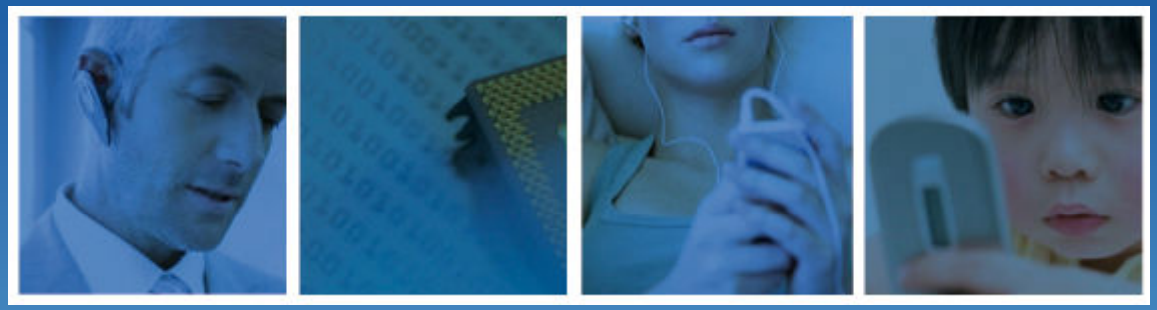
azuro | It's about time



Clock Concurrent Optimization

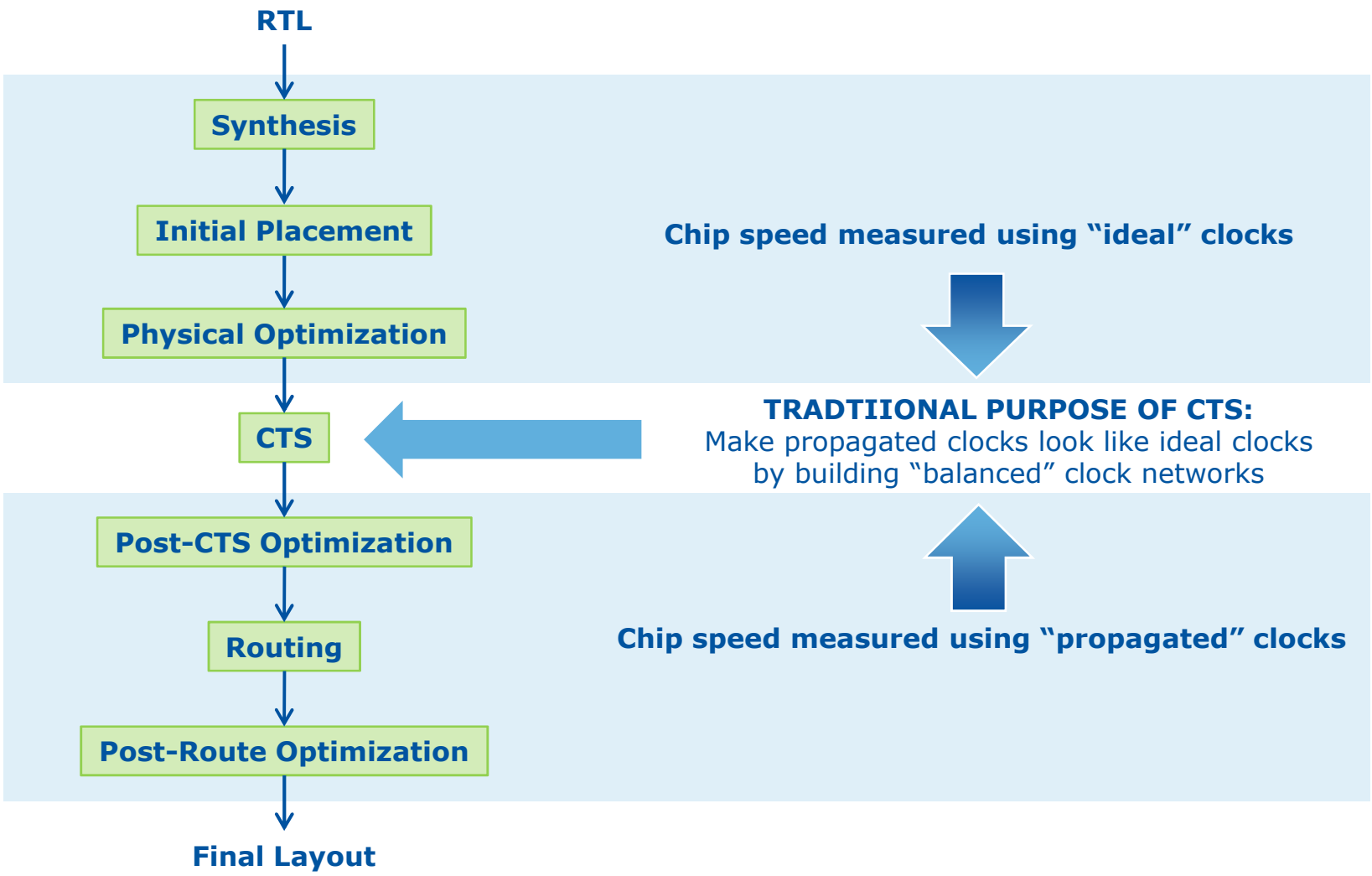
Paul Cunningham, Marc Swinnen, Steev Wilcox

