

-- **** STUDENT: 64000225.....	2
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	2
-- **** STUDENT: 64200100.....	4
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	4
-- **** STUDENT: 64200112.....	6
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	6
-- **** STUDENT: 64200163.....	8
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	8
-- **** STUDENT: 64200238.....	10
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	10
-- **** STUDENT: 64200288.....	12
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	12
-- **** STUDENT: 64200296.....	14
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	14
-- **** STUDENT: 64200385.....	17
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	17
-- **** STUDENT: 64210113.....	19
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	19
-- **** STUDENT: 64210290.....	21
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	21
-- **** STUDENT: 64210382.....	24
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	24
-- **** STUDENT: 64210384.....	26
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	26
-- **** STUDENT: 64210386.....	29
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	29
-- **** STUDENT: 64210457.....	32
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	32

```

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

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY testbench IS
    generic(
        n_addr      : INTEGER := 2;
        bus_width    : INTEGER := 8 );
END testbench;

ARCHITECTURE behavior OF testbench IS

    COMPONENT muxnto1_bus IS
        generic(
            n_addr      : INTEGER := 2;
            bus_width    : INTEGER := 8 );
        PORT (
            s            : IN    std_logic_vector( n_addr - 1 downto 0 );
            w            : IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
            f            : OUT   std_logic_vector( bus_width - 1 downto 0 )
        );
    END COMPONENT;

    SIGNAL      s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
    SIGNAL      w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => ( others =>
'0' ) );

    signal      f, f_prog : std_logic_vector( bus_width - 1 downto 0 );

    signal      trace : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );

BEGIN

    uut: muxnto1_bus

```

```
GENERIC MAP ( n_addr => n_addr, bus_width => bus_width )
```

```
PORT MAP(
```

```
    s => s,      w => w,      f => f
```

```
);
```

```
tb : PROCESS
```

```
variable input_slv : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );
```

```
BEGIN
```

```
    for input in 0 to 2**16 - 1 loop
```

```
        input_slv := std_logic_vector( to_unsigned( input, input_slv'length ) );
```

```
        trace <= input_slv;
```

```
        for idx in 0 to bus_width-1 loop
```

```
            for jdx in 0 to 2**n_addr - 1 loop
```

```
                w( jdx, idx )<= input_slv( idx * 2**n_addr + jdx );
```

```
            end loop;
```

```
        end loop;
```

```
        for s_int in 0 to 2**n_addr - 1 loop
```

```
            s      <= std_logic_vector( to_unsigned( s_int, s'length ) );
```

```
            wait for 1 ns;
```

```
            assert f = f_prog report "FAILURE!";
```

```
        end loop;
```

```
    end loop;
```

```
    wait;
```

```
END PROCESS tb;
```

```
mux_prog : process ( w, s )
```

```
variable f_temp : std_logic_vector( bus_width - 1 downto 0 );
```

```
begin
```

```
    for i in 0 to bus_width-1 loop
```

```
        f_temp( i ) := w( to_integer( unsigned( s ) ), i );
```

```
    end loop;
```

```
    f_prog <= f_temp;
```

```
end process;
```

```
END;
```

```

-- *****
-- **** STUDENT: 64200100
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY test IS
generic( n_addr      :INTEGER := 2;
bus_width:INTEGER := 8 );
END test;

ARCHITECTURE behavior OF test IS

COMPONENT muxnto1_bus IS
generic( n_addr      :INTEGER := 2;
bus_width:INTEGER:=8 );
PORT (
s :    IN      std_logic_vector( n_addr-1 downto 0 );
w :    IN      muxnto1_bus_type( 2**n_addr-1 DOWNT0 0, bus_width-1 DOWNT0 0 );
f :    OUT     std_logic_vector( bus_width-1 downto 0 ) );
END COMPONENT;

SIGNAL      s : std_logic_vector( n_addr-1 downto 0 ):= ( others=>'0' );
SIGNAL      w : muxnto1_bus_type( 2**n_addr-1 DOWNT0 0, bus_width-1 DOWNT0 0 ):= ( others =>( others=>'0' ) );

signal      f, fP : std_logic_vector( bus_width-1 downto 0 );

signal      sl : std_logic_vector( 2**n_addr*bus_width-1 downto 0 ):= ( others => '0' );

BEGIN
uut: muxnto1_bus
GENERIC MAP ( n_addr => n_addr, bus_width => bus_width )
PORT MAP( s => s,w => w,f => f );

testB : PROCESS

