Assignment 01 – Combinatorial logic

EDB HDL

ANM, v03

Introduction

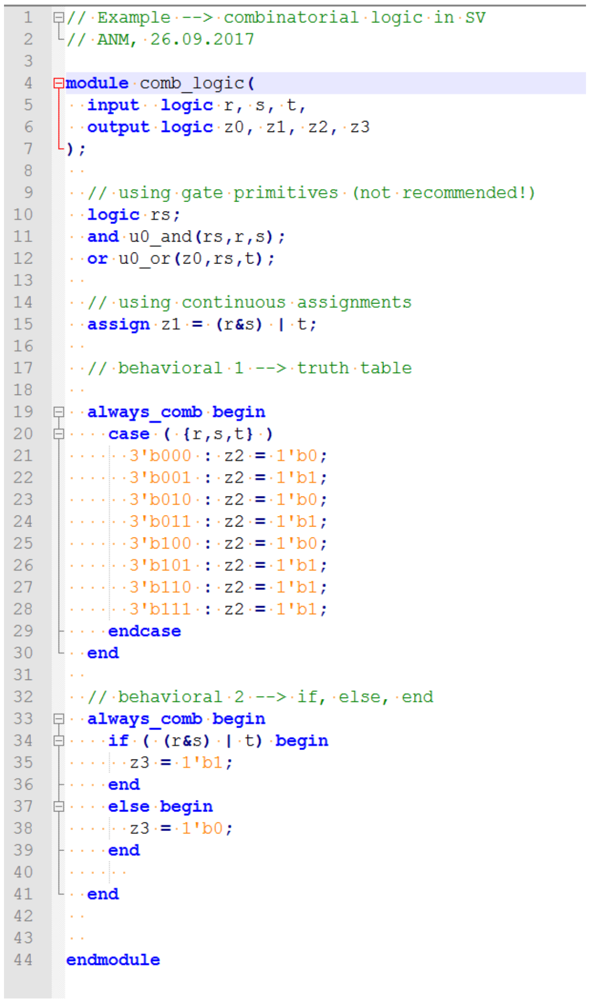
Implementation of combinatorial logic with SystemVerilog is in focus in this assignment.

Combinatorial logic can be implemented in different ways:

- Using gate primitives (not recommended)
- Using continuous assignments
- Using a behavioural description in an `always_comb` block.

A brief example shall illustrate the concepts:

The following combinatorial logic shall be implemented: `Z = (R & S) | T`



```
// Example --> combinatorial logic in SV
// ANM, 26.09.2017

module comb_logic(
  input  logic r, s, t,
  output logic z0, z1, z2, z3
);

  // using gate primitives (not recommended!)
  logic rs;
  and u0_and(rs,r,s);
  or u0_or(z0,rs,t);

  // using continuous assignments
  assign z1 = (r&s) | t;

  // behavioral 1 --> truth table

  always_comb begin
    case ( {r,s,t} )
      3'b000 : z2 = 1'b0;
      3'b001 : z2 = 1'b1;
      3'b010 : z2 = 1'b0;
      3'b011 : z2 = 1'b1;
      3'b100 : z2 = 1'b0;
      3'b101 : z2 = 1'b1;
      3'b110 : z2 = 1'b1;
      3'b111 : z2 = 1'b1;
    endcase
  end

  // behavioral 2 --> if, else, end
  always_comb begin
    if ( (r&s) | t) begin
      z3 = 1'b1;
    end
    else begin
      z3 = 1'b0;
    end

  end


endmodule
```

Figure 1: Different ways to implement combinatorial logic in SystemVerilog.

