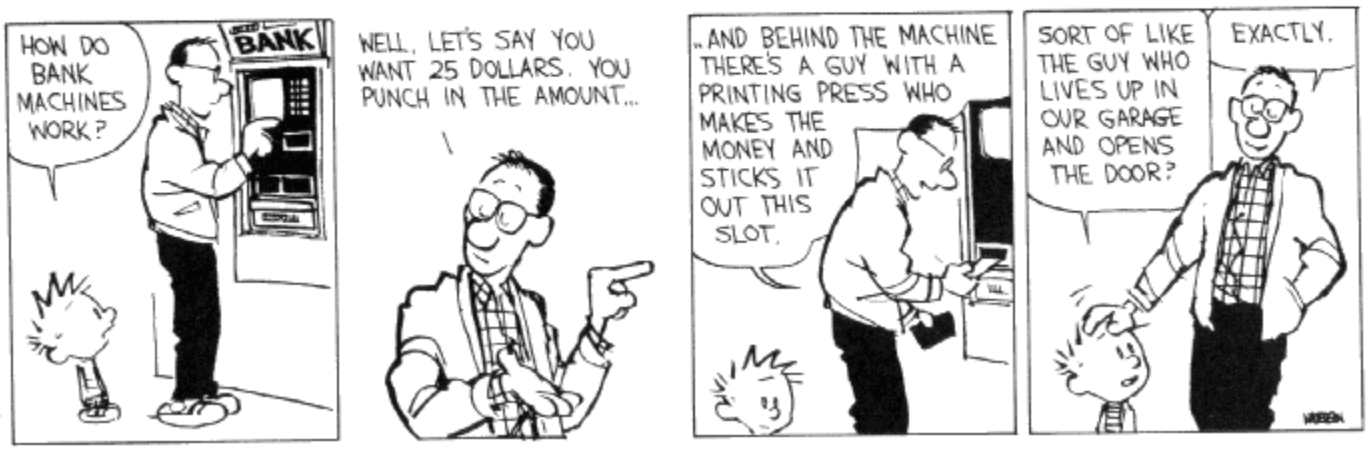
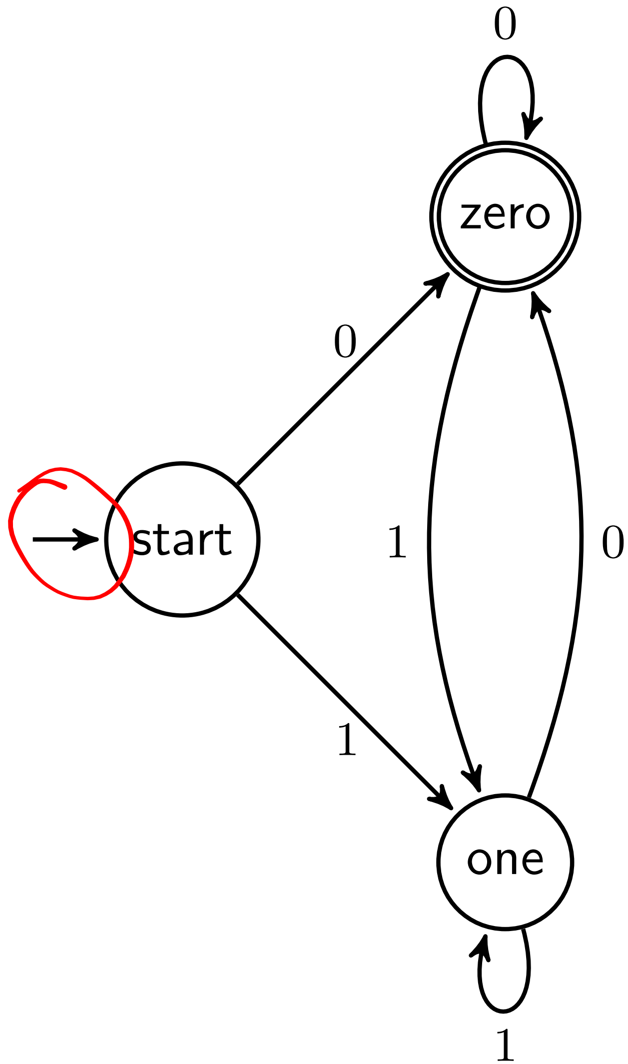
CSE 311: Foundations of Computing

Lecture 22: Finite State Machines



Last class: Strings this machine says are OK?

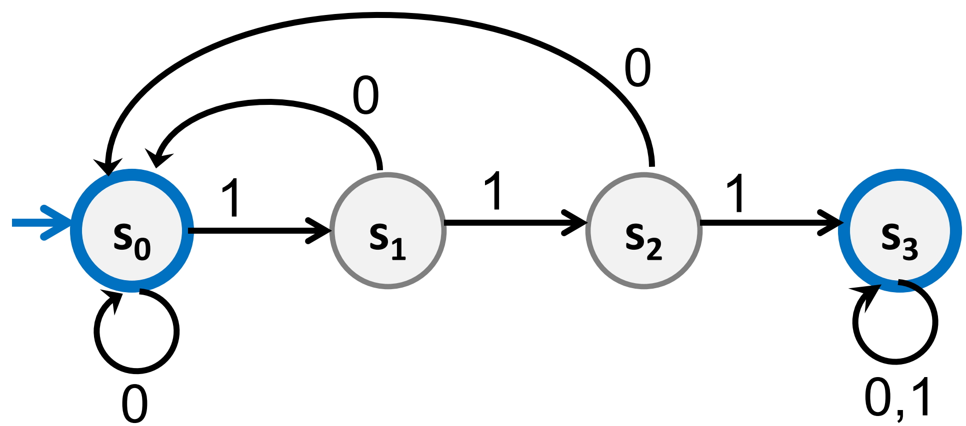


The set of all binary strings that end in 0

Finite State Machines

- States
- Transitions on input symbols
- Start state and final states
- The “language recognized” by the machine is the set of strings that reach a final state from the start

