



Module: Electronics & Telecommunication, 5rd year Hardware Acceleration of Telecommunication Protocols

Introduction



Communication with teachers

Paweł J. Rajda, PhD EE pjrajda@agh.edu.pl

Jerzy Kasperek PhD EE kasperek@agh.edu.pl

C-3, room 502, phone (12) 617 3980

<http://www.embedded.agh.edu.pl/fpga-embedded-systems/fpga-dydaktyka/przedmioty/>

Embedded Systems Group

OPERATING SYSTEMS FPGA SYSTEMS DSP SYSTEMS

PRZEDMIOTY

WdE
LP
JOS
JOS z
HDL
PSC
ZZPSC
PUC
PLD
MPIMS

NAJNOWSZE WIŚPIY
Astronarium: CTA
Narzędzia mroku

PRZEDMIOTY

Przedmioty prowadzone przez pracownię Systemów Wbudowanych FPGA (wybór w lewym menu):

- **WdE**: Wprowadzenie do Elektroniki (E + EIT 1st)
- **LP**: Laboratorium Projektowe (EIT 1st)
- **JOS**: Języki Opisu Sprzętu (EIT 1st)
- **JOS z**: Języki Opisu Sprzętu (EIT zaoczne)
- **HDL**: Hardware Description Languages (EIT 1st)
- **PSC**: Projektowanie Systemów Cyfrowych (E 1st)
- **ZZPSC**: Zastosowane Zagadnienie Projektowania Systemów Cyfrowych (E + EIT 1st)
- **PLD**: Programowalne Układy Cyfrowe (1 2st) teraz pod nazwą **SAPT**: Sprzętowa akceleracja protokołów telekomunikacyjnych
- **PLD**: Programmable Logic Devices (1 2st) now **HATP**: Hardware Acceleration of Telecommunication Protocols
- **MPIMS**: Metody Projektowania i Modelowania Systemów (E + EIT 2st)



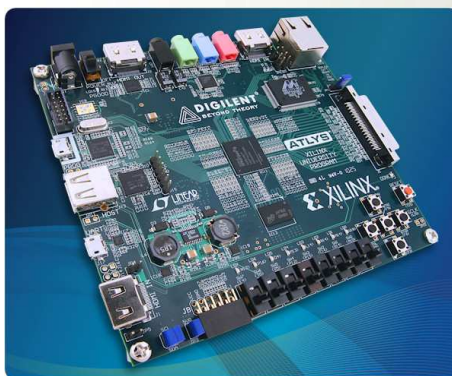
Hardware Acceleration of Telecommunication Protocols

- Hardware Description Languages - VHDL Verilog
- PLD architectures (SPLD,CPLD,FPGA)
- Design methodology (AHDL/MATLAB/LabView)
- Verification techniques (TB,HES)
- DSP in FPGA
- Embedded systems „softprocesor” – MicroBlaze, Nios
- PLD telecomm applications (NetFPGA)
- High Level Synthesis (Impuls C)



Lab

<http://www.embedded.agh.edu.pl/fpga-embedded-systems/fpga-dydaktyka/sprzet/>



DIGILENT

Hardware examples

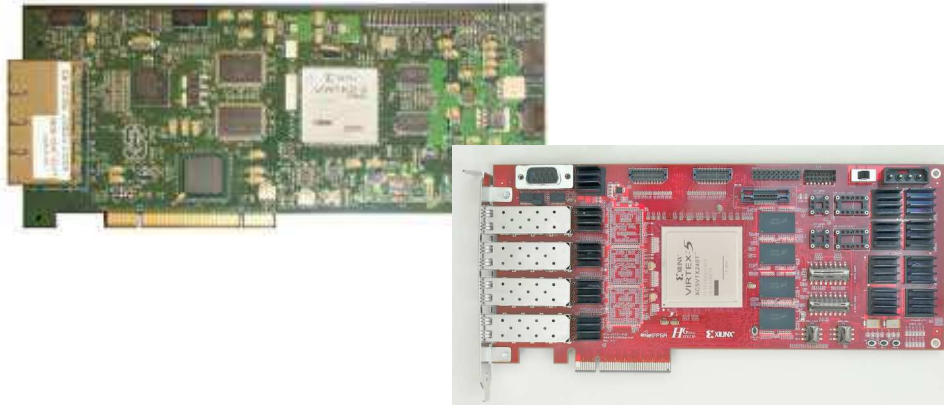
Nexys™ 3 Spartan-6 FPGA Board

Atlys™ Spartan-6 FPGA Development Board



Lab

Project NetFPGA



Hardware

NetFPGA1G NetFPGA10G

[Lab grading policy](#) [presence /tutorials/projects](#)

