



Kierunek Elektronika i Telekomunikacja, zaoczne, IV rok  
**Projektowanie Systemów Cyfrowych  
w Językach Opisu Sprzętu**

# Wprowadzenie



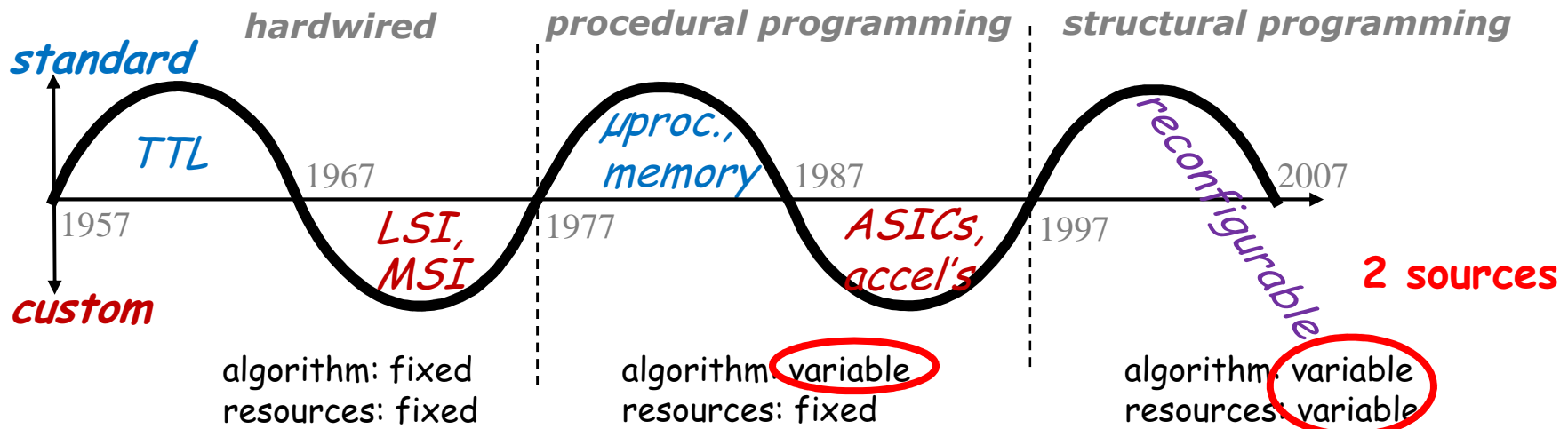
# Kontakt z prowadzącymi

dr inż. Paweł J. Rajda [pjrajda@agh.edu.pl](mailto:pjrajda@agh.edu.pl)  
dr inż. Jerzy Kasperek [kasperek@agh.edu.pl](mailto:kasperek@agh.edu.pl)  
<http://www.embedded.agh.edu.pl>  
C-3, p.502, tel. (12) 617 3980



- 1. VHDL – po co?**
- 2. VHDL – co to jest?**
- 3. VHDL – jak, gdzie, kiedy?**
- 4. VHDL – i co z tego?**
- 5. VHDL – wczoraj, dziś i jutro**
- 6. VHDL – standardy**
- 7. VHDL – literatura**
- 8. VHDL – źródła**
- 9. VHDL – firmy**
- 10. VHDL – Aldec: Active HDL**

# Makimoto's Wave Technologiczne wahadło



Single chip  $\mu$ C circa 1974



TI TMS 1000  
Fairchild F8  
Intel 8048  
Mostek 3870  
etc.

40 pins  
10,000 gates  
10,000 RAM bits  
1MHz clock

Single chip FPGA circa 2004



Xilinx  
Altera  
Actel  
etc.

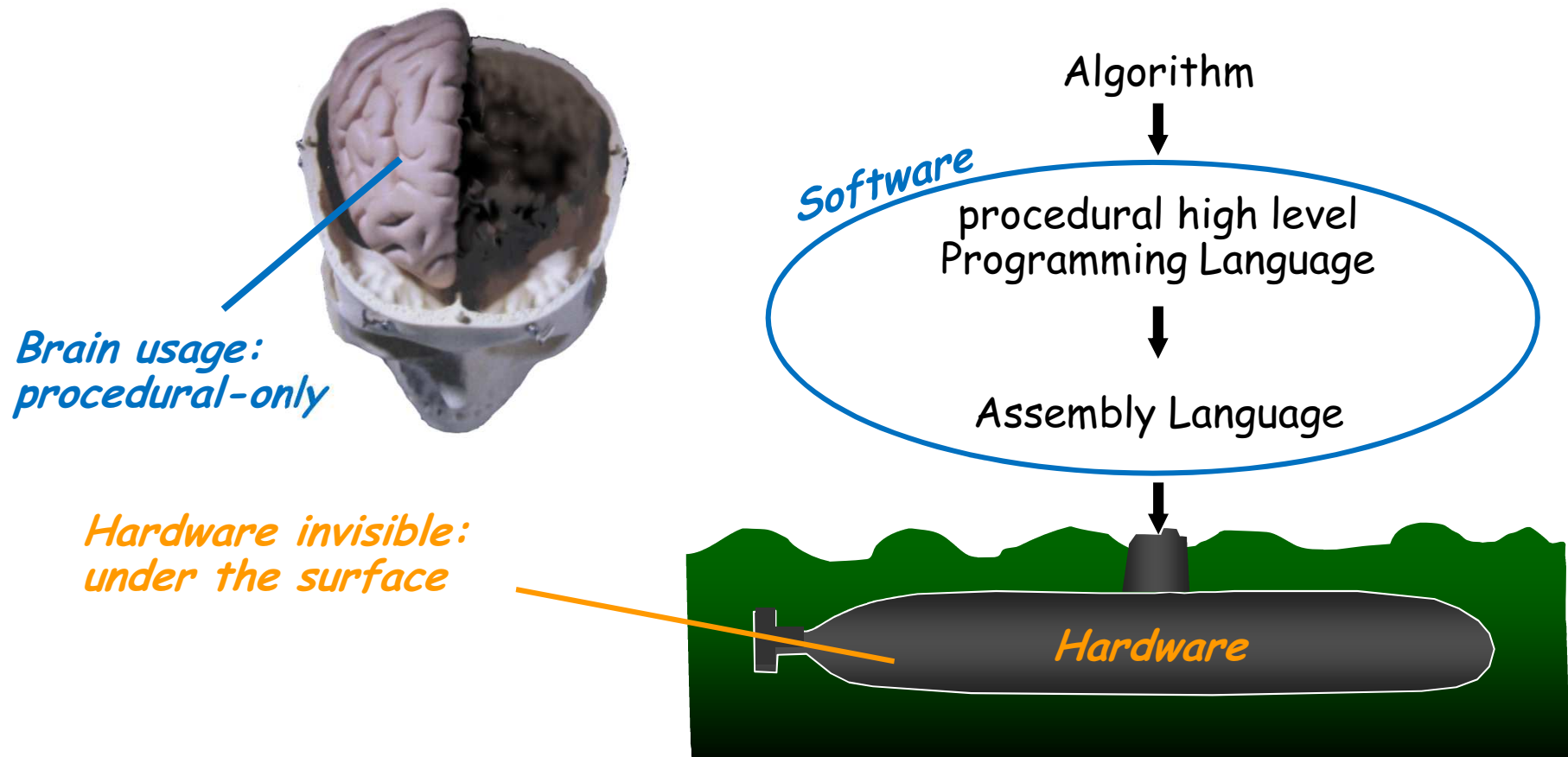
200+ pins  
10,000,000 gates  
10,000,000 RAM bits  
100MHz clock

