

## BARREL SHIFTER TESTBENCH

-- **** STUDENT: 64000225 .....	2
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	2
-- **** STUDENT: 64190088 .....	5
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	5
-- **** STUDENT: 64200163 .....	8
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	8
-- **** STUDENT: 64200296 .....	11
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	11
-- **** STUDENT: 64200385 .....	14
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	14
-- **** STUDENT: 64210132 .....	17
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	17
-- **** STUDENT: 64210290 .....	20
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	20
-- **** STUDENT: 64210382 .....	23
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	23
-- **** STUDENT: 64210384 .....	26
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	26
-- **** STUDENT: 64210445 .....	29
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	29
-- **** STUDENT: 64210455 .....	32
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	32
-- **** STUDENT: 64210457 .....	35
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	35
-- **** PREDLOGA VAJE .....	38
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	38

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
    input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output      : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
        VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
    
```

```
write( RES_LINE, string'( " ," ) );  
hwrite( RES_LINE, input, right, 4 );  
write( RES_LINE, string'( " ," ) );  
hwrite( RES_LINE, output, right, 4 );  
writeline( RESULTS, RES_LINE );  
END;
```

```
variable HDR_line : LINE;
```

```
BEGIN
```

```
write( HDR_line, string'( " n, input, output" ) );  
writeline( RESULTS, HDR_line );
```

```
WAIT FOR PERIOD;
```

```
FOR i IN 0 TO width - 1 LOOP  
  n      <= std_logic_vector( to_unsigned( i, n'length ) );  
  WAIT FOR PERIOD;  
  Log_variables( n, input, output );  
END LOOP;
```

```
WAIT;
```

```
END PROCESS;
```

```
END rtl;
```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output      : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
        VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
        write( RES_LINE, string'( ",", " ) );
    END IS
END PROCESS;

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```



```

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output     : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
        VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
        write( RES_LINE, string'( ",", " ) );
    END IS
END PROCESS;

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output      : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
        VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
        write( RES_LINE, string'( ",", " ) );
    END IS
END PROCESS;

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

    SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
    SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
    SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

    constant    PERIOD : time := 200 ns;
    constant    DUTY_CYCLE : real := 0.5;
    constant    OFFSET : time := 100 ns;

BEGIN
    UUT : barrel_shifter_sra
        GENERIC MAP (
            width => width,          size => size
        )
    PORT MAP (
        input => input,          output => output,          n => n
    );

    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 (
            n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
            input      : std_logic_vector ( width - 1 DownTo 0 );
            output      : std_logic_vector ( width - 1 DownTo 0 )
        ) IS
            VARIABLE RES_LINE : LINE;
        BEGIN
            hwrite( RES_LINE, "000" & n, right, 2 );
            write( RES_LINE, string'( ",", " ) );

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```



```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

    SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
    SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
    SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

    constant    PERIOD : time := 200 ns;
    constant    DUTY_CYCLE : real := 0.5;
    constant    OFFSET : time := 100 ns;

BEGIN
    UUT : barrel_shifter_sra
        GENERIC MAP (
            width => width,          size => size
        )
    PORT MAP (
        input => input,          output => output,          n => n
    );

    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 (
            n
                : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
            input : std_logic_vector ( width - 1 DownTo 0 );
            output : std_logic_vector ( width - 1 DownTo 0 )
        ) IS
            VARIABLE RES_LINE : LINE;
        BEGIN
            hwrite( RES_LINE, "000" & n, right, 2 );
            write( RES_LINE, string'( ",", " ) );

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " ," ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output     : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
    VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
        write( RES_LINE, string'( ",", " ) );

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output     : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
        VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
        write( RES_LINE, string'( ",", " ) );
    END IS
END PROCESS;

```



```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output     : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
    VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
        write( RES_LINE, string'( ",", " ) );

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

    SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
    SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
    SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

    constant    PERIOD : time := 200 ns;
    constant    DUTY_CYCLE : real := 0.5;
    constant    OFFSET : time := 100 ns;

BEGIN
    UUT : barrel_shifter_sra
        GENERIC MAP (
            width => width,          size => size
        )
    PORT MAP (
        input => input,          output => output,          n => n
    );

    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 (
            n
                : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
            input : std_logic_vector ( width - 1 DownTo 0 );
            output : std_logic_vector ( width - 1 DownTo 0 )
        ) IS
            VARIABLE RES_LINE : LINE;
        BEGIN
            hwrite( RES_LINE, "000" & n, right, 2 );
            write( RES_LINE, string'( ",", " ) );

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```



```

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output      : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
        VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
        write( RES_LINE, string'( ",", " ) );
    END IS
END PROCESS;

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output      : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
        VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
        write( RES_LINE, string'( ",", " ) );
    END IS
END PROCESS;

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " , " ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```

```

-- *****
-- **** 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 barrel_shifter_sra_tb IS
GENERIC(
    width : integer := 32;
    size : integer := 32
);
END barrel_shifter_sra_tb;

ARCHITECTURE rtl OF barrel_shifter_sra_tb IS

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

    COMPONENT barrel_shifter_sra IS
    GENERIC(
        width : integer := 32;
        size : integer := 32
    );
    PORT(
        input : IN std_logic_vector ( WIDTH-1 DOWNT0 0 );
        output : OUT std_logic_vector ( WIDTH-1 DOWNT0 0 );
        n : IN std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 )
    );
    END COMPONENT;

    SIGNAL        clk : std_logic := '0';

```

```

SIGNAL      input : std_logic_vector ( WIDTH-1 DOWNT0 0 ) := ( ( WIDTH-1 ) => '1', others => '0' );
SIGNAL      output : std_logic_vector ( WIDTH-1 DOWNT0 0 );
SIGNAL      n : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );

constant    PERIOD : time := 200 ns;
constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;

BEGIN
UUT : barrel_shifter_sra
    GENERIC MAP (
        width => width,          size => size
    )
PORT MAP (
input => input,          output => output,          n => n
);

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 (
        n          : std_logic_vector ( sizeof( size - 1 ) - 1 DOWNT0 0 ) := ( others => '0' );
        input      : std_logic_vector ( width - 1 DownTo 0 );
        output     : std_logic_vector ( width - 1 DownTo 0 )
    ) IS
    VARIABLE RES_LINE : LINE;
    BEGIN
        hwrite( RES_LINE, "000" & n, right, 2 );
        write( RES_LINE, string'( ",", " ) );

```

```

        hwrite( RES_LINE, input, right, 4 );
        write( RES_LINE, string'( " ," ) );
        hwrite( RES_LINE, output, right, 4 );
        writeline( RESULTS, RES_LINE );
        END;

    variable HDR_line : LINE;

BEGIN
    write( HDR_line, string'( " n, input, output" ) );
    writeline( RESULTS, HDR_line );

    WAIT FOR PERIOD;

    FOR i IN 0 TO width - 1 LOOP
        n      <= std_logic_vector( to_unsigned( i, n'length ) );
        WAIT FOR PERIOD;
        Log_variables( n, input, output );
    END LOOP;

    WAIT;

END PROCESS;

END rtl;

```



