5.29 Write an algebraic expression for s_2, the third sum bit of a binary adder, as a function of inputs x_0, x_1, x_2, y_0, y_1, and y_2. Assume that $c_0 = 0$, and do not attempt to "multiply out" or minimize the expression.

5.30 Using the information in Table 5-3 for 74LS components, determine the maximum propagation delay from any input to any output of the 16-bit group ripple adder of Figure 5-91. You may use the "worst-case" analysis method.

Exercises

5.31 A possible definition of a BUT gate (Exercise 4.45) is "Y1 is 1 if A1 and B1 are 1 *but* either A2 or B2 is 0; Y2 is defined symmetrically." Write the truth table and find minimal sum-of-products expressions for the BUT-gate outputs. Draw the logic diagram for a NAND-NAND circuit for the expressions, assuming that only uncomplemented inputs are available. You may use gates from 74HCT00, '04, '10, '20, and '30 packages.

5.32 Find a gate-level design for the BUT gate defined in Exercise 5.31 that uses a minimum number of transistors when realized in CMOS. You may use gates from 74HCT00, '02, '04, '10, '20, and '30 packages. Write the output expressions (which need not be two-level sums-of-products), and draw the logic diagram.

5.33 For each circuit in the two preceding exercises, compute the worst-case delay from input to output, using the delay numbers for 74HCT components in Table 5-2. Compare the cost (number of transistors), speed, and input loading of the two designs. Which is better?

5.34 Butify the function $F = \Sigma_{W,X,Y,Z}(3,7,11,12,13,14)$. That is, show how to perform F with a single BUT gate as defined in Exercise 5.31 and a single 2-input OR gate. *butification*

5.35 Design a 1-out-of-4 checker with four inputs, A, B, C, D, and a single output ERR. The output should be 1 if two or more of the inputs are 1, and 0 if no input or one input is 1. Use SSI parts from Figure 5-18, and try to minimize the number of gates required. (*Hint:* It can be done with seven two-input inverting gates.)

5.36 Suppose that a 74LS138 decoder is connected so that all enable inputs are asserted and C B A = 101. Using the information in Table 5-3 and the '138 internal logic diagram, determine the propagation delay from input to all relevant outputs for each possible single-input change. (*Hint:* There are a total of nine delay numbers, since a change on A, B, or C affects two outputs, and a change on any of the three enable inputs affects one output.)

5.37 Suppose that you are asked to design a new component, a decimal decoder that is optimized for applications in which only decimal input combinations are expected to occur. How can the cost of such a decoder be minimized compared to one that is simply a 4-to-16 decoder with six outputs removed? Write the logic equations for all ten outputs of the minimized decoder, assuming active-high inputs and outputs and no enable inputs.

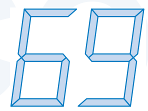
5.38 How many Karnaugh maps would be required to work Exercise 5.37 using the formal multiple-output minimization procedure described in Section 4.3.8?

5.39 Suppose that a system requires a 5-to-32 binary decoder with a single active-low enable input, a design similar to Figure 5-39. With the EN1 input pulled HIGH, either the EN2_L or the EN3_L input in the figure could be used as the enable, with the other input grounded. Discuss the pros and cons of using EN2_L versus EN3_L.

5.40 Determine whether the circuits driving the a, b, and c outputs of the 74x49 seven-segment decoder correspond to minimal product-of-sums expressions for these segments, assuming that the nondecimal input combinations are “don’t cares” and BI = 1.

5.41 Redesign the MSI 74x49 seven-segment decoder so that the digits 6 and 9 have tails as shown in Figure X5.41. Are any of the digit patterns for nondecimal inputs 1010 through 1111 affected by your redesign?

Figure X5.41



5.42 Starting with the ABEL program in Table 5-21, write a program for a seven-segment decoder with the following enhancements:

- The outputs are all active low.
- Two new inputs, ENHEX and ERRDET, control the decoding of the segment outputs.
- If ENHEX = 0, the outputs match the behavior of a 74x49.
- If ENHEX = 1, then the outputs for digits 6 and 9 have tails, and the outputs for digits A–F are controlled by ERRDET.
- If ENHEX = 1 and ERRDET = 0, then the outputs for digits A–F look like the letters A–F, as in the original program.
- If ENHEX = 1 and ERRDET = 1, then the output for digits A–F looks like the letter S.

5.43 A famous logic designer decided to quit teaching and make a fortune by fabricating huge quantities of the MSI circuit shown in Figure X5.47.

5-44 (a)Label the inputs and outputs of the circuit with appropriate signal names, including active-level indications.

