FPGA LAB REPORT 6B

Hamza Shabbir, Hassan Ahmed & Neha Chaudhary

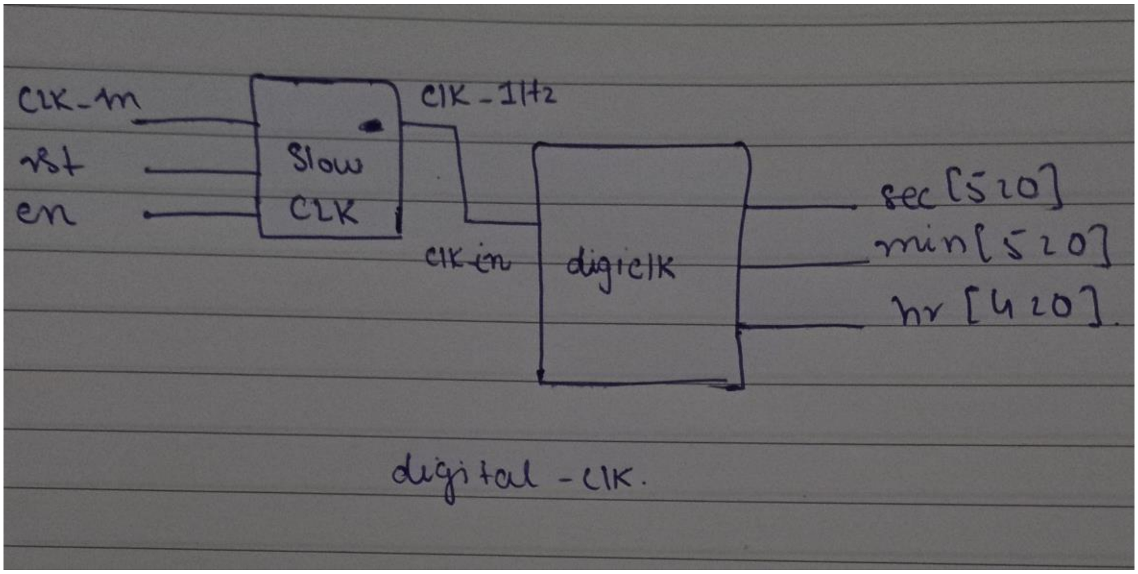
JULY 28, 2021

Implementation of Digital Watch on LEDs of Spartan 3e Starter Kit (Implementation)

6.1 Background

In the last lab we implemented the behavioral abstractions of digital watch using clock divider and counter instances with connections. In this second part of the lab we will implement it on the Spartan 3E FPGA Board by uploading the bit file onto the FPGA.

RTL Schematic Diagram:



6.4 Procedure

1. Designing of block diagram using clock divider and counter instances and mentioning the intermediate connections.
2. Implementation of behavioral abstractions of digital watch
3. Design a test bench in Verilog to check the outputs of digital watch.
4. Opening this project in ISE, synthesize the module and then do the implementation check.
5. Finally generate the BIT file to be uploaded on the FPGA.
6. Observe the output on LEDs.

6.5 Code and Simulations

As we are going to use just LEDs of the Sparten 3E the code is only for the seconds part of the digital watch.

Code:

```
module counter (clk, reset, enable, count_val,
over_flow); parameter Bits=4; parameter Max=9;
input clk, reset, enable;
output[Bits-1:0]
count_val; output
over_flow; reg[Bits-1:0]
count_val; reg
over_flow;

always@(posedge
clk) if(reset) begin
count_val<=0;
over_flow<=0; end
else if(enable)
if(count_val==Max-1)
begin count_val<=0;
over_flow<=1; end else
begin

count_val<=count_val
+1; over_flow<=0; end
endmodule

module digital_watch(clk, reset, enable, seconds);
input clk, reset, enable;
output[7:0] seconds;
wire[3:0]                    seconds_least,
seconds_most;  wire  w1,w2;  wire[7:0]
seconds;
//For sparten 3E we will use values 32 and 50_000_000.
counter #(2,2) C0(.clk(clk), .reset(reset), .enable(enable), .over_flow(w1), .count_val());
counter #(4,10) C1(.clk(clk), .reset(reset), .enable(w1), .count_val(seconds_least[3:0]),
.over_flow(w2));
```

```
counter #(4,6) C2(.clk(clk), .reset(reset), .enable(w2),
.count_val(seconds_most[3:0])); assign seconds={seconds_most,seconds_least};
endmodule
```

Test bench

```
module test_watch();
reg clk, reset, enable;
wire[7:0] seconds;
digital_watch D1(.clk(clk), .reset(reset), .enable(enable), .seconds(seconds));

initial begin
clk=1'b0;
reset=1'b0;
enable=1'b0;
#3 reset=1'b1;
#7 reset=1'b0;
#10
enable=1'b1;
#1000     $stop;
end
always #5 clk=~clk;
endmodule
```

