2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Practical Scheme to Enhance Verification Turn-Around-Time by Using Reusable Harness Interface (RHI)

Jong pil Jung, Hyunju Lee, Jaejin Ha, and Yonghee Im

SK hynix

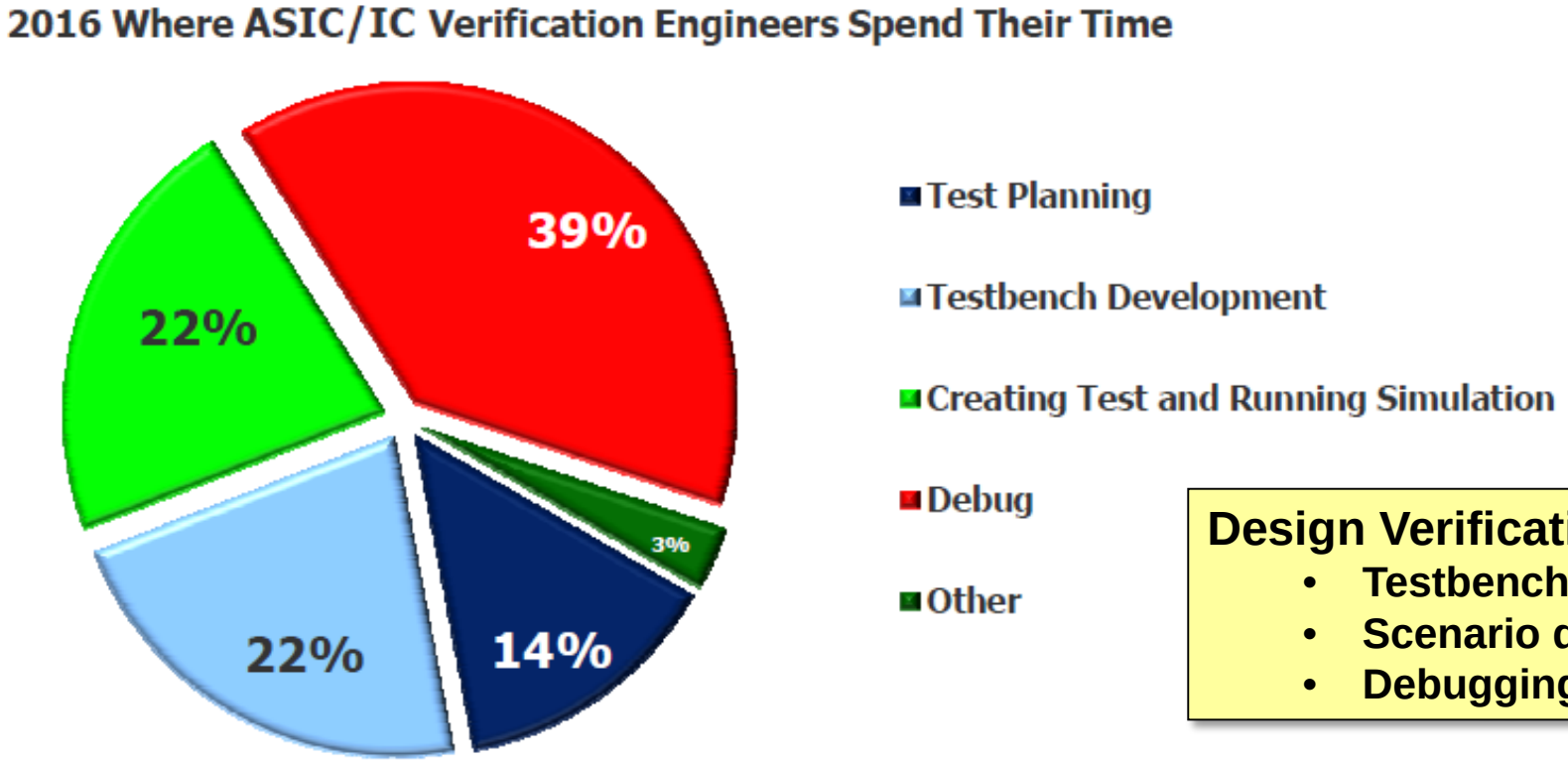
2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Agenda

- Introduction
  - Verification Turn-Around-Time(TAT) Issue
- Proposed Scheme
  - Merge DPI-C Sequences into UVM Environment - Step1
  - Reusable Harness Interface(RHI) - Step2
- Experiments
- Conclusions

