

# Digital Logic Theory Assignment 4

12312706 Zhou Liangyu

1. State table:

$x$	$A$	$B$	$A(t+1)$	$B(t+1)$	$T_A$	$T_B$
0	0	0	0	1	0	1
0	0	1	1	1	1	0
0	1	0	1	1	0	1
0	1	1	0	0	1	1
1	0	0	0	0	0	0
1	0	1	1	0	1	1
1	1	0	1	0	0	0
1	1	1	1	1	0	0

	$x$	
	0	1
$AB$	00	0
	01	1
	11	1
	10	0

$$T_A = B(x' + A')$$

	$x$	
	0	1
$AB$	00	1
	01	0
	11	1
	10	1

$$T_B = x'(A + B') + xA'B$$

$$\therefore A(t+1) = T'_A A(t) + T_A A'(t) = (B' + xA)A + A'B(x' + A') = A \oplus B + x'A'B,$$

$$B(t+1) = T'_B B(t) + T_B B'(t) = x(A + B') + x'A'B.$$

2. State table:

Present State	Inputs		Next State	Flip-Flop Input
$Q$	$G$	$In$	$Q$	$D$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1

Present State	Inputs		Next State	Flip-Flop Input
1	0	0	1	1
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

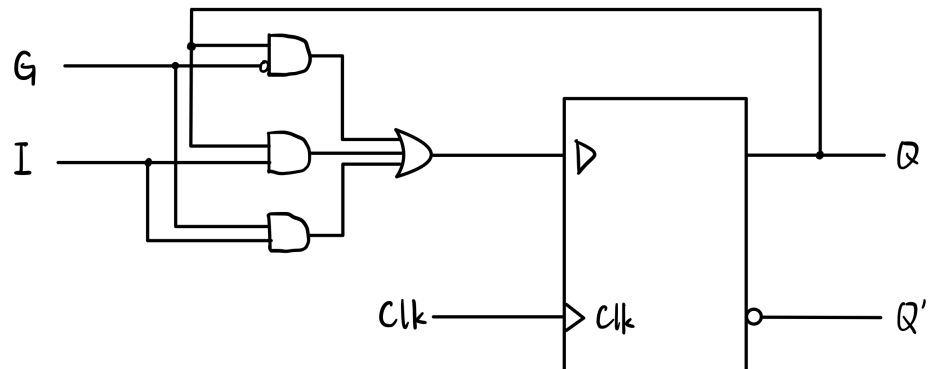
Characteristic Table:

$G$	$In$	$Q(t+1)$
0	x	$Q(t)$
1	0	0
1	1	1

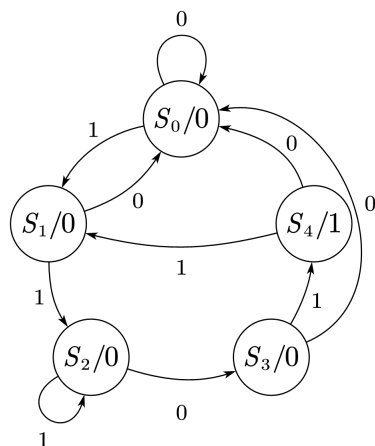
		$Q(t)$	
		0	1
$GIn$	00	0	1
	01	0	1
	11	1	1
	10	0	0

$\therefore D = Q(t+1) = G'Q(t) + GIn.$

Logic Implementation:



3. State diagram:



State table:  $S_0 = 000, S_1 = 001, S_2 = 010, S_3 = 011, S_4 = 100$ .

Present	State		Input	Next	State		Flip-Flop	Inputs				
$Q_2$	$Q_1$	$Q_0$	x	$Q_2$	$Q_1$	$Q_0$	$J_{Q_2}$	$K_{Q_2}$	$J_{Q_1}$	$K_{Q_1}$	$J_{Q_0}$	$K_{Q_0}$
0	0	0	0	0	0	0	0	x	0	x	0	x
0	0	0	1	0	0	1	0	x	0	x	1	x
0	0	1	0	0	0	0	0	x	0	x	x	1
0	0	1	1	0	1	0	0	x	1	x	x	1
0	1	0	0	0	1	1	0	x	x	0	1	x
0	1	0	1	0	1	0	0	x	x	0	0	x
0	1	1	0	0	0	0	0	x	x	1	x	1
0	1	1	1	1	0	0	1	x	x	1	x	1
1	0	0	0	0	0	0	x	1	0	x	0	x
1	0	0	1	0	0	1	x	1	0	x	1	x

K-maps and Input/Excitation equations:

$Q_2Q_1$	$Q_0x$		00	01	11	10
	00	0	0	0	0	0
	01	0	0	1	0	0
	11	X	X	X	X	X
	10	X	X	X	X	X
	$J_{Q_2} = Q_1Q_0x$					
	00	X	X	X	X	X
	01	X	X	X	X	X
	11	X	X	X	X	X
	10	1	1	X	X	X
	$K_{Q_2} = 1$					
	00	0	0	1	0	0
	01	X	X	X	X	X
	11	X	X	X	X	X
	10	0	0	X	X	X
	$J_{Q_1} = Q_0x$					
	00	X	X	X	X	X
	01	0	0	1	1	1
	11	X	X	X	X	X
	10	X	X	X	X	X
	$K_{Q_1} = Q_0$					
	00	0	1	X	X	X
	01	1	0	X	X	X
	11	X	X	X	X	X
	10	0	1	X	X	X
	$J_{Q_0} = Q_1x' + Q_1'x = Q_1 \oplus x$					
	00	X	X	1	1	1
	01	X	X	1	1	1
	11	X	X	X	X	X
	10	X	X	X	X	X
	$K_{Q_1} = 1$					

4. If we use 3 flip-flops:

$clk$	$Q_2$	$Q_1$	$Q_0$	$Z$
↑	1	0	1	0
↑	0	1	0	1
↑	1	0	1	1
↑	1	1	0	1
↑	1	1	1	1
↑	1	1	1	0
↑	0	1	1	1

State 101 and 111 will occur twice, thus we need 4 flip-flops to generate 1011110.

