A Power-Centric Timing Optimization Flow for a Quad-Core ARM Cortex-A7 Processor

Bernard Ortiz de Montellano  
Product Manager  
Processor Division

Dale Lomelino  
Staff Applications Consultant

March 26, 2013
Agenda

ARM-Synopsys Project Introduction

Power-Centric Timing Optimization Flow for a Quad-Core ARM® Cortex®-A7 Processor

Bernard Ortiz de Montellano

Dale Lomelino
ARM-Synopsys Project Introduction

The ARM Cortex-A7 MPCore™ Processor

Introducing big.LITTLE™ Processing

Implementation Optimization for big.LITTLE

ARM-Synopsys Collaboration
Introducing the Cortex-A7 MPCore Processor

The most energy-efficient v7A (32-bit) application processor

- Power efficient microarchitecture
  - In-order 8-stage, partial dual-issue
  - Integrated NEON™ and FPU, L2, improved memory system
- Architecture aligned with Cortex-A15 MPCore
  - Hardware enhanced OS virtualization
  - AMBA® 4 ACE system coherency
  - 1 TB physical memory addressable

big.LITTLE processing with Cortex-A15 MPCore and CCI-400

- Processor cluster includes
  - 1-4 processor cores with integrated L2, SCU and bus interface
- IP available now

Compelling performance at <100mW for big.LITTLE technology and standalone use
big.LITTLE Processing: 2013

- Tightly coupled combination of two ARM CPU clusters:
  - Cortex-A15 and Cortex-A7 processors - functionally identical CPUs
  - Same programmers view, looks the same to OS and applications
- big.LITTLE combines high performance and low power
  - Automatically selects the right processor for the right job
  - Redefines the efficiency/performance trade-off

Current smartphone

big.LITTLE

>2x Performance

“Demanding tasks”

“Always on, always connected tasks”

30% of the Power
(select use cases)
Right Size Core for the Task

**LITTLE**

- Best efficiency for light tasks
- Simple, in-order, 8-stage pipeline
- Recommended implementation target: highest efficiency

**Cortex-A7**

**big**

- Performance for heavy tasks
- Complex, out-of-order, multi-issue pipeline
- Up to 2x performance of today’s high-end smartphones
- Recommended implementation target: high performance

**Cortex-A15**
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
Performance and Energy-Efficiency

Typical big.LITTLE DVFS Single-Core Curves

“big” core provides “TURBO” Power: Much greater IPC, much greater f_max

f_max maximizes performance

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

Power

Performance
Typical big.LITTLE DVFS Single-Core Curves

“big” processor cluster for Turbo performance

High end smartphone power budget

Mid-range smartphone power budget

Separation in power between big and LITTLE enables optimal benefit from big.LITTLE technology

Entry-level smartphone power budget

“LITTLE” processor cluster for highly efficient processing

Power

Performance

"big" processor cluster for Turbo performance

Mid-range smartphone power budget

Entry-level smartphone power budget

Separation in power between big and LITTLE enables optimal benefit from big.LITTLE technology

“LITTLE” processor cluster for highly efficient processing

High end smartphone power budget

Performance

Power
big.LITTLE Measured Results

big.LITTLE Performance and Energy - Web Browsing + mp3 audio

The same performance as Cortex-A15 alone

~50% energy consumed

Optimized Power and Responsiveness
Performance and Energy-Efficiency

big.LITTLE Effective Power Draw

Superphone performance

High end smartphone power budget

Mid-range smartphone power budget

Entry-level smartphone power budget

big.LITTLE Technology delivers superphone performance in an effective mid-range power budget
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

**Excellent Implementation For Cortex-A7 Processor**

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

<table>
<thead>
<tr>
<th>Schedule</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Develop quad-core Cortex-A7 flow quickly for stand-alone or</td>
<td>big.LITTLE</td>
</tr>
<tr>
<td>• Enable ARM and Synopsys customers timely access</td>
<td></td>
</tr>
</tbody>
</table>

