Centurion
UNIVERSITY
Shaping Lives...
Empowering Communities...

Advanced RTL Simulation

Centurion University
Shaping Lives...
Empowering Communities...

Abstract

RTL simulations do not consider timing in all aspects and therefore, the results produced by simulation may differ from silicon behavior. Simulation can miss design bugs due to limitations in simulator. These differences are due to incorrect estimation of propagation time between clock and data paths causing data to propagate earlier than in silicon. This problem is known as shoot-thru.

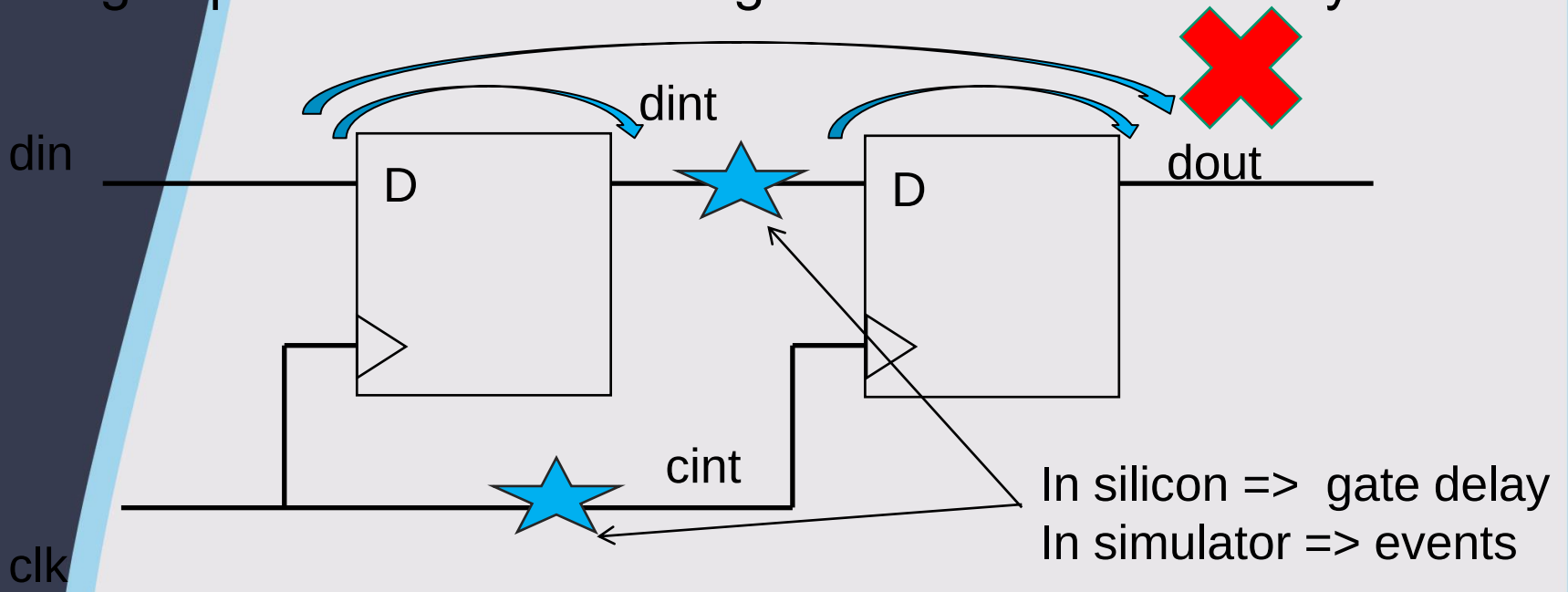
We faced a silicon failure which prompted us to understand how shoot-thru situations occur during simulation, identify these situations with higher accuracy of analysis and fix them with minimal impact using simple RTL coding rules.

We teamed up with Atrenta to automate efficiently the check of those coding rules.

What is Shoot-thru?

When any signal jumps over an *extra* register within one clock cycle.

E.g. Input “din” crosses 2 register within one clock cycle.

