Engineering Trade-offs in the Implementation of a High Performance ARM® Cortex™-A15 Dual Core Processor

Bernard Ortiz de Montellano
Product Manager
Processor Division

Joe Walston
Staff Applications Consultant

March 26th, 2013
Today’s Session

ARM-Synopsys Project Introduction

Engineering Trade-offs in the Implementation of a High Performance Cortex-A15 Dual Core Processor

Bernard Ortiz de Montellano

Joe Walston
The ARM® Cortex™-A15 MPCore™ Processor

Implementation Optimization for the “big” core in a big.LITTLE SoC
The ARM Cortex™-A15 MPCore Processor

- Highest performance ARMv7 (32b) application processor
  - Multi-issue, out-of-order pipeline
- Best for superphones, tablets, laptops, servers, infrastructure
- Processor cluster includes
  - 1-4 processor cores with NEON and FPU
  - ACP, SCU, L2 and bus interface
- Architectural enhancements
  - Hardware enhanced OS virtualization
  - 1TB of addressable physical memory
- Performance and power scalability
  - Implementation options include smartphone, tablet and server power envelopes
  - System coherency with ACE
  - big.LITTLE processing with Cortex-A7 and CCI-400
- IP available now

Cortex-A15 is the high-performance engine for your highly-connected device
Implementation Targeting for a big.LITTLE System-on-Chip

- **Big cluster**: Cortex-A15 processor
  - Choose aggressive frequency target
  - Power is mitigated ~50% with MP software

- **LITTLE cluster**: Cortex-A7 processor
  - Choose high efficiency target
  - Very small area for quad core!

- **CoreLink™ CCI-400 Cache Coherent Interconnect**
  - Implement to favor performance
  - Do not starve the big cluster

- **GIC-400**
  - Provides transparent virtualized interrupt control
  - Implement to favor performance
Typical big.LITTLE DVFS Single-Core Curves

“big” core provides “TURBO” Power: Much greater IPC, much greater $f_{\text{max}}$

“LITTLE” core provides much of the workload at the base frequency

$f_{\text{max}}$ maximizes performance
Collaboration Expanded

To Deliver Optimized Methodologies For ARM Cortex Processors

ARM and Synopsys Expand Collaboration to Optimize Power and Performance, and Accelerate Design and Verification for ARM Technology-based SoCs

CAMBRIDGE, United Kingdom and MOUNTAIN VIEW, Calif., Aug. 28, 2012

ARM and Synopsys Collaborate to Deliver Optimized Reference Implementations for ARM Processors
Optimized Methodologies for ARM’s Cortex-A15, Cortex-A7 and CCI-400 Solutions Help Designers Achieve Processor Performance and Power Objectives Faster

CAMBRIDGE, UK, and MOUNTAIN VIEW, Calif. Mar. 21, 2013
Collaboration Objectives

Optimal Starting Point For Cortex-A15 Processor Implementation

<table>
<thead>
<tr>
<th>QOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Meet power target while optimizing for best timing within power budget, best area within power and timing budgets</td>
</tr>
<tr>
<td>- Target market requires a power centric implementation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Develop Cortex-A15 quad core flow quickly for stand-alone or big.LITTLE</td>
</tr>
<tr>
<td>- Enable ARM and Synopsys customers timely access</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>- RTL through Route</td>
</tr>
<tr>
<td>- Repeatable, robust, easily modifiable scripts</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Guidelines for joint customers to follow when targeting a different configuration</td>
</tr>
<tr>
<td>- Best practices and pitfalls</td>
</tr>
</tbody>
</table>

Primary Deliverables: Reference Implementations (RI) with real, repeatable results
• Cortex-A15 dual core processor
• TSMC 28HPM process
• ARM POP™ IP: core optimized standard cells and fast cache instances

Reference Implementation for an ARM Cortex-A15 MPCore processor optimized for balanced timing and power
Today’s Session

ARM-Synopsys Project Introduction

Bernard Ortiz de Montellano

Engineering Trade-Offs in the Implementation of a High Performance Cortex-A15 Dual Core Processor

