FPGA Design
EECE 277

Documentation for Project Design

Dr. William H. Robinson
March 28, 2005

http://eecs.vanderbilt.edu/courses/eece277/

Topics

“…Write the vision, and make it plain upon tables, that he may run that readeth it.”

– Habakkuk 2:2

• Administrative stuff
  – Homework Assignment #4 (still grading)
  – Proposals for final project (due Monday, April 4)
  – Exam 2 (Wednesday, April 6)
  – Laboratory Assignment #3 (due TODAY)

• Using RAM

• Design documentation

• Design process

Library of Parameterized Modules

• Altera LPM library has many elements useful for building common datapath functions
  – lpm_ram_dq - recommended for either asynchronous or synchronous RAM. Uses EAB in Flex 10K family.
  – lpm_ram_io - recommended for synchronous RAM. Uses EAB in Flex 10K in family.
  – lpm_compare - comparing two values. Outputs are AEQB, ALB, ALEB, AGB, AGBE
  – lpm_counter - versatile counter function

Laboratory Assignment #3

• The Good?

• The Bad?

• The Ugly?
Asynchronous vs. Synchronous RAM

- Asynchronous RAM looks like a combinational element
  - No Clock
  - Data available after propagation delay from address
  - Address MUST BE stable while WE (write enable) is high so that only ONE location is written to. Data must also be stable during write cycle.

- Synchronous RAM has a clock input and will latch input data and control lines (address, data)

Preference: Synchronous RAM

- Sync RAM is easier to use from a timing perspective but adds latency to operations
  - If address coming from counter, then have an extra clock cycle of latency from when counter value is updated to when RAM data is available for that address

- Normally use synchronous RAM
  - Will not latch output data unless specifically needed
  - Options to latch control, input data, output data available on LPM_RAM_DQ

LPM_RAM_DQ Timing (1)

Asynchronous – no latching
LPM_Address_Control = "Unregistered"
LPM_indata = "Unregistered"
LPM_Outdata = "Unregistered"
Assume 16x4 RAM
Contents: loc S1 = 3, S0 = 5, S1 = S8, S2 = S1, S3 = S8, S4 = SB

LPM_RAM_DQ Timing (2)

Synchronous, latch input data and control
LPM_Address_Control = "Registered" (on 'inclock')
LPM_indata = "Registered" (on 'inclock')
LPM_Outdata = "Unregistered"
Assume 16x4 RAM
Contents: loc S1 = 3, S0 = 5, S1 = S8, S2 = S1, S3 = S8, S4 = SB
Asynchronous vs. Synchronous Control

- Some LPMS have both synchronous and asynchronous control lines
  - Counter has ‘aload’ (asynchronous load), and ‘sload’ (synchronous load); ‘aclr’ and ‘sclr’ (async and sync clear)

- Should always use a Synchronous control line if possible, especially if connected to a FSM output
  - Any glitch on an asynchronous control line can trigger it
  - If using a FSM output for an asynchronous control, the output should come directly from a Flip-Flop output, NOT from combinational gating

Ideas for Final Project

http://microsys6.engr.utk.edu/ece/bouldin_courses/551/candidates.html

- 01. Differential Scoreboard
- 02. Elevator Controller
- 03. Encryption/Compression
- 04. Decompression/Decryption
- 05. Hamming Code Transceiver
- 06. Digital Frequency Meter
- 07. Blackjack Game
- 08. Music Generator with 4-Part Harmony
- 09. Race Track Timer
- 10. Tic-Tac-Toe Game
- 11. String Comparator
- 12. Simon Game
- 13. Other (submit proposal for approval)

Final proposals due: Monday, April 4, 2005
Format will be given this week

Design Methodologies

- Every company has its own design methodology

- Methodology depends on:
  - Size of chip
  - Design time constraints
  - Cost/performance
  - Available tools
Design Teams

- Almost all interesting projects are too big for one person to handle
- Need a team of people with varying skills
- Who is in charge?

Documents

- Documents are critical
  - Writing it helps you decide what to do
  - Provides information for maintenance, next generation
- Each document serves as the contract between the provider of the document and the consumer of the document

Major Documents

- Requirements
- Specification
- Architecture
- Module designs
- Manuals
  - Reference, user

Types of Information

- Functional description
- Non-functional description
  - Cycle time, power, etc.
- Timetables
- Design verification methods
- Quality metrics
- Job assignments
Starting the Project

- **The requirements document**
  - English description of what is to be done
  - Customer-oriented
  - High-level

- **May be written by marketing**

- **Author of requirements should verify that the requirements are accurate**

Specifications

- **The specification is the contract between marketing and the design team**

- **The specification is more technical than the requirements:**
  - Delays, etc.

