Lecture #4: Potpourri

Paul Hartke
Phartke@stanford.edu
Stanford EE183
April 15, 2002

Tutorial/Verilog Questions?

- Tutorial is mostly done, right?
  - Due tonight at Midnight (Mon 4/14/02)
  - Turn in copies of all verilog, copy of "verification" scripts you wrote, corresponding waveforms annotated, FGPA Editor output (use PrtSc and copy to MS Paint), the part of the implementation output that shows the speed and the amount of logic units utilized.
  - Try to print side by side and duplex to save the forests.
- As usual, I'll be in the lab after lecture to answer questions.

Course Logistics

- Labs due every two weeks
  - Writeup due by following Monday at Midnight
    - Lab 1 is due Wed April 24th at Midnight
    - Lab 2 is due Wed May 8th at Midnight
    - Lab 3 is due Wed May 22nd at Midnight
    - Lab 4 is due Wed June 5th at Midnight
- Final Quiz 7-9pm TBD
  - Most likely week of May 20th or 27th

Lab 1 Questions?

- VGA
- TCGROM
- Sega Game Controller
- Two dual port memories
  - Why?
  - 4Kx1 architecture
- CoreGen

Lab 1: Optional Fun Things

- Display the number of iterations
- Capability to clear the screen
  - Instead of the cheesy (but perfectly fine) board reset
- Capability to start with a random game board
  - LFSR seeded by counter from powerup
- Fastforward
  - Have the next N game states be computed in rapid succession
    - Perhaps use a third BRAM

Lab 1: Known Interesting Initial States

- Some starting states are more interesting than others
  - Have the initialization of the BRAM be one of them.
  - Have multiple ones and switch between them
- Use Memutils.zip to generate the BRAMs init files.
  - http://groups.yahoo.com/group/xsboard-users/files/

Lab 1: Background Image

- When the game state location is off show a background image
  - Have another 64x64 BRAM storing the image and index it the same way as the game board. If the location is vacant then display whatever is in the background image.
  - we only have 10 4kbit BRAMs

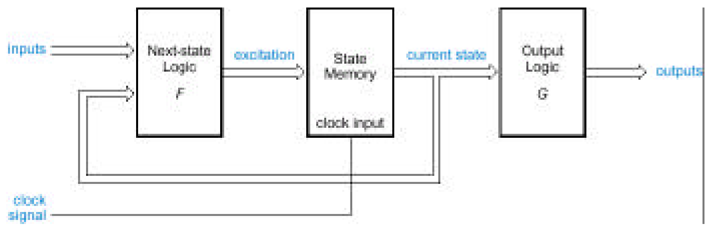
EE121 Topics (cont 2.)

- FSM Timing
  - Skew, jitter, H clock distribution tree
  - Max path, min path, critical path
- Metastability, latches and flops
  - Async Input Synchronizer Circuit
- Memory Architectures
  - ROM, SRAM, Dual Port SRAM, DRAM

We'll have a guest lecture on metastability

FSM Timing

- Now that we now how to design a state machine, how fast can we make it run?
- The register-to-register performance is the key metric to consider.

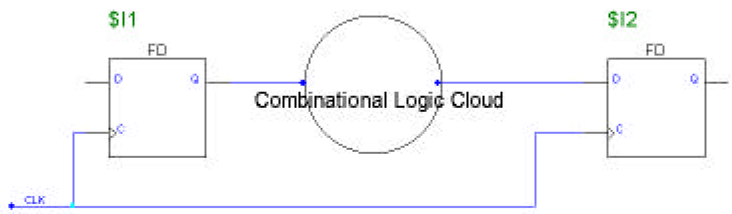


Clock Skew

- We have assumed that the clock reaches each DFF simultaneously.
  - It should be no surprise that this assumption is not entirely valid.
- Clock Skew is the non-time varying (static) difference in the clock arrival time at two different DFFs.

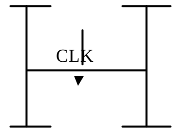
Clock Skew II

- The wire propagation delay is non trivial and the difference in arrival times for this type of layout is unacceptable.



H Clock Distribution Tree

- Make Clock distribution tree in the form of an H so that all flops are equidistant to the root of the tree.
- An FPGA is a very regular structure but for an ASIC, there are a variable number of DFFs in each sector.



Spartan II Skew Data

XILINX — Spartan-II 2.5V FPGA Family: DC and Switching Characteristics

Clock Distribution Guidelines[1]

Symbol	Description	Speed Grade -6 Max	Speed Grade -5 Max	Units
GCLK Clock Skew				
$T_{GSKEWIOB}$	Global clock skew between IOB flip-flops	0.13	0.14	ns

Notes:
1. These clock distribution delays are provided for guidance only. They reflect the delays encountered in a typical design under worst-case conditions. Precise values for a particular design are provided by the timing analyzer.

Clock Jitter

- Clock Jitter is the time-varying (cycle to cycle) difference in the clock arrival time at the ***same*** DFF.
- There are many sources of jitter—inaccuracies in the source oscillator, drifting of the Phase Lock Loop (PLL), and crosstalk between the clock and other transitioning signals.

