Contents

Package

```
-- -------------------------------------------------------------
--
-- File Name:
C:\Users\Mahsa\Desktop\Th1008_Code\tanh_vhld_piecewise_linear\codegen\vhdl_ap
proximate_tanh_design\hdlsrc\vhdl_approximate_tanh_design_fixpt_pkg.vhd
-- Created: 2021-08-21 19:45:46
--
-- Generated by MATLAB 9.8, MATLAB Coder 5.0 and HDL Coder 3.16
--
--
-- -------------------------------------------------------------


LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE IEEE.numeric_std.ALL;

PACKAGE vhdl_approximate_tanh_design_fixpt_pkg IS
  TYPE vector_of_std_logic_vector14 IS ARRAY (NATURAL RANGE <>) OF
std_logic_vector(13 DOWNTO 0);
  TYPE vector_of_signed14 IS ARRAY (NATURAL RANGE <>) OF signed(13 DOWNTO 0);
  TYPE vector_of_signed32 IS ARRAY (NATURAL RANGE <>) OF signed(31 DOWNTO 0);
  TYPE vector_of_real IS ARRAY (NATURAL RANGE <>) OF real;
  TYPE vector_of_signed29 IS ARRAY (NATURAL RANGE <>) OF signed(28 DOWNTO 0);
  TYPE vector_of_signed28 IS ARRAY (NATURAL RANGE <>) OF signed(27 DOWNTO 0);
  TYPE vector_of_unsigned32 IS ARRAY (NATURAL RANGE <>) OF unsigned(31 DOWNTO
0);
  TYPE vector_of_signed33 IS ARRAY (NATURAL RANGE <>) OF signed(32 DOWNTO 0);
END vhdl_approximate_tanh_design_fixpt_pkg;
```

vhdl_approximate_tanh_design_fixpt

```
-- -------------------------------------------------------------
--
```

```
-- File Name:
C:\Users\Mahsa\Desktop\Th1008_Code\tanh_vhld_piecewise_linear\codegen\vhdl_ap
proximate_tanh_design\hdlsrc\vhdl_approximate_tanh_design_fixpt.vhd
-- Created: 2021-08-21 19:45:46
--
-- Generated by MATLAB 9.8, MATLAB Coder 5.0 and HDL Coder 3.16
--
--
--
-- -------------------------------------------------------------
-- Rate and Clocking Details
-- -------------------------------------------------------------
-- Design base rate: 1
--
-- -------------------------------------------------------------



-- -------------------------------------------------------------
--
-- Module: vhdl_approximate_tanh_design_fixpt
-- Source Path: vhdl_approximate_tanh_design_fixpt
-- Hierarchy Level: 0
--
-- -------------------------------------------------------------
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE IEEE.numeric_std.ALL;
USE work.vhdl_approximate_tanh_design_fixpt_pkg.ALL;

ENTITY vhdl_approximate_tanh_design_fixpt IS
  PORT( x                                 :   IN
vector_of_std_logic_vector14(0 TO 125);  -- sfix14_En10 [126]
        y_out                             :   OUT
vector_of_std_logic_vector14(0 TO 125)  -- sfix14_En13 [126]
        );
END vhdl_approximate_tanh_design_fixpt;


ARCHITECTURE rtl OF vhdl_approximate_tanh_design_fixpt IS

  -- Constants
  CONSTANT One                            : real :=
    2.0;  -- double
  CONSTANT C_divbyzero_p                  : real :=
    1.0E+308;  -- double

  -- Functions
  -- HDLCODER_TO_SIGNED
  FUNCTION hdlcoder_to_signed(arg: real; width: integer) RETURN signed IS
  BEGIN
    RETURN to_signed(integer(arg), width);
  END FUNCTION;


  -- Signals
  SIGNAL x_signed                         : vector_of_signed14(0 TO 125);  --
sfix14_En10 [126]
  SIGNAL y_out_tmp                        : vector_of_signed14(0 TO 125);  --
sfix14_En13 [126]

BEGIN
  outputgen1: FOR k1 IN 0 TO 125 GENERATE
    x_signed(k1) <= signed(x(k1));
  END GENERATE;

  vhdl_approximate_tanh_design_fixpt_1_output : PROCESS (x_signed)
    VARIABLE y : vector_of_signed32(0 TO 125);
    VARIABLE y_0 : vector_of_signed32(0 TO 125);
    VARIABLE b : vector_of_signed32(0 TO 125);
    VARIABLE z1 : vector_of_signed32(0 TO 125);
    VARIABLE tmp : signed(31 DOWNTO 0);
    VARIABLE cast : vector_of_real(0 TO 125);
    VARIABLE div_temp : vector_of_real(0 TO 125);
    VARIABLE mul_temp : vector_of_signed29(0 TO 125);
    VARIABLE cast_0 : vector_of_signed28(0 TO 125);
    VARIABLE cast_1 : vector_of_signed28(0 TO 125);
    VARIABLE cast_2 : vector_of_unsigned32(0 TO 125);
    VARIABLE sll_temp : vector_of_unsigned32(0 TO 125);
    VARIABLE add_temp : vector_of_signed33(0 TO 125);
    VARIABLE sub_temp : vector_of_signed33(0 TO 125);
    VARIABLE cast_3 : vector_of_signed32(0 TO 125);
  BEGIN
    tmp := to_signed(16#00000000#, 32);
    --HDL code generation from MATLAB function:
vhdl_approximate_tanh_design_fixpt
    --
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    --
%
    --           Generated by MATLAB 9.8 and Fixed-Point Designer 7.0
%
    --
%
    --
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    FOR k IN 0 TO 125 LOOP
      mul_temp(k) := to_signed(16#2A00#, 15) * x_signed(k);
      cast_0(k) := mul_temp(k)(27 DOWNTO 0);
      cast_1(k) := resize(cast_0(k)(27 DOWNTO 22), 28);
      y(k) := resize(cast_1(k), 32);
      IF y(k) > to_signed(16#0000001E#, 32) THEN
        tmp := to_signed(16#7FFFFFFF#, 32);
      ELSE
        IF y(k)(31) = '1' THEN
          cast_2(k) := X"00000000";
        ELSE
          cast_2(k) := unsigned(y(k));
        END IF;
        sll_temp(k) := to_unsigned(16#00000001#, 32) sll
to_integer(cast_2(k));
```

```
        IF sll_temp(k)(31) /= '0' THEN
          tmp := X"7FFFFFFF";
        ELSE
          tmp := signed(sll_temp(k));
        END IF;
      END IF;
      y_0(k) := tmp;
      add_temp(k) := resize(y_0(k), 33) + to_signed(1, 33);
      IF (add_temp(k)(32) = '0') AND (add_temp(k)(31) /= '0') THEN
        b(k) := X"7FFFFFFF";
      ELSIF (add_temp(k)(32) = '1') AND (add_temp(k)(31) /= '1') THEN
        b(k) := X"80000000";
      ELSE
        b(k) := add_temp(k)(31 DOWNTO 0);
      END IF;
      cast(k) := real(to_integer(b(k)));
      IF cast(k) = 0.0 THEN
        div_temp(k) := C_divbyzero_p;
      ELSE
        div_temp(k) := One / cast(k);
      END IF;
      z1(k) := hdlcoder_to_signed(div_temp(k), 32);
      sub_temp(k) := to_signed(1, 33) - resize(z1(k), 33);
      IF (sub_temp(k)(32) = '0') AND (sub_temp(k)(31) /= '0') THEN
        cast_3(k) := X"7FFFFFFF";
      ELSIF (sub_temp(k)(32) = '1') AND (sub_temp(k)(31) /= '1') THEN
        cast_3(k) := X"80000000";
      ELSE
        cast_3(k) := sub_temp(k)(31 DOWNTO 0);
      END IF;
      y_out_tmp(k) <= signed'(cast_3(k)(0) & '0' & '0' & '0' & '0' & '0' &
'0' & '0' & '0' & '0' & '0' & '0' & '0' & '0');
    END LOOP;

  END PROCESS vhdl_approximate_tanh_design_fixpt_1_output;


  outputgen: FOR k1 IN 0 TO 125 GENERATE
    y_out(k1) <= std_logic_vector(y_out_tmp(k1));
  END GENERATE;

END rtl;
```