Rajda & Kasperek © 2017 Katedra Elektroniki IET AGH

5



Today's agenda VHDL 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

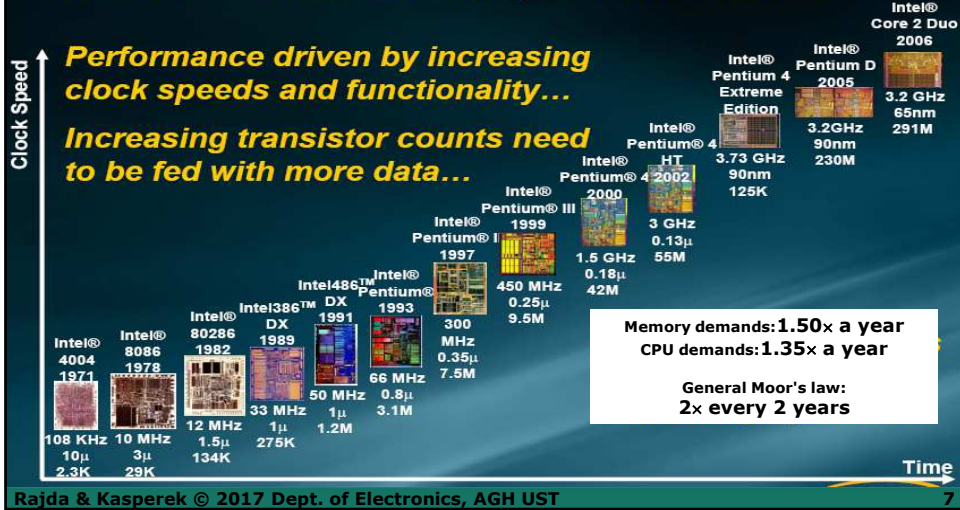
Rajda & Kasperek © 2017 Dept. of Electronics, AGH UST

6



VHDL – what for?
Design complexity

Moore's Law Driving Performance



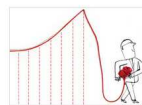
<http://spectrum.ieee.org/static/special-report-50-years-of-moores-law>
50 years of Moore's Law



Fifty years ago this month, Gordon Moore forecast a bright future for electronics. His ideas were later distilled into a single organizing principle—Moore's Law—that has driven technology forward at a staggering clip. We have all benefited from this miraculous development, which has forcefully shaped our modern world. In this special report, we find that the end won't be sudden and apocalyptic but rather gradual and complicated. Moore's Law truly is the gift that keeps on giving—and surprising, as well.



The Multiple Lives of Moore's Law
Why Gordon Moore's grand prediction has endured for 50 years
20 May



The Death of Moore's Law Will Spur Innovation
As transistors stop shrinking, open-source hardware will have its day
27 May



Moore's Law Will Be Slowing Down, But Not Stopping
Efficient optimization may be tough, but there's still room to drive down power consumption in modern computers
21 May



Gordon Moore: The Man Whose Name Means Progress
The visionary engineer reflects on 50 years of Moore's Law
20 May



Moore's Curse
There is a dark side to the revolution in electronics: unjustified technological expectations
19 Mar



Graphic: Transistor Production Has Reached Astronomical Scales
A look at Moore's Law in action
2 Apr



Q&A: Carver Mead
A longtime collaborator recalls the first time he met Gordon Moore
15 Apr



Opinion: What Kind of Thing Is Moore's Law?
The trend has more to do with collective behavior than the laws of nature
20 Mar



Opinion: Is This Really The Anniversary of Moore's Law?
Gordon Moore's path to his famous prediction
7 Apr



VHDL – what for? Design complexity

Strong need for proper tool:

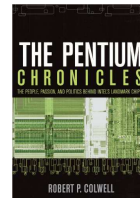
- 1970 - INTEL 4004 4 engineers ~1k transistors
- 1982 - INTEL 80286 20 engineers ~100k transistors
- 1992 - INTEL PENTIUM 100 engineers ~3M transistors
- 20?? - ??? ??? engineers ??? ~150M transistors ???

