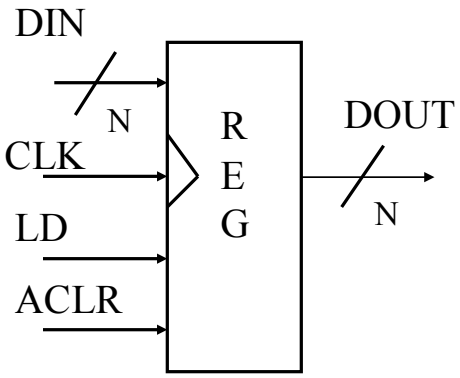
Registers

The most common sequential building block is the register. A register is N bits wide, and has a load line for loading in a new value into the register.

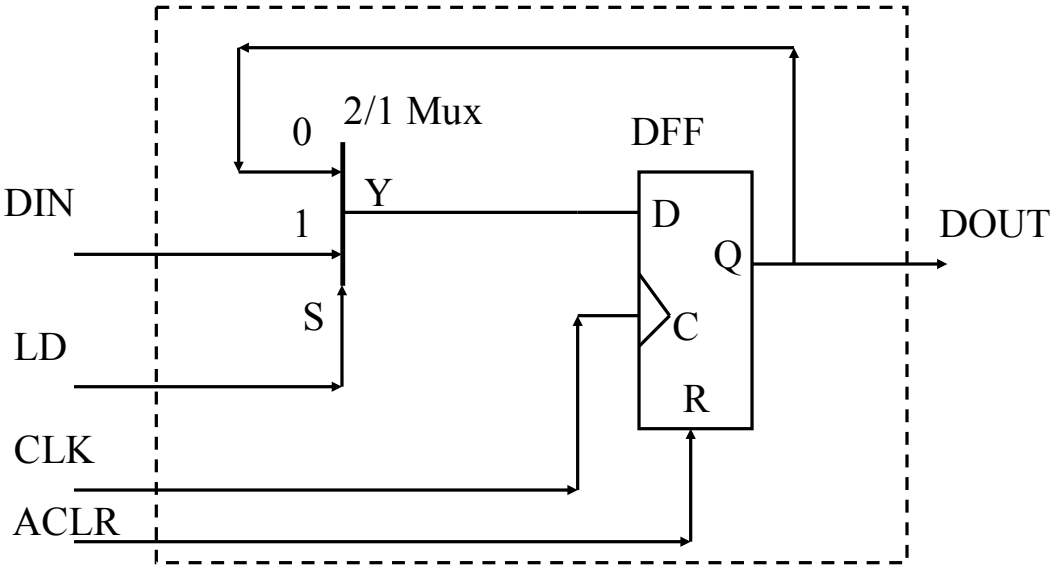


Register contents do not change unless LD = 1 on active edge of clock.

A DFF is NOT a register! DFF contents change every clock edge.

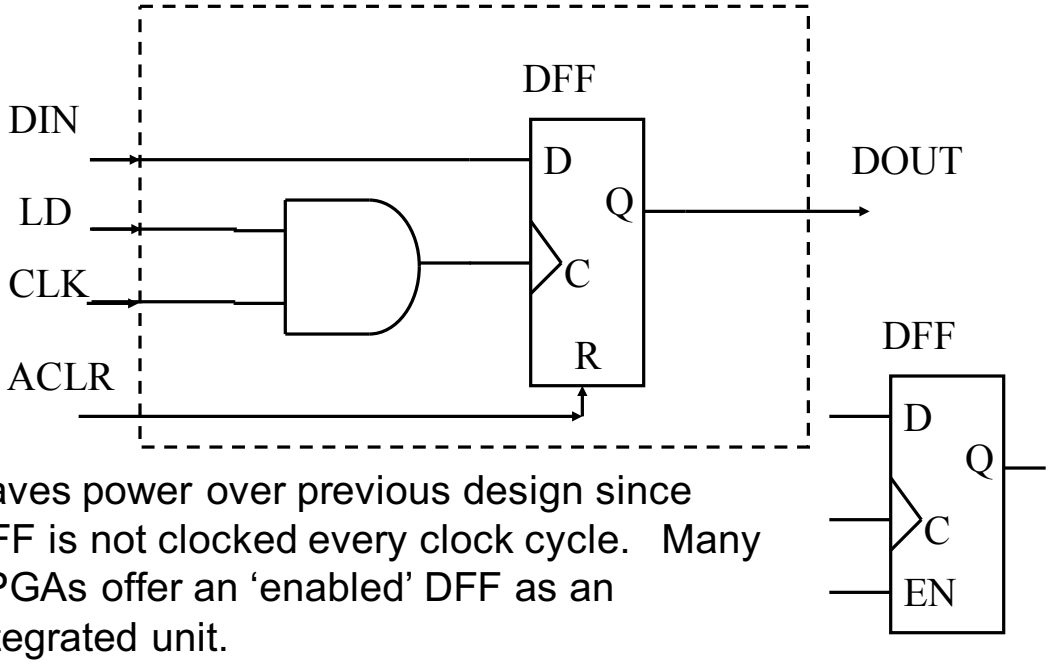
ACLR used to asynchronously clear the register

