Lecture 1

Tinoosh Mohsenin
Today

- Administrative items
- Syllabus and course overview
- Digital systems and optimization overview
Course Communication

- Email
  - Urgent announcements

- Web page
  - http://www.csee.umbc.edu/~tinoosh/cmpe650/

- Office hours
  - By appointment
This course focuses on

- Advanced topics for a complete digital system design
- Advanced topics in logic design
  - Fixedpoint arithmetic
  - Pipelining
  - Memory system design
  - Timing Analysis
  - Low power design
- FPGA implementation and its features
- Evaluation of the system on FPGA
Course Description

- Computer Aided Design of large/complex digital system
  - Verilog
  - Xilinx ISE flow
    - Simulation (isim)
    - Synthesis and place & route
  - FPGA verification
    - Artix-7 FPGA

- Prerequisite
  - CMPE 415
  - CMPE 310
Course Description

- Lectures
- Handouts
- Homework/ projects
  - Three/four HWs
- Midterm Exam
  - End of March (or early April)
- Final Project and Presentation (or Final exam)
  - A simple communication system design and optimization.
- Active participation (5% of your grade)
Lectures

- Ask questions at any time
- Participate in the class (%5 of your grade)
- Silence phones
- Hold conversations outside of class
Advanced FPGA Design

- **FPGA**: Field Programmable Gate Arrays
- **Advanced**: Basic knowledge of FPGA and verilog coding
- **Design**: Meeting functional requirements while satisfying performance, delay, power and cost budgets
Trends in Cellphone Chip Integration

- Chip integration is increasing every generation
- Cell phone size is decreasing
- Users want more features every generation
- Power budget is very limited

Y. Neuvo, ISSCC 2004
Cellphone Architecture Example

- Cellphone chips have multiple processing cores and support multiple applications and features
  - Ex: Integrated Transceiver: WiFi (802.11a/b/g), Bluetooth, FM
Digital Systems

- Electronic circuits that use discrete representations of information
  - Discrete time and values
Digital Processing vs Analog Processing

- Digital arithmetic is completely stable over process, temperature, and voltage variations
  - Ex: \(2.0000 + 3.0000 = 5.0000\) will always be true as long as the circuit is functioning correctly

- Digital design energy-efficiencies are rapidly increasing

- Once a digital processor has been designed in a portable format (gate netlist, HDL, software), very little effort is required to “port” (re-target) the design to a different processing technology. Analog circuits typically require a nearly-complete re-design.

- Digital circuit capabilities are rapidly increasing

- Analog A/D speed x resolution product doubles every 5 years

- Digital processing performance doubles every 18-24 Months (6x to 10x every 5 years)
Common DSP Applications

- Early applications
  - Instrumentation
  - Radar
  - Imaging

- Current applications
  - Audio, video
  - Networking
  - Telecommunications
  - Biomedical application
Common Trends

- Analog based → Digital based
  - Music: records, tapes → CDs
  - Video: VHS, 8mm → DVD, Blu-ray
  - Telephony, cell phones: analog (1G) → digital (2G, 3G, 4G, …)
  - Television: NTSC → digital (DVB, ATSC, ISDB, …)
  - Many new things use digital data and “speak” digital: computers, networks, digital appliances
Basic Digital Circuit Components

- Primitive components for logic design

- AND gate
- OR gate
- Inverter
- Multiplexer
Sequential Circuits

- Circuit whose output values depend on current and previous input values
  - Include some form of storage of values

- Nearly all digital systems are sequential
  - Mixture of gates and storage components
  - Combinational parts transform inputs and stored values
Flipflops and Clocks

- **Edge-triggered D-flipflop**
  - stores one bit of information at a time

- **Timing diagram**
  - Graph of signal values versus time

![Diagram of edge-triggered D-flipflop and timing diagram]
Hierarchical Design

Design

Functional Verification

OK?

Y

N

Architecture Design

Unit Design

Unit Verification

OK?

Y

N

Integration Verification

OK?

Y

N
What we learn by the end of semester

- Processor building blocks
  - Binary number representations
  - Types of Adders
  - Multipliers
  - Complex arithmetic hardware
  - Memories

- Communication algorithms and systems

- Design optimization targeted for FPGA
  - Verilog synthesis to a gate netlist
  - Delay estimation and reduction
  - Area estimation and reduction
  - Power estimation and reduction
A Simple Design Methodology

1. Requirements and Constraints
2. Design
   - Functional Verification
     - OK?
       - Y: Post-synthesis Verification
       - N: Return to Design
     - N: Return to Design
3. Synthesize
   - Post-synthesis Verification
     - OK?
       - Y: Physical Implementation
       - N: Return to Synthesize
     - N: Return to Synthesize
4. Physical Implementation
   - Physical Verification
     - OK?
       - Y: Manufacture
       - N: Return to Physical Implementation
     - N: Return to Physical Implementation
5. Manufacture
6. Test

Digital Design — Chapter 1 — Introduction and Methodology
Hierarchical Design

- Circuits are too complex for us to design all the detail at once
- Design subsystems for simple functions
- Compose subsystems to form the system
  - Treating subcircuits as “black box” components
  - Verify independently, then verify the composition
- Top-down/bottom-up design
Synthesis

- We usually design using register-transfer-level (RTL) Verilog
  - Higher level of abstraction than gates
- Synthesis tool translates to a circuit of gates that performs the same function
- Specify to the tool
  - the target implementation fabric
  - constraints on timing, area, etc.
- Post-synthesis verification
  - synthesized circuit meets constraints
Physical Implementation

- Implementation fabrics
  - Application-specific ICs (ASICs)
  - Field-programmable gate arrays (FPGAs)
- Floor-planning: arranging the subsystems
- Placement: arranging the gates within subsystems
- Routing: joining the gates with wires
- Physical verification
  - physical circuit still meets constraints
  - use better estimates of delays
Codesign Methodology

Requirements and Constraints

Partitioning

Software Requirements and Constraints

Software Design and Verification

Hardware Requirements and Constraints

Hardware Design and Verification

OK?