```

```

variable vho : std_logic_vector( 2**n_addr*bus_width-1 downto 0 ):= ( others=>'0' );
BEGIN
for VH in 0 to 2**16 - 1 loop
vho := std_logic_vector( to_unsigned( VH, vho'length ) );
s1    <= vho;
for i in 0 to bus_width-1 loop
for j in 0 to 2**n_addr - 1 loop
w( j, i )    <= vho( i * 2**n_addr + j );
end loop;
end loop;
for Si in 0 to 2**n_addr - 1 loop
s    <= std_logic_vector( to_unsigned( Si, s'length ) );
wait for 1 ns;
assert f = fP;
end loop;
end loop;
wait;
END PROCESS testB;

mux2 : process ( w, s )
variable fZ : std_logic_vector( bus_width - 1 downto 0 );
begin
for i in 0 to bus_width-1 loop
fZ( i ) := w( to_integer( unsigned( s ) ), i );
end loop;
fP    <= fZ;
end process;
END;

```

```

-- *****
-- **** STUDENT: 64200112
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY muxnto1_bus_tb IS
    generic(
        n_addr      : INTEGER := 2;
        bus_width    : INTEGER := 8 );
END muxnto1_bus_tb;

ARCHITECTURE behavior OF testbench IS

    COMPONENT muxnto1_bus IS
        generic(
            n_addr      : INTEGER := 2;
            bus_width    : INTEGER := 8 );
        PORT (
            s            : IN    std_logic_vector( n_addr - 1 downto 0 );
            w            : IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
            f            : OUT   std_logic_vector( bus_width - 1 downto 0 );
        );
    END COMPONENT;

    -- Inputs
    SIGNAL s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
    SIGNAL w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => ( others =>
'0' ) );

    -- Outputs
    signal f, f_correct : std_logic_vector( bus_width - 1 downto 0 );
    signal trace : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );

BEGIN

    uut: muxnto1_bus

```

```
GENERIC MAP ( n_addr => n_addr, bus_width => bus_width )  
PORT MAP( s => s, w => w, f => f );
```

```
P0 : process
```

```
    variable input_sig : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );
```

```
BEGIN
```

```
    for input in 0 to 2**16 - 1 loop
```

```
        input_sig := std_logic_vector( to_unsigned( input, input_sig'length ) );
```

```
        trace <= input_sig;
```

```
        for i in 0 to bus_width-1 loop
```

```
            for j in 0 to 2**n_addr - 1 loop
```

```
                w( j, i ) <= input_sig( i * 2**n_addr + j );
```

```
            end loop;
```

```
        end loop;
```

```
        for k in 0 to 2**n_addr - 1 loop
```

```
            s <= std_logic_vector( to_unsigned( k, s'length ) );
```

```
            wait for 10 ns;
```

```
            assert f = f_correct report "-- MISTAKE OCCURED-- ";
```

```
        end loop;
```

```
    end loop;
```

```
    wait;
```

```
end process;
```

```
P1 : process ( w, s )
```

```
    variable f_cor_sig : std_logic_vector( bus_width - 1 downto 0 );
```

```
begin
```

```
    for l in 0 to bus_width-1 loop
```

```
        f_cor_sig( l ) := w( to_integer( unsigned( s ) ), l );
```

```
    end loop;
```

```
    f_correct <= f_cor_sig;
```

```
end process;
```

```
END;
```

```

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

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use work.reg_file_functions.all;

entity testbench is
    generic( n_addr : integer := 2;
             bus_width : integer := 8 );
end testbench;

architecture behavior of testbench is
    component muxnto1_bus is
        generic( n_addr : integer := 2;
                 bus_width : integer := 8 );
        port(
            s : in std_logic_vector( n_addr - 1 downto 0 );
            w : in muxnto1_bus_type( 2**n_addr - 1 downto 0, bus_width - 1 downto 0 );
            f : out std_logic_vector( bus_width - 1 downto 0 )
        );
    end component;
    signal s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
    signal w : muxnto1_bus_type( 2**n_addr - 1 downto 0, bus_width - 1 downto 0 ) := ( others => ( others =>
'0' ) );
    signal f : std_logic_vector( bus_width - 1 downto 0 );
begin
    uut: muxnto1_bus
        generic map( n_addr => n_addr, bus_width => bus_width )
        port map(
            s => s, w => w, f => f
        );
    tb : process
        variable data : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );
        begin
            for i in 0 to 2**16 - 1 loop