Contemporary requirements:

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



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



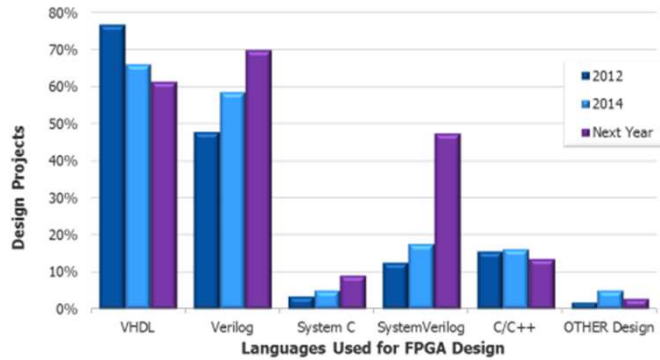
Hardware Description Languages

AGH

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

- **VHDL**
- **VERILOG, System VERILOG**
- **C, C++, Handel-C, System-C, Impuls -C**
- **others...**

FPGA Design Language Adoption Trends

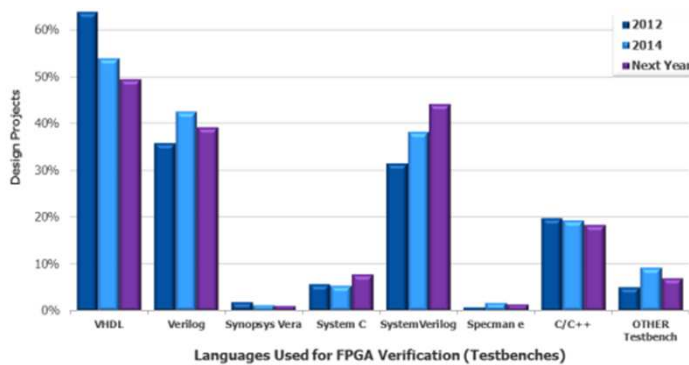


Source: Wilson Research Group and Mentor Graphics, 2014 Functional Verification Study

© 2015 Mentor Graphics Corp. Company Confidential
www.mentor.com



FPGA Verification Language Adoption Trends



Source: Wilson Research Group and Mentor Graphics, 2014 Functional Verification Study

© 2015 Mentor Graphics Corp. Company Confidential
www.mentor.com

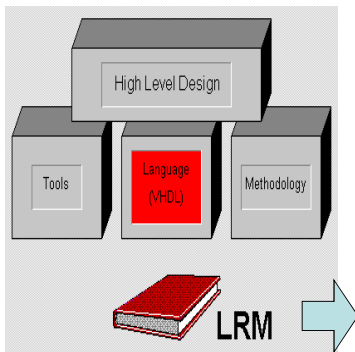




VHDL – what's this? Definition

VHDL - V_{HSIC} H_{ardware} D_{escription} L_{anguage}

↳ Very High Speed Integrated Circuit

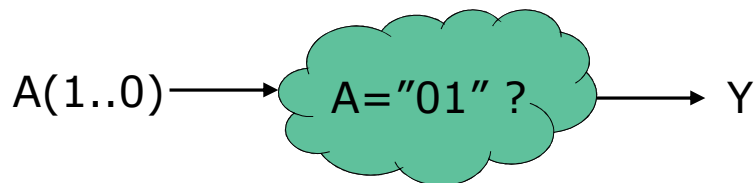


It is "a formal notation intended for use in all phases of the creation of electronic systems. (...) it supports the development, verification, synthesis, and testing of hardware designs, the communication of hardware design data ..."

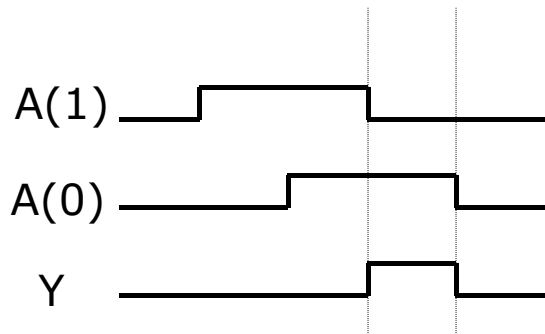
[IEEE Standard VHDL Language Reference Manual]



VHDL – how, where, when? Modeling

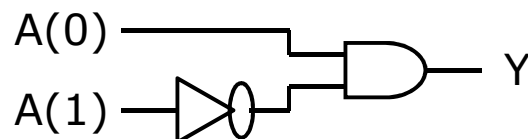


```
Y <= '1' when A = "01" else '0' ;
```



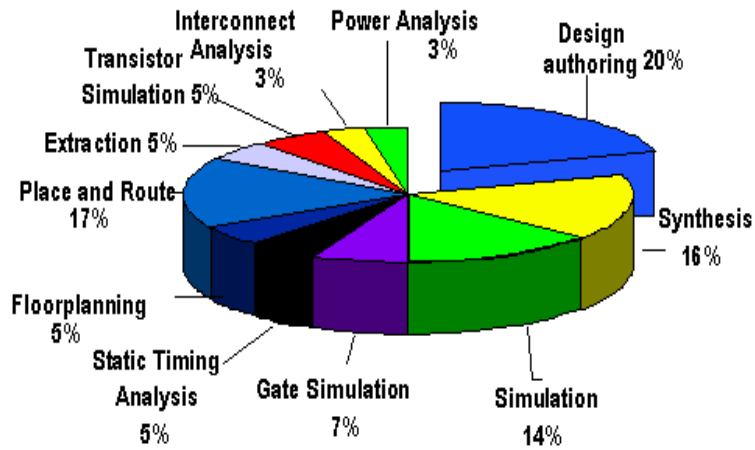
```
Stim <= "10", "11" after 4 ns ...
```

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...).





