Contents

Package

```
-- -------------------------------------------------------------
--
-- File Name: hdlsrc\untitled\untitled_pkg.vhd
-- Created: 2021-08-24 21:36:42
--
-- Generated by MATLAB 9.8 and HDL Coder 3.16
--
-- -------------------------------------------------------------


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

PACKAGE untitled_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_signed16 IS ARRAY (NATURAL RANGE <>) OF signed(15 DOWNTO 0);
  TYPE vector_of_signed15 IS ARRAY (NATURAL RANGE <>) OF signed(14 DOWNTO 0);
  TYPE vector_of_signed28 IS ARRAY (NATURAL RANGE <>) OF signed(27 DOWNTO 0);
  TYPE vector_of_signed48 IS ARRAY (NATURAL RANGE <>) OF signed(47 DOWNTO 0);
END untitled_pkg;
```

Rate and Clocking Details

```
-- -------------------------------------------------------------
--
-- File Name: hdlsrc\untitled\untitled.vhd
-- Created: 2021-08-24 21:36:42
--
-- Generated by MATLAB 9.8 and HDL Coder 3.16
--
```

```
--
-- -------------------------------------------------------------
-- Rate and Clocking Details
-- -------------------------------------------------------------
-- Model base rate: 0.2
-- Target subsystem base rate: 0.2
--
-- -------------------------------------------------------------



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

ENTITY untitled IS
  PORT( in1                                     :   IN
vector_of_std_logic_vector14(0 TO 125);  -- sfix14_En10 [126]
        out1                                    :   OUT
vector_of_std_logic_vector14(0 TO 125)  -- sfix14_En13 [126]
        );
END untitled;


ARCHITECTURE rtl OF untitled IS

  -- Component Declarations
  COMPONENT vhdl_approximate_tanh_design_fixpt_slcfg
    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 COMPONENT;

  -- Component Configuration Statements
  FOR ALL : vhdl_approximate_tanh_design_fixpt_slcfg
    USE ENTITY work.vhdl_approximate_tanh_design_fixpt_slcfg(rtl);

  -- Signals
  SIGNAL y_out                                  : vector_of_std_logic_vector14(0 TO
125);  -- ufix14 [126]

BEGIN
  u_vhdl_approximate_tanh_design_fixpt_slcfg :
vhdl_approximate_tanh_design_fixpt_slcfg
    PORT MAP( x => in1,  -- sfix14_En10 [126]
              y_out => y_out  -- sfix14_En13 [126]
```

```
              );

  out1 <= y_out;

END rtl;
```

