

-- **** STUDENT: 64000225.....	2
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	2
-- **** STUDENT: 64200100.....	11
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	11
-- **** STUDENT: 64200112.....	20
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	20
-- **** STUDENT: 64200238.....	29
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	29
-- **** STUDENT: 64200288.....	38
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	38
-- **** STUDENT: 64200296.....	47
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	47
-- **** STUDENT: 64200385.....	56
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	56
-- **** STUDENT: 64210113.....	65
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	65
-- **** STUDENT: 64210290.....	74
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	74
-- **** STUDENT: 64210382.....	83
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	83
-- **** STUDENT: 64210384.....	92
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	92
-- **** STUDENT: 64210386.....	101
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	101
-- **** STUDENT: 64210445.....	110
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	110
-- **** STUDENT: 64210455.....	119
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	119
-- **** STUDENT: 64210457.....	128
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	128
-- **** STUDENT: 64240430.....	137
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	137
-- **** PREDLOGA VAJE.....	146
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	146

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
        generic map( ctr_width => ctr_width      )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
PROCEDURE Log_variables(
    clk : std_logic;
    nRST : std_logic;
    N : std_logic;
    C : std_logic;
    V : std_logic;
    Z : std_logic;
    PL : std_logic;
    JB : std_logic;
    BC : std_logic;
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );
    COMMENT : string
    ) IS
VARIABLE RES_LINE : LINE;
BEGIN
    write( RES_LINE, clk, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, nRST, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, N, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, C, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, V, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, Z, right, 1 );
    write( RES_LINE, HT );

```

```

write( RES_LINE, PL, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address  <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST  <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';

```

```

JB      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL      <= '0';
JB      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
N       <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N       <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```


[illegible]

```
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

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

```
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

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

```
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

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

```
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width )
PORT MAP (
    clk => clk,
    nRST => nRST,
    N => N,
    C => C,
    V => V,
    Z => Z,
    PL => PL,
    JB => JB,
    BC => BC,
    JB_address => JB_address,
    PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
    BEGIN  
        write( RES_LINE, clk, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, nRST, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, N, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, C, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, V, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, Z, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address    <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST    <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```



```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

-- *****
-- **** STUDENT: 64200112
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
        generic map( ctr_width => ctr_width      )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
    BEGIN  
        write( RES_LINE, clk, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, nRST, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, N, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, C, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, V, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, Z, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address    <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST    <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```



```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

-- *****
-- **** STUDENT: 64200238
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
        generic map( ctr_width => ctr_width
        )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
BEGIN  
    write( RES_LINE, clk, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, nRST, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, N, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, C, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, V, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, Z, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

    write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
    writeline( RESULTS, HDR_line );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

    JB_address    <= ( 0=>'0', others=>'1' );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

    nRST    <= '1';
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```



```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
JB     <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

-- *****
-- **** STUDENT: 64200288
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width      )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
    BEGIN  
        write( RES_LINE, clk, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, nRST, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, N, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, C, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, V, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, Z, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, PL, right, 1 );
```



```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address    <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST    <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
JB     <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
        generic map( ctr_width => ctr_width
        )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
    BEGIN  
        write( RES_LINE, clk, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, nRST, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, N, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, C, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, V, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, Z, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

    write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
    writeline( RESULTS, HDR_line );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

    JB_address    <= ( 0=>'0', others=>'1' );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

    nRST    <= '1';
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```



