Counters Revisited

VOLKAN KURSUN

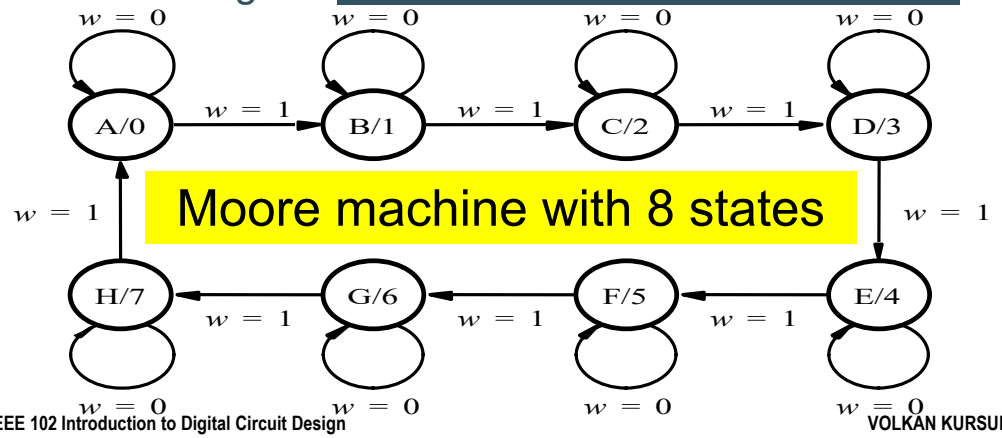
Some material from McGraw Hill

Outline

- Counter Design as FSM
- A Different Counter
- Self Correcting Ring Counter
- Self Correcting Johnson Counter

Counter Design as an FSM

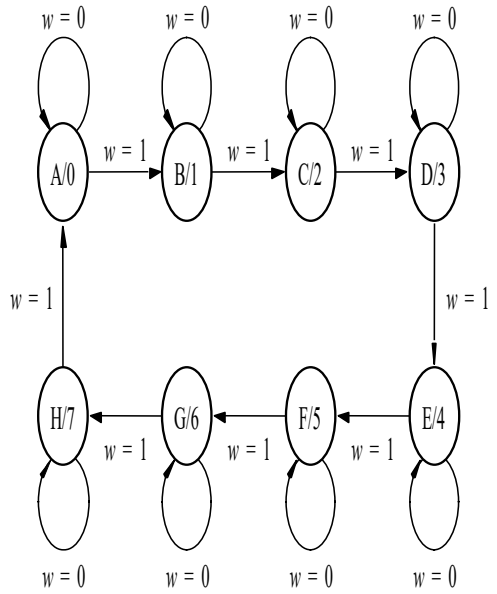
- The counting sequence is 0, 1, 2, 3, 4, 5, 6, 7, 0, 1…
- An input signal w: if w = 0, maintain state (pause counting). If w = 1, increment the counter
- **Design** the counter as a synchronous sequential circuit using the **formal fsm design methodology**



Counter State Table

- The counting sequence is 0, 1, 2, 3, 4, 5, 6, 7, 0, 1…

Present state	Next state $w=0$	Next state $w=1$	Output
A	A	B	0
B	B	C	1
C	C	D	2
D	D	E	3
E	E	F	4
F	F	G	5
G	G	H	6
H	H	A	7



Counter State Assigned Table

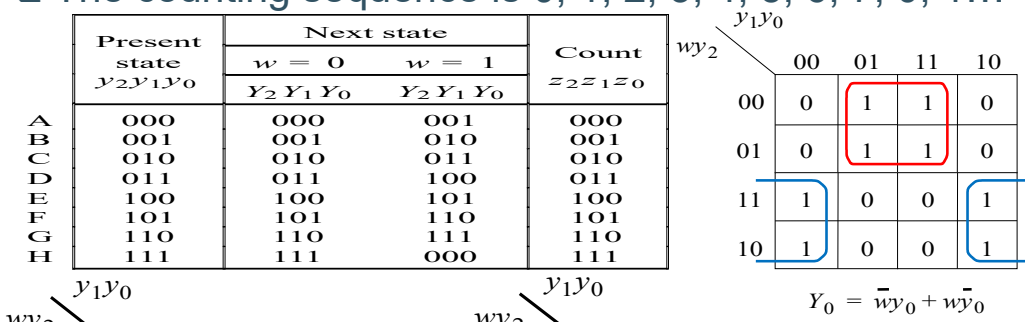
- The counting sequence is 0, 1, 2, 3, 4, 5, 6, 7, 0, 1…

	Present state $y_2y_1y_0$	Next state $w=0$ $Y_2Y_1Y_0$	Next state $w=1$ $Y_2Y_1Y_0$	Count $z_2z_1z_0$
A	000	000	001	000
B	001	001	010	001
C	010	010	011	010
D	011	011	100	011
E	100	100	101	100
F	101	101	110	101
G	110	110	111	110
H	111	111	000	111

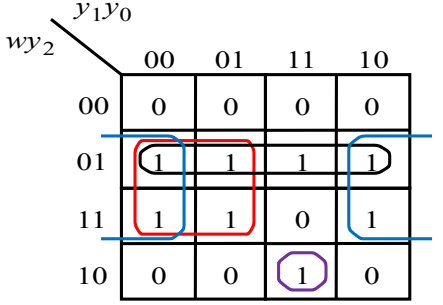
Counter Implementation with D

- The counting sequence is 0, 1, 2, 3, 4, 5, 6, 7, 0, 1…

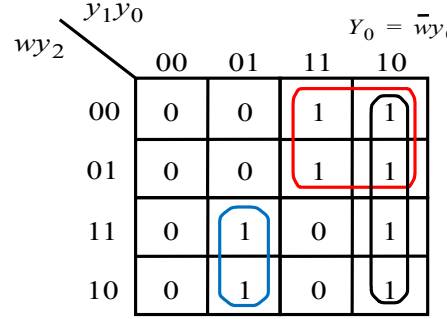
	Present state $y_2y_1y_0$	Next state $w=0$ $Y_2Y_1Y_0$	Next state $w=1$ $Y_2Y_1Y_0$	Count $z_2z_1z_0$
A	000	000	001	000
B	001	001	010	001
C	010	010	011	010
D	011	011	100	011
E	100	100	101	100
F	101	101	110	101
G	110	110	111	110
H	111	111	000	111



$Y_0 = \bar{w}y_0 + w\bar{y}_0$



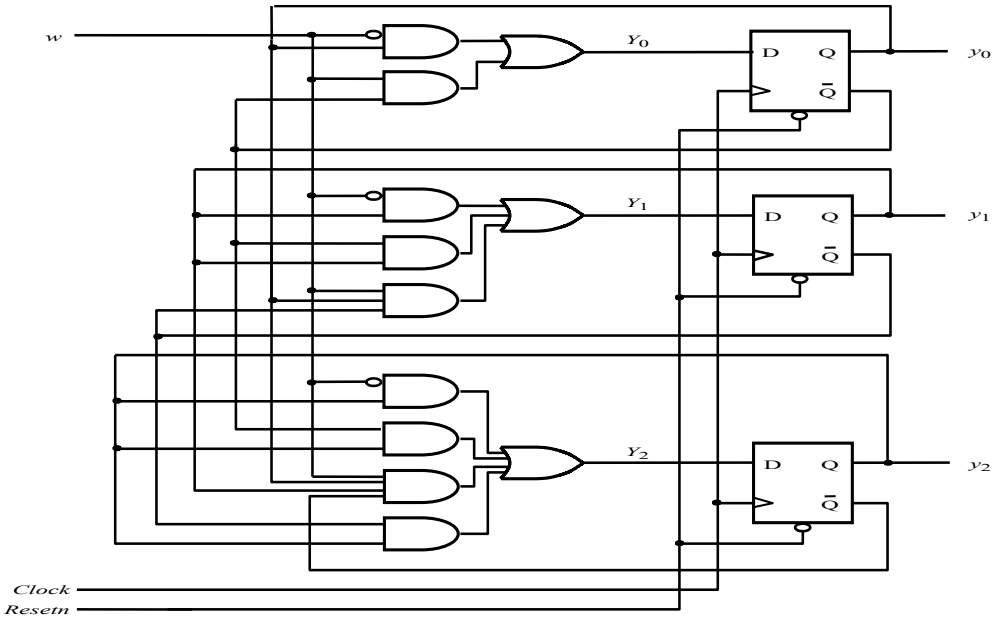
$Y_2 = \bar{w}y_2 + \bar{y}_0y_2 + \bar{y}_1y_2 + wy_0y_1\bar{y}_2$



$Y_1 = \bar{w}y_1 + y_1\bar{y}_0 + wy_0\bar{y}_1$

Counter Implementation with D

$Y_0 = \bar{w}y_0 + w\bar{y}_0$ $Y_1 = \bar{w}y_1 + y_1\bar{y}_0 + wy_0\bar{y}_1$ $Y_2 = \bar{w}y_2 + \bar{y}_0y_2 + \bar{y}_1y_2 + wy_0y_1\bar{y}_2$



Counter Implementation with JK

- The counting sequence is 0, 1, 2, 3, 4, 5, 6, 7, 0, 1…

	Present state $y_2y_1y_0$	Flip-flop inputs $w=0$ $Y_2Y_1Y_0$	J_2K_2	J_1K_1	J_0K_0	Flip-flop inputs $w=1$ $Y_2Y_1Y_0$	J_2K_2	J_1K_1	J_0K_0	Count $z_2z_1z_0$
A	000	000	0d	0d	0d	001	0d	0d	1d	000
B	001	001	0d	0d	d0	010	0d	1d	d1	001
C	010	010	0d	d0	0d	011	0d	d0	1d	010
D	011	011	0d	d0	d0	100	1d	d1	d1	011
E	100	100	d0	0d	0d	101	d0	0d	1d	100
F	101	101	d0	0d	d0	110	d0	1d	d1	101
G	110	110	d0	d0	0d	111	d0	d0	1d	110
H	111	111	d0	d0	d0	000	d1	d1	d1	111

J	K	$Q(t+1)$
0	0	$Q(t)$
0	1	0
1	0	1
1	1	$\bar{Q}(t)$

JK Flip-Flop Excitation Table			
q(t)	Q(t+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Counter Implementation with JK

