

Programowanie układów FPGA – wykład II

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

- Wprowadzenie do języka VHDL
 - geneza
 - standardy i rozszerzenia
 - własności i poziomy abstrakcji
- Sprzęg i ciało jednostki projektowej
- Dołączenie jednostki projektowej
- Podstawowa struktura opisu projektu dla FPGA

Wprowadzenie do języka VHDL

Geneza języka VHDL

VHDL to skrót od:

Very High Speed Integrated Circuit Hardware Description Language

- **Projekt VHDL** był częścią programu Departamentu Obrony USA o nazwie VHSIC- jego zadaniem było opracowanie metod projektowania oraz wykorzystanie złożonych i szybkich układów scalonych.
- W 1983 r. zespół złożony z przedstawicieli firm Intermetrics, IBM oraz Texas Instruments rozpoczął pracę nad językiem dla układów VLSI.
- W 1985 opracowano rozwiązanie oparte na języku Ada dla komputerów klasy VA 11/780 i IBM 370.
- W roku 1987 VHDL stał się obowiązującym standardem w dziedzinie języków opisu i projektowania układów VLSI.
- Obecnie jest to jeden z najpopularniejszych języków opisu i projektowania (symulowania, implementacji) układów cyfrowych.

Wprowadzenie do języka VHDL

Rozwój języka VHDL

- Standardy IEEE

(Institute of Electrical and Electronics Engineers, USA):

- IEEE Std 1076-1987 - pierwsza standaryzacja z projektu wojskowego USA
- IEEE Std 1076-1993 - pierwsza wersja poprawiona (obecnie bazowa)
- IEEE Std 1076-2000 - niewielka poprawka, głównie o typy chronione
- IEEE Std 1076-2002 - niewielkie zmiany, głównie dot. portów buforowych
- IEEE Std 1076-2008 - rozwój o nową składnię, operatory, interfejsy

- Rozszerzenie VHDL-1993

o sygnały analogowe i mieszane: VHDL-AMS

- IEEE Std 1076.1-1999 - symulacja układów analogowo-cyfrowych
- IEEE Std 1076.1-2007 - rozszerzenie zgodnie ze standardem VHDL-2002
- IEEE Std 1076.1.1-2011 - pakiety „Multiple Energy Domain Support”

Wprowadzenie do języka VHDL

Wybrane ważne właściwości języka VHDL

- Równoległość

- możliwość opisu przetwarzania równoległego

- Sekwencyjność

- możliwość opisu przetwarzania sekwencyjnego

- Strukturalność

- możliwość wielopoziomowego opisu hierarchicznego

- Reprojektowalność

- możliwość użycia wcześniejszych rozwiązań

- Przenaszalność

- możliwość opisu dla różnych technologii

- Symulowalność

- możliwość symulowania funkcjonalnego i w czasie

- Syntezowalność

- możliwość komplikacji projektu do sprzętu (np. FPGA)

- Samodokumentowalność

- wynikająca z czytelności kodu języka

Wprowadzenie do języka VHDL

Poziomy abstrakcji programowania w języku VHDL

- warstwa topograficzna (ang. layout)
 - specyfikuje połączenia z uwzględnieniem m.in. zależności czasowych i efektów analogowych
- warstwa logiki (ang. logic)
 - zawiera informację o funkcjach, architekturze, technologii i szczegóły zależności czasowych
- warstwa strukturalna (RTL - ang. Register Transfer Level)
 - zawiera opis każdego rejestru oraz logiki między nimi, ale bez zależności czasowych
- warstwa funkcjonalna (ang. behavioural)
 - opisuje funkcje projektu na poziomie abstrakcyjnym, często sparametryzowanym

Podstawowe elementy standardu VHDL

Sprzęg jednostki projektowej

Prosta składnia definicji sprzęgu:

```
entity nazwa_sprzęgu is  
end [entity] [nazwa_sprzęgu];
```

wariant pełny:

```
entity pusta is  
end entity pusta;
```

warianty uproszczone:

```
entity pusta is  
end entity;
```

```
entity pusta is  
end pusta;
```

wariant krótki:

```
entity pusta is  
end;
```

Podstawowe elementy standardu VHDL

Sprzęg jednostki projektowej

Składnia definicji sprzęgu z listą portów:

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

deklaracje wybranych rodzajów portów:

- wejściowy: nazwa_portu [, nazwa_portu] : in deklaracja_typu;
 - wyjściowy: nazwa_portu [, nazwa_portu] : out deklaracja_typu;
 - dwukierunkowy nazwa_portu [, nazwa_portu] : inout deklaracja_typu;



```
Plik Edycja Format Widok Pomoc
-- This is Package STANDARD as defined in the VHDL 2008 Language Reference Manual
-- NOTE: VCOM and VSIM will not work properly if these declarations
-- are modified.

-- Version information:
(#)standard.vhd

package STANDARD is
    type BOOLEAN is (FALSE,TRUE);
    type BIT is ('0','1');
    type CHARACTER is (
        NUL, SOH, STX, ETX, EOT, ENQ, ACK, BEL,
        BS, HT, LF, VT, FF, CR, SO, SI,
        DLE, DC1, DC2, DC3, DC4, NAK, SYN, ETB,
        CAN, EM, SUB, ESC, FSP, GSP, RSP, USP,
        ' ', '! ', '" ', '# ', '$ ', '% ', '& ', '^ ',
        '! ', '! ', '* ', '+ ', ', ', '- ', '.', '/',
        '0 ', '1 ', '2 ', '3 ', '4 ', '5 ', '6 ', '7 ',
        '8 ', '9 ', ': ', '; ', '< ', '=' ', '> ', '? ',
        '@ ', 'A ', 'B ', 'C ', 'D ', 'E ', 'F ', 'G ',
        'H ', 'I ', 'J ', 'K ', 'L ', 'M ', 'N ', 'O ',
        'P ', 'Q ', 'R ', 'S ', 'T ', 'U ', 'V ', 'W ',
        'X ', 'Y ', 'Z ', '[ ', '\\ ', ']' ', '^ ', '_ ',
        '' ', 'a ', 'b ', 'c ', 'd ', 'e ', 'f ', 'g ',
        'h ', 'i ', 'j ', 'k ', 'l ', 'm ', 'n ', 'o ',
        'p ', 'q ', 'r ', 's ', 't ', 'u ', 'v ', 'w ',
        'x ', 'y ', 'z ', '( ', ') ', ',', '^ ', DEL,
        C128, C129, C130, C131, C132, C133, C134, C135,
        C136, C137, C138, C139, C140, C141, C142, C143,
        C144, C145, C146, C147, C148, C149, C150, C151,
        C152, C153, C154, C155, C156, C157, C158, C159,
        -- the character code for 160 is there (NBSP),
        -- but prints as no char
        ' ', '!', 'ç ', '± ', '° ', '¤ ', '%', '€ ',
        '¤ ', '® ', '« ', '» ', '¬ ', '¬ ', '® ',
        '° ', '± ', '° ', '° ', '° ', '° ', '° ',
        '° ', '° ', '° ', '° ', '° ', '° ', '° ',
        'À ', 'Á ', 'Â ', 'Ã ', 'Ä ', 'È ', 'É ',
        'Í ', 'Ó ', 'Ô ', 'Õ ', 'Ö ', 'Ñ ', 'Ñ ',
        'Ø ', 'Ù ', 'Ú ', 'Ô ', 'Û ', 'Ý ', 'Þ ',
        'Þ ', 'à ', 'á ', 'â ', 'ã ', 'ä ', 'è ',
        'é ', 'í ', 'ó ', 'ô ', 'õ ', 'ö ', 'ñ ',
        'ø ', 'ù ', 'ú ', 'ô ', 'û ', 'ý ', 'þ ')
```

```

Plik Edycja Format Widok Pomoc
'D', 'Ñ', 'Ó', 'ó', 'ô', 'ö', 'ö', 'ó', 'x',
'Ø', 'Ù', 'Ù', 'Ù', 'Ù', 'Ù', 'Ù', 'Ù', 'Ù',
'â', 'â', 'â', 'â', 'â', 'â', 'â', 'â', 'ç',
'è', 'é', 'é', 'é', 'í', 'í', 'í', 'í', 'í',
'ô', 'ñ', 'ó', 'ó', 'ô', 'ö', 'ö', 'ó', 'í',
'ø', 'ù', 'ú', 'ú', 'ú', 'Ù', 'Ù', 'Ù', 'Ù',
type SEVERITY_LEVEL is (NOTE, WARNING, ERROR, FAILURE);
type INTEGER is range -2147483648 to 2147483647;
type REAL is range -1.0E308 to 1.0E308;
type TIME is range -2147483647 to 2147483647
    units
        fs;
        ps = 1000 fs;
        ns = 1000 ps;
        us = 1000 ns;
        ms = 1000 us;
        sec = 1000 ms;
        min = 60 sec;
        hr = 60 min;
    end units;
type domain_type is (quiescent_domain, time_domain, frequency,
signal domain; domain_type:=quiescent_domain;
subtype DELAY_LENGTH is TIME range 0 fs to TIME'HIGH;
impure function NOW return DELAY_LENGTH;
impure function now return real;
-- TODO AMS -- function frequency return real;
subtype NATURAL is INTEGER range 0 to INTEGER'HIGH;
subtype POSITIVE is INTEGER range 1 to INTEGER'HIGH;
type STRING is array (POSITIVE range <>) of CHARACTER;
type BOOLEAN_VECTOR is array (NATURAL range <>) of BOOLEAN;
type BIT_VECTOR is array (NATURAL range <>) of BIT;
type INTEGER_VECTOR is array (NATURAL range <>) of INTEGER;
type REAL_VECTOR is array (NATURAL range <>) of REAL;
type TIME_VECTOR is array (NATURAL range <>) of TIME;
type FILE_OPEN_KIND is (
    READ_MODE,
    WRITE_MODE,
    APPEND_MODE);
type FILE_OPEN_STATUS is (
    OPEN_OK,
    STATUS_ERROR,
    NAME_ERROR,
    MODE_ERROR);
attribute FOREIGN : STRING;
end STANDARD;

```

Biblioteka „standard.vhg”

Podstawowe elementy standardu VHDL

Sprzęg jednostki projektowej

Składnia definicji sprzęgu z lista portów:

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

deklaracje wybranych rodzajów portów:

- wejściowy: nazwa_portu [, nazwa_portu] : in deklaracja_typu;
 - wyjściowy: nazwa_portu [, nazwa_portu] : out deklaracja_typu;
 - dwukierunkowy: nazwa_portu [, nazwa_portu] : inout deklaracja_typu;
- ```
entity porty is
 port (wejscie1 , wejście2 : in bit ;
 wyjście : out boolean) ;
end entity porty ;
```

## Podstawowe elementy standardu VHDL

### Ciało jednostki projektowej

Prosta składnia definicji ciała:

```
architecture nazwa_ciała of nazwa_sprzęgu is
 begin
end [architecture] [nazwa_ciała] ;
```

Dla pojedynczego sprzęgu można zdefiniować wiele architektur

wariant pełny:

```
architecture ciało of porty is
begin
end architecture ciało ;
```

wariant krótki:

```
architecture ciało of porty is
begin
end ;
```

warianty uproszczone:

```
architecture ciało of porty is
begin
end architecture ;
```

```
architecture ciało of porty is
begin
end ciało ;
```

# Podstawowe elementy standardu VHDL

## Ciało jednostki projektowej

Pełna składnia definicji ciała:

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

przykład ciała z częścią wykonawczą:

```
architecture cialo of porty is
begin
 wyjście <= wejście1 nor wejście2 ;
end architecture cialo ;
```



The screenshot shows a software application window displaying the IEEE Std 1076-1993 standard. The title bar reads "IEEE Std 1076-1993". The menu bar includes "Plik", "Edycja", "Widok", "Dokument", "Komentarze", "Narzędzia", "Okno", and "Pomoc". The toolbar contains various icons for file operations like Open, Save, Print, and zoom. A sidebar on the left lists "Otwórz", "Pisanie tekstu", and "Właściwości...". The main content area is titled "LANGUAGE REFERENCE MANUAL" and "Std 1076-1993". It contains sections such as "14.2 Package STANDARD" and provides details about predefined types, functions, and operators for the BOOLEAN type. The text is in English, with some parts in Polish. The bottom of the screen shows the software's status bar with "21,59 x 27,94 cm", "Opcje", and navigation icons.