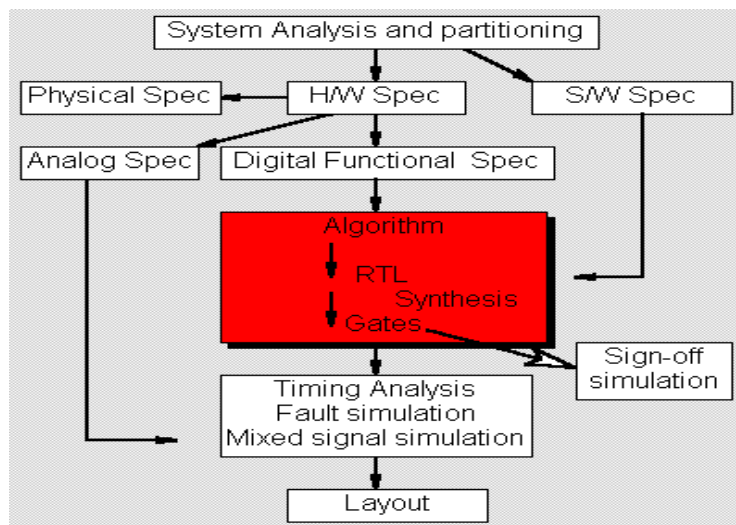
VHDL – how, where, when? Design phases

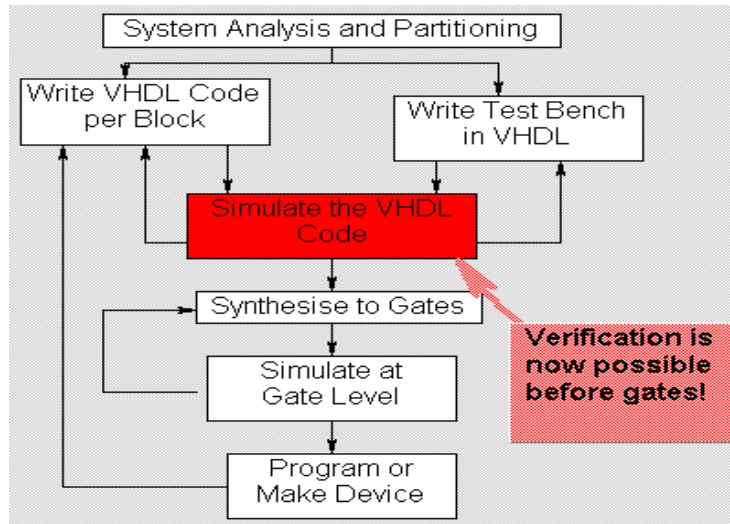


Average iterations between design and layout = 20



VHDL – how, where, when? System design flow





entity name

```

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

architecture style of name

```

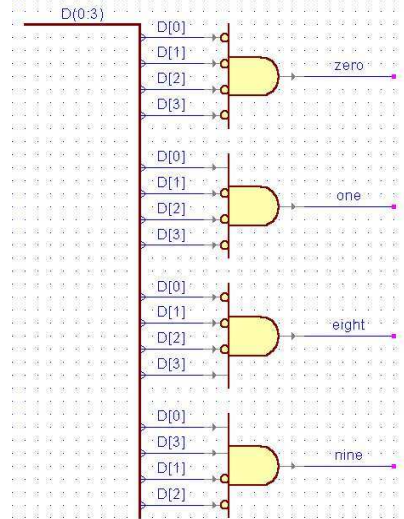
architecture BEHAVIORAL of COMPARE is
begin
  process (A,B)
  begin
    if (A=B) then
      C <= '1';
    else
      C <= '0';
    end if;
  end process;
end BEHAVIORAL;
  
```



VHDL – examples 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;

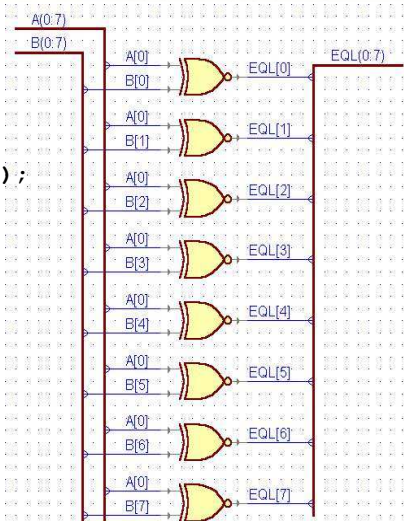
architecture FIRST of DECODER is
begin
  ZERO <= (D="0000");
  ONE <= (D="0001");
  EIGHT <= (D="1000");
  NINE <= (D="1001");
end FIRST;
```



