ДОДАТКИ:

Додаток А. Моделювання компонентів БПП на мові VHDL

MMPC.VHD

```
library IEEE;
use IEEE.std_logic_1164.all;


entity mpmc_sch is
  port(
       RD_A : in STD_LOGIC;
       RD_RAM : in STD_LOGIC;
       TI1 : in STD_LOGIC;
       TI2 : in STD_LOGIC;
       WR_A : in STD_LOGIC;
       WR_RAM : in STD_LOGIC;
       reset : in STD_LOGIC;
       CS_RAM : out STD_LOGIC;
       RD_WR0 : out STD_LOGIC;
       AB_RAM : out STD_LOGIC_VECTOR(7 downto 0);
       DB : inout STD_LOGIC_VECTOR(7 downto 0);
       DB_RAM : inout STD_LOGIC_VECTOR(7 downto 0)
  );
end mpmc_sch;

architecture mpmc_sch of mpmc_sch is

---- Component declarations -----

component ag
  port (
       CLK : in STD_LOGIC;
       LOAD : in STD_LOGIC;
       OE_AB : in STD_LOGIC;
       OE_C : in STD_LOGIC;
       DO : out STD_LOGIC_VECTOR(7 downto 0);
       DI : inout STD_LOGIC_VECTOR(7 downto 0)
  );
end component;
component f
  port (
       CLK : in STD_LOGIC;
       FI : in STD_LOGIC;
       RESET : in STD_LOGIC;
       FO : out STD_LOGIC
  );
end component;
component reg
  port (
       CLK : in STD_LOGIC;
       DIN : in STD_LOGIC_VECTOR(7 downto 0);
       RD : in STD_LOGIC;
       DOUT : out STD_LOGIC_VECTOR(7 downto 0)
  );
end component;

---- Signal declarations used on the diagram ----

signal C1 : STD_LOGIC;
signal DI : STD_LOGIC;
signal f1 : STD_LOGIC;
signal f2 : STD_LOGIC;
signal f3 : STD_LOGIC;
```

```
signal y : STD_LOGIC;

begin

---- Processes ----

BF1 :
process (DI, TI2)
-- Section above this comment may be overwritten according to
-- "Update sensitivity list automatically" option status
        begin
                if TI2 = '1' then
                        CS_RAM <= DI;
                end if;
        end process;

BF2 :
process (TI1, f3)
-- Section above this comment may be overwritten according to
-- "Update sensitivity list automatically" option status
        begin
                if TI1 = '1' then
                        RD_WR0 <= f3;
                end if;
        end process;

----  Component instantiations  ----

U1 : f
  port map(
      CLK => TI1,
      FI => WR_A,
      FO => f1,
      RESET => reset
  );

U2 : f
  port map(
      CLK => TI1,
      FI => RD_RAM,
      FO => f2,
      RESET => reset
  );

U3 : f
  port map(
      CLK => TI1,
      FI => WR_RAM,
      FO => f3,
      RESET => reset
  );

U4 : ag
  port map(
      CLK => C1,
      DI => DB,
      DO => AB_RAM,
      LOAD => WR_A,
      OE_AB => DI,
      OE_C => RD_A
  );

U5 : reg
```

```
  port map(
       CLK => WR_RAM,
       DIN => DB,
       DOUT => DB_RAM,
       RD => f3
  );

U6 : reg
  port map(
       CLK => y,
       DIN => DB_RAM,
       DOUT => DB,
       RD => RD_RAM
  );

C1 <= f3 and RD_RAM;

DI <= f3 and f2 and f1;

y <= f2 and f1;


end mpmc_sch;
```

F.VHD

```
library IEEE;
use IEEE.std_logic_1164.all;


entity F is
  port(
       CLK : in STD_LOGIC;
       FI : in STD_LOGIC;
       RESET : in STD_LOGIC;
       FO : out STD_LOGIC
  );
end F;

architecture F of F is

---- Signal declarations used on the diagram ----

signal AND11 : STD_LOGIC;
signal AND12 : STD_LOGIC;
signal AND21 : STD_LOGIC;
signal AND22 : STD_LOGIC;
signal FF_11 : STD_LOGIC;
signal FF_12 : STD_LOGIC;
signal FF_21 : STD_LOGIC;
signal FF_22 : STD_LOGIC;
signal FF_c : STD_LOGIC;
signal FF_c0 : STD_LOGIC;
signal OR11 : STD_LOGIC;
signal OR21 : STD_LOGIC;

begin

---- Processes ----

D_FF1 :
process (FF_11, RESET, CLK)
-- Section above this comment may be overwritten according to
-- "Update sensitivity list automatically" option status
```

```
begin
        if RESET='1' then           --asynchronous RESET active High
                FF_12 <= '0';
        elsif CLK'event and CLK='1' then  --CLK rising edge
                FF_12 <= FF_11;
        end if;
end process;

D_FF2 :
process (CLK, RESET, FF_21)
-- Section above this comment may be overwritten according to
-- "Update sensitivity list automatically" option status
begin
        if RESET='1' then           --asynchronous RESET active High
                FF_22 <= '0';
        elsif CLK'event and CLK='1' then  --CLK rising edge
                FF_22 <= FF_21;
        end if;
end process;

FF_contr :
process (FI)
-- Section above this comment may be overwritten according to
-- "Update sensitivity list automatically" option status
-- declarations
begin
        FF_c<= '0';
   FF_c0<= '1';
        if rising_edge(FI) then
                FF_c <= '1';
                FF_c0<= '0';
        end if;
end process;

T_FF1 :
process (AND11)
-- Section above this comment may be overwritten according to
-- "Update sensitivity list automatically" option status
begin
                if AND11'event then FF_11<=not AND11;
end if;
end process;

T_FF2 :
process (AND21)
-- Section above this comment may be overwritten according to
-- "Update sensitivity list automatically" option status
begin
                if AND21'event then FF_21<=not AND21;
end if;
end process;

----  Component instantiations  ----

OR11 <= FF_c or FI;

OR21 <= FI or FF_c0;

AND11 <= AND12 and OR11;

AND21 <= OR21 and AND22;

FO <= AND22 and AND12;
```

```
AND12 <= not(CLK and FF_12 and FF_11);

AND22 <= not(CLK and FF_22 and FF_21);
end F;
```

