Corrections for *Digital Systems Design Using VHDL*  

Look on the back of the title page in your copy of the textbook. If there is a line that reads:  

00 01 -- 10 9 8 7 6 5 you have the 5th printing of the text, and you should use the errata sheet given below. If there is a line that reads 00 01 -- 10 9 8 7 6 5 4 you have the 4th printing of the text, etc. If you find any additional errors in the 5th printing, please report them to Dr. Roth: roth@ece.utexas.edu and receive a $5 reward (except for formatting and other trivial errors). Corrected VHDL code is at [http://www.ece.utexas.edu/~roth/VHDL_book/vhdl_code/](http://www.ece.utexas.edu/~roth/VHDL_book/vhdl_code/)

line –7 means 7th line from bottom, etc.

**Corrections for *Digital Systems Design Using VHDL*, 4th printing**  
(corrections marked * also apply to the 5th printing)

- p. 12, Figure 1-8 (c): B’ should be B
- p. 27, Figures 1-26(b) and (c), lower right square: e-f should be c-f
- p. 93, 4th line above Figure 3-9: F_3 should be F_2
- *p. 106, Figure 3-20 should be *
- p. 111, last line: C_1 = C_2 = C_3 should be C_0 = C_1 = C_2
- *p. 132, lines 1 and 2: 2n should be (3n – 4)
- p. 147, line 4: 4 should be 5
- *p. 168, line 11: add Done <= '0';
- p. 182, line –13: Figure 5-17 should be Figure 5-19
- p. 190, line 8: Figure 5-28 should be Figure 5-29
- p. 213, line 12: Figure 5-32 should be Figure 5-33
- p. 213, line –3: S_0 should be S_1
- *p. 254, Figure 7-8: LZ should be L_2, and LFZ should be L_2
- p. 286, line 9: std_logic vectors should be std_logic
- p. 290, Figure 8-18, 6th line: 0 to 7 should be 0 to 6
- p. 322, 8th line of Figure 9-21: t_{11_{max}} should be t_{11}
- p. 323, lines 9 and 14: t_{10_{min}} should be t_{11}
- p. 368, Figure 10-29: interchange connections to B_1 and B_2 as follows:

![Diagram](http://example.com/diagram.png)

- p. 368, line 2 after Figure 10-29: Figure 8-12 should be Figure 8-13

8/30/03
Corrections for *Digital Systems Design Using VHDL, 3rd printing*

p. 12, Figure 1-8 (c): B' should be B
p. 27, Figures 1-26(b) and (c), lower right square: e-f should be c-f
p. 93, 4th line above Figure 3-9: F_3 should be F_2
p. 111, last line: C_1 = C_2 = C_3 should be C_0 = C_1 = C_2
p. 112, Figure 3-25, replace with the following:

![Diagram](image)

p. 132, lines 1 and 2: 2n should be (3n – 4)

p. 147, line 4: 4 should be 5

p. 168, line 11: add Done <= '0';

p. 182, line –13: Figure 5-17 should be Figure 5-19

p. 190, line 8: Figure 5-28 should be Figure 5-29

p. 213, line 12: Figure 5-32 should be Figure 5-33

p. 213, line –3: S0 should be S1

p. 226, line –15: interchange G and F

p. 259, Fig. 7-13, lower right, add label to xor gate output: S4

p. 262, line 12: are are should be are

p. 286, line 9: std_logic vectors should be std_logic

p. 290, Figure 8-18, 6th line: 0 to 7 should be 0 to 6

p. 292, lines 5-7, replace "added Nextstate <= '0'; ... unwanted latch." with "changed the if statements so that Nextstate is always specified even if there is no change of state. For example, in the fifth line of the first process, we added else Nextstate <= 0; before end if, and we added Nextstate <= 1; at the end of the seventh line."

p. 311, line 24, add: next_state <= 3;

p. 322, 8th line of Figure 9-21: t11_max should be t11

p. 323, line 14: t10_min should be t11

p. 324, line 7: now /= 0 ns should be state = T2

p. 324, line –7 should read: where dbus'last_event is the time since the
p. 324, line –10: when the chip is selected and now = 0 should be in state T2
p. 335, problem 9.2(b), last line: unknown should be the value just stored
p. 368, Figure 10-29: interchange connections to B1 and B2 as follows:

```
<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
```

B1  B2

p. 384, line –15: clk_divdr should be ctr2
p. 385, line 7: RDRE should be RDRF
p. 392, line –4: Table 11-1(b) should be Table 11-2(b)

p. 414, line –15: clk_divdr should be ctr2
p. 439, line 23: delete C <= '0';

p. 441, line 20: OP=JMP should be (OP=JMP and reg_mem)

p. 447, line 1, add before end if: selMd2 <= '0';

p. 449, line –13: JMP='1' should be (JMP='1' and reg_mem)

p. 449, line –11: JSR='1' should be (JSR='1' and reg_mem)

p. 451, add after line 4: end process;