```



```

data := std_logic_vector( to_unsigned( i, data'length ) );
for j in 0 to bus_width - 1 loop
for k in 0 to 2**n_addr - 1 loop
w( k, j )    <= data( j * 2**n_addr + k );
end loop;
end loop;
for addr in 0 to 2**n_addr - 1 loop
s    <= std_logic_vector( to_unsigned( addr, s'length ) );
wait for 10 ns;
end loop;
end loop;
wait;
end process;
end;

```

```

-- *****
-- **** STUDENT: 64200238
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY testbench IS
    generic(
        n_addr      : INTEGER := 3;
        bus_width    : INTEGER := 3 );
END testbench;

ARCHITECTURE behavior OF testbench IS

    COMPONENT muxnto1_bus IS
        generic(
            n_addr      : INTEGER := 2;
            bus_width    : INTEGER := 8 );
        PORT (
            s            : IN    std_logic_vector( n_addr - 1 downto 0 );
            w            : IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
            f            : OUT   std_logic_vector( bus_width - 1 downto 0 )
        );
    END COMPONENT;

    SIGNAL
        s : std_logic_vector( n_addr - 1 DOWNT0 0 );
    SIGNAL
        w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
    SIGNAL
        f : std_logic_vector( bus_width - 1 DOWNT0 0 );

BEGIN

DUT: ENTITY work.muxnto1_bus
GENERIC MAP(
    n_addr => n_addr,
    bus_width => bus_width
)
PORT MAP(
    s => s,

```

```

w => w,      f => f
);

TEST_PROCESS: PROCESS
VARIABLE addr : INTEGER;
BEGIN

s    <= ( OTHERS => '0' );
w    <= ( OTHERS => ( OTHERS => '0' ) );

    -- Testiraj vse možne kombinacije naslova ( s )
FOR addr IN 0 TO 2**n_addr - 1 LOOP
    -- Nastavi naslov
s    <= std_logic_vector( to_unsigned( addr, n_addr ) );

    -- Nastavi podatkovne vrednosti za trenutni naslov
FOR i IN 0 TO bus_width - 1 LOOP
w( addr, i )<= '1';      -- Nastavi trenutni naslov na '1'
END LOOP;

WAIT FOR 10 ns;

FOR i IN 0 TO bus_width - 1 LOOP
ASSERT f( i ) = w( addr, i )
REPORT "Napaka: izhod ni pravilen za naslov " & integer'image( addr ) & " in bit " & integer'image( i )
SEVERITY ERROR;
END LOOP;

    -- Ponastavi trenutni naslov za naslednji test
FOR i IN 0 TO bus_width - 1 LOOP
w( addr, i )<= '0';      -- Ponastavi na '0'
END LOOP;

END LOOP;

    -- Konec simulacije
REPORT "Testiranje zaključeno." SEVERITY NOTE;
WAIT;
END PROCESS;
END behavior;

```

```

-- *****
-- **** STUDENT: 64200288
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY testbench IS
    generic(
        n_addr      : INTEGER := 2;
        bus_width    : INTEGER := 8 );
END testbench;

ARCHITECTURE behavior OF testbench IS

    COMPONENT muxnto1_bus IS
        generic(
            n_addr      : INTEGER := 2;
            bus_width    : INTEGER := 8 );
        PORT (
            s            : IN    std_logic_vector( n_addr - 1 downto 0 );
            w            : IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
            f            : OUT   std_logic_vector( bus_width - 1 downto 0 );
        );
    END COMPONENT;

    signal s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
    signal w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => ( others =>
'0' ) );
    signal trace : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );
    signal f, f_prog : std_logic_vector( bus_width - 1 downto 0 );

BEGIN

    Tb: muxnto1_bus
    GENERIC MAP ( n_addr => n_addr, bus_width => bus_width )
    PORT MAP(
        s => s,      w => w,      f => f

```

```
);
```

```
TBproces : PROCESS
```

```
variable input_slv : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );
```

```
BEGIN
```

```
    for input in 0 to 2**16 - 1 loop
```

```
        input_slv := std_logic_vector( to_unsigned( input, input_slv'length ) );
```

```
        trace <= input_slv;
```

```
        for idx in 0 to bus_width-1 loop
```

```
            for jdx in 0 to 2**n_addr - 1 loop
```

```
                w( jdx, idx )<= input_slv( idx * 2**n_addr + jdx );
```

```
            end loop;
```

```
        end loop;
```

```
        for s_int in 0 to 2**n_addr - 1 loop
```

```
            s      <= std_logic_vector( to_unsigned( s_int, s'length ) );
```

```
            wait for 2 ns;
```

```
            assert f = f_prog report "Napaca";
```

```
        end loop;
```

```
    end loop;
```

```
        wait;
```

```
END PROCESS tb;
```

```
mux_prog : process ( w, s )
```

```
variable f_temp : std_logic_vector( bus_width - 1 downto 0 );
```

```
begin
```

```
    for i in 0 to bus_width-1 loop
```

```
        f_temp( i ) := w( to_integer( unsigned( s ) ), i );
```

```
    end loop;
```

```
    f_prog <= f_temp;
```

```
end process;
```

```
END;
```

```

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

