

-- **** STUDENT: 64200100.....	2
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	2
-- **** STUDENT: 64210382.....	8
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	8
-- **** STUDENT: 64210384.....	14
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	14

```

-- *****
-- **** STUDENT: 64200100
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;
USE std.textio.all;
USE work.reg_file_functions.all;

ENTITY reg_file_tb IS
    GENERIC (
        nr_regs          : natural := 4;
        reg_width        : natural := 8
    );
END reg_file_tb;

ARCHITECTURE ideal OF reg_file_tb IS

    -- ATTRIBUTE ENUM_ENCODING : STRING;
    -- ATTRIBUTE ENUM_ENCODING OF reg_addr_type : TYPE IS "00 01 10 11";
    -- type reg_addr_type is ( R0, R1, R2, R3 );

    SIGNAL      nRST          : std_logic := '0'; -- reset input      ( active '0' )
    SIGNAL      LE            : std_logic := '0'; -- load enable input      ( active '1' )

    SIGNAL      dest_select   : std_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 ); -- register number
    destination select input
    SIGNAL      A_select      : std_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 ); -- A bus destination
    select input
    SIGNAL      B_select      : std_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 ); -- B bus destination
    select input
    SIGNAL      D              : std_logic_vector( reg_width - 1 downto 0 ); -- data input bus input
    SIGNAL      A              : std_logic_vector( reg_width - 1 downto 0 ); -- A, B bus output
    SIGNAL      B              : std_logic_vector( reg_width - 1 downto 0 ); -- A, B bus output
    SIGNAL      clk            : STD_LOGIC := '0';
    constant   PERIOD : time := 400 ns;

```

```

constant    DUTY_CYCLE : real := 0.5;
constant    OFFSET : time := 100 ns;
constant    R0      : integer := 0;
constant    R1      : integer := 1;
constant    R2      : integer := 2;
constant    R3      : integer := 3;

```

```

constant    DATA0 : integer := 11;
constant    DATA1 : integer := 22;
constant    DATA2 : integer := 33;
constant    DATA3 : integer := 44;

```

```

COMPONENT reg_file is
    generic( nr_regs          : natural := 4;
             reg_width       : natural := 8 );
    PORT ( clk,  -- clock input
           LE    : IN std_logic;    -- Load enable input      ( active '1' )
           nRST  : in std_logic;    -- reset input          ( active '0' )
           dest_select, -- register number destination select input
           A_select,  -- A, B bus destination select input
           B_select : instd_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 );
           D        : in std_logic_vector( reg_width - 1 downto 0 );    -- data input bus input
           A, B     : out std_logic_vector( reg_width - 1 downto 0 )    -- A, B bus output
    );
end COMPONENT;

```

```

BEGIN

```

```

    UUT: reg_file
    GENERIC MAP( nr_regs => nr_regs, reg_width => reg_width )
    PORT MAP( clk => clk, LE => LE, nRST => nRST, dest_select =>
dest_select, A_select => A_select, B_select => B_select, D => D, A => A,
B => B
    );

```

```

PROCESS    -- clock process for clk
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
BEGIN
    D      <= std_logic_vector( to_unsigned( DATA0, D'length ) );
    dest_select <= std_logic_vector( to_unsigned( R0, dest_select'length ) );    -- select R0 for D
input destination
    A_select  <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
    B_select  <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

    WAIT FOR ( PERIOD );
    nRST    <= '1';    -- reset off

    WAIT FOR ( PERIOD );
    LE      <= '1';    -- enable loading of data for one clock cycle
    WAIT FOR ( PERIOD );
    LE      <= '0';    -- disable loading of data

    WAIT FOR ( PERIOD );
    D      <= std_logic_vector( to_unsigned( DATA0, D'length ) );
    dest_select <= std_logic_vector( to_unsigned( R0, dest_select'length ) );    -- select R0 for D
input destination
    A_select  <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
    B_select  <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

    WAIT FOR ( PERIOD );
    LE      <= '1';    -- enable loading of data for one clock cycle
    WAIT FOR ( PERIOD );
    LE      <= '0';    -- disable loading of data

    WAIT FOR ( PERIOD );
    D      <= std_logic_vector( to_unsigned( DATA1, D'length ) );