	Present state $y_2y_1y_0$	Flip-flop inputs $w=0$ $Y_2Y_1Y_0$	J_2K_2	J_1K_1	J_0K_0	$w=1$ $Y_2Y_1Y_0$	J_2K_2	J_1K_1	J_0K_0	Count $z_2z_1z_0$
A	000	000	0d	0d	0d	001	0d	0d	1d	000
B	001	001	0d	0d	d0	010	0d	1d	d1	001
C	010	010	0d	d0	0d	011	0d	d0	1d	010
D	011	011	0d	d0	d0	100	1d	d1	d1	011
E	100	100	d0	0d	0d	101	d0	0d	1d	100
F	101	101	d0	0d	d0	110	d0	1d	d1	101
G	110	110	d0	d0	0d	111	d0	d0	1d	110
H	111	111	d0	d0	d0	000	d1	d1	d1	111

wy_2 \ y_1y_0

	00	01	11	10
00	0	0	0	0
01	d	d	d	d
11	d	d	d	d
10	0	0	1	0

$$J_2 = wy_0y_1$$

wy_2 \ y_1y_0

	00	01	11	10
00	d	d	d	d
01	0	0	0	0
11	0	0	1	0
10	d	d	d	d

$$K_2 = wy_0y_1$$

Counter Implementation with JK

	Present state $y_2y_1y_0$	Flip-flop inputs $w=0$ $Y_2Y_1Y_0$	J_2K_2	J_1K_1	J_0K_0	$w=1$ $Y_2Y_1Y_0$	J_2K_2	J_1K_1	J_0K_0	Count $z_2z_1z_0$
A	000	000	0d	0d	0d	001	0d	0d	1d	000
B	001	001	0d	0d	d0	010	0d	1d	d1	001
C	010	010	0d	d0	0d	011	0d	d0	1d	010
D	011	011	0d	d0	d0	100	1d	d1	d1	011
E	100	100	d0	0d	0d	101	d0	0d	1d	100
F	101	101	d0	0d	d0	110	d0	1d	d1	101
G	110	110	d0	d0	0d	111	d0	d0	1d	110
H	111	111	d0	d0	d0	000	d1	d1	d1	111

wy_2 \ y_1y_0

	00	01	11	10
00	0	0	d	d
01	0	0	d	d
11	0	1	d	d
10	0	1	d	d

$$J_1 = wy_0$$

wy_2 \ y_1y_0

	00	01	11	10
00	d	d	0	0
01	d	d	0	0
11	d	d	1	0
10	d	d	1	0

$$K_1 = wy_0$$

Counter Implementation with JK

	Present state $y_2y_1y_0$	Flip-flop inputs $w=0$ $Y_2Y_1Y_0$	J_2K_2	J_1K_1	J_0K_0	$w=1$ $Y_2Y_1Y_0$	J_2K_2	J_1K_1	J_0K_0	Count $z_2z_1z_0$
A	000	000	0d	0d	0d	001	0d	0d	1d	000
B	001	001	0d	0d	d0	010	0d	1d	d1	001
C	010	010	0d	d0	0d	011	0d	d0	1d	010
D	011	011	0d	d0	d0	100	1d	d1	d1	011
E	100	100	d0	0d	0d	101	d0	0d	1d	100
F	101	101	d0	0d	d0	110	d0	1d	d1	101
G	110	110	d0	d0	0d	111	d0	d0	1d	110
H	111	111	d0	d0	d0	000	d1	d1	d1	111

wy_2 \ y_1y_0

	00	01	11	10
00	0	d	d	0
01	0	d	d	0
11	1	d	d	1
10	1	d	d	1

$$J_0 = w$$

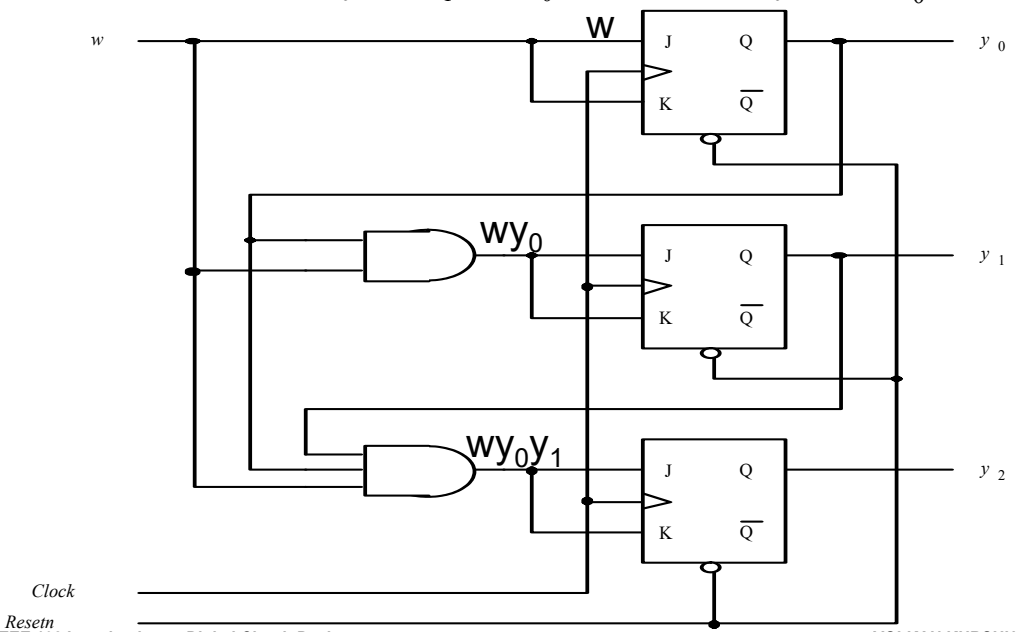
wy_2 \ y_1y_0

	00	01	11	10
00	d	0	0	d
01	d	0	0	d
11	d	1	1	d
10	d	1	1	d

$$K_0 = w$$

Counter Implementation with JK

$J_2 = wy_0y_1$ $K_2 = wy_0y_1$ $J_1 = wy_0$ $K_1 = wy_0$ $J_0 = w$ $K_0 = w$



Outline

- Counter Design as FSM
- A Different Counter
- Self Correcting Ring Counter
- Self Correcting Johnson Counter

A Different Counter Sequence

- The counting sequence is 0, 4, 2, 6, 1, 5, 3, 7, 0, 4…
- Assume the counter is always enabled

State Table

Present state	Next state	Output $z_2z_1z_0$
A	B	000
B	C	100
C	D	010
D	E	110
E	F	001
F	G	101
G	H	011
H	A	111

State Assigned Table

Present state $y_2y_1y_0$	Next state $Y_2Y_1Y_0$	Output $z_2z_1z_0$
000	100	000
100	010	100
010	110	010
110	001	110
001	101	001
101	011	101
011	111	011
111	000	111

Counter Implementation

y_2 \ y_1y_0	00	01	11	10
0	1	1	1	1
1	0	0	0	0

$Y_2 = y_2'$

y_2 \ y_1y_0	00	01	11	10
0	0	0	1	1
1	1	1	0	0

$Y_1 = y_2'y_1 + y_2y_1'$
$= y_2 \oplus y_1$

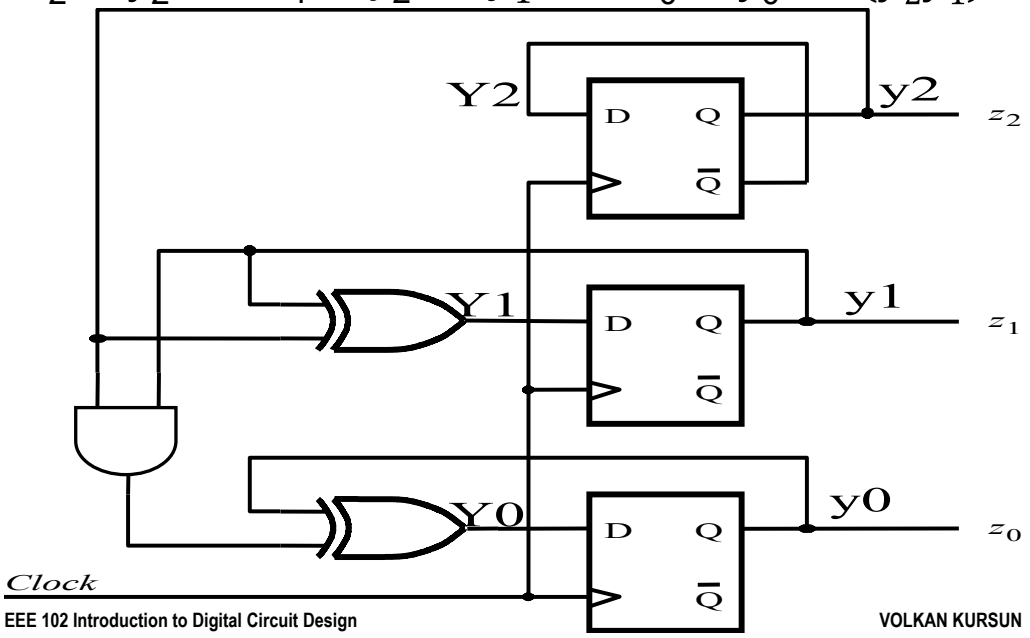
Present state $y_2y_1y_0$	Next state $Y_2Y_1Y_0$	Output $z_2z_1z_0$
000	100	000
100	010	100
010	110	010
110	001	110
001	101	001
101	011	101
011	111	011
111	000	111

y_2 \ y_1y_0	00	01	11	10
0	0	1	1	0
1	0	1	0	1

$Y_0 = y_2'y_0 + y_1'y_0 + y_2y_1y_0'$
$= y_0(y_1'+y_2') + y_0'y_2y_1 = y_0 \oplus (y_2y_1)$