Design

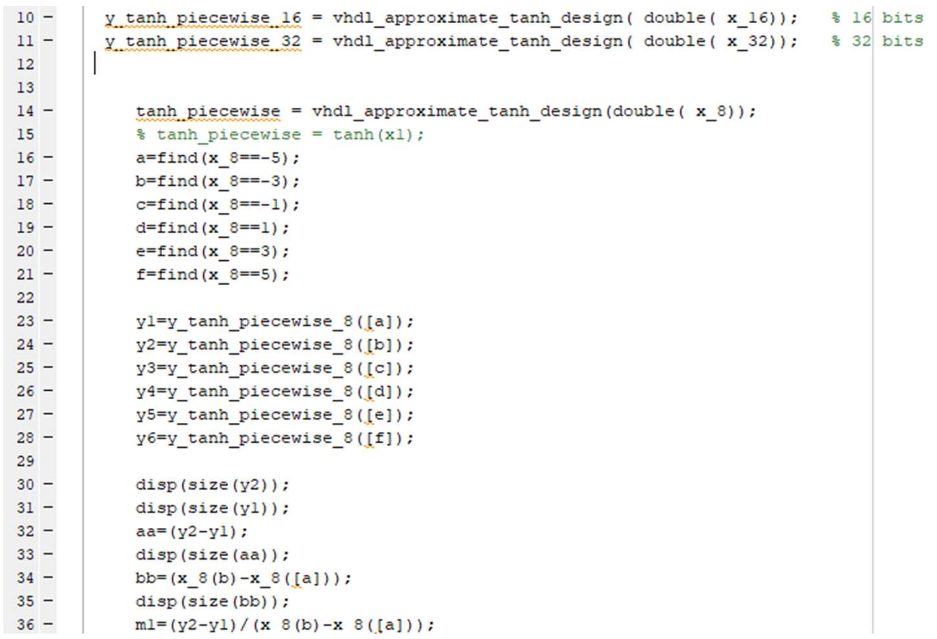
```
function [y_out] = vhdl_approximate_tanh_design(x)
%  y_out = 1-(2./(power(2,((2.625)*x))+1));
y_out = tanh(x);
end
```