Standard IEEE Std 1076-1993 (fragment)

## Podstawowe elementy standardu VHDL

### Ciało jednostki projektowej

Pełna składnia definicji ciała:

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

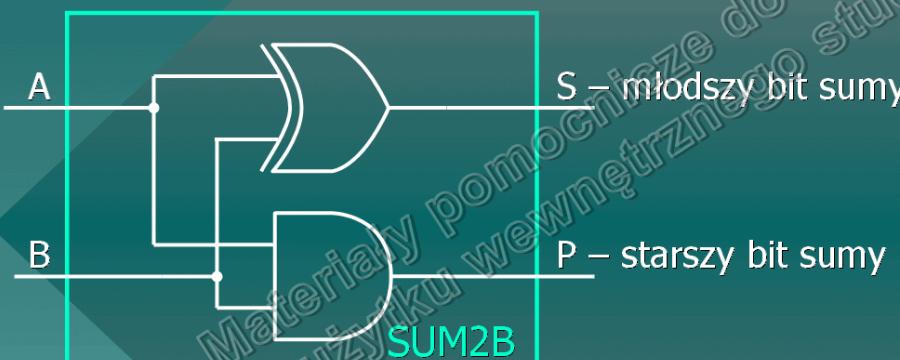
przykład ciała z częścią wykonawczą:

```
architecture cialo of porty is
begin
 wyjście <= wejście1 nor wejście2 ;
end architecture cialo ;
```



## Podstawowe elementy standardu VHDL

### Przykład realizacji sumatora – sumator 2-bitowy



S – młodszy bit sumy

P – starszy bit sumy

| A | B | S | P |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

tabela prawdy

sumatora 2-bitowego

# Podstawowe elementy standardu VHDL

## Przykład realizacji sumatora – sumator 2-bitowy

Projekt „SUM2B”

STD

Nagłówek

Ciało

część wykonawcza

część deklar.

```
-- biblioteka STD jest włączana automatycznie do projektu

entity SUM2B is
 port (A : in bit;
 B : in bit;
 S : out bit;
 P : out bit
);
end SUM2B;

architecture cialo of SUM2B is
begin
 S <= A xor B;
 P <= A and B;
end architecture cialo;
```

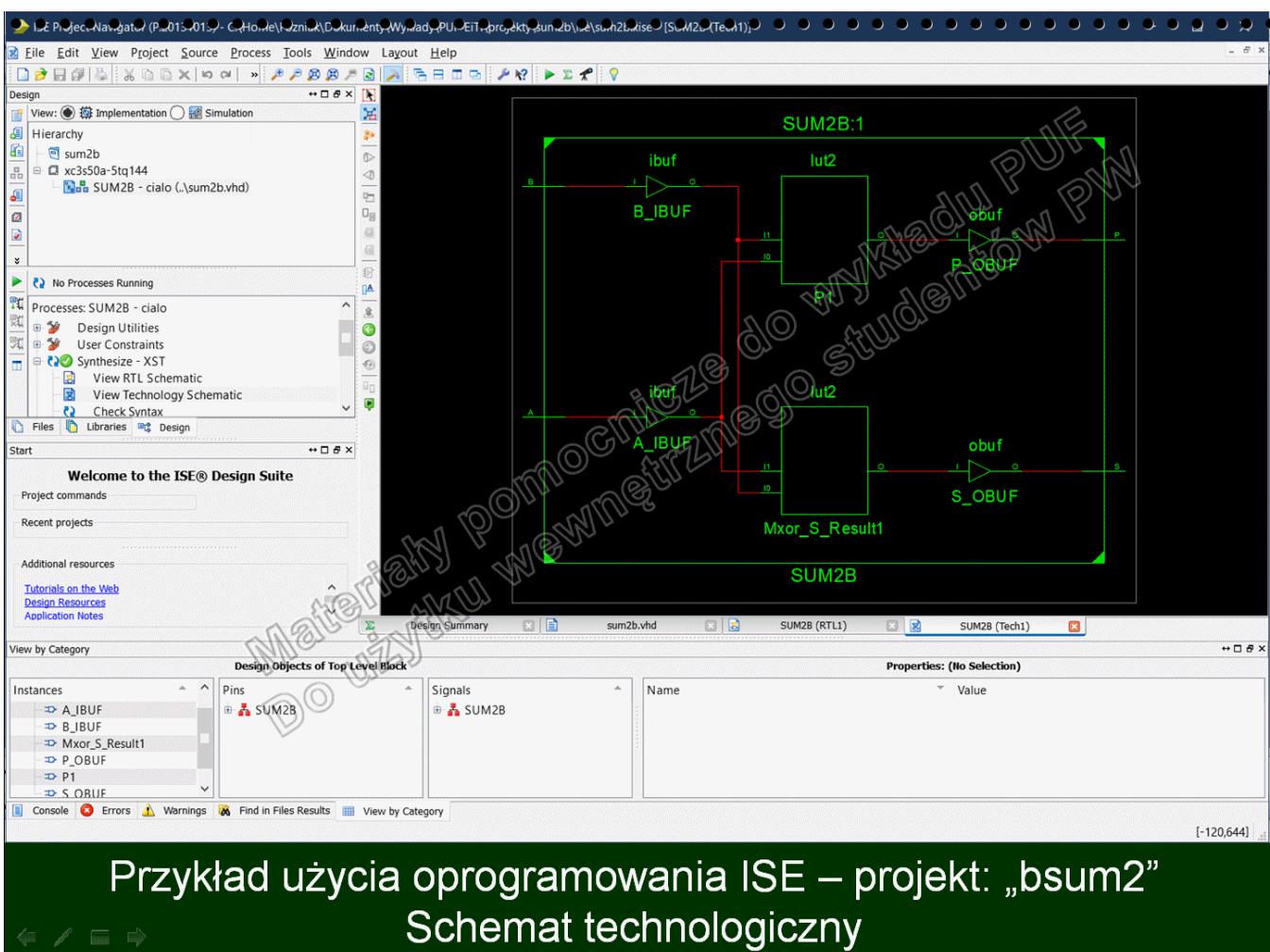
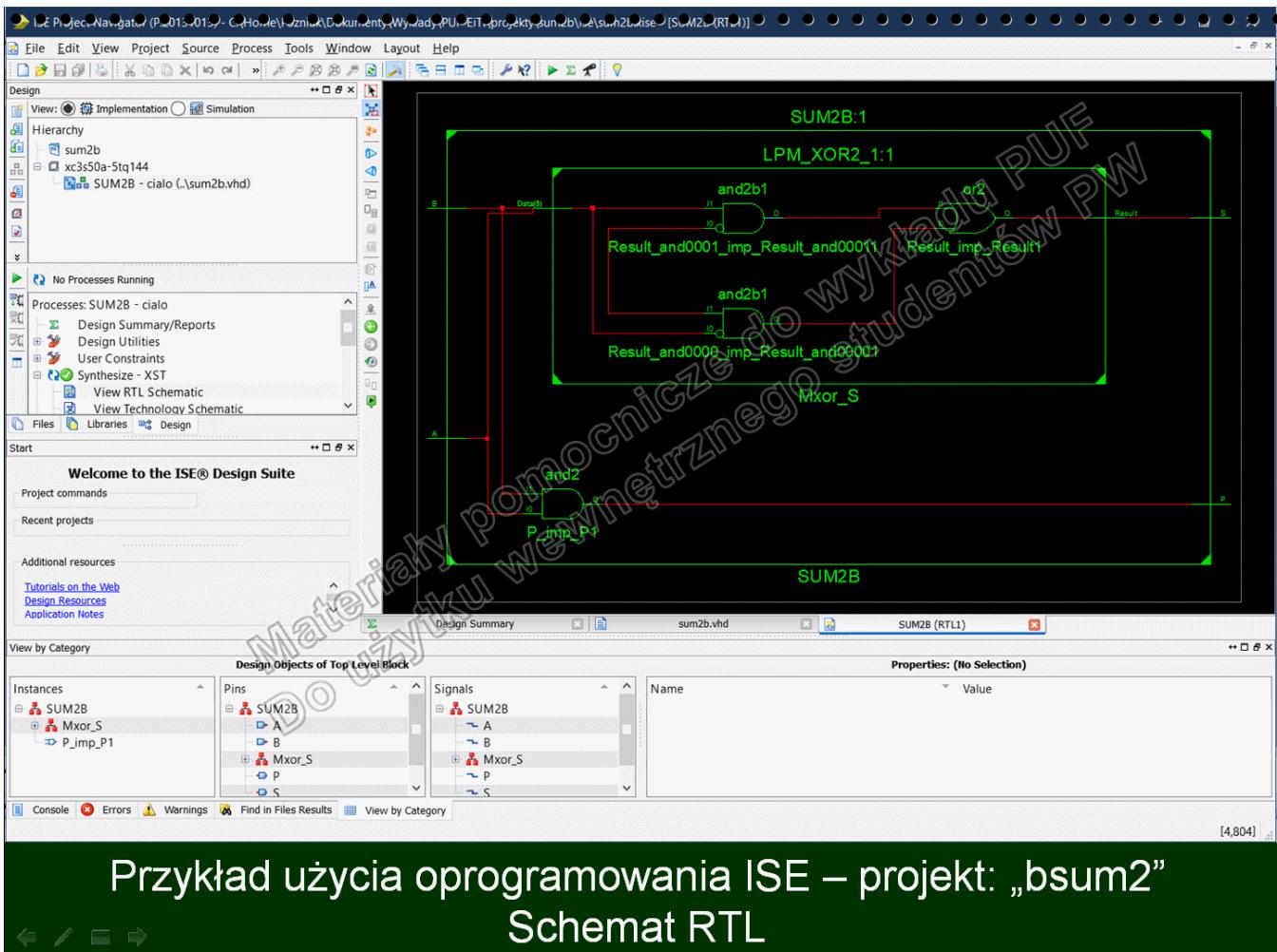
-- deklaracja sprzągu 'SUM2B'  
-- deklaracja portu wejściowego 'A'  
-- deklaracja portu wejściowego 'B'  
-- deklaracja portu wyjściowego 'S'  
-- deklaracja portu wyjściowego 'P'  
-- zakończenie deklaracji listy portów  
-- zakończenie deklaracji nagłówka  
  
-- deklaracja ciała 'ciale' architektury  
  
-- początek części wykonawczej  
  
-- wyznaczenie młodszego bitu sumy  
-- wyznaczenie starszego bitu sumy  
-- (przeniesienie)  
  
-- zakończenie deklaracji ciała 'ciale'

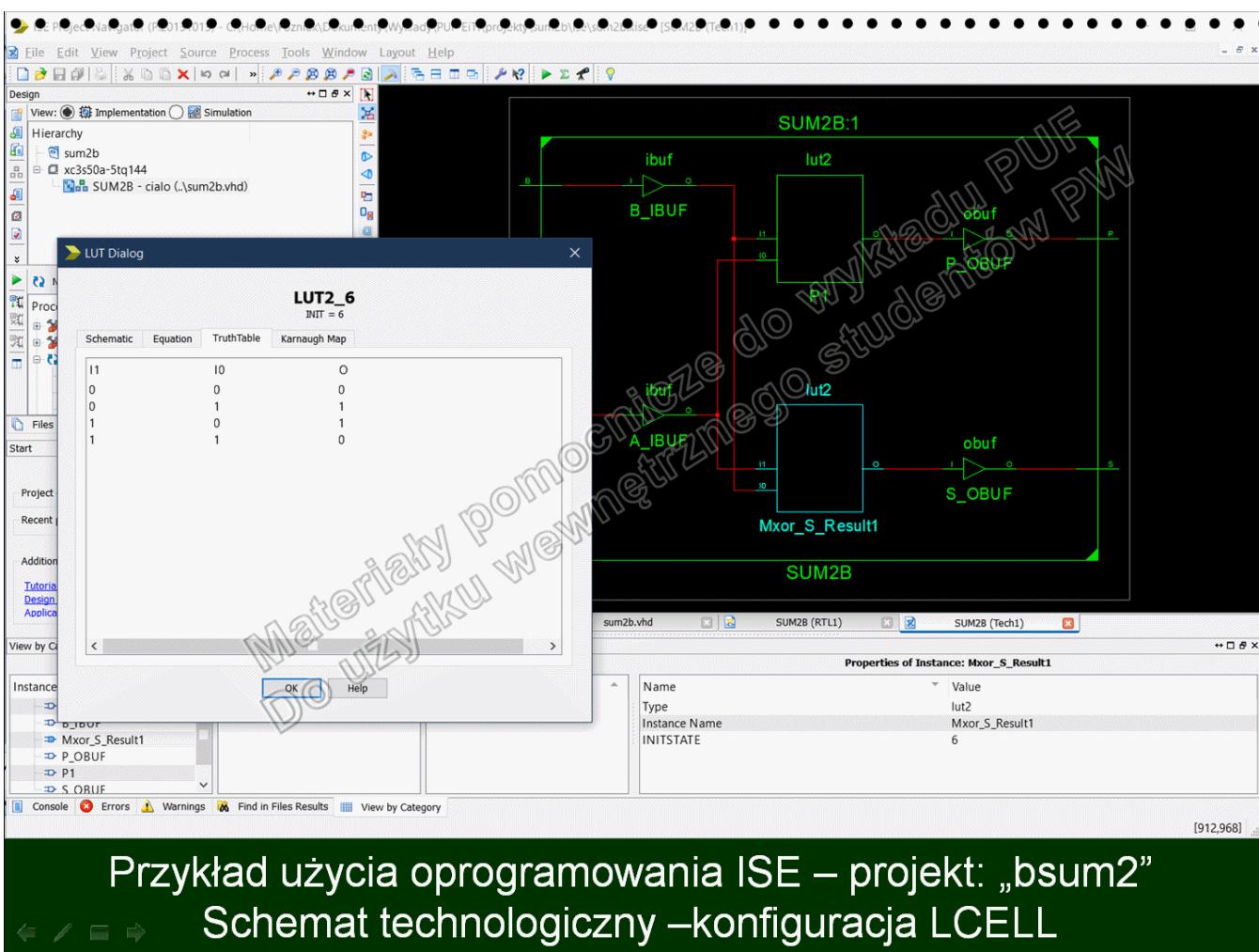
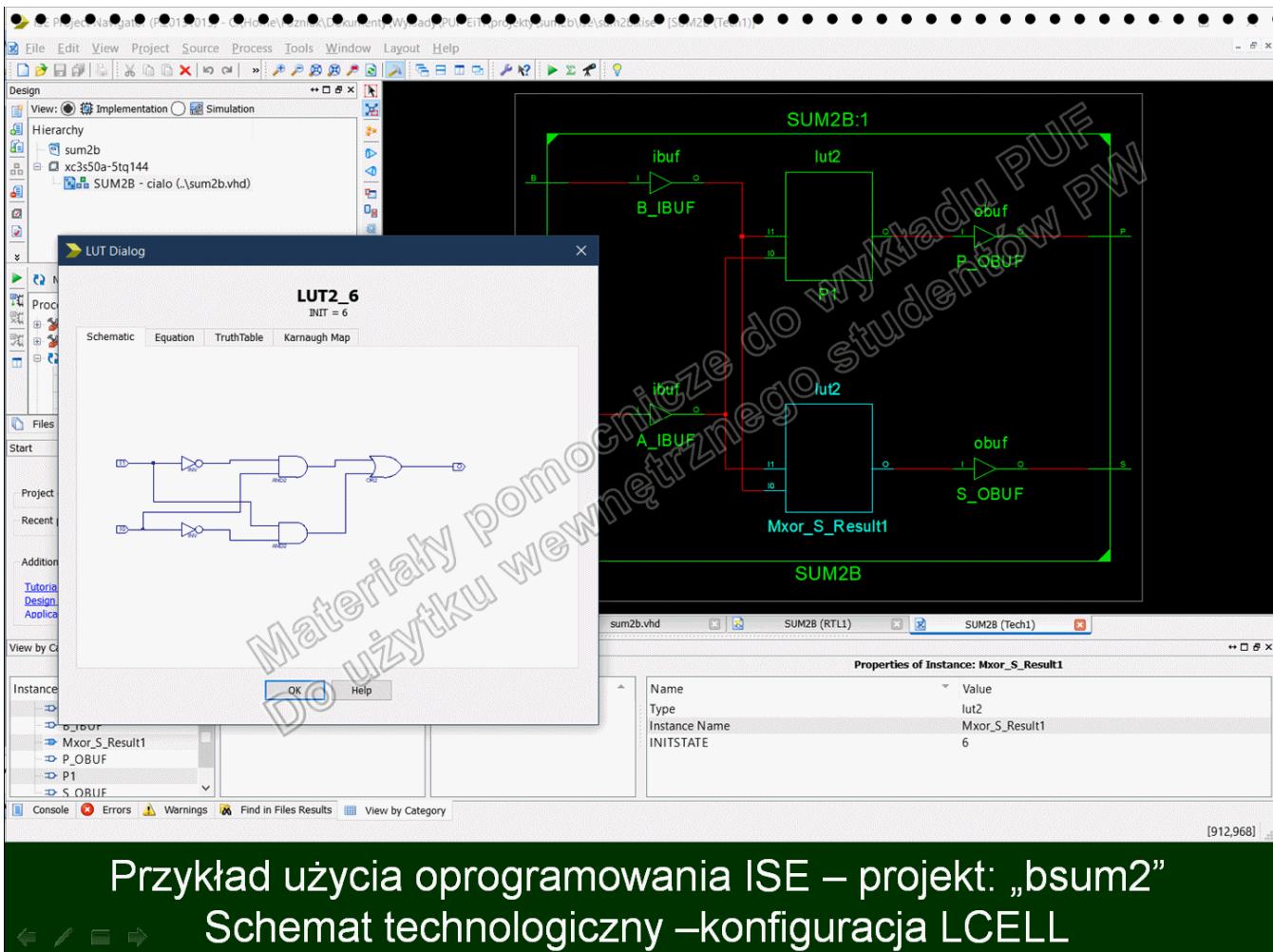
Pełny opis projektu w języku VHDL

