

Programowanie układów FPGA – wykład III i IV

prof. nzw. dr hab. inż. Krzysztof Poźniak
Wydział Elektroniki i Technik Informacyjnych
Instytut Systemów Elektronicznych
e-mail: pozniak@ise.pw.edu.pl,
pok. 262 GE w kor. IIB, tel: (22) 234-7954
konsultacje: wtorek 14-16

- Część deklaracyjna ciała jednostki projektowej
 - deklaracje typów i podtypów
 - definicje stałych
 - deklaracje sygnałów
- Część wykonawcza ciała jednostki projektowej
 - instrukcje przypisania sygnału
 - część deklaracyjna i wykonawcza instrukcji procesu
 - deklaracje typów, stałych i zmiennych
 - instrukcje przypisania sygnałów i zmiennych
 - instrukcja warunkowa wyboru instrukcji



Podstawowe elementy standardu VHDL

Podstawowa struktura opisu projektu dla FPGA



Podstawowe elementy standardu VHDL

Podstawowa struktura opisu projektu dla FPGA



Podstawowe elementy standardu VHDL

Sprzęg i ciało jednostki projektowej

Składnia definicji sprzęgu z listą portów:

```
entity nazwa_sprzęgu is  
[ port ( deklaracja_portu {; deklaracja_portu}) ; ]  
end [entity] [nazwa_sprzęgu] ;
```

Pełna składnia definicji ciała:

```
architektura nazwa_ciała of nazwa_sprzęgu is  
[ część_deklaracyjna ]  
begin  
[ część_wykonawcza ]  
end [architecture] [nazwa_ciała] ;
```

Podstawowe elementy standardu VHDL

Wybrane deklaracje w ciele jednostki projektowej

Składnie definicji typu i podtypu:

type nazwa_typu is definicja_typu ;

subtype nazwa_podtypu is nazwa_typu [ograniczenie] ;

- definicja typu wyliczeniowego: type znak is ('1', 'a', 'A');
- definicja typu całkowitego: type bajt is range 0 to 255;
- definicja typu rzeczywistego: type skala is 5.0 downto 0.0;
- definicja typu fizycznego:
type napiecie is range -1E18 to 1E18 units
nV; uV = 1000 nV;, kV = 1000 V;
end units napiecie;
- definicja typu tablicowego:
type wektor1 is array (bajt range <>) of bajt;
type wektor2 is array (bajt) of bajt
- definicja typu rekordowego:
type parametry_bloku is record
dana :bajt; zakres : skala
end record parametry_bloku;
subtype subwektor is wektor1(20 to 50);
- definicja podtypu:

Podstawowe elementy standardu VHDL

Wybrane deklaracje w ciele jednostki projektowej

Podstawowa składnia definicji stałej:

constant nazwa_stalej : nazwa_typu := wyrażenie ;

- definicje proste:

constant PI : real := 3.14159 ;
constant PI2 : real := PI + PI ;
constant \2PI\ : real := 2.0 * PI ;
constant krok : time := 2.23 ms;

- definicje złożone:

type wektor is array (1 to 4) of integer ;
constant wektor_stalych : wektor := (-5 , 4 , 8 , -2300);
type zestaw is record
dana : bit_vector(5 downto 0);
krok : time;
end record zestaw;
constant wzorzec : zestaw := ("111000" , 4 us);

Podstawowe elementy standardu VHDL

Wybrane deklaracje w ciele jednostki projektowej

Podstawowa składnia definicji sygnału:

**signal nazwa_sygnału {, nazwa_sygnału } : nazwa_typu
[:= wyrażenie] ;**

Brak klauzuli **wyrażenie** jest równoważne przypisaniu wartości początkowej sygnału równej **lewej** granicy definicji typu



Podstawowe elementy standardu VHDL

Wybrane deklaracje w ciele jednostki projektowej

Podstawowa składnia definicji sygnału:

**signal nazwa_sygnału {, nazwa_sygnału } : nazwa_typu
[:= wyrażenie] ;**

- definicje proste:

signal **przewod** : bit ; -- niejawne przypisanie '0'
signal **przewod1, przewod2** : bit := '1';

- definicje złożone:

type wektor is array (1 to 4) of bit ;
signal **magistrala1, magistrala2** : wektor; -- niejawne przypisanie "0000"
signal **magistrala** : wektor := "0011";
type zestaw is record

dana1 : bit_vector(5 downto 0);
 dana2 : bit;
end record **zestaw**;
: zestaw := ("111000", '1');

signal **pakiet**

Podstawowe elementy standardu VHDL

Sprzęg i ciało jednostki projektowej

Składnia definicji sprzęgu z listą portów:

```
entity nazwa_sprzęgu is
  [ port ( deklaracja_portu {; deklaracja_portu}) ] ;
end [entity] [nazwa_sprzęgu] ;
```

Pełna składnia definicji ciała:

```
architektura nazwa_ciała of nazwa_sprzęgu is
  [ część_deklaracyjna ]
  begin
    [ część_wykonawcza ]
  end [architecture] [nazwa_ciała] ;
```

Instrukcje
współbieżne



Podstawowe elementy standardu VHDL

Wybrane instrukcje w ciele jednostki projektowej

Podstawowa składnia prostego przypisania sygnału:

[etykieta :] nazwa_sygnału <= wyrażenie ;



- przypisanie bezpośrednie:

przewod1 <= '0' ; -- przypisanie wartości stałej
przewod2 <= przewod1; -- przypisanie innego sygnału

- przypisanie złożone:

rezultat <= argA xor argB ; -- przypisanie rezultatu operacji
type zestaw is record

dana1 : bit_vector(5 downto 0);
 dana2 : bit;
end record zestaw;

pakiet <= ("111000" , rezultat); -- przypisanie stałej i sygnału

- przypisanie częściowe:

pakiet.dana1 <= "111000"; -- przypisanie składowej rekordu

↳ pakiet.dana1(2) <= '1'; -- przypisanie subskładowej rekordu

Materiały pomocnicze do wykładowego projektu studentów PUF

```

1 entity SIGNAL_SIMP is
2     port (
3         argA      : in bit;
4         argB      : in bit;
5         wynik     : out bit
6     );
7 end SIGNAL_SIMP;
8
9 architecture cialo of SIGNAL_SIMP is
10    signal przewod1, przewod2 : bit;
11
12 begin
13
14     et1: przewod1 <= argA ;
15     przewod2 <= not(przewod1);
16     wynik <= przewod2 xor argB ;
17
18 end cialo;
19
20

```

Console

```

WARNING:ProjectMgmt - File D:/Home/Pozniak/Dokumenty/Wykłady/PUF/projekty/signal_simp/ise/SIGNAL_SIMP_vhdl.prj is missing.
WARNING:ProjectMgmt - File D:/Home/Pozniak/Dokumenty/Wykłady/PUF/projekty/signal_simp/ise/SIGNAL_SIMP.xst.xrpt is missing.
WARNING:ProjectMgmt - File D:/Home/Pozniak/Dokumenty/Wykłady/PUF/projekty/signal_simp/ise/xmsgs/xst.xmsgs is missing.
WARNING:ProjectMgmt - File D:/Home/Pozniak/Dokumenty/Wykłady/PUF/projekty/signal_simp/ise/webtalk_pn.xml is missing.
WARNING:ProjectMgmt - File D:/Home/Pozniak/Dokumenty/Wykłady/PUF/projekty/signal_simp/ise/xst is missing.

```

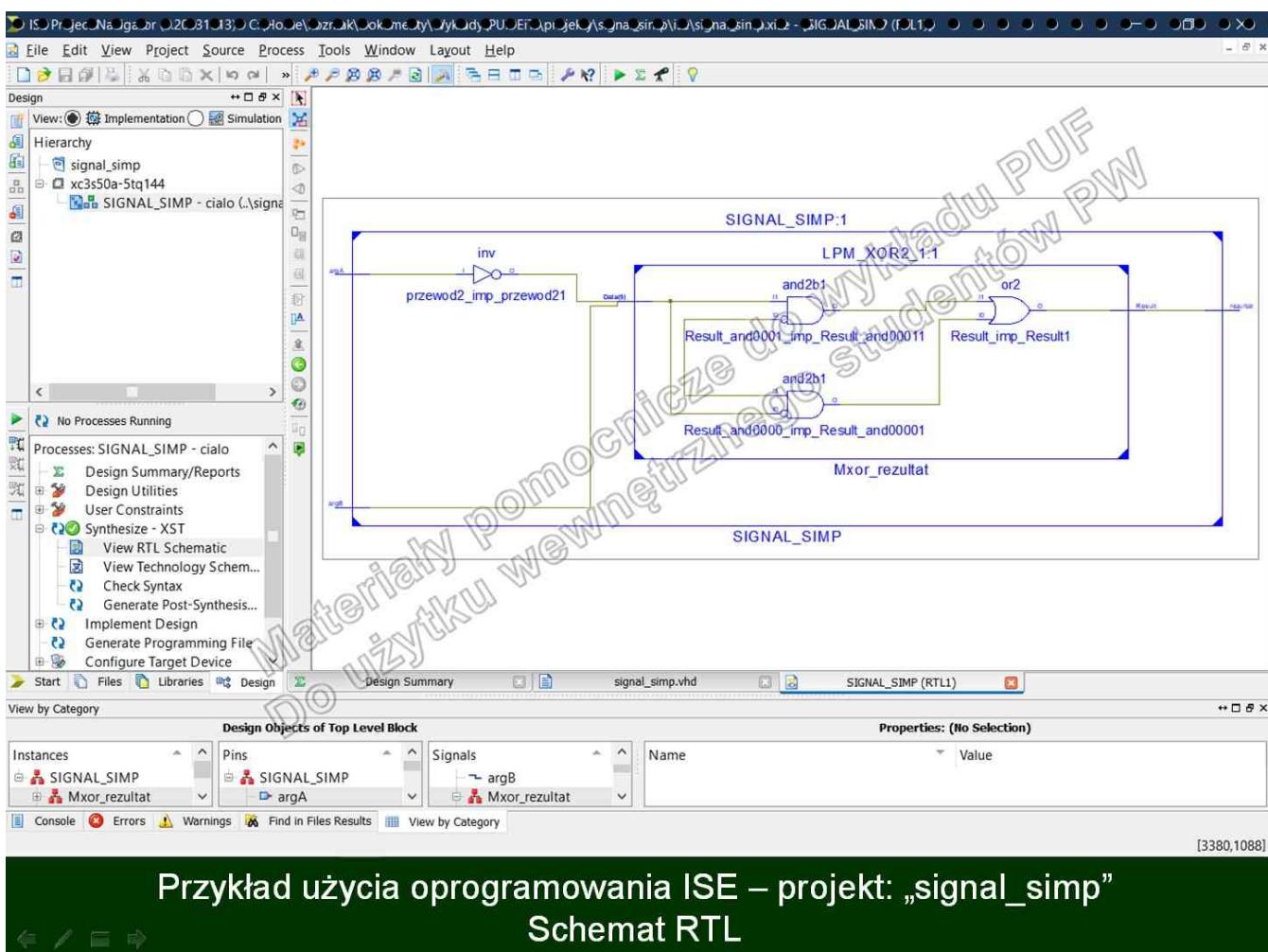
Design Summary

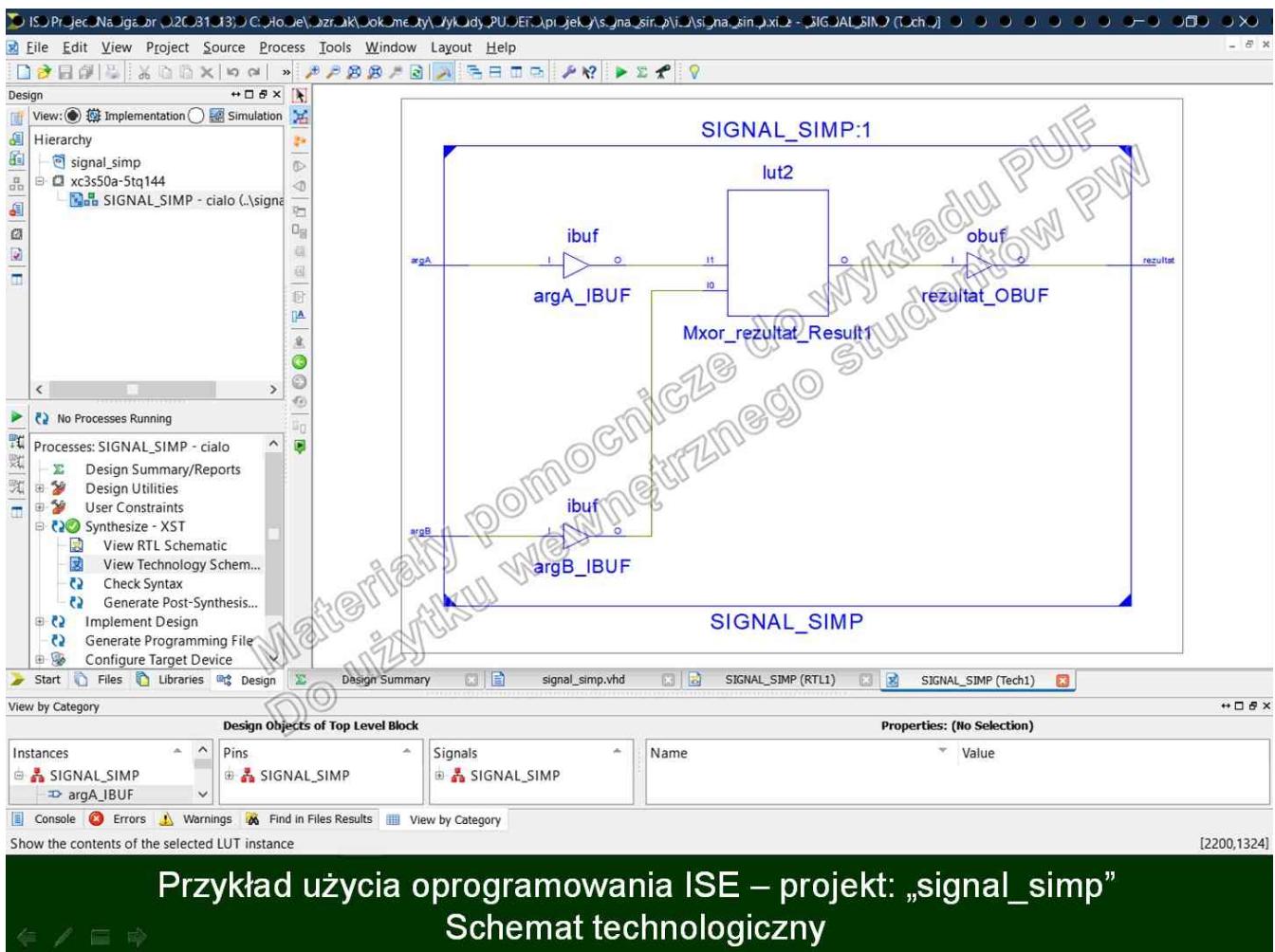
signal_simp.vhd

Ln 9 Col 1 | VHDL

Przykład użycia oprogramowania ISE – projekt: „signal_simp”

Plik źródłowy „signal_simp.vhd”





Przykład użycia oprogramowania ISE – projekt: „signal_simp” Schemat technologiczny

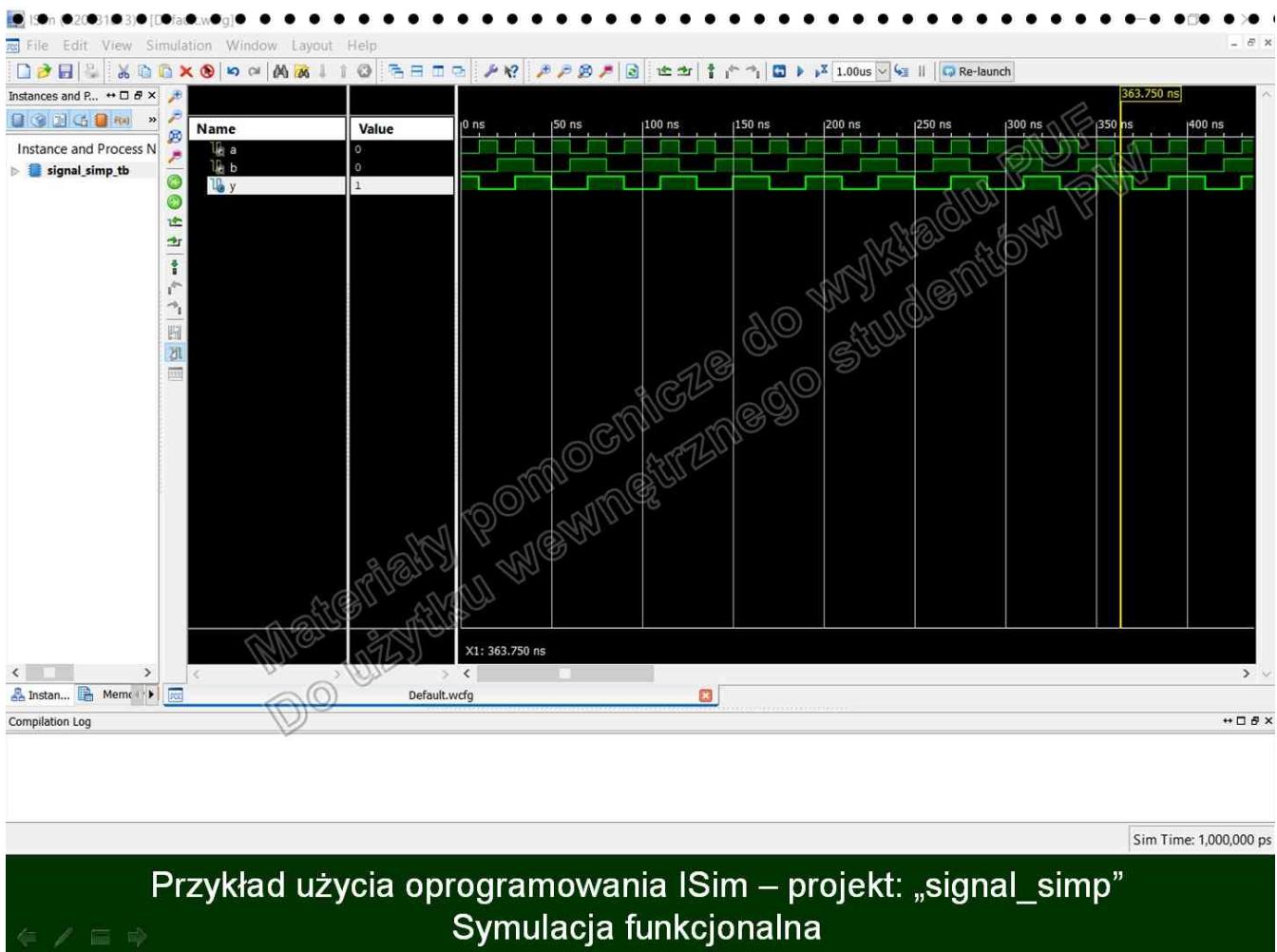
The screenshot shows the Xilinx ISE Design Suite interface with the `Design` tab selected. The left pane shows the project hierarchy with the `signal_simp` project and its sub-components. The right pane displays the VHDL code for the behavioral testbench (`signal_simp_tb.vhd`). The code defines an entity `SIGNAL_SIMP_TB` with an architecture `behavioural`. The architecture contains two processes. The first process sets signal `A` to '0', waits 10 ns, then sets it to '1', waits 10 ns, and repeats. The second process does the same for signal `B`. It also instantiates the `SIGNAL_SIMP` entity and maps its ports: `argA` to `argA`, `argB` to `argB`, and `resultat` to `Y`.

```

1 entity SIGNAL_SIMP_TB is          -- pusty sprzeg projektu symulacji
2 end SIGNAL_SIMP_TB;
3
4 architecture behavioural of SIGNAL_SIMP_TB is -- cialo architektoniczne projektu
5
6   signal A    :bit;           -- symulowane wejscie 'A'
7   signal B    :bit;           -- symulowane wejscie 'B'
8   signal Y    :bit;           -- obserwowane wyjoscie 'Y'
9
10 begin                         -- poczatek ciala wykonawczego architektury
11
12   process is
13     begin
14       A <= '0';
15       wait for 10 ns;
16       A <= '1';
17       wait for 10 ns;
18     end process;
19
20   process is
21     begin
22       B <= '0';
23       wait for 20 ns;
24       B <= '1';
25       wait for 20 ns;
26     end process;
27
28   signal_simp_inst: entity work.SIGNAL_SIMP(cialo) -- instancja projektu 'SIGNAL_SIMP'
29   port map(
30     argA    => A,           -- przypisanie sygnalu 'A' do portu 'arga'
31     argB    => B,           -- przypisanie sygnalu 'B' do portu 'argb'
32     resultat => Y);        -- przypisanie sygnalu 'Y' do portu 'resultat'
33
34 end behavioural;
35

```

Przykład użycia oprogramowania ISE – projekt: „signal_simp” Plik źródłowy „signal_simp_TB.vhd”



Przykład użycia oprogramowania ISim – projekt: „signal_simp” Symulacja funkcjonalna

Podstawowe elementy standardu VHDL

Wybrane instrukcje w ciele jednostki projektowej

Podstawowa składnia warunkowego przypisania sygnału:

```
[etykieta :] nazwa_sygnału <=
wyrażenie1 when warunek1 else
{wyrażenie when warunek else}
wyrażenie_defaultowe ;
```



- przypisanie warunkowe np. do sygnału typu bit_vector:
przewod <= argA when selA = TRUE else argB ; -- przypisanie sygnału
- przypisanie warunkowe np. do sygnału typu natural:
przewod <= argA - argB when argA - 5 > argB else -- przypisanie operacji
argA when argA > argB else -- przypisanie sygnału
argB - argA when argB - 7 > argA else -- przypisanie operacji
7 ; -- przypisanie stałej

Przykład użycia oprogramowania ISE – projekt: „signal_cond”

Plik źródłowy „signal_cond.vhd”

