CSE 241

Lecture 6

University at Buffalo
School of Engineering and Applied Sciences

Overview for this lecture

- Reminders
- Sequential Logic

Reminders

- Regrades due now

The Plan

- Monday- Finish Sequential Logic and start Verilog
- Wednesday- Verilog
- Following Monday- Review and Practice Exam
- Following Wednesday- Final Exam

Questions?

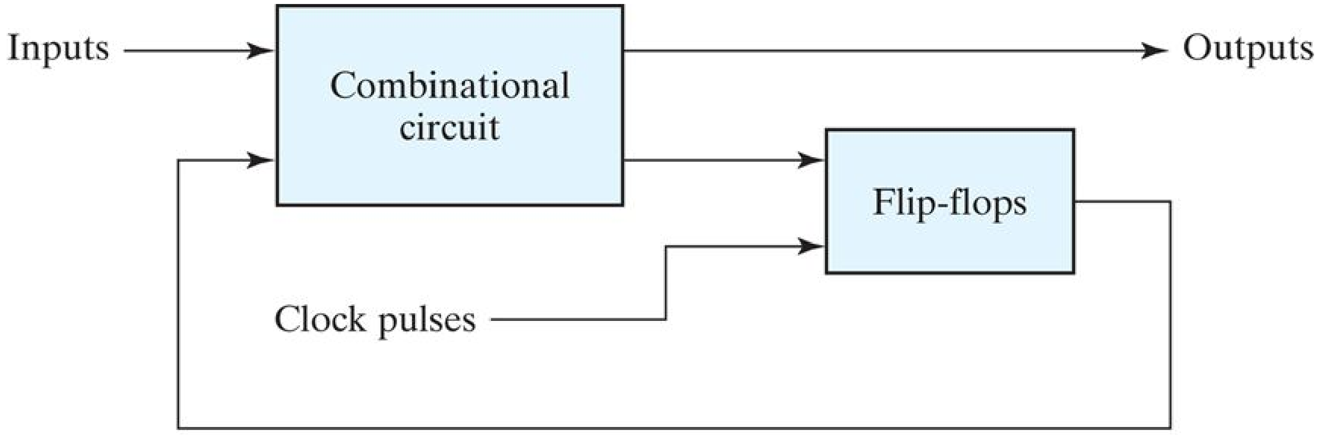
Vocabulary

- Clock- periodic function, 50% duty cycle
- Period- time it takes for one segment of the pattern to repeat
- Duty Cycle- The ratio of the time the wave is at the high state in the total time in the periodic behavior, we like to call it DC
- Frequency- Number of event occurrences in a period of time
- Latch- Sequential design but not clock dependent
- FlipFlop- Sequential circuit that changes can only happen with the clock change condition
- State- describes the values that all variables in the system take at a given time
- Parallel- n-bits are transmitted across n-wires at one time
- Series- n-bits are transmitted one at a time, it takes n-clock cycles

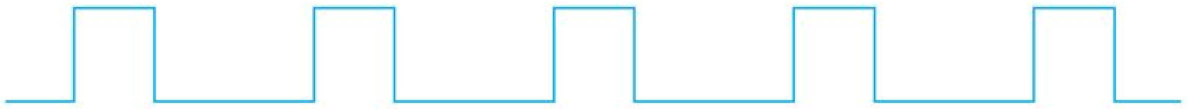
Sequential Logic

- Sequential logic involves one or more memory elements that retain some of the history of the state of the system.
- It is analogous to a combination lock with a dial that must be set to various numbers in order to unlock.

What does this look like?



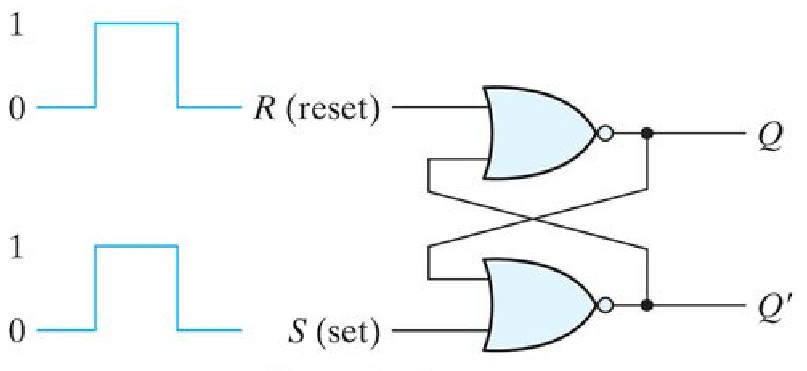
(a) Block diagram



(b) Timing diagram of clock pulses

Latch

Inputs: R and S



(a) Logic diagram

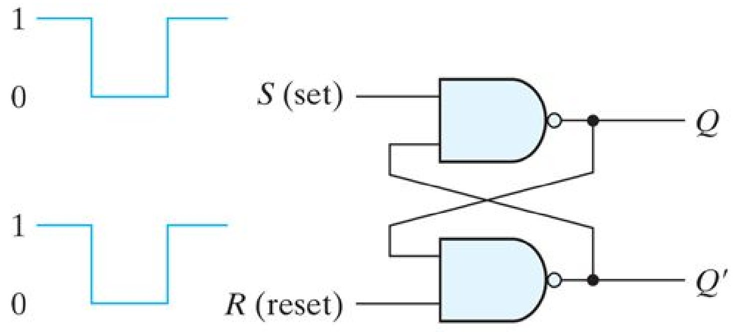
S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after $S = 1, R = 0$)
0	1	0	1	
0	0	0	1	(after $S = 0, R = 1$)
1	1	0	0	(forbidden)

(b) Function table

When R and S are

- both 0, the latch stays in the same state
- 0 (R) and 1 (S), the latch SETS to 1.
- 1 (R) and 0 (S), the latch RESETS to 0.
- both 1, both outputs are 0.

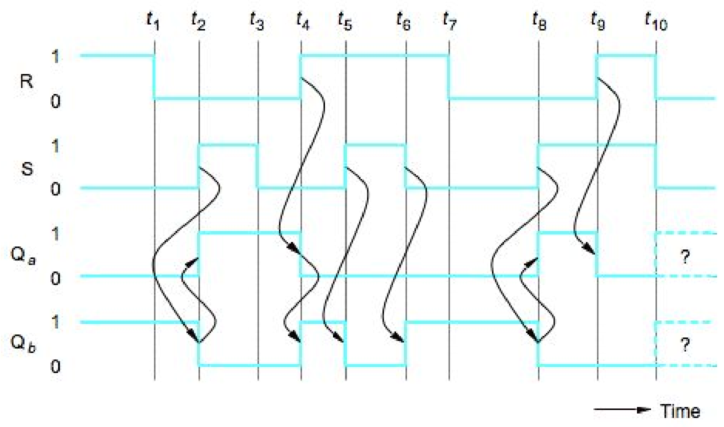
SR Latch



(a) Logic diagram

S	R	Q	Q′	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	(forbidden)

(b) Function table

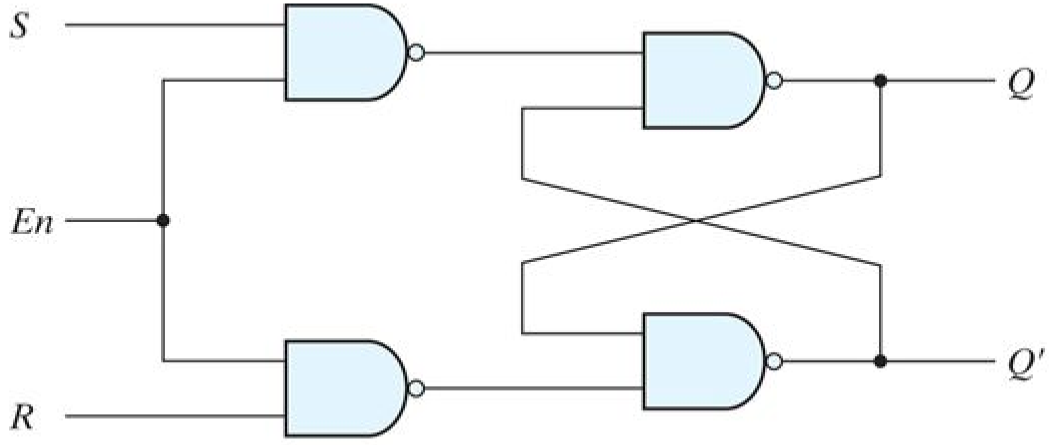


(c) Timing diagram

Figure 7.5. A latch built with NOR gates.

