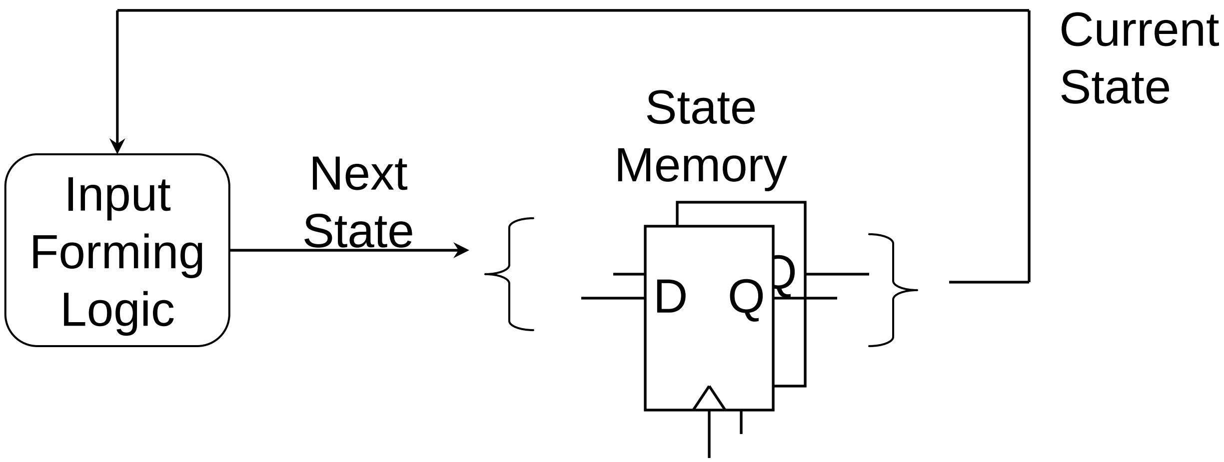
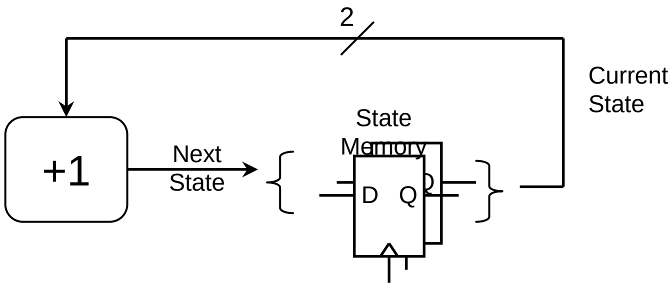
Quick Overview of Counters

Counters
Transition Tables
Moore Outputs
Stategraphs

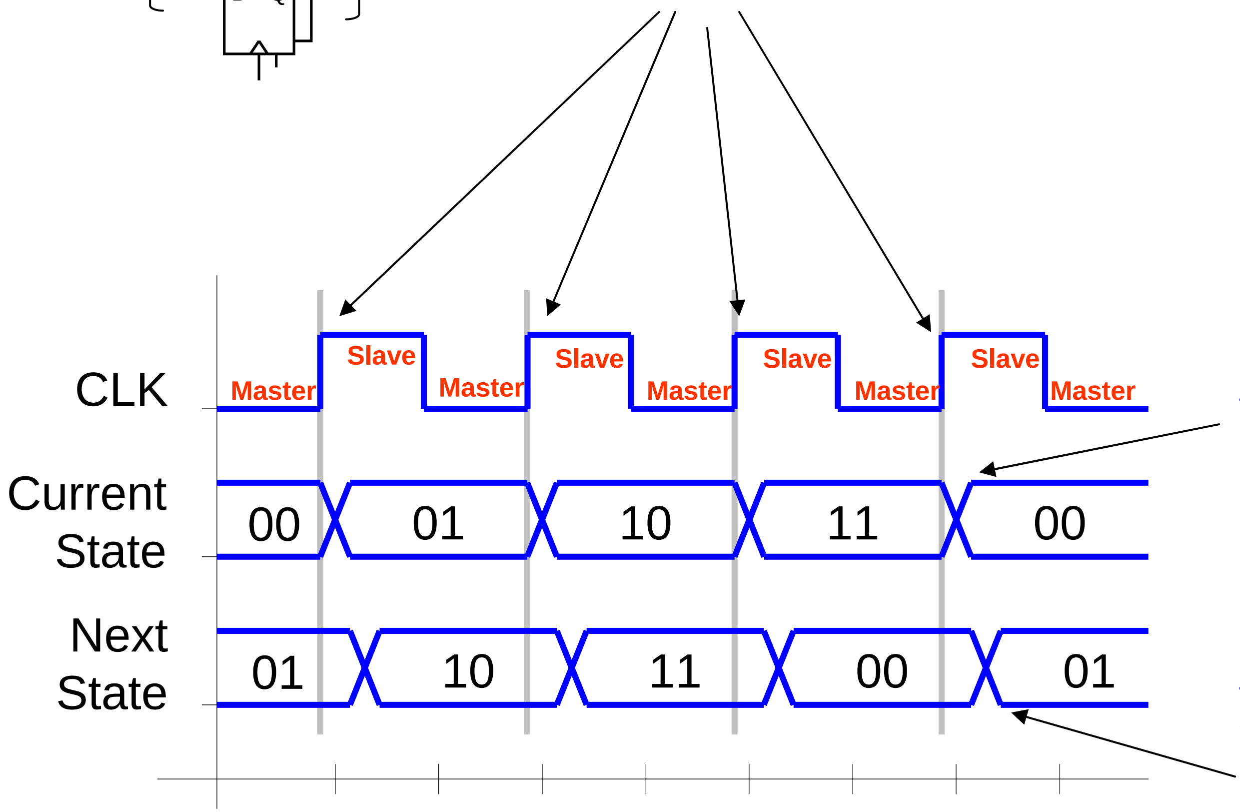
General Sequential Systems



A Sequential Counter



Clock edges…



The current state loads the next state values in response to the clock edge.

IFL reacts after some gate delays to produce a new next state.