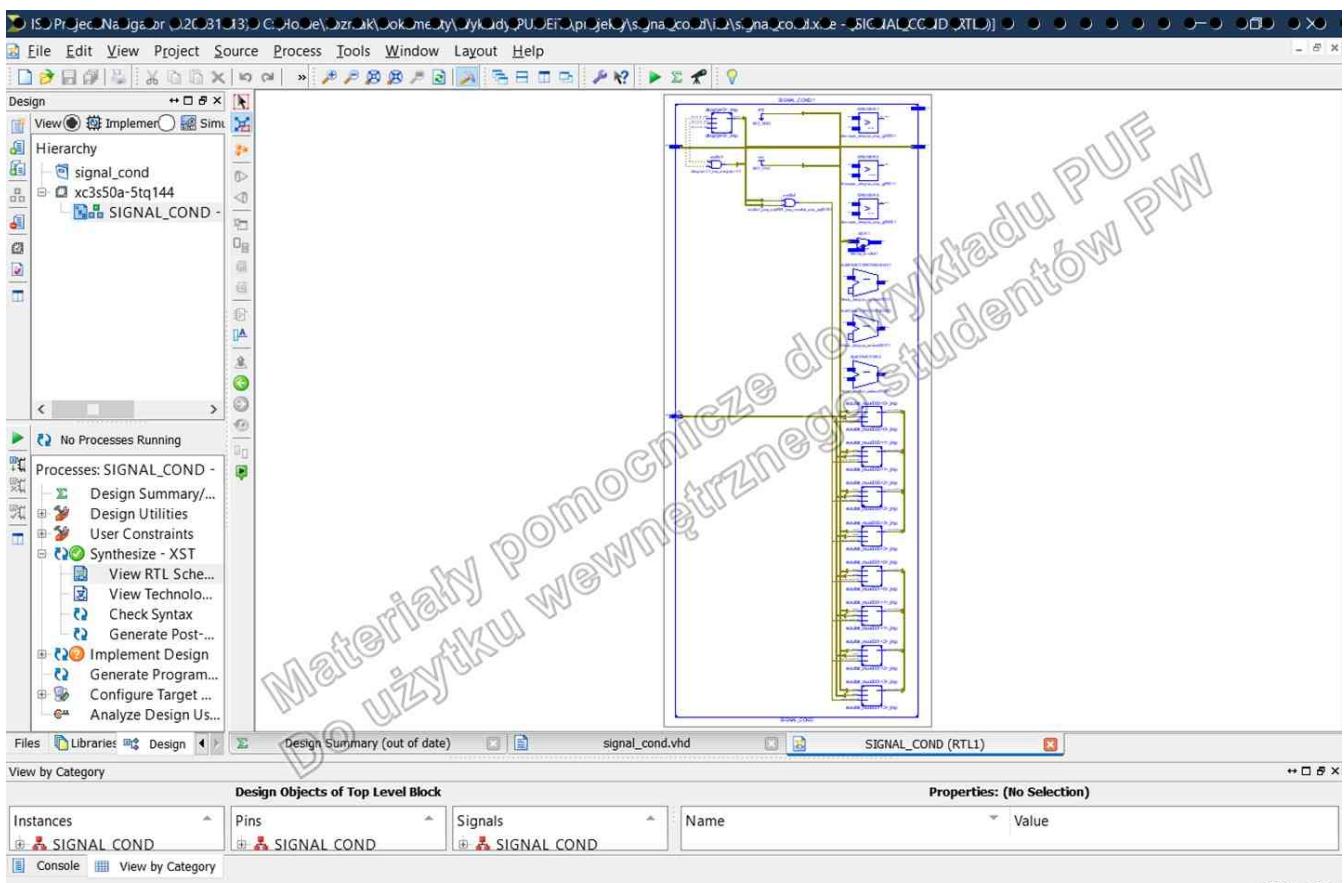
```

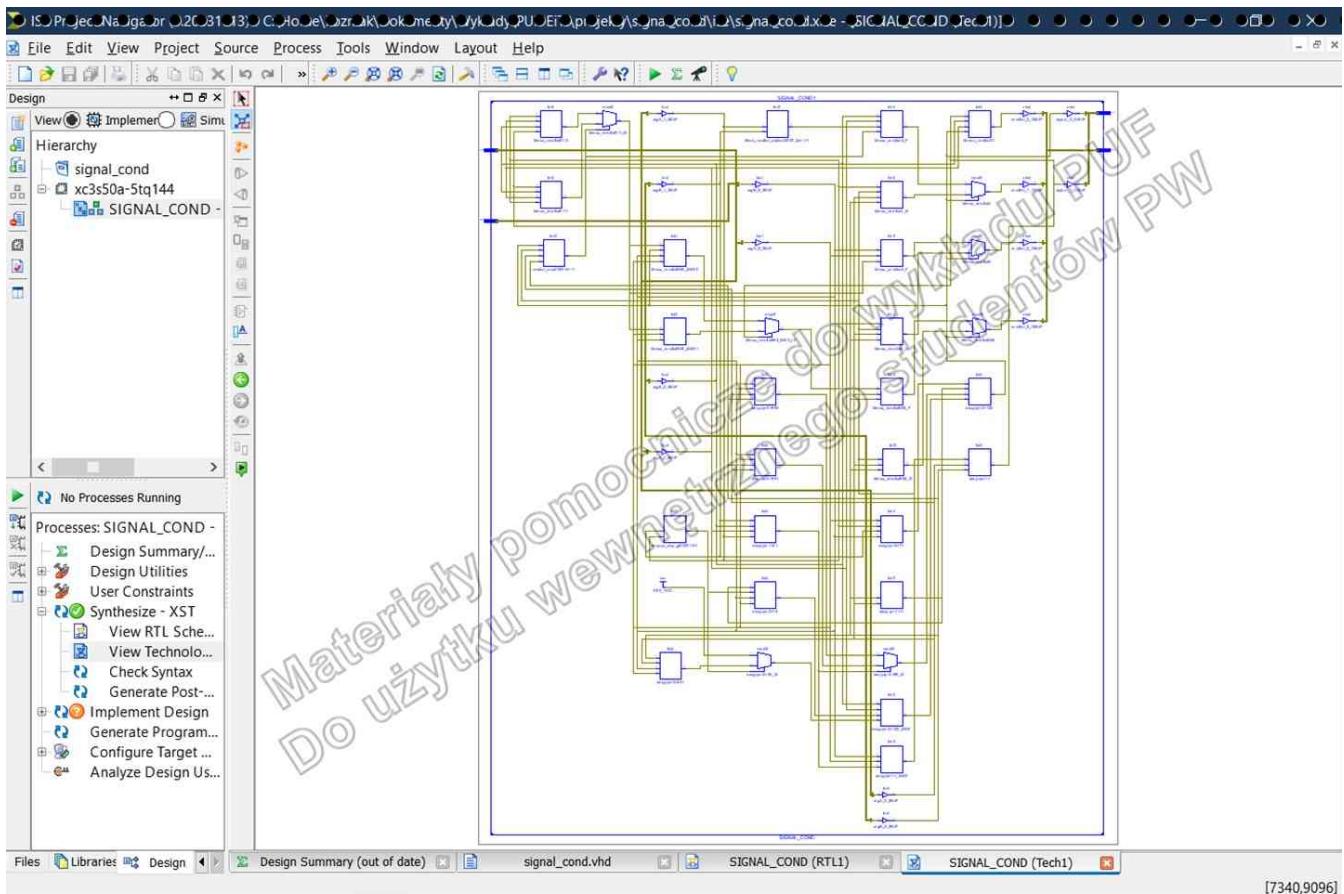
1 entity SIGNAL_COND is
2 port (
3     argA      :in natural range 0 to 15;
4     argB      :in natural range 0 to 15;
5     rezultat  :out natural range 0 to 15;
6     wybor     :out natural range 0 to 3
7 );
8 end SIGNAL_COND;
9
10 architecture cialo of SIGNAL_COND is
11
12     signal decyzja : natural range 0 to 3;
13
14 begin
15
16     decyzja <= 0 when argA = 5 > argB else
17         1 when argA > argB else
18             2 when argB = 7 > argA else
19                 3;
20
21     rezultat <= argA - argB when decyzja = 0 else
22         argA when decyzja = 1 else
23             argB - argA when decyzja = 2 else
24                 7;
25
26     wybor <= decyzja;
27
28 end cialo;
29

```

-- deklaracja sprzegu 'SIGNAL_COND'
-- deklaracja portow wejscia-wyjcia
-- deklaracja portu wejsciowego 'argA'
-- deklaracja portu wejsciowego 'argB'
-- deklaracja portu wyjsciowego 'rezultat'
-- deklaracja portu wyjsciowego 'wybor'
-- zakończenie deklaracji sprzegu
-- deklaracja ciala 'cialo' architektury
-- deklaracja sygnalu
-- początek części wykonawczej architektury
-- wybór 0 gdy 'argA = 5 > argB'
-- wybór 1 gdy 'argA > argB'
-- wybór 2 gdy 'argB = 7 > argA'
-- wybór 3 w pozostałych przypadkach
-- wybór 'argA - argB' gdy 'argA = 5 > argB'
-- wybór 'argA' gdy 'argA > argB'
-- wybór 'argB - argA' gdy 'argB = 7 > argA'
-- wybór '7' w pozostałych przypadkach
-- przypisanie sygnalu do portu wyjsciowego
-- zakończenie deklaracji ciala 'cialo'

Started : "Launching ISE Text Editor to edit signal_cond.vhd".





Przykład użycia oprogramowania ISE – projekt: „signal_cond”
Schemat technologiczny

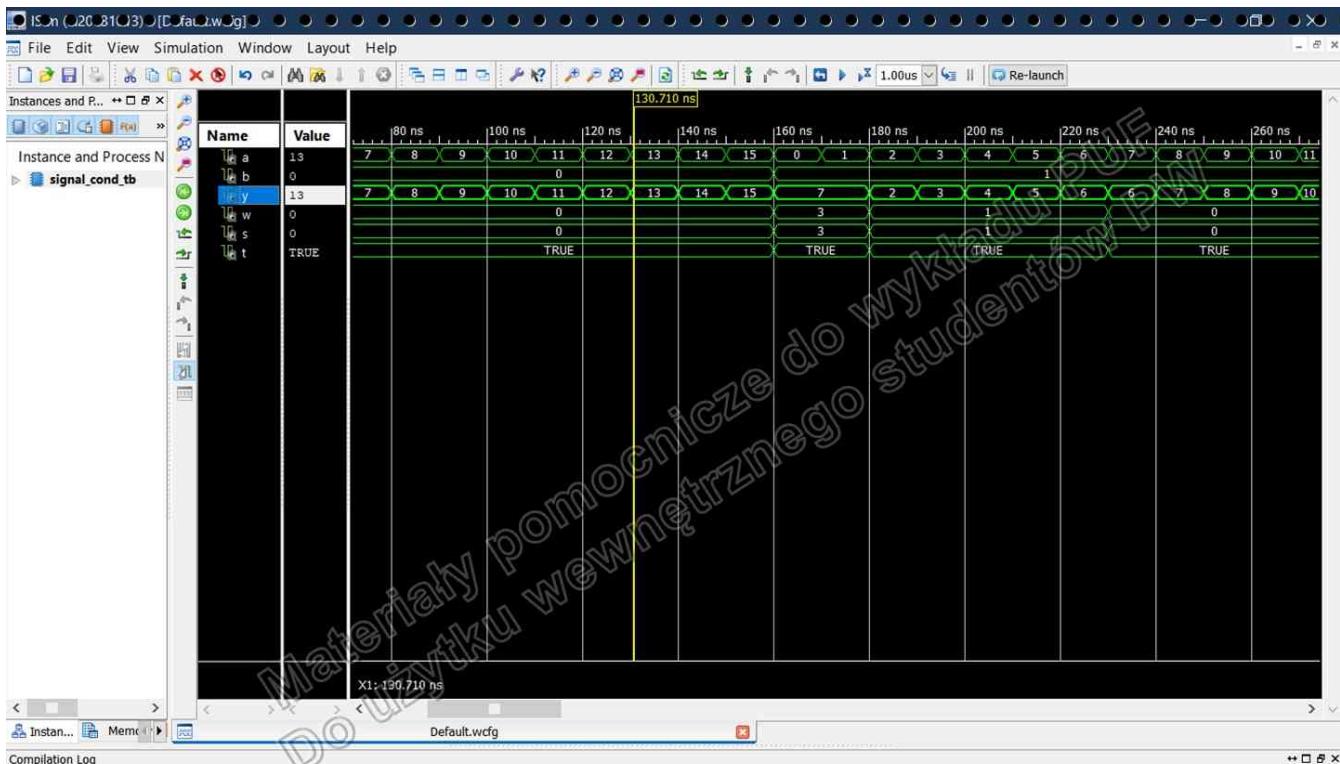
```

1 entity SIGNAL_COND_TB is
2 end SIGNAL_COND_TB;                                -- pusty szkielet projektu symulacji
3
4 architecture behavioural of SIGNAL_COND_TB is      -- cialo architektoniczne projektu
5   signal A    :natural range 0 to 15 := 0;          -- symulowane wejscie 'A' inicjowane na 0
6   signal B    :natural range 0 to 15 := 0;          -- symulowane wejscie 'B' inicjowane na 0
7   signal Y    :natural range 0 to 15;               -- obserwowane wyjście 'Y'
8   signal W    :natural range 0 to 3;                -- obserwowane wyjście 'W'
9   signal S    :natural range 0 to 3;                -- testowy sygnał wyboru 'S'
10  signal T    :boolean;                            -- testowy sygnał zgodnosci 'T'
11
12 begin
13
14   process is
15     begin
16       wait for 10 ns; A <= A+1;
17       wait for 10 ns; A <= (A+1) mod 16;
18   end process;
19
20   process is begin
21     wait for 16*10 ns; B <= (B+1) mod 16;           -- odczekanie 160 ns i zwiększenie o 1 sygnału B
22   end process;
23
24   signal_cond_inst: entity work.SIGNAL_COND(cialo) -- instancja projektu 'SIGNAL_COND'
25     port map (
26       argA    => A,
27       argB    => B,
28       rezultat => Y,
29       wybor   => W);
30
31   S <= 0 when A = 5 > B else
32     1 when A > B else
33     2 when B = 7 > A else
34     3;                                              -- wybór w pozostałych przypadkach
35
36   T <= (S = W);                                    -- zakonczenie ciala architektonicznego
37
38 end behavioural;
39

```

Design Summary (out of date) signal_cond.vhd SIGNAL_COND (RTL1) SIGNAL_COND (Tech1) signal_cond_imp_tb.vhd signal_cond_tb.vhd

Przykład użycia oprogramowania ISE – projekt: „signal_cond”
Plik źródłowy „signal_cond_TB.vhd”



Przykład użycia oprogramowania ISim – projekt: „signal_cond” Symulacja funkcjonalna

Podstawowe elementy standardu VHDL

Wybrane instrukcje w ciele jednostki projektowej

Podstawowa składnia selektywnego przypisania sygnału:

```
[etykieta :] with wyrażenie select nazwa_sygnału <=
    wyrażenie1 when wyrażenie_wyboru1 ,
    {wyrażenie when wyrażenie_wyboru ,}
    wyrażenie_defaultowe when others ;
```



- przypisanie warunkowe np. do sygnału typu bit_vector:

```
et1 : with sel select przewod <= argA when TRUE , argB when others ;
```

with sel select -- sel typu natural:

```
przewod <= argA and argB when 0 ,
            argA or argB when 1 ,
            argA xor argB when 2 ,
            argA xor argA when others ;
```

Materiały pomocnicze do wykładu PUF
Do użytku wewnętrznego studentów PW

```

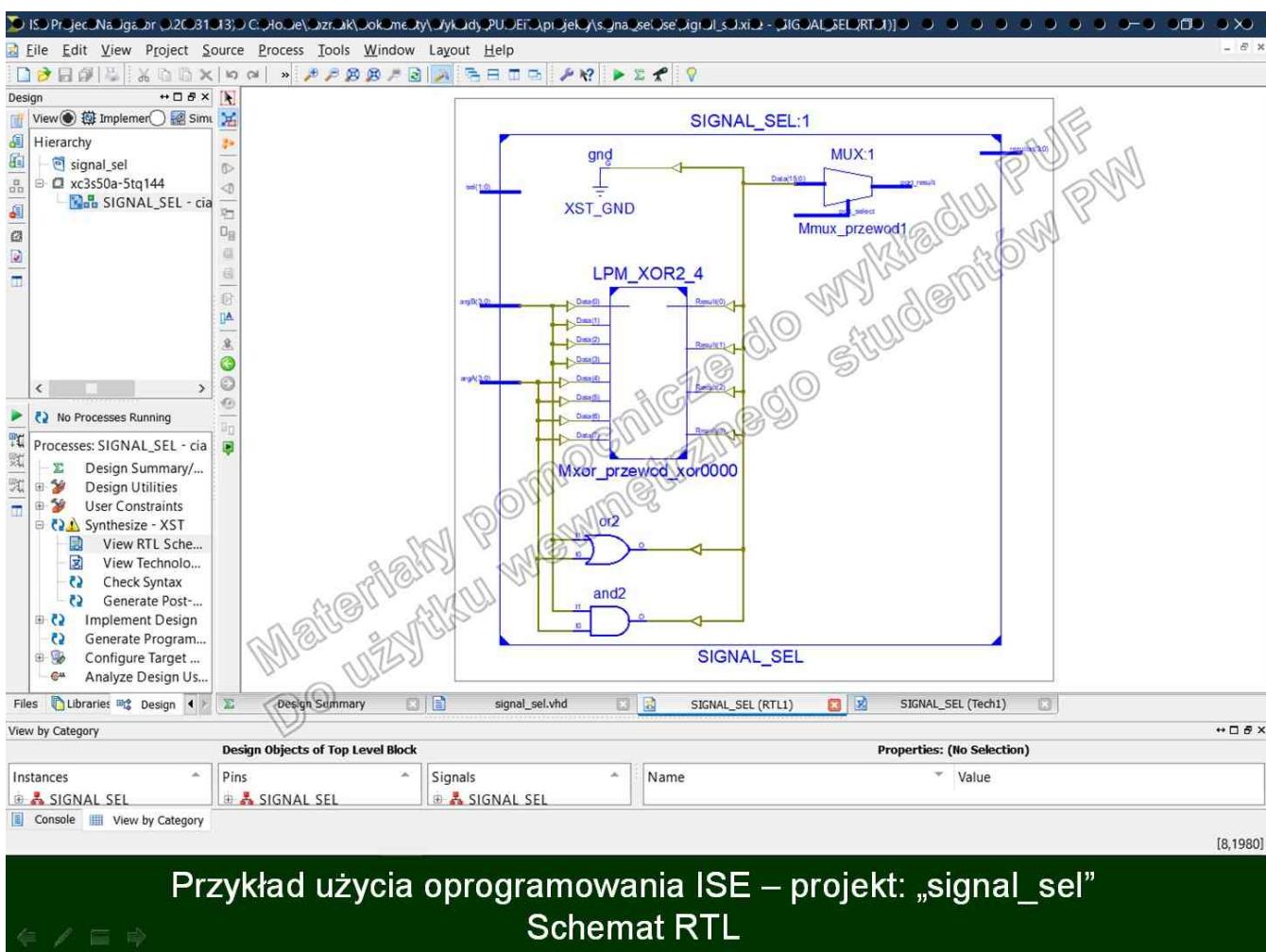
1 entity SIGNAL_SEL is
2 port (
3     sel      :in natural range 0 to 3;
4     argA    :in bit_vector(3 downto 0);
5     argB    :in bit_vector(3 downto 0);
6     rezultat :out bit_vector(3 downto 0)
7 );
8 end SIGNAL_SEL;
9
10 architecture ciale of SIGNAL_SEL is
11 begin
12     signal przewod : bit_vector(3 downto 0);
13
14     begin
15         with sel select
16             przewod <= argA and argB when 0,
17                         argA or argB when 1,
18                         argA xor argB when 2,
19                         argA xor argA when others;
20
21         rezultat <= przewod;
22     end ciale;
23
24 end;
25
-- deklaracja sprzedu 'SIGNAL_SEL'
-- deklaracja portow wejscia-wyjcia
-- deklaracja portu wejsciowego 'sel'
-- deklaracja portu wejsciowego 'argA'
-- deklaracja portu wejsciowego 'argB'
-- deklaracja portu wyjsciowego 'rezultat'
-- zakończenie deklaracji sprzedu
-- deklaracja ciała 'ciale' architektury
-- deklaracja sygnału
-- początek części wykonawczej architektury
-- klawzula wyboru poprzez sygnał 'sel'
-- wybór 'argA and argB' gdy sel=0
-- wybór 'argA or argB' gdy sel=1
-- wybór 'argA xor argB' gdy sel=2
-- wybór 'argA xor argA' w innych przypadkach
-- przypisanie sygnału do portu wyjściowego
-- zakończenie ciała architektonicznego

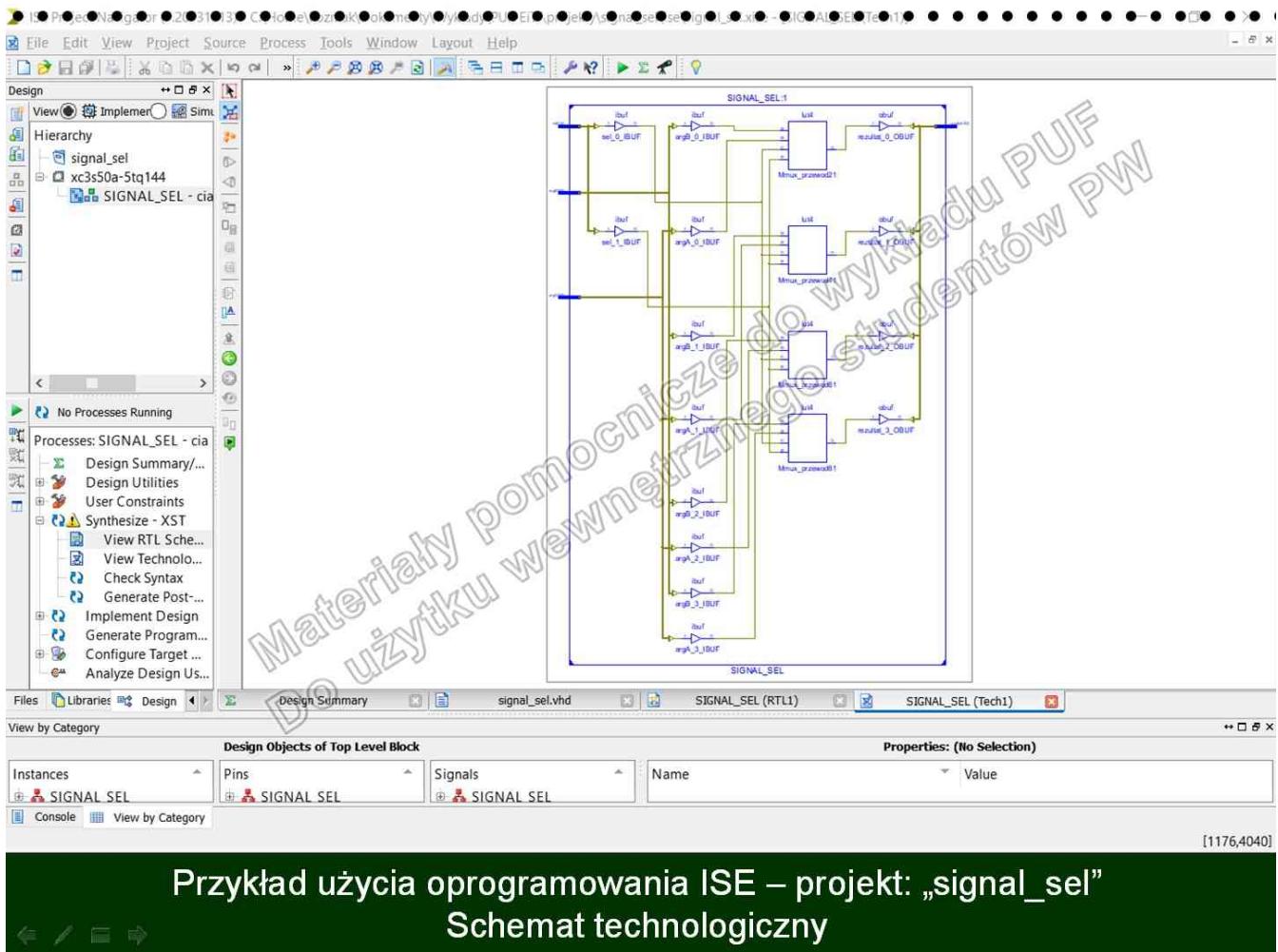
```

Started : "Launching ISE Text Editor to edit signal_sel.vhd".

