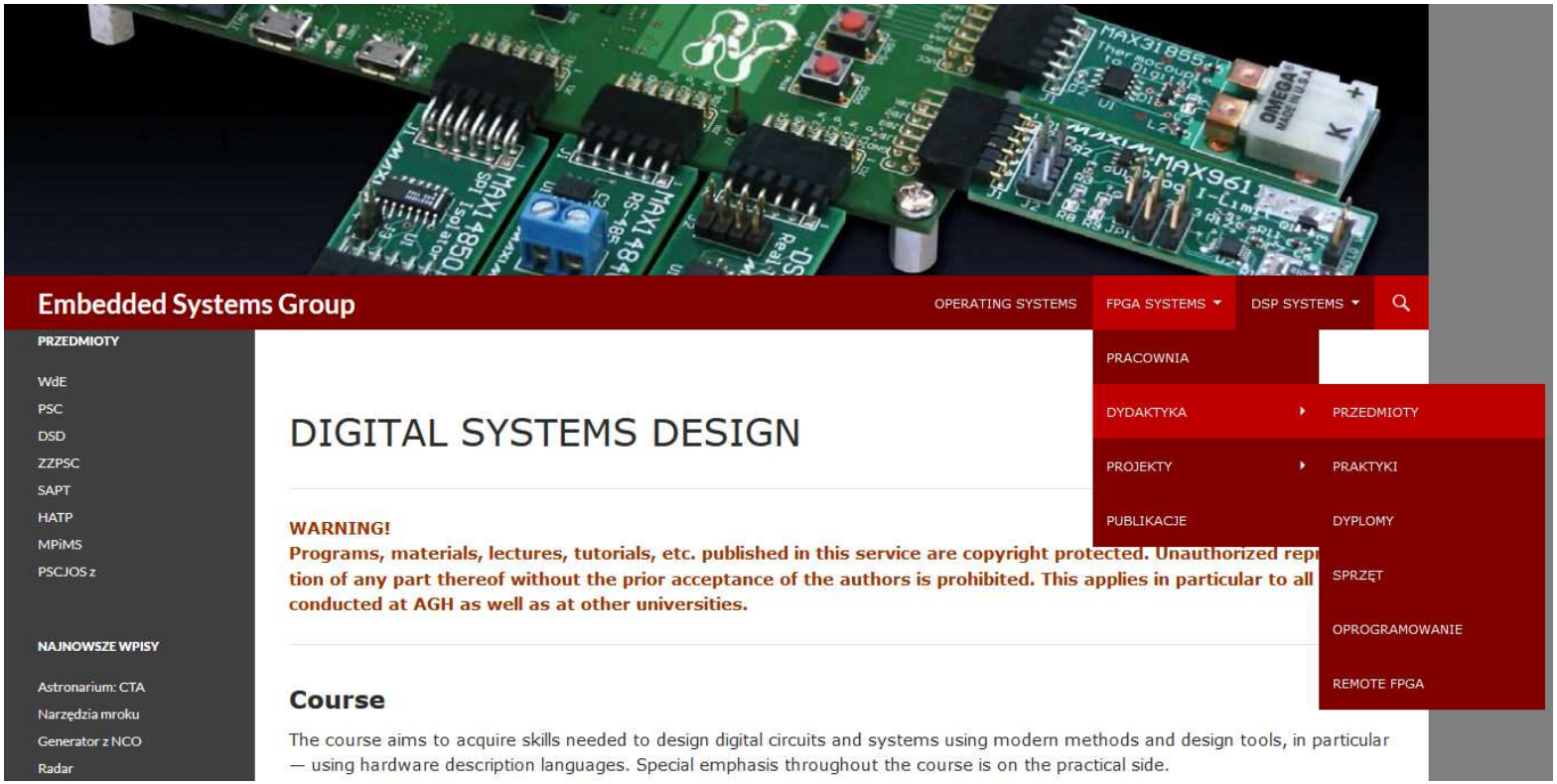
Introduction

Contact with lecturer

Paweł J. Rajda, PhD EE pjrajda@agh.edu.pl
C-3, room 502, phone (12) 617 3980
http://www.embedded.agh.edu.pl

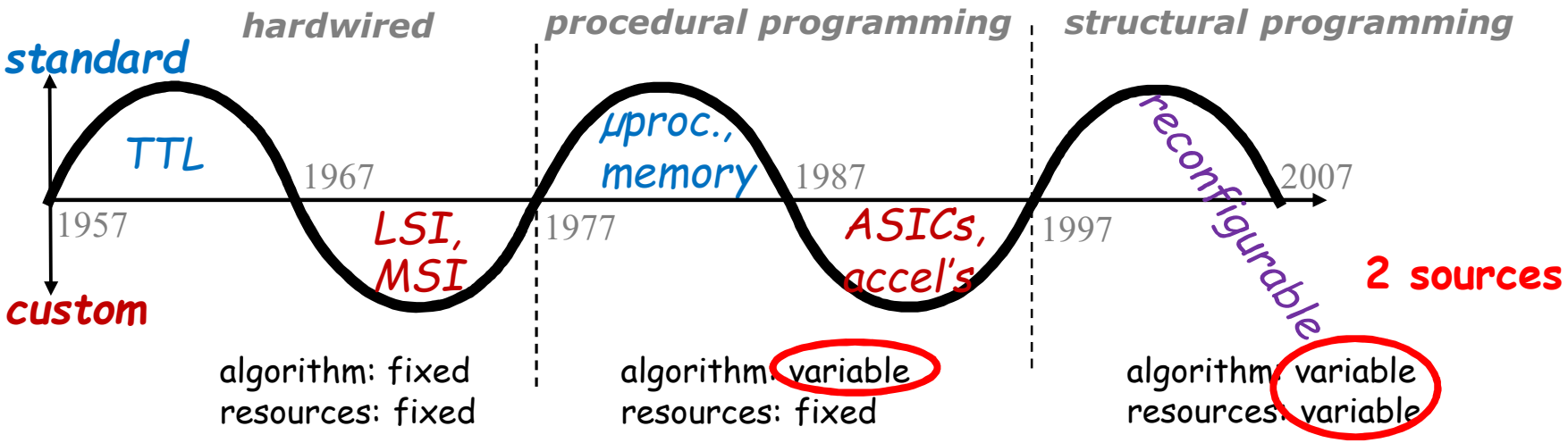


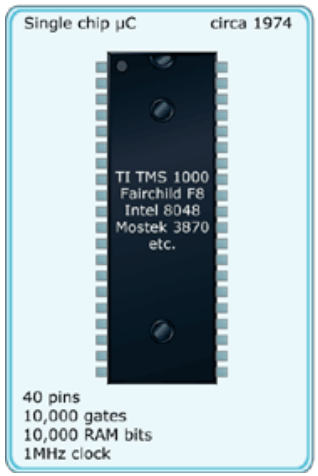
Agenda
A decalogue

1. VHDL – what for?
2. VHDL – what's this?
3. VHDL – how, where, when?
4. VHDL – so what we have?
5. VHDL – yesterday, today, tomorrow
6. VHDL – standards
7. VHDL – literature
8. VHDL – sources
9. VHDL – companies
10. VHDL – Aldec: Active HDL

Makimoto's Wave

Technological pendulum



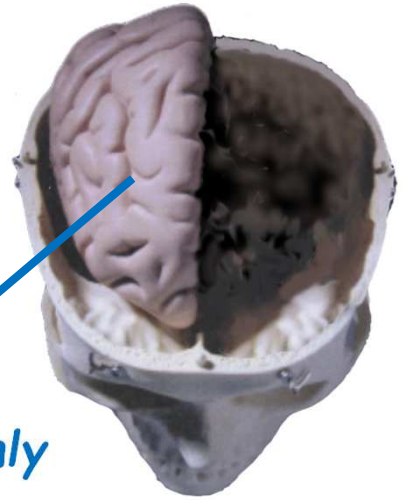


vN machine paradigm

new machine paradigm needed

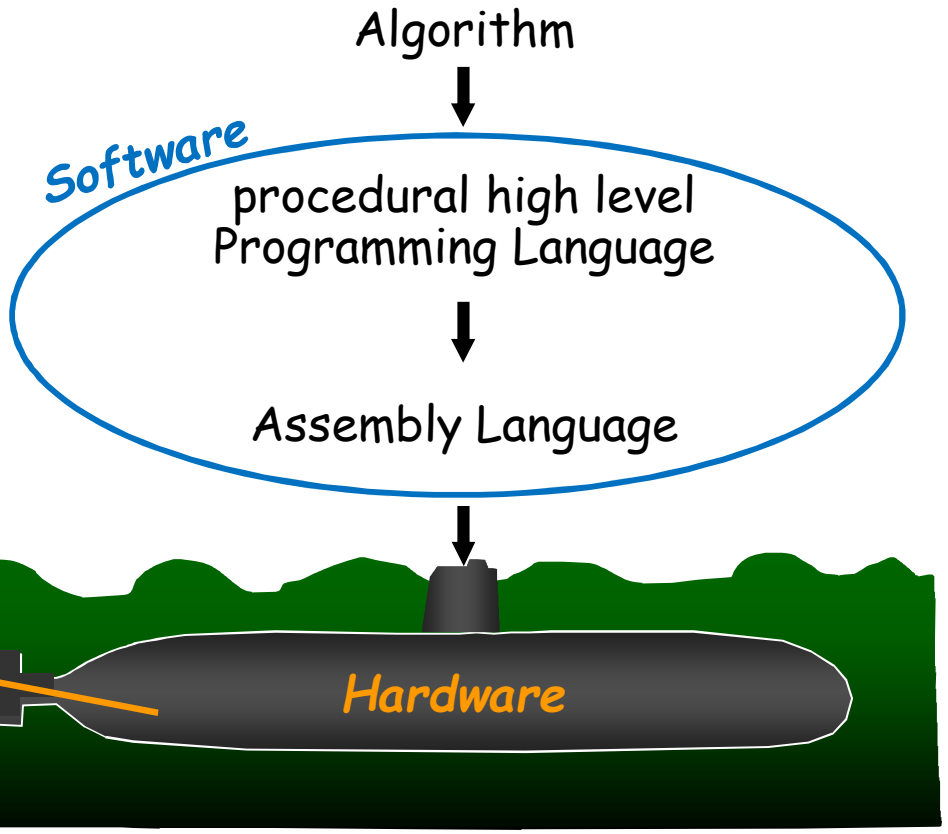
The Programmable System-on-a-Chip IS THE NEXT WAVE