1 Bit Register using DFF, Mux



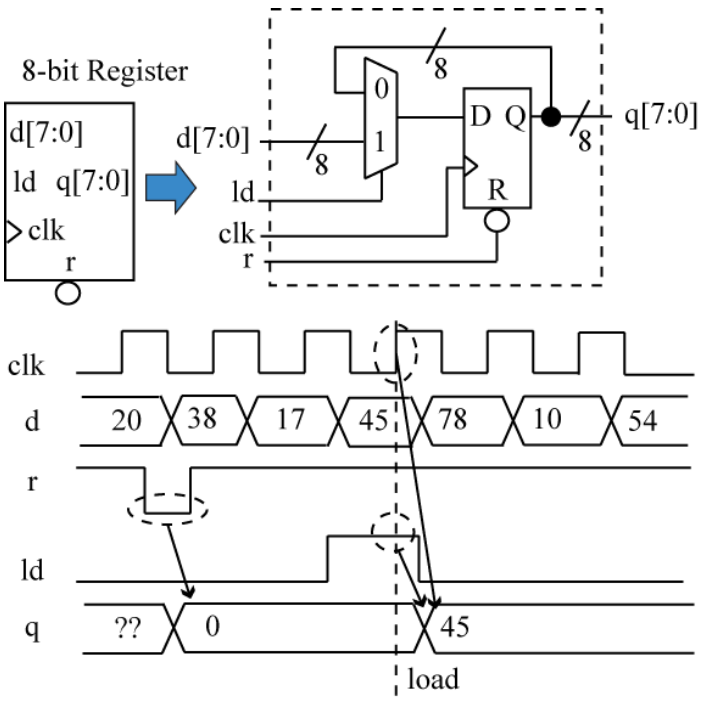
Note that DFF simply loads old value when LD = 0. DFF is loaded every clock cycle.

1 Bit Register using Gated Clock



Saves power over previous design since DFF is not clocked every clock cycle. Many FPGAs offer an 'enabled' DFF as an integrated unit.

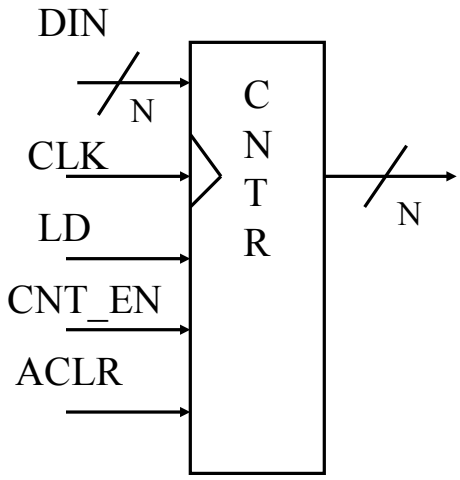
8-bit Register




```
module reg8bit(clk,r,d,ld,q);

input clk,r,ld;
input [7:0] d;
output [7:0] q;

reg [7:0] q;

//register
always @(posedge clk
         or negedge r)
begin
 //async reset
 if (!r) q <= 8'b0;
 else begin
 //synchronous inputs
  if (ld) q <= d;
 end
end

endmodule
```

Counter

Very useful sequential building block. Used to generate memory addresses, or keep track of the number of times a datapath operation is performed.

