CHAPTER 1

INTRODUCTION

1.1 EXERCISES

Section 1.2: The World of Digital Systems

1.1. What is a digital signal and how does it differ from an analog signal? Give two everyday examples of digital phenomena (e.g., a window can be open or closed) and two everyday examples of analog phenomena.

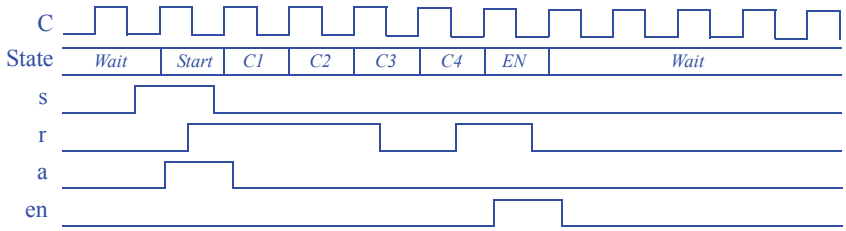
A digital signal at any time takes on one of a finite number of possible values, whereas an analog signal can take on one of infinite possible values. Examples of digital phenomena include a traffic light that is either be red, yellow, or green; a television that is on channel 1, 2, 3, ..., or 99; a book that is open to page 1, 2, ..., or 200; or a clothes hangar that either has something hanging from it or doesn't. Examples of analog phenomena include the temperature of a room, the speed of a car, the distance separating two objects, or the volume of a television set (of course, each analog phenomena could be digitized into a finite number of possible values, with some accompanying loss of information).

1.2 Suppose an analog audio signal comes in over a wire, and the voltage on the wire can range from 0 Volts (V) to 3 V. You want to convert the analog signal to a digital signal. You decide to encode each sample using two bits, such that 0 V would be encoded as `00`, 1 V as `01`, 2 V as `10`, and 3 V as `11`. You sample the signal every 1 millisecond and detect the following sequence of voltages: 0V 0V 1V 2V 3V 2V 1V. Show the signal converted to digital as a stream of `0`s and `1`s.

00 00 01 10 11 10 01

1.3 Assume that 0 V is encoded as `00`, 1 V as `01`, 2 V as `10`, and 3 V as `11`. You are given a digital encoding of an audio signal as follows: `1111101001010000`. Plot

3.32 Draw a timing diagram for the FSM in Figure 3.108with the FSM starting in state *Wait*. Choose input values such that the FSM reaches state *EN,* and returns to *Wait*.



3.33 For FSMs with the following numbers of states, indicate the smallest possible number of bits for a state register representing those states:

a. 4
b. 8
c. 9
d. 23
e. 900

a) 2 bits
b) 3 bits
c) 4 bits
d) 5 bits
e) 10 bits

3.34 How many possible states can be represented by a 16-bit register?

2^{16} = 65,536 possible states

3.35 If an FSM has N states, what is the maximum number of possible transitions that could exist in the FSM? Assume that no pair of states has more than one transition in the same direction, and that no state has a transition point back to itself. Assuming there are a large number of inputs, meaning the number of transitions is not limited by the number of inputs? Hint: try for small N, and then generalize.

For two states A and B, there are only 2 possible transitions: A->B and B->A. For three states A, B, and C, possible transitions are A->B, A->C, B->A, B->C, C->A, and C->B, for 6 possible transitions. For each of N states, there can be up to N-1 transitions pointing to other states. Thus, the maximum possible is N*(N-1).

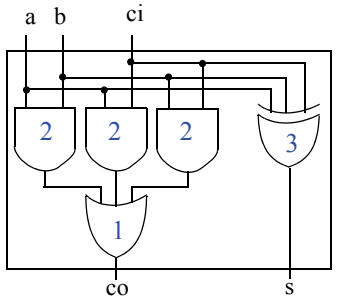
3.36 *Assuming one input and one output, how many possible four-state FSMs exist?

The complete solution to this challenge problem is not provided.The solution involves determining a way to enumerate all possible transitions from each state, and all possible actions in a state.

