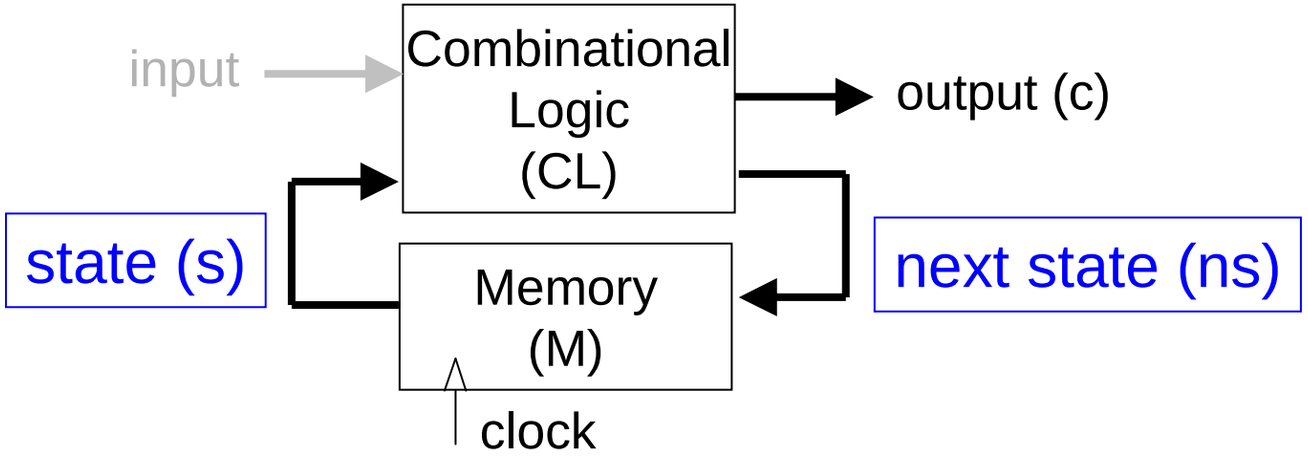
10 Finite State Machine (FSM) II

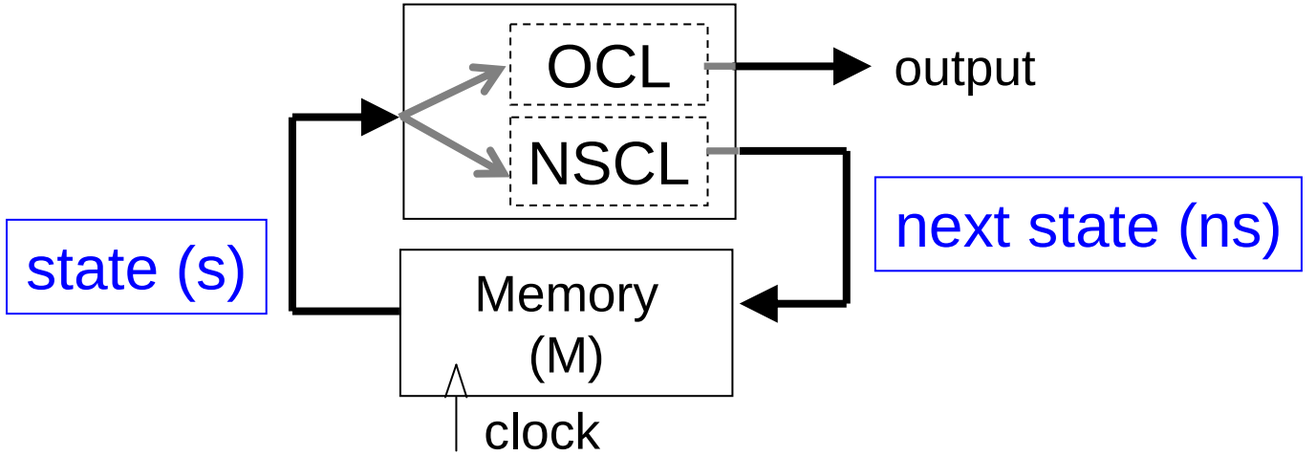
- Inputs & Changes to NSCL and OCL
- Inputs & Changes to STD
    - Example: Stop/Go counter
- Output Models: Moore/Mealy/Mixed
- Unused States and Reset
    - Examples: Vending Machine
    - Sequence Recognizer
- Next State as Outputs
- Shift Registers and SERDES

Previously on lecture set 09:

FSM with no input

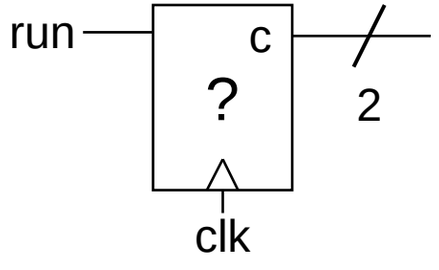


FSM with no input separated into NSCL & OCL



FSM with input example:

Example: - stop/go 0, 1, 2, 3 counter



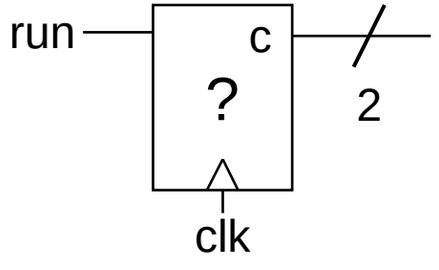
When run = 1, c = 0, 1, 2, 3, 0, 1, 2, 3, …
When run = 0, c holds the previous value

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1	2	3	0	1	2	3	3	3	3	3	0	0	1	2	3	3

input controls behavior of FSM

FSM with input example:

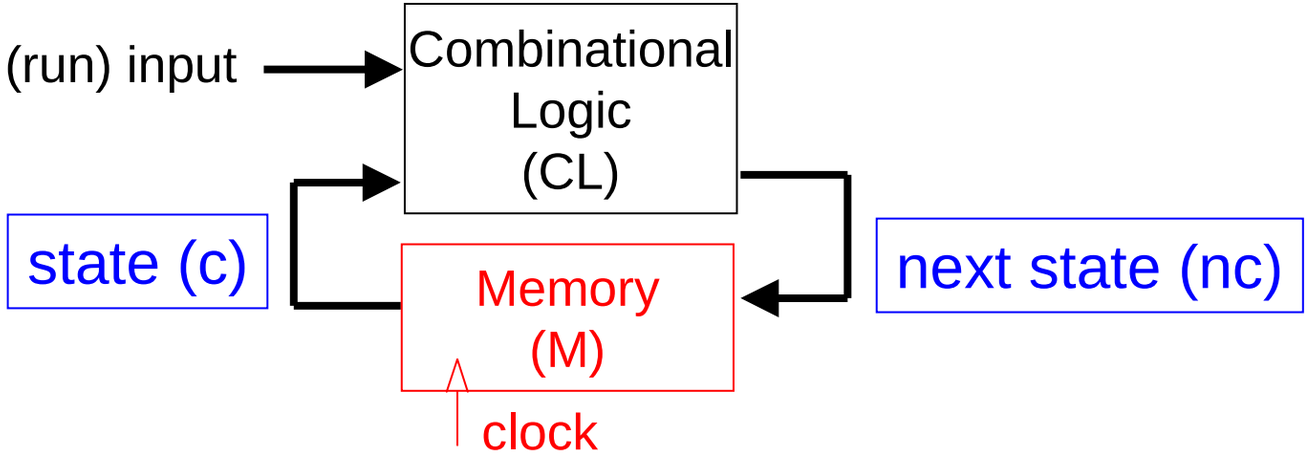
Example: - stop/go 0, 1, 2, 3 counter



When run = 1, c = 0, 1, 2, 3, 0, 1, 2, 3, …
When run = 0, c holds the previous value

State = ?
How many states?
How many flops?

FSM example: stop/go counter c: 0, 1, 2, 3, …

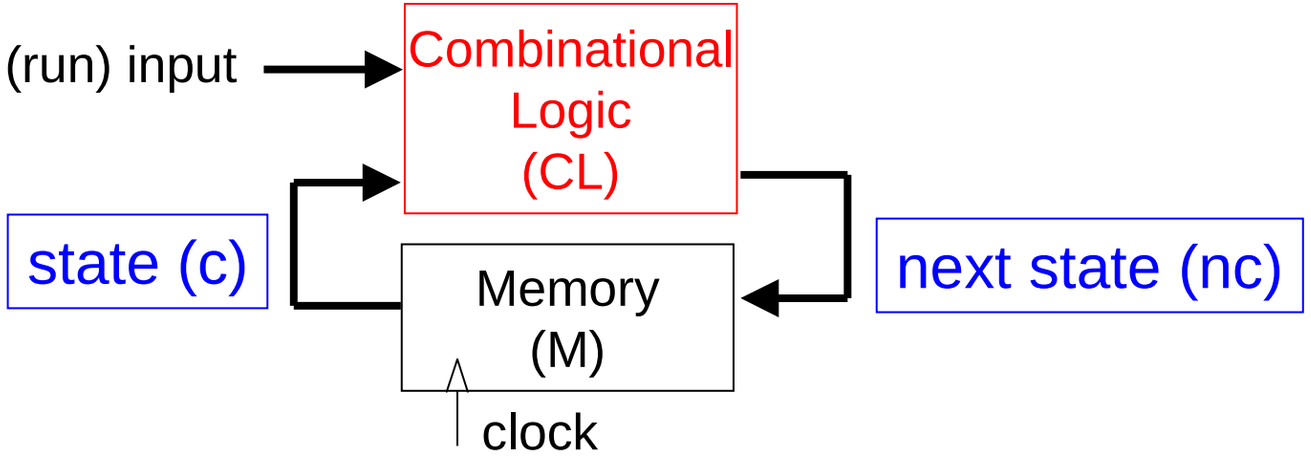


What does Mike do?

store nc that CLaire gives in this clock cycle, and give it back to CLaire next clock cycle, calling it c

Mike works in the same way as before

FSM example: stop/go counter c: 0, 1, 2, 3, …

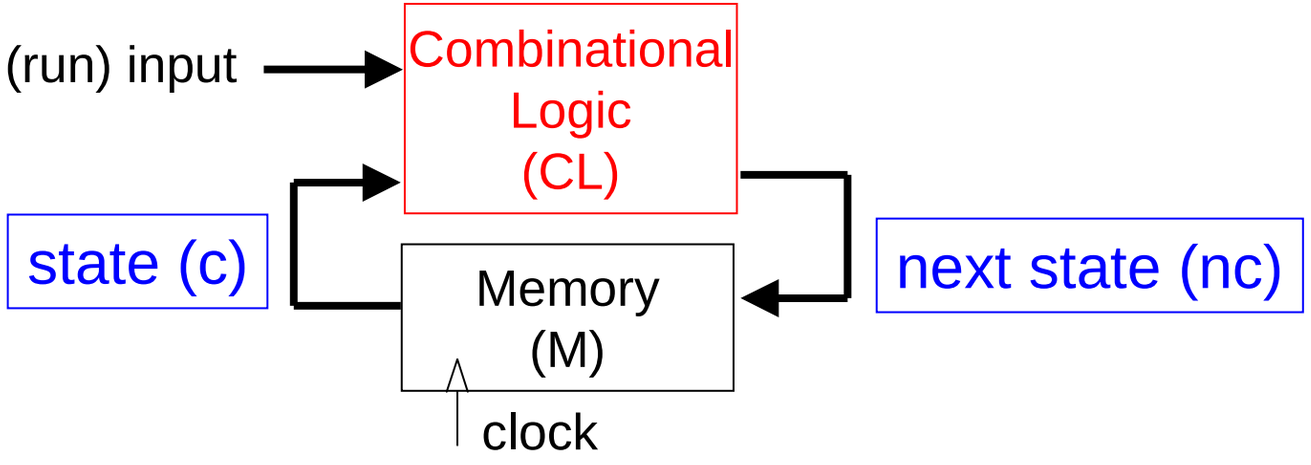


What does CLaire do?

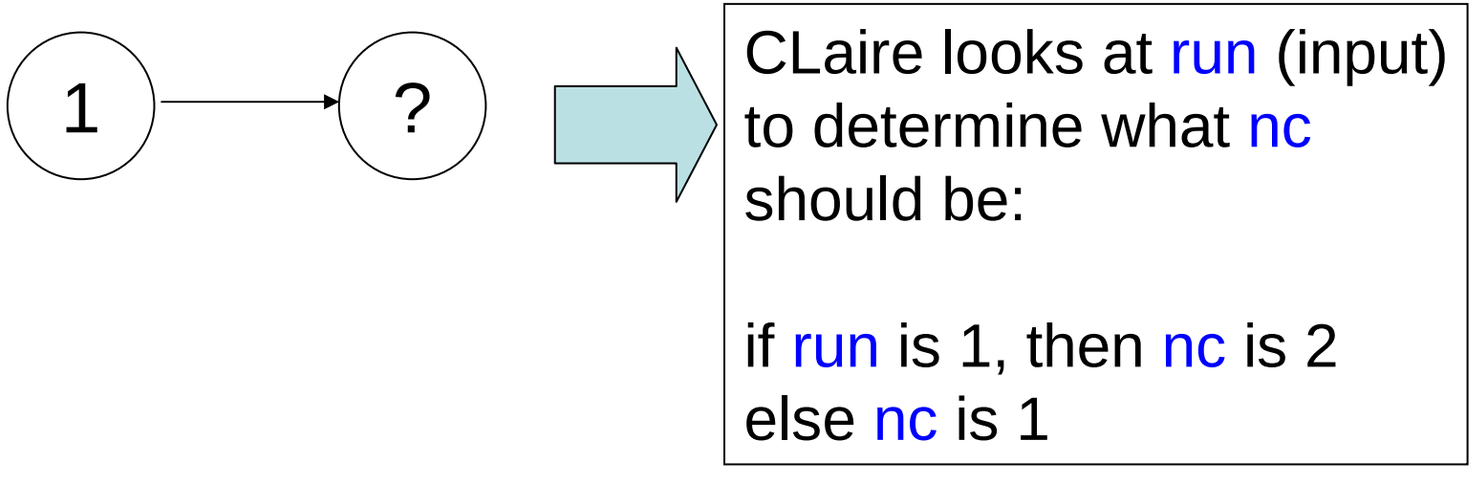
Look at c AND run (input) and figure out what nc should be

CLaire needs to look at to look at both input and state to determine next state

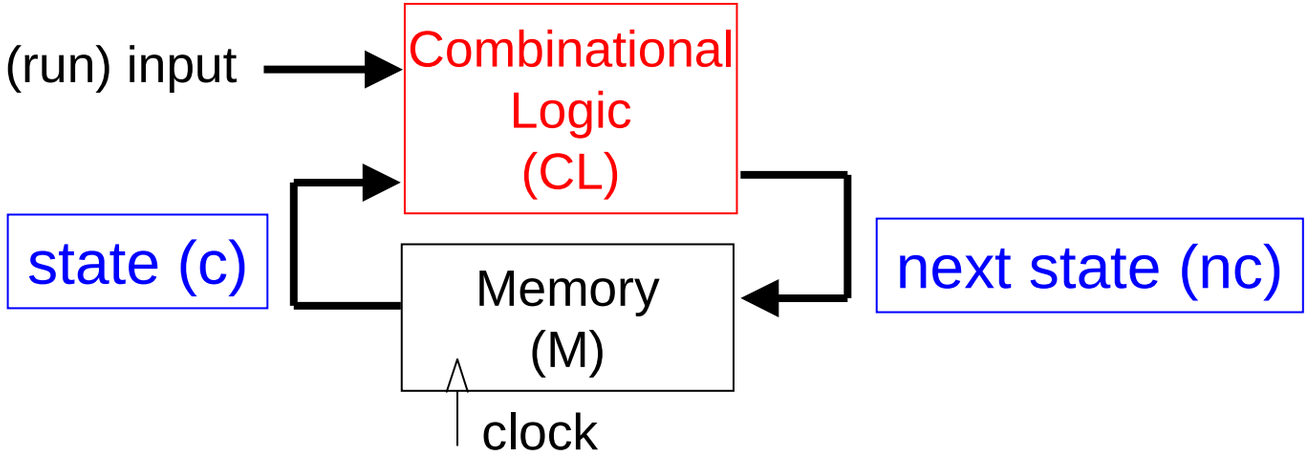
FSM example: stop/go counter c: 0, 1, 2, 3, …



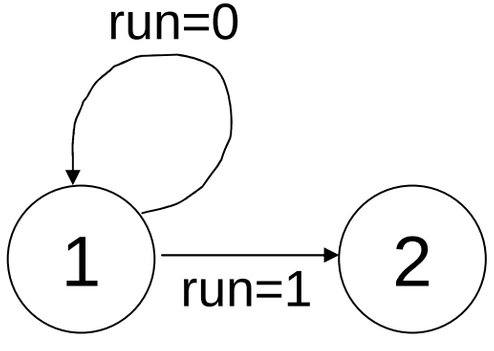
CLaire sees c=1, CLaire needs to write nc



FSM example: stop/go counter c: 0, 1, 2, 3, …



CLaire sees c=1, CLaire needs to write nc

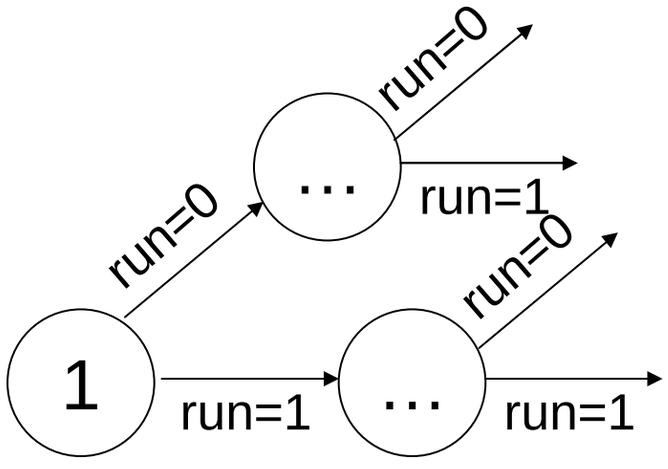


Input creates more arcs

1-bit input = 2 possibilities
= 2 arcs coming out of each state

n-bit input → 2^n arcs out from each state

FSM example:

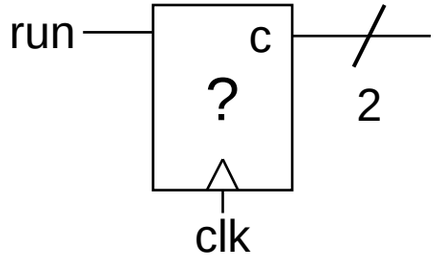


If the system has 1-bit input, then
every state must have 2 arcs coming out.
Arcs going in do not matter.

