

Electrical and Computer Engineering Department
Rutgers University
14:332:431 Digital Systems Design
Fall 1998
Final Examination – Closed Book

Prof. Michael L. Bushnell

Dec. 17, 1998

Name: _____ Student Number (SS No.): _____

Write all of your work on this examination in the spaces provided.

Problem 1	8	
Problem 2	6	
Problem 3	16	
Problem 4	20	
Problem 5	11	
Problem 6	8	
Problem 7	11	
Problem 8	2	
Problem 9	7	
Problem 10	11	
Total	100	

Problem 1 (8 Points, 15 Minutes) *State Machine Synthesis.* Design a Mealy synchronous sequential circuit with one input and one output. The machine produces an output $Z = 1$ **ONLY** whenever one of the following input sequences occur: 1101, 0011. Sequences may start at any time, so avoid designing the state transition diagram so that sequences are recognized only if the sequences line up on some multiple of four clock periods. The circuit resets to its initial state (which should be coded as all zeroes) after a 1 output has been generated. If the machine receives an input sequence other than these two, it should reset itself immediately to the initial state.

1. Draw the state diagram for the machine. If the machine enters any unused states, it should go to the initial state on the next clock edge.
2. Write the coded state transition table for the machine. Code the machine state to have *reduced input dependency* in the transition from the states recognizing the first and second input sequence digits to their successor states.

Problem 2 (6 Points, 10 Minutes) *State Reduction.* Find the equivalent state partition for the state machine shown in Table 1 that has two inputs x_1 and x_2 and one output z . This machine has unnecessary state transitions and logic.

Table 1: State Transition Table for Problem 2

Present State	Next State / Output			
	$x_1x_2 = 00$	$x_1x_2 = 01$	$x_1x_2 = 11$	$x_1x_2 = 10$
s0	s1/0	s1/1	s0/0	s2/0
s1	s4/0	s4/1	s1/0	s3/0
s2	s3/0	s7/1	s2/0	s0/1
s3	s7/0	s6/1	s3/0	s1/1
s4	s0/0	s1/1	s4/0	s6/0
s5	s1/1	s0/0	s7/1	s6/1
s6	s2/0	s3/1	s7/0	s0/1
s7	s6/0	s3/1	s6/0	s4/1

1. Show the sets of states for the reduced machine. Use the method of *partitioning*.
2. Which of the two kinds of synchronous state machine is this machine? _____

Problem 3 (16 Points, 29 Minutes) *Built-In Logic Block Observer for Testing.*

1. Please design a five-bit Built-In Logic Block Observer cell with the control modes set by signals B_1 and B_2 in Table 2. The LFSR feedback polynomial for the circuit should be

Table 2: BILBO Control Signals

B1	B2	Mode
0	0	SCAN Chain
0	1	LFSR
1	0	Normal Mode
1	1	MISR

1. $1 + x^3$. Assume that the BILBO has five bits. Please use a Karnaugh map to optimize the hardware for the BILBO MUX and the individual BILBO cells.
2. Now that you have a logic-level realization of the BILBO, please express the BILBO design (including the flip-flops) in VHDL using your Boolean logic equations from your Karnaugh maps.

Problem 4 (20 Points, 35 Minutes) *Fault-Tolerant State Machine Synthesis.*

1. Realize the following *Coded State Transition Table* in Table 3 for a state machine for Flip-Flops Q_1 , Q_2 , Q_3 and for the output Z using T Flip-Flops. Show the *Excitation Table* for the T Flip-Flop. Assume don't cares for the next states of any states not listed in the table. Show all Karnaugh maps and write the Boolean equations for the minimal two-level sum-of-product forms for the Boolean input equations for the *Next State* and *Output Decoders*.
2. Now, describe the finite state machine using VHDL at the behavioral level. Let the VHDL language processor choose the state assignment.

Table 3: State Transition Table for Problem 4

Present State			Next State / Output	
Q1	Q2	Q3	x = 0	x = 1
<i>a</i>	0	00	110/1	111/1
<i>b</i>	0	01	101/0	110/0
<i>c</i>	0	10	001/1	001/0
<i>d</i>	0	11	101/1	001/1
<i>e</i>	1	00	010/0	101/1
<i>f</i>	1	01	101/1	011/0
<i>g</i>	1	10	111/1	100/0
<i>h</i>	1	11	011/0	001/0

- Finally, design a triple modular redundancy system for fault tolerance, in which there are three copies of this finite state machine and a triplicated voting circuit to check the behavior of the three copies. Write the VHDL code for the voting circuit.