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

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

**Primary Deliverables: Reference Implementations (RI)**

with real, repeatable results
ARM + Synopsys Collaboration

- Quad-core Cortex-A7 processor
- TSMC 28HPM process
- ARM POP™ IP: core optimized standard cells and fast cache instances

Synopsys Engineering and Low Power Expertise

Reference Implementation for an ARM Cortex-A7 MPCore processor optimized for excellent efficiency
Recommended Implementations for your big.LITTLE SoC

- Implementation recommendations developed as a result of the ARM and Synopsys collaboration
- Step 1: Download your ARM Deliverables
  - Processor and fabric IP
- Step 2: Download your Synopsys Deliverables
  - Available now via SolvNet for joint Synopsys and ARM customers
  1. “big” cluster: dual-core Cortex-A15 processor
     - Scripts, design information, documentation
  2. “LITTLE” cluster: Cortex-A7 processor
     - Scripts, design information, documentation
  3. Corelink™ CCI-400 Cache Coherent Interconnect
     - Scripts, design information, documentation

Reference Implementations are an excellent starting point for implementing an effective big.LITTLE SoC processor
ARM-Synopsys Project Introduction

Bernard Ortiz de Montellano

Power-Centric Timing Optimization Flow for a Quad-Core Cortex-A7 Processor

Dale Lomelino
Power-centric Timing Opt. Flow
For a Quad-Core Cortex-A7 Processor

Introduction

Galaxy Low Power Technologies

Setup

Libraries
RTL
Floorplan
Power Plan

Low Power Implementation

Implementation Methodology
Synthesis + DFT
Place & Route

Conclusions

Top 10 Best Practices

© Synopsys 2013

Synopsys Proprietary. All Rights Reserved
### Galaxy Low Power Technologies

**Feature Subset Used In Quad-Core Cortex-A7 Implementation**

<table>
<thead>
<tr>
<th>Dynamic Power</th>
<th>Leakage Power</th>
<th>Multivoltage (UPF)</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Advanced clock gating</td>
<td>• Multi-threshold (Multi-Vt) libraries</td>
<td>• Multivoltage power domains</td>
</tr>
<tr>
<td>• CTS w/ low power placement (LPP)</td>
<td>• Limit usage of LVT cells</td>
<td>• Shutdown regions</td>
</tr>
<tr>
<td></td>
<td>• Channel length cell variants</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Multicorner / Multimode (MCMM)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Final-stage leakage recovery (FSLR)</td>
<td></td>
</tr>
</tbody>
</table>
Power-centric Timing Opt. Flow
For a Quad-Core Cortex-A7 Processor

- Introduction
- Setup
- Low Power Implementation
- Conclusions

Galaxy Low Power Technologies

Libraries  RTL  Floorplan  Power Plan

Implementation Methodology  Synthesis + DFT  Place & Route

Top 10 Best Practices
Libs: Vt-classes / Channel-length
For Power/Timing Tradeoff

Source: www.arm.com/images/PIPD_Logic_MC_animation_small.gif
Choosing Library Subset

**ARM Libraries for Quad-Core Cortex-A7 Processor**

- **Technology Details**
  - TSMC28HPM process
  - 10 layer metal (1p10m_5x2y2z)
  - ARM POP™ IP libraries
    - Fast Cache Instance (FCI) RAMs
    - Standard cells
  - 9T high-density libraries
    - exclude "CL" for stand-alone config

- **PVT Configuration - 4 corners**
  - Setup (OC_WC): SSG / 0.81V / 0C
  - Hold (OC_BC): FF / 1.05V / 125C
  - Power (OC_LEAK): TT / 0.90V / 85C
  - IR (OC_IR): FFG / 1.0V / 125C

- **Three transistor channel lengths**
  - CS = "short" (faster, more power)
  - CM = "medium" (standard)
  - CL = "long" (slower, less power)
  - same cell_footprint, swappable

---

<table>
<thead>
<tr>
<th>Vt Class</th>
<th>Channel Length</th>
<th>Cell Family</th>
</tr>
</thead>
<tbody>
<tr>
<td>ULVT</td>
<td>CS</td>
<td>not used</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>not used</td>
</tr>
<tr>
<td>SVT (RVT)</td>
<td>CS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CM</td>
<td>not used</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>

