

ECE 474A/574A Exam1 Cheat Sheet

Boolean Algebra			
Commutative $a + b = b + a$ $a * b = b * a$	Distributive $a * (b + c) = a*b + a*c$ $a + (b * c) = (a + b) * (a + c)$	Associative $(a + b) + c = a + (b + c)$ $(a * b) * c = a * (b * c)$	Identity $0 + a = a + 0 = a$ $1 * a = a * 1 = a$
Complement $a + a' = 1$ $a * a' = 0$	Null Elements $a + 1 = 1$ $a * 0 = 0$	Idempotent Law $a + a = a$ $a * a = a$	Involution $(a')' = a$

** you should know DeMorgan's Law

Combinational Logic Design Process

Step 1	Capture the function	Create a truth table or equations, <i>whichever is most natural for the given problem</i> , to describe the desired behavior of the combinational logic.
Step 2	Convert to equations	This step is only necessary if you captured the function using a truth table instead of equations. Create an equation for each output by ORing all the minterms for that output. Simplify the equations if desired.
Step 3	Implement as a gate-based circuit	For each output, create a circuit corresponding to the output's equation. (Sharing gates among multiple outputs is OK optionally.)

Controller Design Process

Step 1	Capture the FSM	Create an FSM that describes the desired behavior of the controller
Step 2	Create the architecture	Create the standard architecture by using a state register of appropriate width, and combinational logic with inputs being the state register bits and the FSM inputs and outputs being the next state bits and the FSM outputs.
Step 3	Encode the states	Assign a unique binary number to each state. Each binary number representing a state is known as an encoding. Any encoding will do as long as each state has a unique encoding.
Step 4	Create the state table	Create a truth table for the combinational logic such that the logic will generate the correct FSM outputs and next state signals. Ordering the inputs with state bits first makes this truth table describe the state behavior, so the table is a state table.
Step 5	Implement the combinational logic	Implement the combinational logic using any method.

RTL Design Method

Step 1	Capture a high-level state machine	Describe the system's desired behavior as a high-level state machine. The state machine consists of states and transitions. The state machine is "high-level" because the transition conditions and the state actions are more than just Boolean operations on bit inputs and outputs.
Step 2	Create a datapath	Create a datapath to carry out the data operations of the high-level state machine.
Step 3	Connect the datapath to a controller	Connect the datapath to a controller block. Connect external Boolean inputs and outputs to the controller block.
Step 4	Drive the controller's FSM	Convert the high-level state machine to a finite-state machine (FSM) for the controller, by replacing data operations with setting and reading of control signals to and from the datapath.