Clocked or “Gated” SR Latch

- A clock signal (Clk) is added to enable or disable the latch.
  - When Clk = 0, the latch holds its state.
    - That is, it ignores the changes to S and R.
  - When Clk = 1, the latch state is determined by the S and R inputs.

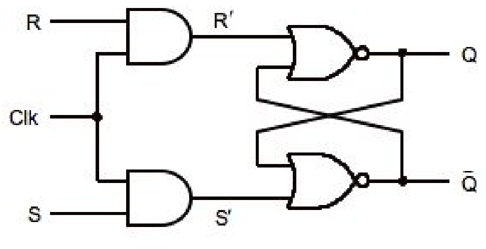


(a) Logic diagram

En	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	$Q = 0$; reset state
1	1	0	$Q = 1$; set state
1	1	1	Indeterminate

(b) Function table

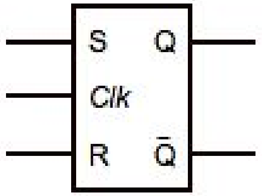
Gated SR Latch



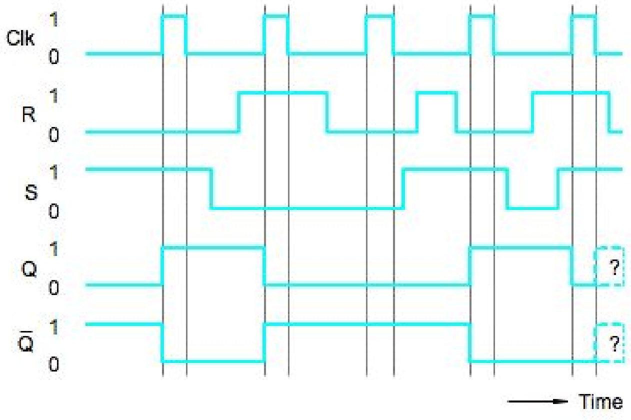
(a) Circuit

Clk	S	R	$Q(t+1)$
0	x	x	$Q(t)$ (no change)
1	0	0	$Q(t)$ (no change)
1	0	1	0
1	1	0	1
1	1	1	x

(b) Truth table



(d) Graphical symbol



(c) Timing diagram

Figure 7.6. Gated SR latch.

SR Latch with NAND Gates

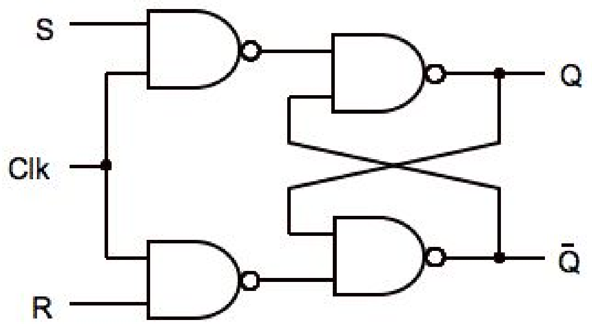


Figure 7.7. Gated SR latch with NAND gates.

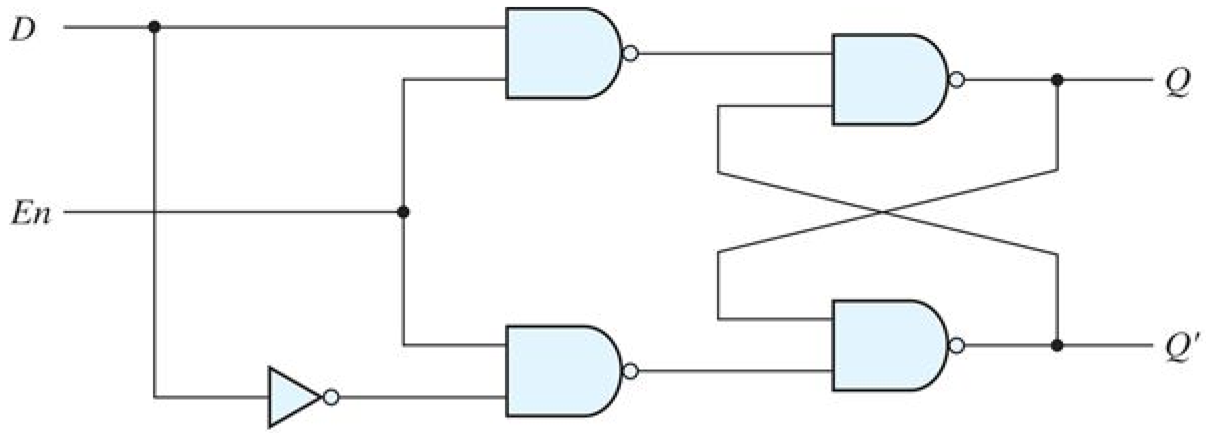
Clocked Signals

- Clocked signals may be:
  - Level Sensitive- on when the clock has some value (usually 1)
  - Edge Sensitive- on during a transition
    - positive edge or rising edge
      - 0=>1
    - negative edge or falling edge
      - 1=> 0
- Edge sensitive flip-flops respond to inputs during only a very small time interval.

D Latch

- D latches or flipflops are designed to copy data from the input and hold it.
- Think of D for Data or Delay.
- A D latch can be constructed from a SR latch by inputting D to S and D’ to R.

D	Q	Qn
0	0	1
0	1	0
1	0	1
1	1	1

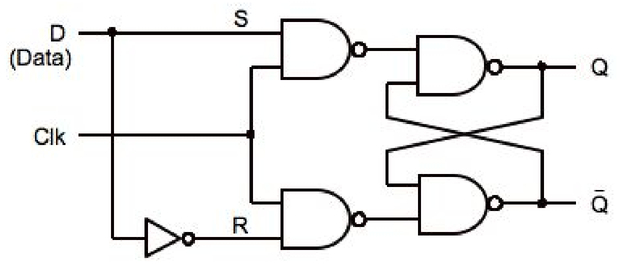


(a) Logic diagram

En	D	Next state of Q
0	X	No change
1	0	$Q = 0$; reset state
1	1	$Q = 1$; set state

(b) Function table

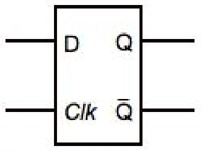
D Latch



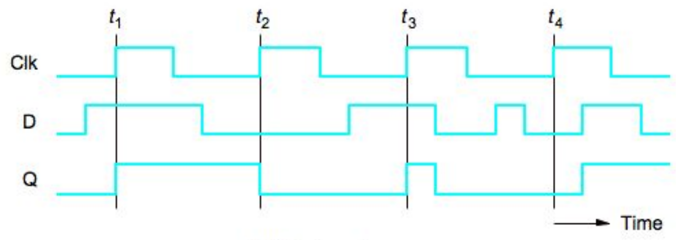
(a) Circuit

Clk	D	$Q(t+1)$
0	x	$Q(t)$
1	0	0
1	1	1

(b) Truth table



(c) Graphical symbol



(d) Timing diagram

Figure 7.8. Gated D latch.