Transition Table for 2-Bit Counter

Current State	Next State
00	01
01	10
10	11
11	00

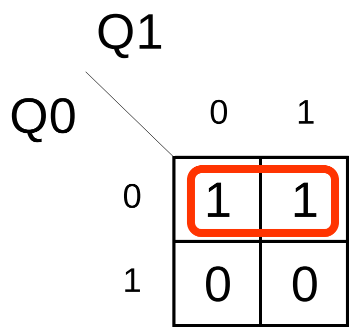
Current State		Next State	
Q1	Q0	N1	N0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

It is the truth table for the input forming logic…

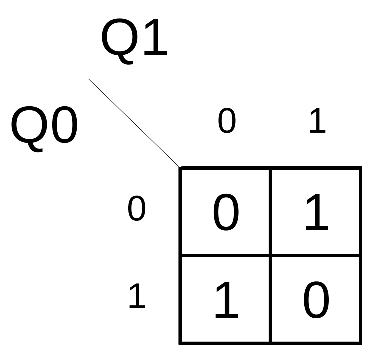
It describes what the *next state* values are as a function of the *current state* (clock is assumed)

Implementation of 2-Bit Counter

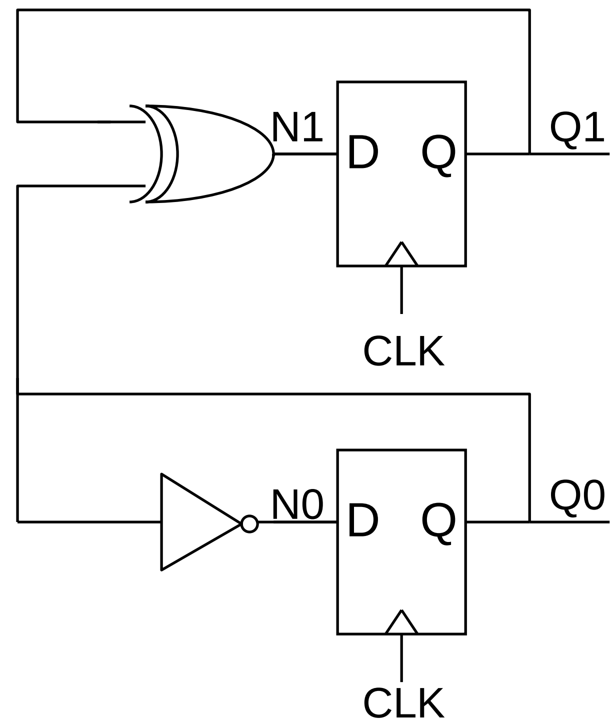
Current State		Next State	
Q1	Q0	N1	N0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0



N0 =Q0’



N1 =Q1 ⊕ Q0



Example 2 - A Gray Code Counter

Q1	Q0	N1	N0
0	0	0	1
0	1	1	1
1	1	1	0
1	0	0	0

Ordering of TT rows chosen to make it easier to see the count pattern.

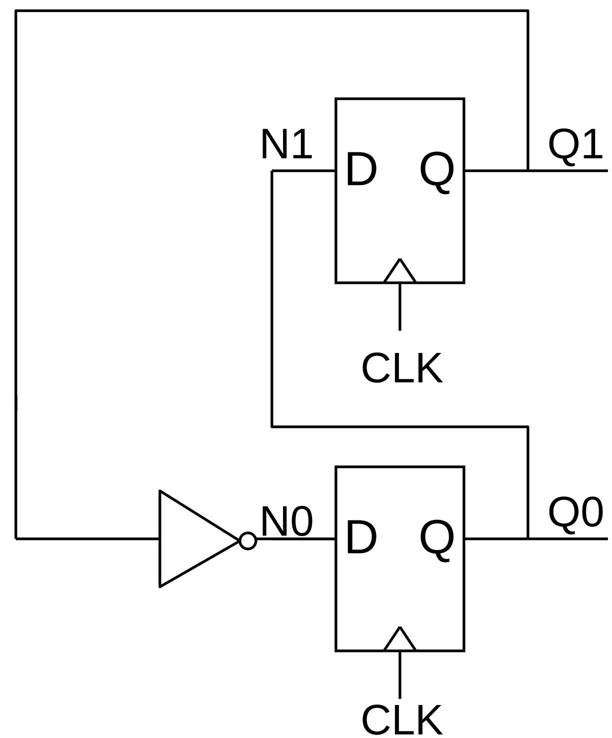
It doesn't change result.

Q0 \ Q1	0	1
0	0	0
1	1	1

N1= Q0

Q0 \ Q1	0	1
0	1	0
1	1	0

N0 = Q1'



Example 3 – Not All Count Values Used

Desired count sequence = 00 – 01 – 11 - 00 …

Q1	Q0	N1	N0
0	0	0	1
0	1	1	1
1	1	0	0
1	0	?	?

What should next state for 10 be?

Example 3 – Not All Count Values Used

Q1	Q0	N1	N0
0	0	0	1
0	1	1	0
1	0	X	X
1	1	0	0

Q0 \ Q1	0	1
0	0	X
1	1	0

$$N1 = Q0 \oplus Q1$$

Q0 \ Q1	0	1
0	1	X
1	0	0

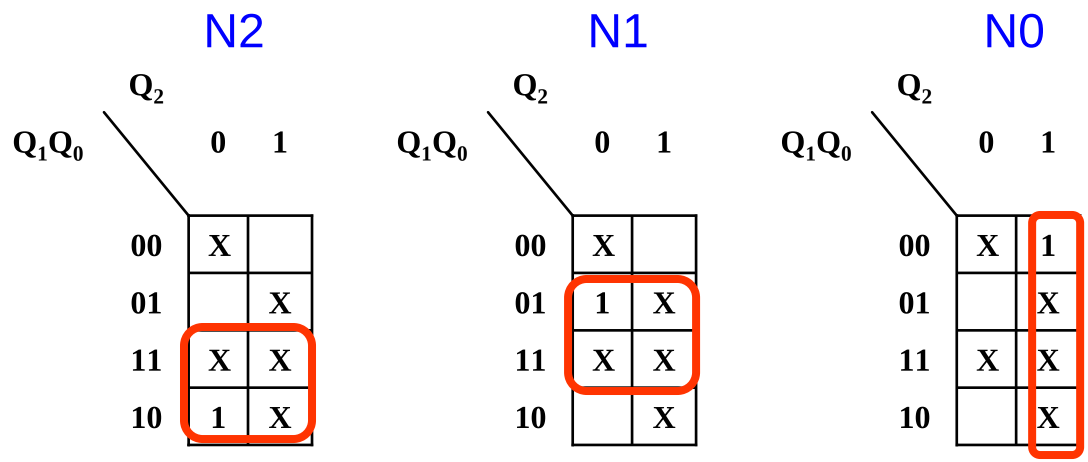
$$N0 = Q0'$$

Do the normal KMap w/don't cares minimization…

Example 4 – A Ring Counter

Desired count sequence = 001 – 010 – 100 - 001 …

Q2	Q1	Q0	N2	N1	N0
0	0	0	X	X	X
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	X	X	X
1	0	0	0	0	1
1	0	1	X	X	x
1	1	0	X	X	X
1	1	1	X	X	X



Doing KMaps leads to:

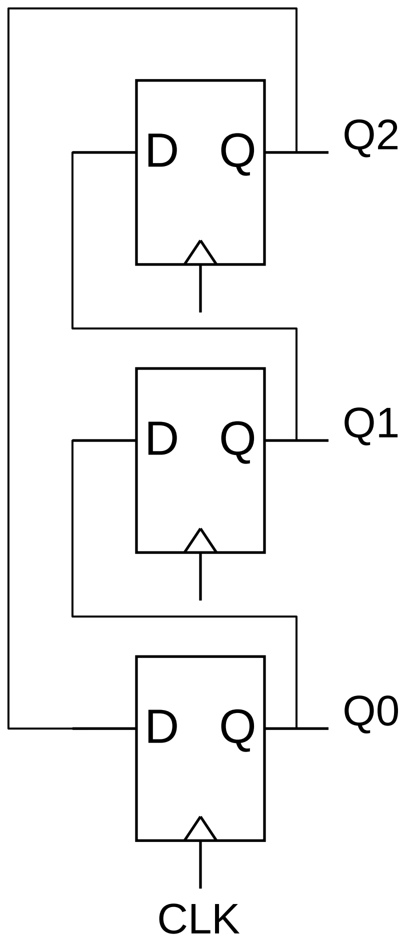
N2 = Q1

N1 = Q0

N0 = Q2

No big surprise here!!!!!

