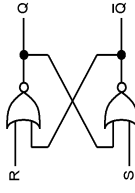


## Storing Data

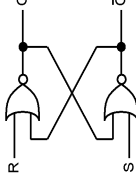
- So far, we've only covered circuits that give an output value, based on a logical combinations of inputs.
- How can these circuits implement data storage?
  - Answer: Use feedback to perpetuate an output value.

• Example:



- A circuit like this is called a **latch**, or a **flip-flop**.

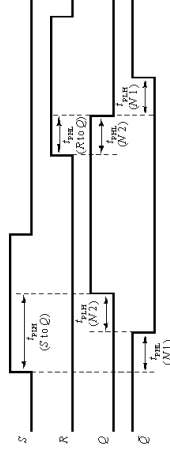
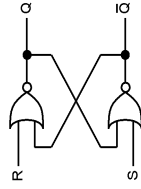
## SR Latches



S	R	Q	Q̄
0	0	1/0	1/0
0	1	0	1
1	0	1	0
1	1	0	0

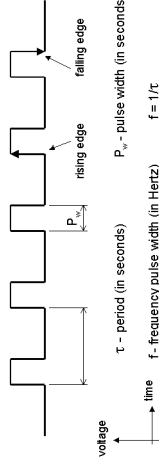
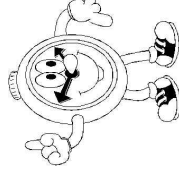
- By examining this circuit's behaviour, we see that it exhibits certain characteristics, based on the inputs it receives:
  - When R and S are 0, it maintains the previous output values.
  - When S is 1, Q is set to 1 and Q is set to 0.
  - When R is 1, Q is set to 0 and Q is set to 1.
  - S and R should not be set to 1 at the same time. However, when they are, both outputs are cleared.
- Because of this behaviour, S and R are generally labelled as **set** and **reset**.

## SR Latches



- The change in output is not instantaneous.
  - **Timing diagrams** illustrate **propagation delays**.
- This can be problematic in some cases:
  - Other circuits whose timing depends on this circuit's output
  - SR goes from 11 to 00 → what happens?

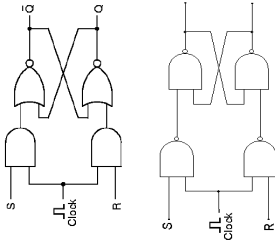
## Synchronous Circuits



- In order to make sure that timing isn't an issue, use a signal to tell the flip-flop when to change.
- This signal is called a **clock**, and produces a high pulse at regular intervals
  - duration of clock pulse (**pulse width**) must be long enough to allow all propagation delays to settle down.
  - time between clock pulses (**period**) must be long enough for the rest of the circuit to process the new flip-flop output value.

## Sequential Circuits

- A latch whose output can only change when the clock pulse rises is called a **gated latch**.



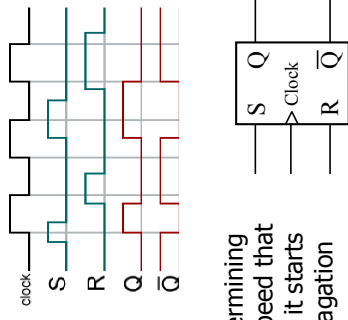
Clock	S	R	Q	Q
0	X	X	1/0	1/0
1	0	0	1/0	1/0
1	0	1	0	1
1	1	0	1	0
1	1	1	0	0

CSC258 Lecture Slides © Steve Engels, 2006

Slide 5 of 12

## Sequential Circuits

- This implementation ensures that the output values for the latch are stable when processing occurs.
- When a computer's processor speed is described, one of the determining factors is the maximum speed that the clock can have before it starts overlapping with the propagation delays.