Setup and Hold Times

- Setup time, t_s, is the minimum time the input must be stable prior to the clock signal.
- Hold time, t_h, is the minimum time the input must be stable after the clock signal.
- Typical CMOS values:
  - t_s = 3 ns and t_h = 2 ns
  - CMOS=Complementary Metal-oxide Semiconductor
  - TTL or Transistor-Transistor Logic ICs may have different timing

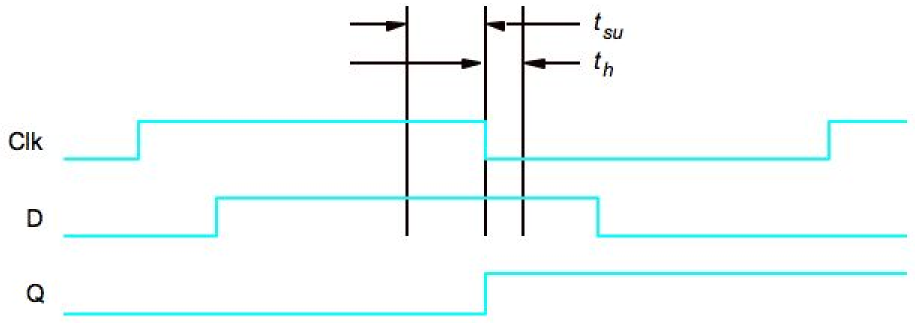
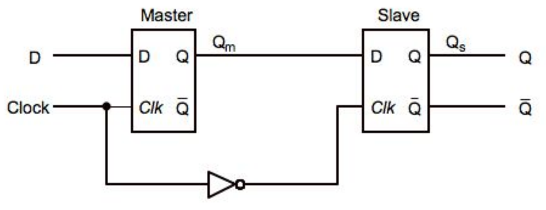


Figure 7.9. Setup and hold times.

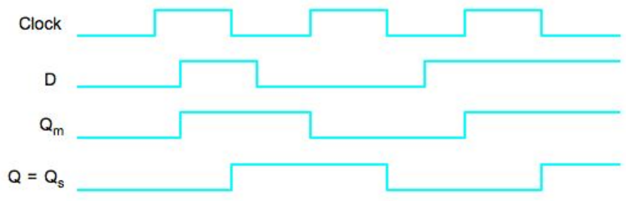
Master - Slave Flip-Flops

- A master-slave flip-flop has two latch circuits.
  - The master responds to the inputs when Clk = 1.
  - The slave responds to the master when Clk = 0
  - Since only half the flip-flop can change for any value of Clk, input changes can affect the flip-flop only once for each clock cycle.

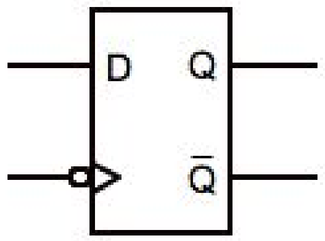
Master - Slave D Flip-Flop



(a) Circuit



(b) Timing diagram



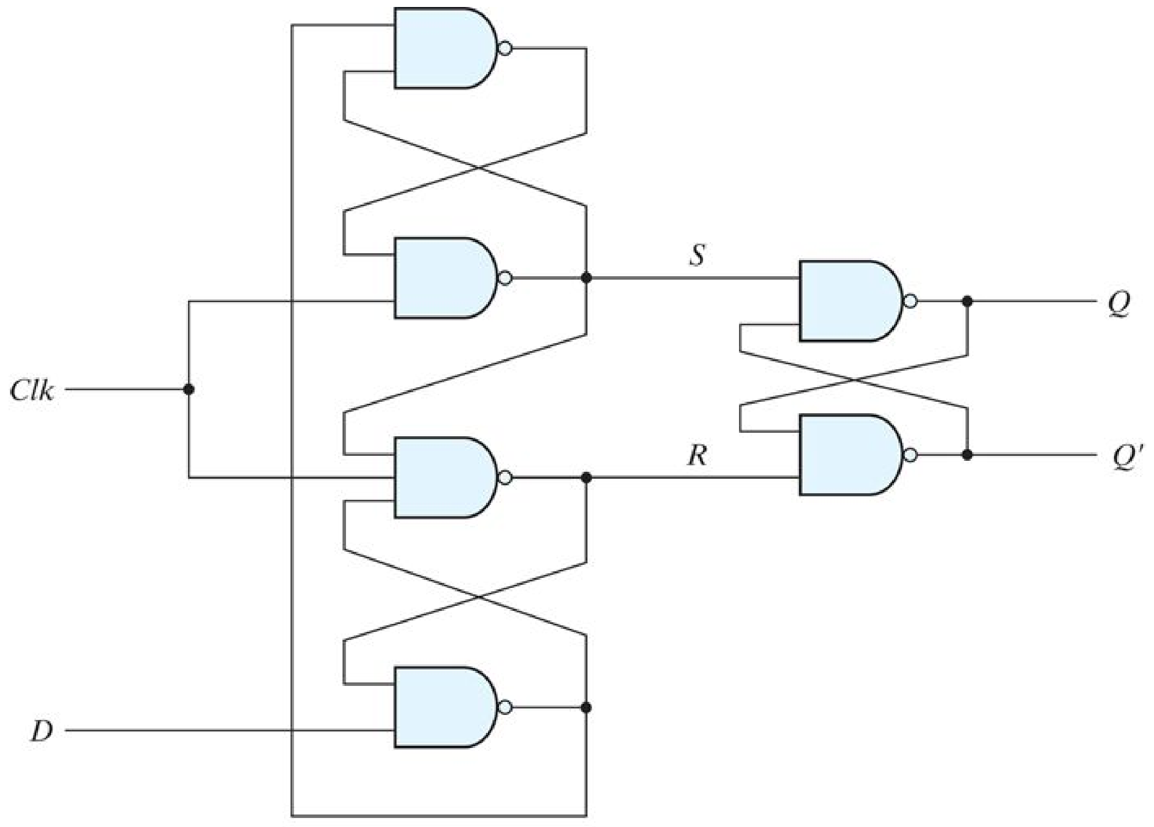
(c) Graphical symbol

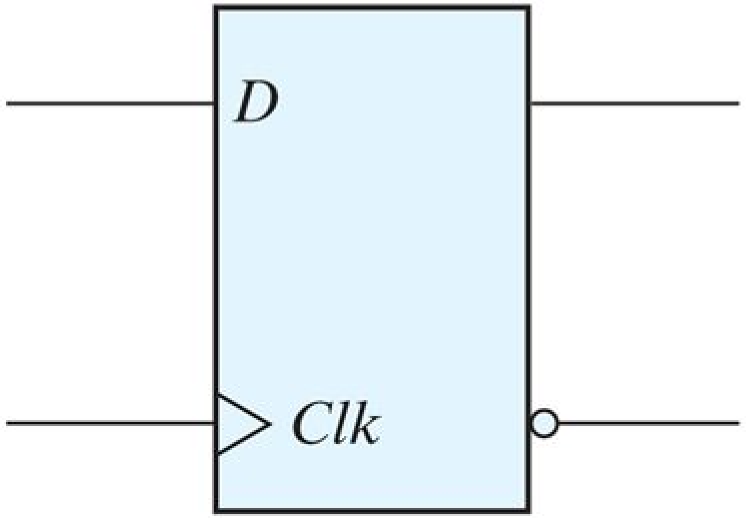
Figure 7.10. Master-slave D flip-flop.