Ln 10 Col 1 | VHDL

Przykład użycia oprogramowania ISE – projekt: „signal_sel” Plik źródłowy „signal_sel.vhd”





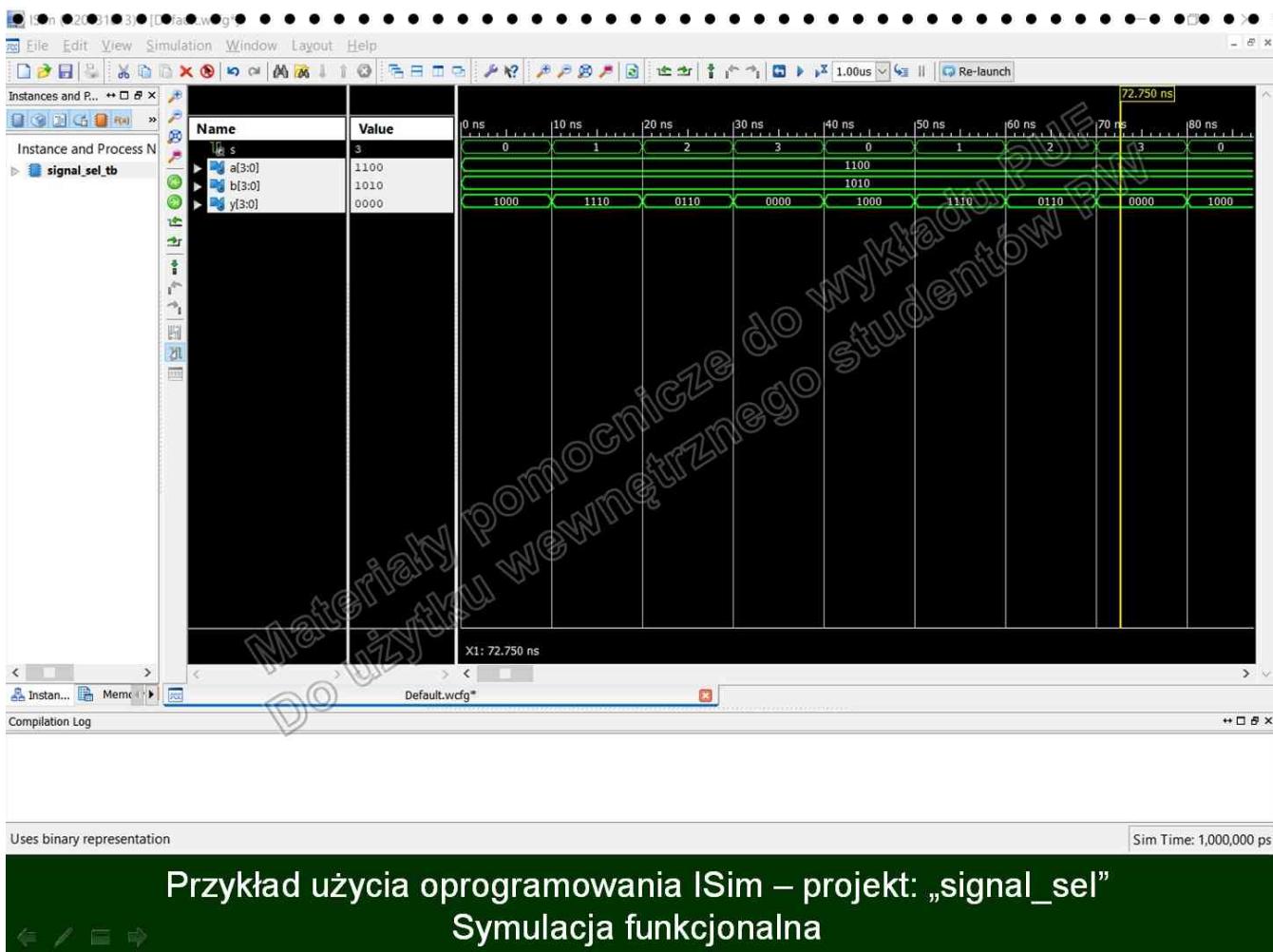
```

1 entity SIGNAL_SEL_TB is -- pusty sprzeg projektu symulacji
2 end SIGNAL_SEL_TB;
3
4 architecture behavioural of SIGNAL_SEL_TB is -- cialo architektoniczne projektu
5
6   signal S      :natural range 0 to 3; -- symulowane wejście A
7   signal A      :bit_vector(3 downto 0); -- symulowane wejście A
8   signal B      :bit_vector(3 downto 0); -- symulowane wejście B
9   signal Y      :bit_vector(3 downto 0); -- obserwowane wyjście
10
11 begin
12
13   process is
14     begin
15       A <= "1100"; B <= "1010";
16       S <= 0; wait for 10 ns;
17       S <= 1; wait for 10 ns;
18       S <= 2; wait for 10 ns;
19       S <= 3; wait for 10 ns;
20     end process;
21
22   signal_sel_inst: entity work.SIGNAL_SEL(cialo) -- instancja projektu 'SIGNAL_SEL'
23   port map (
24     sel      => S, -- przypisanie sygnalu 'S' do portu 'sel'
25     argA    => A, -- przypisanie sygnalu 'A' do portu 'argA'
26     argB    => B, -- przypisanie sygnalu 'B' do portu 'argB'
27     rezultat => Y -- przypisanie sygnalu 'Y' do portu 'rezultat'
28   );
29
30 end behavioural;
31

```

Materiały pomocnicze do wykładu PUF
Do użytku wewnętrznego studentów PW

Przykład użycia oprogramowania ISE – projekt: „signal_sel”
Plik źródłowy „signal_sel_TB.vhd”



Podstawowe elementy standardu VHDL

Wybrane instrukcje w ciele jednostki projektowej

Podstawowa składnia procesu z listą czułości:

```
[etykieta :] process (sygnał1 {, sygnał2 }) is
  [część_deklaracyjna]
  begin
    [część_wykonawcza]
  end process [etykieta];
```

Od wersji języka VHDL z 2008 r. można użyć słowa kluczowego

all

który zastępuje wszystkie wymagane sygnały w liście czułości

Podstawowe elementy standardu VHDL

Wybrane instrukcje w ciele jednostki projektowej

Podstawowa składnia procesu z listą czułości:

```
[etykieta :] process (sygnał1 {, sygnał2 } ) is
  [ część_deklaracyjna ]
    begin
      [ część_wykonawcza ]
    end process [ etykieta ];
```

- Wybrane składniki części deklaracyjnej:

- definicja typu (podobnie jak w ciele jednostki projektowej)
- definicja stałej (podobnie jak w ciele jednostki projektowej)
- definicja zmiennej (istotne różnice funkcjonalne względem sygnału!)

Podstawowa składnia definicji zmiennej:

```
variable nazwa_zmiennej {, nazwa_zmiennej } : nazwa_typu
[ := wyrażenie ] ; -- niejawnie lewa granica definicji typu
```

Podstawowe elementy standardu VHDL

Wybrane instrukcje w ciele jednostki projektowej

Podstawowa składnia procesu z listą czułości:

```
[etykieta :] process (sygnał1 {, sygnał2 } ) is
  [ część_deklaracyjna ]
    begin
      [ część_wykonawcza ]
    end process [ etykieta ];
```

Instrukcje sekwencyjne

- Wybrane składniki części deklaracyjnej:

- definicja typu (podobnie jak w ciele jednostki projektowej)
- definicja stałej (podobnie jak w ciele jednostki projektowej)
- definicja zmiennej (istotne różnice funkcjonalne względem sygnału!)

Podstawowa składnia definicji zmiennej:

```
variable nazwa_zmiennej {, nazwa_zmiennej } : nazwa_typu
[ := wyrażenie ] ; -- niejawnie lewa granica definicji typu
```

Podstawowe elementy standardu VHDL

Wybrane instrukcje w procesie

Podstawowa składnia prostego przypisania sygnału:

[etykieta :] nazwa_sygnału <= wyrażenie ;

Podstawowa składnia prostego przypisania zmiennej:

[etykieta :] nazwa_zmiennej := wyrażenie ;

- przypisanie bezpośrednie:

sygnal <= zmienna ; -- przypisanie zmiennej do sygnału
zmienna := sygnal ; -- przypisanie sygnału do zmiennej

- przypisanie złożone:

sygnal <= argA xor argB ; -- przypisanie rezultatu operacji
zmienna.dana1 := "111000"; -- przypisanie składowej rekordu
pakiet.dana1(2) := '1'; -- przypisanie subskładowej rekordu



Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automatycznie do projektu

Projekt „PROCESS_VAR”

Ciało
część wykonawcza

Nagłówek
część dekl.

```
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S: natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

-- deklaracja sprzęgu PROCESS_VAR'
-- deklaracja portu wejściowego 'A'
-- deklaracja portu wyjściowego 'Y1'
-- deklaracja portu wyjściowego 'Y2'
-- zakończenie deklaracji listy portów
-- zakończenie deklaracji nagłówka
-- deklaracja ciała 'ciało' architektury
-- deklaracja sygnału z wartością pocz.
-- początek części wykonawczej
-- lista czułości procesu
-- deklaracja zmiennej z wartością pocz.
-- część wykonawcza procesu
-- przypisanie sygnału do sygnału
-- przypisanie wyrażenia do sygnału
-- przypisanie sygnału do sygnału
-- przypisanie sygnału do zmiennej
-- przypisanie wyrażenia do zmiennej
-- przypisanie zmiennej do sygnału
-- zakończenie procesu
-- zakończenie deklaracji ciała 'ciało'

Pełny opis projektu w języku VHDL



Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

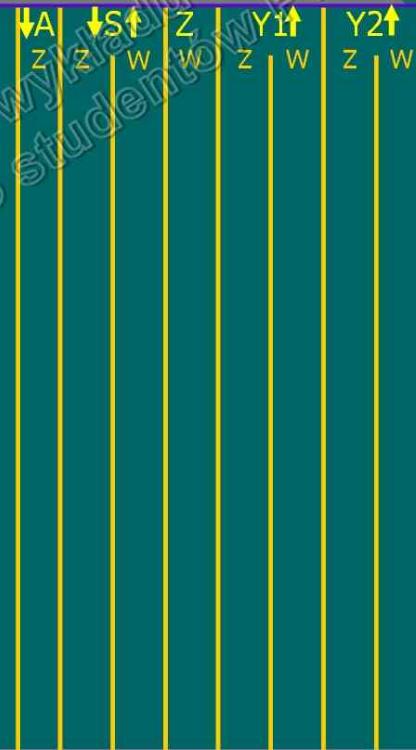
część dekl.
wykonawcza

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL



STD

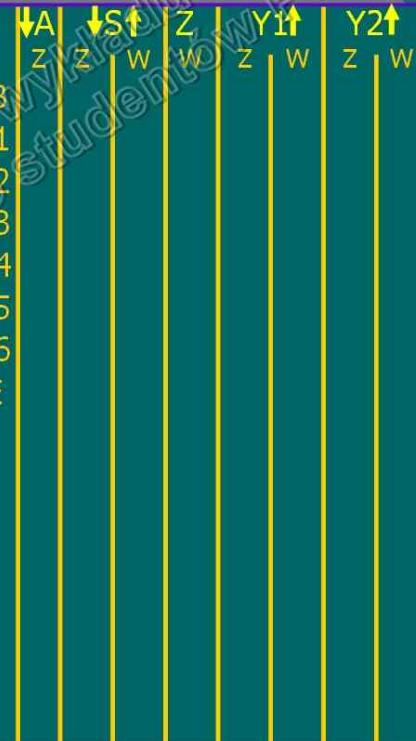
Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Pełny opis projektu w języku VHDL

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```



Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	A ↓	S ↓	Z ↑	Y1 ↑	Y2 ↑
B	z	z	w	w	w
i1	0	3	3	z	w
i2					
i3					
i4					
i5					
i6					
E					

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
    variable Z: natural := 5;
begin
    process (A) is
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	A ↓	S ↓	Z ↑	Y1 ↑	Y2 ↑
B	z	3	3	z	w
i1	0	3	3	z	w
i2					
i3					
i4					
i5					
i6					
E					

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S + 1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	A ↓	S ↑	Z ↓	Y1 ↑	Y2 ↑
B	0	3	3	5	
i1					
i2					
i3					
i4					
i5					
i6					
E					

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S + 1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	A ↓	S ↑	Z ↓	Y1 ↑	Y2 ↑
B	0	3	3	5	
i1				0	0
i2				0	0
i3				0	0
i4				0	0
i5				0	0
i6				0	0
E				0	0

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
    begin
      i1: S <= A;
      i2: S <= S + 1;
      i3: Y1 <= S;
      i4: Z := A;
      i5: Z := Z + 1;
      i6: Y2 <= Z;
    end process;
  end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	0	3	3	5	0
i1	0	3	0	0	0
i2	0	3	0	0	0
i3	0	3	0	0	0
i4	0	3	0	0	0
i5	0	3	0	0	0
i6	0	3	0	0	0
E	0				

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
    begin
      i1: S <= A;
      i2: S <= S + 1;
      i3: Y1 <= S;
      i4: Z := A;
      i5: Z := Z + 1;
      i6: Y2 <= Z;
    end process;
  end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	0	3	3	5	0
i1	0	3	0	0	0
i2	0	3	0	0	0
i3	0	3	0	0	0
i4	0	3	0	0	0
i5	0	3	0	0	0
i6	0	3	0	0	0
E	0				

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	0	3	5	0	0
i1	0	3	0	0	0
i2	0	3	0	0	0
i3	0	3	0	0	0
i4	0	3	0	0	0
i5	0	3	0	0	0
i6	0	3	0	0	0
E	0				

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	0	3	5	0	0
i1	0	3	0	0	0
i2	0	3	4	0	0
i3	0	3	0	0	0
i4	0	3	0	0	0
i5	0	3	0	0	0
i6	0	3	0	0	0
E	0				

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	0	3	3	5	0
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	5	0	0
i4	0	3	5	0	0
i5	0	3	5	0	0
i6	0	3	5	0	0
E	0				

STD

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	0	3	3	5	0
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	5	0	3
i4	0	3	5	0	0
i5	0	3	5	0	0
i6	0	3	5	0	0
E	0				

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

Nagłówek
część dekl.
wykonawcza

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S+1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z+1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Pełny opis projektu w języku VI

	↓A	↓S↑	Z	Y1↑	Y2↑
B	z	z	w	z	w
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	4	5	3
i4	0	3		0	0
i5	0	3		0	0
i6	0	3		0	0
E	0				

STD

Projekt „PROCESS_VAR”

Ciało

Nagłówek
część dekl.
wykonawcza

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S+1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z+1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Pełny opis projektu w języku VI

	↓A	↓S↑	Z	Y1↑	Y2↑
B	z	z	w	z	w
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	4	5	3
i4	0	3		0	0
i5	0	3		0	0
i6	0	3		0	0
E	0				

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało
część wykonawcza

Nagłówek
część dekl.

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S+1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z+1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	z	z	w	z	w
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	4	5	0
i4	0	3	4	0	3
i5	0	3		0	0
i6	0	3		0	0
E	0				

STD

Projekt „PROCESS_VAR”

Ciało
część wykonawcza

Nagłówek
część dekl.

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S+1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z+1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	z	z	w	z	w
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	4	5	0
i4	0	3	4	0	3
i5	0	3		1	0
i6	0	3		0	0
E	0				

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S+1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z+1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku VI

	↓A	↓S↑	Z	Y1↑	Y2↑
B	0	3	3	5	0
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	4	5	0
i4	0	3	4	0	3
i5	0	3	4	1	0
i6	0	3	4	0	0
E	0				

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S+1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z+1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku VI

	↓A	↓S↑	Z	Y1↑	Y2↑
B	0	3	3	5	0
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	4	5	0
i4	0	3	4	0	3
i5	0	3	4	1	0
i6	0	3	4	0	0
E	0				

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

Nagłówek
część dekl.
wykonawcza

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S+1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z+1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	z	z	w	w	w
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	4	5	3
i4	0	3	4	0	3
i5	0	3	4	1	3
i6	0	3	4	1	3
E	0				1

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

Nagłówek
część dekl.
wykonawcza

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S+1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z+1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑
B	z	z	w	w	w
i1	0	3	0	5	0
i2	0	3	4	5	0
i3	0	3	4	5	3
i4	0	3	4	0	3
i5	0	3	4	1	3
i6	0	3	4	1	3
E	0	4	1	3	1

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

Projekt „PROCESS_VAR”

Ciało

Nagłówek
część dekl.
wykonawcza

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3		1
B	6					
i1						
i2						
i3						
i4						
i5						
i6						
E						

STD

Projekt „PROCESS_VAR”

Ciało

Nagłówek
część dekl.
wykonawcza

```
-- biblioteka STD jest włączana automatycznie
entity PROCESS_VAR is
    port ( A : in natural;
           Y1 : out natural;
           Y2 : out natural
      );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
    signal S:natural := 3;
begin
    process (A) is
        variable Z: natural := 5;
    begin
        i1: S <= A;
        i2: S <= S + 1;
        i3: Y1 <= S;
        i4: Z := A;
        i5: Z := Z + 1;
        i6: Y2 <= Z;
    end process;
end architecture ciało;
```

Pełny opis projektu w języku VHDL

	↓A	↓S↑	Z	Y1↑	Y2↑	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3		1
B	6	4	4			
i1						
i2						
i3						
i4						
i5						
i6						
E						

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 0;
  begin
    i1: S <= A;
    i2: S <= S + 1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
B	z	z	w	z	w	
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3		1
B	6	4	4	1		
i1						
i2						
i3						
i4						
i5						
i6						
E						

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S + 1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
B	z	z	w	z	w	
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3		1
B	6	4	4	1	3	1
i1						
i2						
i3						
i4						
i5						
i6						
E						

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
    begin
      i1: S <= A;
      i2: S <= S + 1;
      i3: Y1 <= S;
      i4: Z := A;
      i5: Z := Z + 1;
      i6: Y2 <= Z;
    end process;
  end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4		1	3	1
B	6	4	4	1	3	1
i1	6	4			3	1
i2	6	4			3	1
i3	6	4			3	1
i4	6	4			3	1
i5	6	4			3	1
i6	6	4			3	1
E	6					

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
    begin
      i1: S <= A;
      i2: S <= S + 1;
      i3: Y1 <= S;
      i4: Z := A;
      i5: Z := Z + 1;
      i6: Y2 <= Z;
    end process;
  end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4		1	3	1
B	6	4	4	1	3	1
i1	6	4	6		3	1
i2	6	4			3	1
i3	6	4			3	1
i4	6	4			3	1
i5	6	4			3	1
i6	6	4			3	1
E	6					

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
    begin
      i1: S <= A;
      i2: S <= S + 1;
      i3: Y1 <= S;
      i4: Z := A;
      i5: Z := Z + 1;
      i6: Y2 <= Z;
    end process;
  end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4			3	1
i3	6	4			3	1
i4	6	4			3	1
i5	6	4			3	1
i6	6	4			3	1
E	6					

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
    begin
      i1: S <= A;
      i2: S <= S + 1;
      i3: Y1 <= S;
      i4: Z := A;
      i5: Z := Z + 1;
      i6: Y2 <= Z;
    end process;
  end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4	5		3	1
i3	6	4			3	1
i4	6	4			3	1
i5	6	4			3	1
i6	6	4			3	1
E	6					

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S + 1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4	5	1	3	1
i3	6	4			3	1
i4	6	4			3	1
i5	6	4			3	1
i6	6	4			3	1
E	6					

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S + 1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4	5	1	3	1
i3	6	4			3	1
i4	6	4			3	1
i5	6	4			3	1
i6	6	4			3	1
E	6					

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S+1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z+1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4	5	1	3	1
i3	6	4	5	1	3	1
i4	6	4		3	1	
i5	6	4		3	1	
i6	6	4		3	1	
E	6					

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S+1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z+1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4	5	1	3	1
i3	6	4	5	1	3	1
i4	6	4		6	3	
i5	6	4		3	1	
i6	6	4		3	1	
E	6					

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S+1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4	5	1	3	1
i3	6	4	5	1	3	1
i4	6	4	5	6	3	1
i5	6	4			3	1
i6	6	4			3	1
E	6					

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S+1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4	5	1	3	1
i3	6	4	5	1	3	1
i4	6	4	5	6	3	1
i5	6	4			3	1
i6	6	4			3	1
E	6					

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S + 1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4	5	1	3	1
i3	6	4	5	1	3	1
i4	6	4	5	6	3	1
i5	6	4	5	7	3	1
i6	6	4		3	4	1
E	6			3		1

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S + 1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	↓A	↓S↑	Z	Y1↑	Y2↑	
	z	z	w	z	w	
B	0	3	3	5	0	0
i1	0	3	0	5	0	0
i2	0	3	4	5	0	0
i3	0	3	4	5	0	0
i4	0	3	4	0	0	0
i5	0	3	4	1	0	0
i6	0	3	4	1	0	1
E	0	4	1	3	1	
B	6	4	4	1	3	1
i1	6	4	6	1	3	1
i2	6	4	5	1	3	1
i3	6	4	5	1	3	1
i4	6	4	5	6	3	1
i5	6	4	5	7	3	1
i6	6	4		3	4	1
E	6			3		1

Podstawowe elementy standardu VHDL

Przykłady instrukcji prostego postawienia w procesie

STD

-- biblioteka STD jest włączana automat