Old State	0	1
s_0	s_0	s_1
s_1	s_0	s_2
s_2	s_0	s_3
s_3	s_3	s_3



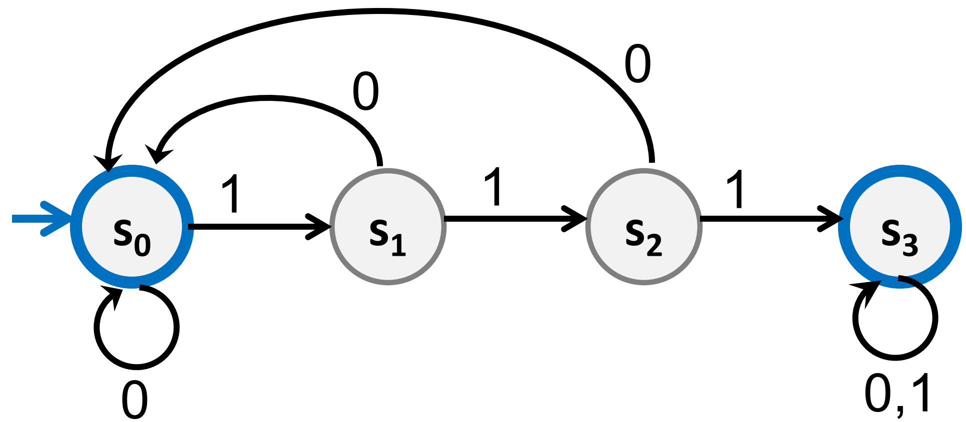
Finite State Machines

$\Sigma = \{0, 1\}$

- Each machine designed for strings over some fixed alphabet Σ.

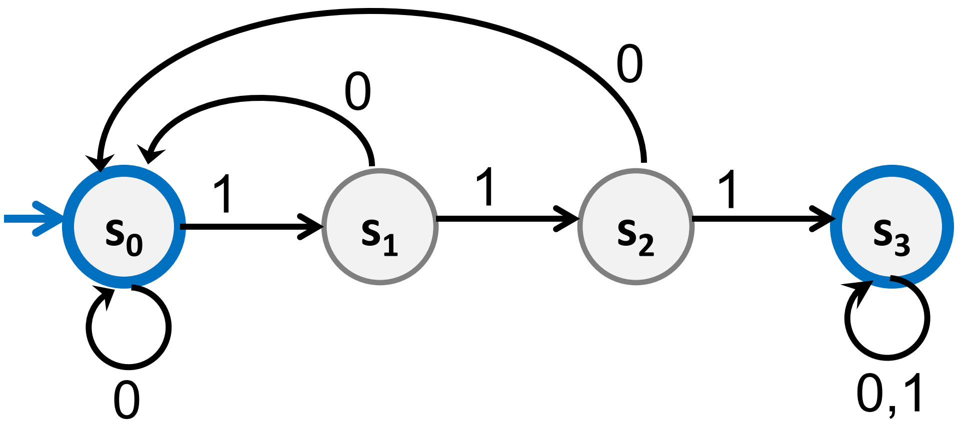
- Must have a transition defined from each state for ***every*** symbol in Σ.

Old State	0	1
s_0	s_0	s_1
s_1	s_0	s_2
s_2	s_0	s_3
s_3	s_3	s_3



What language does this machine recognize?

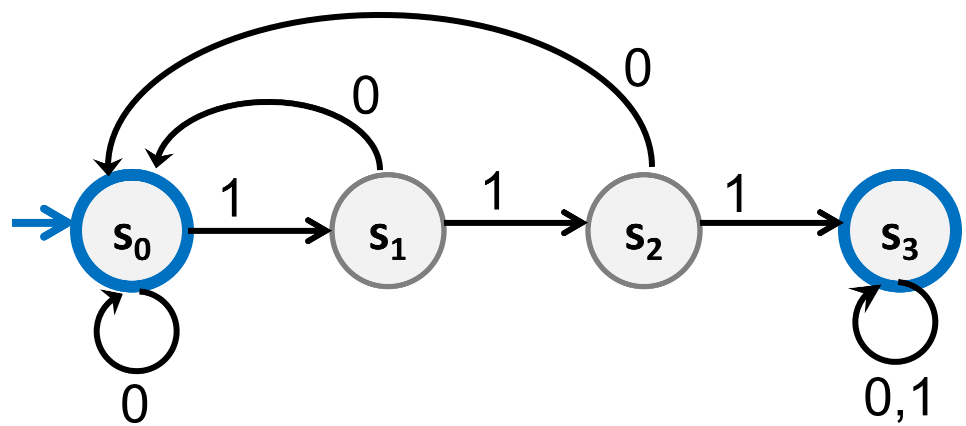
Old State	0	1
s_0	s_0	s_1
s_1	s_0	s_2
s_2	s_0	s_3
s_3	s_3	s_3



What language does this machine recognize?