Synchronous vs. Asynchronous Inputs

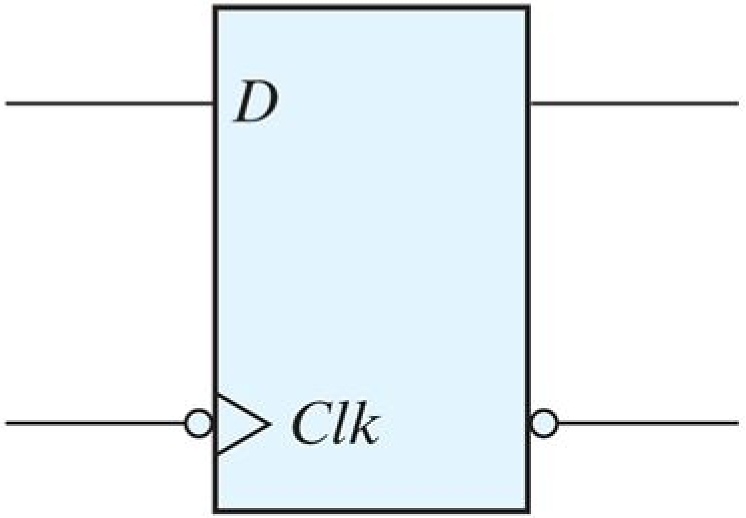
- Synchronous inputs affect the circuit only when the clock is active.
- Asynchronous inputs affect the circuit whenever they occur.
  - Clear - clears the flip-flop to 0
  - Preset - sets the flip-flop to 1.
  - Asynchronous inputs are usually active low.