For system with n-bit input, every state must have 2^n arcs coming out.

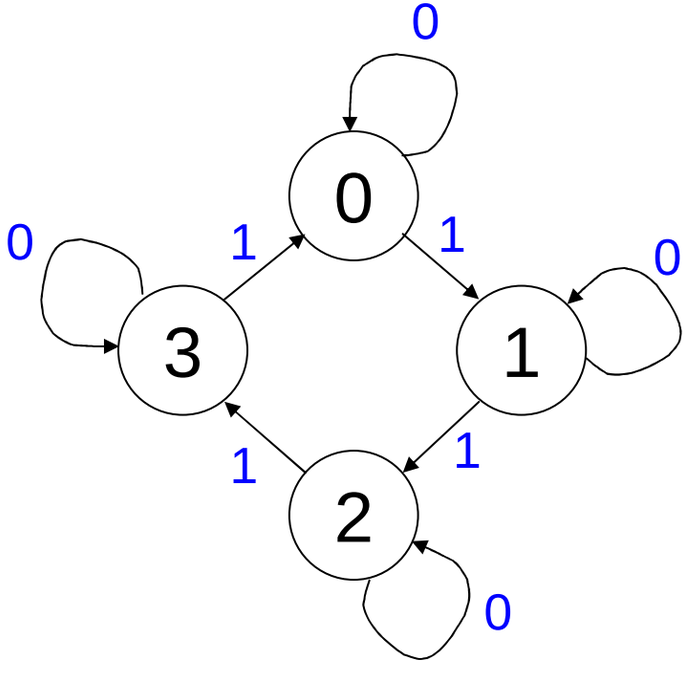
FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



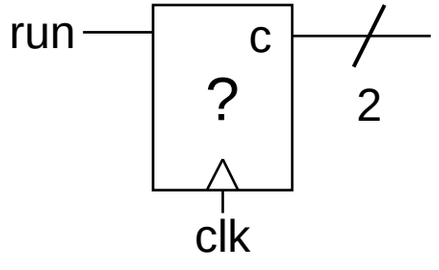
When run = 1, c = 0, 1, 2, 3, 0, 1, 2, 3, …
When run = 0, c holds the previous value

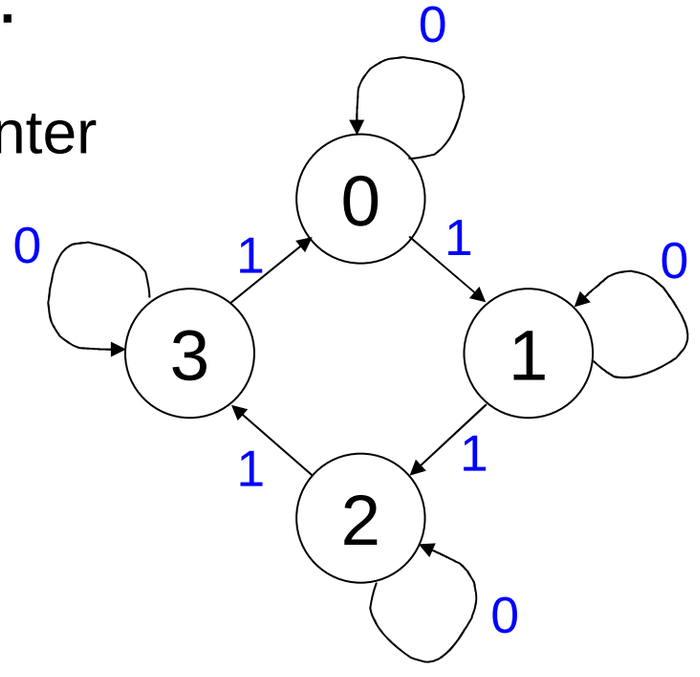
STD of stop/go counter



FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



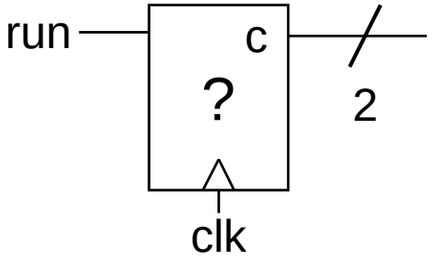


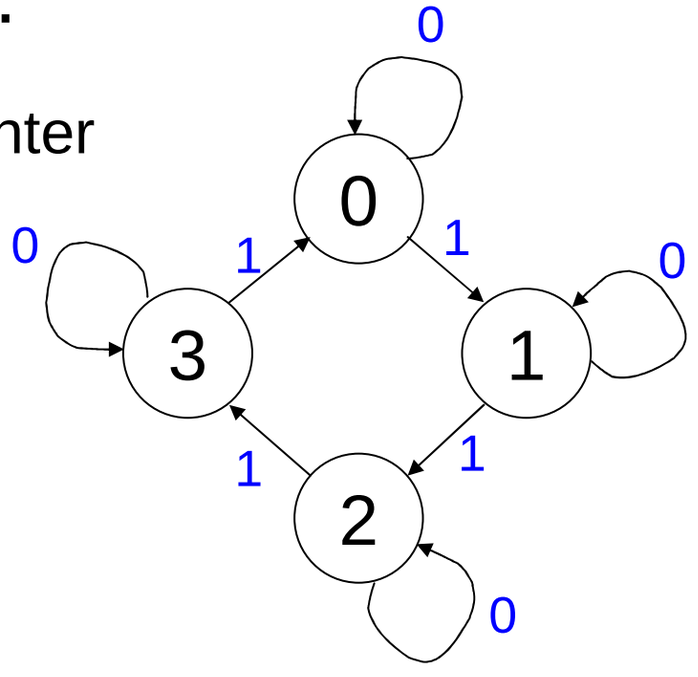
let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0																	

FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



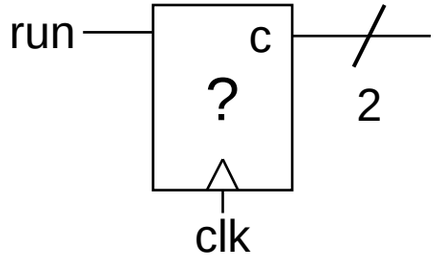


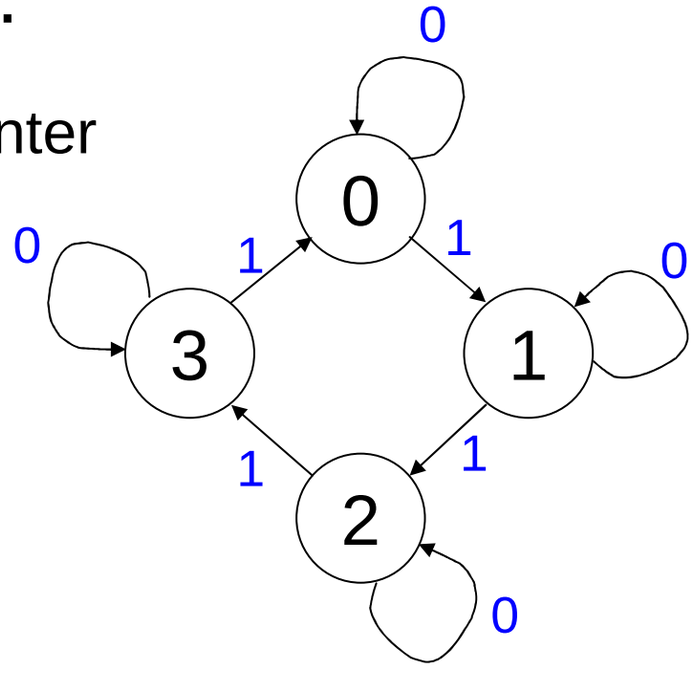
let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	?																

FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



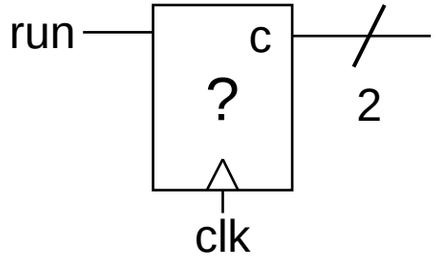


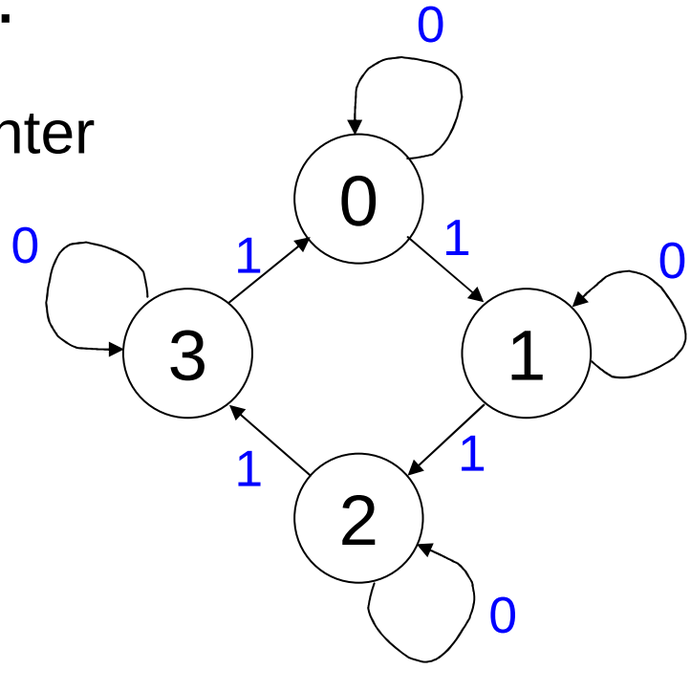
let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1																

FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



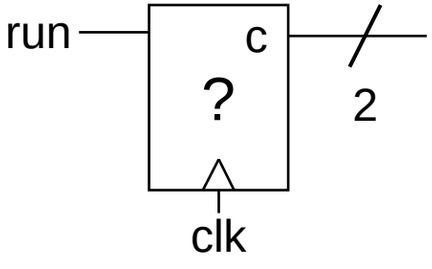


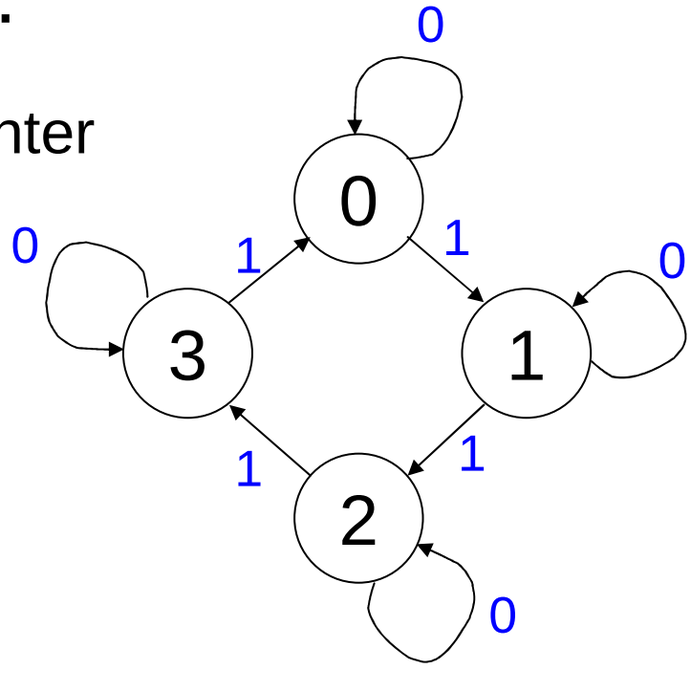
let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1	2															

FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



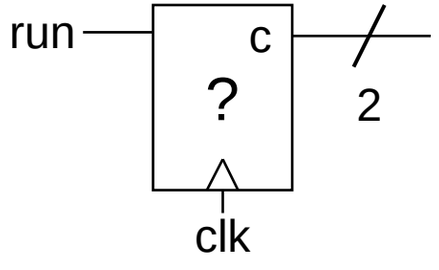


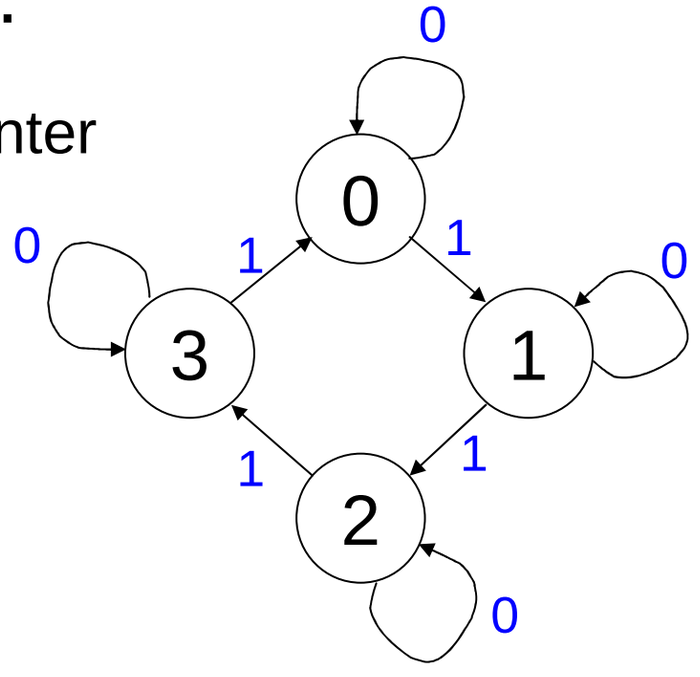
let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1	2	3	0	1	2											

FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



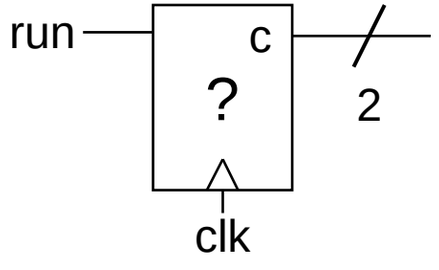


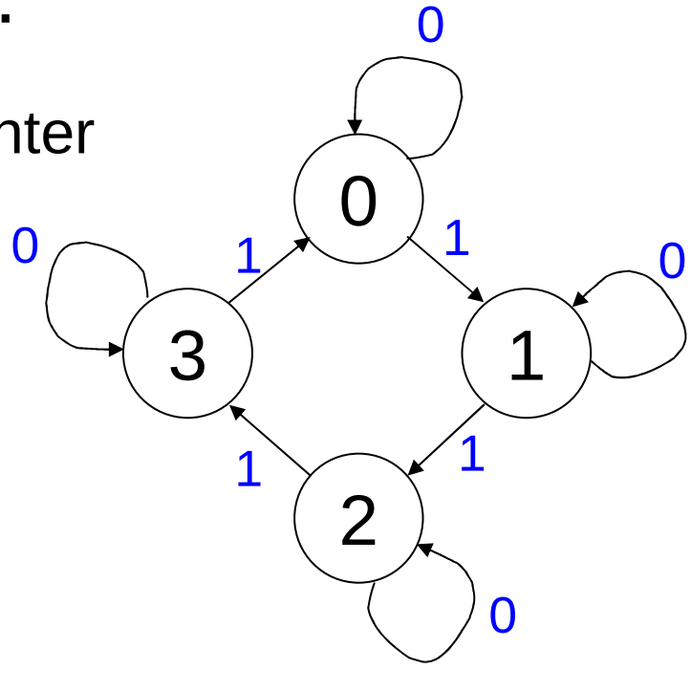
let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1	2	3	0	1	2	3										

FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



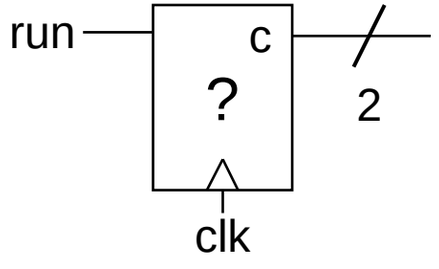


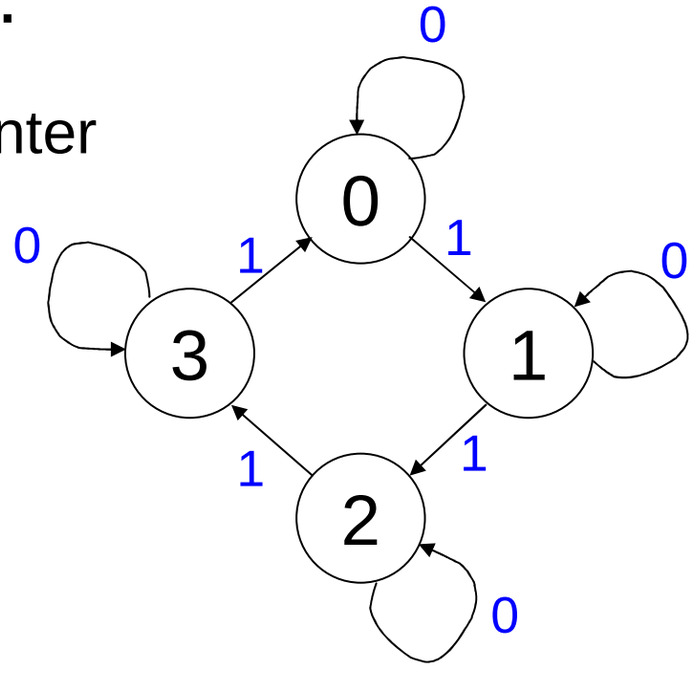
let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1	2	3	0	1	2	3	3	3	3							

FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



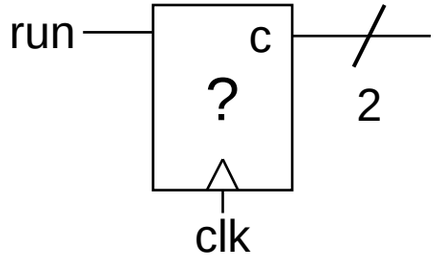


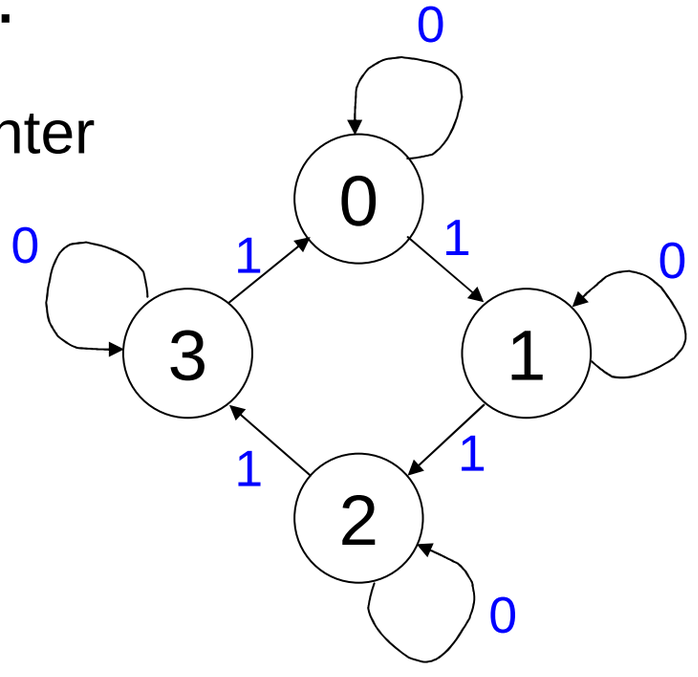
let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1	2	3	0	1	2	3	3	3	3	3						

FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter



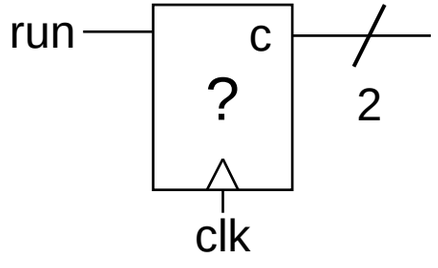


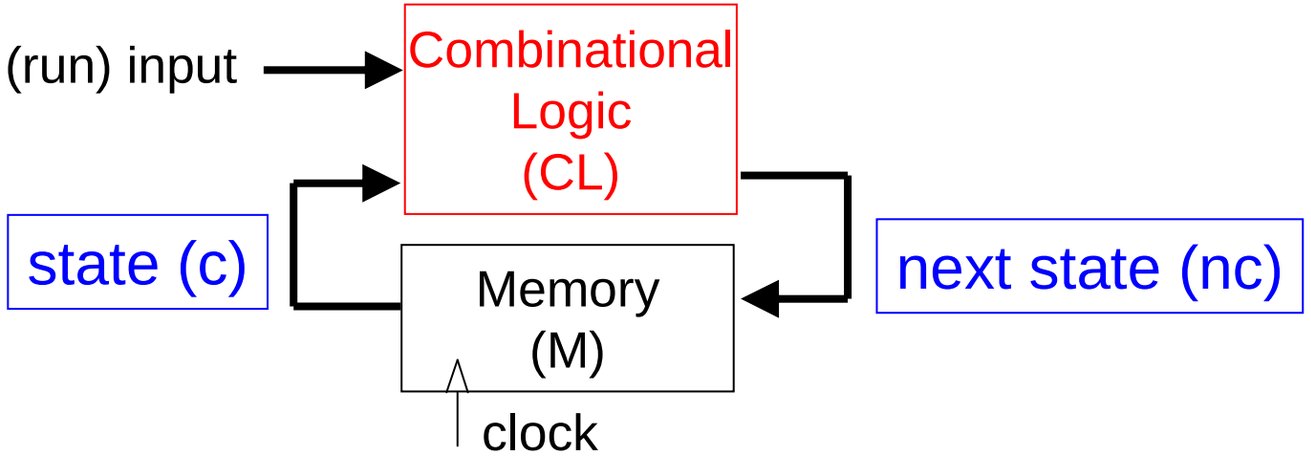
let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1	2	3	0	1	2	3	3	3	3	3	0	0	1	2	3	3

FSM with inputs example:

Example: - stop/go 0, 1, 2, 3 counter





NSCL: next_state = f(input, state)

FSM with inputs (summary):

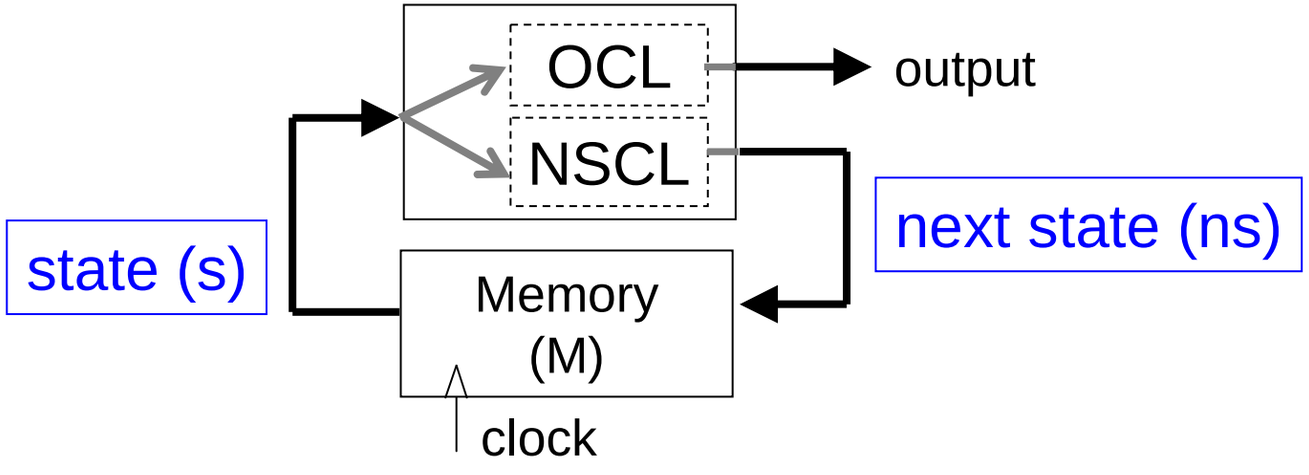
- inputs mean more arcs

- n-bit input → 2^n arcs out from each state
- no (0) input → 1 arc out from each state

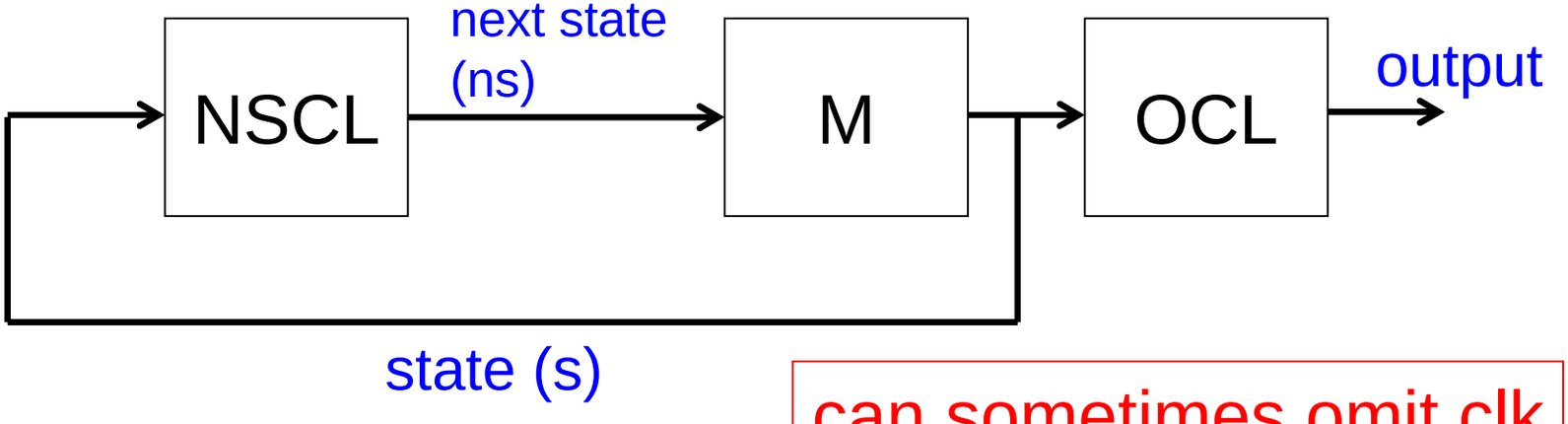
- NSCL is

$$ns = f(in, s)$$

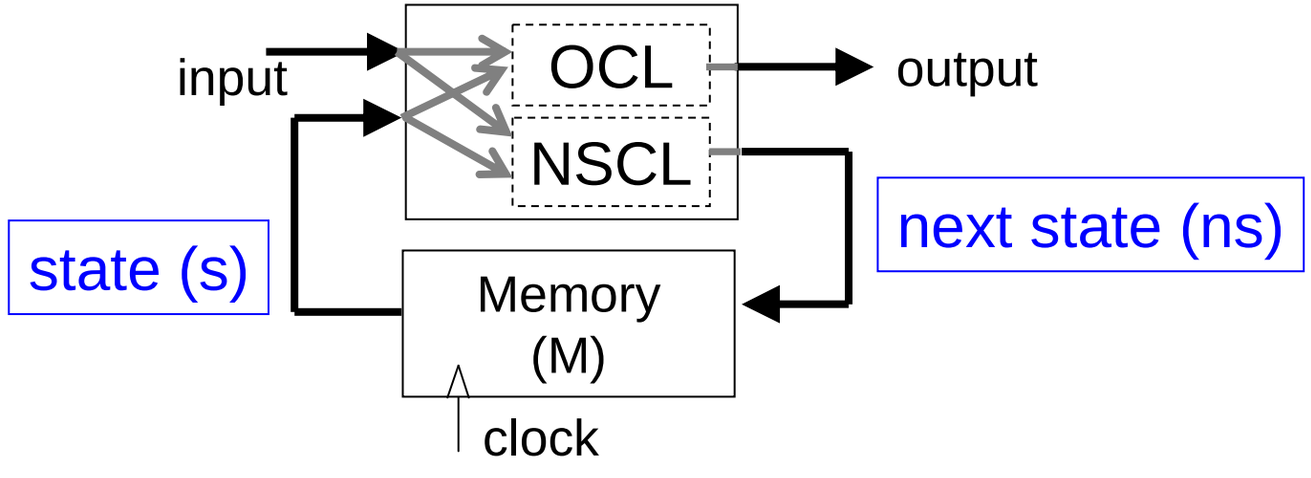
Output Models:



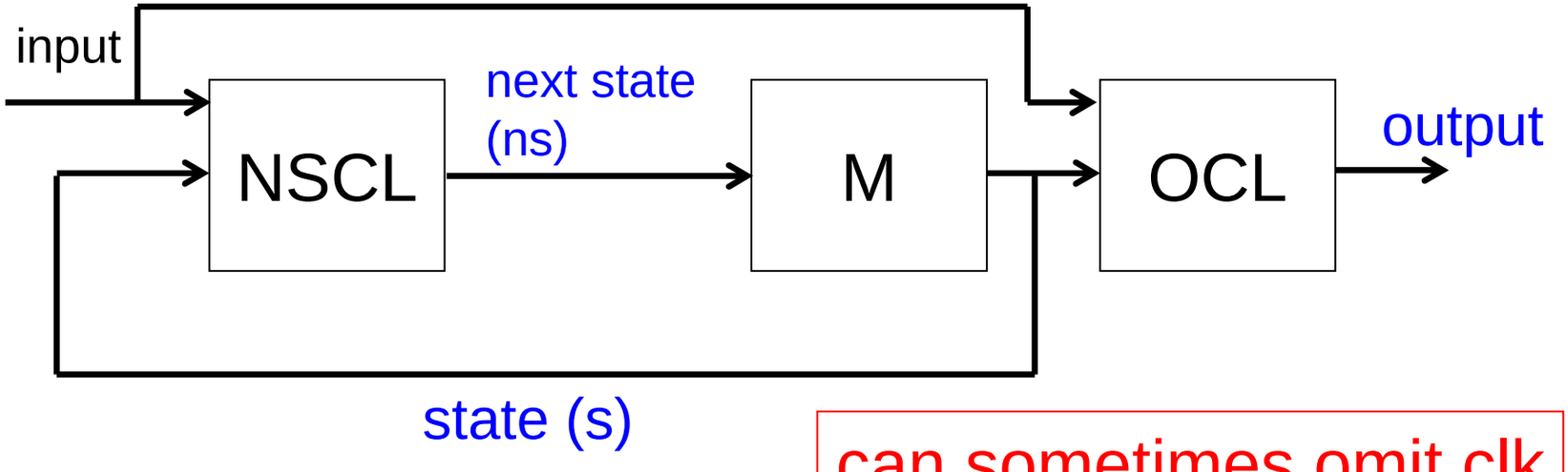
Rewrite .



Output Models:

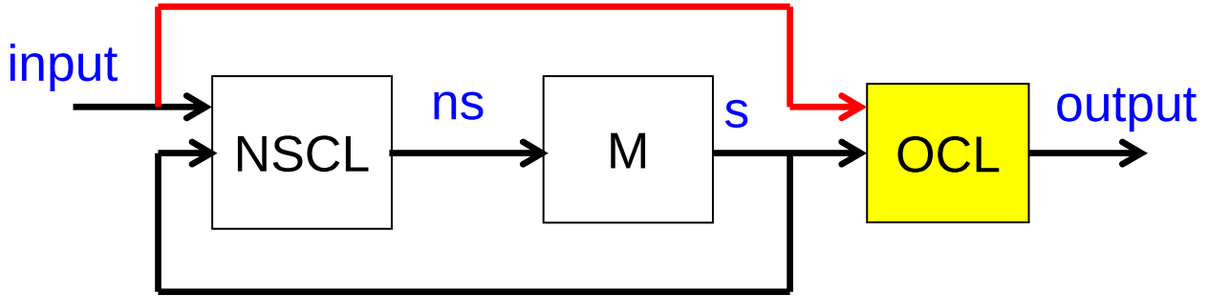


Rewrite .

