Ten tutorial uczy jak tworzyć projekty wykorzystujące edytor schematów Block Diagram Editor. Zakładamy, że student jest zaznajomiony z podstawową obsługą programu Active-HDL. W tym tutorialu użyjemy języka VHDL do zamodelowania pełnego sumatora (1-bitowego, z wejściem i wyjściem przeniesienia). Następnie, z pomocą edytora schematów, zaprojektujemy trzybitowy sumator z przeniesieniem szeregowym. Na kolejnym schemacie połączymy sumator z dekoderem wyświetlacza siedmiosegmentowego. Przy czym, dekoder będzie zrealizowany w języku VHDL.

## Tworzymy nowy projekt

1. Uruchomić "Aldec Actve-HDL" (skrót na pulpicie). Program uruchamia się dosyć długo – cierpliwości.



Rys. 1. Ikona na pulpicie – skrót do Aldec Active-HDL.

2. Pojawi się okno z pytaniem o licencję. Kliknąć "Next".

License Configuration	5 X
	Select one of the Active-HDL product configurations and click [Next>]. EDU Mixed Design Entry Reserve simulation features at startup Simulation features For more details click license information. License information
	<back next=""> Cancel</back>

Rys. 2. Okno konfiguracji licencji.

3. Jeżeli program zapyta się o sprawdzenie aktualizacji, to wybrać "Cancel".

Download Active-HDL Updates
Please, choose how frequently Active-HDL should check for available updates on the Aldec website. To disable automatic checks, choose 'never'.
Check for updates: weekly
If updates are not enabled you may miss some important software and library upgrades.
Press OK to check for updates on Aldec website now.
OK Cancel

Rys. 3. Okno aktualizacji programu.

4. W oknie wybrać "Create new workspace" i kliknąć "OK".



Rys. 4. Wybór przestrzeni roboczej.

5. Wprowadzić nazwę przestrzeni projektowej (tekst bez odstępów, bez polskich liter oraz bez znaków specjalnych) i kliknąć "OK".



Rys. 5. Nadanie nazwy przestrzeni roboczej.

6. W kolejnym oknie wybrać "Create an Empty Design with Design Flow" i kliknąć "Dalej".

How would you like to	create Design Resources?
C C	Create an Empty Design
	Create an Empty Design with Design Flow
C.	Add existing Resource Files
	Import a Design from Active-CAD
This option cr Row Manage implementatio default HDL I	reates an empty design and enables Design r: You can select a vendor of your synthesis or in tool, technology, libraries, and specify the anguage of your new design entry sources.
	Freate New Worksnace

Rys. 6. Wybór rodzaju projektu. Design Flow umożliwia uruchomienie syntezy i implementacji projektu.

 Sprawdzić czy w ustawieniach Design Flow są wskazane narzędzia "Synthesis tool" oraz "Implementation tool" (zapytać prowadzącego która wersja Xilinx ISE jest zainstalowana w laboratorium).
 Sprawdzić czy wybrana jest poprawna rodzina układów FPGA (zależy od wykorzystywanego zestawu ewaluacyjnego).

Block Diagram Configuration: Default HDL Language

Default HDL Language: VHDL

W przypadku różnic kliknąć "Flow Settings" i dokonać odpowiednich zmian. Jeżeli wszystko się zgadza, kliknąć "Dalej".

