Table 1. Learning objectives (84 discrete modules) for the EECS 140/141 flipped classroom model.

Module Number	Module Description	Module Number	Module Description
1	Give the truth table for the AND and OR functions.	43	Design a carry-lookahead adder.
2	Give the truth table for the NOT of a logic function.	44	Design a hierarchical carry-lookahead adder.
3	Create a truth table for a logic function.	45	Design an array multiplier for unsigned binary numbers.
4	Draw the logic network of gates that implements a logic function.	46	Multiply signed binary numbers with 2's compliment arithmetic.
5	Use Boolean Algebra to reduce a logic function.	47	Convert a fixed-point binary number to decimal.
6	Prove a Boolean identity with a Venn Diagram.	48	Give the decimal exponent range and precision of a single- or double-precision IEEE floating point number.
7	Give the canonical sum-of-products (SOP) for a logic function.	49	Design an n-to-1 multiplexer.
8	Five the canonical product-of-sums (POS) to implement a logic function.	50	Design a switch with a multiplexer.
9	Determine whether to use a canonical sum-of-products (SOP) or product-of-sums (POS) to implement a logic function.	51	Synthesize a logic function with a multiplexer.
10	Give the truth table for the NAND and NOR functions.	52	Factor a single variable out of a Boolean expression with Shannon's Expansion Theorem.
11	Draw the logic network of only NAND gates that implements a logic function.	53	Apply Shannon's Expansion Theorem to a single variable of a logic function to implement it with a multiplexer.
12	Draw the logic network of only NOR gates that implements a logic function.	54	Apply Shannon's Expansion Theorem to multiple variables of a logic function to implement it with a multiplexer.
13	Give the truth table for the XOR function.	55	Design a binary decoder.
14	Give the differences between a positive logic and a negative logic implementation using voltage levels.	56	Design a binary encoder.
15	Know which voltage (high or low) applied to a NMOS transistor opens or closes the transistor switch.	57	Design a priority encoder.
16	Know which voltage (high or low) applied to a PMOS transistor opens or closes the transistor switch.	58	Design a comparator for two unsigned numbers.
17	Draw the NMOS realization of a logic network.	59	Design a comparator for two signed numbers.
18	Draw the CMOS realization of a logic network.	60	Design a basic SR latch with NOR and NAND gates.
19	Use a Karnaugh map to find the minimum-cost sum-of-product (SOP) for a 2-variable logic function.	61	Design a gated SR latch with NAND gates.
20	Use a Karnaugh map to find the minimum-cost sum-of-product (SOP) for a 2-variable logic function.	62	Design a gated D latch.
21	Use a Karnaugh map to find the minimum-cost sum-of-product (SOP) for a 4 and 5 variable logic function.	63	Design a master-slave D flip-flop with two gated D latches.
22	Identify the literals, implicants, prime implicants, cover, and cost of a logic function.	64	Design a positive edge triggered D flip-flop from NAND gates.
23	Know how to use essential and non-essential prime implicants to find a minimum cost cover for a sum-of-products (SOP) implementation.	65	Design a negative edge triggered D flip-flop from NOR gates.
24	Know how to use essential and non-essential prime implicants to find a minimum cost cover for a product-of-sums (POS) implementation.	66	Describe how an asynchronous and synchronous D flip-flop clear and preset work.
25	Give the canonical SOP for a K-map with don't cares.	67	Explain the differences between types of flip-flops and latches.
26	Given two logical functions, draw a logical circuit with multiple output circuit sharing.	68	Design a shift register with D flip-flops.
27	Explain what fan-in means, why high fan-in is a problem, and the engineering trade-offs of avoiding high fan-in.	69	Design a parallel-access shift register with D flip-flops.
28	Simplify a logical function with factoring.	70	Design an asynchronous up-counter or down-counter.
29	Simplify a logical function with disjoint functional decomposition.	71	Design a synchronous up-counter or down-counter.
30	Simplify a logical function with non-disjoint functional decomposition.	72	Design a modulo-n counter with a synchronous reset.
31	Convert a multi-level circuit to NAND or NOR gates.	73	Create a state diagram for a Moore-type finite state machine (FSM).
32	Trace a multi-level circuit to determine its logical function.	74	Derive the state assigned table for a Moore-type finite state machine (FSM).
33	Convert a binary number to decimal.	75	Implement the digital logic circuit for a Moore-type finite state machine (FSM).
34	Convert a binary number to octal and hexadecimal.	76	Demonstrate how different state assignments can have an effect on the cost of a finite state machine (FSM).
35	Design a full adder (FA).	77	Simplify the logical expressions of a finite state machine (FSM) with one-hot encoding.
36	Design an n-bit ripple-carry adder.	78	Design a Mealy-type finite state machine (FSM).
37	Represent a signed binary number in 2's compliment form.	79	Design a Mealy-type finite state machine (FSM) serial adder.
38	Add signed binary numbers with 2's compliment arithmetic.	80	Design a Moore-type finite state machine (FSM) serial adder.
39	Subtract signed binary numbers with 2's compliment arithmetic.	81	Design a finite state machine (FSM) counter with D flip-flops.
40	Design an adder/subtractor unit.	82	Design a finite state machine (FSM) counter with JK flip-flops.
41	Design an adder/subtractor unit that detects arithmetic overflow.	83	Design a finite state machine (FSM) that counts non-sequential pulses on a line.
42	Calculate the critical path delay for a multi-level circuit.	84	Analyze the behavior of an existing finite state machine (FSM).