Introduction to Digital System Design

Outline
1. Why Digital?
2. Device Technologies
3. System Representation
4. Abstraction
5. Development Tasks
6. Development Flow

Advantages
- Advantage of digital devices
  - Reproducibility of information
  - Flexibility and functionality: easier to store, transmit and manipulate information
  - Economy: cheaper device and easier to design
- Moore’s law
  - Transistor geometry
  - Chips double its density (number of transistor) in every 18 months
  - Devices become smaller, faster and cheaper
  - Now a chip consists of hundreds of million gates
  - And we can have a “wireless-PDA-MP3-camera-GPS-cell-phone” gadget very soon

Applications of digital systems
- “Digitization” has spread to a wide range of applications, including information (computers), telecommunications, control systems etc.
- Digital circuitry replaces many analog systems:
  - Audio recording: from tape to music CD to MP3 (MPEG Layer 3) player
  - Image processing: from silver-halide film to digital camera
  - Telephone switching networks
  - Control of mechanical system: e.g., “flight-by-wire”
How to implement a digital system

• No two applications are identical and every one needs certain amount of customization
  • Basic methods for customization
    – “General-purpose hardware” with custom software
      • General purpose processor: e.g., performance-oriented processor (e.g., Pentium), cost-oriented processor (e.g., PIC micro-controller)
    – Special purpose processor: with architecture to perform a specific set of functions: e.g., DSP processor (to do multiplication-addition), network processor (to do buffering and routing), “graphic engine” (to do 3D rendering)

2. Device Technologies

Fabrication of an IC

• Transistors and connection are made from many layers (typical 10 to 15 in CMOS) built on top of one another
• Each layer has a special pattern defined by a mask
• One important aspect of an IC is the length of a smallest transistor that can be fabricated
  – It is measured in micron (μm, 10^-6 meter)
  – E.g., we may say an IC is built with 0.50 μm process
  – The process continues to improve, as witnessed by Moore’s law
  – The state-of-art process approaches less than a fraction of 0.1 μm (known as deep sub-micron)

Classification of device technologies

• Where customization is done:
  – In a fab (fabrication facility): ASIC (Application Specific IC)
  – In the “field”: non-ASIC
• Classification:
  – Full-custom ASIC
  – Standard cell ASIC
  – Gate array ASIC
  – Complex field programmable logic device
  – Simple field programmable logic device
  – Off-the-shelf SSI (Small Scaled IC)/MSI (Medium Scaled IC) components
**Full-custom ASIC**

- All aspects (e.g., size of a transistor) of a circuit are tailored for a particular application.
- Circuit fully optimized
- Design extremely complex and involved
- Only feasible for small components
- Masks needed for all layers

**Standard-Cell ASIC**

- Circuit made of a set of pre-defined logic, known as standard cells
- E.g., basic logic gates, 1-bit adder, D FF etc
- Layout of a cell is pre-determined, but layout of the complete circuit is customized
- Masks needed for all layers

**Gate array ASIC**

- Circuit is built from an array of a single type of cell (known as base cell)
- Base cells are pre-arranged and placed in fixed positions, aligned as one- or two-dimensional array
- More sophisticated components (macro cells) can be constructed from base cells
- Masks needed only for metal layers (connection wires)

**Complex Field Programmable Device**

- Device consists of an array of generic logic cells and general interconnect structure
- Logic cells and interconnect can be "programmed" by utilizing "semiconductor fuses or "switches"
- Customization is done "in the filed"
- Two categories:
  - CPLD (Complex Programmable Logic Device)
  - FPGA (Field Programmable Gate Array)
- No custom mask needed

**Simple Field Programmable Device**

- Programmable device with simple internal structure
- E.g.,
  - PROM (Programmable Read Only Memory)
  - PAL (Programmable Array Logic)
- No custom mask needed
- Replaced by CPLD/FPGA

**SSI/MSI components**

- Small parts with fixed, limited functionality
- E.g., 7400 TTL series (more than 100 parts)
- Resource (e.g., power, board area, manufacturing cost etc.) is consumed by "package" but not "silicon"
- No longer a viable option
Three viable technologies

- Standard Cell ASIC
- Gate Array ASIC
- FPGA/CPLD

Comparison of technology

- Area (Size): silicon “real-estate”
  - Standard cell is the smallest since the cells and interconnect are customized
  - FPGA is the largest
    - Overhead for “programmability”
    - Capacity cannot be completely utilized
- Speed (Performance)
  - Time required to perform a task
- Power
- Cost