Asynchronous D Flip-Flop with positive edge trigger



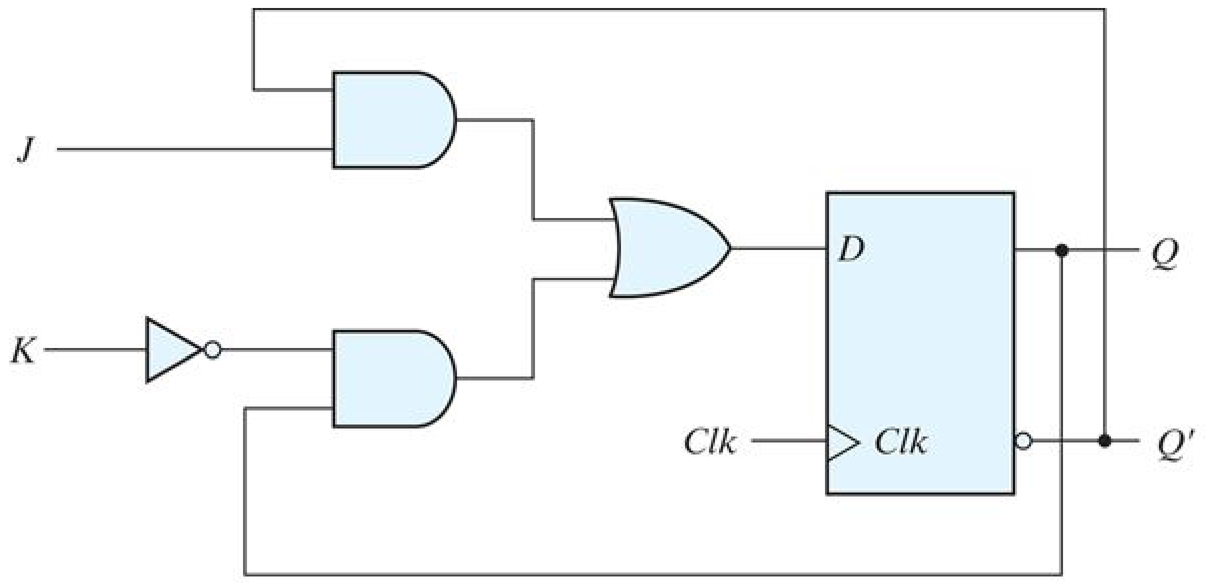


(a) Positive-edge

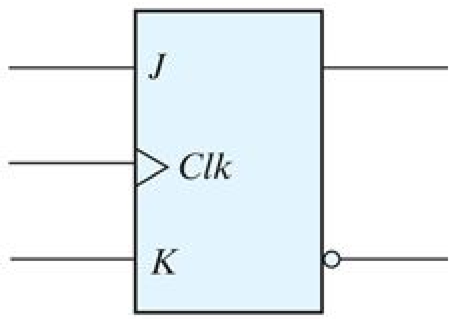


(a) Negative-edge

JK Flip-Flop

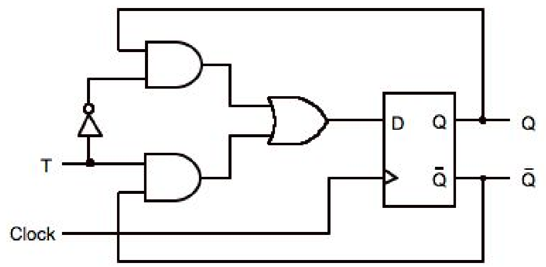


(a) Circuit diagram



(b) Graphic symbol

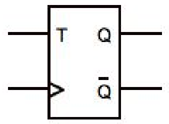
T Flip-Flop Detail



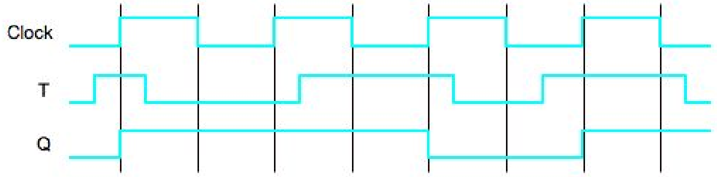
(a) Circuit

T	Q(*t* + 1)
0	Q(*t*)
1	$\bar{Q}(t)$

(b) Truth table

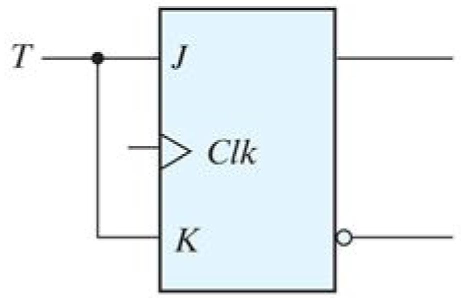


(c) Graphical symbol

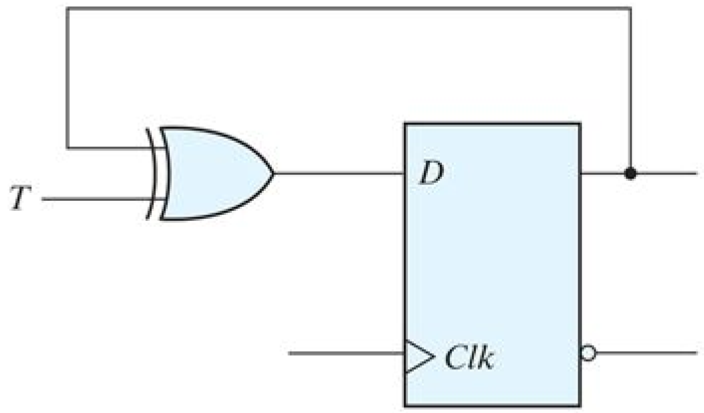


(d) Timing diagram

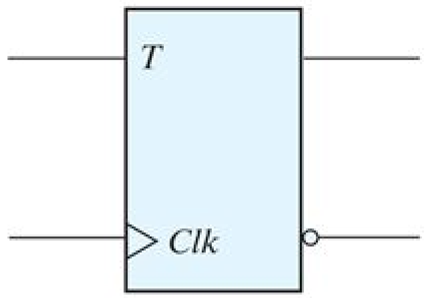
Other ways to look at T



(a) From *JK* flip-flop



(b) From *D* flip-flop



(c) Graphic symbol

Characteristic Equations

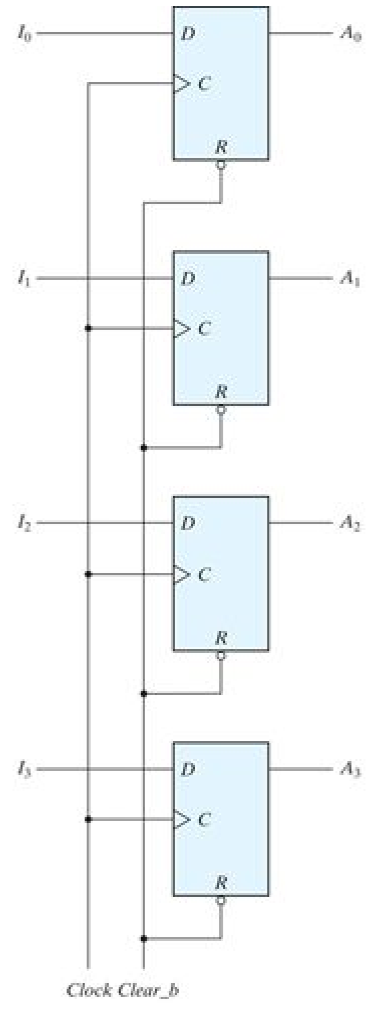
- A characteristic equation gives the next state of a flip-flop as a function of its current state and inputs.
- It can be derived from a Karnaugh map from the flip-flop truth table (transition table).

Flip Flop Characteristic Equations

- D : Qn = D
- T : Qn = T ^ Q
- JK : Qn = JQ’ + K’Q
- SR : Qn = SR’ + R’Q

Registers

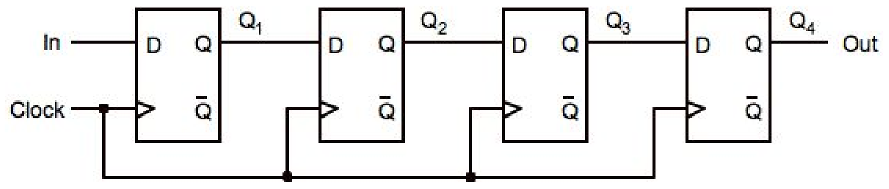
- Flip-flops store precisely one bit of information.
- When we put a bunch of flip-flops together to store n bits of data, we call it a register.



Shift Register

- A shift register is a group of D flip-flops whose Q outputs are connected to the D inputs of the FFs to the right (or left).
- Sometimes there is parallel access to:
    - D inputs and/or the
    - Q outputs.

Shift Register Details



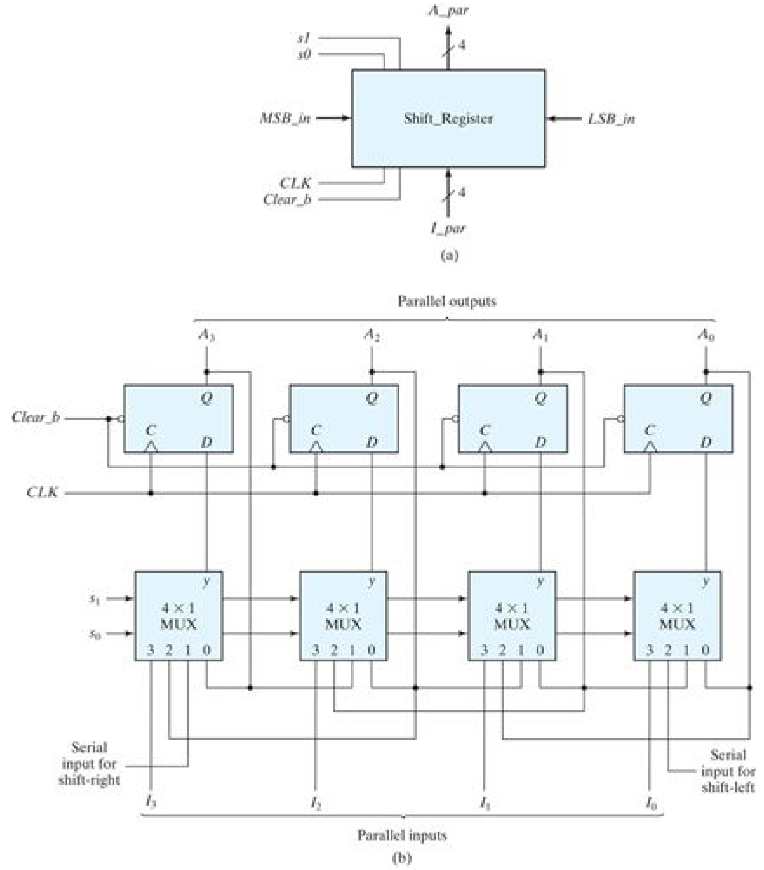
(a) Circuit

	In	Q_1	Q_2	Q_3	Q_4 = Out
t_0	1	0	0	0	0
t_1	0	1	0	0	0
t_2	1	0	1	0	0
t_3	1	1	0	1	0
t_4	1	1	1	0	1
t_5	0	1	1	1	0
t_6	0	0	1	1	1
t_7	0	0	0	1	1

(b) A sample sequence

Figure 7.18. A simple shift register.

4 bit universal shift register



Parallel Access Shift Register

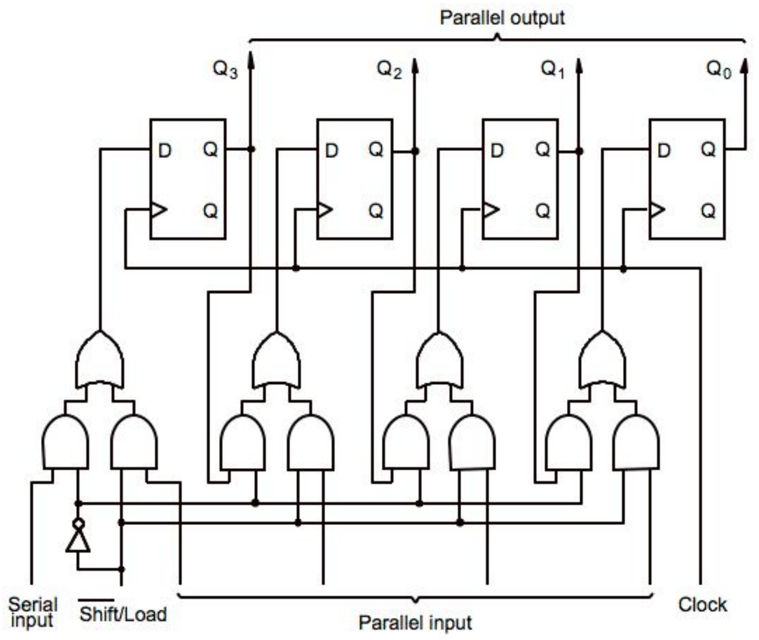
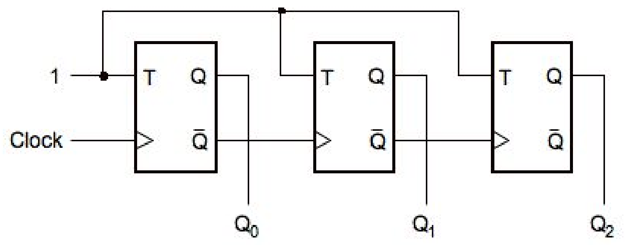


Figure 7.19. Parallel access shift register.

