

RTL Timing

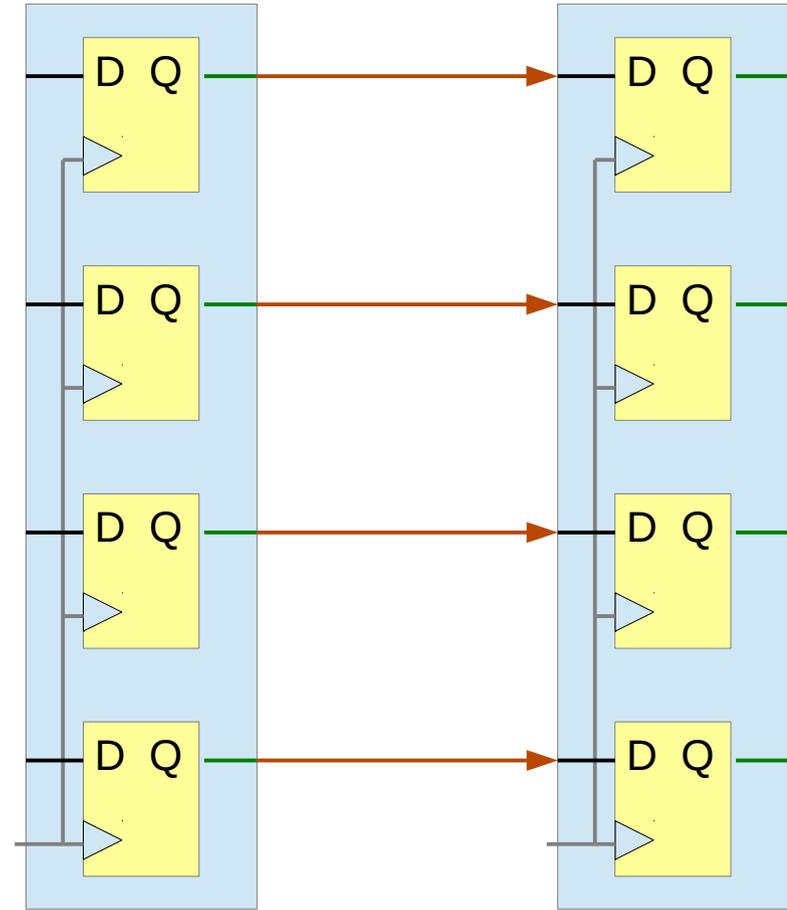
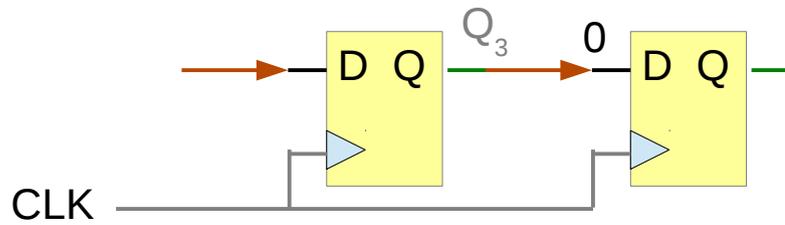
Copyright (c) 2011 - 2015 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

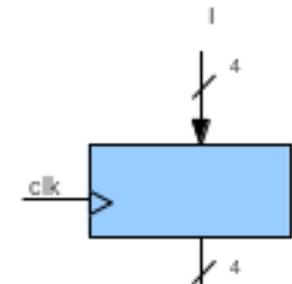
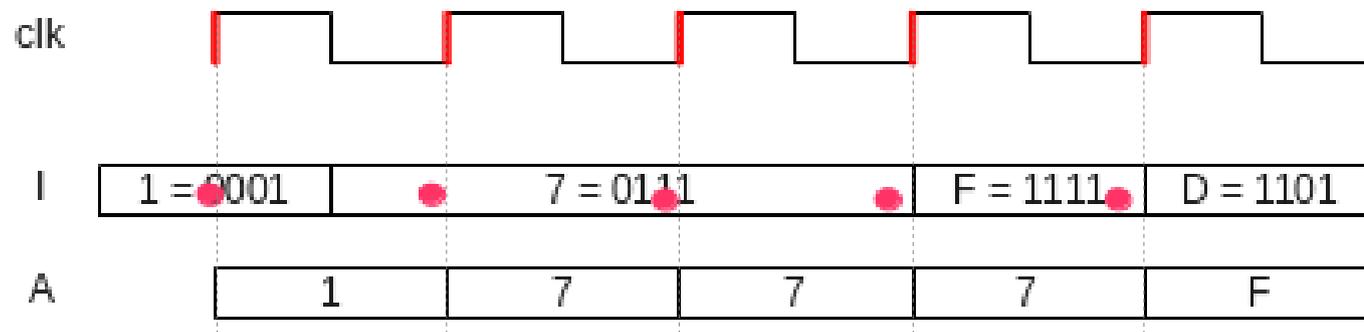
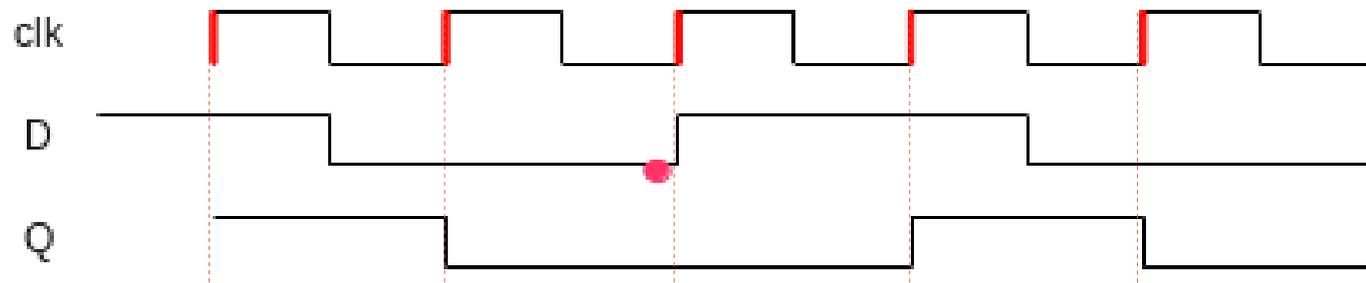
Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Shift Register v.s. Pipeline Stage Register

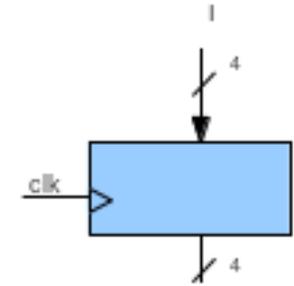
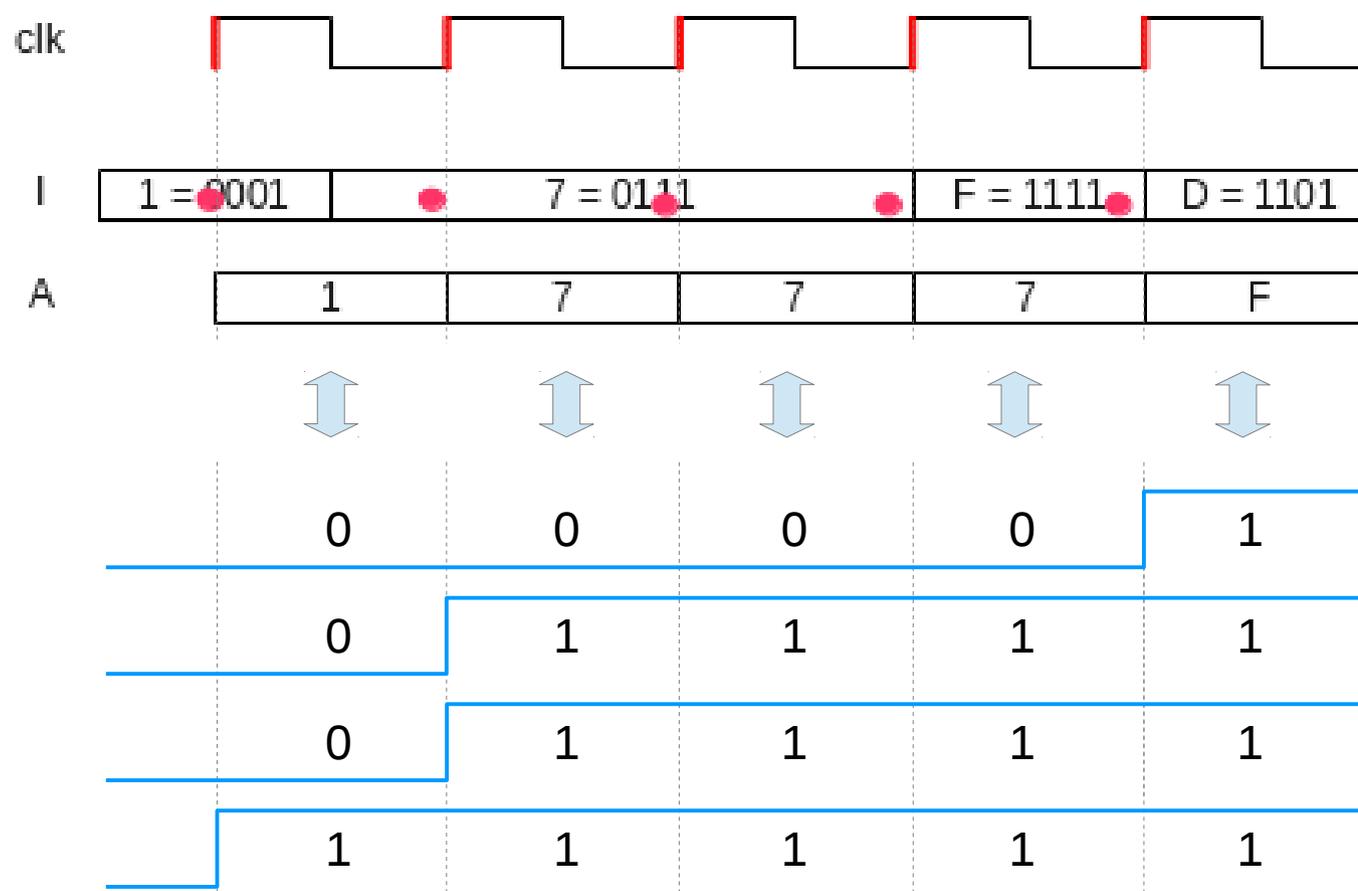


FF and Register Timing



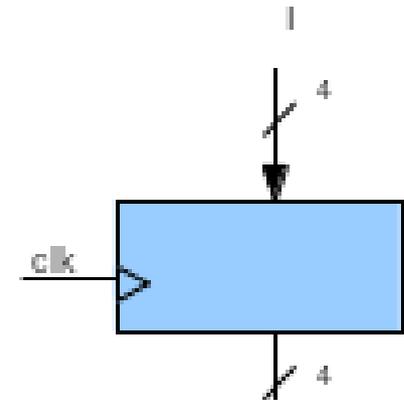
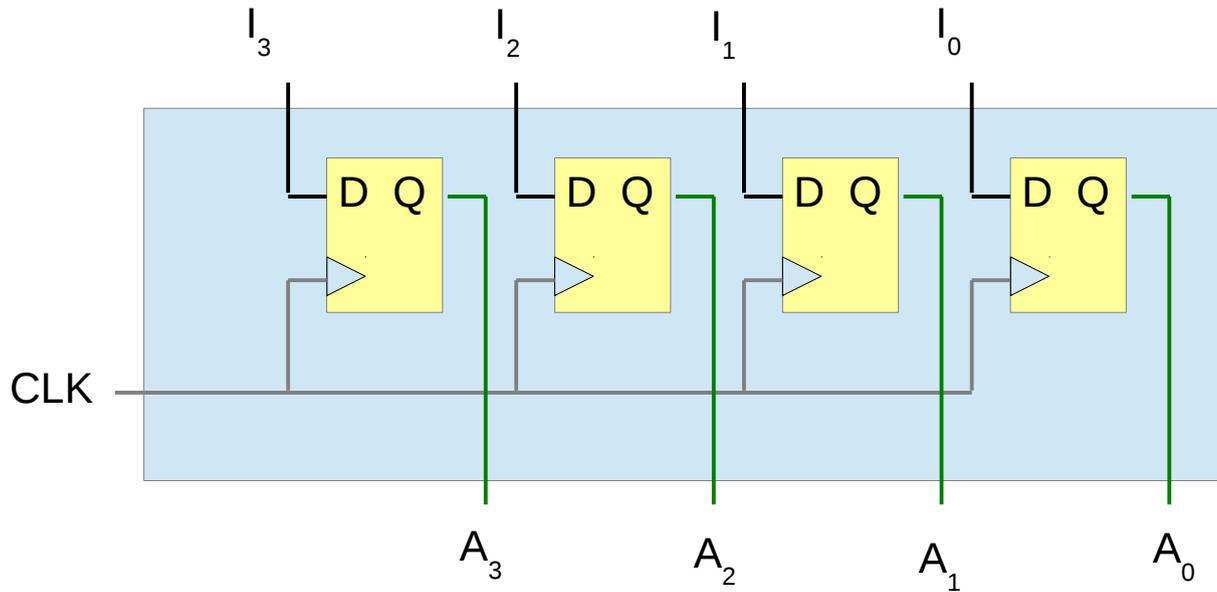
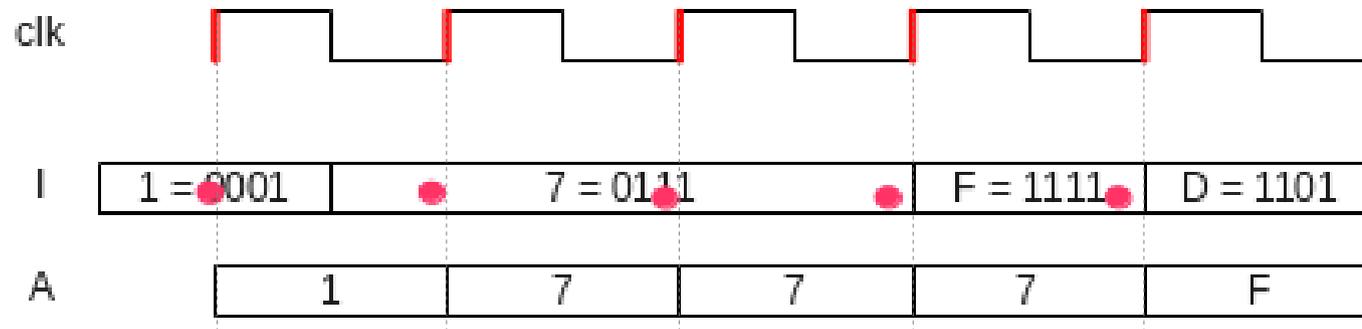
Decimal