Counter Implementation

$Y_2 = y_2'$ $Y_1 = y_2 \oplus y_1$ $Y_0 = y_0 \oplus (y_2y_1)$

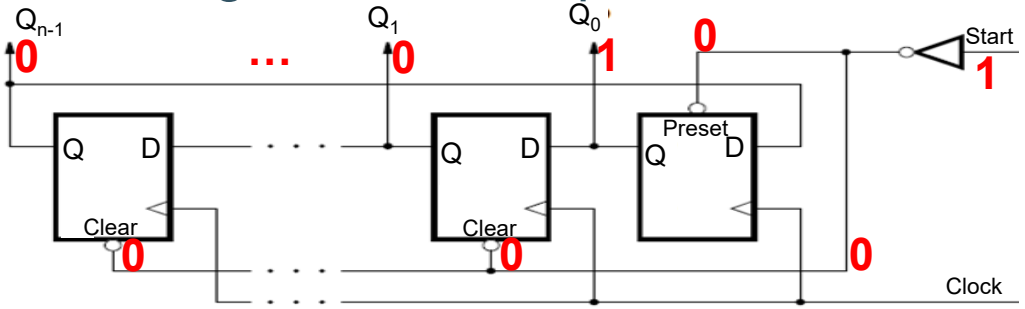


Outline

- Counter Design as FSM
- A Different Counter
- Self Correcting Ring Counter
- Self Correcting Johnson Counter

Shift Left Ring Counter

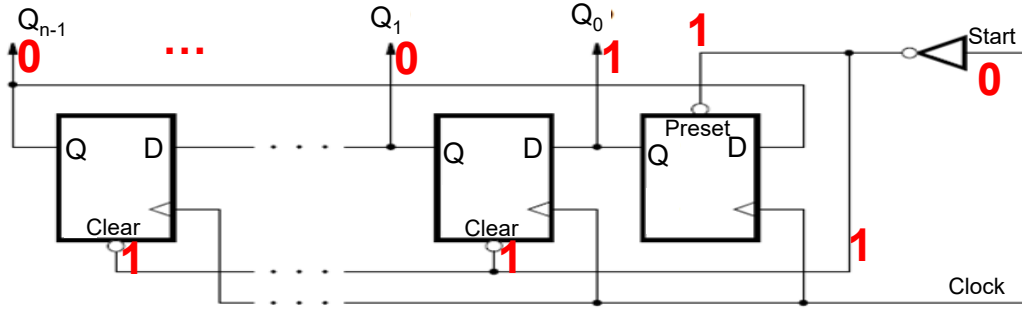
- 0001→0010→0100→1000→0001…
- Can be implemented with a shift-register with the following connections
- **Start = 1**: inject 1 into the LSB while clearing the other bit positions



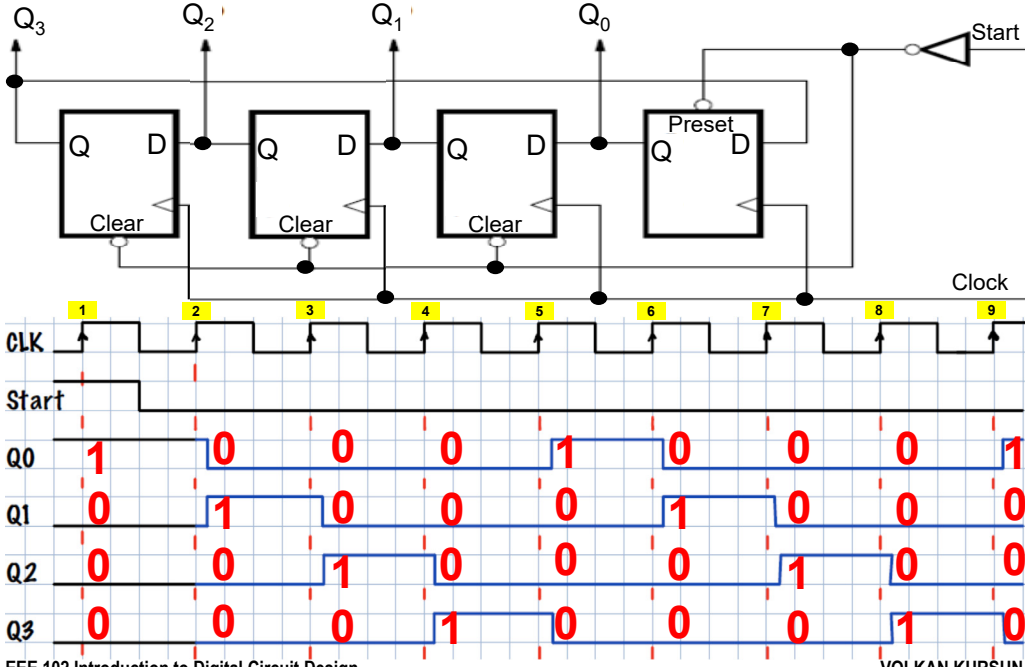
Shift Left Ring Counter

- 0001→0010→0100→1000→0001…
- **Start = 0**: count (shift left) with the positive edges of the clock

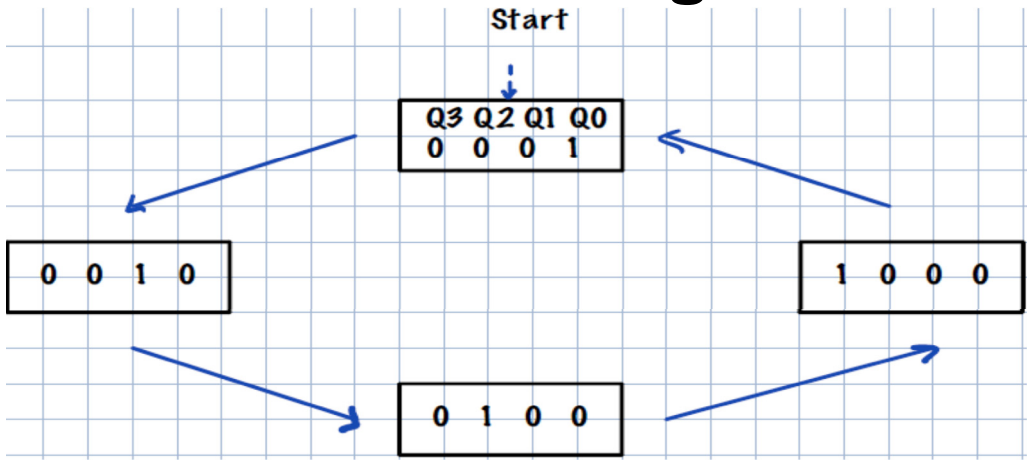
Q(N-1) <= Q(N-2), Q(N-2) <= Q(N-3),…,
Q(1) <= Q(0), Q(0) <= Q(N-1)



4-Bit Shift Left Ring Counter



4-Bit Shift Left Ring Counter



This is the normal mode of counting. In the normal mode, states are one-hot.

What is the total number of states? 16

What will happen if we enter a state which is not given above?

4-Bit Ring Counter State Table

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	d	d	d	d
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	d	d	d	d
0	1	0	0	1	0	0	0
0	1	0	1	d	d	d	d
0	1	1	0	d	d	d	d
0	1	1	1	d	d	d	d
1	0	0	0	0	0	0	1
1	0	0	1	d	d	d	d
1	0	1	0	d	d	d	d
1	0	1	1	d	d	d	d
1	1	0	0	d	d	d	d
1	1	0	1	d	d	d	d
1	1	1	0	d	d	d	d
1	1	1	1	d	d	d	d

- Counter **without self correction**: assume the counter will always go through the valid states. **Invalid states are don't care**
- **0001→0010→0100→1000→** 0001…

4-Bit Shift Left Ring Counter Maps

y_3y_2 \ y_1y_0	00	01	11	10
00	d	0	d	0
01	1	d	d	d
11	d	d	d	d
10	0	d	d	d

$Y_3 = y_2$

y_3y_2 \ y_1y_0	00	01	11	10
00	d	0	d	1
01	0	d	d	d
11	d	d	d	d
10	0	d	d	d

$Y_2 = y_1$

y_3y_2 \ y_1y_0	00	01	11	10
00	d	1	d	0
01	0	d	d	d
11	d	d	d	d
10	0	d	d	d

$Y_1 = y_0$

y_3y_2 \ y_1y_0	00	01	11	10
00	d	0	d	0
01	0	d	d	d
11	d	d	d	d
10	1	d	d	d

$Y_0 = y_3$

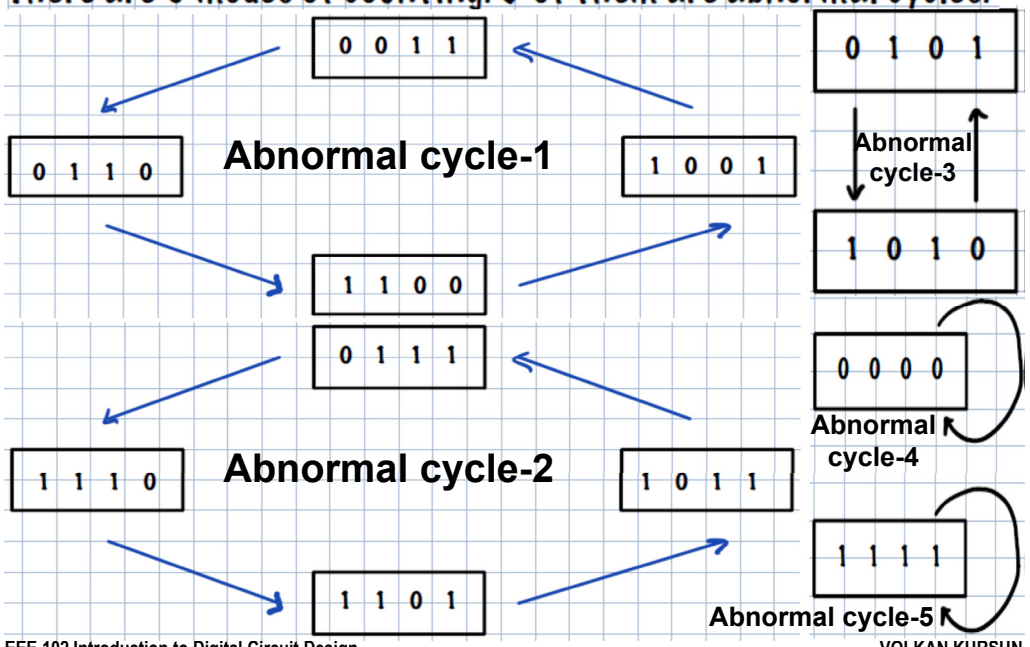
4-Bit Ring Counter State Table

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	**0**	**0**	**0**	**0**
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	**0**	**1**	**1**	**0**
0	1	0	0	1	0	0	0
0	1	0	1	**1**	**0**	**1**	**0**
0	1	1	0	**1**	**1**	**0**	**0**
0	1	1	1	**1**	**1**	**1**	**0**
1	0	0	0	0	0	0	1
1	0	0	1	**0**	**0**	**1**	**1**
1	0	1	0	**0**	**1**	**0**	**1**
1	0	1	1	**0**	**1**	**1**	**1**
1	1	0	0	**1**	**0**	**0**	**1**
1	1	0	1	**1**	**0**	**1**	**1**
1	1	1	0	**1**	**1**	**0**	**1**
1	1	1	1	**1**	**1**	**1**	**1**

