

EECS 355
ASIC & FPGA Design

Seda Öğrenci Memik
seda@ece.northwestern.edu

<http://www.ece.northwestern.edu/~seda/eecs355.html>

EECS 355
ASIC & FPGA Design 1

Course Objectives

- Mastering VHDL
- Design flows for different technologies
 - Application-Specific Integrated Circuits
 - Field Programmable Gate Arrays
- Learning about computer-aided design tools for FPGAs and ASICs
- Designing complete digital systems

EECS 355
ASIC & FPGA Design 2

Course Outcomes

- Students should be able to
 - Given specifications: describe a digital system in VHDL
 - Through simulation verify that it operates correctly as well as to the satisfaction of performance criteria
 - Debug VHDL code
 - Execute major design tasks in commercial CAD tools, know how to supply proper inputs, set parameters correctly, and interpret the outputs
 - Mentor Graphics (ModelSim), Synopsys (Design Compiler), Synplify (Synplify), etc.
 - Describe main design steps in FPGA and ASIC Design Flows and differentiate the needs and relevance of them for each context
 - When presented with a design scenario and associated economics, choose the right design style, flow, technology

EECS 355
ASIC & FPGA Design 3

The VLSI Design Problem

- Realization of a specification
- Optimization of
 - Area
 - Speed
 - Power dissipation
 - Design time
 - Testability
 - Reliability

EECS 355
ASIC & FPGA Design 4

Increasing Device and Context Complexity

- Exponential increase in device complexity
 - Increasing with Moore's law (or faster)!
- More complex system contexts
 - System contexts in which devices are deployed are increasing in complexity
- Require exponential increases in design productivity

We have exponentially more transistors!

Complexity ↑

EECS 355
ASIC & FPGA Design 5
[©Keutzer]

Deep Submicron Effects

- Smaller geometries are causing a wide variety of effects that we have largely ignored in the past:
 - Cross-coupled capacitances
 - Signal integrity
 - Resistance
 - Inductance

Design of each transistor is getting more difficult!

DSM Effects ↓

EECS 355
ASIC & FPGA Design 6
[©Keutzer]

Heterogeneity on Chip

- Greater diversity of on-chip elements
 - Processors
 - Reconfigurable fabric
 - Software
 - Memory
 - Analog

Heterogeneity →

More transistors doing different things!

7
[©Keutzer]

Stronger Market Pressures

- Decreasing design window
- Less tolerance for lengthy design revisions

← Time-to-market

Exponentially more complex, greater design risk, greater variety, and a smaller design window!

EECS 355 ASIC & FPGA Design [©Keutzer]

A Quadruple-Whammy

9
[©Keutzer]

Evolution of the EDA Industry

EECS 355 ASIC & FPGA Design 10

IC Design Steps

EECS 355 ASIC & FPGA Design 11 Figs. [©Sherwani]

IC Design Steps Cntd.

EECS 355 ASIC & FPGA Design Figs. [©Sherwani]

Optimization: Levels of Abstraction

- Algorithmic/Architectural
 - Encoding data, computation scheduling, balancing delays of components, etc.
- Gate-level
 - Reduce fan-out, capacitance
 - Gate duplication, buffer insertion
- Layout / Physical-Design
 - Move cells/gates around to shorten wires on critical paths
 - Abut rows to share power / ground lines

EECS 355
ASIC & FPGA Design

13

How to deal with the complexity?

- Hierarchy: structure of a design at different levels of description
- Abstraction: hiding the lower level details