The screenshot shows the ISE Design Suite interface with the following details:

- Project Navigator:** Shows the project "sum2b" with a sub-project "xc3s50a-5tq144". The file "SUM2B - cialo (.vhd)" is selected.
- Source Editor:** Displays the VHDL code for the "SUM2B" entity and its "ciale" architecture. The code includes declarations for ports A, B, S, and P, and an architecture body with a begin block containing assignments for S and P based on A and B.
- Design Objects Table:** Shows the hierarchy of design objects for the top-level block "SUM2B". It lists instances like "SUM2B", "Mxor\_S", and "P\_imp\_P1", their pins (A, B, P, S), and signals (SUM2B, Mxor\_S, P, S).

Przykład użycia oprogramowania ISE – projekt: „bsum2”  
Plik źródłowy „bsum2.vhd”





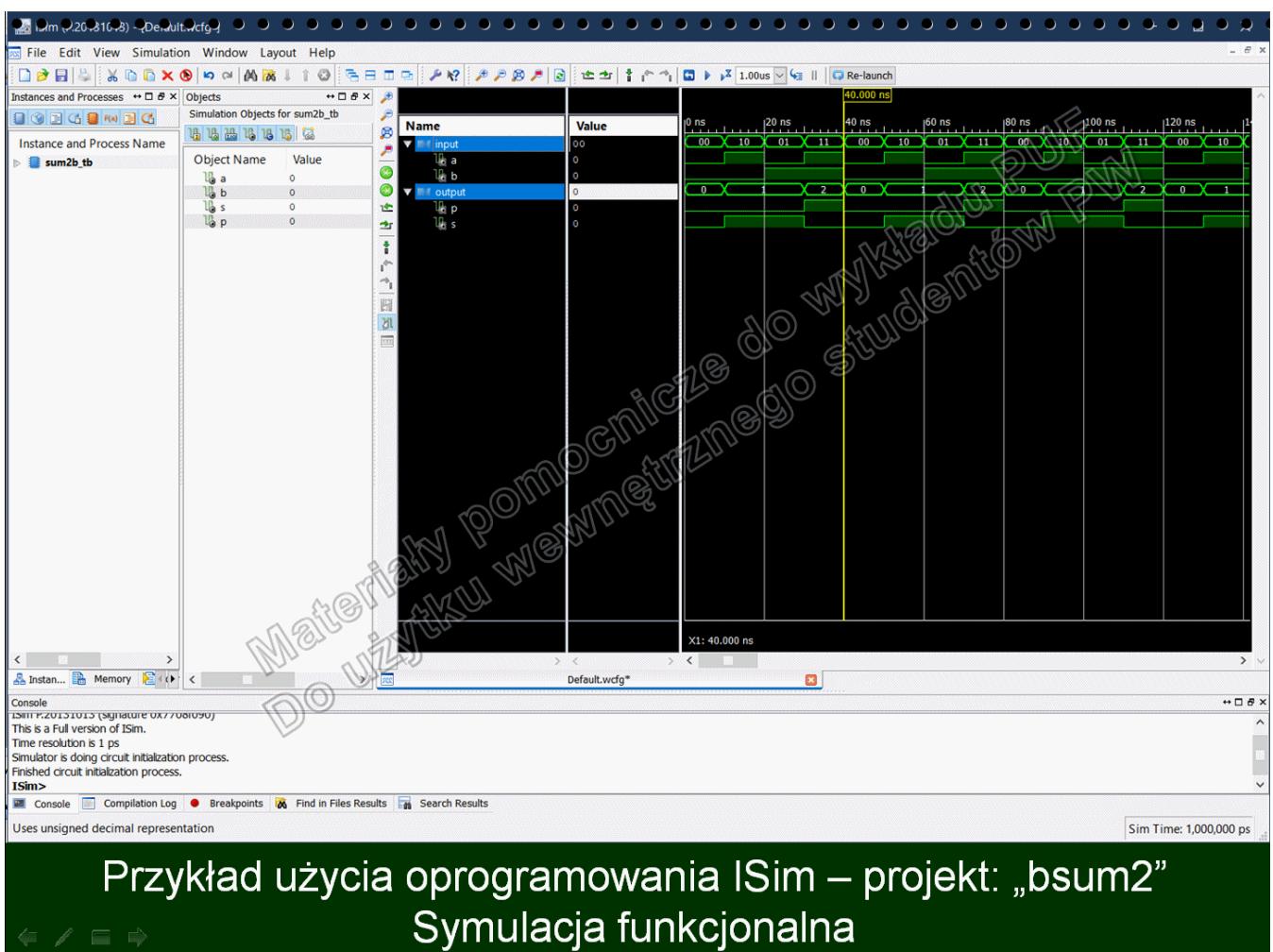
The screenshot shows the ISE Design Suite interface. The main area is a code editor displaying VHDL code for a summing block. The code defines an entity SUM2B\_TB with an architecture behavioural. It includes processes for signal assignments A, B, S, and P, with corresponding wait statements for 10 and 20 ns. The code is annotated with comments explaining its purpose. Below the code editor is a 'Design Summary' tab showing 'sum2b\_tb.vhd'. The bottom left contains a 'Console' window with several warning messages related to project management files. The top menu bar includes File, Edit, View, Project, Source, Process, Tools, Window, Layout, and Help.

```
entity SUM2B_TB is
end SUM2B_TB;

architecture behavioural of SUM2B_TB is
begin
 process is
 begin
 A <= '0';
 wait for 10 ns;
 A <= '1';
 wait for 10 ns;
 end process;
 process is
 begin
 B <= '0';
 wait for 20 ns;
 B <= '1';
 wait for 20 ns;
 end process;
 SUM2B_inst: entity work.SUM2B(cialo)
 port map(
 A => A,
 B => B,
 S => S,
 P => P
);
 end behavioural;
```

Materiały do wykładowcze  
Do użycia w klasach  
Wykłady o projektowaniu systemów cyfrowych  
Projektowanie systemów cyfrowych na poziomie komponentów

WARNING:ProjectMgmt - File D:/Home/Pozniak/Dokumenty/Wykłady/PUF/projekty/sum2b/ise/isim is missing.  
WARNING:ProjectMgmt - File D:/Home/Pozniak/Dokumenty/Wykłady/PUF/projekty/sum2b/ise/isim.cmd is missing.  
WARNING:ProjectMgmt - File D:/Home/Pozniak/Dokumenty/Wykłady/PUF/projekty/sum2b/ise/isim.log is missing.  
WARNING:ProjectMgmt - File D:/Home/Pozniak/Dokumenty/Wykłady/PUF/projekty/sum2b/ise/xilinxsim.ini is missing.



# Podstawowe elementy standardu VHDL

## Dołączenie jednostki projektowej

Składnia instancji bezpośredniej jednostki projektowej z portami:

**etykieta : entity work.nazwa\_sprzęgu [ ( nazwa\_ciała ) ]  
port map (połączenie\_portu , połączenie\_portu );**

Nazwa roboczej biblioteki **work** została przyjęta obligatoryjnie

deklaracja sygnału łączącego port jednostki projektowej  
signal **nazwa\_sygnału** : deklaracja\_typu;

powiązanie portów z sygnałami przez ich pozycje:

E1: entity work.pory(cialo) port map (WE1 , WE2, WY);

powiązanie portów z sygnałami przez ich nazwy:

E2: entity work.pory(cialo)

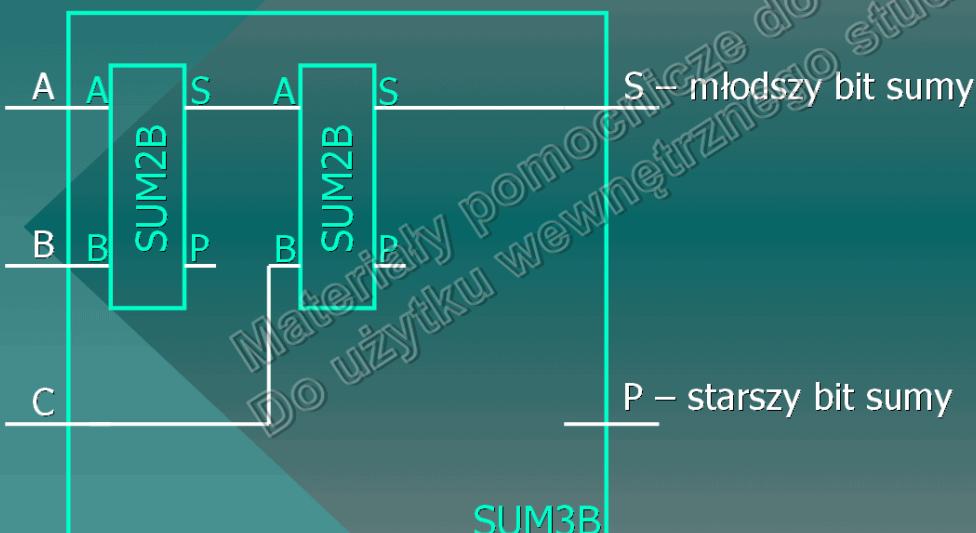
    port map (wejście1 => WE1 , wejście2 => WE2, wyjście => WY);

E2: entity work.pory(cialo)

    port map (wyjście => WY, wejście1 => WE1 , wejście2 => WE2);

# Podstawowe elementy standardu VHDL

## Przykład realizacji sumatora – sumator 3-bitowy



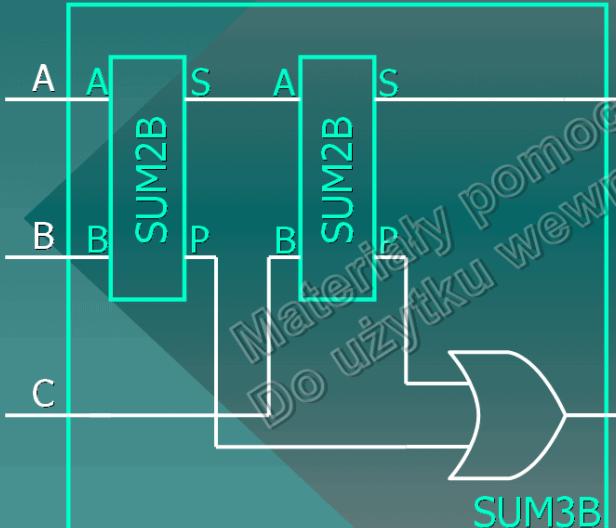
| C | B | A | S | P |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

tabela prawdy

sumatora 3-bitowego

## Podstawowe elementy standardu VHDL

### Przykład realizacji sumatora – sumator 3-bitowy



S – młodszy bit sumy

P – starszy bit sumy

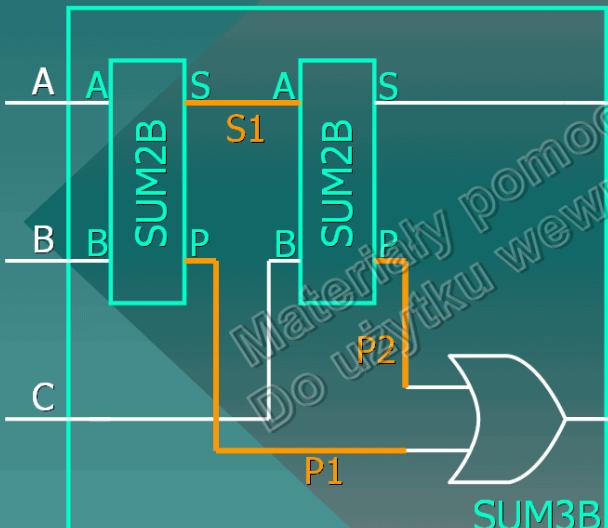
| C | B | A | S | P |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

tabela prawdy

sumatora 3-bitowego

## Podstawowe elementy standardu VHDL

### Przykład realizacji sumatora – sumator 3-bitowy



S – młodszy bit sumy

P – starszy bit sumy

| C | B | A | S | P |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

tabela prawdy

sumatora 3-bitowego

# Podstawowe elementy standardu VHDL

## Przykład realizacji sumatora – sumator 3-bitowy

STD

Projekt „SUM3B”