- The state table after the don't care outputs are assigned values for logic minimization for the ring counter without error correction
- **0001→0010→0100→1000→**0001…

Shift Left Ring Counter Invalid Modes

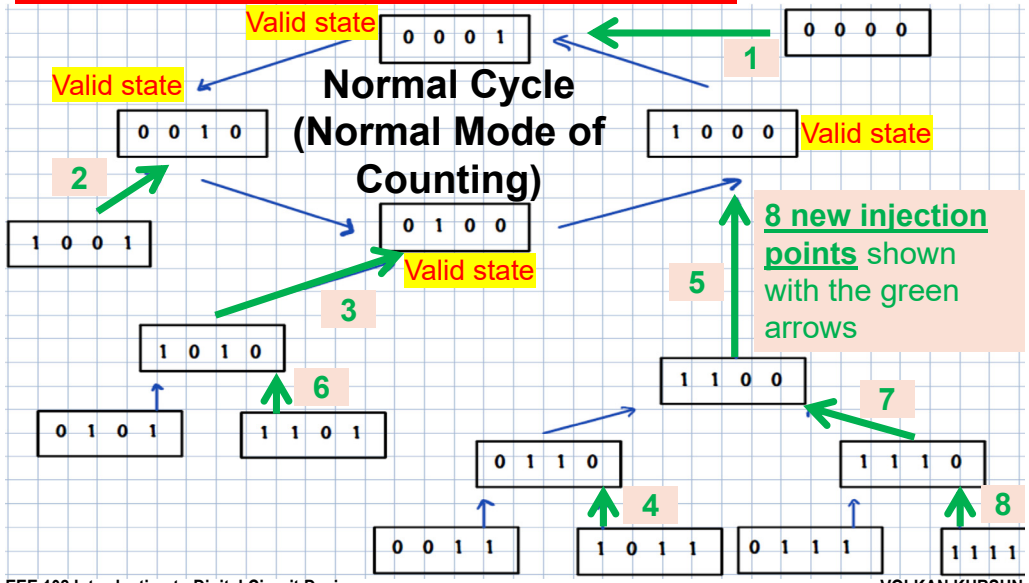
There are 6 modes of counting. 5 of them are abnormal cycles.



Self Correcting Ring Counter Cycles

Corrects abnormal state (moves to a valid state)

after at most 3 active clock edges



Ring Counter State Table Without Error Correction

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	0	0
0	1	0	1	1	0	1	0
0	1	1	0	1	1	0	0
0	1	1	1	1	1	1	0
1	0	0	0	0	0	0	1
1	0	0	1	0	0	1	1
1	0	1	0	0	1	0	1
1	0	1	1	0	1	1	1
1	1	0	0	1	0	0	1
1	1	0	1	1	0	1	1
1	1	1	0	1	1	0	1
1	1	1	1	1	1	1	1

- **Injection point-1: Invalid state 0000**
  - Without error correction, the next state is 0000
- **Injection point-2: Invalid state 1001**
  - Without error correction, the next state is 0011
- **Injection point-3: Invalid state 1010**
  - Without error correction, the next state is 0101
- **Injection point-4: Invalid state 1011**
  - Without error correction, the next state is 0111

Ring Counter State Table Without Error Correction

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	1	0
0	1	0	0	1	0	0	0
0	1	0	1	1	0	1	0
0	1	1	0	1	1	0	0
0	1	1	1	1	1	1	0
1	0	0	0	0	0	0	1
1	0	0	1	0	0	1	1
1	0	1	0	0	1	0	1
1	0	1	1	0	1	1	1
1	1	0	0	1	0	0	1
1	1	0	1	1	0	1	1
1	1	1	0	1	1	0	1
1	1	1	1	1	1	1	1

- **Injection point-5: Invalid state 1100**
  - Without error correction, the next state is 1001
- **Injection point-6: Invalid state 1101**
  - Without error correction, the next state is 1011
- **Injection point-7: Invalid state 1110**
  - Without error correction, the next state is 1101
- **Injection point-8: Invalid state 1111**
  - Without error correction, the next state is 1111

Self Correcting Ring Counter State Table

	Present State				Next State			
	y3	y2	y1	y0	Y3	Y2	Y1	Y0
1	0	0	0	0	0	0	0	1
	0	0	0	1	0	0	1	0
	0	0	1	0	0	1	0	0
	0	0	1	1	0	1	1	0
	0	1	0	0	1	0	0	0
	0	1	0	1	1	0	1	0
	0	1	1	0	1	1	0	0
	0	1	1	1	1	1	1	0
	1	0	0	0	0	0	0	1
2	1	0	0	1	0	0	1	0
3	1	0	1	0	0	1	0	0
4	1	0	1	1	0	1	1	0
	1	1	0	0	1	0	0	1
	1	1	0	1	1	0	1	1
	1	1	1	0	1	1	0	1
	1	1	1	1	1	1	1	1

- **Injection point-1: Invalid state 0000**
  - For error correction, the next state is **0001**
- **Injection point-2: Invalid state 1001**
  - For error correction, the next state is **0010**
- **Injection point-3: Invalid state 1010**
  - For error correction, the next state is **0100**
- **Injection point-4: Invalid state 1011**
  - For error correction, the next state is **0110**

Self Correcting Ring Counter State Table

	Present State				Next State			
	y3	y2	y1	y0	Y3	Y2	Y1	Y0
	0	0	0	0	0	0	0	1
	0	0	0	1	0	0	1	0
	0	0	1	0	0	1	0	0
	0	0	1	1	0	1	1	0
	0	1	0	0	1	0	0	0
	0	1	0	1	1	0	1	0
	0	1	1	0	1	1	0	0
	0	1	1	1	1	1	1	0
	1	0	0	0	0	0	0	1
	1	0	0	1	0	0	1	0
	1	0	1	0	0	1	0	0
	1	0	1	1	0	1	1	0
5	1	1	0	0	1	0	0	0
6	1	1	0	1	1	0	1	0
7	1	1	1	0	1	1	0	0
8	1	1	1	1	1	1	1	0

- **Injection point-5: Invalid state 1100**
  - For error correction, the next state is **1000**
- **Injection point-6: Invalid state 1101**
  - For error correction, the next state is **1010**
- **Injection point-7: Invalid state 1110**
  - For error correction, the next state is **1100**
- **Injection point-8: Invalid state 1111**
  - For error correction, the next state is **1110**

Error Correcting Ring Counter Maps

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	1	1
10	0	0	0	0

$Y_3 = y_2$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	0	0	1	1
10	0	0	1	1

$Y_2 = y_1$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	0

$Y_1 = y_0$

y_3y_2 \ y_1y_0	00	01	11	10
00	1	0	0	0
01	0	0	0	0
11	0	0	0	0
10	1	0	0	0

$Y_0 = y_2'y_1'y_0' = (y_2+y_1+y_0)'$

Self Correcting Ring Counter

1000→**0001**→0010→0100→**1000**→**0001**…

- $Q_{N-2} = Q_{N-3} = \ldots = Q_1 = Q_0 = 0$ is the only legitimate state after which a 1 should be injected into Q_0
- From all other states, including the erroneous states, 0 should be injected into Q_0 to be able to return to the normal cycles
- To construct an n-bit self correcting ring counter: $D_0 = (Q_{n-2} + Q_{n-3} + \ldots Q_1 + Q_0)'$

Self Correcting Ring Counter

1000→**0001**→0010→0100→**1000**→**0001**…

- Assume an abnormal state with n bits:

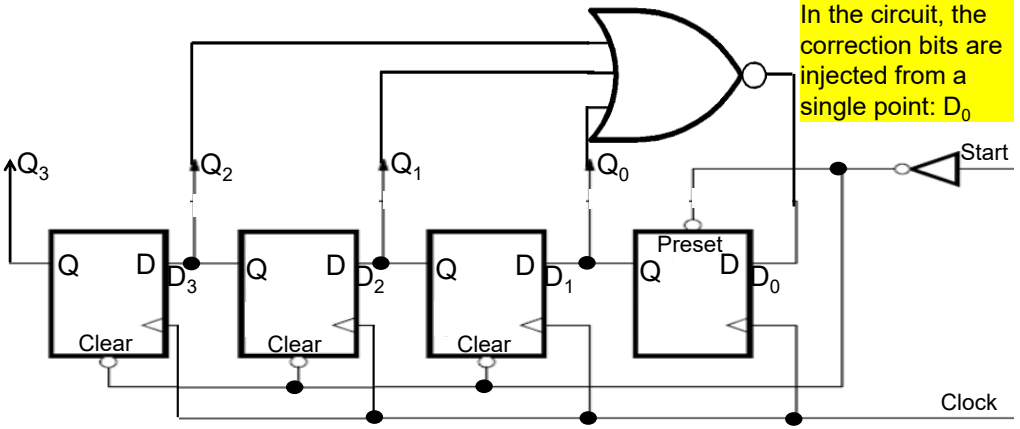
X111….1111 $\xrightarrow{1}$ 1111….1110 $\xrightarrow{2}$ 1111….1100
$\xrightarrow{3}$ 1111….1000 $\xrightarrow{4}$ … $\xrightarrow{n-2}$ 1100….0000
$\xrightarrow{n-1}$ 1**000….0000** $\xrightarrow{n}$ 0000….0001
→0000….0010→…