- 6 Vt/channel-length classes used
- ULVT not used due to leakage power
- **CL** has add'l monetary cost, not used
ARM POP™ IP Core-Hardening Acceleration Technology

POP™ IP libraries are used in the quad-core Cortex-A7 MPCore cluster

Core-Hardening Acceleration by ARM
Power-centric Timing Opt. Flow
For a Quad-Core Cortex-A7 Processor

- Introduction
- Setup
- Low Power Implementation
- Conclusions

1. Galaxy Low Power Technologies
2. Libraries
3. RTL
4. Floorplan
5. Power Plan
6. Implementation Methodology
7. Synthesis + DFT
8. Place & Route
9. Top 10 Best Practices

© Synopsys 2013
Quad-Core Cortex-A7 Configuration
Representative Across Many Applications

<table>
<thead>
<tr>
<th>Configurable Feature</th>
<th>Selected Value</th>
</tr>
</thead>
<tbody>
<tr>
<td># Cores</td>
<td>4</td>
</tr>
<tr>
<td>L2 cache size</td>
<td>1MB</td>
</tr>
<tr>
<td>L1 Instruction cache</td>
<td>32KB</td>
</tr>
<tr>
<td>L1 Data cache</td>
<td>32KB</td>
</tr>
<tr>
<td>NEON™</td>
<td>Included</td>
</tr>
<tr>
<td>FPU</td>
<td>Included</td>
</tr>
<tr>
<td>Generic Interrupt Controller (GIC)</td>
<td>Included</td>
</tr>
<tr>
<td>Embedded Trace Macro Cell (ETM)</td>
<td>Included</td>
</tr>
<tr>
<td>Shared Peripheral Interrupts</td>
<td>128</td>
</tr>
</tbody>
</table>

"The Spec": Worked Example Specification for big.LITTLE, Cortex™-A15, Cortex™-A7, Cortex™-A9, and CCI-400
Edit RTL = Configuration
Configure ETM/GIC Modules, Instantiate RAMs

- Design configuration defined in global configuration file

- Behavioral RAM components changed to library RAM macro instantiations.

- Behavioral clock gating cells changed to library clock gating cell instantiations

- RTL changes during project

CortexA7Integration_Config.v

Fast Cache Instance (FCI) RAMs, from POP IP libraries

Used LVT ICGs, for smallest insertion delay

Parameters included: ETM_PRESENT/GIC_PRESENT
Power-centric Timing Opt. Flow
*For a Quad-Core Cortex-A7 Processor*

- **Introduction**
  - Galaxy Low Power Technologies

- **Setup**
  - Libraries
  - RTL
  - **Floorplan**
  - Power Plan

- **Low Power Implementation**
  - Implementation Methodology
  - Synthesis + DFT
  - Place & Route

- **Conclusions**
  - Top 10 Best Practices
Floorplan Refinement

Leveraged ARM’s Experience, Adjust For This Cfg

Early Floorplan

Refined Floorplan

Saved Area - Made Use Of Corners, Reshaped CPU To Match
Power-centric Timing Opt. Flow
For a Quad-Core Cortex-A7 Processor

Introduction

Setup

Low Power Implementation

Conclusions

Galaxy Low Power Technologies

Libraries

RTL

Floorplan

Power Plan

Implementation Methodology

Synthesis + DFT

Place & Route

Top 10 Best Practices
UPF & Power Plan - ARM iRM Based

Use Design Compiler Design Vision GUI To Visualize
UPF Modifications

Changes From ARM's Cortex-A7 iRM

- Started with `supply_net` UPF
- For top-level (NON_CPU)
- Updated to use new UPF constructs supported in 2011.09 and later versions
- Updated port_state(s) and PST setup