N

OK?

Manufacture and Test

N

Digital Design — Chapter 1 — Introduction and Methodology
Summary

- Digital systems use discrete (binary) representations of information
- Basic components: gates and flipflops
- Combinational and sequential circuits
- Real-world constraints
  - logic levels, loads, timing, area, etc
- Verilog models: structural, behavioral
- Design methodology
Integrated Circuits (ICs)

- Circuits formed on surface of silicon wafer
  - Minimum feature size reduced in each technology generation
  - Currently 90nm, 65nm
  - Moore’s Law: increasing transistor count
  - CMOS: complementary MOSFET circuits
Logic Levels

- Actual voltages for “low” and “high”
  - Example: 1.4V threshold for inputs
Logic Levels

- TTL logic levels with noise margins

\[ V_{OL} : \text{output low voltage} \]
\[ V_{OH} : \text{output high voltage} \]
\[ V_{IL} : \text{input low voltage} \]
\[ V_{IH} : \text{input high voltage} \]
Static Load and Fanout

- Current flowing into or out of an output
  - High: SW1 closed, SW0 open
    - Voltage drop across R1
    - Too much current: $V_O < V_{OH}$
  - Low: SW0 closed, SW1 open
    - Voltage drop across R0
    - Too much current: $V_O > V_{OL}$
  - Fanout: number of inputs connected to an output
    - determines static load
Capacitive Load and Prop Delay

- Inputs and wires act as capacitors

- \( t_r \): rise time
- \( t_f \): fall time
- \( t_{pd} \): propagation delay
  - delay from input transition to output transition
Other Constraints

- **Wire delay**: delay for transition to traverse interconnecting wire

- **Flipflop timing**
  - delay from clk edge to Q output
  - D stable before and after clk edge

- **Power**
  - current through resistance => heat
  - must be dissipated, or circuit cooks!
Area and Packaging

- Circuits implemented on silicon chips
  - Larger circuit area => greater cost
- Chips in packages with connecting wires
  - More wires => greater cost
  - Package dissipates heat
- Packages interconnected on a printed circuit board (PCB)
  - Size, shape, cooling, etc, constrained by final product
Models

- Abstract representations of aspects of a system being designed
  - Allow us to analyze the system before building it

- Example: Ohm’s Law
  - $V = I \times R$
  - Represents electrical aspects of a resistor
  - Expressed as a mathematical equation
  - Ignores thermal, mechanical, materials aspects
Verilog

- Hardware Description Language
  - A computer language for modeling behavior and structure of digital systems

- Electronic Design Automation (EDA) using Verilog
  - Design entry: alternative to schematics
  - Verification: simulation, proof of properties
  - Synthesis: automatic generation of circuits
Module Ports

- Describe input and outputs of a circuit

```
>30°C
above_30_0

>25°C
above_25_0

low level
low_level_0

>30°C
above_30_1

>25°C
above_25_1

low level
low_level_1

temp_bad_0

or_0a

or_0b

below_25_0

inv_0

wake_up_0

select_mux

buzzer

or_1a

or_1b

temp_bad_1

wake_up_1

select_vat_1

buzzer

+V

low_level_1

above_30_1

inv_1

below_25_1
```
module vat_buzzer_struct
  (  output buzzer,
      input above_25_0, above_30_0, low_level_0,
      input above_25_1, above_30_1, low_level_1,
      input select_vat_1 );

wire below_25_0, temp_bad_0, wake_up_0;
wire below_25_1, temp_bad_1, wake_up_1;

// components for vat 0
  not inv_0 (below_25_0, above_25_0);
  or  or_0a (temp_bad_0, above_30_0, below_25_0);
  or  or_0b (wake_up_0, temp_bad_0, low_level_0);

// components for vat 1
  not inv_1 (below_25_1, above_25_1);
  or  or_1a (temp_bad_1, above_30_1, below_25_1);
  or  or_1b (wake_up_1, temp_bad_1, low_level_1);

mux2 select_mux (buzzer, select_vat_1, wake_up_0, wake_up_1);
endmodule
module vat_buzzer_struct
  ( output buzzer,
    input above_25_0, above_30_0, low_level_0,
    input above_25_1, above_30_1, low_level_1,
    input select_vat_1 );

assign buzzer =
  select_vat_1 ? low_level_1 | (above_30_1 | ~above_25_1)
  : low_level_0 | (above_30_0 | ~above_25_0);
endmodule
Design Methodology

- Simple systems can be designed by one person using *ad hoc* methods.
- Real-world systems are designed by teams, which require a systematic design methodology.

**Specifies**
- Tasks to be undertaken
- Information needed and produced
- Relationships between tasks
  - Dependencies, sequences
- EDA tools used
Design using Abstraction

- Circuits contain millions of transistors
  - How can we manage this complexity?
- Abstraction
  - Focus on relevant aspects, ignoring other aspects
  - Don’t break assumptions that allow aspect to be ignored!
- Examples:
  - Transistors are on or off
  - Voltages are low or high
Embedded Systems

- Most real-world digital systems include embedded computers
  - Processor cores, memory, I/O
- Different functional requirements can be implemented
  - by the embedded software
  - by special-purpose attached circuits
- Trade-off among cost, performance, power, etc.