- To construct an n-bit self correcting ring counter:

$D_0 = (Q_{n-2} + Q_{n-3} + \dots Q_1 + Q_0)'$

NOR function of all the bits (all state variables) except the MSB
$D_0 = 1$, iff $Q_0 = Q_1 = \dots = Q_{N-2} = 0$
$D_0 = 0$, for all other states → corrects an abnormal state in at most n-1 clock cycles

Self Correcting 4-Bit Ring Counter



- To construct an n-bit self correcting ring counter:

$D_0 = (Q_2 + Q_1 + Q_0)'$

NOR function of all the bits (all state variables) except the MSB

$D_0 = 1$ iff $Q_2 = Q_1 = Q_0 = 0$ (corrects an abnormal state in at most 3 clock cycles)

Ring Counter State Table for Single Cycle Error Correction

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	1	0	0
0	0	1	1	0	0	1	0
0	1	0	0	1	0	0	0
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	0
0	1	1	1	1	0	0	0
1	0	0	0	0	0	0	1
1	0	0	1	0	0	0	1
1	0	1	0	0	0	0	1
1	0	1	1	0	0	1	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

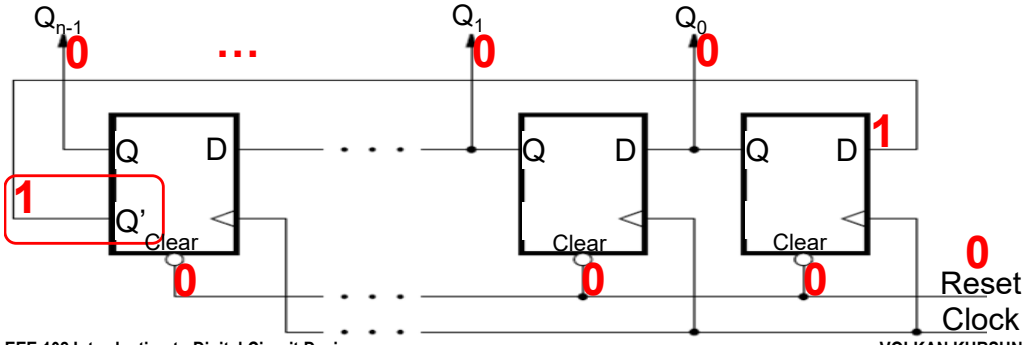
- **For fastest recovery**, design a circuit that can **recover from any invalid state in only one clock cycle**
- **All the invalid states must transition to a valid state in the next clock cycle** for one cycle recovery
- **More complex circuit needed**: logic expressions for Y_2, Y_1, and Y_0 change
- No change in Y_3: $Y_3 = y_2$

Outline

- Counter Design as FSM
- A Different Counter
- Self Correcting Ring Counter
- Self Correcting Johnson Counter

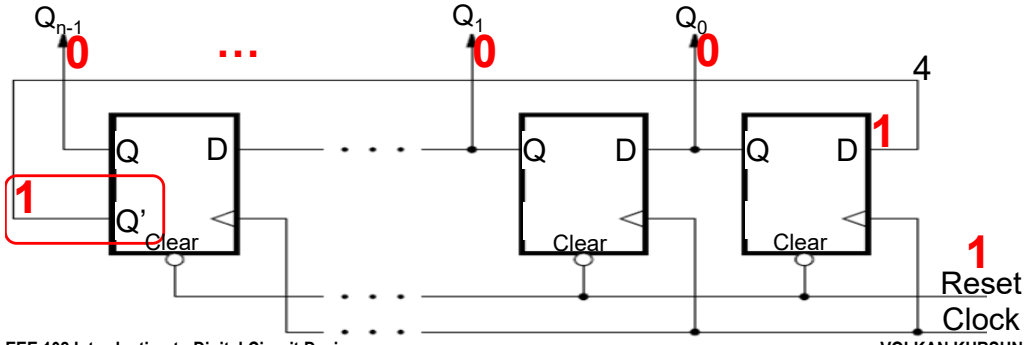
Shift Left Johnson Counter

- Instead of the Q output of the final stage as in a ring counter, connect the Q' output of the final stage to the D input of the first stage flip-flop in a Johnson counter (AKA twisted ring, switched tail)
- First **initialize** with **Reset = 0** (**clear all bit positions to 0**): $Q_{n-1}Q_{n-2}\ldots Q_1Q_0$ = 0b00…00

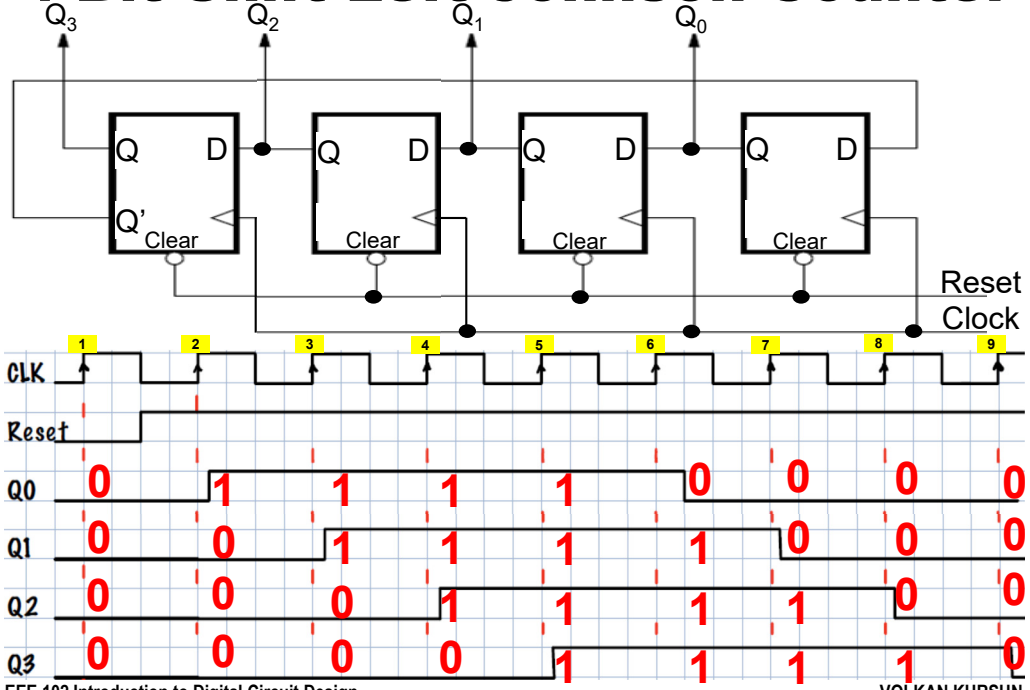


Shift Left Johnson Counter

- When **Reset = 1**, counter **starts counting** with the positive edges of the clock signal
- n-bit Johnson counter (**shift left with Q_{n-1}' (MSB complement) injected to LSB from right**):
  00..00 $\xrightarrow{1}$ 00..01 $\xrightarrow{2}$ 00..11 → … → 01..11 $\xrightarrow{n}$ 11..11 → 11..10 → 11..00 → … → 10..00 $\xrightarrow{2n}$ **00..00** → 00..01 → …

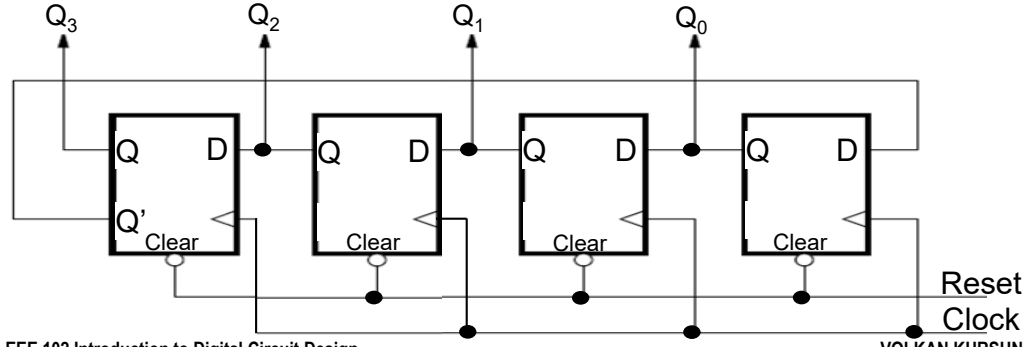


4-Bit Shift Left Johnson Counter



Johnson Counter States

- An n-bit Johnson counter goes through 2n valid states
- There are 2^n-2n invalid states
- Example: 4-bit Johnson counter goes through **8 valid states**. There are also **8 invalid states**.
- **0000→0001→0011→0111→1111→1110→1100→1000**→0000→0001…



4-Bit Johnson Counter State Table

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	0	d	d	d	d
0	0	1	1	0	1	1	1
0	1	0	0	d	d	d	d
0	1	0	1	d	d	d	d
0	1	1	0	d	d	d	d
0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
1	0	0	1	d	d	d	d
1	0	1	0	d	d	d	d
1	0	1	1	d	d	d	d
1	1	0	0	1	0	0	0
1	1	0	1	d	d	d	d
1	1	1	0	1	1	0	0
1	1	1	1	1	1	1	0

- Counter without self correction: assume the counter will always go through the valid states. **Invalid states are don't care**
- **0000→0001→0011→0111→1111→1110→1100→1000→**0000→0001…

4-Bit Johnson Counter Maps

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	0	d
01	d	d	1	d
11	1	d	1	1
10	0	d	d	d

$Y_3 = y_2$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	1	d
01	d	d	1	d
11	0	d	1	1
10	0	d	d	d

$Y_2 = y_1$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	1	1	d
01	d	d	1	d
11	0	d	1	0
10	0	d	d	d

