

CSE 140: Components and Design Techniques for Digital Systems

Lecture 9: Sequential Networks: Implementation

CK Cheng

Dept. of Computer Science and Engineering

University of California, San Diego

Implementation

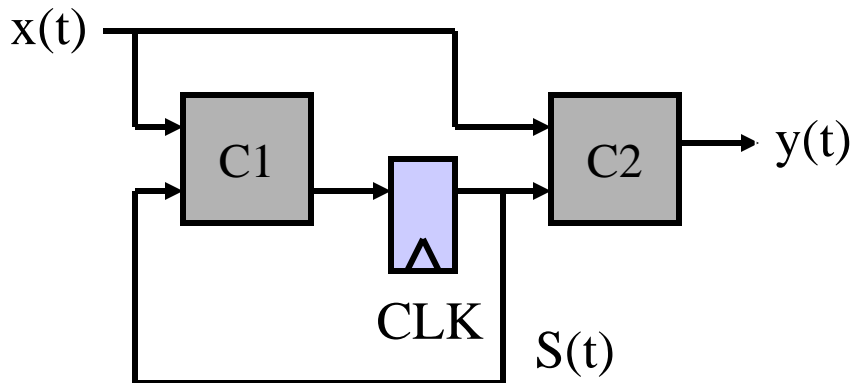
- Format and Tool
 - Mealy & Moore Machines, Excitation Table
- Procedure
 - State Table to Logic Diagram
- Excitation Tables
 - FFs
- Examples

Canonical Form: Mealy and Moore Machines

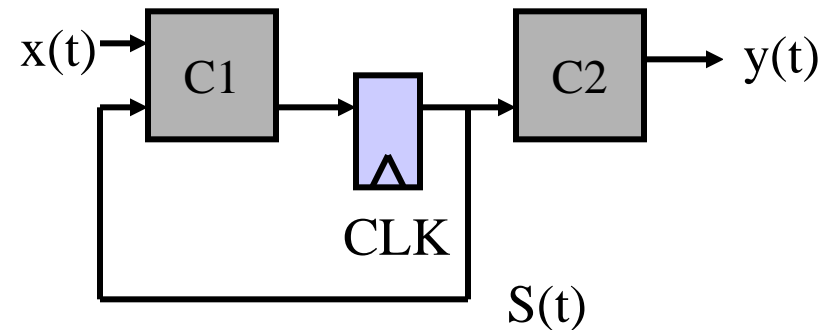
Mealy Machine: $y_i(t) = f_i(X(t), S(t))$

Moore Machine: $y_i(t) = f_i(S(t))$

$$s_i(t+1) = g_i(X(t), S(t))$$



Mealy Machine



Moore Machine

iClicker

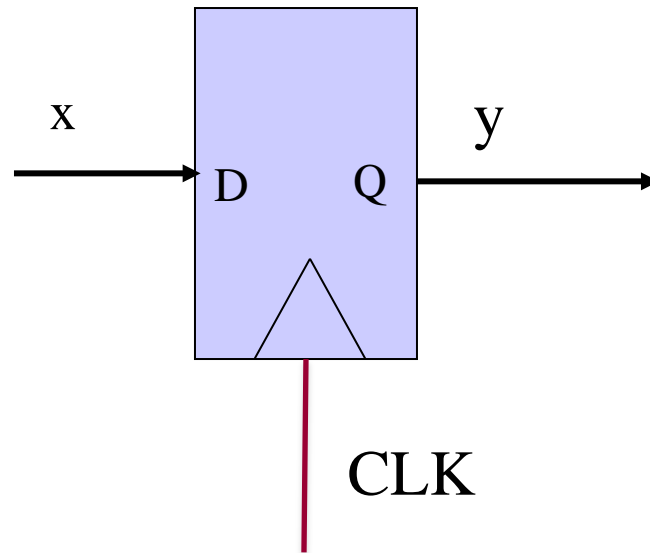
In the logic diagram below, a D flip-flop has input x and output y .

