Medical Electronics IV – Part II
Sheet #2
April 2008

---

1. Given signal sources **a**, **b**, and **c**; selection lines **s0**, **s1**, and **s2**; and a desired output **out** write a VHDL code to implement the following truth table:

s0	s1	s2	out
0	0	0	a
1	0	0	b
0	1	0	c
1	1	0	a AND b
0	0	1	b AND c
1	0	1	a AND c
0	1	1	a OR b OR c
1	1	1	(a AND b) OR C

---

2. Given signal sources **a**, **b**, and **c**; selection lines **s0**, **s1**, and **s2**; and a desired output **out** write a VHDL code to implement the following logic expression:

Out = (a AND s0) OR (b AND s1) OR (c AND s2)

---

3. Repeat Problems 1 and 2 using sequential model.

---

4. What does the following VHDL code do? Comment on the output and suggest a correction if anything is wrong.

```
USE WORK.std_logic_1164.ALL;
ENTITY mux IS
PORT (i0, i1, i2, i3, a, b: IN std_logic;
     q : OUT std_logic);
END mux;

ARCHITECTURE test OF mux IS
BEGIN
     q <= i0 WHEN a = '0' AND b = '0' ELSE '0';
     q <= i1 WHEN a = '1' AND b = '0' ELSE '0';
     q <= i2 WHEN a = '0' AND b = '1' ELSE '0';
     q <= i3 WHEN a = '1' AND b = '1' ELSE '0';
END TEST;
```

---

5. What does the following VHDL code do? Comment on the output and suggest a correction if anything is wrong.

```
LIBRARY IEEE;
```

```
USE IEEE.std_logic_1164.ALL;
ENTITY mux4 IS
PORT ( i0, i1, i2, i3, a, b : IN std_logic;
       q : OUT std_logic);
END mux4;

ARCHITECTURE mux4 OF mux4 IS
SIGNAL sel: INTEGER;
BEGIN
WITH sel SELECT
     q <= i0 AFTER 10 ns WHEN 0,
     q <= i1 AFTER 10 ns WHEN 1,
     q <= i2 AFTER 10 ns WHEN 2,
     q <= i3 AFTER 10 ns WHEN 3,
     q <= ‘X’ AFTER 10 ns WHEN OTHERS;

sel <= 0 WHEN a = ‘0’ AND b = ‘0’ ELSE
       1 WHEN a = ‘1’ AND b = ‘0’ ELSE
       2 WHEN a = ‘0’ AND b = ‘1’ ELSE
       3 WHEN a = ‘1’ AND b = ‘1’ ELSE
       4 ;
END mux4;
```

---

6. Write a VHDL code to model an up counter with the following features:
   a. One input: clock **`CLK1`**
   b. Two outputs: 2 bit count **`C1`** and **`C2`**
   c. Counts on the leading edge of the clock only
   d. Zero initial values for **`C1`** and **`C2`**
   e. Runs until **`C1=’1’`** and **`C2=’1’`** then stops