

Achieve a BCD Counter

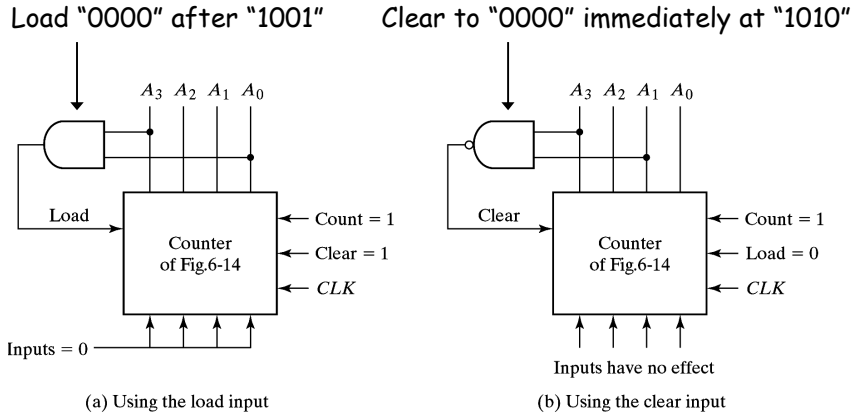


Fig. 6-15 Two ways to Achieve a BCD Counter Using a Counter with Parallel Load

6-25

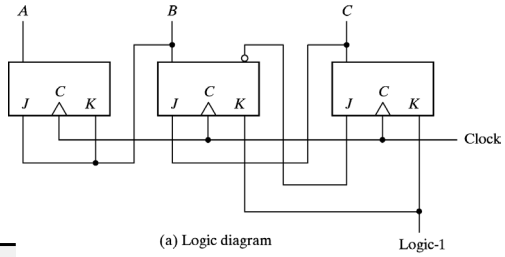
Outline

- Registers
- Shift Registers
- Ripple Counters
- Synchronous Counters
- Other Counters

6-26

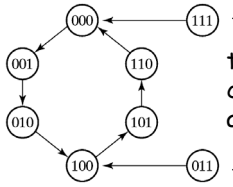
Counter with Unused States

- If the machine falls into the unused states, we have to bring it back !!
  - Cannot assign the used states as don't cares



(a) Logic diagram

Present State	Next State	Flip-Flop Inputs
A B C	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

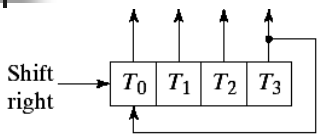


the destination can be assigned arbitrarily

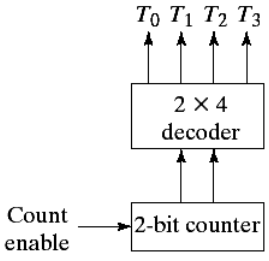
(b) State diagram

6-27

Ring Counter

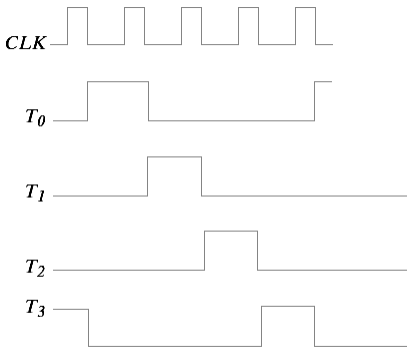


(a) Ring-counter (initial value = 1000)



(b) Counter and decoder

Ring counter: a circular shift register with only one flip-flop being set at any time

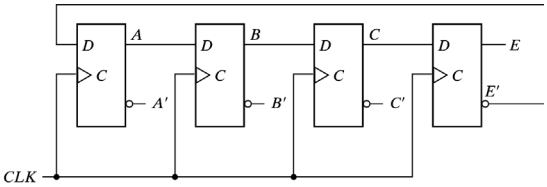


(c) Sequence of four timing signals

6-28

Johnson Counter

- Double the number of states of a ring counter by switch-tail connection
- If this counter falls into an unused state, it will never come back to normal states !!
- To avoid this situation, use some gates to form the input equation of each flip-flop instead of connecting directly
  - Ex: Dc = (A + C) B



(a) Four-stage switch-tail ring counter

Sequence number	Flip-flop outputs A	B	C	E	AND gate required for output
1	0	0	0	0	$A'E'$
2	1	0	0	0	AB'
3	1	1	0	0	BC'
4	1	1	1	0	CE'
5	1	1	1	1	AE
6	0	1	1	1	$A'B$
7	0	0	1	1	$B'C$
8	0	0	0	1	$C'E$

(b) Count sequence and required decoding

6-29