A: $x=Q(t)$, $y=Q(t)$

B: $x=Q(t+1)$, $y=Q(t)$

C: $x=Q(t)$, $y=Q(t+1)$

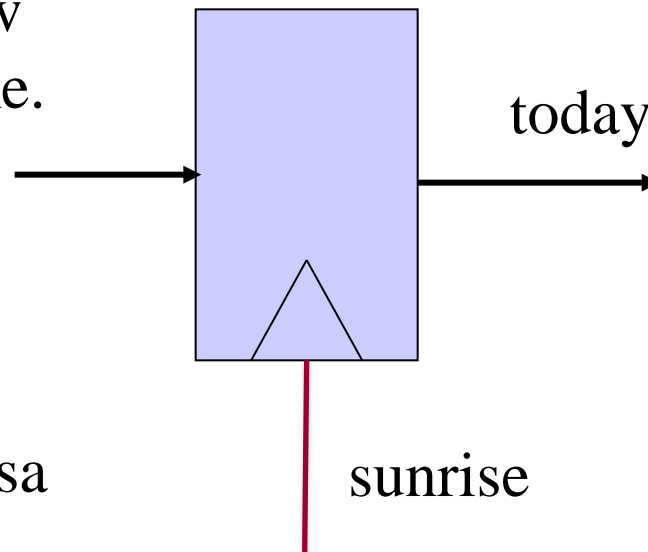
D: None of the above



Understanding Current State and Next State in a sequential circuit

“Yesterday is gone. Tomorrow has not yet come. We have only today. Let us begin.”

— Mother Teresa

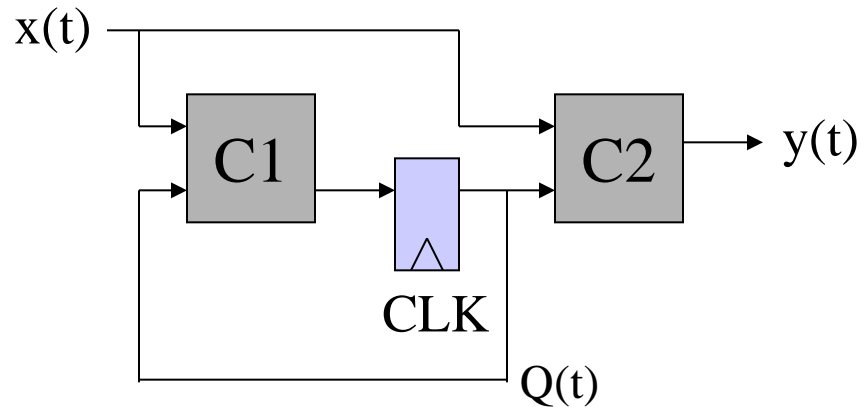


Implementation Format

Canonical Form: Mealy & Moore machines

State Table \rightarrow Netlist

Tool: Excitation Table



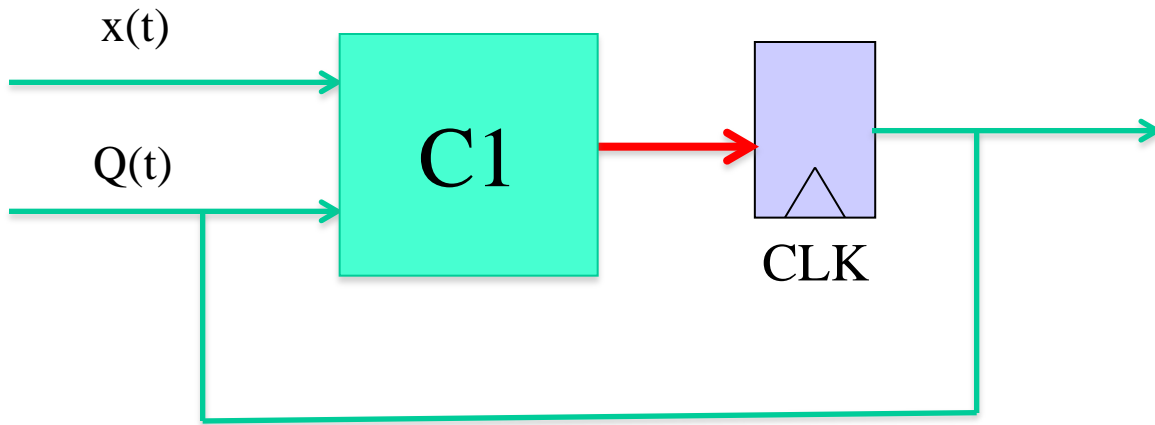
$$Q(t+1) = g(x(t), Q(t)) \quad \text{Circuit C1}$$