Current model

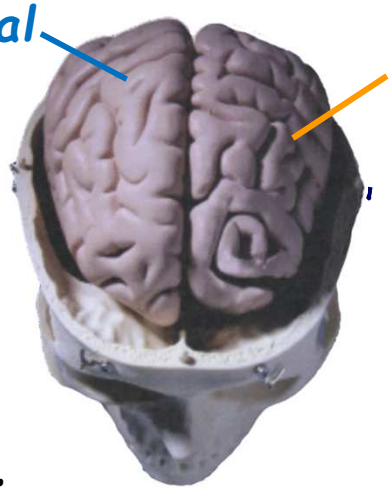


Brain usage:
procedural-only

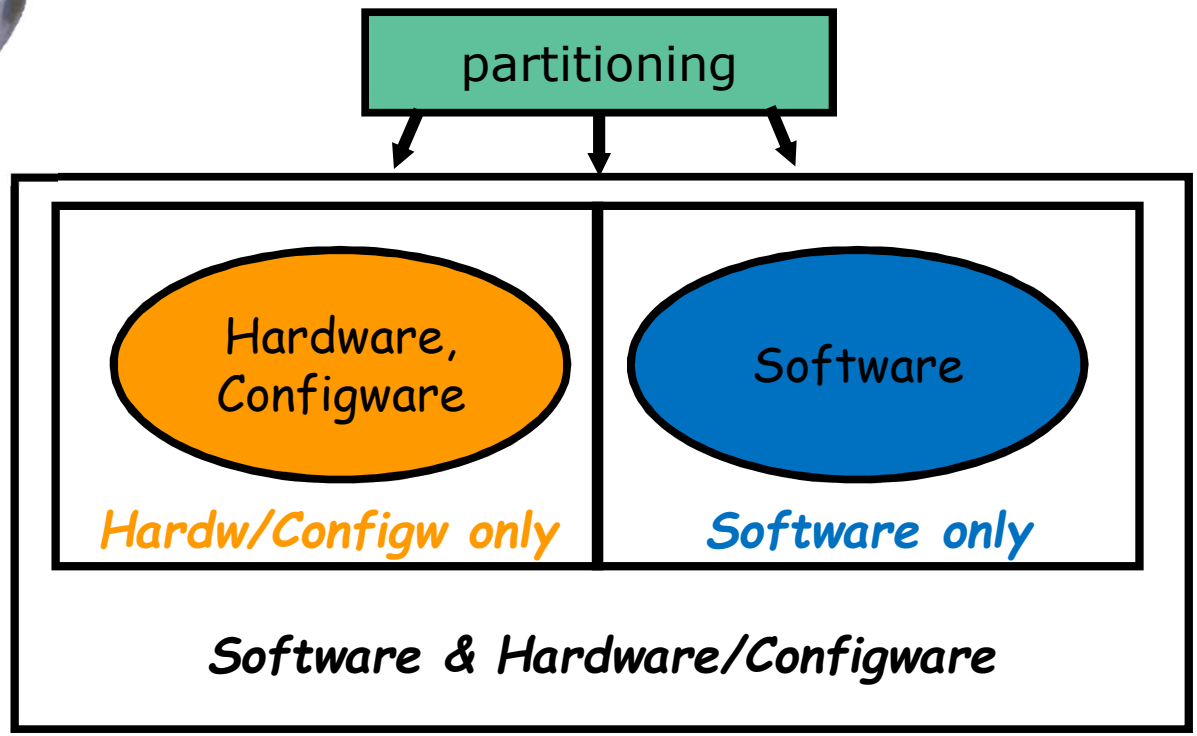
Hardware invisible:
under the surface



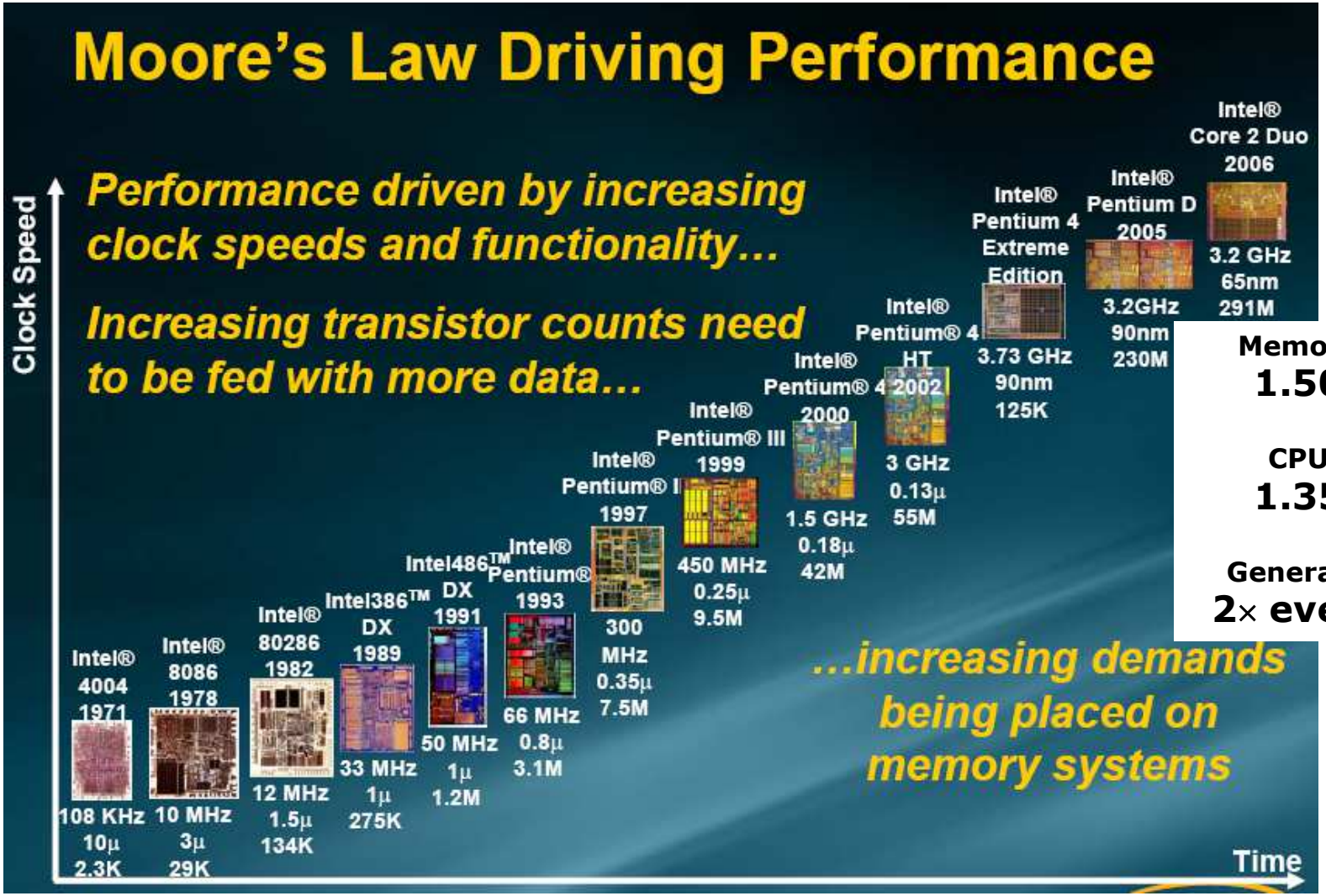
New model



Brain usage:
both hemispheres



Design complexity



Memory demands:
1.50× a year

CPU demands:
1.35× a year

General Moor's law:
2× every 2 years

Design complexity

Strong need for proper tool:

• 1970 - INTEL 4004	4 engineers	~1k transistors
• 1982 - INTEL 80286	20 engineers	~100k transistors
• 1992 - INTEL PENTIUM	100 engineers	~3M transistors
• 2003 - ???	1000 engineers ???	~150M transistors ???

Contemporary requirements:

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

Solution: HDL usage!
(VHDL, Verilog, ...)

Hardware Description Languages

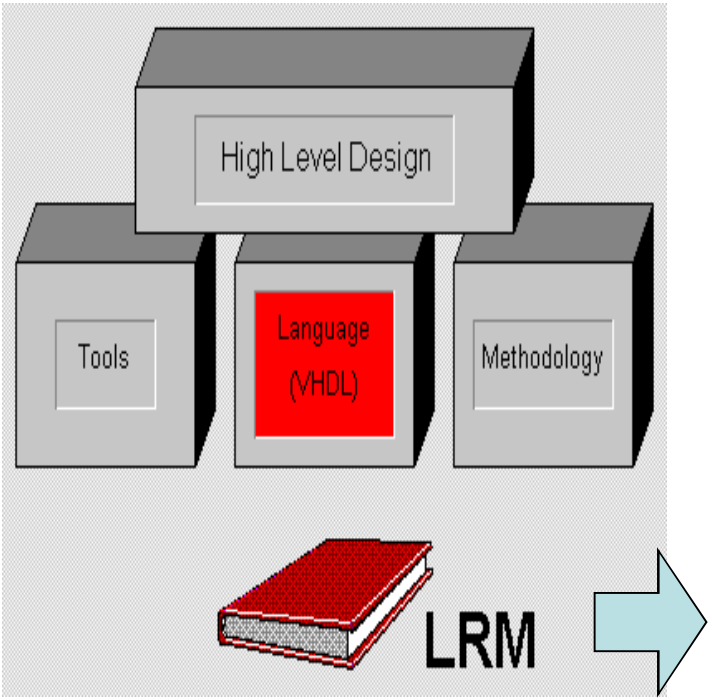
- **PALASM**
- **ABEL**
- **CUPL**

- **VHDL**
- **VERILOG, System VERILOG**
- **C, C++, Handel-C, System-C**
- **others…**

Definition

VHDL - VHSIC Hardware Description Language

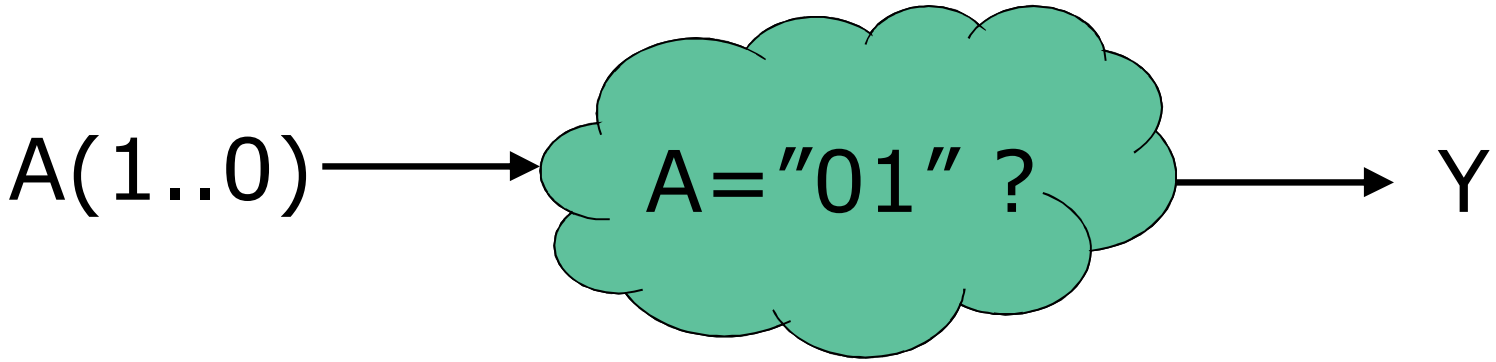
↳ Very High Speed Integrated Circuit



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 ..."

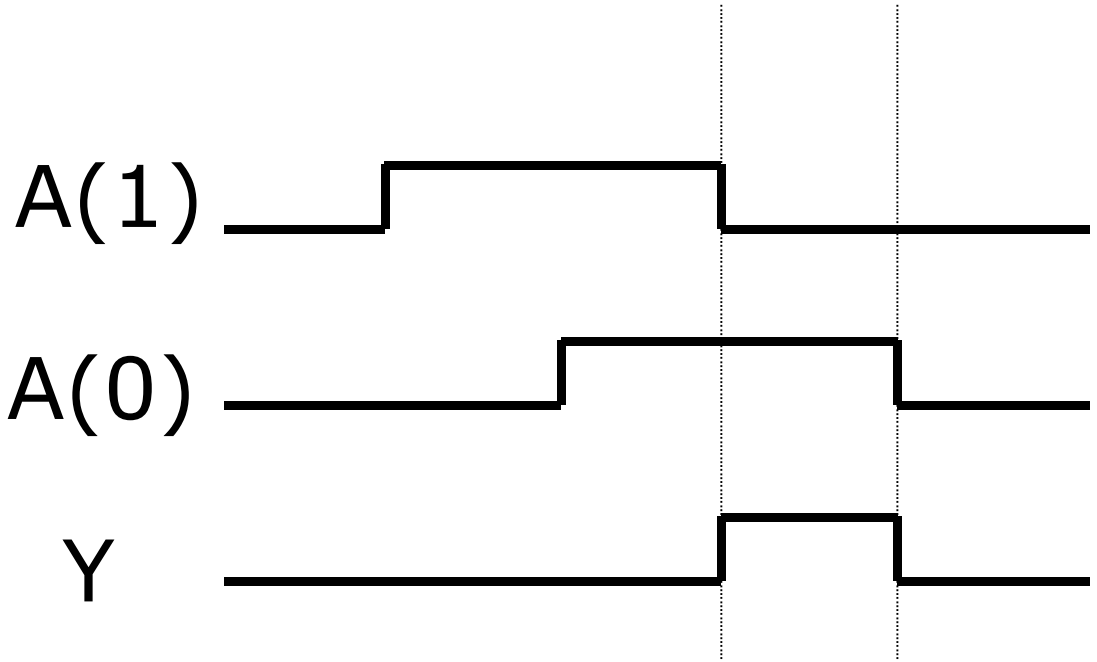
[*IEEE Standard VHDL Language Reference Manual*]