Simulation Results:

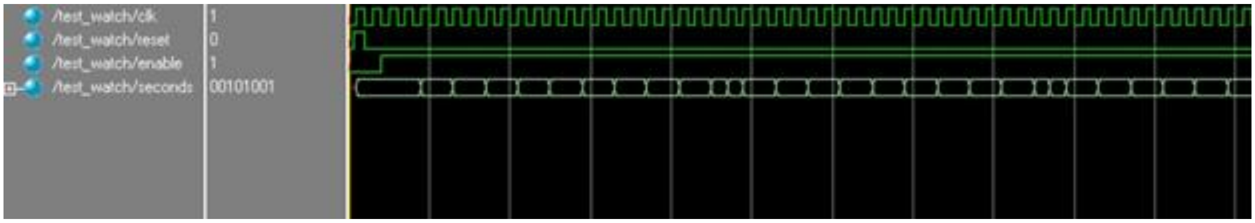
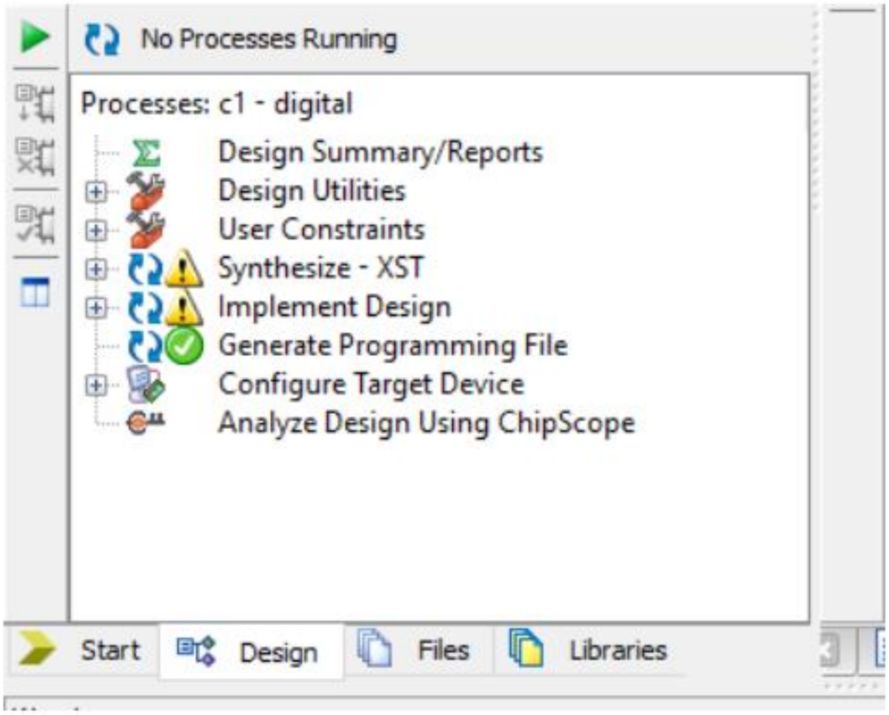


Figure 2: Simulation results of seconds part of digital clock

User Constraint File:

```
NET "clk" LOC = "C9" | IOSTANDARD = LVCMOS33 ;
#NET "clk" PERIOD = 20.0ns HIGH 50%;
NET "reset" LOC = "H13" | IOSTANDARD = LVTTL | PULLDOWN ;
NET "enable" LOC = "L13" | IOSTANDARD = LVTTL | PULLUP ;
NET "seconds<0>" LOC = "F12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "seconds<1>" LOC = "E12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "seconds<2>" LOC = "E11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "seconds<3>" LOC = "F11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "seconds<4>" LOC = "C11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "seconds<5>" LOC = "D11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
```

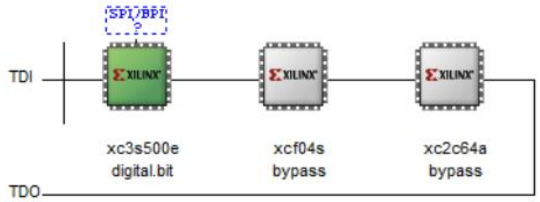
Synthesis and Generation of .bit file



No Processes Running

Processes: c1 - digital

Upload on FPGA Spartan 3E Kit:



Program Succeeded

Discussion and Conclusion:

Using the 50-megahertz clock we implemented the digital watch for which the clock needs to be divided such that each clock cycle shows one second. We divided that 50-megahertz clock into 1HZ clock using counter and if condition. On each positive edge of this 1HZ clock there is an increment of one second. After 59 seconds this second's overflow increments a minute and similarly after 59 minutes the minutes overflow increments the hour while 24 hours reset the hour cycle. .UCF file is written where the specific pins are mapped to the pin codes for each switch or an LED to make a connection. The bit file being generated is burned onto the board and the results are verified. On the FPGA Board.