---

**Corrections for Digital Systems Design Using VHDL, 1st and 2nd printing**

p. 9, line 18: "ABC' or AC'D. Thus one" should be "AB. Thus the"

p. 9, line 20: ABC' should be AB

p. 9, line 22: (A' + B' + C) should be (A' + B')

p. 12, Figure 1-8 (c): B' should be B

p. 18, Figure 1-19, map for Z, within loop on right: change 0 to 1

p. 26, line 18: e ≠ f should be a ≠ g ; line 19: e–f should be e–g

p. 27, Figures 1-26(a) and (c), lower right square: e-f should be c-f

p. 41, top figure: insert > in front of CK

p. 53, line 13: -- see Note 3 should be moved right to align with other comments

p. 55, first line after Figure 2-9: conditional assignment should be conditional signal assignment

p. 55, lines 11 & 12 should be: A behavioral model of the 4-to-1 MUX of Figure 2-9 using a conditional signal assignment statement is

p. 67, line –6: vlaues should be values

p. 81, Problem 2.4, line –7: A <= '1' should be A = '1'

p. 82, Problem 2.9(b): Z should be D and B

p. 82, Prob. 2.10(a): Problem 1.12 should be Problem 1.10;

p. 82, Prob. 2-10(b): Z is should be S and V are

p. 89, Figure 3-3, line 6: omit blank line above end ROM1_2;

p. 93, 4th line above Figure 3-9: F_3 should be F_2

p. 105, Figure 3-19, line 3: out should be inout

---

8/30/03
p. 106, line 9: 30 should be 20
p. 110, Figure 3-22: add horizontal line at top of figure
p. 111, last line: \( C_1 = C_2 = C_3 \) should be \( C_0 = C_1 = C_2 \)
p. 112, Figure 3-25, replace with the following:

```
S_0\quad S_1\quad S_2\quad S_3\quad S_4
\begin{array}{c}
0 \\
C_0C_1C_2 \\
Kd \\
1 \\
Kd'
\end{array} \quad \begin{array}{c}
Kd \quad K' \quad K' \\
K/V \\
K/V \\
1/V \\
Kd'
\end{array} \quad \begin{array}{c}
S_5 \quad S_4 \quad S_3 \quad S_2 \quad S_1 \\
C_0C_1C_2 \\
Kd \\
K/V \\
K/V \\
Kd'
\end{array}
```

p. 115, 10th line above Figure 3-27, near end of line, q3 should be Q3.
p. 120, Problem 3-10(c): an SM chart should be a state graph
p. 132, lines 1 and 2: 2n should be \( (3n - 4) \)
p. 133, line -5: move 1.110001 right to align decimal point with line above
p. 136, Figure 4-11: add a self-loop labeled St'0 to state S0; line -2: use should be in bold face type
p. 137, line -10: move "--output product" to end of previous line
p. 140, line -2: delete "part of the"
p. 140, line -1: replace "After" with "In the second process, after"
p. 141-142, Figure 4-16: Replace with attached copy. (Although the code in the book is correct, it is not consistent with the test bench of Figure 4-14. It is better to use a two-process model with one process for the combinational logic and one for the register updates.)
p. 143, Figure 4-18, add before begin: \textbf{alias} M: bit is B(0);
p. 145, first row of boxes below figure: add a 1 in a box at the right end
p. 145, at bottom of page: move dashed line one place to the right
p. 146, line 4: move 1 1 0 1 one place to the left
p. 147, line 4: 4 should be 5
p. 150, Figure 4-22, two occurrences of: Co1 should be Cm1
p. 154, line -3: multiplier should be divider
p. 157, problem 4.8(a): delete last sentence. Add: Use a serial adder to add the multiplicand to the accumulator.
p. 158, Problem 4-13(b): 3 states should be 5 states; 4-13(c): 4-5 should be 4-12
p. 158, Problem 4.13 (d): 4-12 should be 4-16
p. 164, Figure 5-4(b): add horizontal line at top of figure
p. 165, line -7: 10 should be 11
p. 168, line 11: add \( \text{Done} <= '0'; \)
p. 179, line -2: A'BM' should be A'BM
p. 182, line -13: Figure 5-17 should be Figure 5-19
p. 190, line 8: Figure 5-28 should be Figure 5-29
p. 194, Problem 5.3: add 0 to right of X_3 diamond, add 1 below X_3 diamond
leftmost output box: \( Z_1 \) should be \( Z_3 \)