Introduction : Verification TAT Issue

- Resource Trends



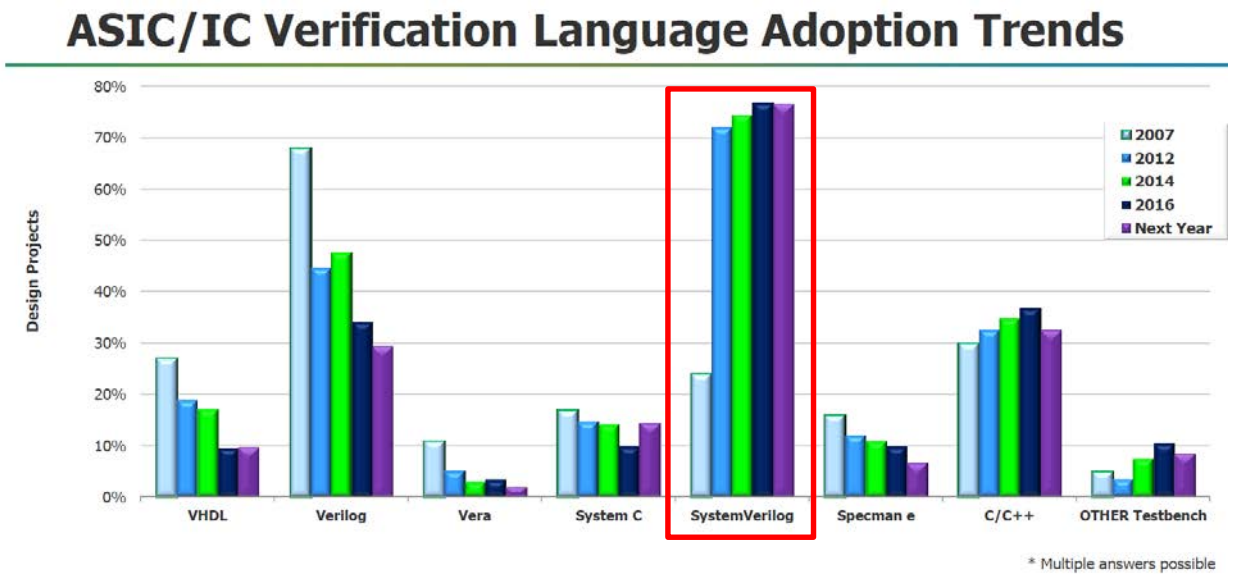
Design Verification Turn-Around-Time(TAT)

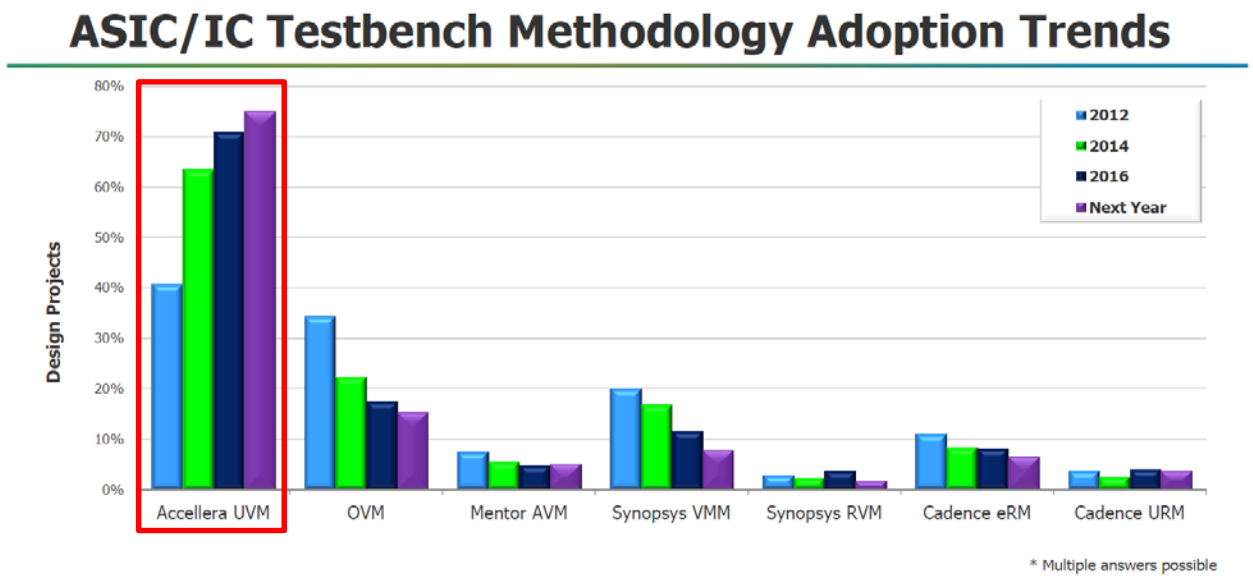
- **Testbench build-up TAT**
- **Scenario development TAT**
- **Debugging TAT**

