Guaranteed Vertical Reuse

C Execution In A UVM Environment
Guaranteed Vertical Reuse

A LITTLE BACKGROUND
Team Introduction

• IP & Sub system verification team
  - 18 ST-Ericsson employees
  - 2 subcontractors
• Seniority
  - ~ 4 years
• Expertise
  - Subsystem verification with C/TLM platform or with SV/UVM
  - Constrained random verification
  - Formal verification
What If?

- You could run SW tests in your UVM environment?
- You could get coverage from these SW tests?
- You had the flexibility of implementing your tests in UVM or SW when appropriate?
- You could reuse the same SW across your different verification platforms?
Our Project Context

MIPI standard for Low Latency Interface interconnecting 2 chips remotely using memory mapped transactions

i.e. A companion chip (slave) can share and boot from main memory of an application chip (master)
Our Project Context

Multimedia Subsystem

- DDR Memory
- Memory Controller
- IP
- Master Chip
- Slave Chip
- CPU and Caches
- Inter Connect
- Signaling
- Service
- LLI
- M-PHY

IP Under Test
Our Project Context

MIPI standard for Low Latency Interface interconnecting 2 chips remotely using memory mapped transactions.
Where We Start From?

- Only PA layer extensively verified
  - Constrained random
  - Functional coverage
  - Formal properties
Where We Start From?

- Full LLI verified with 2 back to back connected DUTs executing C tests
  - SW layer Early validation/Use Cases
Known Limitations

- Constrained Random/Coverage only @ PA layer
- No functional coverage from SW tests
- DUT not verified against a golden model
Addressing Known Limitations

Limitations
• DUT not verified against a golden model
• Random/Coverage only @ PA layer
• No coverage for SW executed

Enhancements
• Use a LLI VIP to verify complete LLI stack
• Develop CR tests with coverage @ all LLI stack layers
• Develop C API to run C code on UVM TB and gather associated functional coverage
Enhanced Verification Environment

- C API
- LLI Stack CR Tests
- LLI VIP Golden Model
Guaranteed Vertical Reuse

C API IMPLEMENTATION FOR SYSTEM VERILOG
Where We Start From?

C test LL read

Function test_ll()
...
for(addr=LL_base; addr<addr_max; addr++)
    DPI_HAL_READ32(addr, data)
.... end

SystemC TLM API

Function DPI_HAL_READ32(device, addr, data)
....
do_ll_be_read(device, LL, addr, data);
...
What Need To Be Implemented

C test LL read

Function test_ll()
...
for(addr=LL_base; addr<addr_max; addr++)
    DPI_HAL_READ32(addr, data)
...... end

SystemC TLM API

Function DPI_HAL_READ32(device, addr, data)
...
do_ll_be_read(device, LL, addr, data);
...

SystemVerilog C API Package

Task DPI_HAL_READ32(device, addr, data)
...
do_ll_be_read(device, LL, addr, data);
...

SystemVerilog TLM layer

SV DPI

Driver

SC transactor

HDL layer
SystemVerilog DPI Overview

• Allows C code to call SV tasks

```cpp
package lli_c_api_pkg;

function automatic byte DPI_dmac_drv_is_dma_enabled
  (device_type_e device);
  return DMAC[device].get_dma_enable();
endfunction: DPI_dmac_drv_is_dma_enabled

// DMA Functionality DPI Method Exports:
export "DPI-C" function DPI_dmac_drv_is_dma_enabled;
```

```c
#include "dmac_drv.h"

uint8_t dmac_drv_is_dma_enabled(void) {
  return DPI_dmac_drv_is_dma_enabled(hal_mode);
}
```

```cpp
void lli_test_wait_dma_done(uint32_t dmac_mask) {
  uint8_t i;
  if(!dmac_drv_is_dma_enabled()) return;
  ...
}
```
SystemVerilog DPI Overview