$$y(t) = f(x(t), Q(t)) \quad \text{Circuit C2}$$

Implementation Tool: Excitation Table

Example: State Table

id	$x(t)$	$Q(t)$	$Q(t+1)$
0	0	0	1
1	1	1	0
2	0	0	1
3	1	1	0



Find D, T, (S R), (J K) to drive F-Fs

Implementation Tool: Excitation Table

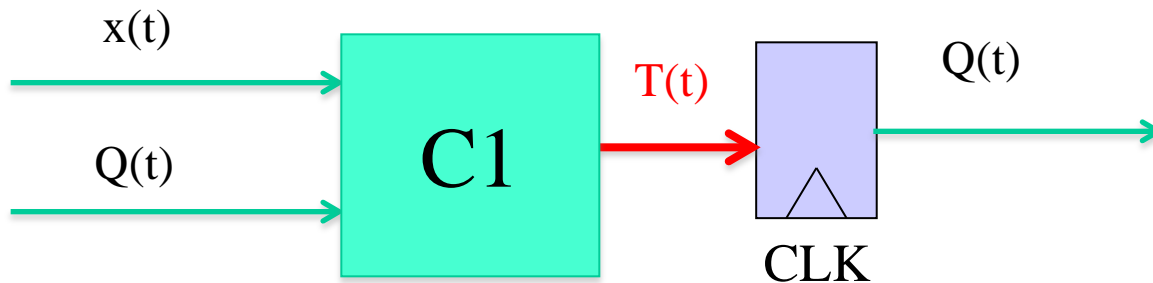
Example:

State Table

id	$x(t)$	$Q(t)$	$Q(t+1)$
0	0	0	1
1	1	1	0
2	0	1	1
3	1	1	0

Excitation Table

id	$x(t)$	$Q(t)$	$T(t)$	$Q(t+1)$
0	0	0	1	1
1	1	1	1	0
2	0	1	0	1
3	1	1	1	0

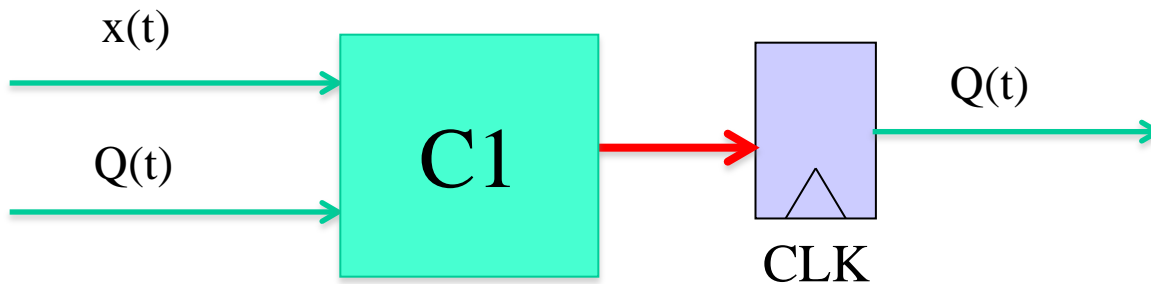


Example with T flip flop

Implementation Tool: Excitation Table

Implement combinational logic **C1**

$D(t)$, $T(t)$, $(S(t) R(t))$, $(J(t) K(t))$ are functions of $(x, Q(t))$



Example:

Excitation Table

id	$x(t)$	$Q(t)$	$T(t)$	$Q(t+1)$
0	0	0	1	1
1	1	1	1	0
2	0	1	0	1
3	1	1	1	0

Implementation: Procedure

State Table \Rightarrow Excitation Table

Problem: To implement C1, we need

$D(t), T(t), (S(t) R(t)), (J(t) K(t))$ as functions of $(x, Q(t))$.

1. From state table, we have

$$NS: Q(t+1) = g(x(t), Q(t))$$

2. Excitation Table of F-Fs: The setting of

$D(t), T(t), (S(t) R(t)), (J(t) K(t))$ to drive $Q(t)$ to $Q(t+1)$.