8. Wprowadzić nazwę projektu (tekst bez odstępów, bez polskich liter i znaków specjalnych) i kliknąć "Dalej".

ype the design name: BDE_Tutorial Select the location of the design folder: d:\My_Designs\Zaoczni_Lab6_8_00 Browse The name of the default working library of the design: BDE_Tutorial The name specified here will be used as the file name for the brary files and as the logical name of the library. You can shange the logical name later on.		ew design.	 
Select the location of the design folder: d:\My_Designs\Zaoczni_Lab6_8_00 Browse The name of the default working library of the design: BDE_Tutorial The name specified here will be used as the file name for the brary files and as the logical name of the library. You can shange the logical name later on.	lype the design name:		
Select the location of the design folder: d:\My_Designs\Zaoczni_Lab6_8_00 Browse The name of the default working library of the design: BDE_Tutorial The name specified here will be used as the file name for the brary files and as the logical name of the library. You can shange the logical name later on.	bbc_rutonai		
d:\My_Designs\Zaoczni_Lab6_8_00         Browse         Browse         BDE_Tutorial         The name specified here will be used as the file name for the brary files and as the logical name of the library. You can shange the logical name later on.	Select the location of the design folde	er:	
Browse The name of the default working library of the design: BDE_Tutorial The name specified here will be used as the file name for the brary files and as the logical name of the library. You can shange the logical name later on.	d:\My_Designs\Zaoczni_Lab6_8_00	)	
The name of the default working library of the design: BDE_Tutorial The name specified here will be used as the file name for the brary files and as the logical name of the library. You can shange the logical name later on.			Browse
The name of the default working library of the design: BDE_Tutorial The name specified here will be used as the file name for the brary files and as the logical name of the library. You can thange the logical name later on.			
BDE_Tutorial The name specified here will be used as the file name for the brary files and as the logical name of the library. You can hange the logical name later on.	The second of the she for the second size of the se	ny of the design :	
The name specified here will be used as the file name for the brary files and as the logical name of the library. You can shange the logical name later on.	i ne name of the default workind libra	IV ULLIE GESIGIT.	
	ne name of the default working libra BDE_Tutorial	ly of the design.	

Rys. 7. Nadanie nazwy projektowi.

9. W kolejnym oknie wybrać "Zakończ". Powinniśmy na ekranie otrzymać widok jak na poniższym rysunku.

Active-nDL 6.1 (Zaoczni_Labo_o_UU, 6DE_Tutonai) - Design Flow Manager	
ile <u>E</u> dit Sea <u>r</u> ch <u>Vi</u> ew W <u>o</u> rkspace <u>D</u> esign <u>Simulation T</u> ools <u>W</u> indow <u>H</u> elp	<b>∲</b> »
🗿 ▼ 🚔 🔜 💥 📖 📓 💱 🔵 🔎 🗱 🧱 🌮 🛝 🏙 習 🖓 🙀 🐼 💁 🗇 🍪 😓 → 📦 🖢 100 ns 🗄 🕂 🔳 🤙 🕨	F∃ ⊊∃ G∃ No simulation
Design Browser	-
Top-Level selection	
Unsorted DUnsorted	
r∰ Workspace 'Zaoczni_Lab6_8_00': 1 design(s)	
BDE_Tutorial Options	
Add New File	
Add New Library	
BDE_lutonal library	
reports implementation Longians simulation	
Flow Settings	
options Continue ?	
PCB Interface	
Files Structure Ca Resources /	
• # DESIGN: Default Design Language: VHDL	
• # DESIGN: C-Synthesis: Not Defined	
• # DESIGN: HDL Synthesis: Not Defined	
<pre># DESIGN: Implementation: Not Defined</pre>	
>	
Sconsole /	
	NUM INS

Rys. 8. Główne okno programu po założeniu projektu.

# Tworzymy nowy plik VHDL

10. W Design Browser dwukrotni kliknij na Add New File. Pojawi się okno dialogowe umożliwiające dołożenie pliku do projektu.

Zmień zakładkę na Wizards i wskaż VHDL Source Code. Kliknij OK.

Active-HDL 8.1 (Zaoczni_Lab6_8_00 ,BDE_Tutorial) - Design Flow	Manager	
<u>File E</u> dit Sea <u>r</u> ch <u>V</u> iew W <u>o</u> rkspace <u>D</u> esign <u>Simulation T</u>	sols <u>W</u> indow <u>H</u> elp	+ <u>+</u> + ≫ ×
🗿 🕶 🚔 🔜 💥 📖 🚳 😻 💭 🚟 🚟 💲 👞	🏙 😨 🖓 🍇 ⊗ 🕹 🄣 🕨 🕨 🕨 100 ns 🗄 👯 🗉 🚽 🕪 🖓 🗐 💭 Simulation	
Design Browser	Price        Price	

Rys. 9. Użycie kreatora do stworzenia pliku VHDL.

- 11. W pierwszym oknie kreatora zostaw domyślnie zaznaczoną opcję "Add the generated file to the design" i kliknij Dalej.
- 12. W kolejnym oknie podaj nazwę pliku źródłowego do stworzenia. Wpisz adder1 jak pokazano na rysunku poniżej i kliknij Dalej.

New Source File W	zard - Name
	Type the name of the source file to create:
	adder1 Browse
	You can use the Browse button to specify the file.
	Type the name of the entity (optional):
	By default, the entity name is the same as the file name.
	Type the name of the architecture body (optional):
	<ul> <li>By default, the architecture name is the same as the entity name.</li> </ul>
	< Wstecz Dalej > Anuluj

Rys. 10. Nadanie nazwy plikowi VHDL. Nazwy Entity i Architecture będą identyczne jak nazwa pliku.