Problem 5 (11 Points, 20 Minutes) *Computer Arithmetic.*

- Show all binary equations for a four-bit binary carry lookahead adder for these signals: $G0-3$, $P0-3$, $SUM0-3$, and $C0-4$. $A0-3$ and $B0-3$ are the addends to the machine, and $C0$ is the least significant carry-in bit. G represents generate signals, and P represents propagate signals. Bit 0 is the least significant bit and bit 3 is the most significant bit except that bit $C0$ is the carry into the least significant bit and bit $C4$ is the carry out of the most significant bit.
- Now, design a 16-bit binary carry lookahead adder by using four 4-bit adders (just to calculate sum bits) and four 4-bit carry lookahead units. Design a *super* lookahead unit that does the carry lookahead for the entire adder using generate and propagate signals from each of the 4-bit carry lookahead units.

Problem 6 (8 Points, 15 Minutes) *Hamming Codes and Fault Tolerance.* Please design a Hamming Code system with four information bits, d_0 d_1 d_2 d_3 , and three parity groups with parity bits P_0 P_1 P_2 . Table 4 shows the parity groups and the error syndromes. Draw a system

Table 4: Parity Groups and Syndromes

Parity Group		Bits		
P_0	d_0	d_1	d_3	
P_1	d_0	d_2	d_3	
P_2	d_1	d_2	d_3	
Syndromes				
Bit in Error		Parity in Error		
d_3	P_2	P_1	P_0	
d_2	P_2	P_1		
d_1	P_2	P_0		
d_0	P_1		P_0	
P_2	P_2			
P_1	P_1			
P_0	P_0			

block diagram of this error correcting code memory system, including the memory address register, memory data register the memory, the parity bit generator, the parity bit regenerator, the syndrome decoder, and the controlled complements to correct the erroneous bits.

Problem 7 (11 Points, 20 Minutes) *Cyclic Codes.* Construct a dividing decoding circuit for a cyclic code for a 512-bit disk block of data. Use the Generator polynomial $1 + x^3 + x^7$. Make sure that you provide a way to initialize the registers in your encoding circuits to zeroes before encoding begins. What serial data will your circuit produce for the input cyclic code bit stream 10100001, assuming that all memory elements were initialized to 0 ? We are only interested in the first 8 bits of the decoded bit stream.

Problem 8 (2 Points, 3 Minutes) *Mixed Logic Notation.* Convert the following circuit schematic in Figure 1 into MIXED logic notation, using signals A , B , $\neg C$, E , and $\neg F$. This is merely a notational change, so you must not alter the Boolean function of the circuit.