VHDL – examples 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;
```



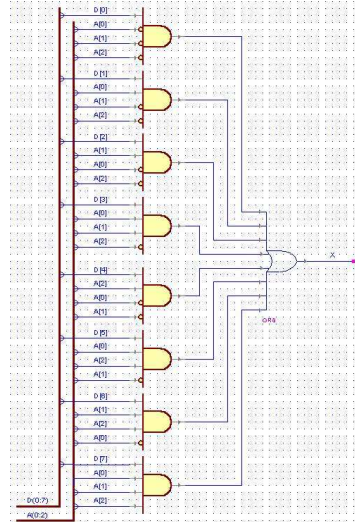
Fault ! →



VHDL – examples 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;
```

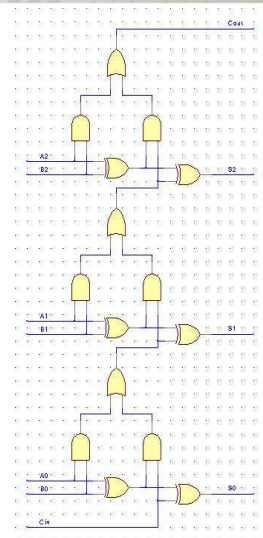


VHDL – examples Adder

```
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;

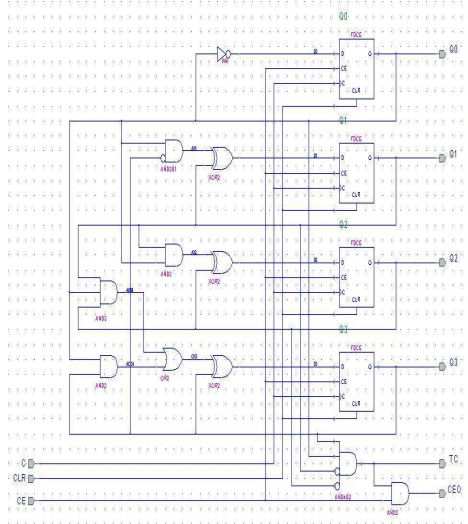
architecture FOURTH of SUM is
  signal V: std_logic_vector (0 to 3);
begin
  V <= A + B + Cin;
  S <= V(0 to 2);
  Cout <= V(3);
end FOURTH;
```





VHDL – examples Counter

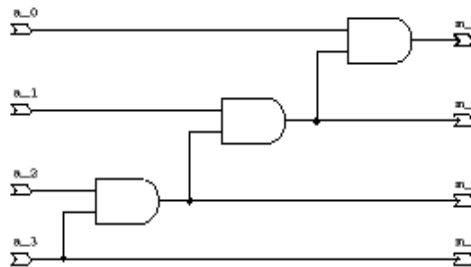
```
architecture FIFTH of COUNT is
begin
  process (C, CLR)
  begin
    if CLR='1' then
      Q := 0;
    elsif C='1' and C'event then
      if CE='1' then
        if Q = 9 then
          Q := 0;
        else
          Q := Q + 1;
        end if;
      end if;
    end if;
  end process;
end FIFTH;
```