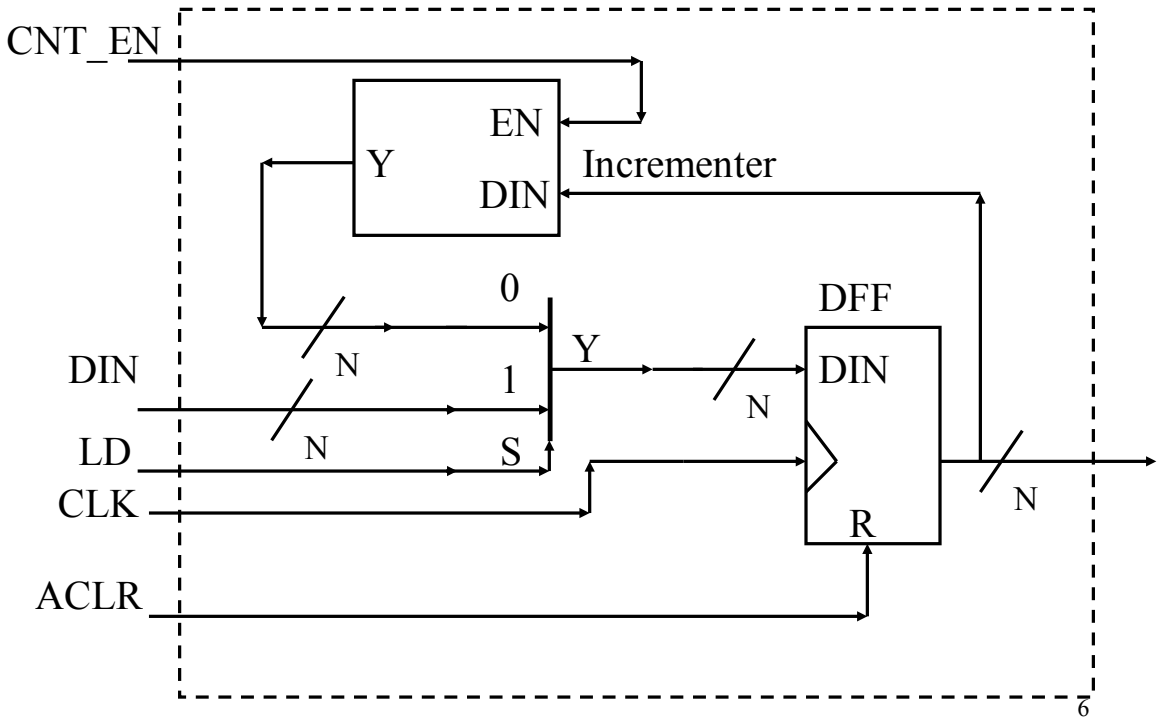


LD asserted: loads counter with DIN value.

CNT_EN asserted will increment counter on next active clock edge.

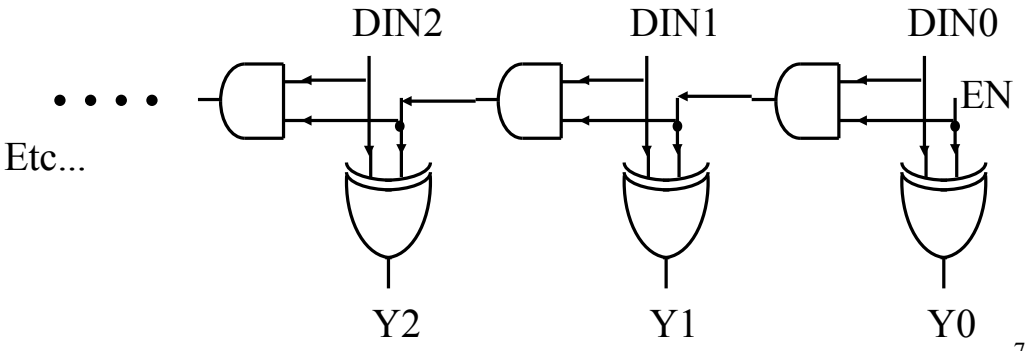
ACLR will asynchronously clear the counter.