- Added commands for I/Os
  - `set_related_supply_net`
- Added supply net connections for CPU domains
- Added logic ports, logic nets for
  - Switch cell controls
  - Isolation cell controls
  - Supports verification in Formality
- Created additional bias.upf file
- Added variable:
  - `set upf_create_implicit_supply_sets false`
Power Switches - ARM iRM Based

Leveraged ARM's Experience To Balance IR-drop And In-rush Current

- "Trickle" switches (blue)
  - Few = limits in-rush current
  - HEADBUFTIE26_X3M_A9TS_CM

- "Hammer" switches (red)
  - Many = low on-resistance (~20:1 hammer:trickle)
  - HEADTIE22_A9TS_CM

- SVT_CM switches for low leakage

- RAMs have built-in switches
Power Analysis w/ PrimeRail

*Used To Debug IR-drop & In-rush Current Trade-offs*

- IR-drop
- CPU VDD
- NON_CPU VDD

- Hammer switches @ 0 ns
- Hammer switches @ 10 ns
- Only trickle switches
- Missing power connections, fixed by aligning pins
Power-centric Timing Opt. Flow
For a Quad-Core Cortex-A7 Processor

Introduction

Galaxy Low Power Technologies

Setup

Libraries
RTL
Floorplan
Power Plan

Low Power Implementation

Implementation Methodology
Synthesis + DFT
Place & Route

Conclusions

Top 10 Best Practices
Quad-Core Cortex-A7 Configuration

A7 = "LITTLE" in big.LITTLE

ARM® Cortex™-A7 MPCore™

- Cortex-A7
  - ARMv7 32b Core
  - 32KB L-Cache
  - 32KB D-Cache
  - NEON SIMD engine
  - Floating Point Unit

- SCU
  - L2 with ECC

- 128-bit AMBA® 4 - Coherent Bus Interface

CCI-400 Cache Coherent Interconnect

GIC-400 Interrupt control

Cortex-A15

Auxiliary Interfaces
Memory Interfaces
Quad-Core Cortex-A7 Priorities

1. **Meet the power target**

2. Optimize for best timing
   (within power budget)

3. Optimize for area
   (within power & timing budgets)

*Goal: Deliver Cortex-A7 "Reference Implementation" flow*

- *scripts for cpu (cortexa7core)*
- *scripts for non_cpu (CORTEXA7INTEGRATION)*

---

**Power is #1 in the "LITTLE" processor**
Synopsys’ Core Optimization Collateral

*Built on Galaxy Tool RMs*

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

Reference Methodologies (RMs)
- 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)
- RI scripts instrumented for Lynx environment

Lynx Plug-Ins
- RI scripts instrumented for Lynx environment

More design/technology specific
Reference Implementation Flow
Captures Best Practices in the Scripts

- Start w/ HPC with `-power` enabled throughout
- Floorplan is HPC input
- DCG with SPG
  - with `-power`
- ICC w/ SPG-based placement
  - with `-power`
- ICC CTS/CTO
- ICC optimized routing
  - with `-power`
- ICC focal_opt timing-closure
  - with `-power`

Reference Implementation is tuned for Quad-Core Cortex-A7 Processor
Power-centric Timing Opt. Flow
For a Quad-Core Cortex-A7 Processor

Introduction

Setup

Low Power Implementation

Conclusions

Galaxy Low Power Technologies

Libraries

RTL

Floorplan

Power Plan

Implementation Methodology

Synthesis + DFT

Place & Route

Top 10 Best Practices
First, Check the Library Limits
Then Adjust The Flow To Meet The Power Target

• Explore the libraries
• Use Synopsys Physical Guidance (SPG) for best DC-G/ICC correlation:
  – `compile Ultra -spg`
• Manage cell density
  `placer_max_cell_density_threshold`
• Reserve 20-30% of power budget for:
  – DC-G/ICC correlation
  – Clock tree insertion
  – Pre/post-route correlation

Find Just Right Libraries To Balance Power/Timing!