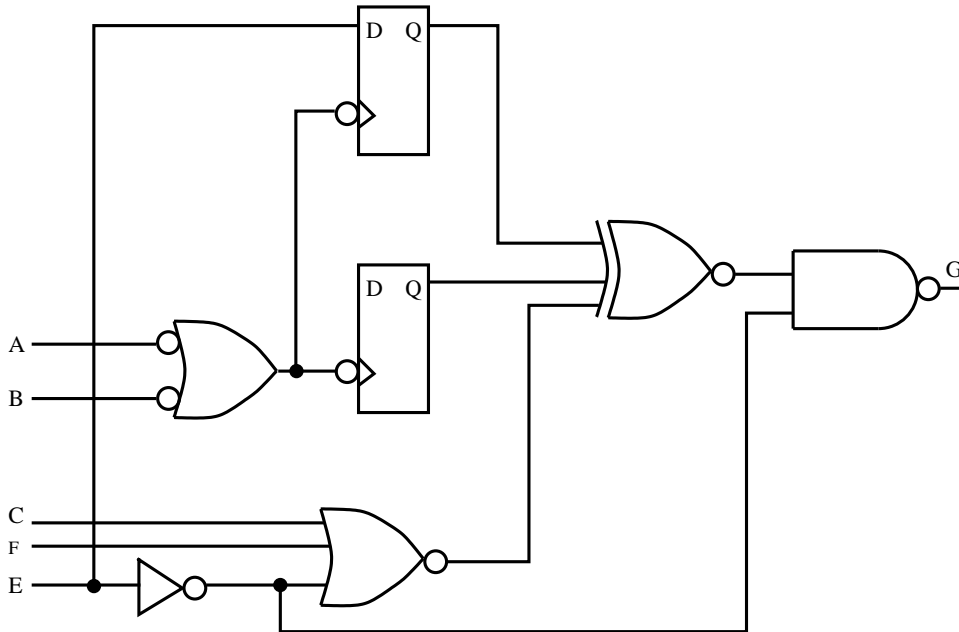


Figure 1: Circuit for Problem 8

Problem 9 (7 Points, 13 Minutes) *Combinational Logic Indirect Addressed MUX Based Design.* Implement the following function $G(a, b, c, d, e, f, g, h, k)$ given in the truth table given in Table 5 using an *indirect-addressed* MUX in Figure 2. Show the Variable-Entered Karnaugh map for G . Use a, b, c, d as *Control Variables* and e, f, g, h, k as *Map-entered Variables*. You may use only the *8-input, 1-output* MUX and the ROM shown below. Your ROM addresses should be selected by a, b, c, d in that order. No credit will be given if you do not use the *8-input, 1-output* MUX, if you do not use the ROM, or if your map entered and control variables violate the specifications. No credit will be given if you do not supply a Variable-Entered Map. The A lines give the ROM address, $Q0-2$ are the ROM outputs, $S0-2$ are the MUX selector lines, $0-7$ are the MUX inputs, and the \overline{ENABLE} signals enable the ROM and the MUX. Use don't cares to abbreviate the ROM programming – you need not write out the programming for each address, just for identical address ranges.

Table 5: Truth Table for Problem 9

a	b	c	d	G
0	0	0	0	$g \oplus h$
0	0	0	1	$\overline{f} \wedge g$
0	0	1	0	$k \oplus e \oplus \overline{h}$
0	0	1	1	\overline{h}
0	1	X	X	$k \oplus e \oplus h$
1	0	0	0	h
1	0	0	1	$\overline{h} \wedge g$
1	0	1	0	$k \oplus e \oplus \overline{h}$
1	0	1	1	\overline{h}
1	1	0	0	0
1	1	0	1	$g \oplus h$
1	1	1	0	\overline{h}
1	1	1	1	$\overline{f} \wedge g$

