

Clock Skew

Clock Skew

Probably the largest source of design failures in FPGA.

Definitions

- **Sequentially-Adjacent Register Pair**

Two registers with only combinational logic or interconnect between them

- **Clock Skew**

Given two sequentially-adjacent registers, R_i and R_j , and an equipotential clock distribution network, the clock skew between these two registers is defined as

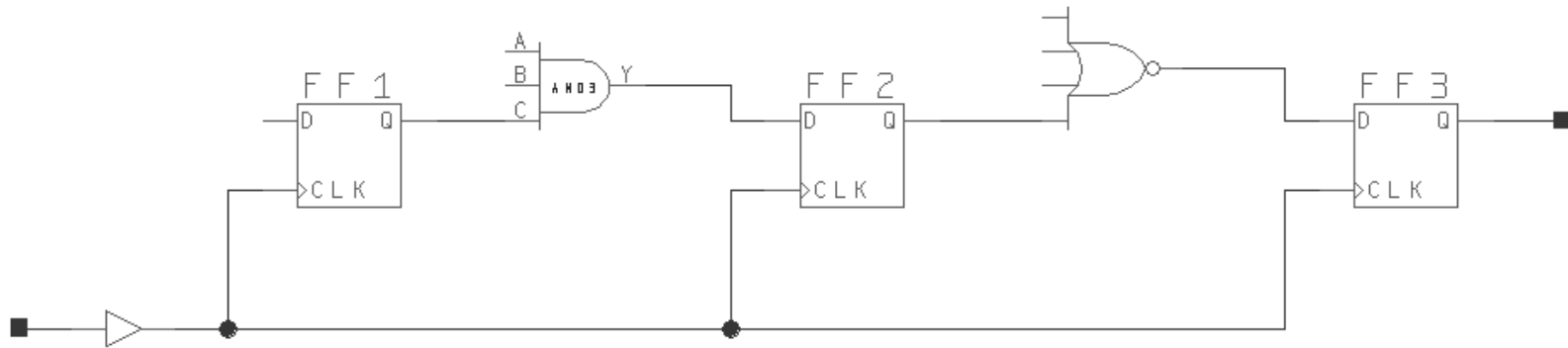
$$T_{\text{skew-}i,j} = T_{ci} - T_{cj}$$

where T_{ci} and T_{cj} are the clock delays from the clock source to the registers R_i and R_j , respectively.

Implications of Definitions

- Sequentially-Adjacent Register Pair

Skew is only meaningful between adjacent pairs of registers, not between any pair of registers in a clock domain.

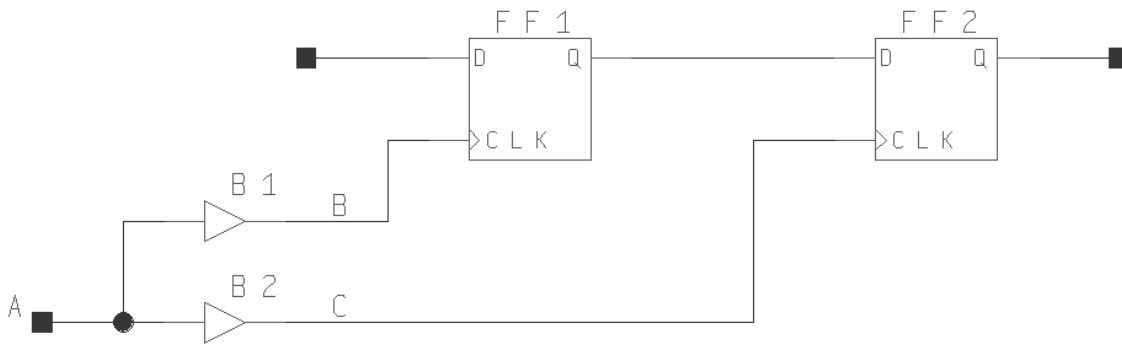


- Clock skew is only important between the pairs (FF1, FF2) and (FF2, FF3).
- The skew between the pair (FF1, FF3) is meaningless.
- Calculations done quickly on sets is often unrealistic.

Implications of Definitions

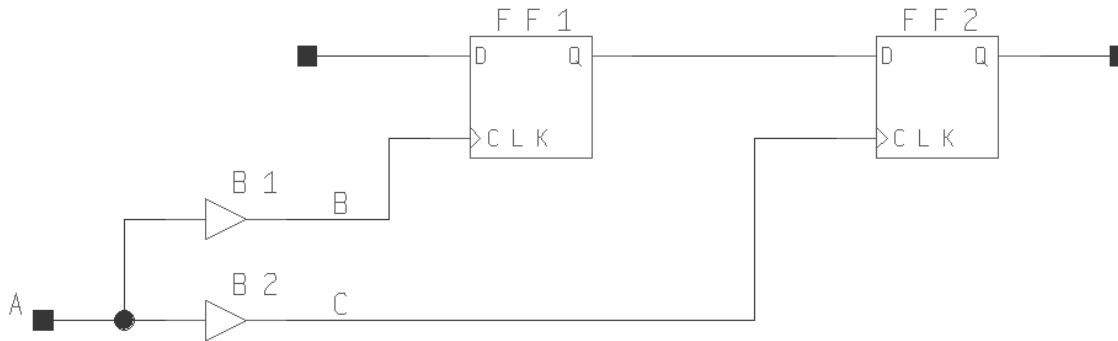
- Definition of skew: “... from the clock source”

A circuit node that is common in time for both paths is the origin of skew calculations.



- ‘A’ is the common point in time for this circuit.
- ‘B’ and ‘C’ are not common. Why? B1 and B2 have different transistors. The loading on B and C are different. B and C can have different line lengths. Parasitic capacitances can be different. Etc. Lines on a schematic are not ideal wires.
- B1 and B2 will not “track” 100%. This is shown later for aging.

A First Skew Calculation



Two cases: 1) $A \rightarrow B$ is fast, $A \rightarrow C$ slow
2) $A \rightarrow B$ is slow, $A \rightarrow C$ fast

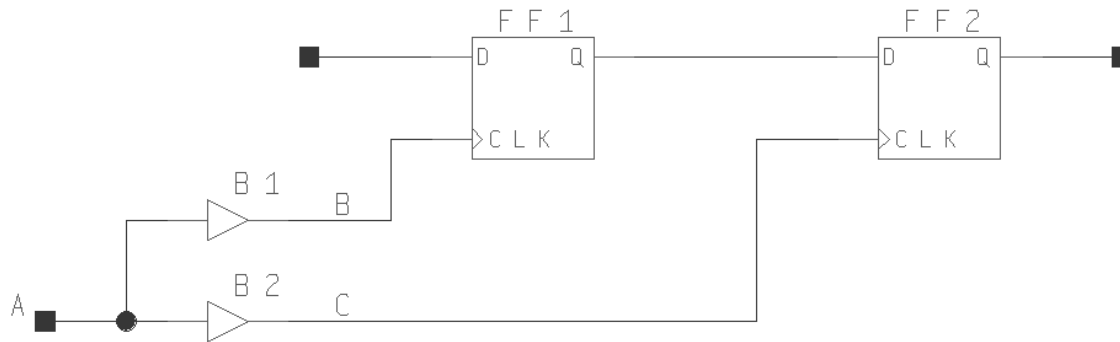
Just show the case 1 calculation (hold time)

$$\text{Skew} = \text{tplh } B1_{\min} - \text{tplh } B2_{\max}$$

Is this correct?

Is this realistic?

A First Skew Calculation

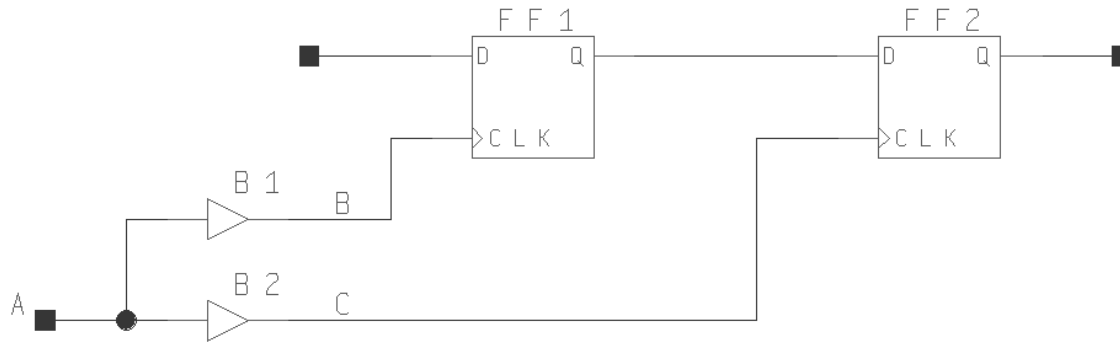


$$\text{Skew} = \text{tplh } B1_{\min} - \text{tplh } B2_{\max}$$

Is this correct?

No, it is wrong. It assumes that the signal 'A' arrives at the inputs to B1 and B2 at the same time. In an FPGA in particular, they will not, making this calculation too optimistic.

