Accelerating RTL Simulation Techniques

Lior Grinzaig

ADVA Optical Networking

Agenda

- Why simulation time is important?
- What can we do to improve performance?
- Performance related tips.

Why simulation time is important?

- Fast simulation environments:
  - remaining focused.
  - low penalty for mistakes.
  - enables “trial and error” method.
- Slow simulation environments:
  - wasting time.
  - frequent context-switches causes mistakes.
  - Frustrating *working environment*.

What can we do?

- Hardware level.
- Simulator level.
- Code level.
  - General programming coding style.
  - **Coding style for HDL simulation & verification.**

Code level acceleration

- There are two types of changes:
  - “Micro code” modifications.
  - “Macro code” modifications.

“Micro” examples - Sensitivity Lists

- Remember:

  every triggering event causes code execution.

- Optimization:

  Remove unnecessary triggering conditions.

Asynchronous Example

- Consider this code for buffer selection:

```
wire IN0 = IN;
wire #(25) IN1 = IN0;
wire #(25) IN2 = IN1;
wire #(25) IN3 = IN2;

always @(*)
begin
        case (DELAY_SEL)
                2'd0 : OUT = IN0 ;
                2'd1 : OUT = IN1 ;
                2'd2 : OUT = IN2 ;
                2'd3 : OUT = IN3 ;
        endcase
end
```

- Complexity of ~N^2 (N is the number of phases).

Asynchronous Example

- Modification:

```
wire         IN0 = IN && (DELAY_SEL=='d0);
wire #(25) IN1 = IN && (DELAY_SEL=='d1);
wire #(50) IN2 = IN && (DELAY_SEL=='d2);
wire #(75) IN3 = IN && (DELAY_SEL=='d3);

always @(*)
begin
        case (DELAY_SEL)
                4'd0 : OUT = IN0 ;
                4'd1 : OUT = IN1 ;
                4'd2 : OUT = IN2 ;
                4'd3 : OUT = IN3 ;
        endcase
end
```

- Complexity of ~N.

Asynchronous Example

- ✓ This modification accelerated simulation of 40M gates SoC by a factor of 2!

How?

- There where 128 possible phases.
- The delay module was instantiated on each bit of 128 bits bus.
- That bus run on the fastest clock in the testbench.

Synchronous Example

- Consider this code for transaction counter:

```
always @(posedge clk)
begin
        if ( (VALID == 1) && (READY == 1) )
        begin
                count++;
        end
end
```

Synchronous Example

- Modification:

```
Initial begin
        forever
        begin
                wait ( (VALID == 1) && (READY == 1) );
                count++;
                @(posedge clk);
        end
end
```

Macro code modifications

- Focusing at the block/systems level.
- Ask yourself how and when components interact.

Examples for macro code modification

- Adjusting your code to your need.
- System modes.

Types of needs

- “Coverage” – quality stamp.
- “Development” – fast feedbacks.
- “Debug” – regenerate errors faster.

	Coverage	Debug	Development
Speed	nice to have	important	important
accuracy	accurate	reasonable accuracy	basic accuracy

What can be change?

- Several examples:
  - Disable analysis components.
  - Using *force*s.
  - Using BFMs and stubs.

Example For flexibility support

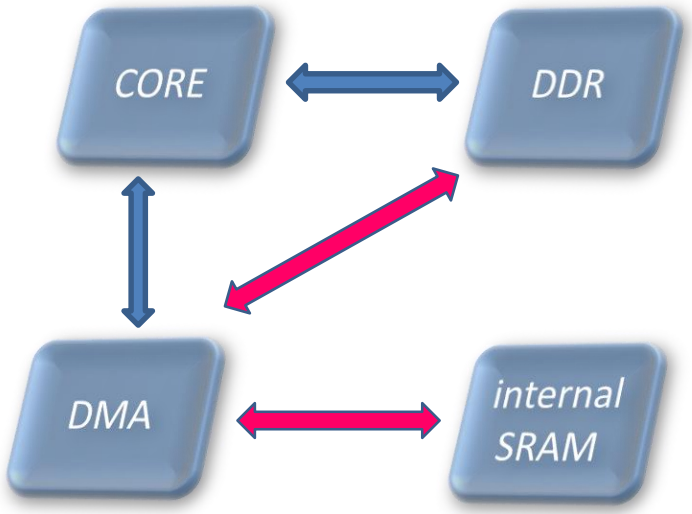
```
parameter DIVIDER1_SIMPLIFIED_MODEL = 0;

generate if ( ! DIVIDER1_SIMPLIFIED_MODEL )
begin :package_model
      divider      u_divider1
      (
            .port_A (a),
            .port_B (b),
            .port_C (c)
      );
end
else begin : simplified_model
      simpified_divider      u_divider1
      (
            .port_A (a),
            .port_B (b),
            .port_C (c)
      );
end
endgenerate
```

System Modes

- "Power down" your DUT/testbench:
  - Gate block clocks, leave them in reset.
  - Don't be afraid to use non-system scenarios for debug mode.
- Find optimal clock scheme for simulation time.

Example for optimal clock scheme



agent	src	dest	Core : DMA : DDR : SRAM
DMA	DDR	SRAM	1 : 3 : 10 : 3
CORE	DDR	DDR	1 : 0 : 5 : 0

Finding the bottlenecks

- How to find out where is my problem?
  - Use profiler.
  - Tip: analyze by stages.

True acceleration measurements

- Measure it right.
- Compare it right.
  - Machine load – use statistical analysis.
  - Random tests.

To Conclude…

- **There is hope!**
  - Slow simulations are not necessarily decreed by fate.
- **There are tools!**
  - Use the simulator profiler.
- **It worth your time!**
  - Invest time to save time.
  - **Engineers** as well as **managers** should pay attention to the performance aspect of their code.

Questions