Daily Assessment format

Date: 09/06/2020
Name: Jyothi S Donne
Course: Desv
Usn: 4AL17EC037
Topic: Hardware modeling using verilog
• FPGA & ASIC interview questions
Github repository: Jyothi-course

forenoon session details

Image of session
course on Hardware modeling using verilog

main objectives of the course

- Learn about the verilog hardware description language
- understand the difference blw behavioural & structural design styles
- Learn to write test benches & analyze simulation results
- Learn to model combinational & sequential ckts
- Distinguish blw good & bad coding practices
- case studies with some complex designs

Moore's law

- Exponential growth
- Design complexity increases rapidly
- Automated tools are essential
- must follow well defined design flow.

simplistic view of design flow

Design ideal
|
Behavioural design
|- - - - -> flow graph, pseudo code
Datapath design
|- - - - - -> bus/register structure
Logic design
|- - - -> gate/F-F netlist
physical design
|- - -> transistor layout
manufacturing
|
chip/Board

Other steps in design flow

- Simulation for verification
  At various levels: logic level, switch level, ckt level
- Formal verification
  used to verify the design through formal techniques
- Testability analysis & test pattern generation
  Required for testing the manufactured device

Report

Introduction

Hardware modeling using verilog VLSI design process

- Design complexity increasing rapidly
- The present testbed.

Moore's Law

- Exponential growth
- Design complexity
- Automated tools are essential
- Must follow well defining flow

VLSI design flow

- Standardized design procedure
- Encompasses many steps
- Specification, synthesis, simulation, layout.
- Need to use Computer Aided design (CAD) tools

→ Two competing HDLs
verilog HDL

→ Simplistic view of design flow

- Behavioural design
- Datapath design
- Logic design
- physical design & manufacturing

→ other steps in the design flow
- simulation for verification
- testability analysis
- formal verification
- test pattern generation

→ FPGA & ASIC interview question

→ T flip flop

```
module tff (input clk, rst, t, output reg, q);
always @ (posedge clk)
begin
if(rst)
  q<=0;
else
  if(t)
    q<=~q;
  else
    q<=q;
end
endmodule.
```