Mid-Term Exam Review

This lecture summarizes the tasks you might be asked to do on the mid-term exam and provides examples of possible questions.

1 Combinational Logic Design

You should be able to convert an informal (text) description of a combinational logic circuit into three forms: (1) a truth table, (2) a sum of products boolean equation, and (3) a VHDL entity/architecture description.

Your VHDL code should be limited to the synthesis subset described in the lecture notes. Marks will not be deducted for *syntax* errors unless the description is ambiguous or unclear. Marks *will* be deducted for code that is not synthesizeable or is incorrect.

Exercise: Design a circuit with a 3-bit input and a 1-bit output. The output should be high if the input, interpreted as a binary number, is a multiple of 3. Describe the circuit in each of the three formats listed above.

2 Sequential Logic Design

You should be able to: (1) design a state machine from an informal description of its operation, and (2) write a VHDL description of the state machine.

You should be able to design the state machine in a methodical fashion by enumerating the states and transition conditions in the form of a table.

Exercise: Design a device with one input (ALE), 4 outputs (T1 to T4) and a clock. The output should only change on the rising edge of the clock. Exactly one of T1 through T4 should be asserted at any time. If ALE is asserted, T1 should be asserted; otherwise the next higher 'T' output is asserted except that if the output is already at T4 it stays at that value.

Give a tabular description of this state machine showing all possible combinations of current state and input, and the corresponding outputs and the next states. Write a synthesizeable VHDL description of such a state machine.

3 The 8088 Architecture

You should be able to:

- write a simple program in 8088 assembly language 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) [un]conditional 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 8088 to NMI, software (INT) and external (IRQ) interrupts and return from interrupts.

As with VHDL, marks will not be deducted for syntax error if the meaning is unambiguous (e.g. using the opcode 'MOVE' instead of 'MOV').

Exercise: Write an 8088 assembly language program to input a byte from input port 3FH and add to it the value of the byte stored in memory location 20100H. If the result is non-zero a byte of value 01H should be stored to memory location 20300H, otherwise the value 00H should be stored there. Your code must initialize all registers, including segment registers.

Exercise: The following table shows the contents of the first 64 bytes of an 8088 microcomputer system's memory. The instruction "INT 05H", located at address 01000 and is executed. Before the instruction is executed the value of SS is 1000 and the value of SP is 200. What is the address of the next instruction that is executed? What values are written to memory as a result of the INT instruction (give the values of each byte written and the physical address to which it is written). (Hint: the INT 05H instruction is encoded as 2 bytes).

4 The Intel 8088 Processor Bus

You should be able to state the values of the processor address, data and the main control bus signals during memory or I/O read and write cycles.

Exercise: During a bus cycle an 8088 CPU reads the value 88H from memory location 12300H. Give the state (high, low, tri-state or indeterminate) of the data bus signals at the falling edge of the CPU clock that starts each of the bus states T1 through T4.

5 Programmable Logic Devices

You should be able to select and justify the choice of a programmable logic device based on: (1) the complexity of the design; (2) maximum allowed propagation delay; (3) device cost; and (4) available development tools.

You should also be familiar with basic PLD terminology including the terms PAL, CPLD, FPGA, OTP, sum of products, CLB, LUT, netlist, and place-and-route.

Exercise: What type of PLD would you most likely use to implement an address decoder that generates chip selects for two 64kB banks from a 20-bit address?

Exercise: Is a RAM OTP? Does the design of a PAL involve place-and-route? Can Design Compiler generate a netlist?

6 Timing Analysis

You should be able to derive the expressions for a chip's timing requirements from the timing diagram and compute the margin for each requirement based on clock periods and the guaranteed responses of the other components.

Exercise: You will be given a schematic plus timing diagrams and timing specifications for a CPU and a memory or I/O device. You will be asked to compute the margins for several device timing requirements. The CPU and memory devices may be new to you.

7 Memory Devices

You should be able to select the appropriate type of memory device for different applications, combine memory ICs to form memory arrays, and design address decoders using SSI decoders and PLDs.

Exercise: You would like to design a microcomputer system with 64 kB of static RAM. The CPU you will use has a 32-bit data bus and a 32-bit address bus. You need to use 16 kByte RAM chips. The CPU uses byte addressing and the address bus pins available on the chip are labeled A2 to A31.

How many RAM chips will you need? If the decoder fully decodes the address bus, how many inputs will it have? How many chip select outputs will it have? Draw a diagram showing the address and data bus connections between the CPU and the RAM chips. Use the bus notation described in Assignment 5. You need not show other control lines.

8 Interrupts

You should be able to: decide and explain why interrupts should (or should not) be used to service a particular peripheral, describe how the 8259 PIC is connected to handle multiple interrupt sources, and write 8088 assembly language code to initialize and service interrupts generated by the 8259 PIC.

Exercise: An ISR for the IBM PC's keyboard interrupt fails to issue an EOI command to the 8259 PIC. What will be the value of the PSW interrupt enable bit after the ISR terminates? Will the timer ISR continue to be executed? Will the user be able to use the keyboard?