※ Reference : Harry D. Foster "Trends in Functional Verification: A 2016 Industry Study" DVCon, 2017

Introduction : Verification TAT Issue (Cont.)

- Verification Language and Methodology Trends





One of drawbacks

- **Irritating re-compiling TAT while developing scenarios**

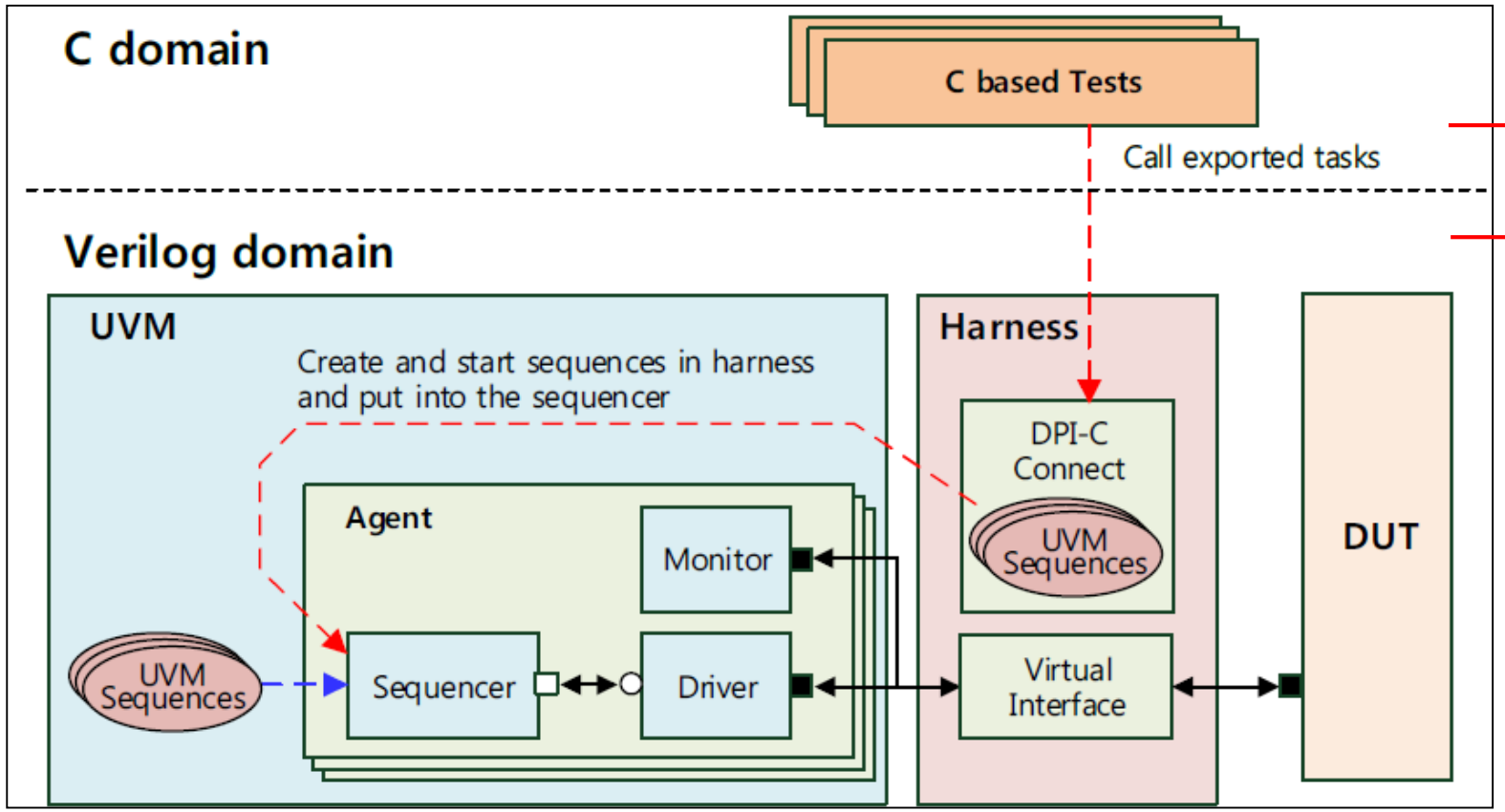
※ Reference : Harry D. Foster "Trends in Functional Verification: A 2016 Industry Study" DVCon, 2017

