Design & Verification of Low Power SoCs

Dr. Gary Delp, LSI
Mr. John Biggs, ARM
Mr. Srikanth Jadcherla, Synopsys
Agenda

- 13:30-13:35  Introduction  
  Dr. Gary Delp, Distinguished Engineer, LSI Corp
- 13:35-13:45  Design Challenges in Automotive, Networking, and Storage  
  Dr. Gary Delp, Distinguished Engineer, LSI Corp  
  with input from: Juergen Karmann, Senior Staff Engineer,  
  Design Methodology, Automotive, Industrial & Multimarket,  
  Infineon Technologies
- 13:45-14:20  Low Power Flow for Design and Verification  
  Dr. Gary Delp, Distinguished Engineer, LSI Corp  
  With input from: Dr. Ed Huijbregts, Vice President,  
  Design Implementation Products, Magma Design Automation
- 14:20-14:50  Logical verification challenges and techniques  
  Srikanth Jadcherla, Group Director, Synopsys
- 14:50-15:20  Requirements and Solutions for Low Power Processor Cores  
  John Biggs, Founder & Consultant Engineer, ARM
- 15:20-15:30  Roundtable and Wrap-up
Power Efficiency: Three Scopes of Activity

3 levels addressed concurrently:

- **System**: System-level issues, e.g., system architecture, software, power supplies, power distribution, data compression, density/real-estate, power monitoring and control

- **SOC Design**: SOC-level issues, e.g., architecture, systems/applications to circuits, design methodologies, design & simulation tools

- **Silicon & Technology**: Device-level issues, e.g., process technologies, libraries, memories, design IP blocks, modeling tools, design flows, packaging

**Motivation**
The Storage Environment: Today...

The Storage Environment: Tomorrow...

Increased Processing
No increase in power budget
Sophisticated Threats Require Complex Responses

- Blended Attack
- Corporate Espionage
- Identity Theft
- Keyboard Loggers
- Malware
- Unwanted Content
- Trojans
- Worms
- Viruses
- Intrusions
- Defacement
- File Deletion
- Theft
- Siege

Increased Processing
No increase in power budget

Design and Verification of Low Power SoCs
LSI Enabling Real-Time High-Bandwidth Services

- Advanced Packet Processing
  - LSI APP/ACP
  - Highly integrated Advanced Communication Processors for IP and TDM Traffic

- Advanced Traffic Management
  - LSI T9000
  - Industry Leading Content Processing Acceleration with Tarari acquisition
  - Steady operation
  - Array scalable
  - Rapid Start
  - Large State memory:
  - Configuration structures
  - Control structures
  - Data memory
  - Data integrity is key
  - Power management
  - Early estimation

- Content Inspection
  - LSI DSP
  - Highest Performance Voice and Video Signal Processors

- Media/Signal Processing

Real Time Quality of Experience, Security and Content Inspection

Design and Verification of Low Power SoCs
Slide 6 (of 118)
Low Power Challenges in Automotive Applications

Thanks to:

Juergen Karmann, Senior Staff Engineer
Design Methodology
Automotive, Industrial & Multimarket Business Group
Infineon Technologies, Munich, Germany
Semi-conductor enabled functions of a typical car

Body & Convenience
- Xenon Light, Seat Position, Climate Control, Dashboard

Safety
- Airbag, ABS Brakes, Adaptive Cruise Control

Chassis
- Active Suspension, Power Steering

Powertrain
- Engine Control, Transmission Control, Battery Management

Climate Control

Airbag

Night Vision, Steering

Dashboard

Hybrid

Transmission, Engine

Blindspot Detection

Car Lock

Light

Battery Management

Adaptive Cruise Control

Cooling

FAN

Steering

Transmission

Central Lock

TPMS

Night Vision

Brake

Door

Mirror

Semi-conductor enabled functions of a typical car
The Future has arrived ...

- **Powertrain**
  - Engine Control
  - Transmission Control
  - Battery Management

- **Body & Convenience**
  - Xenon Light, Window Lift
  - Climate Control, Dashboard

- **Safety & Chassis**
  - Airbag, ABS Brakes
  - Adaptive Cruise Control
  - Power Steering

- More performance
- Higher ambient temperature
- But low power budgets

- More features at higher complexity
- Limited space in the car
- CMOS and Smart Power Technology on a single die

- Smaller technology nodes
- High variability
- But high reliability

... with complex low power systems in automotive applications
Integration of Low and High Power

Current Product (MCM):
Next Step:
and the Challenges:

1. New coupling effects (parasitic) ⇔ full chip power aware AMS simulation
2. Diverging Current Ranges (μA ⇔ some A)
3. Shared power supplies among low power CMOS and ‘Smart Power’ domains
4. Digital / Analogue / High Voltage Co-Design
5. . . . . .

to be addressed by:

- Formal Power Intent Specification
- Interoperability of EDA Tools
Power-aware Design Goals

• Accelerate Design with more function
• Provide “Just the right” voltage for the task
  • Including 0, slow-low power, and High performance
• Provide Formal specification of Power Intent
• Early and Accurate Power Prediction
• Verify and Validate the Design and Implementation
• Use power efficiently
• Maintain very responsive designs
• Train designers – Use a simple abstraction
• Support tool / vendor portability
• World peace and prosperity
Agenda

• 13:30-13:35  Introduction
With Thanks to: Yatin Trivedi, Director,
   All things important, Synopsys

• 13:35-13:45  Design Challenges in Automotive, Networking, and Storage
Dr. Gary Delp, Distinguished Engineer, LSI Corp
   with input from: Juergen Karmann, Senior Staff Engineer,
   Design Methodology, Automotive, Industrial & Multimarket,
   Infineon Technologies

• 13:45-14:20  Low Power Flow for Design and Verification
Dr. Gary Delp, Distinguished Engineer, LSI Corp
   With input from: Dr. Ed Huijbregts, Vice President,
   Design Implementation Products, Magma Design Automation

• 14:20-14:50  Logical verification challenges and techniques
Srikanth Jadcherla, Group Director, Synopsys

• 14:50-15:20  Requirements and Solutions for Low Power Processor Cores
John Biggs, Founder & Consultant Engineer, ARM