A First Skew Calculation



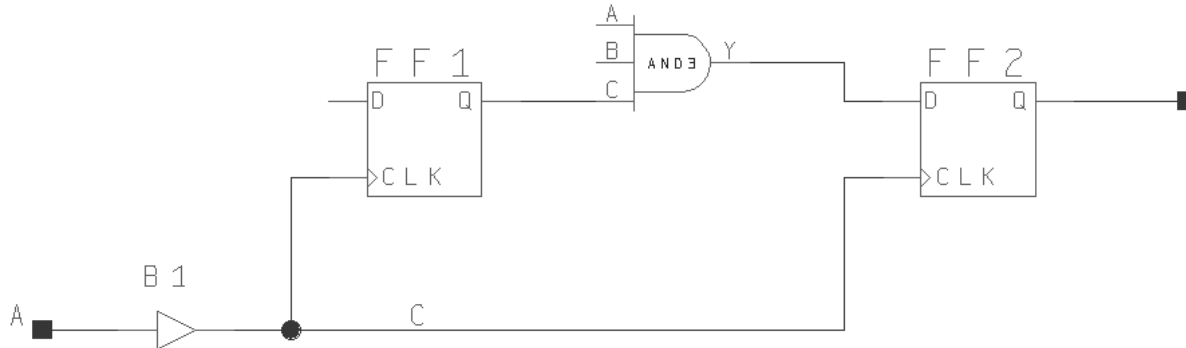
$$\text{Skew} = t_{\text{plh } B1_{\text{min}}} - t_{\text{plh } B2_{\text{max}}}$$

Is this realistic?

It is wrong if you use extreme value analysis without thinking, assuming that this circuit is on one chip and would make this calculation too conservative.

For example, temperature and voltage won't have drastic differences on the same chip at the same time. Life and radiation effects can not be assumed to track. Note that calculations must be done over all corners.

A Zeroth Skew Calculation



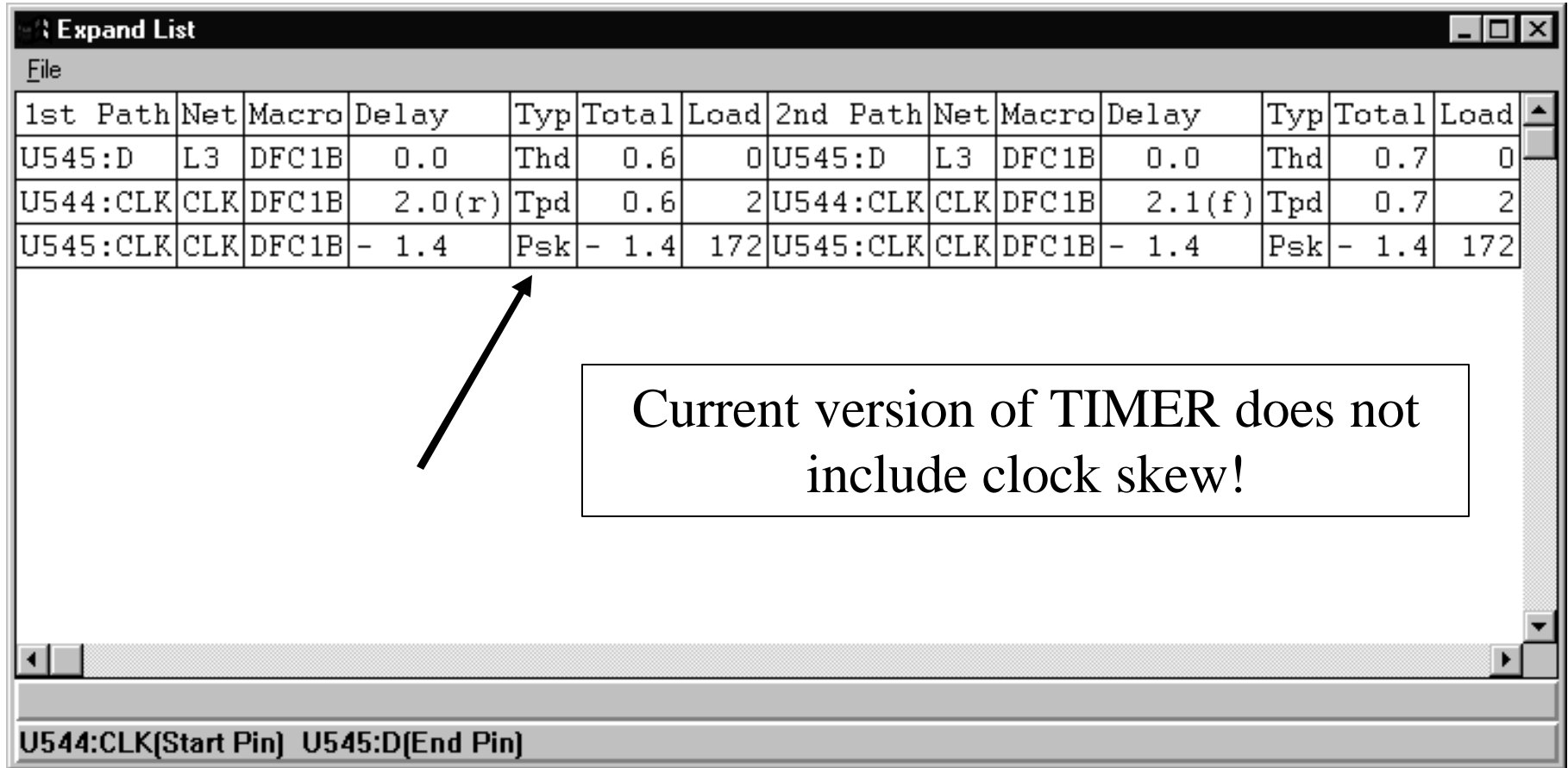
For this simpler case, we have to calculate the time from $A \rightarrow B$ (min) and from $A \rightarrow C$ (max) to compute the worst-case skew.

The input to B1, A, is common to both, so that is our common timing point.

The paths from the output of B1 to FF1:CLK and to FF2:CLK are different.

But $t_{plh}(B1)$ is in common. Legally, that can be factored out, eliminating the unrealistic conservatism of having a transistor be both ‘good’ and ‘bad’ at the same time. Practically, however, some tools lump the gate and routing delays together, so the calculation is needlessly conservative.

Tools: Your Friend?



The screenshot shows a window titled "Expand List" with a menu bar containing "File". Below the menu bar is a table with 14 columns: "1st Path", "Net", "Macro", "Delay", "Typ", "Total", "Load", "2nd Path", "Net", "Macro", "Delay", "Typ", "Total", "Load". The table contains four rows of data. An arrow points from a text box to the "Delay" column of the third row.

1st Path	Net	Macro	Delay	Typ	Total	Load	2nd Path	Net	Macro	Delay	Typ	Total	Load
U545:D	L3	DFC1B	0.0	Thd	0.6	0	U545:D	L3	DFC1B	0.0	Thd	0.7	0
U544:CLK	CLK	DFC1B	2.0(r)	Tpd	0.6	2	U544:CLK	CLK	DFC1B	2.1(f)	Tpd	0.7	2
U545:CLK	CLK	DFC1B	- 1.4	PsK	- 1.4	172	U545:CLK	CLK	DFC1B	- 1.4	PsK	- 1.4	172

Current version of TIMER does not include clock skew!

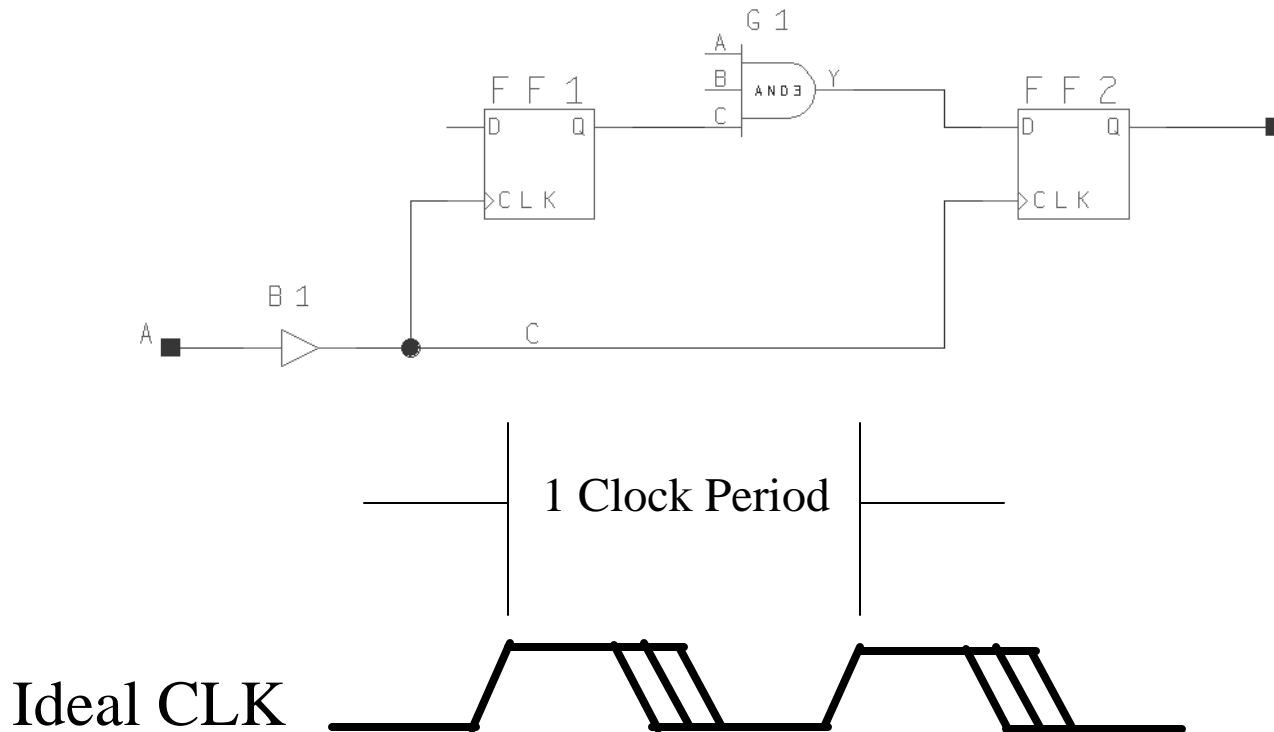
U544:CLK[Start Pin] U545:D[End Pin]