Modeling




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

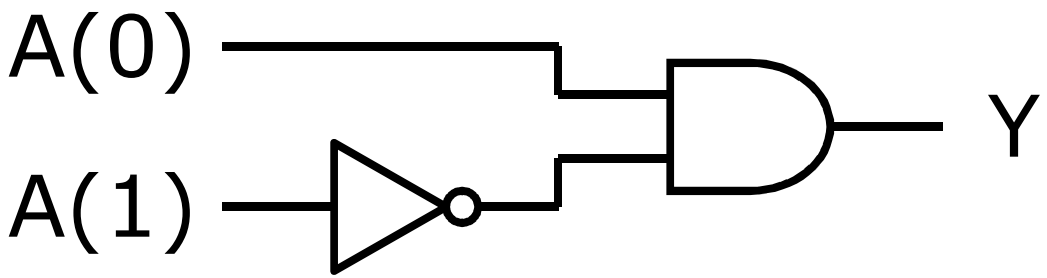
Simulation



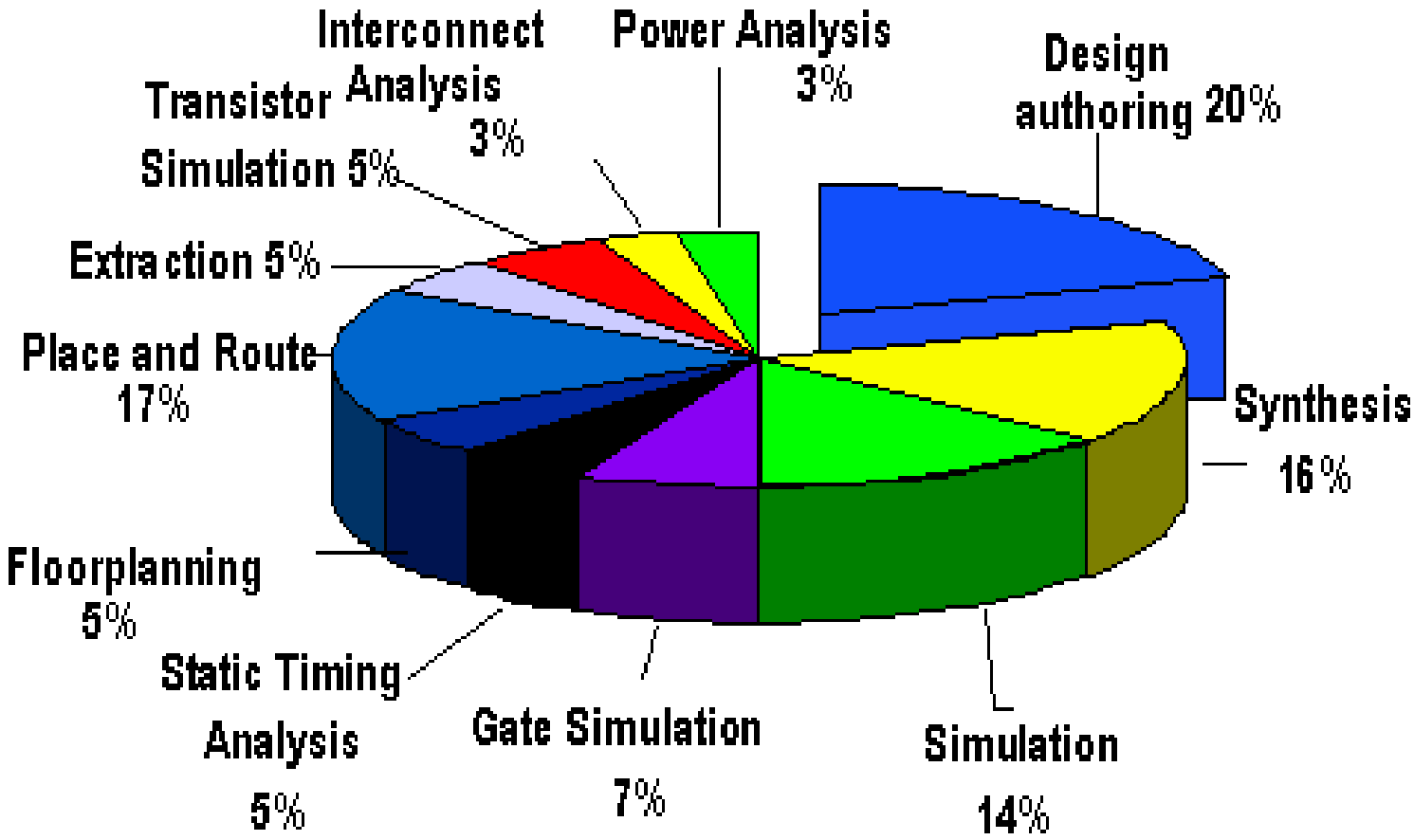

```
Stim <= "10", "11" after 4 ns ...
```

Logic synthesis

Logic synthesis – (automatic) translation of textual hardware description into a structure of connections (netlist) between elementary functional blocks of target hardware platform (gates, flip-flops, memories & others…).

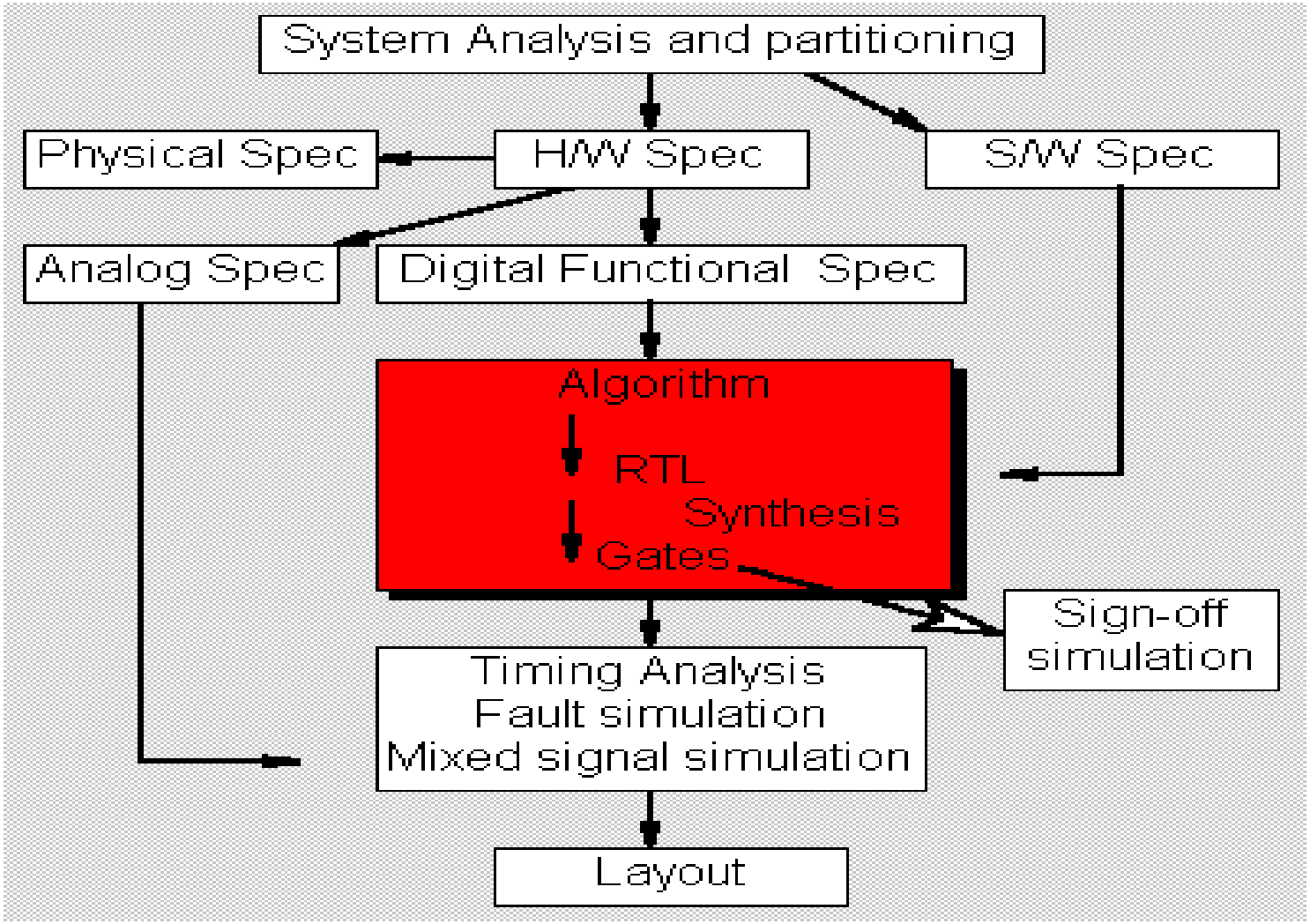


Design phases

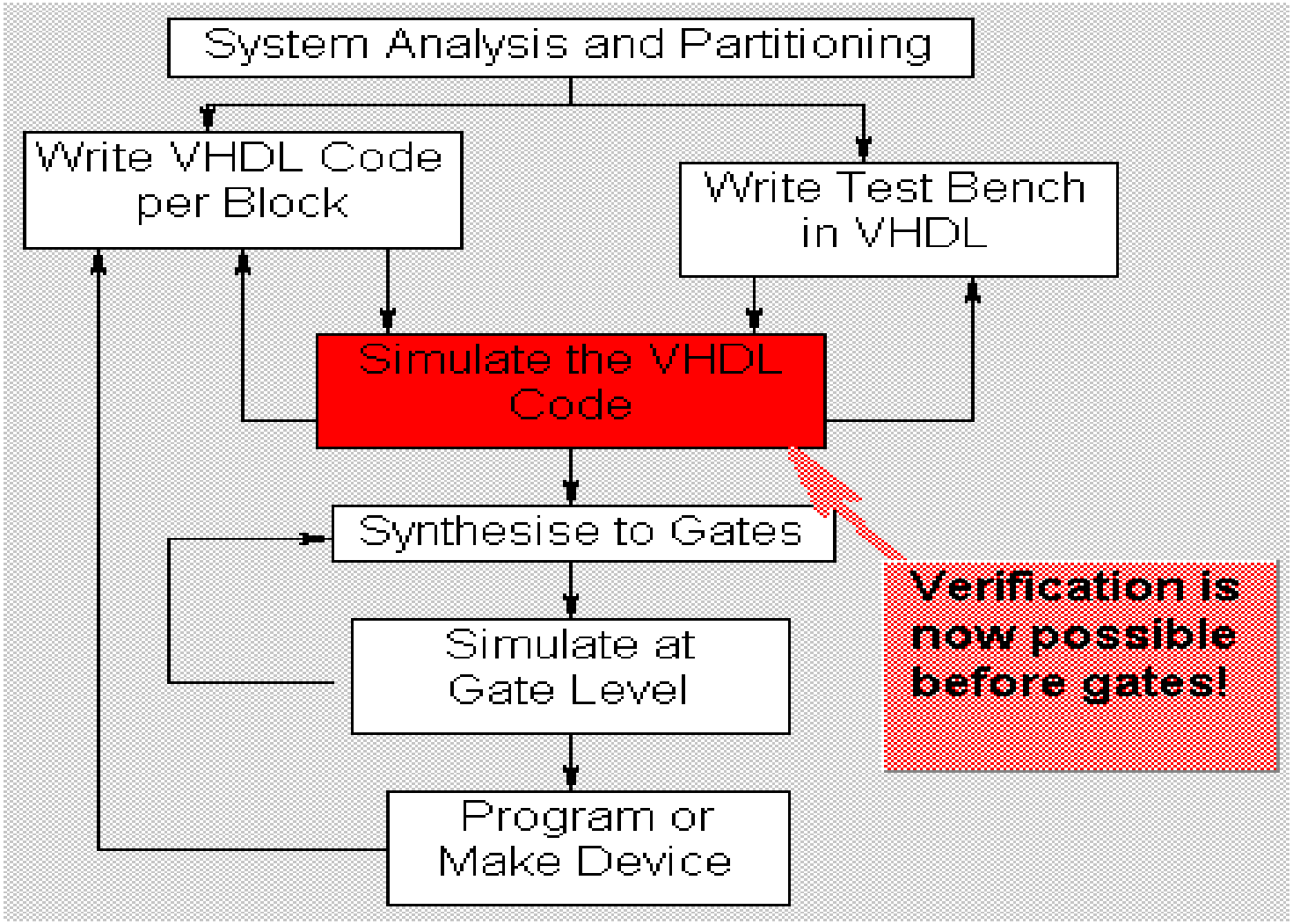


Average iterations between design and layout = 20

System design flow



Design process of digital hardware



Basic design units

entity *name*

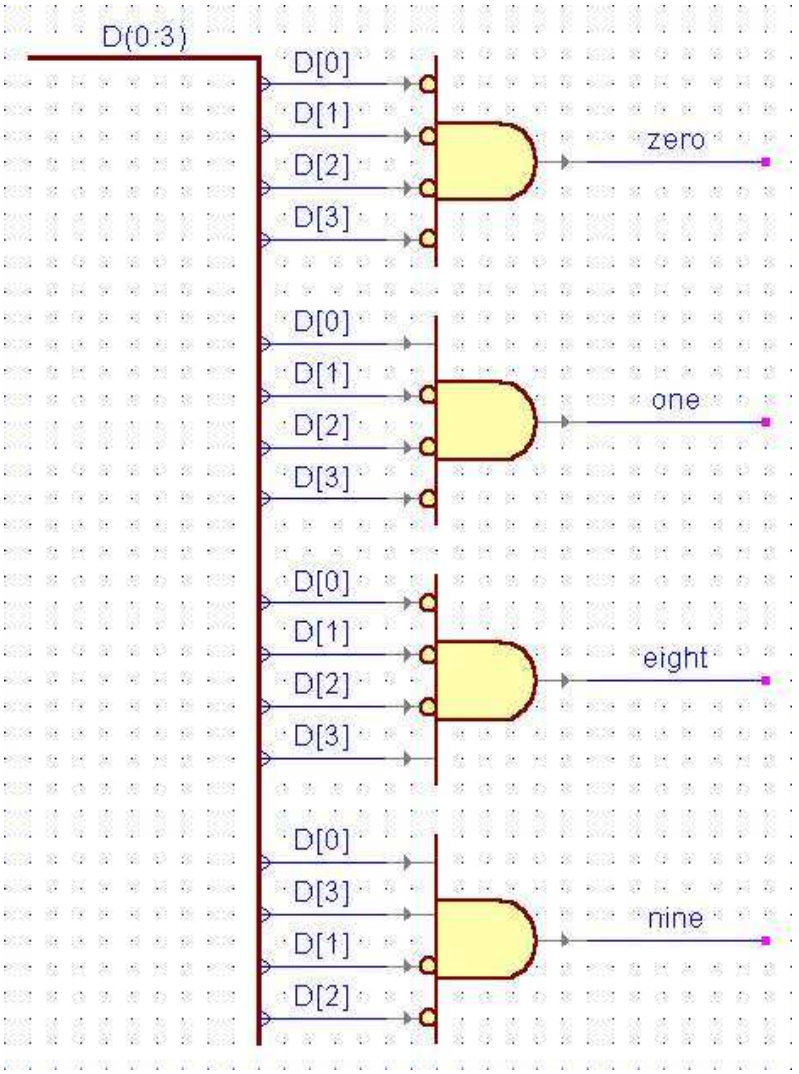
```
entity COMPARE is
  port (A,B: in bit;
        C: out bit);
end COMPARE;
```