VHDL – examples 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;
```





VHDL – so what we have?

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

AGH

- 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

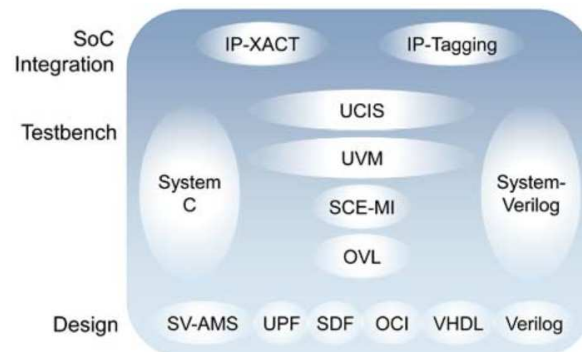


VHDL – today



http://www.eda.org/images/about/Accellera_Overview_2017.pdf

Accellera Systems Initiative
EDA and IP Design Standards and Initiatives



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

AGH

- 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)



VHDL – literature

AGH

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



VHDL –Internet resources

AGH

- Discussion group: **comp.lang.vhdl** (little outdated ☹)
- EDA Industry Working Groups homepage: <http://www.eda.org/>
<http://www.edac.org/>
- Accellera: <http://www.accellera.org/>
- Design Automation Cafe: <http://www.dacafe.com/>
- SOC Central <http://www.soccentral.com/>
- Doulos High Level Design Web site: <http://www.doulos.com/>
- RASSP WWW site: <http://www.eda.org/rassp/>
- The Hamburg VHDL Archive:
<http://tams-www.informatik.uni-hamburg.de/research/vlsi/vhdl/>



VHDL – tutorials

AGH

- 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>
- Interactive VHDL Tutorial from Aldec, Inc.:
<http://www.aldec.com/products/tutorials/>
- VHDL Verification Course by Stefan Doll:
<https://pl.scribd.com/document/48441518/Stefan-Doll-VHDL-Verification-Course-UPDATED>



VHDL – free IP cores