Some tools try and calculate skews for you. However, you must understand the model that it is using and apply it wisely.

Effects of Skew

Assume: Sequentially-Adjacent, Rising Edge-Triggered Flip-Flops
Routing delay bundled with gate delay

Case 1: Setup time



Effects of Skew

Assume: Sequentially-Adjacent, Rising Edge-Triggered Flip-Flops

Case 1: setup time

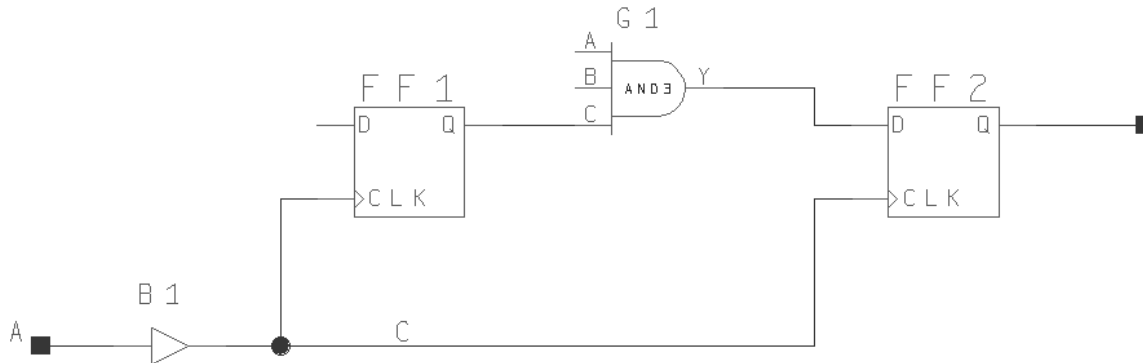
With an ideal, skew-free clock, we have:

Add delays: FF1: CLK \rightarrow Q (max)

G1 t_p (max)

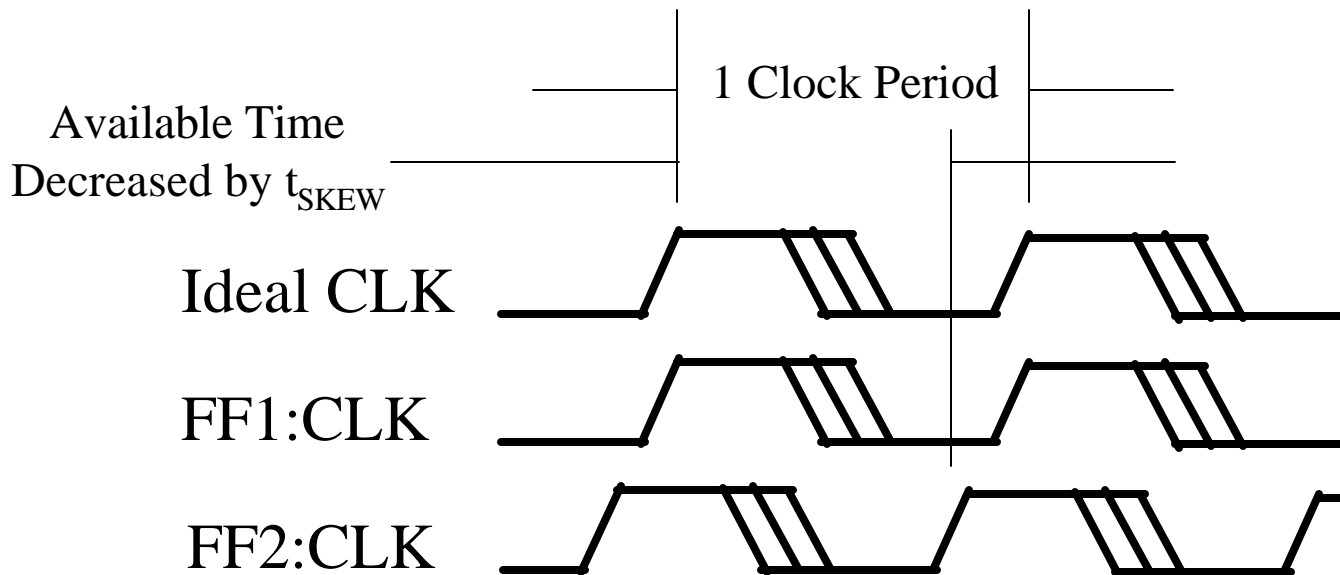
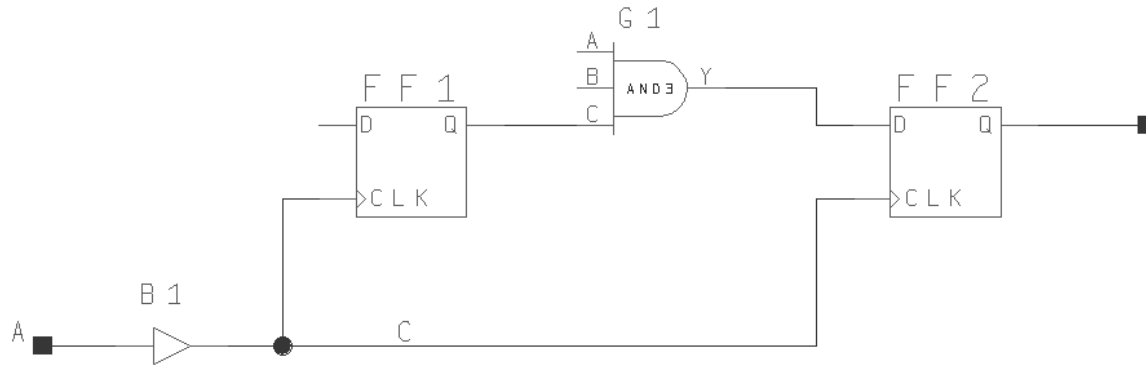
FF2: t_{SU}