High-Performance Goal

Low-Power Goal

Not “too hot,” not “too cold”
Manage Power: `target_library`

`SVT_CM = "Base" Library, Balances Power/Timing`

<table>
<thead>
<tr>
<th><code>target_library</code></th>
<th>UHVT CM</th>
<th>SVT CL</th>
<th>HVT CM</th>
<th>SVT CM</th>
<th>LVT CL</th>
<th>SVT CS</th>
<th>LVT CM</th>
<th>LVT CS</th>
</tr>
</thead>
<tbody>
<tr>
<td>compile Ultra</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>place Opt</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clock_opt_cts</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>clock_opt_psyn</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>clock_opt_rt</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>route</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>route_opt</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>focal_opt</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

* `set_multi_vth_constraint -lvth_percentage 15 -cost area \ -type hard -lvth_groups {stdcell_9t_rvt_CS}`

** Use lvt_CS for minimum insertion delay (clock OCV: 8% setup, 14% hold)
Manage power: clock_uncertainty

**Timing target impacts power**

- Do NOT over-constrain in synthesis
  - impacts power
- Relax hold uncertainty for pre-route fix_hold
  - saves power prior to route
- Use signoff constraints throughout
  - avoids pessimism
- Signoff uncertainty used for both cpu & non_cpu

<table>
<thead>
<tr>
<th>Cortex-A7 CPU &amp; NON_CPU Flow Step</th>
<th>Clock Setup Uncertainty (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>setup</td>
<td>hold</td>
</tr>
<tr>
<td>compile_ultra</td>
<td>50</td>
</tr>
<tr>
<td>place_opt</td>
<td>50</td>
</tr>
<tr>
<td>clock_opt</td>
<td>50</td>
</tr>
<tr>
<td>route_opt</td>
<td>50</td>
</tr>
<tr>
<td>focal_opt</td>
<td>50</td>
</tr>
<tr>
<td>signoff</td>
<td>50</td>
</tr>
</tbody>
</table>
Power-centric Timing Opt. Flow
For a Quad-Core Cortex-A7 Processor

- Introduction
  - Galaxy Low Power Technologies
- Setup
  - Libraries
  - RTL
  - Floorplan
  - Power Plan
- Low Power Implementation
  - Implementation Methodology
  - Synthesis + DFT
  - Place & Route
- Conclusions
  - Top 10 Best Practices

© Synopsys 2013
Synopsys Proprietary. All Rights Reserved
DFT Strategy - Scan Compression

*Mixed Shared/Non-Shared CODEC I/O Architecture*

- **CPU0**
- **CPU1**
- **CPU2**
- **CPU3**
- **NON-CPU**

All internal chains balanced to 256 long (same as Cortex-A15)

Single Top-level CODEC
TetraMAX® ATPG Results
Improved Coverage, Reduced Tester Time

Pattern Count Reduction Compared to Dedicated I/O
- Transition
- Stuck-At

Mixed Shared I/O
- Transition: 55%
- Stuck-At: 52%

Test Time Reduction Compared to Legacy Scan
- Dedicated I/O
- Mixed Shared I/O
- Mixed Production
- Transition: 16X, 36X, 73X
- Stuck-At: 18X, 37X, 43X

Note:
Baseline ATPG patterns: run_atpg -auto, production ATPG patterns: run_atpg -optimize_patterns

Reduces pattern count by >50%
Improves TATR up to 4.5X
DFT-aware Synthesis

Power and Congestion-aware DFT/ATPG

Timing constraints sensitive to DFT architecture

- RTL: Add DFT ports for ease of verification in Formality
- SDC: Constrain CODEC feedthroughs to avoid false timing violations
- MCMM: Test Mode STA scenarios needed as a minimum
  - Slow-speed shift
  - At-speed capture
    - provides timing data for ATPG within TetraMAX

Benefits of Synthesis-Based Test

- Concurrent handling of timing/power/area & physical effects
- Compression logic optimized to remove congestion and chains re-ordered during incremental compile to improve ICC correlation
- UPF power-intent encompasses test signals to ensure DFT insertion ‘correct by construction’

Power-Aware ATPG defines switching activity budgets to test within functional conditions, avoiding false failures on tester due to ground bounce
Power-centric Timing Opt. Flow
For a Quad-Core Cortex-A7 Processor

Introduction
- Galaxy Low Power Technologies

Setup
- Libraries
- RTL
- Floorplan
- Power Plan

Low Power Implementation
- Implementation Methodology
- Synthesis + DFT
- Place & Route

Conclusions
- Top 10 Best Practices
Summary of Power-centric P&R
Micromanage Power (By Limiting Timing Opt.)

