Lab - Finite State Machines

Serial Data Code Conversion

As an example of state graph construction, we will design a converter for serial data. Binary data is frequently transmitted between computers as a serial stream of bits. As shown in Figure 14-19(a), a clock signal is often transmitted along with the data, so the receiver can read the data at the proper time. Alternatively (Figure 14-19(b)), only the serial data is transmitted, and a clock recovery circuit (called a digital phase-locked loop) is used to regenerate the clock signal at the receiver.

Figure 14-20 shows four different coding schemes for serial data together with the clock used to synchronize the data transmission. The example shows the transmission of the bit sequence 0, 1, 1, 1, 0, 0, 1, 0. With the NRZ (non-return-to-zero) code, each bit is transmitted for one bit time without any change. With the NRZI (non-return-to-zero-inverted) code, the data is encoded by the presence or absence of transitions in the output signal. For each 0 in the original sequence, the

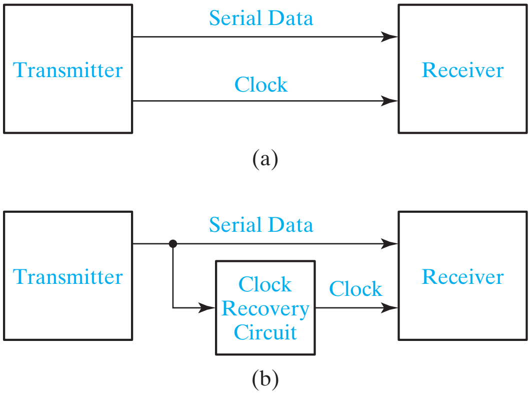


FIGURE 14-19 Serial Data Transmission

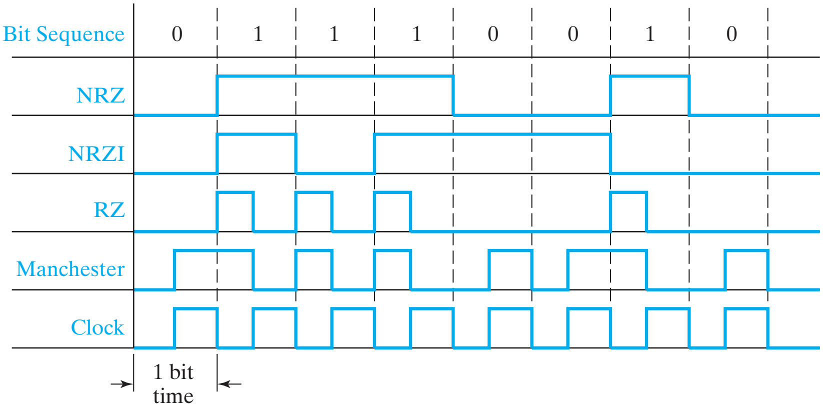


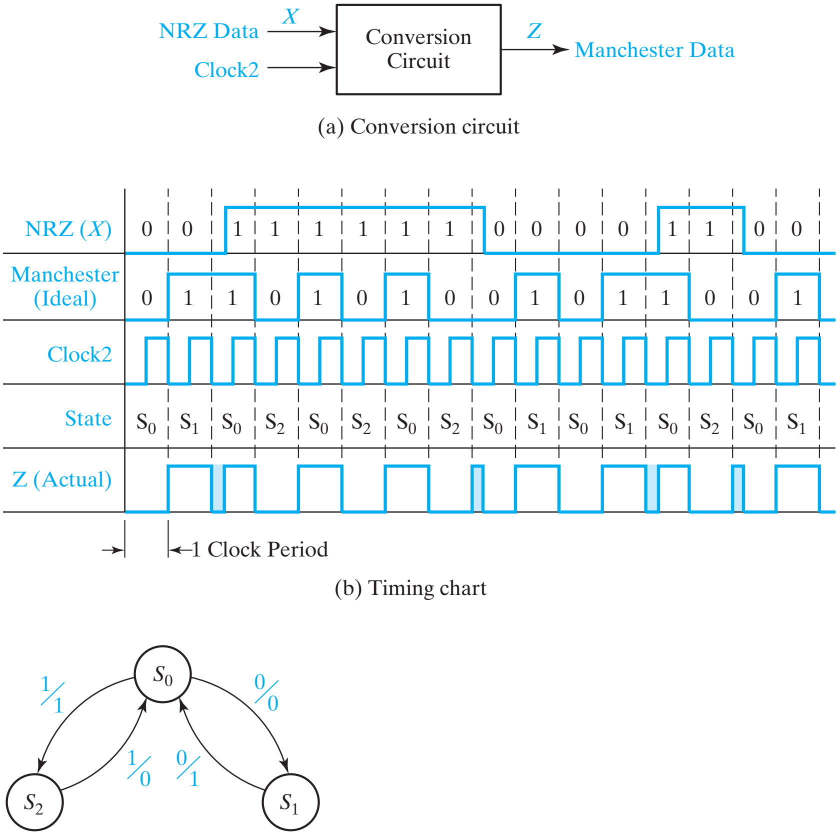
FIGURE 14-20 Coding Schemes for Serial Data Transmission