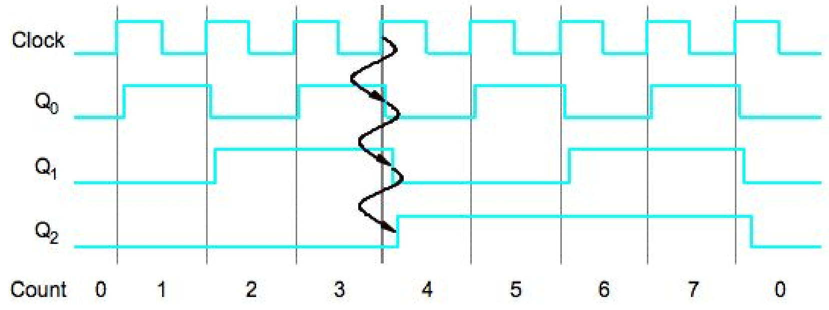
Counters

- Counters are important basic circuits that can be built from any type of flipflop.
  - T and D are most commonly used.
  - Asynchronous counters have delayed transitions similar to ripple carry adders.
- Synchronous counters have all flip-flops changing at once.
- Up counters count up from 0.
- Down counters count down from some value.

Three-Bit Up-Counter



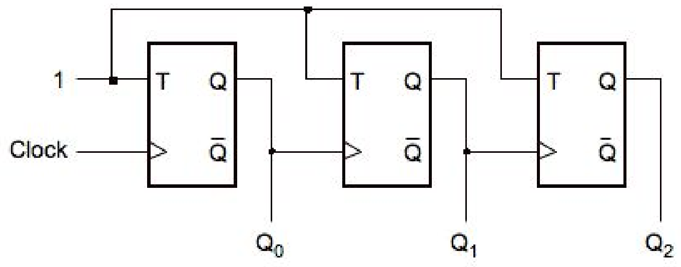
(a) Circuit



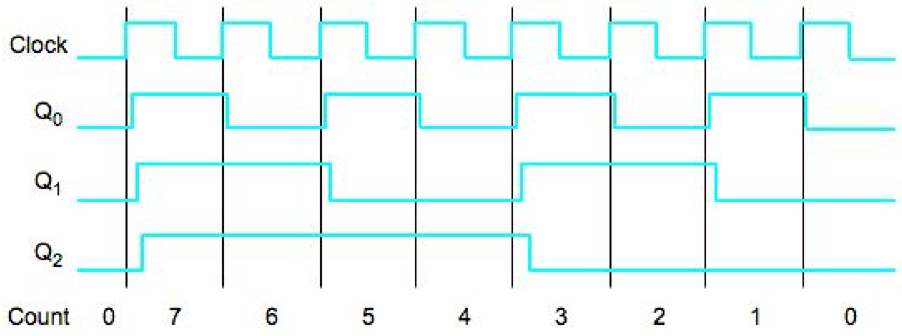
(b) Timing diagram

Figure 7.20. A three-bit up-counter.

Three Bit Down Counter



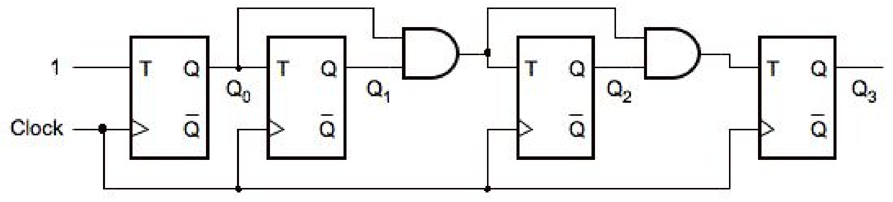
(a) Circuit



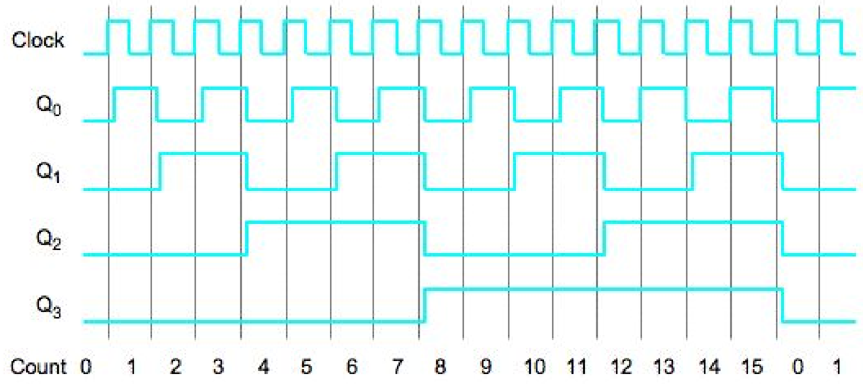
(b) Timing diagram

Figure 7.21. A three-bit down-counter.

Synchronous Counters



(a) Circuit



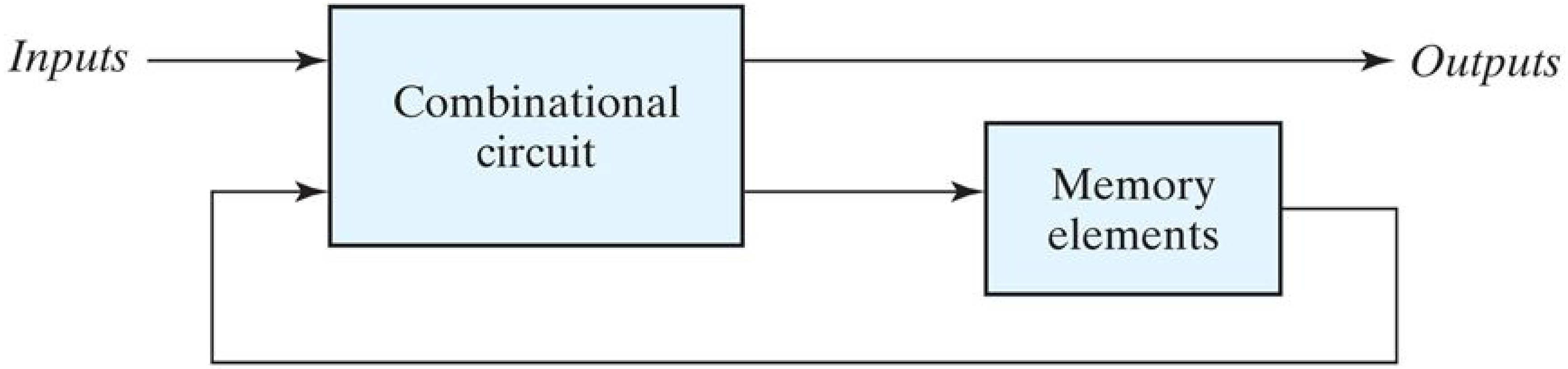
(b) Timing diagram

Figure 7.22. A four-bit synchronous up-counter.

General Form of a Sequential Circuit

- Primary inputs W (from outside world)
- Combinational logic (CL)
  - Inputs are W and flip-flop outputs (states)
- Flip-flops
- Inputs are CL outputs
- Second combinational logic
  - Inputs are W and FF outputs
  - Outputs are Z outputs (to outside world)

What does that look like?



