Matlab VHDL CoSimulation

using the Cosimulation Wizard

and the Matlab System Object

This document outlines the steps needed to cosimulate the VHDL component that computes the reciprocal square root function.

Entity Description:

```
entity rsqrt is
   port (
      clk              : in    std_logic;
      rsqrt_input      : in    std_logic_vector (31 downto 0);
      rsqrt_output     : out   std_logic_vector (31 downto 0)
   );
end rsqrt;
```

Signal Descriptions:

clk – the system clock input to the component.
rsqrt_input – input value (W=32, F=16). The input is assumed to be an unsigned positive fixed-point value (W=32, F=16) between [0 and 2^17].
rsqrt_output– output value y=1/sqrt(x) as a fixed fixed-point value.

Cosimulation Steps:

Files Needed:

rsqrt.vhd (your code to verify, and any dependent .vhdl files)
rsqrt_tb.m (example Matlab test bench, which is supplied, but you will likely need to modify it to match what you are explicitly doing in hardware).

1. Create a working directory, we will use in this example **\rsqrt_verification**
2. Place VHDL code to be tested (i.e. **rsqrt.vhd,** and any dependent .vhd files) into \rsqrt_verification
3. Change Matlab's working directory to \rsqrt_verification
4. At the Matlab command prompt, run the cosimulation wizard, i.e. >>**cosimWizard**
    a. In the ***Cosimulation_Type*** panel select:
        i. HDL cosimulation with: **Matlab System Object**
        ii. HDL simulator: **ModelSim**
        iii. Select one of the two options
            1. *Use HDL simulator executable on the system path* (this assumes that the path variable has already been setup).
            2. Otherwise select *Use the HDL simulator executables at the following location* and set the HDL simulator installation path (for computers in the digital lab) to: C:\modeltech64_10.1c\win64
        iv. Click Next

    b. In the **->HDL Files** panel:
        i. Add: rsqrt.vhd
        ii. Add: <all other VHDL files that are used by rsqrt.vhd>
        iii. Click Next

    c. In the **->HDL Compilation** panel:
        i. Click Next (i.e. accept the defaults)

    d. In the **->HDL Modules** panel:
        i. Name of HDL module to cosimulate with: **rsqrt**
        ii. Click Next

    e. In the **->Input/Output Ports** panel:
        i. Click Next (i.e. accept the defaults)

    f. In the **->Output Port Details** panel:
        i. Data Type: **Unsigned** (for lzc_count)
        ii. Fraction Length: **0** (for lzc_count)
        iii. Click Next

    g. In the **->Clock/Reset Details** panel:
        i. Active Edge: **Rising** (for clk)

        ii. Click Next

   h. In the **->Start Time Alignment** panel:
        i. HDL time to start cosimulation (ns): **0** (Note: if there is a reset, you will want to start after the reset has been de-asserted)
        ii. Click Next

   i. In the **->System Obj. Generation** panel:
        i. HDL Simulator sampling period (ns): **10** (Note: just keep this the same as the clock period, i.e. accept the default.)
        ii. Click Finish

5. Matlab creates a bunch of files in \lzc_verification and will open the following two files in the Matlab Editor:
   a. **launch_hdl_simulator_rsqrt.m**
   b. **hdlcosim_rsqrt.m**

6. Get the cosimulation ready to run by running the following command at the Matlab prompt: >> **launch_hdl_simulator_rsqrt**
   This will open up ModelSim and create a connection to it.

7. Run the cosimulation by running the function **rsqrt_tb.m** in the Matlab Editor. The **rsqrt_tb** requires the following lines:
   a. **`rsqrt_hdl = hdlcosim_rsqrt;`** This calls the function hldcosim_rsqrt.m that was created by the cosimWizard, which sets up the simulation object.
   b. **`output_vector1 = step(rsqrt_hdl,input_vector1);`** The step() function invokes the simulator and passes data to the component under test and receives data from the output. The arguments to the step() function (other than the system object, which is rsqrt_hdl in this case), must be fixed-point objects created by the fixed-point constructor fi(). The word length must match in the word length of the VHDL input signal. The outputs are fixed-point objects as well. If there are multiple I/O in the component, use: **[out1, out2, out3] = step(rsqrt_hdl, in1, in2, in3);**
   c. The rest of the rsqrt_tb function is just to create the appropriate input vectors and then test the outputs (with the appropriate clock cycle latency accounted for).