```

```

input destination    dest_select  <= std_logic_vector( to_unsigned( R1, dest_select'length ) );    -- select R0 for D
output              A_select    <= std_logic_vector( to_unsigned( R1, A_select'length ) );    -- select R0 for A bus
output              B_select    <= std_logic_vector( to_unsigned( R1, B_select'length ) );    -- select R0 for B bus

WAIT FOR ( PERIOD );
LE    <= '1';    -- enable loading of data
WAIT FOR ( PERIOD );
LE    <= '0';    -- disable loading of data

WAIT FOR ( PERIOD );
D      <= std_logic_vector( to_unsigned( DATA2, D'length ) );
input destination    dest_select  <= std_logic_vector( to_unsigned( R2, dest_select'length ) );    -- select R0 for D
output              A_select    <= std_logic_vector( to_unsigned( R2, A_select'length ) );    -- select R0 for A bus
output              B_select    <= std_logic_vector( to_unsigned( R2, B_select'length ) );    -- select R0 for B bus

WAIT FOR ( PERIOD );
LE    <= '1';    -- enable loading of data
WAIT FOR ( PERIOD );
LE    <= '0';    -- disable loading of data

WAIT FOR ( PERIOD );
D      <= std_logic_vector( to_unsigned( DATA3, D'length ) );
input destination    dest_select  <= std_logic_vector( to_unsigned( R3, dest_select'length ) );    -- select R0 for D
output              A_select    <= std_logic_vector( to_unsigned( R3, A_select'length ) );    -- select R0 for A bus
output              B_select    <= std_logic_vector( to_unsigned( R3, B_select'length ) );    -- select R0 for B bus

WAIT FOR ( PERIOD );
LE    <= '1';    -- enable loading of data
WAIT FOR ( PERIOD );
LE    <= '0';    -- disable loading of data
WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );
output              A_select    <= std_logic_vector( to_unsigned( R2, A_select'length ) );    -- select R0 for A bus

```

```

output      B_select      <= std_logic_vector( to_unsigned( R2, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );
output      A_select      <= std_logic_vector( to_unsigned( R1, A_select'length ) ); -- select R0 for A bus

output      B_select      <= std_logic_vector( to_unsigned( R1, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );
output      A_select      <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus

output      B_select      <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );
output      A_select      <= std_logic_vector( to_unsigned( R3, A_select'length ) ); -- select R0 for A bus

output      B_select      <= std_logic_vector( to_unsigned( R3, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );
output      A_select      <= std_logic_vector( to_unsigned( R2, A_select'length ) ); -- select R0 for A bus

output      B_select      <= std_logic_vector( to_unsigned( R2, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );
output      A_select      <= std_logic_vector( to_unsigned( R1, A_select'length ) ); -- select R0 for A bus

output      B_select      <= std_logic_vector( to_unsigned( R1, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );
output      A_select      <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus

output      B_select      <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );

```

```
END PROCESS;
```

```
END ideal;
```

```

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

ENTITY reg_file_tb IS
    GENERIC (
        nr_regs          : natural := 4;
        reg_width        : natural := 8
    );
END reg_file_tb;

ARCHITECTURE ideal OF reg_file_tb IS

    -- ATTRIBUTE ENUM_ENCODING : STRING;
    -- ATTRIBUTE ENUM_ENCODING OF reg_addr_type : TYPE IS "00 01 10 11";
    -- type reg_addr_type is ( R0, R1, R2, R3 );

    SIGNAL      nRST          : std_logic := '0'; -- reset input      ( active '0' )
    SIGNAL      LE            : std_logic := '0'; -- load enable input    ( active '1' )

    SIGNAL      dest_select   : std_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 ); -- register number
    destination select input
    SIGNAL      A_select      : std_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 ); -- A bus destination
    select input
    SIGNAL      B_select      : std_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 ); -- B bus destination
    select input
    SIGNAL      D             : std_logic_vector( reg_width - 1 downto 0 ); -- data input bus input
    SIGNAL      A             : std_logic_vector( reg_width - 1 downto 0 ); -- A, B bus output
    SIGNAL      B             : std_logic_vector( reg_width - 1 downto 0 ); -- A, B bus output
    SIGNAL      clk           : STD_LOGIC := '0';
    constant   PERIOD : time := 400 ns;
    constant   DUTY_CYCLE : real := 0.5;