• 15:20-15:30  Roundtable and Wrap-up
Power Efficiency: Three Scopes of Activity

3 levels addressed concurrently:
- **System**: System-level issues, e.g., system architecture, software, power supplies, power distribution, data compression, density/real-estate, power monitoring and control

- **SOC Design**: SOC-level issues, e.g., architecture, systems/applications to circuits, design methodologies, design & simulation tools

- **Silicon & Technology**: Device-level issues, e.g., process technologies, libraries, memories, design IP blocks, modeling tools, design flows, packaging
A three dimensional view of design closure

Illustration thanks to: Bob Carver
Power Profile Optimization

Dynamic Power Profile

Static Power Profile

Optimized Dynamic Power Profile

Optimized Static Power Profile

System Workload
Opportunity for Power Utilization Improvements

Power Optimization Potential:
- Architectural
- Synthesis
- Gate
- Layout

Percentage breakdown:
- 80%
- 20%
- 0%
The new Abstraction

- For decades designers have worked with the digital abstraction, signals are either logical true or logical false.
- As with all good abstractions, this one had great utility
  - it allowed optimizations in analysis,
  - separated two areas of difficult analysis,
  - making the design task achievable.
- This simple abstraction breaks as parts of digital circuits will be turned off relative to other parts
- The good news is that there is a simple way to express the relationships, boundaries, activities, and side effects of many power domains without having to give up most of the simplifications that the digital abstraction allow us.
- Current tools allow us to manipulate:
  - logical constraints and directives – The circuit will do what we want
  - Physical shape reuse and analysis – We can build the design
  - Static timing analysis – the design will run as fast as we want
- **Power Intent Goals**
  - Scalable transportable methodology for describing and reasoning from:
    - Power Domains
    - Power Supplies / Switches and Supply Sets
    - Acceptable and forbidden Power Modes or States
    - Isolation, Level shifting, retention
- Support Reuse and transport of design equity
  - Identification and Constraints – The Platinum Source
  - Refinement and Configuration – The Golden Source
  - Implementation and analysis – The Silicon Source
Power Management: Source & Flow

- The traditional Synthesis flow
  - Is augmented with Power

- Power source files are part of the design source.
  - Combined with the RTL, the power files are used to describe the intent of the designer.
  - This collection of source files is the input to several tools, e.g., simulation, synthesis, STA, test, formal verification, power consistency checking.

- Multiple source files may be prepared specifically to enable reuse.
- The details of the “What” and the “How” are often produced by different parties.
  - Design refinement
  - Equity preservation
Power management **Structures**: The Data Objects

- **Power Domain**
  - The collection of design objects that share common power attributes

- **Power States**
  - Controlled by *Switches*
  - Memories may require *Retention*
  - States may require sequencing info
  - States will affect simulation

- **Relations & Connections between Domains**
  - *Level shifters*
  - *Isolation* logic
  - “Gas Stations” alternate supply

- **Identify elements**
- **Manage**
- **Implement**
- **Analyze**
- **Reuse**
The concept of corruption – supply Off (or Partially On)

- Supply Nets
  - Net_State
    - Full_on
    - Partial_on
    - Off
  - Voltage
    - May be specified for Analysis and Time based corruption

Partial On or off

Partially On

Full on

Partial On or off
Protection from corruption Electrical – Level Shifting

- **Supply Net**
  - **Net_State**
    - Full_on
    - Partial_on
    - Off
  - **Voltage**
    - May be specified for
    - Analysis and Time based corruption

- Full on, $V_1$
- Full on, $V_2$
- Full on, ground
Supply sets –

- Because a single supply net has no meaning in isolation to the power being supplied to any design element, it would be helpful to have the object type “supply set” which does have meaning. For a domain, the following supply set handles are used: primary, default_retention, and default_isolation.

- Supply sets collect supply nets together so that they can be treated as a unified (and progressively defined) bundle.

- This is done for efficiency, clarity, simulation, and brevity.

- Pg pins may be associated with supply set functions to support automatic connection.

- Completion of supply set specification determines the “equity” of the verification runs.
UPF 2.0 Power States

• Objects which can be attributed with power states:
  • Supply sets
    • Defined in terms of the state of the supply nets of the set
  • Power domains
    • Defined in terms of the state of supply sets, supply nets and the power state(s) of other domain(s)
    • What is included in definition is relevant to the state of the domain and reducible to the state of supply nets
    • This provides the ability to hierarchically specify power state relationships

• Can also specify state transitions
Defining a Power State

- **Same command for both supply sets and domains**
  - `add_power_state object_name`
  - `-state state_name`
  - `-supply_expr {boolean_expr}`
  - `-logic_expr {boolean_expr}`
  - `[-simstate simstate]`
  - `-legal | -illegal`
  - `-update`

- **Can be refined (-update) over time as design evolves**
  - `-supply_expr` is the golden specification of the power state – used by synthesis and LEC
  - `-logic_expr` initial, approximation of the power state definition (in the absence of a `-supply_expr`)
    - `-logic_expr` becomes an assertion check when `-supply_expr` is specified
    - `-supply_expr` and `-logic_expr` state definitions can be refined
      - `supply_expr' = old_supply_expr && new_supply_subexpr`

- **Legality**
  - The default for a user-defined power state is legal
    - Specify `-illegal` to override default; `-legal` to be explicit

  - By default, undefined power states are illegal
    - Override default legality of undefined power states for an object:
      - `add_power_state my_power_domain -legal`
## Simulation States (simstates)

Provide Semantic Behavior

<table>
<thead>
<tr>
<th>Simstate</th>
<th>Combinatorial Logic</th>
<th>Sequential Logic</th>
<th>Corruption Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORMAL</td>
<td>Fully functional</td>
<td>Fully functional</td>
<td>None</td>
</tr>
<tr>
<td>CORRUPT</td>
<td>Non-functional</td>
<td>Non-functional</td>
<td>Wires driven by logic and regs powered by the supply corrupted immediately on entering state</td>
</tr>
<tr>
<td>CORRUPT_ON_ACTIVITY</td>
<td>Non-functional</td>
<td>Non-functional</td>
<td>Wires driven by logic and regs powered by the supply corrupted when any input to the logic is active</td>
</tr>
<tr>
<td>CORRUPT_SEQ_ON_CHANGE</td>
<td>Fully functional</td>
<td>Non-functional</td>
<td>Regs powered by the supply corrupted when the value of the register is changed</td>
</tr>
<tr>
<td>CORRUPT_SEQ_ON_ACTIVITY</td>
<td>Fully functional</td>
<td>Non-functional</td>
<td>Regs powered by the supply corrupted when any input to the reg is active</td>
</tr>
<tr>
<td>NOT_NORMAL</td>
<td>Deferred</td>
<td>Deferred</td>
<td>By default, same as CORRUPT. Tool may provide an override</td>
</tr>
</tbody>
</table>
Simulation Model matching

