Review Stuff – 2nd Exam

- General information
  - Tri state stuff
  - Basic 4-event protocol stuff
- Information representation
  - 2C, UB, Fixed Point, Excess Codes
  - Floating point

Steps for Combinational Design:

Understand the {requirements, algorithms, representations, solutions}.
Create a basic block diagram.
Partition the problem as needed.
Uniquely define the outputs of each section in terms of the inputs of that section.
Reduce the truth table or equations to an implementable form.
Create a logic system to do the work.
(Use hierarchical design where appropriate)
Debug as necessary.
(For VHDL version – make testbench to examine input/output relationships of interest)

Review Topics

- Clocked sequential design
  - System functionality
    - Logic determines next step
    - Logic decodes PSR for work
    - Mealy/Moore machine differences
  - Design of hardware to do work
    - State diagram creation
    - Implementation techniques (classical, one-hot, VHDL case)
  - Limitations and speeds

Review Topics

- How to do math
  - Sequential activities: multiply, divide
  - Floating point math (add/subtract, multiply)
  - Handling the extra bits
  - High speed vs ‘normal’ implementations
    - For multiply – use of RRUs to reduce time
    - For divide – approximation method

Steps for Sequential Design:

Understand the processing requirements.
Understand the processing algorithms.
Create a conceptual level data path block diagram.
Create a detailed data path block diagram.
Create a state diagram describing the sequential nature of the system.
Design the logic of the control section.
Create the logic design for the entire system, implement and debug.

Review Topics

- Asynchronous sequential systems
  - Creation of state diagram
  - Implementation techniques
  - Reason for covering all minterms
  - Steps for synthesis/analysis

Asynchronous Design Steps

Remember that analysis will reverse these steps

- Describe problem - state desired behavior. It is imperative that you understand the desired behavior in the environment in which the system will function
- Create a primitive state diagram of the desired behavior. This diagram will identify the desired output sequences for a specific input sequence. Remember that input events reflect history; two states may have identical in/out behavior, but not be identical since one reflects different history from another.

- Make up primitive state table from primitive state diagram
- Create a merge diagram
- Create a new state table from merged rows of primitive state table.
- Create an excitation map from the new state table.
- Create logic for the expressions of the excitation variables.
- Design - equal gate delays. Be sure to handle:
  - hazards
  - race conditions