HDLs in

VLSI and Embedded System design

By

Vasu

Senior Application Engineer

VLSI & EH group

Unistring Tech Solutions Pvt. Ltd, #D 10, 5th Floor, Eureka complex,
Beside Image hospitals, Ameerpet, Hyderabad

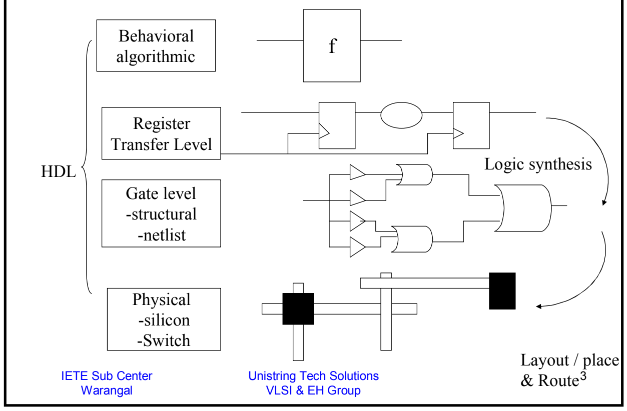
String Technologies, #309, Vederi complex,
opp Dilshuk Nagar bus Depot, Dilsukh nagar, Hyderabad,
ph: 040 – 24151900, 09440318188

String Technologies, Lashkar bazar, Hanamkonda, Warangal,

Introduction

How will you design a digital circuit/system ?

Digital design -Levels of Abstraction



Digital design -Levels of Abstraction (continued ..)

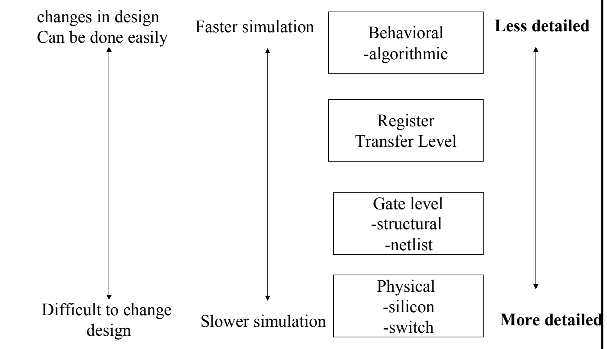
Behavioral level - describing the behavior or operation of the circuit using a hardware description language.

register-transfer level - concerned about how the data is transferred between the various registers and combination units to realize the circuit

Gate level - working with logic gates to build the circuit

Physical level - dealing with discrete transistors and connecting them together to form the circuit

Abstraction Level Trade-offs


changes in design Can be done easily

Faster simulation

Behavioral -algorithmic

Less detailed

Register Transfer Level

Gate level -structural -netlist

Physical -silicon -switch

Difficult to change design

Slower simulation

More detailed

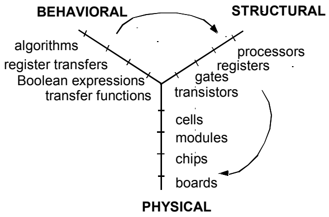
Hardware Description Languages (HDLs)

- Basic idea is a language to describe hardware
- Useful in describing complex digital systems
- Not software programming languages (C, Java, assembly, etc.)
- Examples
  - VHDL
  - Verilog
  - AHDL
  - System Verilog

Hardware Description Languages (HDLs) (conined ….)

- Initial purpose was to allow abstract design and simulation
  - Design could be verified then implemented in hardware
- Now Synthesis tools allow direct implementation from HDL code.
  - Large improvement in designer productivity
  - Combined with modern Field Programmable Gate Array chip technology, HDLs can implement large complex circuits (100000s of gates) in a very short time and without any considerable NRE costs.

The Role of Hardware Description Languages



- Design is structured around a hierarchy of representations
- **HDLs can describe distinct aspects of a design at multiple levels of abstraction**

Think Digital – while using HDL

- When designing a digital system in any HDL, it is important to remember the relation between code constructs and actual hardware
- Always consider the underlying hardware!
  - ask yourself: what does the VHDL code I'm writing actually represent?

HDL Advantages Over Schematic Entry

- Can also describe behavior of circuit
- Shifts focus to specifying functionality
- Produce correct designs in less time
- Produce larger and more complex systems per unit time
- Synthesis tools automate details of connecting gates and devices

Hardware Description Languages

- Verilog
  - Gateway Design Automation (1983; proprietary)
  - Acquired by Cadence 1989
  - IEEE standard in 1995
  - Similar to C
- VHDL
  - Origins in DoD VHSIC program (1980s)
  - IEEE standard in 1987
  - Similar to ADA

VHDL

- ➢ VHDL is a Hardware Description Language
- ➢ VHDL stands for VHSIC Hardware Description Language.