- Allows SV code to call C functions

```verilog
class lli_c_test extends lli_ss_test_base;
...
    task run_phase(uvm_phase phase);
    ...
    fork
        begin: DUT_SW
            start_DUT_c_code();
        end
        begin: TB_SW
            start_TB_c_code();
        end
    ...

    int start_DUT_c_code()
    {
        ... set_hal_mode(DUT);
        wait_n_ns(50); // wait for reset LLI
        //--- Run Test
        run_test(&lli_local);
        printf(" ************ %s Test Done ! ************\n", INSTANCE_NAME);
    }
```

```verilog
package lli_c_api_pkg;
...
// Main c routines:
import "DPI-C" context task start_DUT_c_code();
import "DPI-C" context task start_TB_c_code();
```
C Test Execution Example

```
C tests

Setup LINK
Start Traffic
Check Traffic
End test

```

```
task run_phase(uvm_phase phase);
...
start_DUT_c_code();
...
start_TB_c_code();
```
C Test Execution Example

C API calls on DUT side

• Setup Link
  – Configure DUT via APB3 VIP to R/W DUT registers

• Start traffic
  – Initiate LLI data traffic on LLI IF through DMA request via AXI 3 VIPs sequences

C API calls on TB side

• Setup Link
  – Configure VIP via its configuration API

• Start Traffic
  – Initiate LLI data traffic directly on VIP LLI IF via LLI VIP sequences
C API Implementation Details

• Setup Link implementation

task automatic DPI_HAL_READ32(input device_type_e device, input int address, output int data);

... // AXI Address range:
else if((address >= `LLI_LL_BASE_ADDRESS) && (address <= `LLI_LL_BASE_ADDRESS + `LLI_MEM_RANGE))
    do_ll_be_read(device, LL, 32, address, data);
else if((address >= `LLI_BE_BASE_ADDRESS) && (address <= `LLI_BE_BASE_ADDRESS + `LLI_MEM_RANGE))
    do_ll_be_read(device, BE, 32, address, data);
else//@anything else is configuration access
begin
    if(device == DUT)
        apb_read(device, address[15:0], data);
    else
        begin//TB configuration
            wait_n_us(10);
            if((address >= `LLI_SVC_BASE_ADDRESS) && (address <= (`LLI_SVC_BASE_ADDRESS + `LLI_SVC_RANGE)))
                begin
                    ... lli_cfg.get_config_capability_space(address[15:0], data);
                    ...
                end
        end
end
endtask: DPI_HAL_READ32
C API Implementation Details

• Start Traffic implementation

task automatic DPI_HAL_READ32(input device_type_e device, input int address, output int data);

// AXI Address range:
else if((address >= `LLI_LL_BASE_ADDRESS) && (address <= `LLI_LL_BASE_ADDRESS +
`LLI_MEM_RANGE))
    do_ll_be_read(device, LL, 32, address, data);
else if((address >= `LLI_BE_BASE_ADDRESS) && (address <= `LLI_BE_BASE_ADDRESS +
`LLI_MEM_RANGE))
    do_ll_be_read(device, BE, 32, address, data);
else // anything else is configuration access
begin
    if(device == DUT)
        apb_read(device, address[15:0], data);
    else
        begin // TB configuration
            wait_n_us(10);
            if((address >= `LLI_SVC_BASE_ADDRESS) && (address <= (`LLI_SVC_BASE_ADDRESS +
`LLI_SVC_RANGE)))
                begin
                    ... lli_cfg.get_config_capability_space(address[15:0], data);
                    ...
                end
        end
endtask: DPI_HAL_READ32
Learn More on C API details

• you can go visit https://verificationacademy.com/cookbook/CBasedStimulus
  - Tool-agnostic explanation/code example of C based stimulus testbench
RESULTS & LESSON LEARNS
Reuse Across C/ UVM Tests

**C test LL read**
Function test_ll()
... for(addr=LL_base;addr<addr_max;addr++)
    DPI_HAL_READ32(addr,data)
...... end

**C API Package**
... task do_ll_be_read(device,channel,addr,data)
    ... Launch the appropriate sequence according to initiator/action/channel parameters
    read_DUT_seq.start(m_env.v_sequencer);
    ... end

**UVM test LL read**
class LL_read_test extends uvm_test
... //launch random number of DUT->TB LL read
    seq.initiator=DUT;
    seq.action= READ;
    seq.transfer_channel= LL;
    seq.start(m_env.v_sequencer);
    ...