Margin = t_{period} - total



Effects of Skew

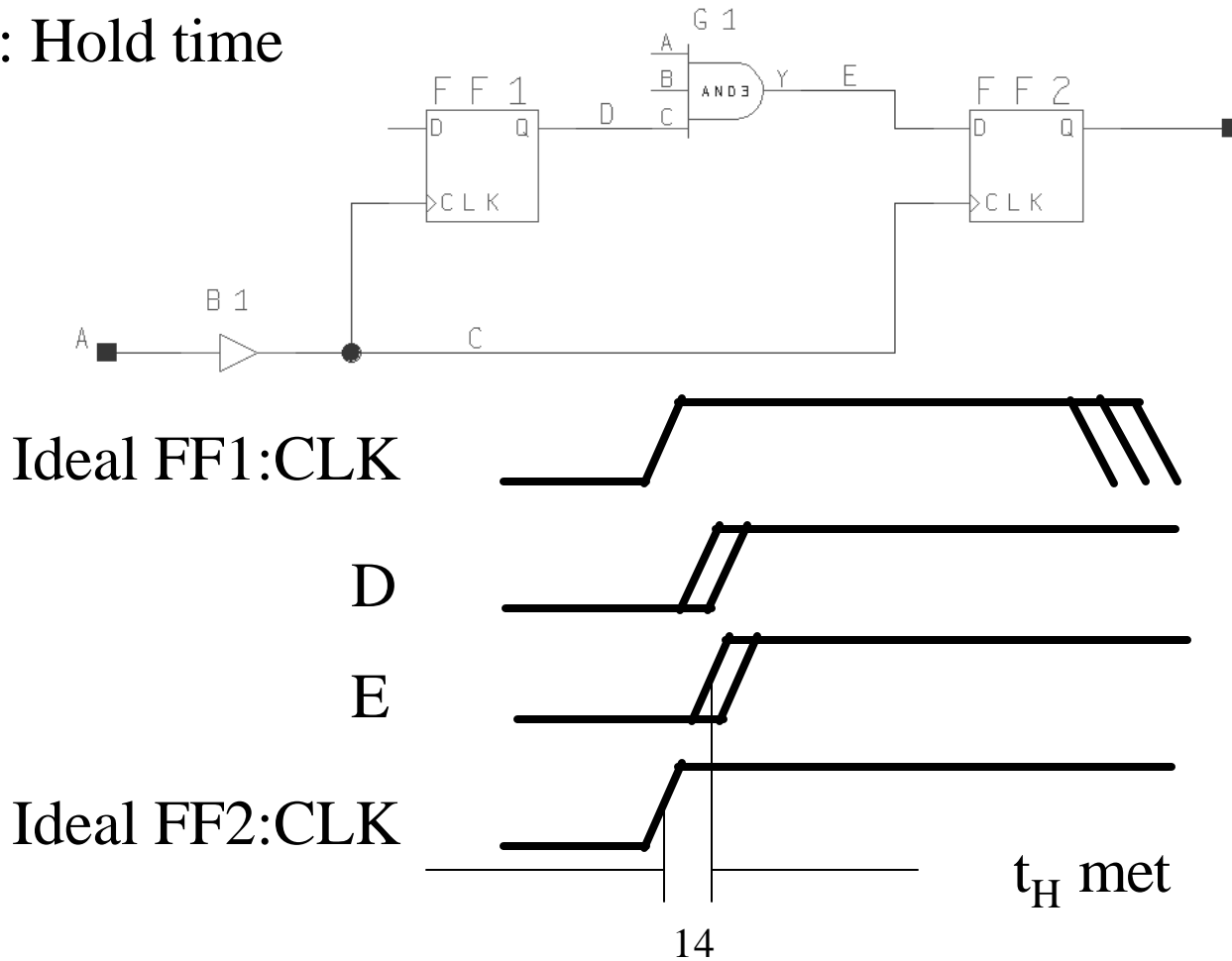
Add Worst-Case Skew for t_{SU}



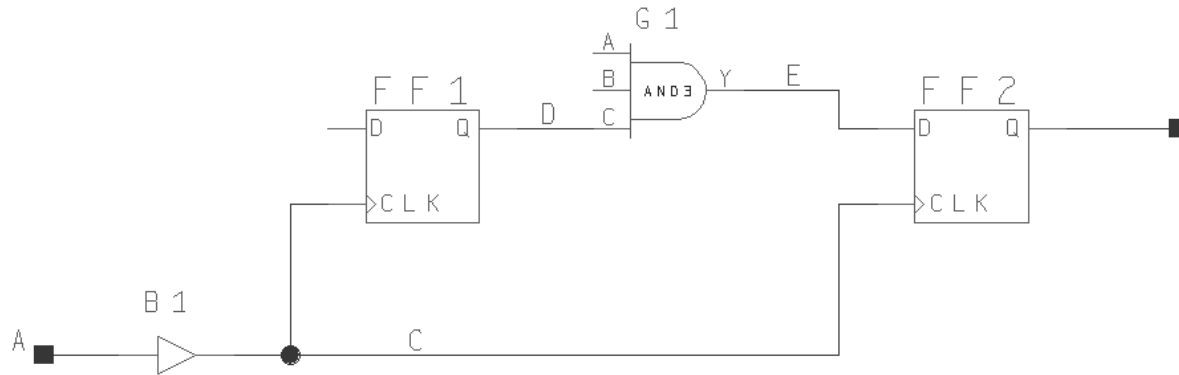
Effects of Skew

Assume: Sequentially-Adjacent, Rising Edge-Triggered Flip-Flops
Routing delay bundled with gate delay