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;
```

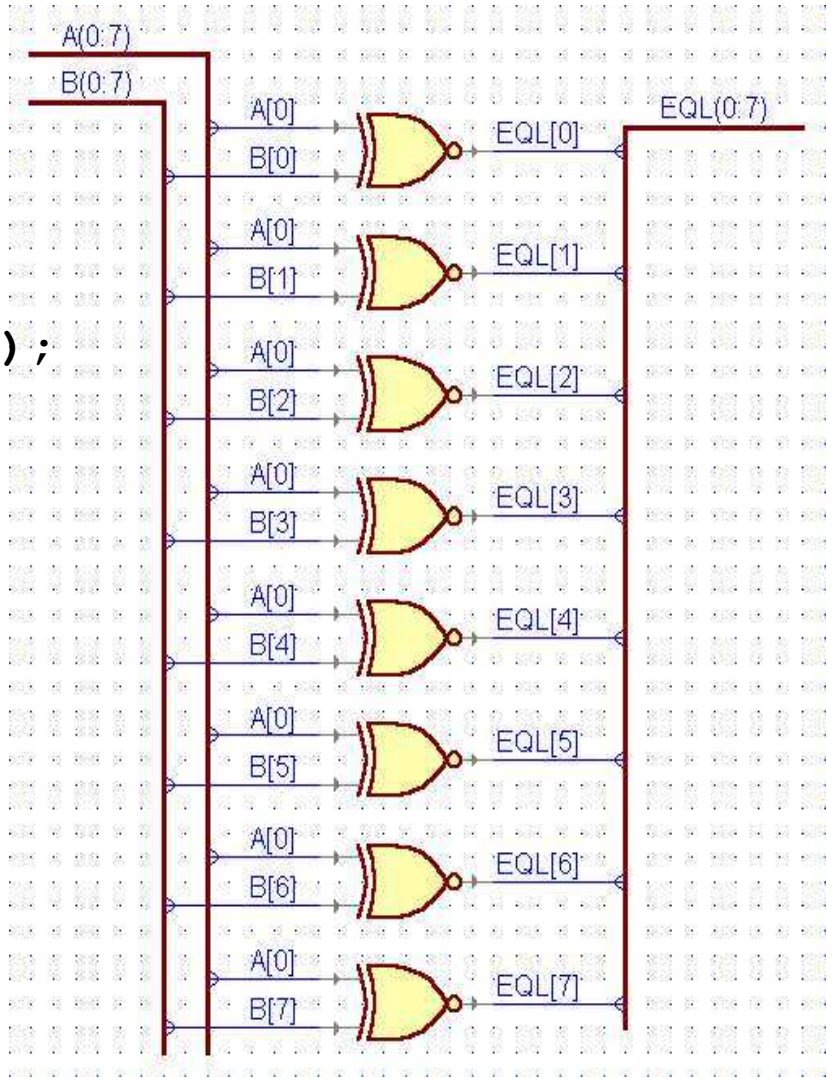
Decoder

```
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;
```



Comparator

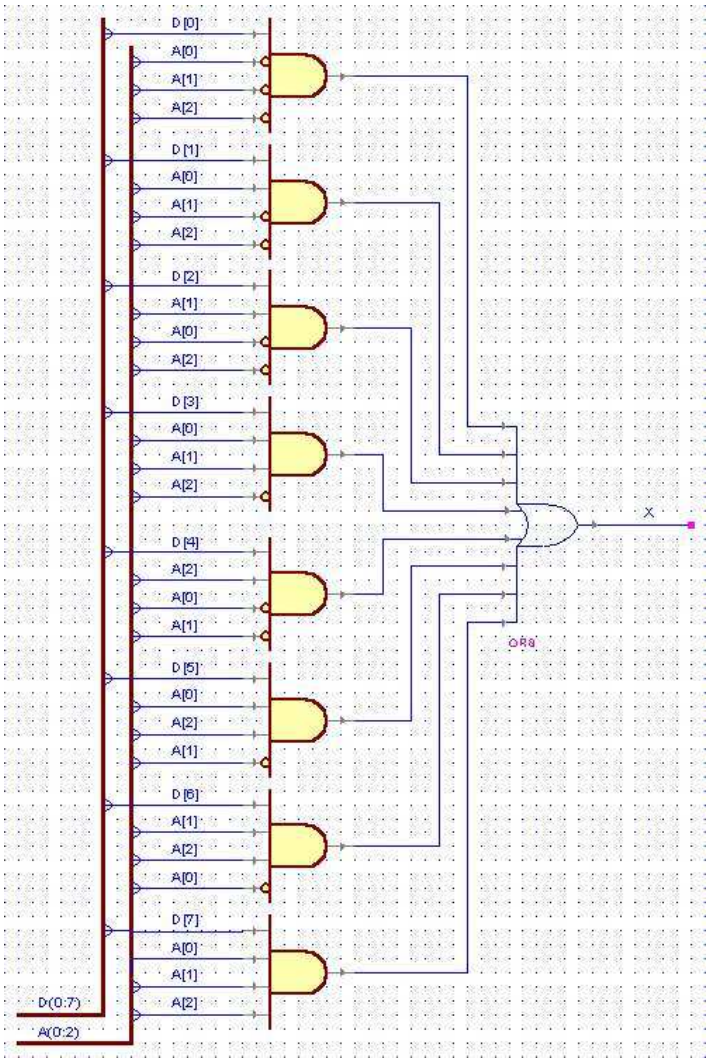
```
entity COMPARE is
  port(A,B: in bit_vector (0 to 7);
       EQL: out bit_vector (0 to 7));
end COMPARE;

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



Fault !

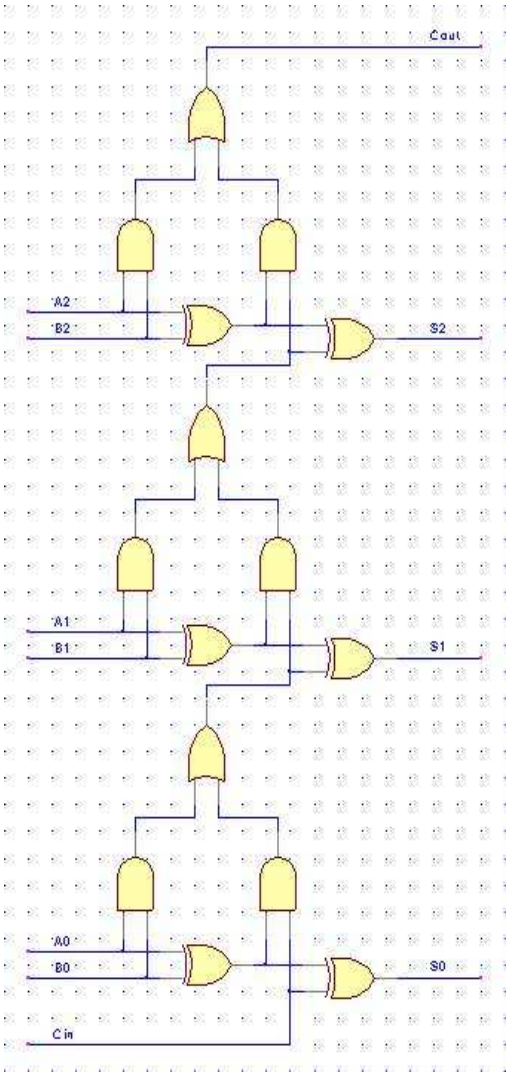
Multiplexer

```
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;
```



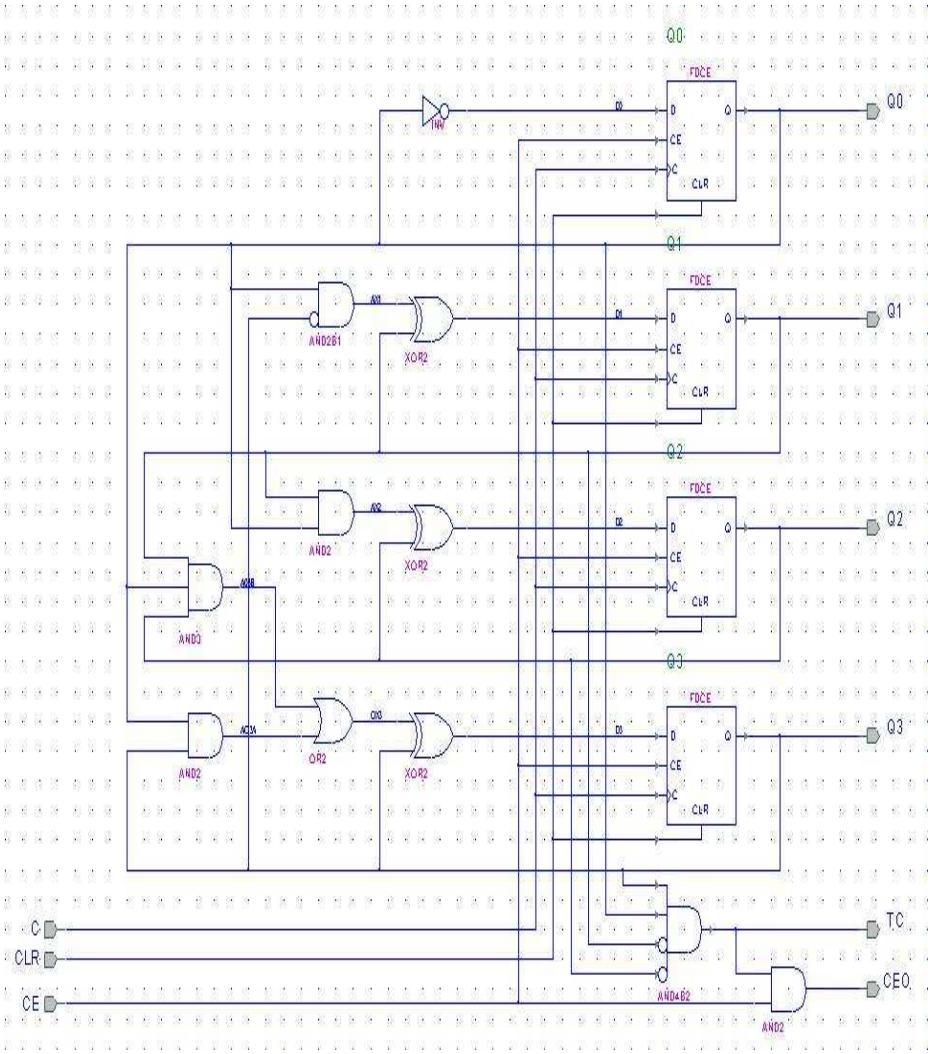
Adder

```
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;
```



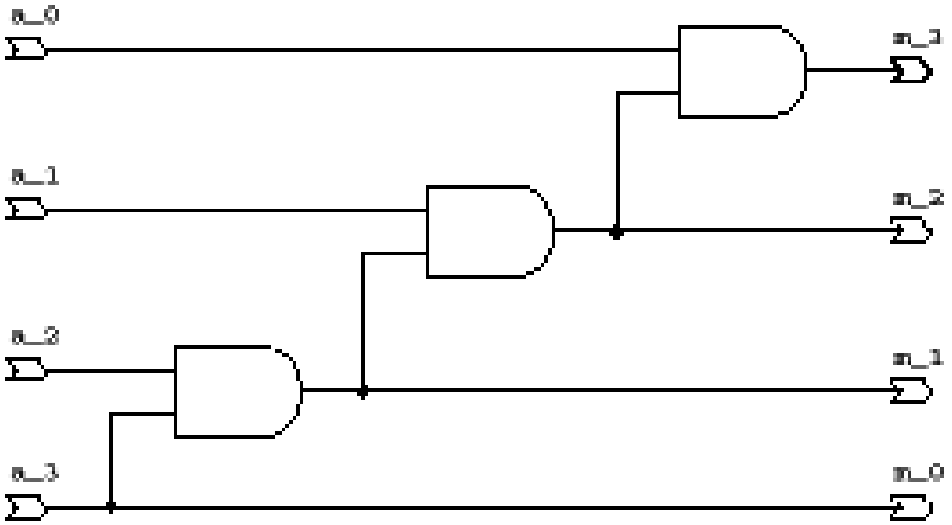
Counter

```
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;
```