vN machine paradigm

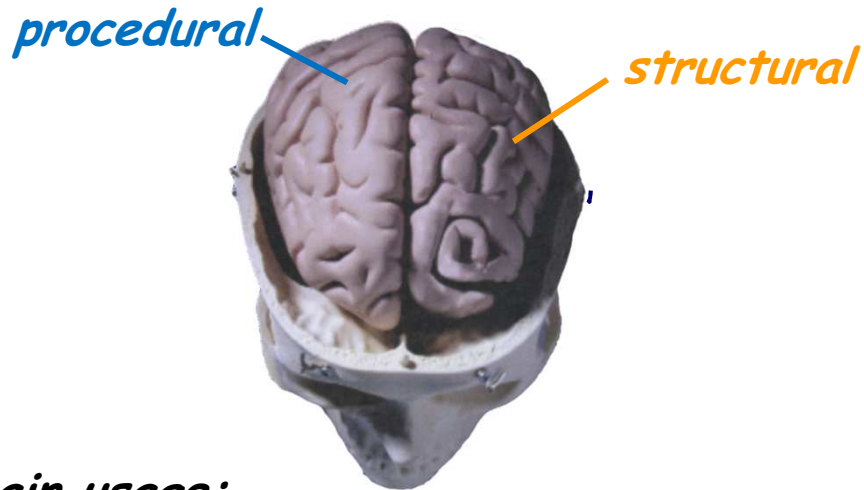
new machine paradigm needed

**The Programmable System-on-a-Chip  
IS THE NEXT WAVE**

# Dotychczasowy model systemu

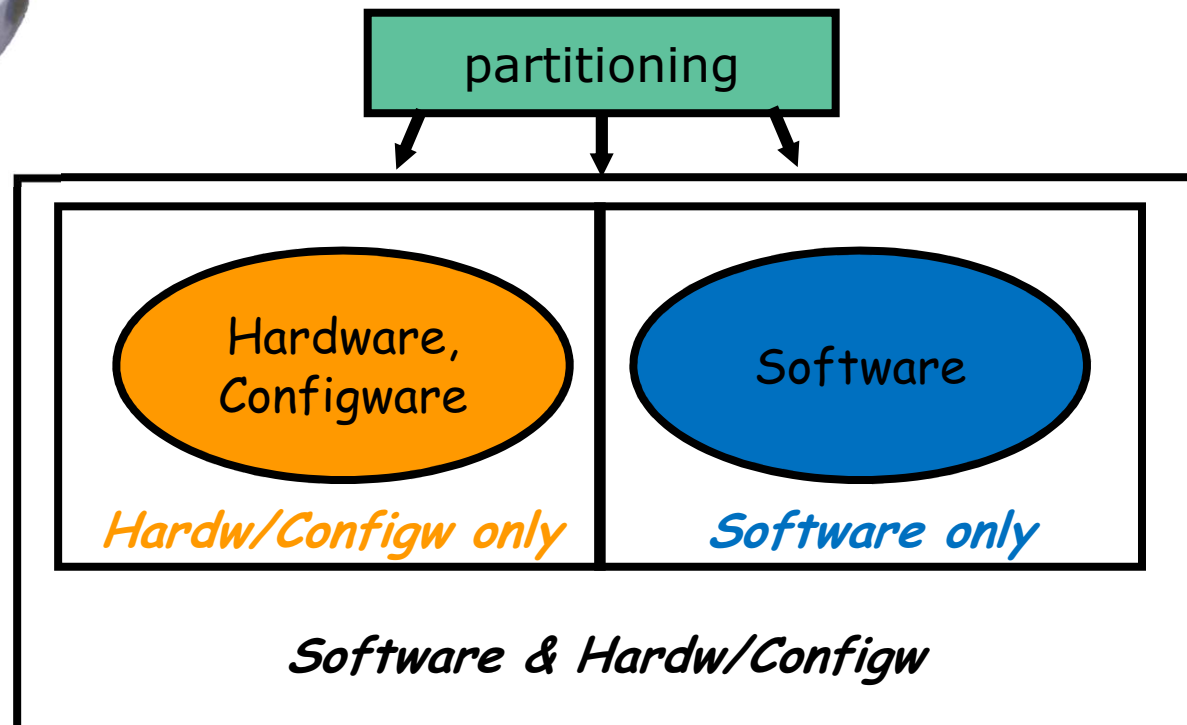


# Nowy model systemu

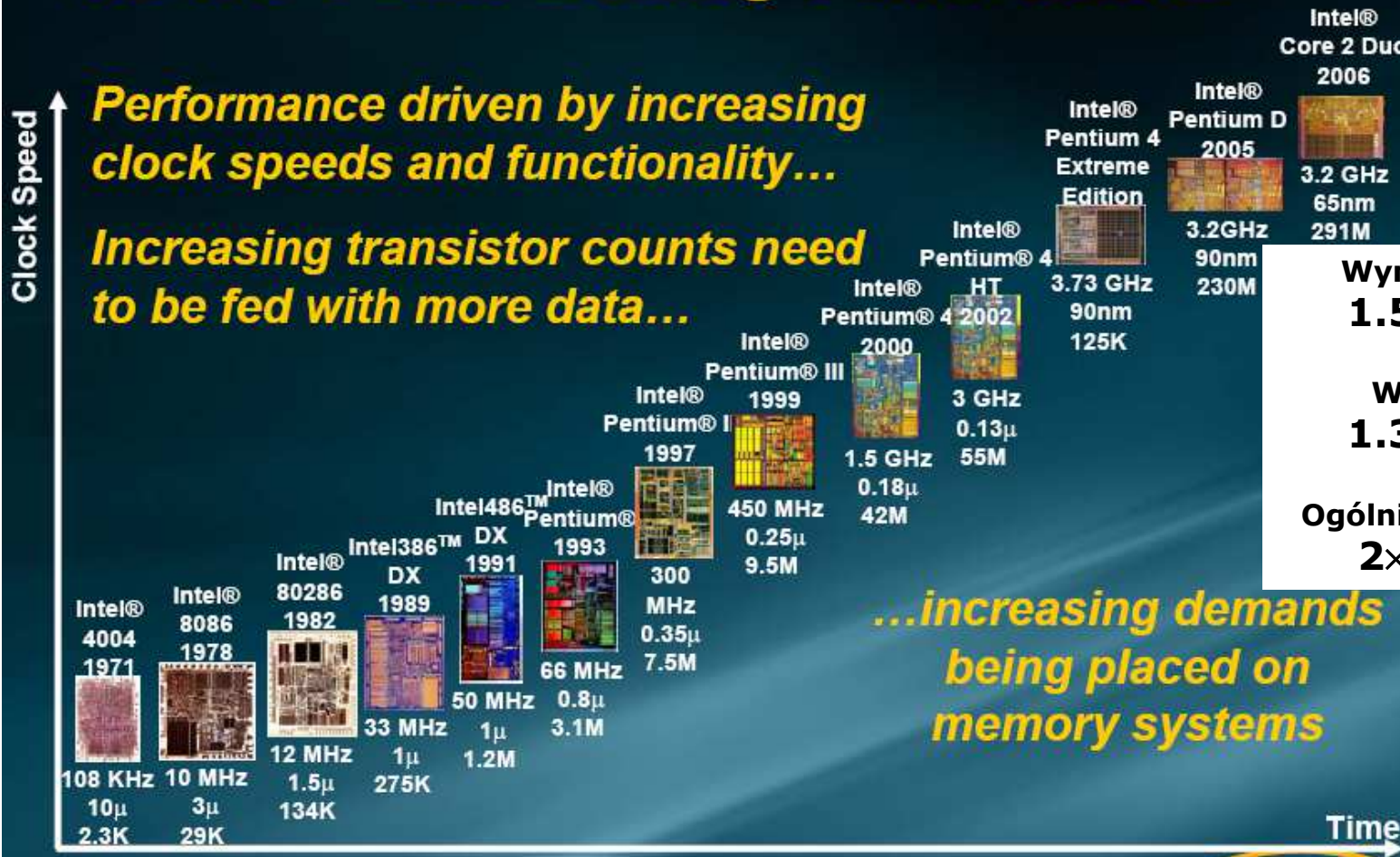


*Brain usage:  
both hemispheres*

## Algorithm



## Moore's Law Driving Performance



Wymogi pamięci:  
**1.50× na rok**

Wymogi CPU:  
**1.35× na rok**

Ogólnie prawo Moora:  
**2× co 2 lata**

## Potrzeba narzędzia:

• 1970 - INTEL 4004	4 projektantów	1 tys tranzystorów
• 1982 - INTEL 80286	20 projektantów	100 tys tranzystorów
• 1992 - INTEL PENTIUM	100 projektantów	3 mln tranzystorów
• 2003 - ???	1000 projektantów	150 mln tranzystorów

Współczesne wymagania:

- *design reuse*
- *hardware-software codesign*



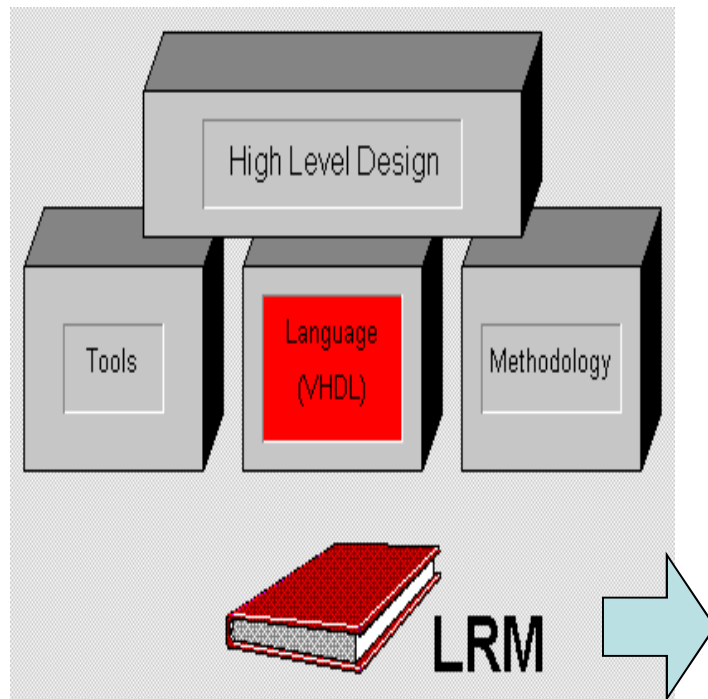
Rozwiązanie:  
**wykorzystanie języka HDL  
(VHDL, Verilog, ...)**



- **PALASM**
- **ABEL**
- **CUPL**
  
- **VHDL**
- **VERILOG, System VERILOG**
- **C, C++, Handel-C, System-C**
- **inne**

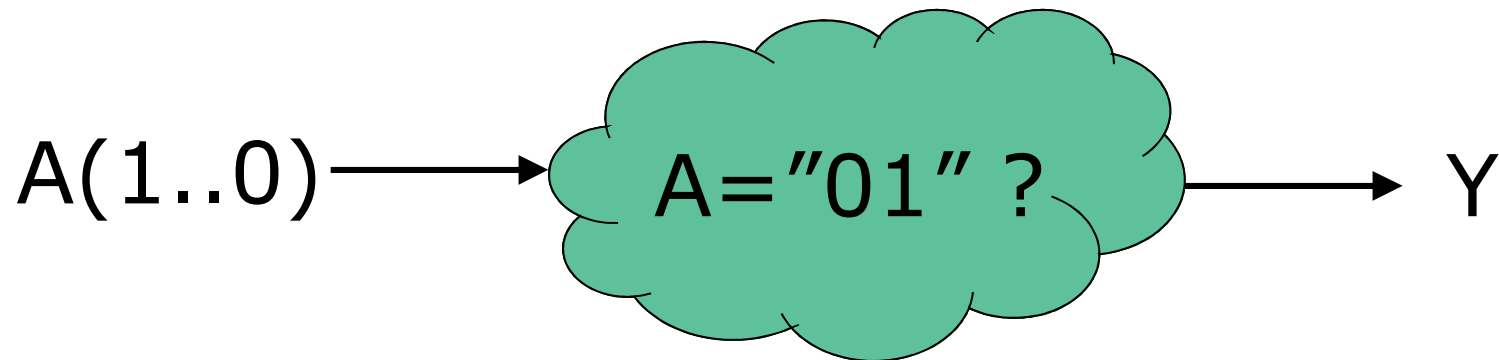
# VHDL - V<sub>HSIC</sub> H<sub>ardware</sub> D<sub>escription</sub> L<sub>anguage</sub>

↳ **V**ery **H**igh **S**peed **I**ntegrated **C**ircuit

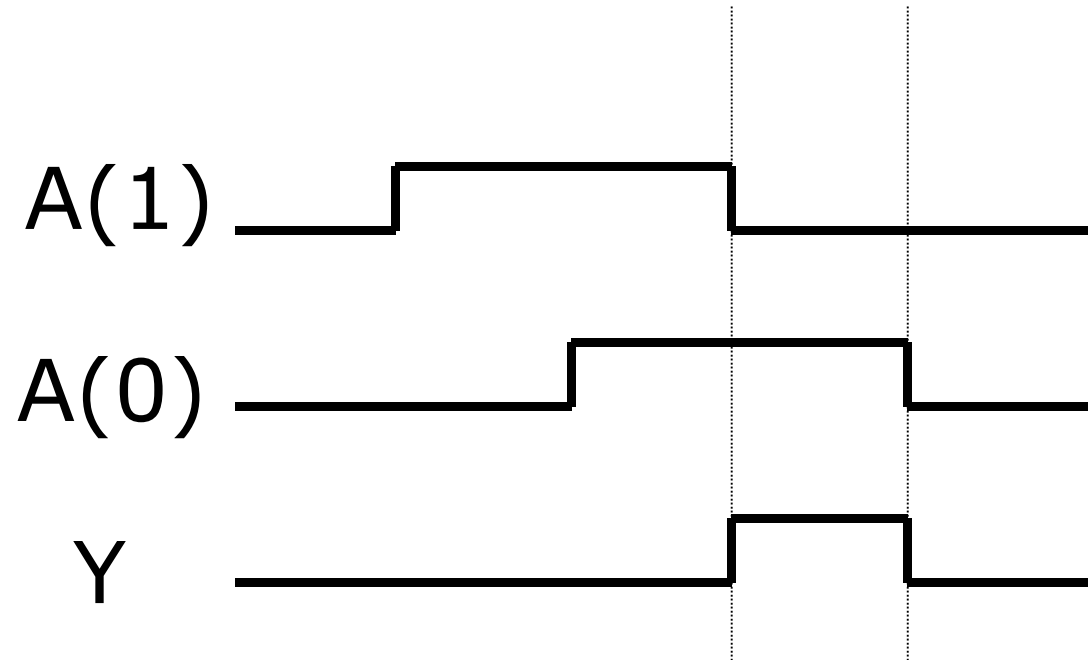


*It is "a formal notation intended for use in all phases of the creation of electronic systems. (...) it supports the development, verification, synthesis, and testing of hardware designs, the communication of hardware design data ..."*

*[IEEE Standard VHDL Language Reference Manual]*

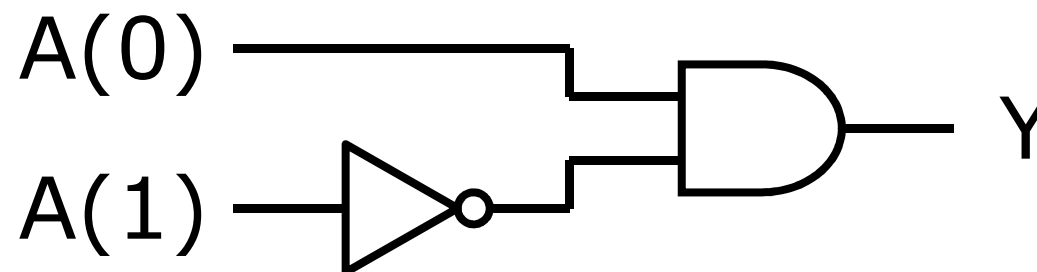


```
Y <= '1' when A = "01" else '0' ;
```



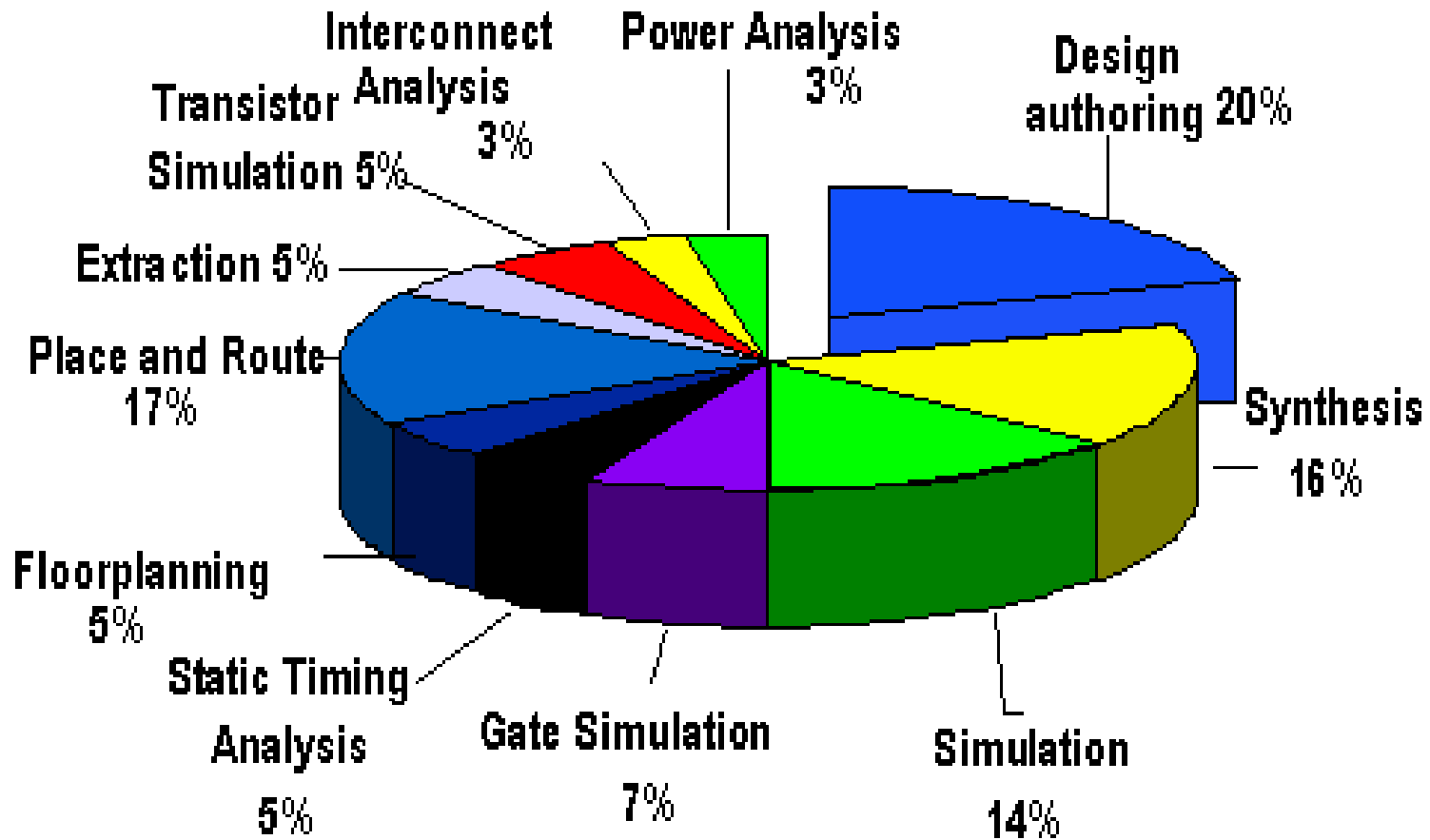
```
Stim <= "10", "11" after 4 ns ...
```

**Synteza** – (automatyczna) translacja opisu w języku HDL na strukturę w postaci listy połączeń elementarnych bloków funkcyjnych docelowej platformy sprzętowej (bramek, przerzutników, pamięci i innych).