vhdl_approximate_tanh_design_fixpt_slcfg

```
-- -------------------------------------------------------------
--
-- File Name: hdlsrc\untitled\vhdl_approximate_tanh_design_fixpt_slcfg.vhd
-- Created: 2021-08-24 21:36:42
--
-- Generated by MATLAB 9.8 and HDL Coder 3.16
--
-- -------------------------------------------------------------



-- -------------------------------------------------------------
--
-- Module: vhdl_approximate_tanh_design_fixpt_slcfg
-- Source Path: untitled/vhdl_approximate_tanh_design_fixpt_slcfg
-- Hierarchy Level: 1
--
-- -------------------------------------------------------------
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
USE IEEE.numeric_std.ALL;
USE work.untitled_pkg.ALL;

ENTITY vhdl_approximate_tanh_design_fixpt_slcfg 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_slcfg;


ARCHITECTURE rtl OF vhdl_approximate_tanh_design_fixpt_slcfg IS

  -- Constants
  CONSTANT nc                             : vector_of_signed14(0 TO 12) :=
    (to_signed(16#0C2F#, 14), to_signed(16#0F6D#, 14), to_signed(16#0FEC#,
14), to_signed(16#0FFD#, 14),
     to_signed(16#1000#, 14), to_signed(16#1000#, 14), to_signed(16#1000#,
14), to_signed(16#1000#, 14),
     to_signed(16#1000#, 14), to_signed(16#1000#, 14), to_signed(16#1000#,
14), to_signed(16#1000#, 14),
     to_signed(16#1000#, 14));  -- sfix14 [13]
  CONSTANT a                              : vector_of_signed14(0 TO 12) :=
    (to_signed(16#08CA#, 14), to_signed(16#0416#, 14), to_signed(16#0203#,
14), to_signed(16#0100#, 14),
```

```
    to_signed(16#0080#, 14), to_signed(16#0040#, 14), to_signed(16#0020#,
14), to_signed(16#0010#, 14),
    to_signed(16#0008#, 14), to_signed(16#0004#, 14), to_signed(16#0002#,
14), to_signed(16#0001#, 14),
    to_signed(16#0001#, 14));  -- sfix14 [13]

  -- 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_slcfg_1_output : PROCESS (x_signed)
    VARIABLE t : vector_of_signed14(0 TO 125);
    VARIABLE y1 : signed(3 DOWNTO 0);
    VARIABLE q : signed(31 DOWNTO 0);
    VARIABLE c : signed(13 DOWNTO 0);
    VARIABLE c_0 : signed(13 DOWNTO 0);
    VARIABLE c_1 : signed(13 DOWNTO 0);
    VARIABLE c_2 : signed(13 DOWNTO 0);
    VARIABLE tmp : signed(13 DOWNTO 0);
    VARIABLE tmp_0 : signed(13 DOWNTO 0);
    VARIABLE tmp_1 : signed(13 DOWNTO 0);
    VARIABLE k : signed(15 DOWNTO 0);
    VARIABLE tmp_2 : signed(13 DOWNTO 0);
    VARIABLE c_3 : signed(13 DOWNTO 0);
    VARIABLE tmp_3 : signed(13 DOWNTO 0);
    VARIABLE tmp_4 : signed(13 DOWNTO 0);
    VARIABLE sub_cast : vector_of_signed14(0 TO 125);
    VARIABLE sub_cast_0 : vector_of_signed14(0 TO 125);
    VARIABLE add_temp : vector_of_signed16(0 TO 12);
    VARIABLE add_temp_0 : vector_of_signed16(0 TO 12);
    VARIABLE cast : vector_of_signed15(0 TO 125);
    VARIABLE cast_0 : vector_of_signed15(0 TO 125);
    VARIABLE add_cast : vector_of_signed14(0 TO 125);
    VARIABLE mul_temp : vector_of_signed28(0 TO 125);
    VARIABLE add_cast_0 : vector_of_signed14(0 TO 125);
    VARIABLE add_cast_1 : vector_of_signed14(0 TO 125);
    VARIABLE sub_cast_1 : vector_of_signed14(0 TO 12);
    VARIABLE add_cast_2 : vector_of_signed14(0 TO 12);
    VARIABLE add_cast_3 : vector_of_signed14(0 TO 125);
    VARIABLE mul_temp_0 : vector_of_signed28(0 TO 125);
