Advances in Low-Power Verification

Janick Bergeron
Synopsys Fellow
The Era Of The Consumer

*Households are the Semiconductor Industry’s #1 Customer*

A very different profile of energy usage than before

Source: J.-H. Huang, SIA 2004
What do consumers want?

High Performance, Multi-media experience

AND

Lowest Active Power,

Zero Idle Power

This is for the System, not just for each individual IC
Most Systems are amenable to Energy Efficient Design

Lowest Active Power Principle:
Apply the Lowest Possible VDD to each functional block at each instant of time

• Systems/ICs don’t have the same performance needs all the time
  – Operating at worst case design corner all the time is wastage

• Systems/ICs don’t need all the functions to be running all the time
  – Operating all the devices all the time is wasteful

Zero Idle Power Principle:
Turn on a block only when needed
Turn it off once it is not needed
What can IC designers do?

• Lowest Active Power
  – Clock Gating, Device Sizing (old)
  – Multi-VDD (Spatial Voltage Control)
  – DVFS Temporal Voltage Control)

• Zero Idle Power
  – Multi-Vt (old)
  – Power Gating/Retention
  – Low VDD Standby, Back Bias

We still have to deal with the System HW and SW need to work together
Enter, the PMU

Power Management Units will aggressively Control Voltage
We are no longer testing ASICs on the tester → SOCs need to work in System Design Verification Tape-Out System/Board level builds Firmware testing Conventional Design-Debug Flow

Design Verification of Firmware especially Power Management Tape-Out SOC development flow
Power Management increases verification complexity enormously

Correct implementation of LP specific design elements must happen
Mode State Space

Phone Mode

Stand-By

PDA Mode
Power State Space

- Standby Mode
- Phone Mode
- PDA Mode
Power State Space
Power State Transitions

- High Perf.
- Normal
- Save
- VDD=3.3V
- VDD=0.7V
- Stand-By

- High Perf.
- Normal
- Save
- VDD=3.3V
- VDD=0.7V
- Stand-By

- VDD=3.3V
- VDD=1.5V

- VDD=1.5V
- VDD=0.7V
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!
Multi-VDD stretches boolean logic

Traditional Boolean Analysis

Voltage-aware Boolean Analysis

Impacts Simulation, Static Analysis, Equivalence models

Unlearn Boolean Analysis you were taught in college!
Voltage Aware Booleans?

Traditional simulators are **not** voltage aware

Voltage-aware simulators are electrically accurate
RTL lacks many power semantics

```verilog
module sio(din, dv, txd, txe, clk, rstn);
    input [7:0] din;
    input dv;
    output txd, txe;
    input clk, rstn;

    reg [2:0] idx;
    reg txd, txe;
    always @ (posedge clk or negedge rstn)
    begin
        if (!rstn) begin
            idx <= 0;
            txe <= 1'b0;
        end
        else if (dv) begin
            txe <= 1'b1;
            txd <= din[idx];
            idx <= idx + 1;
        end
        else idx <= 0;
    end
endmodule
```

No VDD
No V_in
No V_out
No Save/Restore control signals
Clock gating!
No shutdown
No retention
We have a fundamental semantics problem

