Flip-Flops and Registers

This lecture is an introduction to flip-flops and registers.
After this lecture you should be able to:

- *convert between high/low logic levels and true/false truth values for active-high and active-low interfaces*
- *predict the relationship between the input and output waveforms of SR and D flip-flops*
- *identify specifications on a timing diagram*
- *identify a specification as a requirement or guaranteed response*
- *predict relationship between register, shift register and counter input and output waveforms*
- *write Verilog descriptions for these registers*

Numbers, Logic Levels and Truth Values

Numbers are used for counting, logic levels are voltages, and truth values can be true or false. These are different, but related.

Almost universally, 1 and 0 are synonymous with true and false respectively. But there are two common conventions for the relationship between logic levels and truth values:

Number	Truth	Active High	Active Low
`0`	`F`	`L`	`H`
`1`	`T`	`H`	`L`

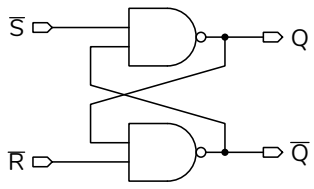
Verilog uses the active-high convention: `0` and `1` are treated as low and high respectively.

Active-low signals can be denoted by:

- a bar over the signal name ($\overline{\text{reset}}$)
- an asterisk after the signal name (`RESET*`)
- a suffix of N (or n) after the signal name (`reset_n`)

Set-Reset Latch

The following circuit "remembers" which input was last active:

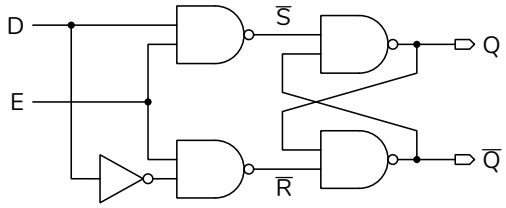


When both $\overline{S}$ and $\overline{R}$ are high the gates act as inverters, outputs are complementary, and they remain unchanged. But if one input is brought low, that output is forced high and the other is forced low.

When both inputs are low, both outputs are high and the condition that Q is the complement of $\overline{Q}$ is not true. The first input to go high will set or reset the output.

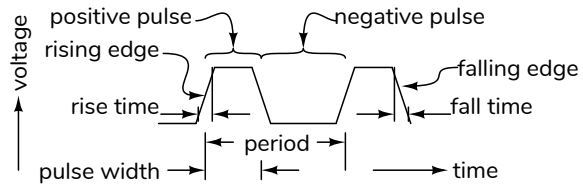
Transparent Latch

We can extend an SR latch by adding an inverter that sets $\overline{S}$ and $\overline{R}$ to complementary levels and two NAND gates to prevent state changes unless an E(nable) signal is asserted:



This is a "transparent latch": the output follows the input when E (enable) is high, but the value is held when the enable is low.

Digital Waveforms



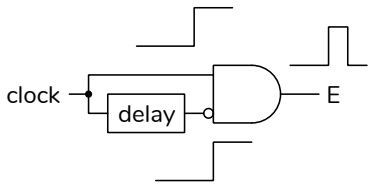
A transition from low to high is called a rising edge. The time it takes is called the rise time. A transition from high to low is called a falling edge. The time it takes is called the fall time. Two adjacent edges define a pulse, which can be negative or positive. The time between these is called the pulse width.

Rise and fall times are typically measured between 10% and 90% of the swing. Other measurements are typically made between 50% levels. But there are exceptions.

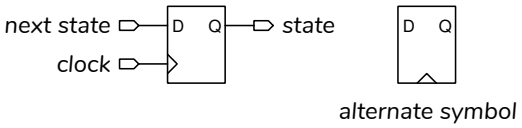
A signal consisting of periodic pulses is a clock. The inverse of the clock period is the clock frequency.

D Flip-Flop

An "edge-triggered" D (delay) flip-flop can be constructed from a transparent latch by adding a circuit that applies a short pulse to the Enable input on each rising edge of a clock input:

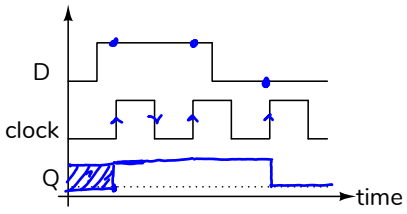


The symbol for the D flip-flop is:



The rising edge of a clock input causes the flip-flop to store the value of the input and make it available on the output. Thus the D flip-flop has a next-state input (D), a state output (Q) and a clock input. The D flip-flop's state only changes on the rising edge of the clock.

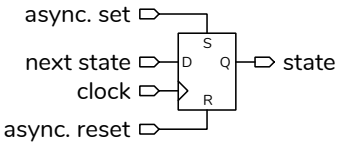
The D flip-flop "stores" one bit and is the memory element used in most sequential logic circuits.



Exercise 1: Fill in the waveform for the `Q` signal in the diagram above.