ag.VHD

```
library IEEE;
use IEEE.std_logic_1164.all;


entity AG is
  port(
      CLK : in STD_LOGIC;
      LOAD : in STD_LOGIC;
      OE_AB : in STD_LOGIC;
      OE_C : in STD_LOGIC;
      DO : out STD_LOGIC_VECTOR(7 downto 0);
      DI : inout STD_LOGIC_VECTOR(7 downto 0)
  );
end AG;

architecture AG of AG is

---- Component declarations -----

component counter
  port (
      CLK : in STD_LOGIC;
      DATA : in STD_LOGIC_VECTOR(7 downto 0);
      LOAD : in STD_LOGIC;
      Q : out STD_LOGIC_VECTOR(7 downto 0)
  );
end component;

---- Signal declarations used on the diagram ----

signal AB : STD_LOGIC_VECTOR (7 downto 0);

begin

---- Processes ----

BF :
process (AB, OE_AB)
-- Section above this comment may be overwritten according to
-- "Update sensitivity list automatically" option status
            begin
                        if OE_AB = '0' then
                                    DO <= AB;
                        end if;
            end process;

BF_1 :
process (OE_C, AB)
-- Section above this comment may be overwritten according to
-- "Update sensitivity list automatically" option status
            begin
                        if OE_C = '0' then
                                    DI <= AB;
else DI<="ZZZZZZZZ";
                        end if;
            end process;
```

```
---- Component instantiations ----

U1 : counter
  port map(
       CLK => CLK,
       DATA => DI,
       LOAD => LOAD,
       Q => AB
  );


end AG;
```

Додаток Б. Програмна реалізація прикладу використання БПП в параллельно-потокових системах

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Grids;

type
  TForm1 = class(TForm)
    StringGrid1: TStringGrid;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    Edit5: TEdit;
    Edit6: TEdit;
    Button1: TButton;
    Button2: TButton;
    StringGrid2: TStringGrid;
    StringGrid3: TStringGrid;
    Button3: TButton;
    Label1: TLabel;
    Label2: TLabel;
    Edit7: TEdit;
    Label3: TLabel;
    Label4: TLabel;
    ListBox1: TListBox;
    Button4: TButton;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
  TArr=class
  a:array of integer;
  n:integer;//kilkist elementiv
  x:integer;//pochatok index
  constructor create(mas:array of integer; poch:integer;kin:integer);
  procedure divscaljar(ch:integer;sr:Tarr;g:integer);
  function add(mas1,mas2:Tarr;h:integer):Tarr;
  end;

var
fLAG:BOOLEAN;
  Form1: TForm1;
  R,n:integer;
  arr1,arr2,ar:Tarr;
implementation
```

```
{$R *.dfm}
constructor
TArr.create(mas:array of integer; poch:integer;kin:integer );
var
i:integer;
begin
self.x:=poch;
self.n:=kin-poch+1;
SetLength(self.a,self.n);
for i:=0 to self.n-1 do
self.a[i]:=mas[i];
end;

procedure TForm1.Button1Click(Sender: TObject);
var k, m, N,z,i: integer;
begin
k:=strtoint(edit1.Text);
m:=strtoint(edit2.Text);
N:=strtoint(edit5.Text);
StringGrid1.colcount:=5+2*m-1;
R:=(k* (k+1))+ (k* (round(N/m))-1) ;
StringGrid1.rowcount:=R-1;
z:=round(N/m+0.5);
StringGrid1.Cells[0,0]:='Такт';
StringGrid1.Cells[1,0]:='БПі';
for i:=0 to m-1 do
StringGrid1.Cells[2+i,0]:='Дані';
StringGrid1.Cells[2+m,0]:='Код';
StringGrid1.Cells[2+m+1,0]:='Розмір';
for i:=0 to m-1 do
StringGrid1.Cells[2+m+2+i,0]:='Результат';
StringGrid2.colcount:=N;
StringGrid2.rowcount:=1;
StringGrid3.colcount:=N;
StringGrid3.rowcount:=1;
for i:=0 to N-1 do begin
StringGrid2.Cells[i,0]:=inttostr(i);
StringGrid3.Cells[i,0]:=inttostr(i+10);
end;
end;