Test Bunch

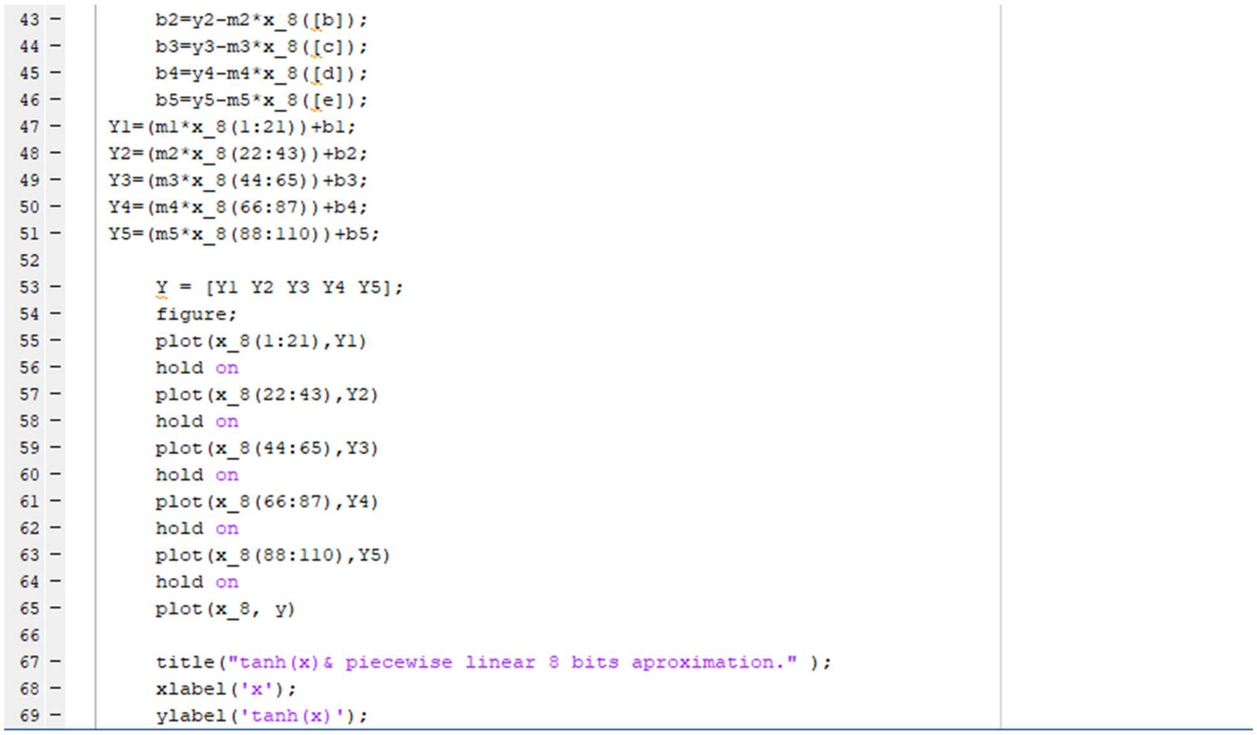
```
function vhdl_approximate_tanh_tb
  % Test inputs
  x_8  = fi(-5:0.08:5,1,8);
  x_16 = fi(-5:0.08:5,1,16);
  x_32 = fi(-5:0.08:5,1,32);
  % Run
 y = tanh( double(x_8));
 y_tanh_piecewise_8 = vhdl_approximate_tanh_design(double(x_8));   % 8 bits
 y_tanh_piecewise_16 = vhdl_approximate_tanh_design( double( x_16));   % 16
bits
 y_tanh_piecewise_32 = vhdl_approximate_tanh_design( double( x_32));   % 32
bits


    tanh_piecewise = vhdl_approximate_tanh_design(double( x_8));
    % tanh_piecewise = tanh(x1);
    a=find(x_8==-5);
    b=find(x_8==-3);
    c=find(x_8==-1);
    d=find(x_8==1);
    e=find(x_8==3);
    f=find(x_8==5);

    y1=y_tanh_piecewise_8([a]);
    y2=y_tanh_piecewise_8([b]);
    y3=y_tanh_piecewise_8([c]);
    y4=y_tanh_piecewise_8([d]);
    y5=y_tanh_piecewise_8([e]);
    y6=y_tanh_piecewise_8([f]);

    disp(size(y2));
    disp(size(y1));
    aa=(y2-y1);
    disp(size(aa));
    bb=(x_8(b)-x_8([a]));
    disp(size(bb));
    m1=(y2-y1)/(x_8(b)-x_8([a]));
    m2=(y3-y2)/(x_8(c)-x_8([b]));
    m3=(y4-y3)/(x_8(d)-x_8([c]));
    m4=(y5-y4)/(x_8(e)-x_8([d]));
    m5=(y6-y5)/(x_8(f)-x_8([e]));

    b1=y1-m1*x_8([a]);
    b2=y2-m2*x_8([b]);
    b3=y3-m3*x_8([c]);
    b4=y4-m4*x_8([d]);
    b5=y5-m5*x_8([e]);
Y1=(m1*x_8(1:21))+b1;
Y2=(m2*x_8(22:43))+b2;
Y3=(m3*x_8(44:65))+b3;
Y4=(m4*x_8(66:87))+b4;
Y5=(m5*x_8(88:110))+b5;

    Y = [Y1 Y2 Y3 Y4 Y5];
    figure;
    plot(x_8(1:21),Y1)
```

```
      hold on
      plot(x_8(22:43),Y2)
      hold on
      plot(x_8(44:65),Y3)
      hold on
      plot(x_8(66:87),Y4)
      hold on
      plot(x_8(88:110),Y5)
      hold on
      plot(x_8, y)

      title("tanh(x)& piecewise linear 8 bits aproximation." );
      xlabel('x');
      ylabel('tanh(x)');


    error8_ = RMSE(y_tanh_piecewise_8,y);
    formatSpec = '%.10f';
    error8_=num2str(error8_,formatSpec);
    disp("error 8 bits:" + error8_);

    figure;
    err = abs(y - double(y_tanh_piecewise_8));
    plot(x_8, err);
    xlabel('theta');
    ylabel('error');
    title("tanh(x)& piecewise linear 8 bits aproximation Error = " + error8_);
    grid on;


%     error = RMSE(t1,Y);
%     formatSpec = '%.10f';
%     error_=num2str(error,formatSpec);
%     disp(error_);
%     title("tanh(x)& piecewise linear Error = " + error_);
%     figure;
%     err = abs(tanh(x_8) - double(Y));
%     plot(x_8, err);
%     xlabel('theta');
%     ylabel('error');
%     title("tanh(x)& piecewise aproximation Error = " + error_);
%     formatSpec = '%.10f';
%     error_=num2str(error,formatSpec);
%     disp(num2str(error,formatSpec));
end
```