    VARIABLE add_cast_4 : vector_of_signed14(0 TO 125);
    VARIABLE add_cast_5 : vector_of_signed14(0 TO 125);
    VARIABLE add_cast_6 : vector_of_signed14(0 TO 12);
    VARIABLE add_cast_7 : vector_of_signed14(0 TO 12);
    VARIABLE add_temp_1 : vector_of_signed14(0 TO 12);
    VARIABLE sub_cast_2 : vector_of_signed14(0 TO 12);
    VARIABLE sub_cast_3 : vector_of_signed14(0 TO 12);
    VARIABLE sub_temp : vector_of_signed14(0 TO 12);
    VARIABLE add_cast_8 : vector_of_signed14(0 TO 12);
```

```
    VARIABLE add_cast_9 : vector_of_signed14(0 TO 12);
    VARIABLE add_temp_2 : vector_of_signed14(0 TO 12);
    VARIABLE sub_cast_4 : vector_of_signed14(0 TO 12);
    VARIABLE sub_cast_5 : vector_of_signed14(0 TO 12);
    VARIABLE sub_temp_0 : vector_of_signed14(0 TO 12);
    VARIABLE add_temp_3 : vector_of_signed16(0 TO 12);
    VARIABLE add_temp_4 : vector_of_signed16(0 TO 11);
    VARIABLE c_4 : vector_of_signed14(0 TO 11);
    VARIABLE add_temp_5 : vector_of_signed16(0 TO 11);
    VARIABLE c_5 : vector_of_signed14(0 TO 11);
    VARIABLE add_temp_6 : vector_of_signed16(0 TO 12);
    VARIABLE add_cast_10 : vector_of_signed14(0 TO 11);
    VARIABLE add_temp_7 : vector_of_signed16(0 TO 11);
    VARIABLE sra_temp : vector_of_signed14(0 TO 11);
    VARIABLE add_temp_8 : vector_of_signed14(0 TO 11);
    VARIABLE sub_cast_6 : vector_of_signed14(0 TO 11);
    VARIABLE add_temp_9 : vector_of_signed16(0 TO 11);
    VARIABLE sra_temp_0 : vector_of_signed14(0 TO 11);
    VARIABLE sub_temp_1 : vector_of_signed14(0 TO 11);
    VARIABLE sub_cast_7 : vector_of_signed14(0 TO 12);
    VARIABLE add_cast_11 : vector_of_signed14(0 TO 12);
    VARIABLE add_cast_12 : vector_of_signed14(0 TO 12);
    VARIABLE add_cast_13 : vector_of_signed14(0 TO 12);
    VARIABLE add_temp_10 : vector_of_signed14(0 TO 12);
    VARIABLE sub_cast_8 : vector_of_signed14(0 TO 12);
    VARIABLE sub_cast_9 : vector_of_signed14(0 TO 12);
    VARIABLE sub_temp_2 : vector_of_signed14(0 TO 12);
    VARIABLE add_cast_14 : vector_of_signed14(0 TO 12);
    VARIABLE add_cast_15 : vector_of_signed14(0 TO 12);
    VARIABLE add_temp_11 : vector_of_signed14(0 TO 12);
    VARIABLE sub_cast_10 : vector_of_signed14(0 TO 12);
    VARIABLE sub_cast_11 : vector_of_signed14(0 TO 12);
    VARIABLE sub_temp_3 : vector_of_signed14(0 TO 12);
    VARIABLE mul_temp_1 : vector_of_signed48(0 TO 12);
    VARIABLE add_cast_16 : vector_of_signed16(0 TO 12);
  BEGIN
    c := to_signed(16#0000#, 14);
    c_0 := to_signed(16#0000#, 14);
    c_1 := to_signed(16#0000#, 14);
    c_2 := to_signed(16#0000#, 14);
    c_3 := to_signed(16#0000#, 14);
    tmp := to_signed(16#0000#, 14);
    tmp_0 := to_signed(16#0000#, 14);
    tmp_1 := to_signed(16#0000#, 14);
    k := to_signed(16#0000#, 16);
    tmp_2 := to_signed(16#0000#, 14);
    tmp_3 := to_signed(16#0000#, 14);
    tmp_4 := to_signed(16#0000#, 14);
    y1 := to_signed(16#0#, 4);
    q := to_signed(16#00000000#, 32);
    --HDL code generation from MATLAB function:
sf_gateway_vhdl_approximate_tanh_design_fixpt_slcfg
    --MATLAB Function 'vhdl_approximate_tanh_design_fixpt_slcfg'
    --
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    --
%
```

```
    --           Generated by MATLAB 9.8 and Fixed-Point Designer 7.0
%
    --
