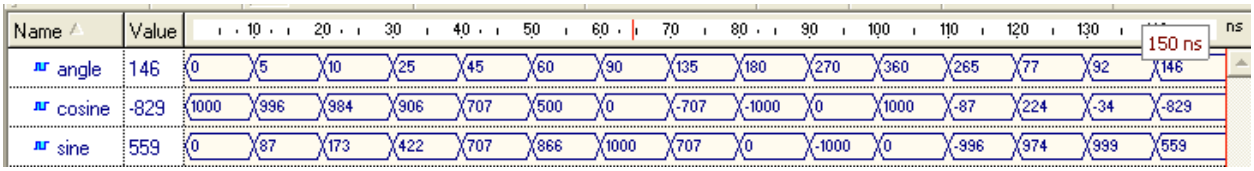
Question #1

```
LIBRARY ieee; USE ieee.std_logic_1164.all;  use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity sine_cosine_block is
port ( angle : in integer range 0 to 360;
cosine : out integer range -1000 to 1000;
sine : out integer range -1000 to 1000);
end entity;

architecture myarch of sine_cosine_block is

procedure return_sin_cosine (
signal angle : in integer;
signal cosine: out integer range -1000 to 1000;
signal sine  : out integer range -1000 to 1000) is

type ROM is array (0 to 90) of integer;
constant cosine_table : ROM :=
(1000,999,999,998,997,996,994,992,990,987,984,981,978,974,970,965,961,956,951,945,939,933,927,
920,913,906,898,891,882,874,866,857,848,838,829,819,809,798,788,777,766,754,743,731,719,707,69
4,681,669,656,642,629,615,601,587,573,559,544,529,515,500,484,469,453,438,422,406,390,374,358,
342,325,309,292,275,258,241,224,207,190,173,156,139,121,104,87,69,52,34,17,0);
constant sine_table : ROM :=
(0,17,34,52,69,87,104,121,139,156,173,190,207,224,241,258,275,292,309,325,342,358,374,390,406,
422,438,453,469,484,499,515,529,544,559,573,587,601,615,629,642,656,669,681,694,707,719,731,74
3,754,766,777,788,798,809,819,829,838,848,857,866,874,882,891,898,906,913,920,927,933,939,945,
951,956,961,965,970,974,978,981,984,987,990,992,994,996,997,998,999,999,1000);
variable cosine_var : integer range -1000 to 1000;
variable sine_var : integer range -1000 to 1000;
begin
          if (angle > 90 and angle <= 180) then
                    cosine_var := -1 * cosine_table(180-angle);
                    sine_var :=  sine_table(180-angle);
          elsif (angle > 180 and angle <= 270) then
                    cosine_var := -1 * cosine_table(-(180-angle));
                    sine_var := -1 * sine_table(-(180-angle));
          elsif (angle > 270 and angle <= 360) then
                    cosine_var := cosine_table(-1*(360-angle));
                    sine_var := -1*sine_table(-1*(360-angle));
          else
                    cosine_var := cosine_table(angle);
                    sine_var := sine_table(angle);
          end if;

          sine <= sine_var;
          cosine <=cosine_var;

end procedure;

begin

          return_sin_cosine(angle,cosine,sine);
end architecture;
```

Test-bench




```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
-- extra package included
USE ieee.std_logic_unsigned.all;

entity test_q1 is
end;

architecture bench of test_q1 is
component sine_cosine_block
        port ( angle : in integer range 0 to 360;
        cosine : out integer range -1000 to 1000;
        sine : out integer range -1000 to 1000);
  end component;

  signal angle  : integer range 0 to 360;
  signal cosine : integer range -1000 to 1000;
  signal sine   : integer range -1000 to 1000;

begin
 angle  <= 0,
 5 after 10 ns,
 10 after 20 ns,
 25 after 30 ns,
 45 after 40 ns,
 60 after 50 ns,
 90 after 60 ns,
 135 after 70 ns,
 180 after 80 ns,
 270 after 90 ns,
 360 after 100 ns,
 265 after 110 ns,
 77 after 120 ns,
 92 after 130 ns,
 146 after 140 ns;

    m: sine_cosine_block port map (angle, cosine, sine);

end bench;
```

Question #2

Main program

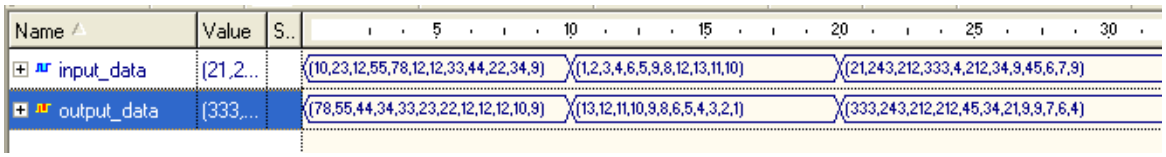
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE work.final_exam_q2.all;

entity topLevel is
        port (  input_data : in matrix;
                        output_data: out matrix);
end entity;

architecture STRUCTURE of topLevel is

begin

output_data <= sort_bus (input_data);

end architecture;
```

Package

```
LIBRARY ieee; USE ieee.std_logic_1164.all;