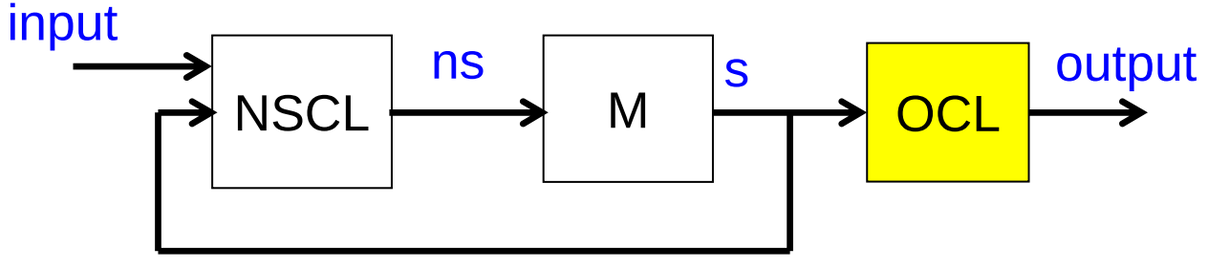


can sometimes omit clk

Output Models:



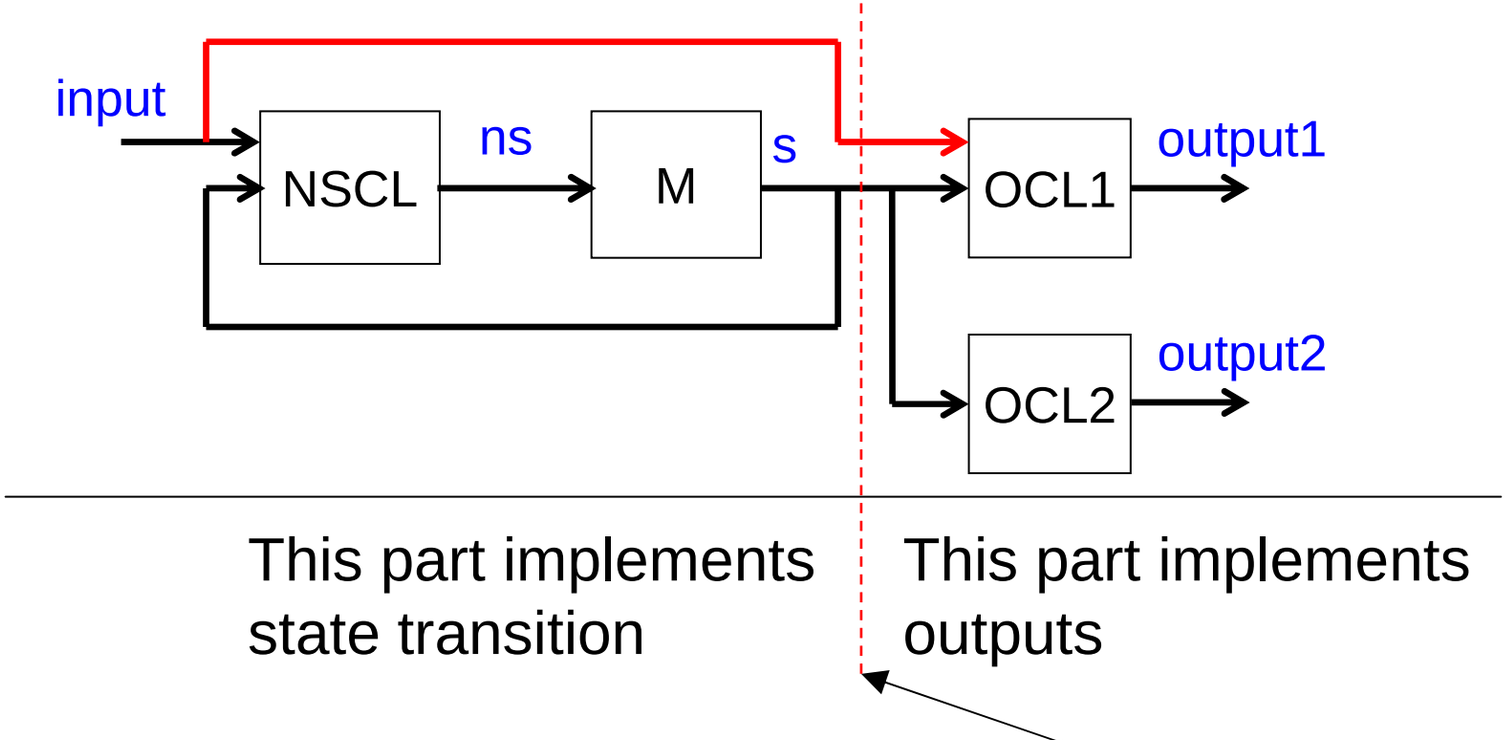
MEALY OUTPUT MODEL: out = f(in, s)



MOORE OUTPUT MODEL: out = f(s)

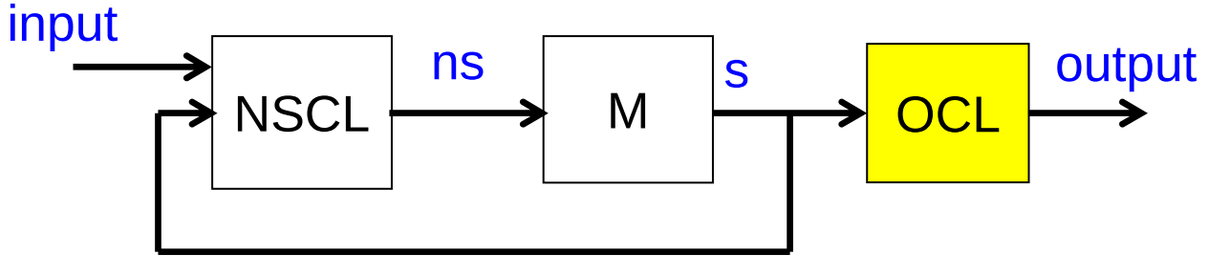
Output Models:

Mixing many output types is possible.
FSM does not need to be all Moore or all Mealy.



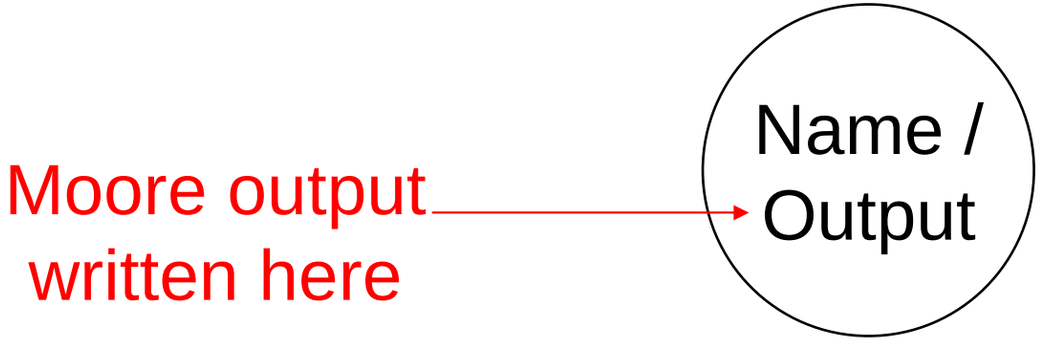
Once information passes through dotted line, it does not come back.

Outputs on STDs:

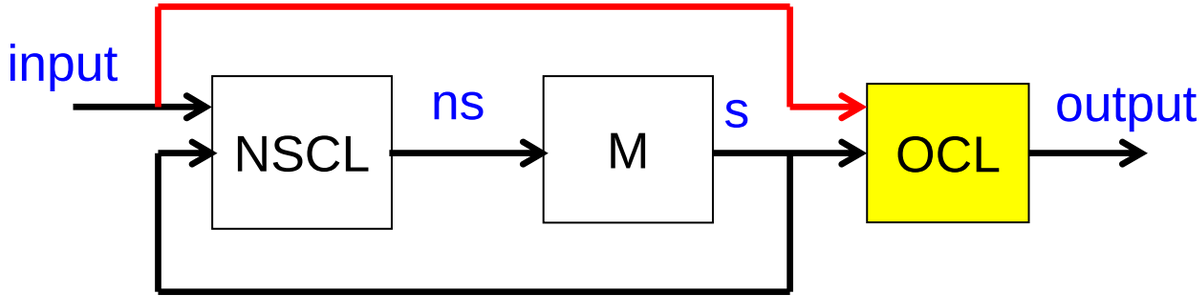


MOORE OUTPUT MODEL: out = f(s)

- know the state, know the output
- so output is written together with state
- output will lag input by at least 1 clock cycle

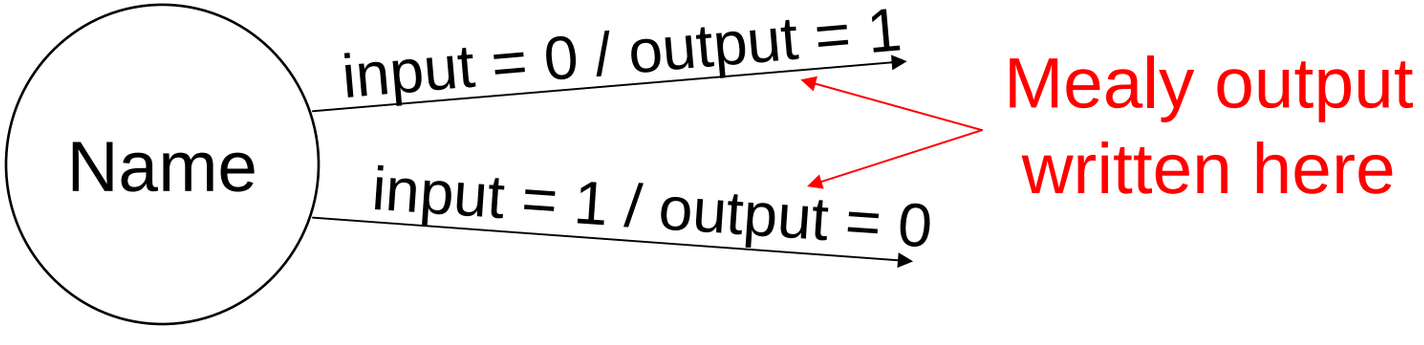


Outputs on STDs:



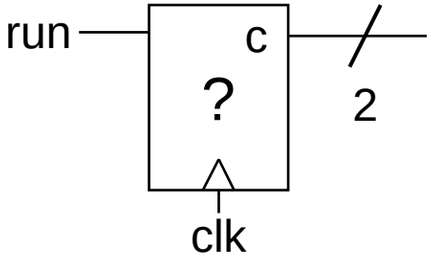
MEALY OUTPUT MODEL: out = f(in, s)

- must know both state and input to know output
- so output is written together with arc
- output can respond to input within same clock cycle



Mealy FSM example:

Example: - **Mealy** stop/go 0, 1, 2, 3 counter



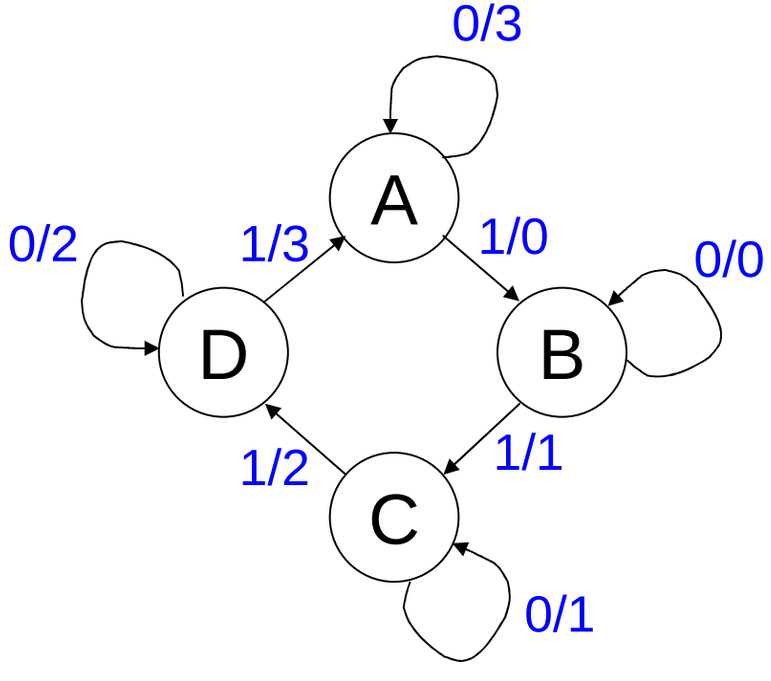
When run = 1, c = 0, 1, 2, 3, 0, 1, 2, 3, …
When run = 0, c holds the previous value

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1	2	3	0	1	2	2	2	2	2	3	3	0	1	2	2	2

Mealy FSM example:

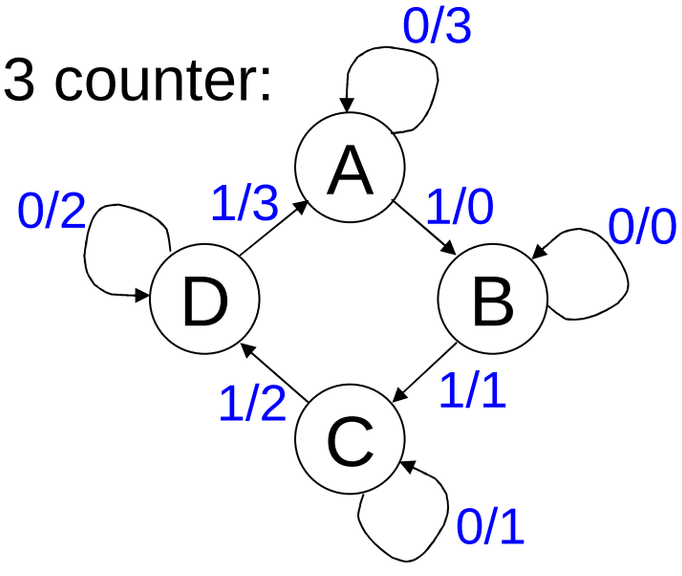
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:

- States are no longer the outputs
- Can stop and start immediately (Mealy output)



Mealy FSM example:

Example: - **Mealy** stop/go 0, 1, 2, 3 counter:

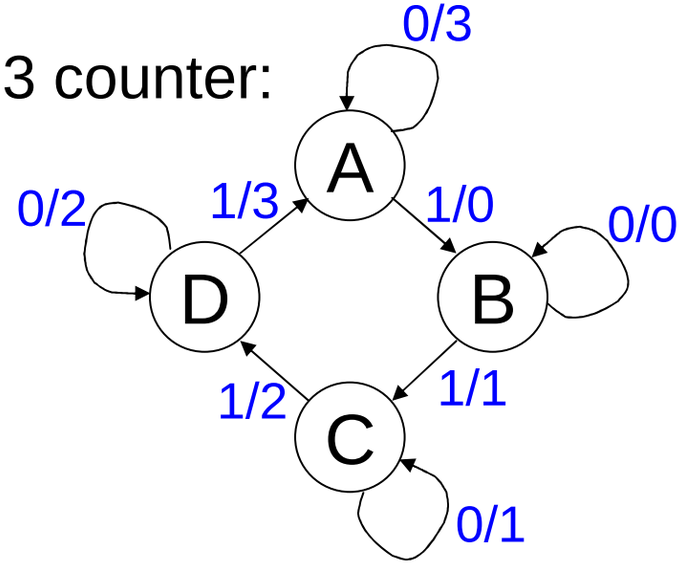


let us simulate

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A																	
output																		

Mealy FSM example:

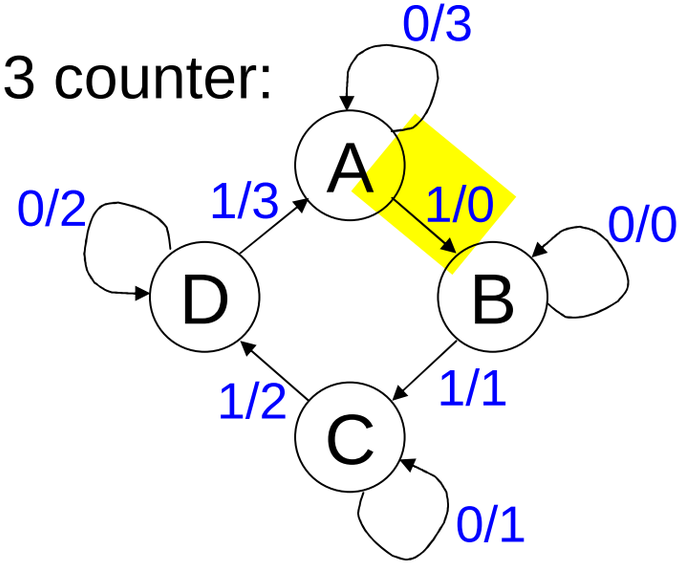
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A																	
output	?																	

Mealy FSM example:

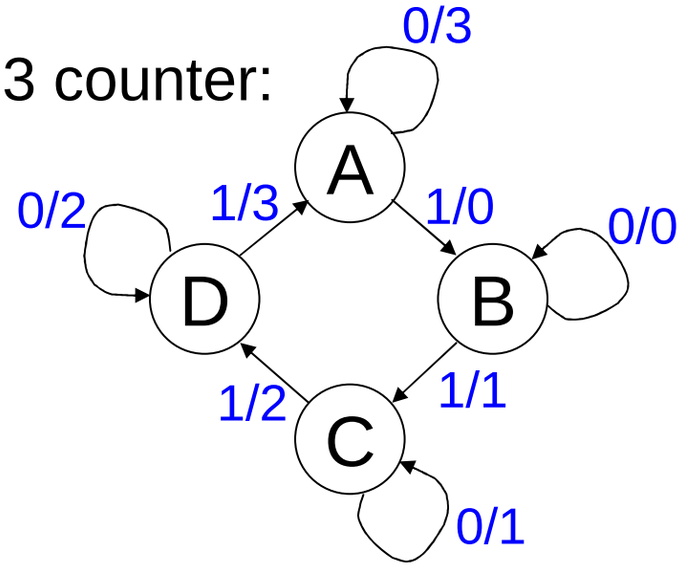
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A																	
output	0																	

Mealy FSM example:

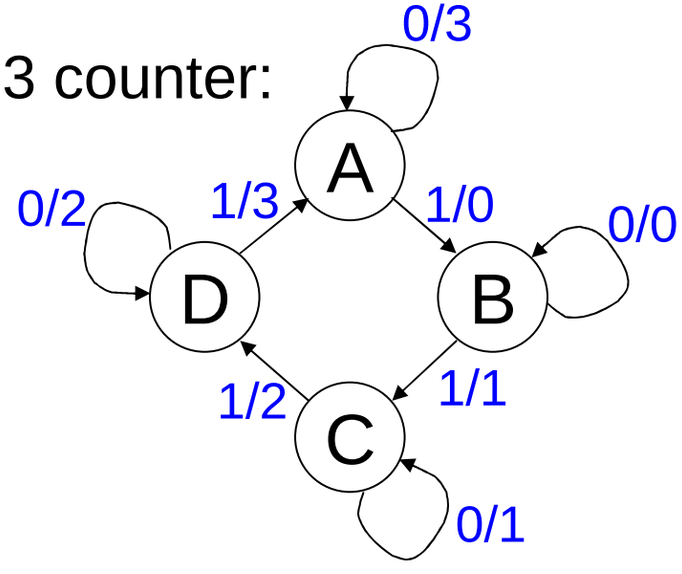
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B																
output	0																	

Mealy FSM example:

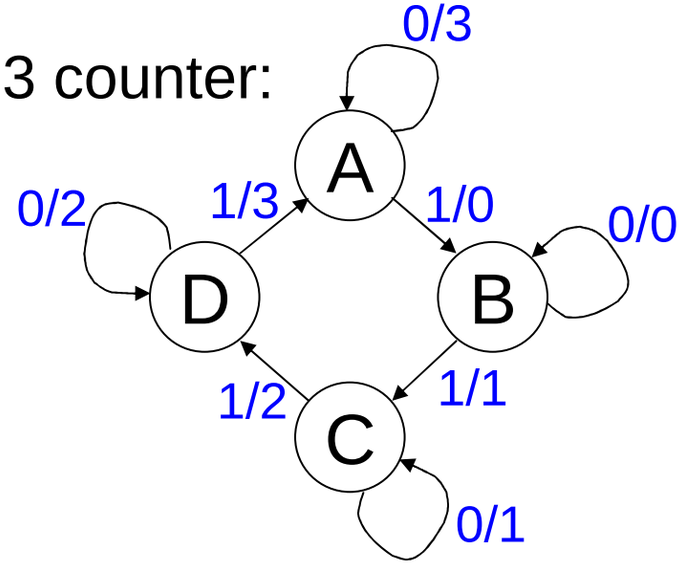
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B																
output	0	1																

Mealy FSM example:

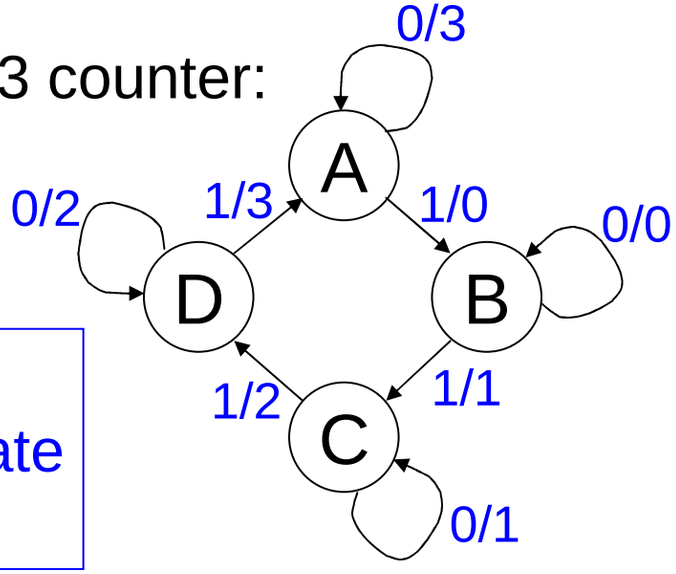
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C															
output	0	1																

Mealy FSM example:

Example: - **Mealy** stop/go 0, 1, 2, 3 counter:

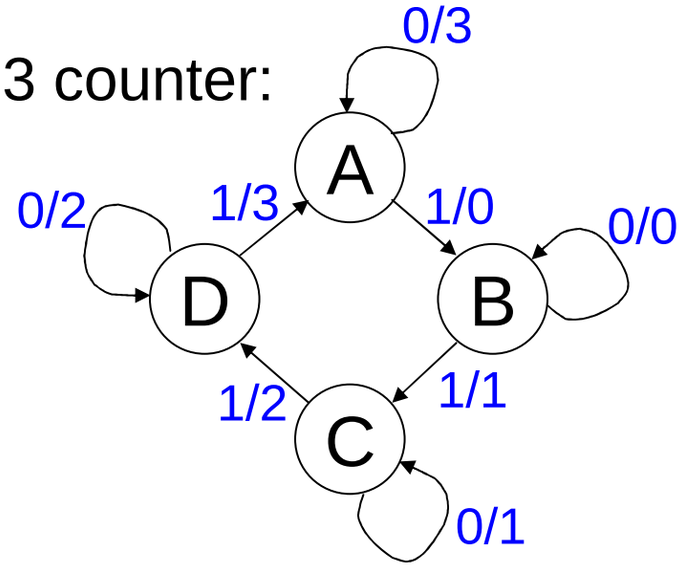


observation:
need only input & state for next state
let's finish up state first

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C															
output	0	1	2															

Mealy FSM example:

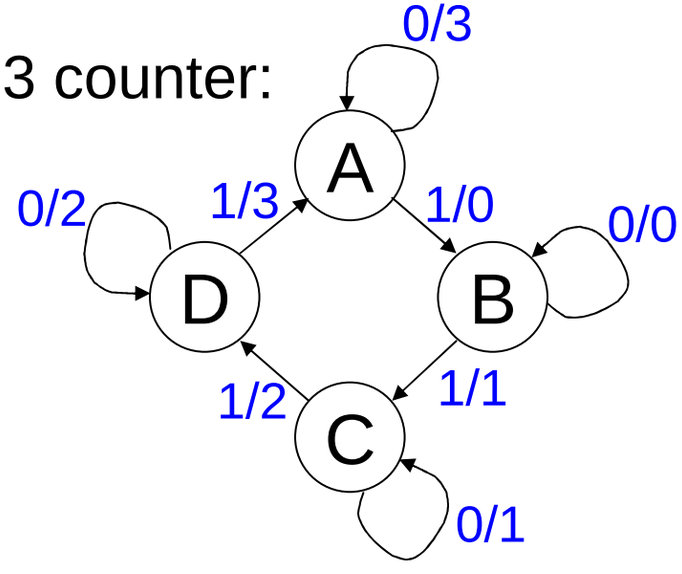
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D														
output	0	1	2															

Mealy FSM example:

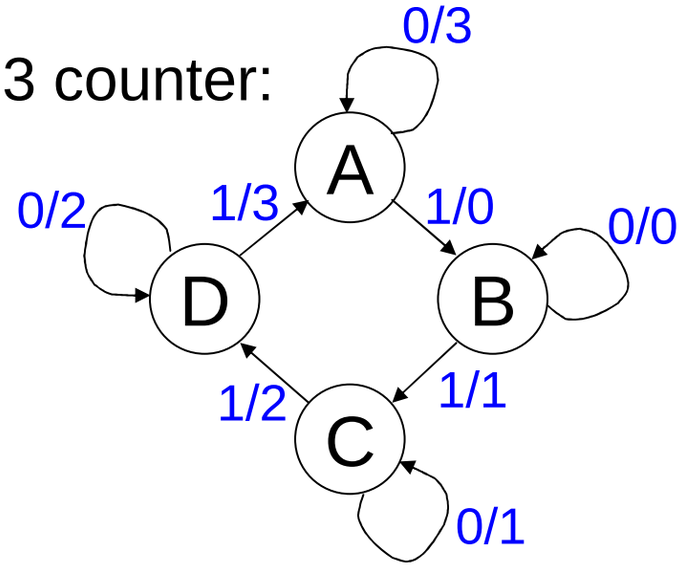
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A													
output	0	1	2															

Mealy FSM example:

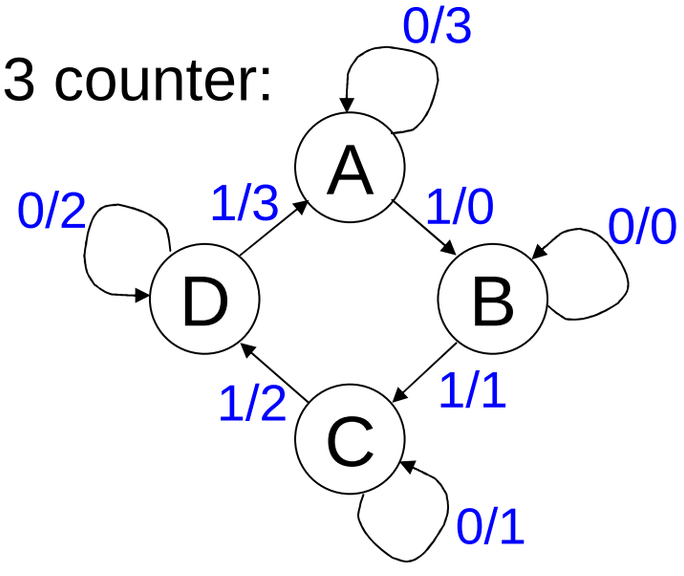
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B												
output	0	1	2															

Mealy FSM example:

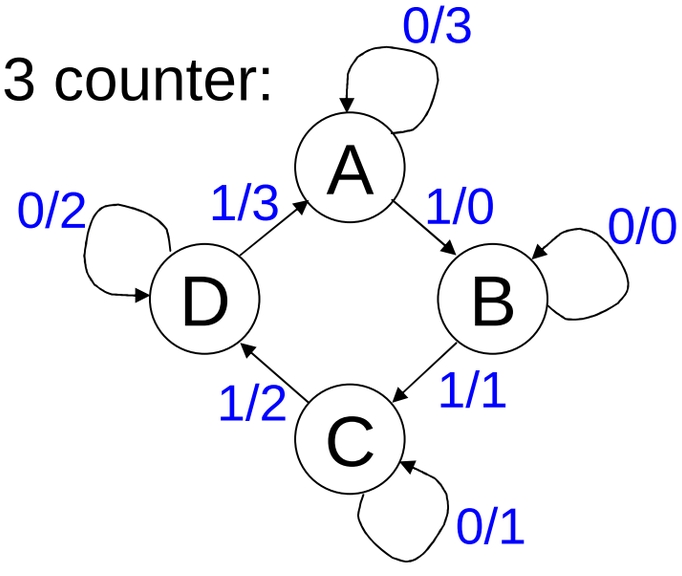
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C											
output	0	1	2															

Mealy FSM example:

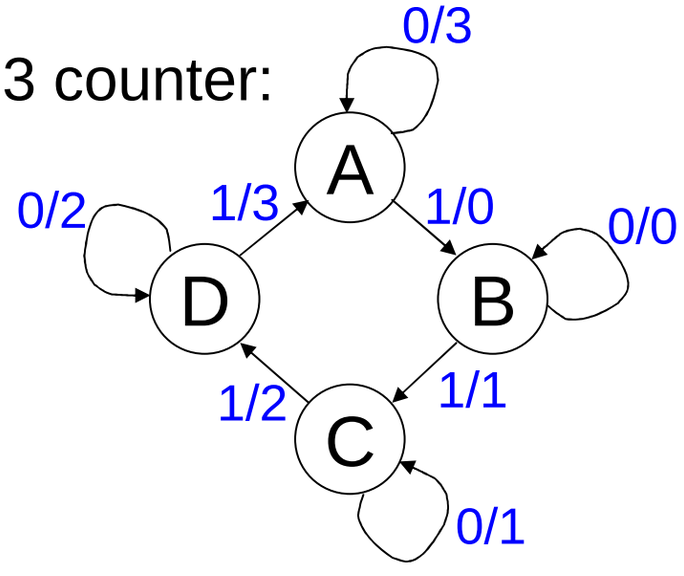
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D										
output	0	1	2															

Mealy FSM example:

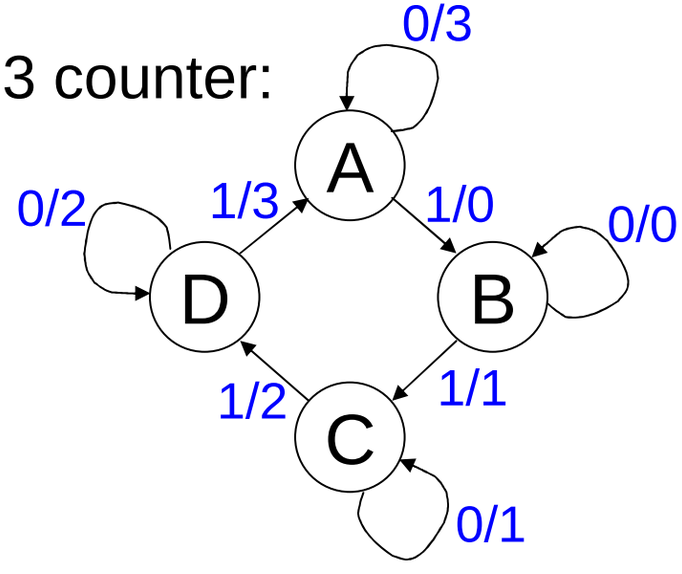
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D									
output	0	1	2															

Mealy FSM example:

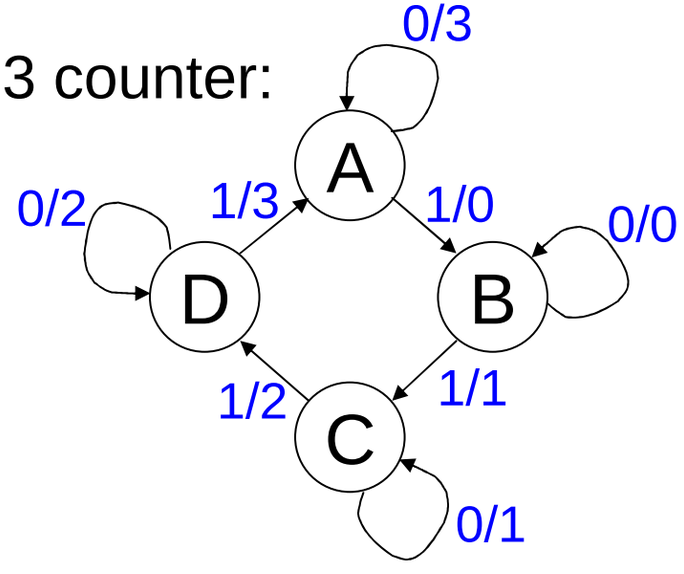
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D								
output	0	1	2															

Mealy FSM example:

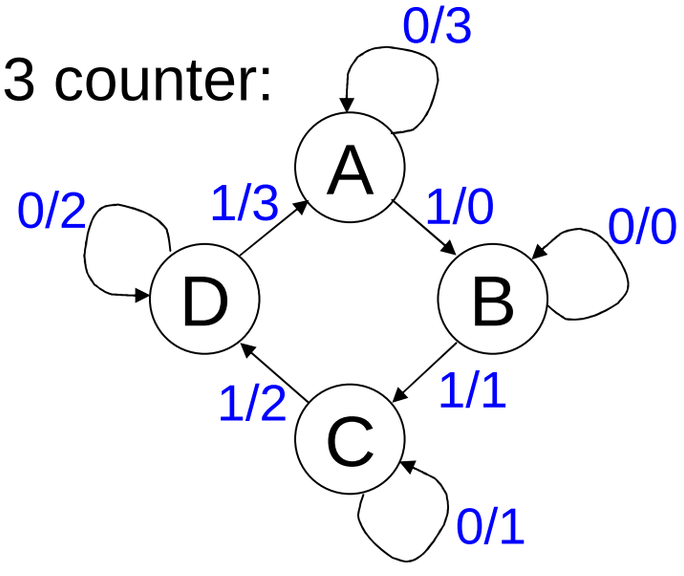
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D							
output	0	1	2															

Mealy FSM example:

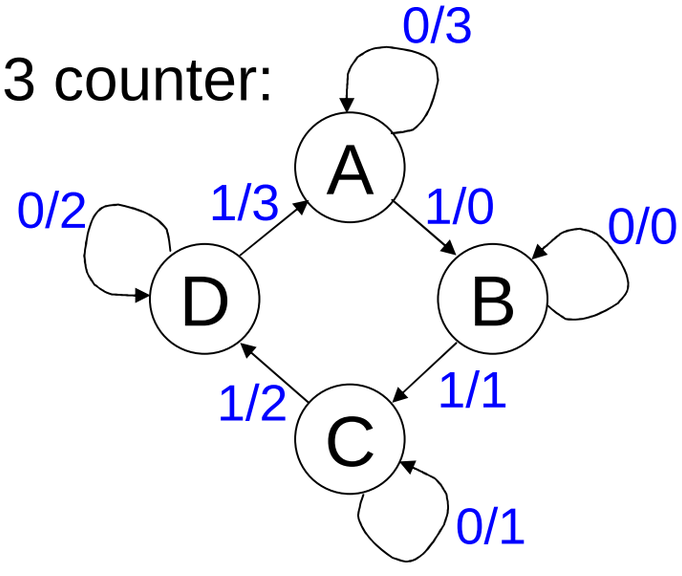
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D						
output	0	1	2															

Mealy FSM example:

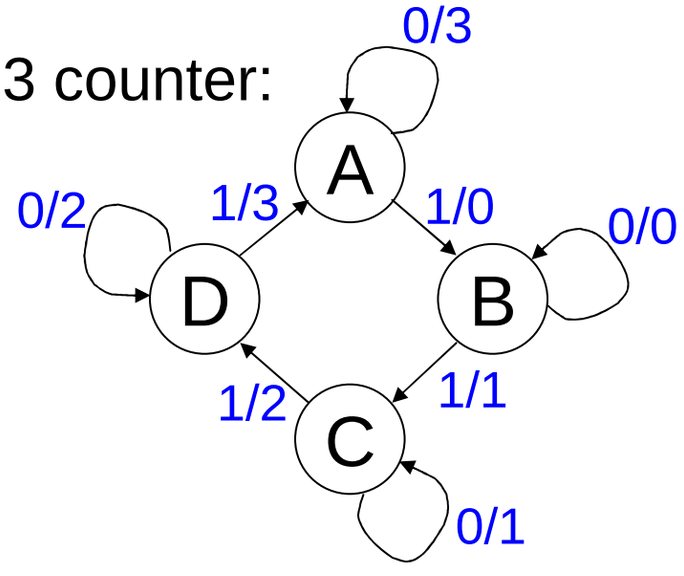
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A					
output	0	1	2															

Mealy FSM example:

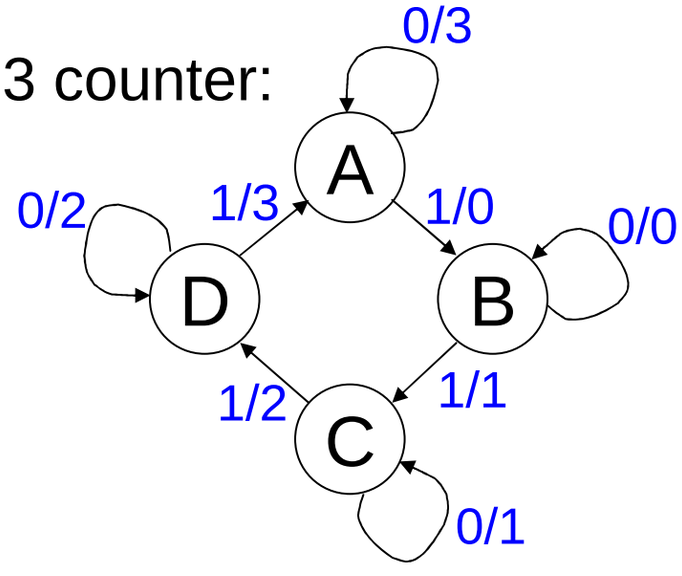
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A				
output	0	1	2															

Mealy FSM example:

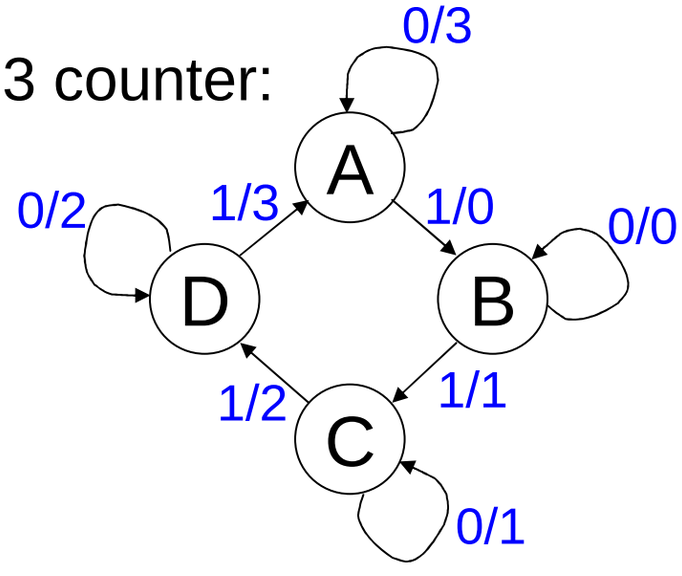
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B			
output	0	1	2															

Mealy FSM example:

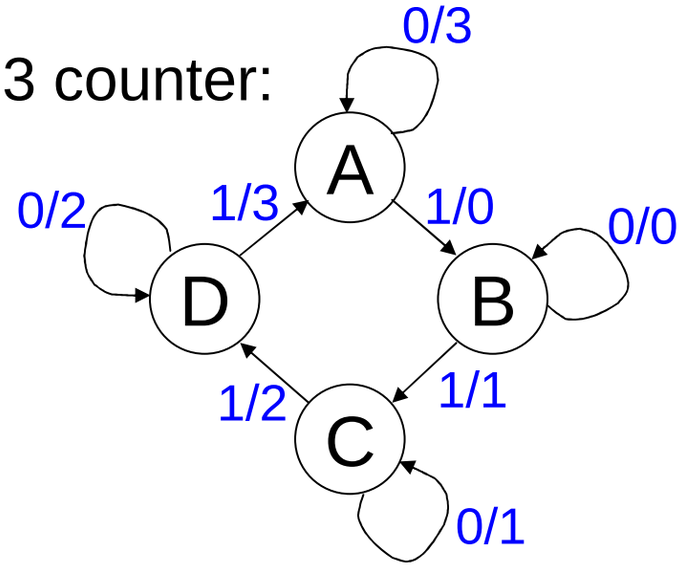
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C		
output	0	1	2															

Mealy FSM example:

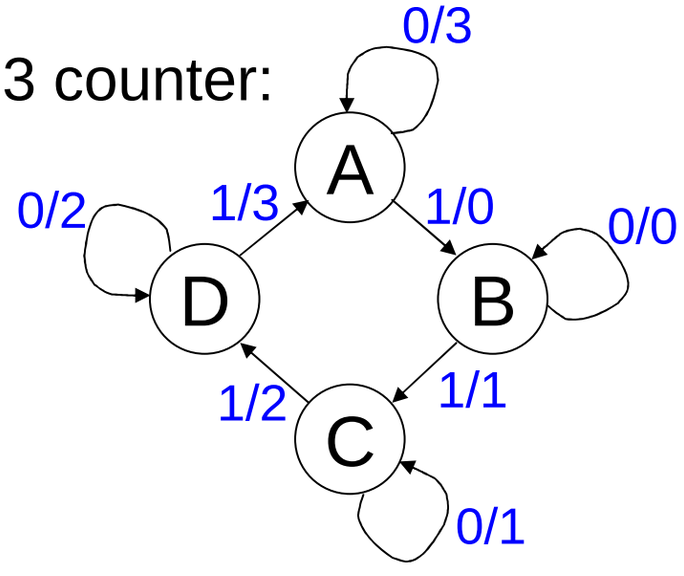
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	
output	0	1	2															

Mealy FSM example:

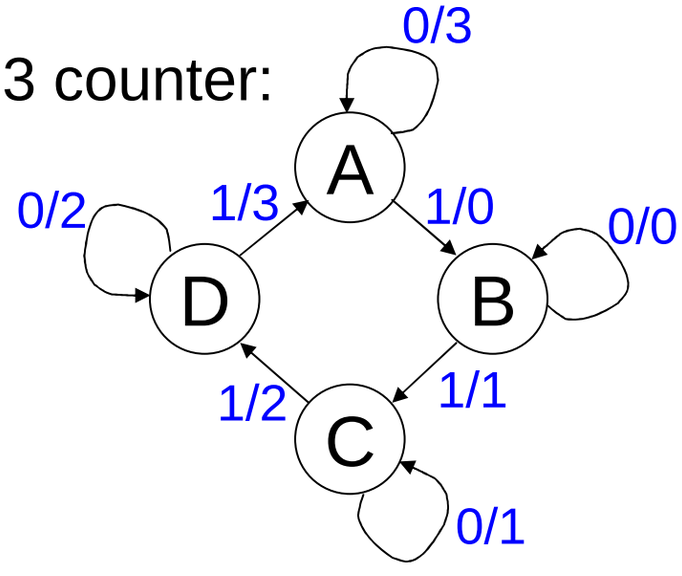
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2															

Mealy FSM example:

Example: - **Mealy** stop/go 0, 1, 2, 3 counter:

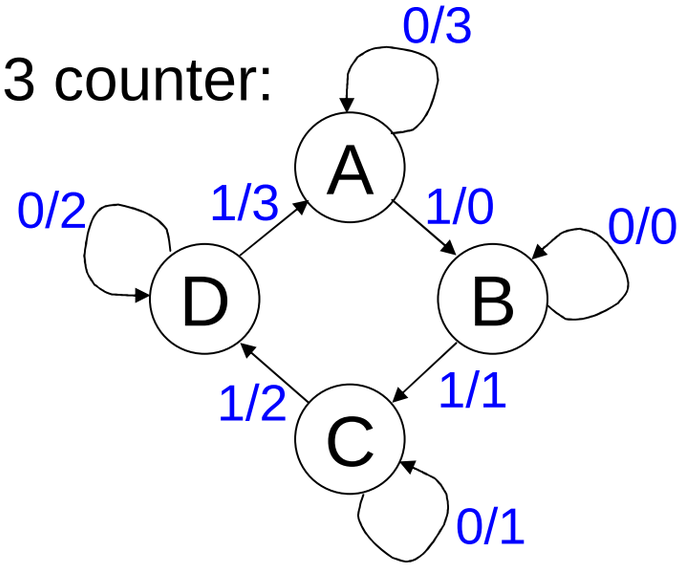


we need to know
input and state to know output

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2															

Mealy FSM example:

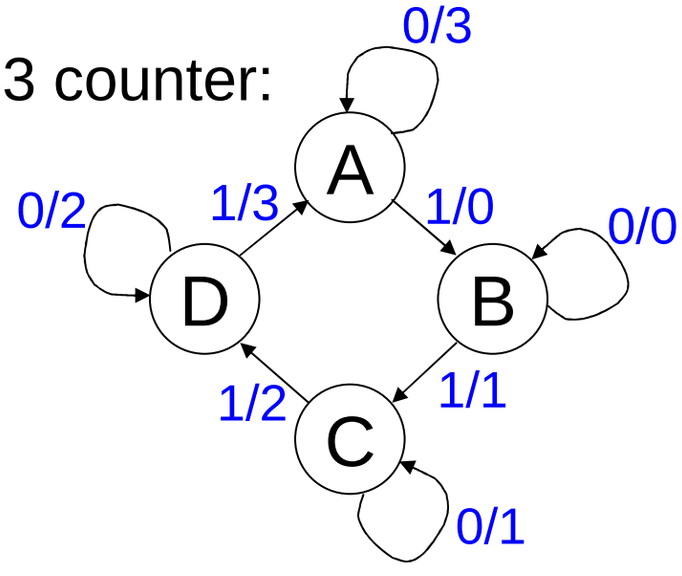
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3														

Mealy FSM example:

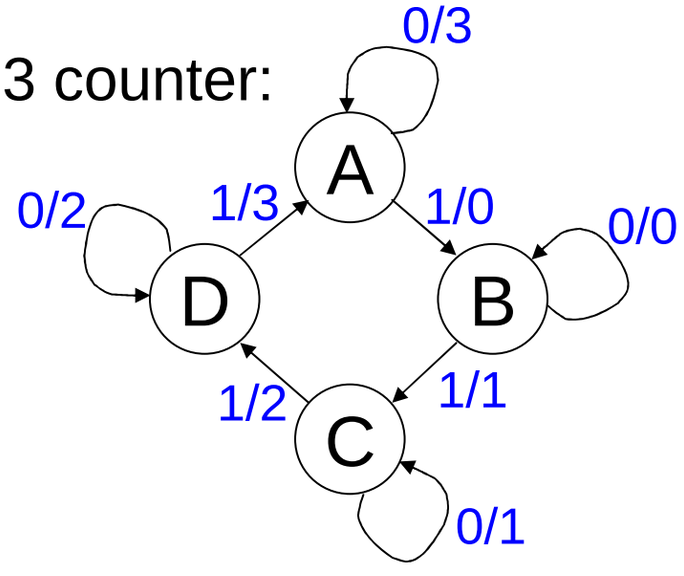
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0													

Mealy FSM example:

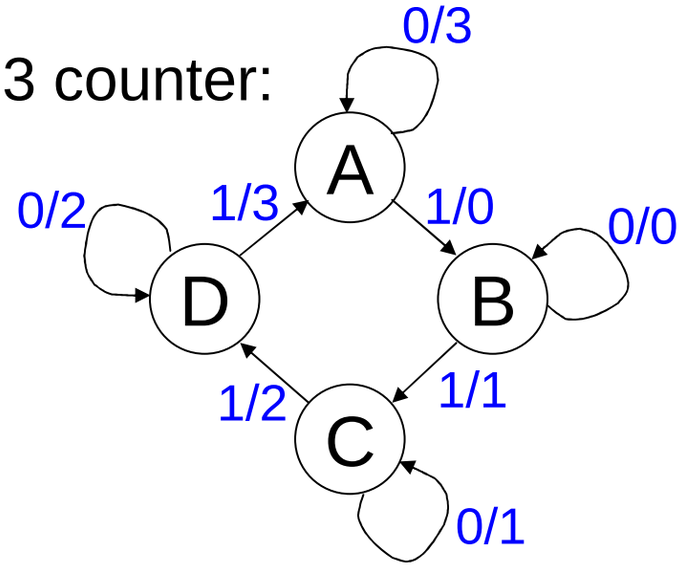
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1												

Mealy FSM example:

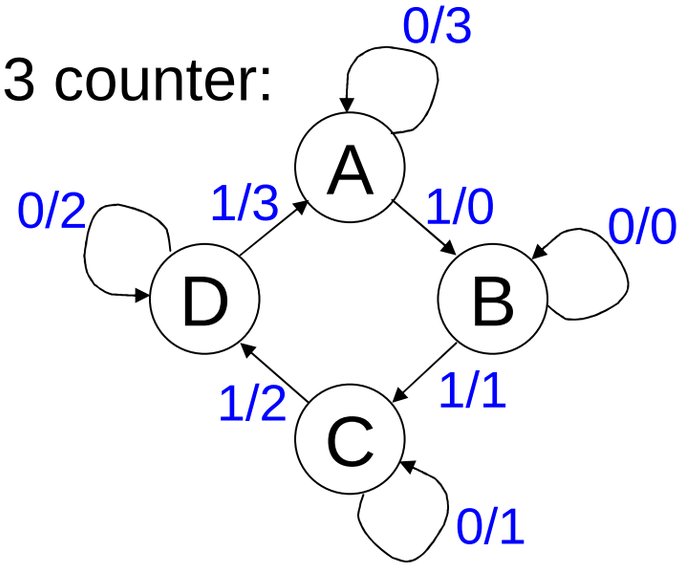
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2											

Mealy FSM example:

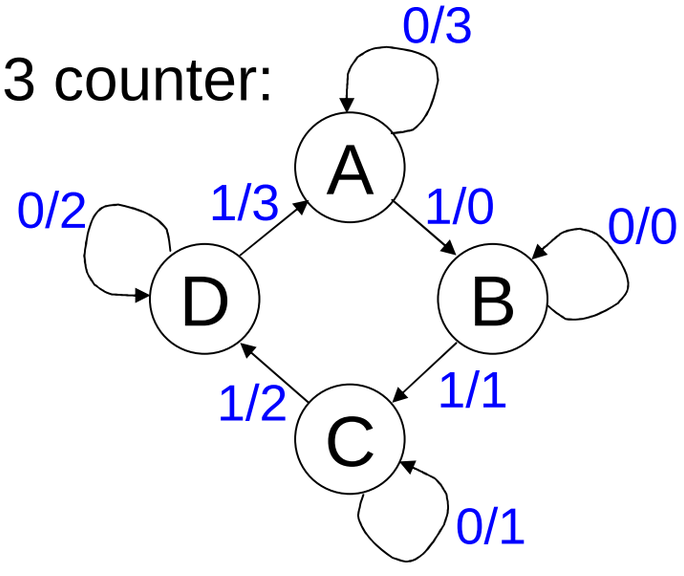
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2										

Mealy FSM example:

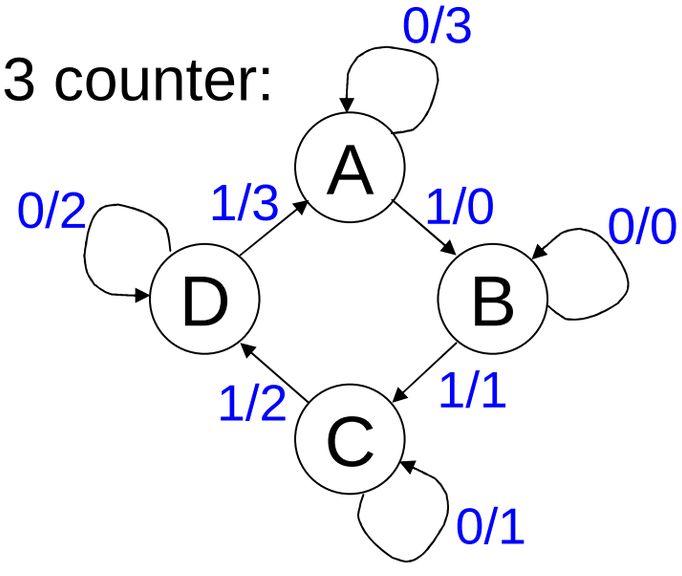
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2							

Mealy FSM example:

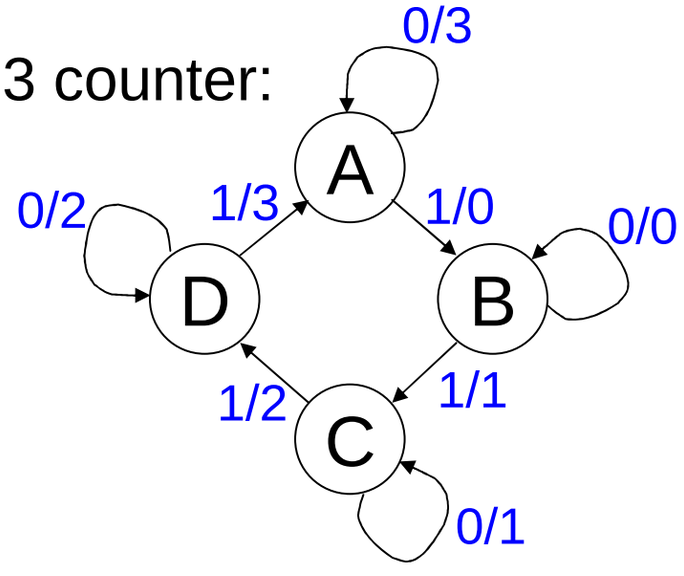
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2	3						

Mealy FSM example:

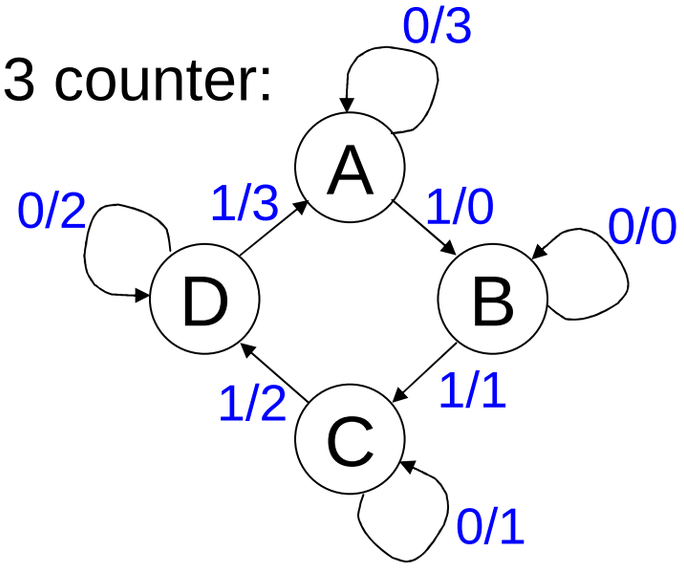
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2	3	3					

Mealy FSM example:

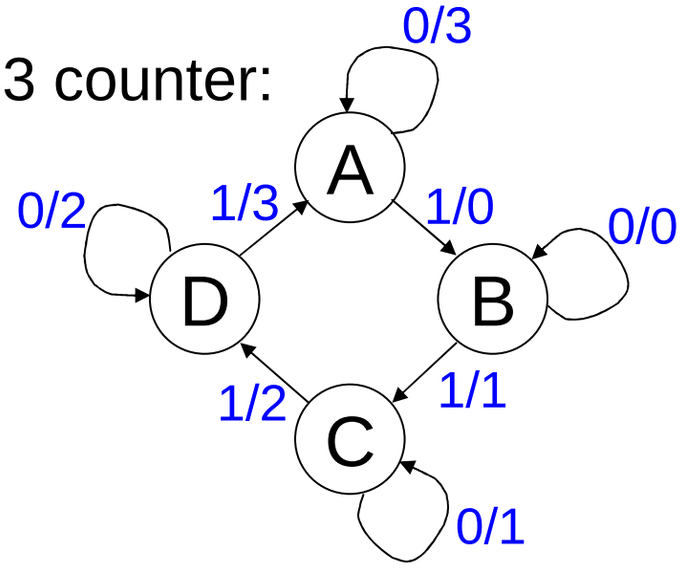
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2	3	3	0				

Mealy FSM example:

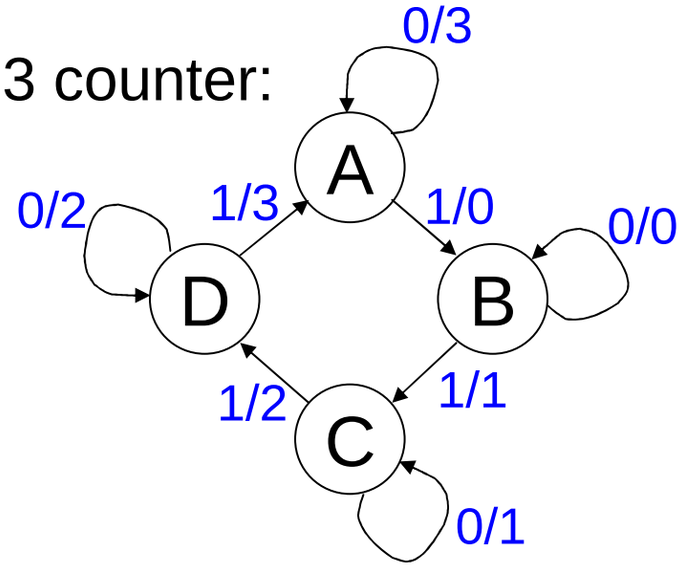
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2	3	3	0	1			

Mealy FSM example:

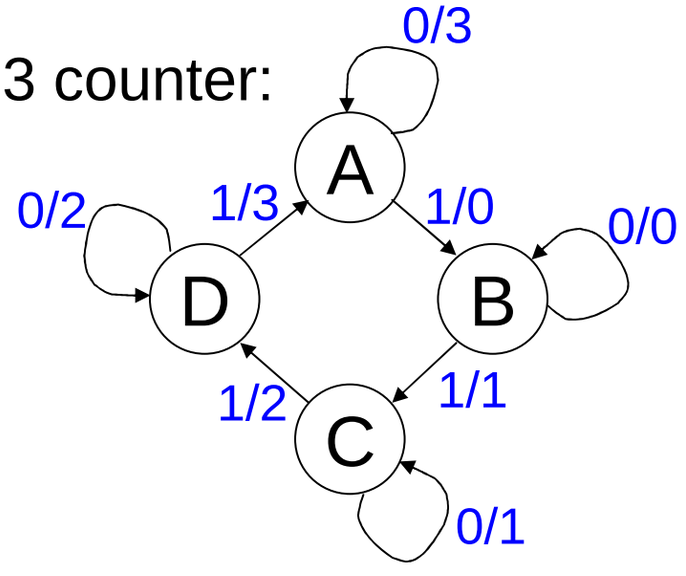
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2	3	3	0	1	2		

Mealy FSM example:

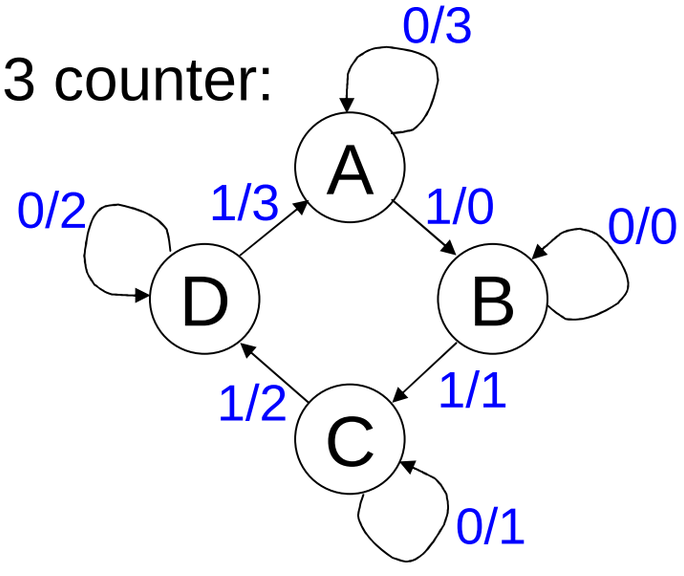
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2	3	3	0	1	2	2	

Mealy FSM example:

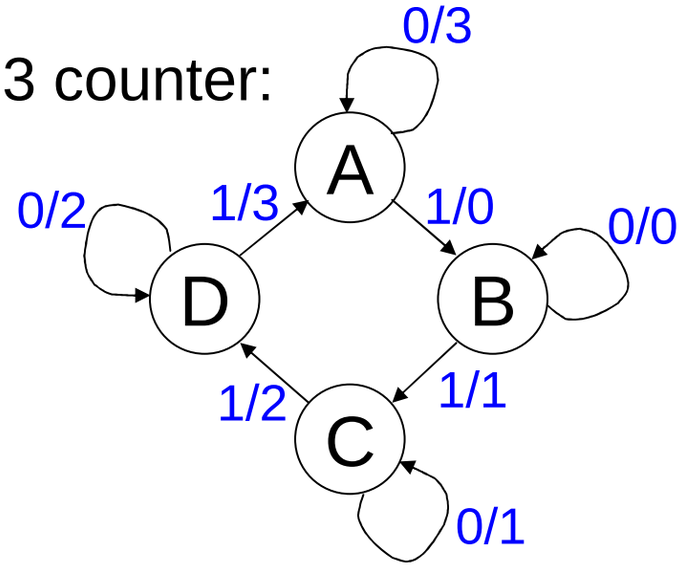
Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2	3	3	0	1	2	2	2

Mealy FSM example:

Example: - **Mealy** stop/go 0, 1, 2, 3 counter:



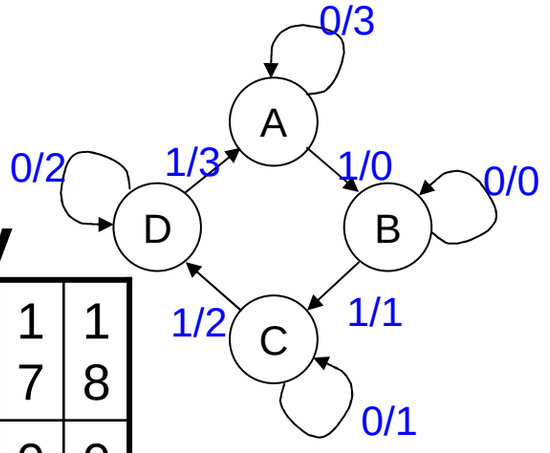
It's possible to have different outputs with different inputs, even when the machine is in the same state

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2	3	3	0	1	2	2	2

Mealy and Moore Comparison:

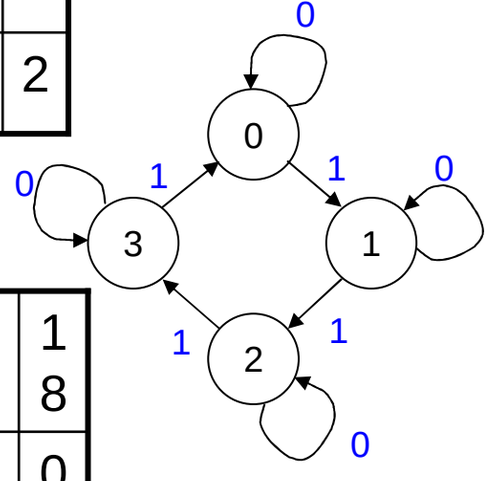
Mealy

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
state	A	B	C	D	A	B	C	D	D	D	D	D	A	A	B	C	D	D
output	0	1	2	3	0	1	2	2	2	2	2	3	3	0	1	2	2	2



Moore

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
c	0	1	2	3	0	1	2	3	3	3	3	3	0	0	1	2	3	3



Mealy and Moore Comparison:

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
Moore out	0	1	2	3	0	1	2	3	3	3	3	3	0	0	1	2	3	3
Mealy out	0	1	2	3	0	1	2	2	2	2	2	3	3	0	1	2	2	2

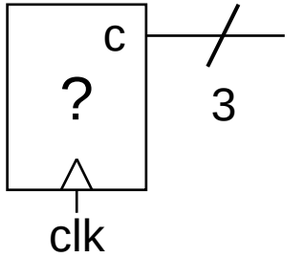
Mealy and Moore Comparison:

clk#	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
run	1	1	1	1	1	1	1	0	0	0	0	1	0	1	1	1	0	0
Moore out	0	1	2	3	0	1	2	3	3	3	3	3	0	0	1	2	3	3
Mealy out	0	1	2	3	0	1	2	2	2	2	2	3	3	0	1	2	2	2

Observations:

- When the 2 machines are running (run = 1), outputs are the same
- Moore machine stops and starts with one clock delay, but Mealy machine has no delay.

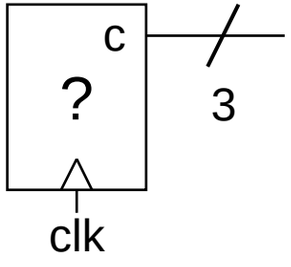
Unused states and reset:



c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

What are the states? How many flops?

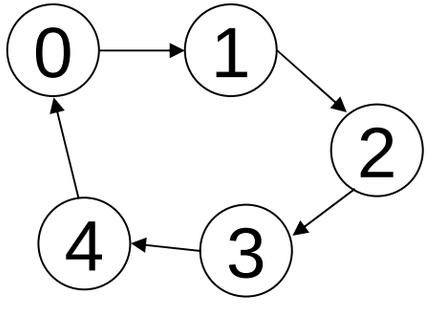
Unused states and reset:



c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

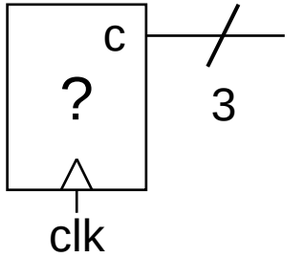
What are the states? How many flops? **3 flops**

STD is



but, 3 flops = 8 states

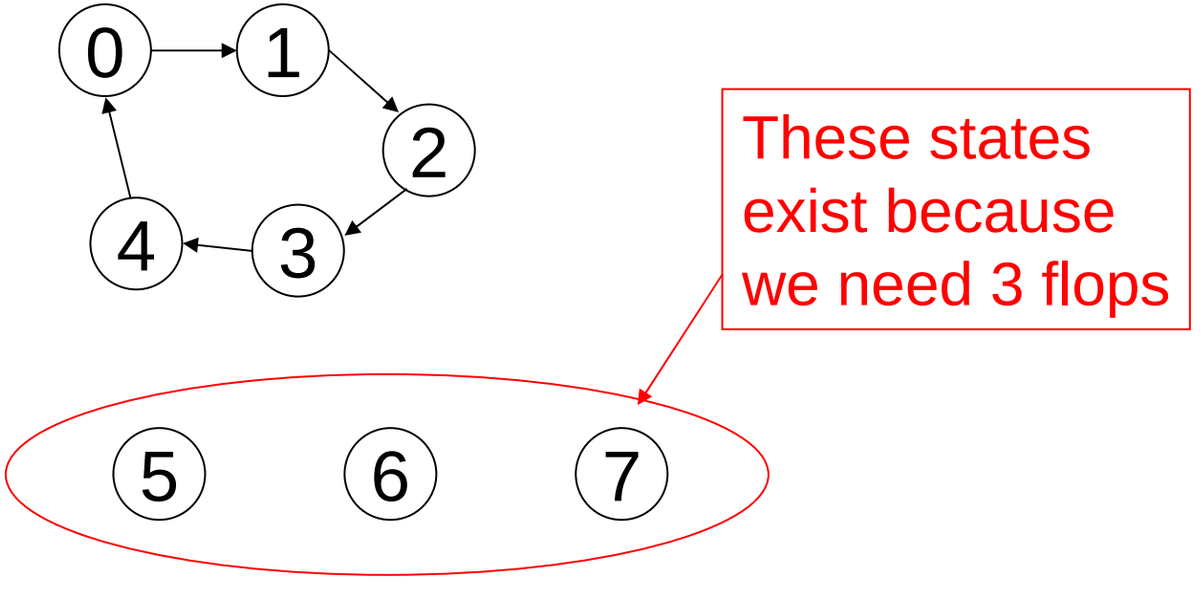
Unused states and reset:



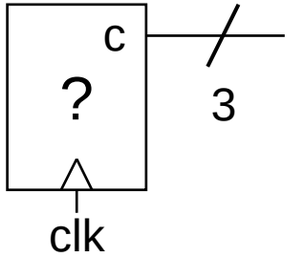
c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

What are the states? How many flops? **3 flops**

STD is



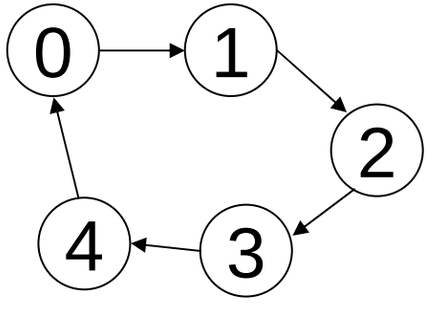
Unused states and reset:



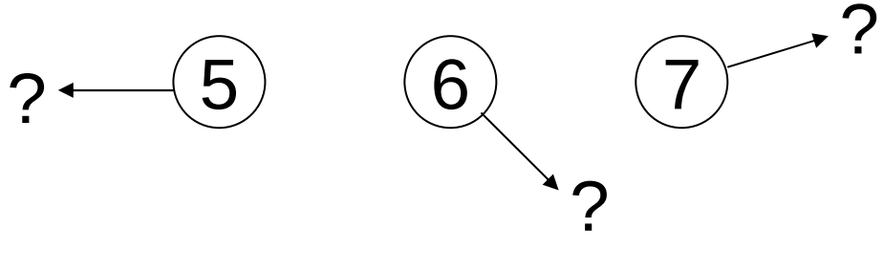
c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

What are the states? How many flops? **3 flops**

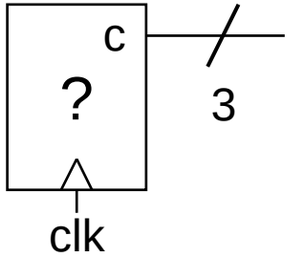
STD is



These states
also have arcs
pointing somewhere



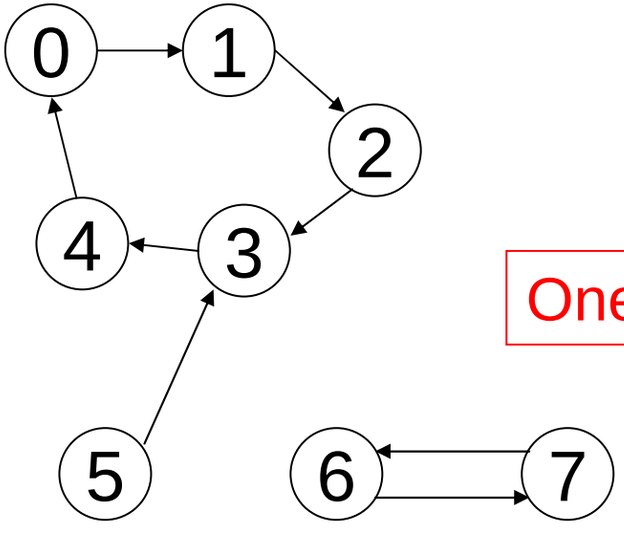
Unused states and reset:



c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

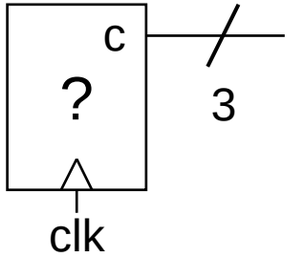
What are the states? How many flops? **3 flops**

STD is



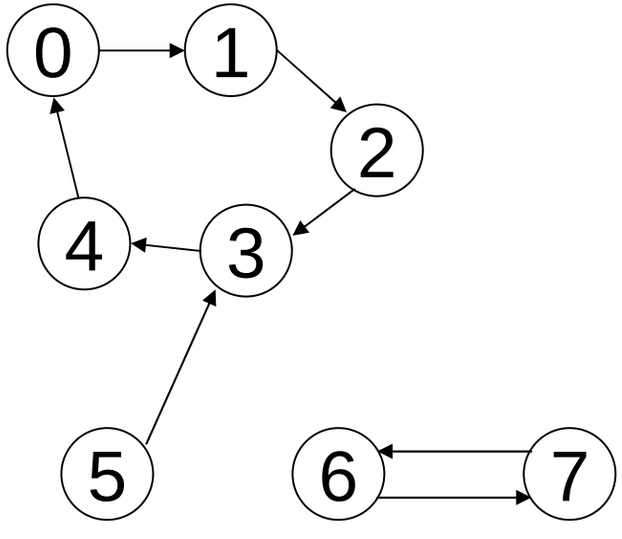
One possible example

Unused states and reset:

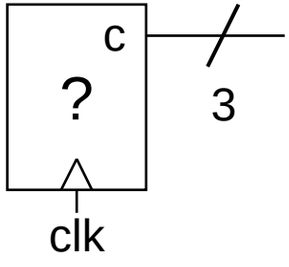


c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

When the circuit is powered on, the initial state is unknown. Why?

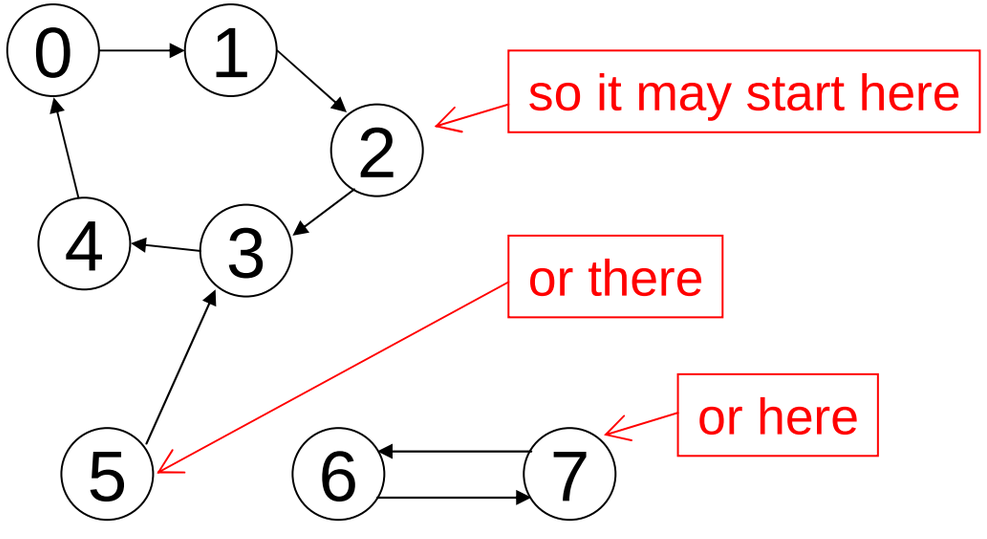


Unused states and reset:

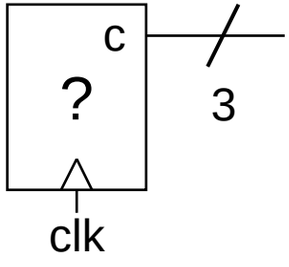


c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

When the circuit is powered on, the initial state is unknown. Why? flip flop outputs = state

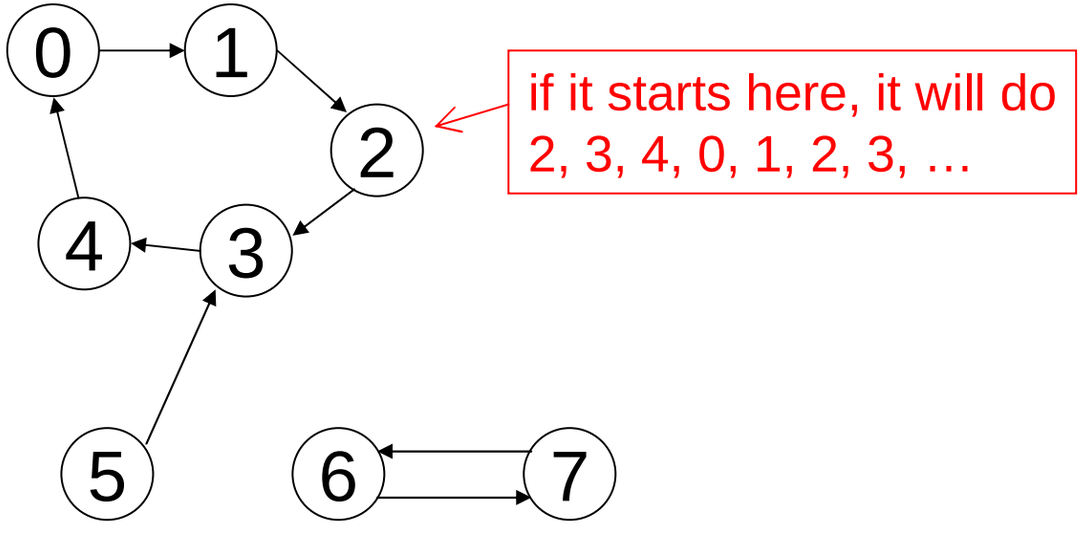


Unused states and reset:

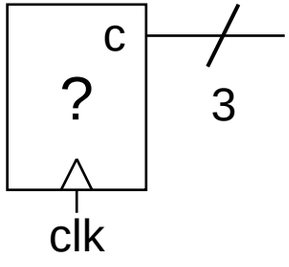


c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

When the circuit is powered on, the initial state is unknown. Why? flip flop outputs = state

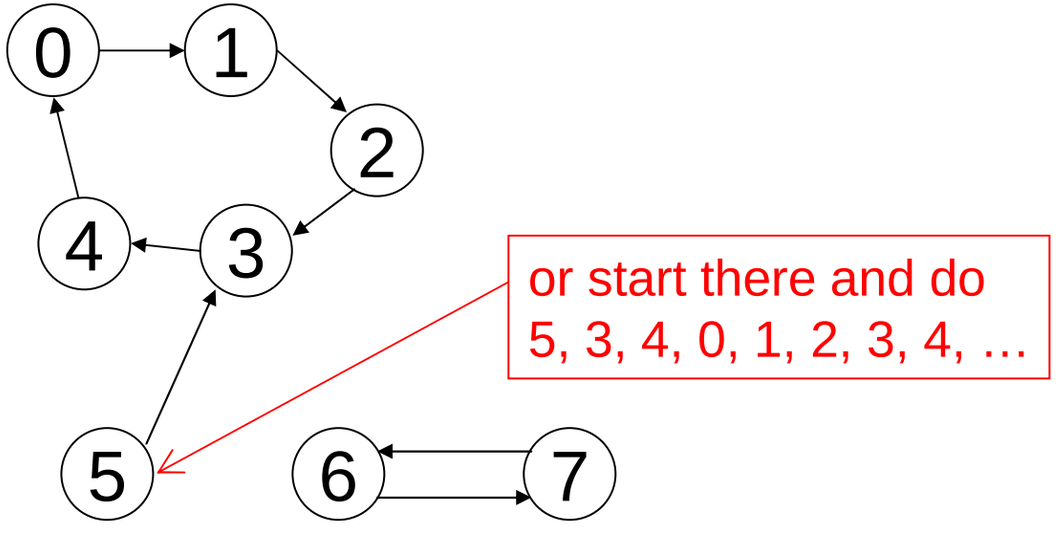


Unused states and reset:

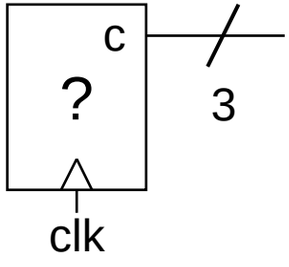


c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

When the circuit is powered on, the initial state is unknown. Why? flip flop outputs = state

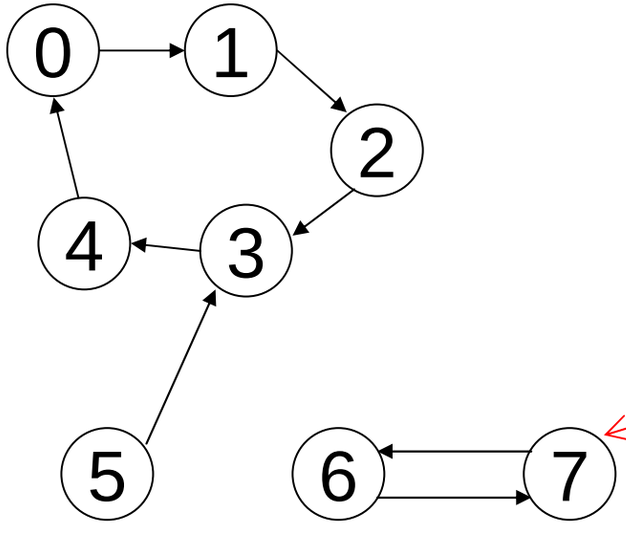


Unused states and reset:



c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

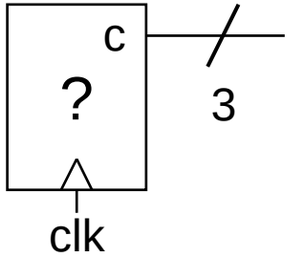
When the circuit is powered on, the initial state is unknown. Why? flip flop outputs = state



if it starts here, it will do 6, 7, 6, 7, 6, 7, …

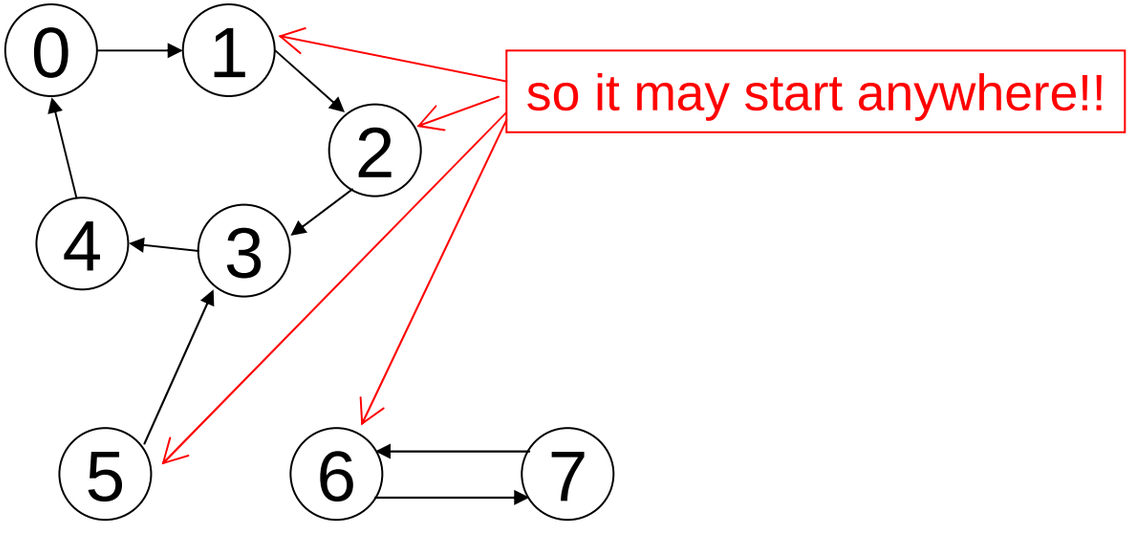
NOT WHAT WE WANT!!!

Unused states and reset:

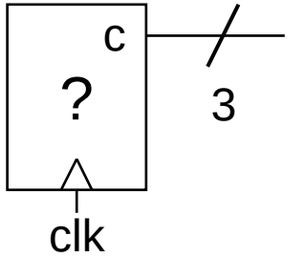


c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

Even worse, the same circuits starting up at the same time may act differently! → No longer singing the same song

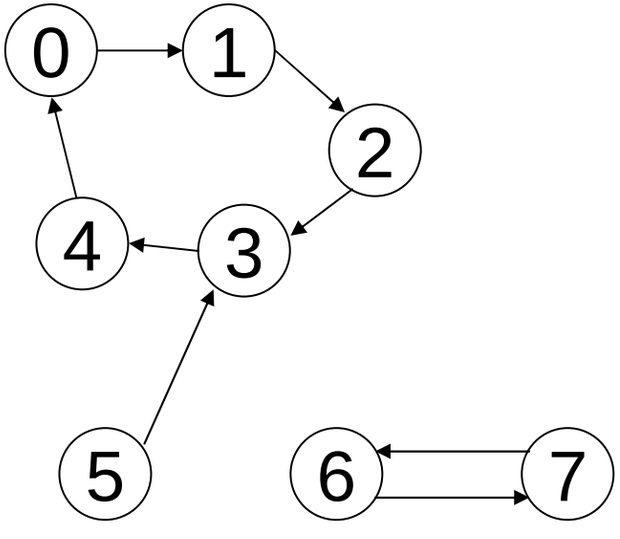


Unused states and reset:



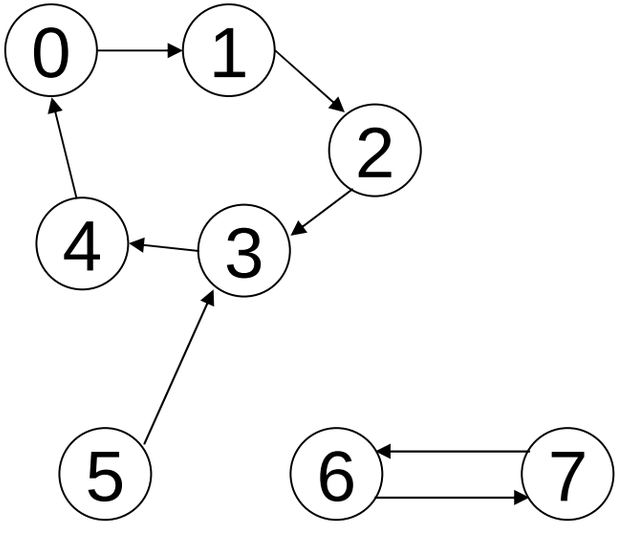
c: 0, 1, 2, 3, 4, 0, 1, 2, 3, 4, …

Most of the time, we want to be able to control the start up behavior: start from this state at this time…



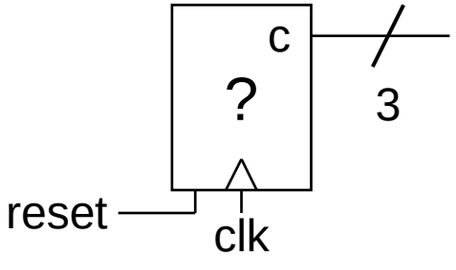
Unused states and reset:

- We modify flip flops (their outputs = state), to start up at a designated place, given a reset signal.
- reset is an external signal into FSM (to the flops only) to force FSM to go to a particular state
- reset is modeled as an arc. like this: R 

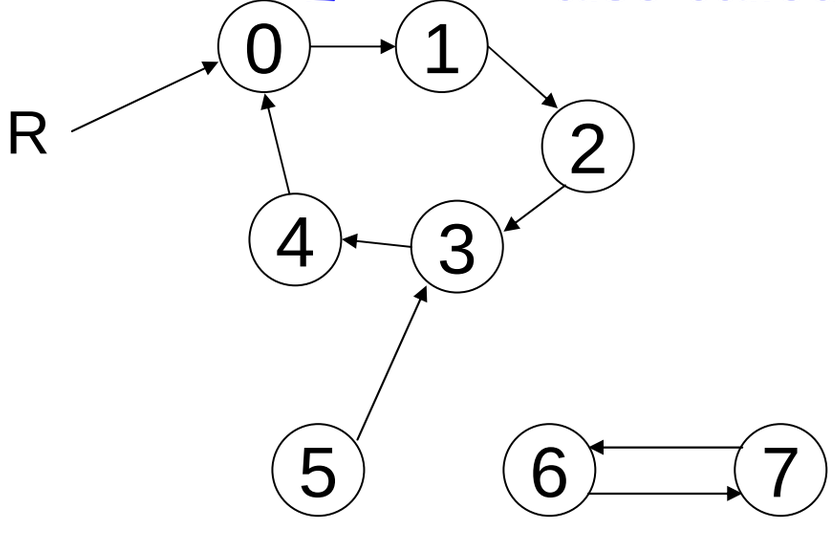


Unused states and reset:

- Now flip-flops needs reset in addition to clock
- reset is not considered an “input” to FSM (neither is clock)

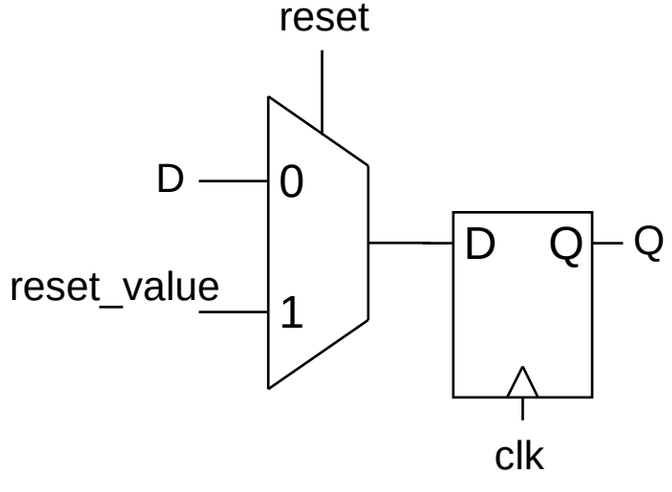


After reset, FSM is guaranteed
so start in its initial state
also called reset state



Unused states and reset:

- some flip-flops needs reset in addition to clock
- reset is not considered an “input” to FSM (neither is clock)
- flip-flop with reset is designed as follow:




```
always @(posedge clk)
   if (reset)
      q <= #1 reset_value;
   else
      q <= #1 d;
```

Think about this as a flip-flop with reset, not a MUX and a flop

Unused states and reset:

- if the reset value is never changed, we can simplify