```



```

constant    OFFSET : time := 100 ns;
constant    R0      : integer := 0;
constant    R1      : integer := 1;
constant    R2      : integer := 2;
constant    R3      : integer := 3;

constant    DATA0 : integer := 11;
constant    DATA1 : integer := 22;
constant    DATA2 : integer := 33;
constant    DATA3 : integer := 44;

COMPONENT reg_file is
    generic( nr_regs          : natural := 4;
             reg_width       : natural := 8 );
    PORT ( clk, -- clock input
           LE : IN std_logic; -- Load enable input ( active '1' )
           nRST : in std_logic; -- reset input ( active '0' )
           dest_select, -- register number destination select input
           A_select, -- A, B bus destination select input
           B_select : instd_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 );
           D : in std_logic_vector( reg_width - 1 downto 0 ); -- data input bus input
           A, B : out std_logic_vector( reg_width - 1 downto 0 ) -- A, B bus output
    );
end COMPONENT;

BEGIN

    UUT: reg_file
    GENERIC MAP( nr_regs => nr_regs, reg_width => reg_width )
    PORT MAP( clk => clk, LE => LE, nRST => nRST, dest_select =>
dest_select, A_select => A_select, B_select => B_select, D => D, A => A,
B => B
    );

    PROCESS -- clock process for clk
    BEGIN
        WAIT for OFFSET;
        CLOCK_LOOP : LOOP
            clk <= '0';
            WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
        END LOOP;
    END PROCESS;

```

```

        clk    <= '1';
        WAIT FOR ( PERIOD * DUTY_CYCLE );
    END LOOP CLOCK_LOOP;
END PROCESS;

PROCESS
BEGIN
    D    <= std_logic_vector( to_unsigned( DATA0, D'length ) );
    dest_select <= std_logic_vector( to_unsigned( R0, dest_select'length ) );    -- select R0 for D
input destination
    A_select    <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
    B_select    <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

    WAIT FOR ( PERIOD );
    nRST    <= '1';    -- reset off

    WAIT FOR ( PERIOD );
    LE    <= '1';    -- enable loading of data for one clock cycle
    WAIT FOR ( PERIOD );
    LE    <= '0';    -- disable loading of data

    WAIT FOR ( PERIOD );
    D    <= std_logic_vector( to_unsigned( DATA0, D'length ) );
    dest_select <= std_logic_vector( to_unsigned( R0, dest_select'length ) );    -- select R0 for D
input destination
    A_select    <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
    B_select    <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

    WAIT FOR ( PERIOD );
    LE    <= '1';    -- enable loading of data for one clock cycle
    WAIT FOR ( PERIOD );
    LE    <= '0';    -- disable loading of data

    WAIT FOR ( PERIOD );
    D    <= std_logic_vector( to_unsigned( DATA1, D'length ) );
    dest_select <= std_logic_vector( to_unsigned( R1, dest_select'length ) );    -- select R0 for D
input destination