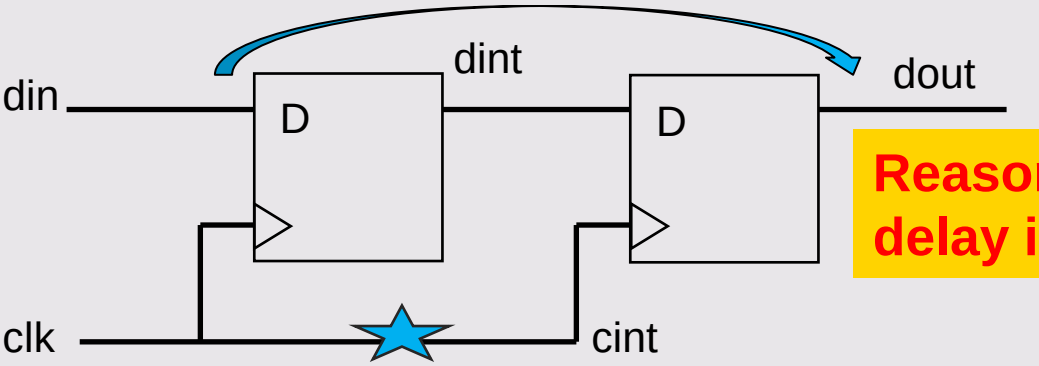


Shoot-thru:

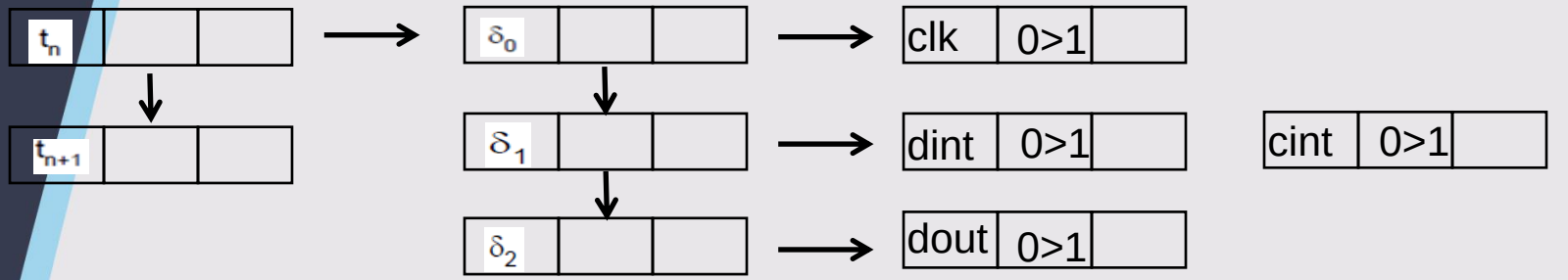
(events in data path) <= (events in clock path)

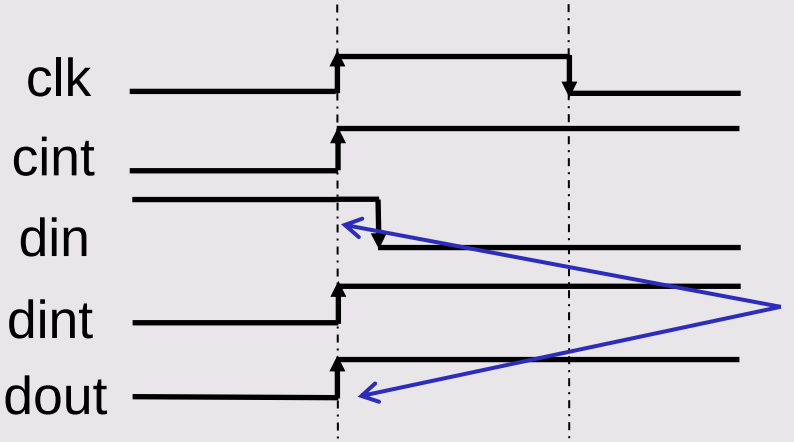
Centurion UNIVERSITY
Shaping Lives...
Empowering Communities...