13. Kolejne okno pozwala na dołożenie portów. Aby dodać nowy port/sygnał należy kliknąć przycisk "New". W polu "Name" wprowadza się jego nazwę. Pola "Array Indexes" umożliwiają definiowanie magistral. Trzeba też zdefiniować kierunek sygnału (in / out / inout) oraz jego typ (używamy wyłącznie STD\_LOGIC oraz STD\_LOGIC\_VECTOR!).

Proszę dołożyć pięć sygnałów zgodnie z poniższą tabelą.

Name	Array Indexes	Port direction	Туре
А	(puste)	In	STD_LOGIC
В	(puste)	In	STD_LOGIC
Cin	(puste)	In	STD_LOGIC
Cout	(puste)	Out	STD_LOGIC
Sum	(puste)	Out	STD_LOGIC

Aby zmodyfikować wcześniej wprowadzony sygnał należy zaznaczyć jego nazwę i dokonać odpowiednich zmian. Błędne / niepotrzebne sygnały można skasować przyciskiem "Delete" na klawiaturze.

W lewej części okna mamy graficzną reprezentację projektowanego elementu. Stworzone wejścia układają się na lewej krawędzi bloczka, z kolei wyjścia układu pojawiają się na prawej krawędzi. Magistrale oznaczane są grubszym "przewodem".



Rys. 11. Konfiguracja portów.

14. Następnie kliknij "Zakończ".

W oknie Design Browser pojawi się nowy plik – adder1.vhd. Plik ten zawiera kompletne entity oraz pustą architekturę. W kolejnych punktach stworzymy kod "Data Flow" opisujący działanie jednobitowego, pełnego sumatora.



Rys. 12. Nowy plik w Design Browser.

15. W Design Browser, dwukrotnie kliknij na pliku adder1.vhd aby wyświetlić jego zawartość. Przepisz kod zgodnie z poniższym rysunkiem.

```
Sum <= A xor B xor Cin;
Cout <= (Cin and A) or (Cin and B) or (A and B);
         24
             library IEEE;
use IEEE.STD_LOGIC_1164.all;
         25
         26
27
             entity adder1 is
          28
          29
                  port (
                      A : in STD_LOGIC;
          30
          31
                       B : in STD_LOGIC;
          32
                       Cin : in STD_LOGIC;
                       Cout : out STD_LOGIC;
          33
          34
                       Sum : out STD_LOGIC
          35
                       );
         36
37
             end adder1;
          38
              --}} End of automatically maintained section
          39
          40
             architecture adder1 of adder1 is
          41
             begin
          42
          43
                   -- enter your statements here --
             Sum <= A xor B xor Cin;
Cout <= (Cin and A) or (Cin and B) or (A and B);
          44
         45
          46
         47 end adder1;
                           Rys. 13. Edycja kodu źródłowego.
```

- 16. Zapisz zmiany w pliku.
- 17. Wybierz z głównego menu Design -> Compile.

### Tworzymy nowy schemat blokowy

- 18. Ponownie, dwukrotnie kliknij Add New File w Design Browser.
- 19. Pozostań na zakładce Empty Files. Wskaż Block diagram. W polu Name wpisz adder3.

Design Browser	🚾 ] 薛 薛 麗 藍 屬 🦉 薄 峻 🏭 🏷 🕼 🧀 👖 🖽 蒜 ] 🔸 🗉
adder1 (adder1)	
O       Unsorted         Image: Straight of the straight of	Add New File
⊕∰ BDE_Tutorial library	Empty Files Wizards
	New Empty File:       Name:       jadder3       Add Existing File       OK     Anuluj

Rys. 14. Tworzenie schematu blokowego.

20. Kliknij OK. Otworzy się okno z pustym schematem blokowym.



Rys. 15. Puste okno edytora schematów blokowych.

21. Kliknij żółtą ikonę bramki AND wskazana na Rysunku 14. Otworzy się biblioteka elementów / symboli. Dostępne elementy są pogrupowane.

W "Built-in symbols" są elementy których NIE używamy!

Jest też "biblioteka" elementów o nazwie takiej samej jak nasz projekt. Jeżeli skompilujemy dowolny plik źródłowy w projekcie (kod VHDL / kod Verilog / schemat blokowy) to będzie on dostępny w Symbols Toolbox.