Proposed Scheme :

Step1 - Merge DPI-C Sequences into UVM Environment

Merge DPI-C Sequences into UVM Environment

- Testbench with DPI-C sequence and UVM environment combined.



Benefits

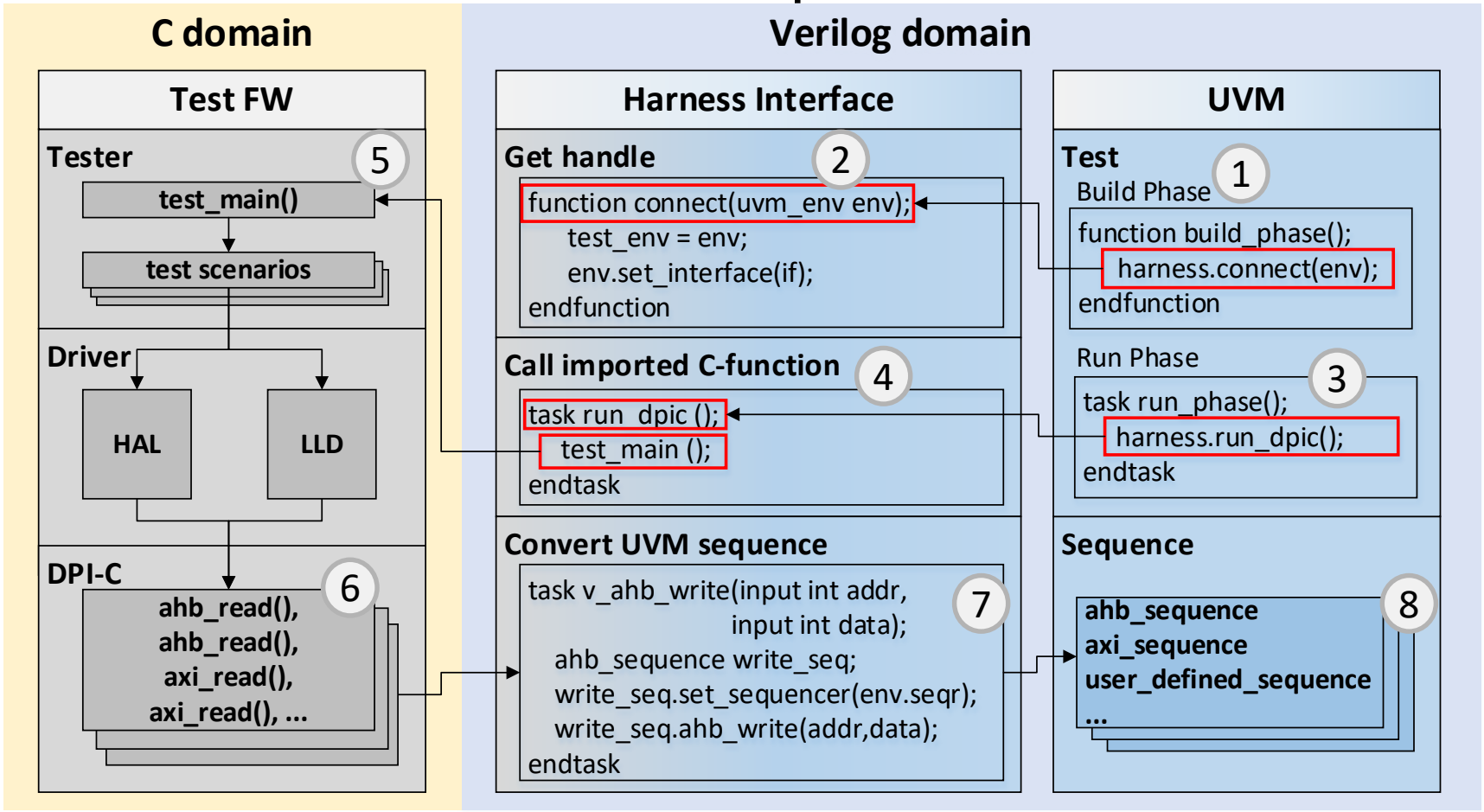
- Faster builds
- Dynamic linking

Conventional Harness :
To bridge DUT and UVM

Proposed Harness :
A new role is added – linking DPI-C and UVM