Logic replication

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

architecture SIXTH of REPLICATOR 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 REPLICATOR;
```



Benefits

- Specification of the project independent of the technology
  - possibility of cooperation with many manufacturers
  - avoiding problems with technologies withdrawn
  - easy upgrades and improvements
- Low-level design automation
  - shorter development time
  - cost reduction
  - elimination of low-level errors
- Improvement of design quality
  - easy testing of optional technologies
  - verification of operation at high abstraction level
  - easy verification of implementations
  - modularity – easy design reuse

VHDL – yesterday

1980 DoD USA – start of the VHSIC technology program (Very High Speed Integrated Circuits)

1981 Woods Hole, Massachusetts – conference on the proposition of the future HDL standard

1983 DoD sets VHDL foundations: – Intermetrics, TI & IBM consortium gathers contract

1984 version 6.0 ready

1985 exemption from ITAR restrictions (US International Traffic and Arm Regulation), VHDL 7.2 with references submitted to IEEE to the standardization and further development

1987 IEEE Std 1076 released

1993 IEEE Std 1076-1993 amendment released

2000 IEEE Std 1076a-1993 erratum released

2003 IEEE Std 1076-2003 amendment released

2005 taking the initiative by Accelera

2006 IEEE Std 1076-2006 amendment released

2008 IEEE Std 1076-2008 amendment released

VHDL Analysis and Standarization Group (VASG)

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

VHDL – tomorrow

- Is hardware engineer still needed?
  - too small efficiency of the Von Neumann architecture
  - need of compensation of poor software performance by using the dedicated hardware
- Language: general or specific?
  - general (RTL)
    - close to the hardware implementation
    - architectural details included in the project
  - specific
    - close to the application (e.g. DSP: Matlab)
    - automatic generation of parallelized hardware
- Language: two approaches
  - new language, dedicated to the needs ⇨ adoption by users needed
  - adaptation of an existing language to the new context

VHDL – standards

- 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 Multivalue Logic System for VHDL Model Interoperability (Std_logic_1164)

