**TRIGONOMETRIC CALCULATOR**

by Sinan Yaman and Umur Yavuz

**Description:**

We made a trigonometric calculator that receives a 9-bit input in the range (0-360) which corresponds to (“000000000” – “101101000”) and prints the Sine, Cosine, Tangent or Cotangent value of the input to the seven segment with 4 digits after decimal. Our design has 4 different buttons that are assigned to 4 distinct functions mentioned above.

For each of the functions (Sin, Cos, Tan, Cot) we used a 0 to 360 lookup table to get the output information. Our lookup tables take the output as a 16-bit std\_logic\_vector which corresponds to the 4 digit after the decimal. For example, cos (1) is 0.9998, so lookup table gives a 16-bit binary number which corresponds to 9998. Then the 16-bit binary number is converted to bcd in which first 4 bits is the first number (in the example, it is 8) and second 4 bits is the second number and so forth.

The bcd output is then used in our MAIN vhdl module to print the output in the seven segment. In Main.vhd, for every distinct function we managed the numbers, again using counter with the clock, that will be printed in the left-hand seven segment which is the left-hand side of the decimal number. There are tons of if statements to assign the correct number to the correct input, but we tried our best to make it easy to read with comments. For every increase in counter, we assigned the 8-bit binary to an\_out which decides the position of the seven-segment to be displayed.

*(With counter select sevseg\_signal <=)* statement is used for the 4 digit number that is after the decimal and also for printing 0.\_\_\_\_, 1.\_\_\_\_\_, -1.\_\_\_\_\_ in the left-hand side of the decimal.

(*Case counter is)* statement is used for tangent and cotangent and to print numbers like 57.\_\_\_\_, -28.\_\_\_\_\_ etc.

Finally, an\_out selection is done inside of a process to prevent displaying random nonsense numbers when the button is not pressed (when the enabler is not = ‘1’).

WARNING: We got an error of “design is too big”, so in order to test functions the user needs to change the multiplexer.vhd and generate programming file afterwards. For example, if cosine function is to be tested every statement in if statements should be changed to output\_mux <= passer\_out\_cos; Accordingly, for cot, tan and sin functions passer\_out\_cot, passer\_out\_tan, passert\_out\_sin should be used respectively. We couldn’t do the convert to radian part because of the same problem, but it would be very easy to use the same design to print the numbers if we had enough space. (Also, note that when tangent or cotangent is undefined for a degree, we printed E in the right-most seven-segment).

Following, we are providing the main.vhd, bin2bcd.vhd, multiplexer.vhd, cos.vhd, LUT\_Cos.vhd and Cos.ucf file will be provided respectively. (It is named Cos.ucf however, it actually is the ucf file of the whole project.)

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

USE IEEE.NUMERIC\_STD.ALL;

entity Main is

Port ( input\_main: in STD\_LOGIC\_VECTOR (8 downto 0);

cos\_enabler\_main : in STD\_LOGIC;

sin\_enabler\_main : in STD\_LOGIC;

tan\_enabler\_main : in STD\_LOGIC;

cot\_enabler\_main : in STD\_LOGIC;

rad\_enabler\_main : in STD\_LOGIC;

CLK : in STD\_LOGIC;

an\_out : out STD\_LOGIC\_VECTOR (7 downto 0);

sevseg\_out : out STD\_LOGIC\_VECTOR (7 downto 0));

end Main;

architecture Behavioral of Main is

component Multiplexer is

Port ( input : in STD\_LOGIC\_VECTOR (8 downto 0);

cos\_enabler : in STD\_LOGIC;

sin\_enabler : in STD\_LOGIC;

tan\_enabler : in STD\_LOGIC;

cot\_enabler : in STD\_LOGIC;

rad\_enabler : in STD\_LOGIC;

output\_mux : out STD\_LOGIC\_VECTOR (15 downto 0));

end component;

component bin2bcd is

port(

input: in std\_logic\_vector (15 downto 0);

ones: out std\_logic\_vector (3 downto 0);

tens: out std\_logic\_vector (3 downto 0);

hundreds: out std\_logic\_vector (3 downto 0);

thousands: out std\_logic\_vector (3 downto 0)

);

end component;

signal bin2bcd\_input : STD\_LOGIC\_VECTOR (15 downto 0);

signal sevseg\_signal : STD\_LOGIC\_VECTOR (3 downto 0);

signal ones1 : STD\_LOGIC\_VECTOR (3 downto 0);

signal tens1 : STD\_LOGIC\_VECTOR (3 downto 0);

signal hundreds1 : STD\_LOGIC\_VECTOR (3 downto 0);

signal thousands1 : STD\_LOGIC\_VECTOR (3 downto 0);

signal counter : STD\_LOGIC\_VECTOR (4 downto 0) :="00000";

signal prescaler : std\_logic\_vector (16 downto 0);

signal dummy : std\_logic\_vector (7 downto 0);

begin

M: Multiplexer port map(input\_main,cos\_enabler\_main,sin\_enabler\_main,tan\_enabler\_main,cot\_enabler\_main,rad\_enabler\_main,

bin2bcd\_input);

G: bin2bcd port map(bin2bcd\_input,ones1,tens1,hundreds1,thousands1);

process(CLK)

begin

if(CLK'EVENT and CLK='1') then

if prescaler < "11000011010100000" then -- Clock HZ

prescaler <= prescaler + 1;

else

prescaler <= (others => '0');

if(sin\_enabler\_main = '1') then -- SINE FUNCTION

COUNTER <= COUNTER + 1;

if(COUNTER = "00110") then --

COUNTER <= "00000";--

end if;--

if(input\_main < "0000000010110101") then -- 181

if(COUNTER = "00100") then

COUNTER<= "00000";

end if;

end if;

if(input\_main = "0000000100001110") then -- 270

if(COUNTER = "00011") then

COUNTER <= "00101"; -- print minus 1

end if;

if(COUNTER = "00110") then --

COUNTER <= "00000";--

end if;--

end if;

if(input\_main = "0000000001011010") then -- 90

if(COUNTER<="00011") then

COUNTER<="00101";

end if;

if(COUNTER="00101") then

COUNTER<="00000"; -- print 1

end if; -- end of 1

--if(COUNTER = "00111") then--

--COUNTER <= "00000";--

--end if;--

end if;

end if; -- END OF SIN

if (cot\_enabler\_main = '1') then -- COTANGENT FUNCTION

COUNTER<=COUNTER + 1;

if(input\_main = "00000" or input\_main="10110100" or input\_main="101101000") then -- 0 and 180 and 360 - Tanımsız