**HPC**
- dc.tcl
- init_design_icc.tcl
- place_opt_icc.tcl
- clock_opt_cts_icc.tcl
- clock_opt_psyn_icc.tcl
- clock_opt_route_icc.tcl
- route_icc.tcl
- route_opt_icc.tcl
- chip_finish_icc.tcl
- focal_opt_icc.tcl

**Power-Centric Additions To HPC**

- **Timing-closure:** limit svt-CS when added @place_opt
  - `set_multi_vth_constraint -lvth_percentage`

- **Pre-route:** relax fix_hold to avoid over-optimization
  - `set_clock_uncertainty -hold -0.050`

- **VR/DR correlation:** Merge clock-routing into psyn step, to get more accurate clock latency timing for optimization

- **Generate net_search_pattern with compare_rc** to align VR RC to final DR RC -- apply in next place_opt

- **Size-only** after adding lvt_CM lib for timing-closure:
  - `route_opt -incremental -size_only`
Upgrade To Latest Tool Release For Best Power Results

- Power optimization enhanced in IC Compiler 2013.03
  - Exact same starting init_design_icc.CEL
  - Exact same script, no changes

<table>
<thead>
<tr>
<th></th>
<th>Frequency</th>
<th>TNS</th>
<th>Area</th>
<th>Leakage</th>
</tr>
</thead>
<tbody>
<tr>
<td>cortexa7core</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>2012.06-SP5</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>2013.03-Beta2</td>
<td>1.02</td>
<td>0.6</td>
<td>0.996</td>
<td>0.93</td>
</tr>
</tbody>
</table>

7% CPU Leakage Power Savings In 2013.03
Add skew_opt

*skew_opt helps to/from_hard_macro path_groups*

- Limit skew_opt to a "few" endpoints for best results

```
set paths [get_timing_paths -slack_less_than -0.010 -max_paths 10000]
append_to -unique skew_opt_pins [get_attr $paths startpoint]
append_to -unique skew_opt_pins [get_attr $paths endpoint_clock_pin]
echo Applying skew_opt on [sizeof $skew_opt_pins] pins ...
skew_opt -no_auto_source -resolution 0.005 -pins $skew_opt_pins -output
```

log: Applying skew_opt on **8353 pins** ...

<table>
<thead>
<tr>
<th>cortexa7core</th>
<th>Frequency</th>
<th>TNS</th>
<th>Area</th>
<th>Leakage</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o skew_opt</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>w/ skew_opt</td>
<td>1.02</td>
<td>0.17</td>
<td>0.99</td>
<td><strong>0.98</strong></td>
</tr>
</tbody>
</table>

Better timing, area, AND power!
Manage Pre/Post-Route Correlation

*Use compare_rc to Generate net_search_pattern*

```plaintext
set enable_net_pattern_rc_scaling TRUE
create_net_search_pattern -net_length_upper_limit 2
set_net_search_pattern_delay_estimation_options \
  -max_horizontal_capacitance_scaling_factor 1.12564 \
  -max_vertical_capacitance_scaling_factor 1.12564 \
  -max_horizontal_resistance_scaling_factor 1.00355 \
  -max_vertical_resistance_scaling_factor 1.00355 \
  -via_count_scale 0.801555 \
  -via_resistance 0.00201186\ 
  -pattern 1

...net_length_upper_limit: 5, 10, 20, 50, 100, 200, 500
```

Net patterns avoid over-constrained timing, for low-power
Manage Cell Density for Low Power
Target Consistency Across Flow

• Check cell density as early as DC-G
  – `restore_spg_placement` shows placement at init_design step
  – "-congestion" is default in `compile_ultra`