Electronic Design Processes
April 10, 2009

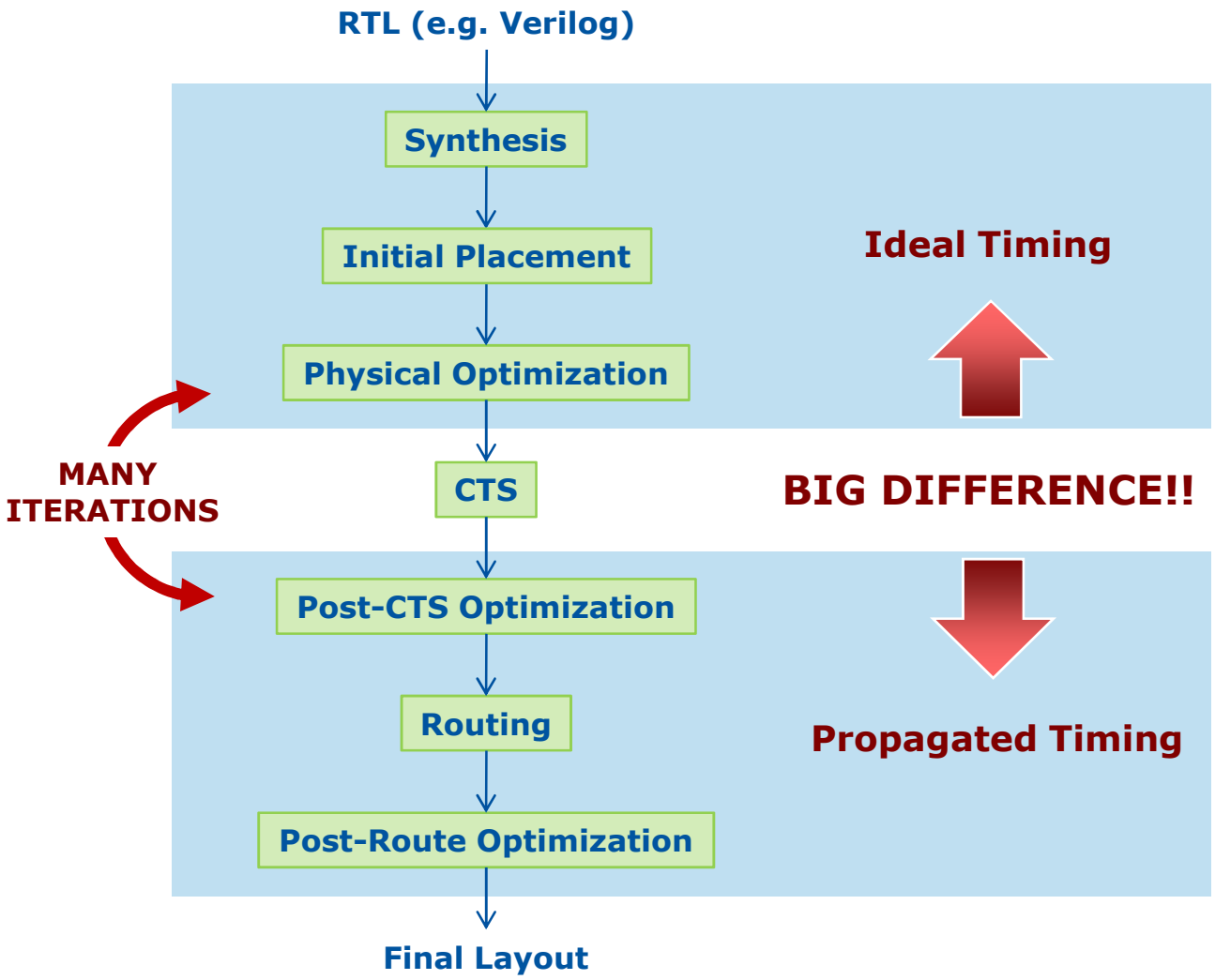
azuro | It's about time

The Clock Timing Gap

Traditional Design Flows



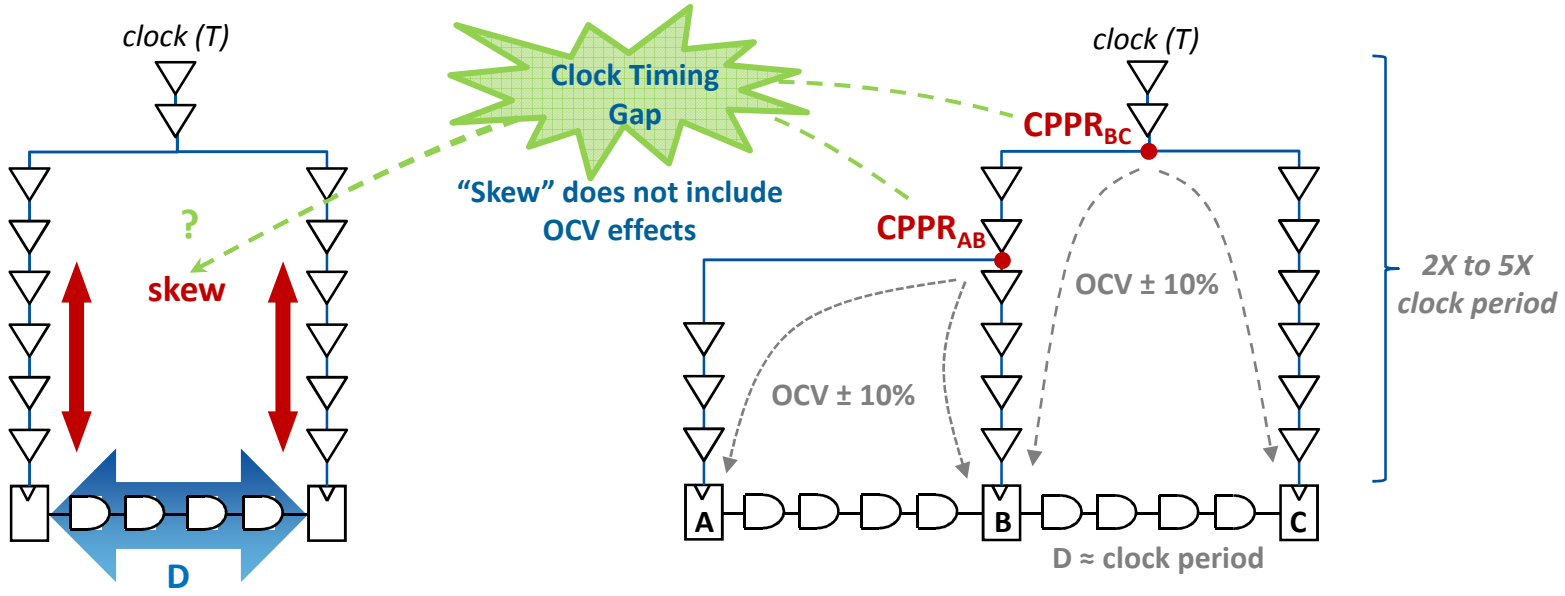
Reality Today



- Clock Gating
- Clock Muxing
- Clock Generators
  - Especially for hold
- Complex Scan Chains
- OCV derates and CPPR
- Multi-corner
  - Especially for hold
- Multi-mode