COUNTER <= "10100";

end if;

if((input\_main >"101101" and input\_main<="1011010") or (input\_main >"11100001" and input\_main <="100001110")) then

-- for 0. (>45 and <90) or (>225 and <270)

if(COUNTER = "00100") then

COUNTER <= "00000";

end if;

end if;--end of 0.

if((input\_main >"1011010" and input\_main<"10000111") or (input\_main >"100001110" and input\_main <"100111011")) then

-- for -0. (>90 and <135) or (>270 and <315)

if(COUNTER = "00100") then

COUNTER <= "10011";

end if;

end if;--end of -0.

if((input\_main > "11010" and input\_main < "101110") or (input\_main>"11001110" and input\_main<"11100010")) then

-- for 1.(> 26 and < 46) or (>206 and <226)

if(COUNTER="00011") then

COUNTER <= "00101";

end if;

if(COUNTER = "00101") then

COUNTER <= "00000";

end if;

end if; -- end of 1.

if((input\_main > "10000110" and input\_main < "10011010") or (input\_main>"100111010" and input\_main<"101001110")) then

-- for -1.(> 134 and < 154) or (>314 and <334)

if(COUNTER="00011") then

COUNTER <= "00101";

end if;

if(COUNTER = "00101") then

COUNTER <= "10011";

end if;

end if; -- end of -1.

if((input\_main > "10010" and input\_main < "11011") or (input\_main>"11000110" and input\_main<"11001111")) then

-- for 2. (>18 and <27) or (>198 and <207)

if(COUNTER = "00011") then

COUNTER <= "01000";

end if;

if(COUNTER = "01000") then

COUNTER <= "00000";

end if;

end if;--end of 2.

if((input\_main > "10011001" and input\_main < "10100010") or (input\_main>"101001101" and input\_main<"101010110")) then

-- for -2. (>153 and <162) or (>333 and <342)

if(COUNTER = "00011") then

COUNTER <= "01000";

end if;

if(COUNTER = "01000") then

COUNTER <= "10011";

end if;

end if;--end of -2.

if((input\_main > "1110" and input\_main < "10011") or (input\_main>"11000010" and input\_main<"11000111")) then

-- for 3. (>14 and <19) or (>194 and <199)

if(COUNTER = "00011") then

COUNTER <= "01001";

end if;

if(COUNTER = "01001") then

COUNTER <= "00000";

end if;

end if;--end of 3.

if((input\_main > "01101000" and input\_main < "01101101") or (input\_main>"100011100" and input\_main<"100100001")) then

-- for -3. (> 104 and <109) or (>284 and <289)

if(COUNTER = "00011") then

COUNTER <= "01001";

end if;

if(COUNTER = "01001") then

COUNTER <= "10011";

end if;

end if;--end of -3.

if((input\_main > "1011" and input\_main < "1111") or (input\_main>"10111111" and input\_main<"11000011")) then

-- for 4. (>11 and <15) or (>191 and <195)

if(COUNTER = "00011") then

COUNTER <= "01010";

end if;

if(COUNTER = "01010") then

COUNTER <= "00000";

end if;

end if;--end of 4.

if((input\_main > "10100101" and input\_main < "10101001") or (input\_main>"101011001" and input\_main<"101011101")) then

-- for -4. (>165 and <169) or (>345 and <349)

if(COUNTER = "00011") then

COUNTER <= "01010";

end if;

if(COUNTER = "01010") then

COUNTER <= "10011";

end if;

end if;--end of -4.

if((input\_main > "1001" and input\_main < "1100") or (input\_main > "10111101" and input\_main < "11000000")) then

-- for 5. (> 9 and <12) or (>189 and <192)

if(COUNTER = "00011") then

COUNTER <= "01011";

end if;

if(COUNTER = "01011") then

COUNTER <= "00000";

end if;

end if;--end of 5.

if((input\_main > "10101000" and input\_main < "10101011") or (input\_main > "101011100" and input\_main < "101011111")) then

-- for -5. (> 168 and <171) or (>348 and <351)

if(COUNTER = "00011") then

COUNTER <= "01011";

end if;

if(COUNTER = "01011") then

COUNTER <= "10011";

end if;

end if;--end of -5.

if(input\_main="1001" or input\_main = "10111101") then -- for 6. (9,189)

if(COUNTER = "00011") then

COUNTER <= "01100";

end if;

if(COUNTER = "01100") then

COUNTER <= "00000";

end if;

end if;--end of 6.

if(input\_main="10101011" or input\_main = "101011111") then -- for -6. (171,351)

if(COUNTER = "00011") then

COUNTER <= "01100";

end if;

if(COUNTER = "01100") then

COUNTER <= "10011";

end if;

end if;--end of -6.

if(input\_main="1000" or input\_main ="10111100") then -- for 7. (8,188)

if(COUNTER = "00011") then

COUNTER <= "01101";

end if;

if(COUNTER = "01101") then

COUNTER <= "00000";

end if;

end if;--end of 7.

if(input\_main="10101100" or input\_main ="101100000") then -- for -7. (172,352)

if(COUNTER = "00011") then

COUNTER <= "01101";

end if;

if(COUNTER = "01101") then

COUNTER <= "10011";

end if;

end if;--end of -7.

if(input\_main="0111" or input\_main = "10111011") then -- for 8. (7,187)

if(COUNTER = "00011") then

COUNTER <= "01110";

end if;

if(COUNTER = "01110") then

COUNTER <= "00000";

end if;

end if;--end of 8.

if(input\_main="10101101" or input\_main = "101100001") then -- for -8. (173,353)

if(COUNTER = "00011") then

COUNTER <= "01110";

end if;

if(COUNTER = "01110") then

COUNTER <= "10011";

end if;

end if;--end of -8.

if(input\_main="110" or input\_main = "10111010") then -- for 9. (6,186)

if(COUNTER = "00011") then

COUNTER <= "01111";

end if;

if(COUNTER = "01111") then

COUNTER <= "00000";

end if;

end if;--end of 9.

if(input\_main="10101110" or input\_main = "101100010") then -- for -9. (174,354)

if(COUNTER = "00011") then

COUNTER <= "01111";

end if;

if(COUNTER = "01111") then

COUNTER <= "10011";

end if;

end if;--end of -9.

if(input\_main="101" or input\_main = "10111001") then -- for 11. (5,185)

if(COUNTER = "00011") then

COUNTER <= "00101";

end if;

if(COUNTER = "00101") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "00000";

end if;

end if;--end of 11.

