Synchronous Sequential Circuits

Instructor: Dr. Vinicius Prado da Fonseca (vpradodafons@online.mun.ca)

Synchronous Sequential Circuits

- Circuits that contain feedback paths are called sequential.
- Latches and flip flops are sequential circuits.
- Combinational circuits have no feedback paths.
- To avoid unwanted oscillations and/or unwanted memory effects the synchronous sequential design discipline is adopted:
  - Isolate any loop and;
  - synchronize the access to memory.

Synchronous Sequential Circuits Design rules

- Every circuit element is either:
  - A register (group of D flop-flops) or;
  - a combinational circuit.
- At least one circuit is a register.
- All registers receive the same clock signal.
- Every cyclic path contains at least one register.

Synchronous Sequential Circuits Design rules

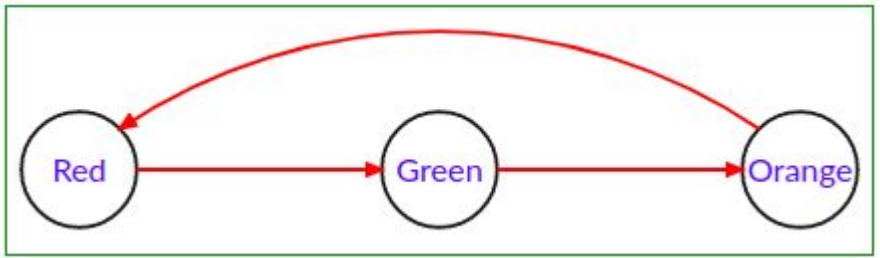
- The registers (DDFs) break any feedback loops, since the registers only change when the clock changes.
  - Stability and control
- Since a common clock is used all register will change at the same instant.
- Since the change is synchronized by the common clock, the circuit is synchronous.

Synchronous Sequential Circuits as FSM

- Finite state machines (FSM).
- Defined by:
  - Set of states.
  - Initial state.
  - Transition between the states.
  - Inputs and outputs.
  - Goal state (sometimes).
- Two block of combinational circuits
  - Next state logic.
  - Output logic.
- Registers
  - Current state

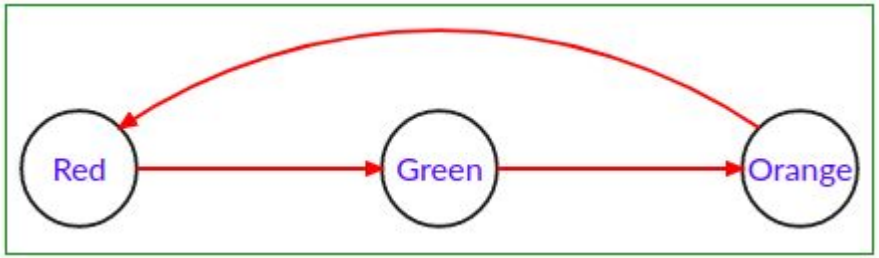
Finite State Machines and State Graphs

- Synchronous digital systems can be designed using Finite State Machines (FSM).
- A FSM models the system as a set of states, and a set of transitions between states.
- FSMs are represented by state graphs (state diagrams)



Finite State Machines and State Graphs

A state diagram can also be represented by a next state table, that shows which states follow which states



Current State	Next State
Red	Green
Green	Orange
Orange	Red

State Encodings For Traffic Lights

- Digital circuit that implements a FSM, the states must be encoding as 0s and 1s
- One possible encoding is:
  - Red = 00
  - Green = 01
  - Orange = 10
- Circuit example
  - a and b (current state)
  - x anb y (next state)
  - x = b
  - y = ~a&~b