  (VHSIC stands for Very High Speed Integrated Circuits)
- ➢ Initiated by by United States DOD in the 1980s
- ➢ The first version was VHDL 87 ; Upgraded to VHDL 93
- ➢ Standardized by the (IEEE) Institute of Electrical and Electronic Engineers -- IEEE 1076
- ➢ An additional standard IEEE1164, was later added to introduced multi valued logic system.
- ➢ VHDL is intended for circuit simulation and as well as circuit synthesis
- ➢ VHDL is a standard Technology/vendor independent language, and is therefore portable and reusable.

Verilog

- Originated at Automated Integrated Design Systems (renamed Gateway) in 1983. Acquired by Cadence in 1989.
- Invented as simulation language. Synthesis was an afterthought. Many of the basic techniques for synthesis were developed at Berkeley in the 80's and applied commercially in the 90's.
- Around the same time as the origin of Verilog, the US Department of Defense developed VHDL. Because it was in the public domain it began to grow in popularity.
- Afraid of losing market share, Cadence opened Verilog to the public in 1990.
- An IEEE working group was established in 1993, and ratified IEEE Standard 1394 in 1995.
- **Today Verilog is the language of choice of Silicon Valley companies, because of high-quality tool support and its similarity to C-language syntax.**

Hardware Description Languages vs. Programming Languages

- Program Structure
  - Instantiation of multiple components of the same type
  - Specify interconnections between modules via schematic
  - Hierarchy of modules
- Assignment
  - Continuous assignment (logic always computes)
  - Propagation delay (computation takes time)
  - Timing of signals is important (when does computation have its effect)
- Data structures
  - Size explicitly spelled out - no dynamic structures
  - No pointers
- Parallelism
  - Hardware is naturally parallel (must support multiple threads)
  - Assignments can occur in parallel (not just sequentially)

Hardware Description Languages vs. Programming Languages (continued ..)

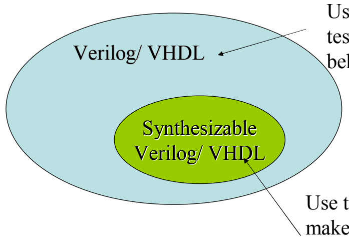
- HDL strengths
  - concurrency
  - bit manipulations
  - module representations
  - timing constraints
  - structural representation support
- Programming Language strengths
  - data structures
  - language support
- Mostly VHDL and Verilog today
  - most vendors support both languages

What does "synthesizable" and "non-synthesizable" mean?

- Simulation is a process in which the logic and functional verification of algorithm/design will be verified, by subjecting it to appropriate input test stimulus.
- In synthesis the VHDL/Verilog code is translated into low level netlist, to program a device or make a chip.
- Synthesizable means that a given design can be compiled into hardware
  - FPGA or CPLD(reprogrammable logic)
  - ASIC
- A non-synthesizable design can be simulated in software and is useful for
  - working out functionality
  - testing out concepts
  - test benche

Synthesizable Subset

- Verilog and VHDL began life as *simulation* and *modeling* tools
- Hardware synthesis developed during the 1990's
- Need to use a subset of Verilog and specific coding styles to allow synthesis tool to *infer* correct (and realizable) hardware



Use this to write testbenches for behavioral simulation

Use this to make hardware in FPGA

Key Advantages of HDL-Based Design Methodology

- Operate at higher level of abstraction
- Can debug earlier (behavioral simulator)
- Parameterized design, easy to make wholesale modifications to a design (e.g., bus width)

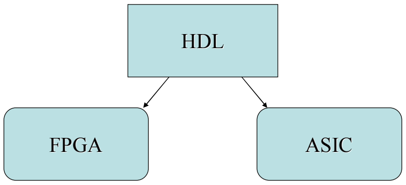
Key Advantages of HDL-Based Design Methodology

- Can quickly specify desired behavior
  - Example: Up-counter with reset

```
if (reset == 1)
      count <= 0;
else
      count <= count + 1;
```

Key Advantages of HDL-Based Design Methodology

- Can easily target multiple devices (eases product migration)



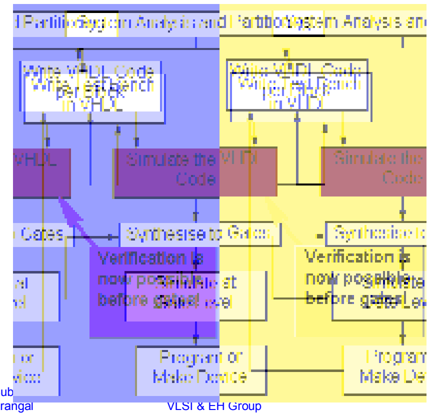
Key Advantages of HDL-Based Design Methodology