```

```

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

```

```

ENTITY testbench IS
  GENERIC(
    n_addr : INTEGER := 2;
    bus_width : INTEGER := 8
  );
END testbench;

```

```

ARCHITECTURE behavior OF testbench IS

```

```

  COMPONENT muxnto1_bus IS
    GENERIC(
      n_addr : INTEGER := 2;
      bus_width : INTEGER := 8
    );
    PORT (
      s : IN std_logic_vector( n_addr - 1 DOWNT0 0 );
      w : IN muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
      f : OUT std_logic_vector( bus_width - 1 DOWNT0 0 )
    );
  END COMPONENT;

```

```

  SIGNAL          s : std_logic_vector( n_addr - 1 DOWNT0 0 ) := ( others => '0' );
  SIGNAL          w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => ( others =>
'0' ) );
  SIGNAL          f : std_logic_vector( bus_width - 1 DOWNT0 0 );
  SIGNAL          f_expected : std_logic_vector( bus_width - 1 DOWNT0 0 );

```

```

BEGIN

```

```

 uut: muxnto1_bus
 GENERIC MAP ( n_addr => n_addr, bus_width => bus_width )
 PORT MAP (
  s => s,
  w => w,
  f => f
 );

 tb_process : PROCESS
 VARIABLE input_vector : std_logic_vector( 2**n_addr * bus_width - 1 DOWNTO 0 ) := ( others => '0' );
 BEGIN
  -- Test za vsako vhodno kombinacijo
  FOR input_index IN 0 TO 2**16 - 1 LOOP
   input_vector := std_logic_vector( to_unsigned( input_index, input_vector'length ) );

   FOR bus_idx IN 0 TO bus_width - 1 LOOP
    FOR addr_idx IN 0 TO 2**n_addr - 1 LOOP
     w( addr_idx, bus_idx ) <= input_vector( bus_idx * 2**n_addr + addr_idx );
    END LOOP;
   END LOOP;

   -- Gre skozi vse selectorje in preveri izhod
   FOR select_val IN 0 TO 2**n_addr - 1 LOOP
    s <= std_logic_vector( to_unsigned( select_val, s'length ) );
    WAIT FOR 1 ns;
    ASSERT f = f_expected REPORT "Output mismatch at selector value " & INTEGER'IMAGE( select_val );
   END LOOP;
  END LOOP;

  WAIT;
 END PROCESS tb_process;

 mux_prog : PROCESS ( w, s )
 VARIABLE temp_output : std_logic_vector( bus_width - 1 DOWNTO 0 );
 BEGIN
  FOR bit_idx IN 0 TO bus_width - 1 LOOP
   temp_output( bit_idx ) := w( to_integer( unsigned( s ) ), bit_idx );
  END LOOP;
  f_expected <= temp_output;
 END PROCESS mux_prog;

```

END;



```

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

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY testbench IS
generic(
    n_addr      :    INTEGER := 2;
    bus_width   :    INTEGER := 8 );
END testbench;

ARCHITECTURE behavior OF testbench IS

    -- Component Declaration
    COMPONENT muxnto1_bus is
generic(
    n_addr      :    INTEGER := 2;
    bus_width   :    INTEGER := 8 );
PORT(
    s :    IN    std_logic_vector( n_addr - 1 downto 0 );
        w      :    IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
        f      :    OUT   std_logic_vector( bus_width - 1 downto 0 )
);
END COMPONENT;

    signal
        signal      s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
        signal      w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => (
others => '0' ) );
        signal      f : std_logic_vector( bus_width - 1 downto 0 );

BEGIN

    -- Component Instantiation
    uut: muxnto1_bus
        GENERIC MAP ( n_addr => n_addr,
            bus_width => bus_width )