- **Default model for Level shifting**
  - Any difference in the current values of the supply set will cause corruption
  - Modification: A range may be declared legal
  - Modification: A level shifter may be inserted
    - Dual supply – corruption caused when either supply is not full on

- **Default model for Clamps (isolation)**
  - Level shifter model applies
  - X’s may be eliminated with logic if the X has the same supply
  - Modification: ordering may be useful in the boundary specification

- **Default Model for Retention**
  - Memory is powered by retention supply, save saves current value, restore restores the saved value, the saved value is corrupted when the retention supply is corrupted.
  - Modification: other models may be introduced, they replace the default
  - Note that implementation may have time dependencies, these must be specified so that corruption can be introduced in simulation.
A UPF Example

Thanks to:

Dr. Ed Huijbregts
Vice President, Design Implementation Products
Magma Design Automation
UPF by example

- 45nm TSMC library
- 4 Domains, 2 Hierarchies
  - Top Level – 1.0v constant
  - Secondary level SUB0
    - 0.99v switched constant
  - Secondary level SUB1
    - 0.89v constant
  - Secondary level SUB2
    - 0.89v constant
UPF – domain creation

Logical → Electrical → Physical

UPF Commands

create_power_domain top -include_scope
create_power_domain SUB0 -elements {u0}
create_power_domain SUB1 -elements {u1}
create_power_domain SUB2 -elements {u2}
UPF – supply network creation

UPF Commands

#Supplynet creation for domain top
create_supply_net VDD - domain top
create_supply_net VDD_SUB1 - domain top
create_supply_net VDD_SUB2 - domain top
create_supply_net GND - domain top

#Supplynet creation for domain SUB0
create_supply_net VDD - domain SUB0 -reuse
create_supply_net VDD_SUB0_SUB1 - domain SUB0
create_supply_net GND - domain SUB0 -reuse

#Supplynet creation for domain SUB1
create_supply_net VDD_SUB1 - domain SUB1 -reuse
create_supply_net GND - domain SUB1 -reuse

#Supplynet creation for domain SUB2
create_supply_net VDD_SUB2 - domain SUB2 -reuse
create_supply_net GND - domain SUB2 -reuse
**UPF Commands**

- `create_supply_port VDD -domain top`
- `create_supply_port VDD_SUB1 -domain top`
- `create_supply_port GND -domain top`
- `create_supply_port VDD_SUB2 -domain top`
- `connect supply_net VDD_SUB1 -ports VDD_SUB1`
- `connect supply_net VDD_SUB2 -ports VDD_SUB2`
- `connect supply_net GND -ports GND`

**Set Domain Supply Net**

- `set_domain_supply_net top`
  - `primary_power_net VDD`
  - `primary_ground_net GND`
- `set_domain_supply_net SUB0`
  - `primary_power_net VDD_SUB0_SW`
  - `primary_ground_net GND`
- `set_domain_supply_net SUB1`
  - `primary_power_net VDD_SUB1`
  - `primary_ground_net GND`
- `set_domain_supply_net SUB2`
  - `primary_power_net VDD_SUB2`
  - `primary_ground_net GND`
UPF – levelshifter strategy

Level shifter considerations:
- Pick a power domain or a set of elements
- Select input ports, output ports, or both
- Tolerate a voltage difference threshold
- UP shift or down SHIFT rule
- Location (self, parent, sibling, fanout, auto)
- Insert or not insert
UPF – levelshifter strategy

UPF Commands

set_level_shifter SUB1_to_TOP
-domain SUB1
-applies_to_outputs
-rule low_to_high
-location parent

set_level_shifter TOP_to_SUB1
-domain SUB1
-applies_to_inputs
-rule high_to_low
-location self

set_level_shifter SUB2_to_TOP
-domain SUB2
-applies_to_outputs
-rule low_to_high
-location parent

set_level_shifter TOP_to_SUB2
-domain SUB2
-applies_to_inputs
-rule high_to_low
-location self
UPF – isolation cell strategy

UPF Commands

```
set_isolation ISO_STRAT
-domain SUB0
-isolation_power_net VDD
-isolation_ground_net GND
-clamp_value 0
```

```
set_isolation_control ISO_STRAT
-domain SUB0
-isolation_signal reg_out[25]
-isolation_sense high
-location parent
```
**UPF - retention cell strategy**

UPF Commands

set_retention key_desIn \n  -domain SUB0 \n  -retention_power_net VDD \n  -elements {u0/uk/ret_key_sel u0/ret_des_key_r \n    u0/ret_desIn_r}

set_retention_control key_desIn \n  -domain SUB0 \n  -save_signal {key_b_r_reg[16:27]/pin:Q.high}\n  -restore_signal {key_b_r_reg[16:26]/pin:Q.low}
UPF - switch cell creation

UPF Commands

```plaintext
create_power_switch SUB0_SW
  -domain SUB0
  -input_supply_port {TVDD VDD}
  -output_supply_port {VDD VDD_SUB0_SW}
  -control_port {NSLEEPIN1 SE_ME_on_1}
  -control_port {NSLEEPIN2 SE_ME_on_2}
  -ack_port {NSLEEPOUT1 SE_ME_on_ack_1}
  -ack_port {NSLEEPOUT2 SE_ME_on_ack_2}
  -on_state {SW_on TVDD {NSLEEPIN2 & NSLEEPIN1}}
  -off_state {SW_off {NSLEEPIN2 & !NSLEEPIN1}}
```

Diagram from Andrew
A power state table defines the legal combinations of states for different domains.

The `create_pst` command creates a PST, using a specific order of supply nets during operation of the design.

Each row defines a valid combination of supply states.