# VHDL – jak, gdzie kiedy?

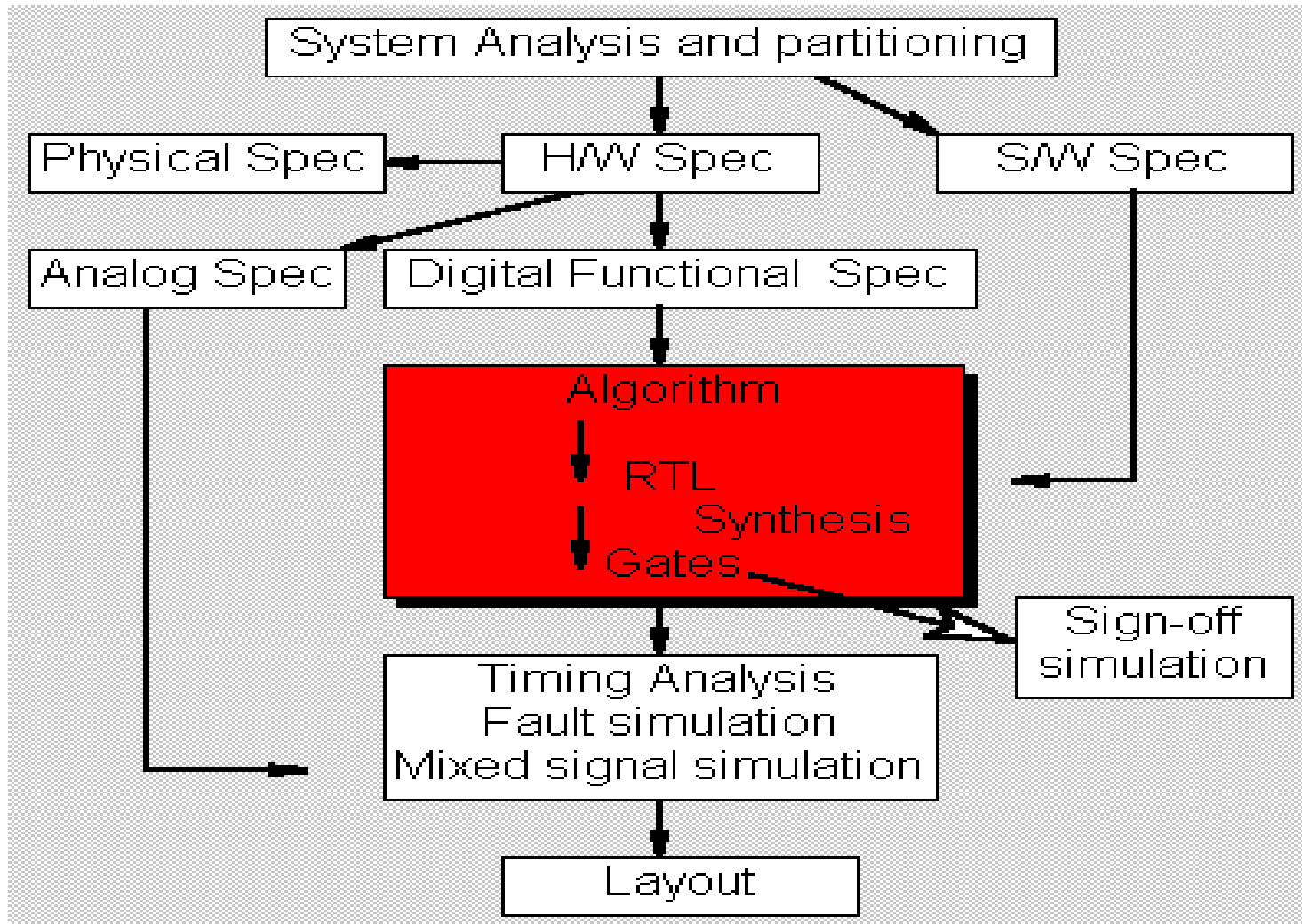
## Etapy projektu



Average iterations between design and layout = 20

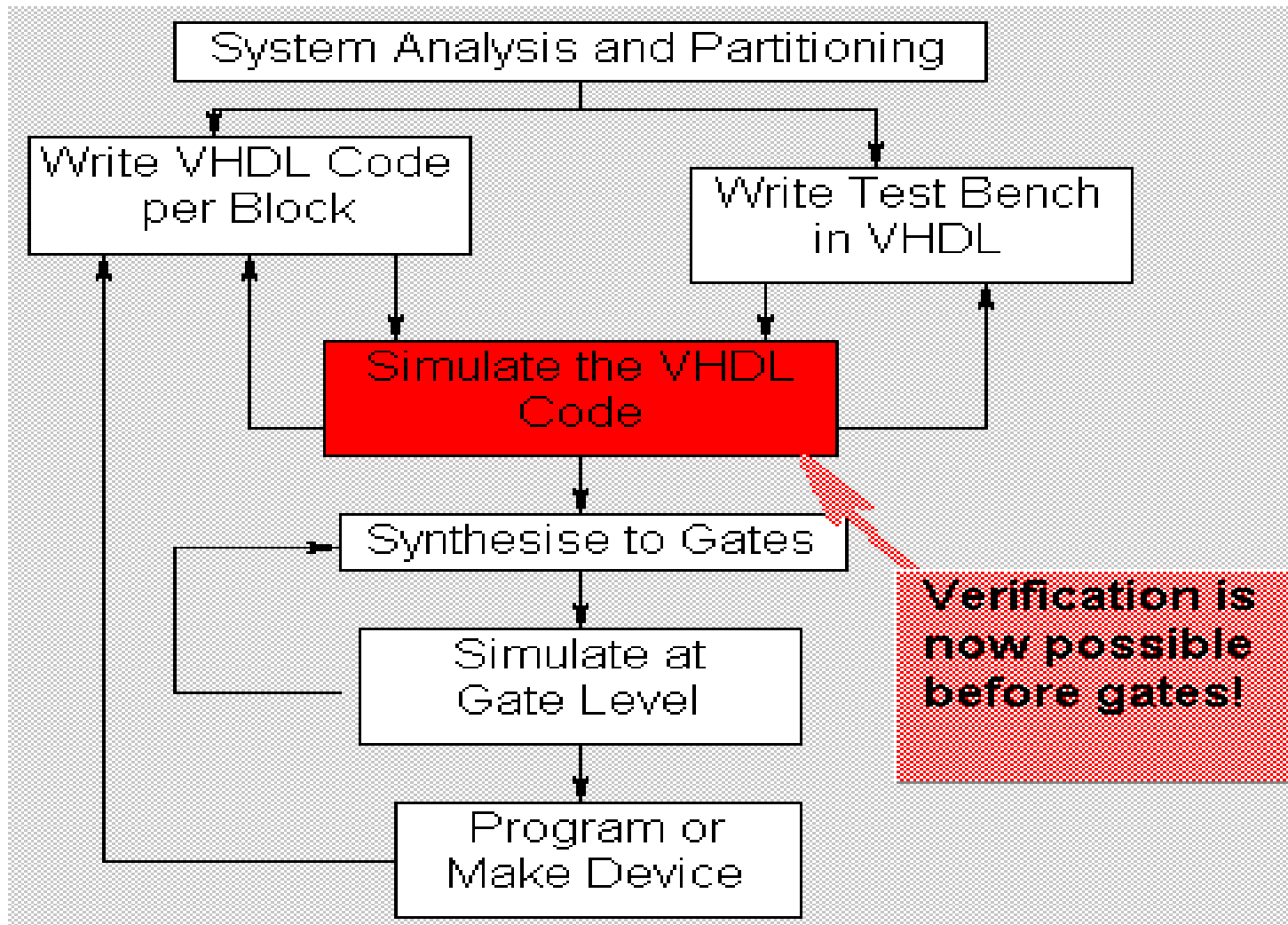
# VHDL – jak, gdzie kiedy?

## Projektowanie systemu



# VHDL – jak, gdzie kiedy?

## Proces projektowania





**entity  
name**

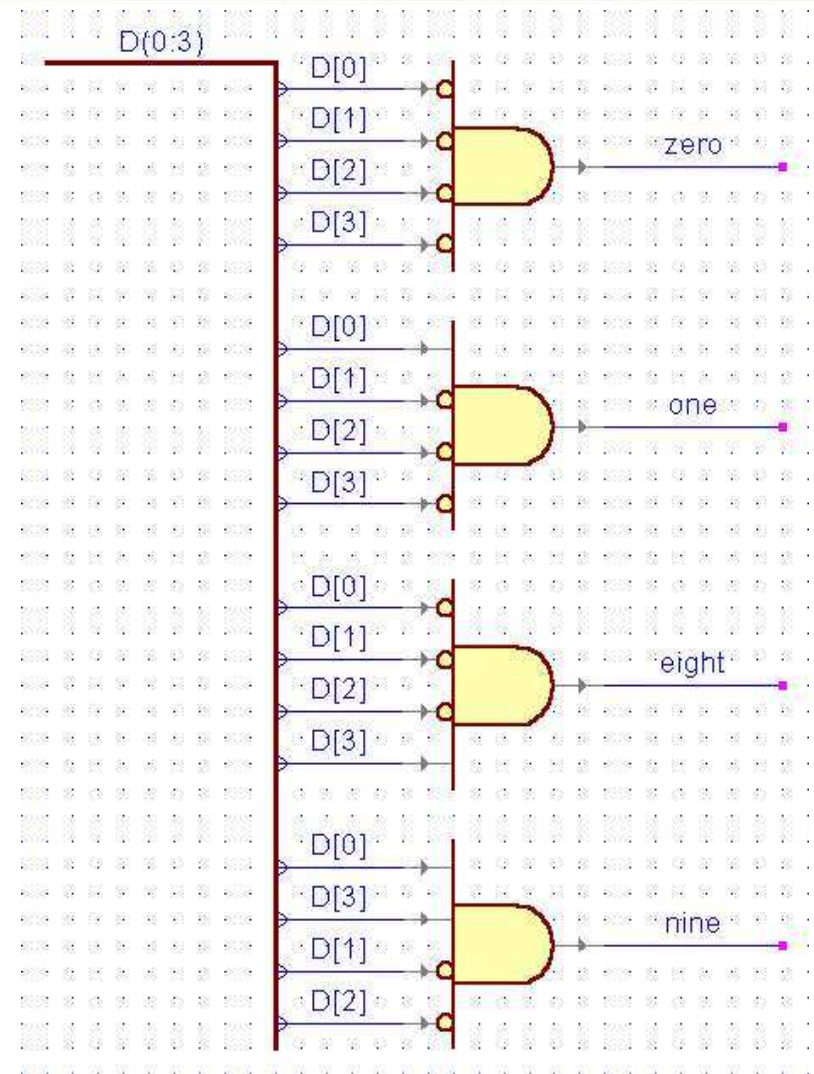
```
entity COMPARE is
    port (A,B: in bit;
          C: out bit);
end COMPARE;
```

**architecture  
style of name**

```
architecture BEHAVIORAL of COMPARE is
begin
    process (A,B)
    begin
        if (A=B) then
            C <= '1';
        else
            C <= '0';
        end if;
    end process;
end BEHAVIORAL;
```

```
entity DECODER is
  port(D: in bit_vector (0 to 3);
        ZERO: out boolean;
        ONE: out boolean;
        EIGHT: out boolean;
        NINE: out boolean);
end DECODER;
```