- 22. Kliknij na nazwie adder1 w Symbols Toolbox, w dolnej części panelu pojawi się jego ikona.
- 23. Trzy razy przeciągnij na schemat symbol adder1.



Rys. 16. Umieszczenie elementu na schemacie blokowym.

24. Narysuj połączenia między wyjściami Cout oraz wejściami Cin.



25. Wybierz Bus Input Terminal tak jak pokazano poniżej.

Używając tego narzędzia można wprowadzić na schemacie porty wejścia/wyjścia. W rozwijanym menu są różne ikony w zależności od kierunku sygnału (in / out / inout) oraz rodzaju portu (pojedynczy sygnał / magistrala).



Rys. 18. Narzędzia do wstawiania porów na schemacie.

- 26. Umieść na schemacie porty dwóch magistral wejściowych. Naciśnij klawisz Esc aby zakończyć wstawianie portów i przełączyć się do trybu zaznaczania.
- Kliknij dwukrotnie na porcie aby wyświetlić jego właściwości.
   Zmień nazwy portów na A i B. Zmień szerokość magistral na 2 downto 0.

, 1,5 2,0 2,5 i	1,0 · 3,5 ·	4,0 4,5 5,0 5,5 1	6,0 6,5 inch
A(2:0) <u>9</u> . BusInput1(7:0)		U1. +A +B	Cout +-
2.0	Terminal Propert	ies	8 23
	General Attribu	ites   View Texts   Comment	
	Simple Bus	•	
₽ U <u>3</u>	Name:	В	Make Default
	Index range:	2 to 0	<u></u>
с. Сп	Direction:	⊳In	<u> </u>
4- adde	Declaration	9 	
<b>4</b> .	Type name:		
▼ ▼ B I 型	Index range:	to downto	÷
🐮 design flow 🖉 adder1.vhd 💒	Initial value:		
ity "adder1" Analysis time : 0.5 [s]			
			OK Anuluj

Rys. 19. Edycja nazw i szerokości portów na schemacie.

Wstaw Input Terminal (pin wejściowy dla pojedynczego przewodu).
 Zmień jego nazwę na Cin i podłącz go do wejścia Cin lewego, dolnego, pełnego sumatora.



29. Umieść na schemacie Bus Output Terminal (pin wyjściowy dla magistrali). Zmień jego nazwę na Sum a szerokość magistrali na 2 downto 0.



Wstaw Output Terminal (pin wyjściowy dla pojedynczego przewodu).
 Zmień jego nazwę na Cout i podłącz go do wyjścia Cout prawego, górnego, pełnego sumatora.



31. Narysuj przewody tak jak na rysunku 23. Użyj narzędzia Wire dostępnego na prawo od ikony Symbols Toolbox. Kliknij dwukrotnie aby zakończyć rysowanie przewodu.



Rys. 23. Przygotowanie przewodów/połączeń.

Użyj etykiet A(0), A(1), A(2), B(0), B(1), B(2), Sum(0), Sum(1), Sum(2) aby połączyć schemat.
 Dwukrotnie kliknij w przewód aby wyświetlić jego właściwości. W polu Segment wpisz nazwę etykiety.