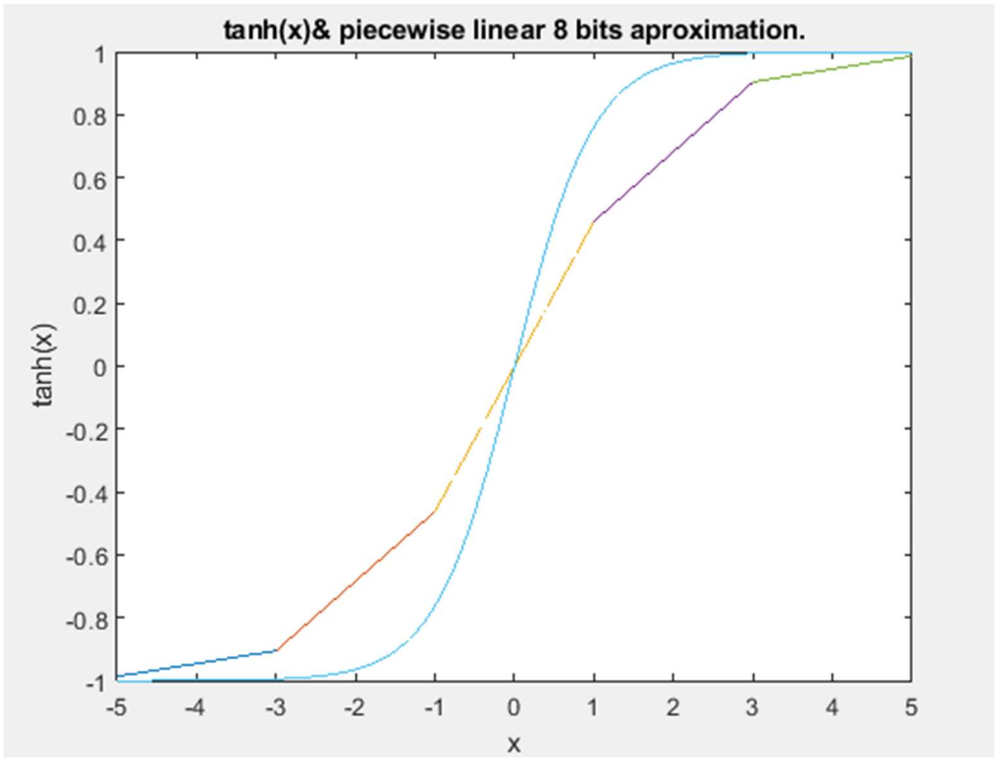
Plots

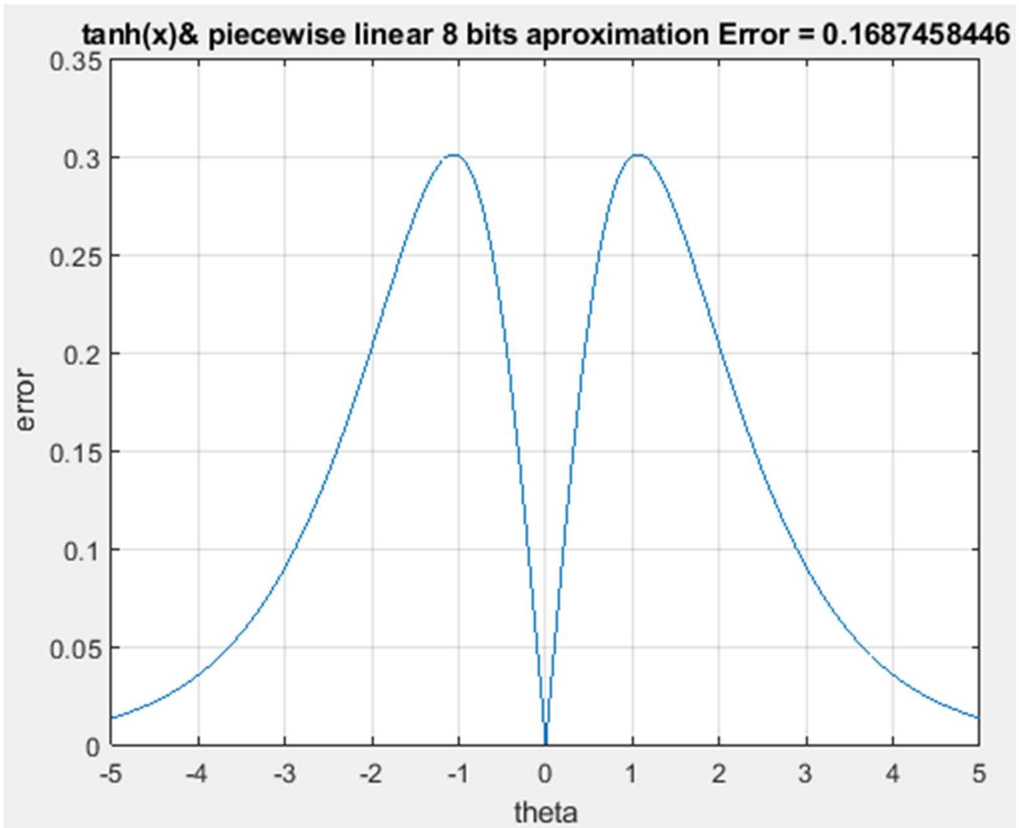


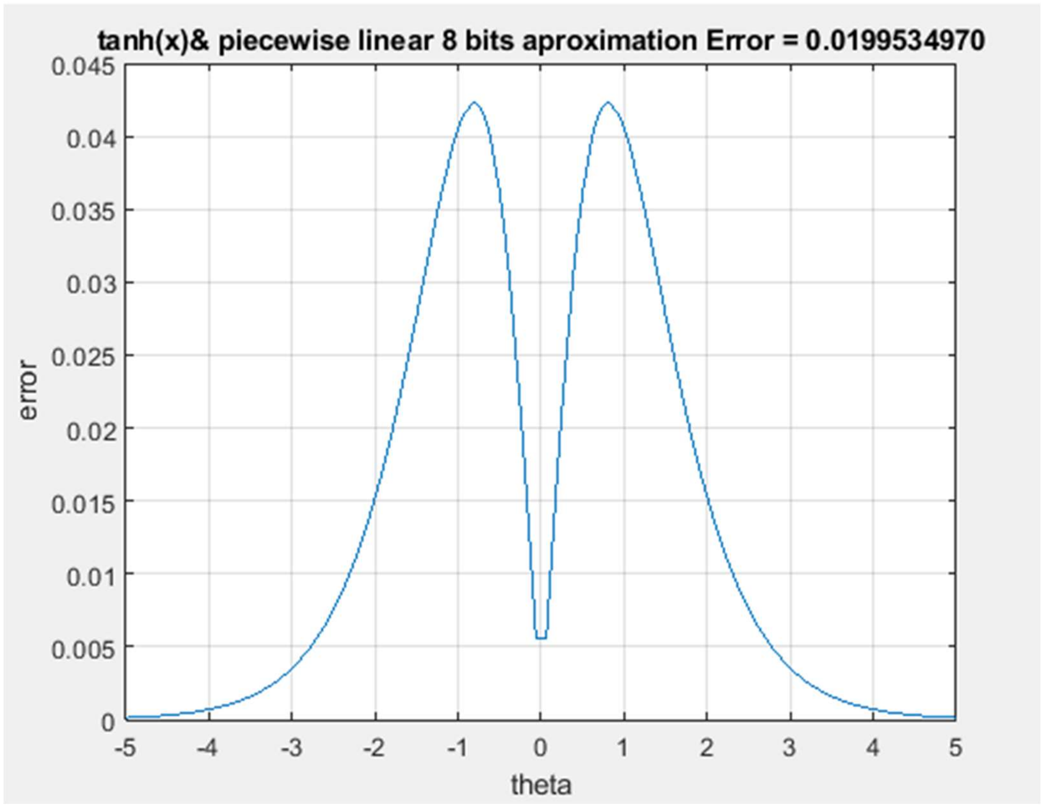
```
y_tanh_piecewise_16 = vhdl_approximate_tanh_design( double( x_16));   % 16 bits
y_tanh_piecewise_32 = vhdl_approximate_tanh_design( double( x_32));   % 32 bits


   tanh_piecewise = vhdl_approximate_tanh_design(double( x_8));
   % tanh_piecewise = tanh(x1);
   a=find(x_8==-5);
   b=find(x_8==-3);
   c=find(x_8==-1);
   d=find(x_8==1);
   e=find(x_8==3);
   f=find(x_8==5);

   y1=y_tanh_piecewise_8([a]);
   y2=y_tanh_piecewise_8([b]);
   y3=y_tanh_piecewise_8([c]);
   y4=y_tanh_piecewise_8([d]);
   y5=y_tanh_piecewise_8([e]);
   y6=y_tanh_piecewise_8([f]);

   disp(size(y2));
   disp(size(y1));
   aa=(y2-y1);
   disp(size(aa));
   bb=(x_8(b)-x_8([a]));
   disp(size(bb));
   m1=(y2-y1)/(x_8(b)-x_8([a]));
```