Types of Sequential Circuits

- Moore
  - outputs depend only on FF states
  - no direct dependence on circuit inputs W
  - simpler to design
  - usually requires one or more extra states
- Mealy
  - outputs depend on FF states and circuit inputs W

Sequential Circuit Design Steps

1. Construct a state diagram.
2. Convert the state diagram to a state table.
3. Minimize the state table.
4. Assign flip-flop values to the states.
5. Construct a transition table.
6. Choose a desired FF type.
7. Determine the FF input and circuit output equations.
8. Draw the circuit diagram.

(We will usually not do this.)

Let’s Do An Moore FSM Example

Let’s look for the pattern of sequential ones for the input w

So if w= 1 for two or more times in a row the output, z=1, otherwise z=0

Let’s Make the “Improved” Logic Functions

General State Assignment Problem

- State assignments can substantially affect the cost of circuits.
  - You can use binary, gray or other patterns
- CAD tools can help find near optimum state assignments.
- Practical circuits have immense numbers of possible state assignments so manual solutions are impractical.

Register Example

Design the control circuit to swap the contents of R1 and R2.

1. Use R3 as temporary storage.
2. Load (R2) into R3.
3. Load (R1) into R2.
4. Load (R3) into R1.
5. Use w as input directing swap.
6. Use Done to signal completion.

State Diagram

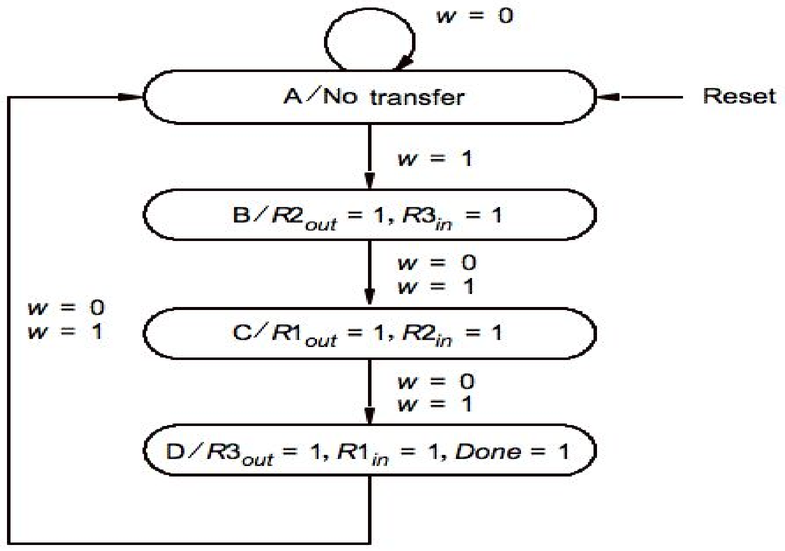

w = 0
A / No transfer
Reset
w = 1
B / $R2_{out}$ = 1, $R3_{in}$ = 1
w = 0
w = 1
C / $R1_{out}$ = 1, $R2_{in}$ = 1
w = 0
w = 1
D / $R3_{out}$ = 1, $R1_{in}$ = 1, Done = 1
w = 0
w = 1

Figure 8.11. State diagram.

Finding the Input and Output Equations

- Treat the transition tables as K-maps.
- This may be easier if you reorder the rows of the table into Grey code order.

Mealy FSM

- The Mealy model has circuit outputs that are functions of both
  - the circuit inputs and the flip-flop states.
  - increases flexibility and may require fewer flip-flops.

Let’s Revisit the Sequence Recognizer

- This was the first Example
- This was a sequence recognizer for 11.
- The timing will change as the sequence will be recognized as soon as the second 1 occurs rather than in the following clock cycle.

What changes

- The state diagram will change and will have one fewer state.
- Consequently, the state table will also have one fewer state.

What about the register swap example?

- We can redo the register swap control circuit example as Mealy.
- Only three states, rather than four, are required.
- Still two flip-flops are required.

Mealy FSM State Diagram for Register Swap

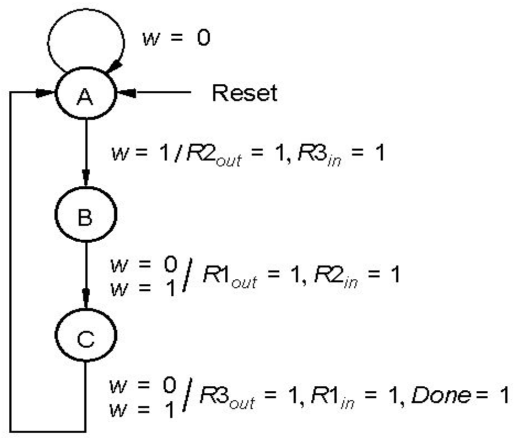

w = 0
Reset
A
w = 1 / $R2_{out}$ = 1, $R3_{in}$ = 1
B
w = 0 / w = 1 / $R1_{out}$ = 1, $R2_{in}$ = 1
C
w = 0 / w = 1 / $R3_{out}$ = 1, $R1_{in}$ = 1, *Done* = 1

Figure 8.28. State diagram for Example 8.4.

More on Moore and Mealy

- How can you tell if a circuit is Moore or Mealy?
- Moore circuits have outputs that depend only on flip-flop states, not directly on circuit inputs.
  - The state diagram has the output in the state circle.
  - The state table has a special column for output.
- Mealy circuits have outputs that depend on flip-flop states and circuit inputs.
  - The state diagram shows the output on the transition arrow.
  - The state table shows outputs in input columns.

Asynchronous Circuits- A High Level Quick Cover

Why?

It is part of sequential circuits and it really is just an extension of the base concepts so you need to have some awareness of it

So how much Asynchronous will be on the exam?

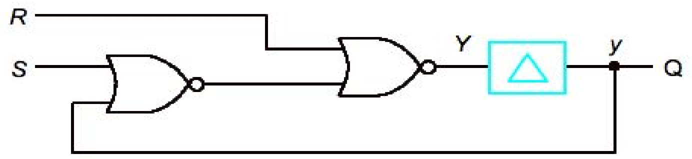
You need to know what the difference between synchronous and asynchronous is, including how to recognize the difference

Asynchronous Behaviour

- Some asynchronous basics:
  - There is no clock pulse to synchronize the circuits here.
  - There are no flip-flops for state variables.
  - To get reliable behavior, circuits must receive their inputs in a nice, controlled manner.
  - We'll assume just one input changes at a time.
  - Moreover, input changes must occur respecting propagation delay so that the circuit is stable before another input arrives.
  - Circuits adhering to these constraints are operating in fundamental mode.

An Asynchronous SR Latch

- The little Δ indicates an arbitrary time delay.
    - The NOR gates are ideal and have no delay.
- Y is the next state, y is the present state.
- Whenever Y = y, the circuit is stable.
    - Stable states are indicated by the little circle.
- Consider the input sequence SR = 00, 01, 11, 01. When does the circuit stabilize?