VHDL – literature

„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. Sjoholm, L. Lindh

„VHDL made easy!”, D. Pellerin, D. Taylor

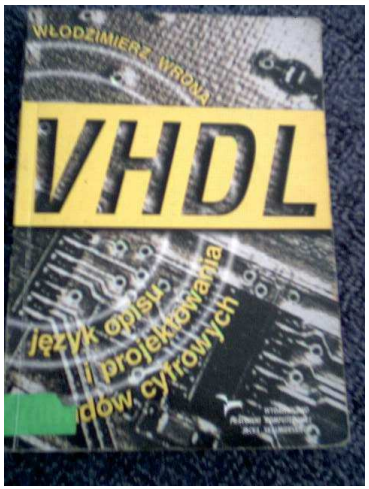
„VHDL answers to frequently asked questions”, B. Cohen

„VHDL and AHDL digital systems implementation”, F. A. Scarpino

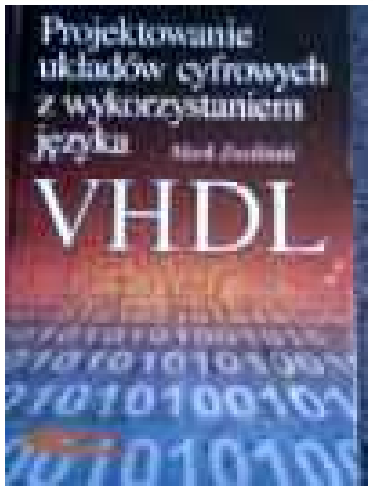
„Active-VHDL Series BOOK#2 – EVITA Interactive Tutorial”,
J. Mirkowski, M. Kapustka, Z. Skowroński, A. Biniszkiewicz

„VHDL: a logic synthesis approach”, D. Naylor, S. Jones

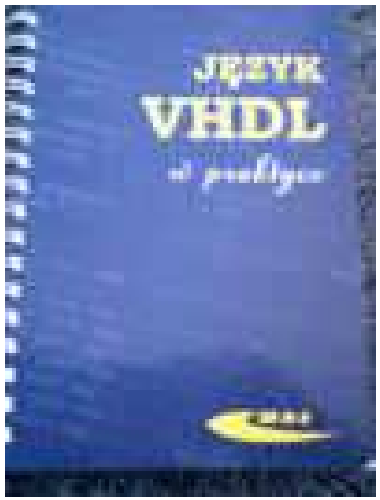
VHDL – literature



Włodzimierz Wrona



Mark Zwoliński



Józef Kalisz

Kevin Skahill

VHDL – Internet resources

- Discussion group: **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/

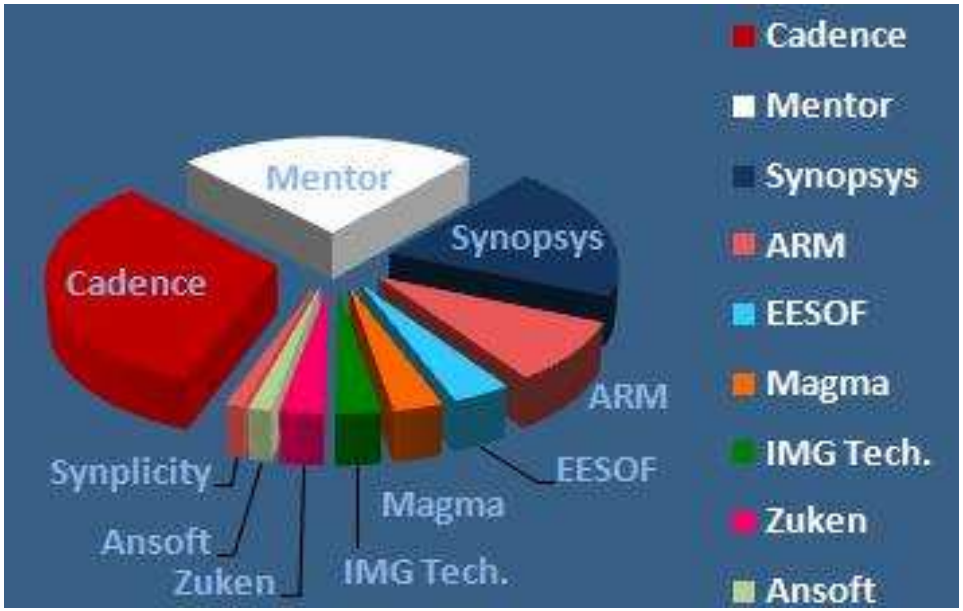
VHDL – tutorials

- 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

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

VHDL – EDA companies (products)



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

- **Green Mountain Computing Systems *(Direct VHDL)***

VHDL – Aldec Inc.

- **Products**
  - **Software (ActiveCAD, ActiveHDL)**
  - **Hardware-based Simulation Accelerators**
  - **IP Cores**

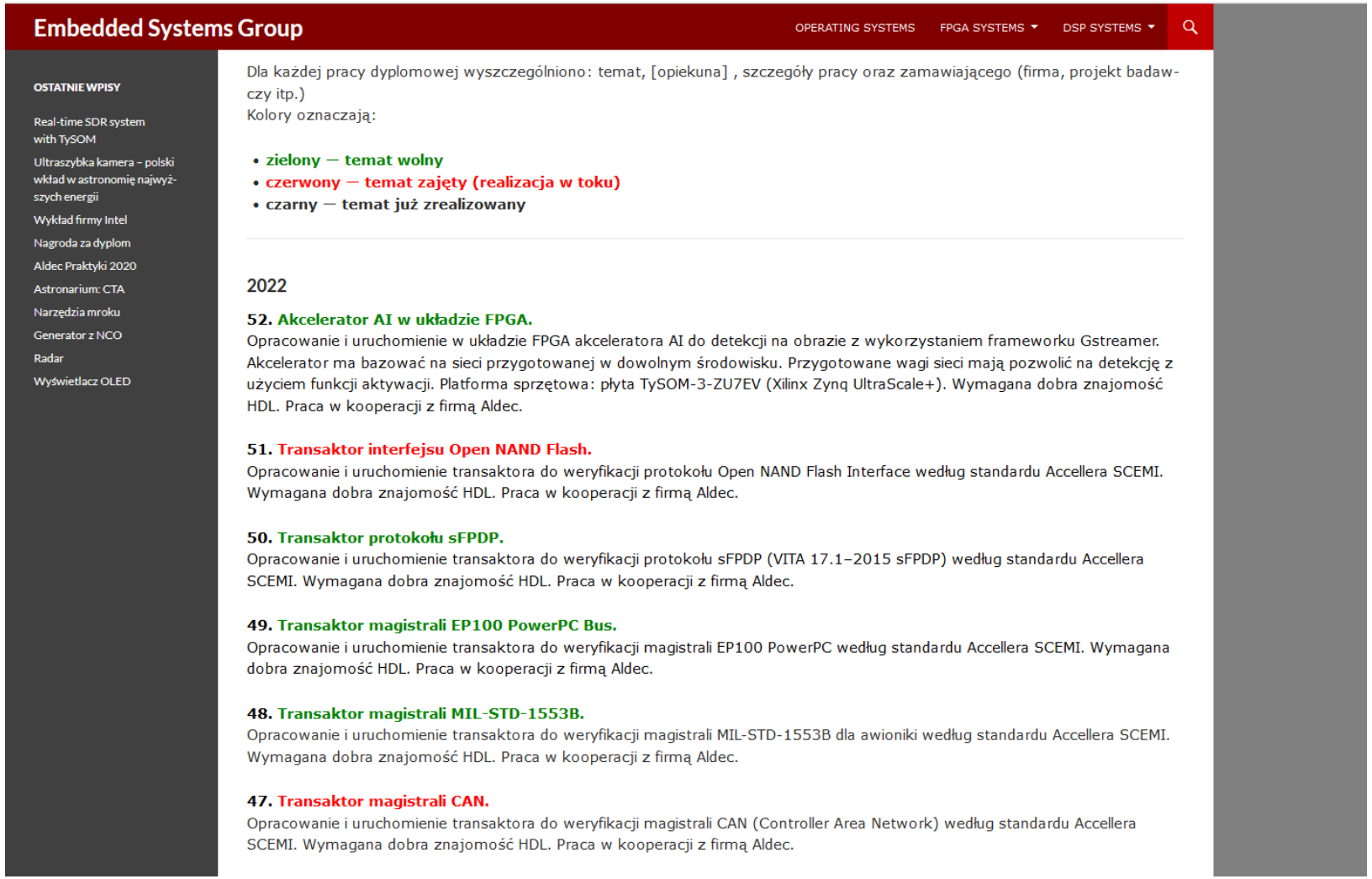
- **Donation to laboratory:**
  - **Computers – 11×PC (former)**
  - **Software (ActiveHDL)**
    - ***20 sites license (Professional Edition)***
    - ***Student Edition***
  - **HES accelerators**

- **<u>Department in Kraków!</u>**

VHDL – Aldec Inc.

- Student traineeships
- Diploma theses (IP Cores)
  - μP/μC: PIC17C4x, PIC16C5x, 8051-SoC
  - peripheral: 8251-SIO, 8255-PIO, 8257-DMA, 8259-INT
  - interfaces: CAN, UART/IrDA, USB, I2S, PS/2, VGA, I2C, SD, ...
  - telecom: Reed-Solomon, Viterbi, Utopia, DTMF, MP3, ADPCM, LDPC, Kasumi, ...
- Professional career

VHDL – thesis



Embedded Systems Group

OPERATING SYSTEMS FPGA SYSTEMS DSP SYSTEMS

OSTATNIE WPISY

- Real-time SDR system with TySOM
- Ultraszybka kamera – polski wkład w astronomię najwyższych energii
- Wykład firmy Intel
- Nagroda za dyplom
- Aldec Praktyki 2020
- Astronarium: CTA
- Narzędzia mroku
- Generator z NCO
- Radar
- Wyświetlacz OLED

Dla każdej pracy dyplomowej wyszczególniono: temat, [opiekuna] , szczegóły pracy oraz zamawiającego (firma, projekt badawczy itp.)
Kolory oznaczają:

- **zielony — temat wolny**