Power states enable optimization and verification:
- Infer or verify level shifters and isolation gates.
UPF – port state information

UPF Commands

- add_port_state VDD
  -state {VDD_N 0.99}

- add_port_state VDD_SUB1
  -state {SUB1_H 0.89}
  -state {SUB1_L 0.69}

- add_port_state VDD_SUB2
  -state {SUB2_H 0.89}
  -state {SUB2_L 0.69}

- add_port_state GND
  -state {default 0}
**UPF – power states**

**UPF Commands**

```plaintext
create_pst PM_pst -supplies
{ VDD  u0/VDD_SUB0_SW  VDD_SUB1  VDD_SUB2 }

add_pst_state pst0 -pst PM_pst -state
{ VDD_N  SW_on  SUB1_H  SUB2_H }

add_pst_state pst1 -pst PM_pst -state
{ VDD_N  SW_off  SUB1_L  SUB2_L }
```

<table>
<thead>
<tr>
<th></th>
<th>VDD</th>
<th>VDD_SUB0_SW</th>
<th>VDD_SUB1</th>
<th>VDD_SUB2</th>
</tr>
</thead>
<tbody>
<tr>
<td>pst0</td>
<td>VDD_N</td>
<td>SW_on</td>
<td>SUB1_H</td>
<td>SUB1_H</td>
</tr>
<tr>
<td>ps1</td>
<td>VDD_N</td>
<td>SW_off</td>
<td>SUB1_L</td>
<td>SUB1_L</td>
</tr>
</tbody>
</table>
### Attributes Specifiable in HDL

<table>
<thead>
<tr>
<th>HDL attribute</th>
<th>Value</th>
<th>Equivalent UPF command</th>
</tr>
</thead>
</table>
| UPF Clamp Value | `<“0” | “1” | “Z” | “latch” | “any” | “value”>` | `set_isolation –clamp_value
set_port_attributes –clamp_value` |
| UPF Sink Off Clamp Value | ditto | `set_isolation –sink_off_clamp_value
set_port_attributes –sink_off_clamp_value` |
| UPF Source Off Clamp Value | ditto | `set_isolation –source_off_clamp_value
set_port_attributes –source_off_clamp_value` |
| UPF PG Type | `pg_type_value` | `set_port_attributes –pg_type` |
| UPF Related Power Pin | `port_name` | `set_pin_related_supply –related_power_pin
set_port_attributes –related_power_port` |
| UPF Related Ground Pin | `port_name` | `set_pin_related_supply –related_ground_pin
set_port_attributes –related_ground_port` |
| UPF Related Bias Pin | `port_name` | `set_port_attributes –related_bias_port` |
| UPF Retention | `<“required” | “optional”>` | `set_retention_elements –retention` |
| UPF Simstate Behavior | `<“ENABLE” | “DISABLE”>` | `set_simstate_behavior` |
| UPF Is Leaf Cell | `<“TRUE” | “FALSE”>` | `set_design_attributes –is_leaf_cell` |
Design/Component Hierarchy

- Managing Complexity:
  - Design and Conquer
  - Divide and Conquer
  - Reuse
  - Provide Robust Interfaces
The Interface specification is common to:
- the design of the component and
- the Reuse of the component.

It is the contract made at the boundary between design teams.

Top–down and Bottom–up are really just two views of the Component/Interface–based design flow.
Power Portion of Robust Interface Designing within a context

UPF Commands: mod_if
Create_power_domain mod_PD
   -include_scope
Create_power_domain G
Create_power_domain B
Set_port_attributes
   -ports {G1, G2}
   -supply_set G.primary
Set_port_attributes
   -ports {B1, B2, B3}
   -supply_set B.primary
<Set_port_attributes
   -ports {M1, M2}
   -supply_set mod_pd.primary>
mod_details:
Create_power_domain G -update
   -elements {Green}
Create_power_domain B-update
   -elements {Blue}

Supply sets and set_port_attributes
Design and Verification of Low Power SoCs

Slide 44 (of 118)
Basics about Power Intent Specification

Objects in a power domain to be supplied:
- Power domain
- Isolation
- Level shifter
- Retention
- Other Elements

Requirements for XML specification of power intent:
- Identifier for objects to be supplied by power
- Power supplies
- Explicit supply net names must be avoided
- Relation of states for power supplies
  ➔ Power State Table (PST)
  ➔ With legality: ok, never, unspecified
- Transition between power states
XML Power Intent Component information

Power Domain Enumeration
(internal: power domain element identification)
Supply Set Identification
(with port punching of tools, power ports are optional)
Ports associated with supply sets
(set port related supply)
Ports with isolation constraints

Named Power states and associated simstates

Isolation Control required/provided
Retention Control required/provided
Explicit Level shifting requirements
XML Power Characteristic Component Information

Power Domain Enumeration
Supply Set Identification

Named Power states and associated power usage
Named Activities and associated power usage

Isolation Control
Retention Control
Power aware reference design flow
(power reference flow workgroup of Si2 Low Power Coalition)


Design and Verification of Low Power SoCs
Slide 48 (of 118)
Power Tasks on both sides of the Abstraction

Front end:
- Functional specification
- Checking the specification
  - Corruption recognized
  - Srikanth will provide more detail
- Specify allowed and forbidden power states and transitions
  - Ensure forbidden states and transitions never present
  - Electrical safety
- Inserts corruption based on power
- Specify and validate Isolation

Back End:
- Structural specification
- Correlation to structural
  - No new corruption
- Electrical protection (level shifting) for all possible states (or specified requirements)
  - Implementation is conservative compared to specification
- Logical protection - Isolation
- Structural checks – all supplies that affect the behavior are accounted for in the front end.