How Simulation Shoot-thru occurs?



Reason : Extra Delta delay in Clock Path





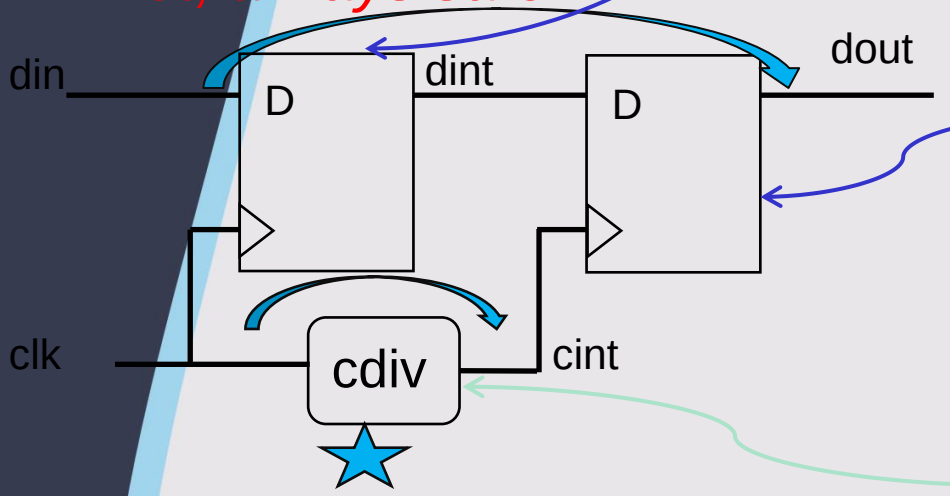
Consequence : “dout” same as “din” within single clock

Will Verilog coding rules help?

blocking assignment in combination blocks and

non-blocking in sequential

Not, always safe….



Extra Delta-delay in Clock Path

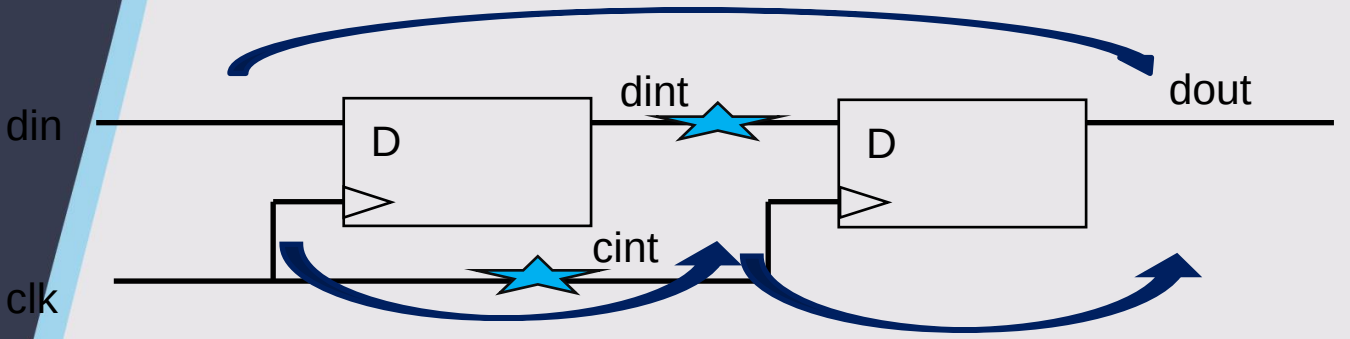
```
// Input data sampling
always @ (posedge clk, negedge rst_n) begin
   if (rst_n==1'b0)
              dint  <= din;
    else
              dint  <= din;
end
// Capture data on divided clock
always @ (posedge cint, negedge rst_n) begin
   if (rst_n==1'b0)
              dout <= 1'b0;
    else
              dout <= dint;
end
// Sequentioal clock generation
always @ (posedge clk, negedge rst_n) begin
   if (rst_n==1'b0)
              cint    <= 1'b0;
else
              cint    <= !cint;
end
```

Non-blocking assignments consume *delta* delay

Case 1: Verification can miss Shoot-thru

- **User specification was to transfer “din” to “dout” in 1 cycle**
- Extra flop was a mistake but not caught by RTL verification.