```
entity PROCESS_VAR is
  port ( A : in natural;
         Y1 : out natural;
         Y2 : out natural
       );
end PROCESS_VAR;
architecture ciało of PROCESS_VAR is
  signal S:natural := 3;
begin
  process (A) is
    variable Z: natural := 5;
  begin
    i1: S <= A;
    i2: S <= S + 1;
    i3: Y1 <= S;
    i4: Z := A;
    i5: Z := Z + 1;
    i6: Y2 <= Z;
  end process;
end architecture ciało;
```

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	$\downarrow A$	$\downarrow S \uparrow$	Z	$Y1 \uparrow$	$Y2 \uparrow$
B	0	3	5	0	0
i1	0	3	0	0	0
i2	0	3	4	0	0
i3	0	3	4	3	0
i4	0	3	4	0	0
i5	0	3	4	3	0
i6	0	3	4	3	0
E	0	4	1	3	1
B	6	4	4	1	3
i1	6	4	6	1	3
i2	6	4	5	1	3
i3	6	4	5	1	3
i4	6	4	5	6	3
i5	6	4	5	7	3
i6	6	4	5	7	3
E	6				

Projekt „PROCESS_VAR”

Ciało

część dekl.
wykonawcza

Nagłówek

Pełny opis projektu w języku V

	$\downarrow A$	$\downarrow S \uparrow$	Z	$Y1 \uparrow$	$Y2 \uparrow$
B	0	3	5	0	0
i1	0	3	0	0	0
i2	0	3	4	0	0
i3	0	3	4	3	0
i4	0	3	4	0	0
i5	0	3	4	3	0
i6	0	3	4	3	0
E	0	4	1	3	1
B	6	4	4	1	3
i1	6	4	6	1	3
i2	6	4	5	1	3
i3	6	4	5	1	3
i4	6	4	5	6	3
i5	6	4	5	7	3
i6	6	4	5	7	3
E	6	5	7	4	7

Materiały pomocnicze do wykładu PUF
Do użytku wewnętrznego studentów PW

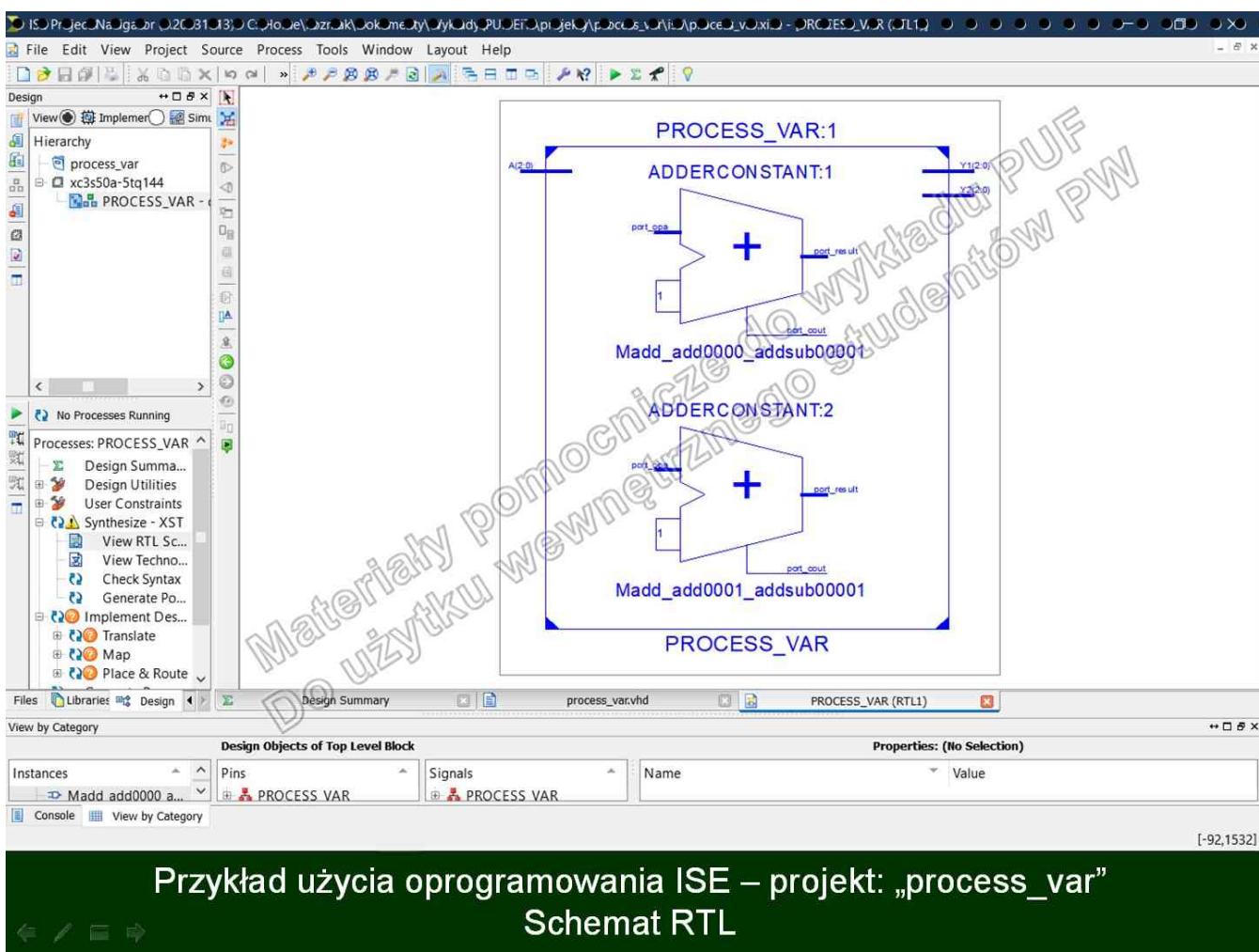
```

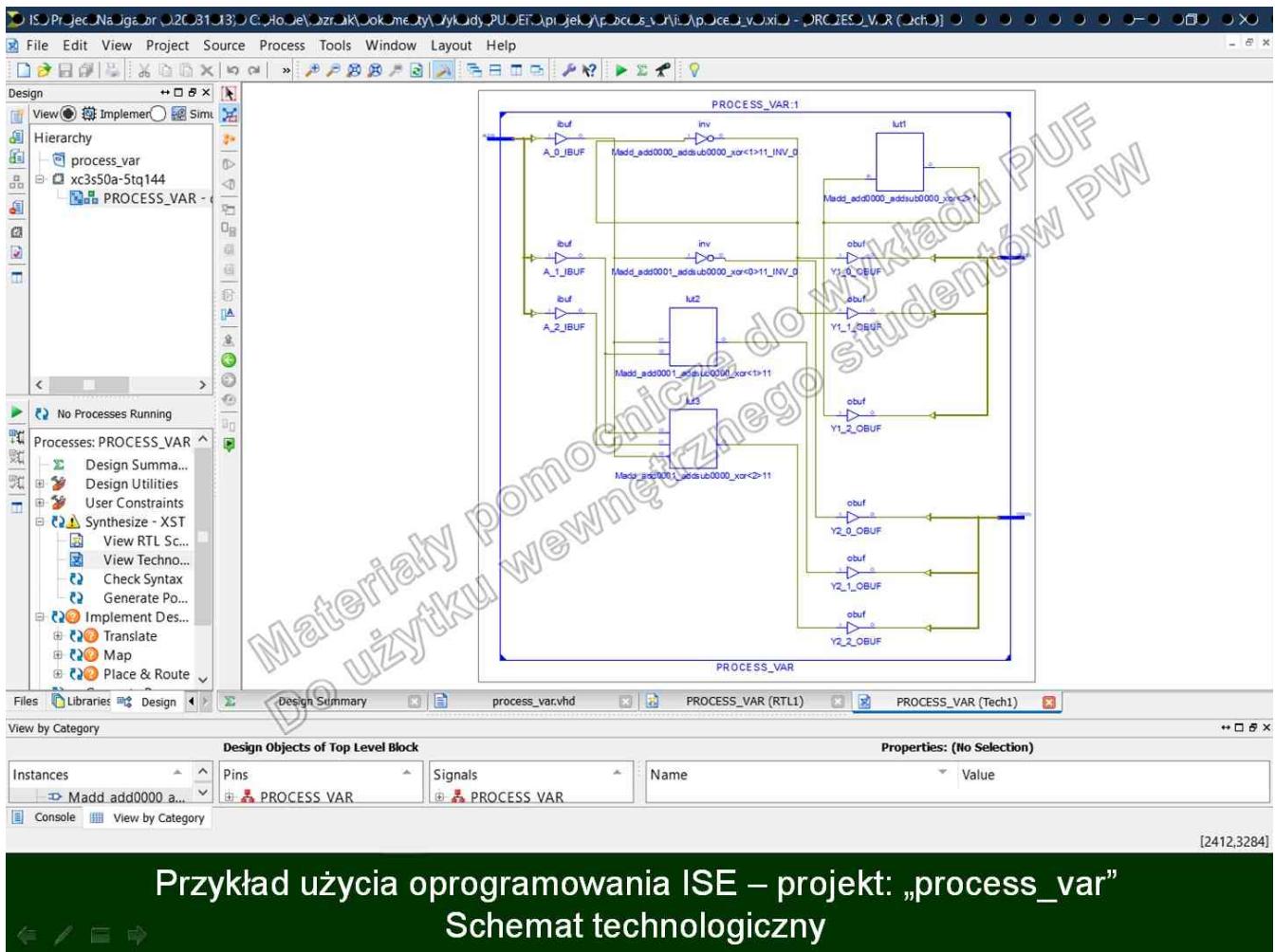
1 entity PROCESS_VAR is
2     port ( A : in natural range 0 to 7; -- deklaracja przegu PROCESS_VAR'
3             Y1 : out natural range 0 to 7; -- deklaracja portu wejściowego 'A'
4             Y2 : out natural range 0 to 7; -- deklaracja portu wyjściowego 'Y1'
5         );
6     end PROCESS_VAR;
7
8 architecture cialo of PROCESS_VAR is -- deklaracja ciała 'cialo' architektury
9     signal S :natural range 0 to 7 := 3; -- deklaracja sygnału 'S'
10    begin
11        process (A) is -- lista czulosci procesu
12            variable Z :natural range 0 to 7 := 5; -- deklaracja zmiennej 'Z'
13        begin
14            begin
15                i1: S <= A; -- przypisanie sygnału do sygnału (z etykieta)
16                i2: S <= (S + 1) mod 8; -- przypisanie wyrażenia do sygnału (z etykieta)
17                i3: Y1 <= S; -- przypisanie sygnału do sygnału (z etykieta)
18                i4: Z := A; -- przypisanie sygnału do zmiennej (z etykieta)
19                i5: Z := (Z + 1) mod 8; -- przypisanie wyrażenia do zmiennej (z etykieta)
20                i6: Y2 <= Z; -- przypisanie zmiennej do sygnału (z etykieta)
21            end process; -- zakończenie procesu
22
23        end architecture cialo; -- zakończenie deklaracji ciała 'cialo'
24

```

Started : "Launching ISE Text Editor to edit process_var.vhd".

Przykład użycia oprogramowania ISE – projekt: „process_var” Plik źródłowy „signal_var.vhd”





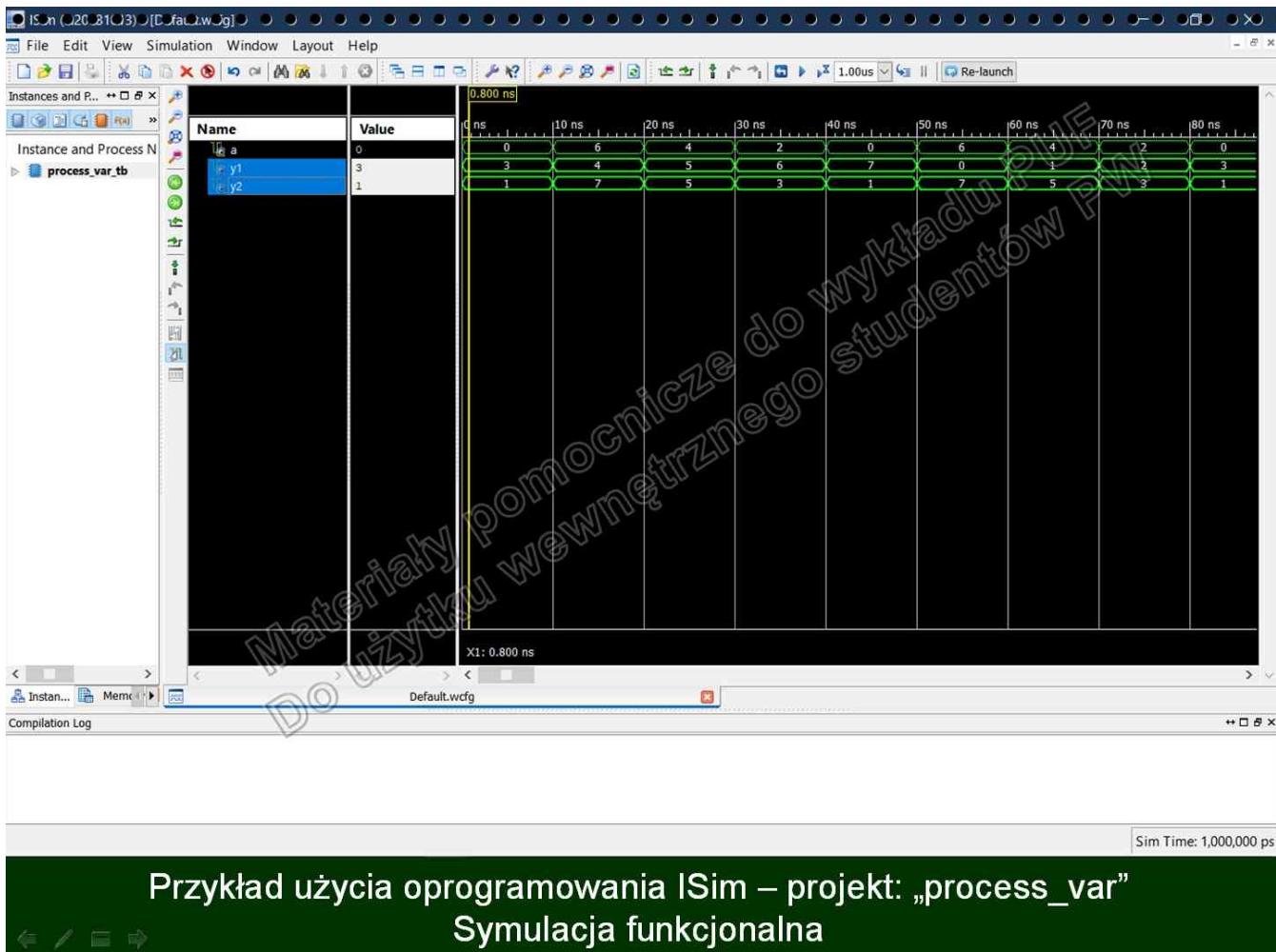
```

1 entity PROCESS_VAR_TB is -- pusty szkielet projektu symulacji
2 end PROCESS_VAR_TB;
3
4 architecture behavioural of PROCESS_VAR_TB is -- cialo architektoniczne projektu
5
6   signal A      :natural range 0 to 7; -- symulowane wejście A
7   signal Y1     :natural range 0 to 7; -- obserwowane wyjście Y1
8   signal Y2     :natural range 0 to 7; -- obserwowane wyjście Y1
9
10 begin
11
12   process is
13     begin
14       wait for 10 ns; A <= (A+6) mod 8; -- czesc wykonywaca proces
15     end process;
16
17   process_var_inst: entity work.PROCESS_VAR(cialo) -- instancja projektu 'PROCESS_VAR2'
18     port map (
19       A    => A,
20       Y1  => Y1,
21       Y2  => Y2
22     );
23
24 end behavioural;
25

```

Materiały pomocnicze do wykładu PUF
Do użytku wewnętrznego studentów PW

Przykład użycia oprogramowania ISE – projekt: „process_var”
Plik źródłowy „process_var_TB.vhd” dla symulacji funkcjonalnej



Przykład użycia oprogramowania ISim – projekt: „process_var” Symulacja funkcjonalna

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity PROCESS_VAR_IMP_TB is
end PROCESS_VAR_IMP_TB; -- pusty sprzeg projektu symulacji

architecture behavioural of PROCESS_VAR_IMP_TB is -- cialo architektoniczne projektu
begin
    process is
    begin
        wait for 10 ns; A <= A + 1;
    end process;
    process_var_inst: entity work.PROCESS_VAR(Structure) -- instancja projektu 'PROCESS_VAR2'
        port map (
            A => A,
            Y1 => Y1,
            Y2 => Y2
        );
    end behavioural;

```

Przykład użycia oprogramowania ISE – projekt: „process_var” Plik źródłowy „process_var_TB.vhd” dla symulacji implementacji

Do użycia domoczyślonego studentów PW

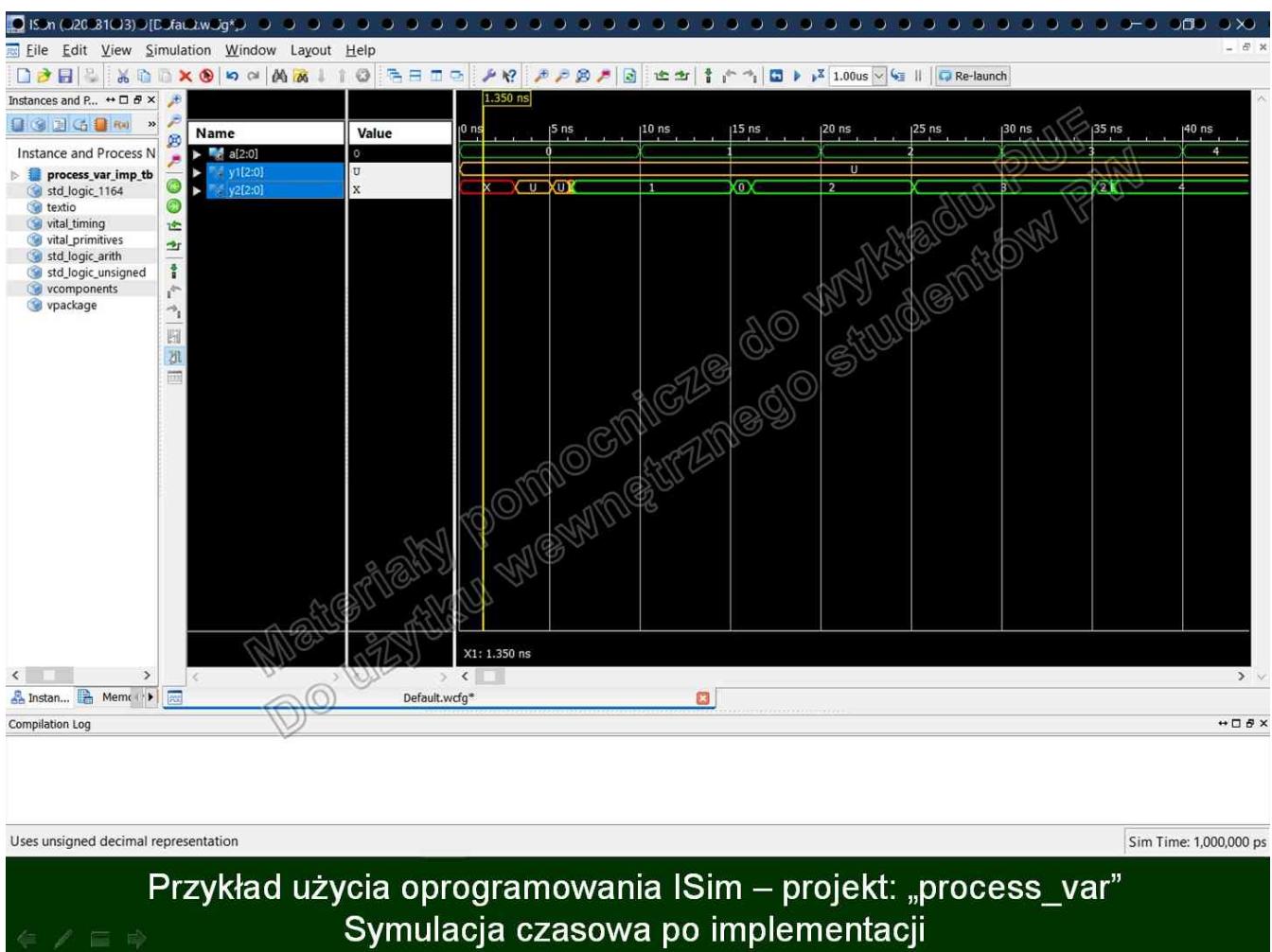
The screenshot shows the Xilinx ISE Design Suite interface. In the top left, the 'Design' menu is open, showing 'View', 'Implementer', 'Sim...', 'Post-Route', and 'Hierarchy'. The 'Hierarchy' tab is selected, displaying a tree structure of the project. The root node is 'process_var', which contains 'xc3s50a-5tq144', 'PROCESS_VAR_IMP', and 'PROCESS_VAR_TB'. Under 'PROCESS_VAR_IMP', there are 'process_var_inst' and two 'NlwBlock' components ('NlwBlockROC' and 'NlwBlockTOC'). The 'NlwBlockROC' component is expanded, showing its internal structure. To the right of the hierarchy tree is a large block of VHDL code. Below the hierarchy tree is a 'Processes' panel showing 'ISim Simulator' with options 'Post-Place & R...' and 'Simulate Post-...'. At the bottom, the 'Console' tab shows the message 'Started : "Launching ISE Text Editor to edit LCELL_timesim.vhd".' The status bar at the bottom right indicates 'Ln 38 Col 29 | VHDL'.