Bus Notation



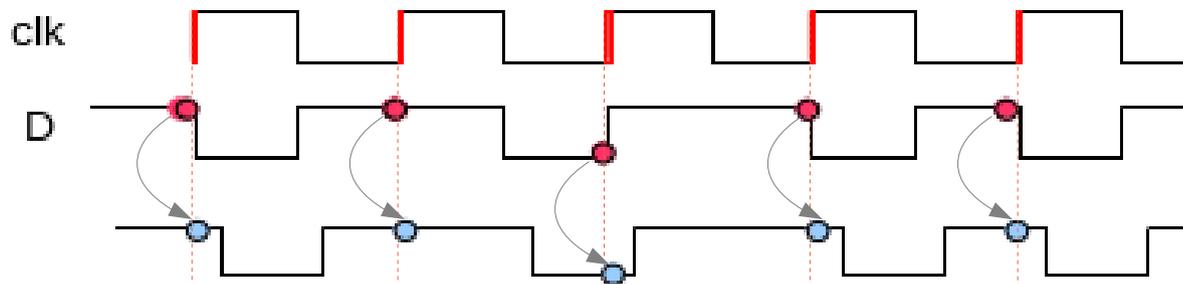
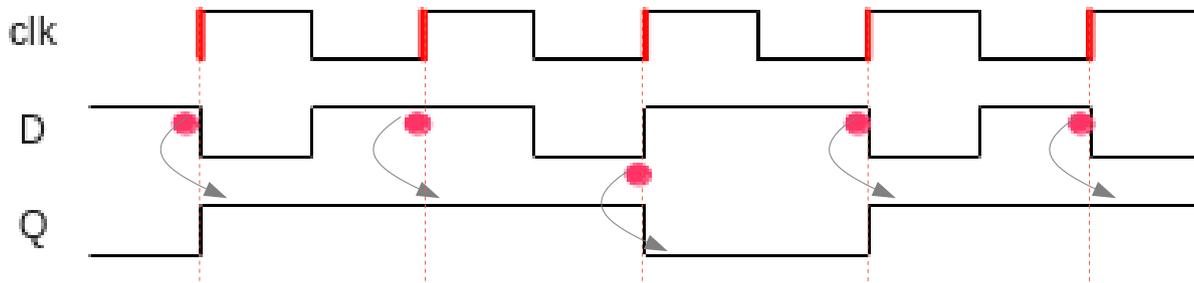
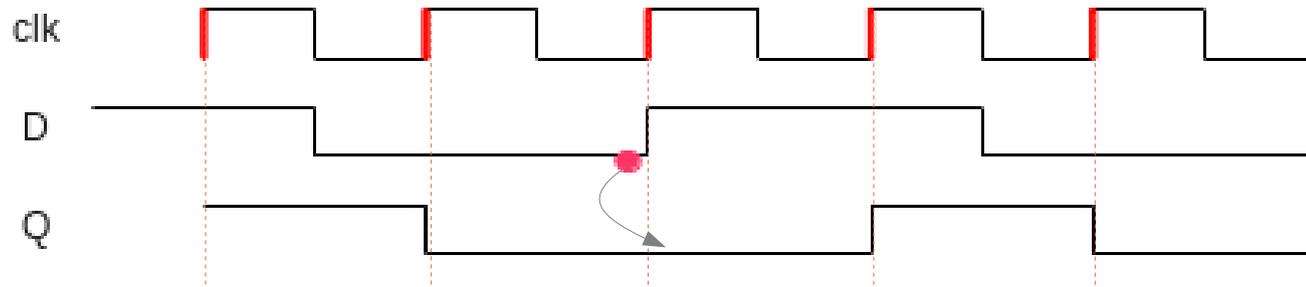
Decimal

Register



Decimal

FF Timing

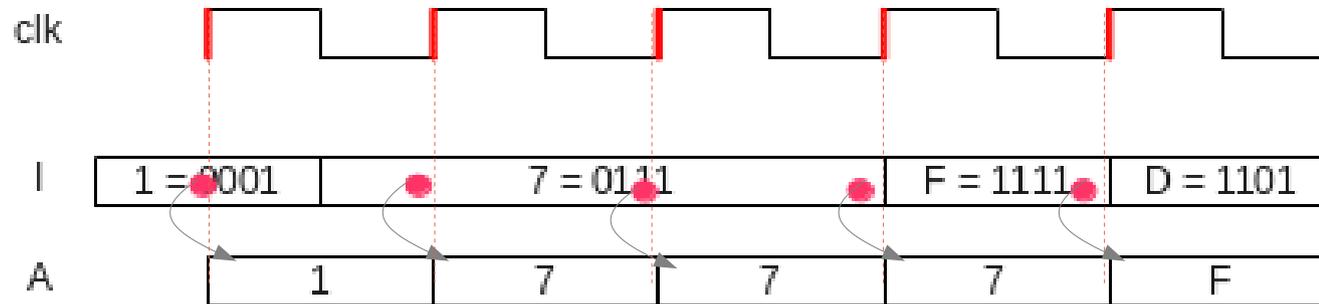


input signal with a delay ignored (ideal case)

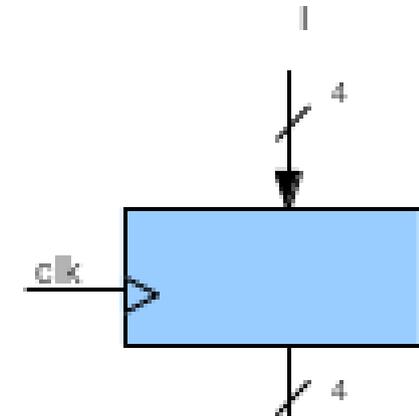
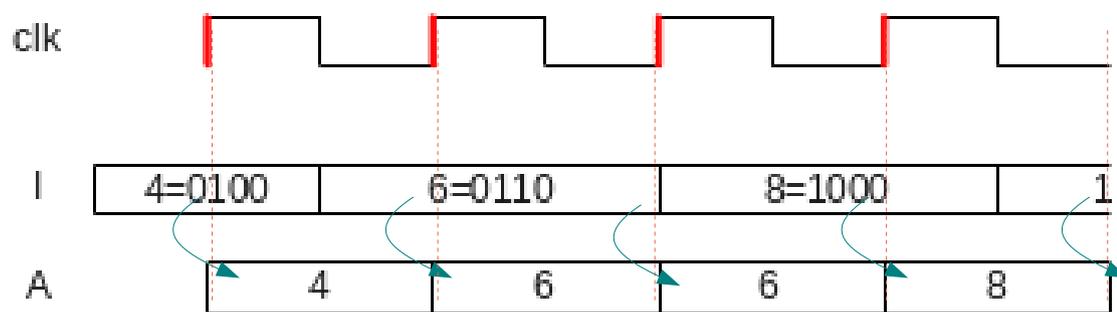
input signal with a delay explicitly shown

Register Timing

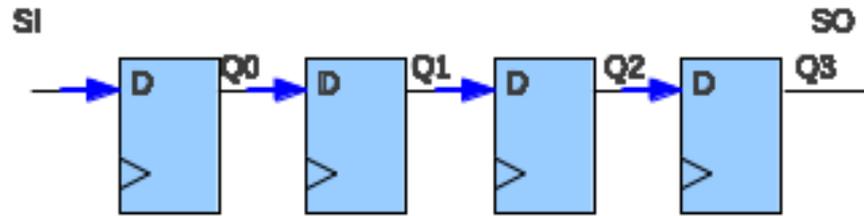
input signal with a delay explicitly shown



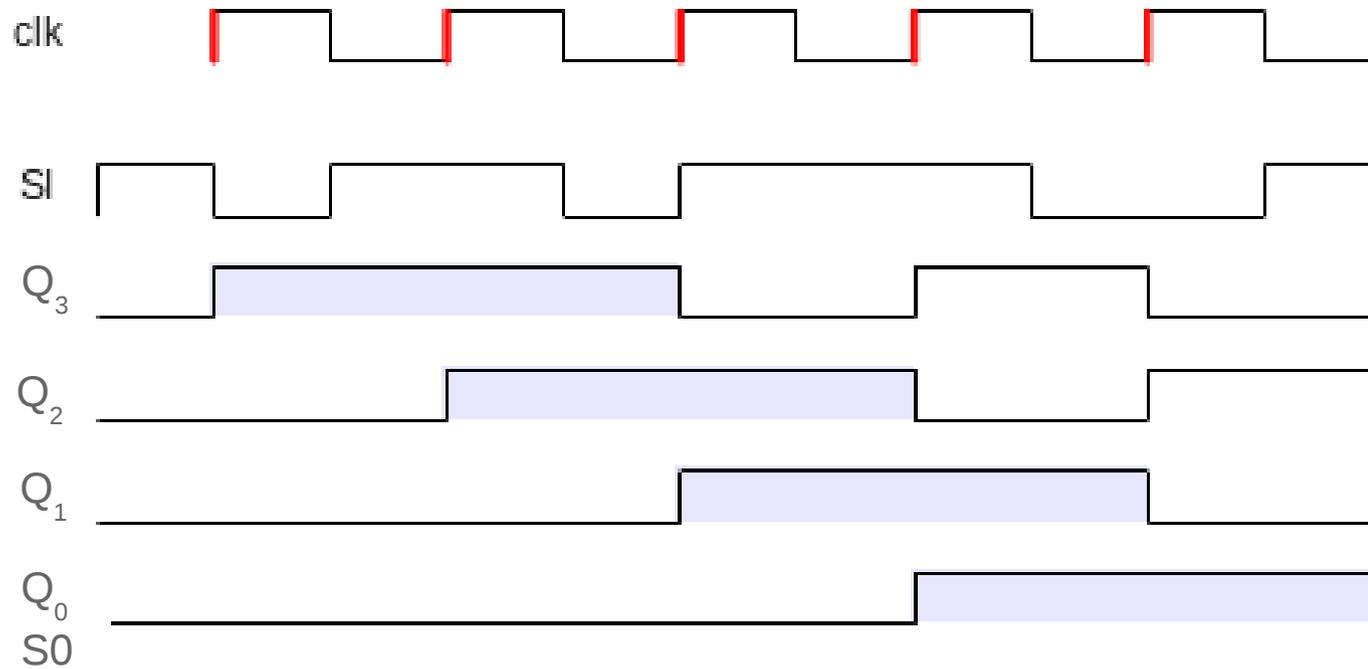
input signal with a delay explicitly shown



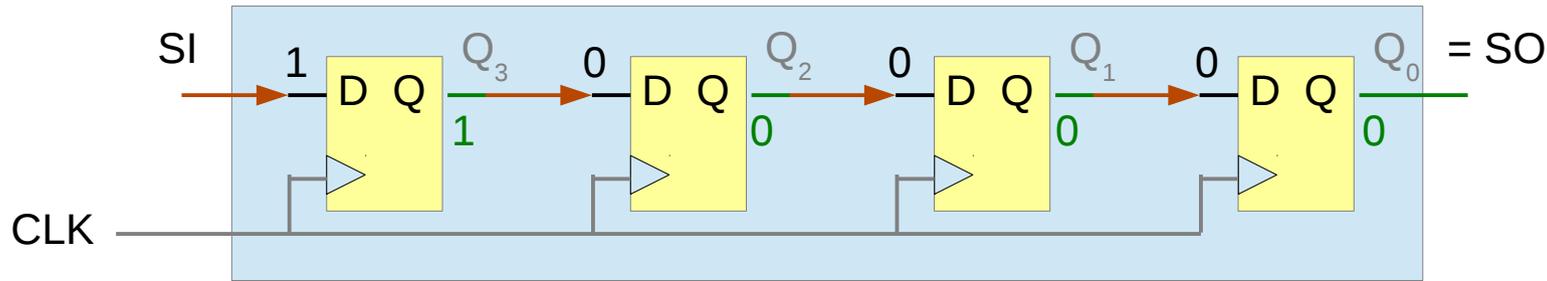
Shift Register Timing



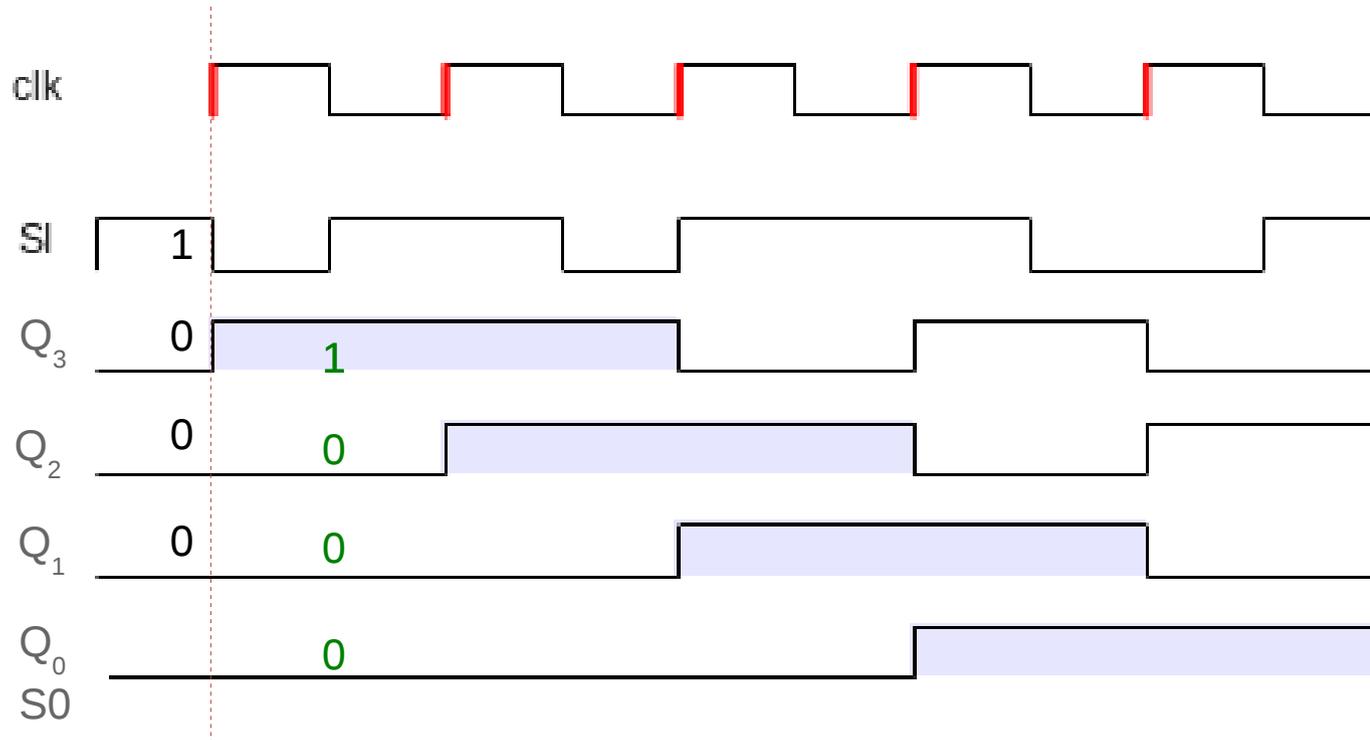
Connected in serial,
but parallel assignments