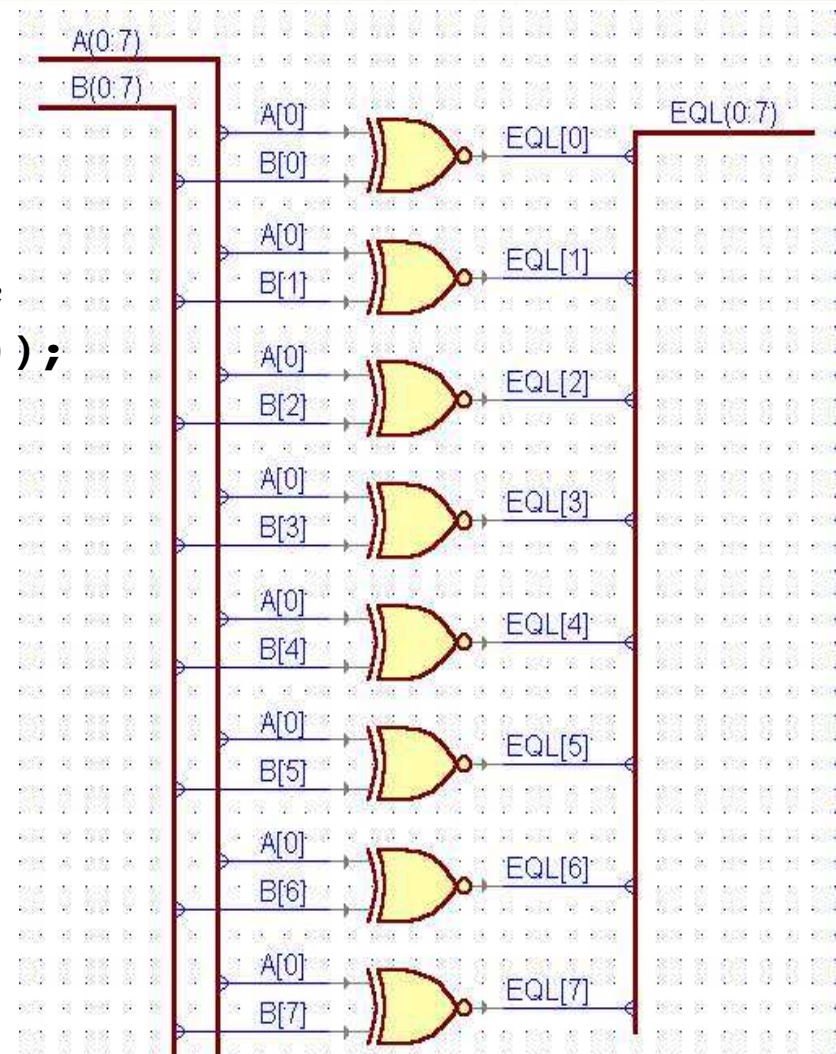
```
architecture FIRST of DECODER is
begin
  ZERO <= (D="0000");
  ONE <= (D="0001");
  EIGHT <= (D="1000");
  NINE <= (D="1001");
end FIRST;
```



```
entity COMPARE is
  port(A,B: in bit_vector (0 to 7);
        EQL: out bit_vector (0 to 7));
end COMPARE;
```

```
architecture SECOND of COMPARE is
begin
  EQL <= not (A xor B);
end SECOND;
```

**Błąd ! →**

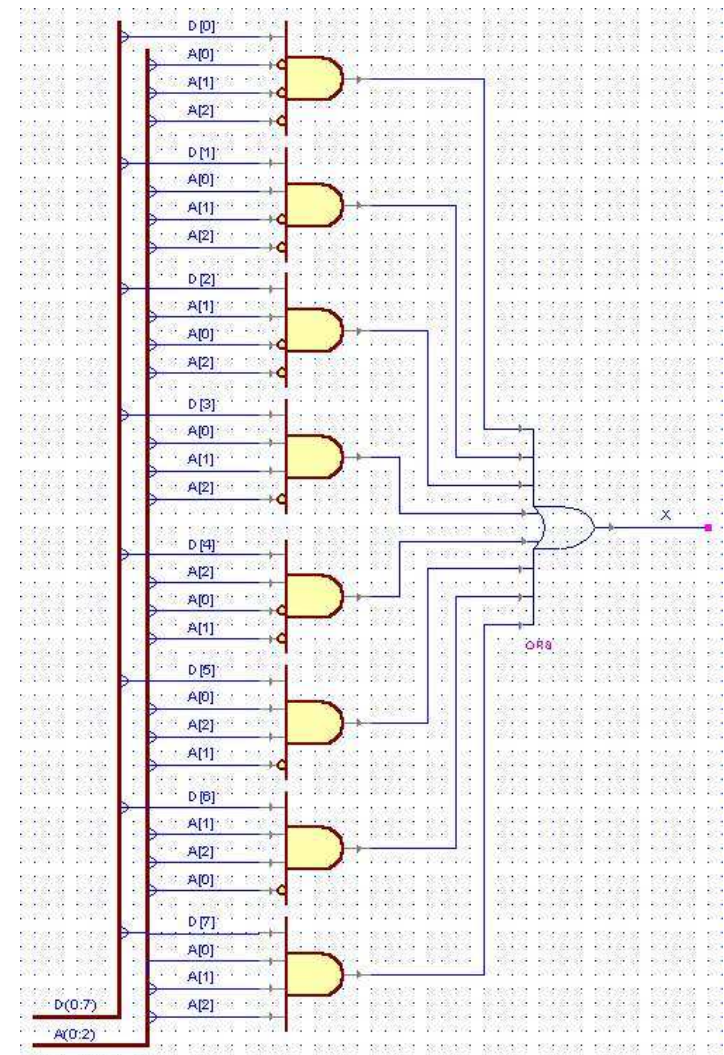


```

entity MPLEXER is
  port(D: in bit_vector (0 to 7);
       A: in integer range 0 to 7;
       X: out bit);
end MPLEXER;

architecture THIRD of MPLEXER is
  begin
    X <= D(A);
  end THIRD;

```



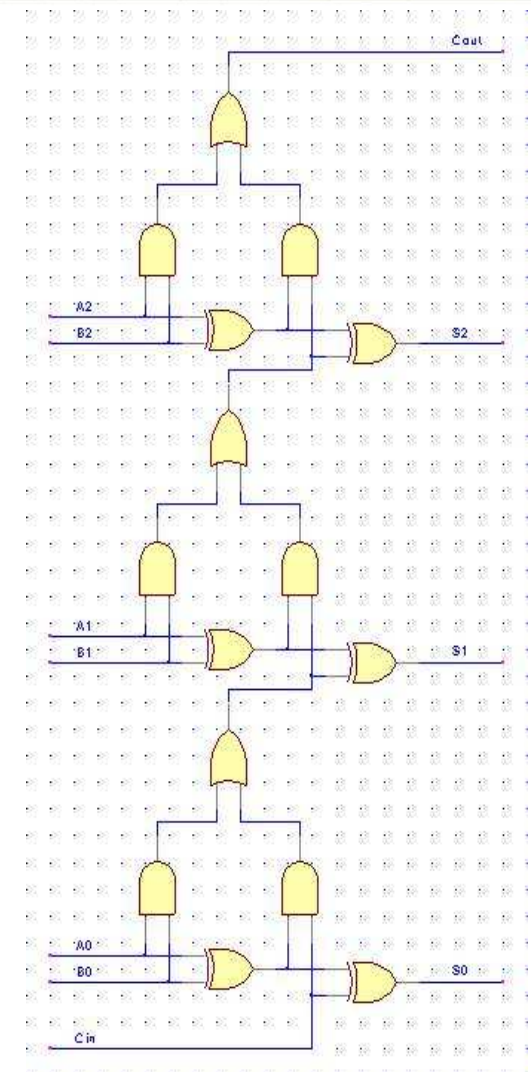
```

use IEEE.STD_LOGIC_UNSIGNED.all;

entity SUM is
  port(A,B: in std_logic_vector (0 to 2);
        Cin: in std_logic;
        S: out std_logic_vector (0 to 2);
        Cout: out std_logic);
end SUM;

architecture FOURTH of SUM is
  signal V: std_logic_vector (0 to 3);
begin
  V <= A + B + Cin;
  S <= V(0 to 2);
  Cout <= V(3);
end FOURTH;

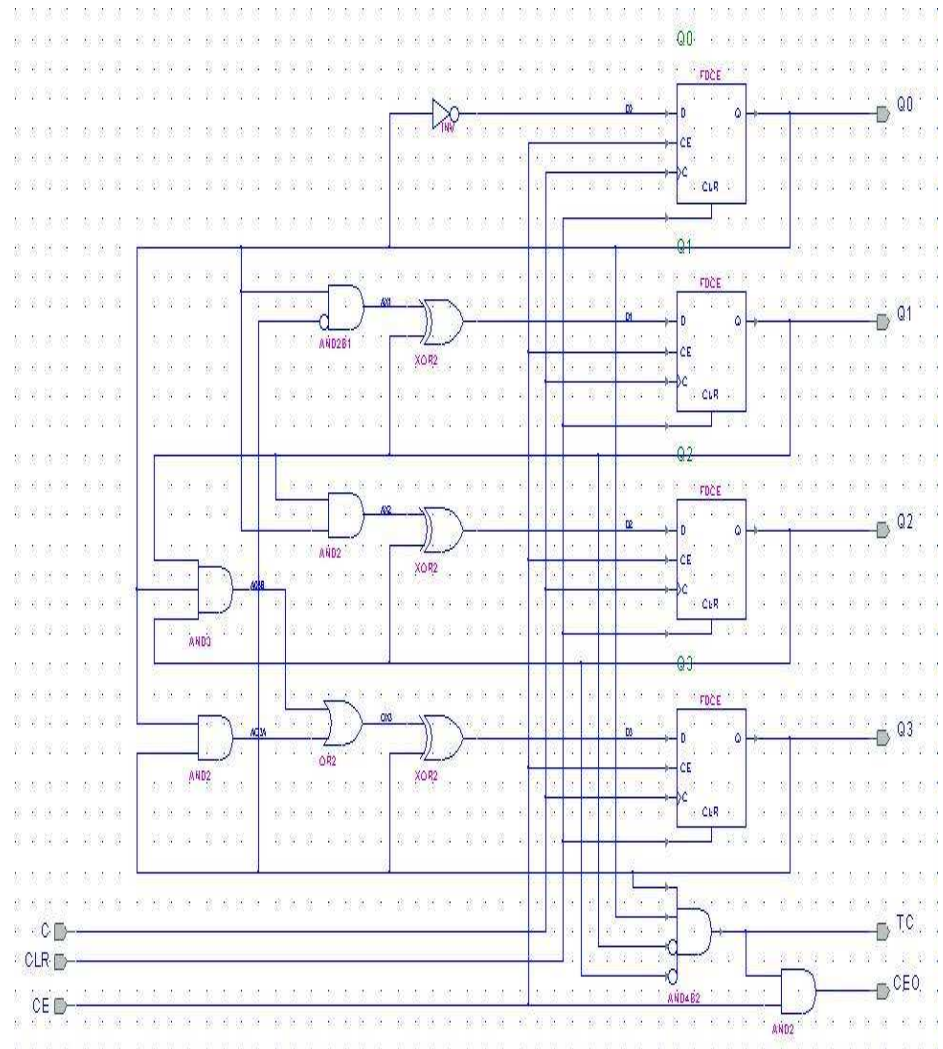
```



```

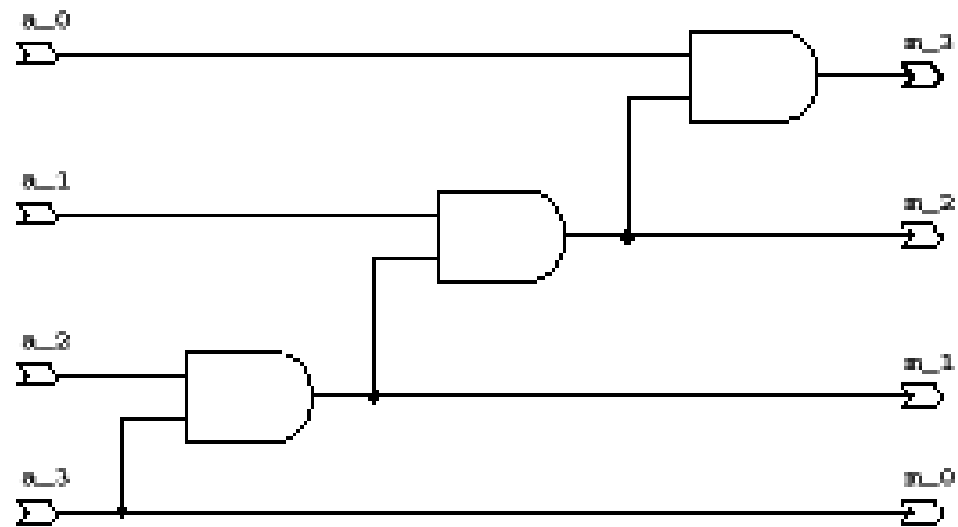
architecture FIFTH of COUNT is
begin
  process (C, CLR)
  begin
    if CLR='1' then
      Q := 0;
    elsif C='1' and C'event then
      if CE='1' then
        if Q = 9 then
          Q := 0;
        else
          Q := Q + 1;
        end if;
      end if;
    end if;
  end process;
end FIFTH;

```



```
entity loop_stmt is
port (a: bit_vector (0 to 3);
      m: out bit_vector (0 to 3));
end loop_stmt;
```

```
architecture example of loop_stmt is
begin
process (a)
  variable b: bit;
begin
  b := '1';
  for i in 0 to 3 loop
    b := a(3-i) and b;
    m(i) <= b;
  end loop;
end process;
end example;
```



