Frequency Counter

Signature and Grading Sheet

Group #:________ Name(s):______________________________________ .

Grading

- Section 4.1(a): VHDL code (60 points):______.
  Attach code
- Section 4.2(b) RT-level simulation waveform (20 points):______.
  Attach simulation waveform screen captures.
- Section 4.3(e): post-synthesis simulation waveform (10 points): ______.
  Attach simulation waveform screen captures.
- VHDL code format and comments (10 points):______.

Total points: .

Experiment
Frequency Counter

1 Purpose

To design a frequency counter

2 Reading

- *FPGA Prototyping by VHDL Examples 2nd edition: Xilinx MicroBlaze MCS SoC.*

3 Project description

A frequency counter is a piece of equipment that measures the frequency of an input square wave. One design scheme is to count the number of transitions in the input signal within the 1-second period. An FPGA-based frequency counter can easily run around 100 MHz. However, this project slows down the rate to around 1K Hz to facilitate simulation (because of the limitation of the free version of ModelSim). Following are the assumptions and specifications:

- The system clock is 1 KHz.
- The frequency range of the input signal is between 1 Hz and 500 Hz.
- The measured frequency is 3 BCD digits.
- There is a control signal, `start`. When it is asserted, the frequency counter takes one measurement. The result stays on the output until the next measurement.
- The entire design must be synchronous (i.e., all memory elements driven by the same system clock signal).

The entity declaration of this design is

```
entity freq_counter is
   port(
      clk: in std_logic;
      start: in std_logic;
      fin: in std_logic;
      ready: out std_logic;
      d2, d1, d0: out std_logic_vector(3 downto 0)
    );
end freq_counter;
```

<u>The design must be synchronous or 50% will be deducted.</u>

4 Design Procedures

4.1 Design

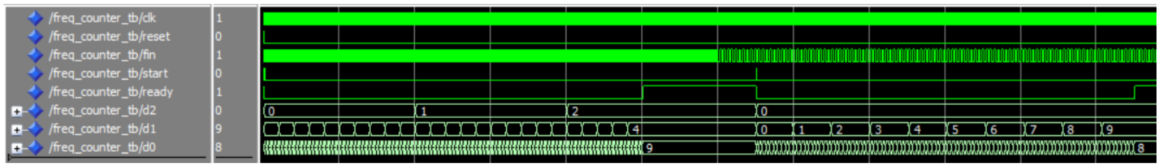
(a) Derive VHDL code

4.2 RT-level ModelSim simulation

(a) Use the testbench (freq_counter_tb.vhd) to simulate your VHDL code.

(b) Perform the screen captures. Use proper number formats (unsigned or binary) and layout.

(c) <u>To get full credits, the input and output signals should be properly arranged and represented in a proper format so that the simulation result can be easily understood</u>.



4.3 Post-synthesis ModelSim simulation

(a) Perform compiling (synthesis/placement and routing) and obtain the .vho file.

(b) Follow the format guideline to add a header to the file and verify the "structure" architecture body is generated.

(c) Revise the testbench to use the "structure" architecture for uut.

(d) Perform post-synthesis simulation

(e) Do screen capture of the simulated result.

(f) Since the waveform is the same as the RT-level simulation, the snapshot should include the expanded uut on the left panel. <u>No point will be given if the uut unit is not expanded in the snapshot.</u>