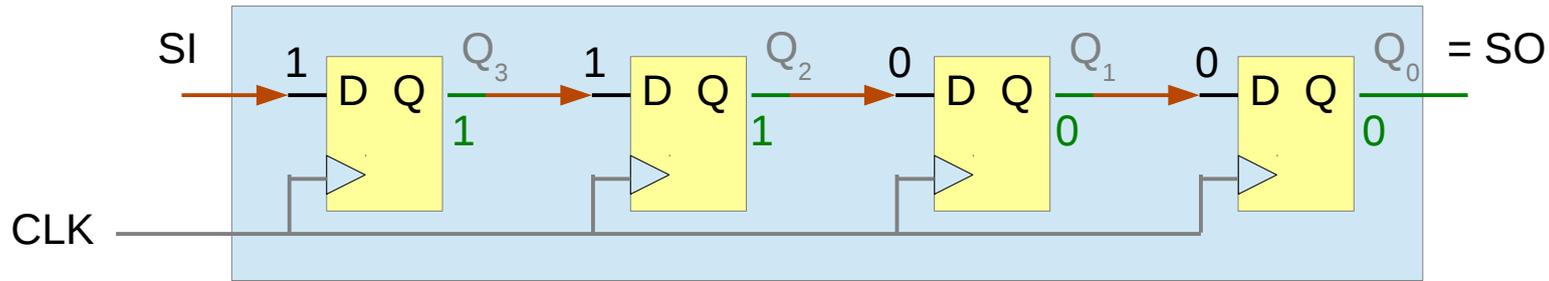
Shift Register Timing - Cycle 1



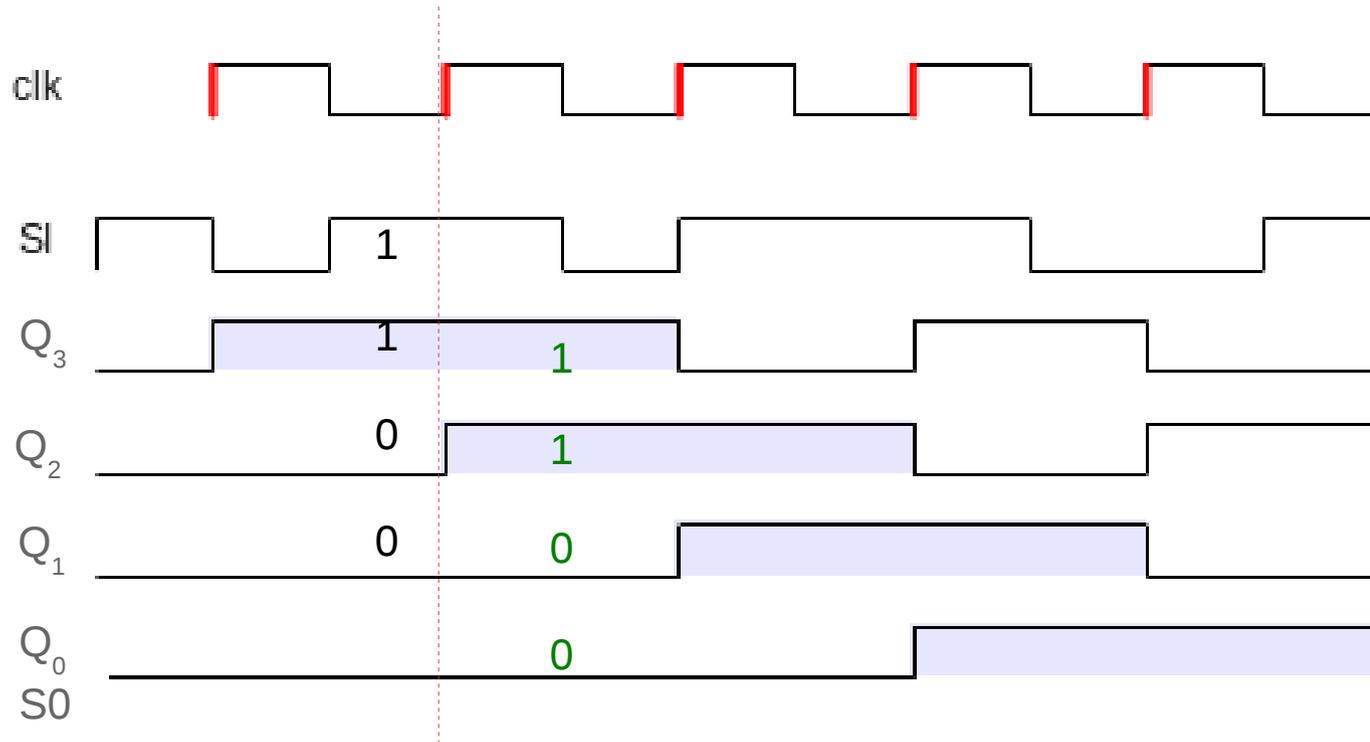
Connected in serial,
but parallel assignments



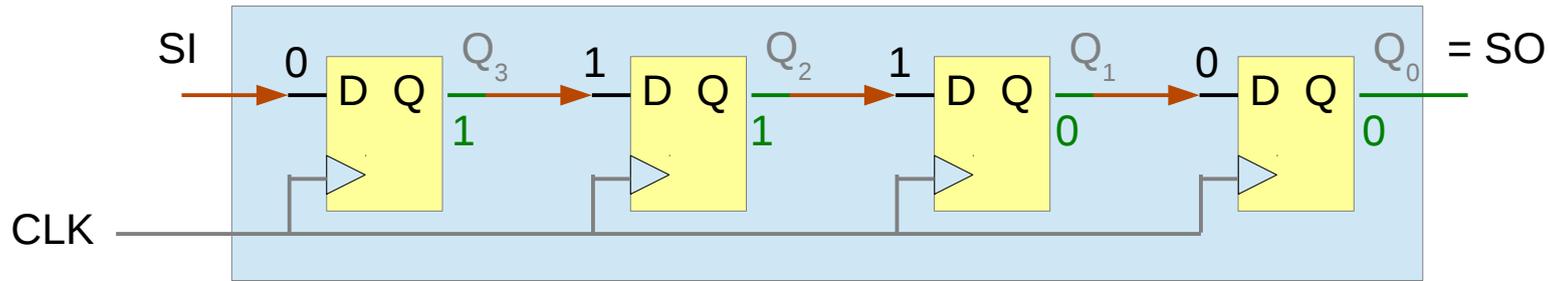
Shift Register Timing - Cycle 2



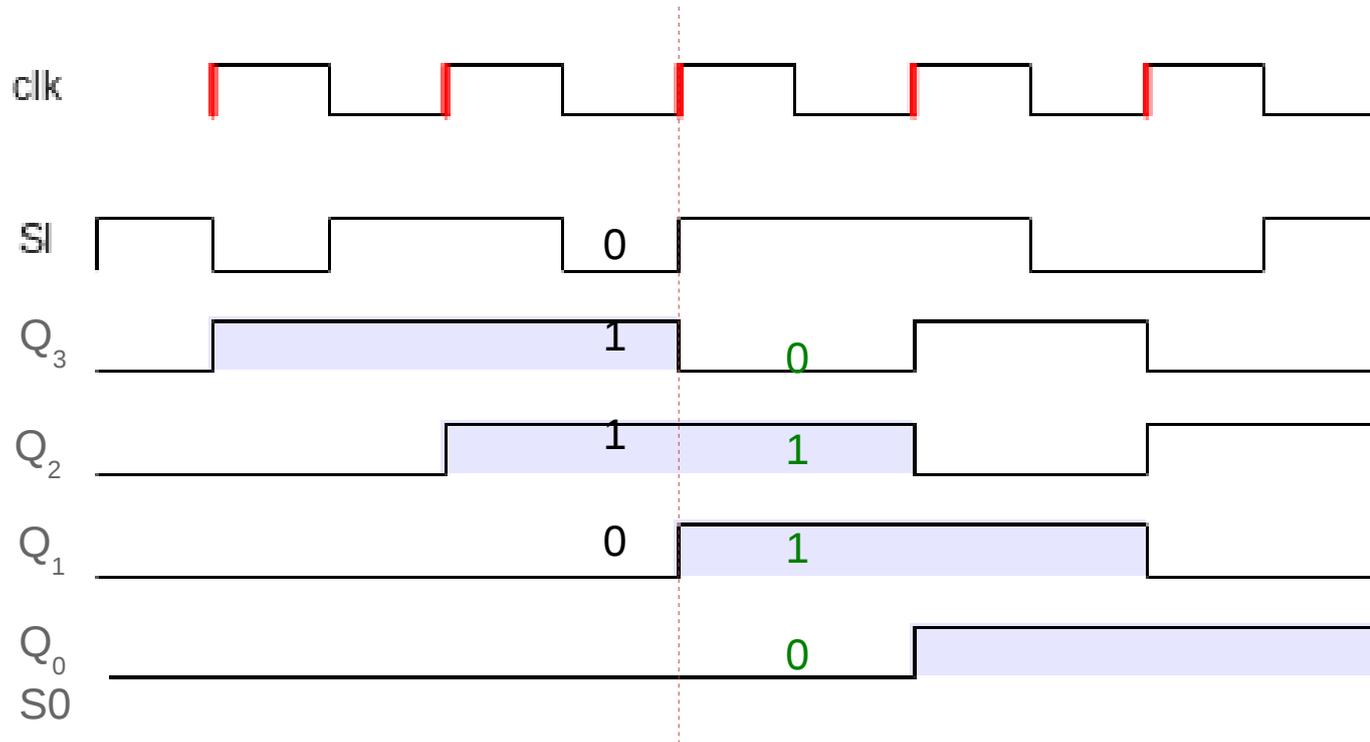
Connected in serial,
but parallel assignments



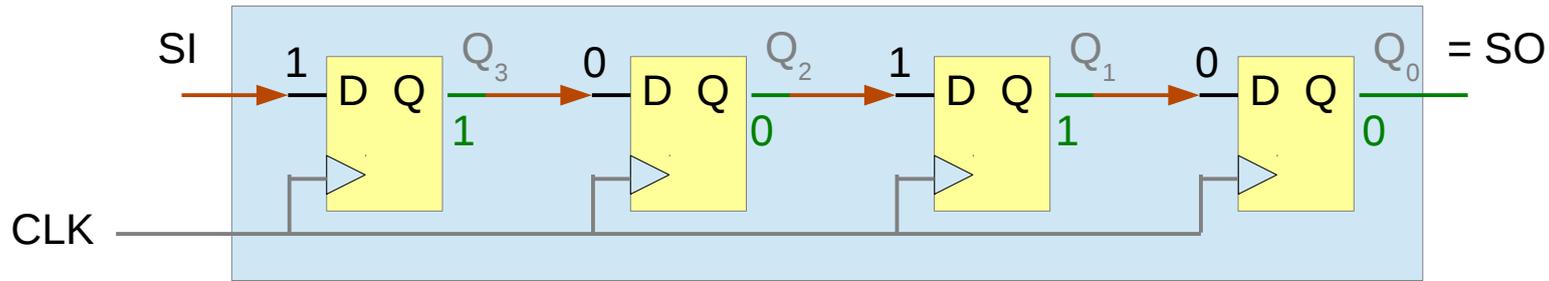
Shift Register Timing - Cycle 3



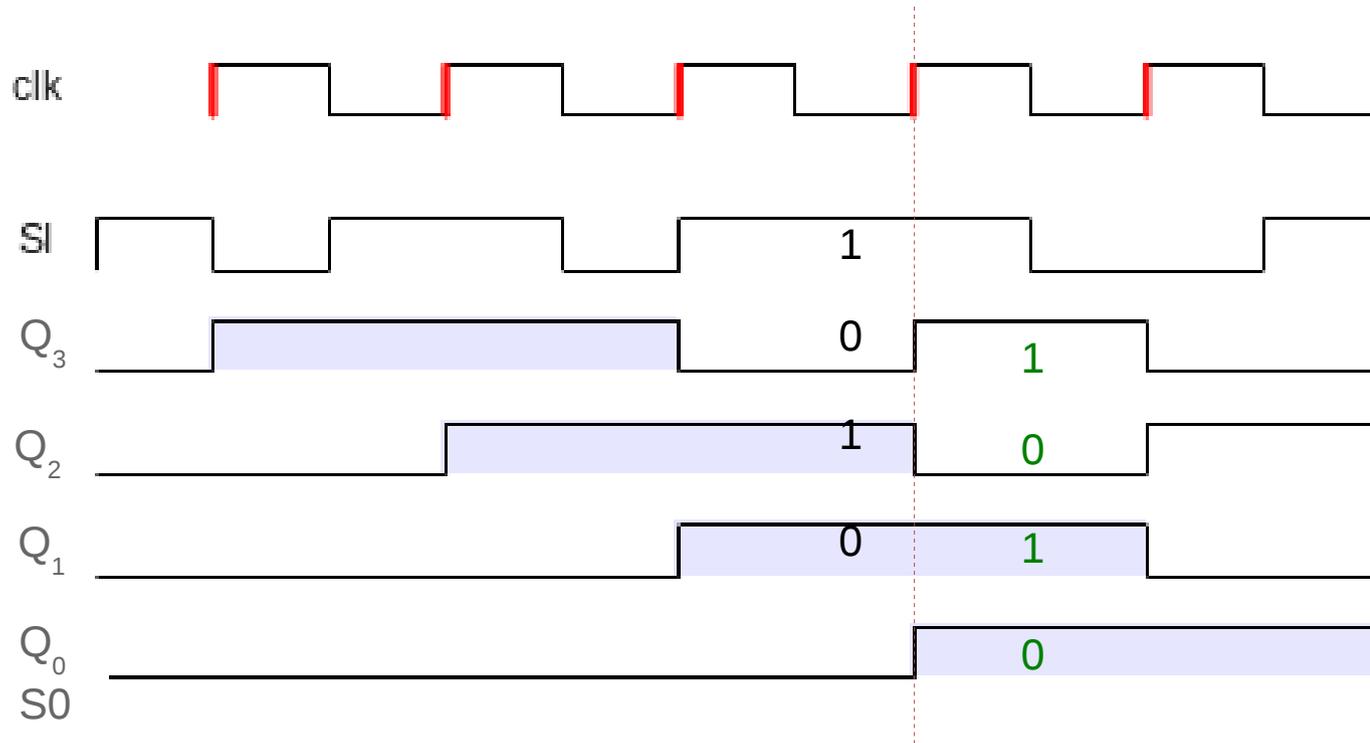
Connected in serial,
but parallel assignments