Joe Walston
Engineering Trade-offs

For a Cortex-A15 Dual Core Processor

Setup
- Standard Cell Libraries
- Core Configuration

Flow Development
- Flow Setup
- Implementation Strategy
- Trade-offs

Future Improvements
- Multisource CTS
- Data Flow Analysis

Conclusions
- Results
- Top 10 Best Practices
Engineering Trade-offs
For a Cortex-A15 Dual Core Processor

- Setup
  - Standard Cell Libraries
  - Core Configuration

- Flow Development
  - Flow Setup
  - Implementation Strategy
  - Trade-offs

- Future Improvements
  - Multisource CTS
  - Data Flow Analysis

- Conclusions
  - Results
  - Top 10 Best Practices
Standard Cell Libraries

**ARM Artisan® Logic for TSMC 28HPM Overview**

- **Technology Details**
  - TSMC 28HPM process
  - 10 layer metal (1p10m_5x2y2z)
  - ARM POP™ IP libraries
    - Fast-Cache Instance RAMs
    - 12T high-speed cells

- **PVT Configuration - 4 corners**
  - Setup (OC_WC): SSG / 0.81v / 0c
  - Hold (OC_BC): FF / 1.05v / 125c
  - Power (OC_LEAK): TT / 0.9v / 85c
  - IR (OC_IR): FFG / 1.0v / 125c

- **Three transistor channel lengths available**
  - CS = short (faster, more power)
  - CM = medium (standard)
  - CL = long (slower, less power)
  - Same cell footprint across all channel lengths

**Standard Cell Selection (Multiple Vt / Channel variants)**

<table>
<thead>
<tr>
<th>Vt Class</th>
<th>Channel Variant</th>
<th>Cell Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>ULVT</td>
<td>CS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CM</td>
<td></td>
</tr>
<tr>
<td>LVT</td>
<td>CS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CL</td>
<td></td>
</tr>
<tr>
<td>SVT (RVT)</td>
<td>CS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CL</td>
<td></td>
</tr>
<tr>
<td>HVT</td>
<td>CM</td>
<td></td>
</tr>
<tr>
<td>UHVT</td>
<td>CM</td>
<td></td>
</tr>
</tbody>
</table>

- 8 different optimization classes
- ULVT not used for this project
- **CL channel has monetary cost**
Standard Cell Libraries

Leakage/Timing Trade-off: Family Comparison

• Comparison for Cortex-A15 leakage/timing trade-off
• Plotting product of leakage and delay for X4 BUF
  – Larger values indicate more leakage cost for a given delay
• Conclusions for Cortex-A15 with leakage/timing trade-off:
  – LVT-CS high leakage cost
  – LVT-CL slightly better trade-off than SVT-CS
  – SVT-CL better low-leakage option than HVT

Only available classes/variants plotted
Engineering Trade-offs
For a Cortex-A15 Dual Core Processor

- Setup
  - Standard Cell Libraries
  - Core Configuration
- Flow Development
  - Flow Setup
  - Implementation Strategy
  - Trade-offs
- Future Improvements
  - Multisource CTS
  - Data Flow Analysis
- Conclusions
  - Results
  - Top 10 Best Practices
## Core Configuration