One way to build a Counter

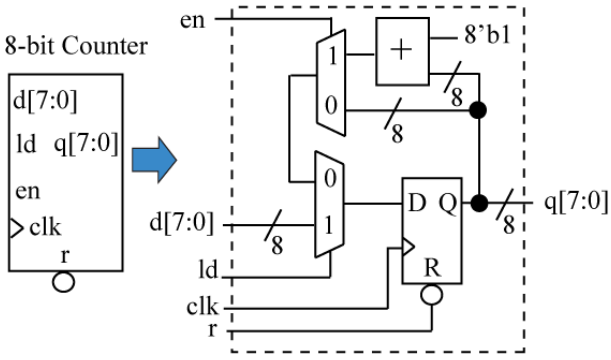


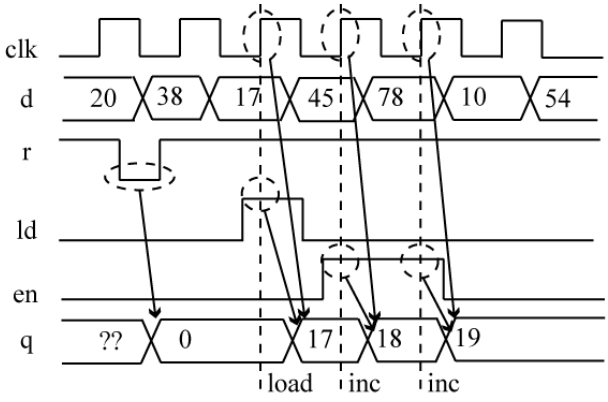
Incrementer: Combinational Building Block

When EN=1, Y = DIN + 1
When EN=0, Y = DIN



8-bit Counter






```
module cnt8bit (clk,r,d,
                ld,en,q);

input clk,r,ld,en;
input [7:0] d;
output [7:0] q;

reg [7:0] q;

//register
always @(posedge clk
         or negedge r) begin
 //async reset
 if (!r) q <= 8'b0;
 else begin
 //synchronous inputs
  if (en) q <= q + 8'b1;
  if (ld) q <= d;
 end
end

endmodule
```

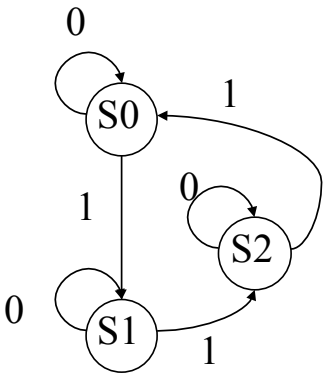
Sequential System Description

- The Q outputs of the flip-flops form a state vector
- A particular set of outputs is the Present State (PS)
- The state vector that occurs at the next discrete time (clock edge for synchronous designs) is the Next State (NS)
- A sequential circuit described in terms of state is a Finite State Machine (FSM)
  - Not all sequential circuits are described this way; i.e., registers are not described as FSMs yet a register is a sequential circuit.

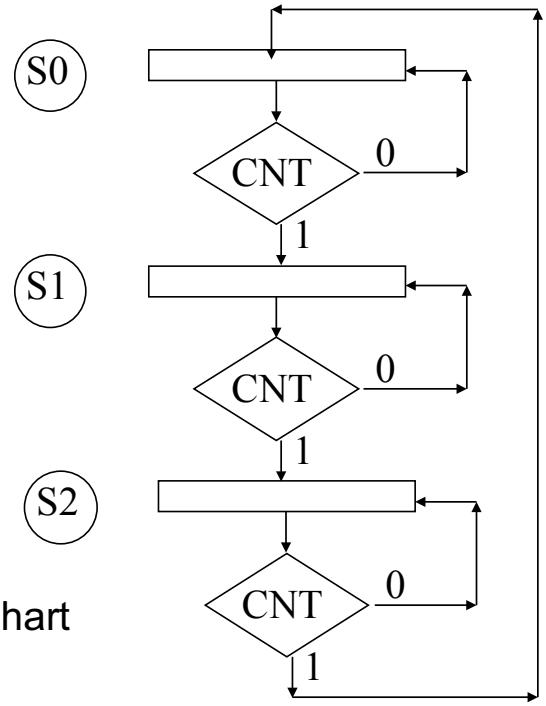
Describing FSMs

- State Tables
- State Equations
- State Diagrams
- Algorithmic State Machine (ASM) Charts
  - Preferred method in this class
- HDL descriptions

Example State Machine



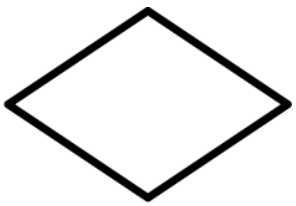
State Diagram
(Bubble Diagram)



ASM Chart

ASM Chart Symbols

State-Representing
Symbol



Basic Decision
Symbol

Conditional Output
Symbol

Flow Direction
Symbol

State Assignment

State assignment is the binary coding used to represent the states

Given N states, need at least $\log_2(N)$ FFs to encode the states

(i.e. 3 states, need at least 2 FFs for state information).