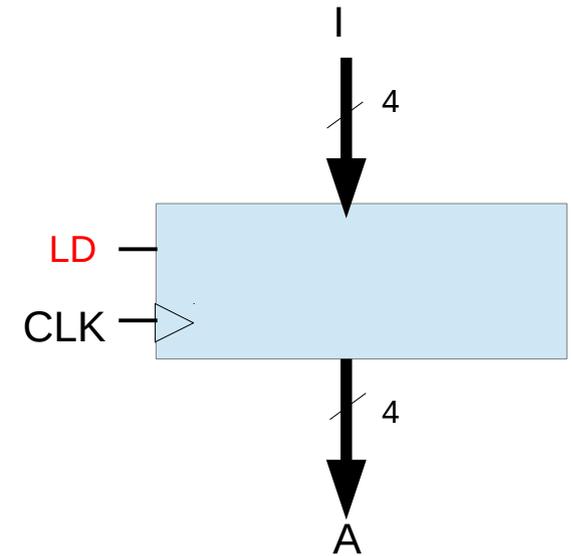
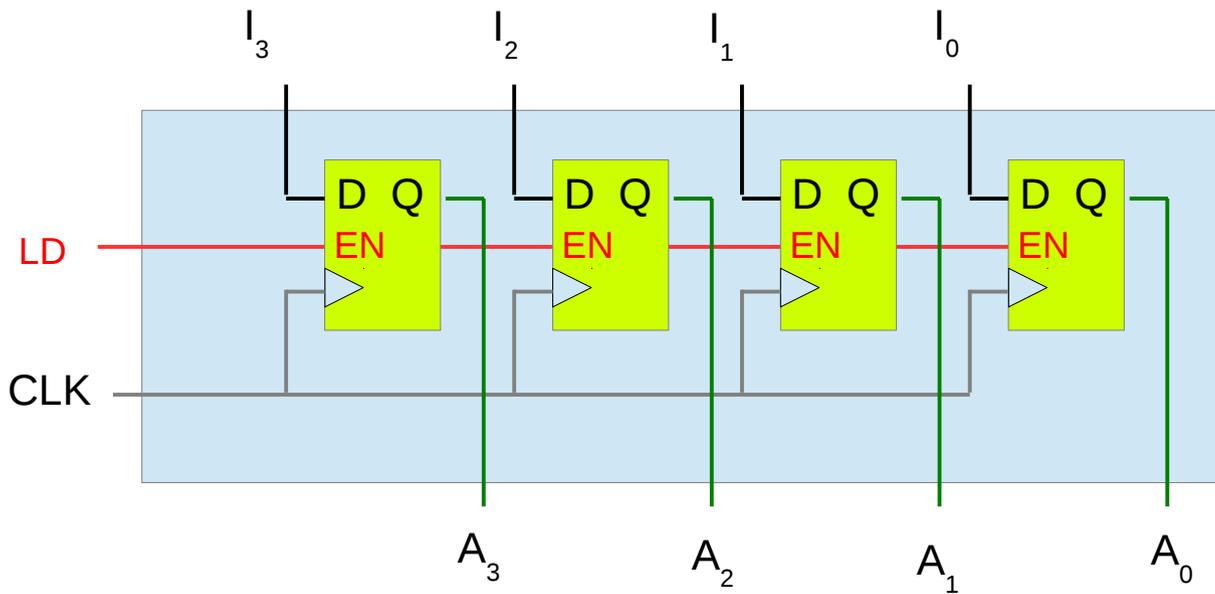
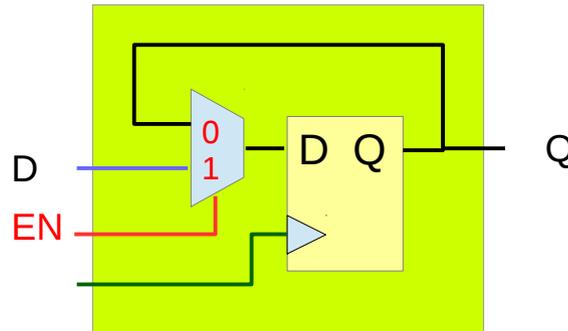
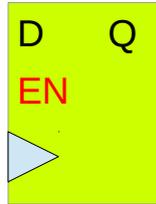
Shift Register Timing - Cycle 4



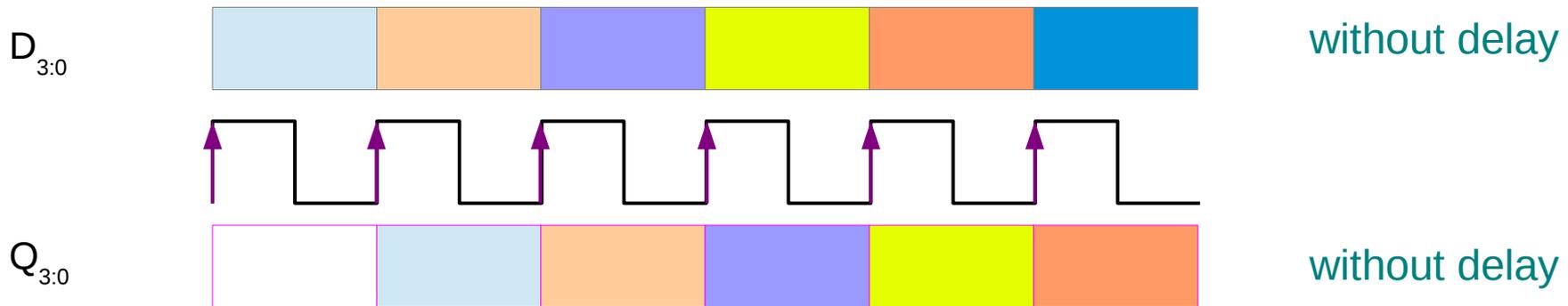
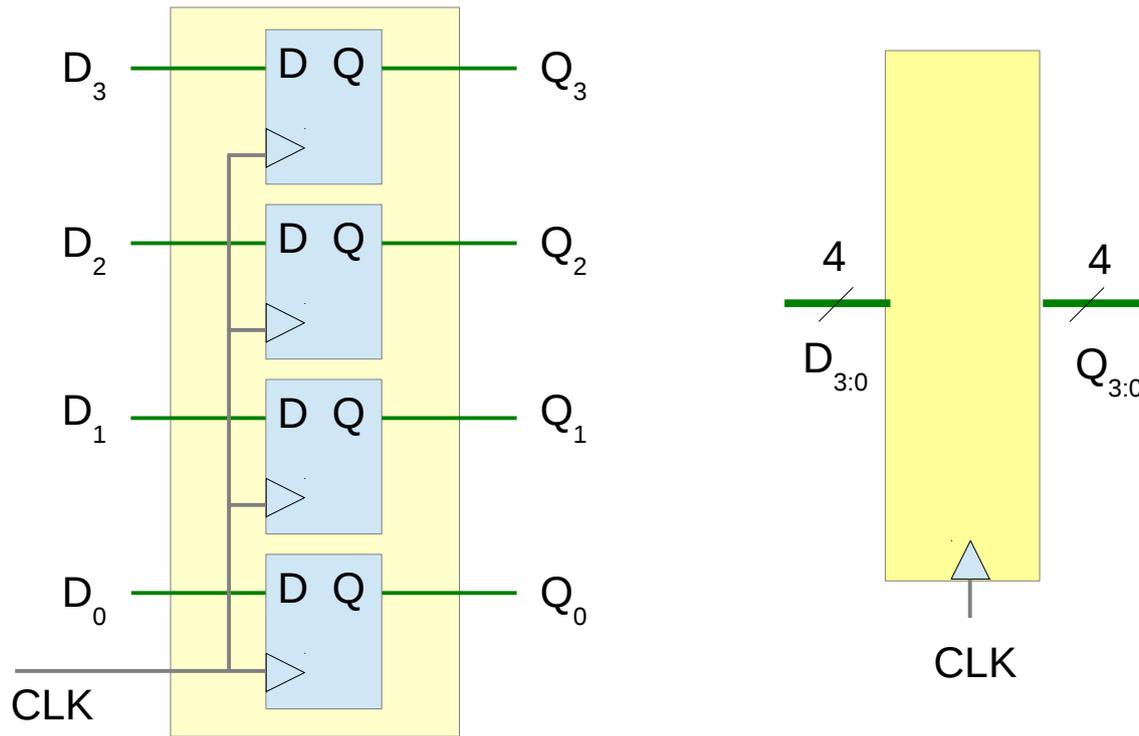
Connected in serial,
but parallel assignments



Register with Parallel Load

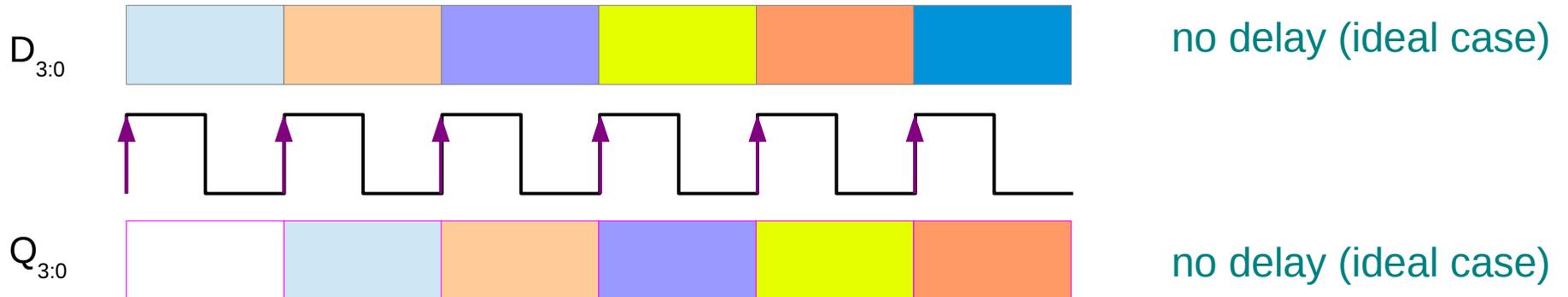


Register

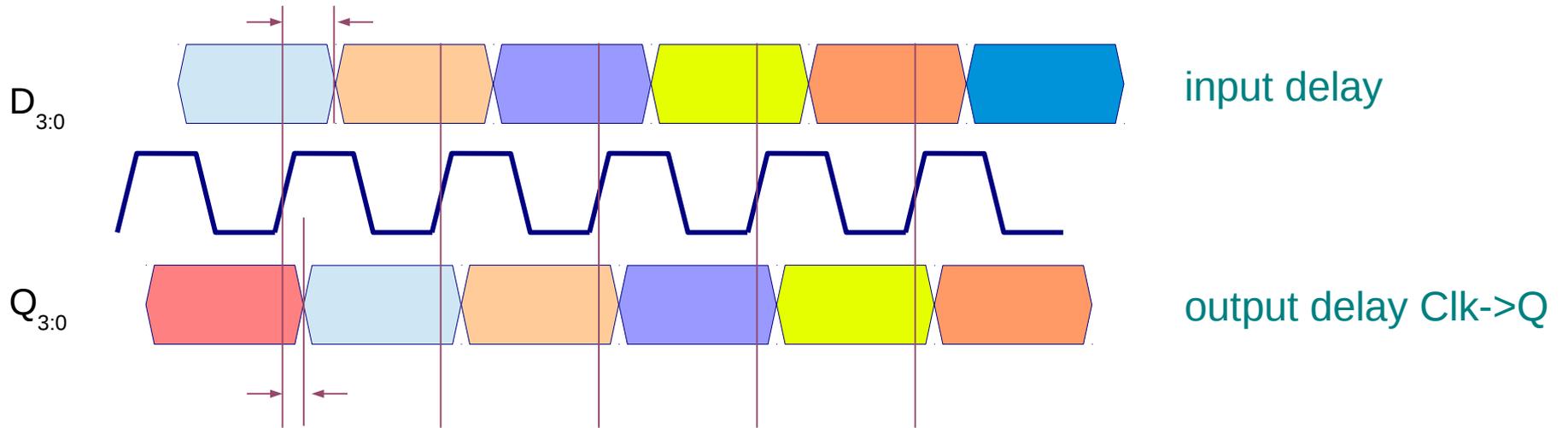


Types of Timing Diagrams

a timing diagram without delays



a timing diagram with delays



DFF Testbench

```
module dff(d, clk, rst, q, qb);
  input d, clk, rst;
  output q, qb;
  reg q;

  always @(posedge clk)
  begin
    if (~rst) q = 0;
    else q = d;
  end

  assign qb = ~q;
endmodule
```

Nonblocking Assignments <=

Blocking Assignments =

```
initial
begin
    clk =0;
    d =0;
    rst =1;
    rst =0;
    #20 rst = 1;

    #10 d <= 1;
    #10 d <= 0;
    #10 d <= 1;
    #10 d <= 0;
    #10 d <= 1;
    #10 d <= 1;

    $finish;
end
```

```
#10 d = 1;
#10 d = 0;
#10 d = 1;
#10 d = 0;
#10 d = 1;
#10 d = 1;
```

```
`timescale 1ns/100ps
```

```
module dff_tb;
  reg d, clk, rst;

  dff U1 (d, clk, rst, q, qb);

  always #10 clk = ~clk;
```



```
initial
begin
    $dumpfile("test.vcd");
    $dumpvars(0, dff_tb);
end
```

```
endmodule
```

Testbench with Nonblocking Assignments

```
module dff(d, clk, rst, q, qb);
  input d, clk, rst;
  output q, qb;
  reg q;

  always @(posedge clk)
  begin
    if (~rst) q = 0;
    else q = d;
  end

  assign qb = ~q;
endmodule
```

```
initial
begin
    clk =0;
    d =0;

    rst =1;
    rst =0;
    #20 rst = 1;

    #10 d <= 1;
    #10 d <= 0;
    #10 d <= 1;
    #10 d <= 0;
    #10 d <= 1;
    #10 d <= 1;
    $finish;
end
```

```
`timescale 1ns/100ps

module dff_tb;

reg d, clk, rst;

dff U1 (d, clk, rst, q, qb);

always #10 clk = ~clk;



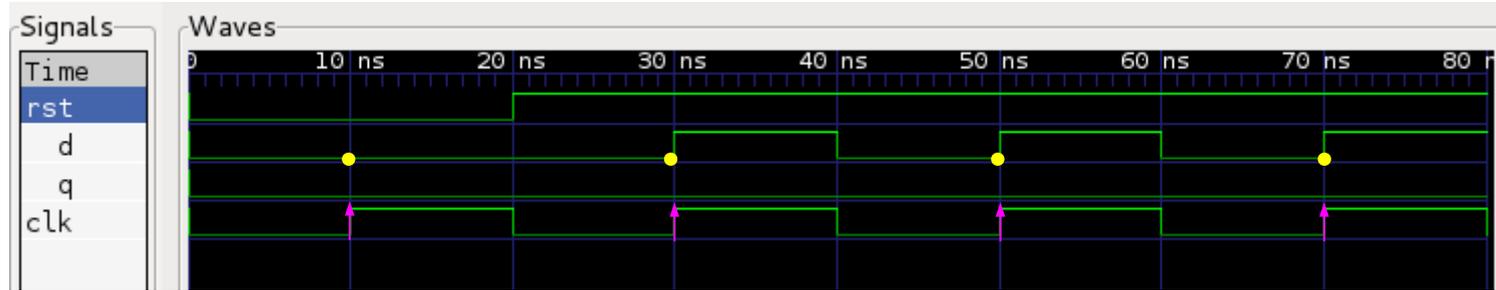
initial
begin
    $dumpfile("test.vcd");
    $dumpvars(0, dff_tb);
end

endmodule
```