Current State	Next State
00	01
01	10
10	00

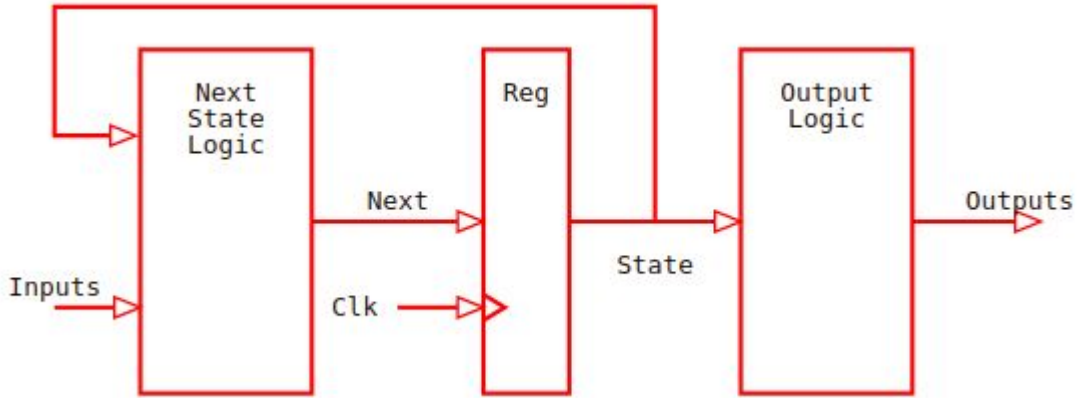
State Encodings For Traffic Lights

- Another common state encoding is one hot
- N state machine requires N bits
- Decoder use?
- Counters?

Current State	Next State
001	010
010	100
100	001

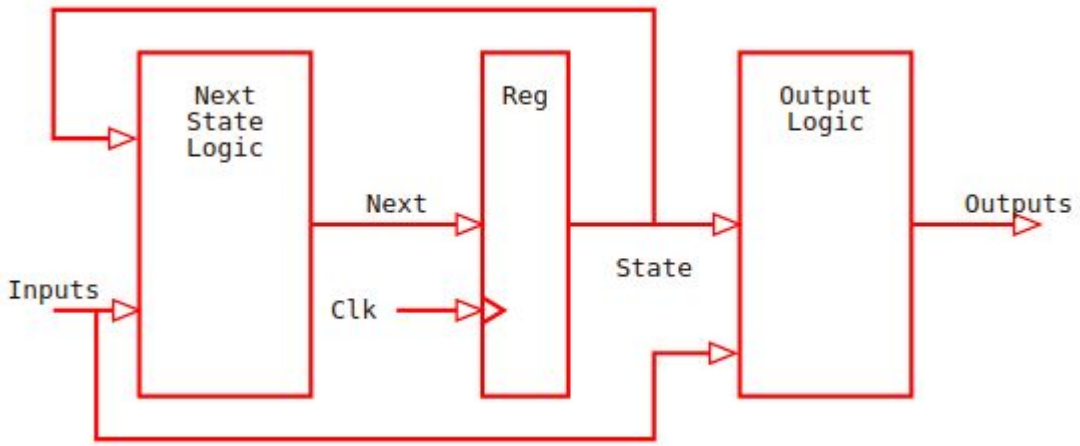
Moore and Mealy Machine

- Theoretical models of Finite State Machines
- State The block diagram for a Moore machine is



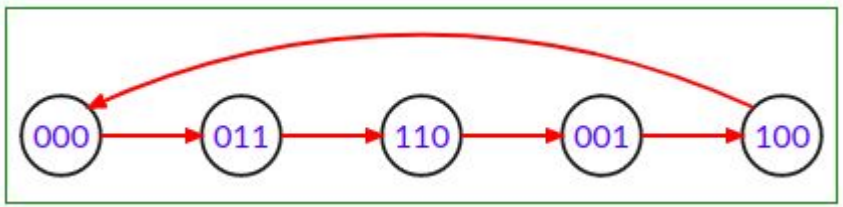
Moore and Mealy Machine

- The block diagram for the Mealy machine
- Mealy machine differs from the Moore machine in the output depends on the current state and current inputs