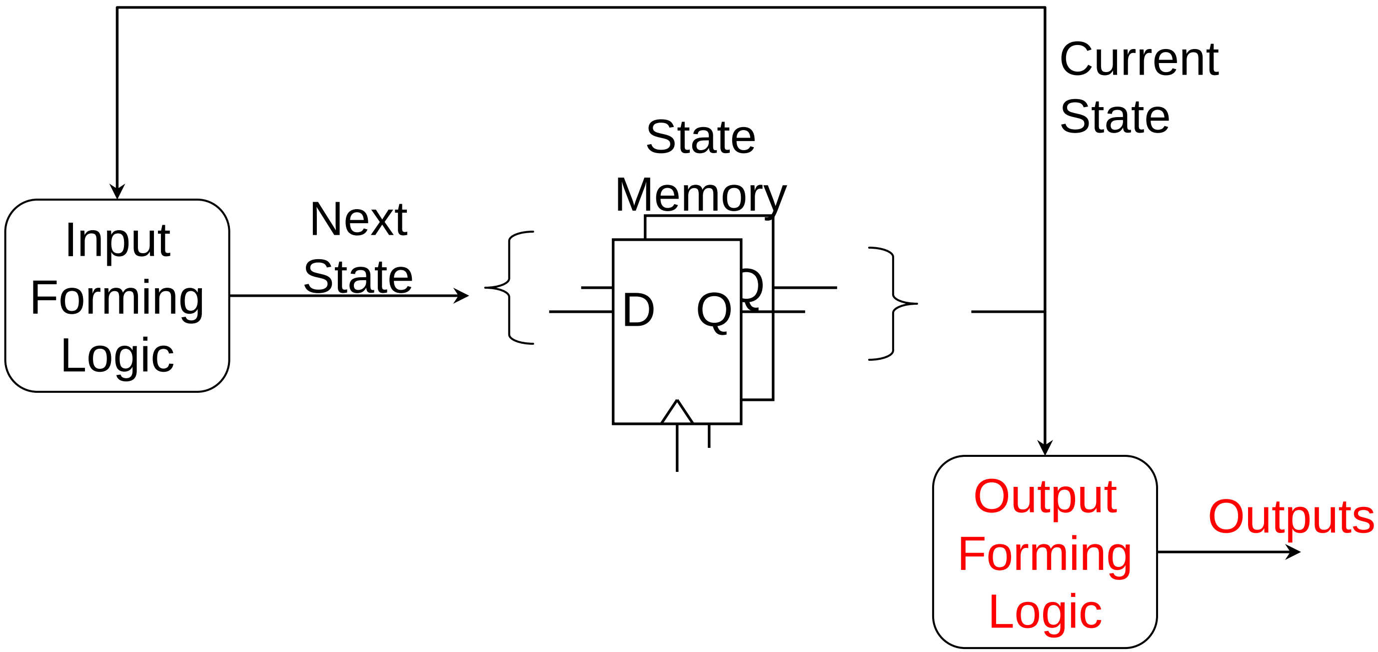
Example 4 – A Ring Counter



General Counter Design Procedure

- Write transition table for counter
  - Use X's as appropriate
- Reduce each Nx variable to an equation
- Implement input forming logic (IFL) using gates
- Draw schematic using FF's + IFL

Counters With Outputs



Outputs = f(CurrentState)

Counters With Outputs

Z=1 when count={0,3,6}

Q2	Q1	Q0	Z
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Z is called a Moore or static output.
It is a function only of the current state

Combined Transition Table

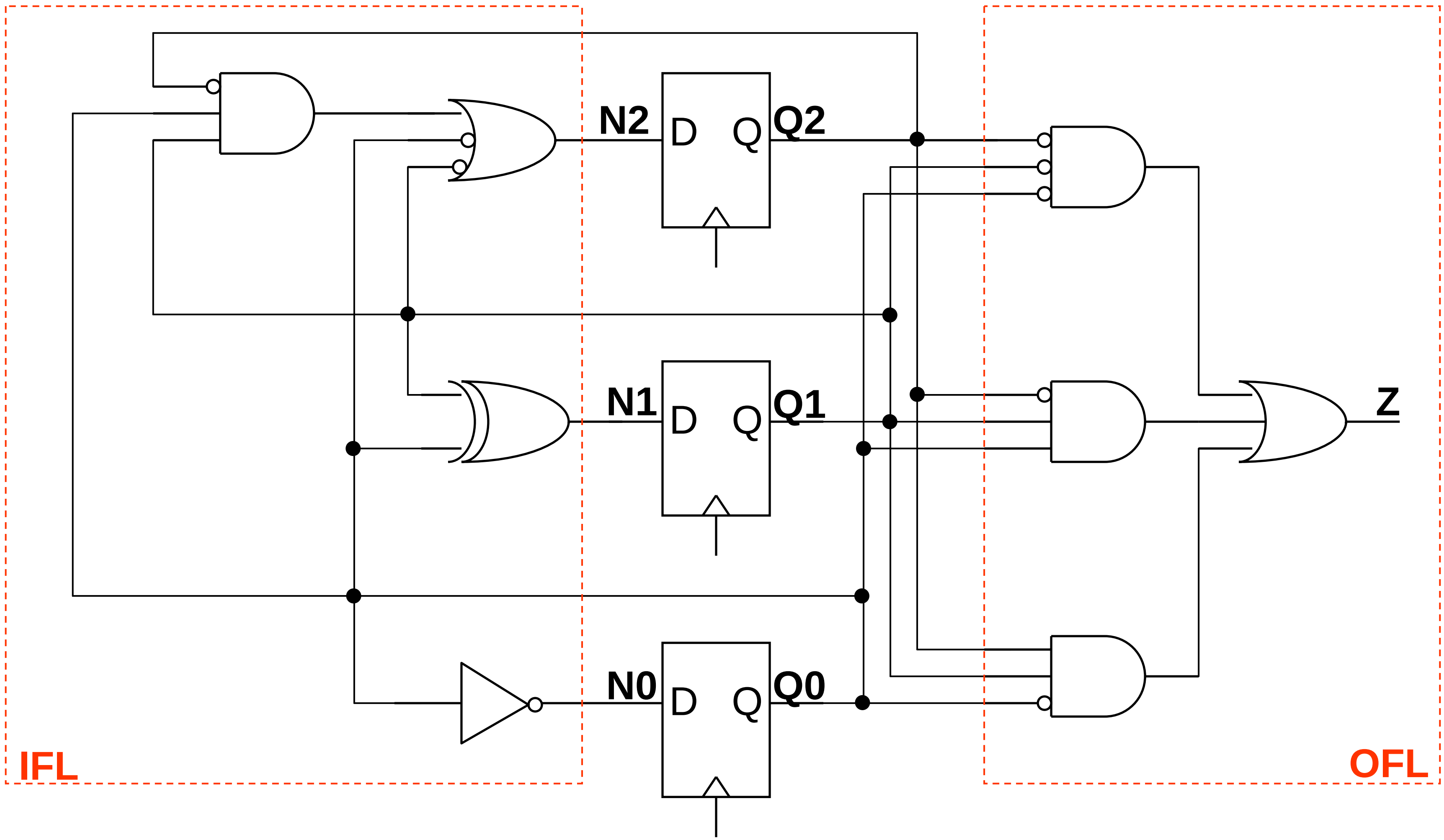
Q2	Q1	Q0	N2	N1	N0	Z
0	0	0	0	0	1	1
0	0	1	0	1	0	0
0	1	0	0	1	1	0
0	1	1	1	0	0	1
1	0	0	1	0	1	0
1	0	1	1	1	0	0
1	1	0	1	1	1	1
1	1	1	0	0	0	0