DFF Testbench Waveforms

Nonblocking Assignments

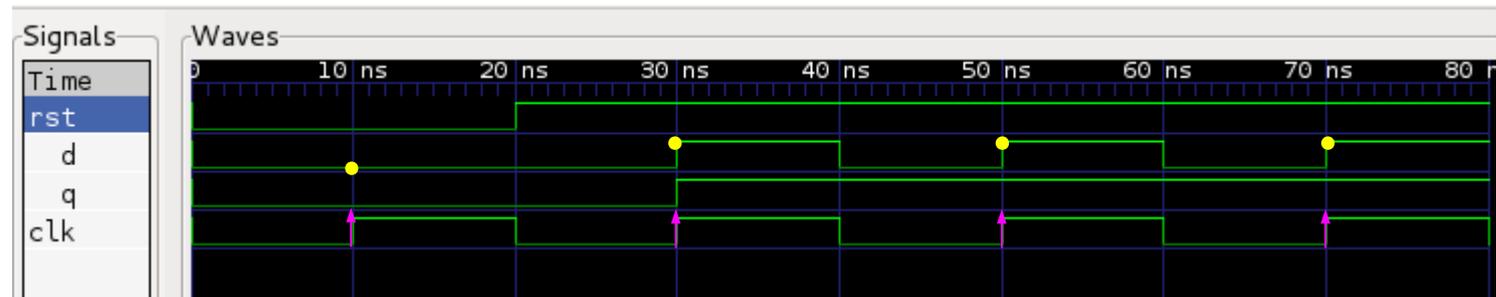
```
#10 d <= 1;  
#10 d <= 0;  
#10 d <= 1;  
#10 d <= 0;  
#10 d <= 1;  
#10 d <= 1;
```



samples the unchanged d input values at the posedge of clk

Blocking Assignments

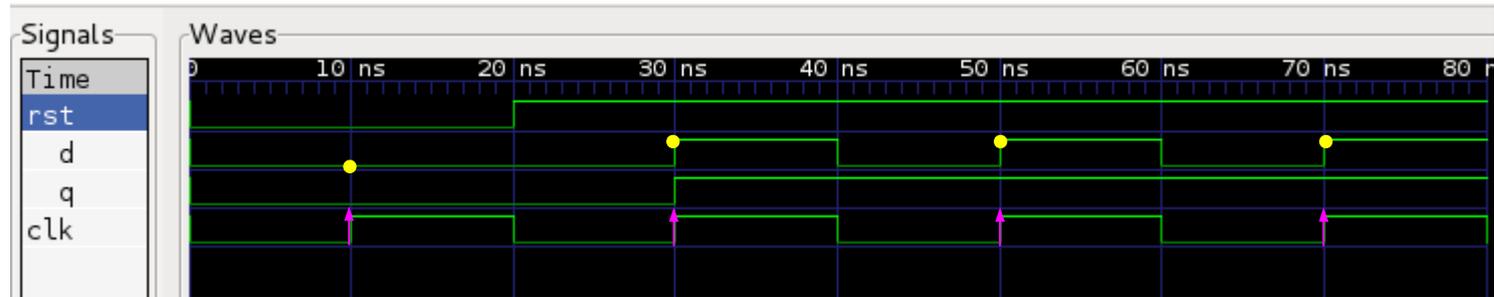
```
#10 d = 1;  
#10 d = 0;  
#10 d = 1;  
#10 d = 0;  
#10 d = 1;  
#10 d = 1;
```



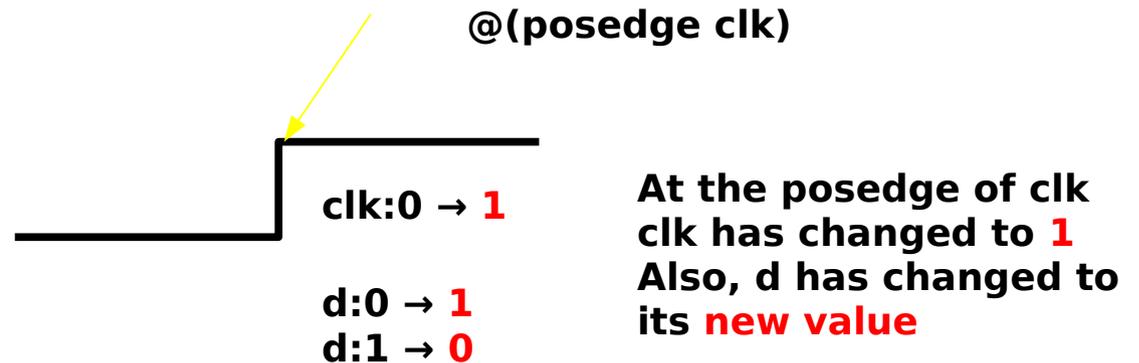
samples the changed d input values at the posedge of clk

Blocking Assignments

```
#10 d = 1;  
#10 d = 0;  
#10 d = 1;  
#10 d = 0;  
#10 d = 1;  
#10 d = 1;
```



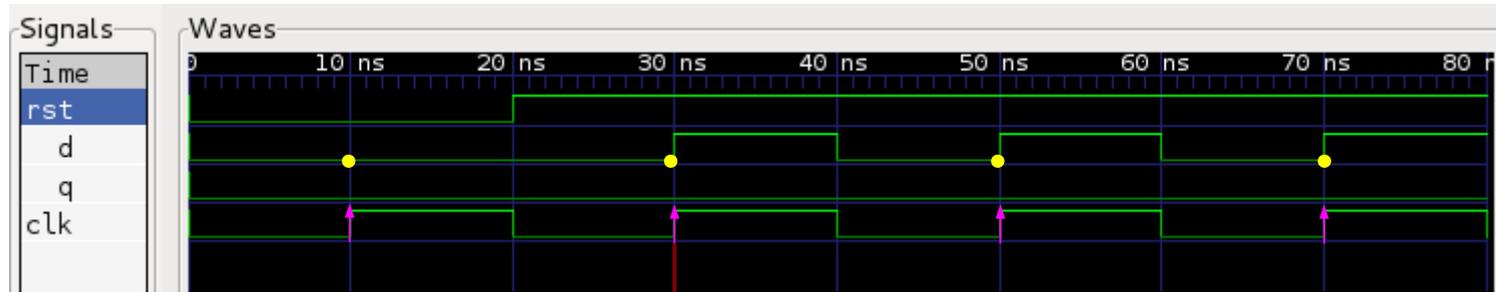
```
always @(posedge clk)  
begin  
  if (~rst) q = 0;  
  else    q = d;  
end
```



Nonblocking Assignments

Nonblocking Assignments

```
#10 d <= 1;  
#10 d <= 0;  
#10 d <= 1;  
#10 d <= 0;  
#10 d <= 1;  
#10 d <= 1;
```



samples the unchanged d input values at the posedge of clk

In each of time slot which is delayed by 10 time units, there is only **one** assignment that can be scheduled simultaneously,

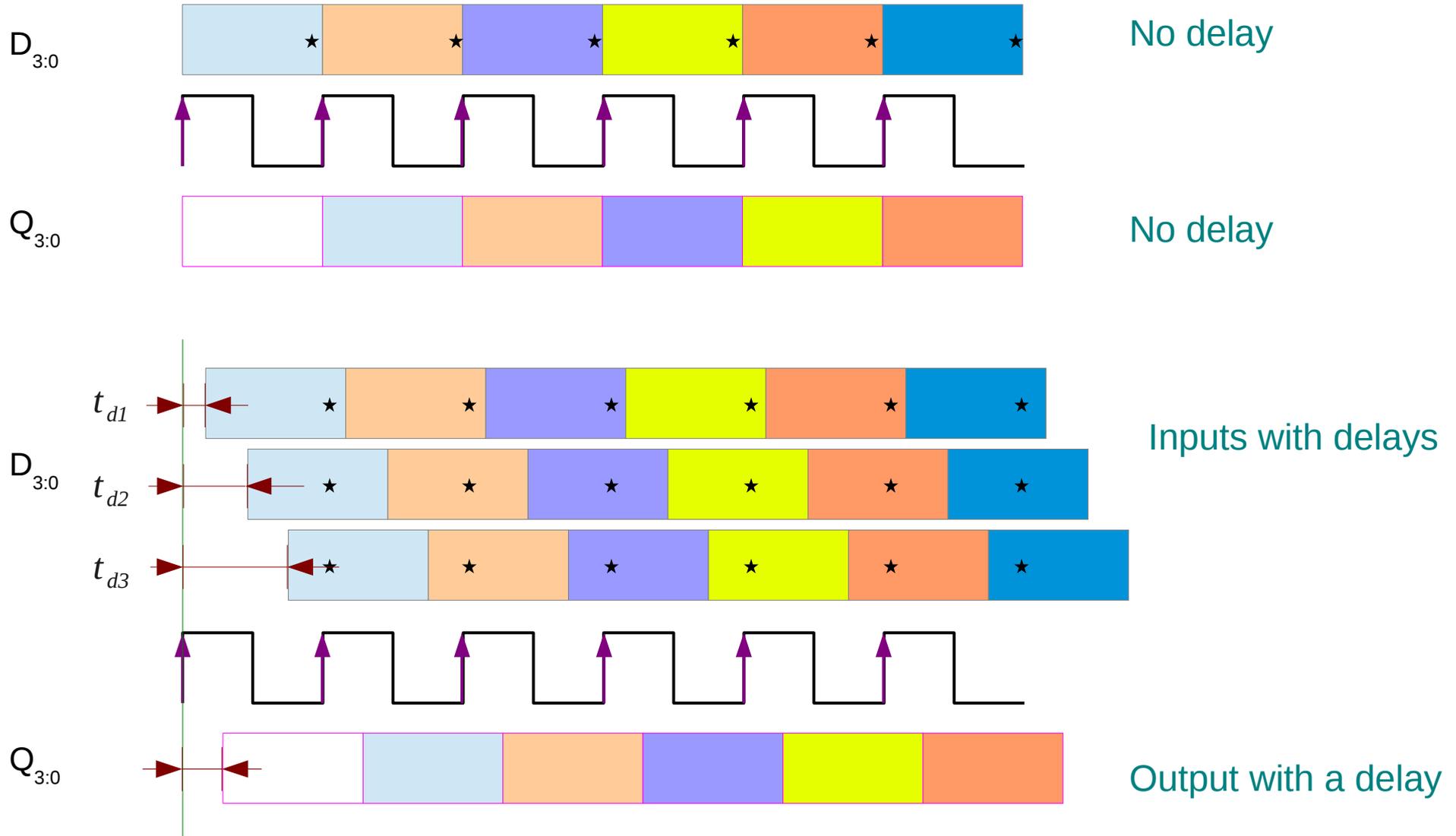
- (1) Allowing scheduling of assignments **without waiting for its completion**
- (2) Executed **last** in the time step in which it is scheduled (after all blocking assignments' execution)

Regular Events	Nonblocking Update Events	Monitor Events
----------------	---------------------------	----------------

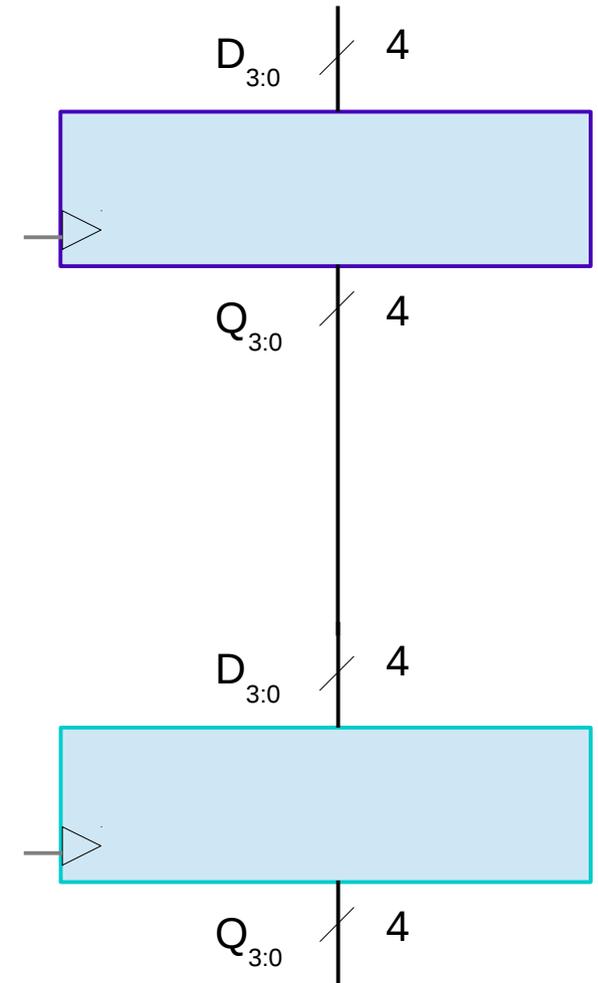
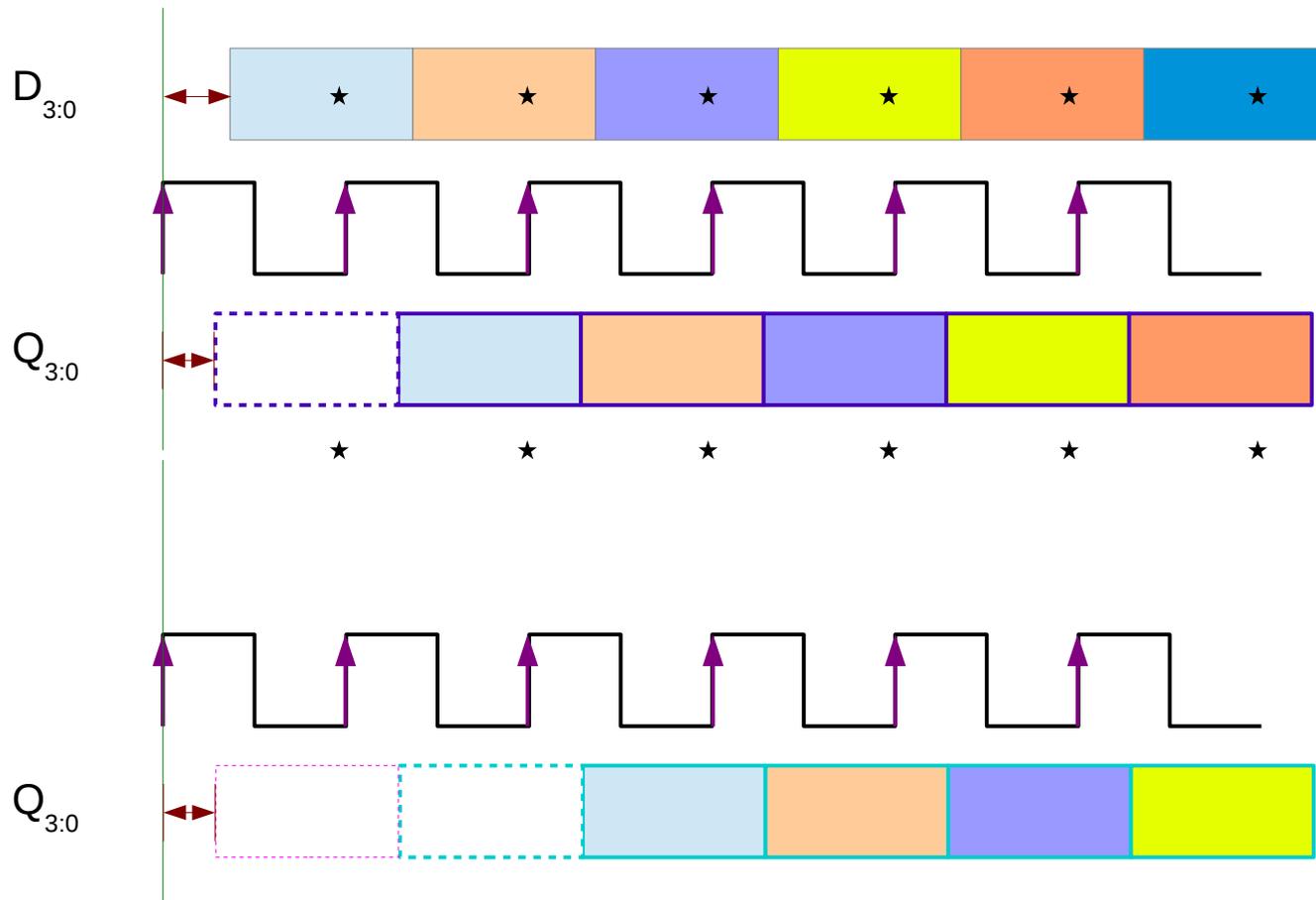
- Evaluate
- Update