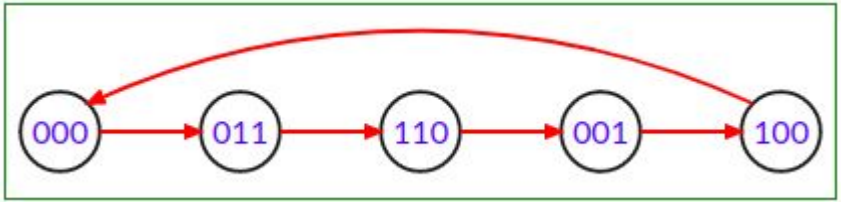
Counters

- Implement a digital circuit to store the next states
- Fixed count sequence
- Simplest synchronous system
  - 000 -> 011 -> 110 -> 001 -> 100 -> 000



Counters

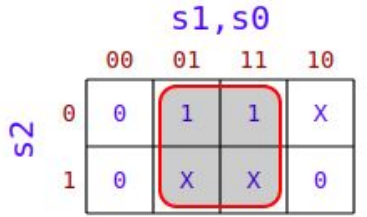
- Three bit register that stores the current state
- Combinational logic that given the current state, outputs the next state
- Three next state functions (n2, n1, n0) -> three-bit states
- 010, 101, and 111 are don't cares.



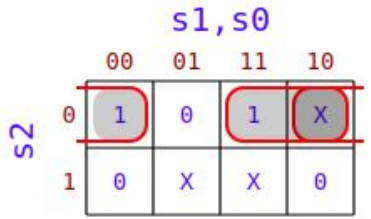
s2	s1	s0	n2	n1	n0
0	0	0	0	1	1
0	0	1	1	0	0
0	1	0	X	X	X
0	1	1	1	1	0
1	0	0	0	0	0
1	0	1	X	X	X
1	1	0	0	0	1
1	1	1	X	X	X

Counters

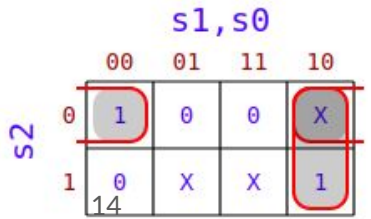
s2	s1	s0	n2	n1	n0
0	0	0	0	1	1
0	0	1	1	0	0
0	1	0	X	X	X
0	1	1	1	1	0
1	0	0	0	0	0
1	0	1	X	X	X
1	1	0	0	0	1
1	1	1	X	X	X