If Simulation has shoot-thru => “din” will be captured at “dout” in 1 cycle
=> All tests pass as per user specification

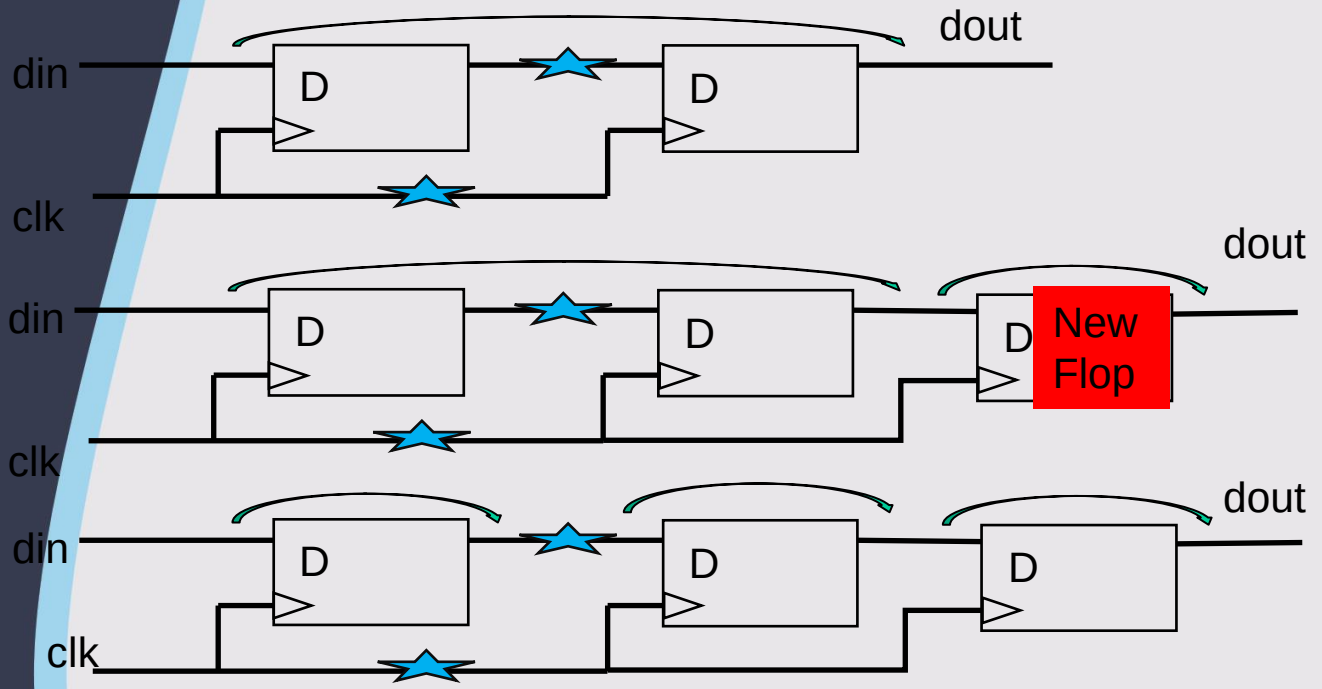


Silicon fails as “din” will be captured at “dout” in 2 cycles
=> *one cycle extra delay vs. Specification*

- Existence of One extra flop causes silicon failure.