### Cortex-A15 Dual Core Processor

<table>
<thead>
<tr>
<th>Configurable Feature</th>
<th>Selected Value</th>
</tr>
</thead>
<tbody>
<tr>
<td># Cores</td>
<td>2</td>
</tr>
<tr>
<td>L2 cache size</td>
<td>1MB</td>
</tr>
<tr>
<td>L2 tag RAM register slice</td>
<td>0</td>
</tr>
<tr>
<td>L2 data RAM register slice</td>
<td>0</td>
</tr>
<tr>
<td>L2 arbitration register slice</td>
<td>Not Included</td>
</tr>
<tr>
<td>L2 logic idle gated clock</td>
<td>Included</td>
</tr>
<tr>
<td>Regional gated clocks</td>
<td>Included</td>
</tr>
<tr>
<td>ECC/parity support</td>
<td>Include Parity/ECC in L1 and ECC in L2</td>
</tr>
<tr>
<td>NEON</td>
<td>Included</td>
</tr>
<tr>
<td>VFP</td>
<td>Included</td>
</tr>
<tr>
<td>Generic Interrupt Controller</td>
<td>Included</td>
</tr>
<tr>
<td>Shared Peripheral Interrupts</td>
<td>128</td>
</tr>
<tr>
<td>DFT Strategy</td>
<td>Scan compression</td>
</tr>
<tr>
<td>UPF/Power Strategy</td>
<td>Shut-down w/ isolation</td>
</tr>
</tbody>
</table>
Engineering Trade-offs
For a Cortex-A15 Dual Core Processor

Setup
- Standard Cell Libraries
- Core Configuration

Flow Development
- Flow Setup
- Implementation Strategy
- Trade-offs

Future Improvements
- Multisource CTS
- Data Flow Analysis

Conclusions
- Results
- Top 10 Best Practices
Synopsys’ Core Optimization Collateral
Built on Galaxy Tool RM\textsubscript{s}

- Leverages HPC
- Core and technology library specific
- Includes scripts, floorplan, constraints

Reference Methodologies (RM\textsubscript{s})
- Tool- and release-specific scripts
- Core and technology library independent

Hi-Performance Core (HPC) Methodology
- Leverages RMs, tuned for high perf cores
- Core and technology library independent

Reference Implementations (RIs)

Lynx Plug-Ins
- RI scripts instrumented for Lynx environment

More design/technology specific
Reference Implementation Flow Development

• Basic RM-style flow with HPC-related add-ons
• Each flow step has driver script
• HPC adds customizations like:
  – ICG handling
  – Path group weighting
• Chip finishing not part of collaboration framework
Engineering Trade-offs
For a Cortex-A15 Dual Core Processor

Setup
- Standard Cell Libraries
- Core Configuration

Flow Development
- Flow Setup
- Implementation Strategy
- Trade-offs

Future Improvements
- Multisource CTS
- Data Flow Analysis

Conclusions
- Results
- Top 10 Best Practices
Implementation Strategy

• Hierarchical implementation
  – 2 CPUs instantiated as *multiple instantiated modules* (MIMs)
  – Top-level includes Non-CPU

• Block-level
  – *Block abstracts* (BA) created for top-level closure
  – BAs reduce memory footprint and runtime in hierarchical flow

• Top-level
  – Used *Transparent Interface Optimization* (TIO) for top-level timing closure
Engineering Trade-offs

For a Cortex-A15 Dual Core Processor

Setup
- Standard Cell Libraries
- Core Configuration

Flow Development
- Flow Setup
- Implementation Strategy
- Trade-offs

Future Improvements
- Multisource CTS
- Data Flow Analysis

Conclusions
- Results
- Top 10 Best Practices
Engineering Trade-offs

1. Synopsys Physical Guidance (SPG)
2. Placement Bounds
3. Managing Uncertainty
4. Power Managed Flow
5. CTS Customization
6. Crosstalk Mitigation
1. Synopsys Physical Guidance

*Improved Correlation & QoR for Place & Route*

- DC-Graphical Synopsys Physical Guidance (SPG) improves timing correlation between synthesis and placement
- More QoR exploration possible early in the flow
- Placement bounds used during DC placement, removed for ICC
- SPG synthesis TNS needs to be consistent to `place_opt` TNS
Engineering Trade-offs

1. Synopsys Physical Guidance (SPG)
2. Placement Bounds
3. Managing Uncertainty
4. Power Managed Flow
5. CTS Customization
6. Crosstalk Mitigation
2. Placement Bounds

Use During Synthesis

- QoR improves with synthesis placement bounds
  - Bounds are created from data flow and unbounded placement
    - Cortex-A15 CPU has known data flow which bounds should reflect
  - Examine DC placement with bounds
  - Tune/add bounds for better placement QoR
2. Placement Bounds