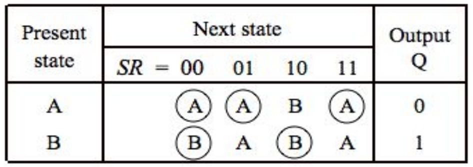
(b) Circuit with modeled gate delay

Present state	Next state			
	SR = 00	01	10	11
y	Y	Y	Y	Y
0	$\textcircled{0}$	$\textcircled{0}$	1	$\textcircled{0}$
1	$\textcircled{1}$	0	$\textcircled{1}$	0

(b) State-assigned table

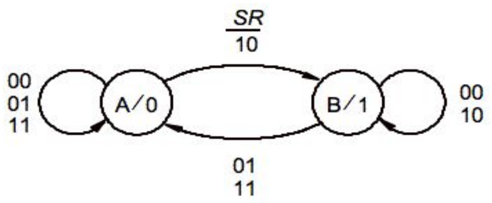
Turning that into an FSM

- Going backwards from circuit to state machine.
- Let A be when y = 0 and B be when y = 1.
- Notice how we’ve notated the two inputs to the state machine.
- One would be led to believe synchronous and asynchronous circuits have the same design path; they don’t.
    - Asynchronous circuits are much harder.



Present state	Next state				Output Q
	SR = 00	01	10	11	
A	Ⓐ	Ⓐ	B	Ⓐ	0
B	Ⓑ	A	Ⓑ	A	1

(a) State table



(b) State diagram

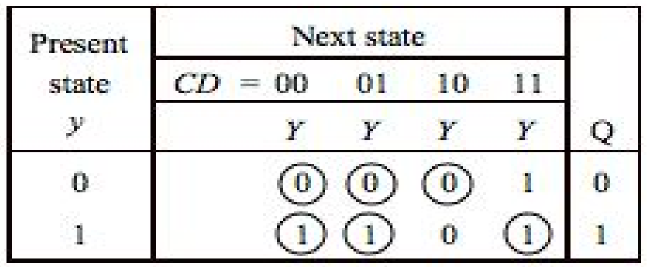
Figure 9.2. FSM model for the SR latch.

Some More Terminology

- One the previous slides, the terminology from our synchronous circuits was used, but that's not common. Perform the following substitutions:
  - State Table = Flow Table
    - We flow from one state to another.
  - Transition Table = Excitation Table
    - Depicts transitions of state variable by exciting the next-state variables.

Gated D Latch Redux

- Let’s redo a gated D latch, treating the clock as just another random input.
- Let’s build it up. Shown at right is the excitation table.
    - Remember, C = Clock,   D = Data.
- The minimal function is: Y=CD+C’y
- The best function is: Y=CD+C’+Dy
    - It’s better because it’s hazard-free.

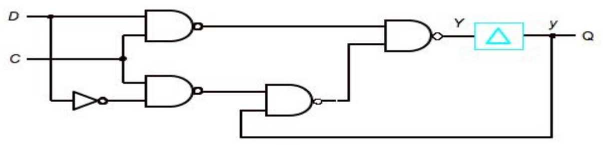


Present state	Next state				
	CD = 00	01	10	11	
y	*Y*	*Y*	*Y*	*Y*	Q
0	⓪	⓪	⓪	1	0
1	①	①	0	①	1

(b) Excitation table

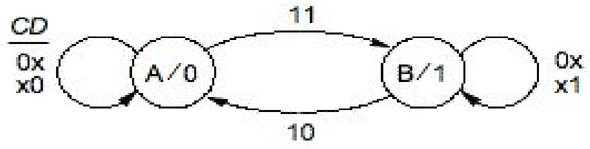
More on that D Latch

Here’s the circuit, flow table and state diagram for the D latch.



Present state	Next state CD = 00	01	10	11	Q
A	A	A	A	B	0
B	B	B	A	B	1

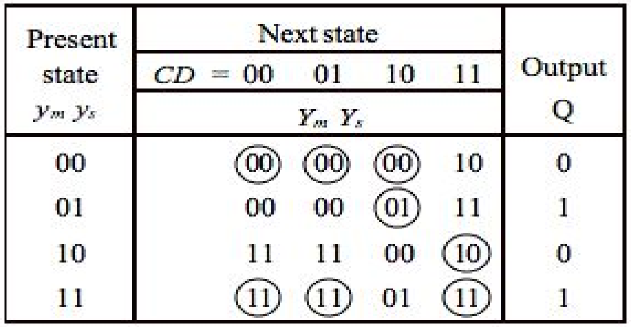
(c) Flow table



(d) State diagram

D Flip-Flop Asynchronously

- This is the master-slave flip-flop.
- Y_m and y_m are the state of the master, Y_s and y_s are the state of the slave.
- C and D are the same as before.
- Let’s walk through this excitation table and try to follow it.

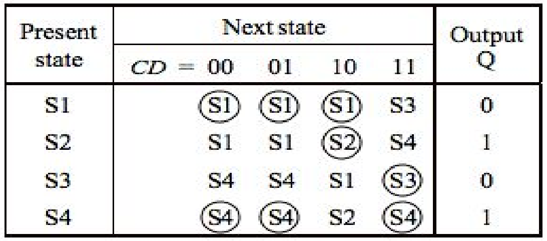


Present state	Next state				Output
	CD = 00	01	10	11	
$y_m\ y_s$	$Y_m\ Y_s$				Q
00	$\boxed{00}$	$\boxed{00}$	$\boxed{00}$	10	0
01	00	00	$\boxed{01}$	11	1
10	11	11	00	$\boxed{10}$	0
11	$\boxed{11}$	$\boxed{11}$	01	$\boxed{11}$	1

(a) Excitation table

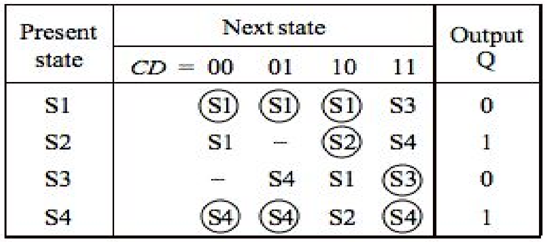
D Flip-Flop Flow Tables

- The top flow table is a pretty direct translation of the excitation table.
  - But consider inputs CD = 01 in state S1 and CD = 00 in state S3. Are these meaningful?
  - No.
- The bottom flow table removes entries that can't happen because only one input can change at a time.



Present state	Next state CD = 00	01	10	11	Output Q
S1	(S1)	(S1)	(S1)	S3	0
S2	S1	S1	(S2)	S4	1
S3	S4	S4	S1	(S3)	0
S4	(S4)	(S4)	S2	(S4)	1

(b) Flow table



Present state	Next state CD = 00	01	10	11	Output Q
S1	(S1)	(S1)	(S1)	S3	0
S2	S1	–	(S2)	S4	1
S3	–	S4	S1	(S3)	0
S4	(S4)	(S4)	S2	(S4)	1

(c) Flow Table with unspecified entries

State Diagram

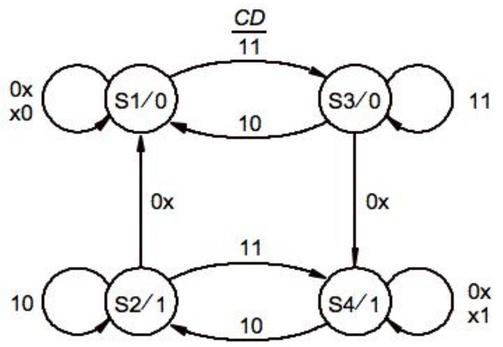


Figure 9.7. State diagram for the master-slave D flip-flop.

Synthesis of Asynchronous Circuits

- Devise a state diagram from the textual description of the required behavior.
- Derive the flow table.
- Reduce states if possible.
- Perform state assignment.
- Derive the excitation table.
- Obtain the next-state and output expressions.
- Draw the circuit (optional).

What makes it hard?

- When creating the diagram (or flow table), we have to make sure the circuit is stable before we try to output anything.
- Minimization is non-trivial.
- State assignments are not just done with the intention of minimizing the combinational logic.
    - We have to avoid unreliable state assignments.