

## CORDIC TESTBENCH

-- **** STUDENT: 64000225 .....	2
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	2
-- **** STUDENT: 64190088 .....	7
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	7
-- **** STUDENT: 64200163 .....	12
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	12
-- **** STUDENT: 64200296 .....	17
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	17
-- **** STUDENT: 64200385 .....	22
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	22
-- **** STUDENT: 64210132 .....	27
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	27
-- **** STUDENT: 64210290 .....	32
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	32
-- **** STUDENT: 64210382 .....	37
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	37
-- **** STUDENT: 64210384 .....	42
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	42
-- **** STUDENT: 64210445 .....	47
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	47
-- **** STUDENT: 64210455 .....	52
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	52
-- **** STUDENT: 64210457 .....	57
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	57
-- **** PREDLOGA VAJE .....	62
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	62

```

-- *****
-- **** STUDENT: 64000225
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant      PERIOD : time := 200 ns;

```

```
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
    VARIABLE RES_LINE : LINE;
```

```
    variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
    variable actual_sinus : real := sin( fi );
```

```
    variable actual_cosinus : real := cos( fi );
```

```
    variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
    variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```

BEGIN
  write( RES_LINE, nRST, right, 1 );
  write( RES_LINE, string'( "," ) );
  write( RES_LINE, start, right, 1 );
  write( RES_LINE, string'( "," ) );
  write( RES_LINE, done, right, 1 );
  write( RES_LINE, string'( "," ) );
  write( RES_LINE, real'( fi ) );
  write( RES_LINE, string'( "," ) );
  write( RES_LINE, real'( cordic_sin ) );
  write( RES_LINE, string'( "," ) );
  write( RES_LINE, real'( cordic_cosinus ) );
  write( RES_LINE, string'( "," ) );
  write( RES_LINE, real'( actual_sin ) );
  write( RES_LINE, string'( "," ) );
  write( RES_LINE, real'( actual_cosinus ) );
  write( RES_LINE, string'( "," ) );
  write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
  write( RES_LINE, string'( "," ) );
  write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
  writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

```

BEGIN
  write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
  writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

```

```

angle <= Conv2fixedPt( PI / real( 2.0 ), width );
WAIT FOR PERIOD;
nRST <= '0';
WAIT FOR PERIOD;
nRST <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST <= '0';
WAIT FOR PERIOD;
nRST <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST <= '0';
WAIT FOR PERIOD;
nRST <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST <= '0';
WAIT FOR PERIOD;
nRST <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```
END PROCESS;
```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64190088
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
    VARIABLE RES_LINE : LINE;
```

```
    variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
    variable actual_sinus : real := sin( fi );
```

```
    variable actual_cosinus : real := cos( fi );
```

```
    variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
    variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```
END PROCESS;
```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64200163
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
        VARIABLE RES_LINE : LINE;
```

```
        variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
        variable actual_sinus : real := sin( fi );
```

```
        variable actual_cosinus : real := cos( fi );
```

```
        variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
        variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```

END PROCESS;

```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64200296
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
    VARIABLE RES_LINE : LINE;
```

```
    variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
    variable actual_sinus : real := sin( fi );
```

```
    variable actual_cosinus : real := cos( fi );
```

```
    variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
    variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```

END PROCESS;

```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64200385
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
        VARIABLE RES_LINE : LINE;
```

```
        variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
        variable actual_sinus : real := sin( fi );
```

```
        variable actual_cosinus : real := cos( fi );
```

```
        variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
        variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
    BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```

END PROCESS;

```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64210132
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
        VARIABLE RES_LINE : LINE;
```

```
        variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
        variable actual_sinus : real := sin( fi );
```

```
        variable actual_cosinus : real := cos( fi );
```

```
        variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
        variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```

END PROCESS;

```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64210290
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
        VARIABLE RES_LINE : LINE;
```

```
        variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
        variable actual_sinus : real := sin( fi );
```

```
        variable actual_cosinus : real := cos( fi );
```

```
        variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
        variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```

END PROCESS;

```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64210382
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
        VARIABLE RES_LINE : LINE;
```

```
        variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
        variable actual_sinus : real := sin( fi );
```

```
        variable actual_cosinus : real := cos( fi );
```

```
        variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
        variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sinuso ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sinuso ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sinuso-cordic_sinuso ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sinuso ), CORDIC( cosinus ),
ACCURATE( sinuso ), ACCURATE( cosinus ), ABSDELTA( sinuso ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST <= '1';
WAIT FOR PERIOD;
nRST <= '0';
WAIT FOR PERIOD;
nRST <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinuso, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```
END PROCESS;
```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64210384
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
        VARIABLE RES_LINE : LINE;
```

```
        variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
        variable actual_sinus : real := sin( fi );
```

```
        variable actual_cosinus : real := cos( fi );
```

```
        variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
        variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```