%
    --
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    --   y_out = 1-(2./(power(2,((2.625)*x))+1));

    FOR i IN 0 TO 125 LOOP
      y1 := x_signed(i)(13 DOWNTO 10);
      q := resize(y1, 32);
      sub_cast(i) := x_signed(i)(12 DOWNTO 0) & '0';
      sub_cast_0(i) := y1(2 DOWNTO 0) & '0' & '0' & '0' & '0' & '0' & '0' &
'0' & '0' & '0' & '0' & '0';
      tmp := sub_cast(i) - sub_cast_0(i);
      tmp_0 := to_signed(16#0000#, 14);
      tmp_1 := to_signed(16#1000#, 14);
      k := to_signed(16#0004#, 16);

      FOR n IN 0 TO 12 LOOP
        add_temp(n) := to_signed(n + 1, 16);
        c := SHIFT_RIGHT(tmp_1, to_integer(add_temp(n)));
        add_temp_0(n) := to_signed(n + 1, 16);
        c_0 := SHIFT_RIGHT(tmp_0, to_integer(add_temp_0(n)));
        IF tmp < to_signed(16#0000#, 14) THEN
          add_cast_2(n) := (resize(a((n + 1) - 1)(13 DOWNTO 1), 14)) + ('0' &
a((n + 1) - 1)(0));
          tmp := tmp + add_cast_2(n);
          sub_cast_2(n) := (resize(tmp_0(13 DOWNTO 1), 14)) + ('0' &
tmp_0(0));
          sub_cast_3(n) := (resize(c(13 DOWNTO 1), 14)) + ('0' & c(0));
          sub_temp(n) := sub_cast_2(n) - sub_cast_3(n);
          tmp_0 := sub_temp(n)(12 DOWNTO 0) & '0';
          sub_cast_4(n) := (resize(tmp_1(13 DOWNTO 1), 14)) + ('0' &
tmp_1(0));
          sub_cast_5(n) := (resize(c_0(13 DOWNTO 1), 14)) + ('0' & c_0(0));
          sub_temp_0(n) := sub_cast_4(n) - sub_cast_5(n);
          tmp_1 := sub_temp_0(n)(12 DOWNTO 0) & '0';
        ELSE
          sub_cast_1(n) := (resize(a((n + 1) - 1)(13 DOWNTO 1), 14)) + ('0' &
a((n + 1) - 1)(0));
          tmp := tmp - sub_cast_1(n);
          add_cast_6(n) := (resize(tmp_0(13 DOWNTO 1), 14)) + ('0' &
tmp_0(0));
          add_cast_7(n) := (resize(c(13 DOWNTO 1), 14)) + ('0' & c(0));
          add_temp_1(n) := add_cast_6(n) + add_cast_7(n);
          tmp_0 := add_temp_1(n)(12 DOWNTO 0) & '0';
          add_cast_8(n) := (resize(tmp_1(13 DOWNTO 1), 14)) + ('0' &
tmp_1(0));
          add_cast_9(n) := (resize(c_0(13 DOWNTO 1), 14)) + ('0' & c_0(0));
          add_temp_2(n) := add_cast_8(n) + add_cast_9(n);
          tmp_1 := add_temp_2(n)(12 DOWNTO 0) & '0';
        END IF;
        IF to_signed(n + 1, 16) = k THEN
          add_temp_3(n) := to_signed(n + 1, 16);
          c_1 := SHIFT_RIGHT(tmp_1, to_integer(add_temp_3(n)));
```

```
          add_temp_6(n) := to_signed(n + 1, 16);
          c_2 := SHIFT_RIGHT(tmp_0, to_integer(add_temp_6(n)));
          IF tmp < to_signed(16#0000#, 14) THEN
            add_cast_11(n) := (resize(a((n + 1) - 1)(13 DOWNTO 1), 14)) +
('0' & a((n + 1) - 1)(0));
            tmp := tmp + add_cast_11(n);
            sub_cast_8(n) := (resize(tmp_0(13 DOWNTO 1), 14)) + ('0' &
tmp_0(0));
            sub_cast_9(n) := (resize(c_1(13 DOWNTO 1), 14)) + ('0' & c_1(0));
            sub_temp_2(n) := sub_cast_8(n) - sub_cast_9(n);
            tmp_0 := sub_temp_2(n)(12 DOWNTO 0) & '0';
            sub_cast_10(n) := (resize(tmp_1(13 DOWNTO 1), 14)) + ('0' &
tmp_1(0));
            sub_cast_11(n) := (resize(c_2(13 DOWNTO 1), 14)) + ('0' &
c_2(0));
            sub_temp_3(n) := sub_cast_10(n) - sub_cast_11(n);
            tmp_1 := sub_temp_3(n)(12 DOWNTO 0) & '0';
          ELSE
            sub_cast_7(n) := (resize(a((n + 1) - 1)(13 DOWNTO 1), 14)) + ('0'