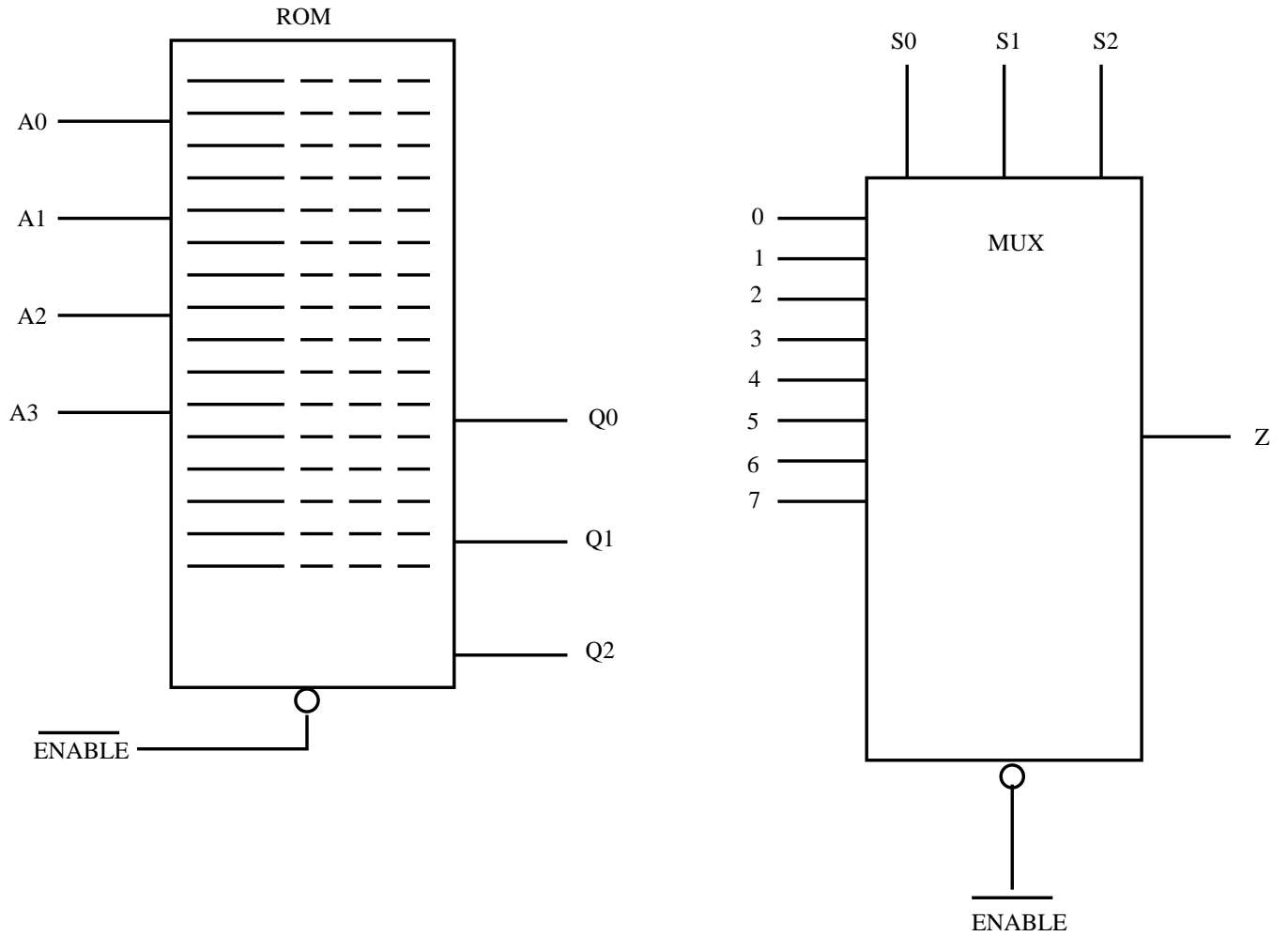


Figure 2: Hardware for Problem 9

Problem 10 (11 Points, 20 Minutes) *Built-in Self-Testing Circuits.* In the logic diagram in Figure 3, the portion of the circuit outlined in dashed lines is the actual circuit to be tested. Signals $A1$, $B1$, and $C1$ are the inputs to the circuit and signal F is the circuit output. The logic to the left of the circuit is very similar to the hardware used to calculate a cyclic code, and is called a *Linear Feedback Shift Register*. For normal circuit operation, inputs A , B and C are used normally to drive signals $A1$, $B1$ and $C1$ because $TEST = 0$. The circuit output is taken from signal F . The logic to the right of the circuit output F is also very similar to the cyclic code generator, and is called a *Multiple Signature Analysis Register* (MISR). The MISR is not used in normal circuit operation, and outputs $S1$, $S2$ and $S3$ are ignored during normal operation. The $TEST$ signal is set to a 0 to route real circuit inputs through the input MUX during normal operation, and set to a 1 to drive the circuit from the LFSR during testing.

In test mode, we set $TEST = 1$. We initialize all flip-flops by setting signal \overline{RESET} to 0 and then restoring it to 1. This initializes $Q1Q2Q3$ to 001 and $S1S2S3$ to 000. We then clock the circuit for 7 clock periods on the clock lines. The Linear Feedback Shift Register will generate the input sequence $001 \rightarrow 100 \rightarrow 010 \rightarrow 101 \rightarrow 110 \rightarrow 111 \rightarrow 011 \rightarrow 001$. During the seven clock periods while we are generating this sequence, the MISR is also clocked, and will compute a polynomial division remainder very like the cyclic codes discussed in the course. After the seven clock periods, signals $S1S2S3$ will contain a three-bit number known as the *signature* of the circuit. If the circuit has no signal lines permanently stuck-at 0 or 1, this signature will be the signature of the *good machine*. When we manufacture this circuit in volume, we will repeat this testing sequence for each copy of the circuit, and examine $S1S2S3$ after testing. If each manufactured circuit generates the good machine signature, then it is assumed to be correct. Circuits that generate incorrect (*failing machine*) signatures are rejected.

