Electronic Design Automation (EDA)

Electronic Design Automation (EDA): (Short Definition)
The use of software to automate electronic (digital and analog) design.

Electronic Design Automation (EDA) (Longer Definition)

Electronic design in which

the design is entered using *design capture tools*

or using a text editor and a *hardware description language*

possibly consisting of "parts" from a vendor's library

the functionality of the design is verified by simulation

the correctness, testability, and compliance of a design is checked by software

and the design is converted to a manufactureable form using *synthesis tools*.

Hardware Description Languages

01-2

Hardware Description Language:
A language used for describing the structure of hardware or how the hardware should behave.

Some Hardware Description Languages

A Hardware Programming Language (AHPL)

APL-like syntax. (APL was an early language for representing math.)

Developed at the University of Arizona.

Used in Hill and Peterson's *Digital Systems* textbook.

Not used in industry or in EE 3755 Fall 2001.

01-3

Verilog

Widely used in industry.

C-like syntax. (Many elements of the language are different from C.)

Developed by Gateway Design Automation in 1984, later bought by Cadence.

Became IEEE standard 1364 in 1995.

A newer version, Verilog 2000, should be approved by 2002.

VHDL (VHSIC Hardware Description Language)

Widely used in industry.

Ada-like syntax. (Ada is a DoD-developed language for large embedded systems.)

Developed as part of U.S. Department of Defense (DoD) VHSIC program in 1983

Became IEEE standard 1076 in 1987.

Standard updated in 1993.

01-4

Language Popularity

Both Verilog and VHDL are widely used in industry.

VHDL is more refined.

Verilog has steeper learning curve than VHDL (can be learned more quickly than).

Design Flow

01-5

Design Flow:
The steps used to produce a design.

Simple Design Flow

Step 1: Design Capture

Step 2: Behavioral Verification

Step 3: Synthesis and Timing Verification

01-6

Simple Flow Step 1: Design Capture

Using the back of an envelope or some other suitable medium ...
... develop a rough draft of the design.

Using a text editor ...
... write a *Verilog description* of the design.

Using a text editor ...
... write a *Verilog description* of a *testbench* used to test the design.

The testbench generates inputs for the design and verifies the design's outputs.

01-6

01-7

Simple Flow Step 2: Behavioral Verification

Using a *simulator* and *waveform viewer* ...
... check if design passes testbench tests ...
... and if not, debug.

Waveform Viewer:
Sort of a virtual logic analyzer, used to view signals in any part of design.

Simulator output includes messages generated by behavioral code ...
... including "pass" or "fail" message produced by testbench.

Using text editor ...
... fix bugs, and tune performance.

01-7

01-8

Simple Flow Step 3: Synthesis and Timing Verification

Using synthesis programs . . .
. . . generate *design database*.

Design database has information needed to fabricate the chip . . .
. . . and to perform simulations with accurate timing.

Re-simulate, and verify that timing is acceptable . . .
. . . if timing is not acceptable edit the Verilog structural description and repeat steps above.

Using the Internet, E-mail design database and credit card number to fab.

After a few weeks, get parts back in mail.

01-9

Topics Covered in This Course

Coding in Verilog.

Using simulation, waveform viewers and similar tools.

How Verilog descriptions synthesize.

Topics **Not** Covered in This Course

Details of using synthesis programs.

Writing testbenches.

These are valuable topics, but there's no time.

The curious might want to visit `http://www.ece.lsu.edu/v/` (ECE Verilog Course).

01-10

Software Used in This Course

Workstation Labs: Mentor Graphics

Programs available for simulation (Modelsim) and synthesis (Leonardo).

Full versions of programs used in industry.

01-10

Synthesis Design Target

01-11

Design Target:
The type of device to be manufactured or programmed.

Synthesis programs generate output for a particular design target.

Design Targets

Programmable Logic Array (PLA):
Chip that can be programmed (once) to implement a logic function.

Usually programmed at the factory.

PLAs might be used in prototypes or when only a few parts are needed.

Application-Specific Integrated Circuit (ASIC):
A fully custom chip.

Usually the fastest design target, can have the most components.

01-12

Gate Array:

A chip full of gates manufactured in two steps.

First generic layers containing gates are fabricated . . .
. . . but gates are not connected to each other.

Later, metal layers connecting gates are added.

Designer using gate arrays specifies only metal layer.

Since gates fabricated in advance time is saved.

Field-Programmable Gate Array (FPGA):

A chip full of logic whose connection and function can be programmed and later re-programmed.

Not as fast as the other design targets.