• Manage cell density, because it also impacts power/area
  – `set_placer_max_cell_density_threshold 0.7`
  – `set_congestion_options -max_util 0.8 -coord $core`

<table>
<thead>
<tr>
<th>Init_design</th>
<th>place_opt</th>
<th>clock_psyn</th>
<th>route_opt</th>
<th>focal_opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>cortexa7core</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Quad-Core Cortex-A7 Processor
Low-Power Results Address Target Market

cortexa7core

cORTEXA7INTEGRATION

- compressor
- cti
- gic
- *ram*
- scu
- etm
CPU Has Consistent Power/Timing

Tracks @ Each Step Vs Spec Target

- Added SVT_CS
- VR/DR correlation
- Met power target, with margin (can work more on frequency)

OpCond: ssg_typical_max_0p81v_0c + cworst.tluplus
### NON_CPU - Intermediate Results

**Debugging "Jumps" in Power/Timing**

- **@ CTS timing degrades**
  - **Cause:** clock skew
  - **Fix:** relax placer_max_cell_density

- **@route timing degrades**
  - **Cause:** VR/DR correlation + crosstalk
  - **Fix:** net_patterns + NDRs & spread_zrt_wires

- **@post-route power jumps**
  - **Opto tries to recover timing**

---

**OpCond:** ssg_typical_max_0p81v_0c + cworst.tluplus
NON_CPU - Power/Timing Improved

Improved Tracking @ Each Step Vs Spec Target

- @ CTS, timing improved (needs more)
- @ psyn merged clock routing for better timing closure convergence
- Meets power target in PrimeRail
- MET

Frequency (MHz)

Leakage (target%) Normalized power

OpCond: ssg_typical_max_0p81v_0c + cworst.tluplus
SVT Balances Power/Timing

POP IP Libraries Usage At ~15%

CPU
- 88.07%
- 9.47%
- 2.42%
- 0.02%

NON_CPU
- 94.07%
- 1.14%
- 0.03%
- 4.72%

CPU
- 86.32%
- 13.67%

NON_CPU
- 84.51%
- 15.47%
Power-centric Timing Opt. Flow
For a Quad-Core Cortex-A7 Processor

Introduction

Galaxy Low Power Technologies

Setup

Libraries
RTL
Floorplan
Power Plan

Low Power Implementation

Implementation Methodology
Synthesis + DFT
Place & Route

Conclusions

Top 10 Best Practices
## Top 10 Best Practices

**For A Power-Centric Implementation**

1. Optimize for power FIRST, starting at synthesis
2. Manage for power at each step, adjust timing target
   - `target_library`, `-lvth_percentage -type hard`
3. Do NOT over-constrain timing
   - `set_clock_uncertainty -hold -0.050`
4. Focus on correlation - no power/timing "jumps"
   - DCG/ICC: use `compile Ultra -spg; place_opt -spg`
   - Pre/post-route (VR/DR) in ICC
     - `compare_rc -net [<net length>]; net_search_pattern`
5. Manage cell density, because it impacts power/area
   - `set placer_max_cell_density_threshold 0.7`
   - `set_congestion_options -max_util 0.85 -coord $core`
   - Use of `-congestion` has power impact, use only if needed
Top 10 Best Practices
For A Power-Centric Implementation

6. Size-only after introducing new Vt libs
   • `route_opt -incremental -size_only`

7. Use -power & enable power-aware (avoid script errors)
   • `set icc_preroute_power_aware_optimization true`
   • `set_route_opt_strategy -power_aware_optimization true`

8. Use same low power target_lib technique in PT/ECO
   • Otherwise, `fix_eco_timing` may degrade power

9. Iterative design-closure process
   • Good timing => good power & good power => good timing

10. Power-centric flow is possible!
ARM + Synopsys Collaboration

High Performance Core (HPC) scripts + Low-Power Experience

**Reference Implementation** for an ARM Cortex-A7 Processor
Optimized for low power and performance
Available Through SolvNet To Joint Customers Today!