Case 2: Verification can miss Shoot-thru

- **User specification was to transfer “din” to “dout” in 2 cycle**
- However, shoot-thru may cause simulation failure
- User fixes it by adding an extra flop



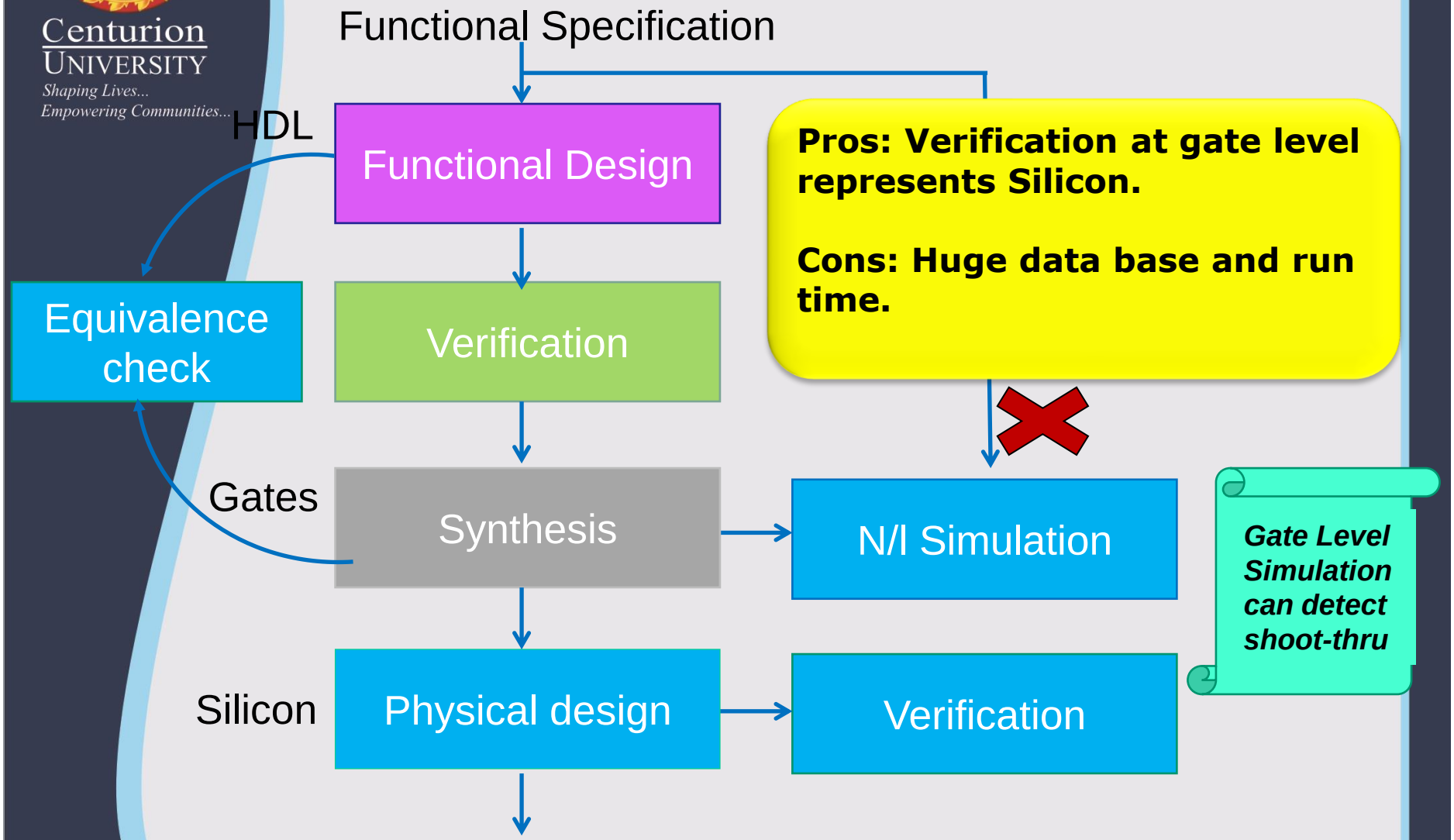
Simulation Fails

Simulation Passes
after rtl changes

Silicon Fails

Centurion University
Shaping Lives...
Empowering Communities...