For this problem, calculate the final values of $S1S2S3$ for the *good machine signature*. Then, calculate the *failing machine signatures* for single circuit fault: signal e stuck-at 0. Was the fault detected?

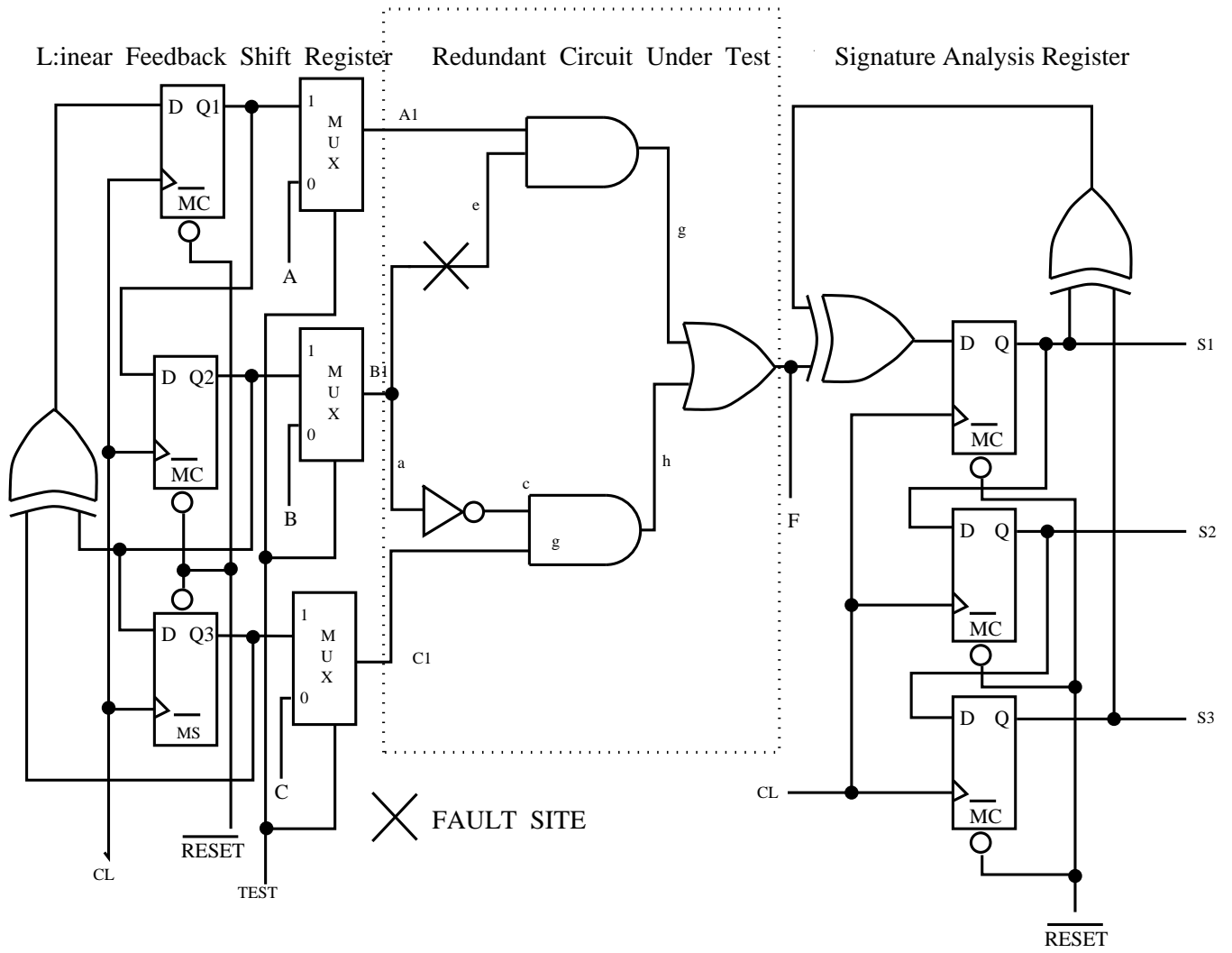


Figure 3: Hardware for Problem 10