procedure TForm1.Button2Click(Sender: TObject);
var str1:array[0..20] of integer;
k,m,N,i,ch,j,ii,iii,jj,takt,index: integer;
begin
k:=strtoint(edit1.Text);
m:=strtoint(edit2.Text);
N:=strtoint(edit5.Text);
for i:=0 to N do begin
str1[i]:=i;
end;
takt:=1;
j:=1;
//vuvid rezults
/////
while(j<=R) do begin
StringGrid1.Cells[0,j]:=inttostr(takt);
j:=j+k;
takt:=takt+1;
end;

//////
j:=1;
```

```
while(j<=R) do begin
for i:=1 to k do begin
StringGrid1.Cells[1,j]:='БП'+inttostr(i);
j:=j+1;
 end;
 end;

/////
j:=1;
for ii:=1 to k do begin   ///po rjadkam BP
jj:=0;   //j:=ii^)
j:=ii+((ii-1)*k);
index:=0;
flag:=true;
while(index<N) do begin
for i:=0 to m-1 do begin//vzdovzh rjadka
if (ii=1) then
begin
 ar:=ar.add(arr1,arr2,n);
 for iii:=0 to N-1 do
 begin
 str1[iii]:= ar.a[iii];
 end;
end;
if ((ii=2)and (flag=true)) then
begin
flag:=false;
 ch:=strtoint(Edit7.Text);
 ar.divscaljar(ch,ar,1);
 for iii:=0 to N-1 do
 begin
 str1[iii]:= ar.a[iii];
 end;
end;
if (ii=k) then begin
StringGrid1.Cells[4+m+i,j+k]:=inttostr(str1[index]);
end;//if
StringGrid1.Cells[2+i,j]:=inttostr(str1[index]);
index:=index+1;
end;
if (ii=1) then
StringGrid1.Cells[2+m,j]:='Add'
else if (ii=2) then
StringGrid1.Cells[2+m,j]:='Mul'
else
StringGrid1.Cells[2+m,j]:='None';
StringGrid1.Cells[2+m+1,j]:=inttostr(N);
j:=j+k;
end; ///  while
end;// for

end;

function Tarr.add(mas1,mas2:Tarr;h:integer):Tarr;
var i:integer;mas4:array of integer;
begin
FLAG:=TRUE;
setlength(mas4,h);
for i:=0 to h-1 do
begin
mas4[i]:=mas1.a[i]+mas2.a[i]
end;
ar:=Tarr.create(mas4,0,h-1);
```

```
result:=ar;
 END;

procedure Tarr.divscaljar(ch:integer;sr:Tarr;g:integer);
 var i:integer;
 mnreal:array of integer;
 begin
 SetLength(mnreaL,sr.n);
 for i:=0 to sr.n-1 do
 begin
 mnreal[i]:=sELF.a[i]*ch;
 end;
ar:=Tarr.create(mnreal,0,self.N-1);



 end;


procedure TForm1.Button3Click(Sender: TObject);
var j,n:integer; str1,str2:array[0..20] of integer;

begin
N:=strtoint(edit5.Text);
for j:=0 to N-1  do
if Length(StringGrid2.Cells[j, 0]) <>0
then str1[j] := StrToInt(StringGrid2.Cells[j,0])
else str1[j]:=0;
arr1:=Tarr.Create(str1,0,N-1);

 for j:=0 to N-1  do
if Length(StringGrid3.Cells[j, 0]) <>0
then str2[j] := StrToInt(StringGrid3.Cells[j,0])
else str2[j]:=0;
arr2:=Tarr.Create(str2,0,N-1);
 MessageDlg('масиви збережені',mtcustom,[mbok],0);
end;

procedure TForm1.Button4Click(Sender: TObject);
var i,k,m,g,f,n,p,Q:integer;
 T,Tc,Tz: extended;
begin
k:=strtoint(edit1.Text);
m:=strtoint(edit2.Text);
g:=strtoint(edit3.Text);
F:=strtoint(edit4.Text);
N:=strtoint(edit5.Text);
Tc:=strtofloat(edit6.Text);
Form1.ListBox1.Items.add('Продуктивність процесора:');
P:=g*F*N*N;
form1.ListBox1.Items.add(IntToStr(p));
Form1.ListBox1.Items.add('Такт роботи системи:');
T:=N/(F*g);
form1.ListBox1.Items.add(FloatToStr(T));
Form1.ListBox1.Items.add('Об єм обчислень кожної підзадачі:');
Q:=0;
for i:=0 to m-1 do
Q:=Q+round(P*T);
form1.ListBox1.Items.add(IntToStr(Q));
Form1.ListBox1.Items.add('Період звертання до БПП:');
Tz:=Tc*m;
form1.ListBox1.Items.add(floatToStr(Tz));
end;
```