Technology Trends
Opening the Clock Timing Gap

Trends Driving the Clock Timing Gap



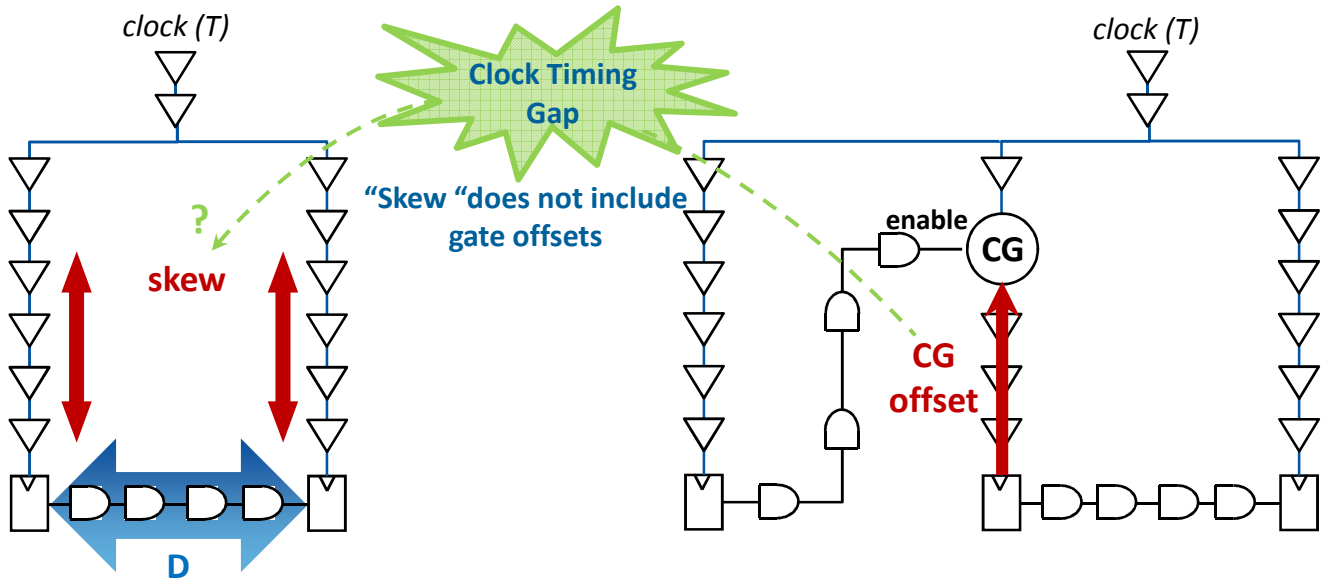
Traditional Optimization

D < T - skew

OCV

- **OCV affects each pair of FFs differently (CPPR)**
- **OCV effect can be very big - e.g. 10% of 3T**
- **CTS cannot predict OCV impact**
- **So, "skew=0" does *not* mean FFs are really balanced**

Trends Driving the Clock Timing Gap



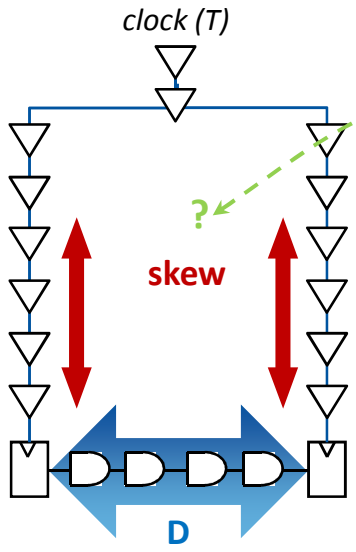
Traditional Optimization

D < T - skew

Clock Gating

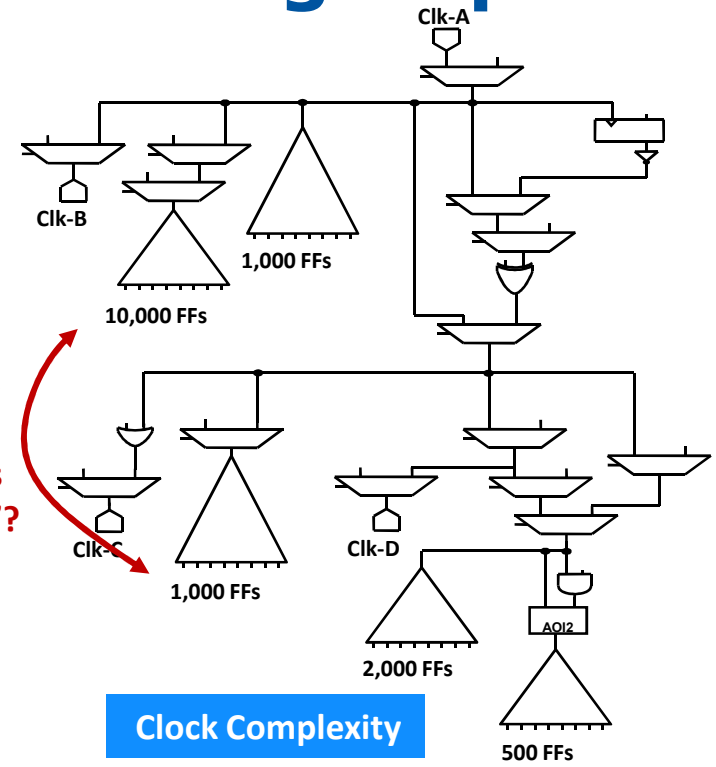
- Clock gates are supposed to have a very big skew
- Traditional optimization tries to prevent this by 'cloning' the gates and pushing them down the tree
- Traditional approach cannot correctly optimize or time CG enable paths

Trends Driving the Clock Timing Gap



Clock Timing Gap

"Skew "does not include interclock skew



Clock Complexity

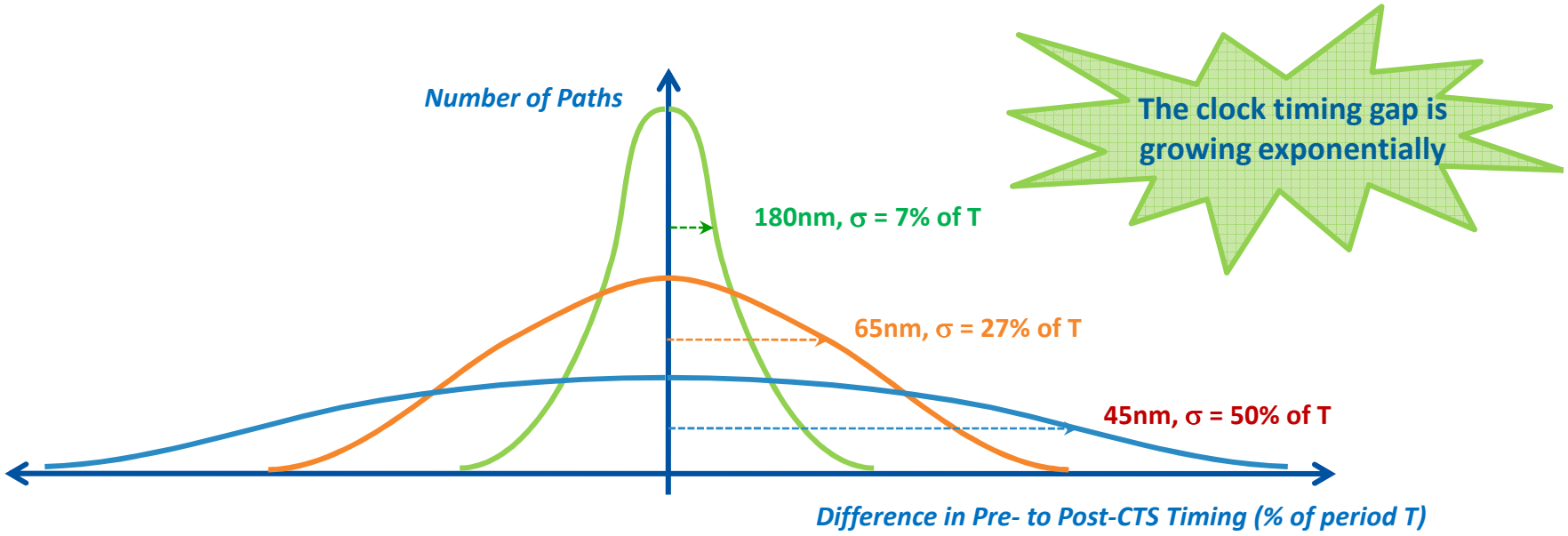
Traditional Optimization

D < T - skew

- Clock balancing becomes very difficult, or even theoretically impossible
- Requires extensive manual intervention
- Final clock implementation is very different from original, ideal assumptions