*Use Data Flow Analysis (DFA) to Qualify Bounds*

- DFA confirms placement guidance and bounds
- Visualize interconnect count and proximity
- No major issues found

**Data Flow Analysis On CPU Block**
2. Placement Bounds

**Bounds Awareness**

- Poor bounds cause problems
  - Increased TNS
  - High local cell utilization
- Review bounds quality to reduce these effects
  - Examine placement after DC
  - Identify bounds w/ high utilization
  - Identify bounds w/ cells at edge
- Resize or move bounds to produce more evenly distributed DC placement
- Always validate bounds quality
2. Placement Bounds

Bounds Refinement

Reference

Experiment 1

Experiment 2
Experiment 2

- Shifted UIX to little left of tag ram
- UDS bound reshaped as per ARM white paper
- UIF bound reshaped to improve density

ARM’s White Paper
Placement using suggested placement bounds
Engineering Trade-offs

1. Synopsys Physical Guidance (SPG)
2. Placement Bounds
3. Managing Uncertainty
4. Power Managed Flow
5. CTS Customization
6. Crosstalk Mitigation
3. Managing Uncertainty

_TNS Consistency Has To Be Managed_

- SPG adds wire delay estimation to synthesis
  - Timing consistency from DC to ICC

- No extra margin needed for synthesis
  - Traditional synthesis adds guard band for wire delay
  - SPG does not need guard band

- Clock uncertainty is reduced during flow development
  - Help flow work when TNS is large (compared to impact of clock tree)
  - Uncertainty in DC and _place_opt_ set to correlate TNS

---

### Cortex-A15 CPU

<table>
<thead>
<tr>
<th>Flow Step</th>
<th>Initial</th>
<th>Flow Dev.</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC SPG</td>
<td>150</td>
<td>0</td>
<td>120</td>
</tr>
<tr>
<td><em>place_opt</em></td>
<td>150</td>
<td>50</td>
<td>120</td>
</tr>
<tr>
<td><em>clock_opt</em></td>
<td>100</td>
<td>0</td>
<td>60</td>
</tr>
<tr>
<td><em>route_opt</em></td>
<td>100</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td><em>focal_opt</em></td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
</tbody>
</table>

Signoff = 50ps
Engineering Trade-offs

1. Synopsys Physical Guidance (SPG)
2. Placement Bounds
3. Managing Uncertainty
4. Power Managed Flow
5. CTS Customization
6. Crosstalk Mitigation
## 4. Power Managed Flow

### Vt & Channel Selection @ Each Implementation Stage

Vt & Channel Selection Through Implementation Flow

<table>
<thead>
<tr>
<th>Vt Class</th>
<th>Channel Variant</th>
<th>Synthesis</th>
<th>Place/CTS</th>
<th>Route/Focal</th>
<th>Top-Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVT</td>
<td>CS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVT</td>
<td>CS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HVT</td>
<td>CM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UHVT</td>
<td>CM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Legend: **Link Lib**  **Target Lib**

[Image of a table and diagram illustrating Vt and channel selection through the implementation flow.]

---

- **Legend:**
  - Link Lib
  - Target Lib

---

*Copyright© 2013 by Synopsys, Inc. All rights reserved.*
4. Power Managed Flow

Vt & Channel Selection During Synthesis

- Started with LVT-CM for flow development
  - Leakage was too high

- Tried SVT-CM in synthesis to keep leakage low
  - Caused high TNS/utilization and FP growth

- Added LVT-CL to synthesis flow
  - Final flow uses SVT-CM and LVT-CL for synthesis

### CPU Synthesis: Vt & Channel Variants Usage

- SVT-CM
- LVT-CL

### Vt Class & Channel Variant Table

<table>
<thead>
<tr>
<th>Vt Class</th>
<th>Channel Variant</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVT</td>
<td>CS</td>
</tr>
<tr>
<td>SVT</td>
<td>CS</td>
</tr>
<tr>
<td>HVT</td>
<td>CM</td>
</tr>
<tr>
<td>UHVT</td>
<td>CM</td>
</tr>
</tbody>
</table>