if(input\_main="10101111" or input\_main = "101100011") then -- for -11. (175,355)

if(COUNTER = "00011") then

COUNTER <= "00101";

end if;

if(COUNTER = "00101") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "10011";

end if;

end if;--end of -11.

if(input\_main="100" or input\_main = "10111000") then -- for 14. (4,184)

if(COUNTER = "00011") then

COUNTER <= "01010";

end if;

if(COUNTER = "01010") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "00000";

end if;

end if;--end of 14.

if(input\_main="10110000" or input\_main = "101100100") then -- for -14. (176,356)

if(COUNTER = "00011") then

COUNTER <= "01010";

end if;

if(COUNTER = "01010") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "10011";

end if;

end if;--end of -14.

if(input\_main="11" or input\_main= "10110111") then -- for 19. (3,183)

if(COUNTER = "00011") then

COUNTER <= "01111";

end if;

if(COUNTER = "01111") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "00000";

end if;

end if;--end of 19.

if(input\_main="10110001" or input\_main = "101100101") then -- for -19. (177,357)

if(COUNTER = "00011") then

COUNTER <= "01111";

end if;

if(COUNTER = "01111") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "10011";

end if;

end if;--end of -19.

if(input\_main="10" or input\_main = "10110110") then -- for 28. (2,182)

if(COUNTER = "00011") then

COUNTER <= "01110";

end if;

if(COUNTER = "01110") then

COUNTER <= "10001";

end if;

if(COUNTER = "10001") then

COUNTER <= "00000";

end if;

end if;--end of 28.

if(input\_main="10110010" or input\_main = "101100110") then -- for -28. (178,358)

if(COUNTER = "00011") then

COUNTER <= "01110";

end if;

if(COUNTER = "01110") then

COUNTER <= "10001";

end if;

if(COUNTER = "10001") then

COUNTER <= "10011";

end if;

end if;--end of -28.

if(input\_main="0001" or input\_main="10110101") then -- for 57. (1,181)

if(COUNTER = "00011") then

COUNTER <= "01101";

end if;

if(COUNTER = "01101") then

COUNTER <= "10010";

end if;

if(COUNTER = "10010") then

COUNTER <= "00000";

end if;

end if;--end of 57.

if(input\_main="10110011" or input\_main="101100111") then -- for -57. (179,359)

if(COUNTER = "00011") then

COUNTER <= "01101";

end if;

if(COUNTER = "01101") then

COUNTER <= "10010";

end if;

end if;--end of -57.

if(input\_main < "1011011" or (input\_main > "0000" and input\_main <"100001111")) then

-- minus'ı basma (180-270 arası ve 0-91 arası)

if(COUNTER ="10010") then

COUNTER <="00000";

end if;

end if;-- minus'ı basma

if(COUNTER = "10011") then

COUNTER <= "00000";

end if;

end if; -- end of COTANGENT

--------------------COT TO TAN------------------------------------------

if (tan\_enabler\_main = '1') then -- TANGENT FUNCTION

COUNTER<=COUNTER + 1;

if(input\_main = "1011010" or input\_main="100001110") then -- 90 and 270 - Tanımsız

COUNTER <= "10100";

end if;

if((input\_main >="0" and input\_main<"101101") or (input\_main >="10110100" and input\_main <"11100001")

or input\_main="101101000") then

-- for 0. (>=0 and <45) or (>180 and <225) or =360

if(COUNTER = "00100") then

COUNTER <= "00000";

end if;

end if;--end of 0.

if((input\_main >"10000111" and input\_main<"10110100") or (input\_main >"100111011" and input\_main <"101101000")) then

-- for -0. (>135 and <180) or (>315 and <360)

if(COUNTER = "00100") then

COUNTER <= "10011";

end if;

end if;--end of -0.

if((input\_main > "101100" and input\_main < "1000000") or (input\_main>"11100000" and input\_main<"11110100")) then

-- for 1.(> 44 and < 64) or (>224 and <244)

if(COUNTER="00011") then

COUNTER <= "00101";

end if;

if(COUNTER = "00101") then

COUNTER <= "00000";

end if;

end if; -- end of 1.

if((input\_main > "1110100" and input\_main < "10001000") or (input\_main>"100101000" and input\_main<"100111100")) then

-- for -1.(> 116 and < 136) or (>296 and <316)

if(COUNTER="00011") then

COUNTER <= "00101";

end if;

if(COUNTER = "00101") then

COUNTER <= "10011";

end if;

end if; -- end of -1.

if((input\_main > "111111" and input\_main < "001001000") or (input\_main>"11110011" and input\_main<"11111100")) then

-- for 2. (> 63 and <72) or (>243 and <252)

if(COUNTER = "00011") then

COUNTER <= "01000";

end if;

if(COUNTER = "01000") then

COUNTER <= "00000";

end if;

end if;--end of 2.

if((input\_main > "1101100" and input\_main < "1110101") or (input\_main>"100100000" and input\_main<"100101001")) then

-- for -2. (> 108 and <117) or (>288 and <297)

if(COUNTER = "00011") then

COUNTER <= "01000";

end if;

if(COUNTER = "01000") then

COUNTER <= "10011";

end if;

end if;--end of -2.

if((input\_main > "1000111" and input\_main < "1001100") or (input\_main>"11111011" and input\_main<"100000000")) then

-- for 3. (> 71 and <76) or (>251 and <256)

if(COUNTER = "00011") then

COUNTER <= "01001";

end if;

if(COUNTER = "01001") then

COUNTER <= "00000";

end if;

end if;--end of 3.

if((input\_main > "1101000" and input\_main < "1101101") or (input\_main>"100011100" and input\_main<"100100001")) then

-- for -3. (> 104 and <109) or (>284 and <289)

if(COUNTER = "00011") then

COUNTER <= "01001";

end if;

if(COUNTER = "01001") then

COUNTER <= "10011";

end if;

end if;--end of -3.

if((input\_main > "1001011" and input\_main < "1001111") or (input\_main>"11111111" and input\_main<"100000011")) then

-- for 4. (> 75 and <79) or (>255 and <259)

if(COUNTER = "00011") then

COUNTER <= "01010";

end if;

if(COUNTER = "01010") then

COUNTER <= "00000";

end if;

end if;--end of 4.

if((input\_main > "1100101" and input\_main < "1101001") or (input\_main>"100011001" and input\_main<"100011101")) then