```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
      generic map( ctr_width => ctr_width      )
PORT MAP (
  clk => clk,
  nRST => nRST,
  N => N,
  C => C,
  V => V,
  Z => Z,
  PL => PL,
  JB => JB,
  BC => BC,
  JB_address => JB_address,
  PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
BEGIN  
    write( RES_LINE, clk, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, nRST, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, N, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, C, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, V, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, Z, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address    <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST    <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```



```

-- *****
-- **** STUDENT: 64210113
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width )
PORT MAP (
    clk => clk,
    nRST => nRST,
    N => N,
    C => C,
    V => V,
    Z => Z,
    PL => PL,
    JB => JB,
    BC => BC,
    JB_address => JB_address,
    PC => PC
);

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
```

```
PROCEDURE Log_variables(
    clk : std_logic;
    nRST : std_logic;
    N : std_logic;
    C : std_logic;
    V : std_logic;
    Z : std_logic;
    PL : std_logic;
    JB : std_logic;
    BC : std_logic;
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );
    COMMENT : string
) IS
    VARIABLE RES_LINE : LINE;
BEGIN
    write( RES_LINE, clk, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, nRST, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, N, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, C, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, V, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, Z, right, 1 );
    write( RES_LINE, HT );

    write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

    write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
    writeline( RESULTS, HDR_line );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

    JB_address    <= ( 0=>'0', others=>'1' );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

    nRST    <= '1';
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]


```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width      )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
BEGIN  
    write( RES_LINE, clk, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, nRST, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, N, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, C, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, V, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, Z, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address  <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST  <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```


[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
        generic map( ctr_width => ctr_width      )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
    BEGIN  
        write( RES_LINE, clk, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, nRST, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, N, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, C, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, V, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, Z, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address  <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST  <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
JB     <= '1';

```



```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width      )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
BEGIN  
    write( RES_LINE, clk, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, nRST, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, N, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, C, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, V, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, Z, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address  <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST  <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```



```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width      )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
    BEGIN  
        write( RES_LINE, clk, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, nRST, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, N, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, C, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, V, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, Z, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

    write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
    writeline( RESULTS, HDR_line );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

    JB_address    <= ( 0=>'0', others=>'1' );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

    nRST    <= '1';
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```



```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width )
PORT MAP (
    clk => clk,
    nRST => nRST,
    N => N,
    C => C,
    V => V,
    Z => Z,
    PL => PL,
    JB => JB,
    BC => BC,
    JB_address => JB_address,
    PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
    BEGIN  
        write( RES_LINE, clk, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, nRST, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, N, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, C, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, V, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, Z, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, PL, right, 1 );
```



```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address  <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST  <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
JB     <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width )
PORT MAP (
    clk => clk,
    nRST => nRST,
    N => N,
    C => C,
    V => V,
    Z => Z,
    PL => PL,
    JB => JB,
    BC => BC,
    JB_address => JB_address,
    PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
BEGIN  
    write( RES_LINE, clk, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, nRST, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, N, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, C, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, V, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, Z, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address  <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST  <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
JB     <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```



```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width )
PORT MAP (
    clk => clk,
    nRST => nRST,
    N => N,
    C => C,
    V => V,
    Z => Z,
    PL => PL,
    JB => JB,
    BC => BC,
    JB_address => JB_address,
    PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
    BEGIN  
        write( RES_LINE, clk, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, nRST, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, N, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, C, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, V, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, Z, right, 1 );  
        write( RES_LINE, HT );  
  
        write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address  <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST  <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```



```

-- *****
-- **** STUDENT: 64240430
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width      )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
BEGIN  
    write( RES_LINE, clk, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, nRST, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, N, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, C, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, V, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, Z, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

    write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
    writeline( RESULTS, HDR_line );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

    JB_address    <= ( 0=>'0', others=>'1' );
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

    nRST    <= '1';
    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

    WAIT FOR ( PERIOD );
    Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
BC      <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '0';
BC      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL      <= '1';
JB      <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```

[illegible]


```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

```

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

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.math_real.all;
library work;
use work.reg_file_functions.all;
use work.branch_ctrl_functions.all;
USE IEEE.STD_LOGIC_TEXTIO.ALL;
USE STD.TEXTIO.ALL;

ENTITY branch_ctrl_tb IS
    generic( ctr_width : natural := 16 );
END branch_ctrl_tb;

ARCHITECTURE ndv OF branch_ctrl_tb IS

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

    COMPONENT branch_ctrl
        generic( ctr_width : natural := 16 );
    PORT (
        clk : In std_logic;
        nRST : In std_logic;
        N : In std_logic;
        C : In std_logic;
        V : In std_logic;
        Z : In std_logic;
        PL : In std_logic;
        JB : In std_logic;
        BC : In std_logic;
        JB_address : In std_logic_vector ( ctr_width-1 DownTo 0 );
        PC : Out std_logic_vector ( ctr_width-1 DownTo 0 )
    );
END COMPONENT;

```