Simulate and Implement same the Design
Agenda

• 13:30-13:35  Introduction
  With Thanks to: Yatin Trivedi, Director, All things important, Synopsys

• 13:35-13:45  Design Challenges in Automotive, Networking, and Storage
  Dr. Gary Delp, Distinguished Engineer, LSI Corp
  with input from: Juergen Karmann, Senior Staff Engineer, Design Methodology, Automotive, Industrial & Multimarket, Infineon Technologies

• 13:45-14:20  Low Power Flow for Design and Verification
  Dr. Gary Delp, Distinguished Engineer, LSI Corp
  With input from: Dr. Ed Huijbregts, Vice President, Design Implementation Products, Magma Design Automation

• 14:20-14:50  Logical verification challenges and techniques
  Srikanth Jadcherla, Group Director, Synopsys

• 14:50-15:20  Requirements and Solutions for Low Power Processor Cores
  John Biggs, Founder & Consultant Engineer, ARM

• 15:20-15:30  Roundtable and Wrap-up
Verification of Power Managed Designs

Srikanth Jadcherla
Group Director, R&D
Low Power Verification

Synopsys, Inc.
Range of Voltage-Control Techniques

Multi-Vdd (MV)

MTCMOS power gating (shut down)

Power gating with State Retention

Dynamic or Adaptive Voltage Frequency Scaling (DVS, DVFS, AVS, AVFS)

Variable $V_{TH}$ (Back Bias – P/N)

Low-VDD Standby
Power Management increases verification complexity enormously

Correct implementation of LP specific design elements must happen
Seriously,

WHAT COULD GO WRONG?
Power Management brings new bug types!

- Isolation/Level Shifting Bugs
- Control Sequencing bugs
- Retention scheme/control errors
- Retention selection errors
- Electrical Problems like memory corruption
- Power Sequencing/Voltage Scheduling errors
- Hardware-Software deadlock
- Power Gating collapse/dysfunction
- Power On Reset/bring up problems
- Thermal runaway/ Overheating
- ...

These are not traditional functional bugs!
Verification Engineers Need Training on these
Bug Classification

• **Structural Errors**
  - Missing Isolation, Level Shifters
  - Devices in wrong domains
  - Wrong Rail connections

• **Control Errors**
  - Mistimed Control signals
  - Incorrect control activation sequence
  - Incorrect gating/ungating in off/low power states

• **Architectural Errors**
  - Incorrect partitions, policies
  - Incorrect scheduling of resources
Structural Errors

Structure needs to be checked constantly through out the implementation flow

Checks protection logic against

- Missing cell
- Redundant cell
- Incorrect cell type
- Incorrect power domain
- Incorrect isolation polarity
- Incorrect iso-enable
Incorrect Isolation Sequence

Control Error

---

Intended Behavior
1. Gate the clk
2. Assert iso to 0
3. Assert sleep to 0

Actual Behavior
1. Gate the clk
2. Assert sleep to 0
3. Assert iso to 0

Control signal sequence error is discovered due to X propagation

---

Output HighZ based on sleep signal

X is observed due to incorrect iso timing

Registers initialized to X after power restoration
Voltage Scheduling Error

Control/Architecture Error

Logic Simulator cannot distinguish between voltage values - All treated as 1

Need Level Shifter

Verification must be aware of the waveform nature of voltage
Power Intent = More code to verify!

- How does boolean analysis change?
- How do we make testbenches for RTL + Power Intent?
- How do we measure coverage, write assertions etc. to make verification fruitful?
Unlearn traditional booleans!

VOLTAGE AWARE BOOLEANS
Fundamental Technology Shift: Voltage Aware Boolean Analysis

Traditional Boolean Analysis:
Implicit, homogenous, always on voltage

Voltage-aware Boolean Analysis:
Dynamic, Variable voltages as real numbers
Voltage-Aware Simulation is now necessary!

Traditional Simulators are not voltage aware

Voltage-Aware Simulators are Electrically Accurate
Multi-Fanout in DVS

Island 1

1.0 V

1'b1

Voltage at input is not strong enough

Island 2

1.5 V

X

70% of VDD = 1'b1

Island 3

0.8 V

1

Voltage at input is strong enough

Design and Verification of Low Power SoCs
Slide 64 (of 118)
Accurate Voltage Modeling

**True Voltage Aware Simulation**

- Voltage is a ‘real’ value – not a binary logic
- Accurate power-up and power-down voltage ramps must be modeled for accurate simulations
- Template behavioral models provided with MVSIM

```verilog
module VRM(enable, ready);
    parameter VoltageStep = 0.01;
    parameter Von = 1.0;
    .......
    real vdd;
    .......
    always @(enable)
        begin
            .......

        function real vrm();
            .......

        endmodule // VRM
```

Parameterized source code provided
The rise of Retention

- State loss from Power Shut Off may not be OK
  - Performance hit with cache misses
  - Or Latency impact for ‘Cold Start’
- Traditionally, Low Vdd Stby was used to retain state
- As Vtn, Vtp ->0, Vstby becomes impractical
  - Retention flops: Shadow the main element with high Vt
  - Cut off Vdd, but hold on to Shadow element power
  - Restore from Shadow to main element after powerup
- Many Flavors of Retention exist
  - Languages don’t model them well!

Retention is a huge verification challenge
Retention stretches language semantics

- Retention: A balloon latch is used to retain state when power is turned off
- Wait, we lack semantics for shutdown, how do we deal with this?

```verilog
always @ (posedge clk or negedge reset_n)
if (!reset_n) q <= 0;
else q <= d;
```

Need to simulate and verify this!

Retention is a huge verification challenge
Verifying Retention

Complete power Sequence

- CLK
- CLK-EN
- SAVE
- ISO
- PWR EN
- VDD
- PWR RDY
- RESTORE

- Disable Clock
- Enable Clock
- Save
- Disable Isolation
- Enable Isolation
- Disable Power
- Enable Power
- Ramp Voltage
- Ramp Voltage
- Disable Ready
- Enable Ready
- Restore
Verifying Retention
Corner Cases Can Be Tricky

PWR GATED DMA
SAVE
ACTUAL RESTORE
REG A aa Z X
EXPECTED RESTORE
REG B bb Z X bb

PREMATURE RESTORE SIGNAL
LOW POWER TESTBENCHES
A typical low power SOC
A complex, asynchronous, mixed signal control system
Elements of a Low Power Testbench

- Power State Manager
- Tests
- Generator
- Scoreboard
- Pwr Mgmt
- Register Abstraction
- Master
- DUT
- Rx
- Performance Analyzer
- Reused from All-ON testbench

Design and Verification of Low Power SoCs
Slide 72 (of 118)
RAL C API: To generate s/w tests

Diagram:
- Unmodified
- Firmware C Code
- Cosim API
- RTL DUT
- Low-Power Tests
- VCS
- CPI
- Register Abstraction
- RTL DUT
- Embedded SW
- gcc
- Spec
- .o
- Interrupts or transition requests

Text:
Design and Verification of Low Power SoCs
Slide 73 (of 118)
Coding Guidelines

• 1'b0 and 1'b1 – no single supply1 or supply0
  • Use tie_hi_<name> or tie_lo_<name>

• Initial blocks don’t get retriggered again
  • Be careful with readmem and other initializations in on/off

• Asynch reset doesn’t get activated again
  • Could be design or testbench issue

• X-detection monitors can go crazy
  • Avoid stopping the test on “x” in an on/off block!