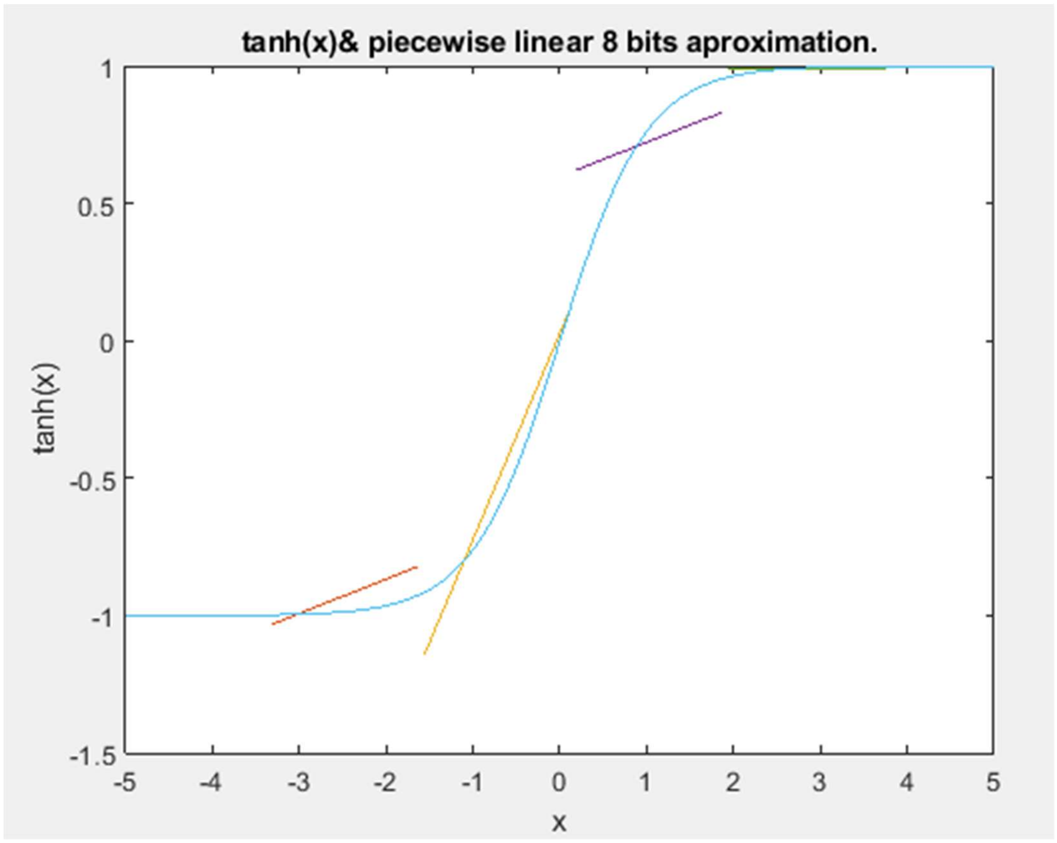


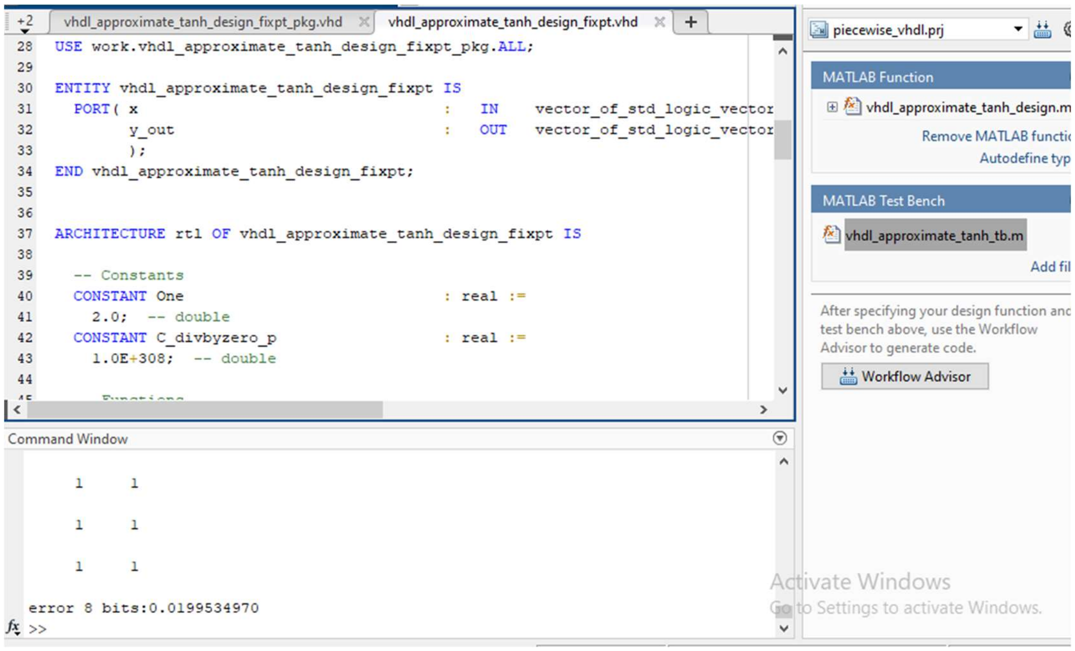
```
    b2=y2-m2*x_8([b]);
    b3=y3-m3*x_8([c]);
    b4=y4-m4*x_8([d]);
    b5=y5-m5*x_8([e]);
Y1=(m1*x_8(1:21))+b1;
Y2=(m2*x_8(22:43))+b2;
Y3=(m3*x_8(44:65))+b3;
Y4=(m4*x_8(66:87))+b4;
Y5=(m5*x_8(88:110))+b5;

    Y = [Y1 Y2 Y3 Y4 Y5];
    figure;
    plot(x_8(1:21),Y1)
    hold on
    plot(x_8(22:43),Y2)
    hold on
    plot(x_8(44:65),Y3)
    hold on
    plot(x_8(66:87),Y4)
    hold on
    plot(x_8(88:110),Y5)
    hold on
    plot(x_8, y)

    title("tanh(x)& piecewise linear 8 bits aproximation." );
    xlabel('x');
    ylabel('tanh(x)');
```