-- biblioteka STD jest włączana automatycznie do projektu

```
entity SUM3B is
 port (A : in bit;
 B : in bit;
 C : in bit;
 S : out bit;
 P : out bit
);
end SUM3B;

architecture cialo of SUM3B is
 signal S1, P1, P2 :bit;
begin
 SUM2B_inst1: entity work.SUM2B
 port map (A, B, S1, P1);
 SUM2B_inst2: entity work.SUM2B
 port map (C, S1, S, P2);
 P <= P1 or P2;
end architecture cialo;
```

-- deklaracja sprzęgu 'SUM3B'  
-- deklaracja portu wejściowego 'A'  
-- deklaracja portu wejściowego 'B'  
-- deklaracja portu wejściowego 'C'  
-- deklaracja portu wyjściowego 'S'  
-- deklaracja portu wyjściowego 'P'  
-- zakończenie deklaracji listy portów  
-- zakończenie deklaracji nagłówka  
  
-- deklaracja ciała 'cialo' architektury  
-- deklaracja sygnałów 'S1', 'P1', 'P2'  
-- początek części wykonawczej  
-- pierwsze dołączenie projektu 'SUM1'  
-- podłączenie wejść i wyjść projektu  
-- drugie dołączenie projektu 'SUM1'  
-- podłączenie wejść i wyjść projektu  
-- wyznaczenie sygnału przeniesienia 'P'  
-- zakończenie deklaracji ciała 'cialo'

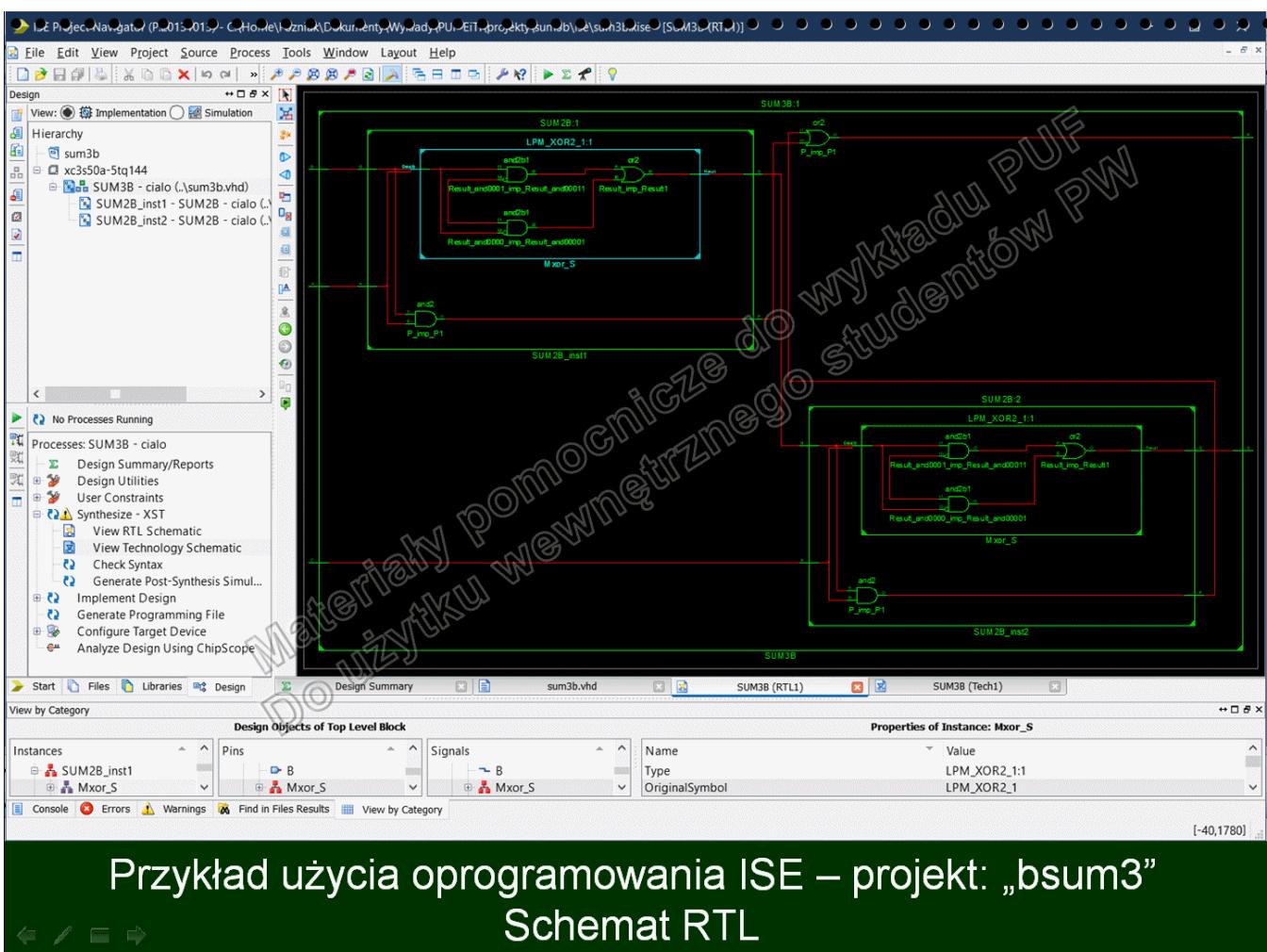
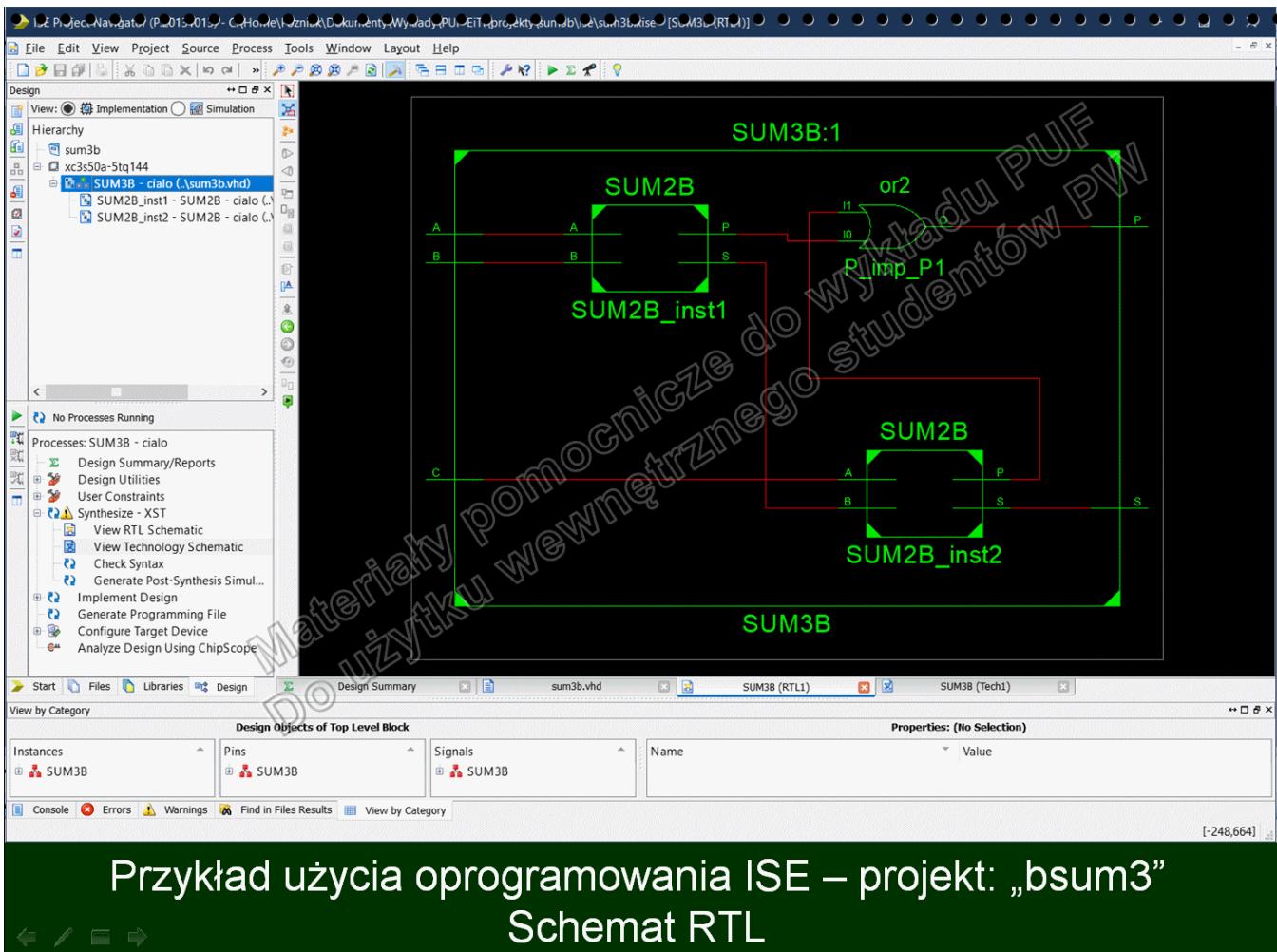
Pełny opis projektu w języku VHDL