Legend:
- Link Lib
- Target Lib

—

(C) Synopsys 2013  Copyright © 2013 by Synopsys, Inc. All rights reserved.
4. Power Managed Flow

Vt & Channel Selection During Placement/CTS

- SVT-only placement and CTS
  - Keep leakage power low through the flow

- Watch for area growth early in project

Area Progression through Project

<table>
<thead>
<tr>
<th>Vt Class</th>
<th>Channel Variant</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVT</td>
<td>CM</td>
</tr>
<tr>
<td></td>
<td>CL</td>
</tr>
<tr>
<td>SVT</td>
<td>CS</td>
</tr>
<tr>
<td></td>
<td>CM</td>
</tr>
<tr>
<td></td>
<td>CL</td>
</tr>
<tr>
<td>HVT</td>
<td>CM</td>
</tr>
<tr>
<td>UHVT</td>
<td>CM</td>
</tr>
</tbody>
</table>

Legend: Link Lib | Target Lib
4. Power Managed Flow
Vt & Channel Selection During Route/Focal_opt

- Introduction of crosstalk increases TNS in route_opt
  - Use LVT cells available to reduce crosstalk

- **Focal_opt** uses all Vt & channel
  - Improves both timing and power
  - Net reduction in leakage even with 22% LVT

### CPU Final Result: Vt & Channel Variants Usage

- **LVT**
- **SVT**
- **UHV**

### Table: Route/Focal

<table>
<thead>
<tr>
<th>Vt Class</th>
<th>Channel Variant</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVT</td>
<td>CM</td>
</tr>
<tr>
<td>SVT</td>
<td>CM</td>
</tr>
<tr>
<td>HVT</td>
<td>CM</td>
</tr>
<tr>
<td>UHVT</td>
<td>CM</td>
</tr>
</tbody>
</table>

Legend:
- **CL**
- **CM**
- **CS**

Legend: Link Lib | Target Lib
4. Power Managed Flow

Vt & Channel Selection @ Top-Level

- Top-level is more timing-challenged
  - Crosstalk in L2 cache RAM channels
  - ICG enable timing more critical
  - Higher connectivity in central area
  - Non-CPU sensitive to area growth

- Added LVT-CM to keep TNS and utilization down

### Non-CPU Placement: Vt & Channel Variants Usage

<table>
<thead>
<tr>
<th>Vt Class</th>
<th>Channel Variant</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVT</td>
<td>CM</td>
</tr>
<tr>
<td></td>
<td>CL</td>
</tr>
<tr>
<td>SVT</td>
<td>CM</td>
</tr>
<tr>
<td></td>
<td>CL</td>
</tr>
<tr>
<td>HVT</td>
<td>CM</td>
</tr>
<tr>
<td>UHVT</td>
<td>CM</td>
</tr>
</tbody>
</table>

Legend:
- LVT
- SVT
- UHVT

Legend: Link Lib Target Lib
4. Power Managed Flow

**Place**\textsubscript{opt}: Cell Density & Power

<table>
<thead>
<tr>
<th></th>
<th>SVT-CM + LVT-CL</th>
<th>SVT-CM + LVT-CM/CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Area Utilization</td>
<td>67%</td>
<td>61%</td>
</tr>
<tr>
<td>Cell Leakage Power</td>
<td>86% target</td>
<td>271% target</td>
</tr>
</tbody>
</table>

Use of LVT-CM in Non-CPU synthesis provided the expected utilization.