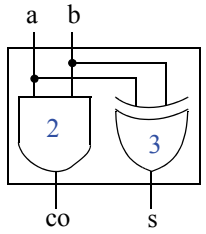
4.9 Assuming all gates have a delay of 1 ns, compute the longest time required to add two numbers using an 8-bit carry-ripple adder.

An 8-bit carry-ripple adder contains 7 full adders and 1 half adder. Each full adder has 2 gate delays and the half adder has 1 gate delay. Therefore a minimum of (7 FA * 2 gate delay/FA * + 1 HA * 1 gate delay/HA) * 1ns/gate delay = 15 ns is required to ensure that the carry-ripple adder's sum is correct.

4.10 Assuming AND gates have a delay of 2 ns, OR gates have a delay of 1 ns, and XOR gates have a delay of 3 ns, compute the longest time required to add two numbers using an 8-bit carry-ripple adder.



Full Adder

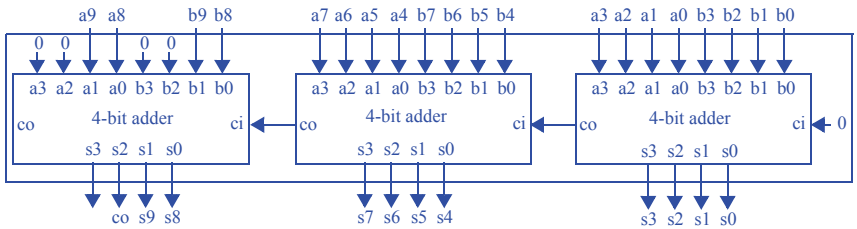


Half Adder

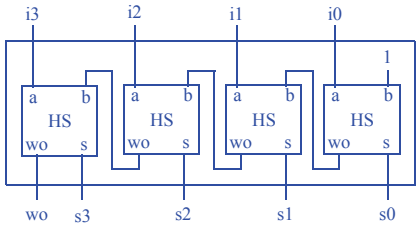
From the illustration above, we see that both the FA and HA have a maximum gate delay of 3 ns. Therefore, 8 adders * 3 ns/adder = 24 ns is required for an 8-bit carry-ripple adder to ensure a correct sum is on the adder's output.

An answer of 23 ns is also acceptable since the carry out of a half-adder will be correct after 2 ns, not 3 ns, and a half-adder may be used for adding the first pair of bits (least significant bits) if the 8-bit adder has no carry-in.

4.11 Design a 10-bit carry-ripple adder using 4-bit carry-ripple adders. *(Component use problem)*.



4.55 Design a circuit for a 4-bit decrementer. *(Component design problem)*.



4.56 Assume an electronic turnstile internally uses a 64-bit counter that counts up once for each person that passes through the turnstile. Knowing that California's Disneyland park attracts about 15,000 visitors per day, and assuming they all pass that one turnstile, how many days would pass before the counter would roll over? (Component use problem.)

$2^{64}/15000 = 1{,}229{,}782{,}938{,}247{,}303$ days. That's a long time.

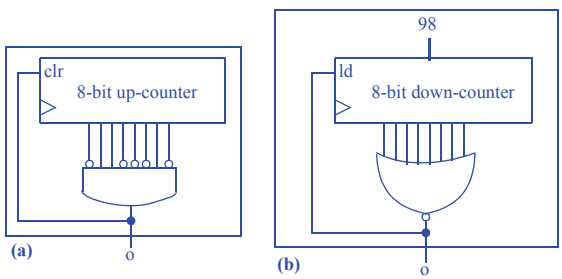
4.57 Design a circuit that outputs a 1 every 99 clock cycles:

a. Using an up-counter with a synchronous clear control input, and using extra logic,

b. Using a down-counter with parallel load, and using extra logic.