```

```

output
output
A_select    <= std_logic_vector( to_unsigned( R1, A_select'length ) ); -- select R0 for A bus
B_select    <= std_logic_vector( to_unsigned( R1, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
LE    <= '1';    -- enable loading of data
WAIT FOR ( PERIOD );
LE    <= '0';    -- disable loading of data

WAIT FOR ( PERIOD );
D      <= std_logic_vector( to_unsigned( DATA2, D'length ) );
dest_select <= std_logic_vector( to_unsigned( R2, dest_select'length ) );    -- select R0 for D
input destination
output
output
A_select    <= std_logic_vector( to_unsigned( R2, A_select'length ) ); -- select R0 for A bus
B_select    <= std_logic_vector( to_unsigned( R2, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
LE    <= '1';    -- enable loading of data
WAIT FOR ( PERIOD );
LE    <= '0';    -- disable loading of data

WAIT FOR ( PERIOD );
D      <= std_logic_vector( to_unsigned( DATA3, D'length ) );
dest_select <= std_logic_vector( to_unsigned( R3, dest_select'length ) );    -- select R0 for D
input destination
output
output
A_select    <= std_logic_vector( to_unsigned( R3, A_select'length ) ); -- select R0 for A bus
B_select    <= std_logic_vector( to_unsigned( R3, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
LE    <= '1';    -- enable loading of data
WAIT FOR ( PERIOD );
LE    <= '0';    -- disable loading of data
WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );
A_select    <= std_logic_vector( to_unsigned( R2, A_select'length ) ); -- select R0 for A bus
output
output
B_select    <= std_logic_vector( to_unsigned( R2, B_select'length ) ); -- select R0 for B bus

```

```

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R1, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R1, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R3, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R3, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R2, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R2, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R1, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R1, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
END PROCESS;

```

```
END ideal;
```

```

-- *****
-- **** STUDENT: 64210384
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;
USE std.textio.all;
USE work.reg_file_functions.all;

ENTITY reg_file_tb IS
    GENERIC (
        nr_regs          : natural := 4;
        reg_width        : natural := 8
    );
END reg_file_tb;

ARCHITECTURE ideal OF reg_file_tb IS

    -- ATTRIBUTE ENUM_ENCODING : STRING;
    -- ATTRIBUTE ENUM_ENCODING OF reg_addr_type : TYPE IS "00 01 10 11";
    -- type reg_addr_type is ( R0, R1, R2, R3 );

    SIGNAL      nRST          : std_logic := '0'; -- reset input      ( active '0' )
    SIGNAL      LE            : std_logic := '0'; -- load enable input    ( active '1' )

    SIGNAL      dest_select   : std_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 ); -- register number
    destination select input
    SIGNAL      A_select      : std_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 ); -- A bus destination
    select input
    SIGNAL      B_select      : std_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 ); -- B bus destination
    select input
    SIGNAL      D              : std_logic_vector( reg_width - 1 downto 0 ); -- data input bus input
    SIGNAL      A              : std_logic_vector( reg_width - 1 downto 0 ); -- A, B bus output
    SIGNAL      B              : std_logic_vector( reg_width - 1 downto 0 ); -- A, B bus output
    SIGNAL      clk            : STD_LOGIC := '0';
    constant   PERIOD : time := 400 ns;
    constant   DUTY_CYCLE : real := 0.5;

```

```

constant    OFFSET : time := 100 ns;
constant    R0      : integer := 0;
constant    R1      : integer := 1;
constant    R2      : integer := 2;
constant    R3      : integer := 3;

```

```

constant    DATA0 : integer := 11;
constant    DATA1 : integer := 22;
constant    DATA2 : integer := 33;
constant    DATA3 : integer := 44;

```

```

COMPONENT reg_file is
    generic( nr_regs          : natural := 4;
             reg_width       : natural := 8 );
    PORT ( clk, -- clock input
           LE : IN std_logic; -- Load enable input ( active '1' )
           nRST : in std_logic; -- reset input ( active '0' )
           dest_select, -- register number destination select input
           A_select, -- A, B bus destination select input
           B_select : instd_logic_vector( sizeof( nr_regs - 1 ) - 1 downto 0 );
           D : in std_logic_vector( reg_width - 1 downto 0 ); -- data input bus input
           A, B : out std_logic_vector( reg_width - 1 downto 0 ) -- A, B bus output
    );
end COMPONENT;

```

```

BEGIN

```

```

    UUT: reg_file
    GENERIC MAP( nr_regs => nr_regs, reg_width => reg_width )
    PORT MAP( clk => clk, LE => LE, nRST => nRST, dest_select =>
dest_select, A_select => A_select, B_select => B_select, D => D, A => A,
B => B
    );