5-45 (b)What does the circuit do? Be specific and account for all inputs and outputs.

5-46 (c)Draw the MSI logic symbol that would go on the data sheet of this wonderful device.

5-47 (d)With what standard MSI parts does the new part compete? Do you think it would be successful in the MSI marketplace?

5.48 An FCT three-state buffer drives ten FCT inputs and a 4.7-KΩ pull-up resistor to 5.0 V. When the output changes from LOW to Hi-Z, estimate how long it takes for the FCT inputs to see the output as HIGH. State any assumptions that you make.

5.49 On a three-state bus, ten FCT three-state buffers are driving ten FCT inputs and a 4.7-KΩ pull-up resistor to 5.0 V. Assuming that no other devices are driving the bus, estimate how long the bus signal remains at a valid logic level when an active output enters the Hi-Z state. State any assumptions that you make.

5.50 Design a 10-to-4 encoder with inputs in the 1-out-of-10 code and outputs in BCD.

5.51 Draw the logic diagram for a 16-to-4 encoder using just four 8-input NAND gates. What are the active levels of the inputs and outputs in your design?

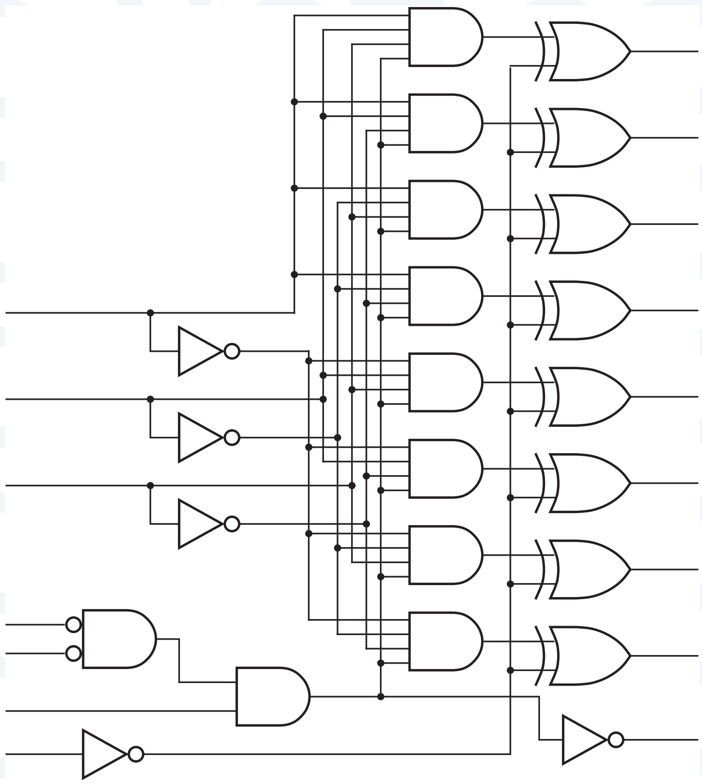


Figure X5.47

5.52 Draw the logic diagram for a circuit that uses the 74x148 to resolve priority among eight active-high inputs, I0–I7, where I7 has the highest priority. The circuit should produce active-high address outputs A2–A0 to indicate the number of the highest-priority asserted input. If no input is asserted, then A2–A0 should be 111 and an IDLE output should be asserted. You may use discrete gates in addition to the '148. Be sure to name all signals with the proper active levels.

5.53 Draw the logic diagram for a circuit that resolves priority among eight active-low inputs, I0_L–I7_L, where I0_L has the highest priority. The circuit should produce active-high address outputs A2–A0 to indicate the number of the highest-priority asserted input. If at least one input is asserted, then an AVALID output should be asserted. Be sure to name all signals with the proper active levels. This circuit can be built with a single 74x148 and no other gates.

5.54 A purpose of Exercise 5.53 was to demonstrate that it is not always possible to maintain consistency in active-level notation unless you are willing to define alternate logic symbols for MSI parts that can be used in different ways. Define an alternate symbol for the 74x148 that provides this consistency in Exercise 5.53.

5.55 Design a combinational circuit with eight active-low request inputs, R0_L–R7_L, and eight outputs, A2–A0, AVALID, B2–B0, and BVALID. The R0_L–R7_L inputs and A2–A0 and AVALID outputs are defined as in Exercise 5.53. The B2–B0 and BVALID outputs identify the second-highest priority request input that is asserted.