Current state (Q2 Q1 Q0) — Next state (N2 N1 N0) — Output (Z)

Z = Q2'Q1'Q0' + Q2'Q1Q0 + Q2Q1Q0'

(implement OFL with gates)

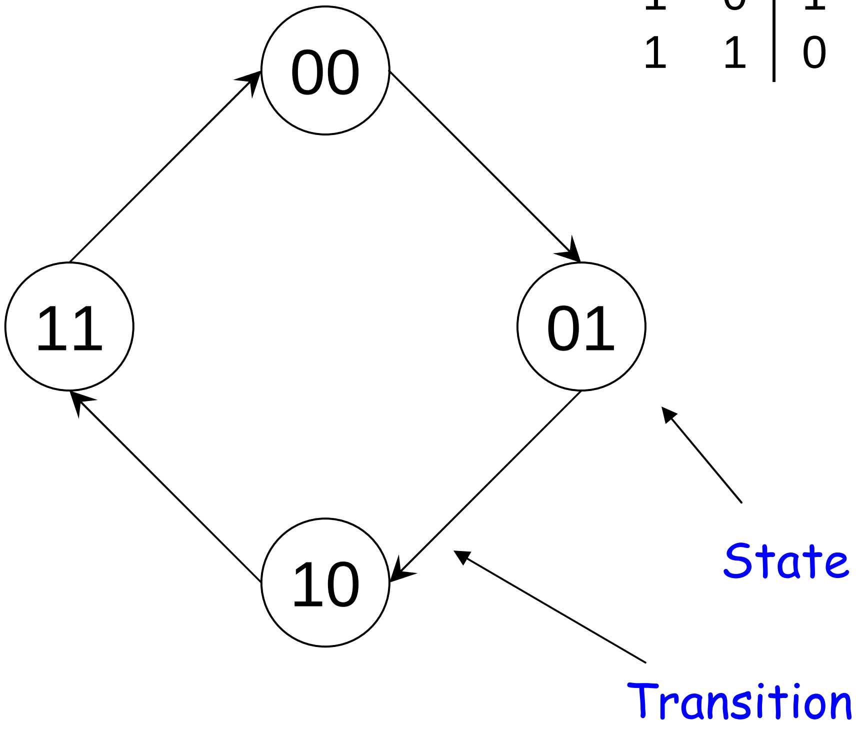
Counter With A Moore Output



State Graphs

Binary Counter State Graph

Q1	Q0	N1	N0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

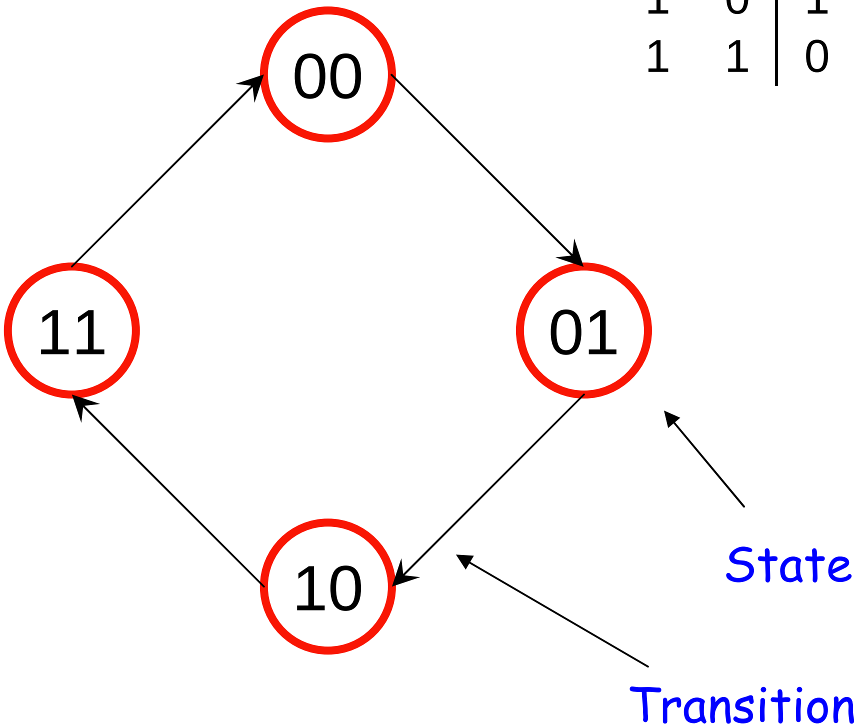


State graphs are graphical representations of TT's

They contain the same information: no more, no less

Binary Counter State Graph

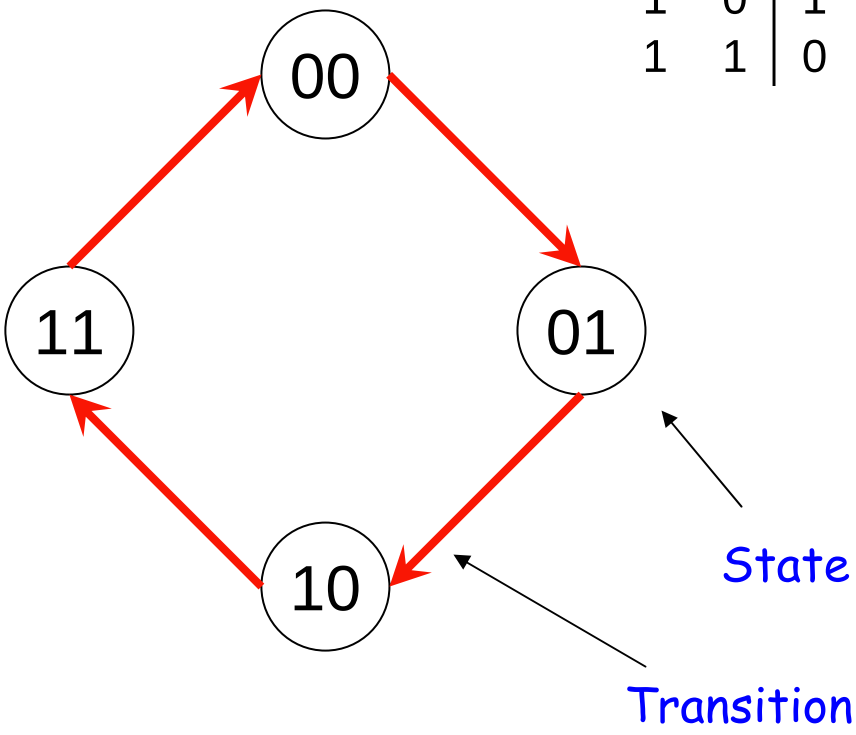
Q1	Q0	N1	N0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0