3. Combining 1 and 2, we have **excitation table of C1**:

$$D(t), T(t), (S(t) R(t)), (J(t) K(t)) = h(x, Q(t)).$$

Implementation: Procedure

F-F State Table \Leftrightarrow F-F Excitation Table

	DTSRJK
PS Q(t)	NS Q(t+1)
	NS Q(t+1)
PS Q(t)	DTSRJK

- D F-F
 - D**(t) = $e_D(Q(t+1), Q(t))$
- T F-F
 - T**(t) = $e_T(Q(t+1), Q(t))$
- SR F-F
 - S**(t) = $e_S(Q(t+1), Q(t))$
 - R**(t) = $e_R(Q(t+1), Q(t))$
- JK F-F
 - J**(t) = $e_J(Q(t+1), Q(t))$
 - K**(t) = $e_K(Q(t+1), Q(t))$

Excitation Table

State table of JK F-F:

		JK				
		00	01	10	11	
0	Q(t)	0	0	1	1	Q(t+1)
1	Q(t)	1	0	1	0	

Excitation table of JK F-F:

		NS	Q(t+1)	
		0	1	
0	PS	0	1	
0	Q(t)	0	1	
1	Q(t)	-1	-0	JK

Ex: If Q(t) is 1, and Q(t+1) is 0, then JK needs to be -1.

Excitation Tables and State Tables

Excitation Tables:

JK	PS	NS		Q(t+1)	
		0	1		
	0	0-	1-		
	Q(t) 1	-1	-0		

State Tables:

JK	Q(t)	Q(t+1)			
		00	01	10	11
	0	0	0	1	1
	1	1	0	1	0

SR	PS	NS		Q(t+1)	
		0	1		
	0	0-	10		
	Q(t) 1	XY	-0		

SR

SR	PS	Q(t+1)			
		00	01	10	11
	0	0	0	1	-
	Q(t) 1	1	0	1	-

iClicker

A. XY = -1

B. XY = 01

C. XY = 10

D. XY = 1-

E. None of the above

Excitation Tables and State Tables

Excitation Tables:

T

		Q(t+1)	
		0	1
PS	NS		
		0	1
Q(t)	0	0	1
	1	1	0

State Tables:

T

		Q(t+1)	
		0	1
PS	T		
		0	1
Q(t)	0	0	1
	1	1	0

D

		Q(t+1)	
		0	1
PS	NS		
		0	1
Q(t)	0	0	1
	1	0	1

D

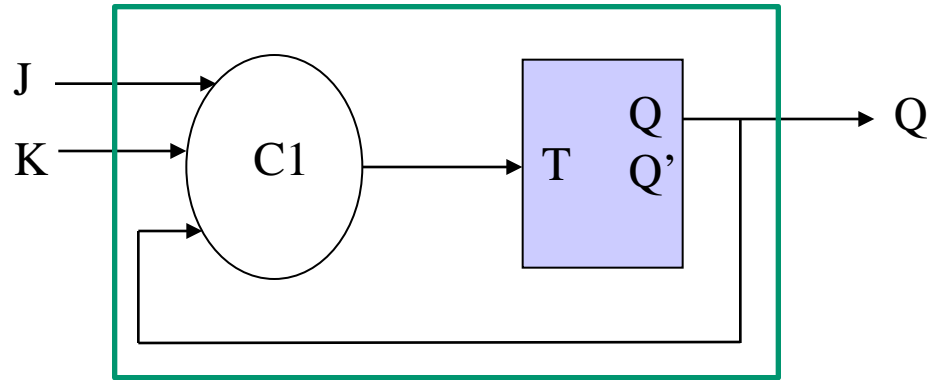
		Q(t+1)	
		0	1
PS	D		
		0	1
Q(t)	0	0	1
	1	0	1