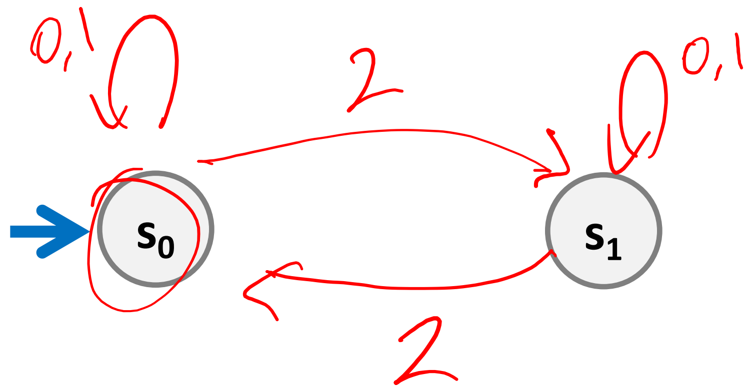
The set of all binary strings that contain 111
or don't end in 1

Old State	0	1
s_0	s_0	s_1
s_1	s_0	s_2
s_2	s_0	s_3
s_3	s_3	s_3



Strings over {0, 1, 2}

M_1: Strings with an even number of 2's

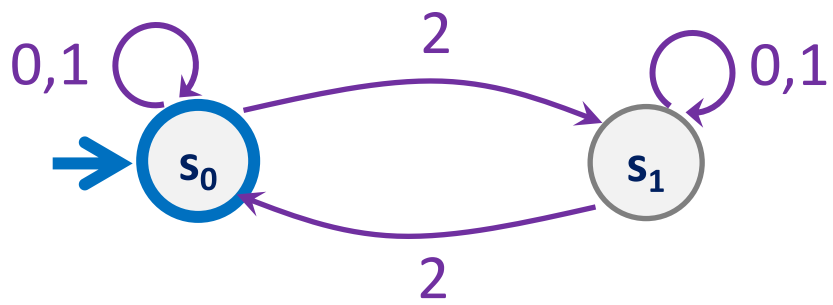


s_0: strings w/ even # of 2s

s_1: — — odd # of 2s

Strings over {0, 1, 2}

M_1: Strings with an even number of 2's



State Machine Design Recipe

Given a language, how do you design a state machine for it?

Create states to remember enough

(about the portion of the input string that it has already seen)

to correctly answer "accept/reject" on the whole string after seeing the rest.

Add labeled edges to show how the memory (state) should be updated for each new symbol.

Strings over {0, 1, 2}

M_2: Strings where the sum of digits mod 3 is 0

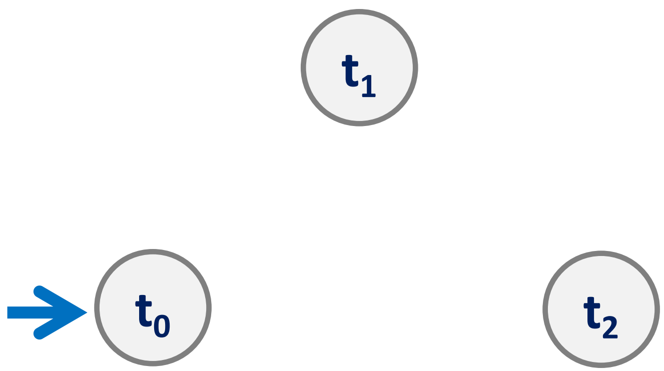
S_0: strings whose sum of digits mod 3 is 0