Cost

- Types of cost:
  - NRE (Non-Recurent Engineering) cost: one-time, per-design cost
  - Part cost: per-unit cost
  - Time-to-market “cost” loss of revenue
- Standard cell: high NRE, small part cost and large lead time
- FPGA: low NRE, large part cost and small lead time

Summary of technology

<table>
<thead>
<tr>
<th></th>
<th>FPGA</th>
<th>Gate array</th>
<th>Standard cell</th>
</tr>
</thead>
<tbody>
<tr>
<td>tailored</td>
<td>0</td>
<td>3 to 5</td>
<td>15 or more</td>
</tr>
<tr>
<td>area</td>
<td></td>
<td></td>
<td>best (smallest)</td>
</tr>
<tr>
<td>speed</td>
<td></td>
<td></td>
<td>best (fastest)</td>
</tr>
<tr>
<td>power</td>
<td></td>
<td></td>
<td>best (minimal)</td>
</tr>
<tr>
<td>NRE cost</td>
<td>best (smallest)</td>
<td></td>
<td>best (smallest)</td>
</tr>
<tr>
<td>part cost</td>
<td>best (smallest)</td>
<td></td>
<td>best (smallest)</td>
</tr>
<tr>
<td>design cost</td>
<td>best (smallest)</td>
<td></td>
<td>best (smallest)</td>
</tr>
<tr>
<td>time to market</td>
<td>best (closest)</td>
<td></td>
<td>best (closest)</td>
</tr>
<tr>
<td>per unit cost</td>
<td>depend on volume</td>
<td></td>
<td>depend on volume</td>
</tr>
</tbody>
</table>

- Trade-off between optimal use of hardware resource and design effort/cost
- No single best technology

3. System Representation (View)
• View: different perspectives of a system
  • Behavioral view:
    – Describe functionalities and i/o behavior
    – Treat the system as a black box
  • Structural view:
    – Describe the internal implementation
      (components and interconnections)
    – Essentially block diagram
  • Physical view:
    – Add more info to structural view: component
      size, component locations, routing wires
    – E.g., layout of a print circuit board

4. Abstraction

• How to manage complexity for a chip with 10 million transistors?
• Abstraction: simplified model of a system
  – show the selected features
  – Ignore associated detail
• E.g., timing of an inverter

Summary
RT level

- RT (Register Transfer) is a misleading term
- Should use "module-level"
- Two meanings:
  - Loosely: represent the module level
  - Formally: a design methodology in which the system operation is described by how the data is manipulated and moved among registers

View and abstraction are two independent aspects.
Combined in a Y-chart

5. Development Tasks

- Developing a digital system is a refining and validating process
- Main tasks:
  - Synthesis
  - Physical design
  - Verification
  - Testing

Synthesis

- A refinement process that realizes a description with components from the lower abstraction level.
- The resulting description is a structural view in the lower abstraction level
- Type of synthesis:
  - High-level synthesis
  - RT level synthesis
  - Gate level synthesis
  - Technology mapping

Physical Design

- Placement and routing
  - Refining from structural view to physical view
  - Derive lay out of a netlist
- Circuit extraction:
  - Determine the wire resistance of capacitance
- Others
  - Derivation of power grid and clock distribution network, assurance of signal integrity etc.
Verification

- Check whether a design meets the specification and performance goals.
- Concern the correctness of the initial design and the refinement processes.
- Two aspects
  - Functionality
  - Performance (timing)

Method of Verification

- Simulation
  - spot check: cannot verify the absence of errors
  - Can be computation intensive
- Timing analysis
  - Just check delay
- Formal verification
  - apply formal math techniques determine its property
  - E.g. equivalence checking
- Hardware emulation

Testing

- Testing is the process of detecting physical defects of a die or a package occurred at the time of manufacturing.
- Testing and verification are different tasks.
- Difficult for large circuit
  - Need to add auxiliary testing circuit in design
  - E.g., built-in self test (BIST), scan chain etc.

Limitation of EDA software

- EDA (Electronic Design Automation)
- EDA software can automate some tasks
- Can software replace human hardware designer? (e.g., C-program to chip)
- Synthesis software
  - should be treated as a tool to perform transformation and local optimization
  - cannot alter the original architecture or convert a poor design into a good one

Development Flow

- Medium design targeting FPGA
- Circuit up to 50,000 gates
Addtional tasks

- Large design targeting FPGA
  - Design partition
  - More verification
- Large design targeting ASIC
  - Thorough verification
  - Testing
  - Physical design

Goal of this course

- Goal:
  - Systematically develop efficient, portable RT level designs that can be easily integrated into a larger system
- Design for efficiency
- Design for "large"
  - Large module, large system, overall development process
- Design for portability
  - Device independent, software dependent, design reuse