-- for -4. (> 101 and <105) or (>281 and <285)

if(COUNTER = "00011") then

COUNTER <= "01010";

end if;

if(COUNTER = "01010") then

COUNTER <= "10011";

end if;

end if;--end of -4.

if((input\_main > "1001110" and input\_main < "1010001") or (input\_main > "100000010" and input\_main < "100000101")) then

-- for 5. (> 78 and <81) or (>258 and < 261)

if(COUNTER = "00011") then

COUNTER <= "01011";

end if;

if(COUNTER = "01011") then

COUNTER <= "00000";

end if;

end if;--end of 5.

if((input\_main > "1100011" and input\_main < "1100110") or (input\_main > "100010111" and input\_main < "100011010")) then

-- for -5. (> 99 and <102) or (>279 and < 282)

if(COUNTER = "00011") then

COUNTER <= "01011";

end if;

if(COUNTER = "01011") then

COUNTER <= "10011";

end if;

end if;--end of -5.

if(input\_main="1010001" or input\_main = "100000101") then -- for 6. (81,261)

if(COUNTER = "00011") then

COUNTER <= "01100";

end if;

if(COUNTER = "01100") then

COUNTER <= "00000";

end if;

end if;--end of 6.

if(input\_main="1100011" or input\_main = "100010111") then -- for -6. (99,279)

if(COUNTER = "00011") then

COUNTER <= "01100";

end if;

if(COUNTER = "01100") then

COUNTER <= "10011";

end if;

end if;--end of -6.

if(input\_main="1010010" or input\_main ="100000110") then -- for 7. (82,262)

if(COUNTER = "00011") then

COUNTER <= "01101";

end if;

if(COUNTER = "01101") then

COUNTER <= "00000";

end if;

end if;--end of 7.

if(input\_main="1100010" or input\_main ="100010110") then -- for -7. (98,278)

if(COUNTER = "00011") then

COUNTER <= "01101";

end if;

if(COUNTER = "01101") then

COUNTER <= "10011";

end if;

end if;--end of -7.

if(input\_main="1010011" or input\_main = "100000111") then -- for 8. (83,263)

if(COUNTER = "00011") then

COUNTER <= "01110";

end if;

if(COUNTER = "01110") then

COUNTER <= "00000";

end if;

end if;--end of 8.

if(input\_main="1100001" or input\_main = "100010101") then -- for -8. (97,277)

if(COUNTER = "00011") then

COUNTER <= "01110";

end if;

if(COUNTER = "01110") then

COUNTER <= "10011";

end if;

end if;--end of -8.

if(input\_main="1010100" or input\_main = "100001000") then -- for 9. (84,264)

if(COUNTER = "00011") then

COUNTER <= "01111";

end if;

if(COUNTER = "01111") then

COUNTER <= "00000";

end if;

end if;--end of 9.

if(input\_main="1100000" or input\_main = "100010100") then -- for -9. (96,276)

if(COUNTER = "00011") then

COUNTER <= "01111";

end if;

if(COUNTER = "01111") then

COUNTER <= "10011";

end if;

end if;--end of -9.

if(input\_main="1010101" or input\_main = "100001001") then -- for 11. (85,265)

if(COUNTER = "00011") then

COUNTER <= "00101";

end if;

if(COUNTER = "00101") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "00000";

end if;

end if;--end of 11.

if(input\_main="1011111" or input\_main = "100010011") then -- for -11. (95,275)

if(COUNTER = "00011") then

COUNTER <= "00101";

end if;

if(COUNTER = "00101") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "10011";

end if;

end if;--end of -11.

if(input\_main="1010110" or input\_main = "100001010") then -- for 14. (86,266)

if(COUNTER = "00011") then

COUNTER <= "01010";

end if;

if(COUNTER = "01010") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "00000";

end if;

end if;--end of 14.

if(input\_main="1011110" or input\_main = "100010010") then -- for -14. (94,274)

if(COUNTER = "00011") then

COUNTER <= "01010";

end if;

if(COUNTER = "01010") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "10011";

end if;

end if;--end of -14.

if(input\_main="1010111" or input\_main= "100001011") then -- for 19. (87,267)

if(COUNTER = "00011") then

COUNTER <= "01111";

end if;

if(COUNTER = "01111") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "00000";

end if;

end if;--end of 19.

if(input\_main="1011101" or input\_main = "100010001") then -- for -19. (93,273)

if(COUNTER = "00011") then

COUNTER <= "01111";

end if;

if(COUNTER = "01111") then

COUNTER <= "10000";

end if;

if(COUNTER = "10000") then

COUNTER <= "10011";

end if;

end if;--end of -19.

if(input\_main="1011000" or input\_main = "100001100") then -- for 28. (88,268)

if(COUNTER = "00011") then

COUNTER <= "01110";

end if;

if(COUNTER = "01110") then

COUNTER <= "10001";

end if;

if(COUNTER = "10001") then

COUNTER <= "00000";

end if;

end if;--end of 28.

if(input\_main="1011100" or input\_main = "100010000") then -- for -28. (92,272)

if(COUNTER = "00011") then

COUNTER <= "01110";

end if;

if(COUNTER = "01110") then

COUNTER <= "10001";

end if;

if(COUNTER = "10001") then

COUNTER <= "10011";

end if;

end if;--end of -28.

if(input\_main="1011001" or input\_main="100001101") then -- for 57. (89,269)

if(COUNTER = "00011") then

COUNTER <= "01101";

end if;

if(COUNTER = "01101") then

COUNTER <= "10010";

end if;

if(COUNTER = "10010") then

COUNTER <= "00000";

end if;

end if;--end of 57.

if(input\_main="1011011" or input\_main="100001111") then -- for -57. (91,271)

if(COUNTER = "00011") then

COUNTER <= "01101";

end if;

if(COUNTER = "01101") then

COUNTER <= "10010";

end if;

end if;--end of -57.

