Overview

- Last lecture
  - Introduction to finite state machines
    - State diagrams
  - Counters as finite state machines
    - Counter design
- Today
  - Finish counter design
    - Our last design example
    - Self-starting counters

Review: Counter design procedure

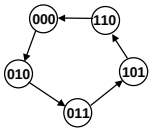
1. Draw a state diagram
2. Draw a state-transition table
3. Encode the next-state functions
   - Minimize the logic using k-maps
4. Implement the design

Example: A 5-state counter

Example: A 5-state counter

- Counter repeats 5 states in sequence
  - Sequence is 000, 010, 011, 101, 110, 000 (not binary)

Step 1: State diagram



Step 2: State transition table

Assume D flip-flops

Present State			Next State		
C	B	A	C+	B+	A+
0	0	0	0	1	0
0	0	1	–	–	–
0	1	0	0	1	1
0	1	1	1	0	1
1	0	0	–	–	–
1	0	1	1	1	0
1	1	0	0	0	0
1	1	1	–	–	–

5-state counter (con't)

Step 3: Encode the next state functions

C+

A \ BC	00	01	11	10
0	0	0	0	X
1	X	1	X	1

C+ = A

B+

A \ BC	00	01	11	10
0	1	1	0	X
1	X	0	X	1

B+ = B' + A'C'

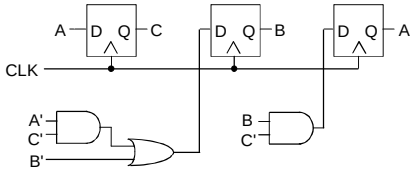
A+

A \ BC	00	01	11	10
0	0	1	0	X
1	X	1	X	0

A+ = BC'

5-state counter (con't)

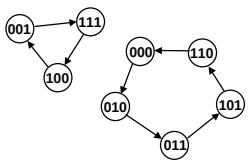
Step 4: Implement the design



5-state counter (con't)

- Is our design robust?
  - What if the counter starts in a 111 state?

Does our counter get stuck in invalid states???



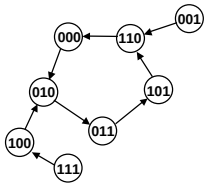
5-state counter (con't)

- Back-annotate our design to check it

Fill in state transition table

Present State			Next State		
C	B	A	C+	B+	A+
0	0	0	0	1	0
0	0	1	1	1	0
0	1	0	0	1	1
0	1	1	1	0	1
1	0	0	0	1	0
1	0	1	1	1	0
1	1	0	0	0	0
1	1	1	1	0	0

Draw state diagram



The proper methodology is to ***design*** your counter to be self-starting

Self-starting counters

- Invalid states should always transition to valid states
  - Assures startup
  - Assures bit-error tolerance
- Design your counters to be self-starting
  - Draw all states in the state diagram
  - Fill in the entire state-transition table
  - May limit your ability to exploit don't cares
    - Choose startup transitions that minimize the logic

Next subject: Finite-state machines

- Generalize the counter-design methodology
  - State machines have input signals
  - State machines have complex transitions
- Finite-state machines control digital systems
  - The core of digital design