S_1: Strings ______ 1

S_2: Strings ______ 2

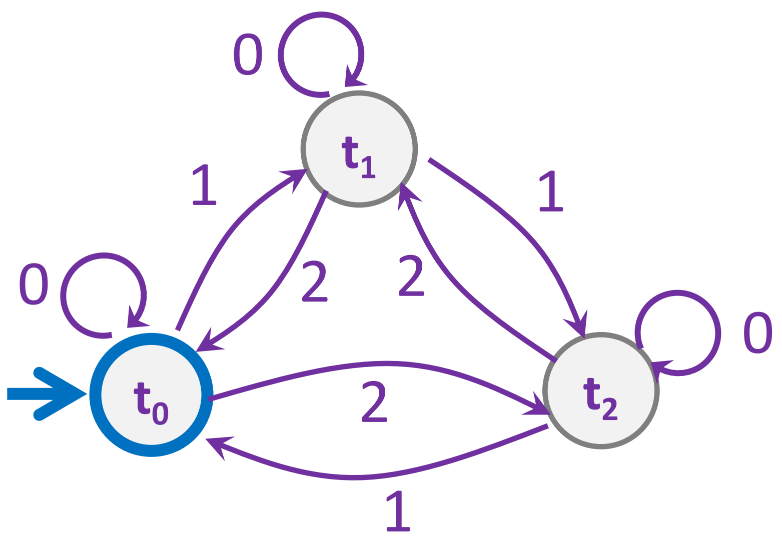
Strings over {0, 1, 2}

M_2: **Strings where the sum of digits mod 3 is 0**

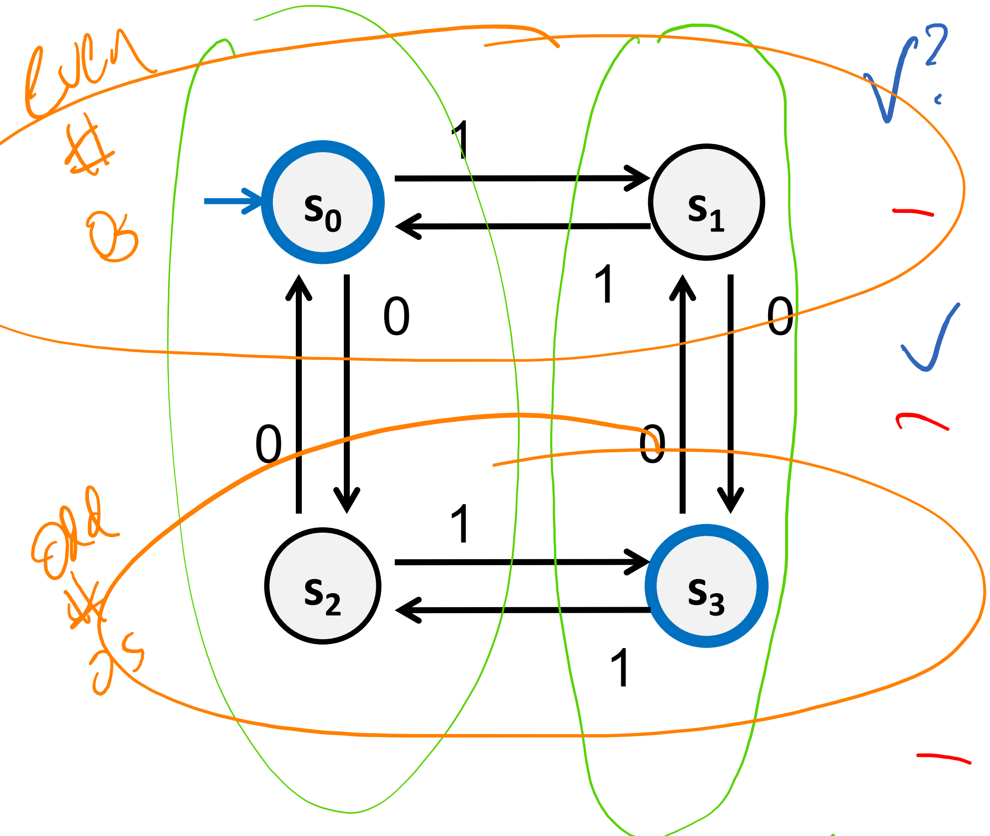


Strings over {0, 1, 2}

M_2: Strings where the sum of digits mod 3 is 0



What language does this machine recognize?



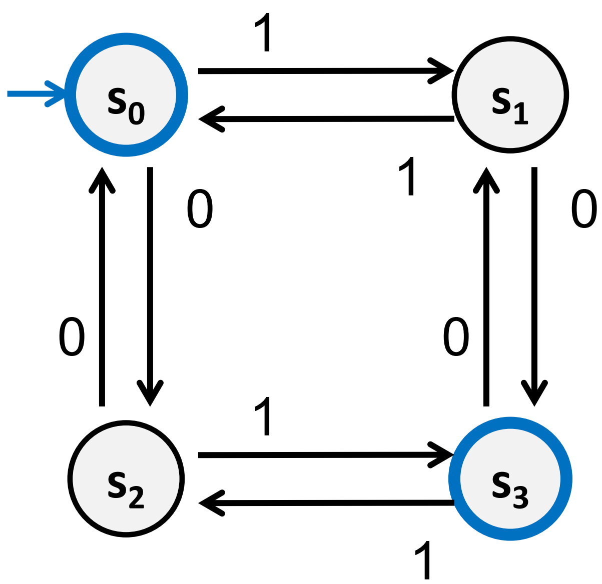
001000

even #'s

odd # 1s

- even # of digits (length)
- even # of 0s and even # of 1s or odd # 0s and odd # 1s
- $10^* \cup 01^*$
- 2nd bullet point and can't have more than 2 consecutive 0s or 1s

What language does this machine recognize?

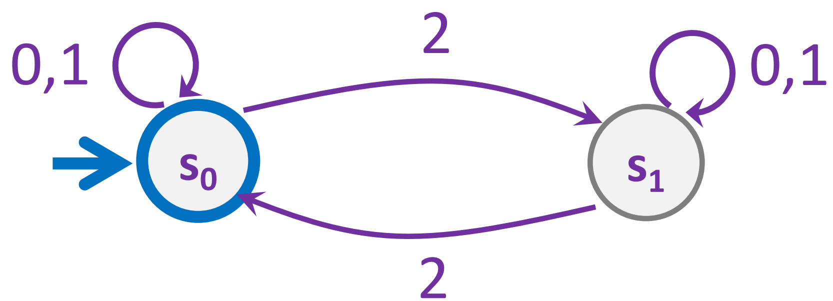


The set of all binary strings with # of 1's ≡ # of 0's (mod 2) (both are even or both are odd).

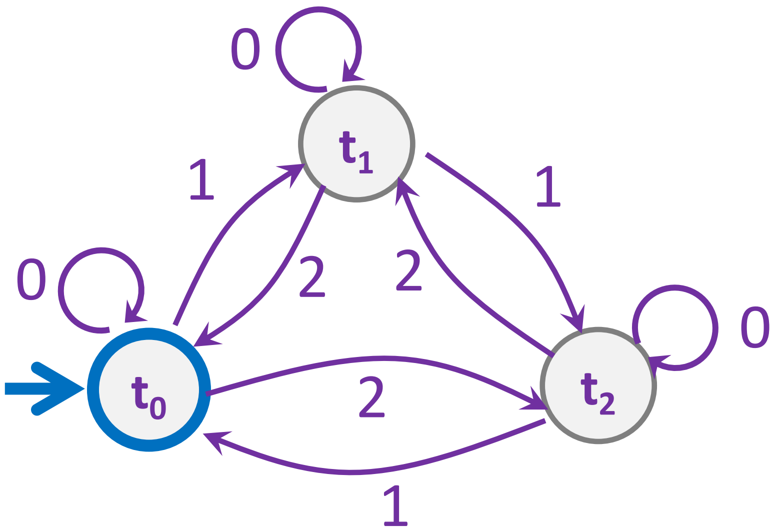
Can you think of a simpler description?