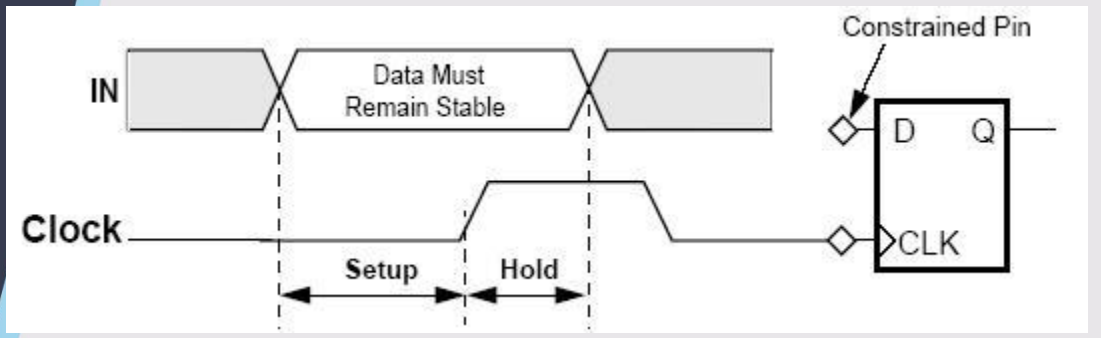
Potential Gap in design flow



So, How to detect shoot-thru issues?

By using an improved simulator?

NO, as root cause of problem is that RTL Simulation has no notion of timing delay & Hold checks.



By doing Gate level simulation?

Yes but it is very costly.

By using robust RTL coding rules?

Yes! this is easy and efficient (see subsequent slides)

Centurion University
Shaping Lives...
Empowering Communities...

Rule 1 - Force scheduling of data

Force **physical** time delay in **Sequential *data* assignments**

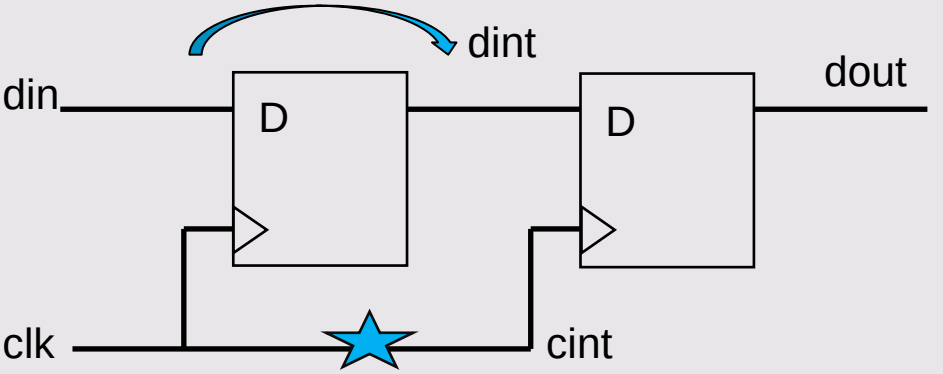
Signal is delayed in ‘physical time’, instead of ‘delta’ time.