So, the input d update is done after the posedge clk

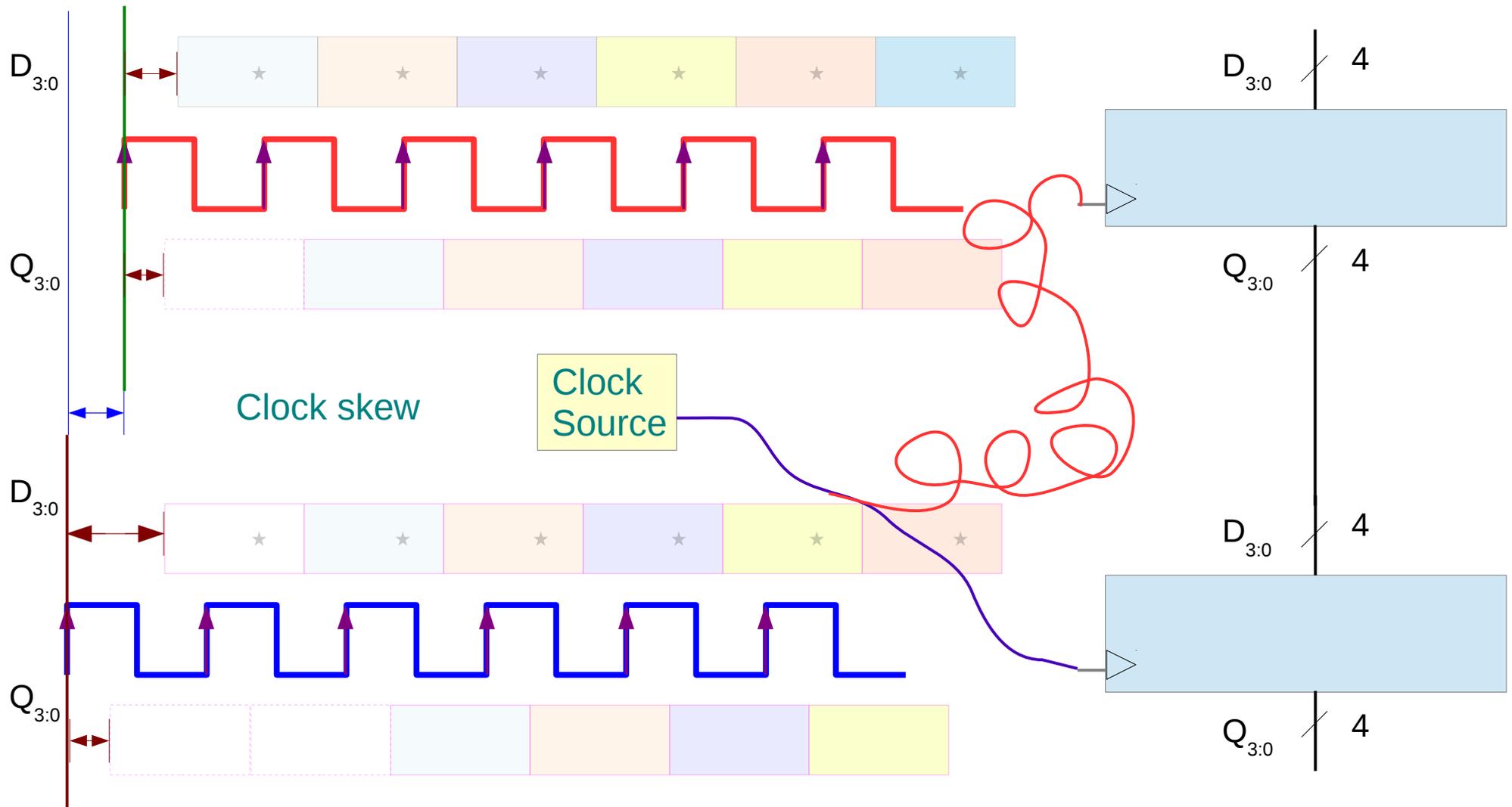
FF Timing - Input and Output Delays



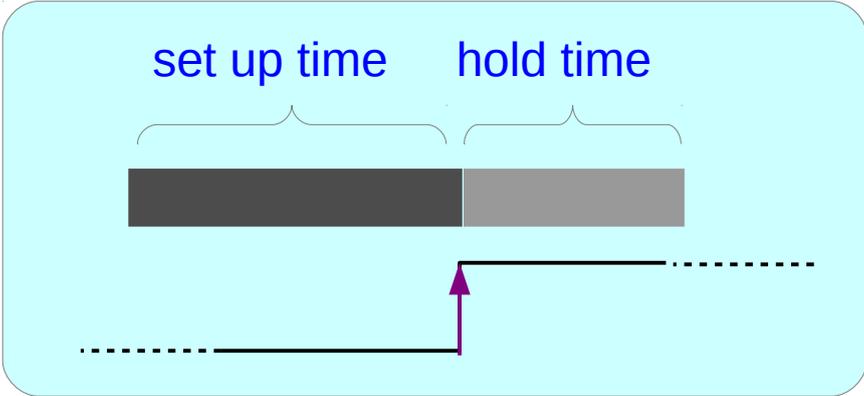
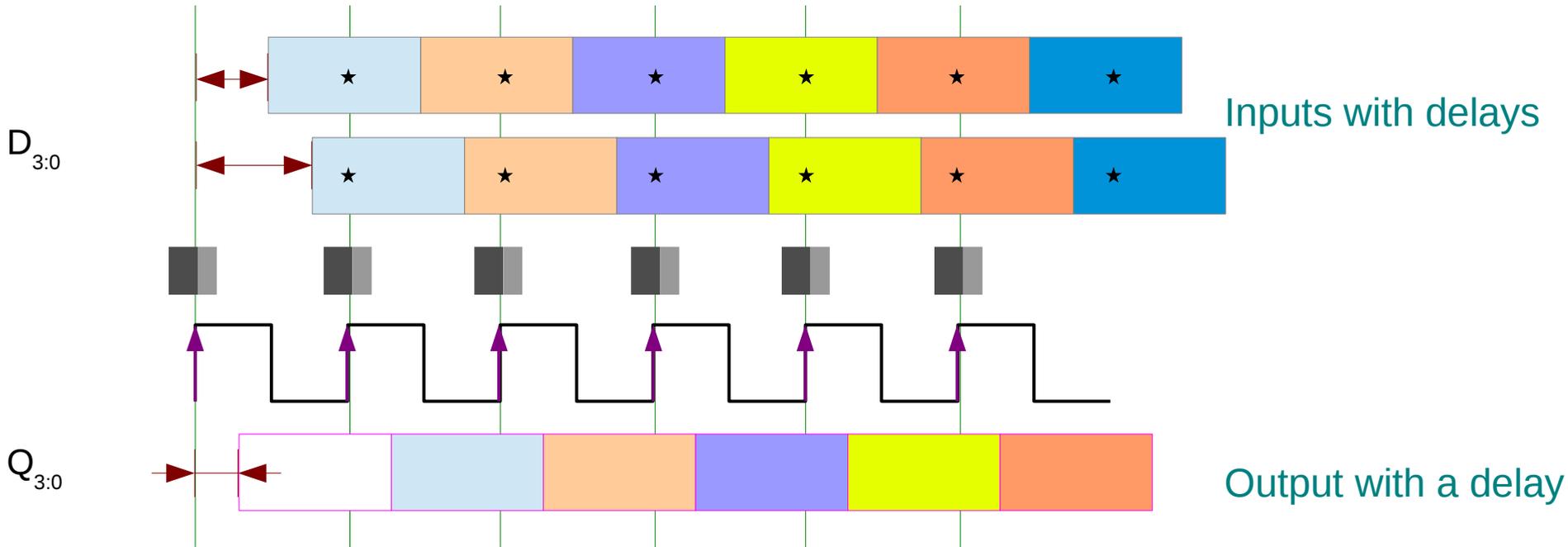
Reg to Reg Timing