$n2 = s0$



$n1 = \overline{s2} \cdot \overline{s0} + \overline{s2} \cdot s1$



$n0 = \overline{s2} \cdot \overline{s0} + s1 \cdot \overline{s0}$

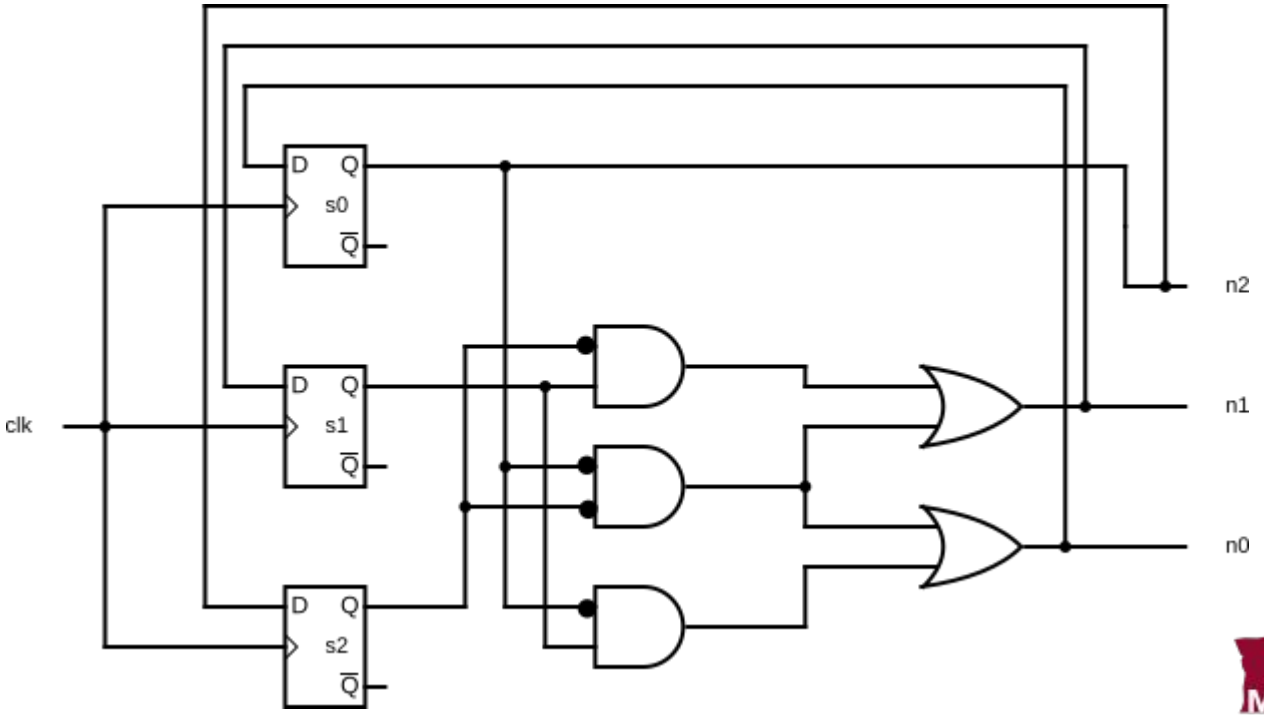
Counters

Assume the current state is 000, then s2=0, s1=0, and s2=0

$$
\begin{aligned}
n2 &= s0 = 0 \\
n1 &= \overline{s2} \cdot \overline{s0} + \overline{s2} \cdot s1 = 1 \cdot 1 + 1 \cdot 0 = 1 \\
n0 &= \overline{s2} \cdot \overline{s0} + s1 \cdot \overline{s0} = 1 \cdot 1 + 0 \cdot 1 = 1
\end{aligned}
$$

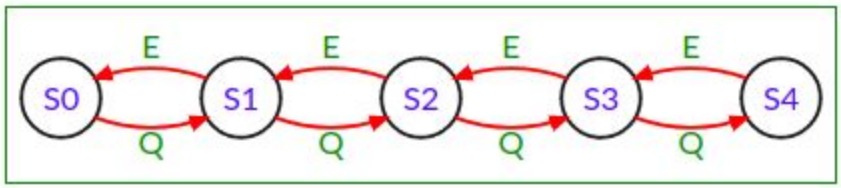
Thus the next state is 011, which is correct.

Counters



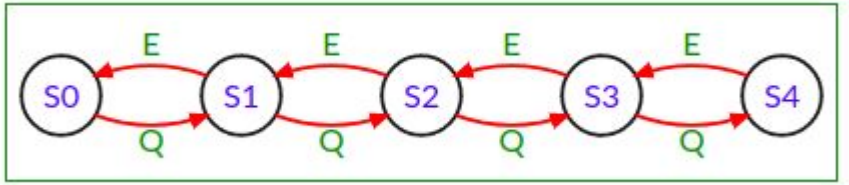
State Diagrams With Inputs

- A parking meter can be modeled by a FSM
- 30 minutes time slots the vehicle can park
- Moves between states
  - A quarter is deposited
  - Half hour time slot has elapsed



State Diagrams With Inputs

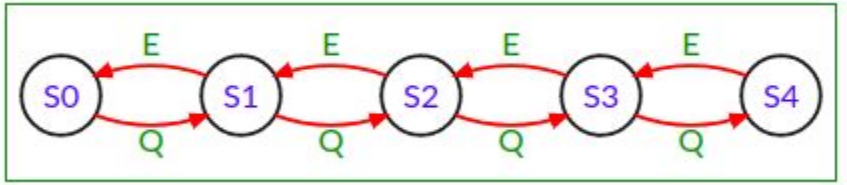
- Parking meter next state table
- State transitions
  - S0 - E -> S0
  - S0 - Q -> S1
  - S1 - Q -> S2
  - S2 - Q -> S3
  - S3 - E -> S2
  - S2 - Q -> S3
- How many bits to encode state?
- How many bits to encode input?