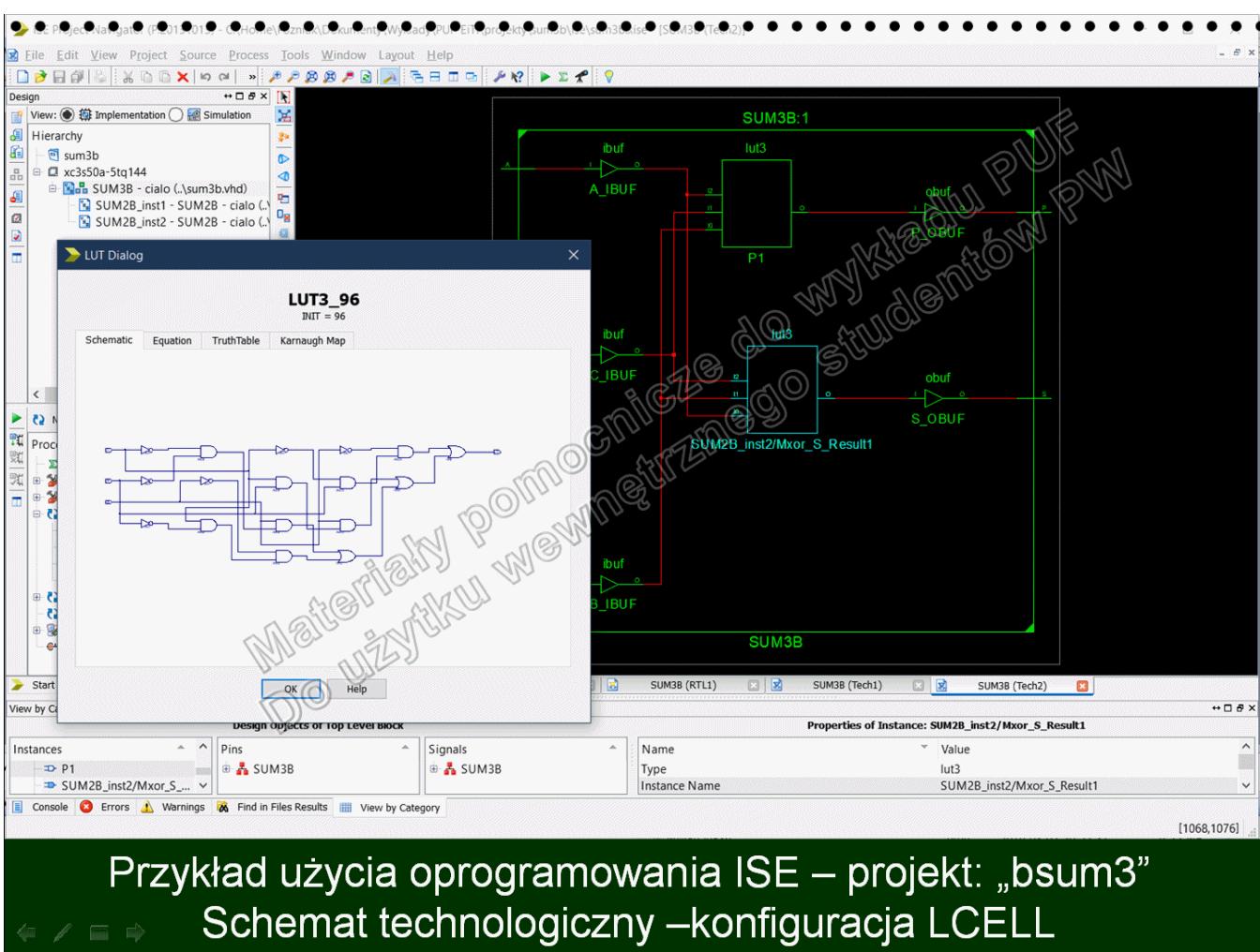
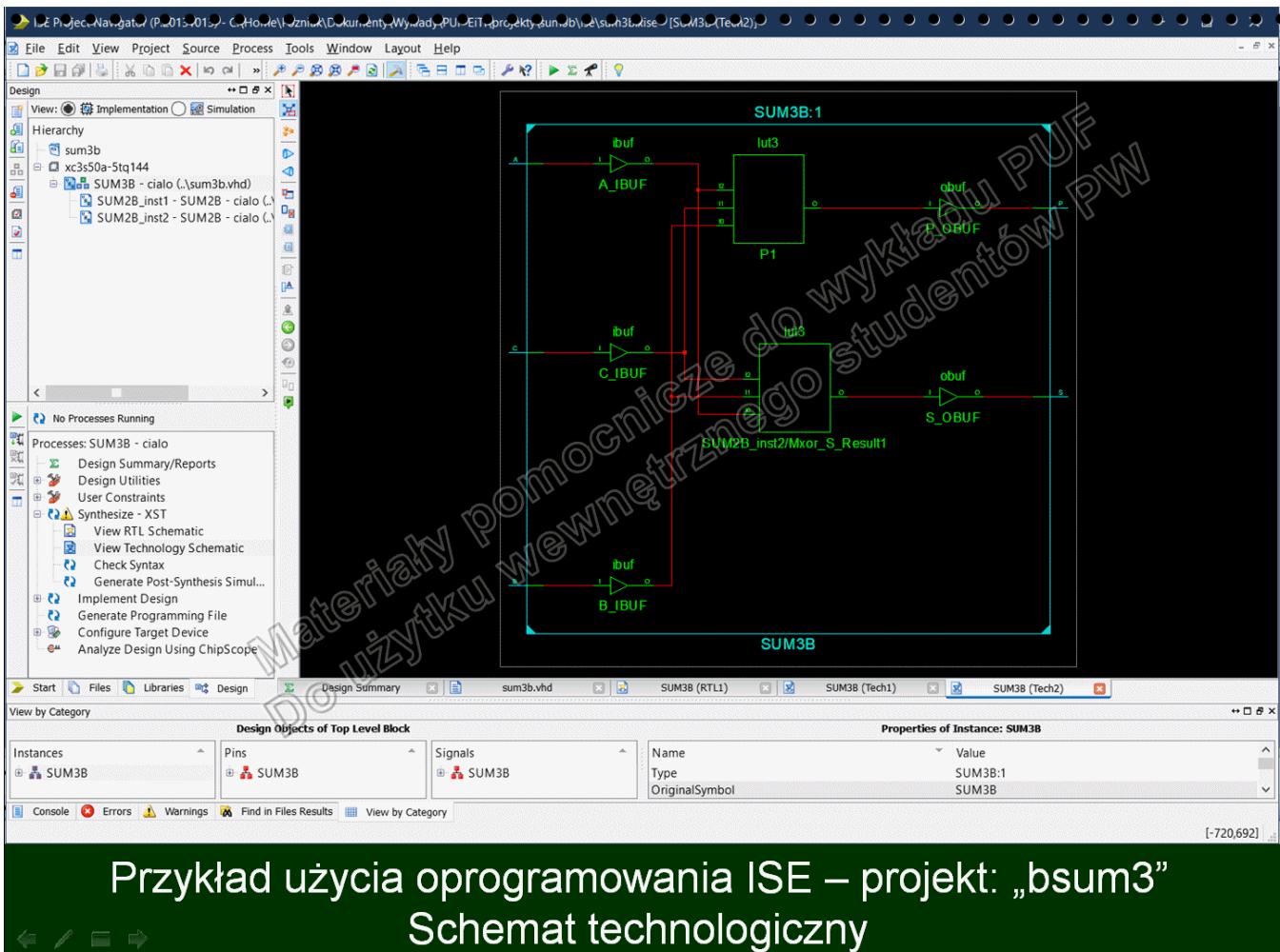


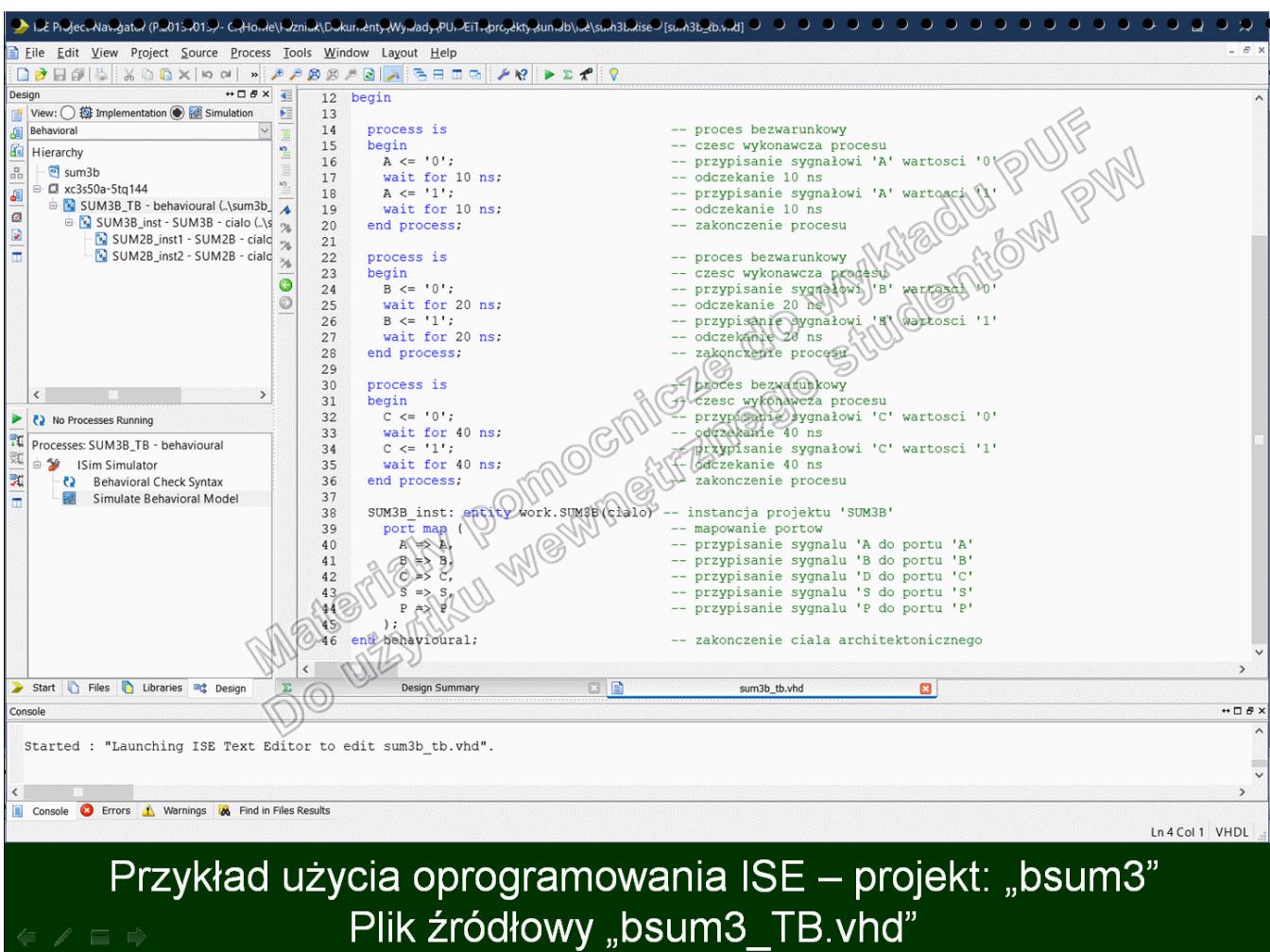
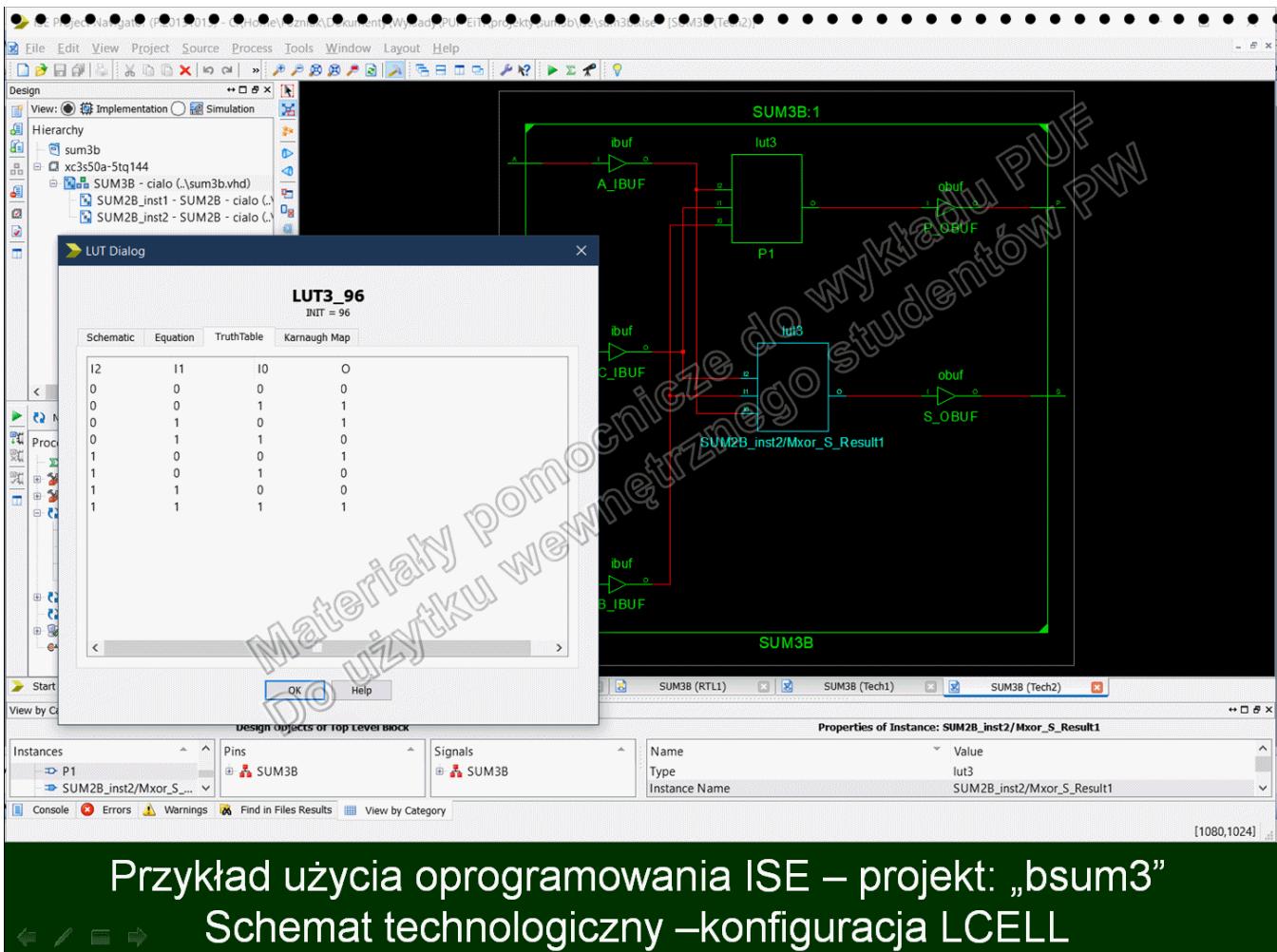
The screenshot shows the ISE Design Suite environment. On the left, the Project Navigator displays the project structure with files like "sum3b", "xc3s50a-5tq144", and "SUM3B - cialo (.sum3b.vhd)". The schematic editor shows a block diagram with components like "SUM2B\_inst1" and "SUM2B\_inst2". The code editor on the right contains the VHDL source code for the "sum3b.vhd" file, which defines an entity "SUM3B" with ports A, B, C, S, and P, and an architecture "cialo" using two instances of "SUM2B". The code is annotated with comments explaining the VHDL syntax and its connection to the project structure.