Merge DPI-C Sequences into UVM Environment (Cont.)

- How to combine DPI-C sequences and UVM environment

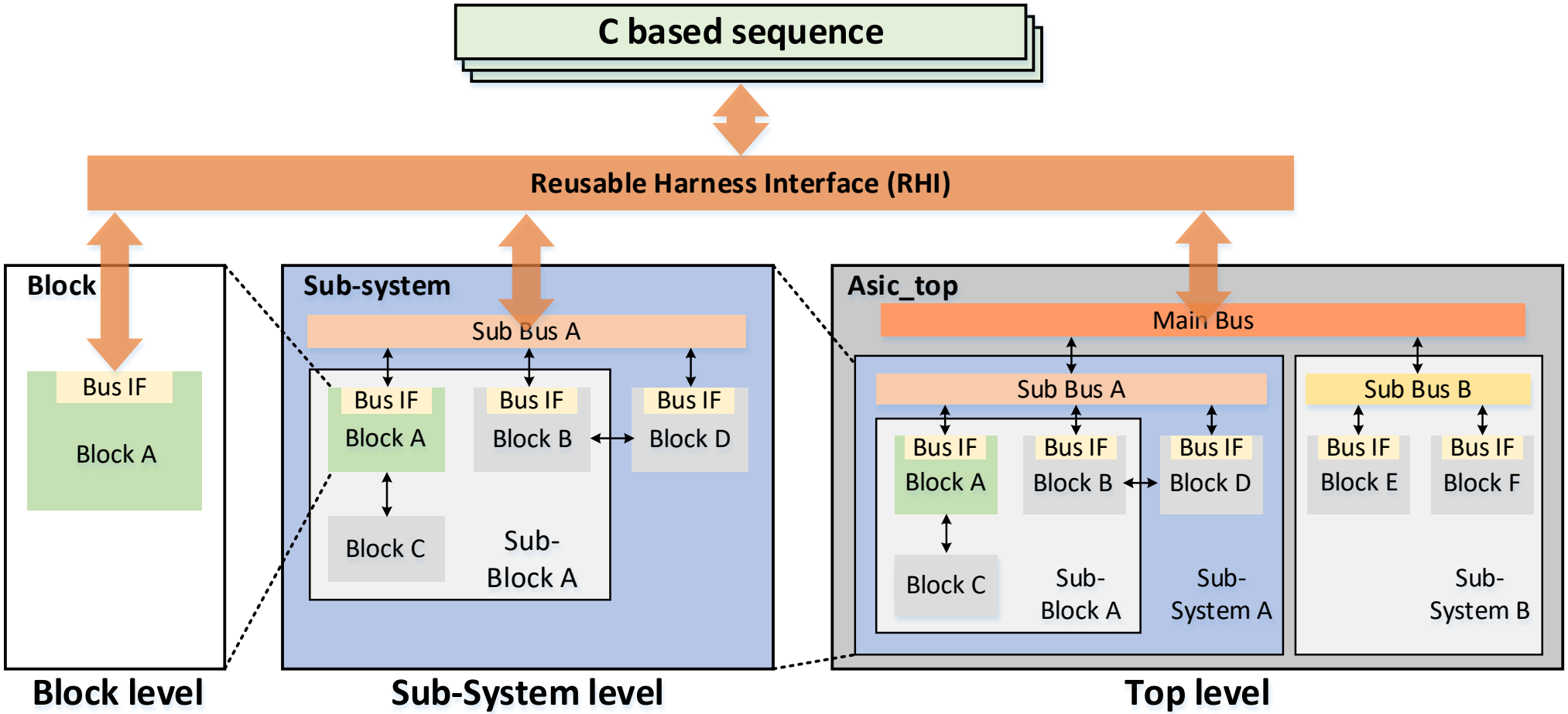


Proposed Scheme :

Step2 - Reusable Harness Interface (RHI)

Reusable Harness Interface (RHI)

- How to apply RHI to each different hierarchical testbench



Reusable Harness Interface (RHI) (Cont.)