- **czerwony — temat zajęty (realizacja w toku)**
- **czarny — temat już zrealizowany**

2022

52. Akcelerator AI w układzie FPGA.
Opracowanie i uruchomienie w układzie FPGA akceleratora AI do detekcji na obrazie z wykorzystaniem frameworku Gstreamer. Akcelerator ma bazować na sieci przygotowanej w dowolnym środowisku. Przygotowane wagi sieci mają pozwolić na detekcję z użyciem funkcji aktywacji. Platforma sprzętowa: płyta TySOM-3-ZU7EV (Xilinx Zynq UltraScale+). Wymagana dobra znajomość HDL. Praca w kooperacji z firmą Aldec.

51. Transaktor interfejsu Open NAND Flash.
Opracowanie i uruchomienie transaktora do weryfikacji protokołu Open NAND Flash Interface według standardu Accellera SCEMI. Wymagana dobra znajomość HDL. Praca w kooperacji z firmą Aldec.

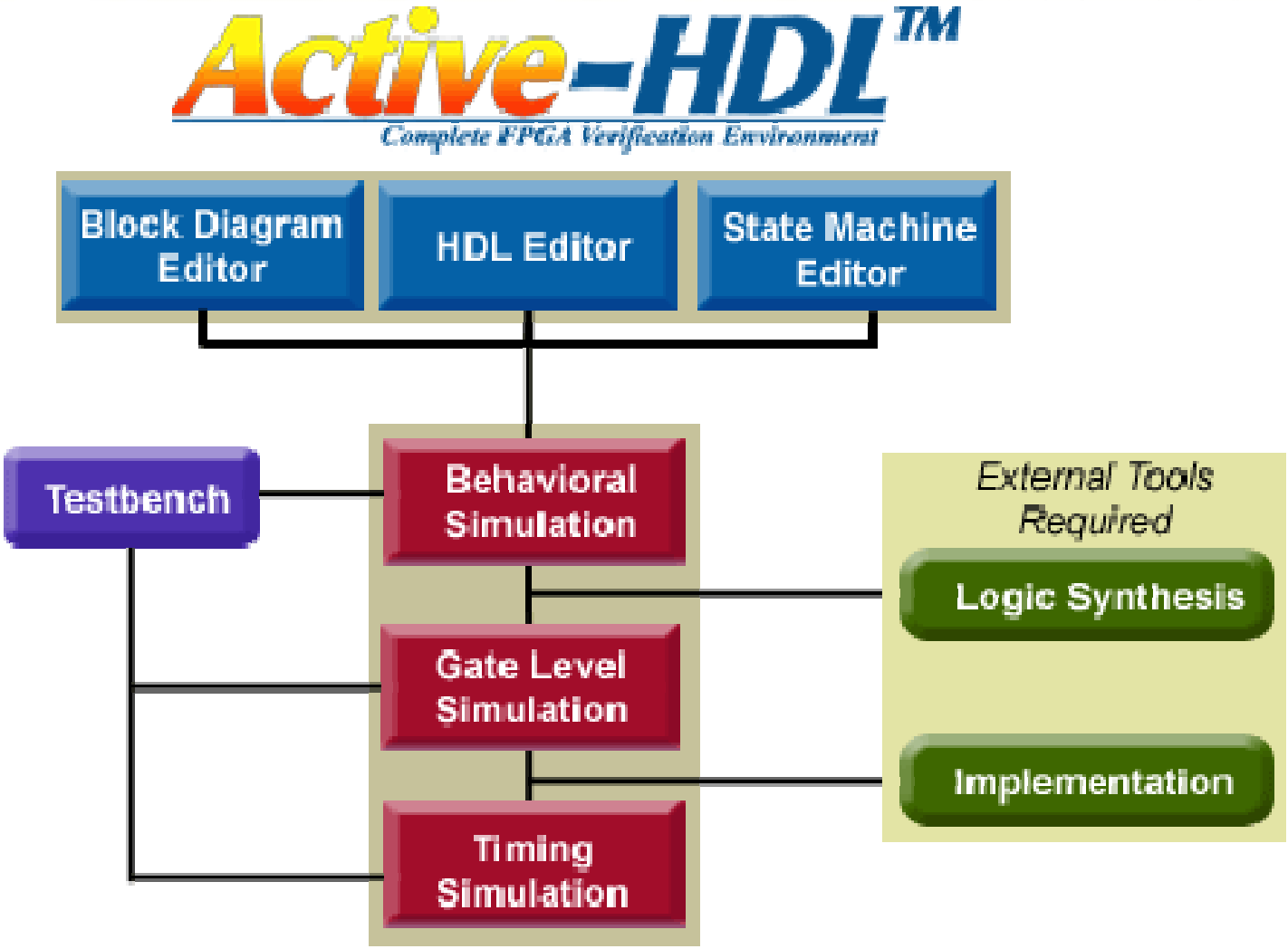
50. Transaktor protokołu sFPDP.
Opracowanie i uruchomienie transaktora do weryfikacji protokołu sFPDP (VITA 17.1–2015 sFPDP) według standardu Accellera SCEMI. Wymagana dobra znajomość HDL. Praca w kooperacji z firmą Aldec.

49. Transaktor magistrali EP100 PowerPC Bus.
Opracowanie i uruchomienie transaktora do weryfikacji magistrali EP100 PowerPC według standardu Accellera SCEMI. Wymagana dobra znajomość HDL. Praca w kooperacji z firmą Aldec.

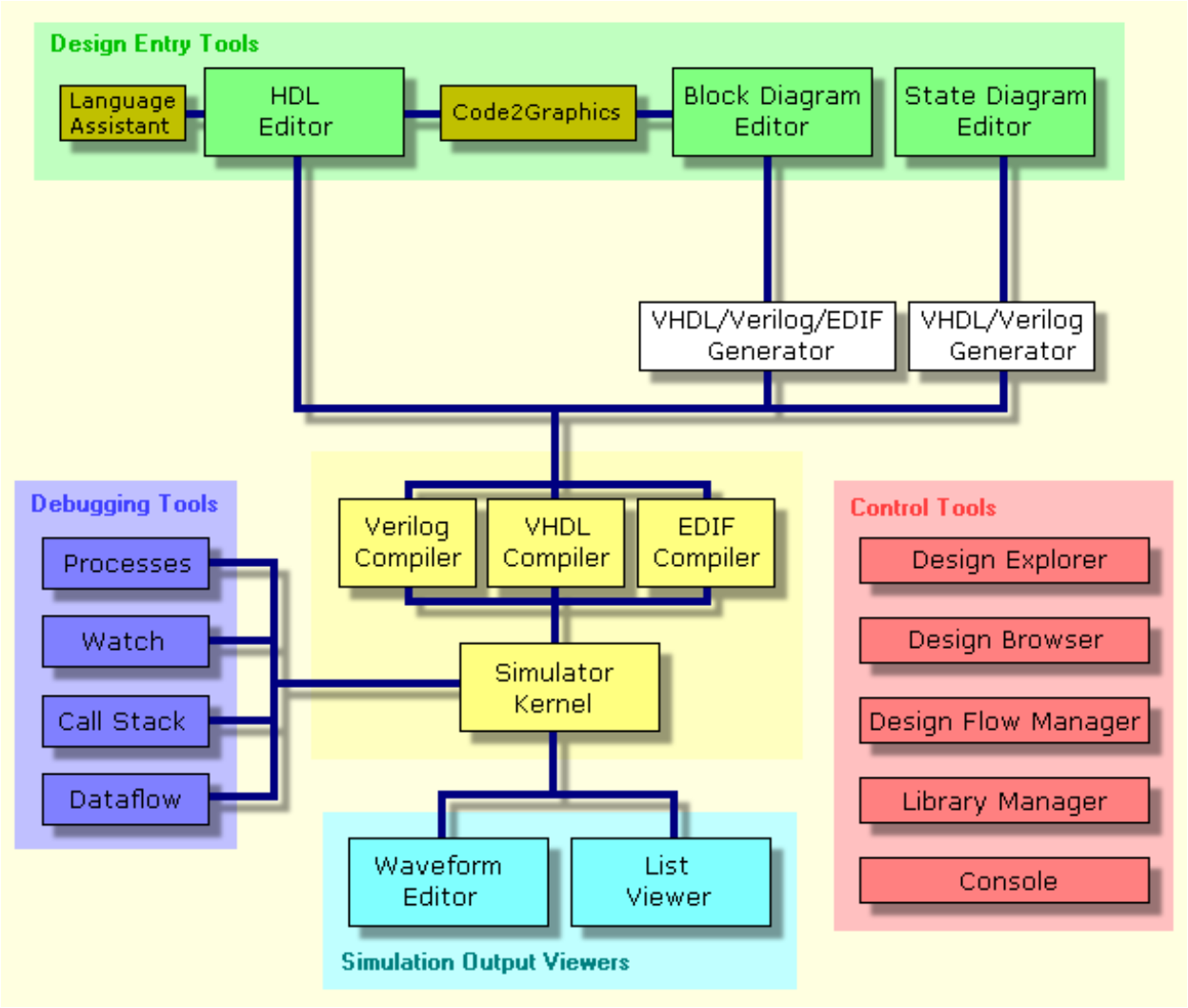
48. Transaktor magistrali MIL-STD-1553B.
Opracowanie i uruchomienie transaktora do weryfikacji magistrali MIL-STD-1553B dla awioniki według standardu Accellera SCEMI. Wymagana dobra znajomość HDL. Praca w kooperacji z firmą Aldec.

47. Transaktor magistrali CAN.
Opracowanie i uruchomienie transaktora do weryfikacji magistrali CAN (Controller Area Network) według standardu Accellera SCEMI. Wymagana dobra znajomość HDL. Praca w kooperacji z firmą Aldec.

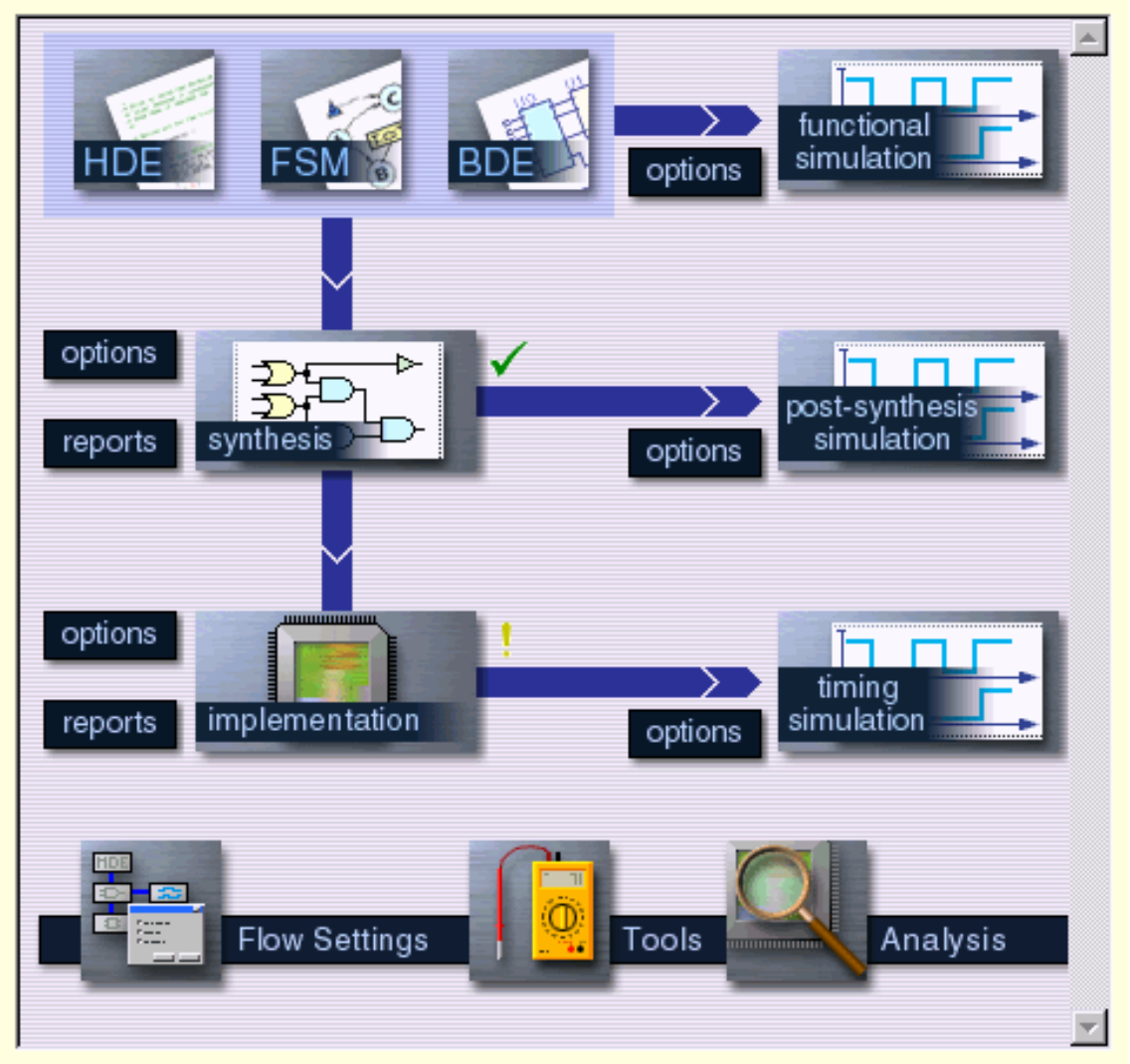
ActiveHDL – modules



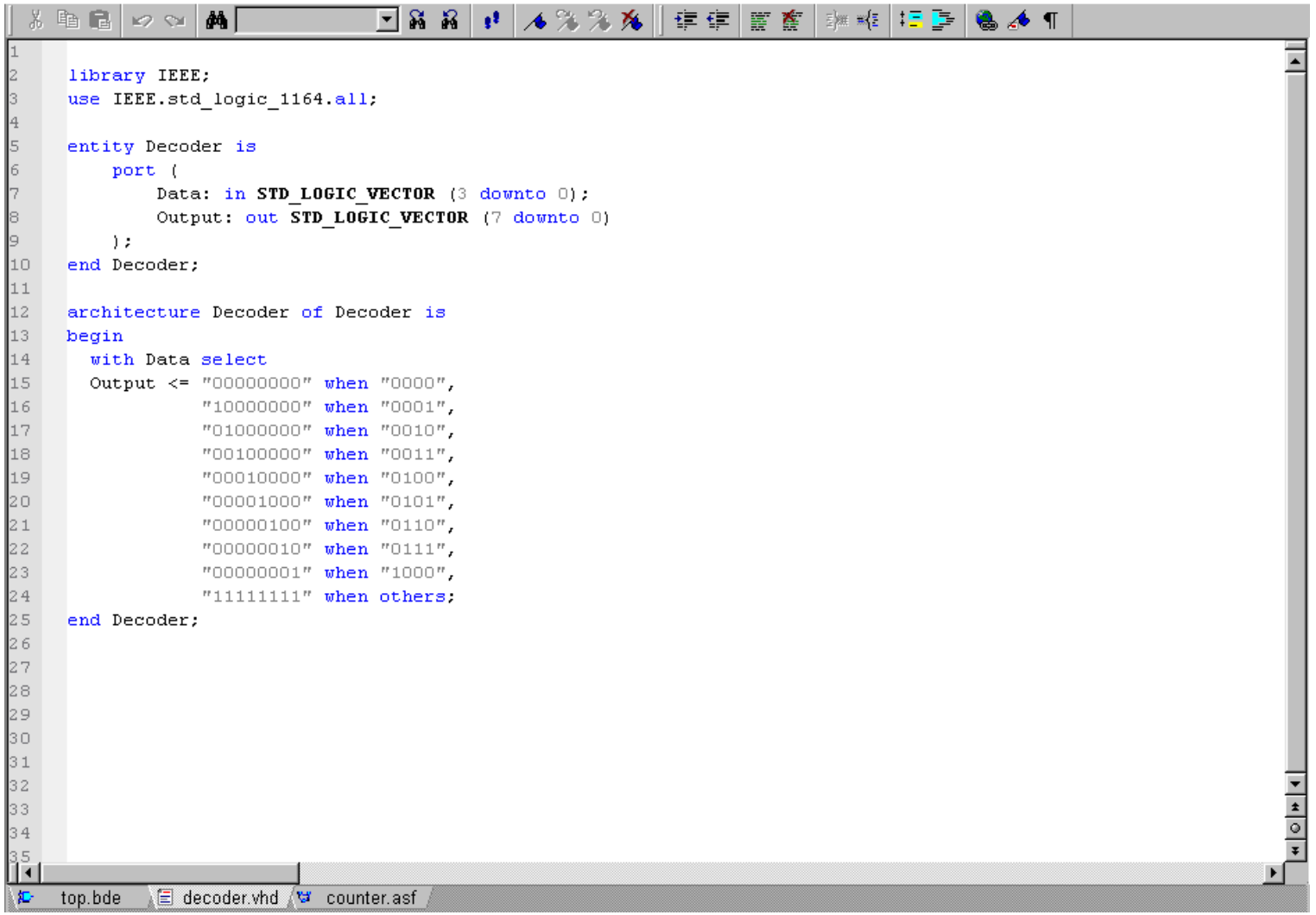
ActiveHDL – modules



ActiveHDL – Design Flow



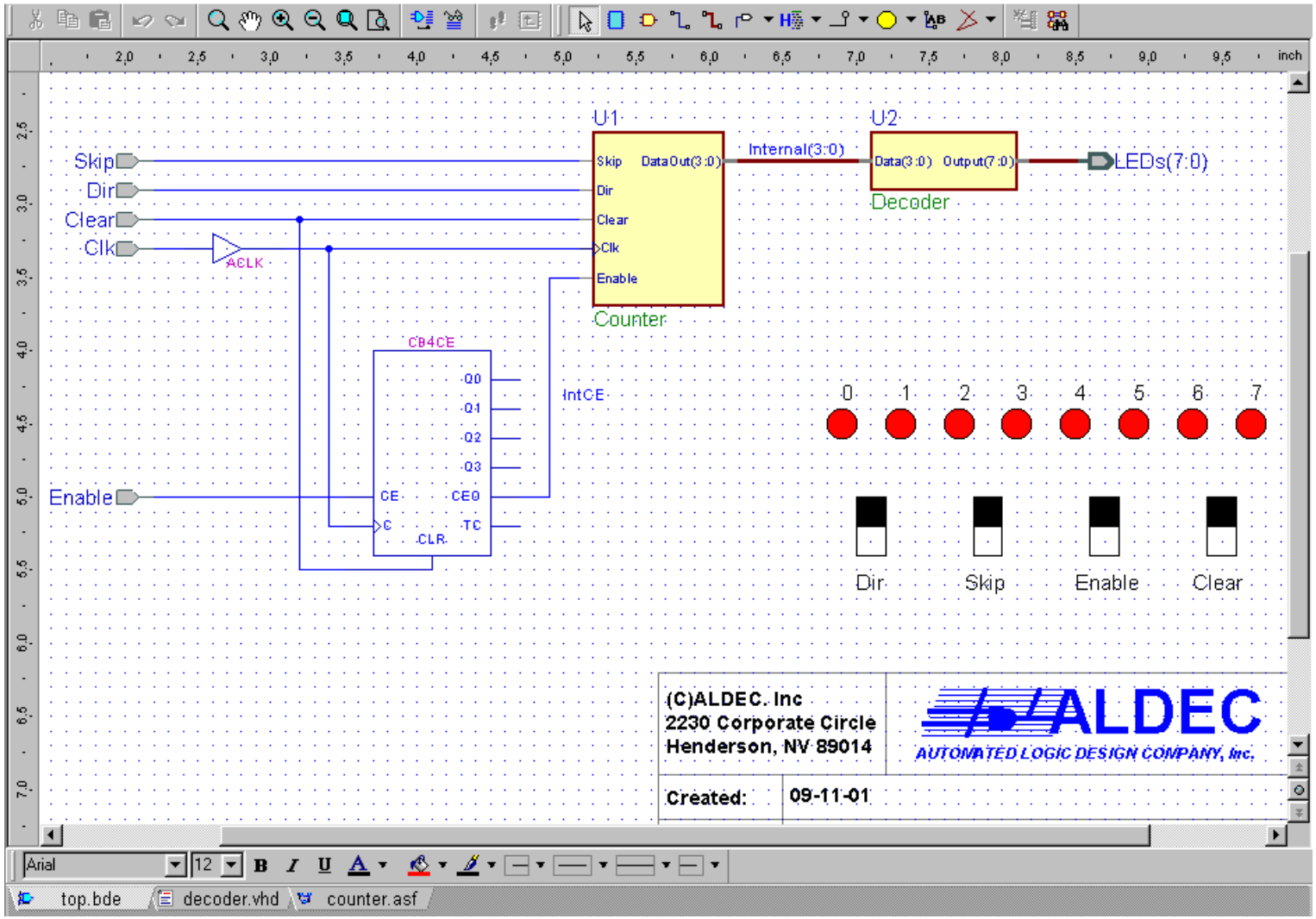
ActiveHDL – HDL editor



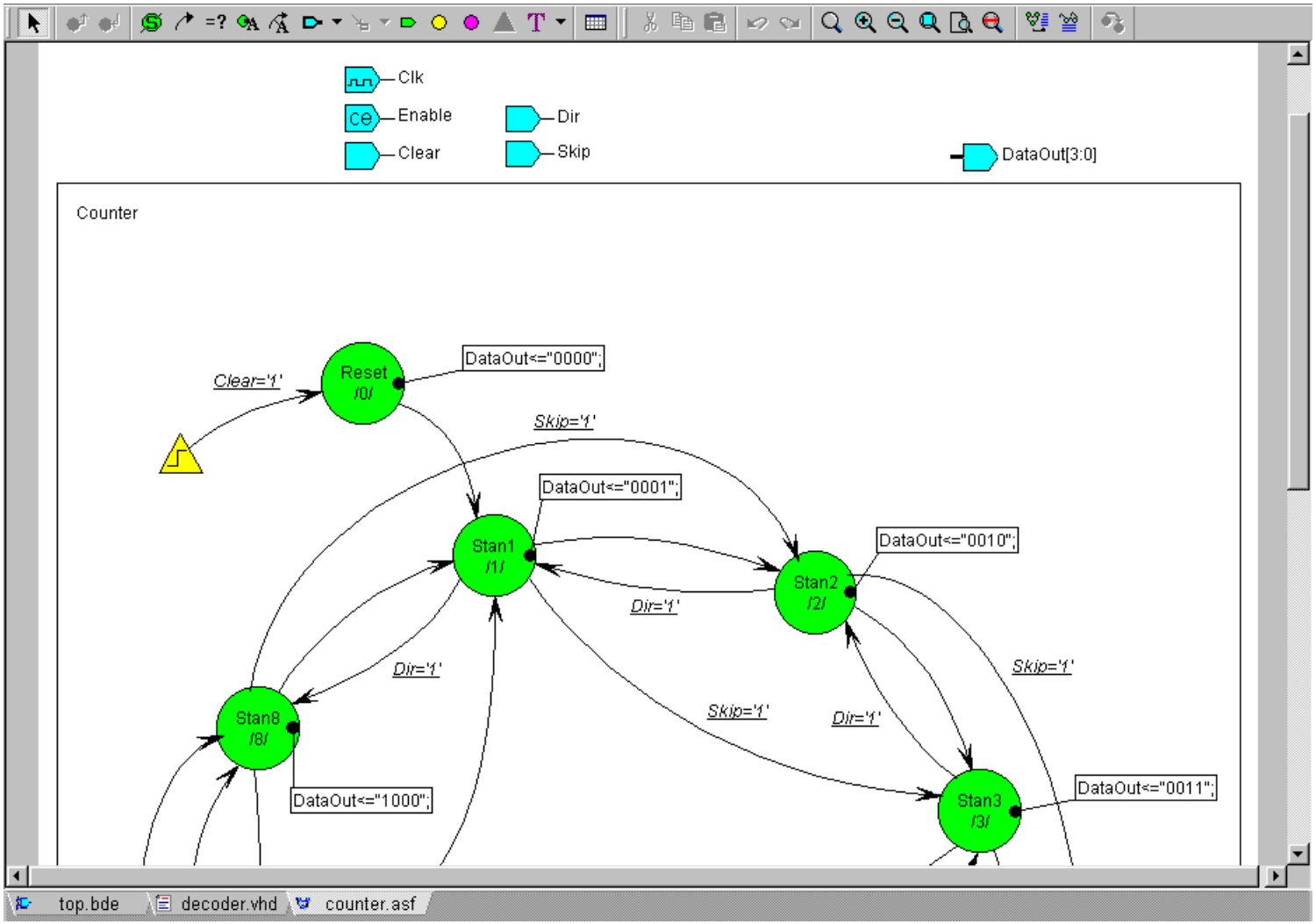
```
1
2   library IEEE;
3   use IEEE.std_logic_1164.all;
4
5   entity Decoder is
6       port (
7           Data: in STD_LOGIC_VECTOR (3 downto 0);
8           Output: out STD_LOGIC_VECTOR (7 downto 0)
9       );
10  end Decoder;
11
12  architecture Decoder of Decoder is
13  begin
14    with Data select
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