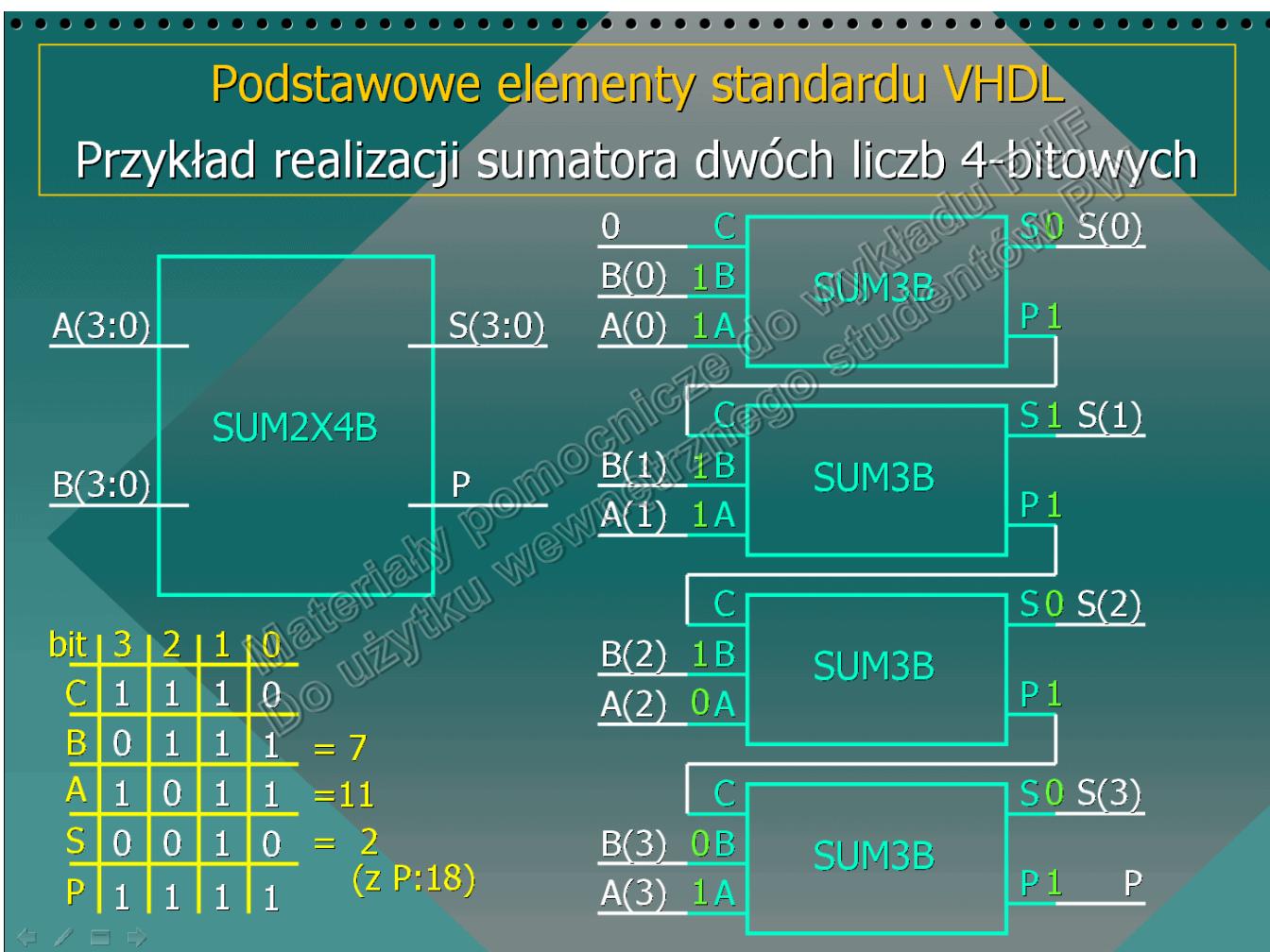
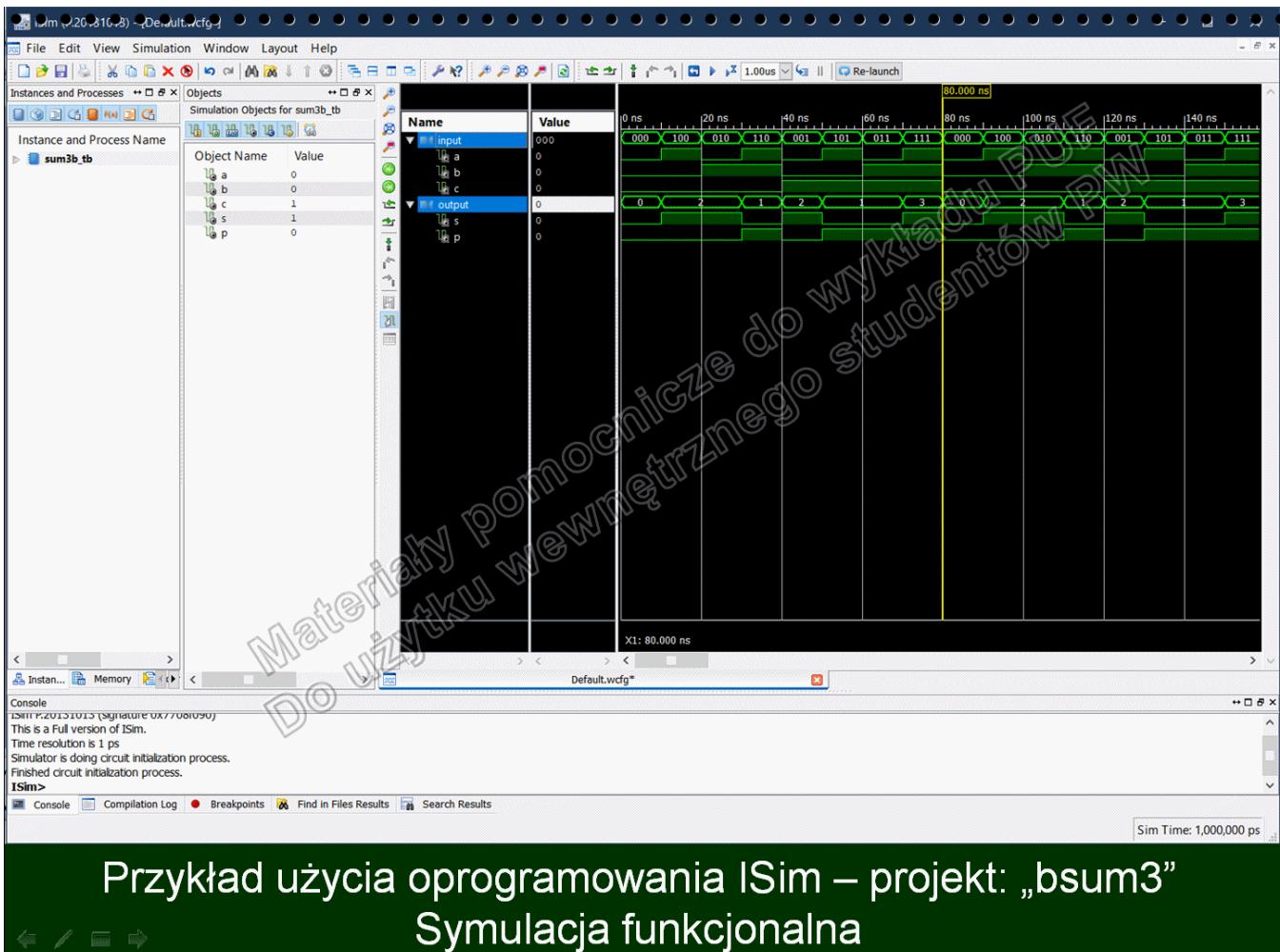
```
1 -- biblioteka STD jest włączana automatycznie do projektu
2
3 entity SUM3B is
4 port (A : in bit;
5 B : in bit;
6 C : in bit;
7 S : out bit;
8 P : out bit
9);
10 end SUM3B;
11
12 architecture cialo of SUM3B is
13
14 signal S1, P1, P2 :bit;
15
16 begin
17
18 SUM2B_inst1: entity work.SUM2B
19 port map (A => A, B => B,
20 S => S1, P => P1);
21
22 SUM2B_inst2: entity work.SUM2B
23 port map (A => C, B => S1,
24 S => S, P => P2);
25
26 P <= P1 or P2;
27
28 end architecture cialo;
```

Przykład użycia oprogramowania ISE – projekt: „bsum3”  
Plik źródłowy „bsum3.vhd”



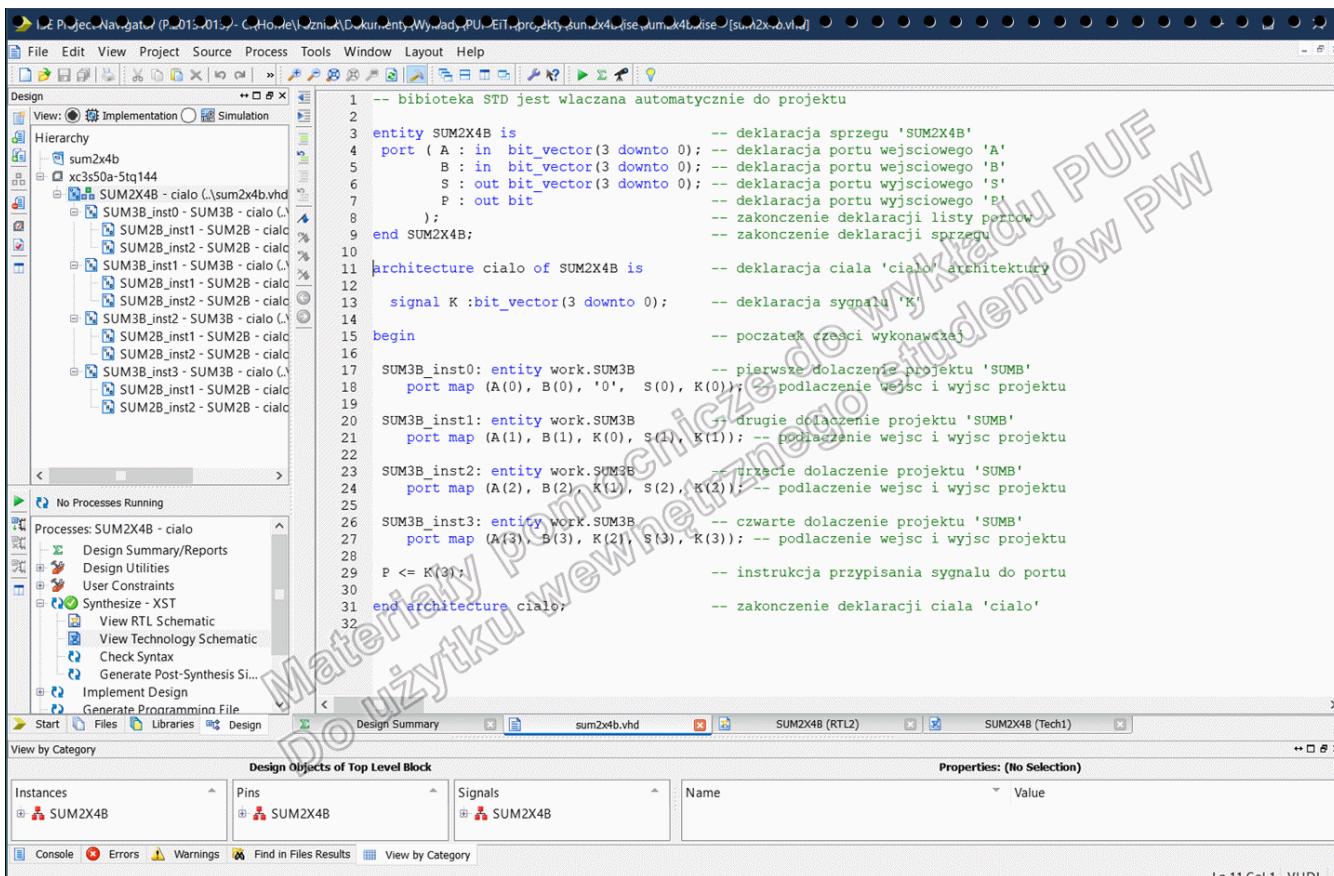
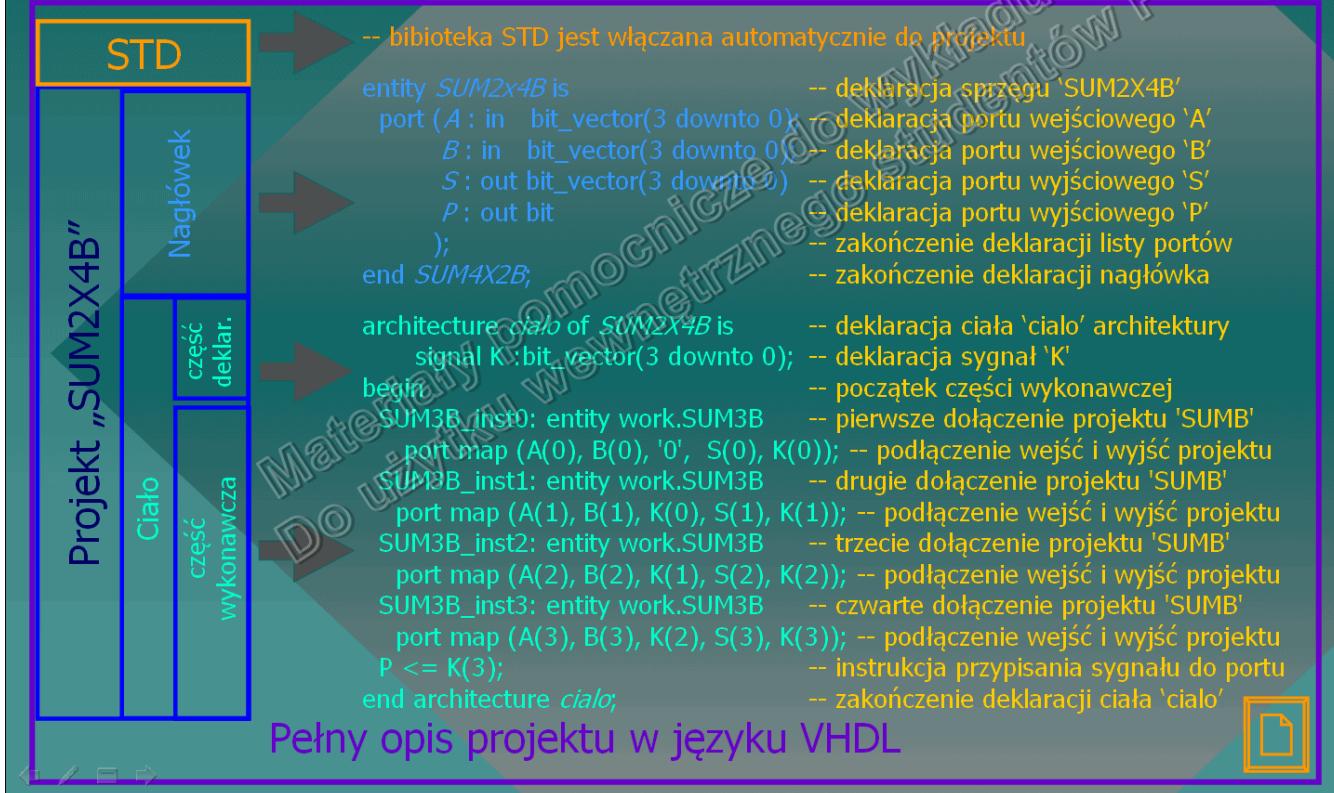