Task 01 – Seven segment decoder

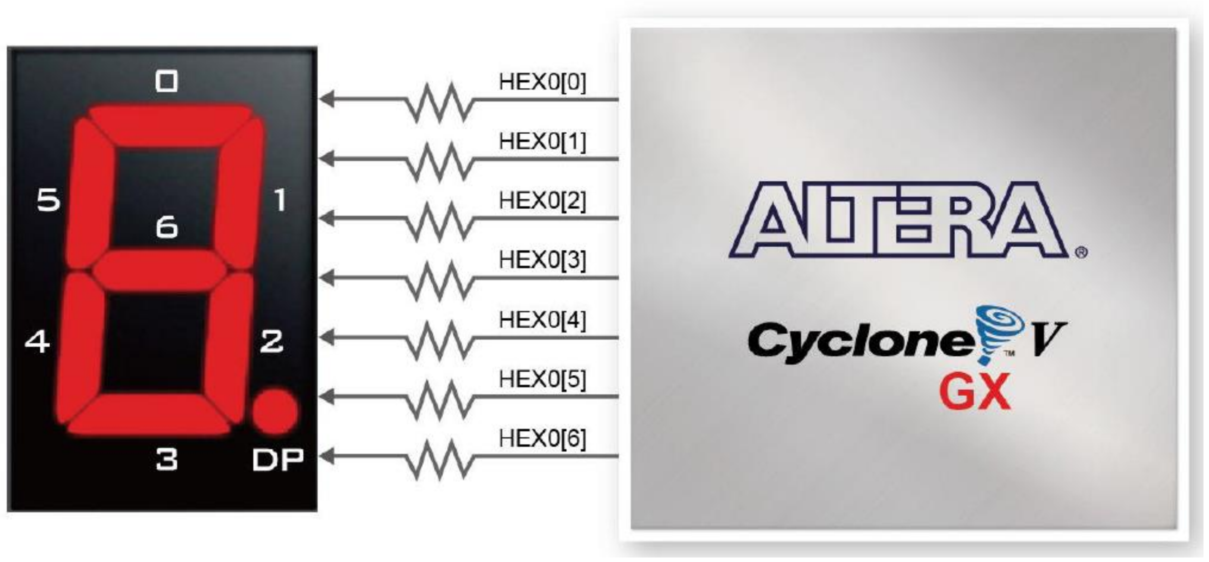
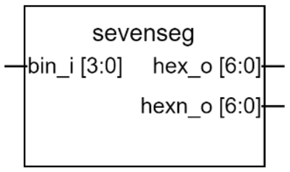


Figure 2: Seven segment display as used on the FPGA development board.

- Create the folder structure
  - ./sevenseg/sim
  - ./sevenseg/src
- [1cp] Create a SystemVerilog module "sevenseg.sv" in the ./sevenseg /src folder that implements the decoder. You can choose in which way you want to implement the logic.



- Requirements
  - Inputs
    - bin_i [3:0]
  - Outputs
    - hex_o [6:0]
    - hexn_o [6:0]
  - Translate binary number as set by "bin_i" to the correct sevenbit output "hex_o". Please refer to the figure in the appendix.
  - "hexn_o" is the negated version of "hex_o". (Would be useful for active low displays.)
  - Do not forget the numbers 10 (0xA) to 15 (0xF)!
- [2cp] Create a SystemVerilog test bench "tb_sevenseg.sv" in the ./sevenseg/sim folder
  - All input values are stimulated.
  - Input and outputs are displayed in the simulator console "transcript".
  - It allows to check the correctness of the implemented logic in the wave window of the simulator.

- [1cp] Create a TCL script file “sim_tb_sevenseg.tcl” that controls the compile and simulation process. A wave window is opened and all relevant signals are displayed.
- [1cp] Create a short summary report “doc_sevenseg.pdf” that shows the simulation result.
  - Show that your design fulfils the specification (verification).
  - Discuss why you chose your implementation method (advantages, disadvantages).

Appendix

The figure below shows the desired segment outputs for all numbers from 0x0 to 0xF.

In the “hex_o” output hex_o[0] refers to segment A and hex_o[6] to segment G.

The DP is not used.

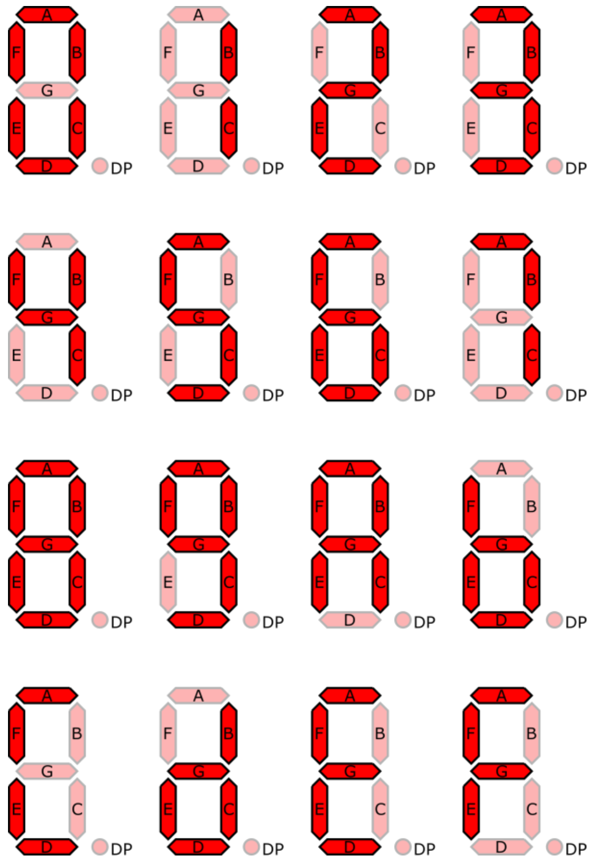


Figure 3: Number representation on a seven segment display.