Asynchronous Inputs

Asynchronous inputs:



can be used to set (to H) or reset (clear to L) the outputs independently of the clock. Asynchronous signals are typically used to reset the circuit to a known state when a circuit is powered up or when a failure is detected.

Timing Specifications

Timing specifications can be:

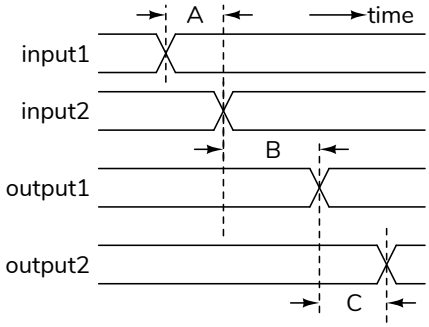
requirements: the manufacturer requires that these specifications be met for a device to operate properly, or

guaranteed responses: the manufacturer guarantees that these specifications will be met.

Since the device manufacturer cannot control the timing on inputs but can guarantee the timing of outputs, a simple rule to distinguish requirements from responses is:

- requirements are measured from a transition on an input *to a transition on an input*.
- guaranteed responses are measured from a transition on an input or an output *to a transition on an output*.

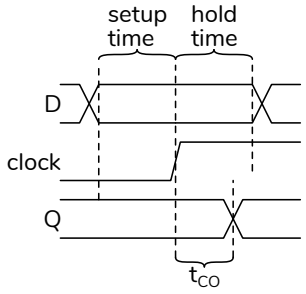
Timing diagrams show the relationship between transitions on inputs and outputs. They are not drawn to scale. Both high and low levels are shown when the specification applies to both.



Exercise 2: Label the specifications A through C as requirements or guaranteed responses.

Combinational logic circuits and D flip-flops have one important timing specification their propagation delay. This is the delay from a change on an input to the corresponding change on an output. The usual symbols are t_{PD} for propagation delay and t_{CO} for the clock-to-output delay for a D flip-flop.

D flip-flops have two important timing requirements:



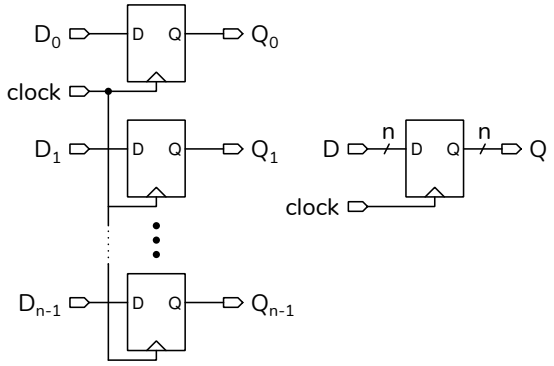
setup time the D input must be at a valid logic level for at least t_S before the rising edge of the clock

hold time the D input must remain a valid logic level for at least t_H after the rising edge of the clock

Timing analysis, be covered later, involves determining whether the timing requirements of the devices in a circuit will be met.

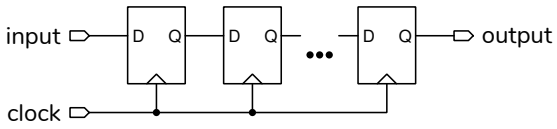
Common Sequential Logic Circuits

A *register* is several D flip-flops with their clocks tied together so that all the flip-flops are loaded simultaneously. The notation below shows the register inputs and outputs are n bits wide.



Exercise 3: What would be another name for a 1-bit register?

A *shift register* is several registers with the output of each register connected to the input of an adjacent register:

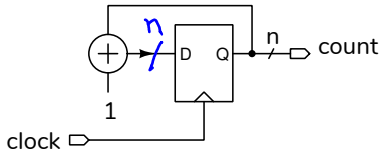


On each rising edge of the clock the state of each register is transferred to the next. This allows the data shifted in at one "end" of the shift register to appear at the other end after a delay equal to the number of stages in the shift register.

A shift register can also be used to make serial[1] data available in parallel by shifting the data into the shift register and then accessing the values at the flip-flop outputs.

Exercise 4: Add the parallel outputs to the shift register diagram.

A *counter* is a register whose value increases by 1 with each clock. It consists of a register whose input is the sum of one plus the current value of the count:



Registers in Verilog

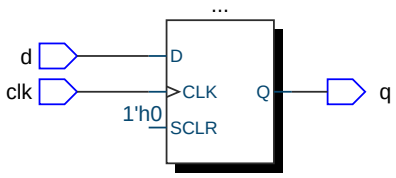
The following Verilog:

```
module ex2 (input logic d, clk,
            output logic q) ;

   always_ff @(posedge clk) begin
      q <= d ;
   end

endmodule
```

synthesizes a D-flip-flop that transfers the `d` input to the `q` output on the rising (positive) edge of `clk`:



Exercise 5: What would you change to make an 8-bit register? A 3-bit (deep) shift register? A 4-bit counter?

[1] One bit or word at a time.