- **Specyfikacja projektu niezależna od technologii**
  - **możliwość współpracy z wieloma producentami**
  - **uniknięcie problemów z wycofywanymi technologiami**
  - **łatwe ulepszenia i poprawy**
- **Automatyzacja projektowania niskiego poziomu**
  - **krótszy czas projektowania**
  - **redukcja kosztów**
  - **eliminacja błędów niskiego poziomu**
- **Poprawa jakości projektu**
  - **łatwe sprawdzanie opcjonalnych technologii**
  - **weryfikacja działania na wysokim poziomie**
  - **łatwa weryfikacja implementacji**
  - **modularność projektu – łatwa wymiana do innych celów**





## VHDL – wczoraj

- 1980 DoD USA – początek programu rozwijania technologii układów VHSIC (*Very High Speed Integrated Circuits*)
- 1981 Woods Hole, Massachusetts - konferencja na temat założeń przyszłego standardu HDL
- 1983 DoD ustala założenia VHDL: *VHSIC Hardware Description Language* – kontrakt otrzymują : Intermetrics, TI i IBM
- 1984 gotowa wersja 6.0
- 1985 zwolnienie z restrykcji ITAR (US International Traffic and Arm Regulation), VHDL 7.2 wraz z referencjami przekazany do IEEE celem standaryzacji i dalszego rozwoju
- 1987 wydany opis IEEE Std 1076
- 1993 wydana nowelizacja IEEE Std 1076-1993
- 2000 wydana errata IEEE Std 1076a-1993
- 2003 wydana nowelizacja IEEE Std 1076-2003
- 2005 przejęcie inicjatywy przez Accelera
- 2006 wydana nowelizacja IEEE Std 1076-2006
- 2008 wydana nowelizacja IEEE Std 1076-2008



# VHDL – dziś

## VHDL Analysis and Standardization Group (VASG)

<http://www.eda-twiki.org/cgi-bin/view.cgi/P1076/WebHome>



P1076

Log In or Register

**P1076 Web**

- Create New Topic
- Index
- Search
- Changes
- Notifications
- RSS Feed
- Statistics
- Preferences

**Webs**

- Main
- P1076
- P10761
- P1647
- P16661
- P1685
- P1734
- P1735
- P1778
- P1800
- P1801
- Sandbox
- TWiki
- VIP
- VerilogAMS

TWiki > P1076 Web > WebHome (2016-01-26, StanKrolkoski) Edit Attach

### IEEE P1076 Working Group

#### VHDL Analysis and Standardization Group (VASG)

#### Mission

VASG is responsible for maintaining and extending the VHDL standard (IEEE 1076).

#### Status

VASG is actively working on proposals for P1076 201X. For more information see [Working Group Status](#)

#### Participating

If you are an experienced VHDL user, digital designer, or verification engineer, then this is your working group. Here you can contribute to the future of VHDL.

P1076 is an individual based standard. The only requirement for membership is to show up and participate. Work is done via on this TWIKI site, and the email reflector, in our phone meetings. Opportunities to participate are available at many different levels of commitment and your participation will help. Join us.

- To participate in TWIKI, send email to TWIKI admin: [Jim Lewis](#)
- Email Reflector: \* [View](#) \* [subscribe](#) \* [unsubscribe](#) \* [Send Email](#) (subscribe first)

#### Meetings & Work Areas

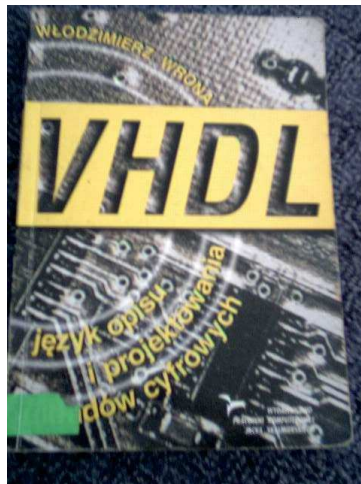
- Czy „sprzętowiec” jest jeszcze potrzebny?
  - zbyt mała wydajność architektury Von Neumanna
  - konieczność kompensacji „niewydajności” oprogramowania przy pomocy dedykowanego sprzętu
- Język: ogólny czy szczególny ?
  - ogólny (RTL)
    - blisko implementacji sprzętowej
    - w projekcie są zawarte szczegóły architektury
  - szczególny
    - blisko aplikacji (np. DSP: Matlab)
    - automatyczna generacja równoległego sprzętu
- Język: dwa podejścia
  - nowy język, dedykowany do potrzeb ⇒ adopcja przez użytkowników
  - adaptacja (do nowego kontekstu) języka już istniejącego



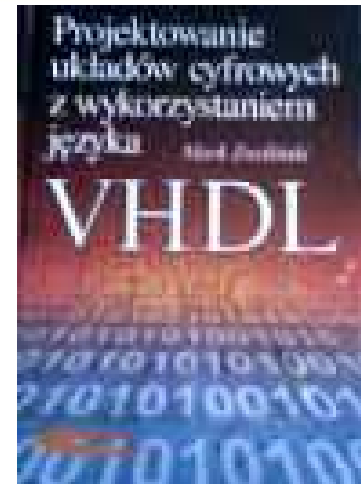
## VHDL – standardy

- IEEE Std 1076/INT-1991, IEEE Standards Interpretations: IEEE Std 1076-1987 IEEE Standard VHDL Language Reference Manual
- IEEE Std 1076-1993, IEEE Standard VHDL Language Reference Manual
- IEEE Std 1076a-2000, Amendment to IEEE Std 1076-1993
- IEEE Std 1029.1-1998, IEEE Standard for VHDL Waveform and Vector Exchange (WAVES) to Support Design and Test Verification
- IEEE Std 1076.1-1999, IEEE Standard VHDL Analog and Mixed-Signal Extensions (VHDL-AMS)
- IEEE Std 1076.2-1996, IEEE Standard VHDL Mathematical Packages
- IEEE Std 1076.3-1997, IEEE Standard VHDL Synthesis Packages
- IEEE Std 1076.4-1995, IEEE Standard for VITAL Application-Specific Integrated Circuit (ASIC) Modeling Specification
- Approved draft of IEEE Std 1076.6-1999, IEEE Standard for VHDL Register Transfer Level Synthesis
- IEEE Std 1149.1-1990, IEEE Standard Test Access Port and Boundary-Scan Architecture
- IEEE Std 1149.1b-1994, Supplement to IEEE Std 1149.1-1990, IEEE Standard Test Access Port and Boundary-Scan Architecture
- IEEE Std 1164-1993, IEEE Standard Multivalued Logic System for VHDL Model Interoperability (Std\_logic\_1164)

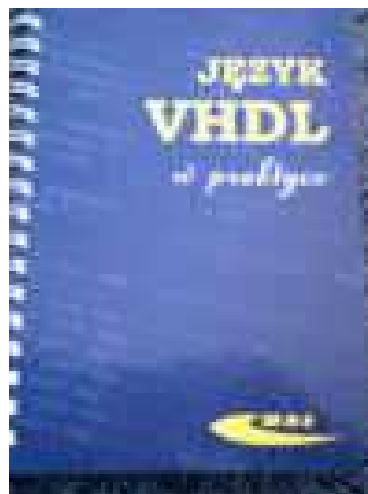
- „A Guide to VHDL”, S. Mazor, P. Langstraat*
- „VHDL Analysis and Modelling of Digital Systems”, Z. Navabi*
- „VHDL Hardware Description and Design”,  
R. Lipsett, C. Schaefer, C. Ussery*
- „The VHDL Cookbook”, P. J. Ashenden*
- „VHDL programming: with advanced topics”, L. Baker*
- „VHDL starter's guide”, S. Yalamanchili*
- „VHDL for designers”, S. Sjöholm, L. Lindh*
- „VHDL made easy!”, D. Pellerin, D. Taylor*
- „VHDL answers to frequently asked questions”, B. Cohen*
- „VHDL and AHDL digital systems implementation”, F. A. Scarpino*
- „Active-VHDL Series BOOK#2 – EVITA Interactive Tutorial”,  
J. Mirkowski, M. Kapustka, Z. Skowroński, A. Biniszkiewicz*
- „VHDL: a logic synthesis approach”, D. Naylor, S. Jones*



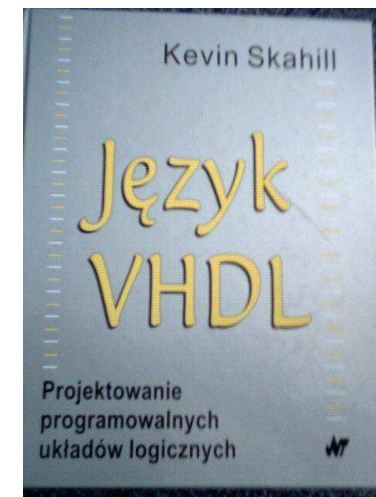
Włodzimierz  
Wrona



Mark  
Zwoliński



Józef  
Kalisz



Kevin  
Skahill



## VHDL – zasoby w Internecie

- Grupa dyskusyjna: **comp.lang.vhdl**
- EDA Industry Working Groups homepage: <http://www.eda.org/>
- Accellera: <http://www.accellera.org/>
- Design Automation Cafe: <http://www.dacafe.com/>
- Doulos High Level Design Web site: <http://www.doulos.com/>
- VHDL-online, University of Erlangen-Nürnberg: <http://www.vhdl-online.de/>
- The Hamburg VHDL Archive: <http://tams-www.informatik.uni-hamburg.de/research/vlsi/vhdl/>



## VHDL – tutorialie

- An Introductory VHDL Tutorial, Green Mountain Computing Systems: <http://www.gmvhdl.com/VHDL.html>
- Doulos High Level Design Web site; A Hardware Engineers Guide to VHDL: <http://www.doulos.com/hegv/index.htm>
- VHDL Synthesis Tutorial from APS: <http://www.associatedpro.com>
- Interactive VHDL Tutorial from Aldec, Inc.: <http://www.aldec.com/products/tutorials/>
- VHDL Verification Course by Stefan Doll: <http://www.stefanVHDL.com/>
- MicroLab VLSI Design courses: [http://www.microlab.ch/images/files/Bachelor/Courses/digital\\_3/VHDL.pdf](http://www.microlab.ch/images/files/Bachelor/Courses/digital_3/VHDL.pdf)





## VHDL – *free IP cores*

- OpenIP home page: <http://www.opencores.org/>
- The Free-IP Project Home Page: <http://www.free-ip.com/>
- The Hamburg VHDL archive:  
<http://tams-www.informatik.uni-hamburg.de/vhdl/index.php?content=06-models>
- RASSP www site: <http://www.eda.org/rassp/>
- Micron Technology, Inc. (memories): <http://www.micron.com/>
- VHDL Library of Arithmetic Units developed by R. Zmmermann:  
[http://www.iis.ee.ethz.ch/~zimmi/arith\\_lib.html](http://www.iis.ee.ethz.ch/~zimmi/arith_lib.html)

## VHDL – firmy (produkty)

- Aldec (*Active-HDL, Riviera*)
- Altium (*Altium Designer*)
- Cadence Design Systems
- Mentor Graphics (Model Technology: *ModelSim*)
- Synopsys (Synplicity: *SynplifyPro*)
- Xilinx (*XST*)
  
- Green Mountain Computing Systems (*Direct VHDL*)





- **Produkty**
  - **Software (ActiveCAD, ActiveHDL)**
  - **Hardware-based Simulation Accelerators**
  - **IP Cores**
- **Donacja dla laboratorium:**
  - **Komputery – 11×PC**
  - **Oprogramowanie (ActiveHDL)**
    - **20 sites license (Professional Edition)**
    - **Student Edition**
  - **Karty HES**
- **Oddział Kraków!**