if(input\_main < "1011011" or (input\_main > "10110100" and input\_main <"100001111")) then -- minus'ı basma (180-270 arası ve 0-91 arası)

if(COUNTER ="10010") then

COUNTER <="00000";

end if;

end if;-- minus'ı basma

if(COUNTER = "10011") then

COUNTER <= "00000";

end if;

end if; -- end of TANGENT

if (cos\_enabler\_main ='1') then -- COSINE FUNCTION

COUNTER<=COUNTER + 1;

if(COUNTER = "00110") then --

COUNTER <= "00000";--

end if;--

if(input\_main > "0000000100001110" or input\_main<"0000000001011010") then -- between 90 and 270

if(COUNTER = "00100") then

COUNTER<= "00000"; -- print minus

end if;

end if;

if (input\_main = "0000000010110100") then -- 180

if(COUNTER = "00011") then

COUNTER<="00101"; -- print -1

end if;

if(COUNTER = "00110") then --

COUNTER <= "00000";--

end if;--

end if;

if(input\_main = "0000000000000000" or input\_main = "0000000101101000") then -- 0 and 360

if(COUNTER<="00011") then

COUNTER<="00101";

end if;

if(COUNTER="00101") then

COUNTER<="00111"; -- print 1

end if; -- end of 1

if(COUNTER = "00111") then--

COUNTER <= "00000";--

end if;--

end if; -- end of 0 and 360

end if; -- end of cosine

end if; -- end of counter

end if; -- end of clock

case sevseg\_signal is

when "0000" => sevseg\_out <="11000000"; ---0

when "0001" => sevseg\_out <="11111001" ; ---1

when "0010" => sevseg\_out <="10100100" ;---2

when "0011" => sevseg\_out <="10110000" ;---3

when "0100" => sevseg\_out <="10011001" ;---4

when "0101" => sevseg\_out <="10010010" ;---5

when "0110" => sevseg\_out <="10000010" ;---6

when "0111" => sevseg\_out <="11111000" ;---7

when "1000" => sevseg\_out <="10000000" ;---8

when "1001" => sevseg\_out <="10010000" ;---9

when "1010" => sevseg\_out <="10111111"; --- minus

when "1011" => sevseg\_out <="01000000"; --- 0.

when "1110" => sevseg\_out <="01111001"; --- 1.

when others => sevseg\_out <="11111111"; --- nothing

end case;

case counter is

when "01000" => sevseg\_out <= "00100100"; -- 2. (4 an)

when "01001" => sevseg\_out <= "00110000"; -- 3.

when "01010" => sevseg\_out <= "00011001"; -- 4.

when "01011" => sevseg\_out <= "00010010"; -- 5.

when "01100" => sevseg\_out <= "00000010"; -- 6.

when "01101" => sevseg\_out <= "01111000"; -- 7.

when "01110" => sevseg\_out <= "00000000"; -- 8.

when "01111" => sevseg\_out <= "00010000"; -- 9. (4 an)

when "10000" => sevseg\_out <= "11111001"; -- 1\_.

when "10001" => sevseg\_out <= "10100100"; -- 2\_.

when "10010" => sevseg\_out <= "10010010"; -- 5\_.

when "10011" => sevseg\_out <= "10111111"; -- -\_\_. (2 an)

when "10100" => sevseg\_out <= "10000110"; -- E

when others => dummy <= "11111111"; -- just a tricky way to avoid when others (because we use others that are not

specified here.)

end case;

end process;

with counter select sevseg\_signal <=

ones1 when "00000",

tens1 when "00001",

hundreds1 when "00010",

thousands1 when "00011",

"1011" when "00100", --- for 0.

"1110" when "00101", --- for 1.

"1010" when "00110", -- for minus

"1111" when others;

process(CLK)

begin

if(cot\_enabler\_main = '0' and sin\_enabler\_main = '0' and tan\_enabler\_main ='0' and cos\_enabler\_main = '0') then

an\_out<="11111111";

else

case counter is

when "00000" =>an\_out<="11111110";

when "00001" =>an\_out<="11111101";

when "00010" =>an\_out<="11111011";

when "00011" =>an\_out<="11110111";

when "00100" =>an\_out<="11101111"; -- 0.

when "00101" =>an\_out<="11101111"; -- 1.

when "00110" =>an\_out<="11011111";

when "01000" =>an\_out<="11101111"; -- 2. (4 an)

when "01001" =>an\_out<="11101111"; -- 3.

when "01010" =>an\_out<="11101111"; -- 4.

when "01011" =>an\_out<="11101111"; -- 5.

when "01100" =>an\_out<="11101111"; -- 6.

when "01101" =>an\_out<="11101111"; -- 7.

when "01110" =>an\_out<="11101111"; -- 8.

when "01111" =>an\_out<="11101111"; -- 9. (4 an)

when "10000" =>an\_out<="11011111"; -- 1\_. (3 an)

when "10001" =>an\_out<="11011111"; -- 2\_.

when "10010" =>an\_out<="11011111"; -- 5\_. (3 an)

when "10011" =>an\_out<="10111111"; -- -\_\_. (2 an)

when "10100" =>an\_out<="11111110"; -- E

when others =>an\_out<="11111111";

end case;

end if;

end process;

end Behavioral;

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.numeric\_std.all;

entity bin2bcd is

port (

input: in std\_logic\_vector (15 downto 0);

ones: out std\_logic\_vector (3 downto 0);

tens: out std\_logic\_vector (3 downto 0);

hundreds: out std\_logic\_vector (3 downto 0);

thousands: out std\_logic\_vector (3 downto 0)

);

end entity;

architecture fum of bin2bcd is

alias Hex\_Display\_Data: std\_logic\_vector (15 downto 0) is input;

alias rpm\_1: std\_logic\_vector (3 downto 0) is ones;

alias rpm\_10: std\_logic\_vector (3 downto 0) is tens;

alias rpm\_100: std\_logic\_vector (3 downto 0) is hundreds;

alias rpm\_1000: std\_logic\_vector (3 downto 0) is thousands;

begin

process (Hex\_Display\_Data)

type fourbits is array (3 downto 0) of std\_logic\_vector(3 downto 0);

variable bcd: std\_logic\_vector (15 downto 0);

variable bint: std\_logic\_vector (13 downto 0);

begin

bcd := (others => '0');

bint := Hex\_Display\_Data (13 downto 0);

for i in 0 to 13 loop

bcd(15 downto 1) := bcd(14 downto 0);

bcd(0) := bint(13);

bint(13 downto 1) := bint(12 downto 0);

bint(0) := '0';

if i < 13 and bcd(3 downto 0) > "0100" then

bcd(3 downto 0) :=

std\_logic\_vector (unsigned(bcd(3 downto 0)) + 3);

end if;

if i < 13 and bcd(7 downto 4) > "0100" then

bcd(7 downto 4) :=

std\_logic\_vector(unsigned(bcd(7 downto 4)) + 3);

end if;

if i < 13 and bcd(11 downto 8) > "0100" then

bcd(11 downto 8) :=

std\_logic\_vector(unsigned(bcd(11 downto 8)) + 3);

end if;

if i < 13 and bcd(15 downto 12) > "0100" then

bcd(11 downto 8) :=

std\_logic\_vector(unsigned(bcd(15 downto 12)) + 3);

end if;

end loop;