bit transmitted is the same as the previous bit transmitted. For each 1 in the original sequence, the bit transmitted is the complement of the previous bit transmitted. Thus, the preceding sequence is encoded as 0, 1, 0, 1, 1, 1, 0, 0. In other words, a 0 is encoded by no change in the transmitted value, and a 1 is encoded by inverting the previous transmitted value. For the RZ (return-to-zero) code, a 0 is transmitted as a 0 for one full bit time, but a 1 is transmitted as a 1 for the first half of the bit time and, then, the signal returns to 0 for the second half. For the Manchester code, a 0 is transmitted as 0 for the first half of the bit time and 1 for the second half, but a 1 is transmitted as 1 for the first half and 0 for the second half. Thus, the encoded bit always changes in the middle of the bit time. When the original bit sequence has a long string of 1's and 0's, the Manchester code has more transitions. This makes it easier to recover the clock signal.

We will design a sequential circuit which converts an NRZ-coded bit stream to a Manchester-coded bit stream (Figure 14-21(a)). In order to do this, we will use a clock, Clock2, that is twice the frequency of the basic clock (Figure 14-21(b)). In this way, all output changes will occur on the same edge of Clock2, and we can use the standard synchronous design techniques which we have been using in this unit. First, we will design a Mealy circuit to do the code conversion. Note that if the NRZ bit is 0, it will be 0 for two Clock2 periods. Similarly, if the NRZ bit is 1, it will be 1 for two Clock2 periods. Thus, starting in the reset state (S_0 in Figure 14-21(c)), the only two possible input sequences are 00 and 11. For the sequence 00, when the first 0 is received, the output is 0. At the end of the first Clock2 period, the circuit goes to S_1. The input is still 0, so the output becomes 1 and remains 1 for one Clock2 period, and then the circuit resets to S_0. For the sequence 11, when the first 1 is received, the output is 1 for one Clock2 period and, then, the circuit goes to S_2. Then, the output is 0 for one Clock2 period, and the circuit resets to S_0.

When we convert the Mealy graph to a state table (Figure 14-21(d)), the next state of S_1 with an input of 1 is not specified and is represented by a dash. Similarly, the next state of S_2 with a 0 input is not specified. The dashes are like don't-cares, in that we do not care what the next state will be because the corresponding input sequence never occurs. A careful timing analysis for the Mealy circuit shows some possible glitches (false outputs) in the output waveform (Figure 14-21(b)). The input waveform may not be exactly synchronized with the clock, and we have exaggerated this condition in the figure by shifting the input waveform to the right so that the input changes do not line up with the clock edges. For this situation, we will use the state table to analyze the occurrence of glitches in the Z output. The first glitch shown in the timing chart occurs when the circuit is in state S_1, with an input $X = 0$. The state table shows that the output is $Z = 1$, and when the clock goes low, the state changes to S_0. At this time, the input is still $X = 0$, so Z becomes 0. Then X changes to 1, Z becomes 1 again, so a glitch has occurred in the output during the time interval between the clock change and the input change. The next glitch occurs in S_2 with $X = 1$ and $Z = 0$. When the clock goes low, the output momentarily becomes 1 until X is changed to 0.

