CS203: Lab 4

Model Logic tile and switch box in Verilog

Objective: Step 1: designing configurable fabric
Language: Verilog
Points: 20
Release date: 12 Oct 2020
Deadline: 25 October 2020

Detailed problem statement:

1. Design 5 input logic tile as discussed in Lab project pdf and lecture on 12 October 2020.

   ```
   module logic_tile(out, clock, in1, in2, in3, in4, in5);
   ```

   out is the output port, clock, in1, in2, in3, in4, in5 are input ports. While configuring logic tile, assume in1 is the least significant and in5 is the most significant bit.

   Configuration for logic_tile should be stored in the following variable
   reg [32:0] mem;

   Mem[31:0] will be used by LUT and mem[32] will be used by MUX.

2. Design 4x4 switch box as discussed in lecture on 12 October 2020

   ```
   module switch_box_4x4(out, in);
   ```

   “out” is a 4 bit wide output port, “in” is a 4 bit wide input port. For both input and output, ports would be declared as [3:0]

   A 16 bit memory configures the switch box. In configuration data: first nibble (least significant) is for first output, second nibble for second output, and so, on. In each nibble, the least significant bit is for in[0] as shown in expression for out[0]. Each output can be described as the following expression:

   out[0] = configure[0] in[0] + configure[1] in[1] + configure[2] in[2] + configure[3] in[3]

   Configuration for switch box should be stored in the following variable
   reg [15:0] configure;

3. Create one test bench for both modules in testbench.v

Clock will be generated in the testbench module.
Give different possible stimulus values, print them on screen using $display. Dump the vcd file using $dumpvars and observe the output in waveform viewer.

4. Memory file format: three lines
    - First line: 32 bits in hex format. Least significant bit is index '0' This is corresponding to LUT
    - Second line : 1 bit (configure mux in in logic tile)
    - Third line: 16 bits in hex format (configure switch box)

For example: The following memory file will configure LUT as a non-latched 5 input OR gate, and switch-box where out0 is connected to in0, out1 is connected to in1, out2 connected to in2, and out3 connected to in3.

FFFFFFFE
0
8421

Logistics:

- Module definitions should be the same as specified in the problem statement. Configuration memory name in modules should also be the same as defined above.
- logic_tile and switch4x4 modules should be in a single file named **`lab4.v`**
- **`testbench.v`** will be a separate file containing testbench.
- Zip verilog files along with the memory file into "Lab4_<entry_number>.zip" and submit on moodle. Use only zip for compression, do not use rar, tgz, bgz or any other compression.
    - It may be a good idea to include your VCD file in submission for reference.

Evaluation:

- Lab will be evaluated automatically using scripts, so it is important to follow the above instructions and naming conventions.
- Use only the latest version of iverilog. Using any other verilog compiler can lead to compatibility issues.
- Course policy of late submission and plagiarism, as given on course webpage would apply.

The theory regarding the assignment is covered in the lecture on 12 October 2020.

FAQs

Designing a flip-flop in Verilog:

```
reg  q, qbar
```

```
always@(posedge clock)
begin
  q <= d;
  qbar = !d;
end
```

Generating a clock signal in testbench

```
reg clk;
initial clk = 0;
always #10 clk = ~clk;
```

Timing: how do we model timing in a behavior model?

Timing in this lab assignment is optional. It would be helpful in understanding, if you add delays in these basic modules.

Can I use a behavior model to model switch or logic tile?

Yes, it is optional. If you like, you may use.

Which one to use : $readmemh or $readmemb?

$readmemh can read one hex number per line, and $readmemb can read an array of bits. Number of bits in one line should match the number bits in the variable where these bits are stored.

In the slides, we use $readmemb, but here we suggest $readmemh, because it will be more manageable. Particularly, the same implementation can be used in the next lab assignment.

Where should we initialize the configuration: in the design file or in the testbench?

Ideally Logic tile and switch box are configured once, and they have separate input for configuration. Usually, one bit input is there in logical tile. For storing 17 bit of information takes 17 clock cycles to store the configuration.

However, in our case, to make things simple, we would be configuring logic tile and switch-box in the testbench without any input pin, but by hierarchical reference.

For example:
$readmemh(“lut.mem”, tb_mem); //tb_mem is defined in testbench
inst_logic_tile.mem = tb_mem;
//inst_logic_tile is instance of logic tile, and mem is a configuration array present in logic tile

What can be a good testing strategy?

You can have multiple configurations written in your memory file. Load these configurations one by one. For each configuration, you can see if your logic tile is functioning correctly or not!