- **3X increase in leakage**
- **Utilization too high**
Engineering Trade-offs

1. Synopsys Physical Guidance (SPG)
2. Placement Bounds
3. Managing Uncertainty
4. Power Managed Flow
5. CTS Customization
6. Crosstalk Mitigation

Diagram:
- Design Compiler
  - SPG
- IC Compiler
  - place_opt
  - clock_opt
  - route_opt
  - focal_opt
5. CTS Customization

• General
  – LVT-CS cells for clock drivers
  – Back-annotated computed latency for integrated clock gating cells (ICGs) to synthesis

• CPU
  – Specify early clock on 2 architectural ICGs

• Non-CPU
  – Fix architectural ICG location in DC
  – Magnet place RAM ICGs
  – Delay clock to RAM ICGs

• Debug Tip: Validate ICG counts
Engineering Trade-offs

1. Synopsys Physical Guidance (SPG)
2. Placement Bounds
3. Managing Uncertainty
4. Power Managed Flow
5. CTS Customization
6. Crosstalk Mitigation
6. Crosstalk Mitigation

- Crosstalk impact at 28nm is large
  - Non-CPU routing density can be a problem
  - RAM channels were widened to accommodate power routing, switches and clock NDRs

- Clock net rules for crosstalk
  - CPU used 3X spacing rule
  - Non-CPU used shielding + 3X spacing

- Crosstalk Timing Optimization:
  - Open all LVT classes to allow footprint-compatible swapping
Engineering Trade-offs
For a Cortex-A15 Dual Core Processor

Setup
- Standard Cell Libraries
- Core Configuration

Flow Development
- Flow Setup
- Implementation Strategy
- Trade-offs

Future Improvements
- Multisource CTS
- Data Flow Analysis

Conclusions
- Results
- Top 10 Best Practices
# Multisource (MS) CTS

**Better Skew And OCV Robustness Than Conventional CTS**

<table>
<thead>
<tr>
<th>Traditional CTS</th>
<th>Multisource CTS</th>
<th>Clock Mesh</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Diagram of Traditional CTS" /></td>
<td><img src="image2.png" alt="Diagram of Multisource CTS" /></td>
<td><img src="image3.png" alt="Diagram of Clock Mesh" /></td>
</tr>
<tr>
<td>Skew/OCV ✓</td>
<td>Skew/OCV ✓ ✓</td>
<td>Skew/OCV ✓ ✓</td>
</tr>
<tr>
<td>Power ✓ ✓</td>
<td>Power ✓ ✓</td>
<td>Power ✓</td>
</tr>
<tr>
<td>• Automated synthesis of complex clock relationships</td>
<td>• Coarse mesh uses less power</td>
<td>• Fine mesh - more power</td>
</tr>
<tr>
<td></td>
<td>• More flexible topology</td>
<td>• Offers lowest skew, highest OCV tolerance</td>
</tr>
</tbody>
</table>

*Copyright© 2013 by Synopsys, Inc. All rights reserved.*
MS-CTS Implementation - CPU

Overall Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Trad. CTS</th>
<th>MS-CTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sinks</td>
<td>155K</td>
<td>155K</td>
</tr>
<tr>
<td>CTBuffers</td>
<td>4.9K</td>
<td>5K</td>
</tr>
<tr>
<td>BufferArea</td>
<td>11K</td>
<td>12K</td>
</tr>
<tr>
<td>Global Skew</td>
<td>61 ps</td>
<td>32 ps</td>
</tr>
<tr>
<td>Local Skew</td>
<td>52 ps</td>
<td>26 ps</td>
</tr>
<tr>
<td>Latency</td>
<td>863 ps</td>
<td>768 ps</td>
</tr>
</tbody>
</table>

Note
Not used in current implementation due to lack of SPICE models
Engineering Trade-offs
For a Cortex-A15 Dual Core Processor

Setup
- Standard Cell Libraries
- Core Configuration

Flow Development
- Flow Setup
- Implementation Strategy
- Trade-offs

Future Improvements
- Multisource CTS
- Data Flow Analysis

Conclusions
- Results
- Top 10 Best Practices
Data Flow Analyzer (DFA)

- Advanced Flyline Analysis
  - Debug macro placement in CPU
  - Analyze IO Critical Paths in CPU

- Data Flow Analysis
  - Confirm Placement guidance and bounds in Non-CPU
Results