## VHDL – Aldec Inc.

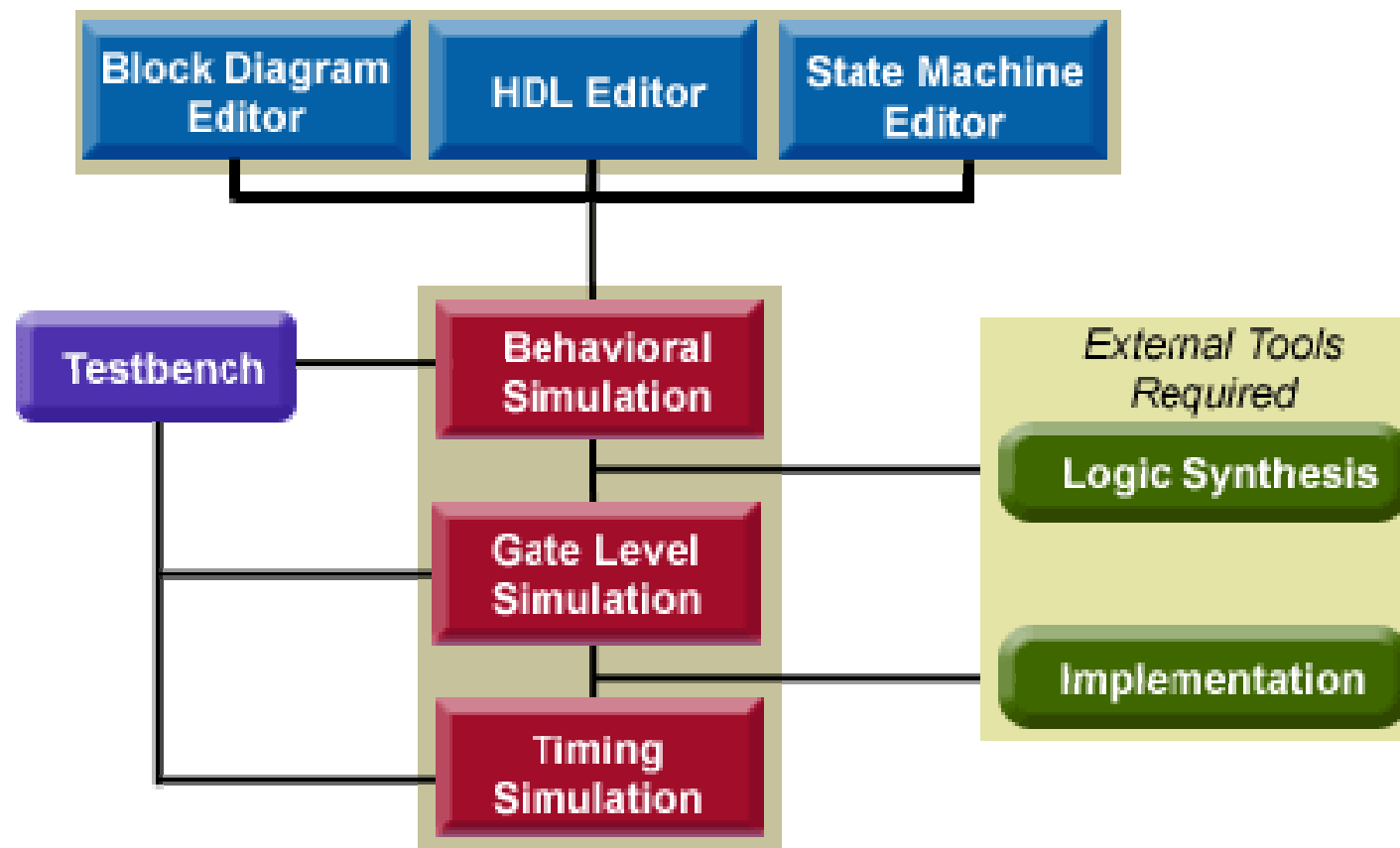


- **Praktyki studenckie**
- **Prace dyplomowe (*IP Cores*)**
  - **$\mu$ P/ $\mu$ C: PIC17C4x, PIC16C5x, 8051-SoC**
  - **periferia: 8251-SIO, 8255-PIO, 8257-DMA, 8259-INT**
  - **interfejsy: CAN, UART/IrDA, USB, I2S, PS/2, VGA, I2C, SD, ...**
  - **telekom: Reed-Solomon, Viterbi, Utopia, DTMF, MP3, ADPCM, LDPC, Kasumi, ...**
- **Praca zawodowa**