Case 2: Hold time



Effects of Skew

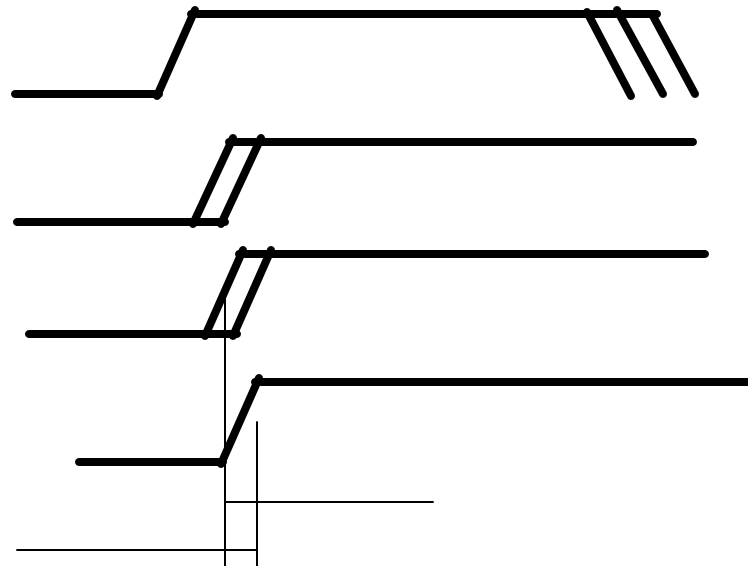


Early FF1:CLK

D

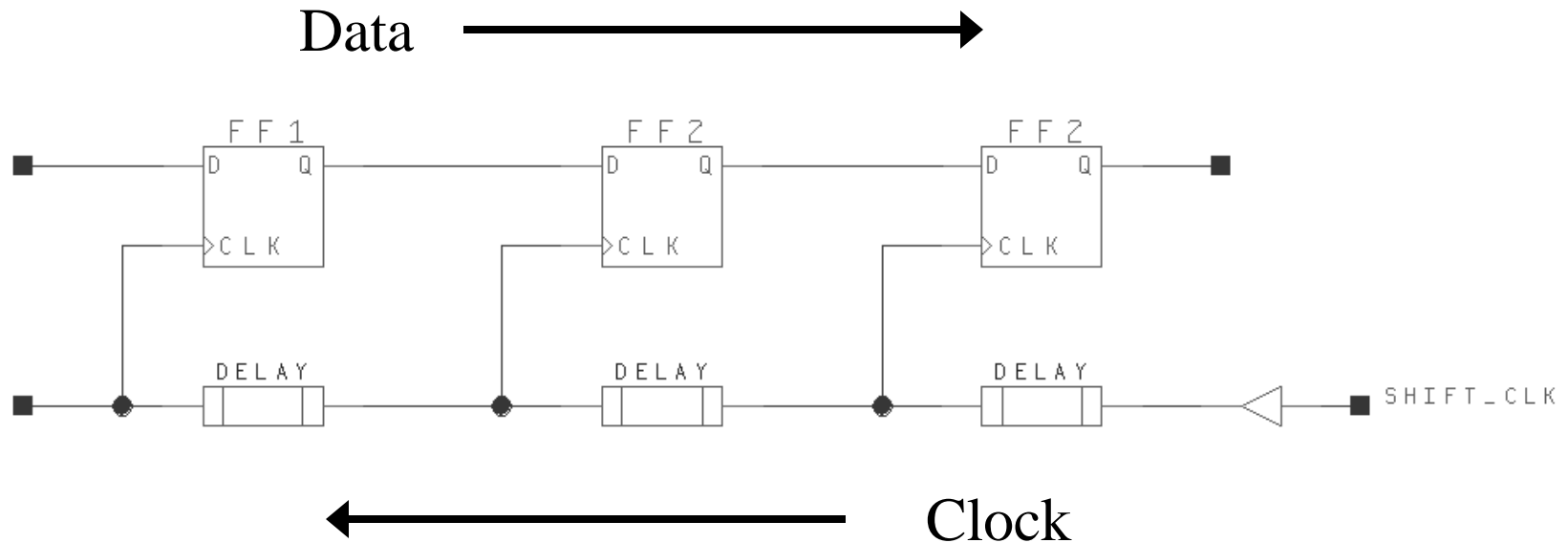
E

Late FF2:CLK



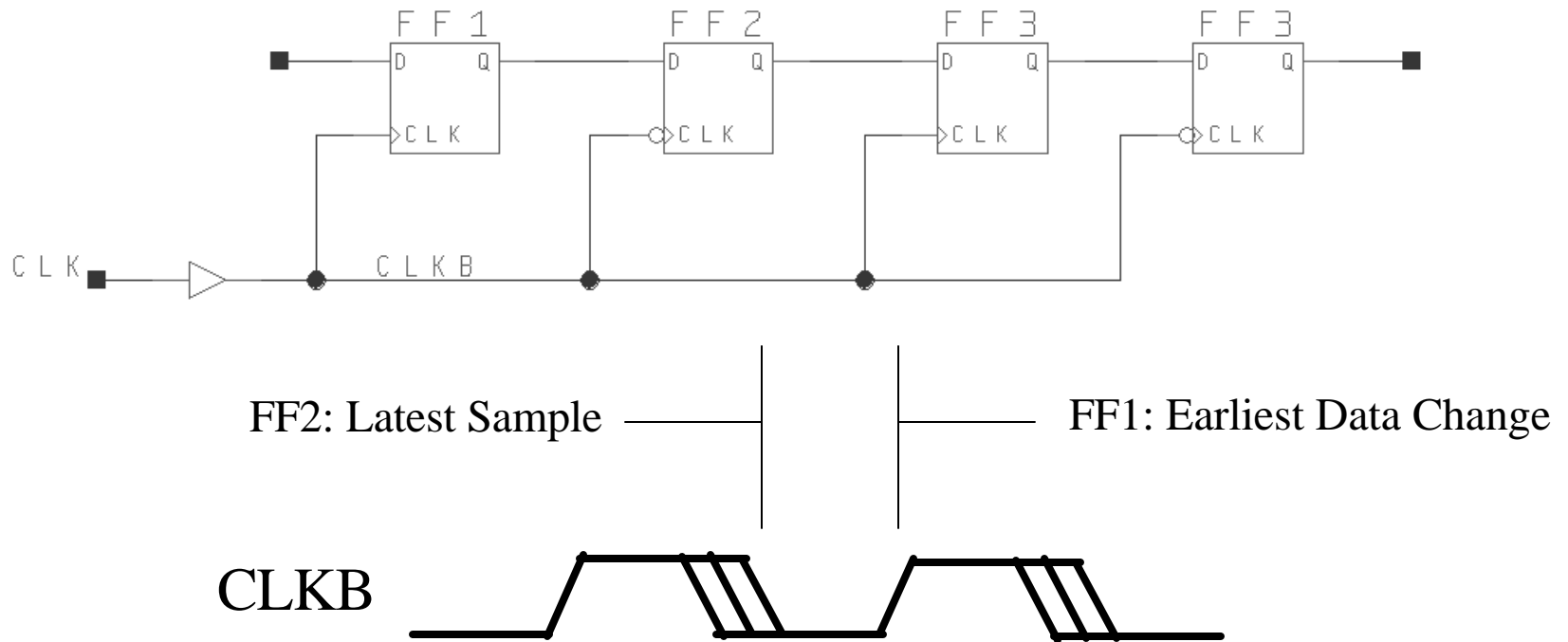
Note: used min, best case for prop delays and max, worst-case for clock path to FF2.

Clock Skew, Our Friend



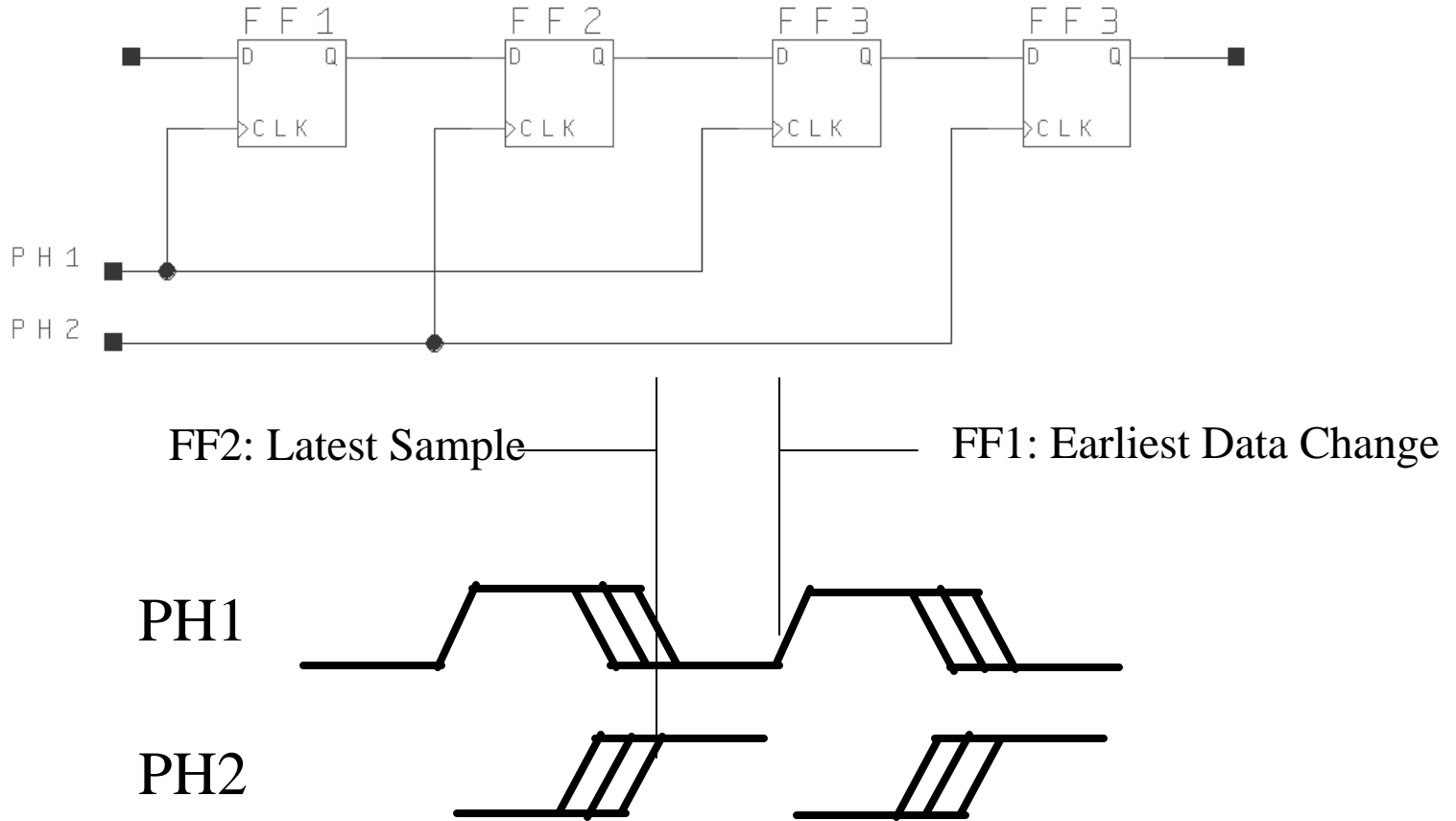
Attempting to force the downstream flip-flops to be clocked before the upstream flip-flops can change their data. This improves hold time at the cost of setup time.

Clock Skew, Our Friend



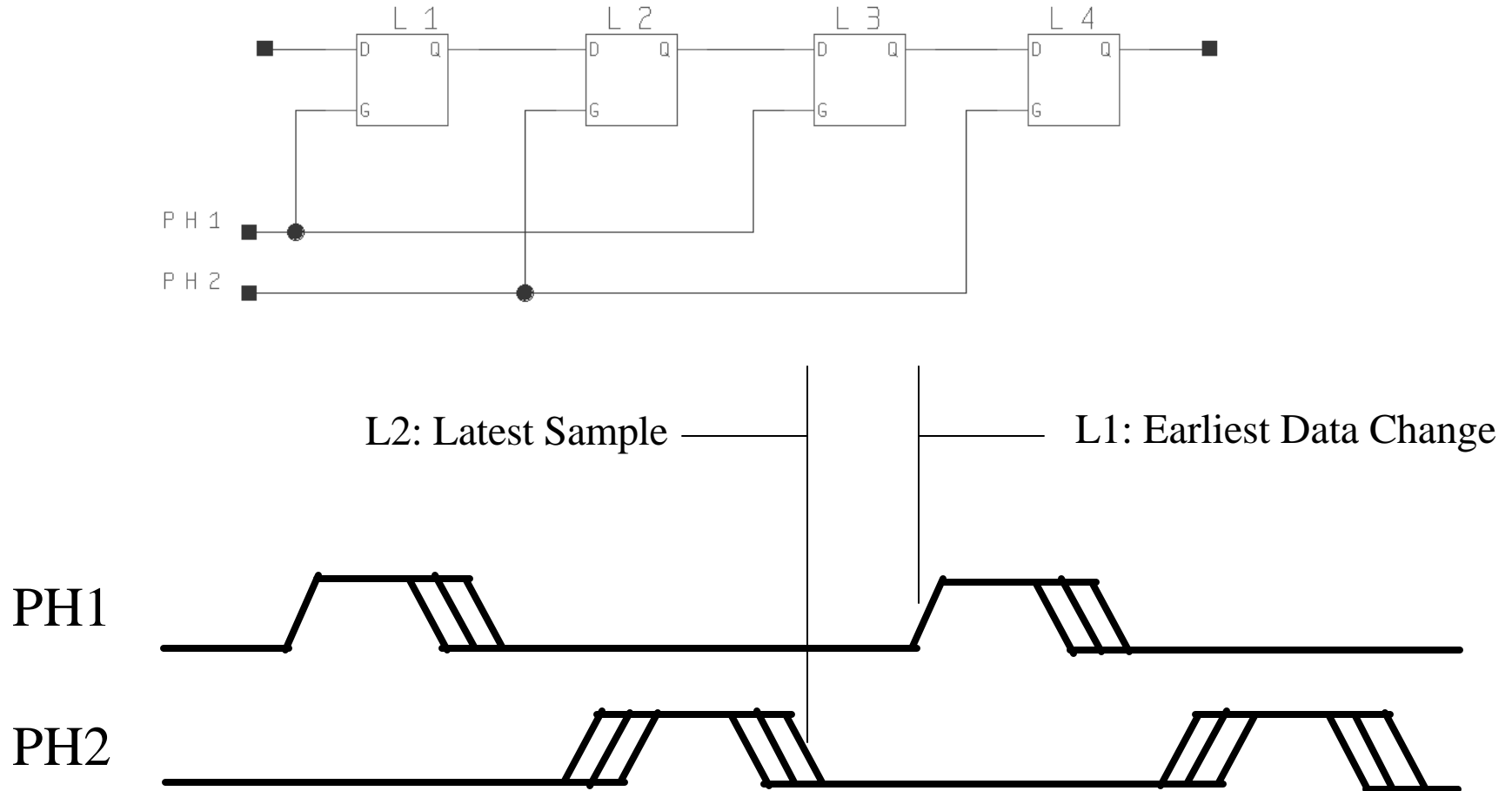
The use of opposite edge clocking makes this circuit skew-tolerant, at the cost of setup time, typically not an issue for shift registers. Additionally, the clock can be run at 1/2 speed, saving power.