The Clock Timing Gap is Growing

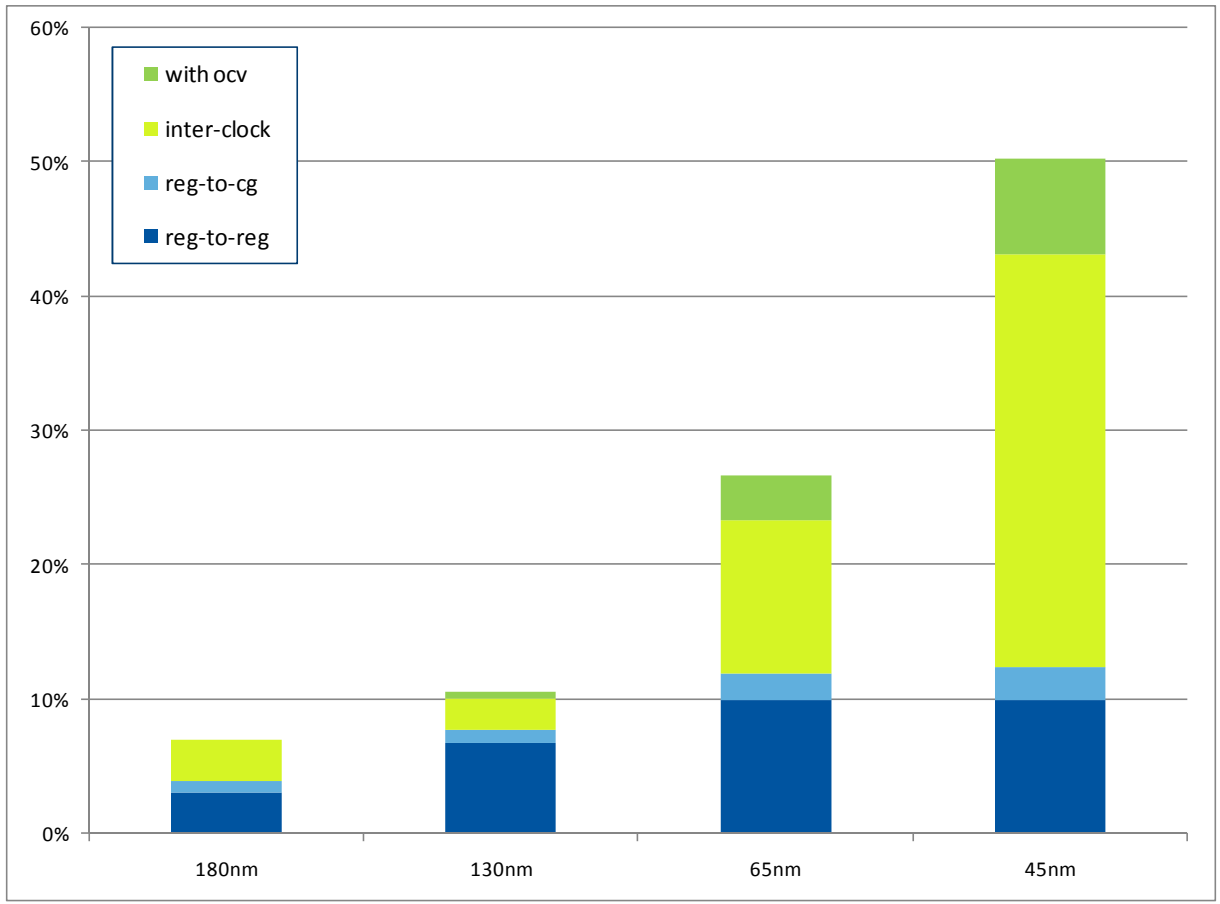
Propagated clocks timing and ideal clocks timing are diverging