& a((n + 1) - 1)(0));
            tmp := tmp - sub_cast_7(n);
            add_cast_12(n) := (resize(tmp_0(13 DOWNTO 1), 14)) + ('0' &
tmp_0(0));
            add_cast_13(n) := (resize(c_1(13 DOWNTO 1), 14)) + ('0' &
c_1(0));
            add_temp_10(n) := add_cast_12(n) + add_cast_13(n);
            tmp_0 := add_temp_10(n)(12 DOWNTO 0) & '0';
            add_cast_14(n) := (resize(tmp_1(13 DOWNTO 1), 14)) + ('0' &
tmp_1(0));
            add_cast_15(n) := (resize(c_2(13 DOWNTO 1), 14)) + ('0' &
c_2(0));
            add_temp_11(n) := add_cast_14(n) + add_cast_15(n);
            tmp_1 := add_temp_11(n)(12 DOWNTO 0) & '0';
          END IF;
          mul_temp_1(n) := to_signed(16#00000003#, 32) * k;
          add_cast_16(n) := mul_temp_1(n)(15 DOWNTO 0);
          k := add_cast_16(n) + 1;
        END IF;
      END LOOP;

      IF q = to_signed(16#00000000#, 32) THEN
        tmp_2 := to_signed(16#0000#, 14);
      ELSIF q > to_signed(16#00000000#, 32) THEN
        tmp_2 := nc(to_integer(q - 1));
      ELSE
        cast(i) := resize(nc(to_integer(resize( - (resize(q, 33)), 32) - 1)),
15);
        cast_0(i) :=  - (cast(i));
        tmp_2 := cast_0(i)(13 DOWNTO 0);
      END IF;
      add_cast(i) := (resize(tmp_1(13 DOWNTO 1), 14)) + ('0' & tmp_1(0));
      mul_temp(i) := tmp_2 * tmp_0;
      add_cast_0(i) := mul_temp(i)(25 DOWNTO 12) + ('0' & mul_temp(i)(11));
      add_cast_1(i) := (resize(add_cast_0(i)(13 DOWNTO 1), 14)) + ('0' &
add_cast_0(i)(0));
      c_3 := add_cast(i) + add_cast_1(i);
      add_cast_3(i) := (resize(tmp_0(13 DOWNTO 1), 14)) + ('0' & tmp_0(0));
```

```
      mul_temp_0(i) := tmp_2 * tmp_1;
      add_cast_4(i) := mul_temp_0(i)(25 DOWNTO 12) + ('0' &
mul_temp_0(i)(11));
      add_cast_5(i) := (resize(add_cast_4(i)(13 DOWNTO 1), 14)) + ('0' &
add_cast_4(i)(0));
      tmp_3 := add_cast_3(i) + add_cast_5(i);
      tmp_4 := to_signed(16#0000#, 14);

      FOR n_0 IN 0 TO 11 LOOP
        IF tmp_3 < to_signed(16#0000#, 14) THEN
          add_temp_5(n_0) := to_signed(n_0 + 1, 16);
          c_5(n_0) := SHIFT_RIGHT(c_3, to_integer(add_temp_5(n_0)));
          tmp_3 := tmp_3 + c_5(n_0);
          sub_cast_6(n_0) := (resize(tmp_4(13 DOWNTO 1), 14)) + ('0' &
tmp_4(0));
          add_temp_9(n_0) := to_signed(n_0 + 1, 16);
          sra_temp_0(n_0) := SHIFT_RIGHT(to_signed(16#0800#, 14),
to_integer(add_temp_9(n_0)));
          sub_temp_1(n_0) := sub_cast_6(n_0) - sra_temp_0(n_0);
          tmp_4 := sub_temp_1(n_0)(12 DOWNTO 0) & '0';
        ELSE
          add_temp_4(n_0) := to_signed(n_0 + 1, 16);
          c_4(n_0) := SHIFT_RIGHT(c_3, to_integer(add_temp_4(n_0)));
          tmp_3 := tmp_3 - c_4(n_0);
          add_cast_10(n_0) := (resize(tmp_4(13 DOWNTO 1), 14)) + ('0' &
tmp_4(0));
          add_temp_7(n_0) := to_signed(n_0 + 1, 16);
          sra_temp(n_0) := SHIFT_RIGHT(to_signed(16#0800#, 14),
to_integer(add_temp_7(n_0)));
          add_temp_8(n_0) := add_cast_10(n_0) + sra_temp(n_0);
          tmp_4 := add_temp_8(n_0)(12 DOWNTO 0) & '0';
        END IF;
      END LOOP;

      t(i) := tmp_4;
      y_out_tmp(i) <= t(i)(12 DOWNTO 0) & '0';
    END LOOP;

  END PROCESS vhdl_approximate_tanh_design_fixpt_slcfg_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 = cordictanh(x);
end
```