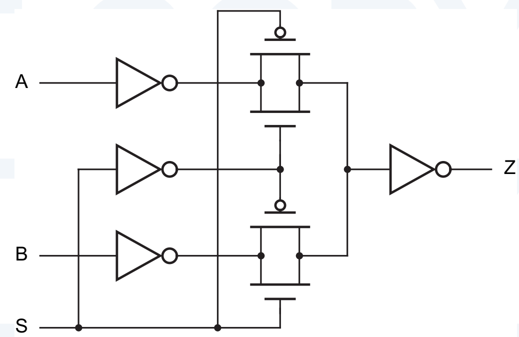


Figure X5.59

You should be able to design this circuit with no more than six SSI and MSI packages, but don't use more than 10 in any case.

5.56 Repeat Exercise 5.55 using ABEL. Does the design fit into a single GAL20V8?

5.57 Repeat Exercise 5.55 using VHDL.

5.58 Design a 3-input, 5-bit multiplexer that fits in a 24-pin IC package. Write the truth table and draw a logic diagram and logic symbol for your multiplexer.

5.59 Write the truth table and a logic diagram for the logic function performed by the CMOS circuit in Figure X5.59. (The circuit contains transmission gates, which were introduced in Section 3.7.1.)

5.60 A famous logic designer decided to quit teaching and make a fortune by fabricating huge quantities of the MSI circuit shown in Figure X5.64.

5-61 (a)Label the inputs and outputs of the circuit with appropriate signal names, including active-level indications.

5-62 (b)What does the circuit do? Be specific and account for all inputs and outputs.

5-63 (c)Draw the MSI logic symbol that would go on the data sheet of this wonderful device.

5-64 (d)With what standard MSI parts does the new part compete? Do you think it would be successful in the MSI marketplace?

barrel shifter

5.65 A 16-bit *barrel shifter* is a combinational logic circuit with 16 data inputs, 16 data outputs, and 4 control inputs. The output word equals the input word, rotated by a number of bit positions specified by the control inputs. For example, if the input word equals ABCDEFGHIJKLMNOP (each letter represents one bit), and the control inputs are 0101 (5), then the output word is FGHIJKLMNOPABCDE. Design a 16-bit barrel shifter using combinational MSI parts discussed in this chapter. Your design should contain 20 or fewer ICs. Do not draw a complete schematic, but sketch and describe your design in general terms and indicate the types and total number of ICs required.

5.66 Write an ABEL program for the barrel shifter in Exercise 5.65.

5.67 Write a VHDL program for the barrel shifter in Exercise 5.65.

5.68 Show how to realize the 4-input, 18-bit multiplexer with the functionality described in Table 5-39 using 18 74x151s.

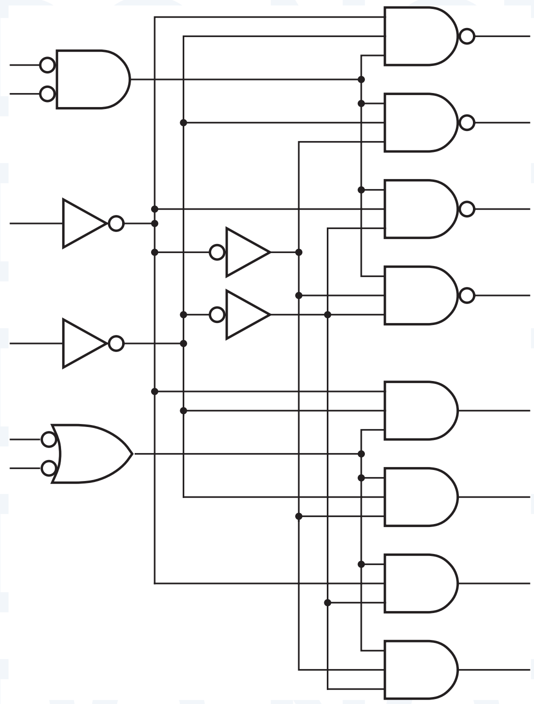


Figure X5.64

5.69 Show how to realize the 4-input, 18-bit multiplexer with the functionality of Table 5-39 using 9 74x153s and a "code converter" with inputs S2–S0 and outputs C1,C0 such that [C1,C0] = 00–11 when S2–S0 selects A–D, respectively.

5.70 Design a 3-input, 2-output combinational circuit that performs the code conversion specified in the previous exercise, using discrete gates.

5.71 Add a three-state-output control input OE to the VHDL multiplexer program in Table 5-42. Your solution should have only one process.

5.72 A digital designer who built the circuit in Figure 5-75 accidentally used 74x00s instead of '08s in the circuit, and found that the circuit still worked, except for a change in the active level of the ERROR signal. How was this possible?