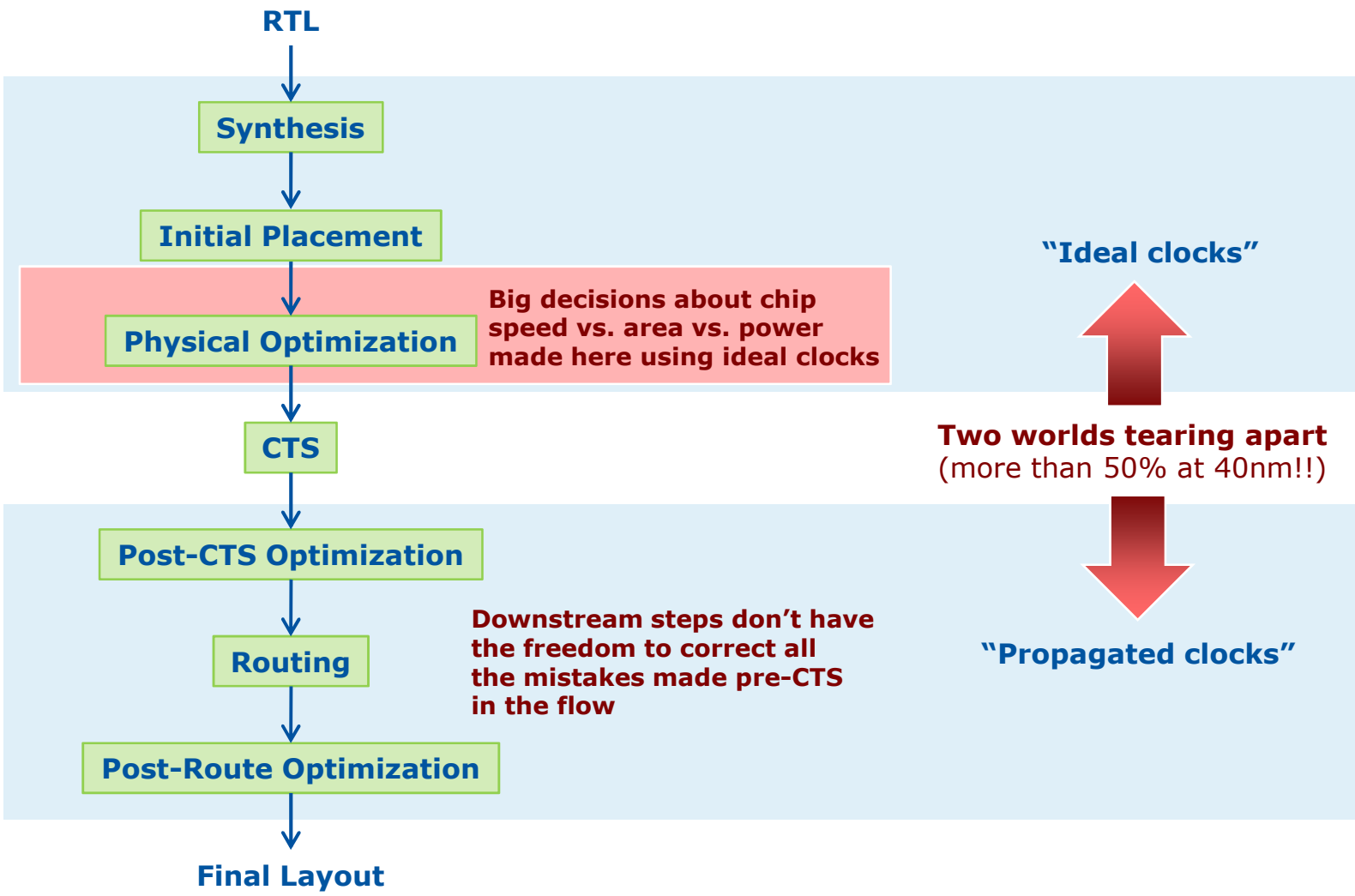
azuro | It's about time

Ideal vs. Propagated Clocks Timing Gap

- **Difference between ideal and propagated timing across 60 chips**
  - Top 10% worst violating paths
  - Difference measured as a %age of clock period



Key Limitation of Traditional Flows

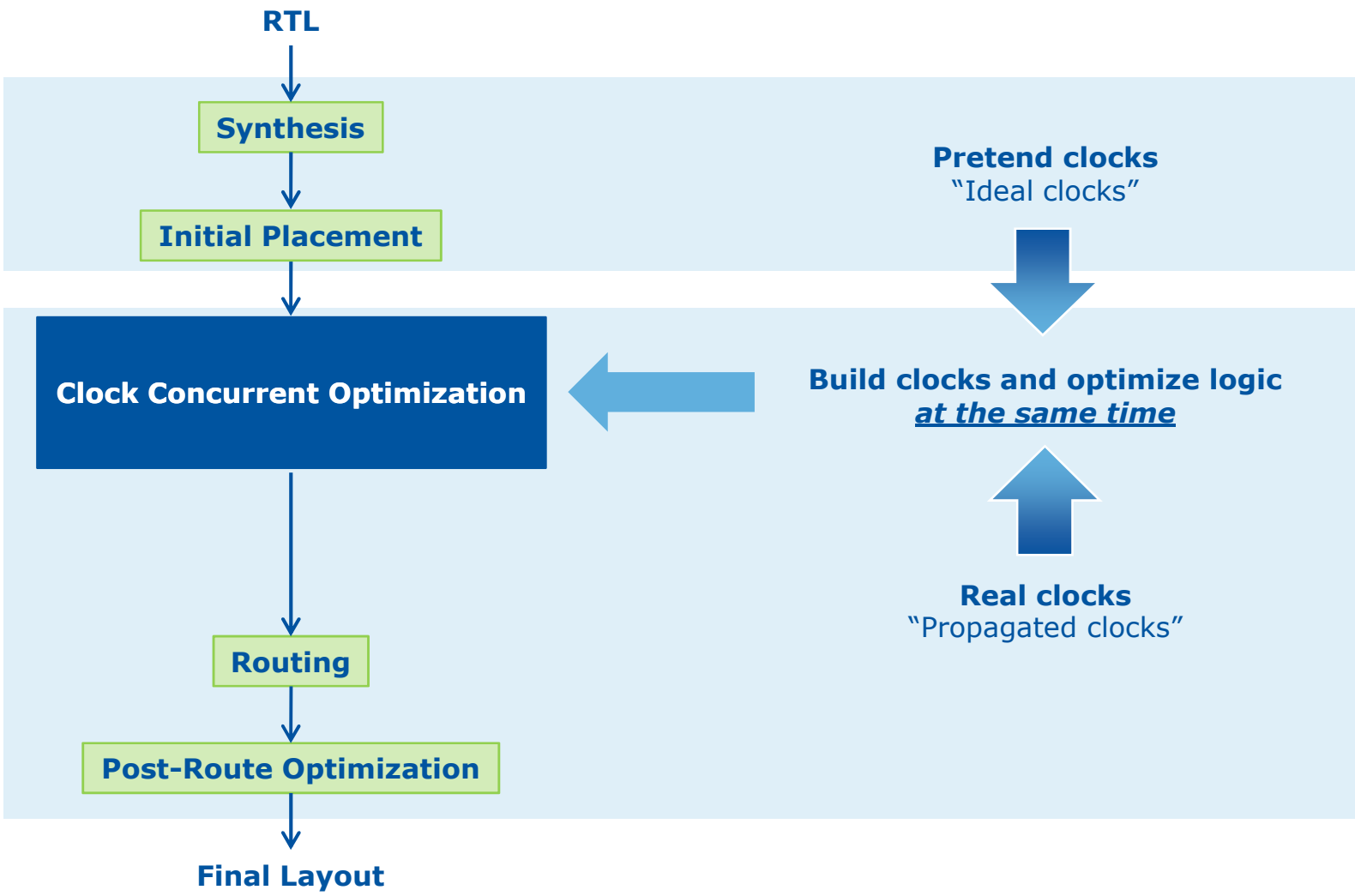


The Key Problems

- **Physical timing optimization today is all based on ideal clocks timing**
  - Timing opt is based on wrong information (like wire load models in the past)
  - Cannot see the real timing situation