S0 = 00, S1 = 01, S2 = 10 (FSM is now a modulo 3 counter)

Do not always have to use the fewest possible number of FFs.
A common encoding is One-Hot encoding - use one FF per state.

S0 = 001, S1 = 010, S2 = 100

State assignment affects speed, gate count of FSM

FSM Implementation

Use DFFs, State assignment: S0 = 00, S1 = 01, S2 = 10

	PS		NS			
CNT	Q1	Q0	Q1	Q0	D1	D0
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	1	0	1	0
0	1	1	x	x	x	x
1	0	0	0	1	0	1
1	0	1	1	0	1	0
1	1	0	0	0	0	0
1	1	1	x	x	x	x

State Table

Equations

D1 = CNT'Q1Q0' + CNT Q1'Q0

D0 = CNT'Q1'Q0 + CNT Q1'Q0'

Minimize Equations

D1

CNT \ Q1Q0	00	01	11	10
0	0	0	x	1
1	0	1	x	0

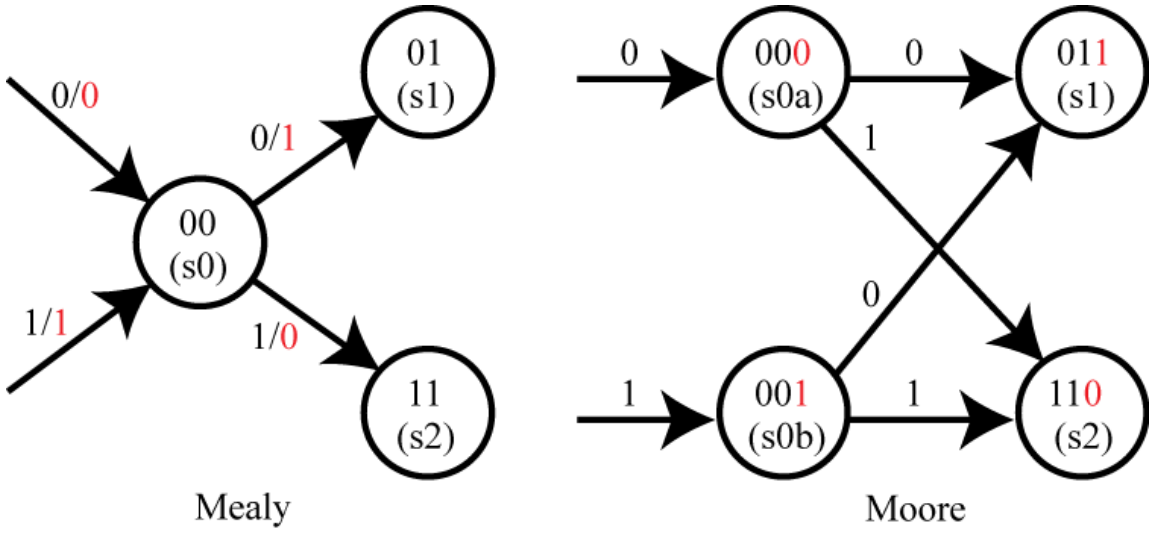
D1 = CNT'Q1 + CNT Q0

D0

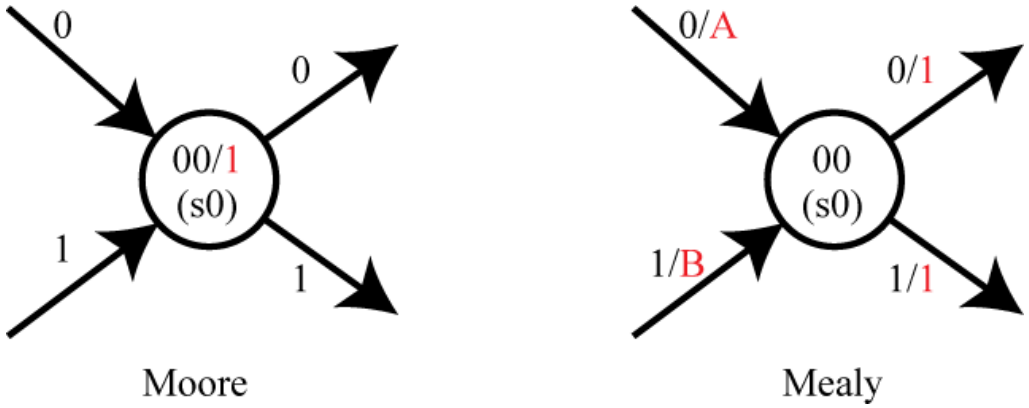
CNT \ Q1Q0	00	01	11	10
0	0	1	x	0
1	1	0	x	0

D0 = CNT'Q0 + CNT Q1'Q0'

