Computer Architecture

Week 5: System Verilog Tutorial II

Fenerbahçe University

Professor & TAs

Prof: Dr. Vecdi Emre Levent

Office: 311

Email: emre.levent@fbu.edu.tr

TA: Arş. Gör. Uğur Özbalkan

Office: 311

Email: ugur.ozbalkan@fbu.edu.tr

Course Plan

- System Verilog for Synthesis II

User Defined Types

- It is not possible to create a user-defined type in Verilog language.
- A keyword named "typedef" has been added to create a user-defined type in Systemverilog.

User Defined Types

- By adding enumeration to increase the readability of the code, it is possible to write readable texts instead of writing fixed numbers, especially in case machines.

System Verilog

User Defined Types

```
 typedef enum logic[2:0] {GETTIMELOW = 3'b000, GETTIMEHIGH = 3'b001,
                GETFREQLOW = 3'b010, GETFREQHIGH = 3'b011,
                GETFREQ2LOW = 3'b100, GETFREQ2HIGH = 3'b101,
                SENDDATA = 3'b110, IDLE = 3'b111} state_t;
state_t state;


...


    case(state)
       GETFREQ2HIGH: begin
          state <= SENDDATA;
       end

       SENDDATA: begin
          if(timeiszero) begin
         state <= GETTIMELOW;
          end else begin
         state <= SENDDATA;
          end
       end
```

```
       IDLE: begin
          if(button_sync2) begin
             state <= GETTIMELOW;
          end
       end

     default: begin
        state <= state.next();
     end
  endcase // case (state)
```

Tasks

- Tasks are structures that can take parameter and argument input.

Tasks

```
reg [3:0] deneme = 1;

task count (input [3:0] in, output [3:0] out );
begin
     out <= in + 1;
end
endtask

always @(posedge clk) begin
    count(deneme, deneme);
end
```

Functions

- Functions are similar to tasks. There is a return value as a difference.
- In Tasks, returning is done through ports.

Functions

```
function [5:0][7:0] testFunc;
    input [3:0] opcode;
    case (opcode)
        NOP     : testFunc = "NOP";
        JMP     : testFunc = "JMP";
        JMPZ    : testFunc = "JMPZ";
        JMPNZ   : testFunc = "JMPNZ";
        LDA     : testFunc = "LDA";
        ADD     : testFunc = "ADD";
        SUB     : testFunc = "SUB";
        AND     : testFunc = "AND";
        OR      : testFunc = "OR";
        NOT     : testFunc = "NOT";
        LSL     : testFunc = "LSL";
        LSR     : testFunc = "LSR";
        STA     : testFunc = "STA";
        default : testFunc = "*ERR*";
    endcase
endfunction
```

Parameters

- parameter: It works on the same task as Verilog standard.
- localparameter: The difference with the parameter cannot be given as a parameter from the top module of a module. It can only be defined when you want to use it as defined inside the module.

Parameters

```
localparam BSER1_NMA = 'h003;
localparam RSTP0_NMA = 'h002;
localparam HALT1_NMA = 'h001;
localparam TRAC1_NMA = 'h1C0;
localparam ITLX1_NMA = 'h1C4;

localparam TVN_SPURIOUS = 12;
localparam TVN_AUTOVEC = 13;
localparam TVN_INTERRUPT = 15;

localparam NANO_DOB_DBD = 2'b01;
localparam NANO_DOB_ADB = 2'b10;
localparam NANO_DOB_ALU = 2'b11;
```