7	1,5		2,0	)	2,	5	а –	3,0	- 1		8,5	æ	4,0	- 4	4,	5	а.	5,0	12	5,5	зċ	6,0	1	6,5	( a	7	0	63	7,5	зе –	8,0	-36	8,5	<u>.</u>	9,0	- 12	
	• •	:::		• •				1		1		1		1		2.		2.1	11	: ; ;			8	11					1	8			:::				
5			A(2	0)	-				1 . 		102	1.1	-	D	Sun	n(2	0)	51 E 19 - 5	4. T 	U1	1.1	101		1.1	1 1. 7 5.		124		10121		1 1 1 • • • •	11		1.42			
5.			B(2	0)	-													A(2	2)	A	6	Cout		1.1.	-	Col	ıt :										
32				• •				8				•		200		1	-	B(2	2)			Sum	, S	u'm'(	2)			•									8
2.0			- 1962 - 525		1923		19023 1913				1962 525			2				-		I Cin	Ű.					2333 1313	182		1923	8 8				1942			
्य											A	(1)	0	2	20	ion.	1			add	er1																
2,5				: :				100			в	(1)		8	S	out um	. 9	u'm'(	1)		• •																
0 <del>0</del> 320			- 1942 - 1942	11	111	1								Cin			iſ	Wire	Dre	nerti		15323	181 1	36.45	9 83	10-1124	104		10101		1.364	3 8	10101	8	5	8	T
30						ļ	3	69	1: 10 		11	1	a	dde	r1	34 		·····	- FIG	, J	= 3																7
- 9				· B	(0)	,	A	1	Cout	1	Su'n	」. 1(0)						G	enen	al Att	tribu	tes	View	Text	ts   (	Comn	hent	Ap	peara	nce	1					Ĩ	
с.		Cir		2.52	5.62	-,	Cin		5 um	10.00			1.1.			19			Wire	1 3																	
₽÷.					1.2.1	a	Idde	er1					111					-	Net I	Name:		Su	im(1)	_													
-				•														L	Segn	nent:		S	um(1)	1									Make	e Def	ault		
4.5				• •								• •						D	ecla	ration		-															
8 <del>7</del> 329		4.13 	1992					8			1942 1942				6-26-5 1-26-5	8	1803 1813		Туре	name	£.	Г													n		
5,0																			(			Ē															
- 5				• •															ii iilidi	value	8	1															
2				11				8			100	11				1																					
ч.																																					
6.5				: :												1																					
			1932								101					19	1333 575																				
1,0																																					
34				: :								•																									
7,5	: :	111	11		1.1.1			3				11	100			5															OH	<		Ar	uluj		1
	Li li	1.11.1	1.1.	1.1	1.1.1		Rv	s.	24	4.	Тν	vo	rze	eni	ie	po	oła	cze	еń	z w	/vł	kor	zvs	sta	ni	em	١e	tv	kie	t.							创

33. Wybierz z menu Design -> Compile.



Rys. 25. Zawartość Design Browser po kompilacji schematu.

# Tworzymy schemat Top

- Postępując podobnie jak w punktach 18 20 stwórz kolejny schemat blokowy o nazwie top.
   Umieścimy na nim sumator 3-bitowy oraz dekoder wyświetlacza 7-segmentowego.
- 35. Umieść na schemacie "top" symbol 3-bitowego sumatora odszukany w Symbols Toolbox.



Rys. 26. Edycja schematu top.

36. Na prawo od sumatora narysuj FUB.

Naciśnij klawisz Esc aby zakończyć wstawianie FUB i przełączyć się do trybu zaznaczania.



37. Kliknij prawym przyciskiem myszy na FUB i wybierz Properties. W polu "Fub name:" zmień nazwę z **Fub1** na **hex2led**. 38. Wybierz narzędzie Bus i narysuj magistralę tak jak pokazano na Rysunku 28. Rysowanie magistrali zakończ na elemencie FUB tak by na jego krawędzi powstał pin wejściowy.



39. Dwukrotni kliknij na magistrali aby wyświetlić jej właściwości.W polu "Segment" wpisz HEX(3:0) i kliknij OK aby zatwierdzić zmiany.

Nazwa pinu na elemencie FUB automatycznie przyjęła nazwę dołączonej magistrali.

40. Wykorzystując narzędzia Wire (pojedynczy przewód), Bus (magistrale) oraz etykiety, połącz sumator z FUB.



Umieść na schemacie Bus Output Terminal (pin wyjściowy dla magistrali).
 Zmień jego nazwę na LED a szerokość magistrali na 6 downto 0.
 Narysuj magistralę od FUB do pinu LED.