(rpm\_1000, rpm\_100, rpm\_10, rpm\_1) <=

fourbits'( bcd (15 downto 12), bcd (11 downto 8),

bcd ( 7 downto 4), bcd ( 3 downto 0) );

end process ;

end architecture;

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

USE IEEE.STD\_LOGIC\_ARITH.ALL;

USE IEEE.STD\_LOGIC\_UNSIGNED.ALL;

USE IEEE.NUMERIC\_STD.ALL;

entity Multiplexer is

Port ( input : in STD\_LOGIC\_VECTOR (8 downto 0);

cos\_enabler : in STD\_LOGIC;

sin\_enabler : in STD\_LOGIC;

tan\_enabler : in STD\_LOGIC;

cot\_enabler : in STD\_LOGIC;

rad\_enabler : in STD\_LOGIC;

output\_mux : out STD\_LOGIC\_VECTOR (15 downto 0));

end Multiplexer;

architecture Behavioral of Multiplexer is

component Rad is

port(

deg\_input\_rad : in STD\_LOGIC\_VECTOR (8 downto 0);

output\_rad : out STD\_LOGIC\_VECTOR (15 downto 0)

);

end component;

component Cos is

port(

deg\_input\_cos : in STD\_LOGIC\_VECTOR (8 downto 0);

output\_cos : out STD\_LOGIC\_VECTOR (15 downto 0)

);

end component;

component Sin is

port(

deg\_input\_sin : in STD\_LOGIC\_VECTOR (8 downto 0);

output\_sin : out STD\_LOGIC\_VECTOR (15 downto 0)

);

end component;

component Tan is

port(

deg\_input\_tan : in STD\_LOGIC\_VECTOR (8 downto 0);

output\_tan : out STD\_LOGIC\_VECTOR (15 downto 0)

);

end component;

component Cot is

port(

deg\_input\_cot : in STD\_LOGIC\_VECTOR (8 downto 0);

output\_cot : out STD\_LOGIC\_VECTOR (15 downto 0)

);

end component;

signal passer\_out\_cos : STD\_LOGIC\_VECTOR (15 downto 0);

signal passer\_out\_sin : STD\_LOGIC\_VECTOR (15 downto 0);

signal passer\_out\_cot : STD\_LOGIC\_VECTOR (15 downto 0);

signal passer\_out\_tan : STD\_LOGIC\_VECTOR (15 downto 0);

signal passer\_out\_rad : STD\_LOGIC\_VECTOR (15 downto 0);

begin

C: Cos port map(input,passer\_out\_cos);

S: Sin port map(input,passer\_out\_sin);

T: Tan port map(input,passer\_out\_tan);

Co: Cot port map(input,passer\_out\_cot);

---R: Rad port map(input,passer\_out\_rad);

process

begin

output\_mux<="0000000000000000";

if(cos\_enabler='1') then

output\_mux<=passer\_out\_cot;

end if;

if (sin\_enabler='1') then

output\_mux<=passer\_out\_cot;

end if;

if (tan\_enabler ='1') then

output\_mux <= passer\_out\_cot;

end if;

if (cot\_enabler ='1') then

output\_mux <= passer\_out\_cot;

end if;

---if(rad\_enabler = '1') then

---output\_mux <= passer\_out\_rad;

---end if;

end process;

end Behavioral;

entity Cos is

port(

deg\_input\_cos : in STD\_LOGIC\_VECTOR (8 downto 0);

output\_cos : out STD\_LOGIC\_VECTOR (15 downto 0)

);

end Cos;

architecture Behavioral of Cos is

component LUT\_cos is

port(

A : in STD\_LOGIC\_VECTOR (8 downto 0);

B : out STD\_LOGIC\_VECTOR (15 downto 0)

);

end component;

signal passer : STD\_LOGIC\_VECTOR (15 downto 0);

begin

G1 : LUT\_cos port map(deg\_input\_cos,passer);

output\_cos<=passer;

end Behavioral; (Sin, Tan, Cot are exactly same except the name and LUT\_sin,LUT\_tan etc.)

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity LUT\_Cos is

port(

A : in STD\_LOGIC\_VECTOR (8 downto 0);

B : out STD\_LOGIC\_VECTOR (15 downto 0)

);

end LUT\_Cos;