Clock Skew, Our Friend



The use of two-phase clocking makes this circuit skew-tolerant, at the cost of setup time.

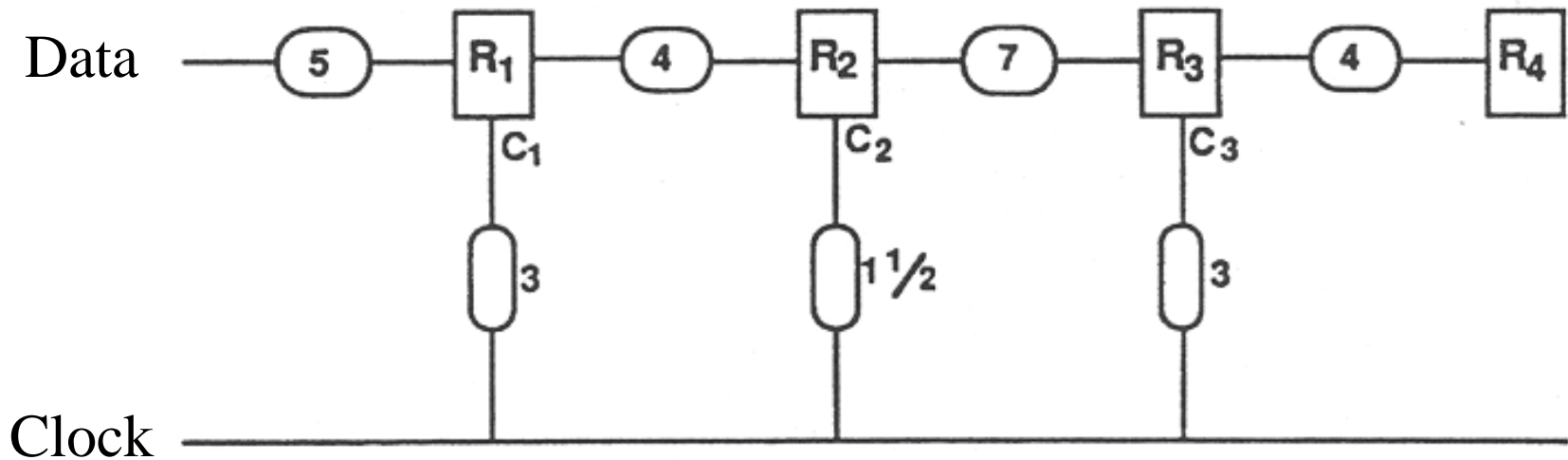
Clock Skew, Our Friend



The use of two-phase, non-overlapping clocking makes this latch-based circuit skew-tolerant.

Clock Skew, Our Friend

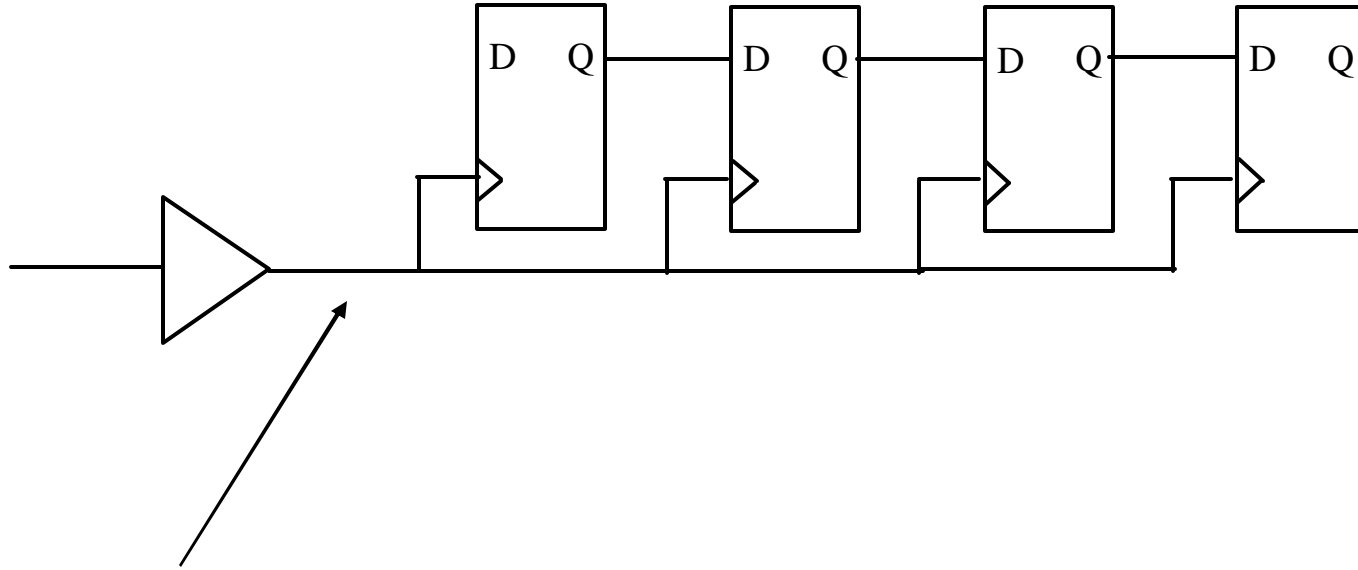
- For pipelined designs with unequal combinational delays between latches or registers, clock skew can provide a better “fit.”



Design Impacts of Clock Skew

- Calculate skews carefully
 - setup time
 - hold time
- Use low-skew clock resources when practical
- Understand skew-tolerant circuit when it is not practical to use low-skew clock resources