Implementation: Procedure

1. State table: $y(t) = f(Q(t), x(t))$, $Q(t+1) = g(x(t), Q(t))$
2. Excitation table of F-Fs:
 - $D(t) = e_D(Q(t+1), Q(t));$
 - $T(t) = e_T(Q(t+1), Q(t));$
 - (S, R), or (J, K)
3. From 1 & 2, we derive excitation table of the system
 - $D(t) = h_D(x(t), Q(t)) = e_D(g(x(t), Q(t)), Q(t));$
 - $T(t) = h_T(x(t), Q(t)) = e_T(g(x(t), Q(t)), Q(t));$
 - (S, R) or (J, K).
4. Use K-map to derive combinational logic implementation.
 - $D(t) = h_D(x(t), Q(t))$
 - $T(t) = h_T(x(t), Q(t))$
 - $y(t) = f(x(t), Q(t))$

Implementation: Example

Implement a JK F-F with a T F-F



Implement a JK F-F: $Q(t+1) = h(J(t), K(t), Q(t)) = J(t)Q'(t) + K'(t)Q(t)$

JK		JK			
		00	01	10	11
PS	0	0	0	1	1
	1	1	0	1	0

Example: Implement a JK flip-flop using a T flip-flop

Excitation Table of T Flip-Flop

		Q(t+1)	
		0	1
PS	NS		
	0	0	1
Q(t)	1	1	0

Excitation Table of the Design

id	J(t)	K(t)	Q(t)	Q(t+1)	T(t)
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	0	0
3	0	1	1	0	1
4	1	0	0	1	1
5	1	0	1	1	0
6	1	1	0	1	1
7	1	1	1	0	1

$$T(t) = Q(t) \text{ XOR } (J(t)Q'(t) + K'(t)Q(t))$$

$$\text{i.e. } Q(t+1)(t) = JQ'(t) + K'Q(t)$$

- State table: $y(t) = f(Q(t), x(t))$,
 $Q(t+1) = g(x(t), Q(t))$
- Excitation table of F-Fs:
 - $D(t) = e_D(Q(t+1), Q(t))$;
 - $T(t) = e_T(Q(t+1), Q(t))$;
 - (S, R), or (J, K)
- From 1 & 2, we derive excitation table of the system
 - $D(t) = h_D(x(t), Q(t)) = e_D(g(x(t), Q(t)), Q(t))$;
 - $T(t) = h_T(x(t), Q(t)) = e_T(g(x(t), Q(t)), Q(t))$;
 - (S, R) or (J, K).
- Use K-map to derive combinational logic implementation.
 - $D(t) = h_D(x(t), Q(t))$
 - $T(t) = h_T(x(t), Q(t))$
 - $y(t) = f(x(t), Q(t))$

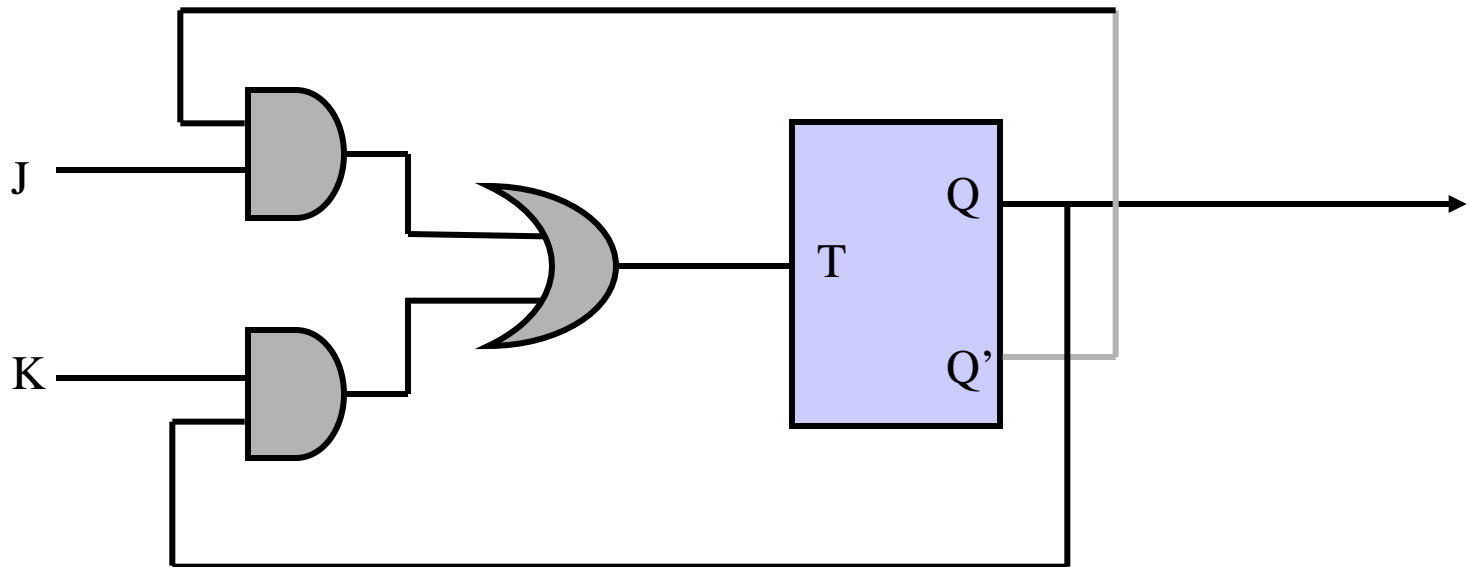
Example: Implement a JK flip-flop using a T flip-flop