Verilog

```
dint <= #1 din ;
```

VHDL

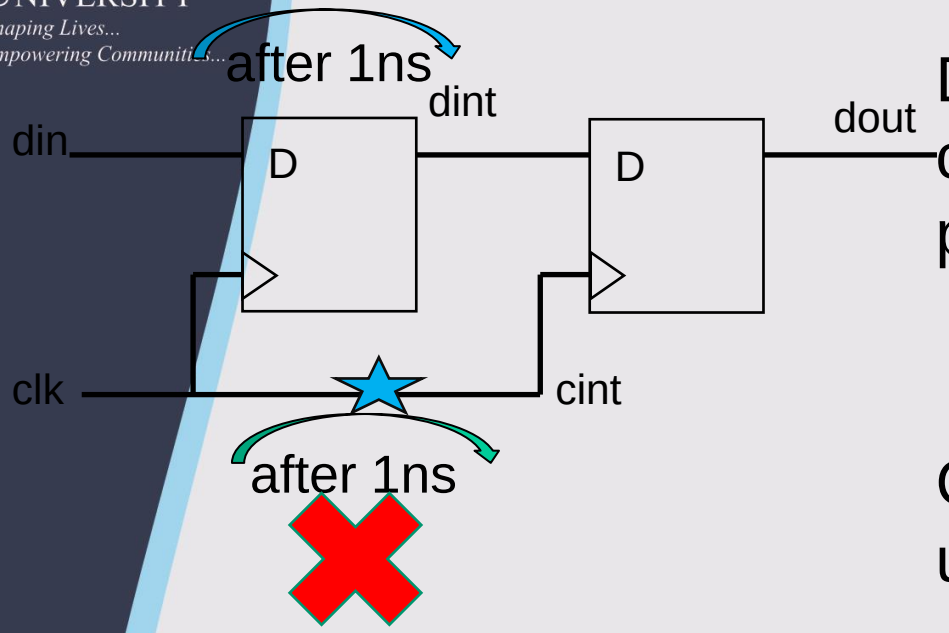
```
dint<= din after 1 ns;
```



This adds an extra delay in data path

Rule 2 - No ‘physical delay’ in clock path

Centurion University
Shaping Lives...
Empowering Communities...



Delay on clock paths competes with delay on Data path

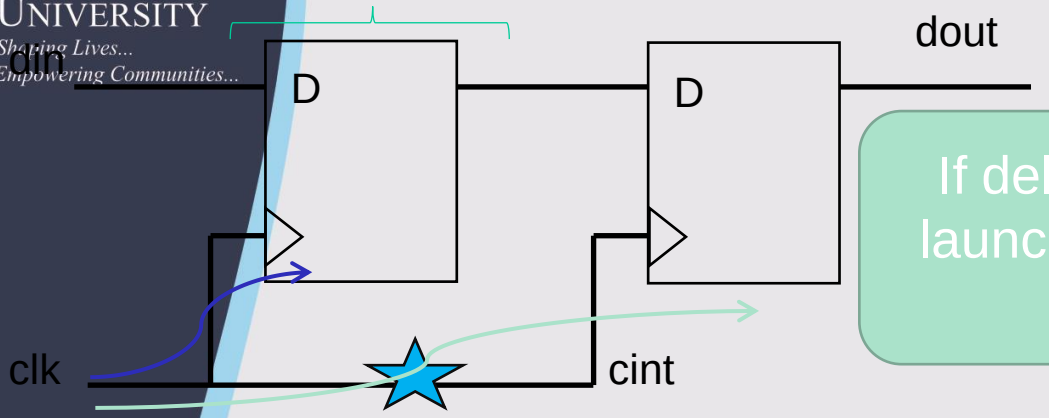
Clock vs Data scheduling still un-reliable

No unwanted delay on clock path

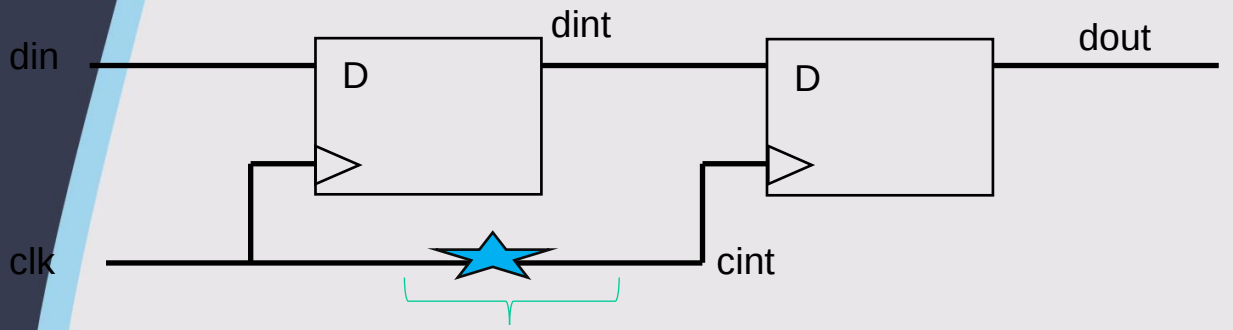
Rule1 + Rule2 ensures no risk of shoot-thru i.e data is updated after clock