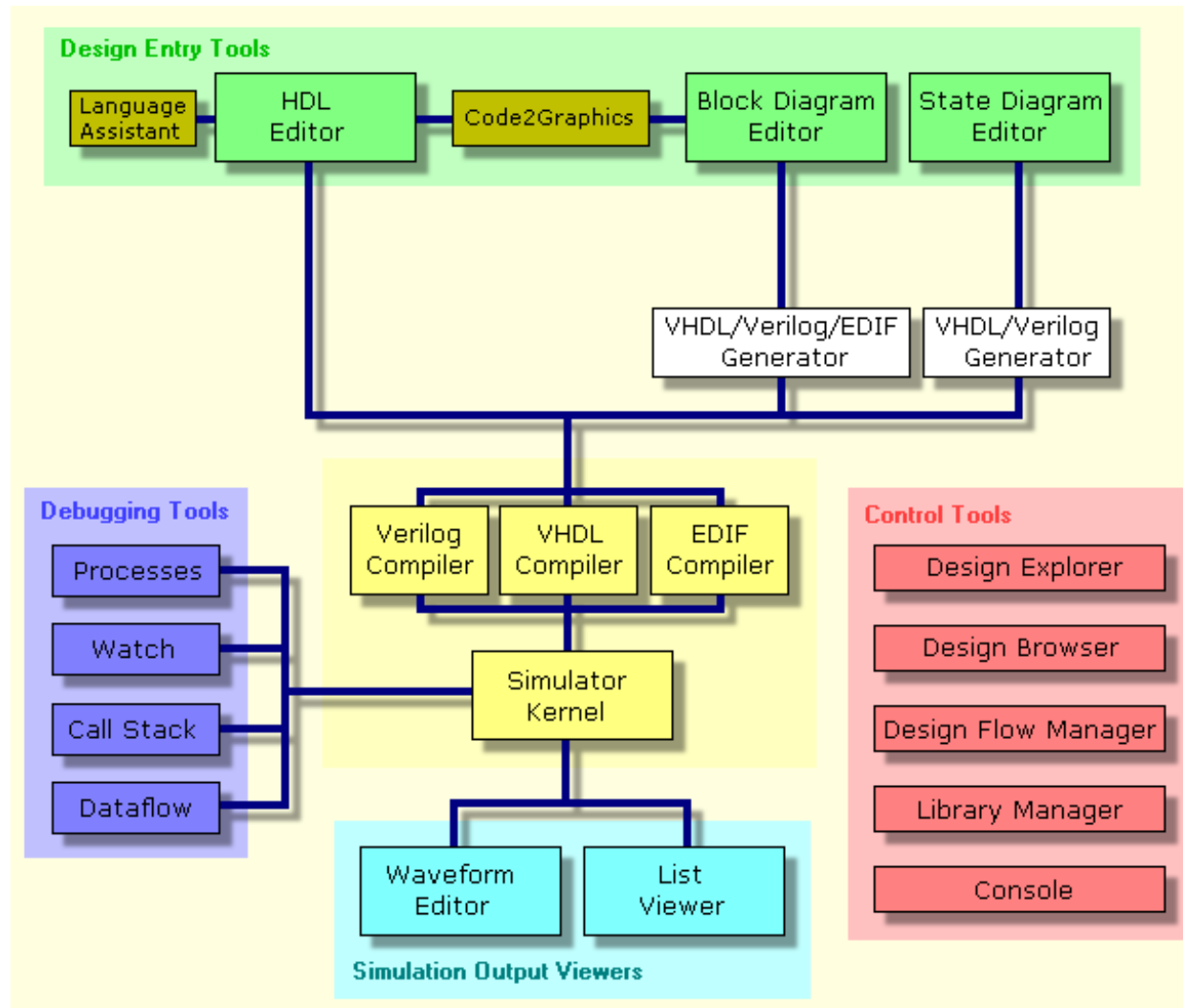
## ActiveHDL – moduły

# Active-HDL™

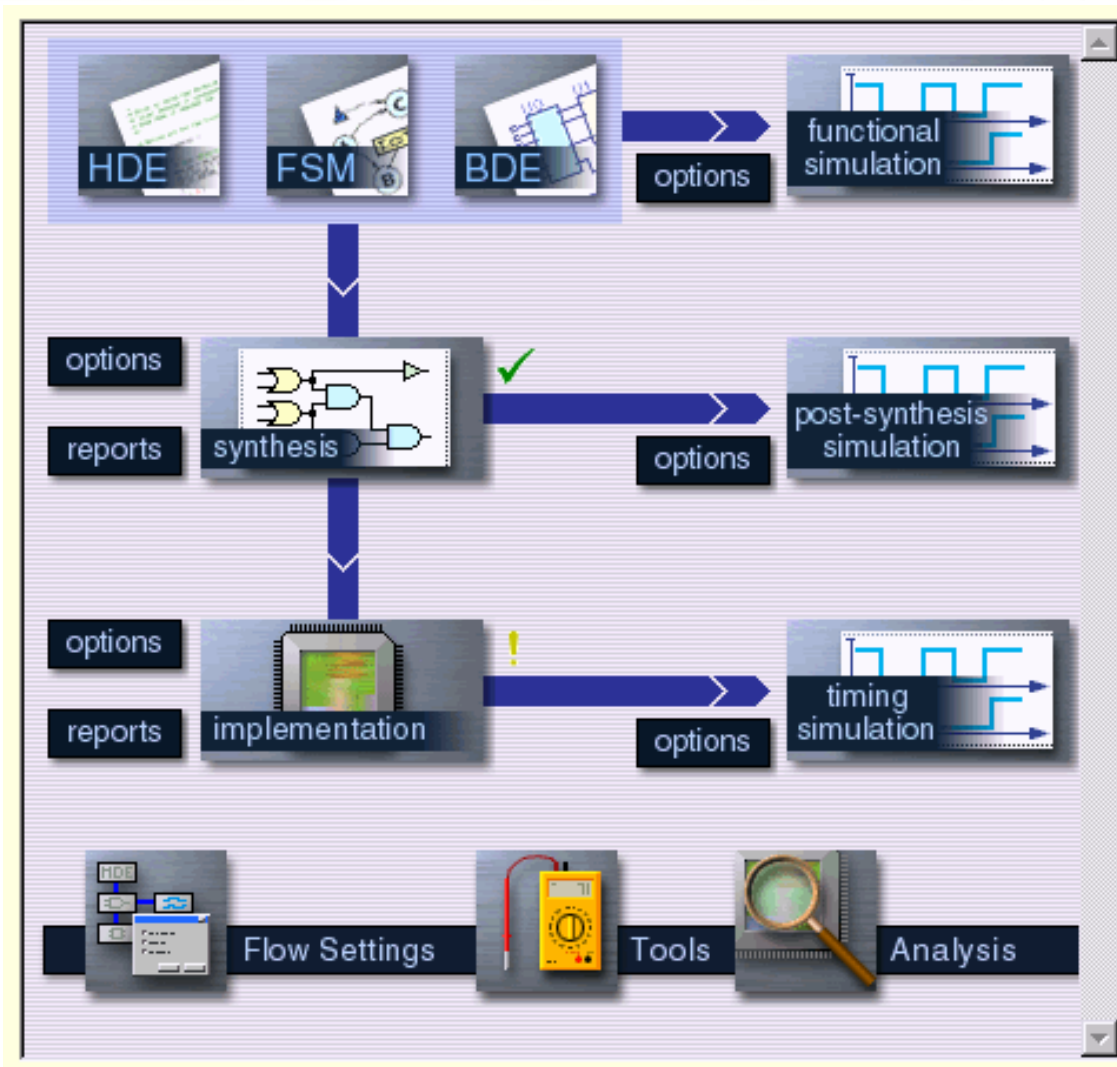
Complete FPGA Verification Environment



# ActiveHDL – moduły



# ActiveHDL – Design Flow

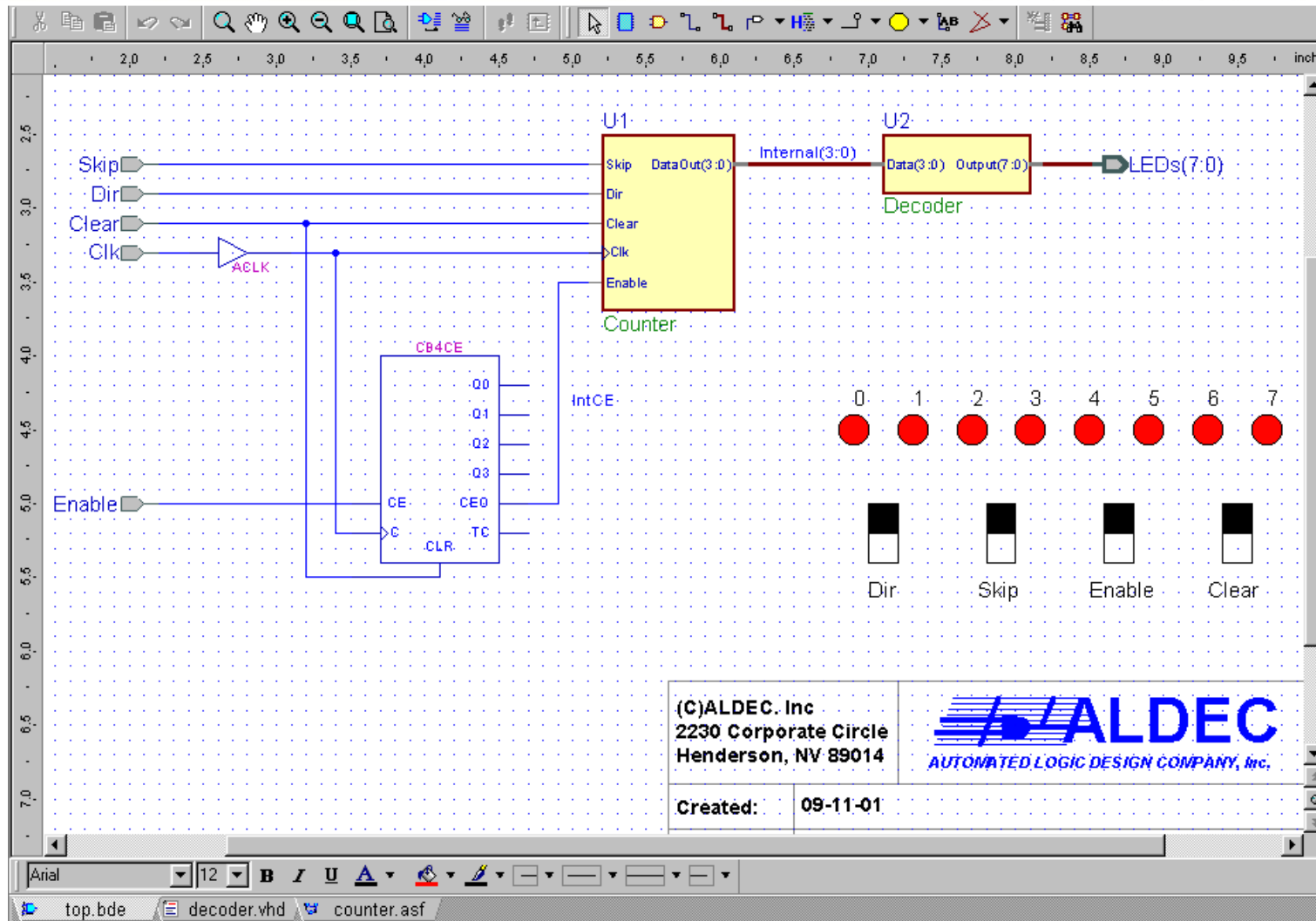


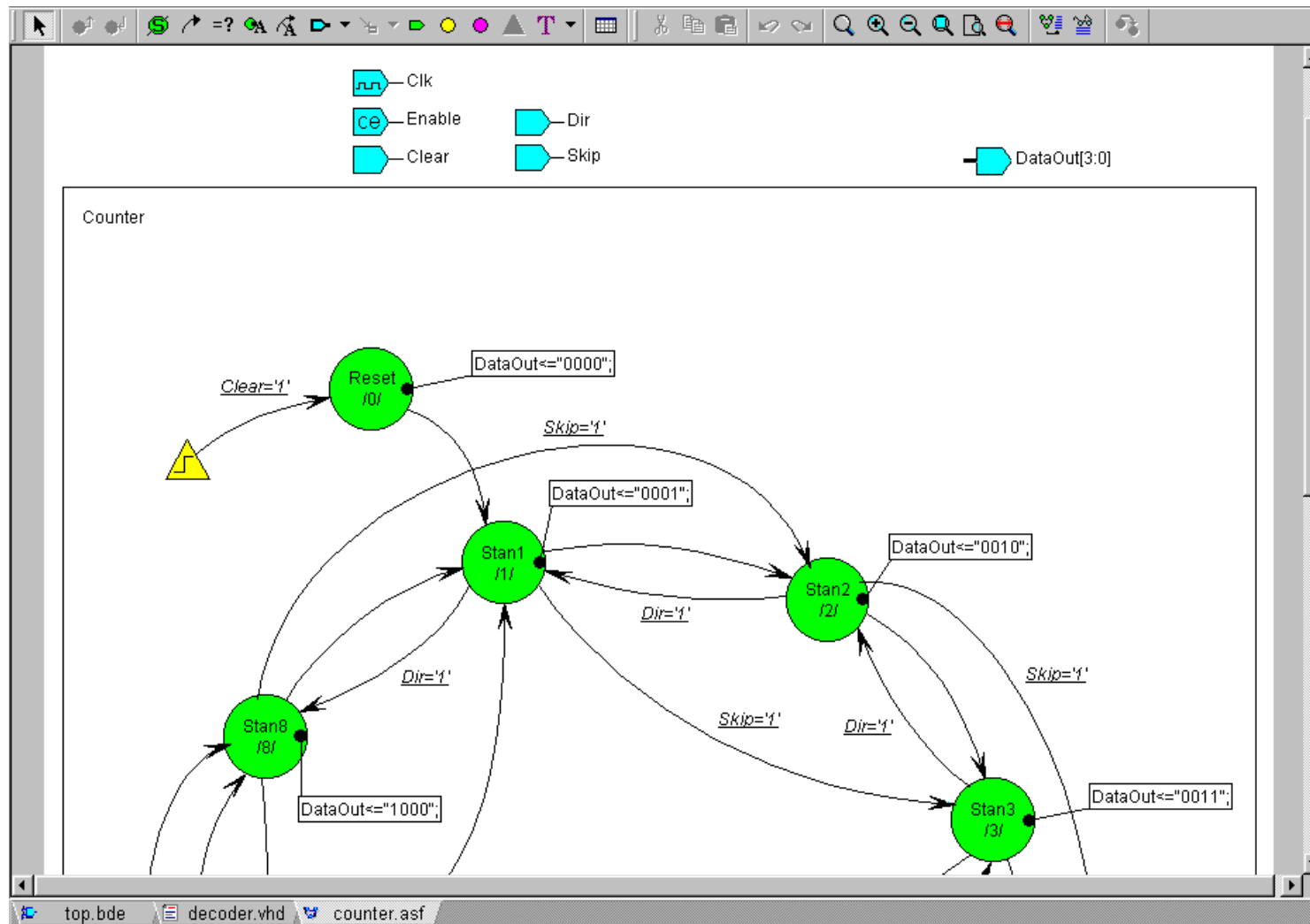


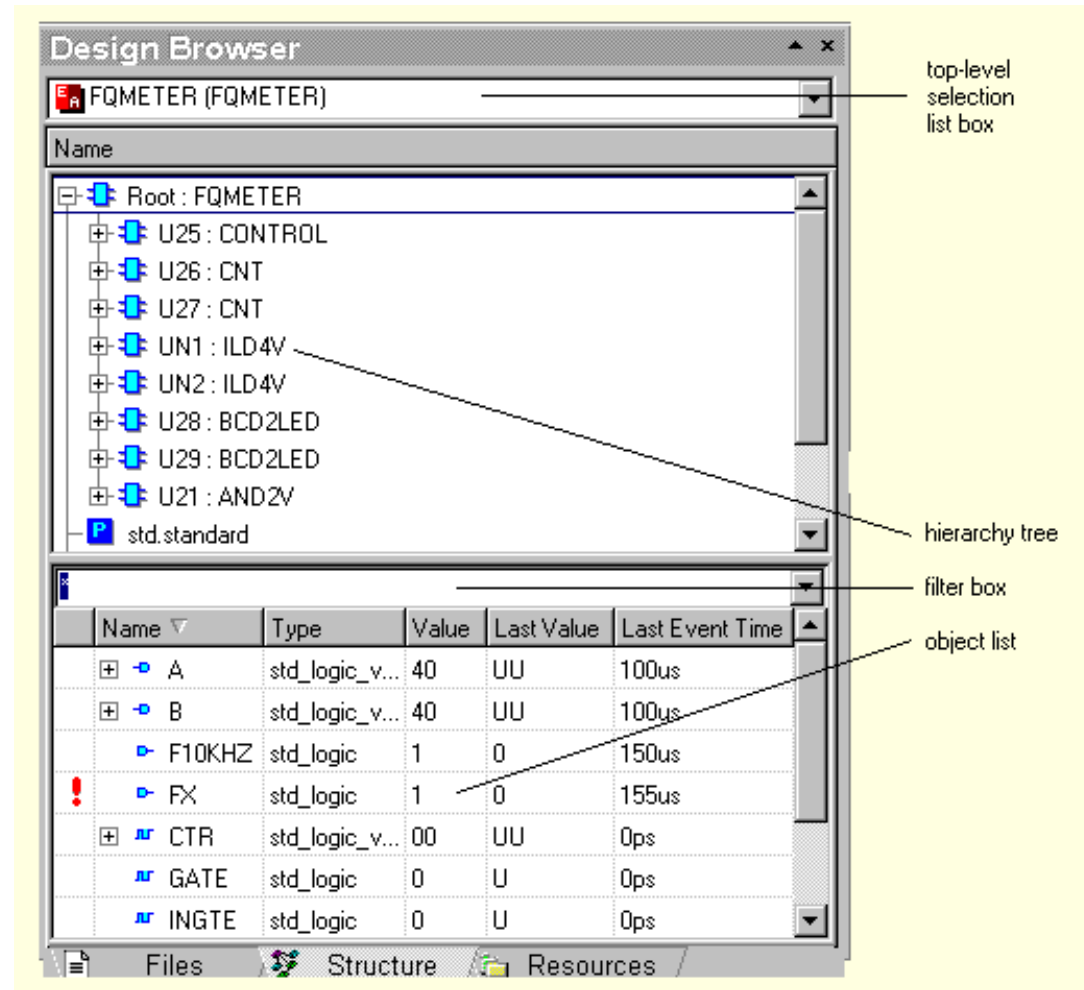
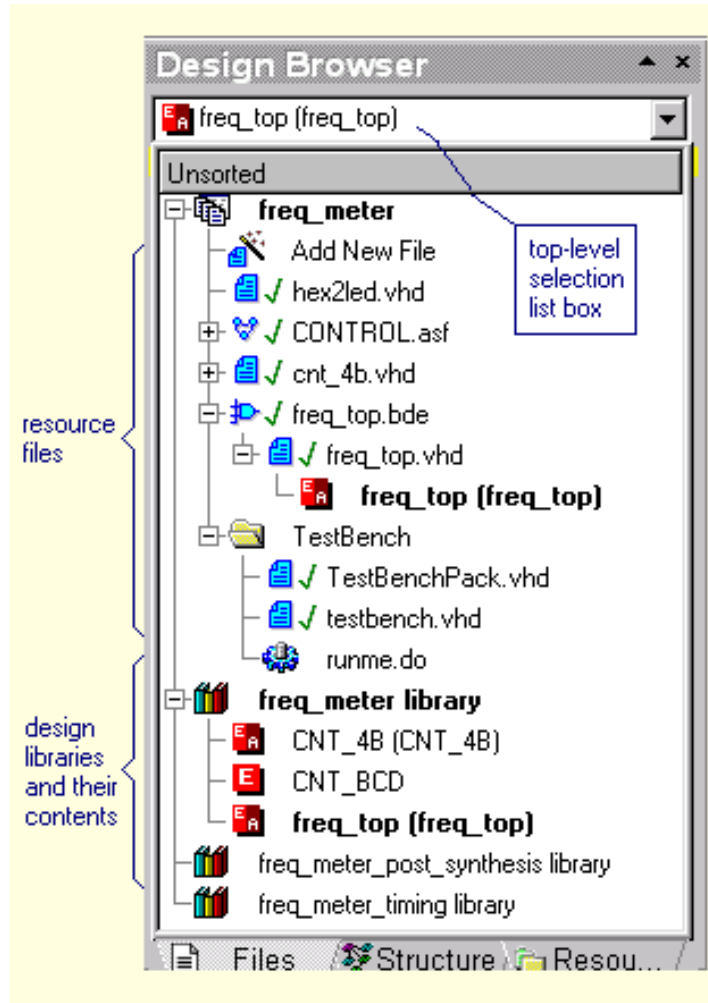
```
1
2  library IEEE;
3  use IEEE.std_logic_1164.all;
4
5  entity Decoder is
6      port (
7          Data: in STD_LOGIC_VECTOR (3 downto 0);
8          Output: out STD_LOGIC_VECTOR (7 downto 0)
9      );
10 end Decoder;
11
12 architecture Decoder of Decoder is
13 begin
14     with Data select
15     Output <= "00000000" when "0000",
16              "10000000" when "0001",
17              "01000000" when "0010",
18              "00100000" when "0011",
19              "00010000" when "0100",
20              "00001000" when "0101",
21              "00000100" when "0110",
22              "00000010" when "0111",
23              "00000001" when "1000",
24              "11111111" when others;
25 end Decoder;
26
27
28
29
30
31
32
33
34
35
```