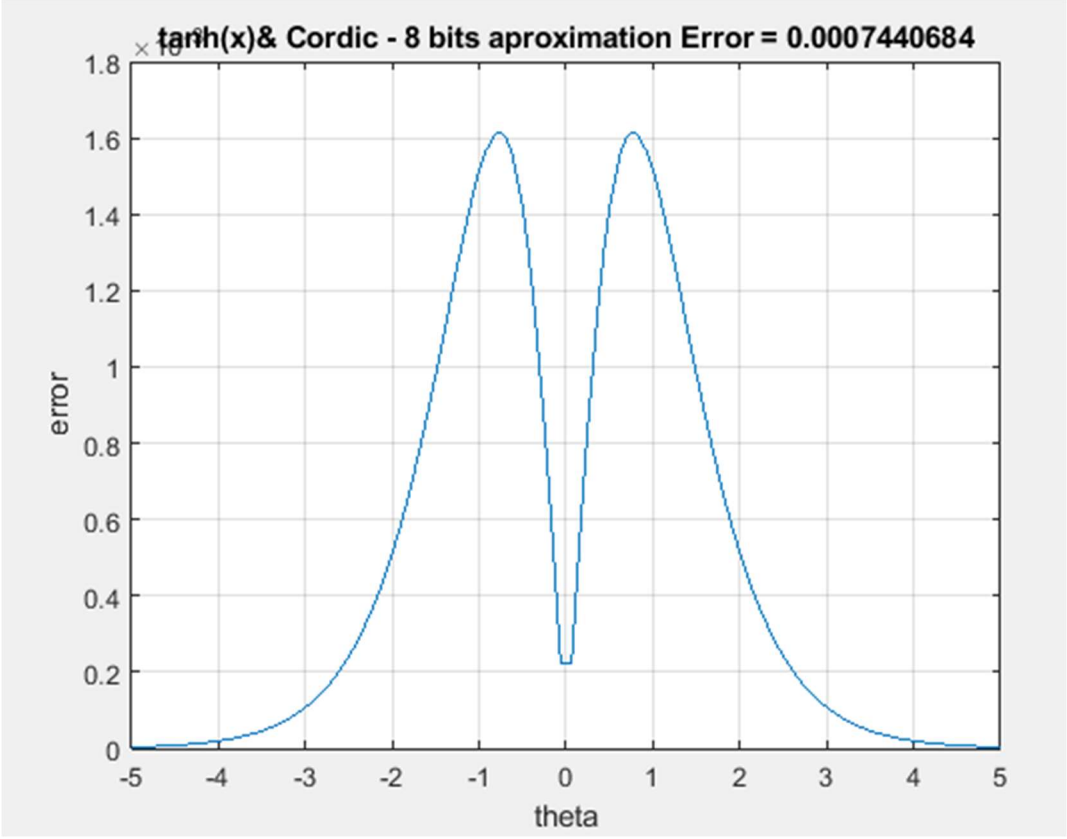
Test Bunch

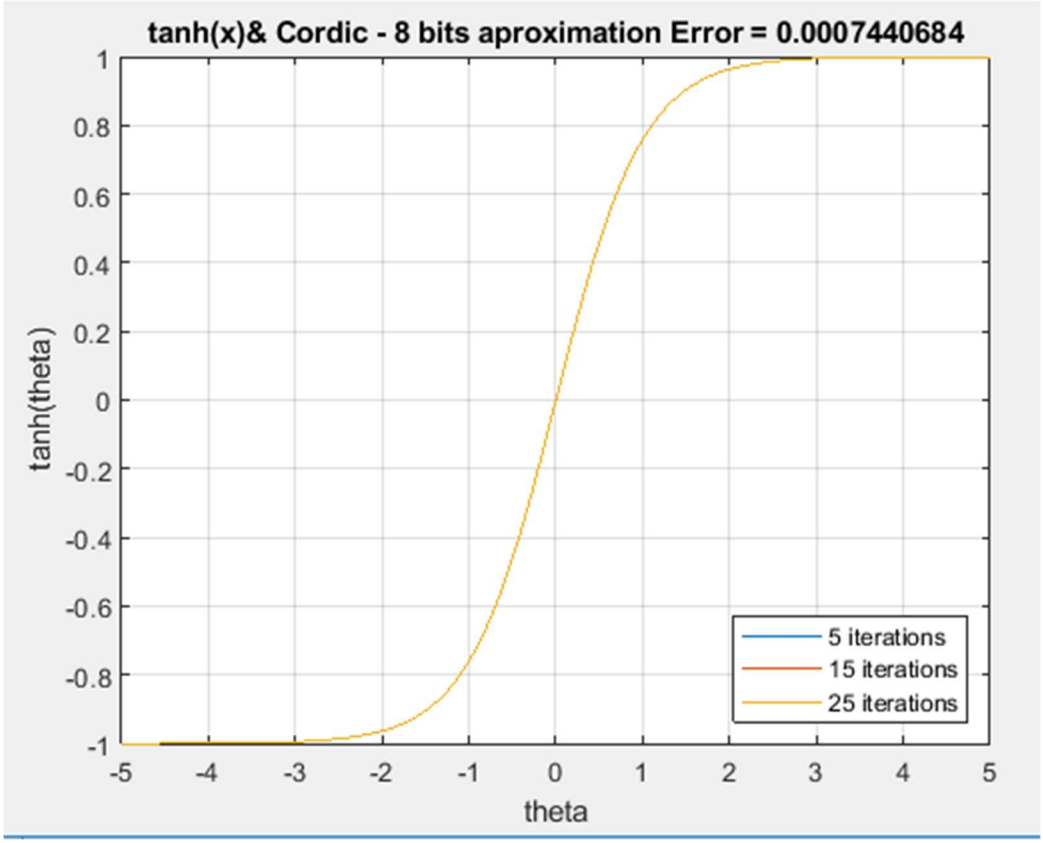
```
function vhdl_approximate_tanh_tb
  % Test inputs
  x_8  = fi(-5:0.08:5,1,8);
  x_16 = fi(-5:0.08:5,1,16);
  x_32 = fi(-5:0.08:5,1,32);
  y = tanh( double(x_8) );
figure;
iteration = [5,15,25];
 for i = 1:length(iteration)
     for niters = iteration(i)
        y_cordic_8 = vhdl_approximate_tanh_design( double( x_8));
        %  y_cordic_16 = vhdl_approximate_tanh_design( double( x_16));   % 16
bits
        %  y_cordic_32 = vhdl_approximate_tanh_design( double( x_32));   % 32
bits
        plot(x_8, y_cordic_8);
        grid on;
        hold on;
     end
 end
  error8_ = RMSE(y_cordic_8,y);
  formatSpec = '%.20f';
  error8_=num2str(error8_,formatSpec);
  disp("error 8 bits:" + error8_);

xlabel('theta');
ylabel('tanh(theta)');
title("tanh(x)& Cordic - 8 bits aproximation Error = " + error8_);

legend('5 iterations', '15 iterations', '25
iterations','Location','southeast');
grid on;


  figure;
  err = abs(y - double(y_cordic_8));
  plot(x_8, err);
  xlabel('theta');
  ylabel('error');
  title("tanh(x)& Cordic - 8 bits aproximation Error = " + error8_);
  grid on;

% figure;
% err = abs(tanh(x) - double(T_cordic));
% plot(x, err);
% xlabel('theta');
% ylabel('error');
% error = RMSE(tanh(x),T_cordic);
% formatSpec = '%.10f';
% error_=num2str(error,formatSpec);
```