- OpenIP home page: <http://www.opencores.org/>
- The Hamburg VHDL archive:
<http://tams-www.informatik.uni-hamburg.de/vhdl/index.php?content=06-models>
- 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

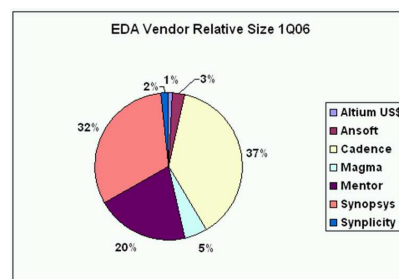


VHDL – EDA companies (products)

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

- Xilinx (*XST*)
- Altera (*Quartus*)

- Green Mountain Computing Systems (*Direct VHDL*)





VHDL – Aldec Inc.



- **Products**
 - Software (ActiveCAD, ActiveHDL)
 - Hardware-based Simulation Accelerators
 - IP Cores
- **Donation to the laboratory:**
 - Computers – 11×PC (former)
 - Software (ActiveHDL)
 - 20 sites license (*Professional Edition*)
 - *Student Edition*
 - HES accelerators
- **Department in Kraków!**



VHDL – Aldec Inc.

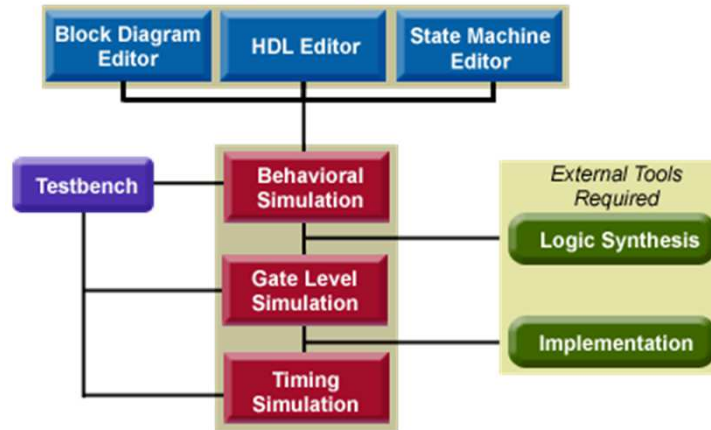


- **Student traineeships**
- **Diploma theses (IP Cores)**
 - μ P/ μ C: PIC17C4x, PIC16C5x, 8051-SoC
 - peripheral: 8251-SIO, 8255-PIO, 8257-DMA, 8259-INT
 - interfaces: CAN, UART/IrDA, USB, I2S, PS/2, VGA, I2C, SD, ...
 - telecom: Reed-Solomon, Viterbi, Utopia, DTMF, MP3, ADPCM, LDPC, Kasumi, ...
-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

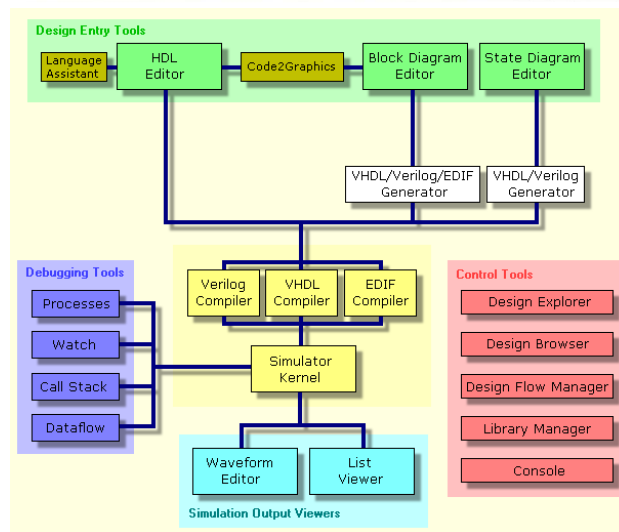


ActiveHDL – modules

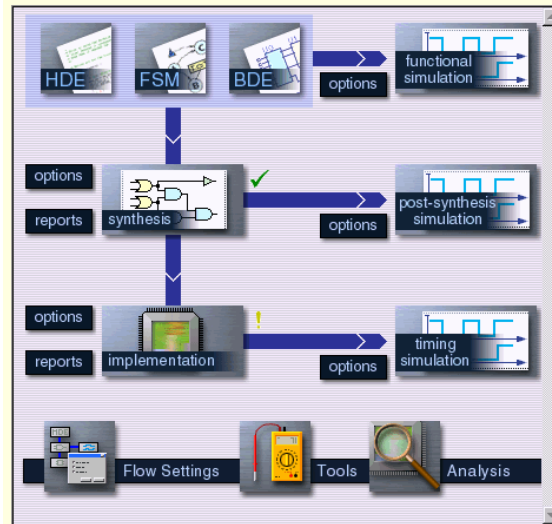
Active-HDL™
Complete FPGA Verification Environment



ActiveHDL – modules



ActiveHDL – Design Flow



ActiveHDL – HDL editor

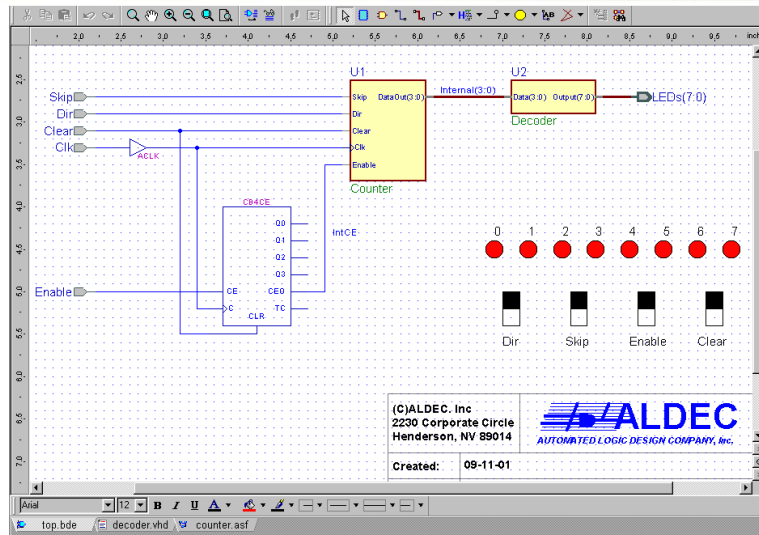
```