Automation with SpyGlass RTL checker

Rule 1 – Reports missing needed delay in data path



If delta delay is different between launch and capture flop, add delay on launch flop



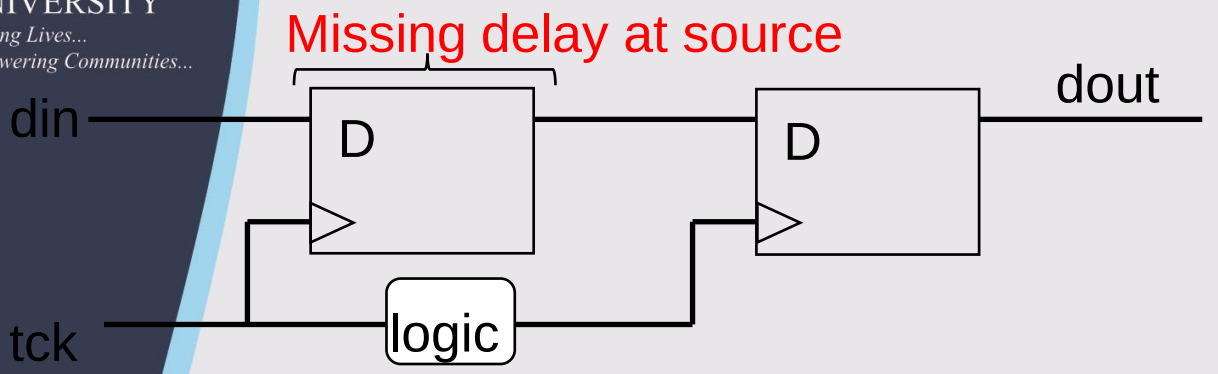
Rule 2 – Reports extra delay (Physical) present in clock path

Easy to identify shoot-thru prone RTL

Centurion UNIVERSITY
Shaping Lives...
Empowering Communities...

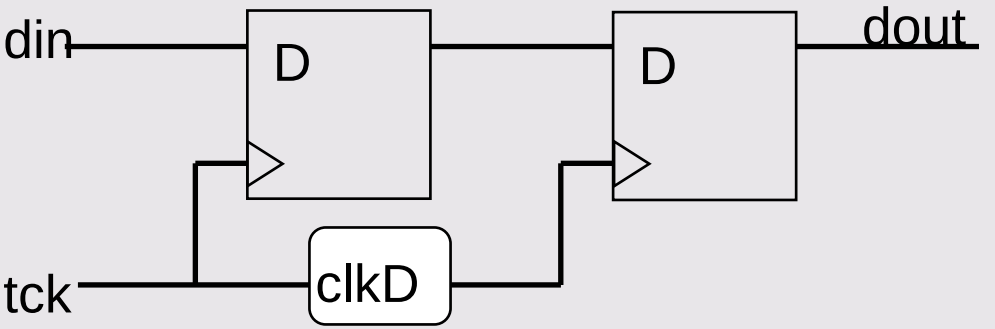
Checker catches Real Silicon Issues

Missing delay was causing silicon failure



Buffer, Clock gating etc.

- Unintentional delay in clock path found on another chip
  - User specified physical delay on source flop - Still problem occurred due to physical delay specified on clock path



Extra delay in clock divider

Conclusion

- Simulators create events and schedule them, but with no notion of timing checks across data and clock events
- Shoot-thru impacts RTL Simulation and may cause false simulation behavior, masking the real issues
- Simple RTL coding rules can prevent the problem to occur
  - Possibly add physical delays on data path to force scheduling
  - Avoid physical delays on clock path
- Spyglass RTL checker can help you to efficiently check those rules and removes need of costly gate level simulations.
- Next Steps
  - Extend the SpyGlass checks to ensure that every output interface is having delayed assignment