Current State	Input Event	Next State
S0	Q	S1
S1	E	S0
S1	Q	S2
S2	E	S1
S2	Q	S3
S3	E	S2
S3	Q	S4
S4	E	S3

State Diagrams With Inputs

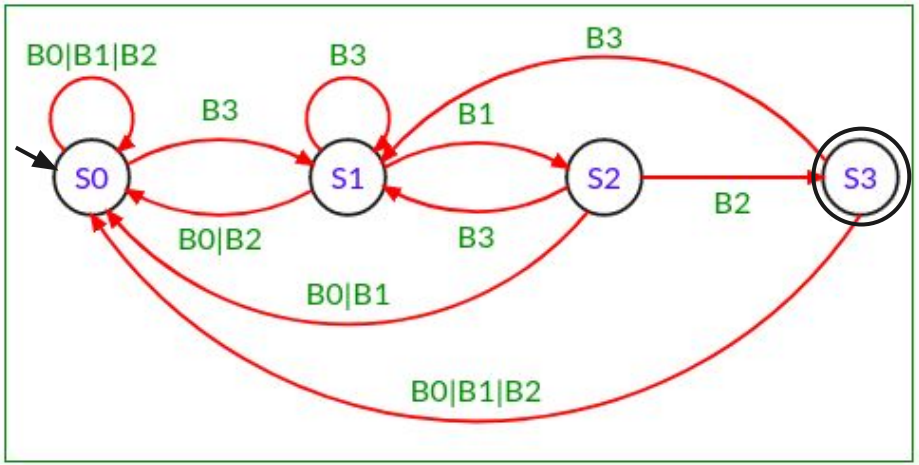
- Parking meter next state table
- State transitions
  - S0 - E -> S0
  - S0 - Q -> S1
  - S1 - Q -> S2
  - S2 - Q -> S3
  - S3 - E -> S2
  - S2 - Q -> S3
- How many bits to encode state?
  - 5 states, 3 bits, $2^3 = 8$
- How many bits to encode input?
  - 1 bit, 0 or 1.
- 3 + 1 bits, $2^4=16$



Current State	Input Event	Next State
S0	Q	S1
S1	E	S0
S1	Q	S2
S2	E	S1
S2	Q	S3
S3	E	S2
S3	Q	S4
S4	E	S3

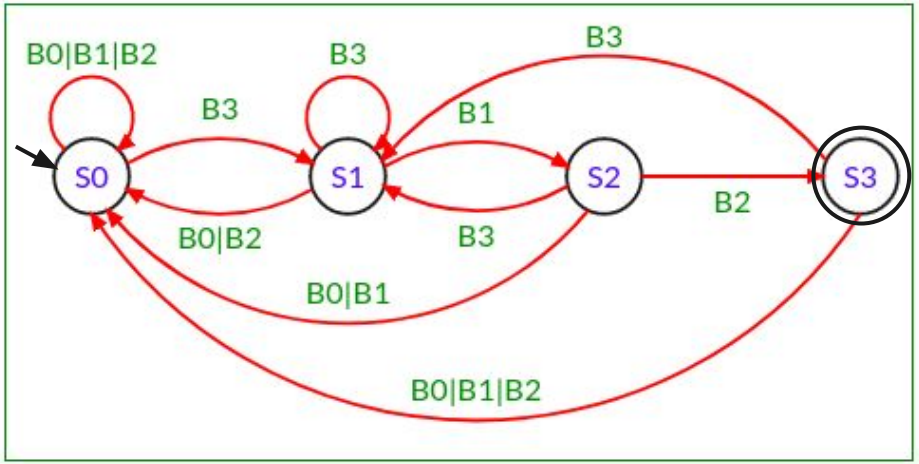
Four Button Electric Lock

- Sequence of button presses to control a lock
- Correct sequence is pressed the lock will be unlocked
- A FSM can be used to detect when the correct sequence has been entered