**UVM virtual sequence DUT/TB LL/BE read**
class lli_ll_be_read_write_seq extends virtual_sequence
... lli_ll_be_read_DUT_seq read_DUT_seq;
    ... read_DUT_seq.start(m_env.v_sequencer);
    ...... end

**UVM virtual sequence DUT LL read**
class lli_ll_be_read_DUT_seq extends virtual_sequence;
... Launch read AXI sequence on master LL channel
    AXI_read_seq.start(AXI_LL_master_sqr)
......
C Tests Functional Coverage

- Reuse existing C tests to reach coverage closure
  - Functional coverage gathered on C test run
- After C test coverage analysis, verifier may choose to
  - add C tests or SV ones to reach coverage goal
  - Remove C test if not contributing to coverage
C Tests Verification Cycle & Debug

- C test use cases mainly used at the beginning of the verification cycle
  - Used as smoke tests prior running full regression and implement additional tests
  - Used as a mini regression test after smoke tests passed
- Useful in that verification window to have C debug available together with UVM debug
- C debug achieved with printf but also with C debugger running together with RTL simulator
Verification Cockpit

• Fully automated test/regression run flow
  - Automatically distinguish C test run and SV test run
    • Load SW shared object for both DUT and TB if C test
    • Add appropriate simulator options
  - Automatically collect/merge coverage
• Common SW framework across the company to maximize reuse
  - SW modules separated in groups
    • IPs modules, including IP, drivers, interfaces and tests
    • platform-specific drivers and their interfaces
    • Project-specific elements, including SoC test bench maps and global test runners
Verification Cockpit Overview

UVM/C tests
Regression Suite

SV tests
C tests

Optimized test suite generator

Coverage Results (merge/rank)

Verification Cockpit
Vertical Reuse

• IP verification level
  - 10 test cases written in C running the common IP Use Cases within our UVM environment
    • required for SoC integration and SW driver validation
  - Remaining tests (~50) developed in SV/UVM for exhaustive verification

• SOC verification level
  - C tests reused with almost no modification

• Software level
  - LLI driver reused with minimal changes
Conclusion

Achievements

- Full functional coverage reached with a mix of HW & SW tests
- 1st LLI driver code developed prior full RTL availability
  - Created a Mixed hardware/software approach
- Horizontal but also vertical reuse becomes a reality thanks to this C API
  - C API Package developed for reuse so we can use it for next projects

Improvements

- For a full reuse of the C tests/drivers SOC and system developers ask for more
  - Wait functions replaced with Interrupt routines
  - Reduced LLI driver code size
  - code size too big → didn’t fit in the memory
Thank You!

QUESTIONS?
Overview:

This template contains

- Information on formatting your presentation for DVCon
- Information on how much time you have for presenting your paper
- Tips for including code examples and other information
- Providing copies of your presentation

Change this in the master slide to reflect the maximum number of slides in your presentation
DVCon Presentation Format

• This template is a guideline for your paper presentation
  - You are not required to use this template
  - If you use your own template, please include the DVCon and Accellera logos on each page, as in this template
    • In the View menu, select Master→Slide Master
    • Select the DVCon logo in the top right, and copy it
    • Paste it into your slide master

• Your company logo may appear on the FIRST SLIDE (title slide) ONLY
Recommended Fonts

• Use a sans serif font, such as Tahoma, Arial or Helvetica
• Page titles should be in 32 point
• 1st level bullets should be 24 to 28 points
  – 2nd level bullets should also be 24 to 28 points
    • Third and fourth level bullets can use 20 to 24 points
DVCon Presentations

• You may use OpenOffice, LibreOffice, or PowerPoint to create/present your slide presentation. All three programs will be available to you at Conference.
  – An LCD projector will be provided
  – You can connect your own laptop to the projector, or…
  – A computer with OpenOffice, LibreOffice, and PowerPoint will be available

• Use a USB drive to load your presentation on to the laptop in your session room. You are responsible for bringing your final presentation with you to your session.
Adding Code and Notes to Your Presentation

• Code examples should use a 16 to 18 point mono spaced font, such a Courier
• You can add note boxes to enrich your presentation and call attention to important points
  – Use 16 to 18 point Arial or Helvetica font for notes

module addbit (input wire a,
               input wire b
               input wire ci,
               output reg sum, co);

functionality & timing

endmodule

ANSI C style port lists
Paper Presentation Time

• You will have a total of **30 minutes** to…
  - Set up and start your presentation
  - Present your paper
  - Answer Questions