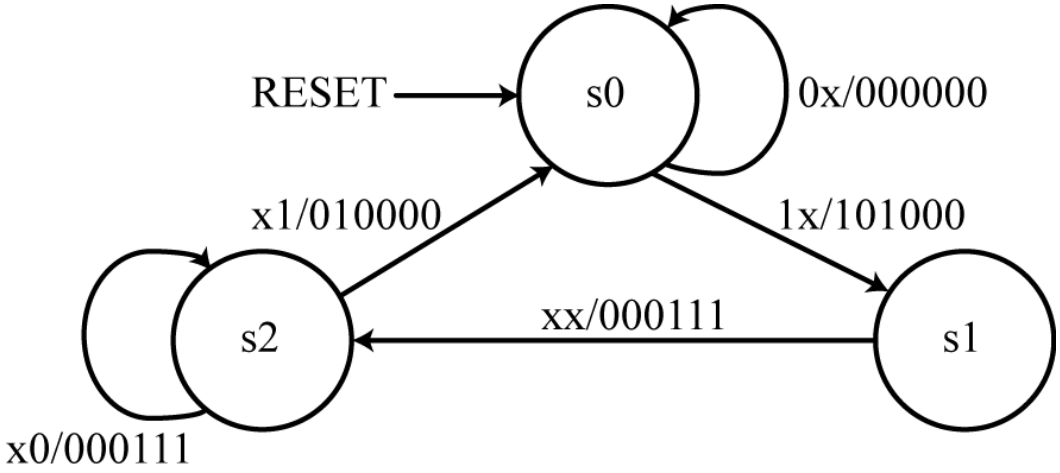
Mealy to Moore Conversion



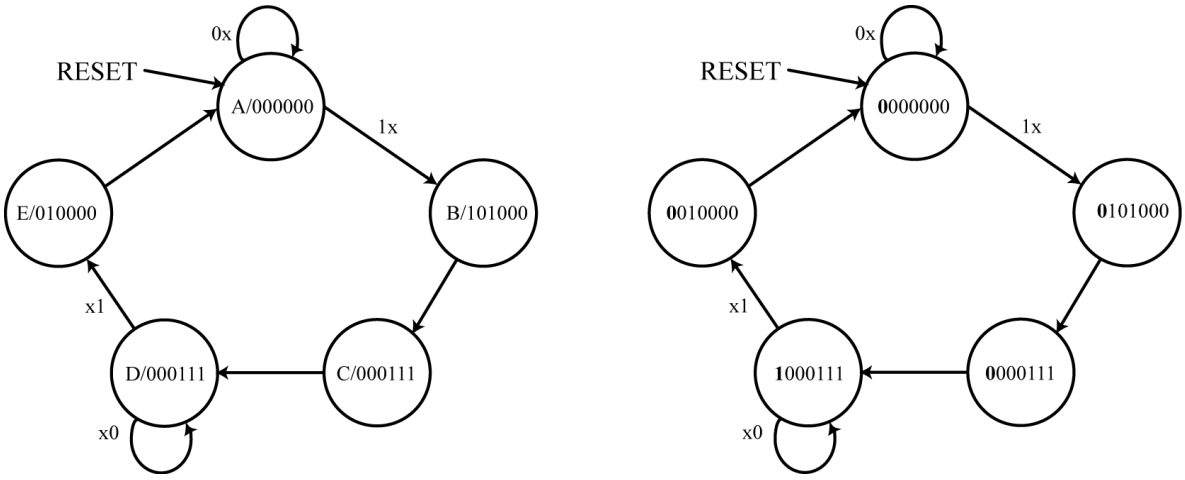
Moore to Mealy Conversion



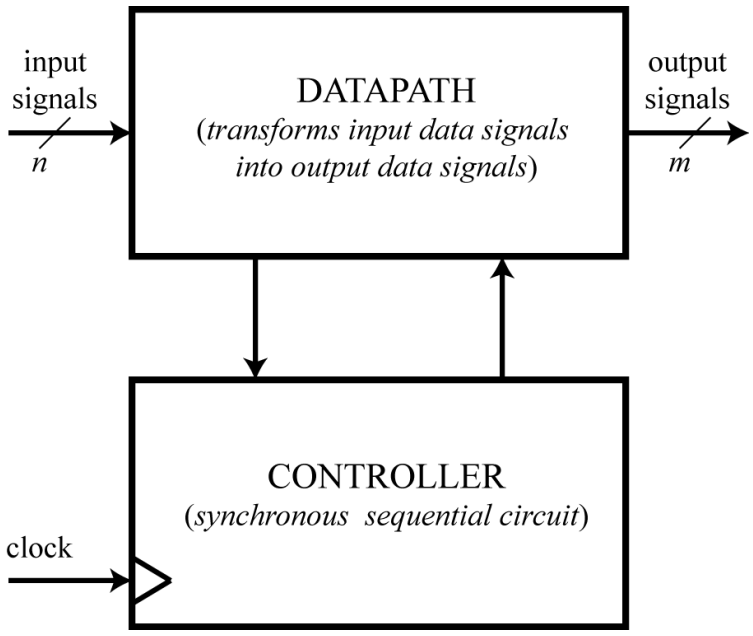
Mealy Model State Diagram



Moore Model State Diagram

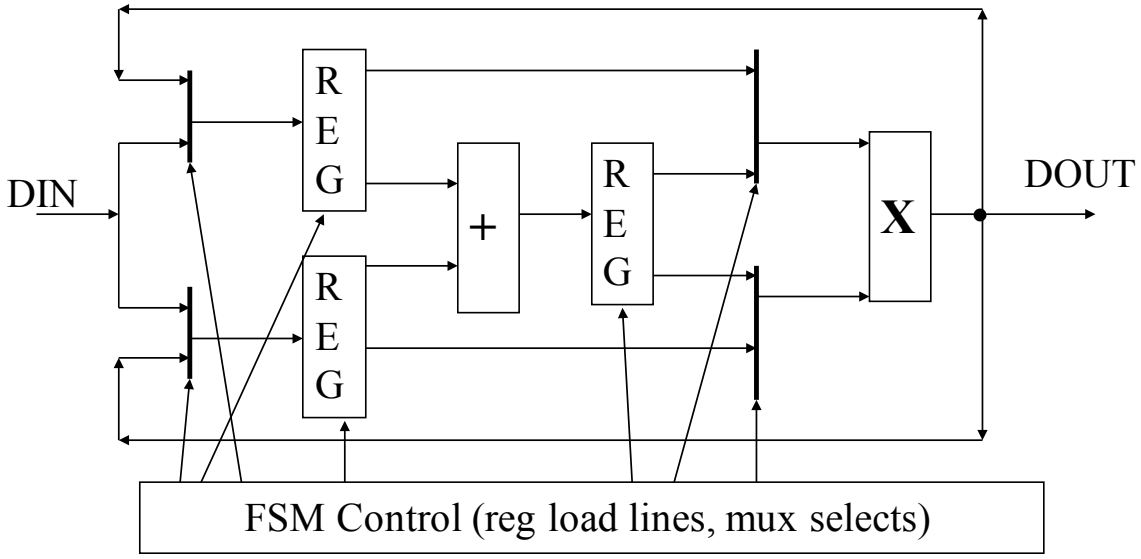


FSMD Block Diagram



FSM Usage as Controller

- Custom counters
- Datapath control



Summary

- We will be describing sequential systems via HDL and ASM charts
  - Use ASM chart for human reader, HDL to allow synthesis of the design
  - Synthesis will perform combinational minimization, but not state reduction.
- Will use common sequential building blocks extensively
  - Registers, Counters, Shift registers, Memories
- Basic storage element will be DFF
- Synchronous (edge-triggered) design methodology