• Assertions need to account for off state!

• Don’t use XMR force statements!

• …
COVERAGE AND ASSERTIONS
Redefining Coverage

Power Intent elements, Power States, Transitions and Sequences need a Coverage strategy
Static Verification covers structural errors
Dynamic Coverage elements

- Power Intent entities
- Isolation device controls
- Last known good state
- Retention elements
- Power Switch structures
- Changes in Power Intent vs. desired coverage
  - E.g. Adding retention to block A increases coverage needs
  - E.g. Changing a power state may move an island pair from isolation to level shifting
Source vs. Leaf level assertions

- Absence of one leaf-level assertion will cause failure of design
- Source level assertions cannot be relied upon to ensure correct operation of design

Source Assertion:
When PS_ENABLE = 1, ISO_ENABLE = 0

<table>
<thead>
<tr>
<th>RTL Lines</th>
<th>Voltage Islands</th>
<th>Power States</th>
<th>RTL Assertions</th>
<th>Gate Level Assertions</th>
</tr>
</thead>
<tbody>
<tr>
<td>6099</td>
<td>4</td>
<td>8</td>
<td>685</td>
<td>1891</td>
</tr>
</tbody>
</table>
A METHODOLOGY FOR LOW POWER VERIFICATION
Industry’s First Verification Methodology for Low-Power

Broad participation by LP experts to contribute and review
VMM-LP - Industry’s First Verification Methodology for Low-Power

Standards
- Builds on VMM standard

Education
- Documented, real LP bug examples

Verification
- LP planning, assertions and coverage

Reuse
- Best-practice rules and guidelines

Base Class Library & Applications
- Source code available under open-source license

Book and pdf are out!
VMM-LP Chapters

- Basics of Multi-Voltage Low Power Designs
- Bug types and profiles
- Dedicated chapter on State retention
- Testbench and coding guidelines
- Static Verification
- Test planning and Dynamic Verification
- Base Classes and Applications
- Rules and guidelines
VMM-LP.

- has a dual focus:
  - Problem identification/education
  - Reusable Verification Methodology
- has Design rules and Verification rules
- has Static verification components
- has new classes
- has extensions to current classes
- has little source code in the book: code is provided online
Conclusions

• Low Power design is functionally quite complex
• Verification engineers need to adapt to the new bug profiles
• A rigorous methodology is needed in the flow involving both static and dynamic verification
• VMM-LP extends VMM for low power designs
• VMM-LP introduces new as well as enhanced base classes and applications
Agenda

13:30-13:35  Introduction
With Thanks to: Yatin Trivedi, Director,
All things important, Synopsys

13:35-13:45  Design Challenges in Automotive, Networking, and Storage
Dr. Gary Delp, Distinguished Engineer, LSI Corp
with input from: Juergen Karmann, Senior Staff Engineer,
Design Methodology, Automotive, Industrial & Multimarket,
Infineon Technologies

13:45-14:20  Low Power Flow for Design and Verification
Dr. Gary Delp, Distinguished Engineer, LSI Corp
With input from: Dr. Ed Huijbregts, Vice President,
Design Implementation Products, Magma Design Automation

14:20-14:50  Logical verification challenges and techniques
Srikanth Jadcherla, Group Director, Synopsys

14:50-15:20  Requirements and Solutions for Low Power Processor Cores
John Biggs, Founder & Consultant Engineer, ARM

15:20-15:30  Roundtable and Wrap-up
Every Joule is sacred.
Every Joule is great.
If a Joule is wasted,
We all get quite irate....

Every Joule Is Sacred...
What Consumers Care About

• Users want more features in their mobile devices:
  • MP3, Camera, Video, GPS...

• But also need long battery life
  • Convenient form factor, affordable price

• Battery technology is not evolving fast enough!
  • Need to manage power consumption
Process Migration Alone Is No Longer Enough

The good old days
- Faster
- Lower power

New reality
- Speed increases at the expense of energy consumption

Everything improves significantly
RISC is a good starting point...

- Pure RISC can go too far in reducing the functionality of each instruction
  - More instruction fetches, less efficient cache usage (more external fetches)

- ARM retains some carefully chosen CISC like features
  - Conditional instruction execution
  - LDM/STM - Load/Store multiple registers
  - LDR/STR - Load/Store Register with base plus offset
  - Flexible “second operand” on ALU (Barrel Shifter)

- ARM7TDMI - 16 bit “Thumb” Instruction Set
  - High code density for system size/cost/power savings
  - Greater than 20% code size savings over 32-bit ARM code
  - Memory footprint comparable to 8/16-bit microcontrollers

The cheapest, fastest, most reliable components of a computer system are those that are not there!

--Gordon Bell
Low Power Is A System Problem

- Operating System with Software Policies
  - Managing the entry and exit to and from system sleep states

- System Level Control IP
  - Architectural design partitioning, hardware control
  - Sleep transition protocol management

- Library Level Support
  - Comprehensive low power components (ISO, Switch, Retention)

- EDA Software
  - Comprehensive automation yielding ultra low power design with optimal QoR

- Power Supply Management
  - External power supply control, power supply tolerances, etc.