EECS 355
ASIC & FPGA Design

14

Y-CHART

EECS 355
ASIC & FPGA Design

15

Top-down structural decomposition / bottom-up layout reconstruction

EECS 355
ASIC & FPGA Design

16

Design methods

- Full custom
 - Maximal freedom
 - High performance blocks
 - Slow design cycle

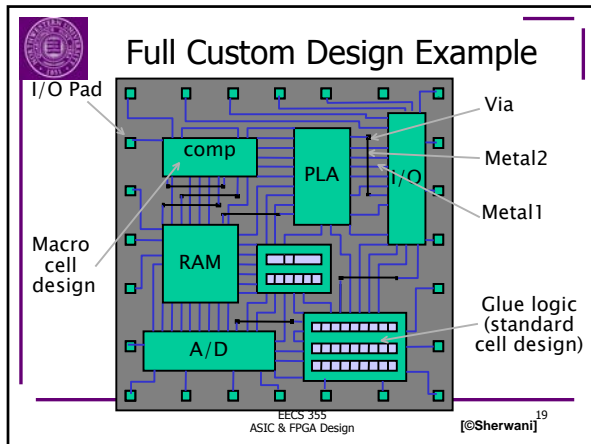
EECS 355
ASIC & FPGA Design

17

Full Custom Design

EECS 355
ASIC & FPGA Design

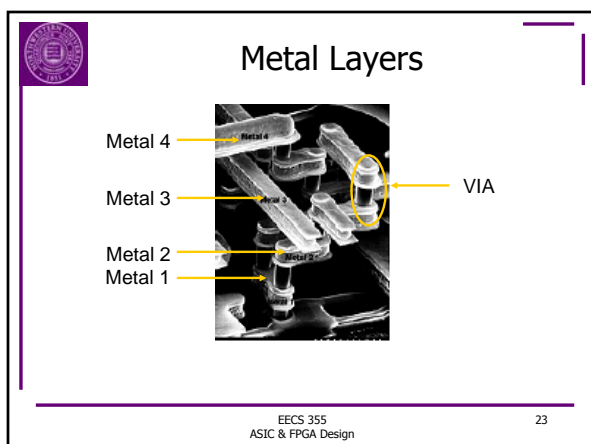
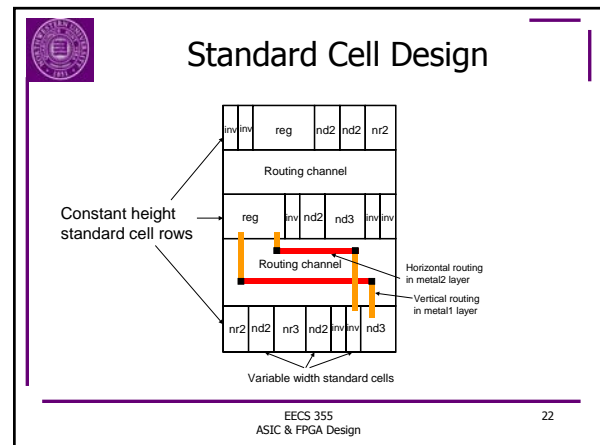
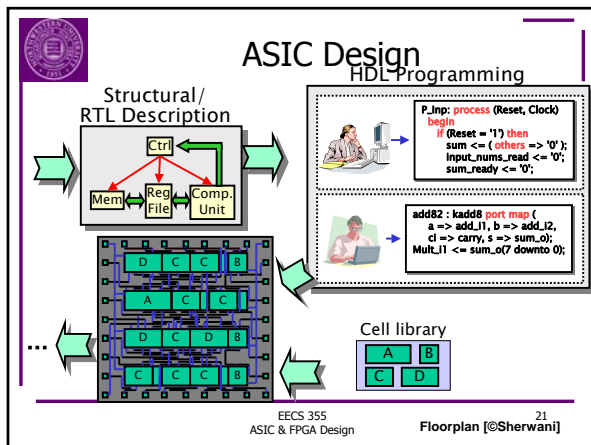
18



Design methods

- Semi-custom (a.k.a. standard cell)
 - gate arrays (mask vs field programmable)
 - standard cells
 - parametrizable modules (adder, multiplier, memories)


EECS 355 ASIC & FPGA Design 20



Programmable Technologies

- PLA
- PAL
- CPLD
- FPGA
- Structured ASIC (semi-programmable)


EECS 355 ASIC & FPGA Design 24



CAD Tools for Automation

- **Design Capture and Simulation**
 - **Algorithmic and system design:** C, DFG (Data Flow Graph), HDL (Hardware Description Language) / Simulator
 - **Structural and logic design:** Schematic editor / Simulator
 - **Transistor level design:** Schematic editor / Simulator
 - **Layout design:** Layout editor

EECS 355
ASIC & FPGA Design 25




CAD Tools for Automation

SYNTHESIS
The process of creating a structural description of a design from a behavioral description

- **High level:** synthesis from algorithmic/behavioral level to structural logic level
 - Data flow graph to gate netlist
- **Logic:** synthesis from logic level to gate level
 - Boolean equations, truth table to gate netlist


EECS 355
ASIC & FPGA Design 26



HARDWARE DESCRIPTION LANGUAGES

- HDL is used to describe the hardware for the purpose of modeling, simulation, testing, design, and documentation.
 - **Modeling:** behavior, flow of data, structure
 - **Simulation:** verification and test
 - **Design:** synthesis


EECS 355
ASIC & FPGA Design 27



Purpose of VHDL

- **Problem**
 - Need a method to quickly design, implement, test, and document increasingly complex digital systems
 - Schematics and Boolean equations inadequate for million-gate IC


EECS 355
ASIC & FPGA Design 28



Purpose of VHDL

- **Solution**
 - A hardware description language (HDL) to express the design
 - Associated computer-aided design (CAD) or electronic design automation (EDA) tools for **capture, simulation, and synthesis**
 - Programmable logic devices for rapid implementation of hardware
 - Custom VLSI application specific integrated circuit (ASIC) devices for low-cost mass production

EECS 355
ASIC & FPGA Design 29



History of VHDL

- Two widely-used HDLs today
 - VHDL
 - Verilog HDL (from Cadence, now IEEE standard)
- VHDL - Very High Speed Integrated Circuit (VHSIC) Hardware Description Language

EECS 355
ASIC & FPGA Design 30

History of VHDL

- VHDL history
 - Created by US Department of Defense to document designs for portability
 - IEEE standard 1076 (VHDL) in 1987
 - Revised IEEE standard 1076 (VHDL) in 1993
 - IEEE standard 1164 (object types standard) in 1993
 - IEEE standard 1076.3 (synthesis standard) in 1996

EECS 355
ASIC & FPGA Design 31

VHDL: Why?