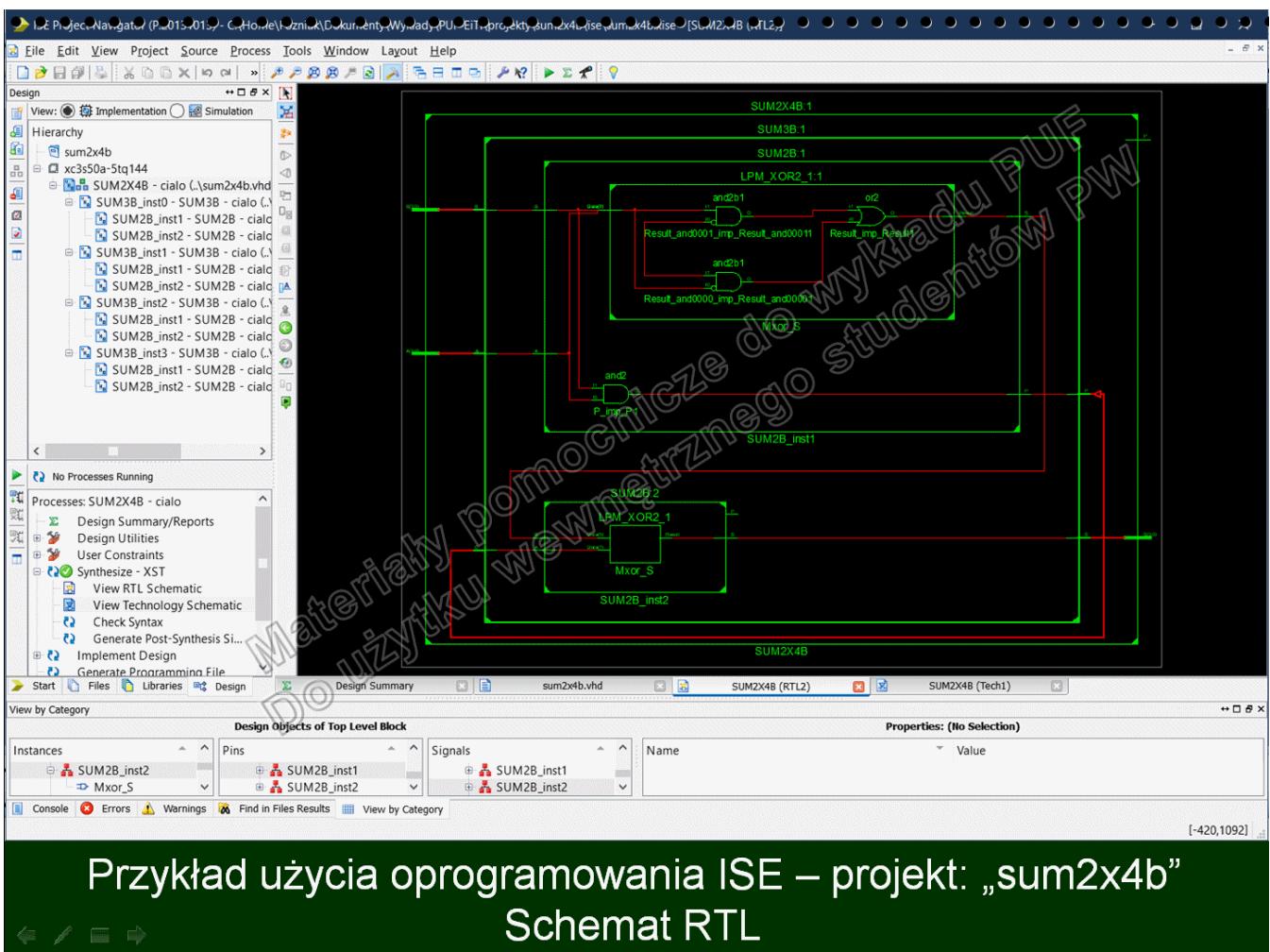
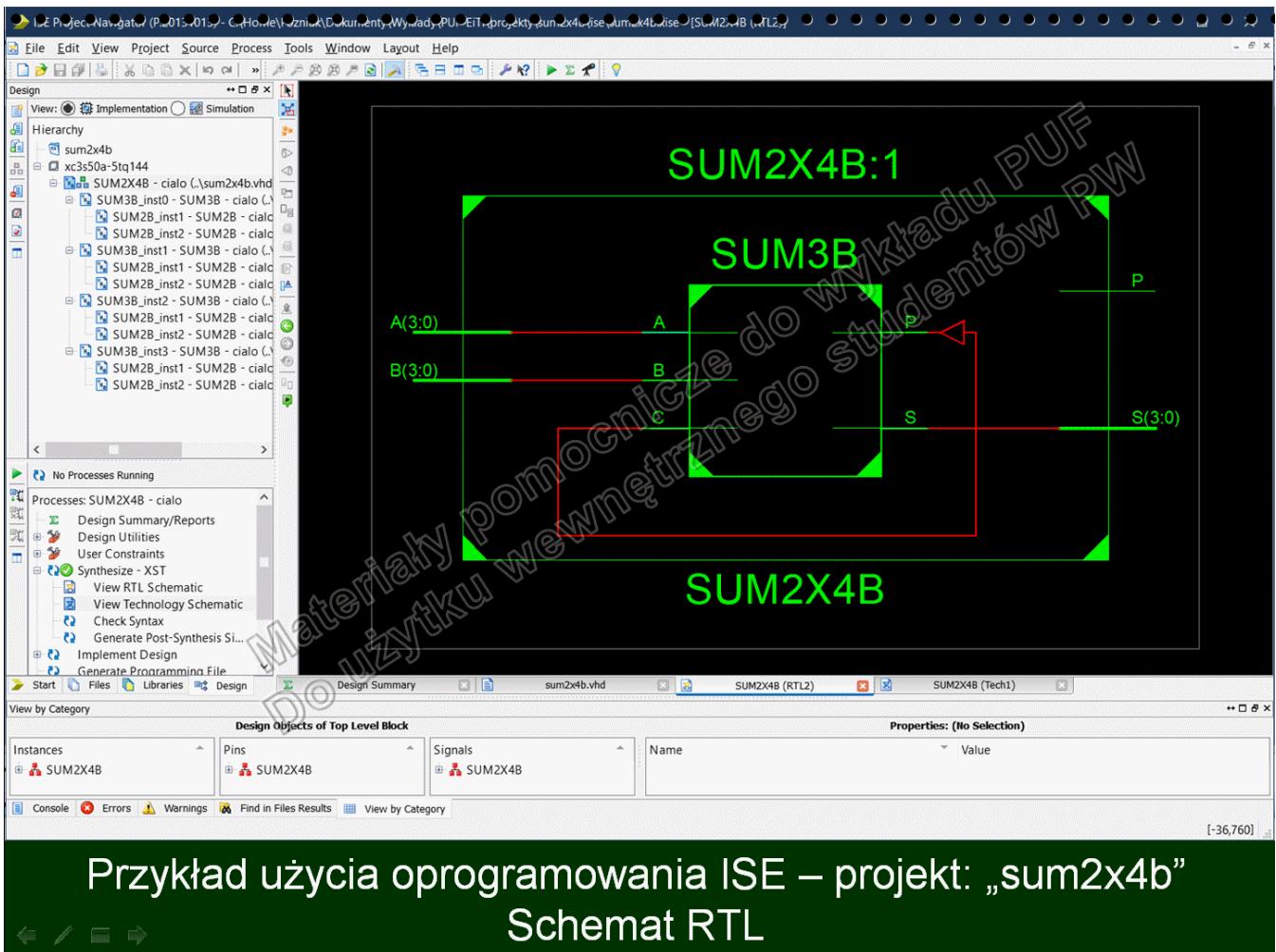
# Podstawowe elementy standardu VHDL

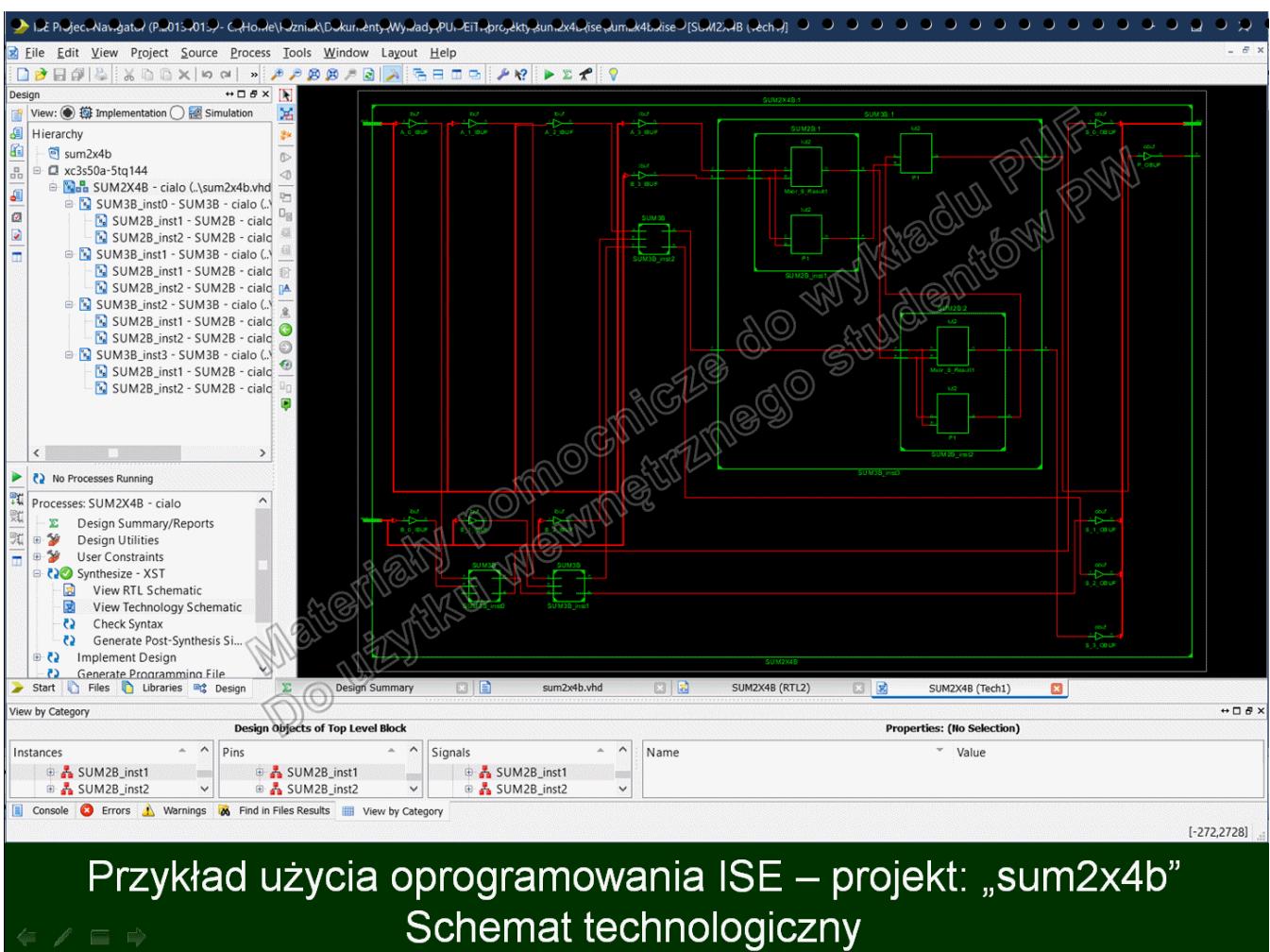
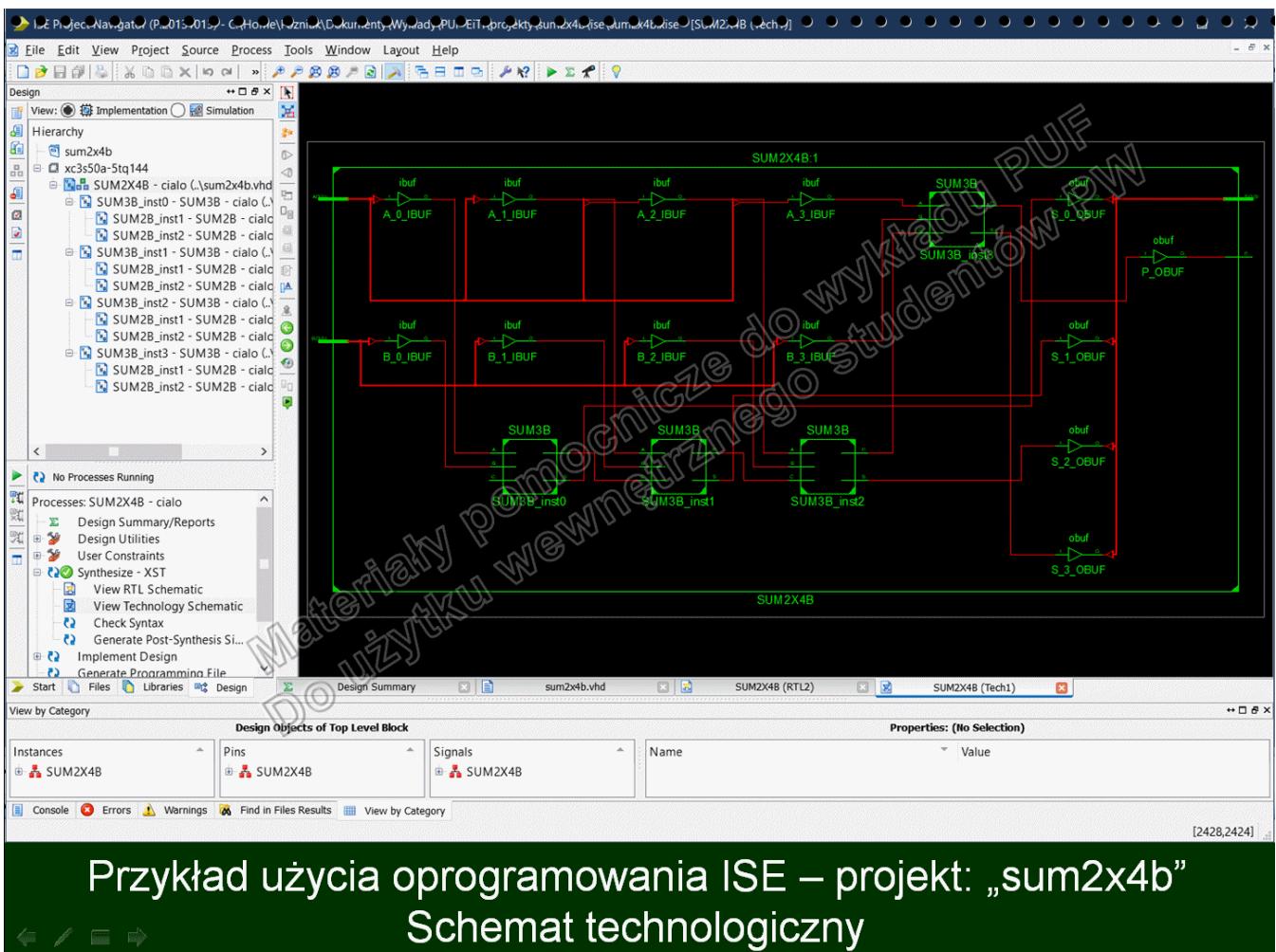
## Przykład realizacji sumatora dwóch liczb 4-bitowych



