The Clock Timing Gap
Traditional Design Flows

- RTL
- Synthesis
- Initial Placement
- Physical Optimization
- CTS
- Post-CTS Optimization
- Routing
- Post-Route Optimization
- Final Layout

Chip speed measured using “ideal” clocks

TRADITIONAL PURPOSE OF CTS:
Make propagated clocks look like ideal clocks by building “balanced” clock networks

Chip speed measured using “propagated” clocks
Reality Today

- RTL (e.g. Verilog)
- Synthesis
- Initial Placement
- Physical Optimization
- CTS
- Post-CTS Optimization
- Routing
- Post-Route Optimization
- Final Layout

> Ideal Timing

> BIG DIFFERENCE!!

> Propagated Timing

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

© Azuro, Inc. 2009
Technology Trends
Opening the Clock Timing Gap
Trends Driving the Clock Timing Gap

- Clock Timing Gap
- "Skew" does not include OCV effects
- 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

Traditional Optimization

D < T - skew

© Azuro, Inc. 2009
Trends Driving the Clock Timing Gap

Clock Timing Gap

Skew does not include gate offsets

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 (T)

1,000 FFs

10,000 FFs

Clk-A

Clk-B

Clk-C

Clk-D

Are all FFs “balanced”? 

“Skew” does not include interclock skew

Traditional Optimization

D < T - skew

Clock Complexity

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

© Azuro, Inc. 2009
The Clock Timing Gap is Growing

Propagated clocks timing and ideal clocks timing are diverging

Number of Paths

180nm, \( \sigma = 7\% \) of \( T \)

65nm, \( \sigma = 27\% \) of \( T \)

45nm, \( \sigma = 50\% \) of \( T \)

The clock timing gap is growing exponentially

Difference in Pre- to Post-CTS Timing (% of period \( T \)
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

![Chart showing the difference between ideal and propagated timing across different chip generations.](chart.png)
Key Limitation of Traditional Flows

Big decisions about chip speed vs. area vs. power made here using ideal clocks

RTL

Synthesis

Initial Placement

Physical Optimization

Big decisions about chip speed vs. area vs. power made here using ideal clocks

Initial Placement

Physical Optimization

CTS

Post-CTS Optimization

Routing

Post-Route Optimization

Final Layout

Two worlds tearing apart
(more than 50% at 40nm!!)

“Ideal clocks”

“Propagated clocks”

Downstream steps don’t have the freedom to correct all the mistakes made pre-CTS in the flow
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

- RTL
  - Synthesis
    - Initial Placement
      - Clock Concurrent Optimization
        - Routing
          - Post-Route Optimization
            - Final Layout

Pretend clocks
“Ideal clocks”

Build clocks and optimize logic
at the same time

Real clocks
“Propagated clocks”
Clock Concurrent Optimization
Clock Concurrent Technology

Traditional Physical Optimization

Clock Concurrent Optimization

Extend physical optimization into the clocks

More degrees of freedom

$G_{\text{max}} \leq T - \text{skew}$

$G_{\text{max}} < T - \text{skew}$

$G_{\text{max}} < T + C$

$L + G_{\text{max}} < T + C$

variable fixed fixed

variable variable fixed variable
Time Borrowing in Clock Concurrent Opt.

Using CC-Opt, slack can flow across register boundaries
Logic Chains Limit Time Borrowing

clock

Looping Chain

IO Chain
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
The “CRITICAL CHAIN” is the focus of CC-Opt
- Critical chain is the chain with the longest delay/stage

![Diagram of critical chain and traditional critical path]

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

\[
\text{Delay Stage} = \frac{15 + 16 + 11}{3} = 14
\]
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 “ret ime” 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