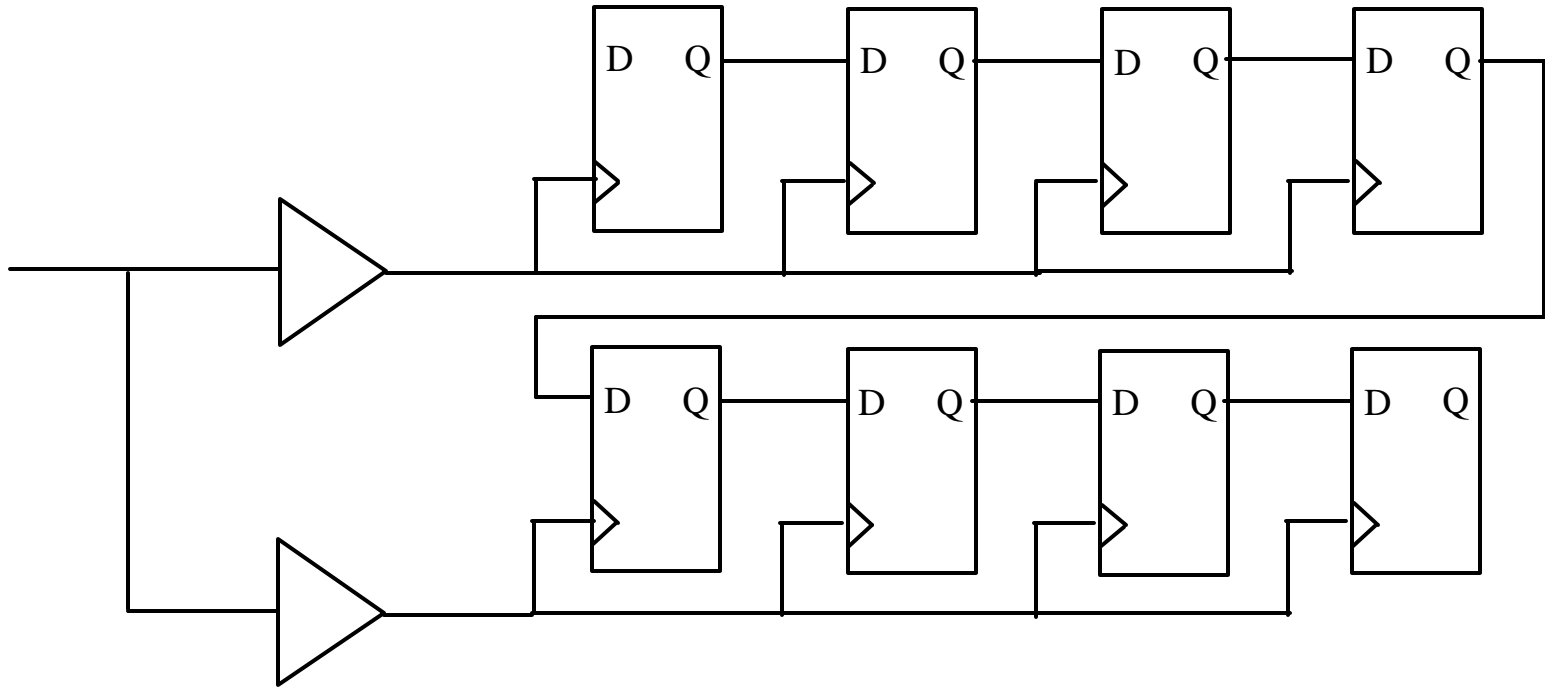
Clock Skew



Normal Routing Resource

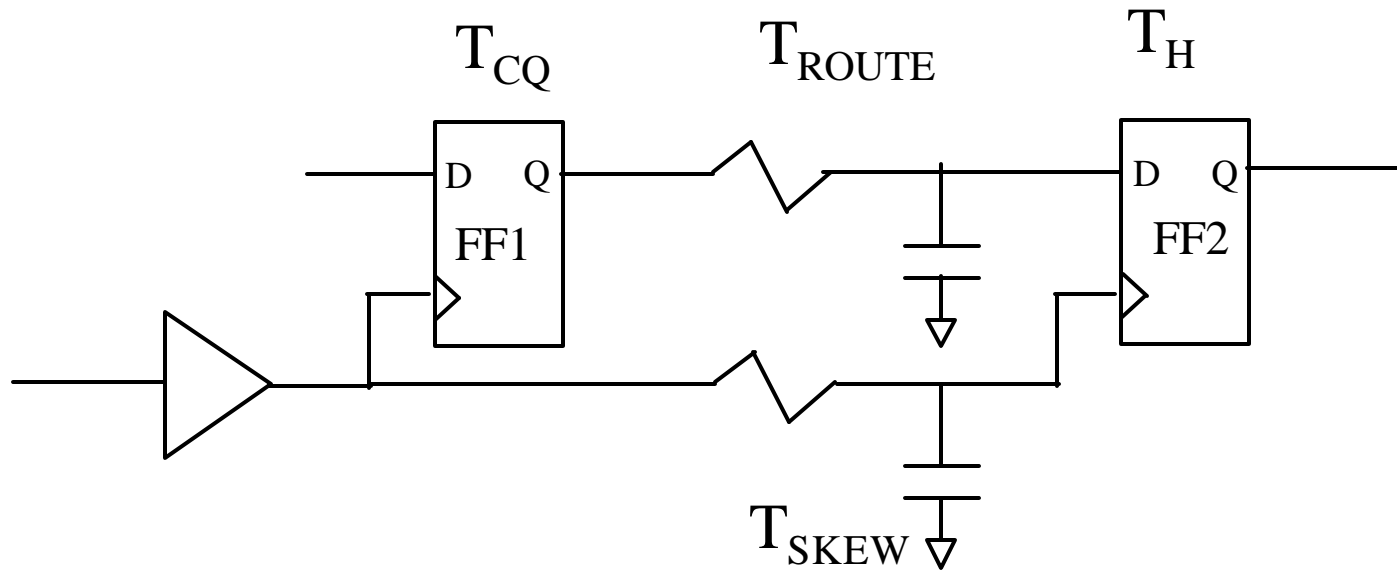
Shift register is given as an example. Also seen in counters and other logic structures.

Clock Skew



- Clock trees are made to increase fanout.
- Not placing buffers and flip-flops on the same row
 - Can increase skew problem.

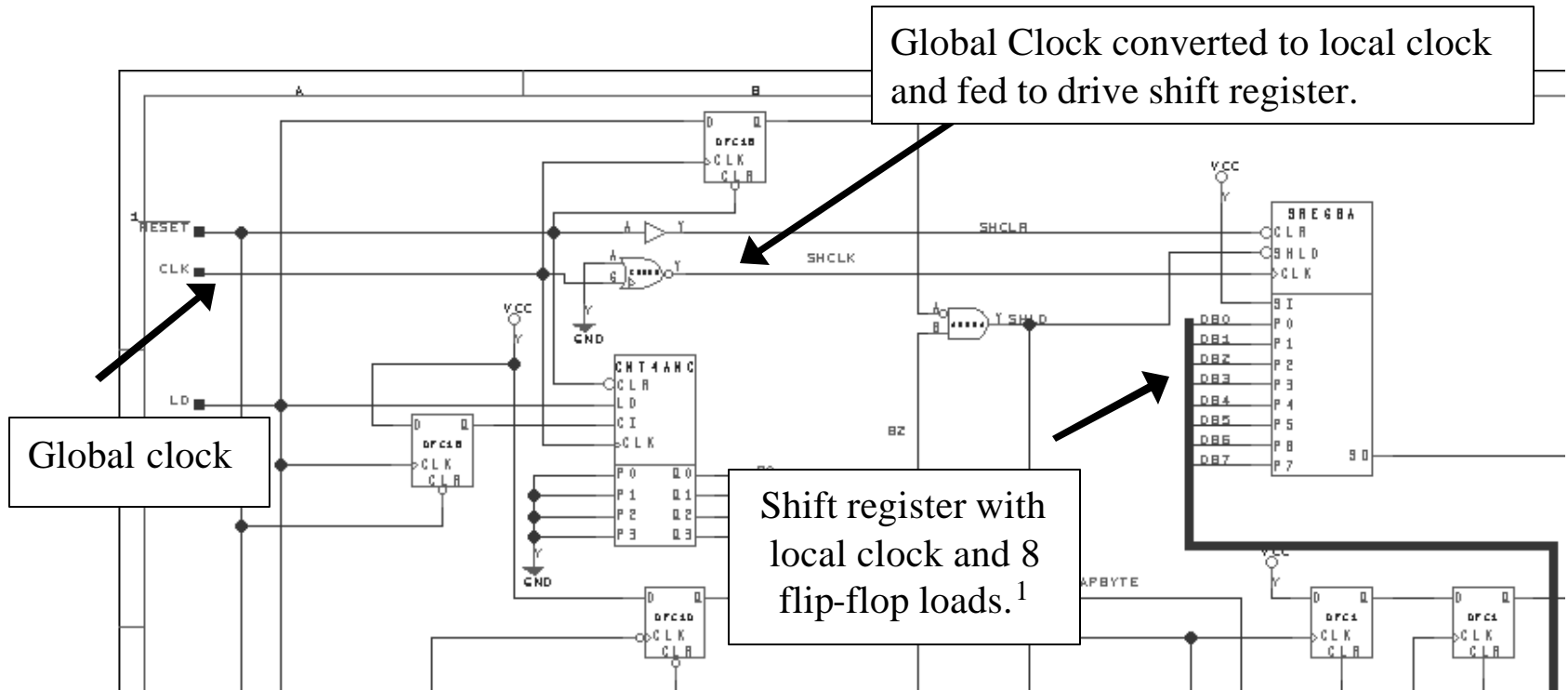
Clock Skew - Timing Model



- Hold time at FF2 is the concern.
 - Worst-case
 - Low V_{IH} FF1
 - High V_{IH} FF2
 - Fast T_{CQ} , T_{Route}
 - High T_{SKEW}
- $T_{CQ} + T_{ROUTE} + T_H > T_{SKEW}$

Design Strategy (2)

Use of Local, High-Skew Clock



¹This project had a design rule of no more than 5 loads on a local, high-skew clock. This was repeatedly violated.

Clock Skew - Timing Analysis

Most static timing analyzers give bounded numbers for min, max.

Just setting “MAX” or “MIN” does not account for variations as a result of fabrication differences, anti-fuse resistance, changes as a result of aging, etc, and will be too liberal.