Each circle is a state

Binary Counter State Graph

Q1	Q0	N1	N0
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

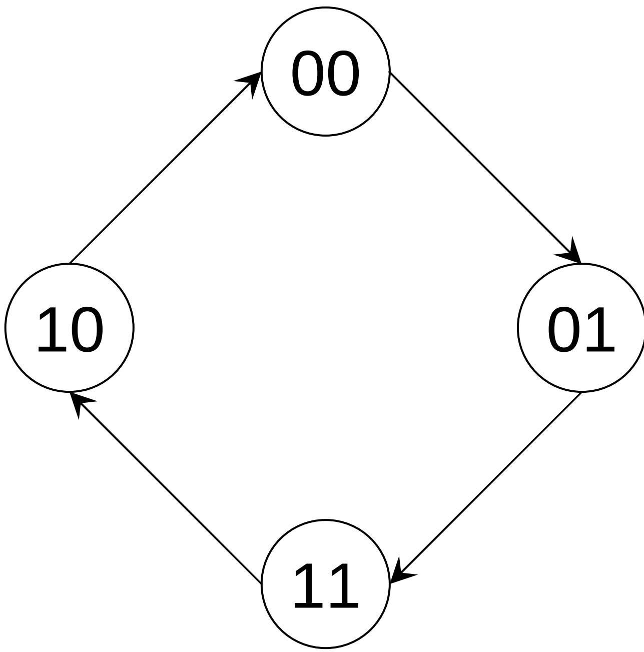


Each arc is a transition from one state to another

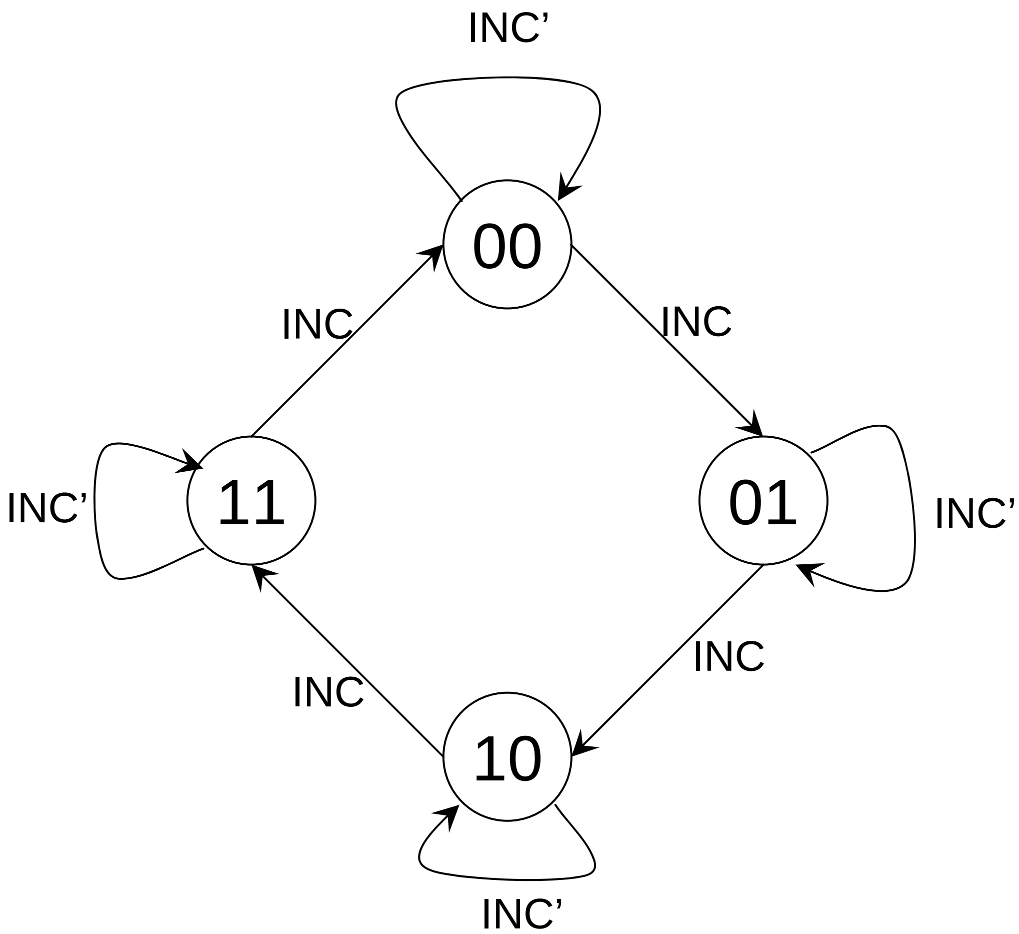
These arcs are unlabelled, meaning the transition is *always* taken on the clock edge