tanh(x)& piecewise linear 8 bits aproximation.
tanh(x)
x
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-5
-4
-3
-2
-1
0
1
2
3
4
5


tanh(x)& piecewise linear 8 bits aproximation Error = 0.1687458446
error
theta
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-5
-4
-3
-2
-1
0
1
2
3
4
5


tanh(x)& piecewise linear 8 bits aproximation Error = 0.0199534970
error
theta
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
-5
-4
-3
-2
-1
0
1
2
3
4
5


tanh(x)& piecewise linear 8 bits aproximation.
tanh(x)
x
1
0.5
0
-0.5
-1
-1.5
-5
-4
-3
-2
-1
0
1
2
3
4
5

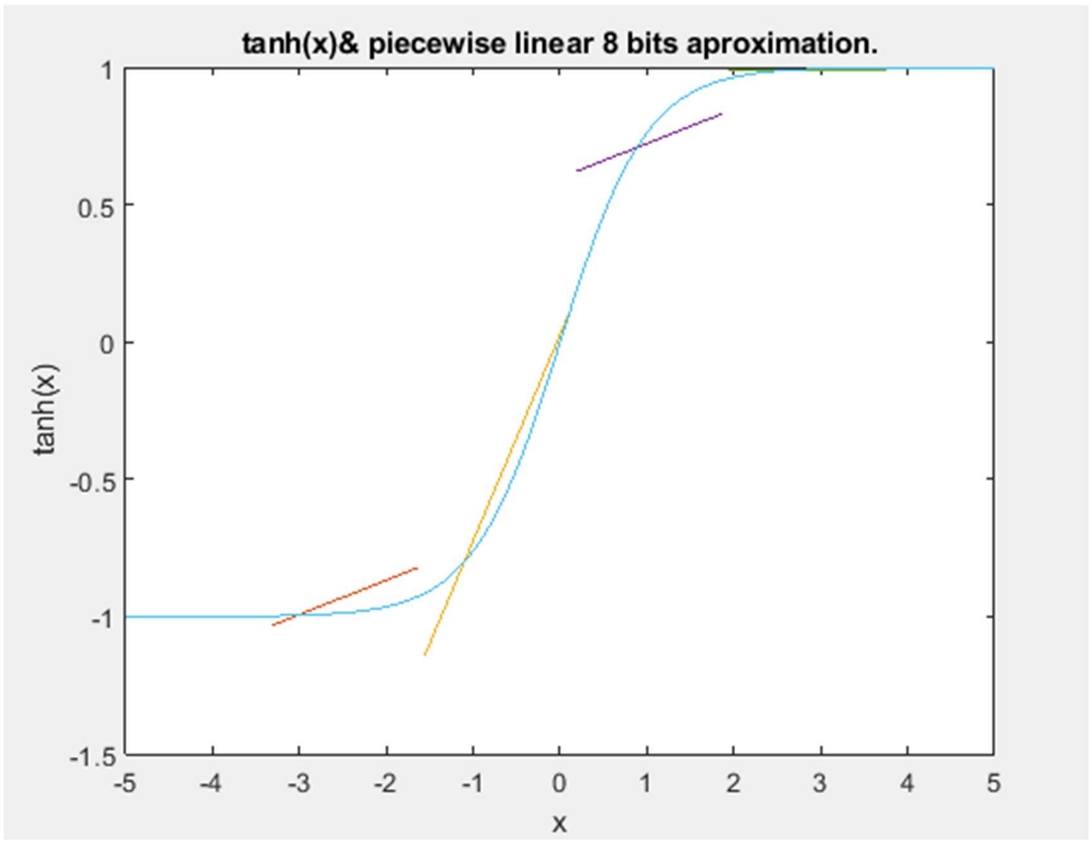


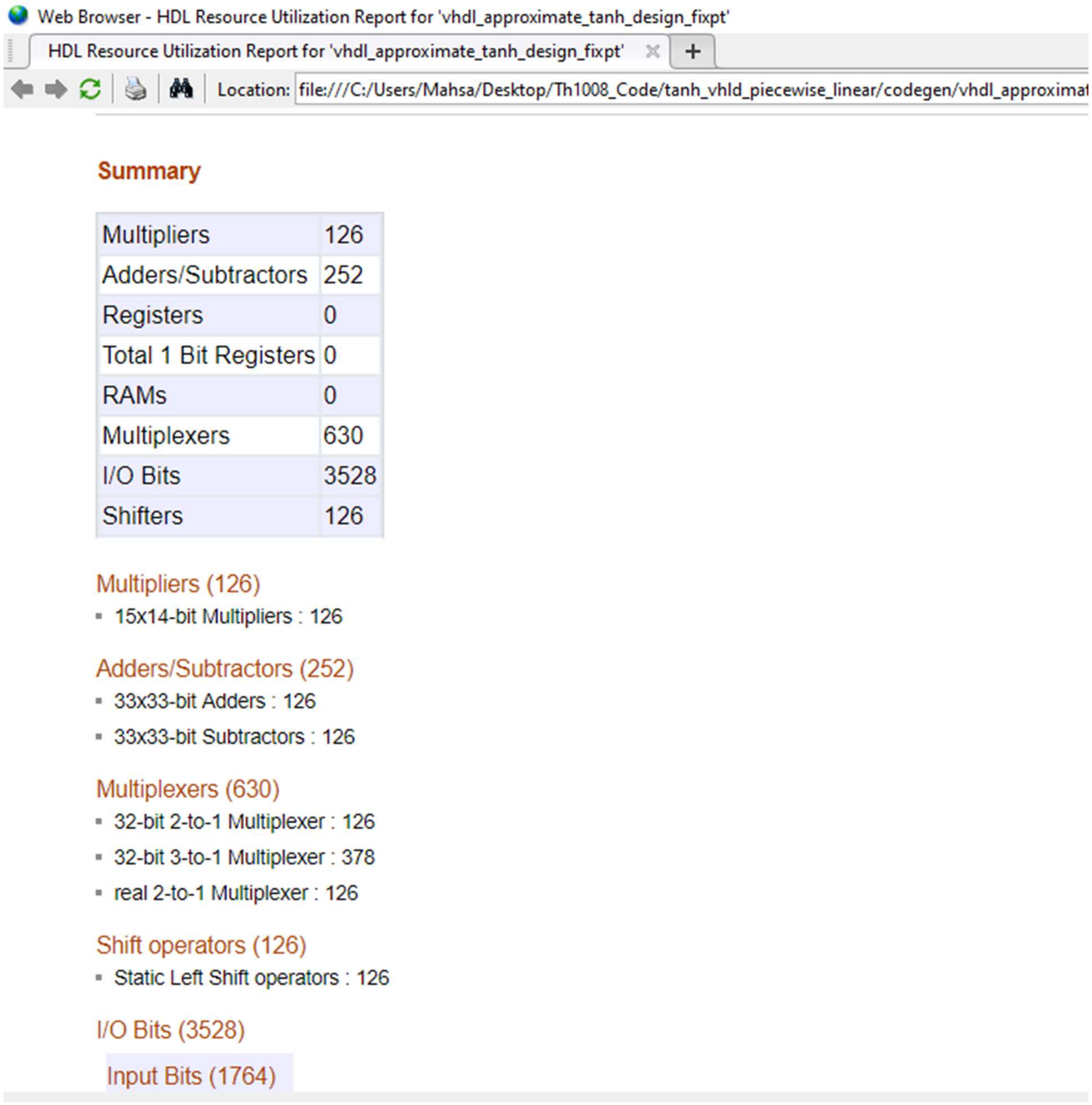
+2
vhdl_approximate_tanh_design_fixpt_pkg.vhd
vhdl_approximate_tanh_design_fixpt.vhd
piecewise_vhdl.prj
MATLAB Function
vhdl_approximate_tanh_design.m
Remove MATLAB functi
Autodefine typ
MATLAB Test Bench
vhdl_approximate_tanh_tb.m
Add fil
Workflow Advisor
Command Window
Activate Windows
Go to Settings to activate Windows.

```
USE work.vhdl_approximate_tanh_design_fixpt_pkg.ALL;

ENTITY vhdl_approximate_tanh_design_fixpt IS
  PORT( x                                 :   IN    vector_of_std_logic_vector
        y_out                             :   OUT   vector_of_std_logic_vector