```

```

    PORT MAP(
        s => s,      w => w,      f => f
    );

    -- Test Bench Statements
tb : PROCESS
variable vh : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );
BEGIN
for i in 0 to 2**16 - 1 loop
vh := std_logic_vector( to_unsigned( i, vh'length ) );
for j in 0 to bus_width - 1 loop
for k in 0 to 2**n_addr - 1 loop
w( k,j )    <= vh( j * 2**n_addr + k );
end loop;
end loop;

for z in 0 to 2**n_addr - 1 loop
s    <= std_logic_vector( to_unsigned( z, s'length ) );
wait for 10ns;
end loop;
end loop;

wait;
END PROCESS tb;

END;

```

```

-- *****
-- **** STUDENT: 64210113
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY testbench IS
    generic(
        n_addr      : INTEGER := 2;
        bus_width    : INTEGER := 8 );
END testbench;

ARCHITECTURE behavior OF testbench IS

    COMPONENT muxnto1_bus IS
        generic(
            n_addr      : INTEGER := 2;
            bus_width    : INTEGER := 8 );
        PORT (
            s            : IN    std_logic_vector( n_addr - 1 downto 0 );
            w            : IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
            f            : OUT   std_logic_vector( bus_width - 1 downto 0 );
        );
    END COMPONENT;

    SIGNAL      s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
    SIGNAL      w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => ( others =>
'0' ) );

    signal      f : std_logic_vector( bus_width - 1 downto 0 );
    signal      f_sp : std_logic_vector( bus_width - 1 downto 0 );
    signal      Ozna : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );

BEGIN

    uut: muxnto1_bus
    GENERIC MAP ( n_addr => n_addr, bus_width => bus_width )

```

```

PORT MAP(
    s => s,      w => w,      f => f
);

tb : process
variable input_S : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );

begin
    for input in 0 to 2**16 - 1 loop

        input_S := std_logic_vector( to_unsigned( input, input_S'length ) );
        Oznz    <= input_S;

        for i in 0 to bus_width-1 loop
            for j in 0 to 2**n_addr - 1 loop
                w( j, i )    <= input_S( i * 2**n_addr + j );
            end loop;
        end loop;

        for s_int in 0 to 2**n_addr - 1 loop
            s    <= std_logic_vector( to_unsigned( s_int, s'length ) );
            wait for 1 ns;
        end loop;

    end loop;

    wait;
END PROCESS tb;

Pm_var : process ( w, s )
variable f_var : std_logic_vector( bus_width - 1 downto 0 );

begin
    for i in 0 to bus_width-1 loop
        f_var( i ) := w( to_integer( unsigned( s ) ), i );
    end loop;

    f_sp    <= f_var;
end process;

END;

```

```

-- *****
-- **** STUDENT: 64210290
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY testbench IS
END testbench;

ARCHITECTURE behavior OF testbench IS

    COMPONENT muxnto1_bus IS
        PORT(
            s : IN STD_LOGIC_VECTOR( 2 - 1 DOWNT0 0 );
            w : IN muxnto1_bus_type( 2**2 - 1 DOWNT0 0, 8 - 1 DOWNT0 0 );
            f : OUT STD_LOGIC_VECTOR( 8 - 1 DOWNT0 0 )
        );
    END COMPONENT;

    signal      s : STD_LOGIC_VECTOR( 2 - 1 DOWNT0 0 );
    signal      w : muxnto1_bus_type( 2**2 - 1 DOWNT0 0, 8 - 1 DOWNT0 0 );
    signal      f : STD_LOGIC_VECTOR( 8 - 1 DOWNT0 0 );

    constant    clock_period : time := 10 ns;
    signal      clock : std_logic;

BEGIN

    uut: muxnto1_bus
        port map(
            s => s,           w => w,           f => f
        );

    clock_process : process
    begin

```

```

        clock <= '0';
        wait for clock_period;
        clock <= '1';
        wait for clock_period;
    end process;

```

```

tb : PROCESS
BEGIN

```

```

    wait for 10*clock_period;

```

```

    w( 0,0 )    <= '1';
    w( 0,1 )    <= '1';
    w( 0,2 )    <= '1';
    w( 0,3 )    <= '1';
    w( 0,4 )    <= '0';
    w( 0,5 )    <= '0';
    w( 0,6 )    <= '0';
    w( 0,7 )    <= '0';
    w( 1,0 )    <= '0';
    w( 1,1 )    <= '0';
    w( 1,2 )    <= '0';
    w( 1,3 )    <= '0';
    w( 1,4 )    <= '1';
    w( 1,5 )    <= '1';
    w( 1,6 )    <= '1';
    w( 1,7 )    <= '1';
    w( 2,0 )    <= '1';
    w( 2,1 )    <= '1';
    w( 2,2 )    <= '0';
    w( 2,3 )    <= '0';
    w( 2,4 )    <= '1';
    w( 2,5 )    <= '1';
    w( 2,6 )    <= '0';
    w( 2,7 )    <= '0';
    w( 3,0 )    <= '0';
    w( 3,1 )    <= '0';
    w( 3,2 )    <= '1';
    w( 3,3 )    <= '1';
    w( 3,4 )    <= '0';