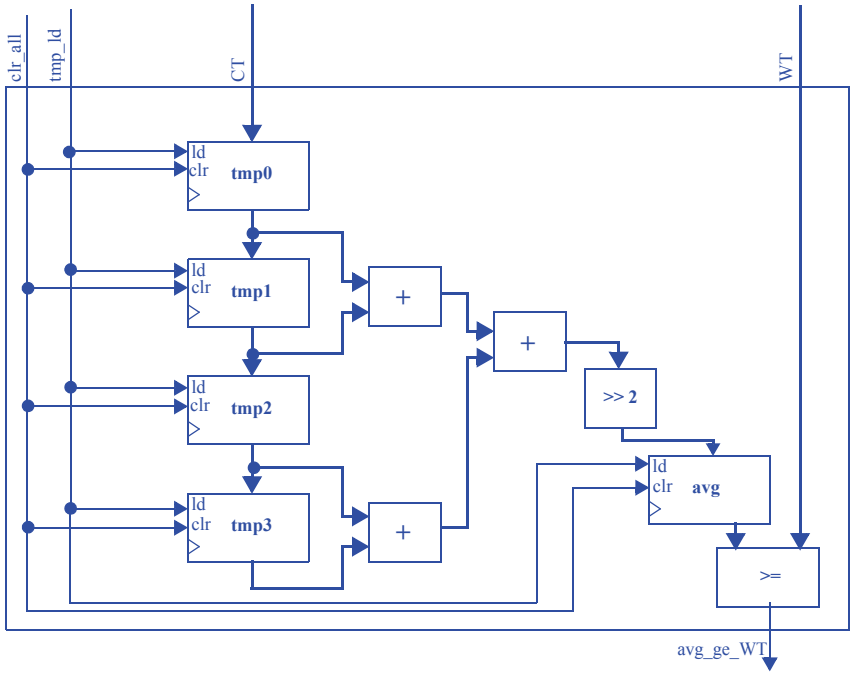
c. What are the tradeoffs between the two designs from parts (a) and (b)?

(Component use problem.)



(c) The circuit implemented in (a) is smaller, while the circuit implemented in (b) is easier to modify to pulse at a different rate.

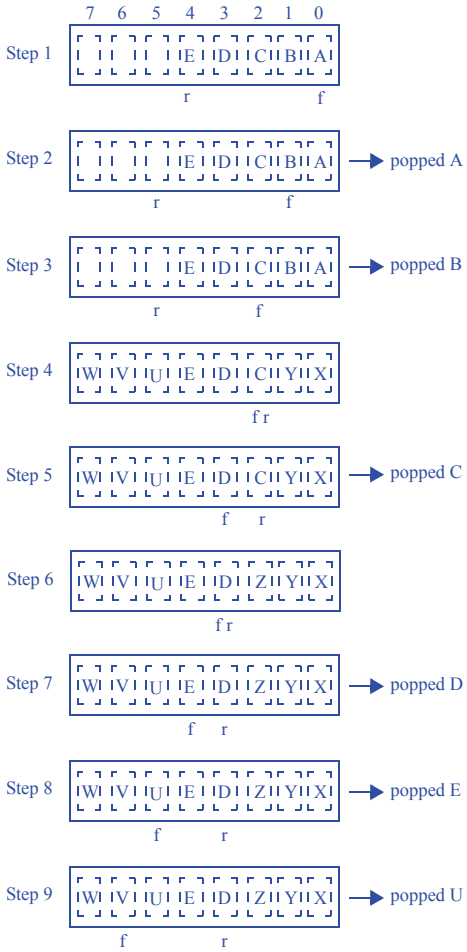
Step 2A - Create a datapath



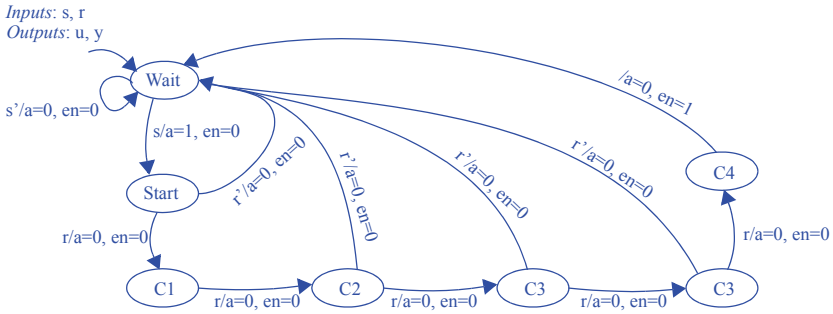
Note: A solution more consistent with the chapter's methdology would use a separate clear and ld signal for each register. In this particular example, a single clr and a single load line happens to work.