architecture Behavioral of LUT\_Cos is

begin

with A select B <=

"0000000000000000" when "000000000",

"0010011100001110" when "000000001",

"0010011100001010" when "000000010",

"0010011100000010" when "000000011",

"0010011011111000" when "000000100",

"0010011011101010" when "000000101",

"0010011011011001" when "000000110",

"0010011011000101" when "000000111",

"0010011010101111" when "000001000",

"0010011010010101" when "000001001",

"0010011001111000" when "000001010",

"0010011001011000" when "000001011",

"0010011000110101" when "000001100",

"0010011000010000" when "000001101",

"0010010111100111" when "000001110",

"0010010110111011" when "000001111",

"0010010110001101" when "000010000",

"0010010101011011" when "000010001",

"0010010100100111" when "000010010",

"0010010011101111" when "000010011",

"0010010010110101" when "000010100",

"0010010001111000" when "000010101",

"0010010000111000" when "000010110",

"0010001111110101" when "000010111",

"0010001110101111" when "000011000",

"0010001101100111" when "000011001",

"0010001100011100" when "000011010",

"0010001011001110" when "000011011",

"0010001001111101" when "000011100",

"0010001000101010" when "000011101",

"0010000111010100" when "000011110",

"0010000101111100" when "000011111",

"0010000100100000" when "000100000",

"0010000011000011" when "000100001",

"0010000001100010" when "000100010",

"0010000000000000" when "000100011",

"0001111110011010" when "000100100",

"0001111100110010" when "000100101",

"0001111011001000" when "000100110",

"0001111001011011" when "000100111",

"0001110111101100" when "000101000",

"0001110101111011" when "000101001",

"0001110100000111" when "000101010",

"0001110010010010" when "000101011",

"0001110000011001" when "000101100",

"0001101110011111" when "000101101",

"0001101100100011" when "000101110",

"0001101010100100" when "000101111",

"0001101000100011" when "000110000",

"0001100110100001" when "000110001",

"0001100100011100" when "000110010",

"0001100010010101" when "000110011",

"0001100000001101" when "000110100",

"0001011110000010" when "000110101",

"0001011011110110" when "000110110",

"0001011001101000" when "000110111",

"0001010111011000" when "000111000",

"0001010101000110" when "000111001",

"0001010010110011" when "000111010",

"0001010000011110" when "000111011",

"0001001110001000" when "000111100",

"0001001011110000" when "000111101",

"0001001001010111" when "000111110",

"0001000110111100" when "000111111",

"0001000100100000" when "001000000",

"0001000010000010" when "001000001",

"0000111111100011" when "001000010",

"0000111101000011" when "001000011",

"0000111010100010" when "001000100",

"0000111000000000" when "001000101",

"0000110101011100" when "001000110",

"0000110010111000" when "001000111",

"0000110000010010" when "001001000",

"0000101101101100" when "001001001",

"0000101011000100" when "001001010",

"0000101000011100" when "001001011",

"0000100101110011" when "001001100",

"0000100011001010" when "001001101",

"0000100000011111" when "001001110",

"0000011101110100" when "001001111",

"0000011011001000" when "001010000",

"0000011000011100" when "001010001",

"0000010101110000" when "001010010",

"0000010011000011" when "001010011",

"0000010000010101" when "001010100",

"0000001101101000" when "001010101",

"0000001010111010" when "001010110",

"0000001000001011" when "001010111",

"0000000101011101" when "001011000",

"0000000010101111" when "001011001",

"0000000000000000" when "001011010",

"0000000010101111" when "001011011",

"0000000101011101" when "001011100",

"0000001000001011" when "001011101",

"0000001010111010" when "001011110",

"0000001101101000" when "001011111",

"0000010000010101" when "001100000",

"0000010011000011" when "001100001",

"0000010101110000" when "001100010",

"0000011000011100" when "001100011",

"0000011011001000" when "001100100",

"0000011101110100" when "001100101",

"0000100000011111" when "001100110",

"0000100011001010" when "001100111",

"0000100101110011" when "001101000",

"0000101000011100" when "001101001",

"0000101011000100" when "001101010",

"0000101101101100" when "001101011",

"0000110000010010" when "001101100",

"0000110010111000" when "001101101",

"0000110101011100" when "001101110",

"0000111000000000" when "001101111",

"0000111010100010" when "001110000",

"0000111101000011" when "001110001",

"0000111111100011" when "001110010",

"0001000010000010" when "001110011",

"0001000100100000" when "001110100",

"0001000110111100" when "001110101",

"0001001001010111" when "001110110",

"0001001011110000" when "001110111",

"0001001110001000" when "001111000",

"0001010000011110" when "001111001",

"0001010010110011" when "001111010",

"0001010101000110" when "001111011",

"0001010111011000" when "001111100",

"0001011001101000" when "001111101",

"0001011011110110" when "001111110",

"0001011110000010" when "001111111",

"0001100000001101" when "010000000",

"0001100010010101" when "010000001",

"0001100100011100" when "010000010",

"0001100110100001" when "010000011",

"0001101000100011" when "010000100",

"0001101010100100" when "010000101",

"0001101100100011" when "010000110",

"0001101110011111" when "010000111",

"0001110000011001" when "010001000",

"0001110010010010" when "010001001",

"0001110100000111" when "010001010",

"0001110101111011" when "010001011",

"0001110111101100" when "010001100",

"0001111001011011" when "010001101",

"0001111011001000" when "010001110",

"0001111100110010" when "010001111",

"0001111110011010" when "010010000",

"0010000000000000" when "010010001",

"0010000001100010" when "010010010",

"0010000011000011" when "010010011",

"0010000100100000" when "010010100",

"0010000101111100" when "010010101",

"0010000111010100" when "010010110",

"0010001000101010" when "010010111",

"0010001001111101" when "010011000",

"0010001011001110" when "010011001",

"0010001100011100" when "010011010",

"0010001101100111" when "010011011",

"0010001110101111" when "010011100",

"0010001111110101" when "010011101",

"0010010000111000" when "010011110",

"0010010001111000" when "010011111",

"0010010010110101" when "010100000",

"0010010011101111" when "010100001",

"0010010100100111" when "010100010",

"0010010101011011" when "010100011",

"0010010110001101" when "010100100",

"0010010110111011" when "010100101",

"0010010111100111" when "010100110",

"0010011000010000" when "010100111",

"0010011000110101" when "010101000",

"0010011001011000" when "010101001",

"0010011001111000" when "010101010",

"0010011010010101" when "010101011",

"0010011010101111" when "010101100",

"0010011011000101" when "010101101",

"0010011011011001" when "010101110",

"0010011011101010" when "010101111",

"0010011011111000" when "010110000",

"0010011100000010" when "010110001",

"0010011100001010" when "010110010",

"0010011100001110" when "010110011",

"0000000000000000" when "010110100",

"0010011100001110" when "010110101",

"0010011100001010" when "010110110",

"0010011100000010" when "010110111",

"0010011011111000" when "010111000",

"0010011011101010" when "010111001",

"0010011011011001" when "010111010",

"0010011011000101" when "010111011",

"0010011010101111" when "010111100",

"0010011010010101" when "010111101",

"0010011001111000" when "010111110",

"0010011001011000" when "010111111",

"0010011000110101" when "011000000",

"0010011000010000" when "011000001",