Spartan II Jitter Data

DLL Clock Tolerance, Jitter, and Phase Information

All DLL output jitter and phase specifications were determined through statistical measurement at the package pins using a clock mirror configuration and matched drivers.

Figure 1, page 13, provides definitions for various parameters in the table below.

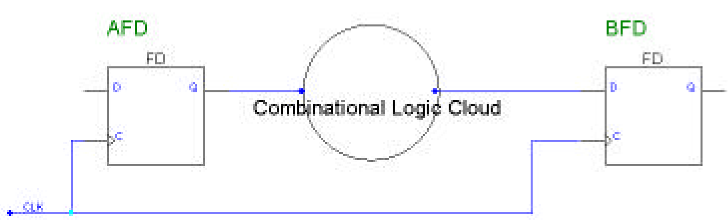
Symbol	Description	F_{CLKIN}	CLKDLLHF Min	CLKDLLHF Max	CLKDLL Min	CLKDLL Max	Units
T_{IPTOL}	Input clock period tolerance		-	1.0	-	1.0	ns
T_{IJITCC}	Input clock jitter tolerance (cycle-to-cycle)		-	±150	-	±300	ps
T_{LOCK}	Time required for DLL to acquire lock	> 60 MHz	-	20	-	20	μs
		50-60 MHz	-	-	-	25	μs
		40-50 MHz	-	-	-	50	μs
		30-40 MHz	-	-	-	90	μs
		25-30 MHz	-	-	-	120	μs
T_{OJITCC}	Output jitter (cycle-to-cycle) for any DLL clock output[1]		-	±60	-	±60	ps
T_{PHIO}	Phase offset between CLKIN and CLKO[2]		-	±100	-	±100	ps
T_{PHOO}	Phase offset between clock outputs on the DLL[3]		-	±140	-	±140	ps
T_{PHIOM}	Maximum phase difference between CLKIN and CLKO[4]		-	±160	-	±160	ps
T_{PHOOM}	Maximum phase difference between clock outputs on the DLL[5]		-	±200	-	±200	ps

Notes:

1. **Output Jitter** is cycle-to-cycle jitter measured on the DLL output clock, excluding input clock jitter.
2. **Phase Offset between CLKIN and CLKO** is the worst-case fixed time difference between rising edges of CLKIN and CLKO, excluding output jitter and input clock jitter.
3. **Phase Offset between Clock Outputs on the DLL** is the worst-case fixed time difference between rising edges of any two DLL outputs, excluding Output Jitter and input clock jitter.
4. **Maximum Phase Difference between CLKIN an CLKO** is the sum of Output Jitter and Phase Offset between CLKIN and CLKO, or the greatest difference between CLKIN and CLKO rising edges due to DLL alone (excluding input clock jitter).
5. **Maximum Phase Difference between Clock Outputs on the DLL** is the sum of Output Jitter and Phase Offset between any DLL clock outputs, or the greatest difference between any two DLL output rising edges due to DLL alone (excluding input clock jitter).

Example Parameters

- DFF values:
  - $T_{clk\text{->}q}$=1ns, T_{setup}=1ns, T_{hold}=1ns
- Clock skew is max 2ns and jitter is 2ns
- Combinational logic T_{cl_pdmax}=10ns, T_{cl_pdmin}=1ns



MaxPath Timing Constraint

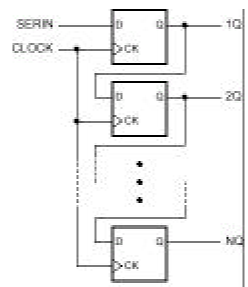
- Add up the components that result in the time budget; the period must be greater than this value.
- $T_{clk\text{->}q} + T_{cl\_pdmax} + T_{setup} + T_{skew} <=$ Clock Period
- 1 + 10 + 1 + 2 <= Clock Period
- 14ns <= Clock Period
- Max Frequency is 71MHz

MinPath Timing Constraint

- Consider what happens when the same clock edge is considered at the far DFF.
- $T_{clk\text{->}q} + T_{cl\_pdmin} >= T_{skew} + T_{hold}$
- 1 + 1 >= 2 + 1
- Whoops!! ☹
- AKA, “Hold-Time Violation”

MinPath and ShiftRegisters

- Shift Registers can easily fall prey to min path timing violations.
- Fix the violations by increasing delay between Ds and Qs
  - Insert pairs of inverters
- FPGA DFF clk->q is big enough so that MinPath violations are rare.



Impacts

- You can “fix” MaxPath timing constraint violations by slowing down the clock after the circuit is implemented.
- You ***cannot*** “fix” MinPath timing constraint violations be modifying the clock.

Static Timing Tool

- Longest MaxPath Constraint is called ***Critical Path*** of design.
  - Find critical path by calculating all the MaxPath constraints of ever every path in the design and picking the largest.
- Perfect tool for a computer.
  - Xilinx Timing Analyzer is an example of a static timing tool.

Timing Closure Challenges

- When integrate individual blocks that meet timing, the combined system might not meet timing.
- In general have registered outputs from ***top-level*** blocks.
  - This doesn't solve the problem if the chip is so large/fast that a signal cannot propagate all the way across the chip.
  - Reason that I/Os are always useful to register
    - Not always certain timing budget available on the board.