		•	•			٠.	• •		1	٠	•	÷	٠		٢.	•	•		•	•	•	•	•	•		•			•		• •	÷	•	•	 •	÷	÷	٠	•	×.	•	•	1	•		•
e	•	•	•									٠.	•	•	•	•					٠.	•	•	• •		•			•	•				•			٠.		•		•	•	•			
e - 1		•																		L		•												•												
											Υ.	χ.								н	F	X	3.	0)													÷									
																				г		$\sim$	Č.,	×4.																						
											2	2		2	2					L	2	2	2											2			2		2	2	2		2			
				11	4						÷.	0		Ċ.,	۰.					L	÷.			1.14	÷.,				÷.	2									0	0			2			
		•		U	1.					1		۰.	÷.,		۰.					L		÷.,		U2	2				۰.			1		•				1		۰.						
	•	•	•	1				-	-	-			•	•	•					L		•	•											1	 1				•	۰.		•	٠.			
		•	•	· 1							L		۰.	11	4 F	° X	(3	١.		L			•																				•			100
0.		•		-	A (2	2:01			C	o ut	H	-					(0	/	-	1																										
																				1																										
			•	. 1										·U	<b>C</b>	$\mathbf{v}$	2.1	n 'e				•	•																							
	1	1	1		B (2				. (2	-01	L	1	1	'H	Ε.	Х(	2:1	0)		L	1	1		н	EX	(3.1	n			÷.	ED	(6)	01		 1	1	1	1		÷.	Ś	ù	Ė	Ġ(	6	<sup>o</sup>
				+	B (2	2:0)	s	un	n (2	:0)	)=	÷		1H	Ε.	X(	2:1	0)	-	t				н	EX	(3:(	)			L	ED	(6:	0)		 -	-	1	-	ł	Ċ	Þ	Ė	Ė	D(	6	0)
				+	B (2	2::0)	S	un	n (2	:0)	)-,	-	•	°H	Е. -	X() · · ·	2:1	0)	÷	-	÷			н	ΕX	(3:1	))			L	ED	(6:	0)		 -	1	ł	÷	ł	C	Ż	Ļ	Ė	Þ(	6	0)
				≠=	B (2 C In	2:0)	S	un	n (2	:0)	)  -,		ł	îH C	Е. :	×(	2:1	0) :	ł	-	i.	i.		н	EX	(3 :	)			L	ED	(6:	0)		1	ł	ł	ł	1	Ē	Ś	Ė	Ė	D(	6	0)
				• • • •	B (2 C In	2::0)	s	un	n (2	:0)	),		1	1H	E.	X()	2:1	0) : :	ł	ļ	1	1		н	EX	(3 :	))			L	ED	(6:	0)		1	1	1	1	1	0	>	Ė.	Ē	D(	6	0)
				• • • •	B (2 C In	2:0)	S	un	n (2	:0)	)=		1	1H	E.	X(	2:1	0)			-	-		н	EX	(3 :(	)			L	ED	(6:	0)		-	1		1	-	Ē		ĖI	Ė	D(	6	0)
				,+ ;}	B(2 Cin	er	s 3 -	un	n (2	:0)	) <b></b>			1H	E.	X(	2:1	0) - - - -	-		-			hе	e X	(3 :: 2] e	) ed			L.	E D	(6:	0)			-	-			-	>	ĖI	Ė	D(	6	0)
				+ +	в (2 С In dd	er	3 :	: un	n (2	:0)	) <b>-</b>			1H	E.	×(	2:1	0) - - - - -	-					hе	e X	(3 :: 2  <del>(</del>	ed	:	:	۰ (	E D	(6:	o)								>	ÈI	Ē	D(	6	0)
				+ 	в (a cin dd	er	3 -	un	n (2	:0)	) <b>-</b> ,			1H	E.	×(	2:1	0) 						hе	ex x2	(3 :: 21 e	ed	:			E D	(6:	o)								>	ÈI	Ē	D(	6	0)
				+ +	в (2 с in dd	er	3 ·	: un	n (2	:0)	, <b>-</b> ,			1H	E.	×(	2:1	•						•н	ex X2	(3 :: 2] <del>(</del>	ed				E D	(6:	o)								>	È.	Ė	D(	6	0)

Rys. 30. Pin wyjściowy LED(6:0).

42. Umieść na schemacie Bus Input Terminal dla pinów A(2:0) oraz B(2:0) a także Input Terminal dla pinu Cin. Połącz je z symbolem sumatora.

	<b>HFX</b> (3)	0)	 
a construction of the construction		112	 
· · · · · · · · · · · · · · · · · · ·		02	 
	· · · · · · · · · · · · · · · · · · ·		 
- A(2:0) Cout	HEX(3)		 
	· · · · · · · · · · · · · · · · · · ·		 
· B(2:0) Sum (2:0)	HEA(2.0)	HEX(3:0) LED(6:0)	ED(6:0)
	·		
····Cin			 
adder3		hex2led · · · · · ·	 
D 114 D.			

- 43. Kliknij FUB prawym przyciskiem myszy i wybierz Push z menu kontekstowego.
- 44. W oknie Create New Implementation kliknij na VHDL Source Code by wskazać że działanie tego komponentu zostanie zaimplementowane w języku VHDL.

plementation			
hex2led			
ype:			
<b>‡</b> D-	문		
Block Diagram	EDIF Netlist	State Diagram	VHDL Source Code
<u> </u>			
d:\My_Designs\Zaoczni	_Lab6_8_00\BDE_Tut	orial\src\hex2led.vhd	Browse

Rys. 32. Wybór sposobu implementacji elementu FUB.