State table:

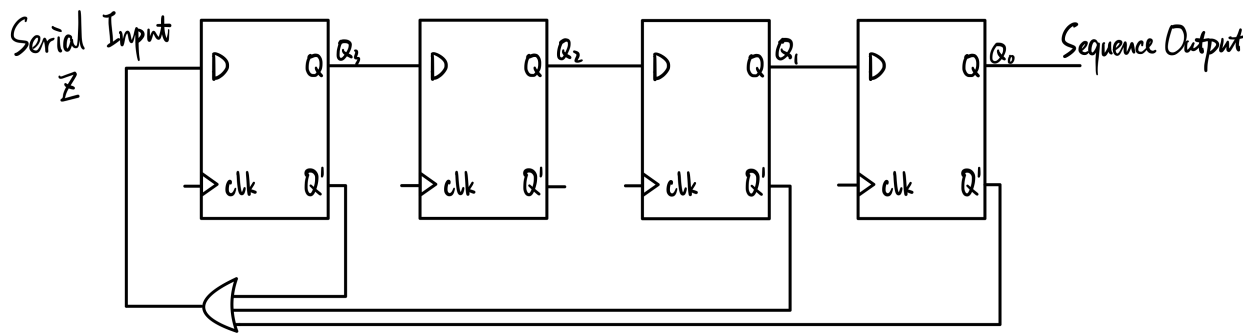
$clk$	$Q_3$	$Q_2$	$Q_1$	$Q_0$	$Z$
↑	1	0	1	1	0
↑	0	1	0	1	1
↑	1	0	1	0	1
↑	1	1	0	1	1
↑	1	1	1	0	1
↑	1	1	1	1	0
↑	0	1	1	1	1

K-map of  $Z$ :

		$Q_1Q_0$			
		00	01	11	10
$Q_3Q_2$	00	X	X	X	X
	01	X	1	1	X
	11	X	1	0	1
	10	X	X	0	1

$$\therefore Z = Q_3' + Q_1' + Q_0'$$

Logic Implementation:



5. State table:

Present State			Next State			Flip-Flop	Inputs		
A	B	C	A	B	C	$T_A$	$T_B$	$T_C$	
0	0	0	0	0	1	0	0	1	
0	0	1	0	1	1	0	1	0	
0	1	0	x	x	x	x	x	x	
0	1	1	1	1	1	1	0	0	
1	0	0	0	0	0	1	0	0	
1	0	1	x	x	x	x	x	x	
1	1	0	1	0	0	0	1	0	
1	1	1	1	1	0	0	0	1	

K-maps:

		$C$	
		0	1
$AB$	00	0	0
	01	X	1
	11	0	0
	10	1	X

		$C$	
		0	1
$AB$	00	0	1
	01	X	0
	11	1	0
	10	0	X

		$C$	
		0	1
$AB$	00	1	0
	01	X	0
	11	0	1
	10	0	X

$$T_A = A'B + AB' = A \oplus B$$

$$T_B = BC' + B'C = B \oplus C$$

$$T_C = A'C' + AC = A \odot C$$

According to the input equation, the next state of state 010 and 101 are 101 and 010 separately.

Present State			Next State			Flip-Flop	Inputs		
A	B	C	A	B	C	$T_A$	$T_B$	$T_C$	
0	1	0	1	0	1	1	1	1	
1	0	1	0	1	0	1	1	1	

If the state reaches 010 or 101 unexpectedly, then the state will switch between 101 and 010 repeatedly.

It's a Lock Out Problem.

To solve this problem, we can let the next state of state 010 and 101 return to 000 and 110 separately.

The corrected state table:

Present State			Next State			Flip-Flop	Inputs		
<i>A</i>	<i>B</i>	<i>C</i>	<i>A</i>	<i>B</i>	<i>C</i>	$T_A$	$T_B$	$T_C$	
0	0	0	0	0	1	0	0	1	
0	0	1	0	1	1	0	1	0	
0	1	0	0	0	0	0	1	0	
0	1	1	1	1	1	1	0	0	
1	0	0	0	0	0	1	0	0	
1	0	1	1	1	0	0	1	1	
1	1	0	1	0	0	0	1	0	
1	1	1	1	1	0	0	0	1	

The corrected k-maps:

		<i>C</i>	
		0	1
<i>AB</i>	00	0	0
	01	0	1
	11	0	0
	10	1	0

		<i>C</i>	
		0	1
<i>AB</i>	00	0	1
	01	1	0
	11	1	0
	10	0	1

		<i>C</i>	
		0	1
<i>AB</i>	00	1	0
	01	0	0
	11	0	1
	10	0	1

$$T_A = A'BC + AB'C'$$

$$T_B = BC' + B'C = B \oplus C$$

$$T_C = A'B'C' + AC$$

Logic Implementation:

