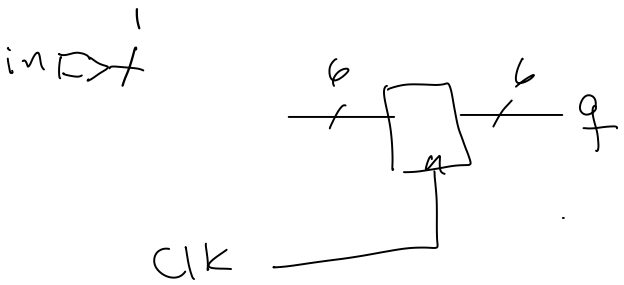
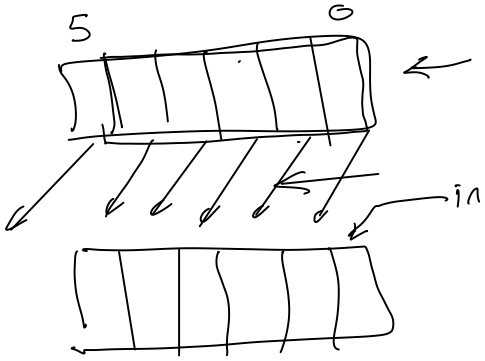
Examples of State Machines

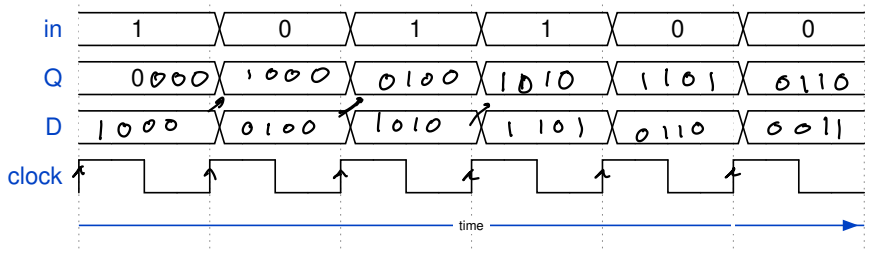
Exercise 1: The example above is an N-bit shift register that shifts the bits right. Draw a block diagram and write the Verilog for a 6-bit shift register that shifts left.



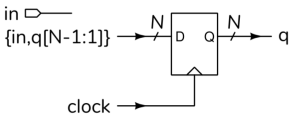



```
always_ff @ (posedge clk)
  q <= {q[4:0], in} ;
```

Exercise 2:

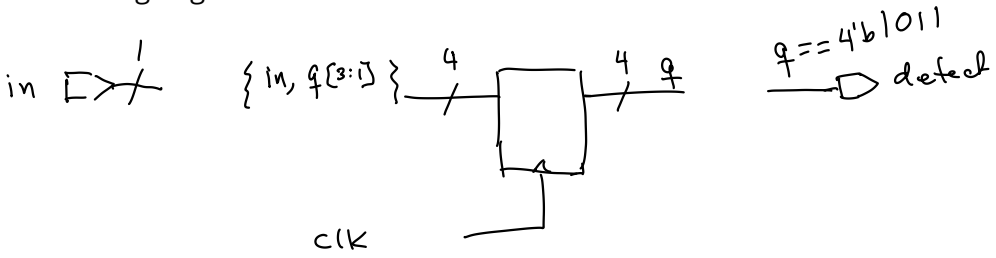


1 bit
Q [3:0]
D [3:0]



Fill in the diagram above for a 4-bit ($N = 4$) right-shift shift register. Assume the initial value is zero. Which bit is the oldest (first) value in the waveform? Which bit of the shift register holds the oldest value?

Exercise 3: Draw a block diagram and write the Verilog for a circuit that sets an output named **`detect`** high when the sequence of values 1, 1, 0, 1 has appeared on an input named **`in`** on successive rising edges of the clock.



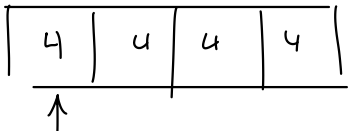

```
module ex100 (input logic in, clk,
              output logic detect);
  logic [3:0] q;
  always_ff @(posedge clk)
    q <= {in, q[3:1]};
  assign detect = q == 4'b1011;
endmodule
```

Exercise 4: How could you modify the code so that **`digits`** is only updated when an **`enable`** input is asserted?

```
always_ff @(posedge clk) digits
   <= { digits[1:3], digit } ;
```

```
<= enable ? {digits[1:3], digit} : digits ;
```

Exercise 5: How many states can this state machine have?