- Reasons to use VHDL
 - Power and flexibility
 - Device-independent design
 - Portability among tools and devices
 - Device and tool benchmarking capability
 - VLSI ASIC migration
 - Quick time-to-market and low cost (with programmable logic)

EECS 355
ASIC & FPGA Design 32

VHDL

- Problems with VHDL
 - Loss of control with gate-level implementation
 - Potentially inefficient logic implementations via synthesis (engineer-dependent)
 - Variations in synthesis quality among tools (always improving, but still a problem)

EECS 355
ASIC & FPGA Design 33

Design Flow in VHDL

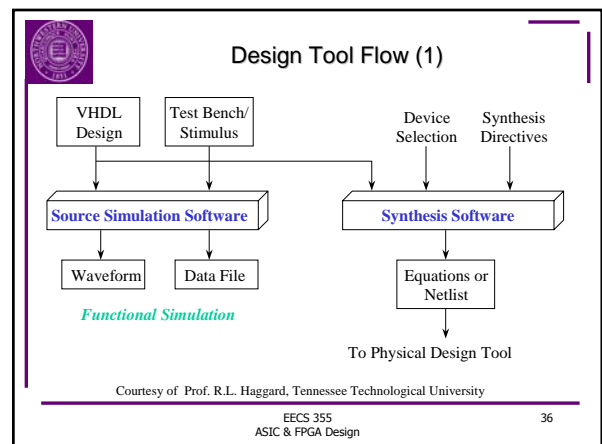
- Define the design requirements
- Describe the design in VHDL
 - Top-down, hierarchical design approach
 - Code optimized for synthesis or simulation
- Simulate the VHDL source code
 - Early problem detection before synthesis

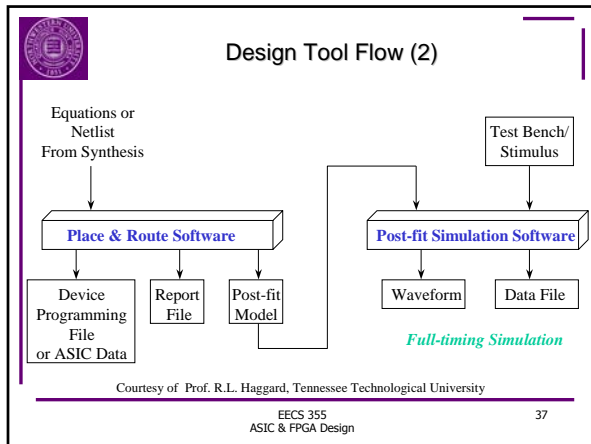
EECS 355
ASIC & FPGA Design 34

Design Flow in VHDL

- Synthesize, optimize, and fit (place and route) the design for a device
 - Synthesize to equations and/or netlist
 - Optimize equations and logic blocks subject to constraints
 - Fit into the components blocks of a given device
- Simulate the post-layout design model
 - Check final functionality and worst-case timing
- Program the device (if PLD) or send data to ASIC vendor

EECS 355
ASIC & FPGA Design 35





EXAMPLES

- Example VHDL code for 4-bit 2-to-1 multiplexer:

```

LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY mymux1 IS
  PORT (a,b: IN bit_vector(3 DOWNTO 0)); --inputs
        sel: IN bit;
        y:  OUT bit_vector(3 DOWNTO 0)); --output
END mymux1;

ARCHITECTURE behavior OF mymux1 IS
BEGIN
  mux: PROCESS (a,b,sel)
  BEGIN
    IF sel = '0' THEN y <= a;
    ELSE y <= b;
    END IF;
  END PROCESS mux;
END behavior;
    
```

EECS 355
ASIC & FPGA Design 38

STYLES in VHDL

- Behavioral**
 - High level, algorithmic
 - Sequential execution
 - Hard to synthesize well
 - depends largely on the "talent" of the synthesis tool
 - Easy to write and understand (like high-level language code)
 - Describes the function and timing of system

EECS 355
ASIC & FPGA Design 39

STYLES in VHDL

- Dataflow**
 - Medium level, register-to-register transfers
 - Concurrent execution
 - Easy to synthesize well
 - Harder to write and understand (like assembly code)
 - Describes how data flows through the system
 - Hides details of the underlying combinational circuit

EECS 355
ASIC & FPGA Design 40

STYLES in VHDL

- Structural**
 - Low level, netlist, component instantiations and wiring
 - Components themselves may be described in behavioral or dataflow model with proper interfaces
 - Trivial to synthesize
 - Hardest to write and understand (very detailed and low level)

EECS 355
ASIC & FPGA Design 41

SUMMARY

- The VLSI digital design problem is described
- Y-Chart and the different levels of abstraction are explained
- VLSI design automation and CAD tools are mentioned
- Purpose and background of VHDL are explained
- VHDL and programmable logic are the best current solution for rapid design, implementation, testing, and documenting of complex digital systems
- A standard 6-step design synthesis process is used with VHDL
- The general flow of information through standard VHDL synthesis CAD tools described

EECS 355
ASIC & FPGA Design 42