45. Uzupełnij kod dekodera wyświetlacza siedmiosegmentowego.

W tym celu:

- umieść kursor w linii 41 pliku hex2led.vhd;
- na głównym pasku menu wybierz Tools -> Language Assistant;
- w oknie Language Assistant rozwiń Synthesis templates i odszukaj HEX2LED Converter;
- kliknij prawym przyciskiem myszy na HEX2LED Converter i wybierz Use;
- zamknij Language Assistant.

Na Rysunku 33 przedstawiono zawartość pliku hex2led.vhd po wprowadzonych zmianach.

```
25 library IEEE;
   use IEEE.STD_LOGIC_1164.all;
26
27
28
   entity hex2led is
       port (
29
             HEX : in STD LOGIC VECTOR (3 downto 0);
30
31
             LED : out STD_LOGIC_VECTOR(6 downto 0)
32
             );
33
    end hex2led;
34
35
     --}} End of automatically maintained section
36
37
    architecture hex2led of hex2led is
38 begin
39
40
         -- enter your statements here --
41
            --HEX-to-seven-segment decoder
42
            -- HEX: in STD LOGIC VECTOR (3 downto 0);
43
            -- LED: out STD_LOGIC_VECTOR (6 downto 0);
44
            ___
45
            -- segment encoding
46
            ___
                0
47
            ___
                   ____
48
            -- 5 | | 1
49
            ___
                  ____
                        <- 6
50
            -- 4 | | 2
51
            ---
                   ____
52
                    з
            ___
53
54
            with HEX select
55
            LED<= "1111001" when "0001",
                                           --1
                    "0100100" when "0010",
                                           --2
56
                    "0110000" when "0011",
57
                                           --3
58
                    "0011001" when "0100",
                                           --4
59
                    "0010010" when "0101",
                                           --5
                    "0000010" when "0110",
60
                                           --6
                    "1111000" when "0111",
61
                                           --7
                    "0000000" when "1000",
62
                                           --8
63
                    "0010000" when "1001",
                                           --9
                    "0001000" when "1010",
64
                                           --A
65
                    "0000011" when "1011",
                                           --b
                    "1000110" when "1100",
66
                                           --C
67
                    "0100001" when "1101",
                                           --d
                    "0000110" when "1110",
68
                                           --E
69
                    "0001110" when "1111", --F
70
                    "1000000" when others; --0
71
    end hex2led;
```

Rys. 33. Implementacja dekodera wyświetlacza siedmiosegmentowego w języku VHDL.

46. Skompiluj cały projekt. Wybierz z głównego menu programu Design -> Compile All. Jeśli kompilacja zakończy się poprawnie to w Design Browser będzie można zobaczyć następującą strukturę projektu.



Rys. 34. Zawartość okna Design Browser po kompilacji całego projektu.

# Generowanie Testbenchu

47. W Design Browser kliknij prawym przyciskiem myszy na parze top(top) i wybierz Generate TestBench z menu kontekstowego.



Rys. 35. Generowanie Testbench dla wskazanej pary entity(architecture).

48. W pierwszym oknie wybierz Single Process dla Testbench Type.

bench Generator Wizard				_
Select the design unit for wh	iich you want t	o generate a	testbench. Th	e wizard will
generate appropriate source	files and a ma	icro file for the	e testbench.	
Entity:				
top				•
Angle in a star and a second				
Architecture:				
top				•
Testbench Type:				
Single Process				
O WAVES Based				
C Aldec TestBench				
2.51	/stecz	)alei >	Acadati	Pomoc

### Rys. 36. Wybór rodzaju testbenchu.

### 10

49. Nie będziemy wczytywać wektorów testowych z żadnego pliku, więc w kolejnym oknie wybierz Next.

tbench Generator Wizard		×
Define test vectors.		
☐ Test vectors from file Select this check box if you want saved in a waveform file.	to use previously created test	vectors
Select a test vector file:		
1		Browse
Signals found in file:	UUT ports:	
	A B Cin	
,		
< Wstec	z Dalej > Ani	uluj Pomoc

Rys. 37. Pomijamy import wektorów testowych.

50. W kolejnym oknie pozostaw niezmienione, domyślne nazwy dla entity i architecture. Kliknij Dalej.

ype the name of	the testbench entity	y:	
op_tb			
ype the name of	the testbench archi	itecture:	
TB_ARCHITECT	URE		
vpe the name of	the testbench soun	ce file:	
op_TB.vhd			
			Browse
pe the name of	the folder for testbe	nch files:	
[estBench			

Rys. 38. Okno w którym można wskazać m.in. nazwy entity i architecture dla generowanego testbenchu.