Gray Code Counter State Graph

Q1	Q0	N1	N0
0	0	0	1
0	1	1	1
1	1	1	0
1	0	0	0



State Graphs for Counters With Inputs

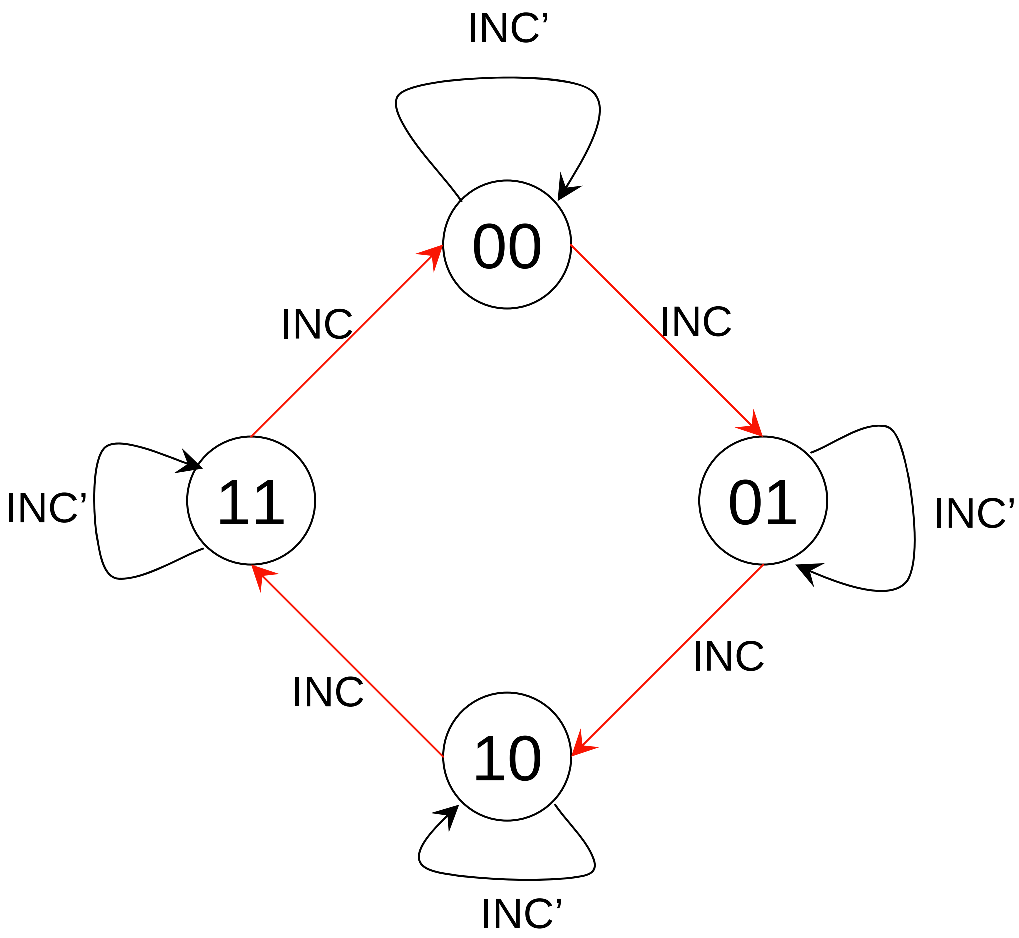


INC controls whether transition is taken or not…

INC	Q1	Q0	N1	N0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

There is a one-to-one correspondence between the rows of the TT and the arcs in the SG