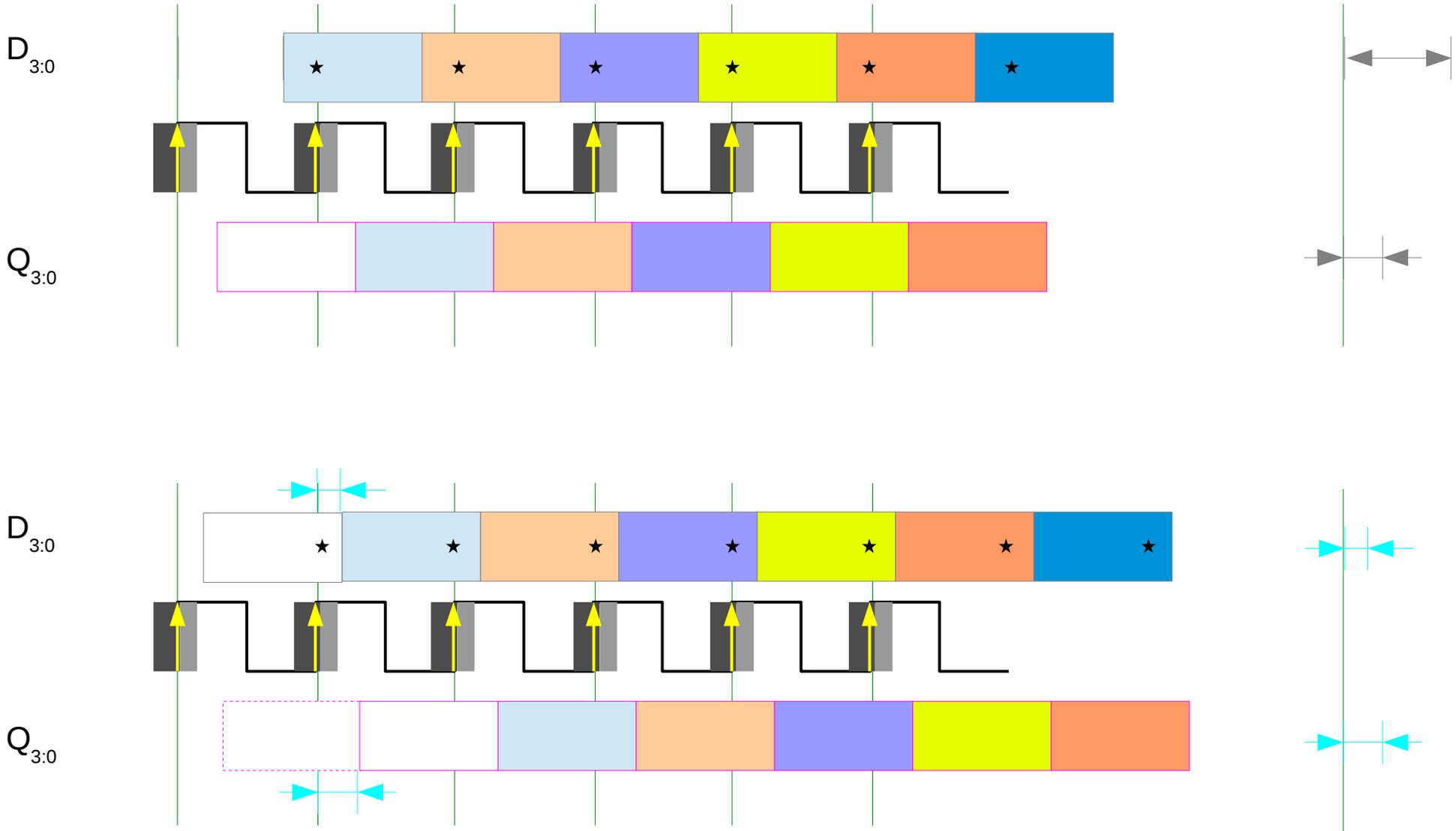
Clock Skew



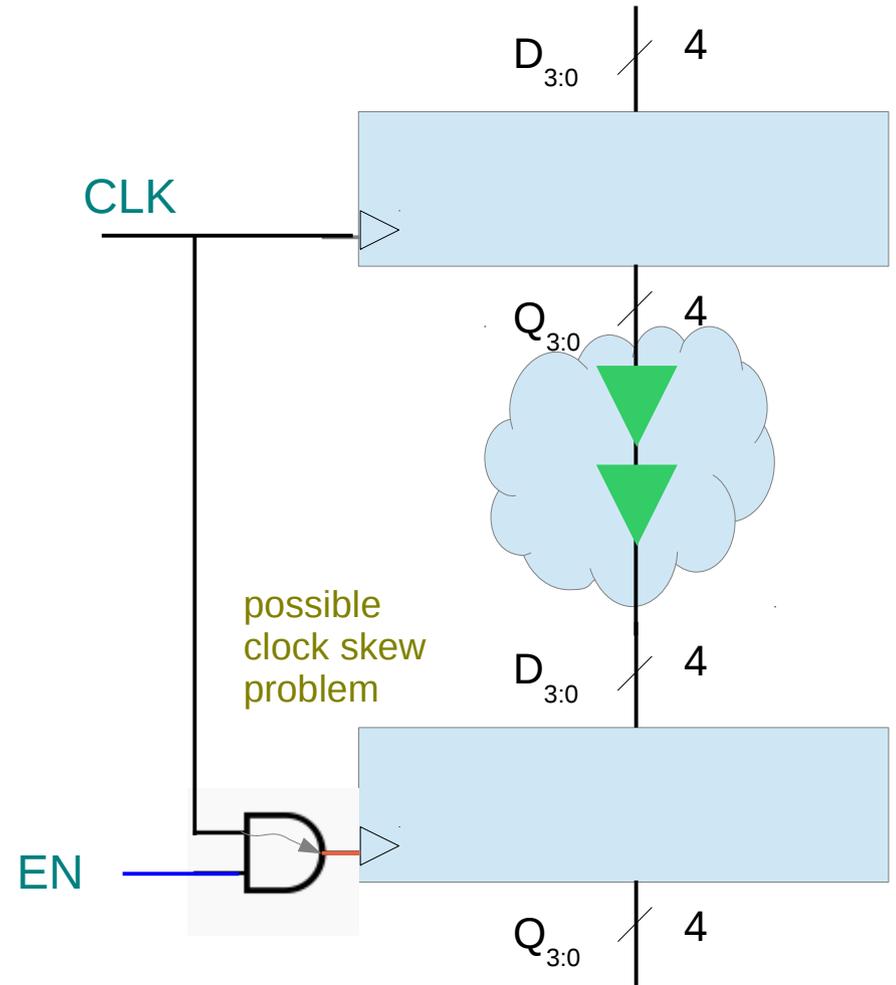
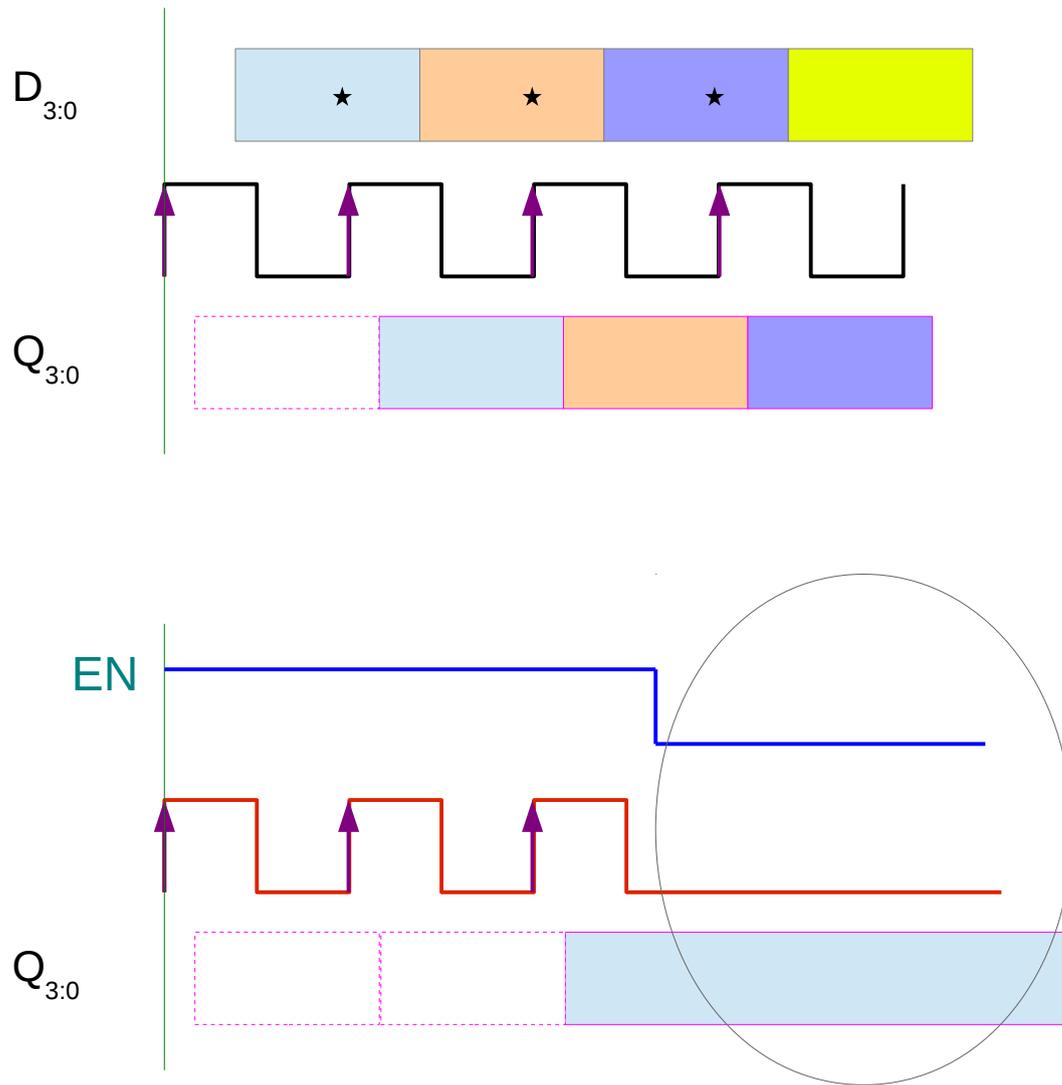
Setup & Hold Time (1)



Setup & Hold Time (2)



Clock Gating

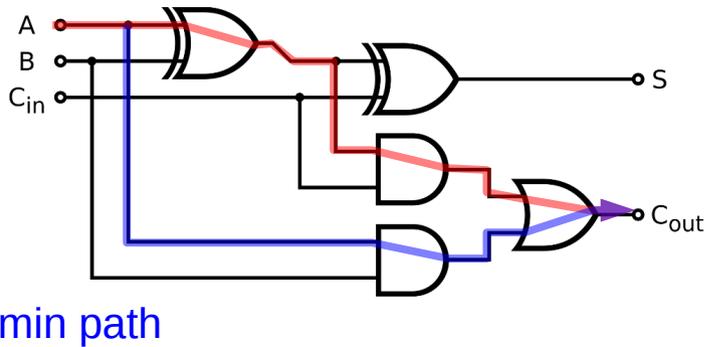


Min Delay, Max Delay

Min Max Delay

Max Path / Min Path

Max path



Max delay

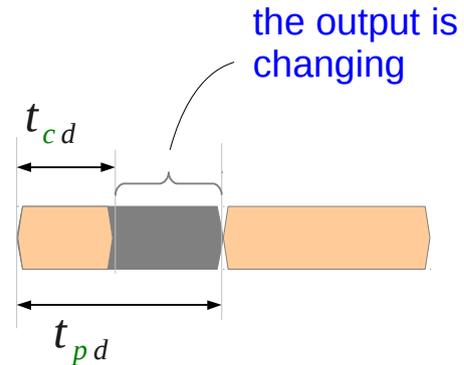
min delay

min path

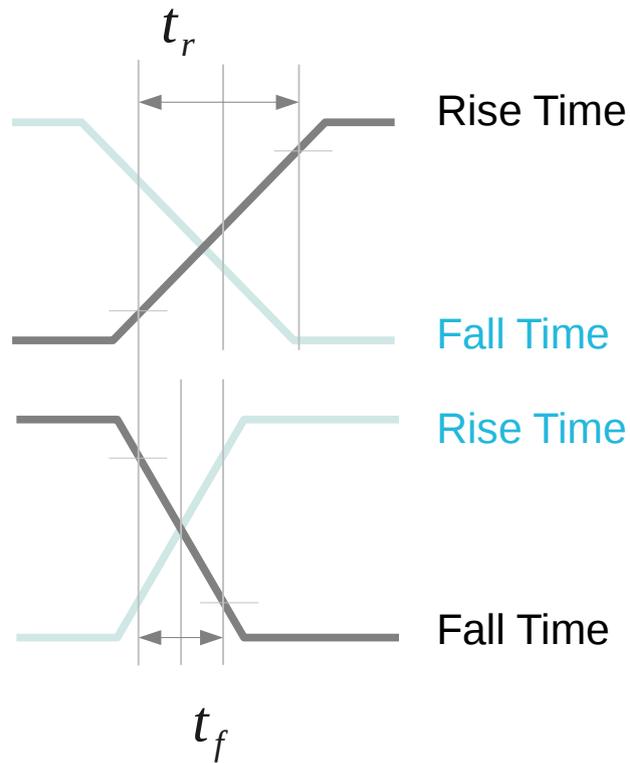
$$t_{cd} \leq t_{delay} \leq t_{pd}$$

min delay

Max delay

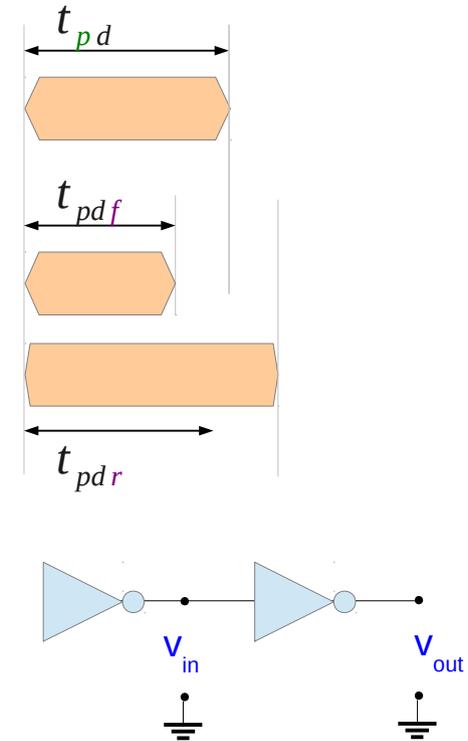


Rise / Fall Times



Max delay

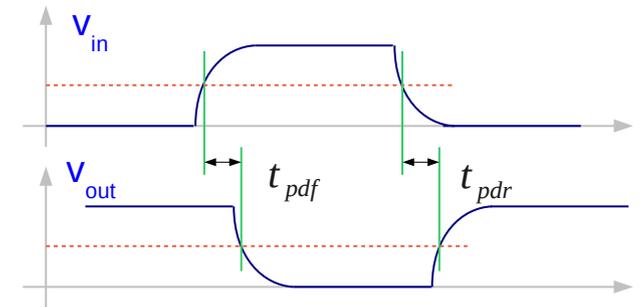
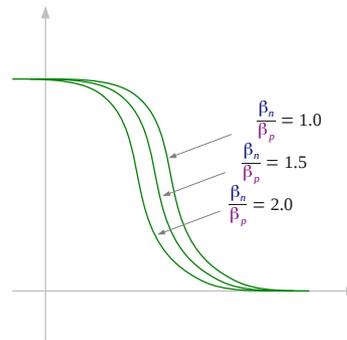
min delay



$$\frac{t_f}{t_r} = \frac{2.2\tau_n}{2.2\tau_p}$$

$$\frac{\beta_n}{\beta_p} > 1 \quad \frac{R_n}{R_p} < 1$$

$$\frac{\tau_n}{\tau_p} = \frac{R_n C_{out}}{R_p C_{out}} = \frac{R_n}{R_p} < 1$$