Section 5.7: Queues (FIFOs)

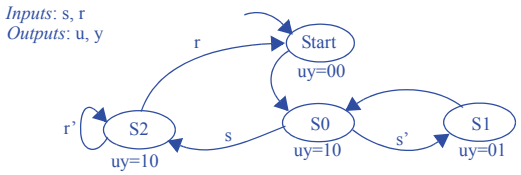
5.44 For an 8-word queue, show the queue's internal state and provide the value of popped data for the following sequences of pushes and pops: (1) push A, B, C, D, E, (2) pop, (3) pop, (4) push U, V, W, X, Y, (5) pop, (6) push Z, (7) pop, (8) pop, (9) pop.



6.22) Convert the Moore FSM in Figure 6.92 to the nearest Mealy FSM equivalent.

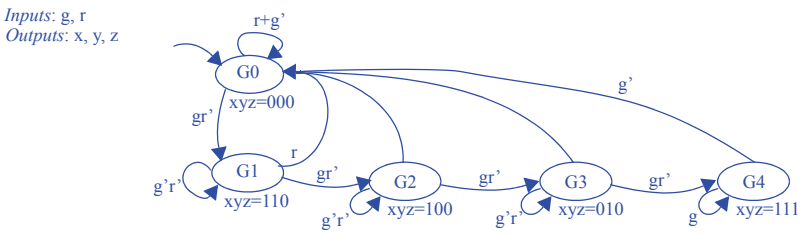


6.23) Convert the Mealy FSM in Figure 6.93 to the nearest Moore equivalent.

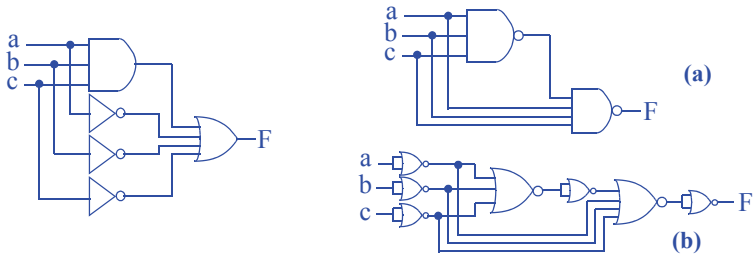


6.24) Convert the Mealy FSM in Figure 6.94 to the nearest Moore equivalent.

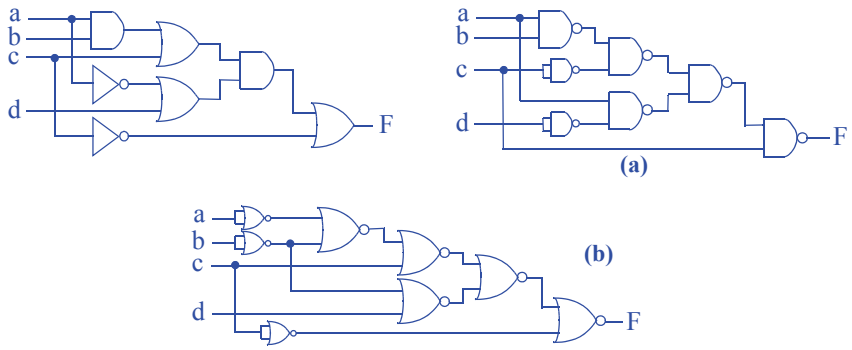
.



7.10 Draw a circuit using AND, OR and NOT gates for the following function: `F(a,b,c) = abc + a' + b' + c'`. Place inversion bubbles on that circuit to convert that circuit to: (a) NAND gates only, (b) NOR gates only.



7.11 Draw a circuit using AND, OR, and NOT gates for the following function: `F(a,b,c) = (ab + c)(a' + d) + c'`. Convert the circuit to a circuit using: (a) NAND gates only, (b) NOR gates only.



7.12 Draw a circuit using AND, OR, and NOT gates for the following function: `F(w,x,y,z) = (w + x)(y + z) + wy + xz`. Convert the circuit to a circuit using: (a) NAND gates only, (b) NOR gates only..