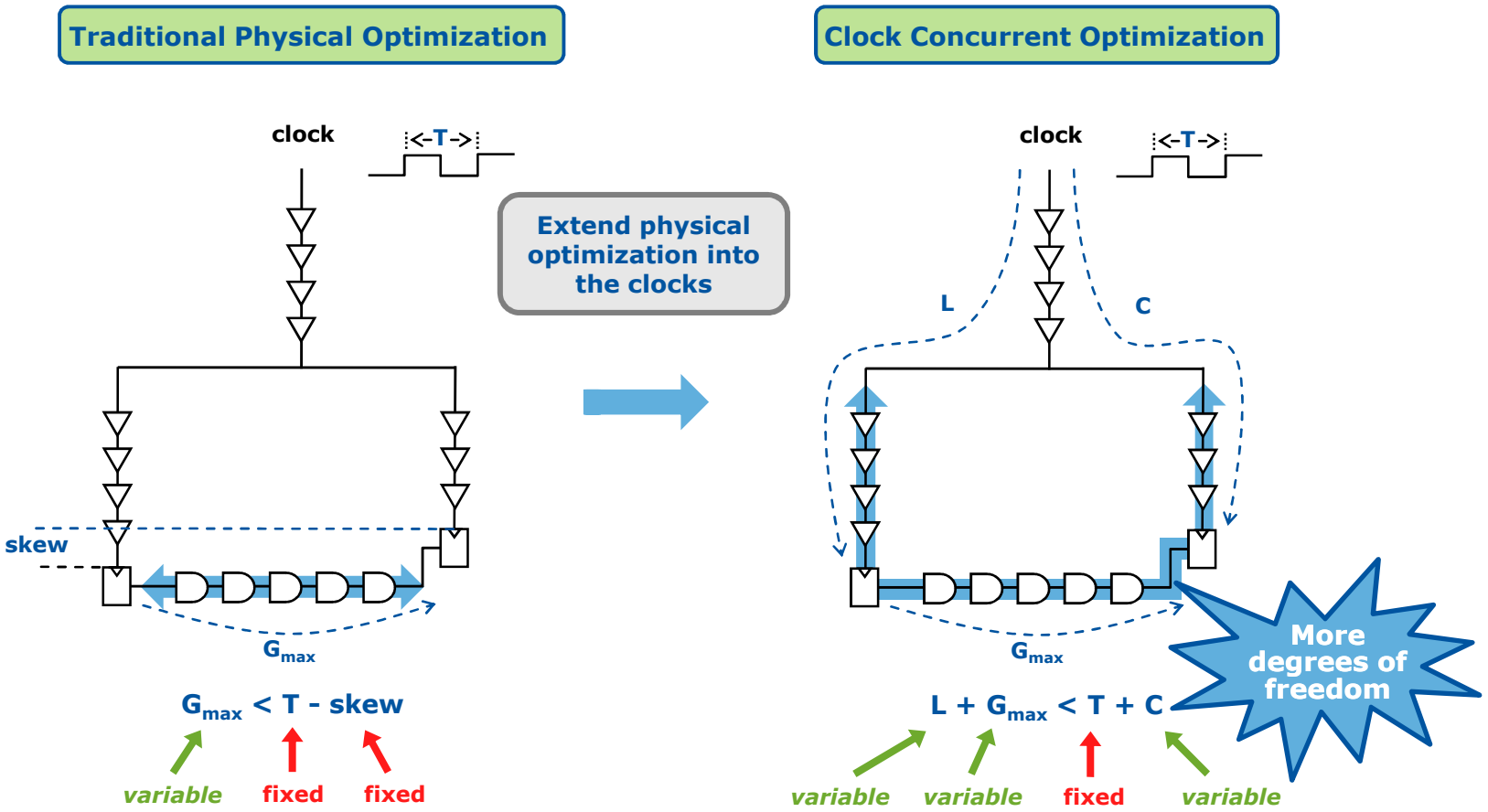
- **Clock balancing is not achievable, not necessary, and not helpful**
  - Even if CTS skew=0, Propagated timing ≠ Ideal timing
  - Clock balancing imposes severe restrictions on timing optimization – for no benefit

