ENGR-433 Midterm notes

The emphasis will be on state machine design (sequential logic) with some combinational logic design. A little introductory VHDL will appear on this exam.

Material that could appear in some way on the exam.

Non-VHDL topics (significant emphasis on this):

- typical combinational logic functions (multiplexers, decoders, etc)
- state machine models - moore and mealy outputs
- state diagram use & terminology
- flip flops - D, T
  - know the D state excitation (i.e. transition) tables
  - know the state diagram of a D F/F
- edge triggered flip flops vs "pulse triggered" latches; set-up time; hold time
- counters, binary and decade
- clocking of flip-flops and counters
- synchronous logic design
  - be able to do it:
    - create state diagrams with correct branching conditions & output signals
    - create next state forming logic
    - create output forming logic
    - analyze a state diagram for adherence to branching rules
  - be able to design next state logic using either
    - binary encoded state numbering
    - one-hot encoded state numbering
- be able to create or read a timing diagram

VHDL topics:

- Basic VHDL grammar and syntax required. I.e., know how to:
  - create an entity with its port statement
  - create an architecture
  - define a signal with std_logic or std_logic_vector data type
  - do a signal assignment
  - AND two signals, OR two signals, complement a signal

I will not ask questions about VHDL sequential statements (i.e. in a process) or sequential logic (state machines defined using VHDL).

Various state machine problems picked from several old exams follow.

1) Design a synchronous FSM (finite state machine) that will create an output according to the following requirements:

- There will be four inputs to the state machine: S, X, Y and clk.
- There will be one output: Z
- If the input S is active (asserted) simultaneous with X and Y being inactive (not-asserted), then output Z will be asserted if a sequence of inputs occur: (Y following XY) or (X following XY), provided that these events (combinations of inputs) are spaced one clock period apart. If either of these sequences does not occur, then Z will not be asserted and the FSM must wait for S to be sampled inactive before the FSM can return to the initial state and start the process over again. When Z is asserted, it must be asserted for only one clock period, after which the FSM will return to the initial state.
- The FSM must initialize to state zero.

a) Construct a state diagram
b) Give it a glitch-free (output) state assignment
c) Draw a <u>block</u> diagram of your FSM. (do not design the next-state or output logic)

(Hint: properly done, 6 states are required.)

2) Here is an operation table for a flip-flop (possibly a bit silly) which is to be edge triggered. Create a state diagram and design the circuit for this. Memory element choice is yours, but please state your rational for the choice made.

A	B	Q_{t+1}
0	0	0
0	1	1
1	0	1
1	1	Q_t

3) Find any static hazards in this function. If hazards are found, show hazard cover. Also, if hazards are found, will the hazard occur on a low-to-high or high-to-low transistion?:

$$Q = AB\underline{X} + BY + \underline{A}C\underline{X} + BCXY$$

Note: complemented variables here are underlined, i.e. $\underline{X}$ is X bar.

4) Assume you need an edge triggered Set-Reset flop-flop but only have a RET D-F/F. Create a RET Set-Reset F/F. Show your work.

a) Draw a fully documented state diagram
b) Create next state and output logic
c) Draw a logic diagram for your design.