$Y_1 = y_0$

y_3y_2 \ y_1y_0	00	01	11	10
00	1	1	1	d
01	d	d	1	d
11	0	d	0	0
10	0	d	d	d

$Y_0 = y_3'$

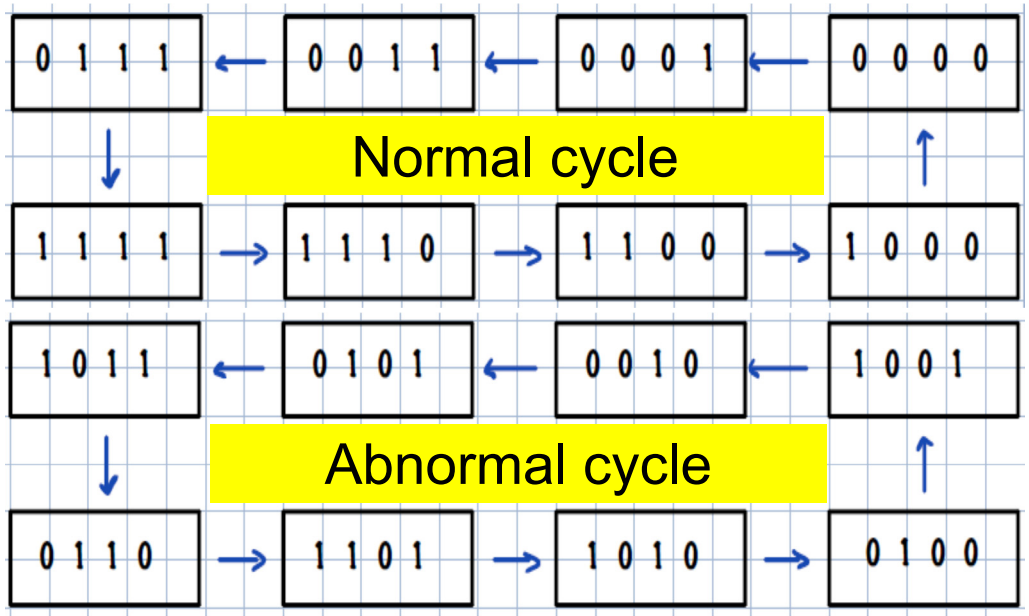
4-Bit Johnson Counter State Table

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	1
0	1	0	0	1	0	0	1
0	1	0	1	1	0	1	1
0	1	1	0	1	1	0	1
0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	1	0
1	1	1	0	1	1	0	0
1	1	1	1	1	1	1	0

- After the assignments of the don't cares for the next state logic minimization, the state table looks like this
- **0000→0001→0011→0111→1111→1110→1100→1000→**0000→0001…

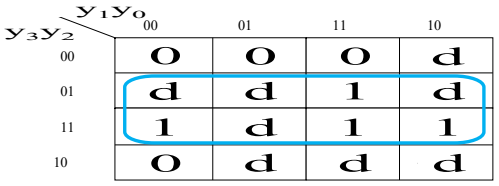
Johnson Counter Cycles

- Original design without error correction capability

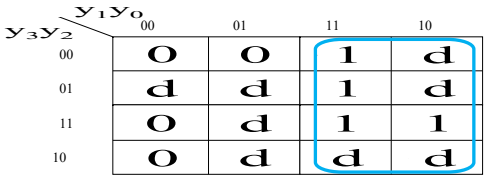


Error Correcting Johnson Counter Design

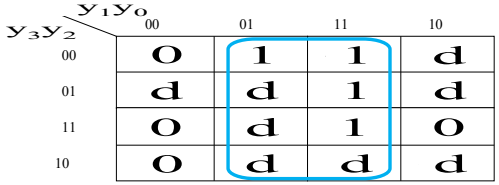
- To design a self-correcting Johnson counter, examine the existing Karnaugh maps of the original design and **identify don't care conditions that can be changed to recover from erroneous states with the minimum hardware overhead**



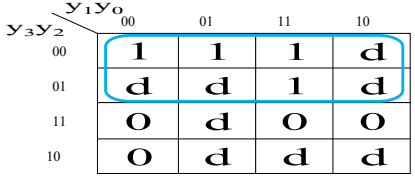
Y3 = y2



Y2 = y1



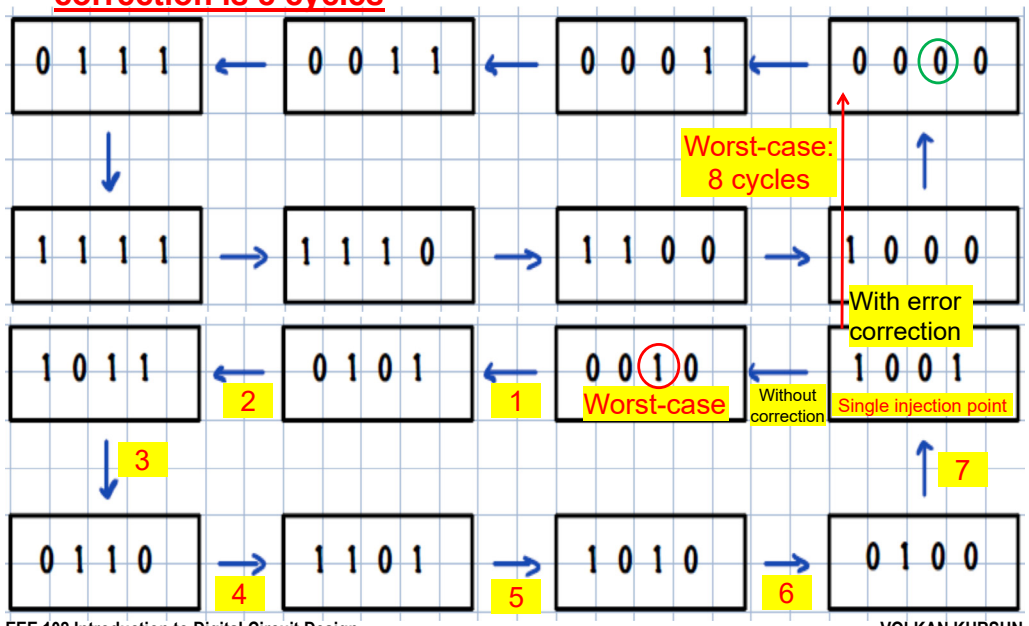
Y1 = y0



Y0 = y3'

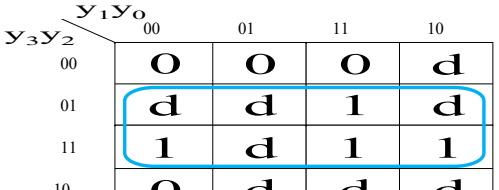
Error Correcting Johnson Counter

- **With single injection point: worst case latency for error correction is 8 cycles**

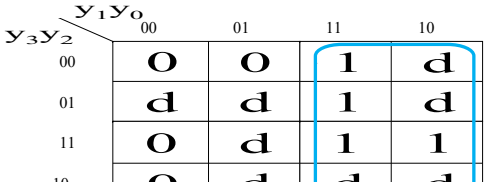


Error Correcting Johnson Counter Design-1

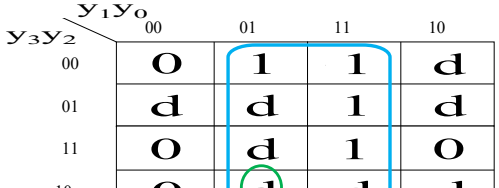
- In design-1, we will explore having only one injection point
- Replace the highlighted don't care condition for Y1 from 1 to 0 to recover from erroneous state with minimum hardware overhead
- **Form three new groups of 4 instead of one group of 8**



Y3 = y2

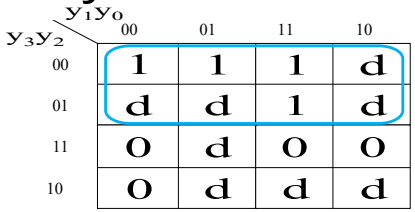


Y2 = y1



Y1 = y0

Change from 1 to 0 in the self-correcting design



Y0 = y3'

Original State Table Without Error Correction

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	1
0	1	0	0	1	0	0	1
0	1	0	1	1	0	1	1
0	1	1	0	1	1	0	1
0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	1	0
1	1	1	0	1	1	0	0
1	1	1	1	1	1	1	0

- **Single injection point: Invalid state 1001**
  - Without error correction, the next state is **0010**

Modified State Table-1 for Error Correction

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	1
0	1	0	0	1	0	0	1
0	1	0	1	1	0	1	1
0	1	1	0	1	1	0	1
0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	1	0
1	1	1	0	1	1	0	0
1	1	1	1	1	1	1	0

- **Single injection point: Invalid state 1001**
  - **With error correction**, the next state is **0000**

Error Correcting Johnson Counter-1 Maps

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	1	1
10	0	0	0	0

$Y_3 = y_2$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	0	0	1	1
10	0	0	1	1

$Y_2 = y_1$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	0	1	0

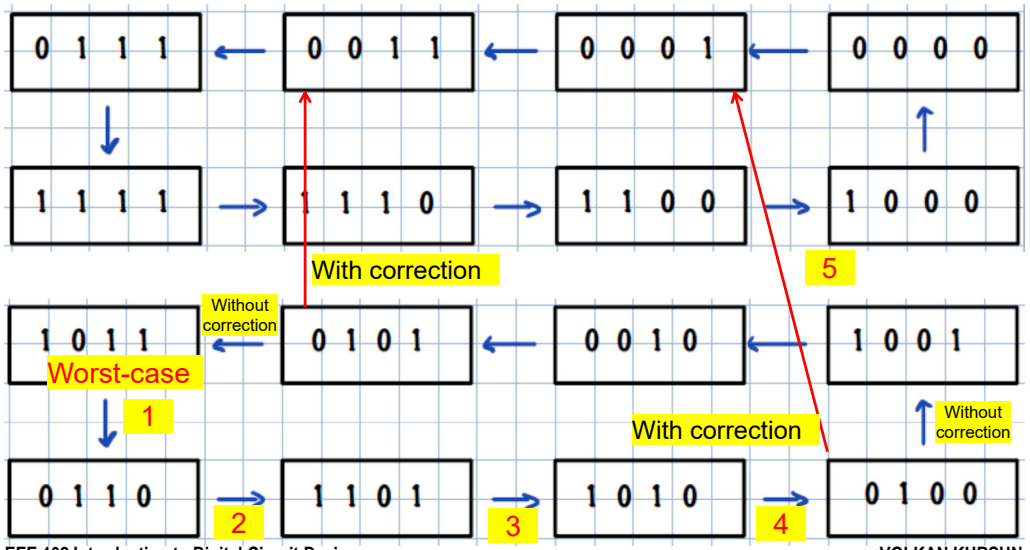
$Y_1 = y_3'y_0 + y_2y_0 + y_1y_0$

y_3y_2 \ y_1y_0	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	0	0	0
10	0	0	0	0

