bit p) is even. For *odd parity* the p bit is given the value that makes the total number of 1s odd. The sender generates the p bit based on the n-bit data item that is to be transmitted. The receiver checks whether the parity of the received item is correct.

Parity generating and checking circuits can be realized with XOR gates. For example, for a four-bit data item consisting of bits $x_3x_2x_1x_0$, the even parity bit can be generated as

$$p = x_3 \oplus x_2 \oplus x_1 \oplus x_0$$

At the receiving end the checking is done using

$$c = p \oplus x_3 \oplus x_2 \oplus x_1 \oplus x_0$$

If $c = 0$, then the received item shows the correct parity. If $c = 1$, then an error has occurred. Note that observing $c = 0$ is not a guarantee that the received item is correct. If two or any even number of bits have their values inverted during the transmission, the parity of the data item will not be changed; hence the error will not be detected. But if an odd number of bits are corrupted, then the error will be detected.

The attractiveness of parity checking lies in its simplicity. There exist other more sophisticated schemes that provide more reliable error-checking mechanisms [4]. We will discuss parity circuits again in section 9.3.

PROBLEMS

5.1 Determine the decimal values of the following unsigned numbers:
(a) $(0111011110)_2$
(b) $(1011100111)_2$
(c) $(3751)_8$
(d) $(A25F)_{16}$
(e) $(F0F0)_{16}$

5.2 Determine the decimal values of the following 1's complement numbers:
(a) 0111011110
(b) 1011100111
(c) 1111111110

5.3 Determine the decimal values of the following 2's complement numbers:
(a) 0111011110
(b) 1011100111
(c) 1111111110

5.4 Convert the decimal numbers 73, 1906, −95, and −1630 into signed 12-bit numbers in the following representations:
(a) Sign and magnitude
(b) 1's complement
(c) 2's complement

5.5 Perform the following operations involving eight-bit 2's complement numbers and indicate whether arithmetic overflow occurs. Check your answers by converting to decimal sign-and-magnitude representation.

00110110	01110101	11011111
+01000101	+11011110	+10111000
00110110	01110101	11010011
−00101011	−11010110	−11101100

5.6 Prove that the XOR operation is associative, which means that $x_i \oplus (y_i \oplus z_i) = (x_i \oplus y_i) \oplus z_i$.

5.7 Show that the circuit in Figure 5.5 implements the full-adder specified in Figure 5.4*a*.

5.8 Prove the validity of the simple rule for finding the 2's complement of a number, which was presented in section 5.3. Recall that the rule states that scanning a number from right to left, all 0s and the first 1 are copied; then all remaining bits are complemented.

5.9 Prove the validity of the expression Overflow = $c_n \oplus c_{n-1}$ for addition of n-bit signed numbers.

5.10 In section 5.5.4 we stated that a carry-out signal, c_k, from bit position $k-1$ of an adder circuit can be generated as $c_k = x_k \oplus y_k \oplus s_k$, where x_k and y_k are inputs and s_k is the sum bit. Verify the correctness of this statement.

5.11 Consider the circuit in Figure P5.1. Can this circuit be used as one stage in a carry-ripple adder? Discuss the pros and cons.

5.12 Determine the number of gates needed to implement an n-bit carry-lookahead adder, assuming no fan-in constraints. Use AND, OR, and XOR gates with any number of inputs.

5.13 Determine the number of gates needed to implement an eight-bit carry-lookahead adder assuming that the maximum fan-in for the gates is four.

5.14 In Figure 5.18 we presented the structure of a hierarchical carry-lookahead adder. Show the complete circuit for a four-bit version of this adder, built using 2 two-bit blocks.

5.15 What is the critical delay path in the multiplier in Figure 5.33? What is the delay along this path in terms of the number of gates?

5.16 (a) Write a Verilog module to describe the circuit block in Figure 5.36*b*. Use the CAD tools to synthesize a circuit from the code and verify its functional correctness.
(b) Write a Verilog module to describe the circuit block in Figure 5.36*c*. Use the CAD tools to synthesize a circuit from the code and verify its functional correctness.
(c) Write a Verilog module to describe the 4×4 multiplier shown in Figure 5.36*a*. Your code should be hierarchical and should use the subcircuits designed in parts (a) and (b). Synthesize a circuit from the code and verify its functional correctness.

5.17 Consider the Verilog code in Figure P5.2. Given the relationship between the signals IN and OUT, what is the functionality of the circuit described by the code? Comment on whether or not this code represents a good style to use for the functionality that it represents.