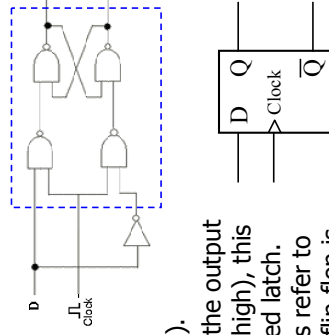


CSC258 Lecture Slides © Steve Engels, 2006

Slide 6 of 12

## The D Flip-Flop

- A common variation on the SR flip-flop is the D flip-flop.
- It is based on a regular SR flip-flop, but with a single input (see diagram).
- Because of its simplicity (the output is high when the input is high), this is the most commonly-used latch. Whenever hardware types refer to a latch or flip-flop, the D flip-flop is generally assumed.

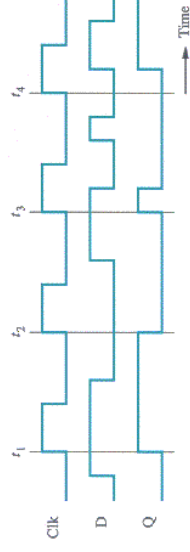


CSC258 Lecture Slides © Steve Engels, 2006

Slide 7 of 12

## The D Flip-Flop

- Simplified behaviour eliminates odd case where S and R are both high.
- Timing diagram for D flip-flop:

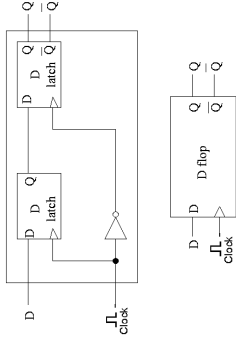


CSC258 Lecture Slides © Steve Engels, 2006

Slide 8 of 12

## Master-Slave Flip-Flop

- The problem with these designs is that we assume that the inputs to S, R & D will not change while the clock pulse is high. This is a BAD assumption. Most circuits that use flip-flops feed the outputs of one flip-flop into the input of another, or even back into itself!
- Solution:** The **master-slave** flip-flop
  - One latch is activated on the rising clock edge (storing input), and the other is activated on the falling clock edge (propagating output).

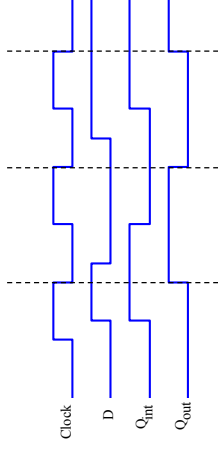


CSC258 Lecture Slides © Steve Engels, 2006

Slide 9 of 12

## Master-Slave Flip-Flop

- Timing diagram:



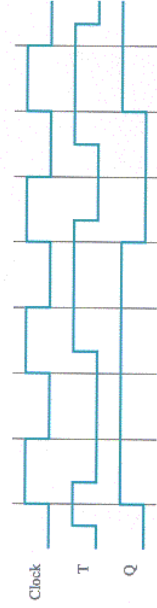
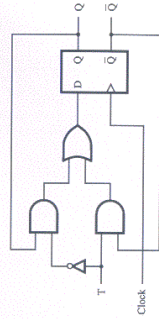
- Notice how the output to the slave is delayed until it can no longer influence the output of the master.

CSC258 Lecture Slides © Steve Engels, 2006

Slide 10 of 12

## Other Flip-Flops

- The T Flip-Flop:**
  - Like the D flip-flop, except that it toggles its value whenever the input to T is high.

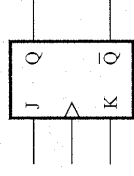
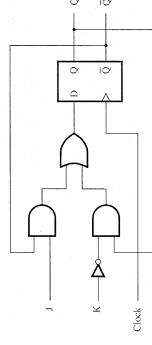


CSC258 Lecture Slides © Steve Engels, 2006

Slide 11 of 12

## Other Flip-Flops

- The JK Flip-Flop:**
  - Takes advantage of all combinations of two inputs (J & K) to produce four different behaviours:
    - if J and K are 0, maintain output.
    - if J is 0 and K is 1, set output to 0.
    - if J is 1 and K is 0, set output to 1.
    - if J and K are 1, toggle output value.



CSC258 Lecture Slides © Steve Engels, 2006

Slide 12 of 12