$Y_0 = y_3'$

Self Correcting Johnson Counter Design-2

- **Multiple injection points reduce the latency for correction**
- **Two injection points for error correction**: worst case latency for correction is reduced to **5 cycles**



Error Correcting Johnson Counter Design-2

- In design option-2, we will replace the highlighted don't care conditions for Y3 from 1 to 0 to recover from erroneous states with minimum hardware overhead
- **Form two new groups of 4 instead of one group of 8**

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	0	d
01	d	d	1	d
11	1	d	1	1
10	0	d	d	d

$Y3 = y2$

Change from 1 to 0 in the self-correcting design

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	1	d
01	d	d	1	d
11	0	d	1	1
10	0	d	d	d

$Y2 = y1$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	1	1	d
01	d	d	1	d
11	0	d	0	0
10	0	d	d	d

$Y1 = y0$

y_3y_2 \ y_1y_0	00	01	11	10
00	1	1	1	d
01	d	d	1	d
11	0	d	0	0
10	0	d	d	d

$Y0 = y3'$

State Table Without Correction

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	1
0	1	0	0	1	0	0	1
0	1	0	1	1	0	1	1
0	1	1	0	1	1	0	1
0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	1	0
1	1	1	0	1	1	0	0
1	1	1	1	1	1	1	0

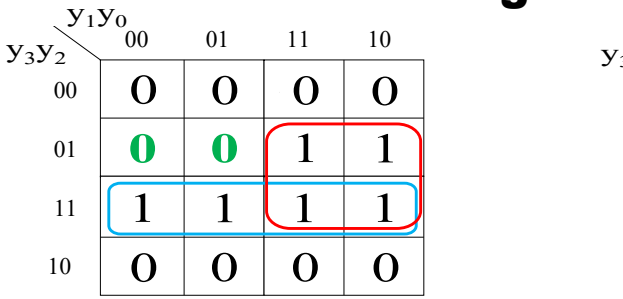
- **Injection point-1: Invalid state 0100**
  - Without error correction, the next state is **1001**
- **Injection point-2: Invalid state 0101**
  - Without error correction, the next state is **1011**

Design-2 for Error Correction

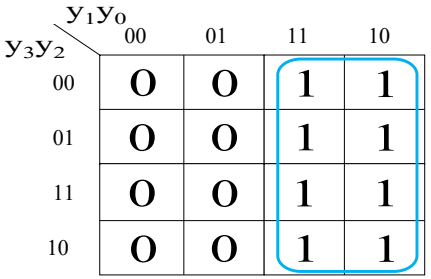
	Present State				Next State			
	y3	y2	y1	y0	Y3	Y2	Y1	Y0
	0	0	0	0	0	0	0	1
	0	0	0	1	0	0	1	1
	0	0	1	0	0	1	0	1
	0	0	1	1	0	1	1	1
1	0	1	0	0	0	0	0	1
2	0	1	0	1	0	0	1	1
	0	1	1	0	1	1	0	1
	0	1	1	1	1	1	1	1
	1	0	0	0	0	0	0	0
	1	0	0	1	0	0	1	0
	1	0	1	0	0	1	0	0
	1	0	1	1	0	1	1	0
	1	1	0	0	1	0	0	0
	1	1	0	1	1	0	1	0
	1	1	1	0	1	1	0	0
	1	1	1	1	1	1	1	0

- **Injection point-1: Invalid state 0100**
  - **With error correction**, the next state is **0001**
- **Injection point-2: Invalid state 0101**
  - **With error correction**, the next state is **0011**
- Y_3 logic function changes
- Y_2, Y_1, and Y_0 do NOT change

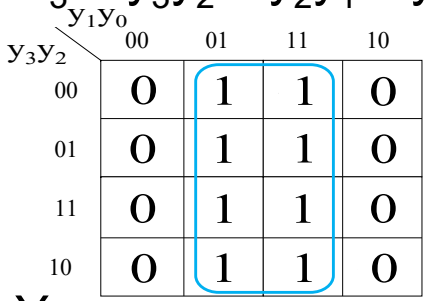
Error Correcting Design-2 Maps



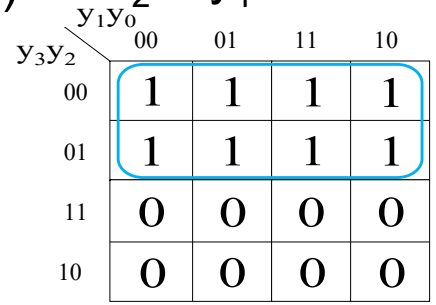
$Y_3 = y_3y_2 + y_2y_1 = y_2(y_3+y_1)$



$Y_2 = y_1$



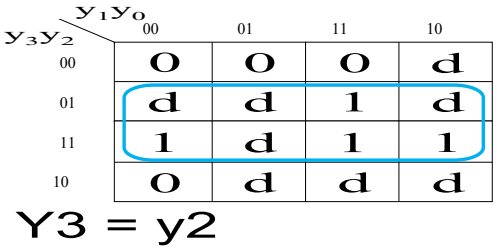
$Y_1 = y_0$



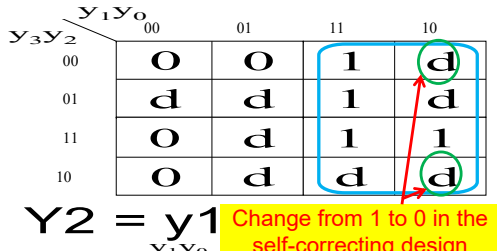
$Y_0 = y_3'$

Error Correcting Johnson Counter Design-3

- In design option-3, we will replace the highlighted don't care conditions for Y2 from 1 to 0 to recover from erroneous states with minimum hardware overhead
- **Form two new groups of 4 instead of one group of 8**

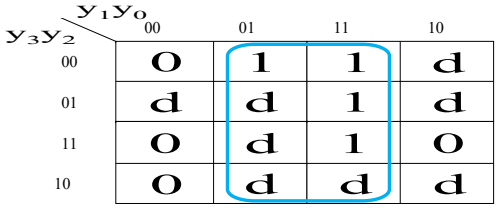


Y3 = y2

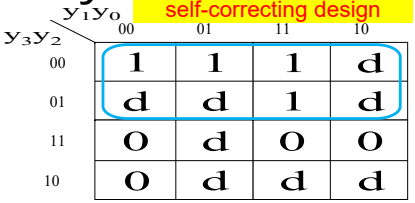


Y2 = y1

Change from 1 to 0 in the self-correcting design



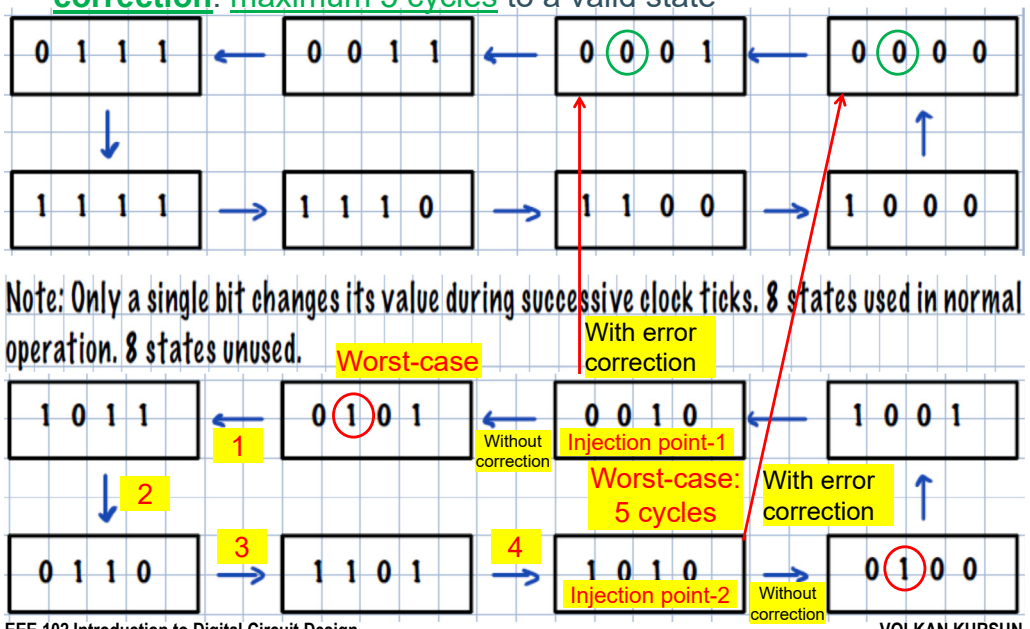
Y1 = y0



Y0 = y3'

4-Bit Johnson Counter State Diagram

- Two injection points for error correction lower the latency for correction: maximum 5 cycles to a valid state



4-Bit Johnson Counter State Table

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	1
0	1	0	0	1	0	0	1
0	1	0	1	1	0	1	1
0	1	1	0	1	1	0	1
0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	1	0
1	1	1	0	1	1	0	0
1	1	1	1	1	1	1	0

- **Injection point-1: Invalid state 0010**
  - Without error correction, the next state is **0101**
- **Injection point-2: Invalid state 1010**
  - Without error correction, the next state is **0100**

Modified State Table for Error Correction

	Present State				Next State			
	y3	y2	y1	y0	Y3	Y2	Y1	Y0
	0	0	0	0	0	0	0	1
	0	0	0	1	0	0	1	1
1	0	0	1	0	0	0	0	1
	0	0	1	1	0	1	1	1
	0	1	0	0	1	0	0	1
	0	1	0	1	1	0	1	1
	0	1	1	0	1	1	0	1
	0	1	1	1	1	1	1	1
	1	0	0	0	0	0	0	0
	1	0	0	1	0	0	1	0
2	1	0	1	0	0	0	0	0
	1	0	1	1	0	1	1	0
	1	1	0	0	1	0	0	0
	1	1	0	1	1	0	1	0
	1	1	1	0	1	1	0	0
	1	1	1	1	1	1	1	0