```

```

PROCESS -- clock process for clk
BEGIN
    WAIT for OFFSET;
    CLOCK_LOOP : LOOP
        clk <= '0';
        WAIT FOR ( PERIOD - ( PERIOD * DUTY_CYCLE ) );
    END LOOP;

```

```

        clk    <= '1';
        WAIT FOR ( PERIOD * DUTY_CYCLE );
    END LOOP CLOCK_LOOP;
END PROCESS;

PROCESS
BEGIN
    D    <= std_logic_vector( to_unsigned( DATA0, D'length ) );
    dest_select <= std_logic_vector( to_unsigned( R0, dest_select'length ) );    -- select R0 for D
input destination
    A_select    <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
    B_select    <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

    WAIT FOR ( PERIOD );
    nRST    <= '1';    -- reset off

    WAIT FOR ( PERIOD );
    LE    <= '1';    -- enable loading of data for one clock cycle
    WAIT FOR ( PERIOD );
    LE    <= '0';    -- disable loading of data

    WAIT FOR ( PERIOD );
    D    <= std_logic_vector( to_unsigned( DATA0, D'length ) );
    dest_select <= std_logic_vector( to_unsigned( R0, dest_select'length ) );    -- select R0 for D
input destination
    A_select    <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
    B_select    <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

    WAIT FOR ( PERIOD );
    LE    <= '1';    -- enable loading of data for one clock cycle
    WAIT FOR ( PERIOD );
    LE    <= '0';    -- disable loading of data

    WAIT FOR ( PERIOD );
    D    <= std_logic_vector( to_unsigned( DATA1, D'length ) );
    dest_select <= std_logic_vector( to_unsigned( R1, dest_select'length ) );    -- select R0 for D
input destination

```



```

output
output
A_select    <= std_logic_vector( to_unsigned( R1, A_select'length ) ); -- select R0 for A bus
B_select    <= std_logic_vector( to_unsigned( R1, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
LE    <= '1';    -- enable loading of data
WAIT FOR ( PERIOD );
LE    <= '0';    -- disable loading of data

WAIT FOR ( PERIOD );
D      <= std_logic_vector( to_unsigned( DATA2, D'length ) );
dest_select <= std_logic_vector( to_unsigned( R2, dest_select'length ) );    -- select R0 for D
input destination
output
output
A_select    <= std_logic_vector( to_unsigned( R2, A_select'length ) ); -- select R0 for A bus
B_select    <= std_logic_vector( to_unsigned( R2, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
LE    <= '1';    -- enable loading of data
WAIT FOR ( PERIOD );
LE    <= '0';    -- disable loading of data

WAIT FOR ( PERIOD );
D      <= std_logic_vector( to_unsigned( DATA3, D'length ) );
dest_select <= std_logic_vector( to_unsigned( R3, dest_select'length ) );    -- select R0 for D
input destination
output
output
A_select    <= std_logic_vector( to_unsigned( R3, A_select'length ) ); -- select R0 for A bus
B_select    <= std_logic_vector( to_unsigned( R3, B_select'length ) ); -- select R0 for B bus

WAIT FOR ( PERIOD );
LE    <= '1';    -- enable loading of data
WAIT FOR ( PERIOD );
LE    <= '0';    -- disable loading of data
WAIT FOR ( PERIOD );
WAIT FOR ( PERIOD );
A_select    <= std_logic_vector( to_unsigned( R2, A_select'length ) ); -- select R0 for A bus
output
output
B_select    <= std_logic_vector( to_unsigned( R2, B_select'length ) ); -- select R0 for B bus

```

```

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R1, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R1, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R3, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R3, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R2, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R2, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R1, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R1, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );
        A_select    <= std_logic_vector( to_unsigned( R0, A_select'length ) ); -- select R0 for A bus
output
        B_select    <= std_logic_vector( to_unsigned( R0, B_select'length ) ); -- select R0 for B bus
output

        WAIT FOR ( PERIOD );
        WAIT FOR ( PERIOD );

END PROCESS;

```

```
END ideal;
```