- Reusable Harness Interface(RHI) is an interface format in which **aforementioned harness interface can be systematically reused** in a difference level.
- RHI consists of three major parts:
  - *Harness interface*
  - *Harness binding*
  - *Path define*

REUSABLE : Harness Interface + Harness Binding
Only path define file is required

Reusable Harness Interface (RHI) (Cont.)

- **Harness interface**
  - Connecting DUT and UVM environment

```
/** HDL defines */
`include hdl_defines.h

/** Import components */
import ahb_env_pkg::*;
import ahb_seq_lib_pkg::*;

/** Interface declaration */
interface ahb_harness(parameter string ENV_PATH_PARAM = "uvm_test_top.xxx.xxx")();

    /** Next slide */

endinterface : ahb_harness
```

Reusable Harness Interface (RHI) (Cont.)

- **Harness interface (Cont.)** – Inside of interface declaration

```
/** Handle of components */
ahb_env dpic_ahb_env;

/** Declaration of imported and exported functions and tasks */
import "DPI-C" context task main(int arg0, int arg1, ... , int arg3);
export "DPI-C" task v_ahb_swrite;
export "DPI-C" task v_ahb_sread;

/** Description imported and exported functions and tasks */
task v_ahb_swrite(input int size, intput int addr, input int wdata);
    /** Convert to uvm_sequence */
    ...
endtask : v_ahb_swrite

...
```

Reusable Harness Interface (RHI) (Cont.)

- **Harness interface(Cont.)** – Inside of interface declaration

```
/** Instantiation of virtual interface */
ahb_interface #()u_ahb_interface (
                                      .xxx ( `DUT_TOP.xxx ),
                                      ...
                                  );
/** Set VIF for ENV */
initial begin
    uvm_config_db#(ahb_interface)::set(uvm_root::get(), ENV_PATH_PARAM,
    "u_ahb_vif", u_ahb_interface);
end

/** Get ENV component */
initial begin
    uvm_config_db#(ahb_env)::get(uvm_root::get(), ENV_PATH_PARAM,
    "dpic_ahb_env", dpic_ahb_env);
end
```

Reusable Harness Interface (RHI) (Cont.)

- **Harness binding**
  - Instantiates harness interface and binds it to `TB_TOP

```
/** import components */
import xxx_env_pkg::*;

/** HDL path defines */
`include "path_defines.sv“

/** Interface to connect with C sequence */
`include "ahb_harness.sv“

/** Binding interface */
bind `TB_TOP ahb_harness#(`ENV_PATH) u_ahb_harness ();
```

Reusable Harness Interface (RHI) (Cont.)

- **Path define**
  - Hierarchical HDL path defines such as block, sub-system and top level

```
/** Define design hierarchy path for block level */
`define `TB_TOP block_tb_top
`define `DUT_TOP `TB_TOP.u_block

/** Define ENV path for block level */
`define `ENV_PATH “uvm_test_top.m_block_env”
```

Block level

```
/** Define design hierarchy path for subsystem level */
`define `TB_TOP subsystem_tb_top
`define `DUT_TOP `TB_TOP.u_subsystem.u_block

/** Define ENV path for subsystem level */
`define `ENV_PATH “uvm_test_top.m_subsystem_env.m_block_env”
```

Sub-system level

Reusable Harness Interface (RHI) (Cont.)