```

SIGNAL      clk : std_logic := '0';
SIGNAL      nRST : std_logic := '0';
SIGNAL      N : std_logic := '0';
SIGNAL      C : std_logic := '0';
SIGNAL      V : std_logic := '0';
SIGNAL      Z : std_logic := '0';
SIGNAL      PL : std_logic := '0';
SIGNAL      JB : std_logic := '0';
SIGNAL      BC : std_logic := '0';
SIGNAL      JB_address : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );
SIGNAL      PC : std_logic_vector ( ctr_width-1 DownTo 0 ) := ( others=>'0' );

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

BEGIN
UUT : branch_ctr1
    generic map( ctr_width => ctr_width      )
PORT MAP (
clk => clk,
nRST => nRST,
N => N,
C => C,
V => V,
Z => Z,
PL => PL,
JB => JB,
BC => BC,
JB_address => JB_address,
PC => PC
);

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
```

```
PROCEDURE Log_variables(  
    clk : std_logic;  
    nRST : std_logic;  
    N : std_logic;  
    C : std_logic;  
    V : std_logic;  
    Z : std_logic;  
    PL : std_logic;  
    JB : std_logic;  
    BC : std_logic;  
    JB_address : std_logic_vector ( ctr_width-1 DownTo 0 );  
    PC : std_logic_vector ( ctr_width-1 DownTo 0 );  
    COMMENT : string  
    ) IS  
    VARIABLE RES_LINE : LINE;  
BEGIN  
    write( RES_LINE, clk, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, nRST, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, N, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, C, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, V, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, Z, right, 1 );  
    write( RES_LINE, HT );  
  
    write( RES_LINE, PL, right, 1 );
```

```

write( RES_LINE, HT );

write( RES_LINE, JB, right, 1 );
write( RES_LINE, HT );

write( RES_LINE, BC, right, 1 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, JB_address, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT & "0x" );

hwrite( RES_LINE, PC, right, sizeof( 2**ctr_width )/4 );
write( RES_LINE, HT );
write( RES_LINE, comment );

writeline( RESULTS, RES_LINE );

END;

variable HDR_line : LINE;

BEGIN

write( HDR_line, string'( "CLK" & HT & "nRST" & HT & "N" & HT & "C" & HT & "V" & HT & "Z" & HT &
"PL" & HT & "JB" & HT & "BC" & HT & "JB_address" & HT & "PC" & HT & "!!!NEXT!!! OPERATION" ) );
writeline( RESULTS, HDR_line );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET ACTIVE" );

JB_address    <= ( 0=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JB ADDRESS SET TO -2" );

nRST    <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "RESET INACTIVE" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

```

```

Z      <= '1';
PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRZ TAKEN" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

Z      <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - RESET ZERO BIT" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRZ NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

N      <= '1';
PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );

```

```

Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N      <= '0';
PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
BC     <= '1';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '0';
BC     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( others=>'0' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - SET JUMP ADDRESS TO
ZERO" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL     <= '1';
JB     <= '1';

```

```

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "JUMP AGAIN" );

PL    <= '0';
JB    <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

JB_address <= ( 0=>'0', 1=>'0', 2=>'0', others=>'1' );
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - JB ADDRESS SET TO -8"
);

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

PL    <= '1';
BC    <= '1';
N     <= '1';

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "BRN TAKEN AGAIN" );

N     <= '0';
WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP - BRN NOT TAKEN" );

WAIT FOR ( PERIOD );
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );

WAIT FOR ( PERIOD );

```


[illegible]

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT FOR ( PERIOD );  
Log_variables( CLK, nRST, N, C, V, Z, PL, JB, BC, JB_address, PC, "COUNT UP" );
```

```
WAIT;
```

```
END PROCESS;
```

```
END ndv;
```