Strings over {0, 1, 2}

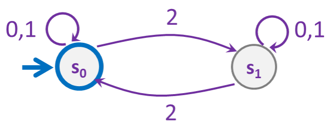
M_1: Strings with an even number of 2's

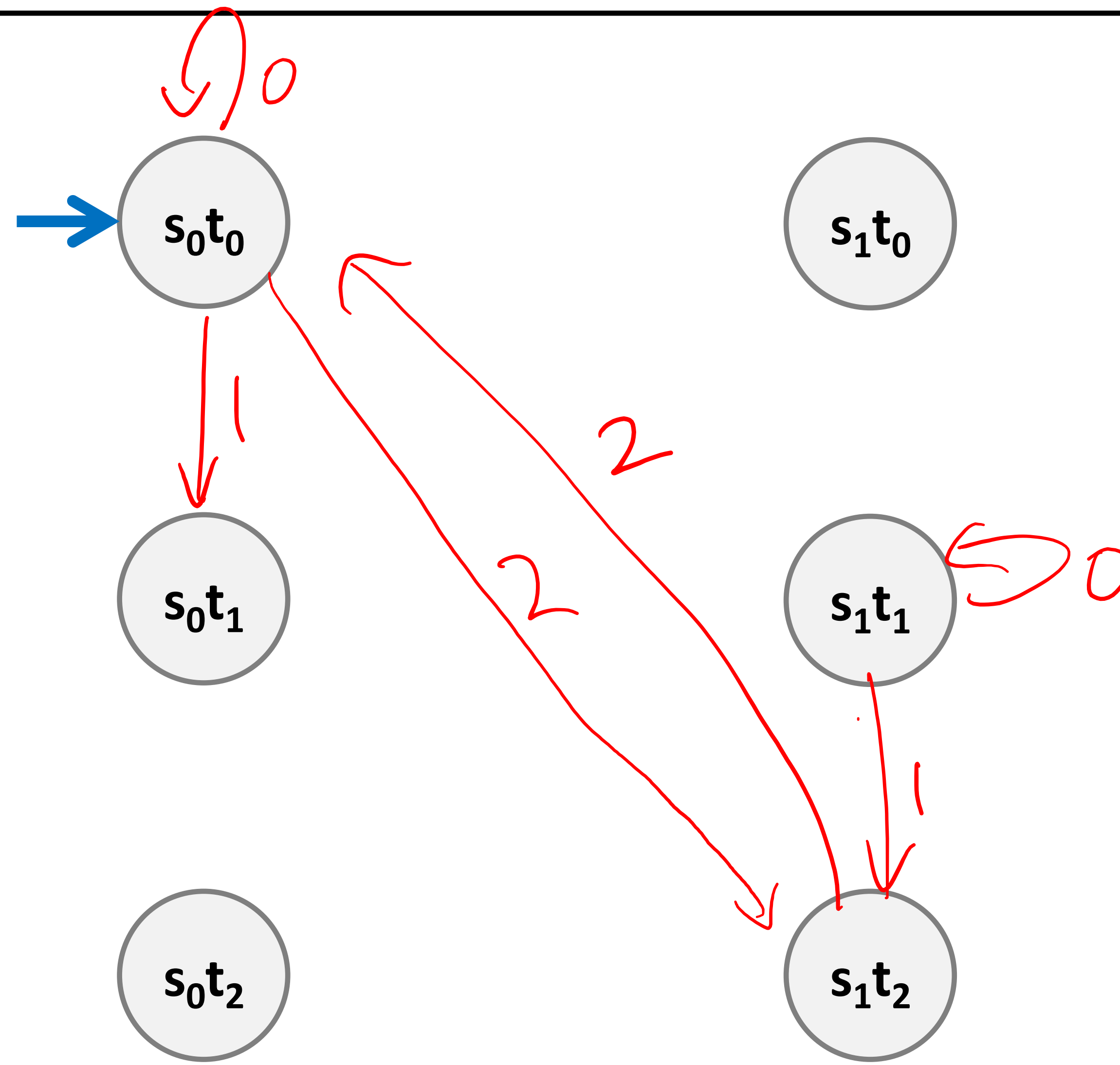


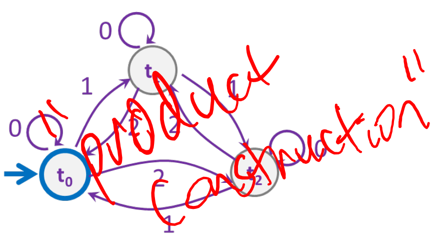
M_2: Strings where the sum of digits mod 3 is 0