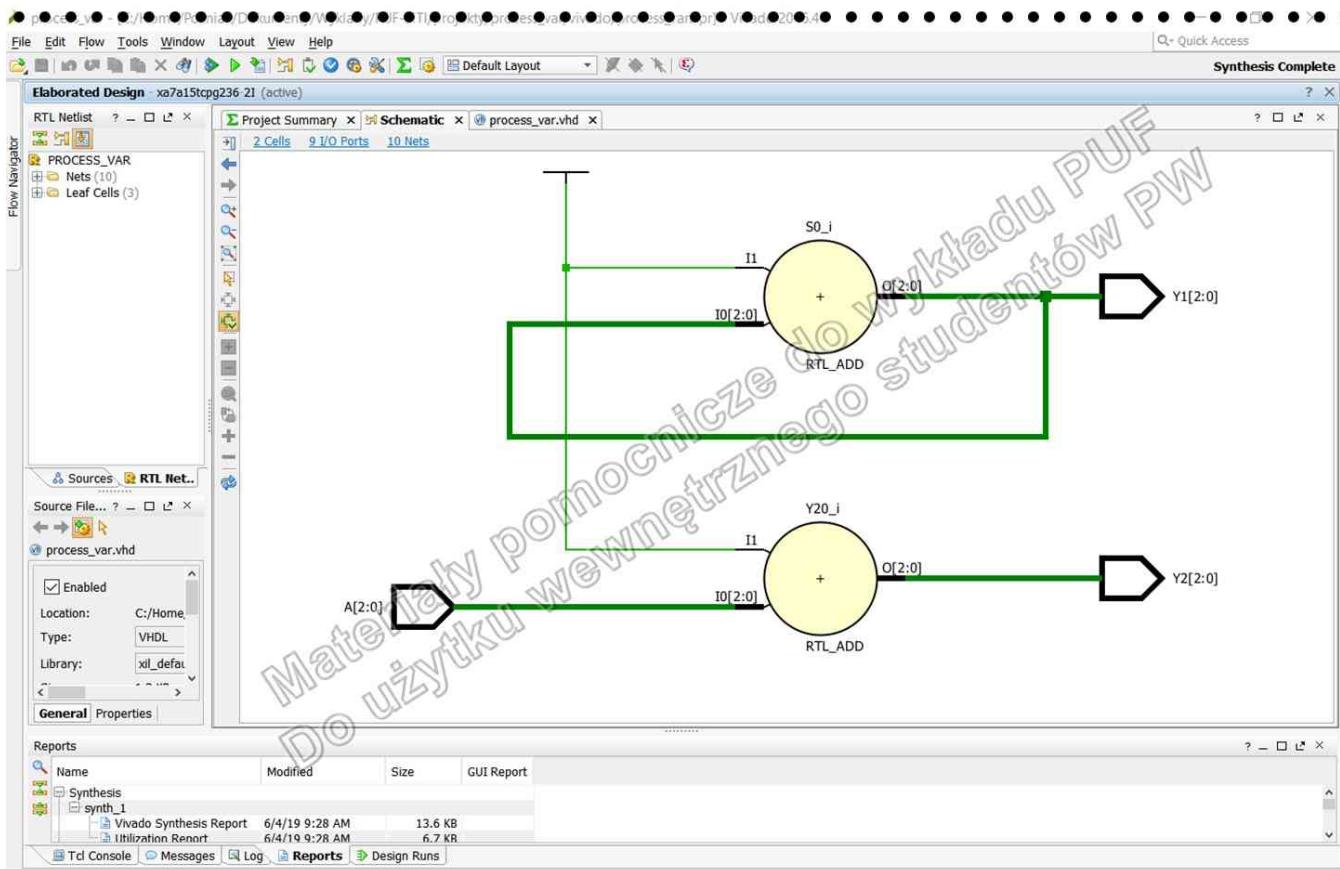
```

1 -- Copyright (c) 1995-2013 Xilinx, Inc. All rights reserved.
2
3
4
5
6   Vendor: Xilinx
7   Version: P.20131013
8   Application: netgen
9   Filename: LCELL_timesim.vhd
10  Timestamp: Tue Jun 04 09:05:52 2013
11
12
13
14  -- Command    : -intstyle ise -s 5 -pcf PROCESS_VAR.pcf -rpw 100 -tpw 0 -ar Structure -vh PROCESS_VAR -insert_pp_buffers true -w -dir netgen/par -of
15  -- Device     : 3s50atq144-5 (PRODUCTION 1.42 2013-10-13)
16  -- Input file  : PROCESS_VAR.ncd
17  -- Output file : D:\Home\Pozniak\Dokumenty\Wykłady\PUF-EiT\projekty\process_var\ise\ngen\par\LCELL_timesim.vhd
18  -- # of Entities: 1
19  -- Design Name : PROCESS_VAR
20  -- Xilinx      : C:\CAD\Xilinx\14.7\ISE_DS\ISE
21
22  -- Purpose:
23  -- This VHDL netlist is a verification model and user simulation
24  -- primitives which may not represent the true implementation of the
25  -- device, however the netlist is functionally correct and should not
26  -- be modified. This file cannot be synthesized and should only be used
27  -- with supported simulation tools.
28
29  -- Reference:
30  -- Command Line Tools User Guide, Chapter 13
31  -- Synthesis and Simulation Design Guide, Chapter 6
32
33
34
35 library IEEE;
36 use IEEE.STD_LOGIC_1164.all;
37 library SIMPRIM;
38 use SIMPRIM.VCOMPONENTS.all;
39 use SIMPRIM.VBALANCE.all;
40

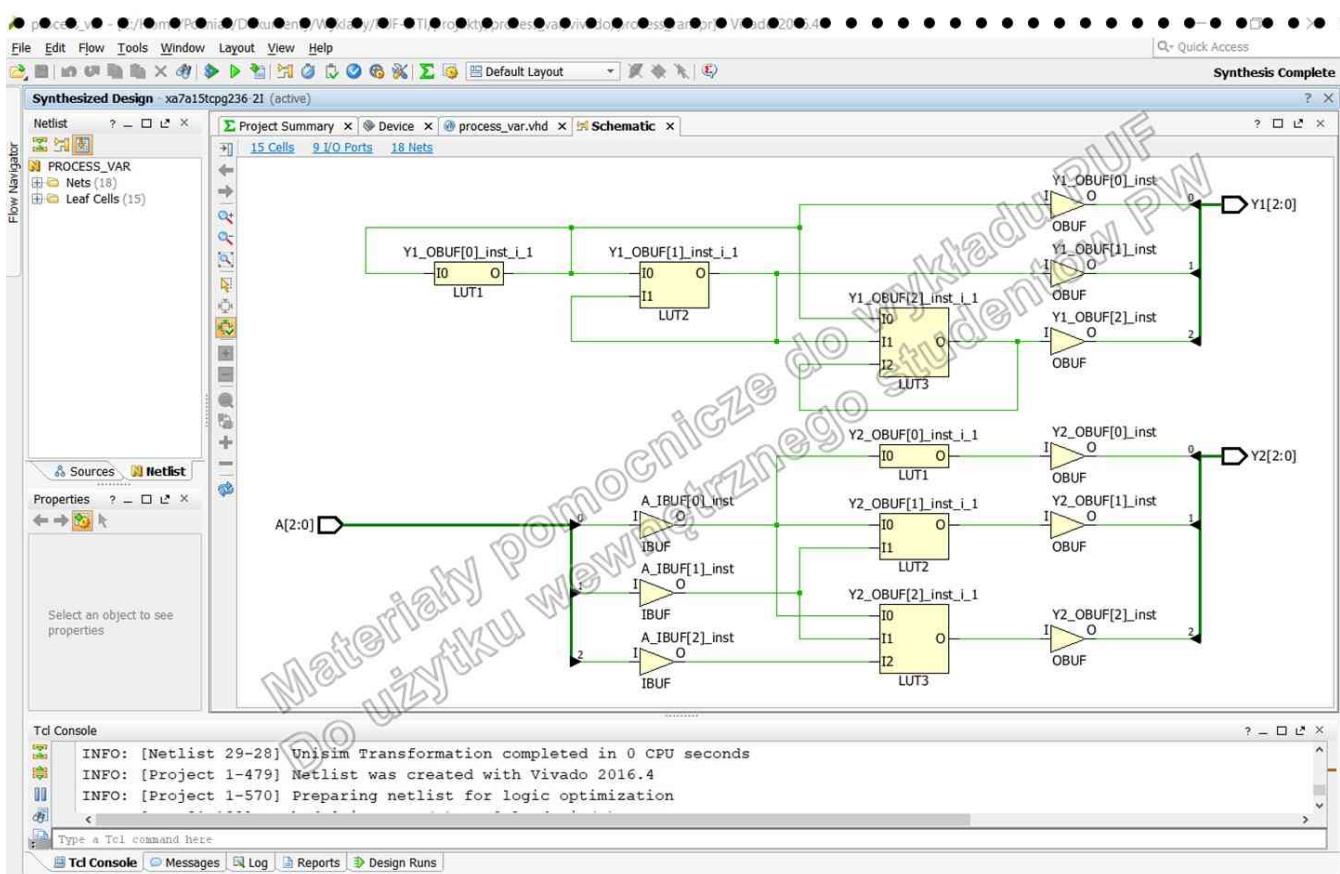
```

Przykład użycia oprogramowania ISE – projekt: „process_var”
Plik źródłowy implementacji wygenerowany przez ISE (fragment)

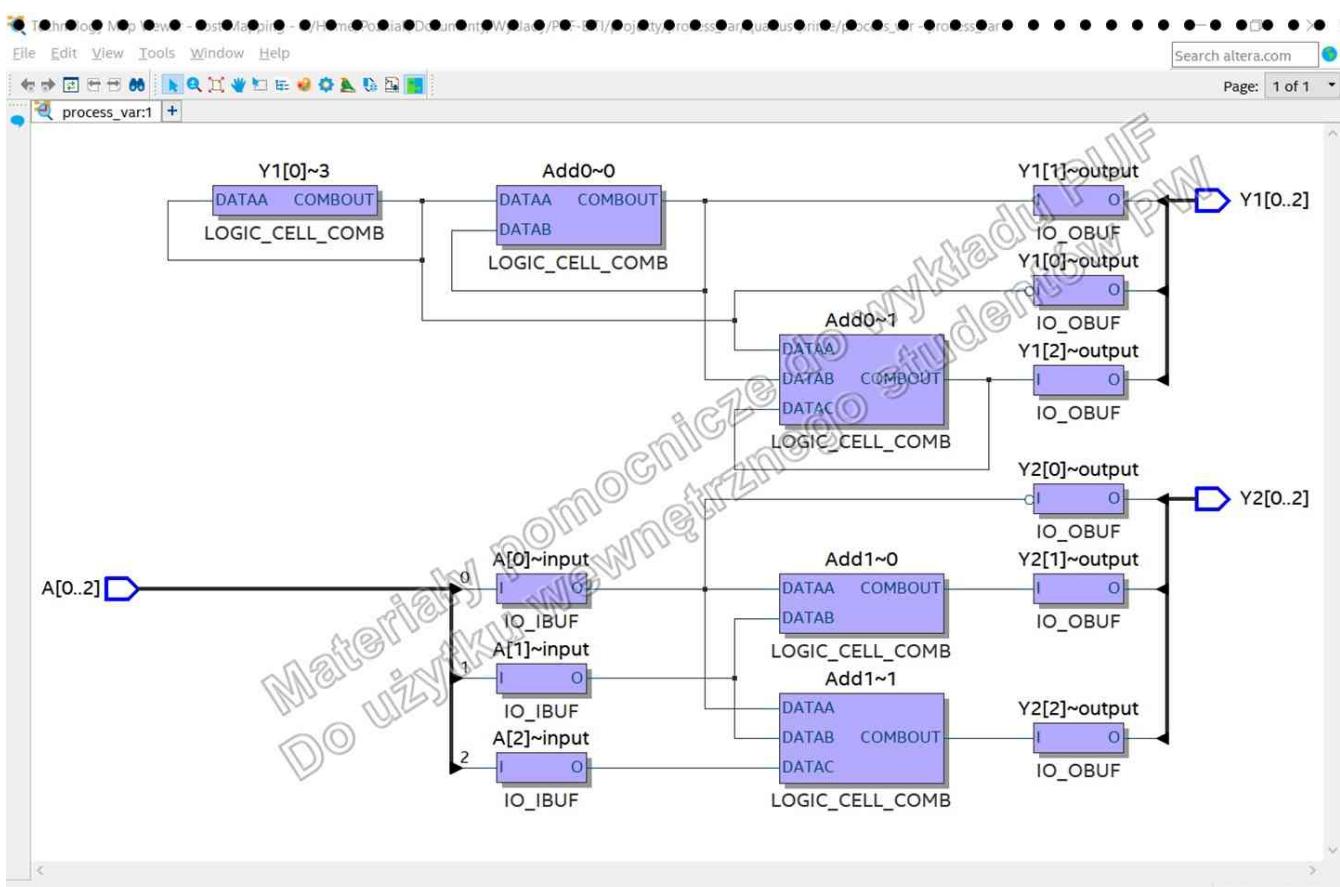
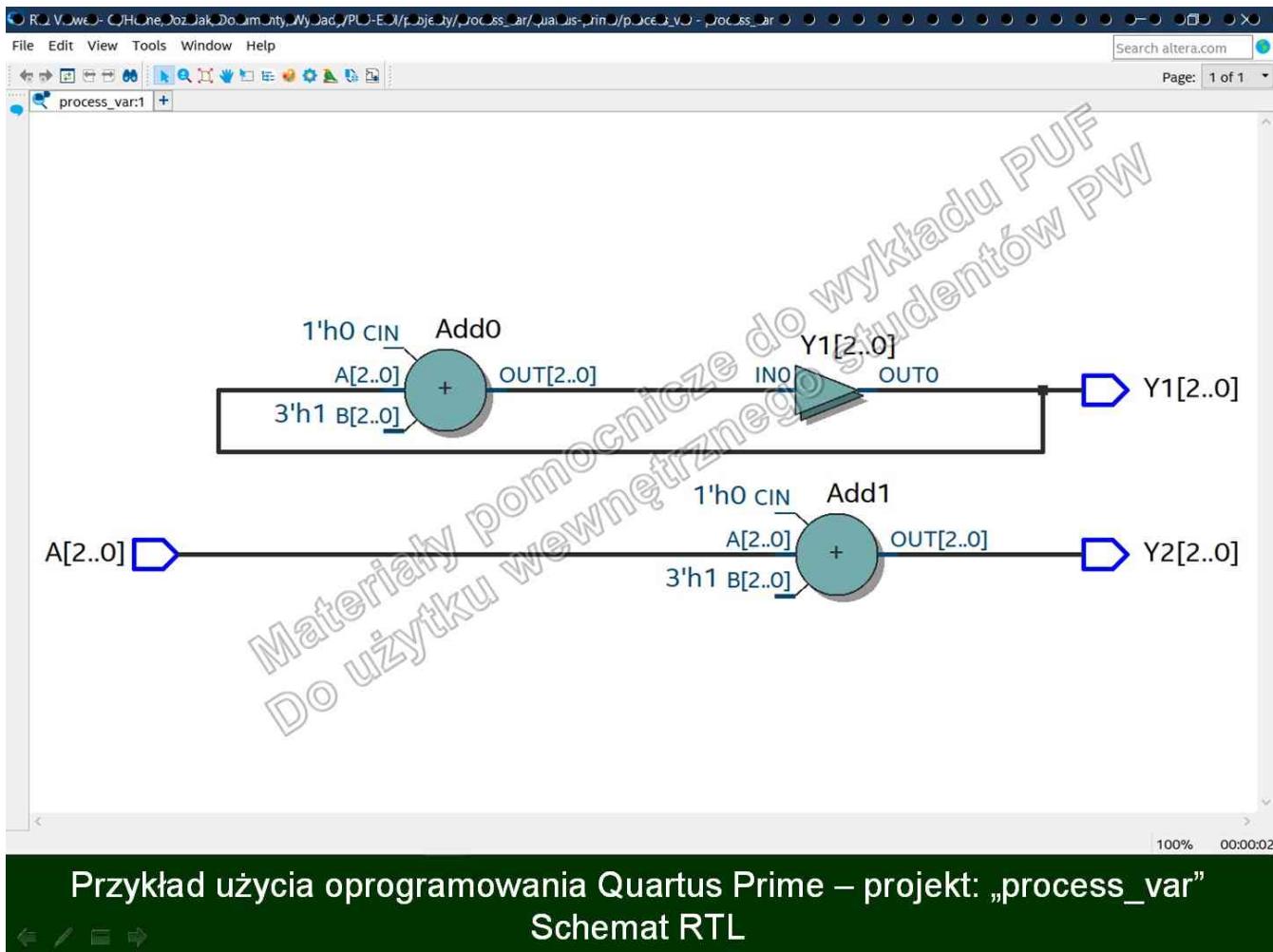




Przykład użycia oprogramowania Vivado – projekt: „process_var”
Schemat RTL



Przykład użycia oprogramowania Vivado – projekt: „process_var”
Schemat technologiczny



Podstawowe elementy standardu VHDL

Wybrane instrukcje w procesie

Podstawowa składnia warunkowego wyboru instrukcji:

```
[etykieta :] if warunek1 then instrukcja { instrukcja }
{ elsif warunek then instrukcje { instrukcja } }
[ else instrukcja_defaultowa {instrukcja_defaultowa} ]
end if [etykieta];
```

- prosty warunkowy wybór przypisania sygnału:

```
et1 : if sel=1 then S <= A; else S <= B; end if;
```

- kilkawariantowy wybór sekwencji instrukcji przypisania:

```
if sel=1 then
```

```
    S1 <= argA + 3 ; S2 <= argB ;
```

```
elsif sel=2 then
```

```
    S1 <= argB + 5 ; S3 <= argB ;
```

```
else
```

```
    S3 <= 7 ;
```

```
end if ;
```



Podstawowe elementy standardu VHDL

Przykłady instrukcji wyboru warunkowego w procesie

STD

Projekt „PROCESS_SYNC”

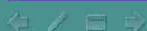
Ciało Nagłówek

część dekl.
część wykonawcza

```
-- biblioteka STD jest włączana automatycznie do projektu
entity PROCESS_SYNC is
port ( R : in bit;
      S : in bit;
      L : in bit;
      E : in bit;
      C : in bit;
      D : in bit;
      Q : out bit
);
end PROCESS_SYNC;
-- deklaracja spłzegu PROCESS_SYNC
-- deklaracja portu kasującego 'R'
-- deklaracja portu ustawiającego 'S'
-- deklaracja portu ładującego 'L'
-- deklaracja portu zezwalającego 'E'
-- deklaracja portu taktującego 'C'
-- deklaracja portu wej. danej 'D'
-- deklaracja portu wej. danej 'Q'
-- zakończenie deklaracji listy portów
-- zakończenie deklaracji nagłówka

architecture ciało of PROCESS_SYNC is
begin
process (R, S, L, D, C, E) is
begin
  if (R='1') then
    Q <= '0';
  elsif (S='1') then
    Q <= '1';
  elsif (L='1') then
    Q <= D;
  elsif (C'event and C='1') then
    if (E='1') then
      Q <= D;
    end if;
  end if;
end process;
end architecture ciało;
-- deklaracja ciała 'ciało' architektury
-- lista człosci procesu
-- czesc wykonawcza procesu
-- warunek dla sygnalu 'R'
-- przypisanie stalej do wyjścia
-- warunek dla sygnalu 'S'
-- przypisanie stalej do wyjścia
-- warunek dla sygnalu 'L'
-- przypisanie danej do wyjścia
-- warunek zbozca narastajacego 'C'
-- warunek zezwolenia na zapis
-- przypisanie danej do wyjścia
-- zakończenie instrukcji wyboru
-- zakończenie instrukcji wyboru
-- zakończenie procesu
-- zakończenie deklaracji ciała 'ciało'
```

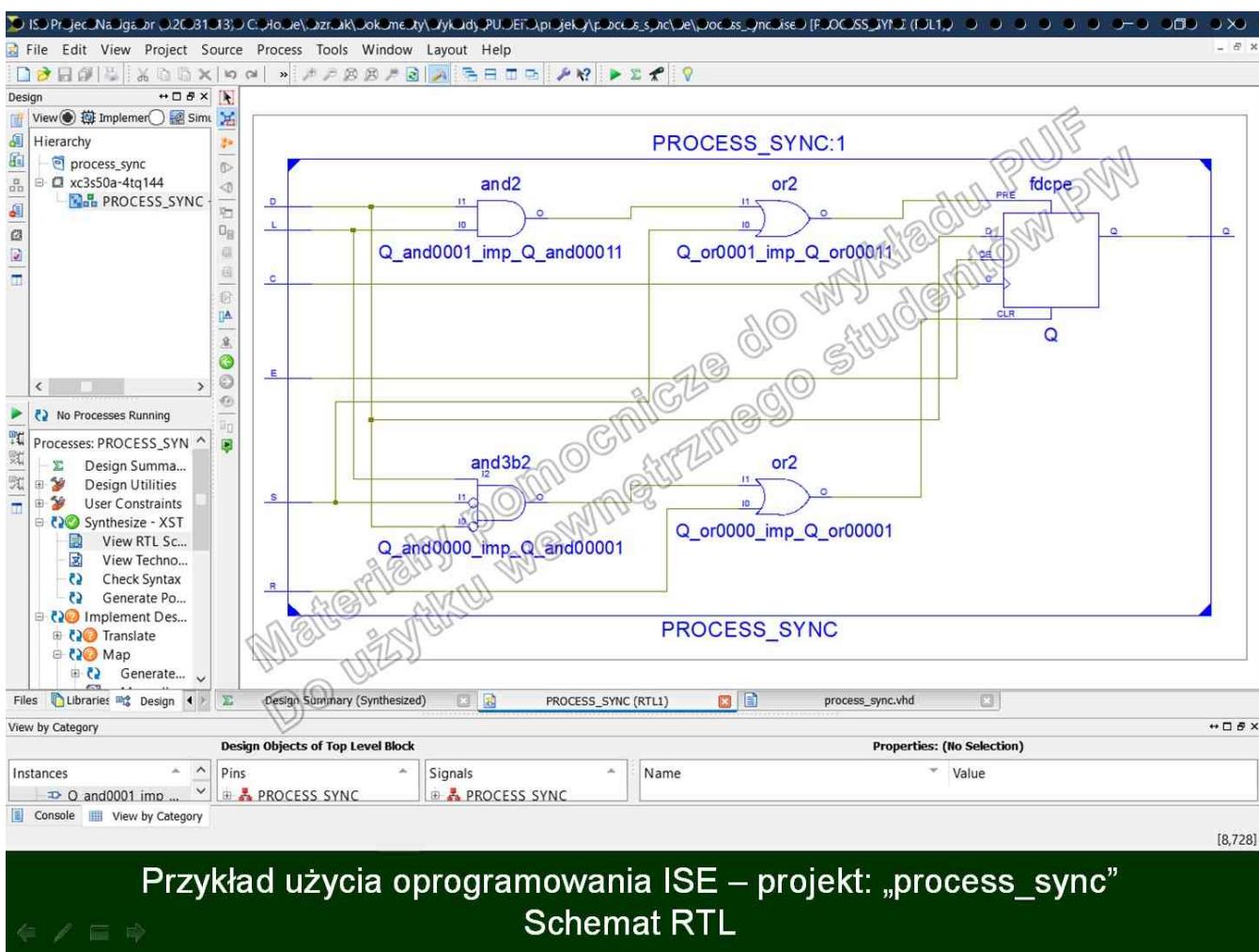
Pełny opis projektu w języku VHDL

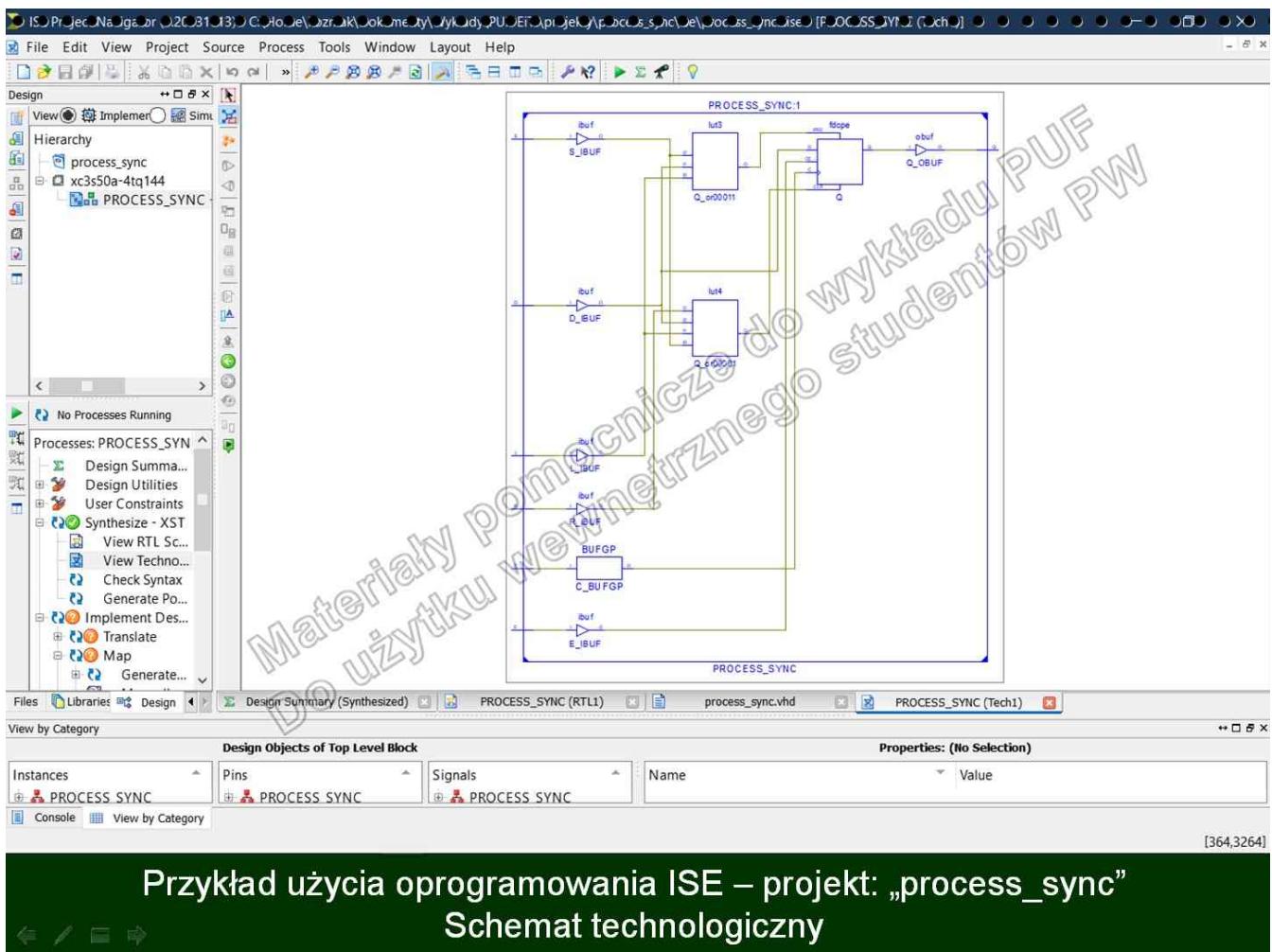


The screenshot shows the ISE 14.7 interface with the following details:

- File Menu:** File, Edit, View, Project, Source, Process, Tools, Window, Layout, Help.
- Design View:** Shows the source code for `process_sync.vhd`. The code defines an entity `PROCESS_SYNC` with a port containing signals R, S, L, E, C, D, Q. It contains an architecture `cialo` with a process that handles various logic conditions based on inputs R, S, L, D, C, E and output Q.
- Hierarchy View:** Shows the project structure with `process_sync` as the main component, which includes `xc3s50a-4tq144` and `PROCESS_SYNC`.
- Processes View:** Shows the synthesized processes:
 - Design Summary (Synthesized)
 - Design Utilities
 - User Constraints
 - Synthesize - XST
 - View RTL Sc...
 - View Techno...
 - Check Syntax
 - Generate Po...
 - Implement Des...
 - Translate
 - Map
 - Generate...
- Design Summary (Synthesized) Tab:** Shows the synthesized design summary.
- process_sync.vhd Tab:** Shows the source code tab.
- Design Objects of Top Level Block Table:** Lists instances, pins, and signals for the top-level block.

Instances	Pins	Signals	Name	Value
O_and0001_im...	PROCESS SYNC	PROCESS SYNC		
- Properties: (No Selection) Tab:** Shows the properties of selected objects.
- Status Bar:** Shows "Ln 12 Col 1 VHDL".





```

1 entity PROCESS_SYNC_TB is -- pusty szkielet projektu symulacji
2 end PROCESS_SYNC_TB;
3
4 architecture behavioural of PROCESS_SYNC_TB is -- cialo architektoniczne projektu
5
6   signal R : bit; -- symulowane wejście kasujące 'R'
7   signal S : bit; -- symulowane wejście ustawiające 'S'
8   signal L : bit; -- symulowane wejście ładujące 'L'
9   signal E : bit; -- symulowane wejście zezwalające 'E'
10  signal C : bit; -- symulowane wejście taktujące 'C'
11  signal D : bit; -- symulowane wejście danej 'D'
12  signal Q : bit; -- obserwowane wyjście danej 'Q'
13
14 begin
15
16   process is
17     begin
18       R <= '1'; wait for 20 ns;
19       R <= '0'; wait for 200 ns;
20     end process;
21
22   process is
23     begin
24       S <= '0'; wait for 300 ns;
25       S <= '1'; wait for 20 ns;
26     end process;
27
28   process is
29     begin
30       L <= '0'; wait for 200 ns;
31       L <= '1'; wait for 50 ns;
32     end process;
33

```

Przykład użycia oprogramowania ISE – projekt: „process_sync”
Plik źródłowy „process_var_TB.vhd” (fragment 1)

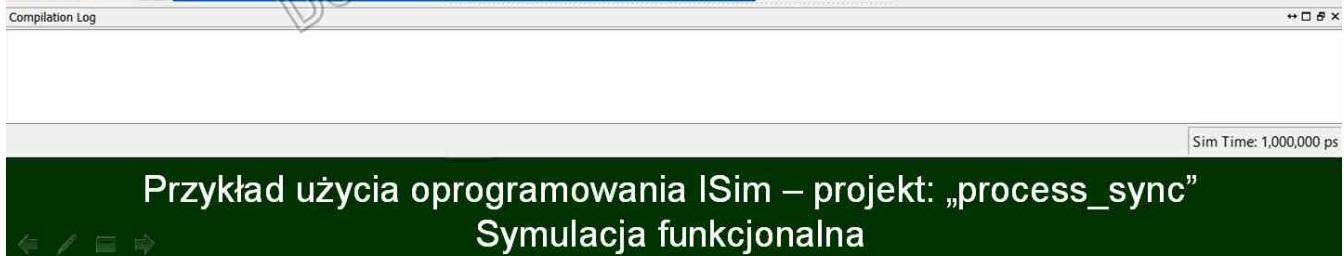
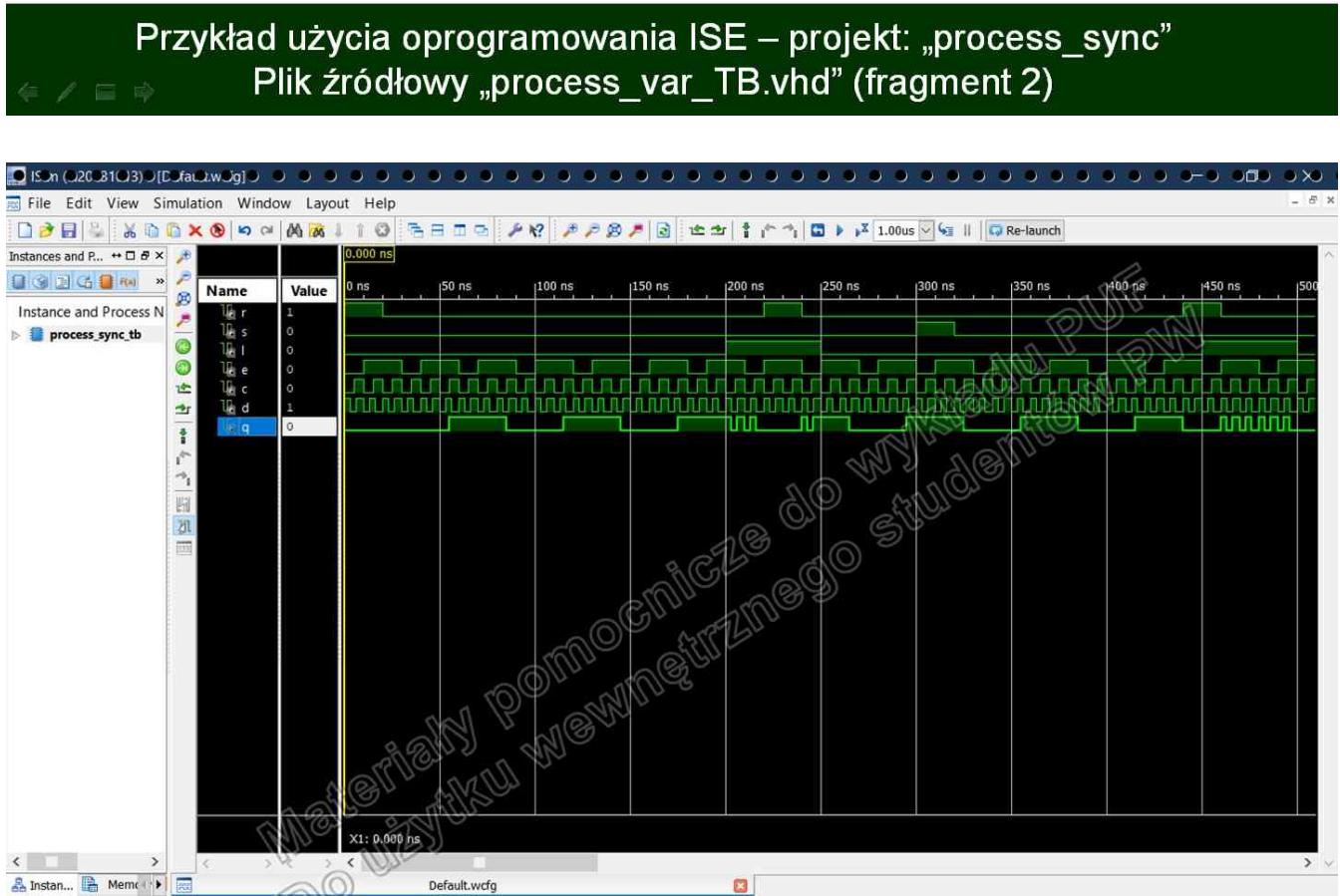
Przykład użycia oprogramowania ISE – projekt: „process_sync”

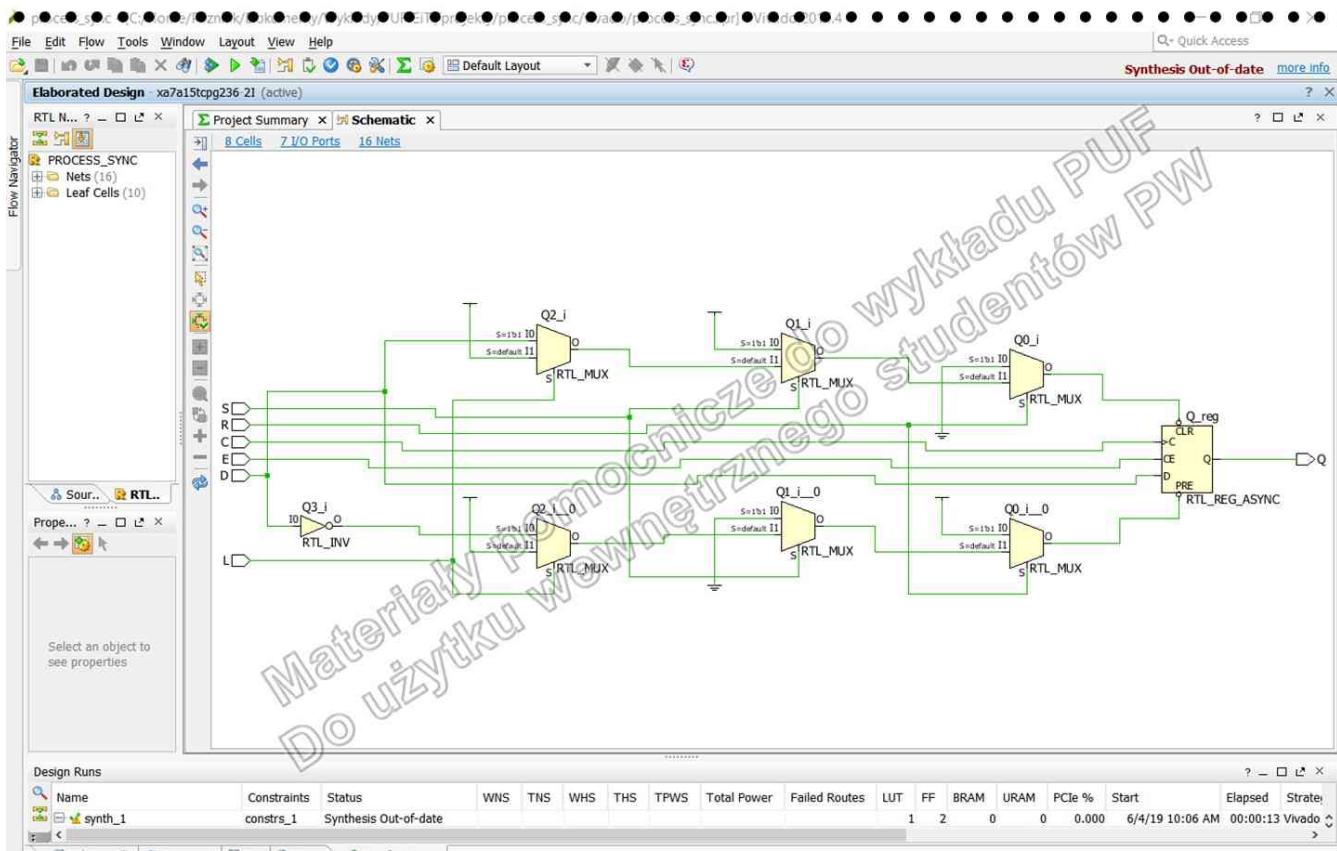
Plik źródłowy „process_var_TB.vhd” (fragment 2)

```

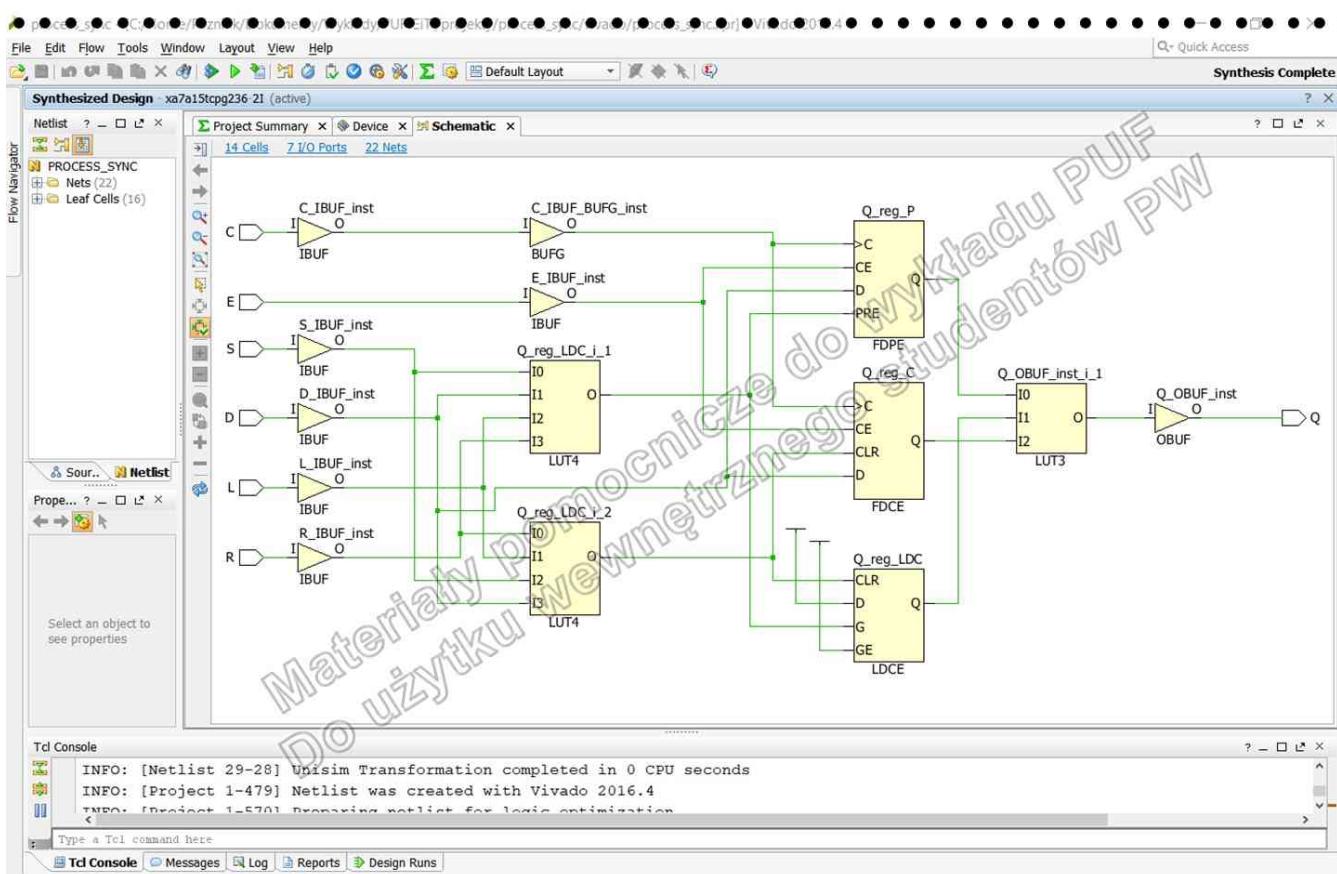
33  process is
34    begin
35      E <= '0'; wait for 10 ns;
36      E <= '1'; wait for 20 ns;
37    end process;
38
39  process is
40    begin
41      C <= '0'; wait for 5 ns;
42      C <= '1'; wait for 5 ns;
43    end process;
44
45  process is
46    begin
47      D <= not(D); wait for 3 ns;
48      D <= not(D); wait for 3 ns;
49      D <= not(D); wait for 4 ns;
50    end process;
51
52
53 process_sync_inst: entity work.PROCESS_SYNC -- instancja projektu 'PROCESS_SYNC'
54   port map (
55     R => R,
56     S => S,
57     L => L,
58     E => E,
59     C => C,
60     D => D,
61     Q => Q
62   );
63
64 end behavioral;
65
66 -- zakonczenie ciala architektonicznego

```

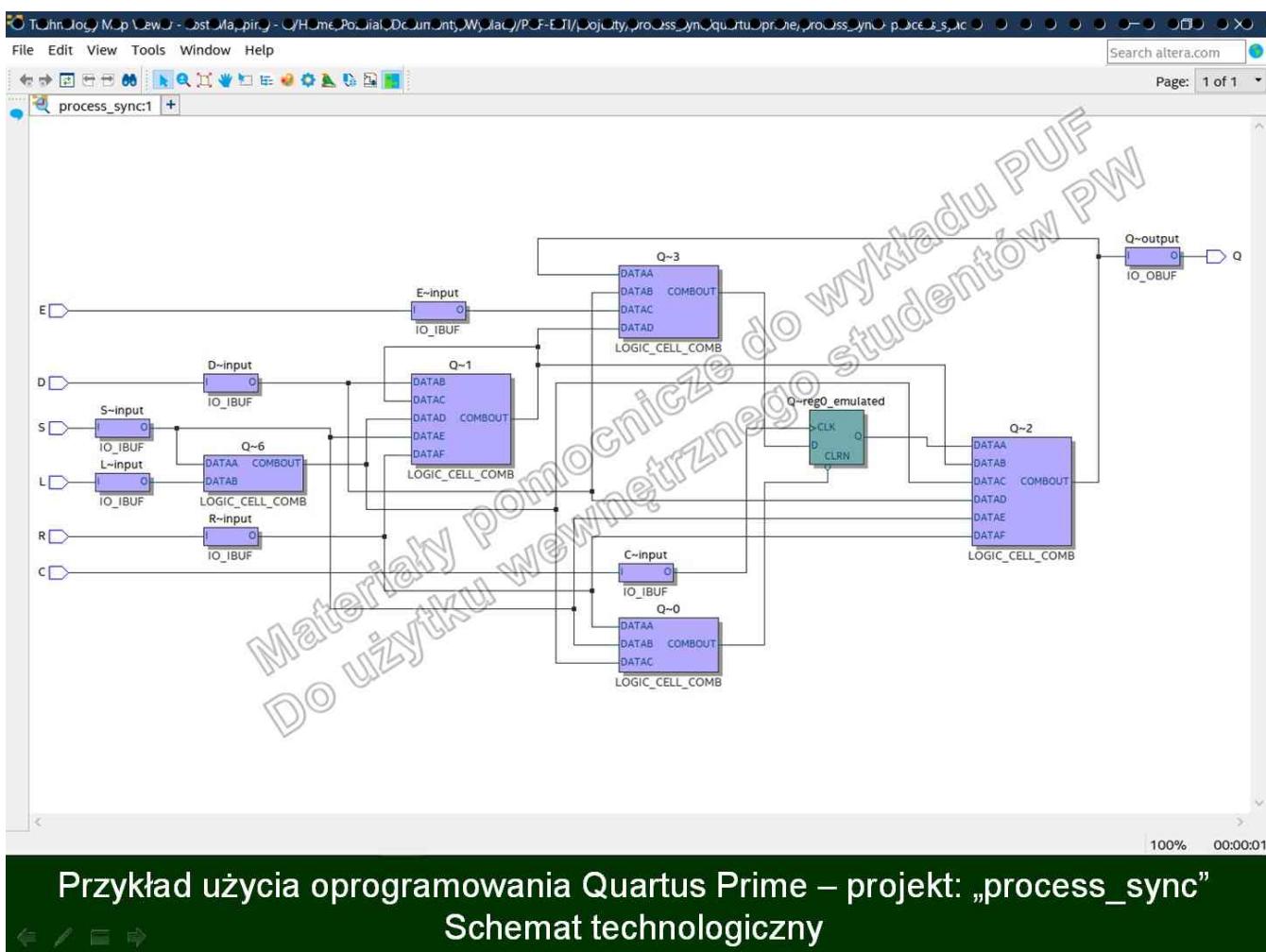
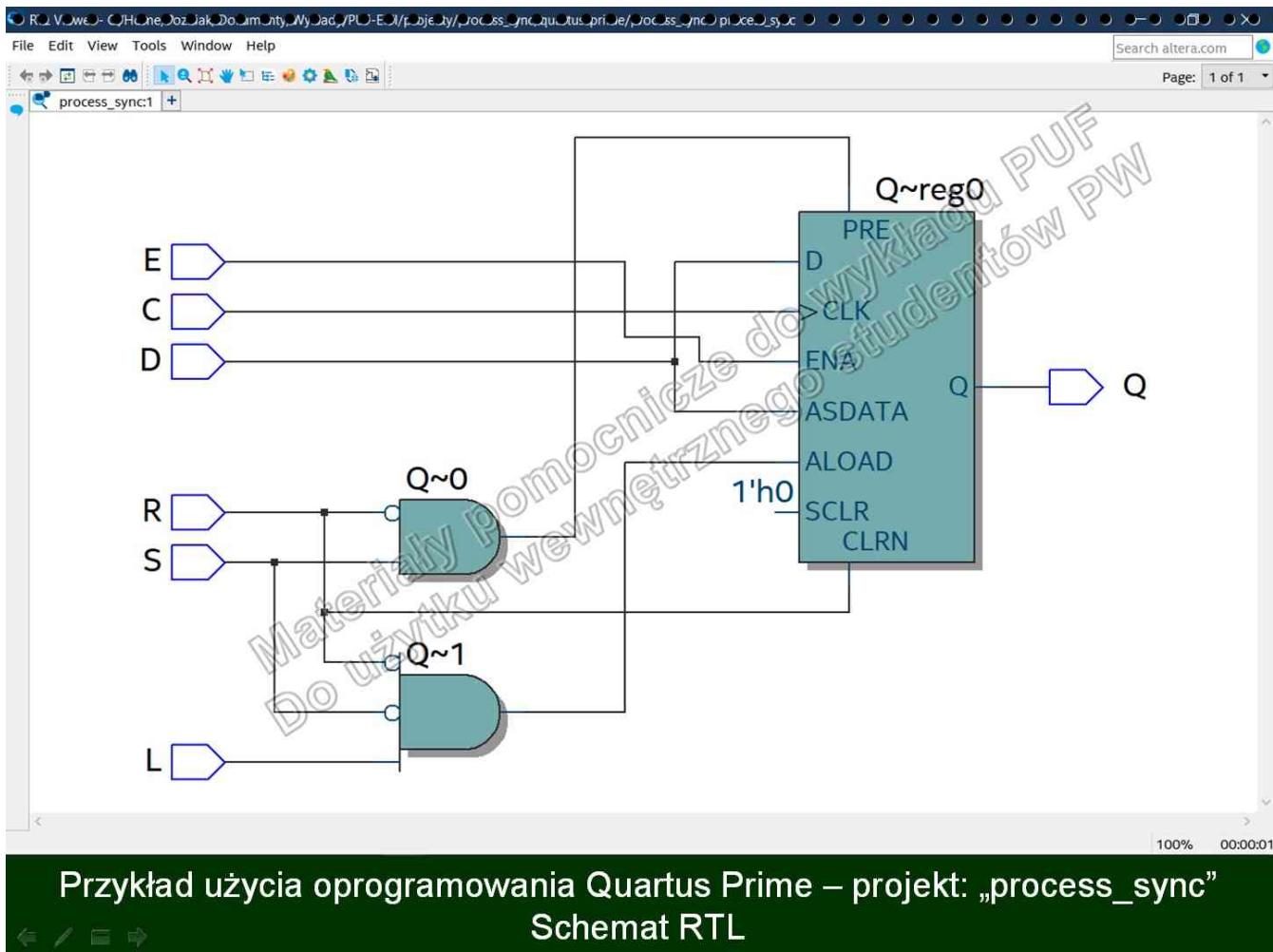