- Process Technology
  - Trade-off between a high performance and low leakage process
Power Dissipation

\[ E = \int_{0}^{t} (V_{DD}I_{\text{leak}} + CV_{DD}^{2}f_{c}) \, dt \]

\[ \int_{0}^{t} V_{DD}I_{\text{leak}} \, dt \]

\[ \int_{0}^{t} CV_{DD}^{2}f_{c} \, dt \]

Minimize \( I_{\text{leak}} \) by:
- Reducing operating voltage
- Fewer leaking transistors
- Reduce transistor leakage

Minimize \( I_{\text{switch}} \) by:
- Reducing operating voltage
- Less switching cap
- Less switching activity
Dynamic Power Optimization

• Dynamic Frequency Scaling (DFS)
  • Reduce operating frequency if possible
  • Reduces average power (but not task energy)
  • Eliminates NOPs

• Dynamic Voltage & Frequency Scaling (DVFS)
  • Requires DFS
  • Reduces voltage if frequency is reduced
  • Reduces task energy
  • Based on characterized frequency – voltage pairs (lookup table)

• Adaptive Voltage Scaling (AVS)
  • Closed loop optimization of VDD at run-time
  • Can save energy even at fixed frequency
ARM IEM Principles

- Batteries have finite amounts of energy stored in them
- Running fast and then idling wastes energy

Only need to run just fast enough to meet the application deadlines
ARM IEM Technology

Hardware and software solution for energy management
Dynamic control of voltage and frequency scaling.

- IEM software connects to OS kernel and collects data.
- Multiple policies categorize the software workload.
- Prediction of future performance requirement is made.
- Suitable operating point (Voltage and Frequency) is set.

IEM software connects to OS kernel and collects data.
Multiple policies categorize the software workload.
Prediction of future performance requirement is made.
Suitable operating point (Voltage and Frequency) is set.
ARM IEM System Implementation
Trends In Power Dissipation

- Static power dissipation can no longer be ignored
  - It became significant at 90nm and dominant at 65nm
- Leakage currents are rising fast
  - Must be controlled by circuit design and optimization tools
Leakage Currents

- Transistors are not perfect switches – they always “leak”
  - Especially the high performance (low Vt) ones

\[ I_{\text{SUB}}: \text{Sub-threshold Leakage} \]
\[ I_{\text{GATE}}: \text{Gate Leakage} \]
\[ I_{\text{GIDL}}: \text{Gate Induce Drain Leakage} \]
\[ I_{\text{REV}}: \text{Reverse Bias Junction Leakage} \]

Total Leakage = \( I_{\text{SUB}} + I_{\text{GATE}} + I_{\text{GIDL}} + I_{\text{REV}} \)

- Currently sub-threshold leakage dominates
  - Multi-threshold and Power Gating most effective

- However gate leakage is becoming significant
  - Can be mitigated by high K dielectric material
Leakage Mitigation Techniques

- Lower Operating Voltage
- Cell sizing
- Dual $V_t$
- Power Gating

- Non minimum size gate lengths
- VT-CMOS
- Stack Effect

Slide 99 (of 118)
Coarse Grain Power Gating

- **Power switches shared by many cells**
  - Reduced area and performance impact

- **Ring Based Switch Topology**
  - Rings of switches encapsulate the power down block
  - Good for legacy IP, non-intrusive for optimized blocks
  - Preferable if no state retention used – no always-on mesh
  - Can add significant area cost to existing IP

- **Distributed Switch Topology**
  - Distributed switch cells in power down block – smaller
  - Can be implemented as a sparse array, in rows or columns
  - Seems to provide better QoR and control of IR drop
  - Better trickle charge management during power-up
Managing Voltage Drop

- **A reduced supply voltage impacts performance**
  - Adding switches to a power network will necessarily induce a voltage drop in that network in addition to the voltage drop due to power distribution across the mesh
  - Must minimize this additional switch induced voltage drop through considered architecture of the switch topology and switch size
  - Voltage drop across switch also causes standard cells to operate slightly reverse biased if well is tied to VDD

- **Distributed switching can help limit voltage drop**
  - Provides a finer control resolution on the switch size and placement
  - Increase the switch density and size in power hot spots
  - Can reuse the always-on power networks (for switch control) to power retention registers and additional always-on logic (save & restore signals)
State Retention Considerations

- Three possible approaches to state retention
  - Software based state save and restore (OS driven)
  - Hardware based state save and restore via scan structures (via AMBA)
  - Hardware based local state retention with retention registers

- Choice of retention scheme dependant on a number of factors:
  - Area overhead of retention registers and size of state space to be maintained
  - Performance impact of retention registers
  - Energy cost for save and restore when saving state externally
  - Real time cost for save and restore when saving state externally

<table>
<thead>
<tr>
<th>Approach</th>
<th>Standby Leakage</th>
<th>Area Overhead</th>
<th>S/R Energy Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Gating Only</td>
<td>Power Switches &amp; AO Logic</td>
<td>Power Switches &amp; AO Logic</td>
<td>Complete Reset Required</td>
</tr>
<tr>
<td>Power Gating with Software Based Retention</td>
<td>Power Switches &amp; AO Logic</td>
<td>Power Switches &amp; AO Logic</td>
<td>State Restore via Software</td>
</tr>
<tr>
<td>Power Gating with Scan Based Hibernation</td>
<td>Power Switches &amp; AO Logic</td>
<td>Power Switches &amp; AO Logic</td>
<td>State Restore via Scan Shift From Memory</td>
</tr>
<tr>
<td>Power Gating with Local State Retention</td>
<td>Power Switches, AO Logic, Retention Registers</td>
<td>Power Switches, AO Logic, Retention Registers</td>
<td>Minimal as state maintained locally</td>
</tr>
</tbody>
</table>
Managing In-Rush Current

- To avoid excessive power rail droop/ground bounce which may:
  - Corrupt locally retained state
  - Impact neighbouring blocks
- Turn on a small number of “weak” switches first
- Turn on the “main” switches as VDD_SW ~= VDD
SALT: Synopsys ARM Leakage Technology Demonstrator

- Joint development with Synopsys to address EDA implementation
- ARM926EJS based SoC in TSMC90G
  - Leakage mitigation technology demonstrator
  - Based on R&D Library, Synopsys MV tools
  - Performance scaling:
    - 33/67/100/133% of 300MHz
  - CPU supports 3 “depths” of leakage management
    - State Retention Power Gating
    - Scan-Hibernation
    - Shutdown
  - Support for back/forward bias VTCMOS
  - Retention integrity diagnostics
SALT Silicon Measured Results

- Graph shows leakage reduction due to power gating over temperature (compared to mission mode with clocks stopped)
- Excellent thermal leakage profile for hand held mobile devices!