```

```
w( 3,5 )    <= '0';  
w( 3,6 )    <= '1';  
w( 3,7 )    <= '1';
```

```
wait for 10*clock_period;
```

```
s    <= "00";  
wait for clock_period;  
s    <= "01";  
wait for clock_period;  
s    <= "10";  
wait for clock_period;  
s    <= "11";  
wait for clock_period;
```

```
wait;  
END PROCESS tb;
```

```
END;
```

```

-- *****
-- **** STUDENT: 64210382
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY testbench IS
    generic(
        n_addr      : INTEGER := 2;
        bus_width    : INTEGER := 8 );
END testbench;

ARCHITECTURE behavior OF testbench IS

    COMPONENT muxnto1_bus IS
        generic(
            n_addr      : INTEGER := 2;
            bus_width    : INTEGER := 8 );
        PORT (
            s            : IN    std_logic_vector( n_addr - 1 downto 0 );
            w            : IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
            f            : OUT   std_logic_vector( bus_width - 1 downto 0 )
        );
    END COMPONENT;

    SIGNAL      s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
    SIGNAL      w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => ( others =>
'0' ) );

    signal      f, f_prog : std_logic_vector( bus_width - 1 downto 0 );

    signal      trace : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );

BEGIN

    uut: muxnto1_bus
    GENERIC MAP ( n_addr => n_addr, bus_width => bus_width )

```



```
PORT MAP(  
    s => s,      w => w,      f => f  
);
```

```
tb : PROCESS
```

```
variable input_slv : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );
```

```
BEGIN
```

```
    for input in 0 to 2**16 - 1 loop  
        input_slv := std_logic_vector( to_unsigned( input, input_slv'length ) );  
        trace <= input_slv;  
        for idx in 0 to bus_width-1 loop  
            for jdx in 0 to 2**n_addr - 1 loop  
                w( jdx, idx ) <= input_slv( idx * 2**n_addr + jdx );  
            end loop;  
        end loop;  
        for s_int in 0 to 2**n_addr - 1 loop  
            s      <= std_logic_vector( to_unsigned( s_int, s'length ) );  
            wait for 1 ns;  
            assert f = f_prog report "FAILURE!";  
        end loop;  
    end loop;
```

```
    wait;
```

```
END PROCESS tb;
```

```
mux_prog : process ( w, s )
```

```
variable f_temp : std_logic_vector( bus_width - 1 downto 0 );
```

```
begin
```

```
    for i in 0 to bus_width-1 loop  
        f_temp( i ) := w( to_integer( unsigned( s ) ), i );  
    end loop;  
    f_prog <= f_temp;
```

```
end process;
```

```
END;
```

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

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use ieee.numeric_std.all;
use work.reg_file_functions.all;
```

```
entity muxnto1_bus_tb is
    generic(
        n_addr : integer := 2;
        bus_width : integer := 8
    );
end muxnto1_bus_tb;
```

```
architecture dn6 of muxnto1_bus_tb is
```

```
    component muxnto1_bus
        generic(
            n_addr : integer := 2;
            bus_width : integer := 8
        );
        port(
            s : in std_logic_vector( n_addr - 1 downto 0 );
            w : in muxnto1_bus_type( 2**n_addr - 1 downto 0, bus_width - 1 downto 0 );
            f : out std_logic_vector( bus_width - 1 downto 0 )
        );
    end component;
```

```
    signal s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
    signal w : muxnto1_bus_type( 2**n_addr - 1 downto 0, bus_width - 1 downto 0 ) := ( others => ( others =>
'0' ) );
```