Solution: Clock Concurrent Optimization

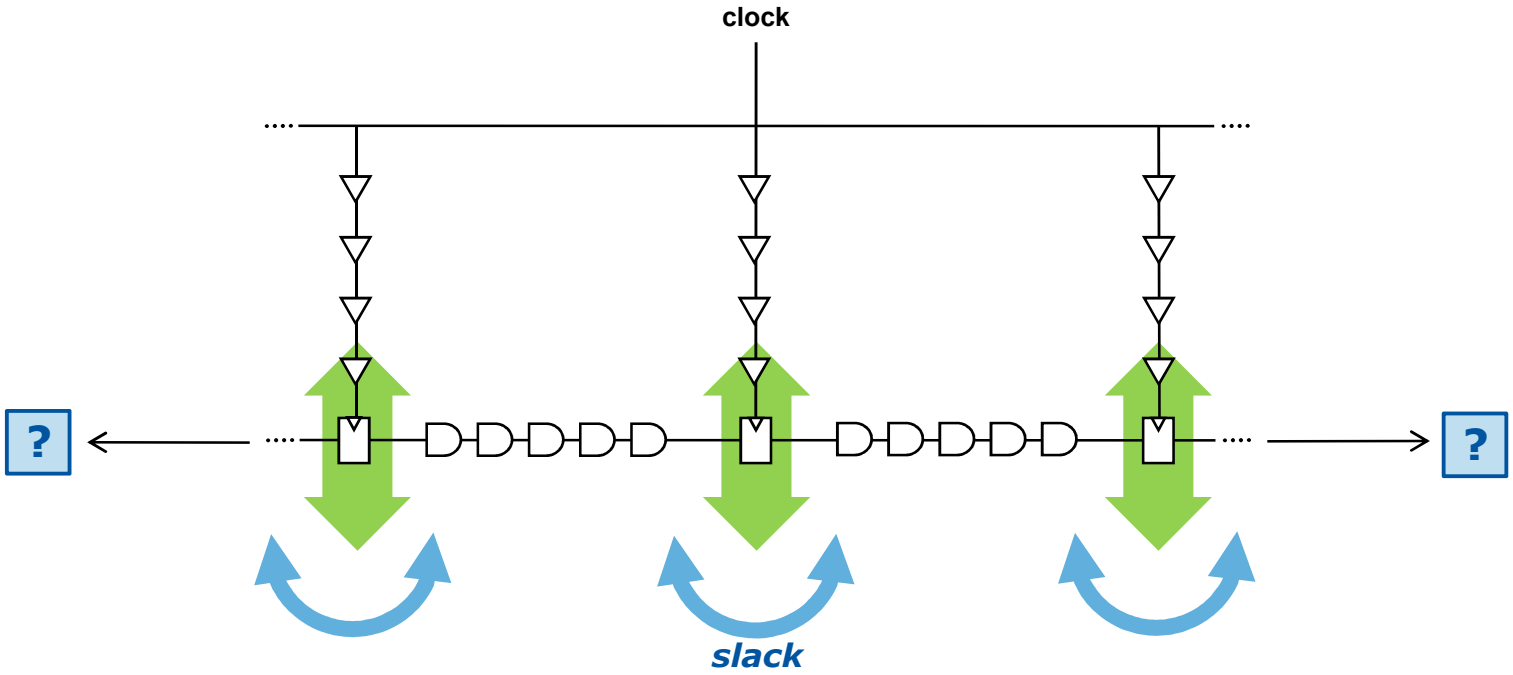


Clock Concurrent Optimization

Clock Concurrent Technology

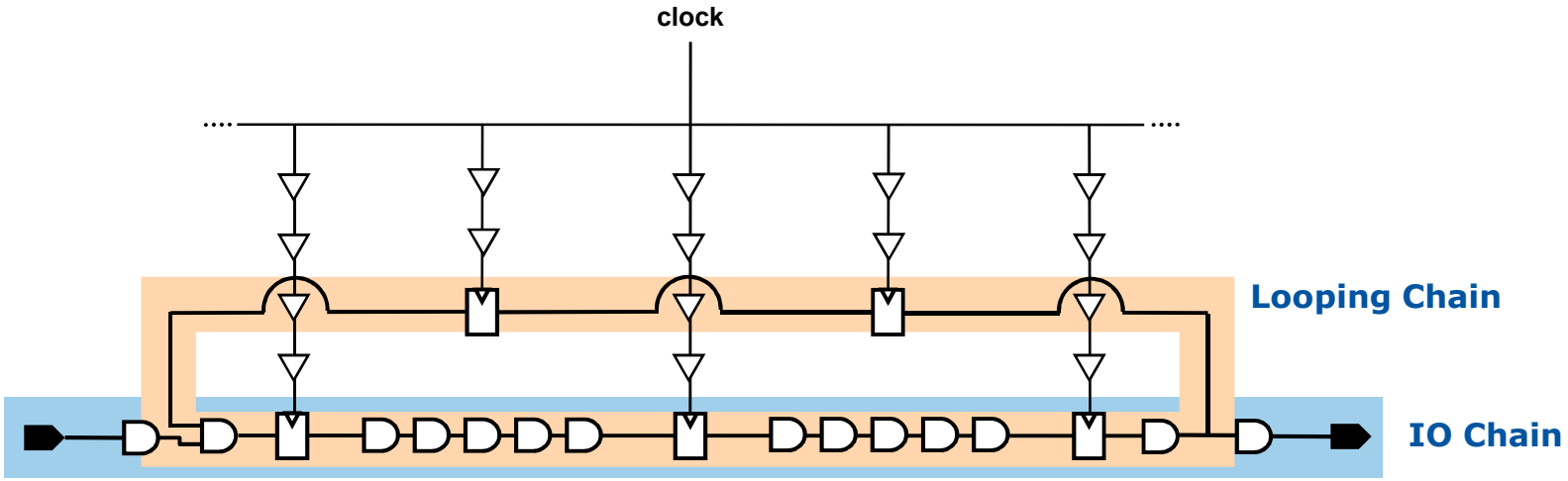


Time Borrowing in Clock Concurrent Opt.



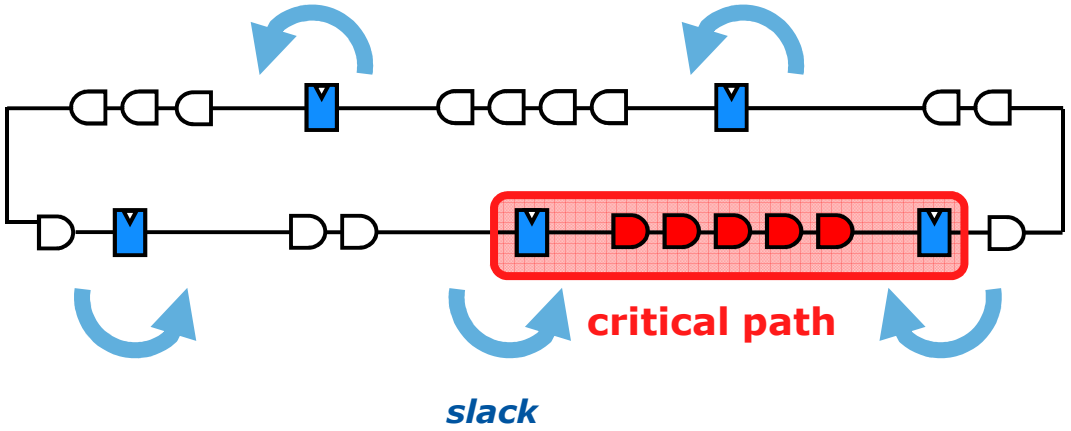
Using CC-Opt, slack can flow across register boundaries

Logic Chains Limit Time Borrowing



Speed is Not Limited by the Critical Path

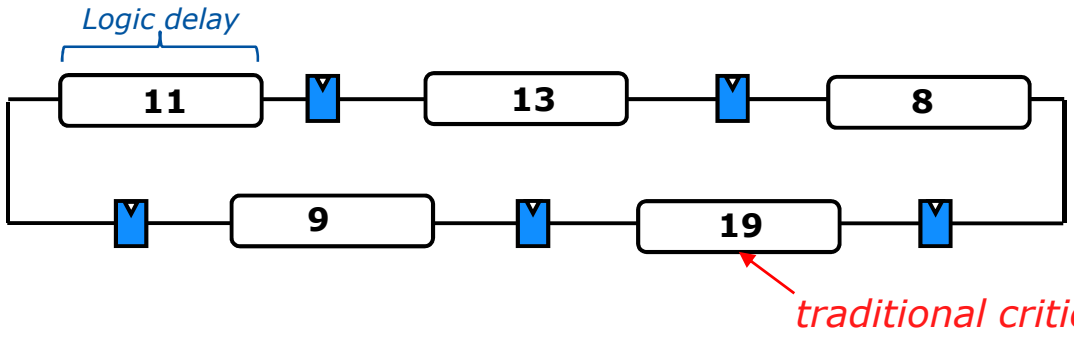
- The "critical path" does NOT limit the chip speed
- CC-Opt can easily move slack along a chain to where it is needed



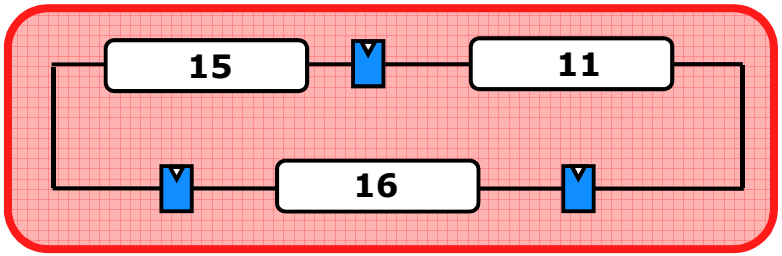
- CC-Opt will optimize "non-critical" paths to create spare slack

