Project Status Report – RMC75E Test Bench

Date: July 21, 2023

Subject: Progress Update

Current Phase: Refining control output, DIO8 test units.

Challenges: Last week I mentioned that there was a problem with the Expansion module clock oscillating at an irregular interval. I managed to solve that problem with some very valuable input from David. It was related to the Slow Enable input signal, which needed to activate every 8th system clock tick. Previously I had the process high for 8 ticks, and low for 8 ticks. It should have been low for 7 ticks, high for 1 tick.

Fig. 1 – Erroneous Slow Enable Process

```
-- SlowEnable signal process definition
SlowEnable_process : process
begin
    SlowEnable <= '0';
    wait for 8 * SysClk_period;
    SlowEnable <= '1';
    wait for 8 * SysClk_period;
end process;
```

Fig. 2 – Corrected Slow Enable Process

```
-- SlowEnable signal process definition
SlowEnable_process : process
begin
    SlowEnable <= '0';
    wait for 7 * SysClk_period;
    SlowEnable <= '1';
    wait for SysClk_period;
end process;
```

Fig. 3 – Analog Module Data Output

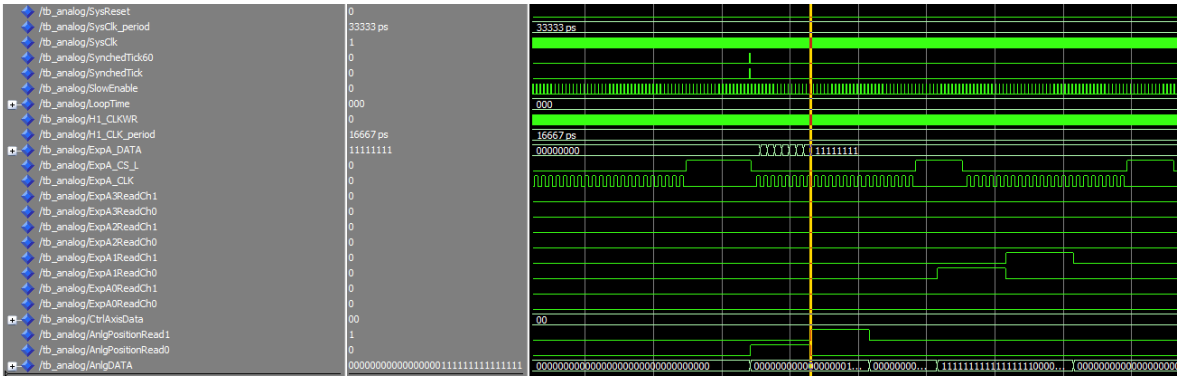

Fig. 4 – ADC Timing Diagram

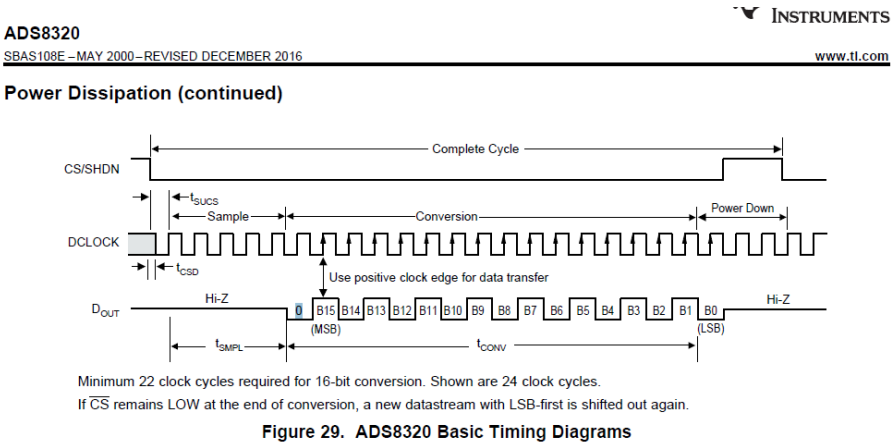

Figure 29. ADS8320 Basic Timing Diagrams

(We can see here that the clock behavior we are seeing in the analog module matches the expected behavior shown in the timing diagram for the ADS8320 Analog-to-Digital Converter.)

There were also some small changes I had to make to the source code to get certain signals to reach a defined state in the analog module:

- By uncommenting the following signal value initializations on line 75 of the databuffer module, I am able to write data to the AnalogData output line:
    - BankSelect, WriteEnable, WriteBank0, WriteBank1: std_logic:= '0';

- I was able to get the ‘o’ signal to properly define by uncommenting the initialization value for the RAM signal on line 89 of the RAM128x16bits module.
- Note that I also commented out the weird signal assignments for ExpA_CS_L and ExpA_CLK on line 121 of analog.vhd that were causing issues with the ExpA_CLK signal.

Work to date: As mentioned I was able to resolve the issue with the expansion module clock. Since then I have been working on refining the test benches for serial memory, I/O control, discrete I/O, and output control. Currently resolved an issue with the control output module where the data output line is only displaying the MSB of the input data, rather than the entire 16-bit vector as it should.

I have also been tinkering around with tcl scripts. Using these scripts has definitely improved my workflow, as they can help automate a lot of the debugging process I have to go through, such as running the tests, setting time constraints, or adding signal to the waveform viewer.

Fig. 5 – TCL Script 1

```
add wave -position insertpoint \
  -radix binary \
  /tb_analog/DUT/StateMach_1/State \
  /tb_analog/DUT/StateMach_1/SampleHoldState \
  /tb_analog/DUT/StateMach_1/ConvertState \
  /tb_analog/DUT/StateMach_1/CycleCounter \
  /tb_analog/DUT/StateMach_1/ExpA_CLK_EN \
  /tb_analog/DUT/StateMach_1/intExpA_CLK \
  /tb_analog/DUT/StateMach_1/StartState \
  /tb_analog/DUT/StateMach_1/intConverting \
  /tb_analog/DUT/StateMach_1/Converting \
  /tb_analog/DUT/StateMach_1/ResetCycleCounter \
```

Fig. 6 – TCL Script 2

```
# Load the libraries (change "work" to your
vlib work

# List of test bench files
set testbenches {
    tb_analog.vhd
    tb_CPUconfig.vhd
    tb_DIO8.vhd
    tb_DataBuffer.vhd
    tb_DiscoverExpansionID.vhd
    tb_ExpModuleLED.vhd
    tb_ExpSigRoute.vhd
    tb_MDTTopSimp.vhd
    tb_RtdExpIDLED.vhd
    tb_SSITop.vhd
    tb_Serial2Parallel.vhd
    tb_Serial2Parallel_v2.vhd
    tb_analog.vhd
    tb_controlio.vhd
    tb_controloutput.vhd
    tb_cpuled.vhd
    tb_decode.vhd
    tb_disccontID.vhd
    tb_discovercontrol.vhd
    tb_latencyCounter.vhd
    tb_list.txt
    tb_mdtssiroute.vhd
    tb_quad.vhd
    tb_ram128x16bits.vhd
    tb_serial_mem.vhd
    tb_statemachine.vhd
    tb_ticksync.vhd
    tb_top.vhd
    tb_watchdogtimer.vhd
}

# Load and run each test bench
foreach tb $testbenches {
    vsim work.[file rootname $tb]
    set StdArithNoWarnings 1
    set NumericStdNoWarnings 1
    run 100 us
    add wave -r /*
    quit -sim
}
```

Next Steps: Continue debugging the control output module, after that I will review the DIO8 module. At that point, test units for most if not all of the major modules should be in good shape.