PACKAGE final_exam_q2 IS
        constant size : integer :=12;
        type matrix is array (size-1 downto 0) of integer;
        function sort_bus (signal x: matrix) return matrix;
END PACKAGE;

package body final_exam_q2 IS
function sort_bus (signal x: matrix) return matrix
is
        variable y : matrix := x;
        variable tempy : matrix := x;
        begin

                for j in 1 to size-1 loop
                        for i in 1 to size-1 loop
                                if (y(i-1) > y(i)) then
                                        tempy(i):= y(i-1);
                                        tempy(i-1):= y(i);
                                end if;

                                y(i):=tempy(i);
                                y(i-1):=tempy(i-1);
                        end loop;
                end loop;
        return y;
        end function;

end package body;
```

Test-bench

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
USE work.final_exam_q2.all;

entity testadder is
end;

architecture bench of testadder is
        component topLevel is
        port (  input_data : in matrix;
                        output_data: out matrix);
        end component;

  signal input_data, output_data : matrix;


begin
 input_data   <= (10,23,12,55,78,12,12,33,44,22,34,9),
 (1,2,3,4,6,5,9,8,12,13,11,10) after 10ns,
 (21,243,212,333,4,212,34,9,45,6,7,9) after 20ns ;

 m: topLevel port map (input_data,output_data);

end bench;
```



Question 3

```
LIBRARY ieee; USE ieee.std_logic_1164.all;
USE work.traffic_light_package.all;

entity traffic_controller is
          port (reset, clk_1hz : in std_logic;
          SRD_sensor : in       std_logic;
          pedestrian_sensor : in std_logic;
          HWY_light, SRD_light : out color
          );
end entity;

architecture my_architecture of traffic_controller is
type states is (HWYgreen_wait,HWYgreen_loop,HWYyellow, HWYred, SRDyellow);
signal pr_state, nx_state : states;
CONSTANT timeMAX : INTEGER := 60; CONSTANT timeY : INTEGER := 10;
CONSTANT timeSRD : INTEGER := 30; CONSTANT timeHWY : INTEGER := 60;
SIGNAL time_signal : INTEGER RANGE 0 TO timeMAX;

begin
          process(reset,clk_1hz)
          VARIABLE count : INTEGER RANGE 0 TO timeMAX;
          begin
                    if reset='1' then
                              pr_state<=HWYgreen_wait;
                              count:=0;
                     elsif (clk_1hz'event and clk_1hz='1') then
                               count := count + 1;

                                if (count=time_signal) then
                                         pr_state<=nx_state;
                                          count:=0;
                               end if;
                    end if;
          end process;

process (pr_state,pedestrian_sensor,SRD_sensor)
begin
          case pr_state is
                    --transition between HWYgreen_wait to HWYgreen_loop
                    when HWYgreen_wait =>
                              nx_state<=HWYgreen_loop;
                              HWY_light<=green;
                              SRD_light<=red;
                              time_signal <= timeHWY;

                    --transition between HWYgreen_loop to HWYyellow
                    when HWYgreen_loop =>
                              if (pedestrian_sensor='1' or SRD_sensor='1') then
                              nx_state<=HWYyellow;
                              end if;
                              time_signal <= 1;


                    --transition between HWYyellow to HWYred (next state requires a 30 sec wait)
                    when HWYyellow =>
                              nx_state<=HWYred;
                              HWY_light<=yellow;
                              SRD_light<=red;
                              time_signal <= timeY;

                    --transition between HWYred to SRDyellow
                    when HWYred =>
                              nx_state<= SRDyellow;
                              HWY_light<=red;
                              SRD_light<=green;
                              time_signal <= timeSRD;

                    --transition between SRDyellow to HWYgreen_wait (next state requires 60 sec wait)
                    when SRDyellow =>
                              HWY_light<=red;
                              SRD_light<=yellow;
                              nx_state<=HWYgreen_wait;
                              time_signal <= timeY;

                    when others => null;
                    end case;
end process; end architecture;
```

Package

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

PACKAGE traffic_light_package IS
TYPE color is (red, yellow, green);
END PACKAGE;
```

Grading worksheet

(Q1=30, Q2=30, Q3=40)

Exam score:

Your grade for the course is:

Deduction type	Deduction
• Q1 - No optimization (is 360 values necessary?)	5
• Q1 - Procedure missing	5
• Q2 - Function missing	5
• Q2 - Minor bug in sorting	10
• Q2 - Major bug in sorting	25
• Q3 - Major bug in the finite state machine	30
• Q3 - Minor bug in the finite state machine	10

Last updated on December 14, 2011 6:50 PM