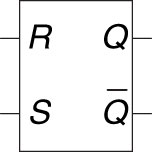
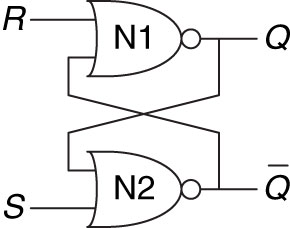
Lesson 14 – Introduction to Synchronous Circuits

Combinational logic – Outputs depend on current inputs

Sequential logic – Outputs depend on current and prior inputs (🡪 i.e. has memory)

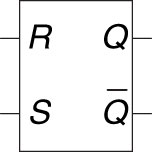
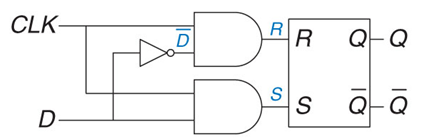
**SR Latch**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | **S** | **R** | **Q** |  | | **1** | **0** | **1** | **0** | | **0** | **0** | **Qprev** |  | | **0** | **1** | **0** | **1** | | **0** | **0** | **Qprev** |  | | **1** | **1** | **0** | **0🡪** | | S = Set  R = Reset  Q = Stored Value  **🡺** **Undefined…next state unknown** | Q changes whenever an input changes |

Problem with SR latch: Biggest problem is when S + R are both 1

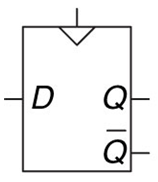
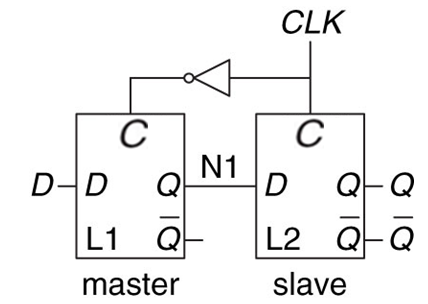
**D Latch - Solves the problem of the SR Latch when both S & R are asserted**



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | **C** | **D** | **Q** |  | | **0** | **X** | **Qprev** |  | | **1** | **0** | **0** | **1** | | **1** | **1** | **1** | **0** |   Problem with D latch: What if we don’t always want D to update? | C = Control/Clock  D = Data  Q = Stored value | transparent – if C is high, Q will follow input (D) (i.e. D flows through)  opaque – if C is low, Q won’t change (i.e. D doesn’t flow through)  Q changes whenever C is high and D changes |

**D Flip-Flop – Two D-Latches back to back with complementary clocks in a Master/Slave Config**

D-Latch updates it’s state continuously while clk = ‘1’

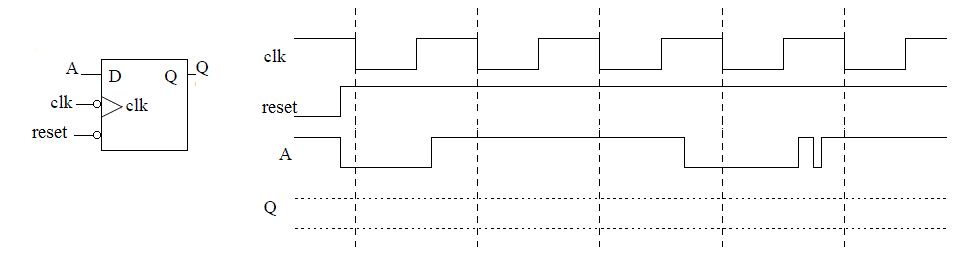
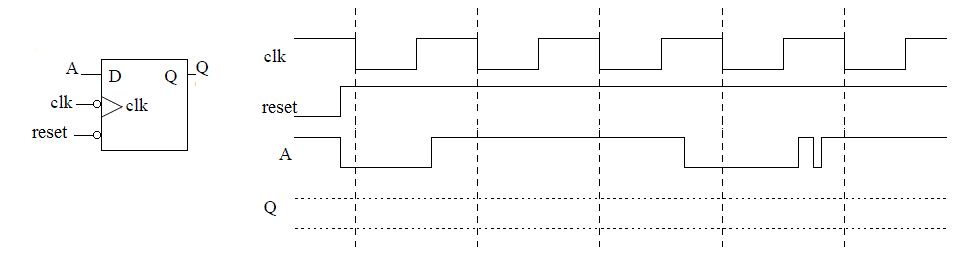
  

Copies D 🡪 Q on Rising Edge of Clock and remembers it’s state on all others

1 🡪0

0 🡪1 Rising Edge of Clock

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | **CLK** | **D** | **Q** |  | | 0 | 0 | **Qprev** |  | | 0 | 1 | **Qprev** |  | | 1 | 0 | **Qprev** |  | | 1 | 1 | **Qprev** |  | | ↑ | 0 | 0 | 1 | | ↑ | 1 | 1 | 0 | | CLK = Clock  D = Data  Q = Stored Value  🡪Called an edge-trigged device | Comprised of: Two latches, a master and a slave  Q changes when clock changes  Enabled flip-flop – Add EN signal to determine if data will load on edge  Resettable flip-flop – resets output to 0 when reset enabled. Useful for hearing known states |

****

FF

Latch

**\*What is the difference between a latch and a flip flop?** A Flip Flop changes on the clock edge a latch changes at any time

Register – An N-Bit Register is a Bank of N-Flip Flops

Synchronous circuit – Combinational circuit followed by bank of flip flops

Properties: a) Each element is either a register or a combinational circuit

b) At least one circuit is a register

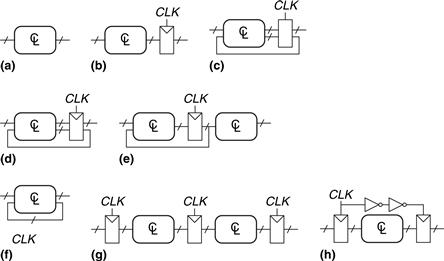
c) All registers have the same clock

d) Every cyclic path contains at least one register

Example: FSMs, Pipelines

Asynchronous circuit – Timing not limited by clock registers

**Example 3.5: Synchronous Sequential Circuits**



**Solutions**

**Circuit:**

* (a) is combinational, not sequential, because it has no registers.
* (b) is a simple sequential circuit with no feedback.
* (c) is neither a combinational circuit nor a synchronous sequential circuit, because it has a latch that is neither a register nor a combinational circuit.
* (d) and (e) are synchronous sequential logic; they are two forms of finite state machines, which are discussed in Section 3.4.
* Same as (d)
* (f) is neither combinational nor synchronous sequential, because it has a cyclic path from the output of the combinational logic back to the input of the same logic but no register in the path.
* (g) is synchronous sequential logic in the form of a pipeline, which we will study in Section 3.6.
* (h) is not, strictly speaking, a synchronous sequential circuit, because the second register receives a different clock signal than the first, delayed by two inverter delays.