- **Injection point-1: Invalid state 0010**
  - **With error correction**, the next state is **0001**
- **Injection point-2: Invalid state 1010**
  - **With error correction**, the next state is **0000**
- Y_2 logic function changes
- Y_3, Y_1, and Y_0 do NOT change

Error Correcting Johnson Counter-3 Maps

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	1	1
10	0	0	0	0

$Y_3 = y_2$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	1	0
01	0	0	1	1
11	0	0	1	1
10	0	0	1	0

$Y_2 = y_2y_1 + y_1y_0 = y_1(y_2 + y_0)$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	0

$Y_1 = y_0$

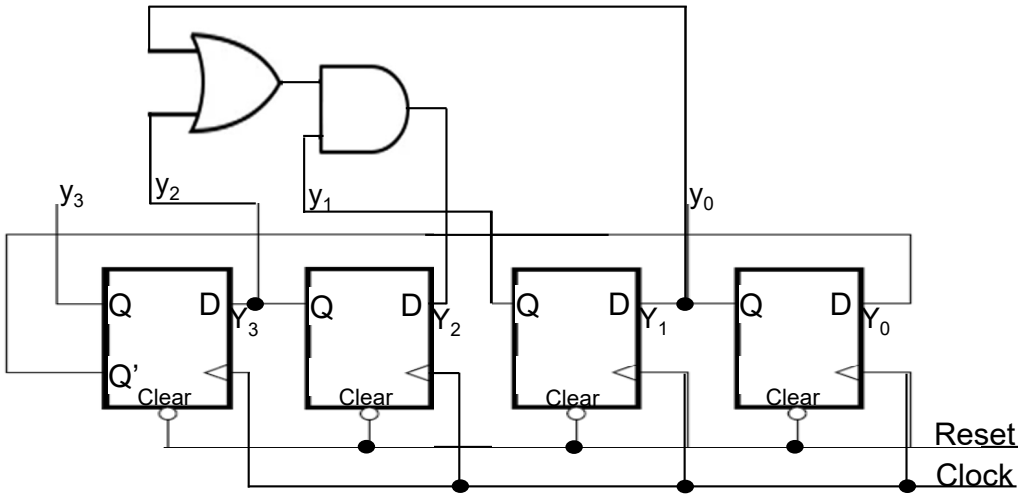
y_3y_2 \ y_1y_0	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	0	0	0
10	0	0	0	0

$Y_0 = y_3'$

Self Correcting Johnson Counter-3

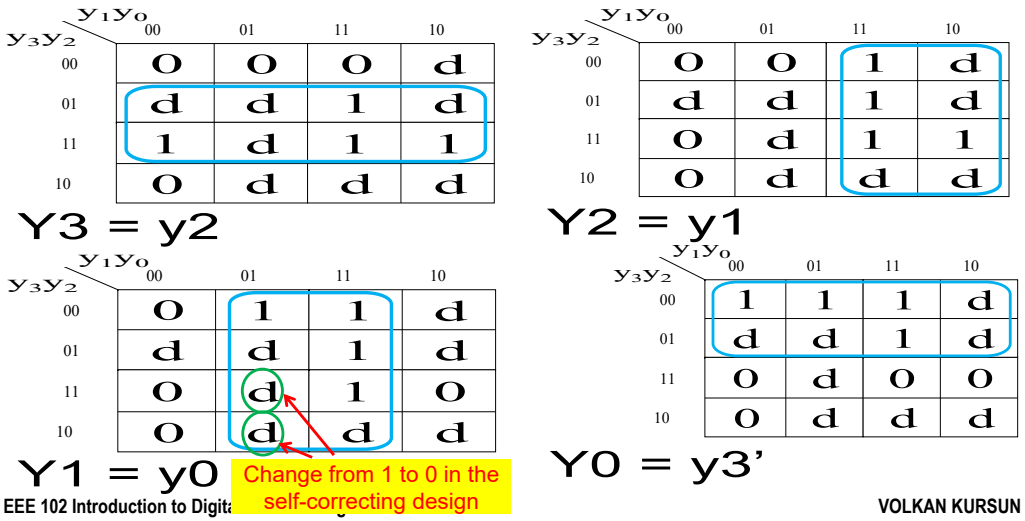
$Y_3 = y_2$ $Y_2 = y_2y_1+y_1y_0 = y_1(y_2 + y_0)$

$Y_1 = y_0$ $Y_0 = y_3'$



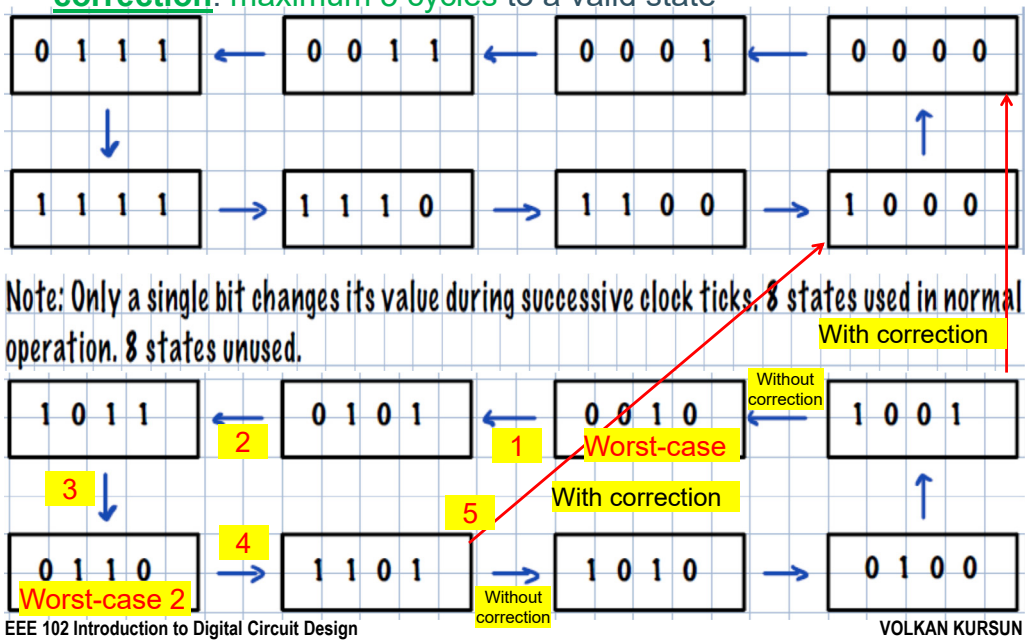
Error Correcting Johnson Counter Design-4

- In design option-4, we will replace the highlighted don't care conditions for Y1 from 1 to 0 to recover from erroneous states with minimum hardware overhead
- **Form two new groups of 4 instead of one group of 8**



Self Correcting Johnson Counter Alternative Design-4

- **Two injection points** for error correction **lower the latency for correction**: maximum 5 cycles to a valid state



State Table Without Correction

Present State				Next State			
y3	y2	y1	y0	Y3	Y2	Y1	Y0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	1
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	1
0	1	0	0	1	0	0	1
0	1	0	1	1	0	1	1
0	1	1	0	1	1	0	1
0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0
1	0	0	1	0	0	1	0
1	0	1	0	0	1	0	0
1	0	1	1	0	1	1	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	1	0
1	1	1	0	1	1	0	0
1	1	1	1	1	1	1	0

- **Injection point-1: Invalid state 1001**
  - Without error correction, the next state is **0010**
- **Injection point-2: Invalid state 1101**
  - Without error correction, the next state is **1010**

Design-4 for Error Correction

	Present State				Next State			
	y3	y2	y1	y0	Y3	Y2	Y1	Y0
	0	0	0	0	0	0	0	1
	0	0	0	1	0	0	1	1
	0	0	1	0	0	1	0	1
	0	0	1	1	0	1	1	1
	0	1	0	0	1	0	0	1
	0	1	0	1	1	0	1	1
	0	1	1	0	1	1	0	1
	0	1	1	1	1	1	1	1
	1	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	0
	1	0	1	0	0	1	0	0
	1	0	1	1	0	1	1	0
	1	1	0	0	1	0	0	0
2	1	1	0	1	1	0	0	0
	1	1	1	0	1	1	0	0
	1	1	1	1	1	1	1	0

- **Injection point-1: Invalid state 1001**
  - **With error correction**, the next state is **0000**
- **Injection point-2: Invalid state 1101**
  - **With error correction**, the next state is **1000**
- Y_1 logic function changes
- Y_3, Y_2, and Y_0 do NOT change

Error Correcting Design-4 Maps

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	1	1
10	0	0	0	0

$Y_3 = y_2$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	0	0	1	1
10	0	0	1	1

$Y_2 = y_1$

y_3y_2 \ y_1y_0	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	0	1	0
10	0	0	1	0

$Y_1 = y_3'y_0 + y_1y_0 = y_0(y_3' + y_1)$

y_3y_2 \ y_1y_0	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	0	0	0
10	0	0	0	0

$Y_0 = y_3'$

Johnson Counter State Table for Single Cycle Error Correction

	Present State				Next State			
	y3	y2	y1	y0	Y3	Y2	Y1	Y0
	0	0	0	0	0	0	0	1
	0	0	0	1	0	0	1	1
	0	0	1	0	0	0	0	1
	0	0	1	1	0	1	1	1
	0	1	0	0	1	0	0	0
	0	1	0	1	1	1	1	1
	0	1	1	0	1	1	0	0
	0	1	1	1	1	1	1	1
	1	0	0	0	0	0	0	0
	1	0	0	1	0	0	1	1
	1	0	1	0	0	0	0	0
	1	0	1	1	0	1	1	1
	1	1	0	0	1	0	0	0
	1	1	0	1	1	1	1	0
	1	1	1	0	1	1	0	0
	1	1	1	1	1	1	1	0

- Modified design that can **recover from any invalid state in one clock cycle**
- **All the invalid states must transition to a valid state in the next clock cycle**
- **More complex circuit needed for 1 cycle recovery**
- Y_2 and Y_0 logic expressions change
- No change in Y_3:
  $Y_3 = y_2$
- No change in Y1:
  $Y_1 = y_0$