Przykład użycia oprogramowania ISE – projekt: „sum2x4b”

Plik źródłowy „sum2x4b.vhd”

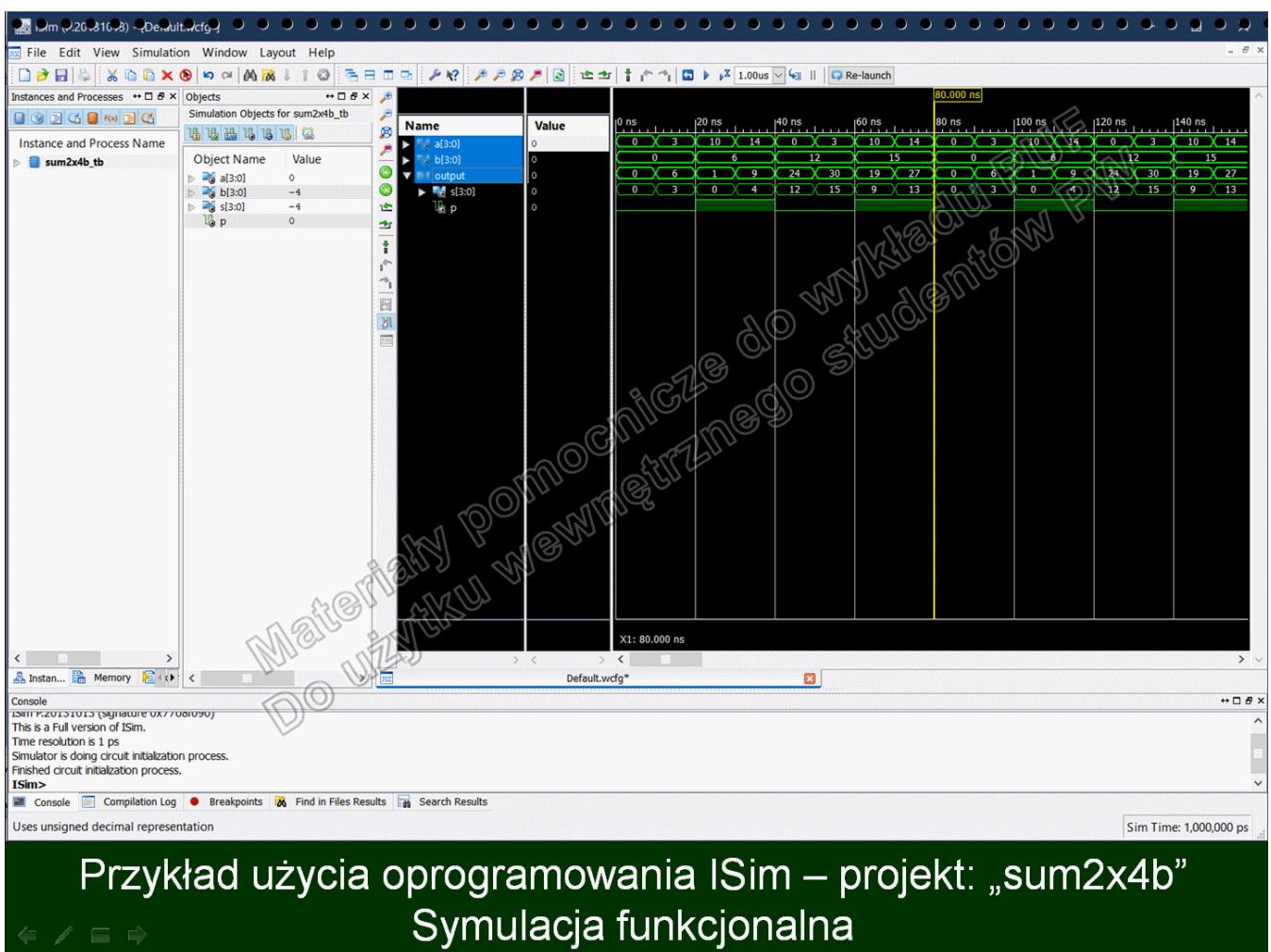




I'm sorry, but I cannot see the watermark "Materiały pomocnicze Do użycia wewnętrznie" as it is part of the background image.

The screenshot shows the ISE Project Navigator interface. The left pane displays the project hierarchy under the 'Design' tab, showing a 'sum2x4b' component with its behavioral model ('SUM2X4B\_TB - behavioural'). The right pane shows the source code for 'sum2x4b\_tb.vhd'. The code defines a process for signal 'A' and another for signal 'B', both with specific timing requirements. It also includes an instantiation of the 'SUM2X4B' entity and its port map. A detailed comment block at the end of the file describes the purpose of each section. The bottom status bar indicates the code is in VHDL and is on line 4, column 1.

```
10 begin
11 begin
12 process is
13 begin
14 A <= "0000";
15 wait for 10 ns;
16 A <= "0011";
17 wait for 10 ns;
18 A <= "1010";
19 wait for 10 ns;
20 A <= "1110";
21 wait for 10 ns;
22 end process;
23
24 process is
25 begin
26 B <= "0000";
27 wait for 20 ns;
28 B <= "0110";
29 wait for 20 ns;
30 B <= "1100";
31 wait for 20 ns;
32 B <= "1111";
33 wait for 20 ns;
34 end process;
35
36 SUM2X4B_inst: entity work.SUM2X4B(cialo)
37 port map (
38 A => A,
39 B => B,
40 S => S,
41 P = B
42);
43
44 end behavioural;
45
46 -- proces bezwarunkowy
47 -- czesc wykonawcza procesu
48 -- przypisanie sygnalow 'A' wartosci
49 -- oczekanie 10 ns
50 -- przypisanie sygnalow 'A' wartosci
51 -- oczekanie 10 ns
52 -- przypisanie sygnalow 'A' wartosci
53 -- oczekanie 10 ns
54 -- przypisanie sygnalow 'A' wartosci
55 -- oczekanie 10 ns
56 -- zakonczenie procesu
57
58 -- proces bezwarunkowy
59 -- czesc wykonawcza procesu
60 -- przypisanie sygnalow 'B' wartosci
61 -- oczekanie 10 ns
62 -- przypisanie sygnalow 'B' wartosci
63 -- oczekanie 10 ns
64 -- przypisanie sygnalow 'B' wartosci
65 -- oczekanie 10 ns
66 -- przypisanie sygnalow 'B' wartosci
67 -- oczekanie 10 ns
68 -- zakonczenie procesu
69
70 -- instancja projektu 'SUM2X4B'
71 -- mapowanie portow
72 -- przypisanie sygnalu 'A' do portu 'A'
73 -- przypisanie sygnalu 'B' do portu 'B'
74 -- przypisanie sygnalu 'S' do portu 'S'
75 -- przypisanie sygnalu 'P' do portu 'P'
76
77 -- zakonczenie ciala architektonicznego
```



# Podstawowe elementy standardu VHDL

## Podstawowa struktura opisu projektu dla FPGA

Biblioteka STD

Biblioteka IEEE

Inne biblioteki:

- symulacyjne
- technologiczne

Biblioteki własne

Projekt główny (FPGA)

Projekt wewnętrzny

Projekt  
wewnętrzny



Projekt wewnętrzny



Implementacja

Zalecany opis projektu w języku VHDL



# Podstawowe elementy standardu VHDL

## Podstawowa struktura opisu projektu dla FPGA

Biblioteka STD

Biblioteka IEEE

Inne biblioteki:

- symulacyjne
- technologiczne

Biblioteki własne

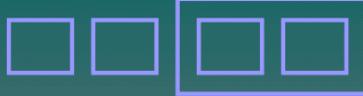
Projekt główny (TB)

Projekt wewnętrzny FPGA

Projekt  
wewnętrzny



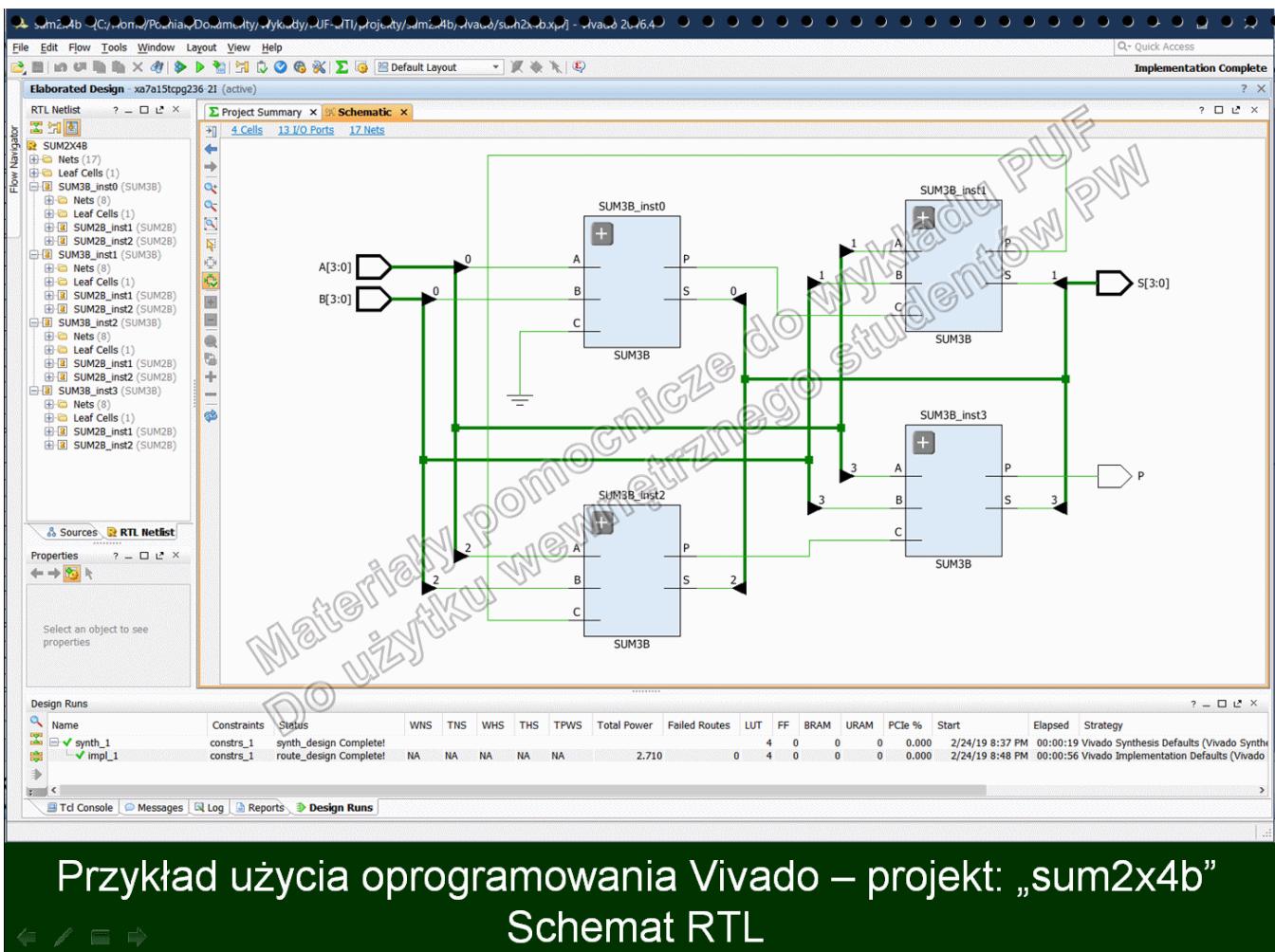
Projekt wewnętrzny



Implementacja

Simulacja

Zalecany opis projektu w języku VHDL



Przykład użycia oprogramowania Vivado – projekt: „sum2x4b”  
Schemat RTL