T(J,K,Q):

	K			
Q(t)	0	2	6	4
	0	0	1	1
	1	3	7	5
	0	1	1	0

J

$$T = K(t)Q(t) + J(t)Q'(t)$$

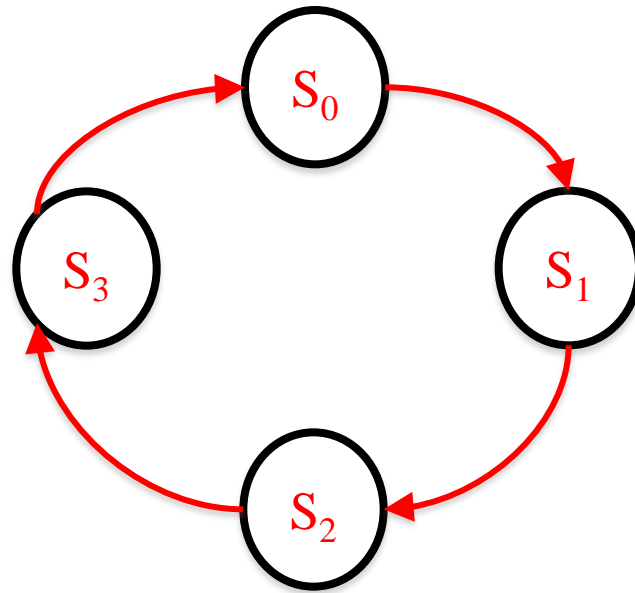


iClicker

Before state assignment, the relation of its state table and excitation table is

- A. One to one
- B. One to many
- C. Many to one
- D. Many to many
- E. None of the above

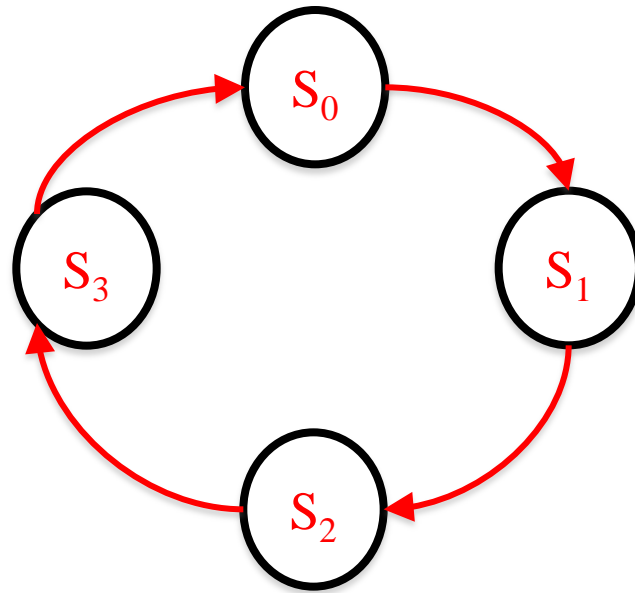
Let's implement our free running 2-bit counter using T-flip flops