1  library IEEE;
2  use IEEE.std_logic_1164.all;
3
4  entity Decoder is
5  port (
6      Data: in STD_LOGIC_VECTOR (3 downto 0);
7      Output: out STD_LOGIC_VECTOR (7 downto 0)
8  );
9  end Decoder;
10
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

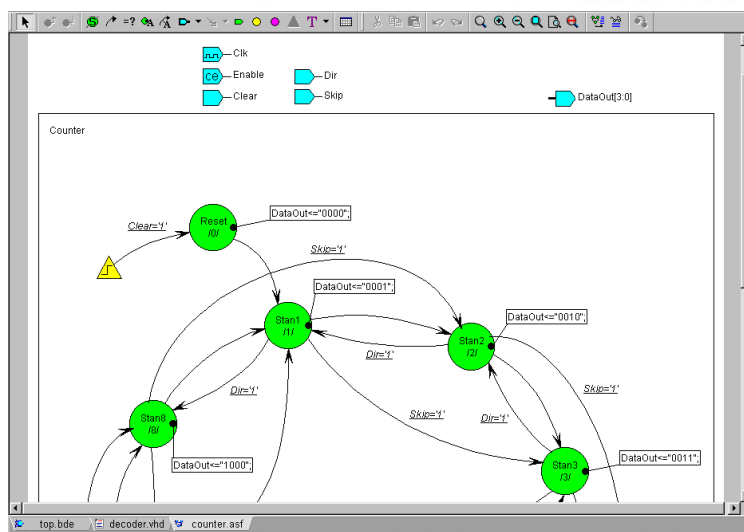
```



ActiveHDL – BDEditor



ActiveHDL – FSM editor





ActiveHDL – Design Browser

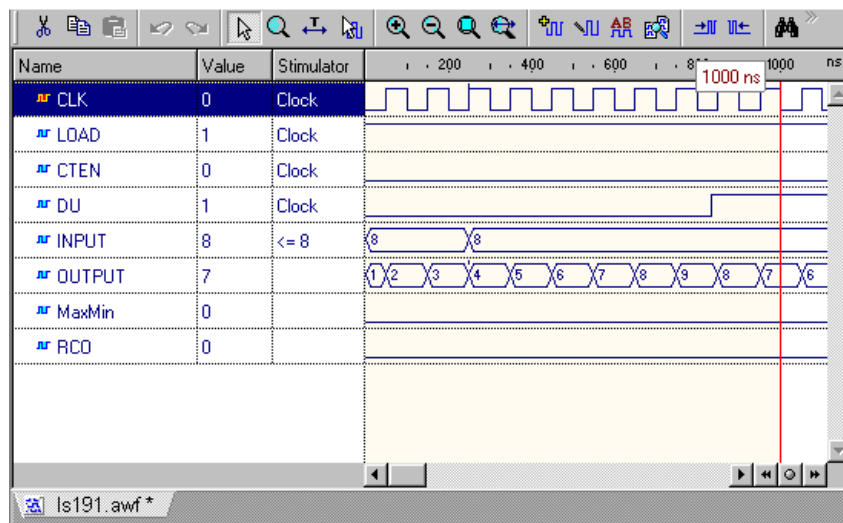
The left screenshot shows the Design Browser for the project 'freq_top (freq_top)'. It displays a tree structure under 'Unsorted' containing files like 'hex2led.vhd', 'CONTROL.asf', 'cnt_4b.vhd', 'freq_top.bde', 'freq_top.vhd', and 'TestBench'. Below this is the 'freq_meter library' containing 'CNT_48 (CNT_48)', 'CNT_BCD', and 'freq_top (freq_top)'. Further down are 'freq_meter_post_synthesis library' and 'freq_meter_timing library'. Annotations point to the 'freq_top (freq_top)' selection box and the 'Unsorted' folder.

The right screenshot shows the Design Browser for the component 'FQMETER (FQMETER)'. The hierarchy tree shows components: U25: CONTROL, U26: CNT, U27: CNT, UN1: ILD4V, UN2: ILD4V, U28: BCD2LED, U29: BCD2LED, and U21: AND2V. Below the tree is a table with columns: Name, Type, Value, Last Value, and Last Event Time. Annotations point to the 'FQMETER (FQMETER)' selection box, the hierarchy tree, a filter box, and the object list table.

Name	Type	Value	Last Value	Last Event Time
A	std_logic_v...	40	UU	100us
B	std_logic_v...	40	UU	100us
F10KHZ	std_logic	1	0	150us
FX	std_logic	1	0	155us
CTR	std_logic_v...	00	UU	Ops
GATE	std_logic	0	U	Ops
INGTE	std_logic	0	U	Ops



ActiveHDL – Waveform Viewer





ActiveHDL – List Viewer

simulation time simulation cycle signals/nets

toolbar Add Signals...

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

list1



ActiveHDL – Library Manager

Library Manager

File Edit Search View Design Simulation Tools Library Help

Library	Mode	Comment	Unit Name	Secondary Unit Name	Source Type	Target Language	Sy
ovi_uni3000	B	Xilinx Verilog library	E xor5	xor5_v	Source Code	VHDL	\
ovi_uni5200	B	Xilinx Verilog library	D xor6	xor6	Block diagram	EDIF	\
ovi_unisim	B	Xilinx Verilog library	D xor7	xor7	Block diagram	EDIF	\
ovi_xilinxco...	B	Xilinx Verilog library for Co	D xor8	xor8	Block diagram	EDIF	\
synopsys	B	Library containing packa	D xor9	xor9	Block diagram	EDIF	\
simprim_1_4	B	Post P&R timing simulatio	P global	global	Source Code	VHDL	\
simprim	B	Post P&R timing simulatio	P vcomponents		Source Code	VHDL	\
std	B	Standard VHDL library	P vpkg	vpkg	Source Code	VHDL	\
simprim_edif	B	Post P&R timing simulatio					
vi	B	Standard Verilog library					
synplify	B	Primitive library for post sy					
unisim	B	Functional components li					
xc4000e	B	Xilinx schematic library					
xabelsim	B	Functional components li					
xilinxcorelib	B	Functional simulation libr					

Package Contents

XDR9	XNDR6	X74_352	X74_174
XDR8	X74_L85	X74_298	X74_168
XDR7	X74_521	X74_283	X74_165S
XDR6	X74_518	X74_280	X74_164
XNDR9	X74_42	X74_273	X74_163
XNDR8	X74_390	X74_195	X74_162
XNDR7	X74_377	X74_194	X74_161

Ready

To be continued...