5.73 What logic function is performed by the CMOS circuit shown in Figure X5.73?

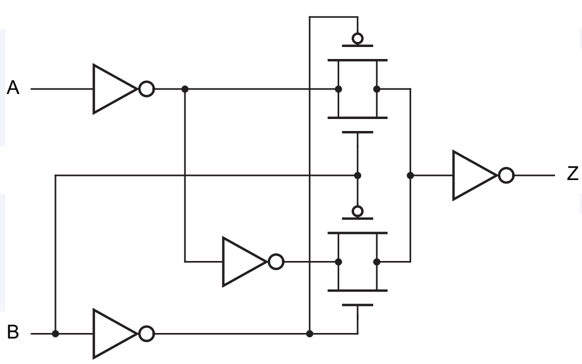


Figure X5.73

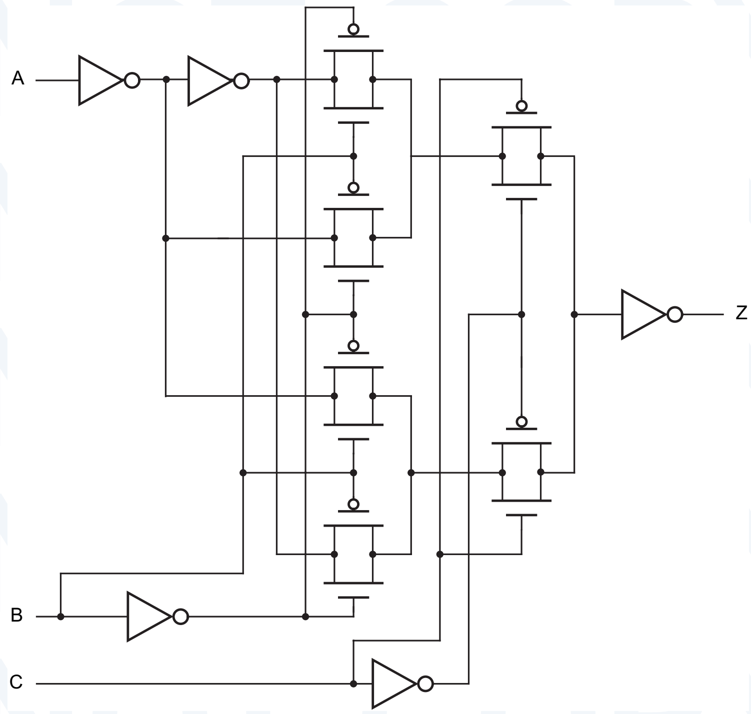


Figure X5.75

5.74 An odd-parity circuit with 2^n inputs can be built with 2^n-1 XOR gates. Describe two different structures for this circuit, one of which gives a minimum worst-case input to output propagation delay and the other of which gives a maximum. For each structure, state the worst-case number of XOR-gate delays, and describe a situation where that structure might be preferred over the other.

5.75 Write the truth table and a logic diagram for the logic function performed by the CMOS circuit in Figure X5.75.

5.76 Write a 4-step iterative algorithm corresponding to the iterative comparator circuit of Figure 5-79.

5.77 Design a 16-bit comparator using five 74x85s in a tree-like structure, such that the maximum delay for a comparison equals twice the delay of one 74x85.

5.78 Starting with a manufacturer's logic diagram for the 74x85, write a logic expression for the ALTBOUT output, and prove that it algebraically equals the expression derived in Drill 5.27.

5.79 Design a comparator similar to the 74x85 that uses the opposite cascading order. That is, to perform a 12-bit comparison, the cascading outputs of the high-order comparator would drive the cascading inputs of the mid-order comparator, and the mid-order outputs would drive the low-order inputs. You needn't do a complete logic design and schematic; a truth table and an application note showing the interconnection for a 12-bit comparison are sufficient.

5.80 Design a 24-bit comparator using three 74x682s and additional gates as required. Your circuit should compare two 24-bit unsigned numbers P and Q and produce two output bits that indicate whether P = Q or P > Q.

5.81 Draw a 6-variable Karnaugh map for the s_2 function of Drill 5.29, and find all of its prime implicants. Using the 6-variable map format of Exercise 4.66, label the variables in the order x_0, y_0, x_2, y_2, x_1, y_1 instead of U, V, W, X, Y, Z. You need not write out the algebraic product corresponding to each prime implicant; simply identify each one with a number (1, 2, 3, …) on the map. Then make a list that shows for each prime implicant whether or not it is essential and how many inputs are needed on the corresponding AND gate.

