The code in Figure 3.26 defines a circuit that generates the n sum bits, but it does not include carry-out or overflow signals. As explained previously, the carry-out signal is useful when the numbers X, Y, and S are interpreted as being unsigned numbers. The carry-out is 1 when the sum of unsigned numbers overflows n bits. But, if X, Y, and S are interpreted as being signed numbers, then the carry-out is not meaningful, and we have to generate the arithmetic overflow output as discussed in Section 3.3.5. One way in which these signals can be generated is given in Figure 3.27.

The carry-out from the MSB position, $n-1$, can be derived by observing that the carry-out must be 1 if both x_{n-1} and y_{n-1} are 1, or if either x_{n-1} or y_{n-1} is 1 and s_{n-1} is 0.

```
module addern (carryin, X, Y, S);
   parameter n = 32;
   input carryin;
   input [n-1:0] X, Y;
   output reg [n-1:0] S;

   always @(X, Y, carryin)
      S = X + Y + carryin;

endmodule
```

Figure 3.26 Specification of an n-bit adder using arithmetic assignment.

```
module addern (carryin, X, Y, S, carryout, overflow);
   parameter n = 32;
   input carryin;
   input [n-1:0] X, Y;
   output reg [n-1:0] S;
   output reg carryout, overflow;

   always @(X, Y, carryin)
   begin
      S = X + Y + carryin;
      carryout = (X[n-1] & Y[n-1]) | (X[n-1] & ~S[n-1]) | (Y[n-1] & ~S[n-1]);
      overflow = (X[n-1] & Y[n-1] & ~S[n-1]) | (~X[n-1] & ~Y[n-1] & S[n-1]);
   end

endmodule
```

Figure 3.27 An n-bit adder with carry-out and overflow signals.

```
module seg7 (hex, leds);
  input [3:0] hex;
  output reg [1:7] leds;

  always @(hex)
    case (hex)   //abcdefg
      0: leds = 7'b1111110;
      1: leds = 7'b0110000;
      2: leds = 7'b1101101;
      3: leds = 7'b1111001;
      4: leds = 7'b0110011;
      5: leds = 7'b1011011;
      6: leds = 7'b1011111;
      7: leds = 7'b1110000;
      8: leds = 7'b1111111;
      9: leds = 7'b1111011;
      10: leds = 7'b1110111;
      11: leds = 7'b0011111;
      12: leds = 7'b1001110;
      13: leds = 7'b0111101;
      14: leds = 7'b1001111;
      15: leds = 7'b1000111;
    endcase

endmodule
```

Figure 4.34 Code for a hex-to-7-segment decoder.

Table 4.1 The functionality of the 74381 ALU.

Operation	Inputs $s_2\ s_1\ s_0$	Outputs F
Clear	0 0 0	0 0 0 0
B–A	0 0 1	$B - A$
A–B	0 1 0	$A - B$
ADD	0 1 1	$A + B$
XOR	1 0 0	A XOR B
OR	1 0 1	A OR B
AND	1 1 0	A AND B