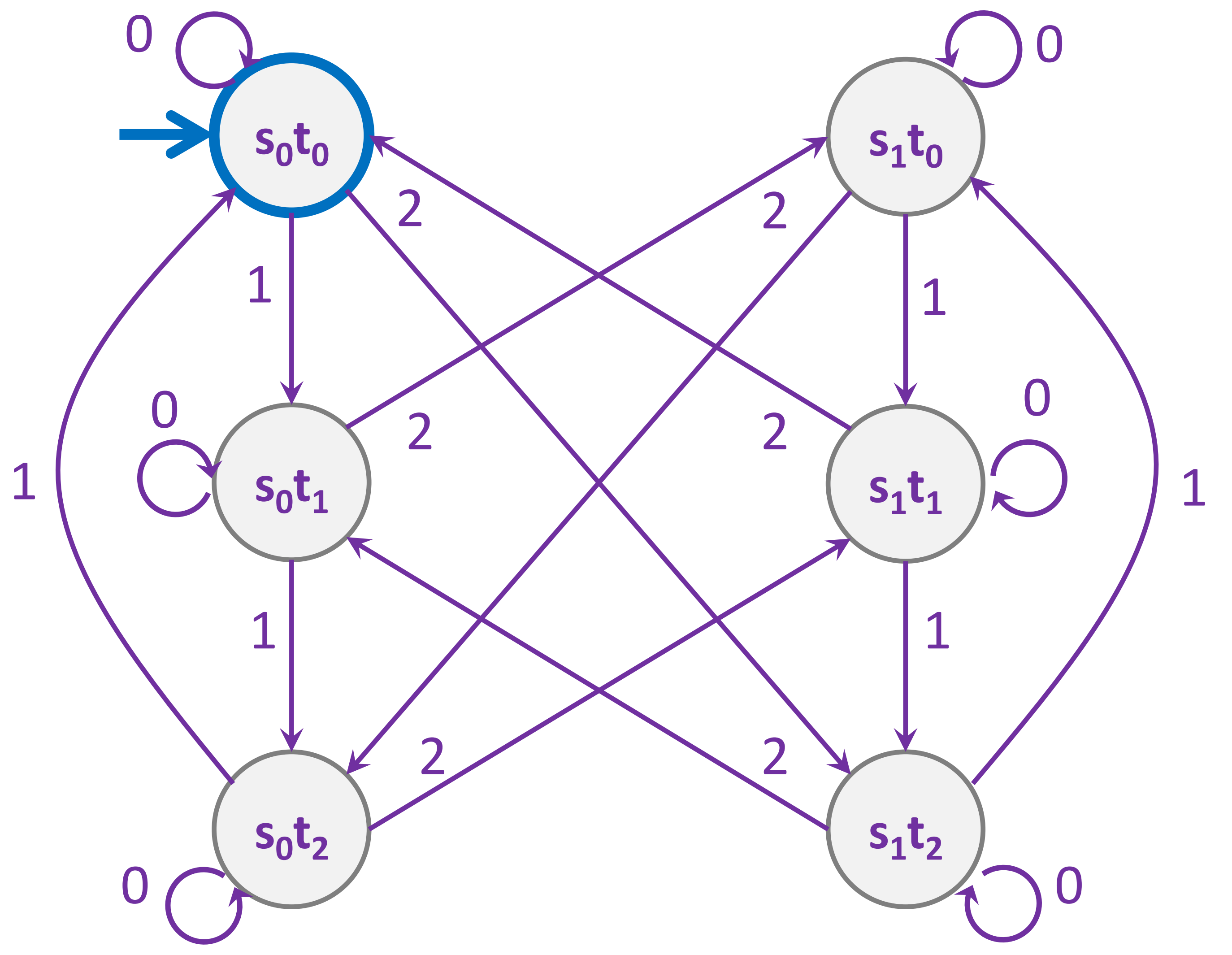
Strings over {0,1,2} w/ even number of 2's and mod 3 sum 0