- HDL is more universal than schematic tools
- Promotes *design reuse*
- Promotes integration of third party designs, or *IP* (*intellectual property*)

What HDL is ***NOT***

- HDL is **not** a programming language (HDL is a *description* language)
- HDL is **not** highly abstract, e.g., implement the DSP algorithm $y(n) = 0.75y(n-1) + 0.3x(n)$
  (HDL is at the RTL level (register transfer))

Design Flow using VHDL



Motivation for Higher abstraction Level of coding (compared to VHDL)

- Higher abstraction level HDLs are crucial for SOC design and testing.
- Simplify hardware/software partitioning by describing both with a single, C-based language.
- Enhanced simulation and debugging performance.
- Many standards (e.g. the GSM and MPEG-standards) are published as C programs
  - ☞ Standards have to be translated if special hardware description languages have to be used
- The functionality of many systems is provided by a mix of hardware and software components
  - ☞ Simulations require an interface between hardware and software simulators unless the same language is used for the description of hardware and software.

Handel C

- Developed by Oxford Hardware Compilation Group
- Based on the language Occam
- Goal: Enable someone not familiar with H/W to do H/W design
- Currently, Celoxica provides a development suite called DK4 that uses Handel C.

What does Handel add to ANSI-C?

- Parallelism
- Timing
- Interfaces
- Clocks
- RAM/ROM
- Shared expression
- Communications
- Handel-C libraries
- Floating Point component library
- Bit manipulation
- Macro functions for hardware block reuse

What doesn't Handel C support

- Recursion
- Side effects
- Standard libraries
- Malloc()
- Standard floating point
- Pointers

- Aside from these, everything else in C is supported by Handel C

System C

- Offered by the Open SystemC Initiative (OSCI)
- Implemented via C++ library, runs on any C++ compiler.
- Includes constructs for testing/simulation as well as synthesis.

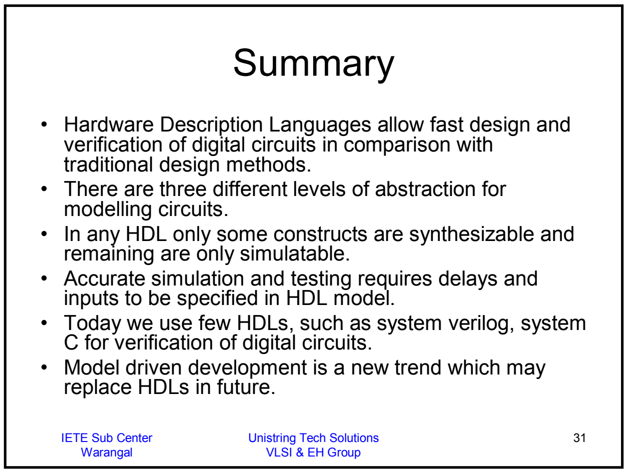
Synthesizable Subset

- Only a subset of the SystemC library is suitable for synthesis of a circuit.
- This subset closely matches VHDL or another underlying HDL.
- In reality, most tools actually convert the SystemC to HDL for synthesis.

Model driven development – A new Era

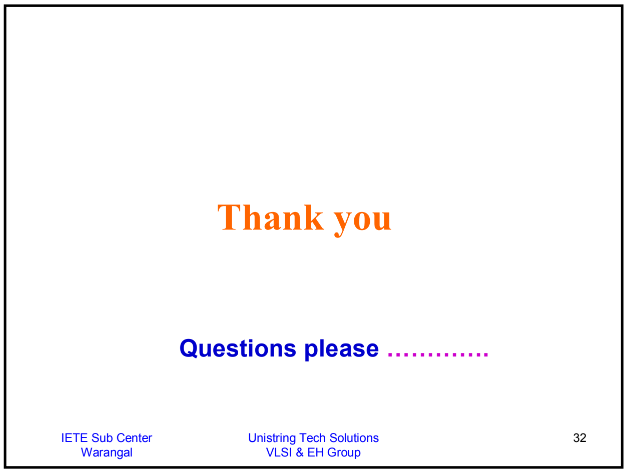
Xilinx System Generator.

Matlab simulink based design flow.



Summary

- Hardware Description Languages allow fast design and verification of digital circuits in comparison with traditional design methods.
- There are three different levels of abstraction for modelling circuits.
- In any HDL only some constructs are synthesizable and remaining are only simulatable.
- Accurate simulation and testing requires delays and inputs to be specified in HDL model.
- Today we use few HDLs, such as system verilog, system C for verification of digital circuits.
- Model driven development is a new trend which may replace HDLs in future.



Thank you

Questions please …………