• We recommend:
  - Arrive early to load and test your presentation before the session begins
  - Allow 20 minutes to present your paper
  - Allow 10 minutes for getting started and then for questions at the end
Additional Tips

• Keep each slide brief
  - Use short phrases, not complete sentences
  - Too many words forces the audience to choose between reading and listening to you

• Limit yourself to just a few bullets per slide
  - Too much information on a slide is distracting
  - Use two or three slides to explain complex ideas

• Use presentation software “builds” to bring up information a little at a time

Add some clip art to keep your audience’s attention (but don’t over do it)
Hard Copies of Your Presentation Materials

- Your full paper is included in the DVCon Proceedings
- Your presentation slides are *not* included in the Proceedings, but will be in the DVCon archives
  - If you want the audience to have copies of your presentation slides, you can...
    - Provide copies that can be picked up after the session
      - Bring 50 to 75 copies
    - Provide a web site location where copies can be downloaded
  - Tutorial presentations are included with the conference materials
    - Tutorial presenters do not need to provide additional copies
Expert Presentation Tips for New and Advanced Presenters

(with a review of *ALL* of the basics!)
Practice!

- Practice your presentation
- Practice your opening
Starting the Presentation

- 1st - talk slow!
- Nervous speakers have a tendency to speak too fast
- When you first stand up to speak …
- Work the first two slides for a killer introduction
  - Who are you?
  - What are you going to present?
  - What will the audience learn?
Be Confident

- Be Confident
- Don't make apologies!
- Don't share inadequacies!

Questions during the presentation -
- Ask audience to hold questions by saying, "Let me hold all questions until the end of the presentation."
- If the questioner persists, the moderator should step in!
Experienced Presenters

• Where to stand?
  – Podium - if you want
  – To the left of the presentation screen

DON’T STAND BETWEEN THE SCREEN AND THE AUDIENCE!!

DON’T TURN YOUR BACK ON THE AUDIENCE!!

• Where possible, stand to the left of the presentation
  – People are used to reading from left-to-right
  – People will focus from you (the speaker) to the presentation
Setting Up To Present

• Clear your walking area
  - Approaching the screen

• Portable microphone
  - Place it high on the collar
  - Do not place it on a side-collar
  - Do not place it too low
    • Moving your head from side-to-side should not be the volume control
    • The volume will have to be turned up and feedback is inevitable

Every screen has at least one leg designed to trip the presenter!

Under the chin – 2nd button on a shirt
The Conference Support Group

• You should think of the conference attendees as a big support group

• The conference attendees aren't looking to catch mistakes
  – The attendees want to learn from your presentation
Learn From Your Audience

• We give as much as we know
  - and then we learn from the audience what we didn't know

• NO MARKETING!!
  - Let the introduction market you
  - Then give a killer technical presentation - that will attract people
Laser Pointers & Use Humor

• Laser pointers
  - Avoid them or learn to control them
  - Do not point them at your audience

• Use Humor!

If you are having fun ... your audience is having fun!
Hands-On Presentation

- People follow your hands - don't be afraid to use them
  - Hands by your side - one of the best resting positions
  - Crossing arms? No!
  - Giving a prayer? No!

- Silence is better than "uh" and "um"
  - Silence gives your audience a chance to think
  - It also gives you a chance to think

It feel awkward but looks normal

Thinking is good!
Common Presentation Mistakes

- Speak to the audience
  ... not to the screen!

- Walking back and forth in front of the audience

- Too many slides
  - the presentation is going long - audience loses interest

- Identify slides you can skip
  Type in the new slide number and press <enter>

Also, keep extra slide page numbers handy
Summarizing and Sharing

- Abbreviate and repeat the significant points
  - Don't summarize the topics
  - Summarize the guidelines

What should the audience take away from your presentation?

- Repeat questions before answering

Moderators should help remind the presenter to repeat the question
Summary of Important Guidelines

- Before presenting, watch those fluids!
- Speak slow!
- Stand to the left and out of the way
- Give a killer introduction
- NO MARKETING
- If you must use a laser pointer, control it!
- If you run long, be prepared to skip slides
- Summarize important guidelines!

- Repeat all audience questions before answering them