"0010010111100111" when "011000010",

"0010010110111011" when "011000011",

"0010010110001101" when "011000100",

"0010010101011011" when "011000101",

"0010010100100111" when "011000110",

"0010010011101111" when "011000111",

"0010010010110101" when "011001000",

"0010010001111000" when "011001001",

"0010010000111000" when "011001010",

"0010001111110101" when "011001011",

"0010001110101111" when "011001100",

"0010001101100111" when "011001101",

"0010001100011100" when "011001110",

"0010001011001110" when "011001111",

"0010001001111101" when "011010000",

"0010001000101010" when "011010001",

"0010000111010100" when "011010010",

"0010000101111100" when "011010011",

"0010000100100000" when "011010100",

"0010000011000011" when "011010101",

"0010000001100010" when "011010110",

"0010000000000000" when "011010111",

"0001111110011010" when "011011000",

"0001111100110010" when "011011001",

"0001111011001000" when "011011010",

"0001111001011011" when "011011011",

"0001110111101100" when "011011100",

"0001110101111011" when "011011101",

"0001110100000111" when "011011110",

"0001110010010010" when "011011111",

"0001110000011001" when "011100000",

"0001101110011111" when "011100001",

"0001101100100011" when "011100010",

"0001101010100100" when "011100011",

"0001101000100011" when "011100100",

"0001100110100001" when "011100101",

"0001100100011100" when "011100110",

"0001100010010101" when "011100111",

"0001100000001101" when "011101000",

"0001011110000010" when "011101001",

"0001011011110110" when "011101010",

"0001011001101000" when "011101011",

"0001010111011000" when "011101100",

"0001010101000110" when "011101101",

"0001010010110011" when "011101110",

"0001010000011110" when "011101111",

"0001001110001000" when "011110000",

"0001001011110000" when "011110001",

"0001001001010111" when "011110010",

"0001000110111100" when "011110011",

"0001000100100000" when "011110100",

"0001000010000010" when "011110101",

"0000111111100011" when "011110110",

"0000111101000011" when "011110111",

"0000111010100010" when "011111000",

"0000111000000000" when "011111001",

"0000110101011100" when "011111010",

"0000110010111000" when "011111011",

"0000110000010010" when "011111100",

"0000101101101100" when "011111101",

"0000101011000100" when "011111110",

"0000101000011100" when "011111111",

"0000100101110011" when "100000000",

"0000100011001010" when "100000001",

"0000100000011111" when "100000010",

"0000011101110100" when "100000011",

"0000011011001000" when "100000100",

"0000011000011100" when "100000101",

"0000010101110000" when "100000110",

"0000010011000011" when "100000111",

"0000010000010101" when "100001000",

"0000001101101000" when "100001001",

"0000001010111010" when "100001010",

"0000001000001011" when "100001011",

"0000000101011101" when "100001100",

"0000000010101111" when "100001101",

"0000000000000000" when "100001110",

"0000000010101111" when "100001111",

"0000000101011101" when "100010000",

"0000001000001011" when "100010001",

"0000001010111010" when "100010010",

"0000001101101000" when "100010011",

"0000010000010101" when "100010100",

"0000010011000011" when "100010101",

"0000010101110000" when "100010110",

"0000011000011100" when "100010111",

"0000011011001000" when "100011000",

"0000011101110100" when "100011001",

"0000100000011111" when "100011010",

"0000100011001010" when "100011011",

"0000100101110011" when "100011100",

"0000101000011100" when "100011101",

"0000101011000100" when "100011110",

"0000101101101100" when "100011111",

"0000110000010010" when "100100000",

"0000110010111000" when "100100001",

"0000110101011100" when "100100010",

"0000111000000000" when "100100011",

"0000111010100010" when "100100100",

"0000111101000011" when "100100101",

"0000111111100011" when "100100110",

"0001000010000010" when "100100111",

"0001000100100000" when "100101000",

"0001000110111100" when "100101001",

"0001001001010111" when "100101010",

"0001001011110000" when "100101011",

"0001001110001000" when "100101100",

"0001010000011110" when "100101101",

"0001010010110011" when "100101110",

"0001010101000110" when "100101111",

"0001010111011000" when "100110000",

"0001011001101000" when "100110001",

"0001011011110110" when "100110010",

"0001011110000010" when "100110011",

"0001100000001101" when "100110100",

"0001100010010101" when "100110101",

"0001100100011100" when "100110110",

"0001100110100001" when "100110111",

"0001101000100011" when "100111000",

"0001101010100100" when "100111001",

"0001101100100011" when "100111010",

"0001101110011111" when "100111011",

"0001110000011001" when "100111100",

"0001110010010010" when "100111101",

"0001110100000111" when "100111110",

"0001110101111011" when "100111111",

"0001110111101100" when "101000000",

"0001111001011011" when "101000001",

"0001111011001000" when "101000010",

"0001111100110010" when "101000011",

"0001111110011010" when "101000100",

"0010000000000000" when "101000101",

"0010000001100010" when "101000110",

"0010000011000011" when "101000111",

"0010000100100000" when "101001000",

"0010000101111100" when "101001001",

"0010000111010100" when "101001010",

"0010001000101010" when "101001011",

"0010001001111101" when "101001100",

"0010001011001110" when "101001101",

"0010001100011100" when "101001110",

"0010001101100111" when "101001111",

"0010001110101111" when "101010000",

"0010001111110101" when "101010001",

"0010010000111000" when "101010010",

"0010010001111000" when "101010011",

"0010010010110101" when "101010100",

"0010010011101111" when "101010101",

"0010010100100111" when "101010110",

"0010010101011011" when "101010111",

"0010010110001101" when "101011000",

"0010010110111011" when "101011001",

"0010010111100111" when "101011010",

"0010011000010000" when "101011011",

"0010011000110101" when "101011100",

"0010011001011000" when "101011101",

"0010011001111000" when "101011110",

"0010011010010101" when "101011111",

"0010011010101111" when "101100000",

"0010011011000101" when "101100001",

"0010011011011001" when "101100010",

"0010011011101010" when "101100011",

"0010011011111000" when "101100100",

"0010011100000010" when "101100101",

"0010011100001010" when "101100110",

"0010011100001110" when "101100111",

"0000000000000000" when "101101000",

"0000000000000000" when others;

end Behavioral; (Sin, Tan, Cot are exactly same except the name and values.)

THE UCF FILE:

NET "input\_main[0]" LOC = P15;

NET "input\_main[1]" LOC = P12;

NET "input\_main[2]" LOC = P5;

NET "input\_main[3]" LOC = P4;

NET "input\_main[4]" LOC = P94;

NET "input\_main[5]" LOC = P90;

NET "input\_main[6]" LOC = P88;

NET "input\_main[7]" LOC = P85;

NET "input\_main[8]" LOC = P82;

NET "cos\_enabler\_main" LOC = P37;

NET "sin\_enabler\_main" LOC = P32;

NET "tan\_enabler\_main" LOC = P36;

NET "cot\_enabler\_main" LOC = P33;

Net "sevseg\_out[0]" LOC = P71;

Net "sevseg\_out[1]" LOC = P62;

Net "sevseg\_out[2]" LOC = P65;

Net "sevseg\_out[3]" LOC = P72;

Net "sevseg\_out[4]" LOC = P73;

Net "sevseg\_out[5]" LOC = P98;

Net "sevseg\_out[6]" LOC = P64;

Net "sevseg\_out[7]" LOC = P70;

Net "an\_out[0]" LOC = P50;

Net "an\_out[1]" LOC = P49;

Net "an\_out[2]" LOC = P52;

Net "an\_out[3]" LOC = P56;

Net "an\_out[4]" LOC = P59;

Net "an\_out[5]" LOC = P57;

Net "an\_out[6]" LOC = P60;

Net "an\_out[7]" LOC = P61;

Net "CLK" LOC = P40;

END.