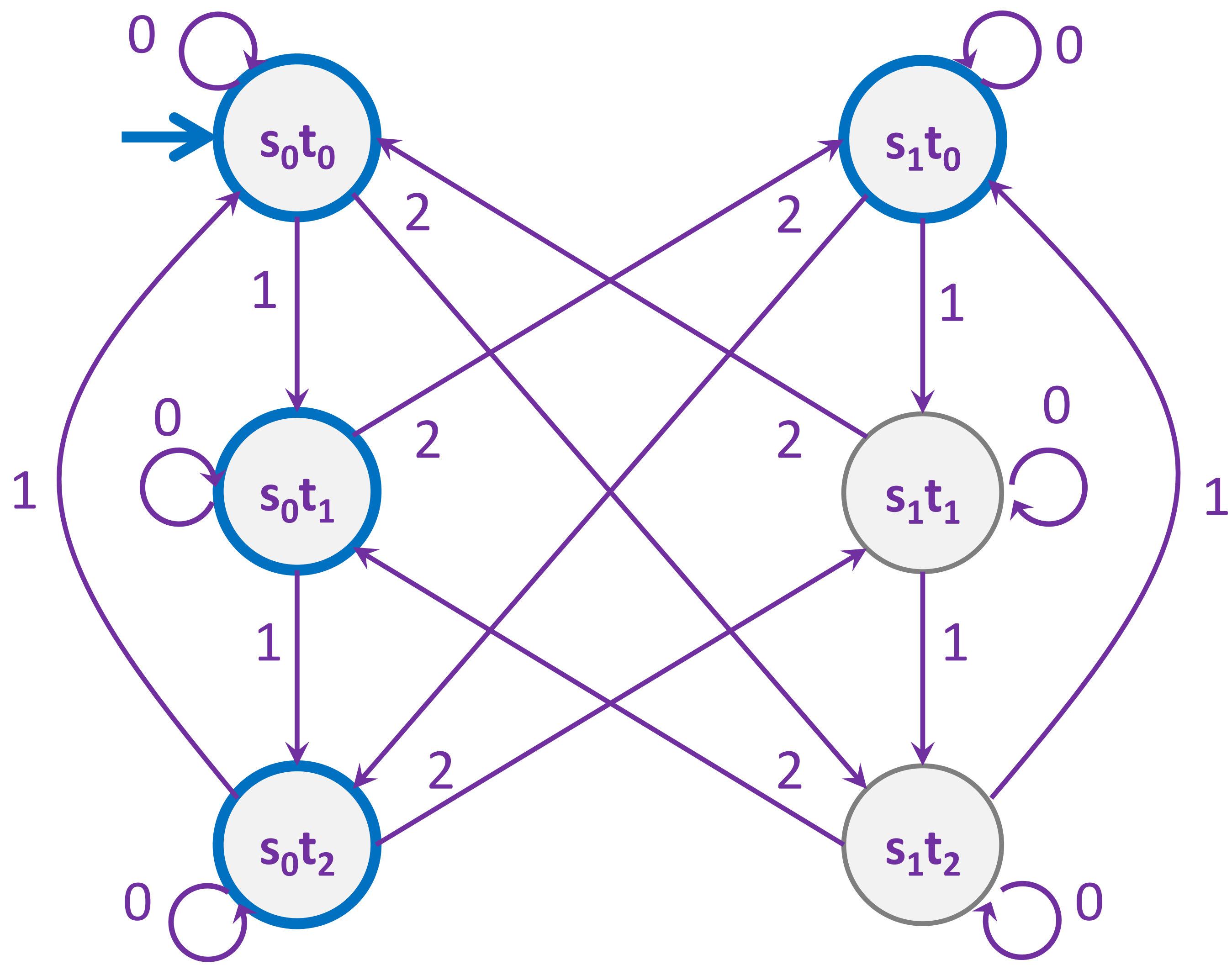




Strings over {0,1,2} w/ even number of 2's and mod 3 sum 0



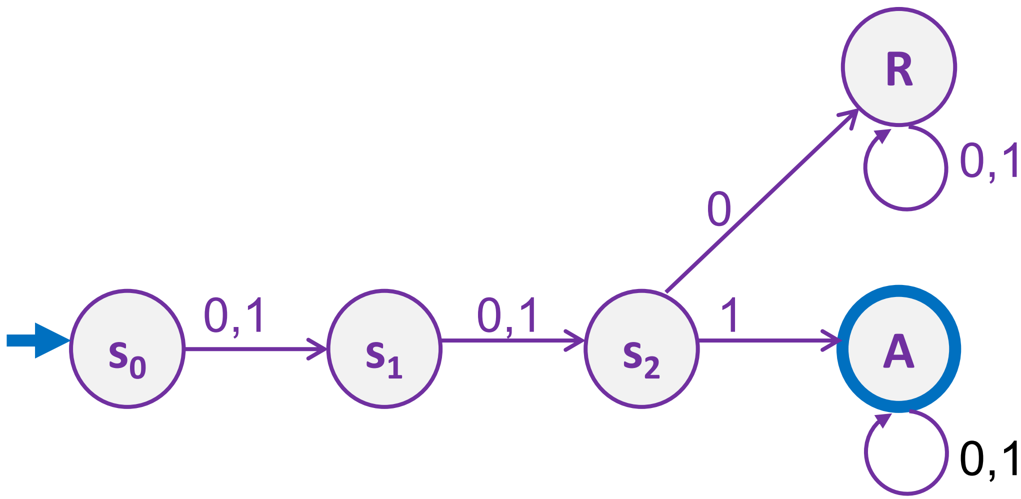
Strings over {0,1,2} w/ even number of 2's OR mod 3 sum 0



$\Sigma = \{0, 1\}$

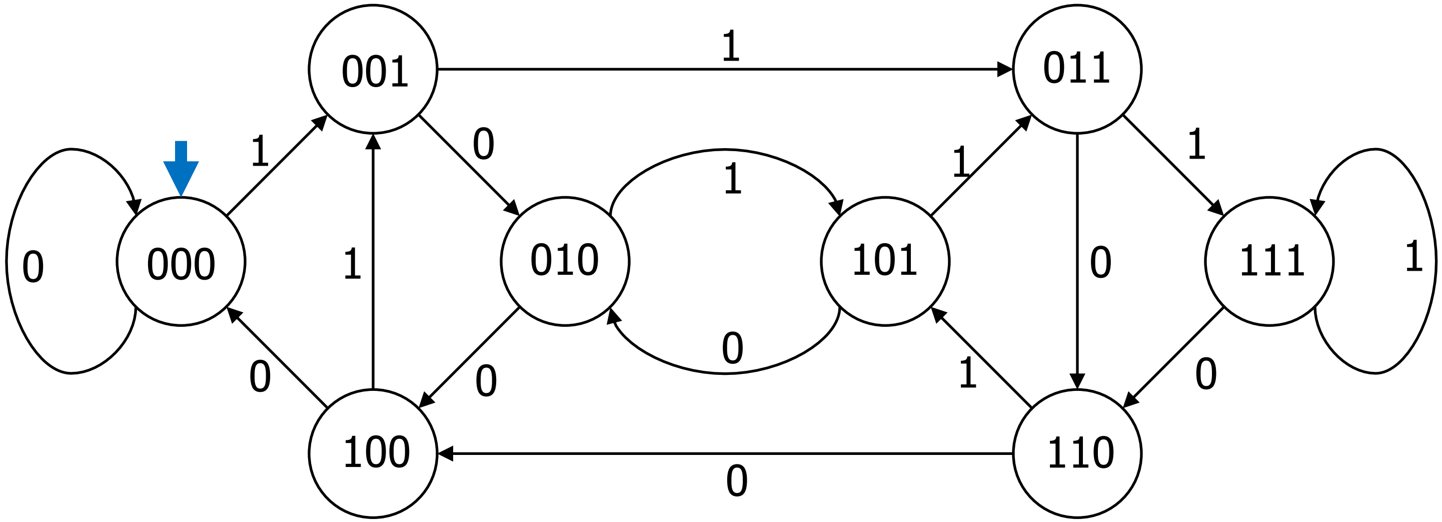
The set of binary strings with a 1 in the 3rd position from the start