- **Path define (Cont.)**
  - Hierarchical HDL path defines such as block, sub-system and top level

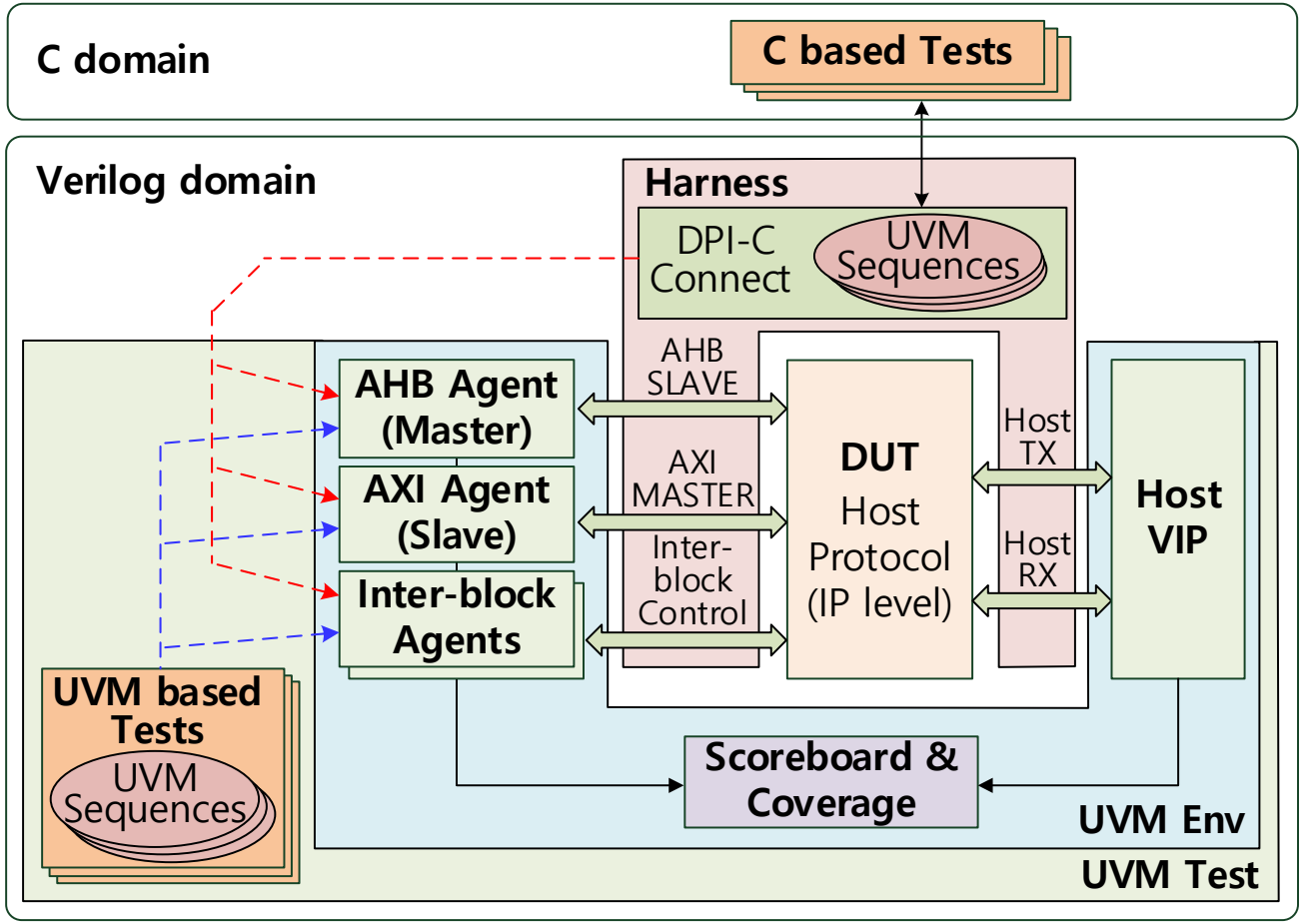
TOP level

```
/** Define design hierarchy path for top level */
`define `TB_TOP top_tb_top
`define `DUT_TOP `TB_TOP.u_top.u_subsystem.u_block

/** Define ENV path for TOP level */
`define `ENV_PATH “uvm_test_top.m_top_env.m_subsystem_env.m_block_env”
```