State Table

PS	Next state
S_0	S_1
S_1	S_2
S_2	S_3
S_3	S_0

Let's implement our free running 2-bit counter using T-flip flops



State Table

S_0	S_1
S_1	S_2
S_2	S_3
S_3	S_0

State Table with Assigned Encoding

Current	Next
0 0	01
0 1	10
1 0	11
1 1	00

Let's implement our free running 2-bit counter using T-flip flops

Excitation table

id	$Q_1(t)$	$Q_0(t)$	$T_1(t)$	$T_0(t)$	$Q_1(t+1)$	$Q_0(t+1)$
0	0	0			0	1
1	0	1			1	0
2	1	0			1	1
3	1	1			0	0

Let's implement our free running 2-bit counter using T-flip flops

Excitation table

id	$Q_1(t)$	$Q_0(t)$	$T_1(t)$	$T_0(t)$	$Q_1(t+1)$	$Q_0(t+1)$
0	0	0	0	1	0	1
1	0	1	1	1	1	0
2	1	0	0	1	1	1
3	1	1	1	1	0	0

Let's implement our free running 2-bit counter using T-flip flops

Excitation table

id	$Q_1(t)$	$Q_0(t)$	$T_1(t)$	$T_0(t)$	$Q_1(t+1)$	$Q_0(t+1)$
0	0	0	0	1	0	1
1	0	1	1	1	1	0
2	1	0	0	1	1	1
3	1	1	1	1	0	0

$$T_0(t) =$$

$$T_1(t) =$$

$$Q_0(t+1) = T_0(t) Q'_0(t) + T'_0(t) Q_0(t)$$

$$Q_1(t+1) = T_1(t) Q'_1(t) + T'_1(t) Q_1(t)$$

Let's implement our free running 2-bit counter using T-flip flops

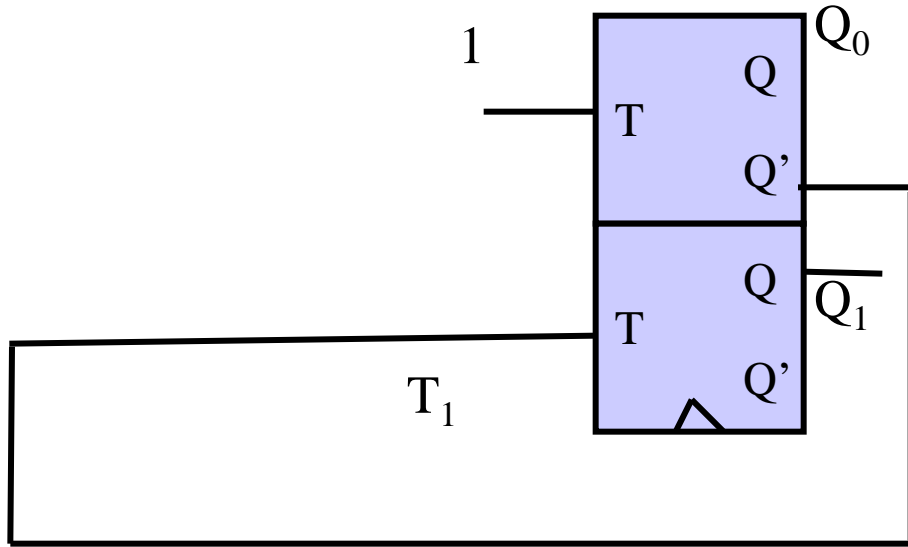
Excitation table

id	$Q_1(t)$	$Q_0(t)$	$T_1(t)$	$T_0(t)$	$Q_1(t+1)$	$Q_0(t+1)$
0	0	0	0	1	0	1
1	0	1	1	1	1	0
2	1	0	0	1	1	1
3	1	1	1	1	0	0

$$T_0(t) = 1$$

$$T_1(t) = Q_0(t)$$

Free running counter with T flip flops



$$T_0(t) = 1$$

$$T_1(t) = Q_0(t)$$

Summary: Implementation

- Set up canonical form
 - **Mealy or Moore machine**
- Identify the next states
 - state diagram \Leftrightarrow state table
 - state assignment
- Derive **excitation table**
 - Inputs of flip flops
- Design the combinational logic
 - don't care set utilization