Speed is Limited by the Critical Chain

- **The "CRITICAL CHAIN" is the focus of CC-Opt**
  - Critical chain is the chain with the longest delay/stage



$$\frac{\text{Delay}}{\text{Stage}} = \frac{11+9+19+8+13}{5} = 12$$

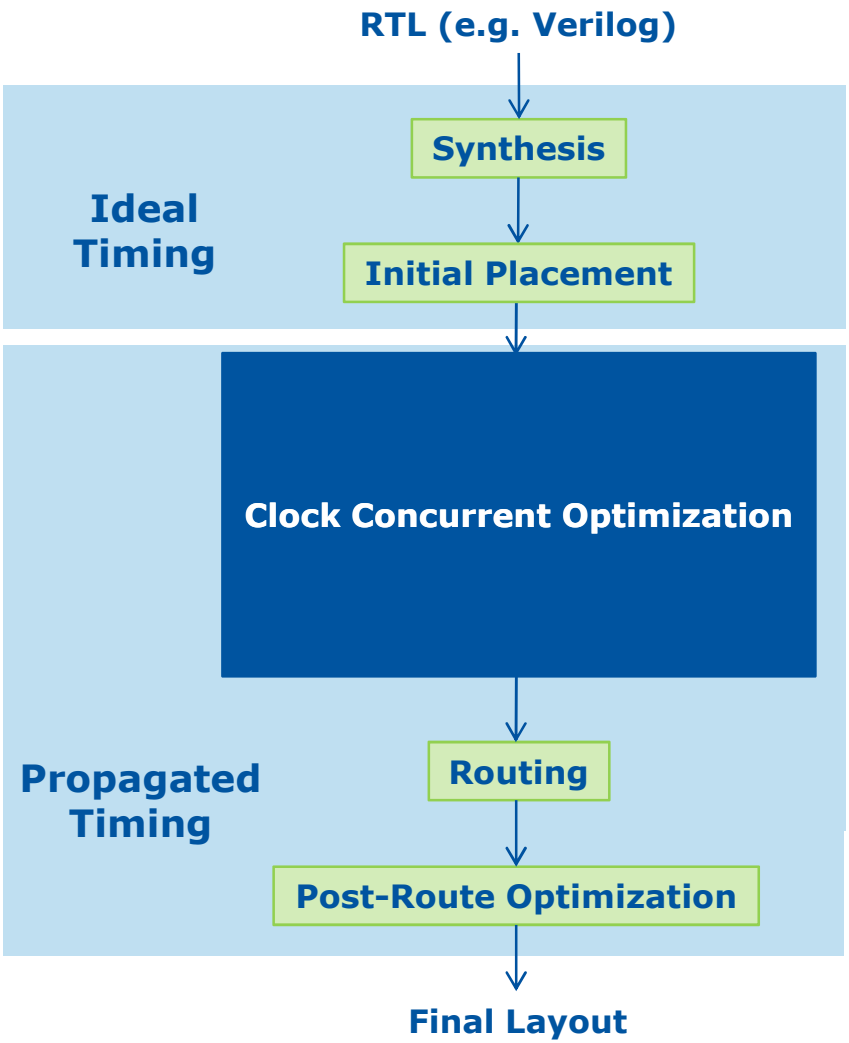


critical chain

$$\frac{\text{Delay}}{\text{Stage}} = \frac{15+16+11}{3} = 14$$

CC-Opt Benefits

Summary of CC-Opt



- **Build clocks directly for timing not skew balancing**
  - Consider setup and hold timing
  - Understand OCV timing
  - Understand clock gate timing
  - Understand clock mux timing
  - Understand clock generator timing
  - Understand multi-corner
  - Understand multi-mode
- **Eliminate need to configure any skew groups**
  - Skew groups are just a work-around for a broken flow!

Key Benefits of CC-Opt.

- **Up to 20% increase in clock speed**
  - Fundamentally more degrees of freedom during optimization
    - All the benefits of useful skew and more!
  - Directly targets propagated timing

- **Accelerated timing closure**
  - No requirement to configure any skew groups
  - Automatically handles clock muxing, clock gating, clock generators, OCV, multi-corner (setup & hold), and multi-mode

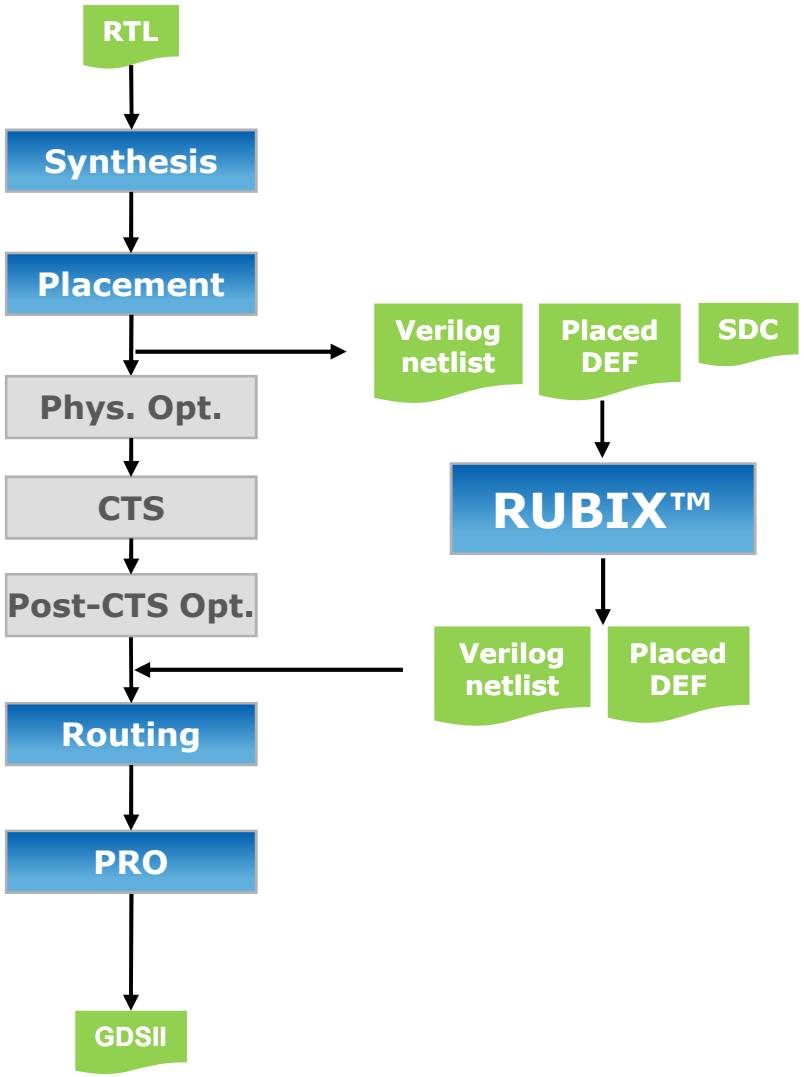
- **Reduced iterations to the frontend**
  - No need manually "retime" logic across register boundaries

- **Reduced IR-drop**
  - Clocks are not balanced!

- **Reduced power**
  - Clock buffers are only used where it is necessary for timing

Rubix™ - An Implementation

Rubix™ Flow and Key Features



- **Full industry standard STA**
  - SDC constraint format
  - Multi-corner and multi-mode
  - OCV derates and CPPR
- **Global routing**
  - Ability to export "route guides"
- **Physical Optimization**
  - Timing-driven incremental placement
  - Timing-driven high-fanout net buffering
  - Cell sizing and logic transformations
  - Legalization
- **Clocks**
  - Comprehensive skew group support (can mix and overlap with timing windows based CTS)
- **Multi-voltage**
  - Clock buffering and net buffering across voltage islands
- **Timing driven scan-chain reordering**
  - Setup and hold aware

Thanks!

For more information see CC-Opt White Paper at

www.azuro.com