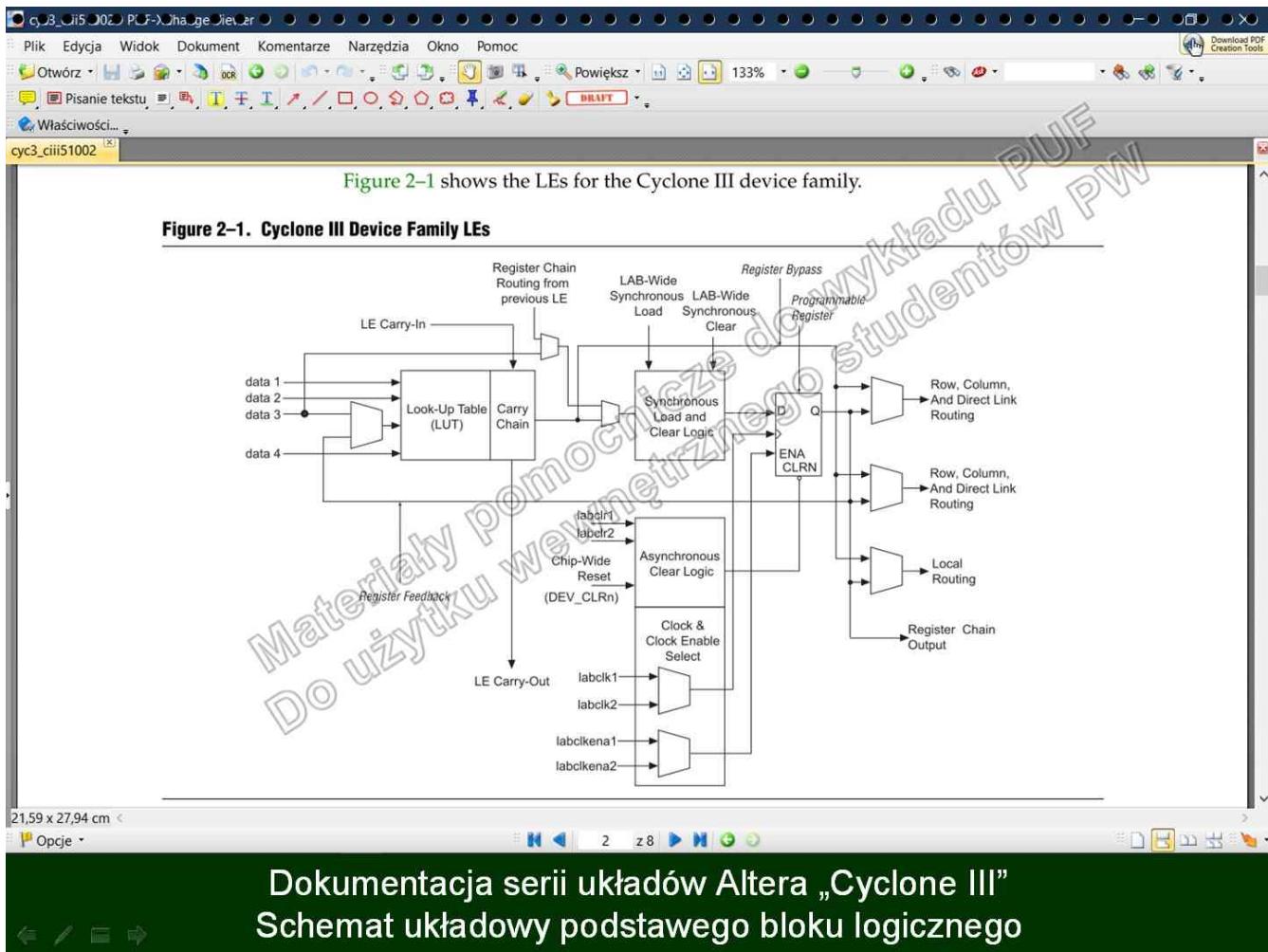


Przykład użycia oprogramowania Vivado – projekt: „process_sync”
Schemat RTL



Przykład użycia oprogramowania Vivado – projekt: „process_sync”
Schemat technologiczny





Dokumentacja serii układów Altera „Cyclone III” Schemat układowy podstawego bloku logicznego

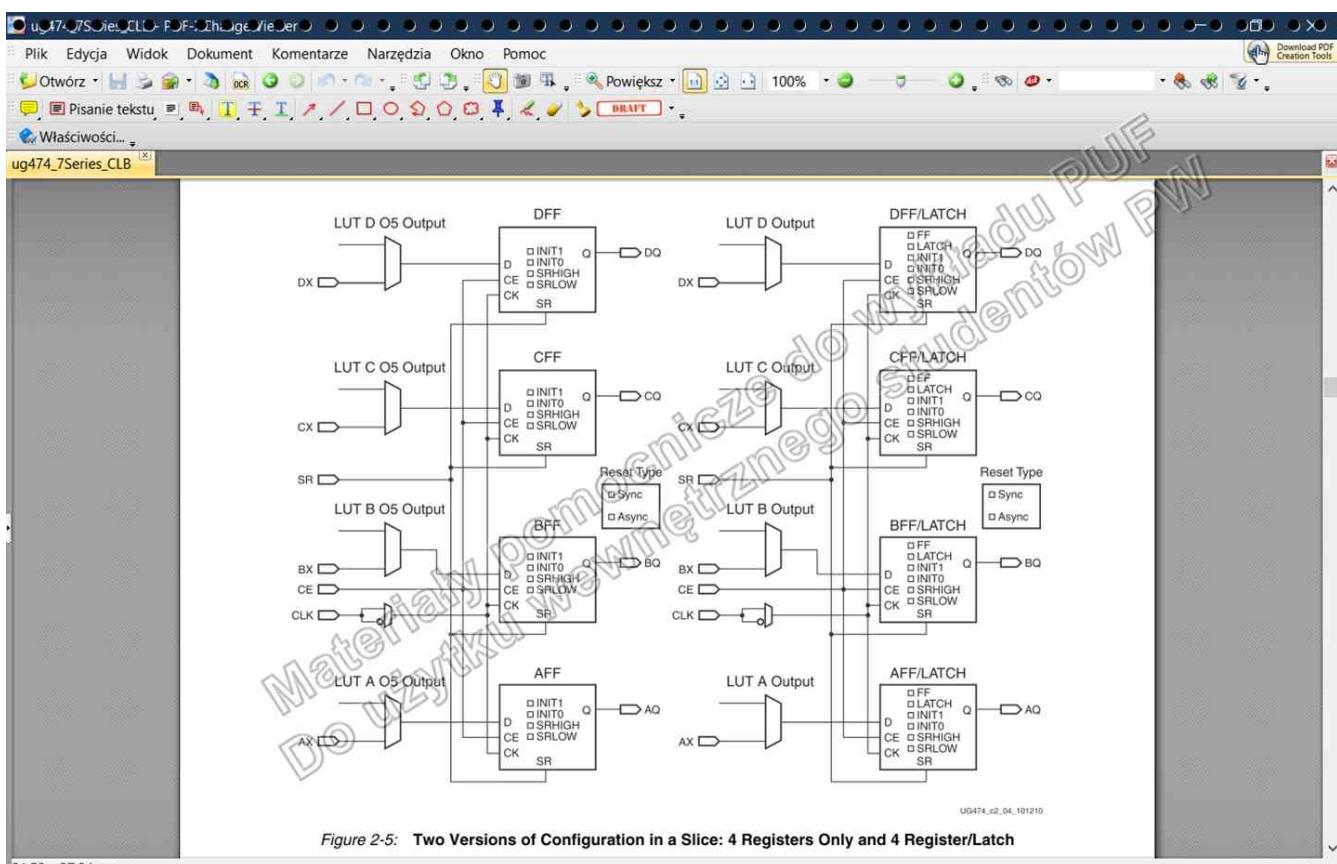
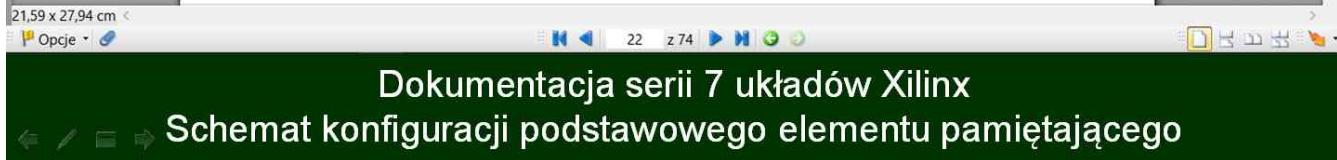


Figure 2–5: Two Versions of Configuration in a Slice: 4 Registers Only and 4 Register/Latch



Podstawowe elementy standardu VHDL

Przykłady instrukcji wyboru warunkowego w procesie

STD

-- biblioteka STD jest włączana automatycznie do projektu

entity PROCESS_DFFE is
port (R : in bit;
C : in bit;
E : in bit;
D : in bit;
Q : out bit
);
end PROCESS_DFFE;

architecture cialo of PROCESS_DFFE is -- deklaracja ciala 'cialo' architektury

begin
process (R, C) is
begin
if (R='1') then
Q<= '0';
elsif (C'event and C='1') then
if (E='1') then
Q <= D;
end if;
end if;
end process;
end architecture cialo;

Nagłówek

Ciało

część wykonawcza

część dekl.

-- deklaracja sprzęgu PROCESS_DFFE
-- deklaracja portu kasującego 'R'
-- deklaracja portu taktującego 'C'
-- deklaracja portu zezwalającego 'E'
-- deklaracja portu wej. danej 'D'
-- deklaracja portu wej. danej 'Q'
-- zakończenie deklaracji listy portów
-- zakończenie deklaracji nagłówka

-- lista czułości procesu
-- część wykonawcza procesu
-- warunek dla sygnału 'R'
-- przypisanie stałej do wyjścia
-- warunek zbocza narastającego 'C'
-- warunek zezwolenia na zapis
-- przypisanie danej do wyjścia
-- zakończenie instrukcji wyboru
-- zakończenie instrukcji wyboru
-- zakończenie procesu
-- zakończenie deklaracji ciala 'cialo'

Pełny opis projektu w języku VHDL

```
File Edit View Project Source Process Tools Window Layout Help
```

Design

View Implement Sim

Hierarchy

process_dffe

xc3s50a-4tq144

PROCESS_DFFE

No Processes Running

Processes: PROCESS_DFFE

Design Summary

process_dffe.vhd

Console

Started : "Launching ISE Text Editor to edit process_dffe.vhd".

Ln 10 Col 1 VHDL

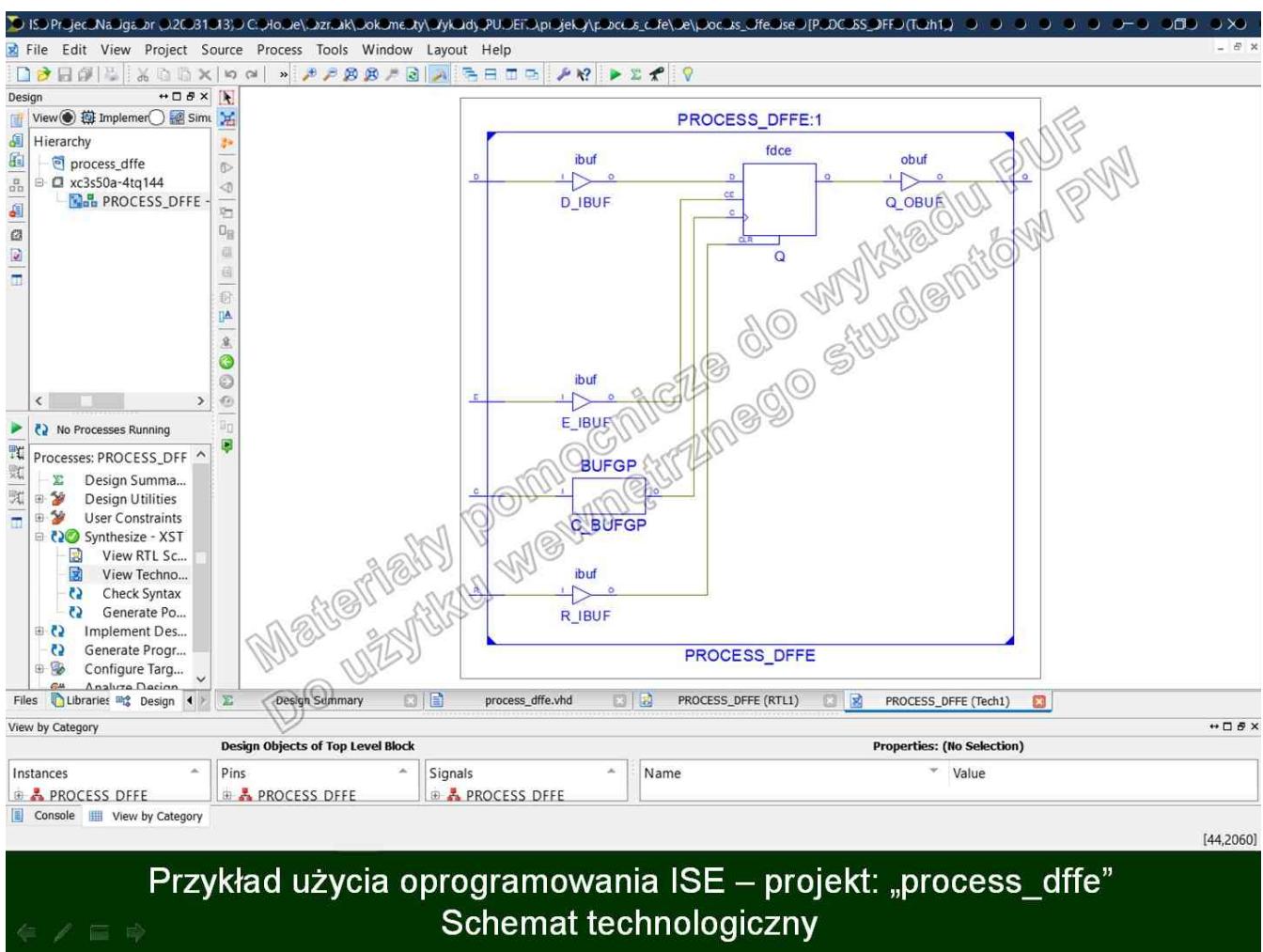
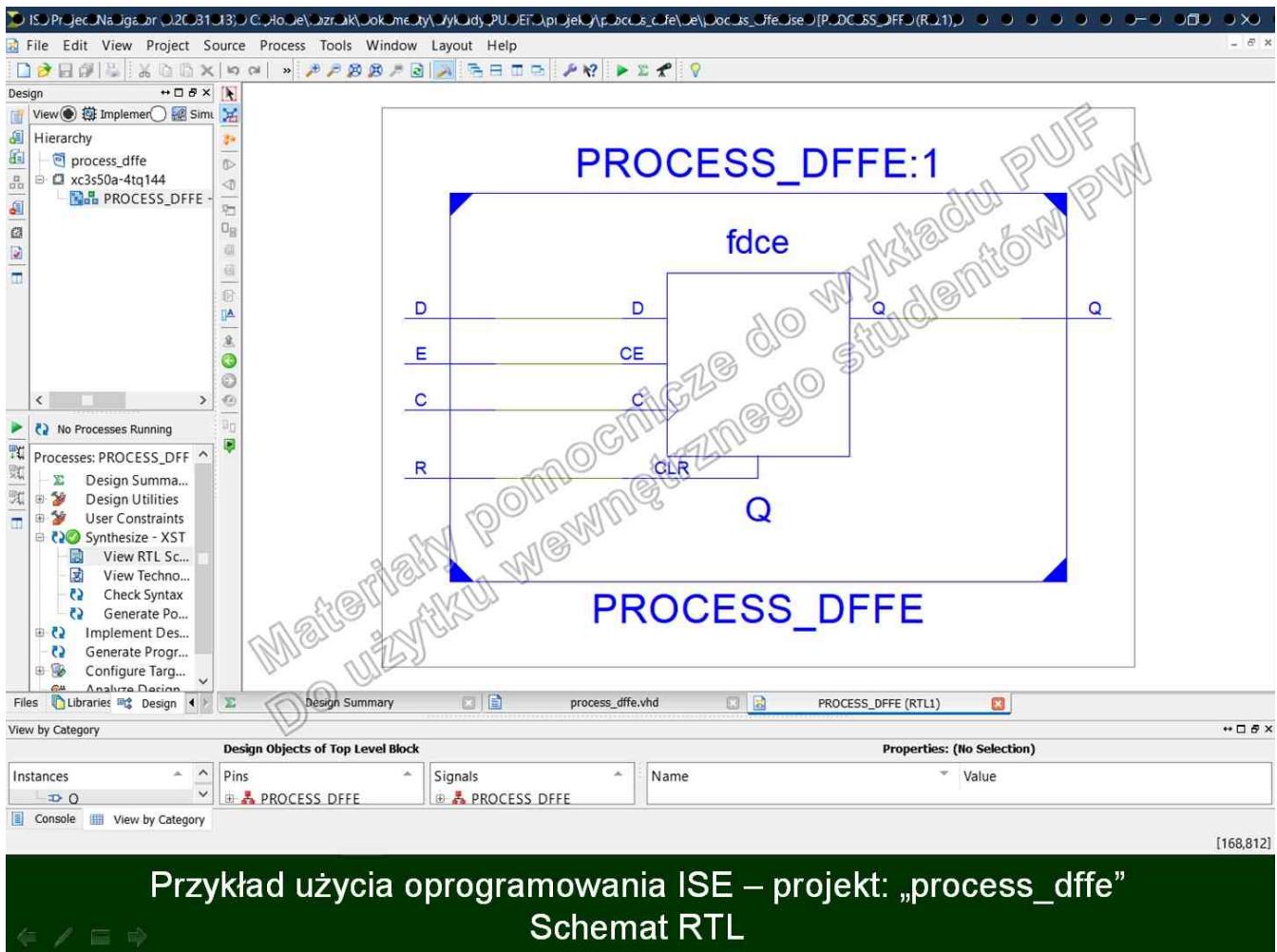
```
1 entity PROCESS_DFFE is  
2 port ( R : in bit;  
3 C : in bit;  
4 E : in bit;  
5 D : in bit;  
6 Q : out bit  
7 );  
8 end PROCESS_DFFE;  
9  
10 architecture cialo of PROCESS_DFFE is  
11 begin  
12 process (R, C) is  
13 begin  
14 if (R='1') then  
15 Q <= '0';  
16 elsif (C'event and C='1') then  
17 if (E='1') then  
18 Q <= D;  
19 end if;  
20 end if;  
21 end process;  
22 end architecture cialo;
```

-- deklaracja sprzęgu PROCESS_DFFE
-- deklaracja portu kasującego 'R'
-- deklaracja portu taktującego 'C'
-- deklaracja portu zezwalającego 'E'
-- deklaracja portu wej. danej 'D'
-- deklaracja portu wej. danej 'Q'
-- zakończenie deklaracji listy portów
-- zakończenie deklaracji nagłówka

-- deklaracja ciala 'cialo' architektury

-- lista czułości procesu
-- część wykonawcza procesu
-- warunek dla sygnału 'R'
-- przypisanie stałej do wyjścia
-- warunek zbocza narastającego 'C'
-- warunek zezwolenia na zapis
-- przypisanie danej do wyjścia
-- zakończenie instrukcji wyboru
-- zakończenie instrukcji wyboru
-- zakończenie procesu
-- zakończenie deklaracji ciala 'cialo'