- **An ideal specification would contain no architectural information**
  - That goal may be hard to achieve in practice
  - The specification says what to do, not how to do it

Specification and Planning

- **Driven by contradictory impulses**
  - Customer-centric concerns about cost, performance, etc.
  - Forecasts of feasibility of cost and performance

- **Features, performance, power, etc. may be negotiated at early stages**
  - Negotiation at later stages creates problems

Architecture

- **The architecture document**
  - Contract between the system designers and the component designers

- **Specifies major subsystems and their interactions**

- **Makes important design decisions**

- **Isn’t a full implementation**
Module Designs

- Specifies details of a module
  - Functionality
  - Non-functional parameters
  - Design verification

Design Reviews

- Have other designers (team + non-team) evaluate a design
  - Relatively simple
  - Proven to work

- Must walk through the design in detail to look for problems, improvements

Do Documents Reflect the Product?

- In a word, no.
  - Things change
  - People don’t have time to conform documents to the final design

- Some amount of updating is important for maintenance, future generations

Generic Design Flow

- Detailed specs
- architectural simulation
- Timing/area budget
- register-transfer design
- logic design
- Final design verification
- physical design
  - configuration
Estimation and Planning

- Estimation techniques vary with module
  - Memories may be generated once size is known
  - Data paths may be estimated from previous design
  - Controllers are hard to estimate without details

- Estimates must include speed, area, power

Floorplanning and Budgeting

- Want some early physical design information: area, delay, power, etc.

- Ways to get info
  - Previous designs
  - Quick design runs

Architecture

- Need to build an executable model of the architecture
  - Run vectors on architecture
  - Use as golden design for comparison with later stages

- Modeling languages
  - C: easier to write, less detailed
  - VHDL: harder to write, synthesizable with effort

Logic Design

- For controllers, good state assignment is usually requires CAD tools

- Logic synthesis is an option
  - Very good for non-critical logic
  - Can work well for speed-critical logic

- Logic synthesis system may be sensitive to changes in the input specification
**Place and Route**

- Most computationally expensive stage
- Metrics take more time to judge than functional vectors
- Deciding how to fix a problem may take effort
  – How to change placement, etc.

**Design Verification**

- Functional verification
  – Runs reasonable set of vectors
- Non-functional verification
  – Performance
  – Power

**Functional Verification**

- At all levels of hierarchy: module, subsystem, system
- At every level of abstraction
  – Compare to previous level of abstraction, golden model
- Must check interfaces
  – Half of bugs are at the interface to other modules

**Functional Verification Input**

- Sources of vectors
  – Previous designs
  – Vectors from higher levels of abstraction
  – Vectors designed precisely for this stage
  – Inputs from other modules
Non-Functional Verification

- **Performance**
  - Static timing analysis

- **Power**
  - Some information from timing analysis
  - Power analysis tools

Static Timing Analysis

- After your gate netlist has been mapped to the FPGA, a timing analysis tool will analyze the paths in the design and compute the timings

- The timing analyzer takes into account the routing delays in the physical routing and the speed grade of the part you have mapped to

- Because routing can sometimes change somewhat drastically for even small changes, often run multiple device mappings to try to get a ‘good route’

Static Timing Analysis Reports

- The static timing analyzer will report the following times
  - Register to Register delays
  - Setup times of all external synchronous inputs
  - Clock to Output delays
  - Pin to Pin combinational delays

- The clock to output delay is usually just reported as simply another pin-to-pin combinational delay

- Timing analysis reports are often pessimistic since they use worst case conditions

Timing Simulation

- The timings extracted by the timing analysis tool (routing delays, gate delays for a particular speed grade, etc) are used in the simulation

- It may be tempting to simply ignore the delays reported by the timing analyzer, and simply simulate the design ‘at speed’ to see if it works
  - If the design simulates correctly, only means that it works for the particular test vectors that you used!
  - Different test vectors exercise different delay paths – you must use test vectors that exercise the LONGEST paths
Breadboards

• May build a board to test an FPGA-based design
  – Takes some time
  – May allow running the design against the real I/O device

Summary

• Documentation is key for overall design and maintenance of a project
• Must capture all relevant information
• Ambiguities leads to multiple interpretations which can affect the design