        );
END vhdl_approximate_tanh_design_fixpt;


ARCHITECTURE rtl OF vhdl_approximate_tanh_design_fixpt IS

  -- Constants
  CONSTANT One                            : real :=
    2.0;  -- double
  CONSTANT C_divbyzero_p                  : real :=
    1.0E+308;  -- double
```

After specifying your design function and test bench above, use the Workflow Advisor to generate code.

```
     1     1

     1     1

     1     1

error 8 bits:0.0199534970
>>
```




Web Browser - HDL Resource Utilization Report for 'vhdl_approximate_tanh_design_fixpt'

HDL Resource Utilization Report for 'vhdl_approximate_tanh_design_fixpt'

Location: file:///C:/Users/Mahsa/Desktop/Th1008_Code/tanh_vhld_piecewise_linear/codegen/vhdl_approximat

Summary

Multipliers	126
Adders/Subtractors	252
Registers	0
Total 1 Bit Registers	0
RAMs	0
Multiplexers	630
I/O Bits	3528
Shifters	126

Multipliers (126)

- 15x14-bit Multipliers : 126

Adders/Subtractors (252)

- 33x33-bit Adders : 126
- 33x33-bit Subtractors : 126

Multiplexers (630)

- 32-bit 2-to-1 Multiplexer : 126
- 32-bit 3-to-1 Multiplexer : 378
- real 2-to-1 Multiplexer : 126

Shift operators (126)

- Static Left Shift operators : 126

I/O Bits (3528)

Input Bits (1764)