Experiments : NAND controller

- Block diagram for test experiments



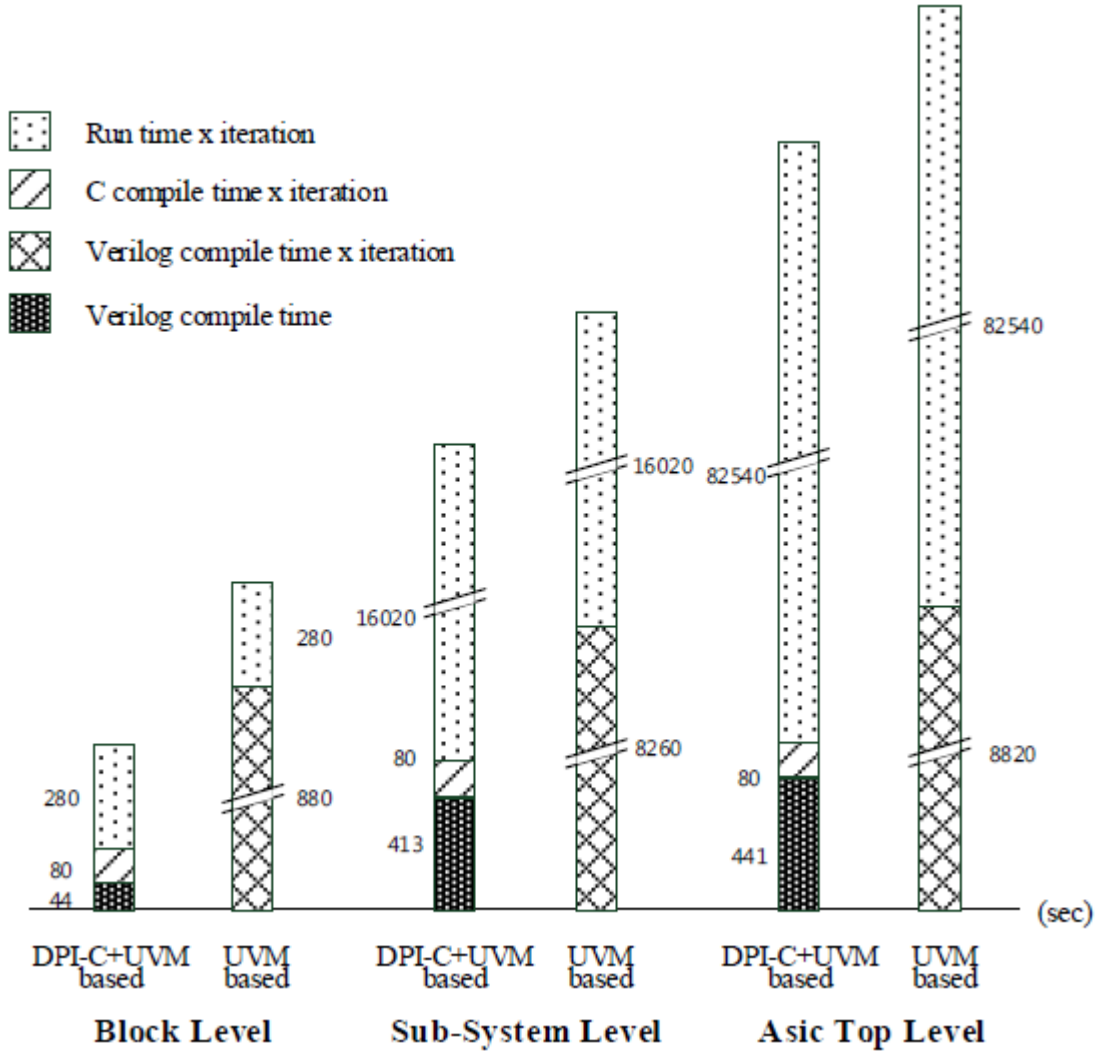
- Host protocol design of a NAND controller
- DPI-C test is translated into AXI/AHB UVM sequences and they are fed to AXI/AHB Agent

Experiments : NAND controller

- Results
  - The merged testbench can give you more benefit in TAT.

	Details	Time (sec)
UVM based	Verilog compile time x Iteration	8260
	Run time x Iteration	16020
	Total sequence develop TAT	**24280**
DPI-C + UVM based	Verilog compile time	413
	C compile time x Iteration	80
	Run time x Iteration	16020
	Total sequence develop TAT	**16513**

※ Iteration Cost : "The amount of time required to iterate once in sequence development TAT."
Assumes 20 as the average iterations



2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Conclusions

- This paper presents a practical scheme to **enhance verification turn-around-time (TAT)**
- **Reduction of sequence development TAT** in case of merging DPI-C based codes into UVM testbench via harness interface
- **Reusable Harness Interface (RHI)**
  - Harness interface, Harness binding, and Path define
- **Reduction of debugging TAT** in case of using across hierarchy via RHI

2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Thank you