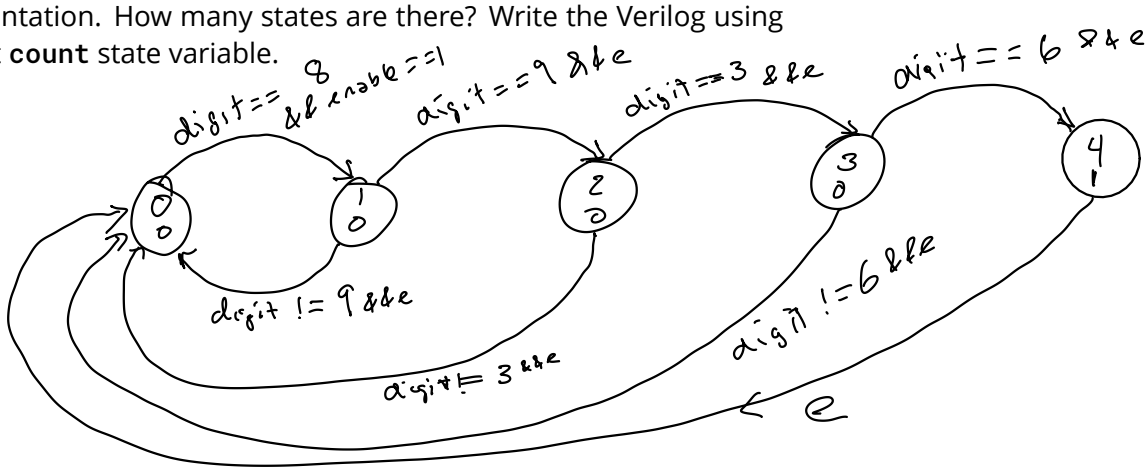


$10 \times 10 \times 10 \times 10 = 10^4$

or $16 \times 16 \times 16 \times 16 =$

$2^4 \times 2^4 \times 2^4 \times 2^4 = 2^{16}$

Exercise 6: Draw the state transition diagram for this simpler implementation. How many states are there? Write the Verilog using a 3-bit **count** state variable.



e: enable == 1
d: digit

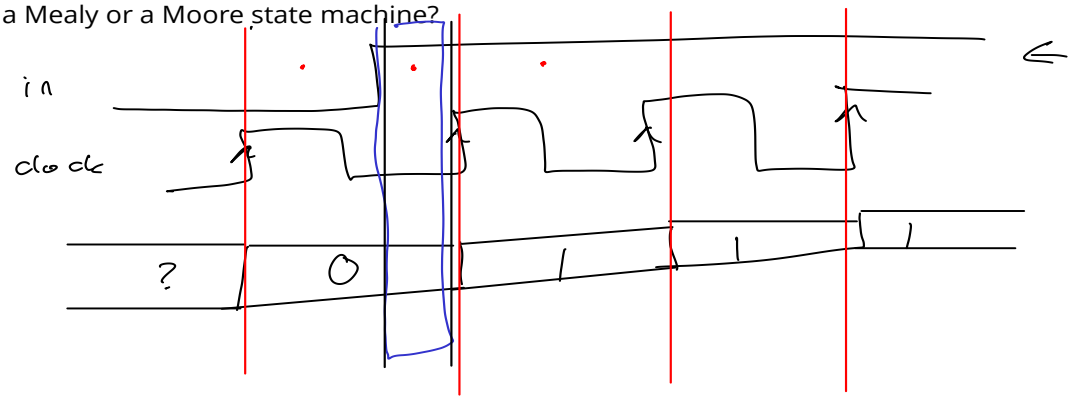
assume the combination is 8936

Exercise 7: For which states would a **fell** output be asserted? A **rose** output? Draw the schematic and write the Verilog for this state machine. Assume an input **in** and a 2-bit register **bits** that holds the two most recent input values.

fell would be asserted in state 10.

rose would be asserted in state 01.

Exercise 8: Can you design an edge detector that uses only one bit? Is this a Mealy or a Moore state machine?



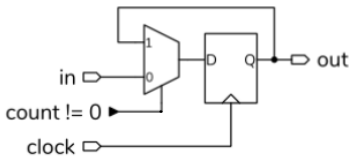
assign rose = !q && in; ← Mealy (output is function of input)

Exercise 9: Write **`always_ff`** statements that implement these state machines.

count	in == out	next count
x	1	$N-1$
0	x	$N-1$
n	0	$n-1$

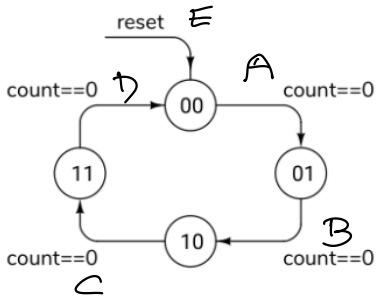
```
logic [15:0] count;

always_ff @(posedge clock)
  count <= in==out ? 49_999 :
           !count  ? 49_999 :
           in!=out ? count-1'b1 : count;
```




```
always_ff @(posedge clock)
  out <= count!=0 ? out : in;
```

Exercise 10: Write the state transition table for this state machine.



state	inputs	next state

state	reset	count==0	next state
X	1	X	00 E
00	0	1	01 A
01	0	1	10 B
		1	11 C
10	0	1	00 D
11	0		