FIGURE 14-21
Mealy Circuit for NRZ to Manchester Conversion



(a) Conversion circuit

(b) Timing chart

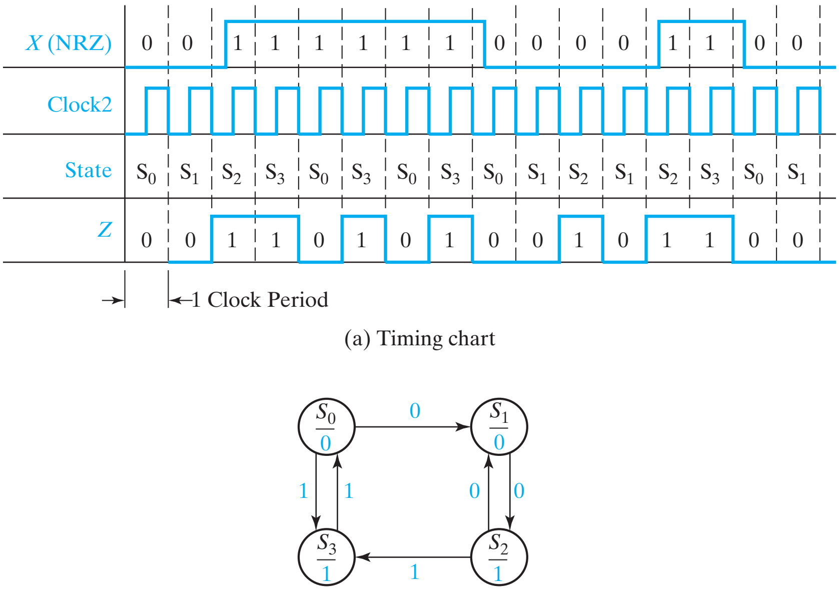
(c) State graph

Present State	Next State $X = 0$	Next State $X = 1$	Output (Z) $X = 0$	Output (Z) $X = 1$
S_0	S_1	S_2	0	1
S_1	S_0	–	1	–
S_2	–	S_0	–	0

(d) State table

To overcome the possible glitch problem with the Mealy circuit, we will redesign the circuit in Moore form (Figure 14-22 on the next page). Because the output of a Moore circuit cannot change until after the active edge of the clock, the output will be delayed by one clock period. Starting in S_0, the input sequence 00 takes us to state S_1 with a 0 output and, then, to S_2 with a 1 output. Starting in S_0, 11 takes us to S_3 with a 1 output, and the second 1 can take us back to S_0 which has a 0 output. To complete the graph, we add the two arrows starting in S_2. Note that a 1 input cannot occur in S_1, and a 0 output cannot occur in S_3, so the corresponding state table has two don't-cares.

FIGURE 14-22 Moore Circuit for NRZ to Manchester Conversion



(a) Timing chart

(b) State graph

Present State	Next State		Present Output (Z)
	$X = 0$	$X = 1$	
S_0	S_1	S_3	0
S_1	S_2	–	0
S_2	S_1	S_3	1
S_3	–	S_0	1

(c) State table

Task 1: (a) Derive the state graph and table for a Mealy sequential circuit that converts a serial stream of bits from Manchester code to NRZ code. Assume that a double frequency clock (Clock2) is available.

(b) Repeat (a) for a Moore sequential circuit.

(c) Draw a timing diagram similar to Figure 14-21(b) for your answer to (a), using the Manchester waveform in Figure 14-21(b) as the input waveform to your circuit. If the input changes occur slightly after the clock edge, indicate places in the output waveform where glitches (false outputs) can occur. If possible, assign the don't-cares in the output part of your state table to eliminate some of the glitches.

(d) Draw the timing diagram for your answer to (b), using the same input waveform as in (c).

(e) Design two Verilog modules that test the NRZ to Manchester code conversion and the Manchester to NRZ code conversion. Use test benches to verify the design and match the timing diagram with the waveform you anticipated.
Define the states as outputs so that they can show in the simulation results. You can decide which type of FSM you will work with, but back your selection with reason.

Task 2: Design a serial encoder that converts NRZ to PAM4. Assume that the output of the encoder is a 2-bit bus that will pass through a DAC to get mapped to the 4 analog levels. Use a test bench to test it.