Consistent Power/Timing @ Each Step Vs Spec Target

Leakage Power & Timing Profile

- **Leakage**
- **Timing**
- **FMAX Timing**

**Target**

- **Crosstalk opto increases power**
- **FMAX PVT 1.80 GHz**
- **Focal_opt recovers**

Increasing Leakage Normalized (% target)

CPU Timing (GHz)
Engineering Trade-offs
For a Cortex-A15 Dual Core Processor

Setup
- Standard Cell Libraries
- Core Configuration

Flow Development
- Flow Setup
- Implementation Strategy
- Trade-offs

Future Improvements
- Multisource CTS
- Data Flow Analysis

Conclusions
- Results
- Top 10 Best Practices
Top 10 Best Practices
For A Power & Timing Balanced Implementation

1. Beat leakage target in synthesis
   - Restrict Vt classes and reduce uncertainty to keep power low

2. Review placement bounds for quality

3. Do not over constrain timing in DC or ICC
   - No extra margin needed for DC SPG
   - Keep ICC uncertainty and derating low to keep power down

4. Timing/Power consistency throughout the flow
   - Use SPG flow in DC and ICC

5. Manage Vt class and channel length throughout the flow
   - Introduce leakier classes only to keep timing/area in check
   - Judicious use of LVT (try not to use LVT-CS until focal_opt)
## Top 10 Best Practices For A Power & Timing Balanced Implementation

1. **Watch cell density, because it impacts power/timing**
   - Areas can rise early in project due to constraints/bounds
   - Grow floorplan, use LVT or reduce margin to keep area in check

2. **Manage for power at each flow step**
   - Understand each rise in power through the flow
   - Use SVT in placement and CTS to reduce power

3. **Use -power & enable power-aware optimization**
   - Enabled by HPC flow variable

4. **Use aggressive clock NDR or shielding for crosstalk prevention**

5. **Aggressive power and timing tradeoff is possible!**
ARM + Synopsys Collaboration

- Cortex-A15 dual core processor
- TSMC 28HPM process
- ARM POP™ IP: core optimized standard cells and fast cache instances

High Performance Core (HPC) scripts + Timing/Power Trade-off Expertise

Reference Implementation for an ARM Cortex-A15 Processor
Optimized for low power and performance
Available Through SolvNet To Joint Customers Today!
Reference Implementation

Collateral & Availability (1/2)

• Available for key components of the ARM big.LITTLE system

Reference Implementation for the ARM Cortex-A15 Processor
Your best starting point for optimized implementation!
Reference Implementation
Collateral & Availability (2/2)

• ARM & Synopsys joint customers can download RI scripts & documentation from:

  www.synopsys.com/ARM-Opto

• For other processor cores, contact Synopsys technical support to help you configure and deploy HPC scripts

• For further optimization and customization support, contact Synopsys Professional Services
# High-Perf. Core Implementation

## Sessions of Interest - Tuesday, March 26th

<table>
<thead>
<tr>
<th>Presenters</th>
<th>Time</th>
<th>Session</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synopsys</td>
<td>12:00 PM to 1:30 PM</td>
<td>1. Optimization Exploration of ARM® Cortex™ Processor-Based Designs with the Lynx Design System</td>
</tr>
<tr>
<td>Lunch &amp; Learn</td>
<td>1:30 PM</td>
<td>2. Power-centric Timing Optimization of an ARM® Cortex™-A7 Quad Core Processor</td>
</tr>
<tr>
<td>ARM &amp; Synopsys</td>
<td>1:30 PM to 3:30 PM</td>
<td>3. Engineering Trade-Offs in the Implementation of a High Performance ARM® Cortex™-A15 Dual Core Processor</td>
</tr>
<tr>
<td>Joint Tutorial</td>
<td>4:15 PM to 5:15 PM</td>
<td>4. Achieving Optimum Results on High Performance Processor Cores</td>
</tr>
<tr>
<td>Broadcom</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MediaTek</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Samsung</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STMicroelectronics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer Panel</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Thank You