Mid-Term Review

Introduction to Logic Design with VHDL

This chapter reviews the design of combinational and sequential logic and introduces the use of VHDL to design combinational logic and state machines.

After this chapter you should be able to:

- convert an informal description of the behaviour of a combinational logic circuit into a truth table, a sum of products boolean equation and a schematic,
- convert an informal description of a combinational logic circuit into a VHDL entity and architecture,
- design a state machine from an informal description of its operation, and
- write a VHDL description of a state machine.

The 80386SX Processor Bus and Real-Mode Instruction Set

This chapter describes the signals and operation of the Intel 80386SX processor bus and describes a *subset* of the 80x86 architecture and instruction set.

After this lecture you should be able to draw diagrams for the processor bus signals described in this lecture and state the values that would appear on the data and address buses during memory and I/O read and write cycles and for interrupt acknowledge cycles.

You should know enough about the 80x86 instruction set to be able to write simple routines to read/write data to/from I/O ports. In particular, you should be able to:

- write a real-mode 8086 assembly language program including: (1) transfer of 8 and 16-bit data between registers and memory using register, immediate, direct, and register indirect addressing, (2) some essential arithmetic and logic instructions on byte and 16-bit values, (3) stack push/pop, (4) input/output, (5) conditional and unconditional branches, (6) call/return, (7) interrupt/return, (8) essential pseudo-ops (org, db, dw).
- compute a physical address from segment and offset values,
- describe response of the 8086 CPU to software (INT) and external (NMI, IRQ) interrupts and return from interrupts.

RTL Design with VHDL

This chapter covers some features of VHDL that are useful for logic synthesis. You should learn to:

- make library packages visible
- declare components in architectures and packages
- declare constants
- instantiate components into an architecture
- declare std_logic, std_logic_vector, signed and unsigned signals
- declare enumerated types and subtypes of array types in architectures and packages
- declare and use entities with generics
- use conditional signal assignments
- convert between std_logic_vector, unsigned and integer types
- instantiate tri-state outputs
- create RAM and ROM memories

We will also learn an approach to logic design called Register Transfer Level (RTL) or "dataflow" design. This is the method currently used for the design of complex logic circuits such as microprocessors.

You should be able to:

- classify a VHDL description as a behavioral, structural, or dataflow (RTL) description
- identify the registers and logic/arithmetic functions required to implement a particular algorithm
- partition this algorithm into a sequence of these operations and register transfers
- write synthesizeable VHDL RTL code to implement the algorithm

We also covered three topics related to the design of interfaces to logic circuits: metastability, input synchronization and glitches. You should be able to: identify circuits where metastable behaviour is possible; compute the mean time between metastable outputs; identify circuits that could fail due to asynchronous inputs; add synchronizer flip-flops to reduce the probability of metastability; remove race conditions by registering inputs; and use registered outputs to eliminate glitches.

Reference Materials

The exam is open-book. You should bring your lecture notes, the 80x86 assembly language summary and the VHDL language reference sheets. All of these are available on the course web page.