```
    signal f : std_logic_vector( bus_width - 1 downto 0 );
```

```
    signal clk : std_logic;
    constant clk_period : time := 100 ps;
```

```

signal      izhod : std_logic_vector( bus_width - 1 downto 0 );

begin
  uut: muxnto1_bus
    generic map(
      n_addr => n_addr,          bus_width => bus_width
    )
    port map(
      s => s,                    w => w,          f => f
    );

  clock_process: process
  begin
    clk    <= '0';
    wait for clk_period/2;
    clk    <= '1';
    wait for clk_period/2;
  end process;

  stimulus_process: process
    variable helper : std_logic_vector( bus_width*2**n_addr - 1 downto 0 );
  begin
    z1: for k in 0 to 2**( bus_width*n_addr ) - 1 loop
      helper := std_logic_vector( to_unsigned( k, helper'length ) );
      for j in 0 to bus_width - 1 loop
        for i in 0 to 2**n_addr - 1 loop
          w( i,j )    <= helper( j*2**n_addr + i );
        end loop;
      end loop;
      for i in 0 to 2**n_addr - 1 loop
        s    <= std_logic_vector( to_unsigned( i, s'length ) );
        wait for clk_period;
        assert ( f = izhod )
          report "Test failed for s = " & integer'image( to_integer( unsigned( s ) ) ) & ". Expected " &
integer'image( to_integer( unsigned( izhod ) ) ) &
          " but got " & integer'image( to_integer( unsigned( f ) ) ) & ". Exiting..." severity error;
        exit z1 when ( f /= izhod );
      end loop;
    end loop;
  end process;

```

```

        wait;
    end process;

    posodobitev_process: process ( w, s )          -- podobno kot pri domaco nalogo 2, proces nujen zaradi
zakasnitve clk periode
        variable rezultat_temp : std_logic_vector( bus_width - 1 downto 0 ) := ( others => '0' );
    begin
        for i in 0 to bus_width - 1 loop
            rezultat_temp( i ) := w( to_integer( unsigned( s ) ), i );
        end loop;
        izhod <= rezultat_temp;
    end process;
end;

```

```

-- *****
-- **** STUDENT: 64210386
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
-- TestBench Template

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY testbench IS
    generic(
        n_addr      : INTEGER := 2;
        bus_width    : INTEGER := 8 );
END testbench;

ARCHITECTURE behavior OF testbench IS

    -- Component Declaration
    COMPONENT muxnto1_bus is
        generic(
            n_addr      : INTEGER := 2;
            bus_width    : INTEGER := 8 );
    PORT(
        s      : IN    std_logic_vector( n_addr - 1 downto 0 );
        w      : IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
        f      : OUT   std_logic_vector( bus_width - 1 downto 0 );
    );
END COMPONENT;

    SIGNAL
        s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
        w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => ( others =>
'0' ) );

        signal
            f, f_sig : std_logic_vector( bus_width - 1 downto 0 );

        signal
            tr : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );

BEGIN

```

```

    -- Component Instantiation
    uut: muxnto1_bus
        GENERIC MAP ( n_addr => n_addr, bus_width => bus_width )
        PORT MAP(
            s => s,          w => w,          f => f
        );

    -- Test Bench Statements
    tb : PROCESS
        variable x : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );
    BEGIN

    for i in 0 to 2**16 - 1 loop
        x := std_logic_vector( to_unsigned( i, x'length ) );
        tr    <= x;
        for i_b in 0 to bus_width-1 loop
            for j_b in 0 to 2**n_addr - 1 loop
                w( j_b, i_b ) <= x( i_b * 2**n_addr + j_b );
            end loop;
        end loop;
        for s_b in 0 to 2**n_addr - 1 loop
            s    <= std_logic_vector( to_unsigned( s_b, s'length ) );
            wait for 1 ns;
            assert f = f_sig report "FAIL!";
        end loop;
    end loop;

    wait; -- will wait forever
END PROCESS tb;

mux : process ( w, s )
    variable f_b : std_logic_vector( bus_width - 1 downto 0 );
    begin
        for i in 0 to bus_width-1 loop
            f_b( i ) := w( to_integer( unsigned( s ) ), i );
        end loop;
        f_sig <= f_b;
    end process;

```

END;

```

-- *****
-- **** STUDENT: 64210457
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY testbench IS
    generic(
        n_addr      : INTEGER := 2;
        bus_width    : INTEGER := 8 );
END testbench;

ARCHITECTURE behavior OF testbench IS

    COMPONENT muxnto1_bus IS
        generic(
            n_addr      : INTEGER := 2;
            bus_width    : INTEGER := 8 );
        PORT (
            s            : IN    std_logic_vector( n_addr - 1 downto 0 );
            w            : IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
            f            : OUT   std_logic_vector( bus_width - 1 downto 0 )
        );
    END COMPONENT;

    SIGNAL      s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
    SIGNAL      w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => ( others =>
'0' ) );

    signal      f, f_prog : std_logic_vector( bus_width - 1 downto 0 );

    signal      trace : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );

BEGIN

    uut: muxnto1_bus
    GENERIC MAP ( n_addr => n_addr, bus_width => bus_width )

```



```
PORT MAP(  
    s => s,      w => w,      f => f  
);
```

```
tb : PROCESS
```

```
variable input_slv : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );
```

```
BEGIN
```

```
    for input in 0 to 2**16 - 1 loop  
        input_slv := std_logic_vector( to_unsigned( input, input_slv'length ) );  
        trace <= input_slv;  
        for idx in 0 to bus_width-1 loop  
            for jdx in 0 to 2**n_addr - 1 loop  
                w( jdx, idx ) <= input_slv( idx * 2**n_addr + jdx );  
            end loop;  
        end loop;  
        for s_int in 0 to 2**n_addr - 1 loop  
            s      <= std_logic_vector( to_unsigned( s_int, s'length ) );  
            wait for 1 ns;  
            assert f = f_prog report "FAILURE!";  
        end loop;  
    end loop;
```

```
    wait;
```

```
END PROCESS tb;
```

```
mux_prog : process ( w, s )
```

```
variable f_temp : std_logic_vector( bus_width - 1 downto 0 );
```

```
begin
```

```
    for i in 0 to bus_width-1 loop  
        f_temp( i ) := w( to_integer( unsigned( s ) ), i );  
    end loop;  
    f_prog <= f_temp;
```

```
end process;
```

```
END;
```

```

-- *****
-- **** STUDENT: 64240430
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE work.reg_file_functions.all;

ENTITY muxnto1_bus_tb IS
    generic(
        n_addr      : INTEGER := 2;
        bus_width    : INTEGER := 8 );
END muxnto1_bus_tb;

ARCHITECTURE behavior OF muxnto1_bus_tb IS

    COMPONENT muxnto1_bus IS
        generic(
            n_addr      : INTEGER := 2;
            bus_width    : INTEGER := 8 );
        PORT (
            s      : IN    std_logic_vector( n_addr - 1 downto 0 );
            w      : IN    muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 );
            f      : OUT   std_logic_vector( bus_width - 1 downto 0 )
        );
    END COMPONENT;

    SIGNAL      s : std_logic_vector( n_addr - 1 downto 0 ) := ( others => '0' );
    SIGNAL      w : muxnto1_bus_type( 2**n_addr - 1 DOWNT0 0, bus_width - 1 DOWNT0 0 ) := ( others => ( others =>
'0' ) );

    signal      f : std_logic_vector( bus_width - 1 downto 0 );

    signal      trace : std_logic_vector( 2**n_addr * bus_width - 1 downto 0 ) := ( others => '0' );

    signal      f_out : std_logic_vector( bus_width - 1 downto 0 );

    signal
    constant    clock : std_logic;
    constant    clock_period : time := 100 ns;

```

```
BEGIN
```

```
 uut: muxnto1_bus
    GENERIC MAP ( n_addr => n_addr,          bus_width => bus_width )
    PORT MAP(
        s => s,          w => w,          f => f
    );
```

```
clock_process: process
begin
    clock <= '0';
    wait for clock_period/2;
    clock <= '1';
    wait for clock_period/2;
end process;
```

```
-- Test Bench Statements
```

```
tb : PROCESS-- proces
    variable vhd_temp : std_logic_vector( bus_width * ( 2**n_addr ) - 1 downto 0 );
```

```
BEGIN
```

```
case1:
```

```
for i in 0 to 2**( n_addr * bus_width ) - 1 loop
    vhd_temp := std_logic_vector( to_unsigned( i, vhd_temp'length ) );
    for j in 0 to bus_width - 1 loop
        for k in 0 to 2**n_addr - 1 loop
            w( k, j )    <= vhd_temp( j * ( 2**n_addr ) + k );
        end loop;
    end loop;
```

```
    for s_int in 0 to 2**n_addr - 1 loop
        s    <= std_logic_vector( to_unsigned( s_int, s'length ) );
        wait for clock_period;
```

```
    assert ( f = f_out )
    report "Napaka!!!";
```

```
    exit case1 when ( f /= f_out );
end loop;
```

```

        end loop;
        wait;
END PROCESS tb;

second_process : process( w, s )
    variable f_temp : std_logic_vector( bus_width - 1 downto 0 );
begin

    for i in 0 to bus_width - 1 loop
        f_temp( i ) := w( to_integer( unsigned( s ) ), i );
    end loop;

    f_out <= f_temp;

end process;

END;
```