Przykład użycia oprogramowania ISE – projekt: „process_dffe”
Plik źródłowy „process_dffe.vhd”



```

1 entity PROCESS_DFFE_TB is           -- pusty szkielet projektu symulacji
2 end PROCESS_DFFE_TB;
3
4 architecture behavioural of PROCESS_DFFE_TB is -- cialo architektoniczne projektu
5
6 signal R : bit;                   -- symulowane wejście kasujące 'R'
7 signal C : bit;                   -- symulowane wejście taktujące 'C'
8 signal E : bit;                   -- symulowane wejście zezwalające 'E'
9 signal D : bit;                   -- symulowane wejście danej 'D'
10 signal Q : bit;                  -- obserwowane wyjście danej 'Q'
11
12 begin                           -- początek części wykonawczej architektury
13
14 process is
15 begin
16   R <= '1'; wait for 20 ns;      -- część wykonawcza procesu
17   R <= '0'; wait for 200 ns;     -- przypisanie R='1' i oczekanie 20 ns
18 end process;                    -- przypisanie R='0' i oczekanie 200 ns
19
20 process is
21 begin
22   C <= '0'; wait for 5 ns;      -- część wykonawcza procesu
23   C <= '1'; wait for 5 ns;      -- przypisanie C='0' i oczekanie 5 ns
24 end process;                    -- przypisanie C='1' i oczekanie 5 ns
25
26 process is
27 begin
28   E <= '0'; wait for 10 ns;     -- część wykonawcza procesu
29   E <= '1'; wait for 20 ns;     -- przypisanie E='0' i oczekanie 10 ns
30 end process;                    -- przypisanie E='1' i oczekanie 10 ns
31
32 process is
33 begin
34   D <= not(D); wait for 3 ns;   -- część wykonawcza procesu
35   D <= not(D); wait for 3 ns;   -- przypisanie negacji wartości D i oczekanie 3 ns
36   D <= not(D); wait for 4 ns;   -- przypisanie negacji wartości D i oczekanie 3 ns
37 end process;                    -- przypisanie negacji wartości D i oczekanie 4 ns
38
39 process_dffe_inst: entity work.PROCESS_DFFE(cialo) -- instancja projektu 'PROCESS_DFFE'
40 port map (
41   R => R,
42   C => C,
43   E => E,
44   D => D,
45   Q => Q
46 );
47
48 end behavioural;                -- zakończenie ciała architektonicznego
49

```

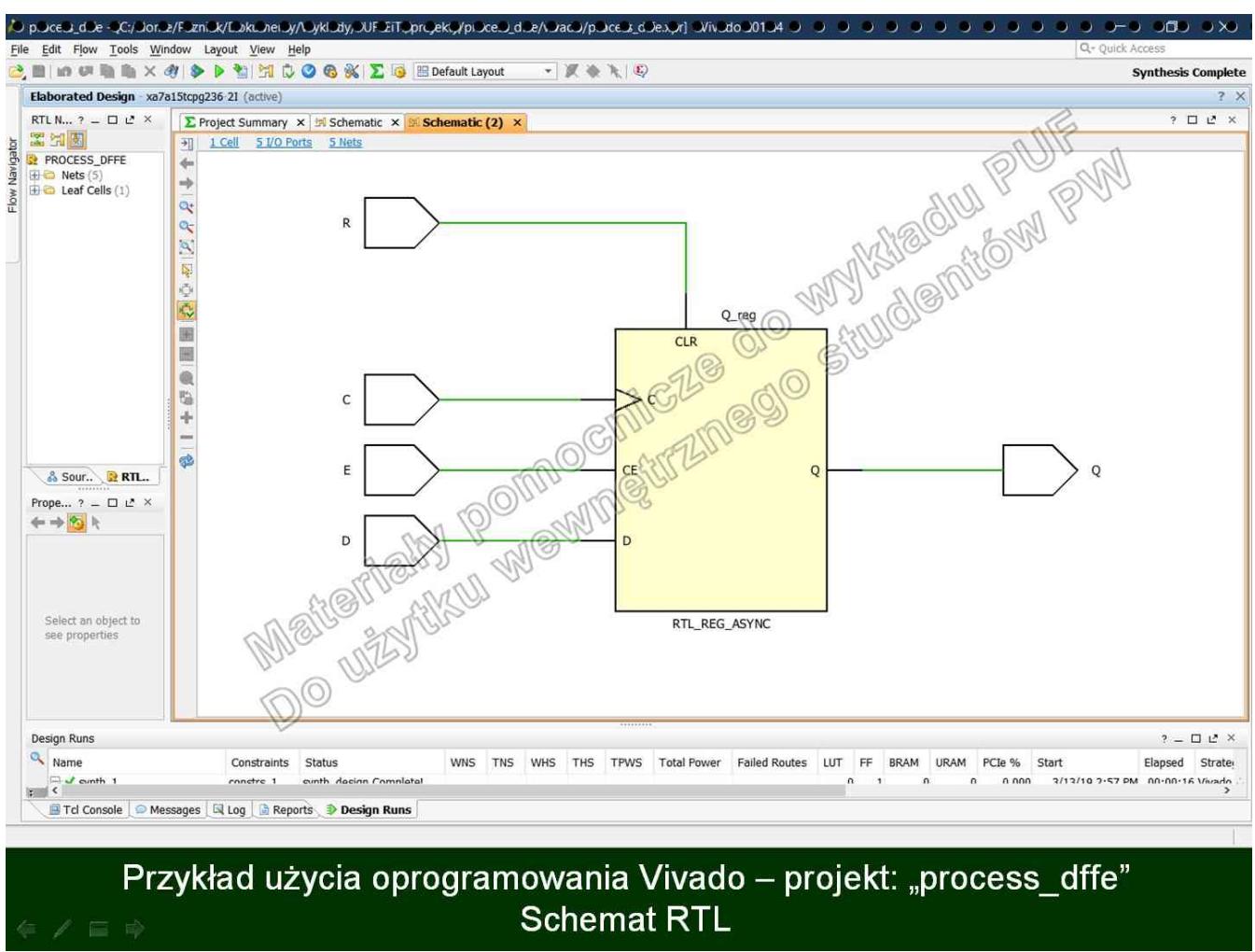
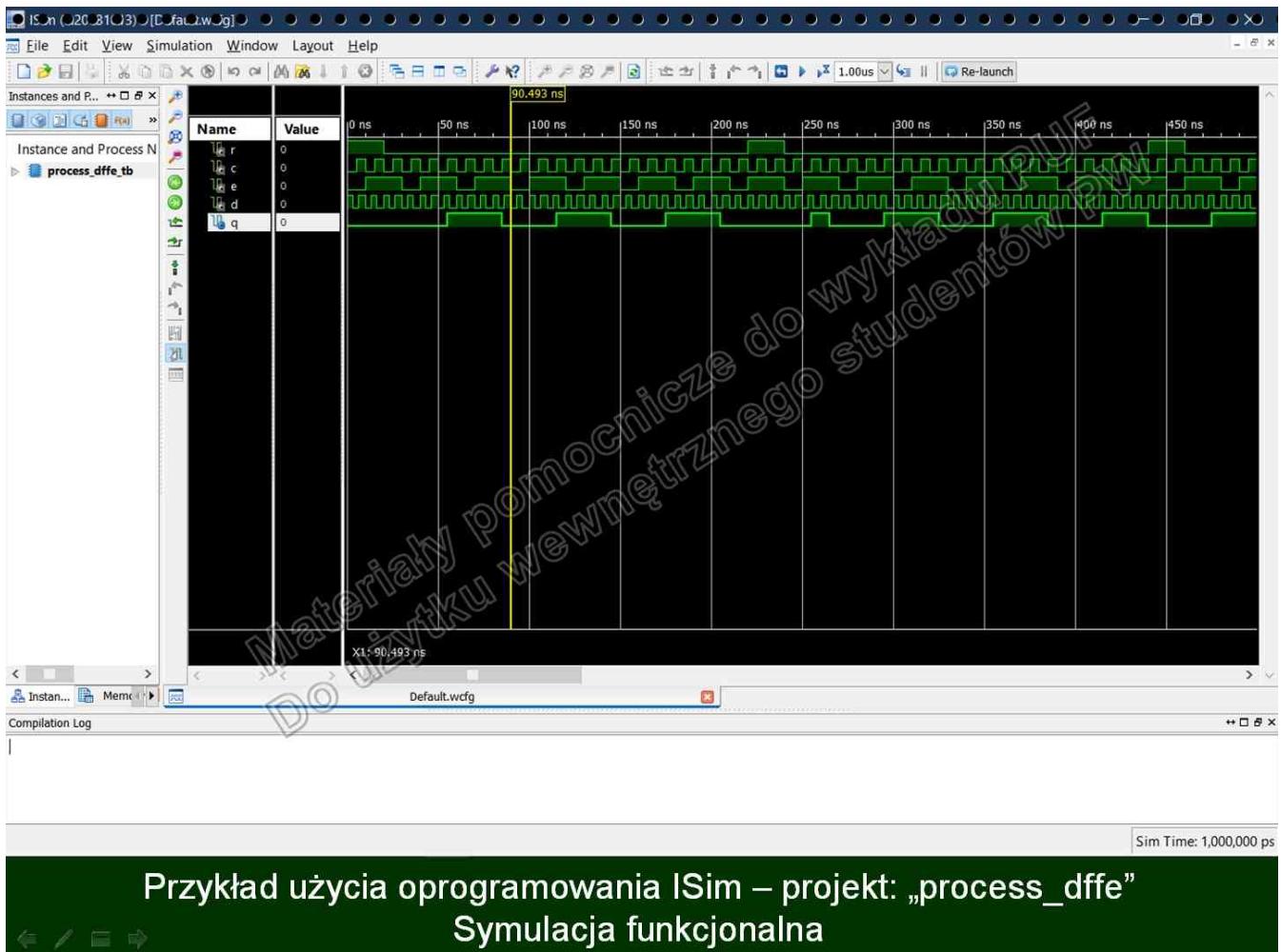
Przykład użycia oprogramowania ISE – projekt: „process_dffe” Plik źródłowy „process_var_TB.vhd” (fragment 1)

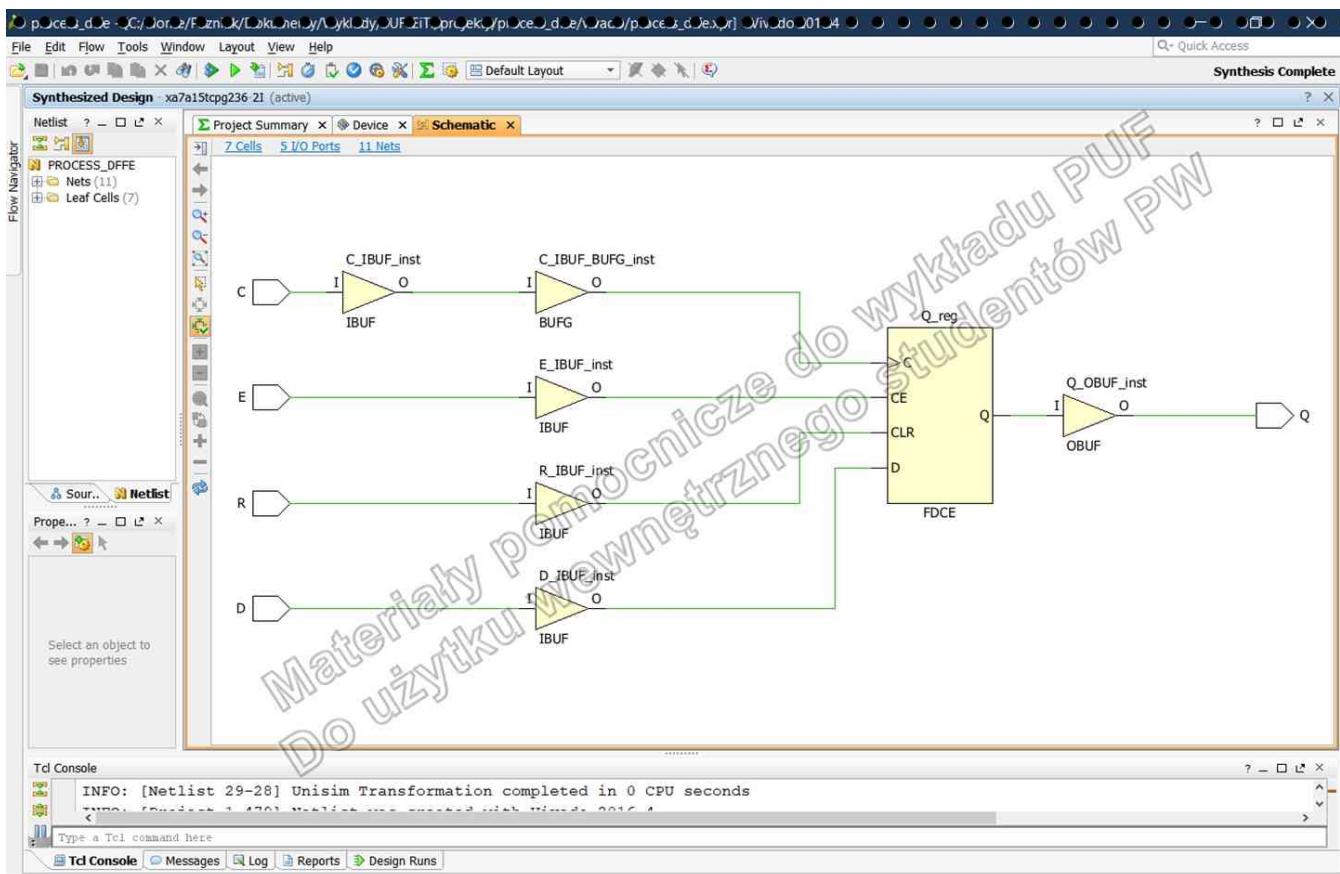
```

18 end process;                      -- zakończenie procesu
19
20 process is
21 begin
22   C <= '0'; wait for 5 ns;        -- część wykonawcza procesu
23   C <= '1'; wait for 5 ns;        -- przypisanie C='0' i oczekanie 5 ns
24 end process;                      -- przypisanie C='1' i oczekanie 5 ns
25
26 process is
27 begin
28   E <= '0'; wait for 10 ns;       -- część wykonawcza procesu
29   E <= '1'; wait for 20 ns;       -- przypisanie E='0' i oczekanie 10 ns
30 end process;                      -- przypisanie E='1' i oczekanie 10 ns
31
32 process is
33 begin
34   D <= not(D); wait for 3 ns;    -- część wykonawcza procesu
35   D <= not(D); wait for 3 ns;    -- przypisanie negacji wartości D i oczekanie 3 ns
36   D <= not(D); wait for 4 ns;    -- przypisanie negacji wartości D i oczekanie 3 ns
37 end process;                      -- przypisanie negacji wartości D i oczekanie 4 ns
38
39 process_dffe_inst: entity work.PROCESS_DFFE(cialo) -- instancja projektu 'PROCESS_DFFE'
40 port map (
41   R => R,
42   C => C,
43   E => E,
44   D => D,
45   Q => Q
46 );
47
48 end behavioural;                  -- zakończenie ciała architektonicznego
49

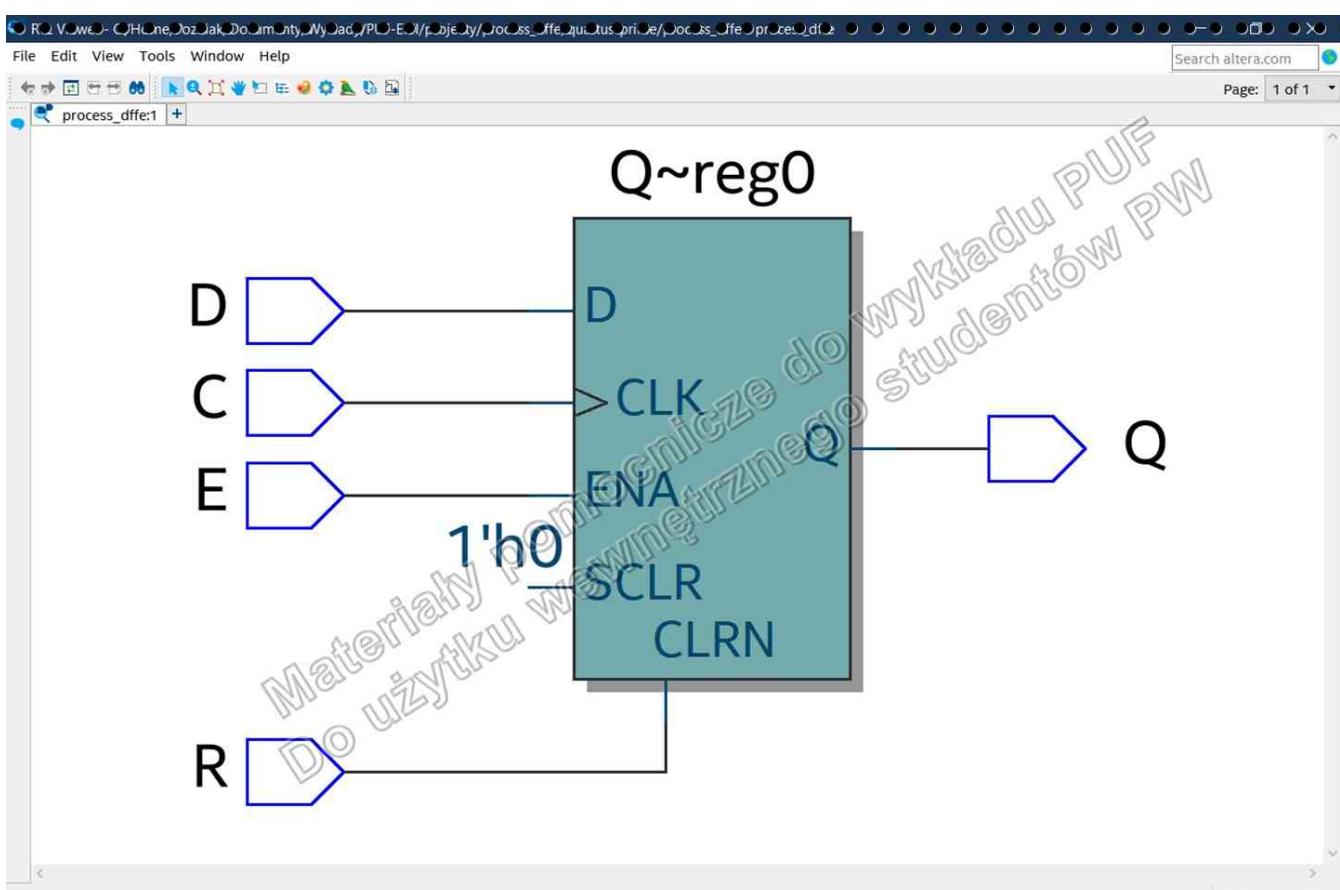
```

Przykład użycia oprogramowania ISE – projekt: „process_dffe” Plik źródłowy „process_var_TB.vhd” (fragment 2)

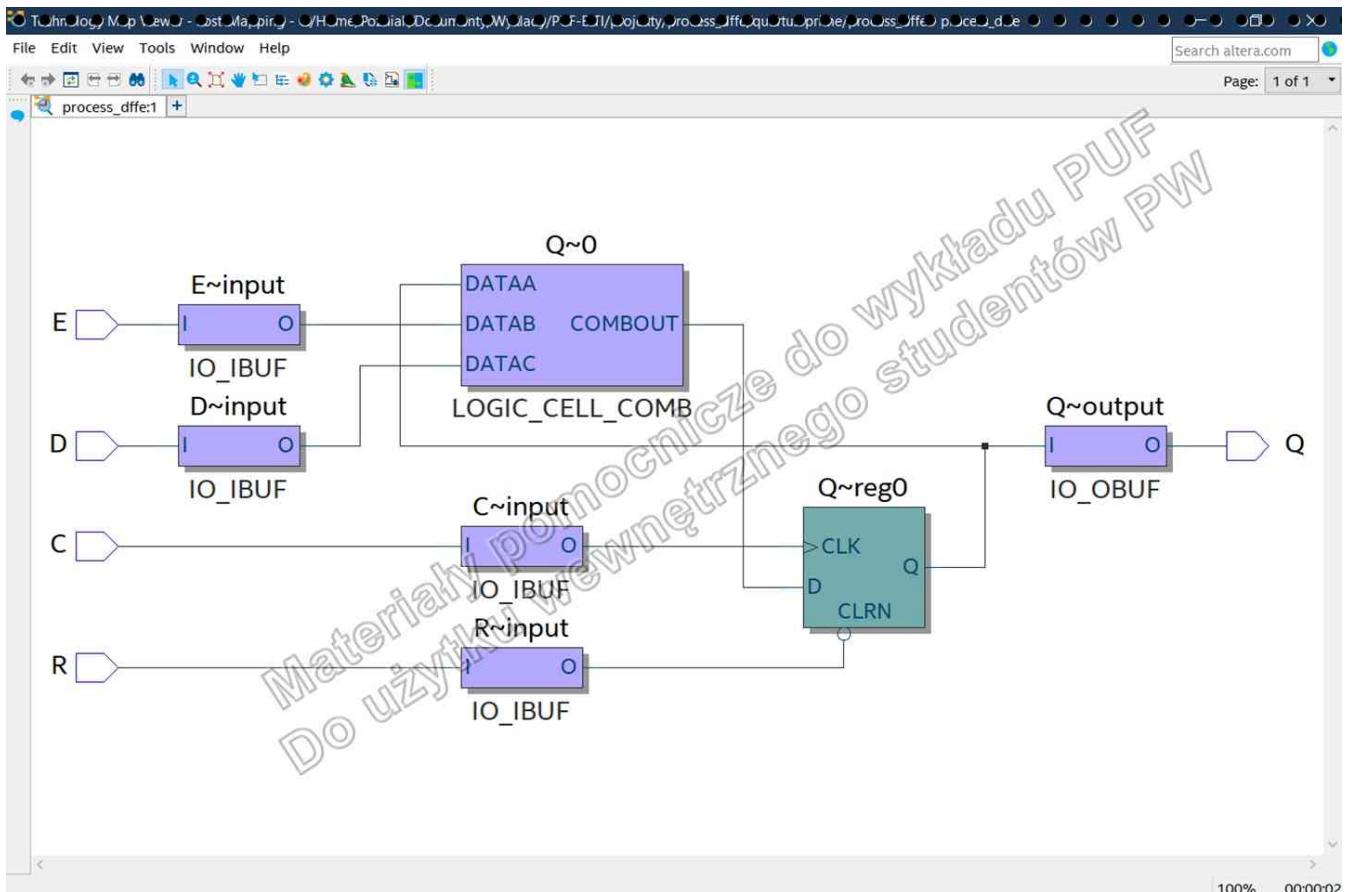




Przykład użycia oprogramowania Vivado – projekt: „process_dffe”
Schemat technologiczny



Przykład użycia oprogramowania Quartus Prime – projekt: „process_dffe”
Schemat RTL



Przykład użycia oprogramowania Quartus Prime – projekt: „process_dffe”
Schemat technologiczny