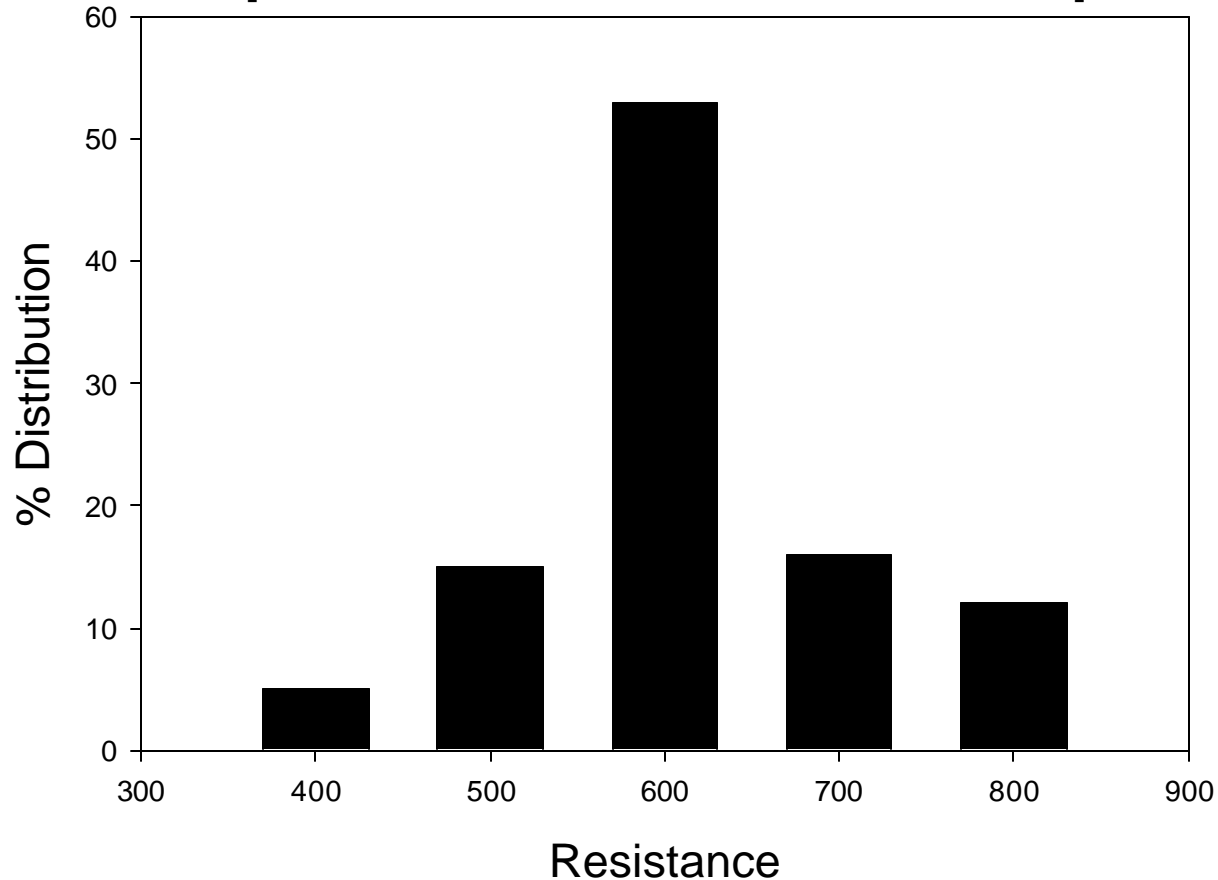
A full MIN/MAX analysis is too conservative since elements near each other on the same die cannot vary that widely. I.e., one part can't be at 4.5VDC, the other at 5.5VDC.

For each environmental condition, it is fair to hold temperature, voltage, fixed.

MIN/MAX will still be a bit conservative, since will range over all manufacturing conditions, not limited to variation within a single die.

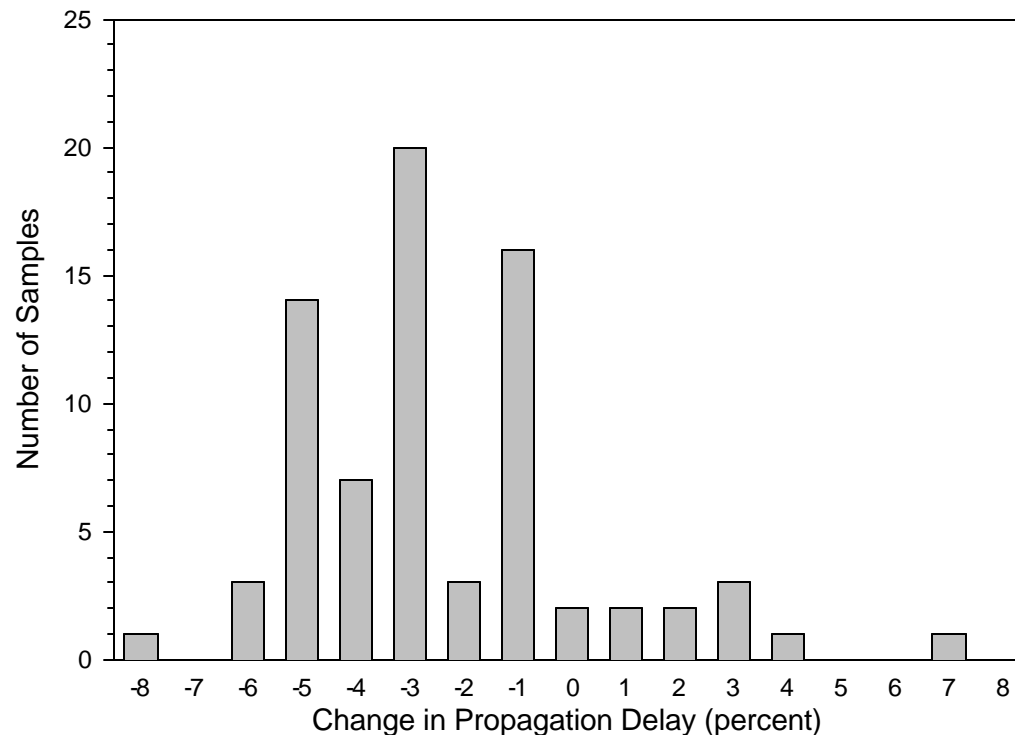
Antifuse Resistance Variation

ONO Antifuse Resistance Distribution
Programming Current = 5mA
[from Antifuse FPGAs, J. Greene, et. al.]



Prop Delay Delta vs. Life

RH1280 Change in Propagation Delay
After 1000 Hour Life Test
Tested at 4.5 volts, 125C



Note: Over a long path, 16 modules + I/O, T_p exceeding 100 ns.

Clock Skew - From VHDL Coding Example

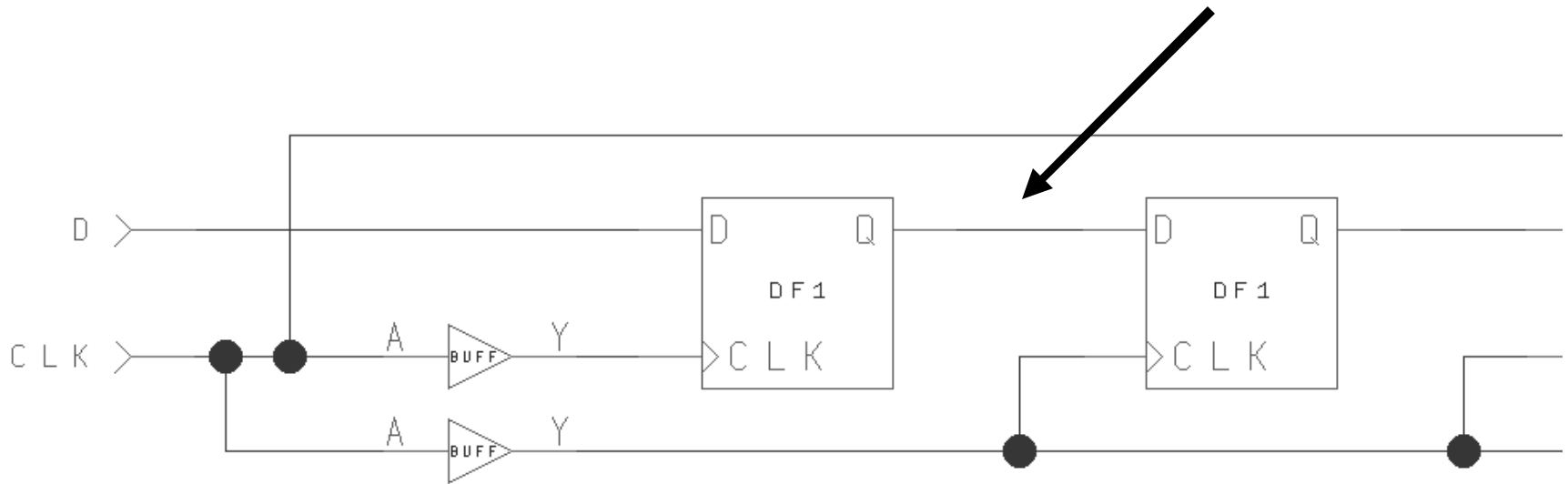
```
Library IEEE;
  Use IEEE.Std_Logic_1164.All;

Entity Skew Is
  Port ( Clk   : In  Std_Logic;
        D     : In  Std_Logic;
        Q     : Out Std_Logic );
End Skew;

Library IEEE;
  Use IEEE.Std_Logic_1164.All;

Architecture Skew of Skew Is
Signal ShiftReg : Std_Logic_Vector (31 DownTo 0);
Begin
  P: Process ( Clk )
    Begin
      If Rising_Edge (Clk)
        Then Q                                     <= ShiftReg(0);
              ShiftReg (30 DownTo 0) <= ShiftReg (31 DownTo 1);
              ShiftReg (31)          <= D;
        End If;
      End Process P;
End Skew;
```

Clock Skew - From VHDL Synthesized Results



Results will depend on coding, directives and attributes, synthesizer, and synthesizer revision.

Here we see that the logic synthesizer generated a poor circuit.

Clock Skew Correction