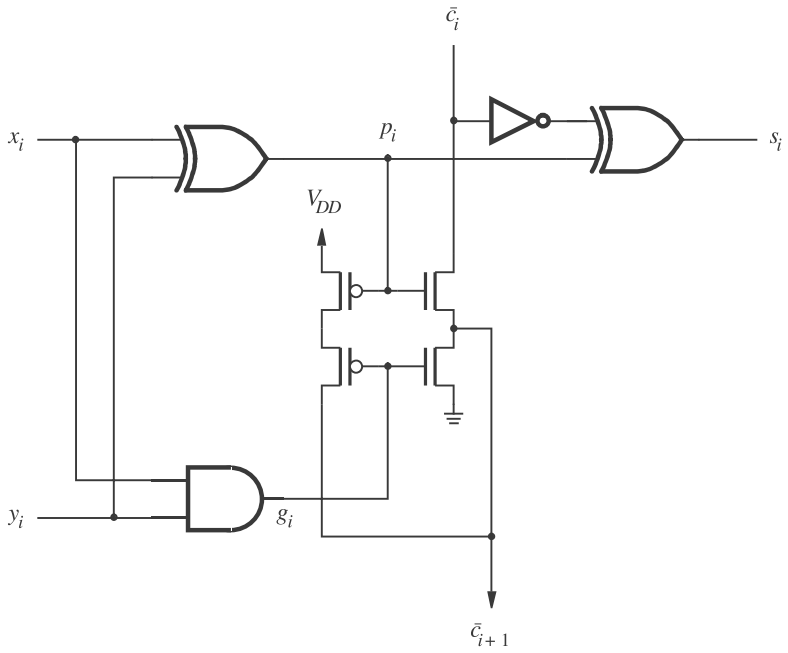

Figure P5.1 Circuit for problem 5.11.

```
module problem5_17 (IN, OUT);
   input [3:0] IN;
   output [3:0] OUT;
   reg [3:0] OUT;

   always @(IN)
      if (IN == 4'b0101)  OUT = 4'b0001;
      else if (IN == 4'b0110)  OUT = 4'b0010;
      else if (IN == 4'b0111)  OUT = 4'b0011;
      else if (IN == 4'b1001)  OUT = 4'b0010;
      else if (IN == 4'b1010)  OUT = 4'b0100;
      else if (IN == 4'b1011)  OUT = 4'b0110;
      else if (IN == 4'b1101)  OUT = 4'b0011;
      else if (IN == 4'b1110)  OUT = 4'b0110;
      else if (IN == 4'b1111)  OUT = 4'b1001;
      else  OUT = 4'b0000;

endmodule
```

Figure P5.2 The code for problem 5.17.

5.18 Design a circuit that generates the 9's complement of a BCD digit. Note that the 9's complement of d is $9 - d$.

5.19 Derive a scheme for performing subtraction using BCD operands. Show a block diagram for the subtractor circuit.
Hint: Subtraction can be performed easily if the operands are in the 10's complement (radix complement) representation. In this representation the sign digit is 0 for a positive number and 9 for a negative number.

5.20 Write complete Verilog code for the circuit that you derived in problem 5.19.

5.21 Suppose that we want to determine how many of the bits in a three-bit unsigned number are equal to 1. Design the simplest circuit that can accomplish this task.

5.22 Repeat problem 5.21 for a six-bit unsigned number.

5.23 Repeat problem 5.21 for an eight-bit unsigned number.

5.24 Show a graphical interpretation of three-digit decimal numbers, similar to Figure 5.12. The left-most digit is 0 for positive numbers and 9 for negative numbers. Verify the validity of your answer by trying a few examples of addition and subtraction.

5.25 In a ternary number system there are three digits: 0, 1, and 2. Figure P5.3 defines a ternary half-adder. Design a circuit that implements this half-adder using binary-encoded signals, such that two bits are used for each ternary digit. Let $A = a_1a_0$, $B = b_1b_0$, and $Sum = s_1s_0$; note that *Carry* is just a binary signal. Use the following encoding: $00 = (0)_3$, $01 = (1)_3$, and $10 = (2)_3$. Minimize the cost of the circuit.

A B	*Carry*	*Sum*
0 0	0	0
0 1	0	1
0 2	0	2
1 0	0	1
1 1	0	2
1 2	1	0
2 0	0	2
2 1	1	0
2 2	1	1

Figure P5.3 Ternary half-adder.

5.26 Design a ternary full-adder circuit, using the approach described in problem 5.25.

5.27 Consider the subtractions $26 - 27 = 99$ and $18 - 34 = 84$. Using the concepts presented in section 5.3.4, explain how these answers (99 and 84) can be interpreted as the correct signed results of these subtractions.

REFERENCES

1. V. C. Hamacher, Z. G. Vranesic and S. G. Zaky, *Computer Organization*, 5th ed. (McGraw-Hill: New York, 2002).
2. D. A. Patterson and J. L. Hennessy, *Computer Organization and Design—The Hardware/Software Interface*, 2nd ed. (Morgan Kaufmann: San Francisco, CA, 1998).
3. Institute of Electrical and Electronic Engineers (IEEE), "A Proposed Standard for Floating-Point Arithmetic," *Computer* 14, no. 3 (March 1981), pp. 51–62.
4. W. W. Peterson and E. J. Weldon Jr., *Error-Correcting Codes*, 2nd ed. (MIT Press: Boston, MA, 1972).