5.82 Starting with the logic diagram for the 74x283 in Figure 5-90, write a logic expression for the S2 output in terms of the inputs, and prove that it does indeed equal the third sum bit in a binary addition as advertised. You may assume that c_0 = 0 (i.e., ignore c_0).

5.83 Using the information in Table 5-3, determine the maximum propagation delay from any A or B bus input to any F bus output of the 16-bit carry lookahead adder of Figure 5-95. You may use the "worst-case" analysis method.

5.84 Referring to the data sheet of a 74S182 carry lookahead circuit, determine whether or not its outputs match the equations given in Section 5.10.7.

5.85 Estimate the number of product terms in a minimal sum-of-products expression for the c_{32} output of a 32-bit binary adder. Be more specific than "billions and billions," and justify your answer.

5.86 Draw the logic diagram for a 64-bit ALU using sixteen 74x181s and five 74S182s for full carry lookahead (two levels of '182s). For the '181s, you need only show the CIN inputs and G_L and P_L outputs.

5.87 Show how to build all four of the following functions using one SSI package and one 74x138.

$$F1 = X' \cdot Y' \cdot Z' + X \cdot Y \cdot Z \qquad F2 = X' \cdot Y' \cdot Z + X \cdot Y \cdot Z'$$

$$F3 = X' \cdot Y \cdot Z' + X \cdot Y' \cdot Z \qquad F4 = X \cdot Y' \cdot Z' + X' \cdot Y \cdot Z$$

5.88 Design a customized decoder with the function table in Table X5.88 using MSI and SSI parts. Minimize the number of IC packages in your design.

Table X5.88

CS_L	A2	A1	A0	*Output to assert*
1	x	x	x	none
0	0	0	x	BILL_L
0	0	x	0	MARY_L
0	0	1	x	JOAN_L
0	0	x	1	PAUL_L
0	1	0	x	ANNA_L
0	1	x	0	FRED_L
0	1	1	x	DAVE_L
0	1	x	1	KATE_L

5.89 Repeat Exercise 5.88 using ABEL and a single GAL16V8.

5.90 Repeat Exercise 5.88 using VHDL.

5.91 Using ABEL and a single GAL16V8, design a customized multiplexer with four 3-bit input buses P, Q, R, T, and three select inputs S2–S0 that choose one of the buses to drive a 3-bit output bus Y according to Table X5.91.

Table X5.91

S2	S1	S0	*Input to select*
0	0	0	P
0	0	1	P
0	1	0	P
0	1	1	Q
1	0	0	P
1	0	1	P
1	1	0	R
1	1	1	T

5.92 Design a customized multiplexer with four 8-bit input buses P, Q, R, and T, selecting one of the buses to drive a 8-bit output bus Y according to Table X5.91. Use two 74x153s and a code converter that maps the eight possible values on S2–S0 to four select codes for the '153. Choose a code that minimizes the size and propagation delay of the code converter.

5.93 Design a customized multiplexer with five 4-bit input buses A, B, C, D, and E, selecting one of the buses to drive a 4-bit output bus T according to Table X5.93. You may use no more than three MSI and SSI ICs.

Table X5.93

S2	S1	S0	*Input to select*
0	0	0	A
0	0	1	B
0	1	0	A
0	1	1	C
1	0	0	A
1	0	1	D
1	1	0	A
1	1	1	E

5.94 Repeat Exercise 5.93 using ABEL and one or more PAL/GAL devices from this chapter. Minimize the number and size of the GAL devices.

5.95 Design a 3-bit equality checker with six inputs, SLOT[2–0] and GRANT[2–0], and one active-low output, MATCH_L. The SLOT inputs are connected to fixed values when the circuit installed in the system, but the GRANT values are changed on a cycle-by-cycle basis during normal operation of the system. Using only SSI and MSI parts that appear in Tables 5-2 and 5-3, design a comparator with the shortest possible maximum propagation delay from GRANT[2–0] to MATCH_L. (*Note:*

The author had to solve this problem "in real life" to shave 2 ns off the critical-path delay in a 25-MHz system design.)

5.96 Design a combinational circuit whose inputs are two 8-bit unsigned binary integers, X and Y, and a control signal MIN/MAX. The output of the circuit is an 8-bit unsigned binary integer Z such that Z = 0 if X = Y; otherwise, Z = min(X,Y) if MIN/MAX = 1, and Z = max(X,Y) if MIN/MAX = 0.

5.97 Design a combinational circuit whose inputs are two 8-bit unsigned binary integers, X and Y, and whose output is an 8-bit unsigned binary integer Z = max(X,Y). For this exercise, you may use any of the 74x SSI and MSI components introduced in this chapter *except* the 74x682.