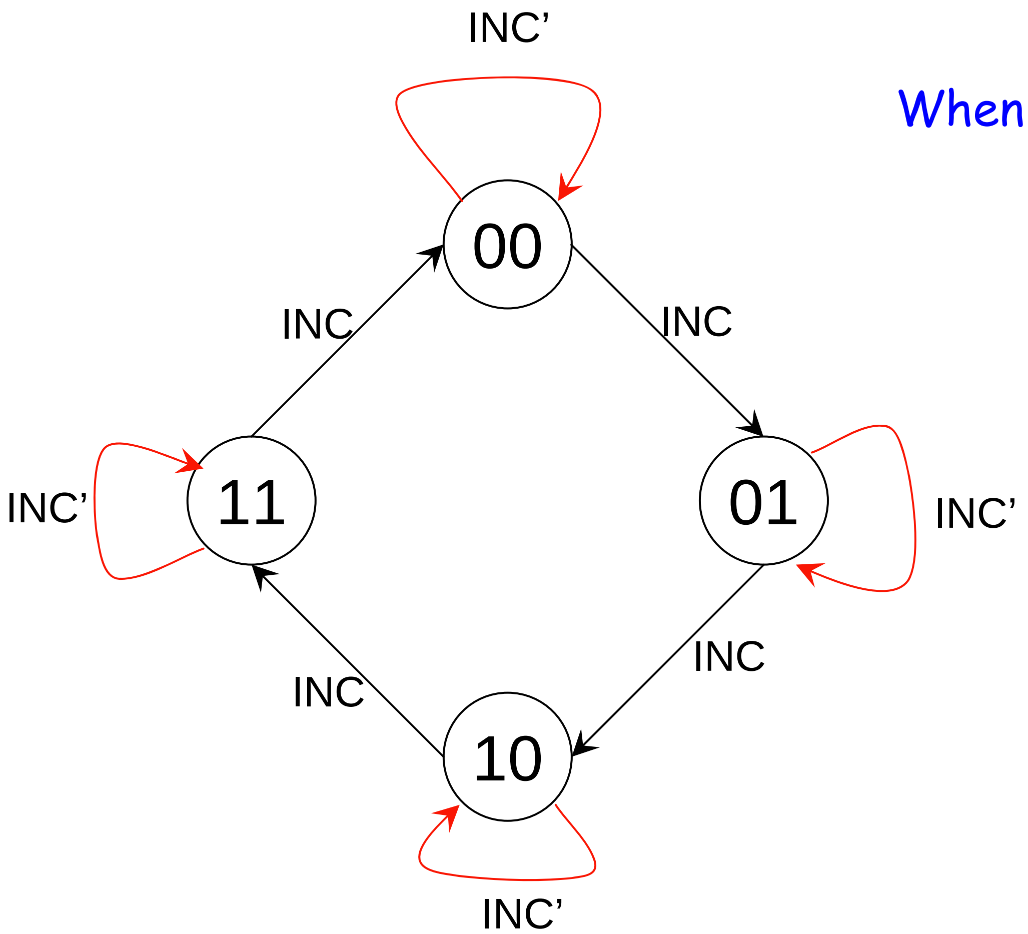
State Graphs for Counters With Inputs



When INC=1, we change state…

INC	Q1	Q0	N1	N0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

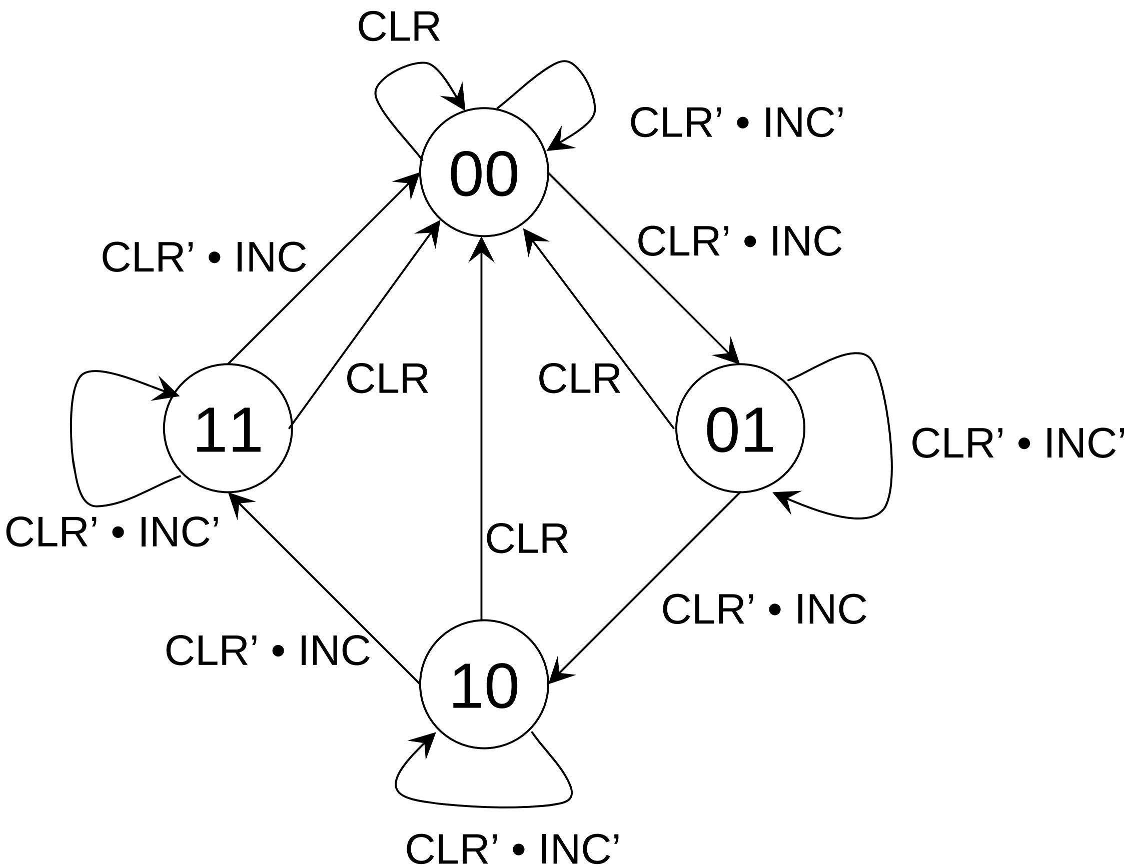
State Graphs for Counters With Inputs



When INC=0, we stay in the same state…

INC	Q1	Q0	N1	N0
0	0	0	0	0
0	0	1	0	1
0	1	0	1	0
0	1	1	1	1
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

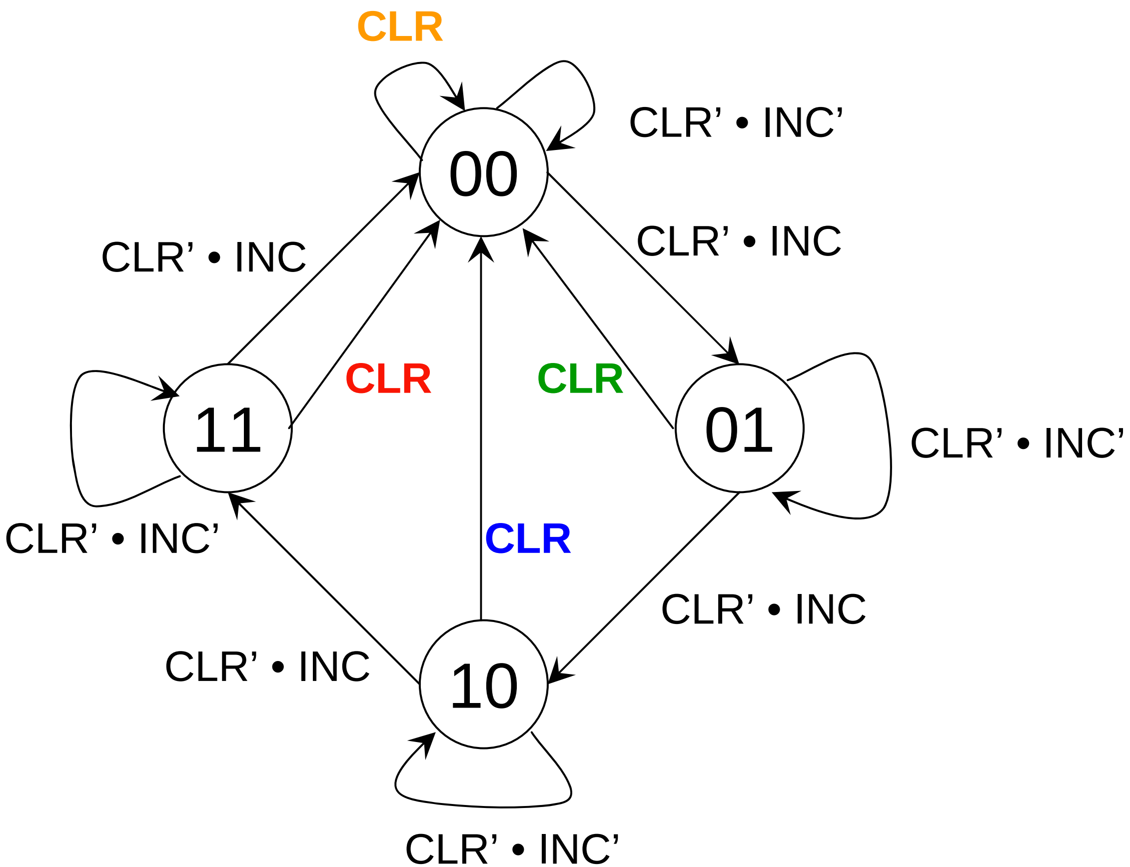
SG for Counter With Multiple Inputs



CLR	INC	Q1	Q0	N1	N0
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	1
0	1	1	1	0	0
1	0	0	0	0	0
1	0	0	1	0	0
1	0	1	0	0	0
1	0	1	1	0	0
1	1	0	0	0	0
1	1	0	1	0	0
1	1	1	0	0	0
1	1	1	1	0	0

NOT an exact correspondence between TT rows and SG arcs, why?

SG for Counter With Multiple Inputs



CLR	INC	Q1	Q0	N1	N0
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	1
0	1	1	1	0	0
1	**0**	**0**	**0**	**0**	**0**
1	**0**	**0**	**1**	**0**	**0**
1	**0**	**1**	**0**	**0**	**0**
1	**0**	**1**	**1**	**0**	**0**
1	**1**	**0**	**0**	**0**	**0**
1	**1**	**0**	**1**	**0**	**0**
1	**1**	**1**	**0**	**0**	**0**
1	**1**	**1**	**1**	**0**	**0**

Some arcs correspond to 2 rows in TT

Transition Table Simplification

CLR	INC	Q1	Q0	N1	N0
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	1
0	1	1	1	0	0
1	**0**	**0**	**0**	**0**	**0**
1	**0**	**0**	**1**	**0**	**0**
1	**0**	**1**	**0**	**0**	**0**
1	**0**	**1**	**1**	**0**	**0**
1	**1**	**0**	**0**	**0**	**0**
1	**1**	**0**	**1**	**0**	**0**
1	**1**	**1**	**0**	**0**	**0**
1	**1**	**1**	**1**	**0**	**0**

→

CLR	INC	Q1	Q0	N1	N0
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	1
0	1	1	1	0	0
1	**-**	**0**	**0**	**0**	**0**
1	**-**	**0**	**1**	**0**	**0**
1	**-**	**1**	**0**	**0**	**0**
1	**-**	**1**	**1**	**0**	**0**

→

CLR	INC	Q1	Q0	N1	N0
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	1
0	1	1	1	0	0
1	**-**	**-**	**-**	**0**	**0**