- Quad-core Cortex-A7 processor
- TSMC 28HPM process
- ARM POP IP 9T libraries and memories
- Galaxy Design Compiler
- PrimeTime IC Compiler
- StarRC
Reference Implementation
Collateral & Availability (1/2)

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

Reference Implementation for the ARM Cortex-A7 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

Reference Implementation for the ARM Cortex-A7 Processor
Your best starting point for optimized implementation!
# 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 Lunch &amp; Learn</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>ARM &amp; Synopsys Joint Tutorial</td>
<td>1:30 PM to 3:30 PM</td>
<td>2. Power-centric Timing Optimization of an ARM® Cortex™-A7 Quad Core Processor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4. Achieving Optimum Results on High Performance Processor Cores</td>
</tr>
</tbody>
</table>
Thank You
Q&A
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTS:</td>
<td>Clock Tree Synthesis</td>
</tr>
<tr>
<td>DCG:</td>
<td>Synopsys Design Compiler Graphical</td>
</tr>
<tr>
<td>DR:</td>
<td>Detail Route</td>
</tr>
<tr>
<td>DRC:</td>
<td>Design Rule Check</td>
</tr>
<tr>
<td>DVFS/AFS:</td>
<td>Dynamic Voltage and Frequency Scaling</td>
</tr>
<tr>
<td>FCI:</td>
<td>Fast Cache Instance (ARM POP IP RAMs)</td>
</tr>
<tr>
<td>FSLR:</td>
<td>Final Stage Leakage Recovery</td>
</tr>
<tr>
<td>HPC:</td>
<td>Synopsys High Performance Core (scripts)</td>
</tr>
<tr>
<td>HVT:</td>
<td>High Threshold Voltage</td>
</tr>
<tr>
<td>ICC:</td>
<td>Synopsys IC Compiler</td>
</tr>
<tr>
<td>ILM:</td>
<td>Interface Logic Model</td>
</tr>
<tr>
<td>iRM:</td>
<td>ARM's Implementation Reference Methodology</td>
</tr>
<tr>
<td>LCRM:</td>
<td>Synopsys Lynx Compatible Reference Methodology</td>
</tr>
<tr>
<td>LPP:</td>
<td>Low Power Placement</td>
</tr>
<tr>
<td>LVS:</td>
<td>Layout vs Schematic</td>
</tr>
<tr>
<td>LVT:</td>
<td>Low Threshold Voltage</td>
</tr>
<tr>
<td>MCMM:</td>
<td>Multi-Corner Multi-Mode</td>
</tr>
<tr>
<td>MIM:</td>
<td>Multiply Instantiated Module</td>
</tr>
<tr>
<td>OCV:</td>
<td>On-Chip Variation</td>
</tr>
<tr>
<td>POP:</td>
<td>Processor Optimization Pack (ARM POP IP library)</td>
</tr>
<tr>
<td>PT:</td>
<td>Synopsys Primetime</td>
</tr>
<tr>
<td>PTSM:</td>
<td>Primetime SI (timing analysis with xtalk &amp; noise)</td>
</tr>
<tr>
<td>PTPX:</td>
<td>Primetime PX (gate-level power analysis)</td>
</tr>
<tr>
<td>RI:</td>
<td>Reference Implementation</td>
</tr>
<tr>
<td>RM:</td>
<td>Synopsys Reference Methodology</td>
</tr>
<tr>
<td>SAIF:</td>
<td>Switching Activity Interchange Format</td>
</tr>
<tr>
<td>SPG:</td>
<td>Synopsys Physical Guidance (synthesis =&gt; place)</td>
</tr>
<tr>
<td>SVT/RVT:</td>
<td>Standard Threshold Voltage</td>
</tr>
<tr>
<td>TATR:</td>
<td>Test Application Time Reduction</td>
</tr>
<tr>
<td>VCD:</td>
<td>Value Change Dump</td>
</tr>
<tr>
<td>VR:</td>
<td>Virtual Route</td>
</tr>
<tr>
<td>Vt:</td>
<td>Threshold Voltage</td>
</tr>
</tbody>
</table>