8/30/03
p. 324, line –10: when the chip is selected and now = 0 should be in state T2
p. 324, line –8: (now – dbus'event) should be dbus'last_event
p. 324, line –7 should read: where dbus'last_event is the time since the
p. 326, replace Figure 9-23 with the following:

```
486 bus
| 486 bus |
|        |
|        |
|        |
| 486 bus |

CLK

<table>
<thead>
<tr>
<th>Ads</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rdy</td>
</tr>
<tr>
<td>Addr Decoder</td>
</tr>
<tr>
<td>Memory Controller</td>
</tr>
<tr>
<td>Memory</td>
</tr>
<tr>
<td>Data</td>
</tr>
<tr>
<td>tce</td>
</tr>
<tr>
<td>tmc</td>
</tr>
<tr>
<td>tacs</td>
</tr>
<tr>
<td>tAA</td>
</tr>
<tr>
<td>CS</td>
</tr>
<tr>
<td>tL</td>
</tr>
<tr>
<td>tmc</td>
</tr>
</tbody>
</table>

p. 326, line –4 should read: Delay from CLK high to CS low = tcc = 5 ns
p. 327, lines –1, –4, –14: tmc should be tcc
p. 328, Figure 9-25: t6max should be t10max; tmc should be tcc
p. 329, line 10: delete new_we_b, line 14: delete new_we_b <= '1';
p. 331, line 1, end of line 1: delete |
p. 335, Problem 9.1(c): delete "and latest"
p. 335, replace Problem 9.1(d) with: For a write cycle, what is the minimum time that valid data must be driven onto the data bus?
p. 335, problem 9.2(b), last line: unknown should be the value just stored
p. 335, Problem 9.3(a): replace "Add code that" with "Verify that the code"
p. 335, Problem 9.5: Figure 9-5 should be Figure 9-6
p. 338, lower figure: add a vertical line from the small circle down to the RW line
p. 344, lines –8 and –9: k should be k+1
p. 346, line –7: S3 should be S1
p. 349, line 6: shift 010 110 011 111 left to align with column headings
p. 354: replace Figure 10-16 with attached figure
p. 355, top flip-flop in IC1: D1 Q1 should be D0 Q0
p. 357, line 9 of item 6: Update-IR should be Update-DR
p. 363, Table 10-4, for n = 5: delete Q1 ⊕
p. 363, line above Figure 10-24, add: as for Figure 10-23
p. 364, 3rd and 6th lines after Figure 10-25: 1010 should be 0010; 5th line: 1000 should be 0000
p. 367, line 2: 10 should be 01
p. 368, Figure 10-29: interchange connections to B1 and B2 as follows:

```
B1
|
|
|
B2
```
p. 368, line 2 after Figure 10-29: Figure 8-12 should be Figure 8-13
p. 369, line 16, add after bit_vector: (3 downto 0)
p. 376, line above Figure 11-5: add semi-colon at end.
p. 377, line 3: 7 should be 8; line –7: TSRout should be TSR(0); line 12: of should not be bold
p. 384, lines 3 and –9: CONVERT_INT should be CONV_INTEGER
p. 384, line –15: clk_divdr should be ctr2
p. 385, table above Figure 11-12: complement all entries in R_W column
p. 385, Figure 11-12, line 4: add rst_b, after clk,
p. 385, line 7: RDRE should be RDRF
p. 386, line 4: BclkX8, should be BclkX8: buffer std_logic;
p. 386, lines 14, 16, and 18: map should be in bold face type
p. 387, lines 1, 2, 3, 5, 6, and 7: ADDR should be ADDR2
p. 391, Table 11-2(b), line 8: LSR should be LSL
p. 392, line –4: Table 11-1(b) should be Table 11-2(b)
p. 395, Table 11-4, 1st column: INC dir should be INC dirm, INC ix should be INC ixm, INC ix1 should be INC ixm1
p. 396, "JSR ext" row, 4th column: P ← SP – 1 should be SP ← SP – 1
p. 396, "SWI" row, 3rd column: SP v SP – 1 should be SP ← SP – 1
p. 397, line –7: <= should be ←
p. 398, line –3 of Table 11-5: 42 should be 57
p. 398, last line of Table 11-5: 0305 should be 0304 (two places)
p. 403, line 11: assert should be assert
p. 410, line 2: that should be than
p. 414, line –17: insert cs, before cpu_wr; line –4: cs, should be cs1, cs2,
p. 414, line –14: ADDR, should be ADDR0, and line –8: ADDR should be ADDR2
p. 414, Figure 11-24, lines –1 and –2: others should be in bold face type type
p. 415, lines 2, 3, 4, 5, 7, and 8: map should be in bold face type
p. 415, 1st and 5th lines below the VHDL code: 4020 should be 4010
p. 428, line –14: delete ;
p. 439, line 23: delete C <= ‘0’;
p. 441, line 20: OP=JMP should be (OP=JMP and reg_mem)
p. 441, line 22: OP=JSR should be (OP=JSR and reg_mem)
p. 443, delete lines 4 and 5
p. 444, insert after line 11: alias JMP : bit is opd(12); alias JSR : bit is opd(13);
p. 446, line 18: CONVTEGER(TO_INTEGER should be conv_integer(TO_stdlogicvector
p. 447, line 1, add before end if: selMd2 <= '0';
p. 449, line –13: JMP='1' should be (JMP='1' and reg_mem)
p. 449, line –11: JSR='1' should be (JSR='1' and reg_mem)
p. 451, add after line 4: end process;
p. 456, last line of table: 32 should be 132
Figure 4-16 Model for 2's Complement Multiplier with Control Signals

-- This architecture of a 4-bit multiplier for 2's complement numbers
-- uses control signals.

architecture behave2 of mult2Cs is
  signal State, Nextstate: integer range 0 to 5;
  signal A, B: bit_vector(3 downto 0);
  signal AdSh, Sh, Load, Cm: bit;
  signal addout: bit_vector(4 downto 0);
  alias M: bit is B(0);
begin
  process (state, st, M)
  begin
    Load <= '0'; AdSh <= '0'; Sh <= '0'; Cm <= '0'; Done <= '0';
    case State is
      when 0=> -- initial State
        if St='1' then Load <= '1'; Nextstate <= 1; end if;
      when 1 | 2 | 3 => -- "add/shift" State
        if M = '1' then AdSh <= '1';
        else Sh <= '1';
        end if;
        Nextstate <= State + 1;
      when 4 => -- add complement if sign
        if M = '1' then
          Cm <= '1'; AdSh <= '1';
          else Sh <= '1';
        end if;
        Nextstate <= 5;
      when 5 => -- output product
        done <= '1';
        Nextstate <= 0;
    end case;
  end process;

  addout <= add4(A, Mcand, '0') when Cm = '0'
    else add4(A, not Mcand, '1');

  process
  begin
    wait until CLK = '1'; -- executes on rising edge
    if Load = '1' then -- load the multiplier
      A <= "0000";
      B <= Mplier;
    end if;
    if AdSh = '1' then -- Add multiplicand to A and shift
      A <= (Mcand(3) xor Cm) & addout(3 downto 1);
      B <= addout(0) & B(3 downto 1);
    end if;
    if Sh = '1' then
      A <= A(3) & A(3 downto 1);
      B <= A(0) & B(3 downto 1);
    end if;
    State <= Nextstate;
  end process;

  Product <= A(2 downto 0) & B;
end behave2;
Figure 10-16. State machine for TAP controller
Cosmetic Corrections for 1st and 2nd Printing

p. 64, line –20: the word all in BITLIB.Bit_pack.all; should be in bold face type

p. 68, line –11: the word to in (1 to 4, 1 to 3) should be in bold face type

p. 72, line –9: the word range in in range loop should be in bold face type

p. 96, Figure 3-11, line 10: the word to in (1 to 3) should be in bold face type

p. 96, Figure 3-11, line 11: the words to and downto in (0 to 6, 7 downto 0) should be in bold face type

p. 117, line 7: the word to in (0 to 11) should be in bold face type

p. 117, line 10: the word downto in (3 downto 0) should be in bold face type

p. 127, line –3: the word to in 0 to 9 should be in bold face type

p. 127, Figure 4-5, line 7: the word use should be in bold face type

p. 139, line 9: the word library in library BITLIB; should be in bold face type

p. 151, Figure 4-23, line 2: the word use should be in bold face type

p. 225, Figure 6-26, line 3: BIT_PACK and ALL should be lower case

p. 306, Figure 9-4, CS waveform: move the vertical line associated with tCHZ to the right so it is centered on the sloping part of the waveform

p. 310, line –3: the word all in bitlib.bit_pack.all; should be in bold face type

p. 310, line –5: the word all in ieee.std_logic_1164.all; should be in bold face type

p. 312, line –7: the word all in bitlib.bit_pack.all; should be in bold face type

p. 312, line –9: the word all in ieee.std_logic_1164.all; should be in bold face type

p. 400, line –12: is should be in bold face type

p. 401, line 2: the word procedure in procedure ALU_OP should be in bold face type