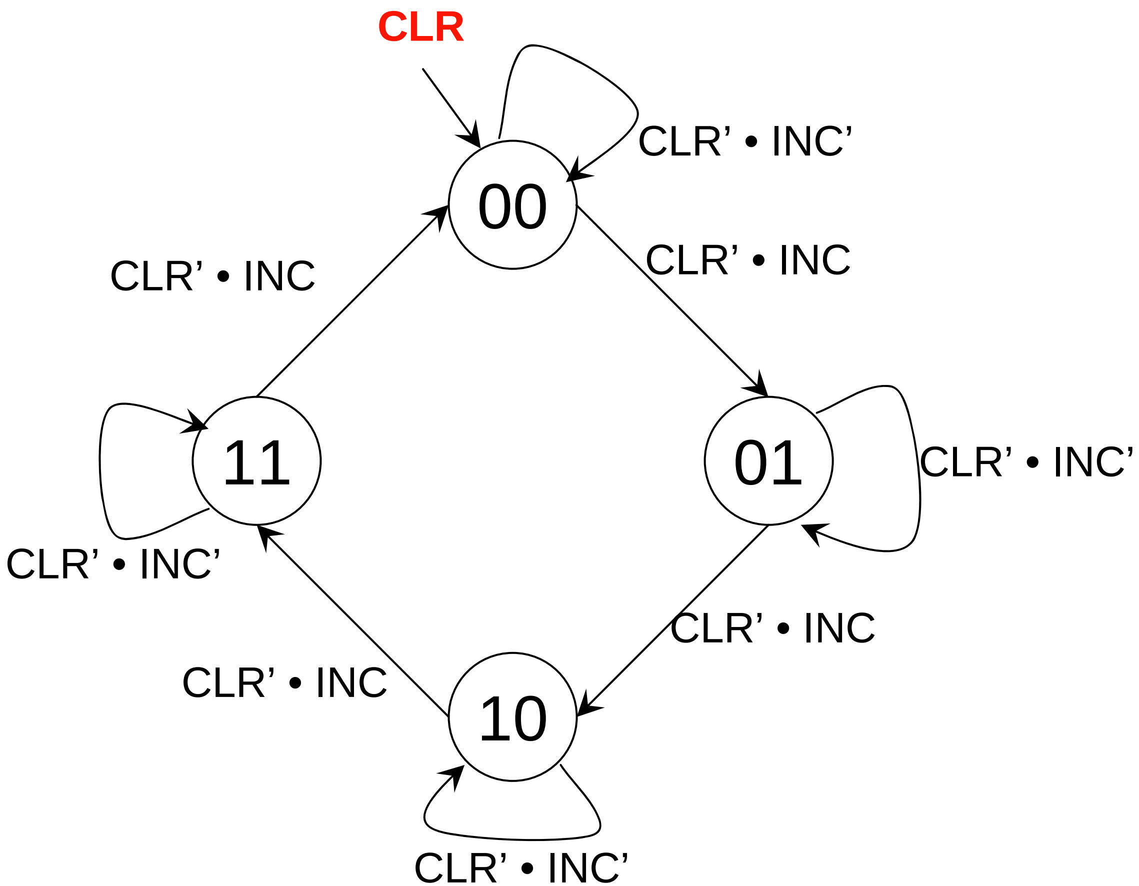
These are ***input*** don't cares.
They are a shorthand for the TT on the left
This TT exactly matches SG on previous page

Simplified Transition Tables With Input Don't Cares

- Contain ***exactly*** same information as original
  - Shorthand way of writing
- Should be able to easily convert back/forth

Simplified State Graph

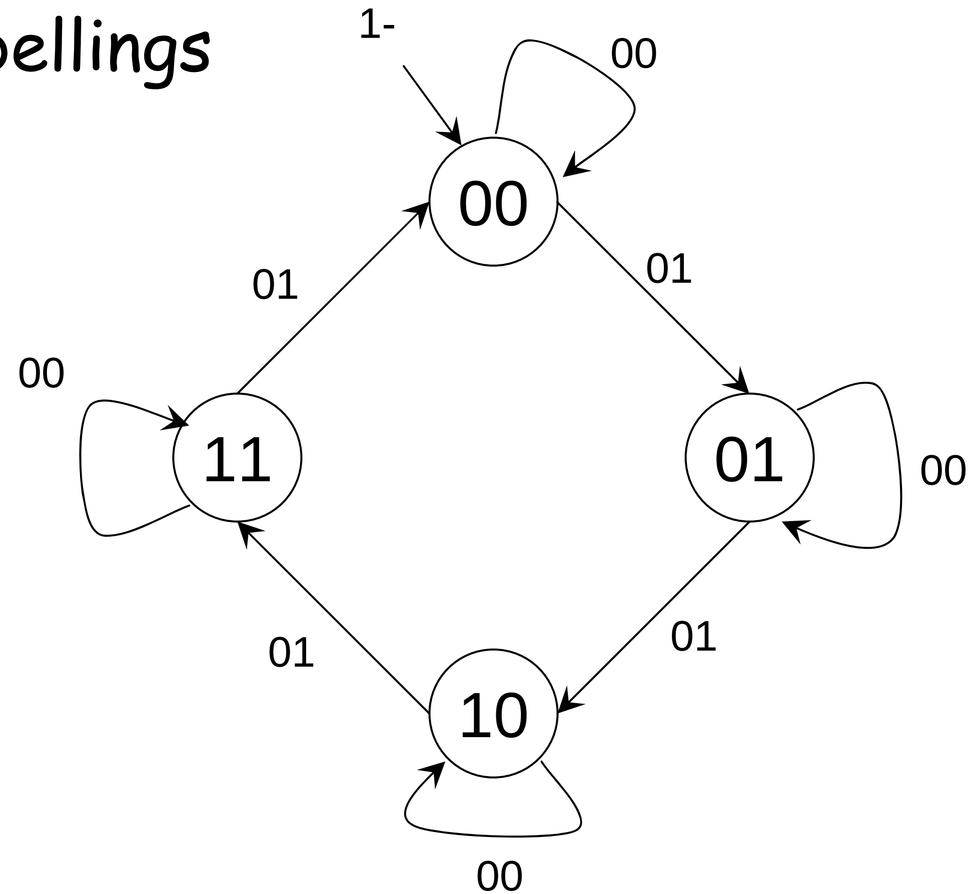
CLR	INC	Q1	Q0	N1	N0
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	1
0	1	1	1	0	0
1	**-**	**-**	**-**	**0**	**0**



There is now a one-to-one correspondence between rows and arcs…

More Simplified State Graphs

- If the order of inputs is known…
  - Use binary labellings



There is no **correct** way to draw a SG as long as you know precisely how it maps back to a transition table.

Design Procedure Using State Graphs

1. Draw the state graph
2. Create an equivalent transition table
3. If transition table contains input don't cares,
   - *unfold* it to a full transition table
4. Complete the design using KMaps, gates, FF's

Cover Examples in Book

- Page 260 – use white board
- Look @ process on 268
- Another Example on 276 – use whiteboard