Verilog HDL:
Programming Language Interface

Pravin Zode

Outline

- Introduction
- Applications of PLI
- Types of Verilog PLI
- How PLI Works ?
- Example of PLI (Hello World)

Introduction

- PLI enables calling C/C++ functions from Verilog simulations
- It provides a way to extend Verilog functionality using custom tasks and functions
- Custom system calls can be created using PLI, allowing features beyond Verilog’s built-in syntax
- Used for debugging, simulation control, and custom tool integration

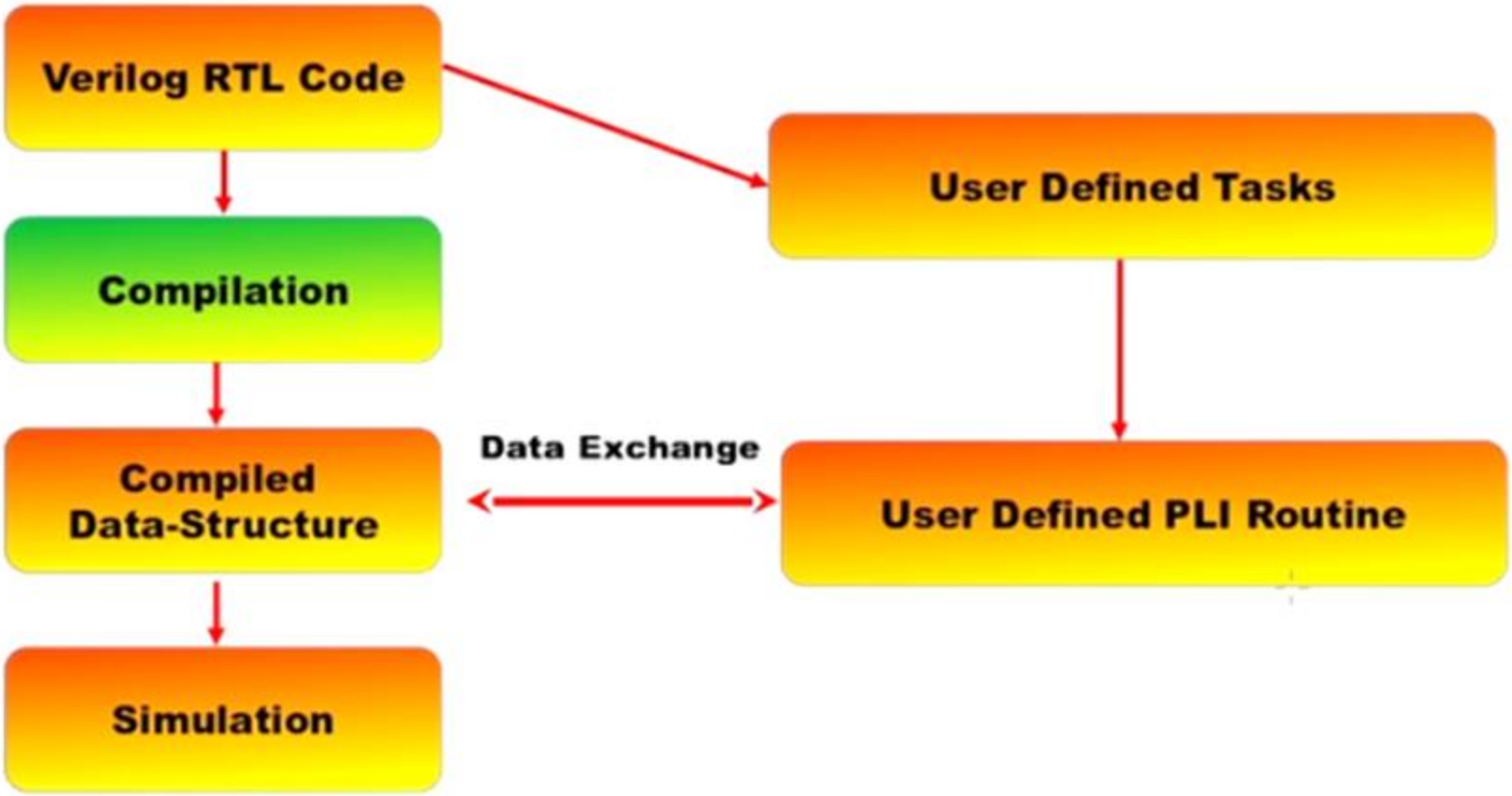
Applications of PLI

- Power analysis (Measure and Optimize)
- Code coverage tools (Test Coverage)
- Modifying simulation data structure (Accurate Delay)
- Custom output displays: (detailed reports and logs)
- Co-simulation: Integrate Verilog with external tools
- Design debug utilities: Access internal signals for debugging
- Simulation analysis: Extract and process simulation data
- C-model interface: Speed up simulations
- Testbench modeling: (Dynamic testbench generation)

Types of PLI

- **TF (Task and Function) Interface**
  - Provides access to Verilog system tasks and functions
  - Allows calling C functions from Verilog
- **ACC (Access) Interface**
  - Provides hierarchical access to internal Verilog objects
  - Used for debugging, signal monitoring, and modifying simulation behavior
- **VPI Routines (Verilog Procedural Interface)**
  - Introduced in Verilog 2001 as a replacement for TF and ACC routines
  - Modern standard for accessing design details in Verilog.

How PLI works ?

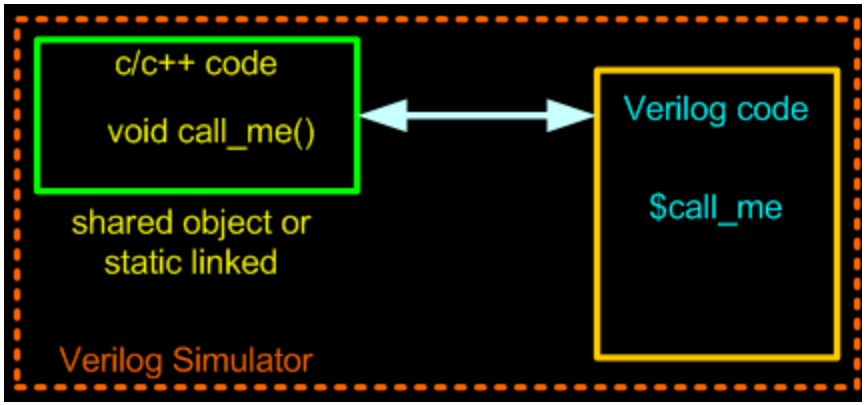


PLI System Tasks

- **$tf_ routines:** Used to interact with Verilog system tasks
- **$acc_ routines:** Used to access and manipulate simulation objects
- **$vpi_ routines:** Modern, flexible API replacing older TF and ACC interfaces.

Steps to use PLI in Verilog

- Write the functions in C/C++
- Compile the C/C++ code to generate shared libraries (*.DLL for Windows, *.so for UNIX)
- Use these functions in Verilog testbench
- Pass the compiled shared library to the simulator during the compilation process (linking process)
- Run the simulation



Hello World using PLI

```
1   #include <stdio.h>
2
3     void hello ()
4     {
5       printf ("\nHello Deepak\n");
6     }
7
8
9     module hello_pli ();
10
11    initial begin
12      $hello;
13       #10  $finish;
14    end
15
16    endmodule
```

Thank you !

Happy Learning