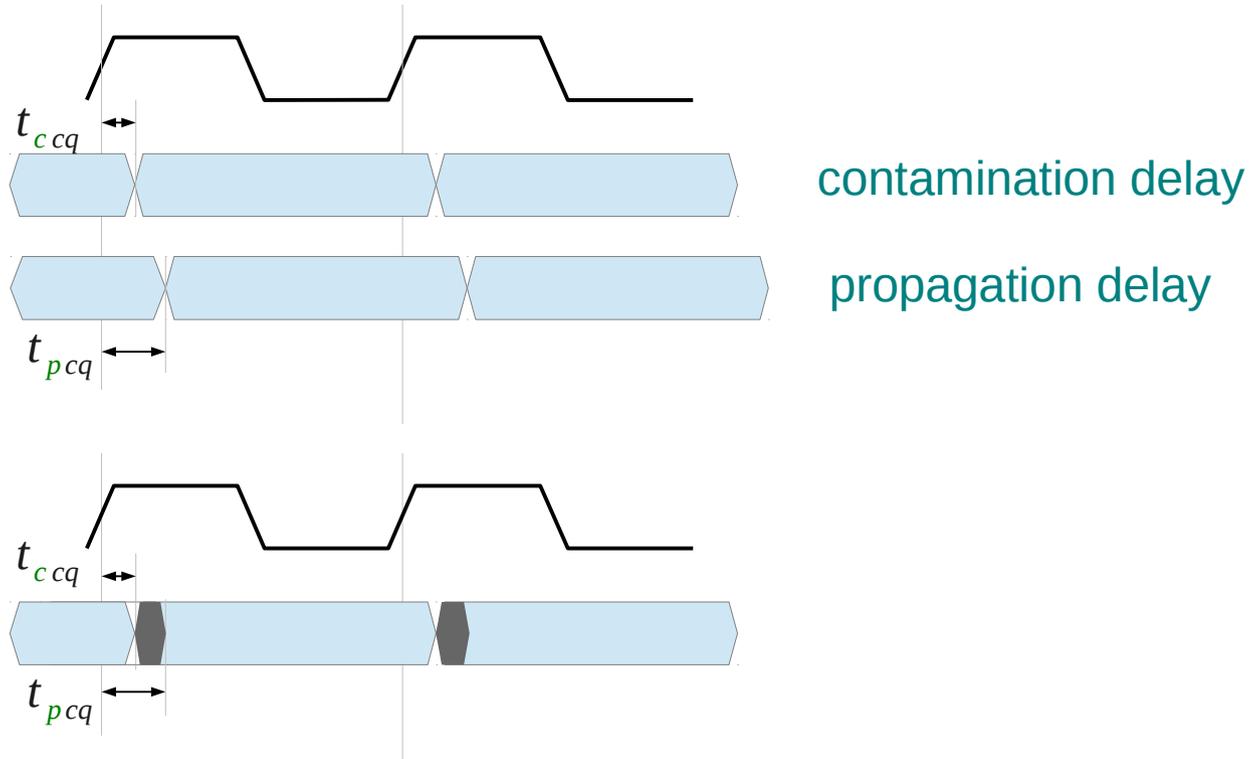


PVT Variation

{ Process
Voltage
Temperature

High temperature **Max delay**
Low temperature **min delay**

FF Output Delay



contamination delay

propagation delay

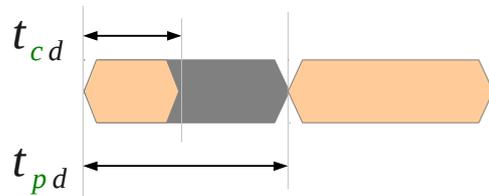
flipflop clock-to-q

$$t_{ccq} \leq t_{delay} \leq t_{pcq}$$

min delay

Max delay

Path Delay



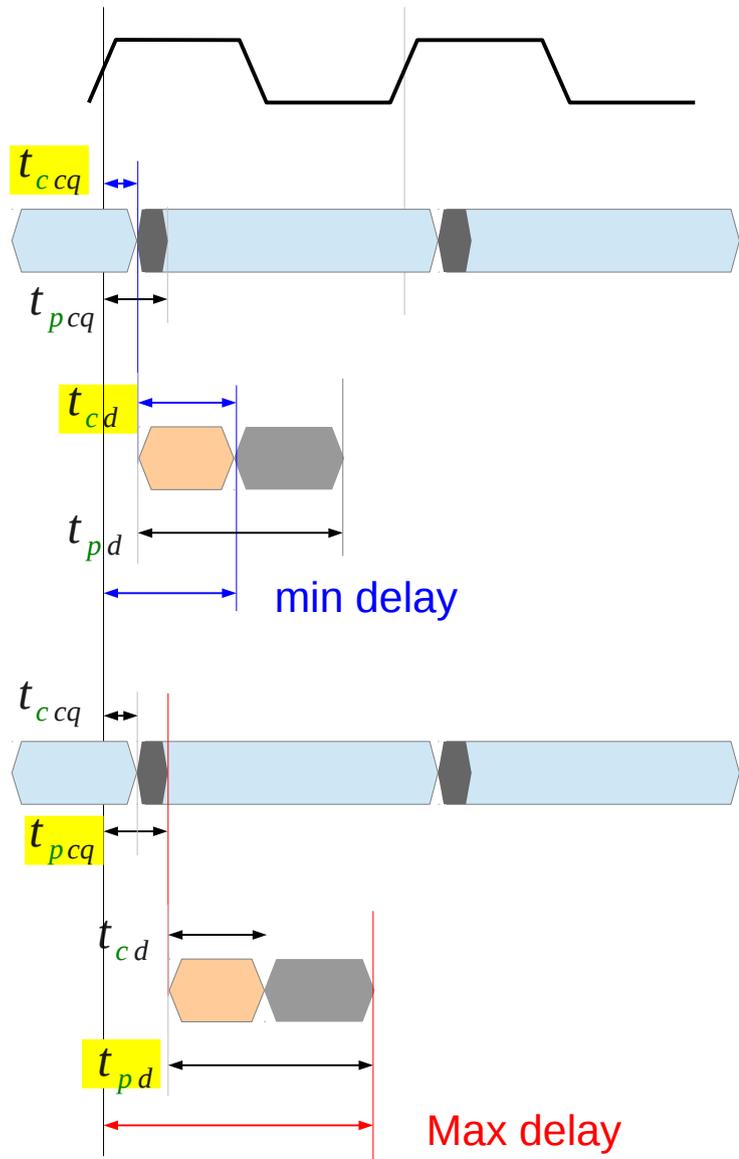
combinational logic delay

$$t_{cd} \leq t_{delay} \leq t_{pd}$$

min delay

Max delay

Reg-to-Reg Delay (1)



$$t_{ccq} \leq t_{FF} \leq t_{pcq}$$

min delay

Max delay

$$t_{cd} \leq t_{comb} \leq t_{pd}$$

min delay

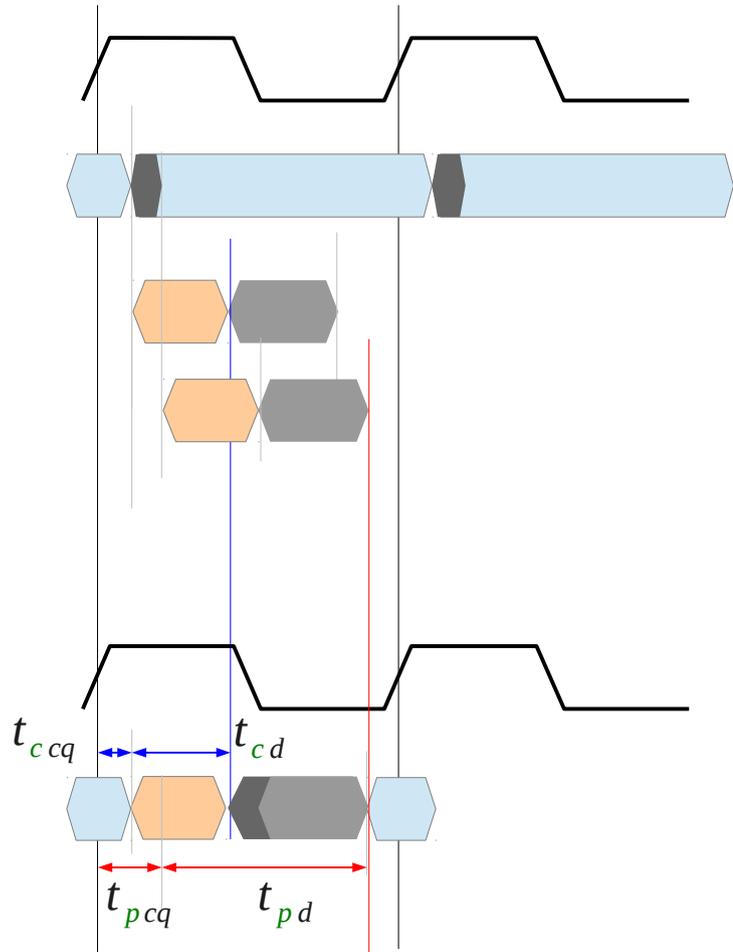
Max delay

$$t_{ccq} + t_{cd} \leq t_{delay} \leq t_{pcq} + t_{pd}$$

min delay

Max delay

Reg-to-Reg Delay (2)



$$t_{cq} \leq t_{FF} \leq t_{pcq}$$

min delay

Max delay

$$t_{cd} \leq t_{comb} \leq t_{pd}$$

min delay

Max delay

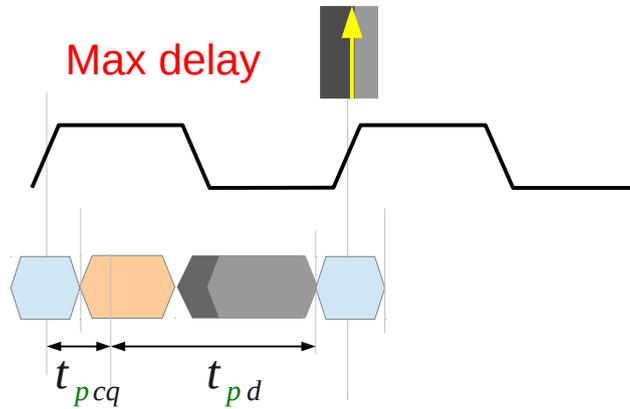
$$t_{cq} + t_{cd} \leq t_{delay} \leq t_{pcq} + t_{pd}$$

min delay

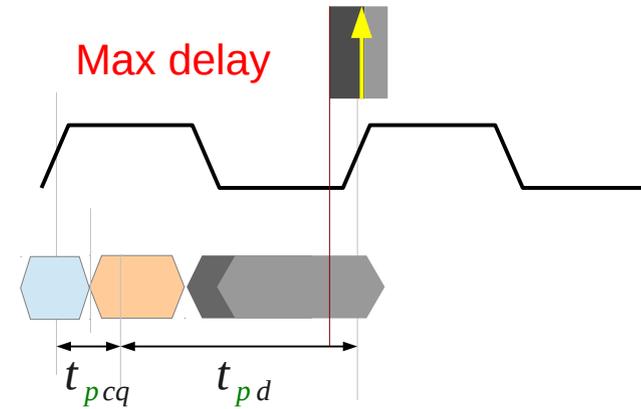
Max delay

Setup Time / Hold Time

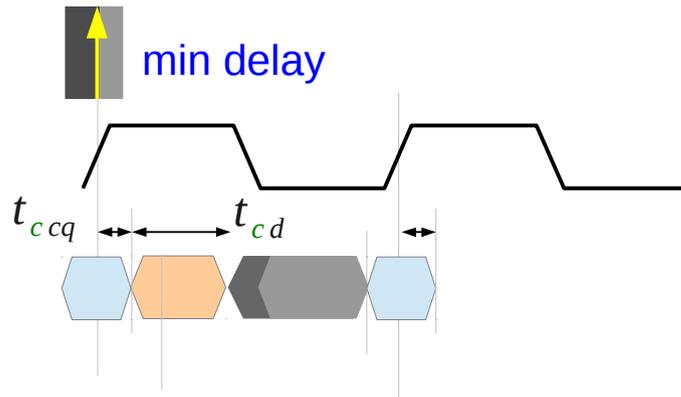
Setup Time OK



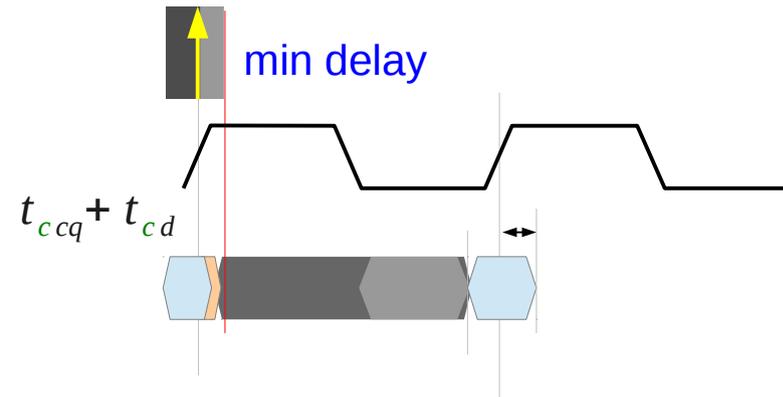
Setup Time Violation



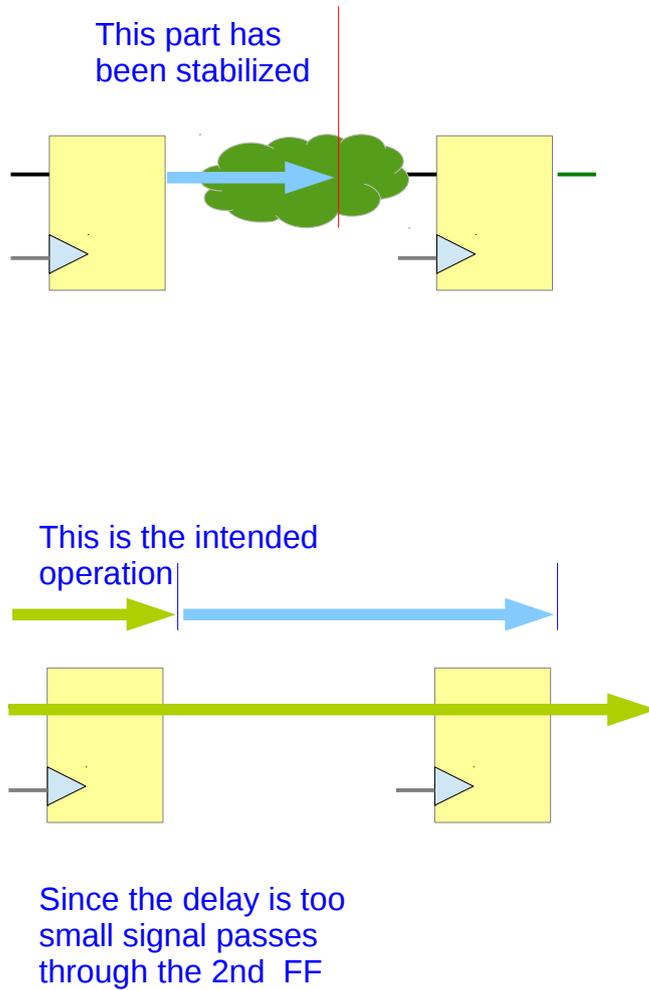
Hold Time OK



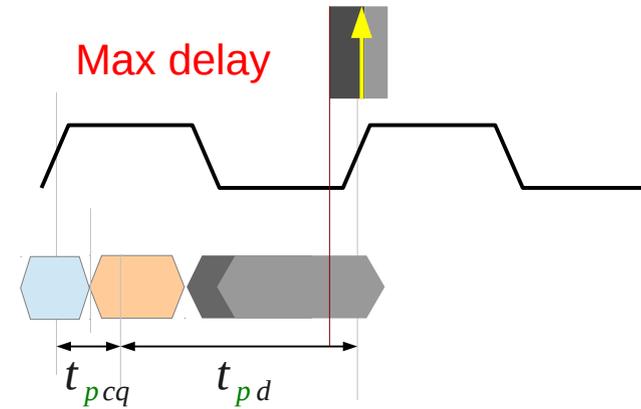
Hold Time Violation



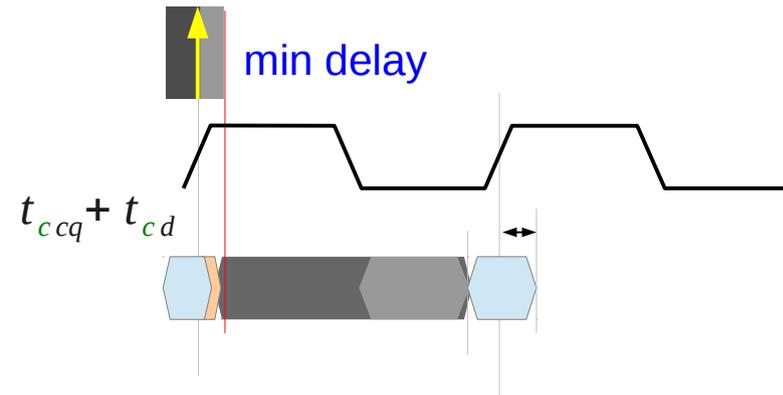
Setup Time / Hold Time



Setup Time Violation



Hold Time Violation



References

- [1] <http://en.wikipedia.org/>
- [2] M. M. Mano, C. R. Kime, “Logic and Computer Design Fundamentals”, 4th ed.
- [3] J. Stephenson, Understanding Metastability in FPGAs. Altera Corporation white paper. July 2009.