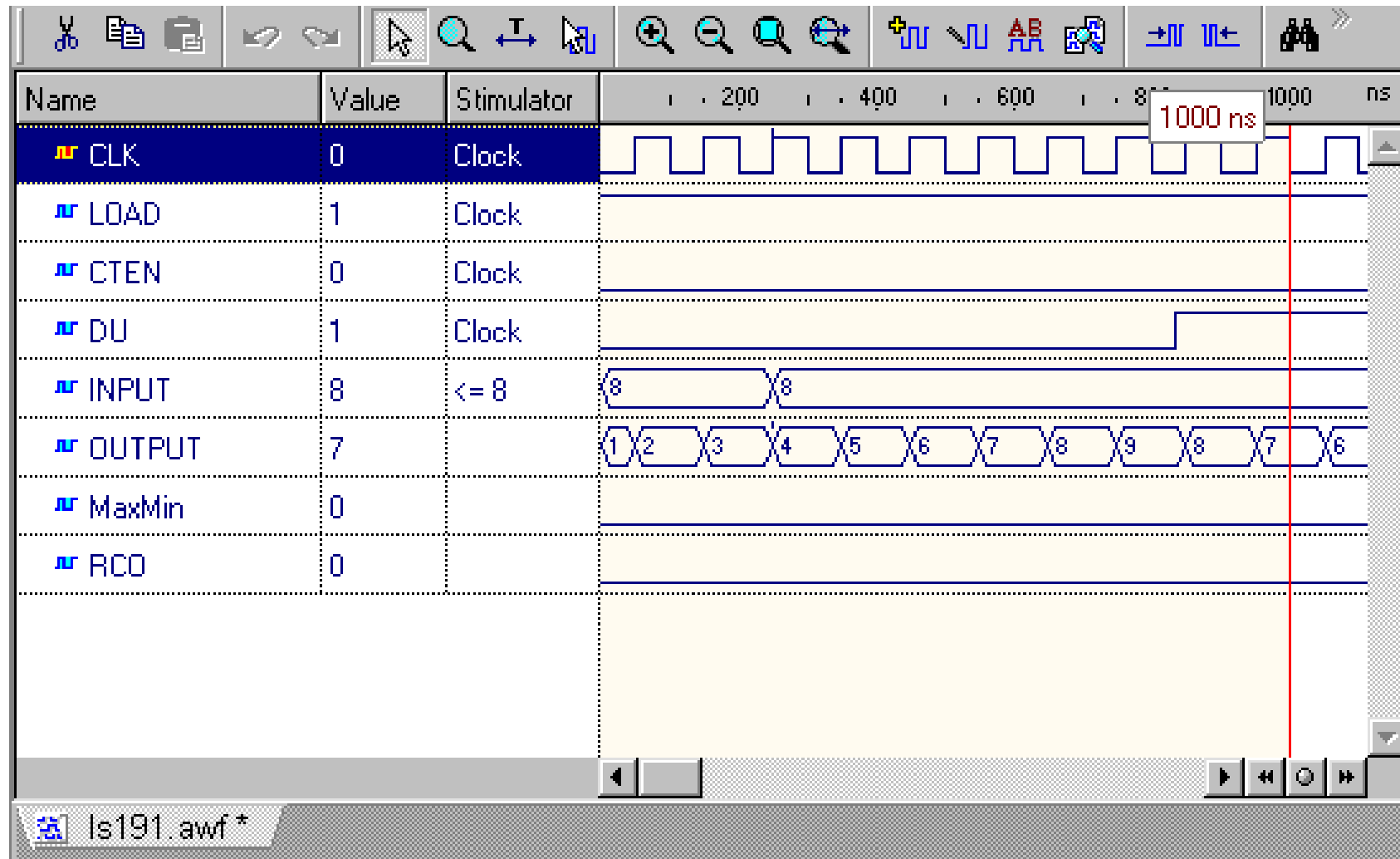
top.bde decoder.vhd counter.asf



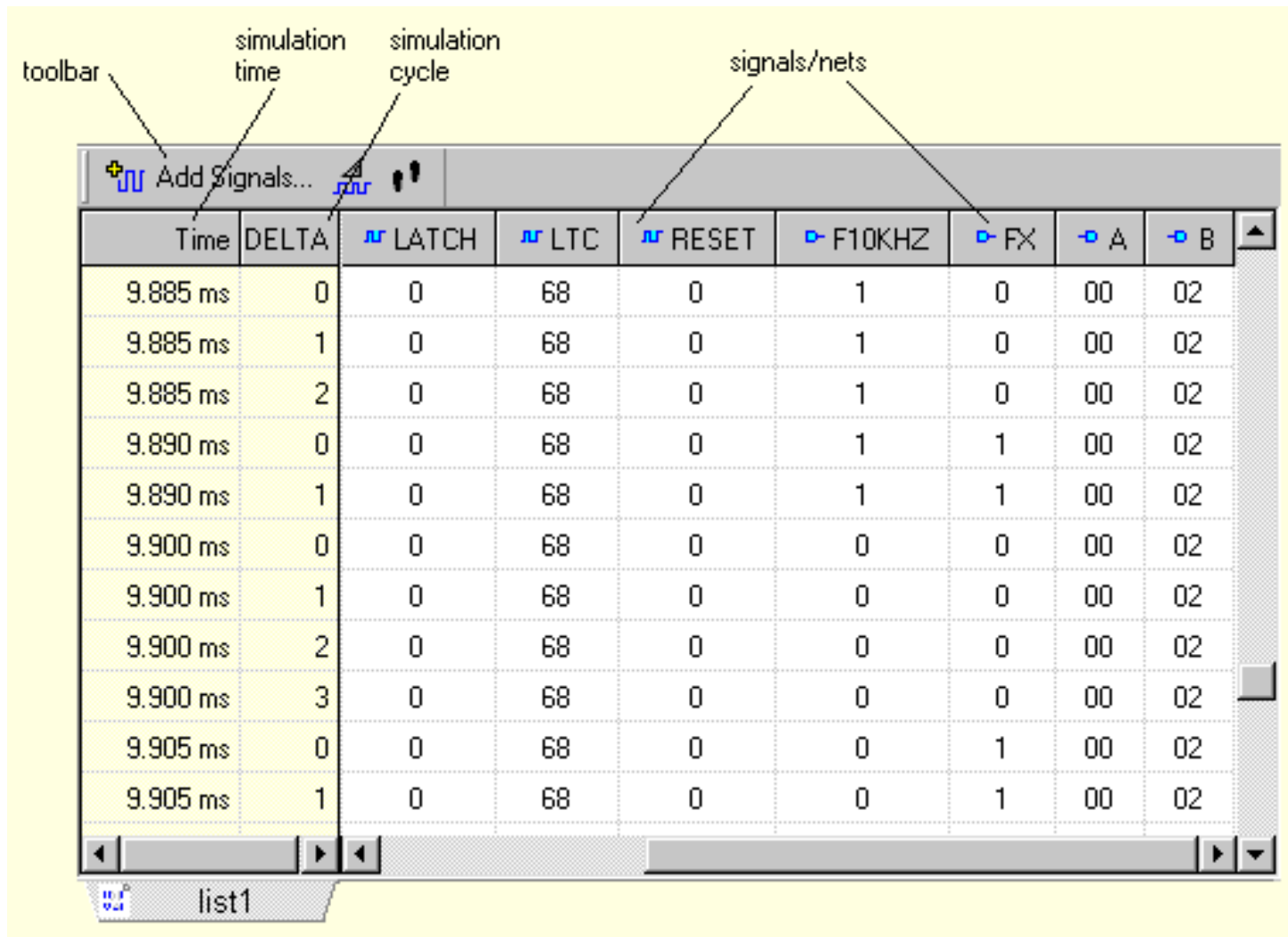








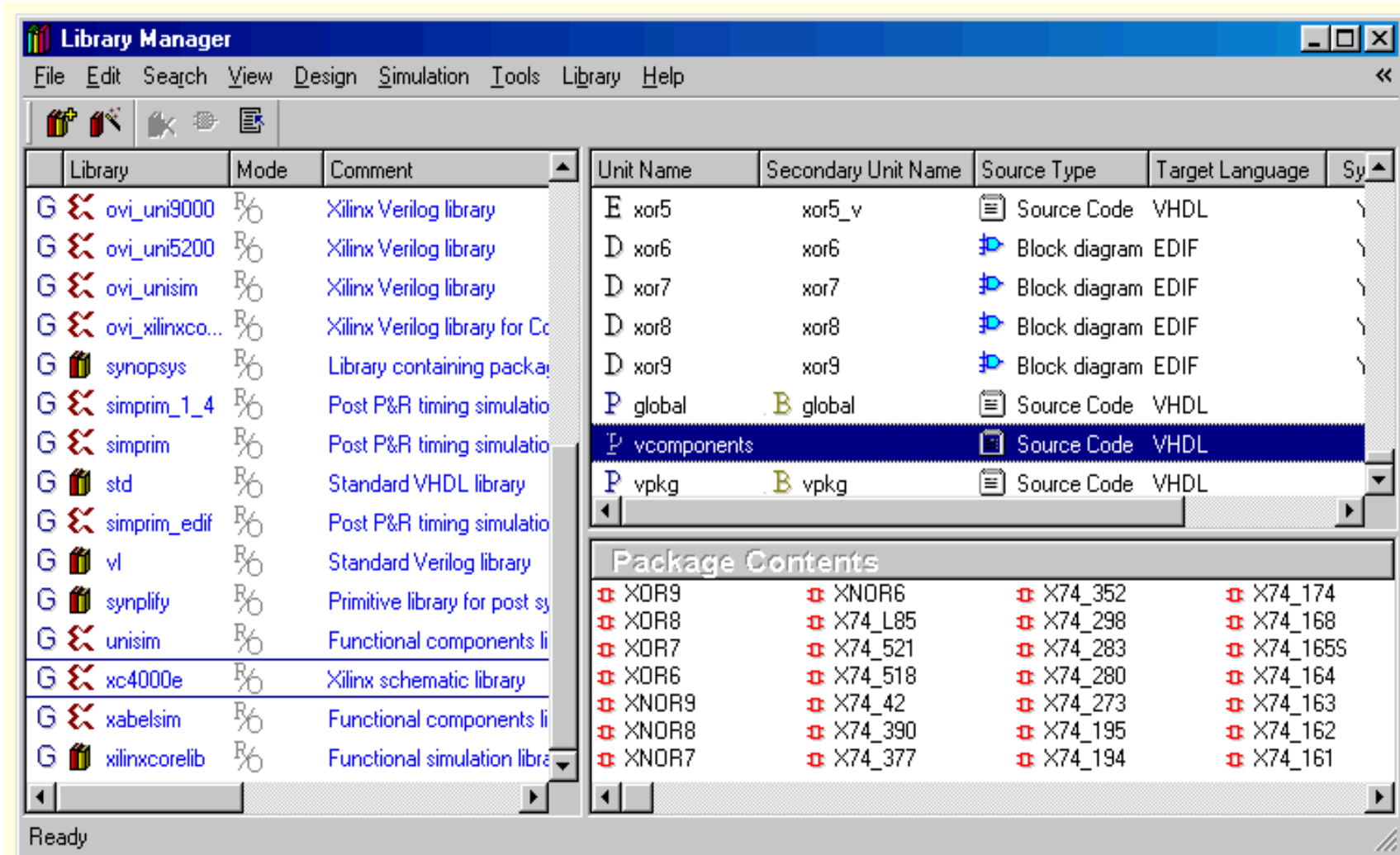
# ActiveHDL - List Viewer



The screenshot shows the ActiveHDL List Viewer window. The toolbar at the top includes an 'Add Signals...' button. The table displays simulation data with columns for Time, DELTA, and various signals/nets (LATCH, LTC, RESET, F10KHZ, FX, A, B). Annotations with arrows point to the toolbar, simulation time, simulation cycle, and signals/nets columns.

Time	DELTA	LATCH	LTC	RESET	F10KHZ	FX	A	B
9.885 ms	0	0	68	0	1	0	00	02
9.885 ms	1	0	68	0	1	0	00	02
9.885 ms	2	0	68	0	1	0	00	02
9.890 ms	0	0	68	0	1	1	00	02
9.890 ms	1	0	68	0	1	1	00	02
9.900 ms	0	0	68	0	0	0	00	02
9.900 ms	1	0	68	0	0	0	00	02
9.900 ms	2	0	68	0	0	0	00	02
9.900 ms	3	0	68	0	0	0	00	02
9.905 ms	0	0	68	0	0	1	00	02
9.905 ms	1	0	68	0	0	1	00	02

# ActiveHDL – Library Manager



The screenshot shows the 'Library Manager' window with a menu bar (File, Edit, Search, View, Design, Simulation, Tools, Library, Help) and a toolbar. The main area is divided into two panes. The left pane shows a list of libraries with columns for Library, Mode, and Comment. The right pane shows a detailed view of the selected 'vcomponents' package, including a table of Unit Name, Secondary Unit Name, Source Type, and Target Language. Below this is a 'Package Contents' section listing various components like XOR9, XNOR6, etc.

Library	Mode	Comment	Unit Name	Secondary Unit Name	Source Type	Target Language
ovi_uni9000	...	Xilinx Verilog library	xor5	xor5_v	Source Code	VHDL
ovi_uni5200	...	Xilinx Verilog library	xor6	xor6	Block diagram	EDIF
ovi_unisim	...	Xilinx Verilog library	xor7	xor7	Block diagram	EDIF
ovi_xilinxco...	...	Xilinx Verilog library for Co	xor8	xor8	Block diagram	EDIF
synopsys	...	Library containing packag	xor9	xor9	Block diagram	EDIF
simprim_1_4	...	Post P&R timing simulation	global	global	Source Code	VHDL
simprim	...	Post P&R timing simulation	vcomponents		Source Code	VHDL
std	...	Standard VHDL library	vpkg	vpkg	Source Code	VHDL
simprim_edif	...	Post P&R timing simulation				
vi	...	Standard Verilog library				
synplify	...	Primitive library for post sy				
unisim	...	Functional components li				
xc4000e	...	Xilinx schematic library				
xabelsim	...	Functional components li				
xilinxcorelib	...	Functional simulation libra				

Package Contents			
XOR9	XNOR6	X74_352	X74_174
XOR8	X74_L85	X74_298	X74_168
XOR7	X74_521	X74_283	X74_165S
XOR6	X74_518	X74_280	X74_164
XNOR9	X74_42	X74_273	X74_163
XNOR8	X74_390	X74_195	X74_162
XNOR7	X74_377	X74_194	X74_161

Ciąg dalszy  
nastąpi...