- Spatial partitions for SOC blocks need to be specified
- Temporal variations in voltage need description
- Interface definitions for each block
- Shutdown, Back Bias, Standby behaviors are not factored into traditional boolean models
- And then there is retention! ➞
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 or posedge save or posedge restore)
if (!reset_n) q <= 0;
else if (!vdd)           q   <= 1'bx;
else if (!reset_n) q <= 0;
else if (save)  q_s <= q;
else if (restore)  q   <= q_s;
else                      q   <= d;
```

Need to simulate and verify this!

Retention is a huge verification challenge
Power-Aware Model

Required for Static Analysis, Simulation, Timing Analysis, Formal Analysis, etc..
But wait…

• Is it enough to provide the design tools?
  – What about coverage, assertions, debug?
• Successful Verification takes rigorous methodology
  – How do we build a low power verification methodology?
  – How do we migrate existing methodology?
Verification Engineers need help/training

- Digesting technology shift and bug types
- Evolving coding styles for low power
  - RTL and UPF
- Formulating/migrating test plans
- Writing constrained-random tests for low-power
- Measuring coverage
- Writing and monitoring assertions
- Debug
- Static Verification
- Post Layout Netlist Verification
Verification must now perform additional tasks

- Verify connection, placement, type of isolation/level shifting
- Include new power intent files such as UPF
- Formulate test plan for architecture correctly
- Reach good power state coverage
- Verify design works in all states, transitions and sequences
- Address firmware control of power management
- Address power-on reset issues
- Address verification at each stage of design, not just RTL
  - Verify netlist at each handoff
  - Verify Power Switch and rail connectivity
- Migrate existing testbenches, assertions, monitors to be low power aware
- Think about exhaustive constrained random and asynchronous logic testing

Bugs will escape without a rigorous methodology!
VMM – The technology of rigorous, reusable Verification

**Clear Guidelines**
- 463 Rules
- 510 Recommendations
- 383 Suggestions

**Base Classes**
- Data models
- Transactors
- Verification Environments

**Utilities**
- Message Service
- Notification Service
- Transaction-Level Interface

**Pre-Defined Functions**
- Assertions
- Random Generators
- Application Packages

**Compliance**
**Consistency**
**Scalability**
**Productivity**
VMM - Components

- Constraints
- Directed Stimulus
- Random Generator
- Transaction-Level Interface
- Generator
- Tests
- Scoreboard
- Scoreboard Application
- Application
- Transactor
- Cfg
- Register Abstraction
- Master
- DUT
- Rx
- Tx
- Performance Analyzer
- High-Level Applications
- Transactor
VMM: Widely used since 2005
VMM-LP: Industry’s First Verification Methodology for Low-Power

Broad participation by LP experts to contribute and review
VMM-LP Extends The Proven VMM Methodology

- **VMM-LP Base Classes** comprehend power shutdown, wakeup, and retention semantics

- **Power State Manager.**
  - Abstracts the power state machine(s)
  - Identifies current power state(s)
  - Notifications of transitions start/end
  - Can generate random power state transitions

- **Power-aware Components**
  - Customization of power-on reset and power-up sequences
  - Partial timeline rollbacks
  - HW/SW interaction to verify power management firmware

- **Source code under Apache 2.0**
Low-Power DUT
VMM-LP Addresses The New Challenge

**Awareness**
- Impact of LP verification
- Bugs related to low power techniques
- Documented examples from real designs

**Verification**
- LP perspective into planning
- LP-related assertions and coverage
- Modeling of LP devices such as VRM

**Reuse**
- LP-aware base classes and infrastructure
- LP-aware methodology applications
- Best-practice rules and guidelines

Source code for base class library
Automated Assertion Technology is needed for power management

• Assertions needed deep in the design/IP, but SOC integration (power intent) changes the assertion

• Writing assertions manually is painful and error prone

• Automated assertions is a reality today  Needs Protocol specification
  – Can be used to check properties as well

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTL Lines</td>
</tr>
<tr>
<td>6099</td>
</tr>
</tbody>
</table>
What about Standards?

- Universal format
  - Interoperable
  - Industry Std tools
  - *Automatic, Optimized Impl.*
  - On/Off sim

- Adv. Verification
  - Automated Assertions
  - Formal Property Checks

- Internal Scripts
  - Not Portable
  - Not Verifiable

- Proprietary formats

- Tcl Based Structural Power Intent

- Protocol Based Power Intent
Unified Power Format (UPF) Language

✓ Open format for defining power intent
✓ Used throughout flow
✓ Accellera UPF1.0 Industry-standard
✓ UPF proposed as IEEE-P1801
✓ Interoperable with multi-vendor support
✓ Industry-wide support
The Eclypse Low Power Solution

The Perfect Alignment of technology, IP, methodology, services and industry standards for Low Power Design
Eclypse Low Power Education Programs
Coming Soon…

• Advanced Verification with UPF
  – Hands-on workshop

• Low Power Curriculum
  – Developed with Taiwan’s National Chiao Tung University

• Low Power Methodology Manual
  – Being translated into Chinese & Japanese

We can work with Academia to develop Low Power Curriculum
Protocol Standards will emerge

- ACPI, USB, PCI, PCIe have Power Management Specs
- No SOC Power Management interface standard exists
  - Standards like PMBus are emerging and evolving
  - A common Power Management Protocol/Interface will increase interoperability of SOC IP

Hierarchical Power Management is on the rise

→ Increased Static(Formal) verification
The Great Unanswered Question

The Chip but does it actually save power/energy? Architecture Selection and Optimization has always been the Holy Grail of EDA!
Conclusions

Low power design impacts every stage of design and verification.

Verification of low power techniques is challenging and requires new verification technology and methodology.

Synopsys is committed to delivering the best low power design and verification solution in the market.