51. Nie wprowadzaj zmian w kolejnym oknie i zakończ generowanie testbenchu.

estbench Generator Wizard	X
The wizard is ready to generate testbench files.	
The following files will be generated:	
Testbench file(s):	
.\src\TestBench\top_TB.vhd	
File with configuration for timing simulation:	
.\src\TestBench\top_TB_tim_cfg.vhd	
Generate	
- Simulation macm (DO file):	
\sm\TestBench\ton_TB_nuntest_do	
(Webers Zelezán Arubii De	
< vvstecz Zakoncz Anuluj Po	moc

Rys. 39. Ostatnie okno narzędzia do generowania Testbenchu.

52. Dołóż wymuszenia w wygenerowanym Testbenchu.

W tym celu:

```
- wyświetl zawartość pliku top_TB.vhd (plik znajduje się w Design Browser, w folderze TestBench);
- po linii -- Add your stimulus here... wstaw poniższy kod:
              Cin <= '0';
              A<="010", "111" after 200 ns, "011" after 400 ns;
              B<="011", "001" after 200 ns, "001" after 400 ns;
-skompiluj plik testbencha.
       30
                 -- Unit Under Test port map
       31
                 UUT : top
       32
                     port map (
       33
                          Cin => Cin,
       34
                          A \Rightarrow A,
       35
                          B => B,
                          LED => LED
       36
       37
                     );
       38
       39
                 -- Add your stimulus here ...
       40
                 Cin <= '0';
       41
                 A<="010", "111" after 200 ns, "011" after 400 ns;
       42
                 B<="011", "001" after 200 ns, "001" after 400 ns;
       43
       44
            end TB ARCHITECTURE;
       45
             configuration TESTBENCH_FOR_top of top_tb is
       46
       47
                 for TB ARCHITECTURE
       48
                     for UUT : top
       49
                          use entity work.top(top);
       50
                     end for;
```

```
Rys. 40. Edycja kodu VHDL w pliku top_TB.vhd.
```

53. W folderze TestBench w Design Browser, oprócz pliku VHDL z kodem testbencha znajduje się makro o nazwie top\_TB\_runtest.do, które służy do automatycznej kompilacji projektu i uruchamiania symulacji. Wyświetl zawartość makra i dołóż na końcu:

#### run 600 ns

```
SetActiveLib -work
     comp -include "$DSN\compile\top.vhd"
     comp -include "$DSN\src\TestBench\top TB.vhd"
     asim TESTBENCH_FOR_top
4
     wave
6
     wave -noreg Cin
7
     wave -noreg A
     wave -noreg B
     wave -noreg LED
10
     # The following lines can be used for timing simulation
11
     # acom <backannotated vhdl file name>
12
     # comp -include "$DSN\src\TestBench\top_TB_tim_cfg.vhd"
13
     # asim TIMING FOR top
14
    run 600 ns
15
      Rys. 41. Edycja makra służącego do uruchomienia symulacji.
```

- 54. Zapisz zmiany wprowadzone w treści makra.

## Symulacja Projektu

55. W Design Browser kliknij prawym przyciskiem myszy na top\_TB\_runtest.do i wybierz Execute z menu kontekstowego.



Rys. 42. Uruchomienie makra.

56. Zweryfikuj poprawne działanie układu.

Magistrala LED to sygnały sterujące wyświetlaczem siedmiosegmentowym (to nie jest wynik dodawania w kodzie naturalnym binarnym).

Segment wyświetlacza jest zapalany poziomem niskim (wynika to z rodzaju użytego wyświetlacza – wspólna Anoda).

Name	Value	Sti	i - 50 -	74.05	150	1 2	. 00	250	т	300	i 3	3 <b>5</b> 0	ı.	4 <u>0</u> 0	ı.	450	ı.	500	ı.	550	ı.	6 <u>0</u> 0
🏧 Cin	0			74 113																		
±лгд	2		(2				χ7							<u>X</u> 3								
т. В	3		(3				χ1															
🖃 🏧 LED	12		(12				X00							X19								
™ LED(6)	0																					
™ LED(5)	0																					
۳ LED(4)	1						1															_
™ LED(3)	0																					_
™ LED(2)	0																					
™ LED(1)	1						1															
™ LED(0)	0																					_

Rys. 43. Wynik symulacji projektu.