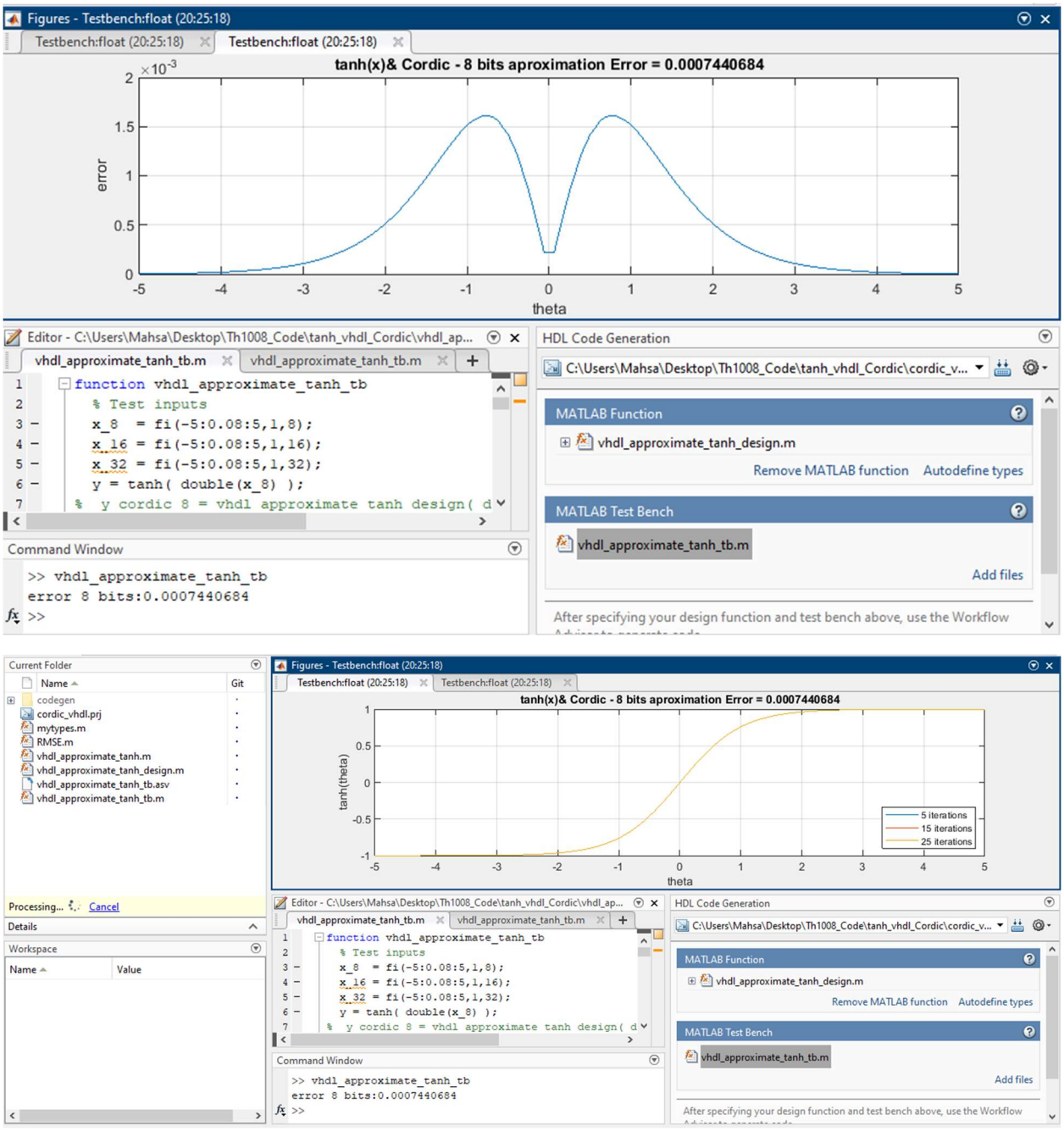
```
% disp(num2str(error,formatSpec));
% title("tanh(x)& cordic aproximation Error = " + error_);
% xlabel('x');
% ylabel('tanh(x)');


end
```