END PROCESS;

```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64210445
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
    VARIABLE RES_LINE : LINE;
```

```
    variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
    variable actual_sinus : real := sin( fi );
```

```
    variable actual_cosinus : real := cos( fi );
```

```
    variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
    variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```

END PROCESS;

```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64210455
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
        VARIABLE RES_LINE : LINE;
```

```
        variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
        variable actual_sinus : real := sin( fi );
```

```
        variable actual_cosinus : real := cos( fi );
```

```
        variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
        variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
    BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```

END PROCESS;

```

```
END testbench_arch;
```

```

-- *****
-- **** STUDENT: 64210457
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
    VARIABLE RES_LINE : LINE;
```

```
    variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
    variable actual_sinus : real := sin( fi );
```

```
    variable actual_cosinus : real := cos( fi );
```

```
    variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
    variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```

END PROCESS;

```

```
END testbench_arch;
```

```

-- *****
-- **** PREDLOGA VAJE
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library WORK;
use WORK.cordic_pkg.all;
library STD;
use STD.textio.all;
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
USE IEEE.MATH_REAL.ALL;
USE IEEE.STD_LOGIC_TEXTIO.ALL;

ENTITY CORDIC_tb IS
GENERIC(
    width : integer := 32
);
END CORDIC_tb;

ARCHITECTURE testbench_arch OF CORDIC_tb IS

    FILE RESULTS: TEXT OPEN WRITE_MODE IS "cordic.csv";

    COMPONENT cordic
    PORT (
        clk, nRST, start : In std_logic;
        angle : In std_logic_vector ( width - 1 DownTo 0 );
        sin, cos : Out std_logic_vector ( width - 1 DownTo 0 );
        done : Out std_logic
    );
    END COMPONENT;

    SIGNAL          clk, nRST, start, done : std_logic := '0';
    SIGNAL          angle, sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 ) := ( others => '0' );

    constant PERIOD : time := 200 ns;
    constant DUTY_CYCLE : real := 0.5;

```

```
constant    OFFSET : time := 100 ns;
```

```
BEGIN
```

```
UUT : cordic
```

```
PORT MAP (
```

```
clk => clk,
```

```
nRST => nRST,
```

```
start => start,
```

```
angle => angle,
```

```
sin => sinus,
```

```
cos => cosinus,
```

```
done => done
```

```
);
```

```
PROCESS
```

```
BEGIN
```

```
WAIT for OFFSET;
```

```
CLOCK_LOOP : LOOP
```

```
clk  <= '0';
```

```
WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
```

```
clk  <= '1';
```

```
WAIT FOR ( PERIOD * DUTY_CYCLE );
```

```
END LOOP CLOCK_LOOP;
```

```
END PROCESS;
```

```
PROCESS
```

```
    PROCEDURE Log_variables(
```

```
        nRST, start : std_logic;
```

```
        angle : std_logic_vector ( width - 1 DownTo 0 );
```

```
        sinus, cosinus : std_logic_vector ( width - 1 DownTo 0 );
```

```
        done : std_logic
```

```
    ) IS
```

```
    VARIABLE RES_LINE : LINE;
```

```
    variable fi : real := Conv2real( to_integer( unsigned( angle ) ), width );
```

```
    variable actual_sinus : real := sin( fi );
```

```
    variable actual_cosinus : real := cos( fi );
```

```
    variable cordic_sinus : real := Conv2real( to_integer( unsigned( sinus ) ), width );
```

```
    variable cordic_cosinus : real := Conv2real( to_integer( unsigned( cosinus ) ), width );
```

```
BEGIN
```

```

write( RES_LINE, nRST, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, start, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, done, right, 1 );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( fi ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( cordic_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_sin ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( actual_cosinus ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_sin-cordic_sin ) ) );
write( RES_LINE, string'( "," ) );
write( RES_LINE, real'( abs( actual_cosinus-cordic_cosinus ) ) );
writeline( RESULTS, RES_LINE );
END;
variable HDR_line : LINE;

```

BEGIN

```

write( HDR_line, string'( " nRST, start, done, angle, CORDIC( sin ), CORDIC( cosinus ),
ACCURATE( sin ), ACCURATE( cosinus ), ABSDELTA( sin ), ABSDELTA( cosinus )" ) );
writeline( RESULTS, HDR_line );

```

```

WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sin, cosinus, done );

angle <= Conv2fixedPt( PI / real( 2.0 ), width );

```

```

WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 3.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );

angle <= Conv2fixedPt( PI / real( 4.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

angle <= Conv2fixedPt( PI / real( 6.0 ), width );
WAIT FOR PERIOD;
nRST  <= '0';
WAIT FOR PERIOD;
nRST  <= '1';
WAIT FOR PERIOD;
start <= '1';
WAIT FOR ( ( width + 2 ) * PERIOD );
Log_variables( nRST, start,angle, sinus, cosinus, done );

WAIT; -- this line to avoid repetition of simulation

```

```
END PROCESS;
```

```
END testbench_arch;
```