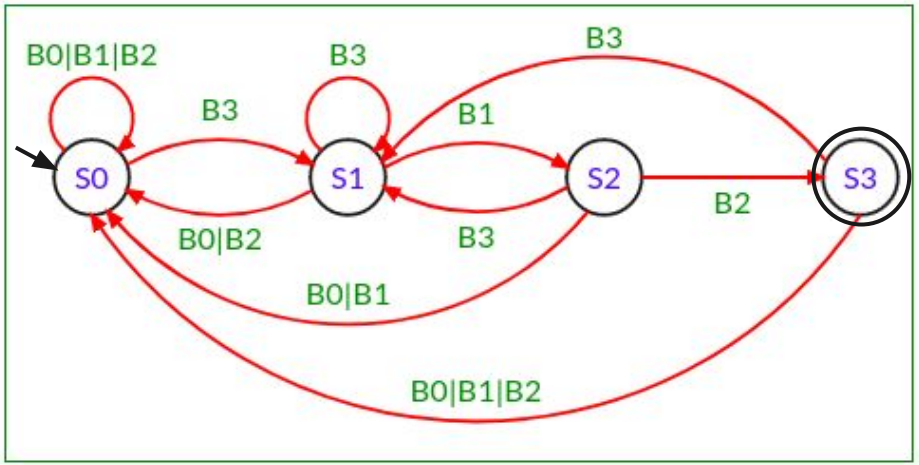
Four Button Electric Lock

- The sequence B3, B1 and B2 will unlock the the lock.
- State S3 indicates that the lock should be unlocked.
- S0 is the starting state.

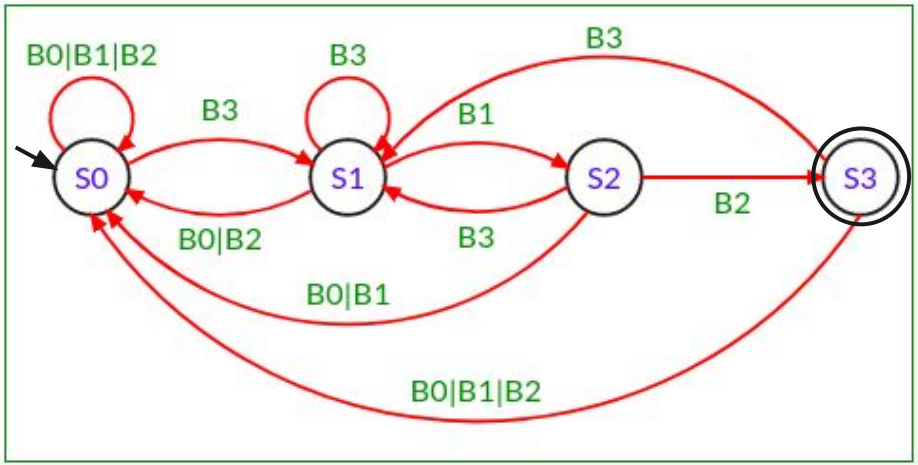


Four Button Electric Lock

- S0 - B0 -> S0
- S0 - B3 -> S1
- S1 - B3 -> S1
- S1 - B1 -> S2
- S2 - B2 -> S3



Four Button Electric Lock



Current State	Input Event	Next State
S0	B0	S0
S0	B1	S0
S0	B2	S0
S0	B3	S1
S1	B0	S0
S1	B1	S2
S1	B2	S0
S1	B3	S1
S2	B0	S0
S2	B1	S0
S2	B2	S3
S2	B3	S1
S3	B0	S0
S3	B1	S0
S3	B2	S0
S3	B3	S1

Questions?

- Next

  15 - Modeling synchronous systems in Verilog