Plots

```
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);
  y = tanh( double(x_8) );
figure;
iteration = [5,15,25];
 for i = 1:length(iteration)
     for niters = iteration(i)
        y_cordic_8 = vhdl_approximate_tanh_design( double( x_8));
        %  y_cordic_16 = vhdl_approximate_tanh_design( double( x_16));   % 16 bits
        %  y_cordic_32 = vhdl_approximate_tanh_design( double( x_32));   % 32 bits
        plot(x_8, y_cordic_8);
        grid on;
        hold on;
     end
 end
  error8_ = RMSE(y_cordic_8,y);
  formatSpec = '%.10f';
  error8_=num2str(error8_,formatSpec);
  disp("error 8 bits:" + error8_);

xlabel('theta');
ylabel('tanh(theta)');
title("tanh(x)& Cordic - 8 bits aproximation Error = " + error8_);
```


tanh(x)& Cordic - 8 bits aproximation Error = 0.0007440684
error
theta


tanh(x)& Cordic - 8 bits aproximation Error = 0.0007440684
tanh(theta)
theta
5 iterations
15 iterations
25 iterations


Figures - Testbench:float (20:25:18)
Testbench:float (20:25:18)
Testbench:float (20:25:18)
tanh(x)& Cordic - 8 bits aproximation Error = 0.0007440684
×10^-3
error
theta
Editor - C:\Users\Mahsa\Desktop\Th1008_Code\tanh_vhdl_Cordic\vhdl_ap...
vhdl_approximate_tanh_tb.m
vhdl_approximate_tanh_tb.m
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);
y = tanh(double(x_8));
% y cordic 8 = vhdl approximate tanh design(d
Command Window
>> vhdl_approximate_tanh_tb
error 8 bits:0.0007440684
>>
HDL Code Generation
C:\Users\Mahsa\Desktop\Th1008_Code\tanh_vhdl_Cordic\cordic_v...
MATLAB Function
vhdl_approximate_tanh_design.m
Remove MATLAB function
Autodefine types
MATLAB Test Bench
vhdl_approximate_tanh_tb.m
Add files
After specifying your design function and test bench above, use the Workflow
Current Folder
Name
Git
codegen
cordic_vhdl.prj
mytypes.m
RMSE.m
vhdl_approximate_tanh.m
vhdl_approximate_tanh_design.m
vhdl_approximate_tanh_tb.asv
vhdl_approximate_tanh_tb.m
Processing...
Cancel
Details
Workspace
Name
Value
Figures - Testbench:float (20:25:18)
Testbench:float (20:25:18)
Testbench:float (20:25:18)
tanh(x)& Cordic - 8 bits aproximation Error = 0.0007440684
tanh(theta)
theta
5 iterations
15 iterations
25 iterations
Editor - C:\Users\Mahsa\Desktop\Th1008_Code\tanh_vhdl_Cordic\vhdl_ap...
vhdl_approximate_tanh_tb.m
vhdl_approximate_tanh_tb.m
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);
y = tanh(double(x_8));
% y cordic 8 = vhdl approximate tanh design(d
Command Window
>> vhdl_approximate_tanh_tb
error 8 bits:0.0007440684
>>
HDL Code Generation
C:\Users\Mahsa\Desktop\Th1008_Code\tanh_vhdl_Cordic\cordic_v...
MATLAB Function
vhdl_approximate_tanh_design.m
Remove MATLAB function
Autodefine types
MATLAB Test Bench
vhdl_approximate_tanh_tb.m
Add files
After specifying your design function and test bench above, use the Workflow

HDL Resource Utilization Report ('vhdl_approximate_tanh_design_fixpt')

Generated on 2021-08-21 20:26:43

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)

- x : 1764 bits

Output Bits (1764)

- y_out : 1764 bits