Over 96% savings across normal mobile operating range

High Vt devices leak more at high temperatures

Gate leakage starts to dominate at low temperatures
In Summary:

- Manage power in all modes in which a design operates
  - Dynamic power during device operation including active leakage
  - Static power dissipation during standby

- Maintain device performance while minimizing power consumption
  - Meet most aggressive performance targets while minimizing power
  - Aggressive power optimization when running at reduced performance levels
  - Minimize impact to performance by employing aggressive low power techniques
  - Employ a number of low power techniques in a single processor implementation

- Aggressive techniques for power management
  - Dynamic power minimized through OS directed performance scaling
  - Dynamic power minimized through use of Multi-Vt and Multi-L libraries
  - Standby power minimized through power gating with state retention
  - Additional standby power savings through use of threshold scaling (bias)
Extent of Soft IP Provider’s Low Power Intent

A Soft IP provider need only declare four things:

1. The "atomic" power domains in the design
   • these can be merged but not split during implementation

2. The state that needs to be retained during shutdown
   • with out prescribing how retention is controlled

3. The signals that need isolating high/low
   • with out prescribing how isolation is controlled

4. The legal power states and sequencing between them
   • with out prescribing absolute voltages
Successive Refinement of Low Power Intent

1. **IP Creation**
   - RTL
   - Constraint UPF
   - Soft IP

2. **IP Configuration**
   - RTL
   - Constraint UPF
   - Configuration UPF
   - Golden Source

3. **IP Implementation**
   - RTL
   - Constr’nt UPF
   - Config’n UPF
   - Impl’tion UPF
   - Synthesis
   - Netlist
   - P&R
   - Netlist

**IP Provider:**
- Creates IP source
- Creates low power implementation constraints

**IP Licensee/User:**
- Configures IP for context
- Validates configuration
- Freezes “Golden Source”
- Implements configuration
- Verifies implementation against “Golden Source”

Simulation, Logical Equivalence Checking, Kelipsis

Design and Verification of Low Power SoCs
Slide 109 (of 118)
Successive Refinement Example

1. UPF Constraints
   • IP provider needs to "identify" what is to be isolated with out prescribing how:
     
     ```
     set_isolation my_iso -domain my_pd \
     -clamp_value 0
     ```

2. UPF Configuration
   • System Level simulation guy needs to configure the logical power controls with out having to specify the power supplies:
     
     ```
     set_isolation -update my_iso -domain my_pd \
     -isolation_signal CLAMP -isolation_sense high
     ```

3. UPF Implementation
   • Finally the details of power supplies are then added during implementation
     
     ```
     set_isolation -update my_iso -domain my_pd \
     -isolation_power_net VDDG -location parent
     ```

Or specify it all at the same time:

```
set_isolation my_iso -domain my_pd \
-clamp_value 0 \
-isolation_signal CLAMP -isolation_sense high \
-isolation_power_net VDDG -location parent
```
However UPF-1.0 has a limitation…

- Unfortunately UPF-1.0 can’t describe power control nets/pins

- So for a “bottom up” flow power control pins like “SLEEP” have to be in the RTL

- However, true Soft IP should be technology independent

- The sense/width of “SLEEP” will depend the implementation
  - Active high for PMOS active low for NMOS switches
  - May be a “thermometer encoded” bus to manage inrush

- Good news: this has been fixed in IEEE1801 UPF-2.0
  - See “create_logic_*” commands
Non-contiguous power domains

- Multi-element power domains can lead to unexpected “intra-domain” isolation
  
  ```
  create_power_domain RED -elements A B
  ```

- These can often be avoided with a different approach
  
  ```
  create_power_domain RED include_scope
  create_power_domain BLUE -elements C
  ```

- Better to align power domains with logic hierarchy if at all possible
  
  ```
  create_power_domain RED -elements RED
  create_power_domain BLUE -elements BLUE
  ```

- Again, fixed in UPF-2.0
  
  ```
  set_isolation -diff_supply_only TRUE
  ```
• Can use UPF inferred clamps to stop clocks and assert reset
  • Needs care to avoid timing issues
• Better to use handshaking controlled by a simple state machine
  • Facilitates design reuse and technology portability
Sequencing Clocks, Resets and Power

**Power down sequence:**
1. Stop the clocks
2. Apply isolation
3. Optionally save state
4. Assert reset
5. Remove power

**Power up sequence:**
1. Apply power
2. Remove reset
3. Optionally restore state
4. Remove isolation
5. Start the clocks
A Few “Best Practices”...

- Avoid non-contiguous power domains
  - Can lead to unwanted isolation cells
  - If in doubt, align power domains with logic hierarchy

- Avoid using clock gating on both edges of the clock
  - Need for specialised ICGs may limit choice of implementation libraries

- Avoid partial retention within a power domain
  - Unless the IP has been explicitly designed to support it.
  - User defined partial state retention will require complete re-verification
  - No support for “set_retention –no retention” in UPF-1.0

- Ensure that every power domain’s clocks and resets can be controlled externally
In Conclusion:

- Power dissipation is the #1 limiter of design performance
  - Can be mitigated with advanced circuit design and optimization tools

- Power management is a system problem
  - Power management strategy must be carefully considered from architecture to silicon

- Need to ease adoption of advanced low power techniques
  - Develop low power IP, tools & techniques (ARM’s IEM & PMK)

- UPF enables portability of low power intent
  - Provides ability to compare and contrast a variety of implementation strategies
  - Portable across EDA tools and supported by commercial low power libraries
  - Low power overlay to existing processor IP from ARM

Transistors (and silicon) are free. Power is the only real limiter. Optimizing for frequency and/or area may achieve neither.

Pat Gelsinger, Intel (DAC2004 Keynote)
Did I mention the book?

• 49% of surveyed customers* identified power management as major concern
  • “How do I describe my power requirements?”
  • “Which advanced techniques are worth the effort?”
  • “I know the concepts, but I don’t know how to implement them”

• ARM & Synopsys partnering to develop solutions to address these concerns
  • Many years of joint investment in advanced low power programs
  • Driving technology into products: processors, libraries & EDA tools
  • Capturing best practice in the Low Power Methodology Manual

http://www.lpmm-book.org

• Free PDF download:
  • http://www.arm.com/lpmm
  • http://www.synopsys.com/lpmm

* Source: 2007, Synopsys LPMM customer survey
Thank You!

Gary.Delp@LSI.com
John.Biggs@ARM.com
Srikanth.Jadcherla@synopsys.com