The set of binary strings with a 1 in the 3rd position from the start

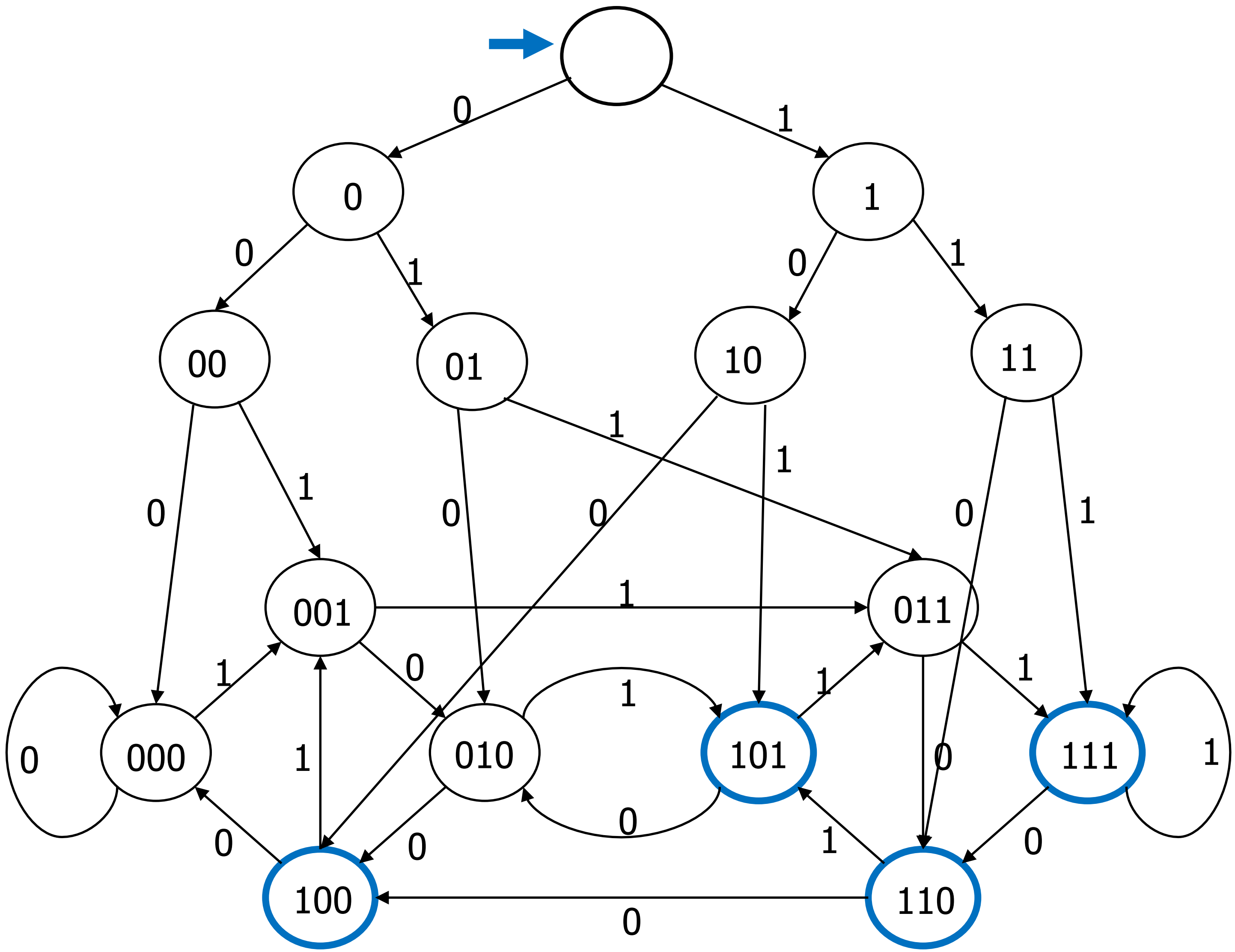


The set of binary strings with a 1 in the 3rd position from the end

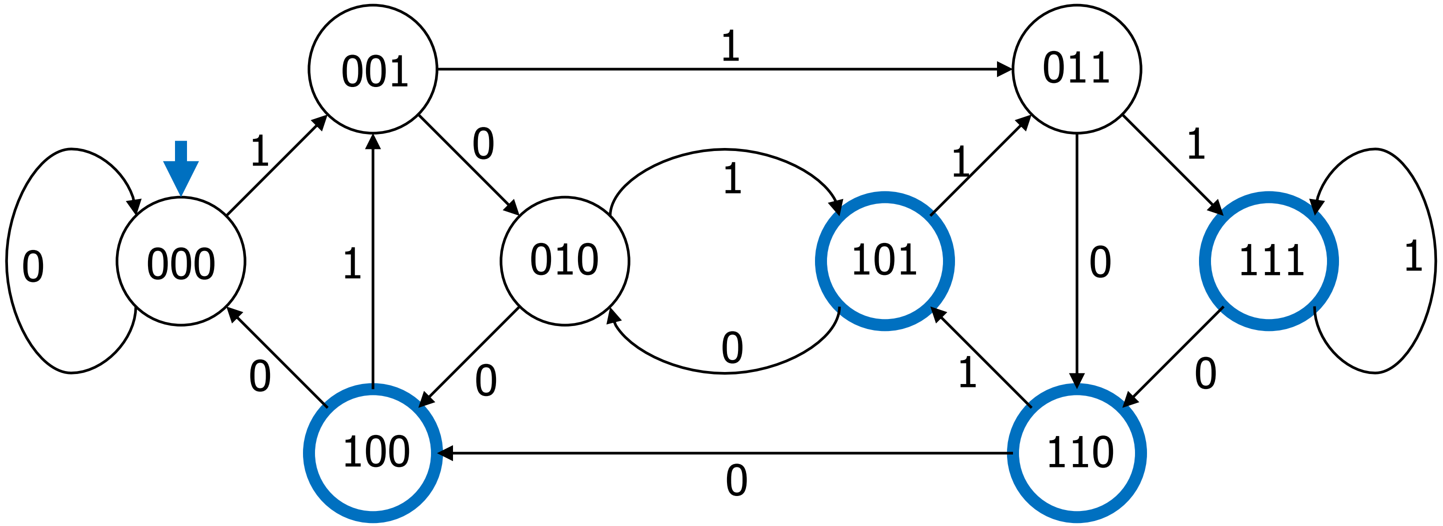
3 bit shift register “Remember the last three bits”



The set of binary strings with a 1 in the 3rd position from the end



The set of binary strings with a 1 in the 3rd position from the end



The beginning versus the end