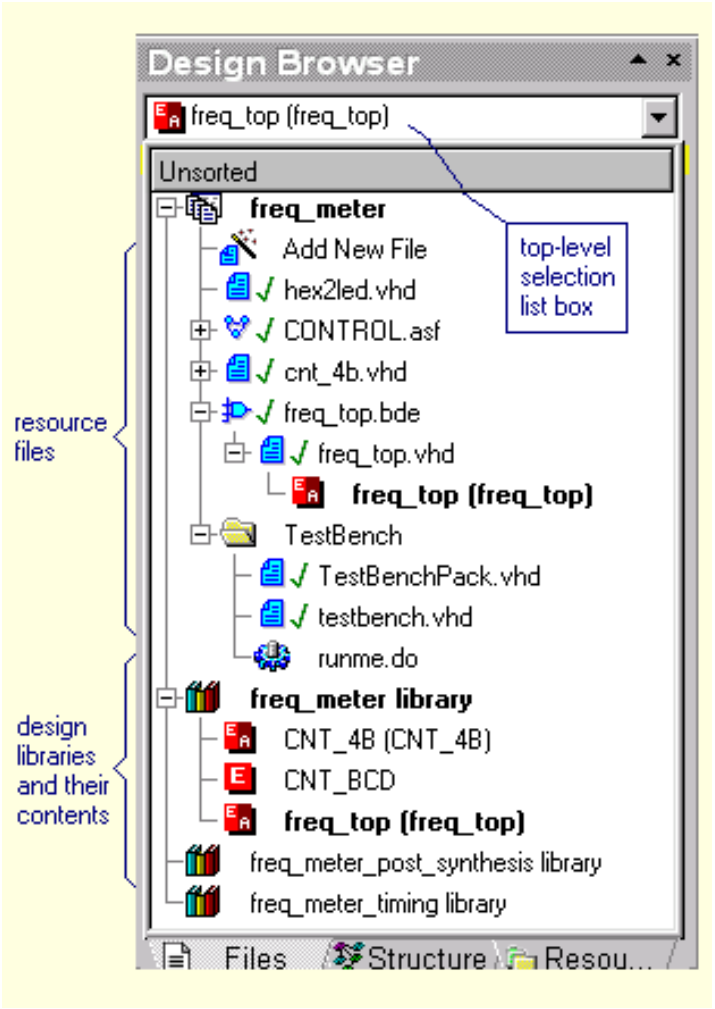
ActiveHDL – BDEditor

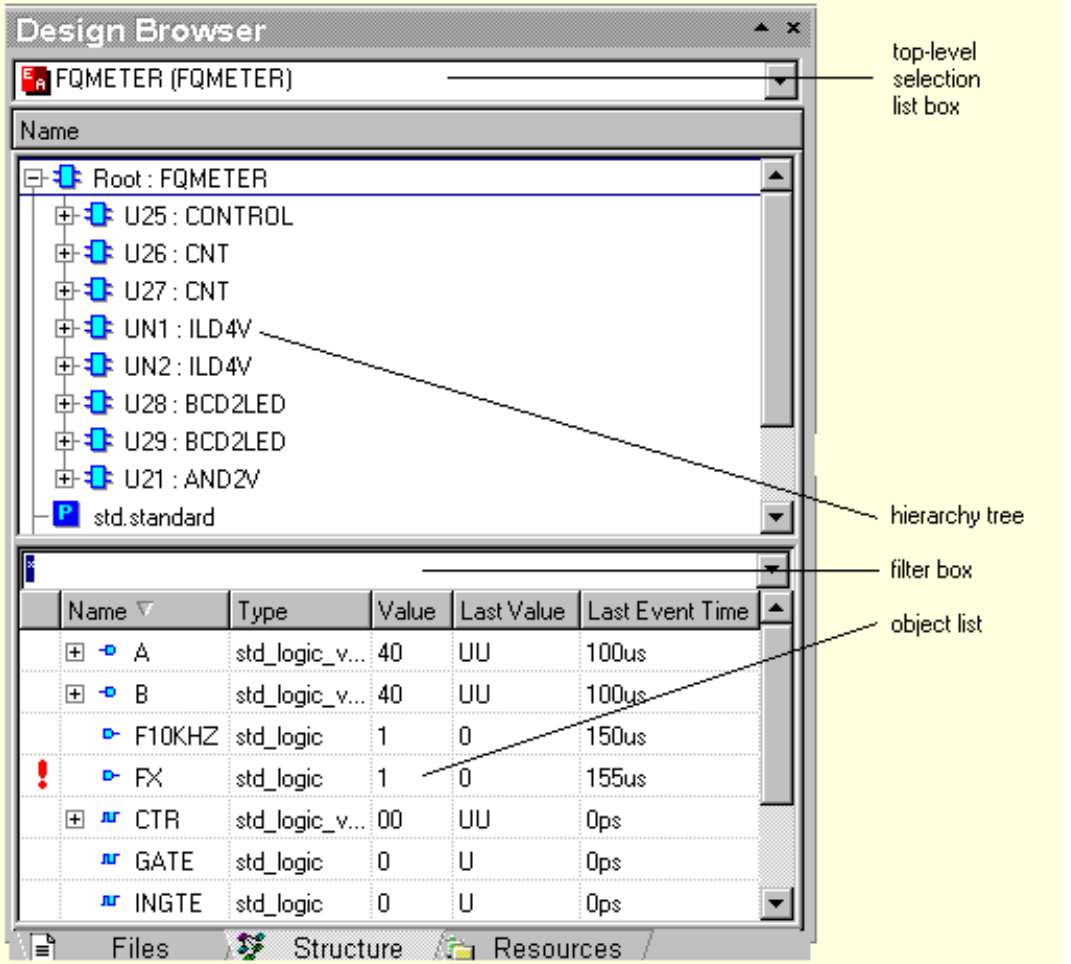


ActiveHDL – FSM editor

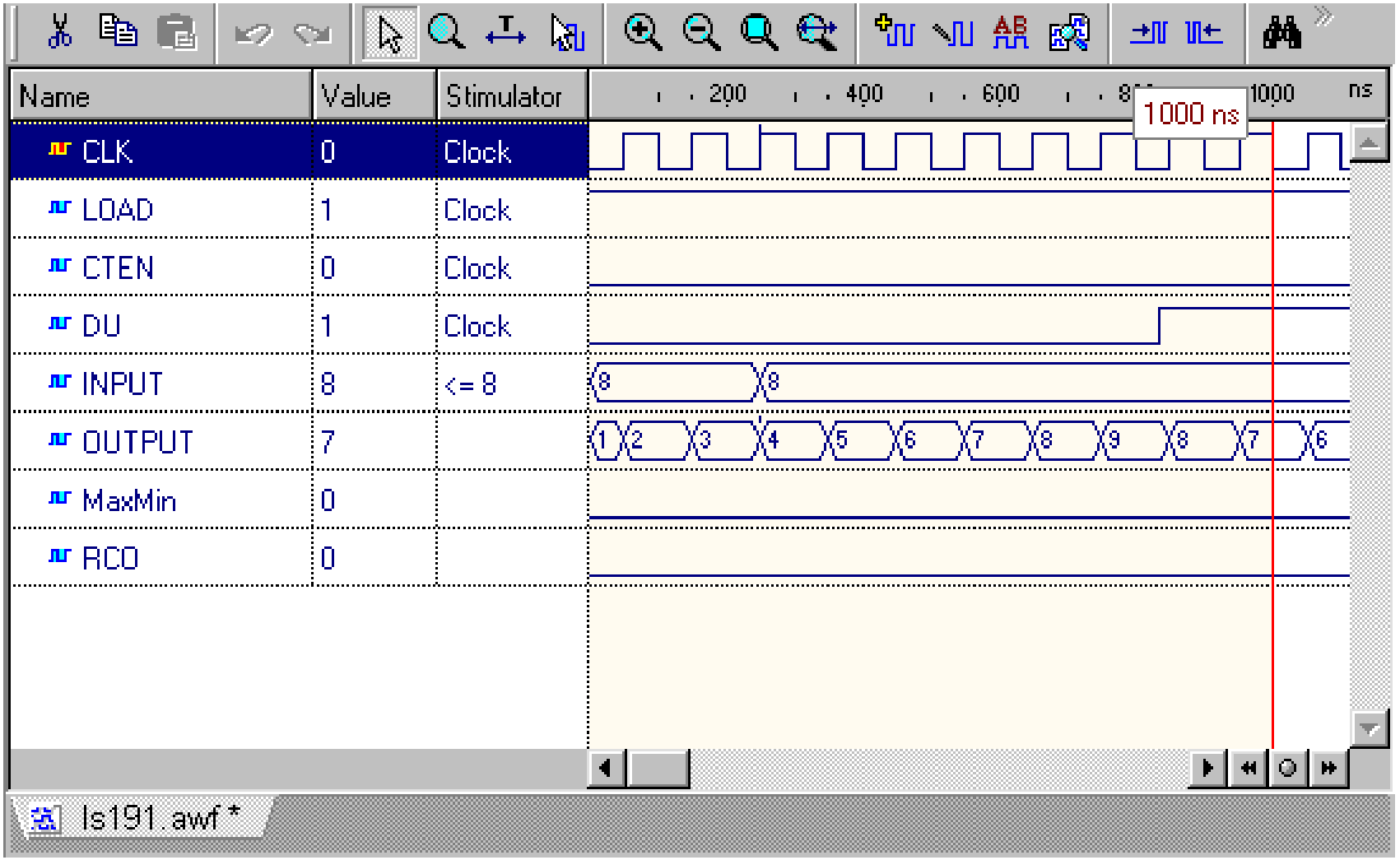


ActiveHDL – Design Browser

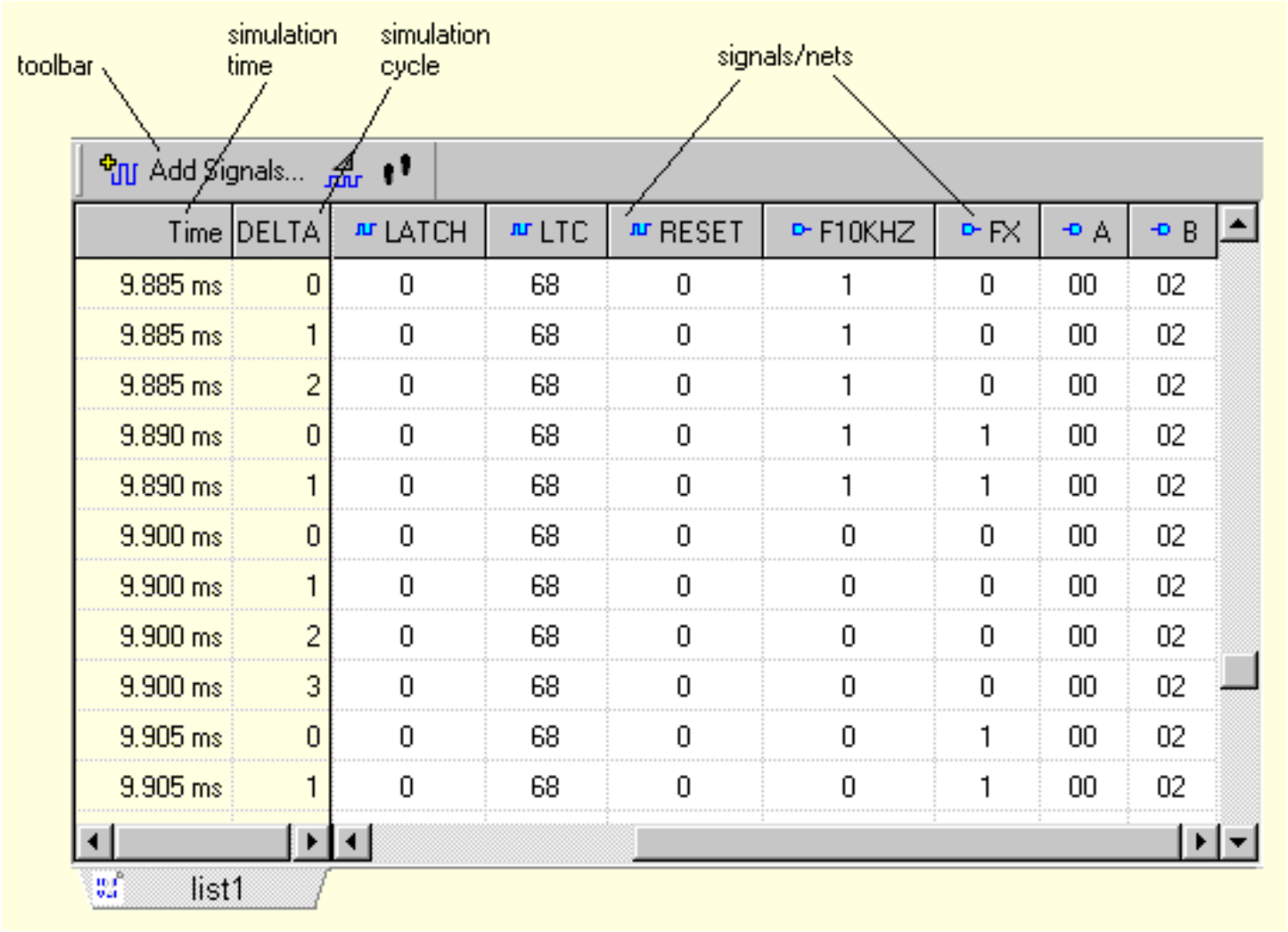




ActiveHDL – Waveform Viewer

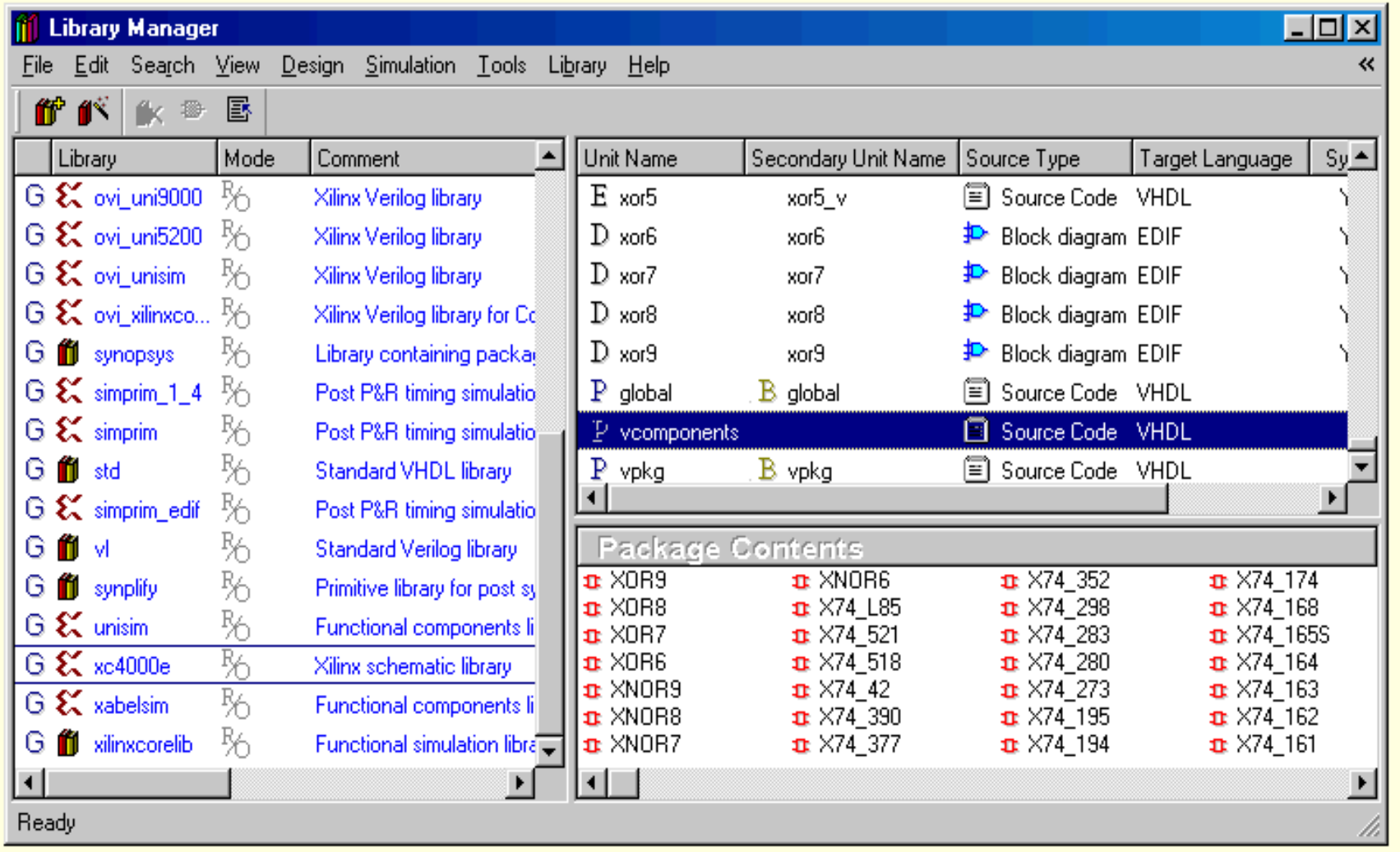


ActiveHDL – List Viewer



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




To be continued…