Chapter 1
Digital System Design

Dr. Curt Nelson
Engr433 – Digital Design

Outline

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

• Advantages of digital devices
  – Reproducibility of information;
  – Flexibility and functionality - easier to store, transmit and manipulate information;
  – Economy - cheaper devices and easier to design.

• Moore’s law
  – Transistor geometry;
  – Chips double in density (number of transistor) every 18 months;
  – Devices become smaller, faster, and cheaper;
  – Now, a chip consists of a few billion gates.
Applications of Digital Systems

- Digital circuitry replaces many analog systems
  - Audio recording
    - From Tape → Music → CD → MP3 → Air Pods → ???;
  - Image processing - from silver-halide film to digital cameras;
  - Telephone switching networks;
  - Replacement of mechanical systems e.g., “fly-by-wire”

Digital Circuits in Communication Systems
Device Technologies - Fabrication of an IC

- Transistors and connections are made from many layers (typically 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
  - Measured in microns ($10^{-6}$ meter);
  - We may say an IC is built with 15 nm process;
  - The process continues to improve, as witnessed by Moore’s law;
  - The state-of-art processes are now 9 nm and shrinking, still using photolithography.

Classification of Device Technologies

- Where customization is done
  - In a fabrication facility - ASIC (Application Specific IC);
  - In the “field” - non-ASIC, like an FPGA.
- Classification
  - Full-custom ASIC;
  - Standard cell ASIC;
  - Gate array ASIC;
  - Field programmable logic device;
  - Off-the-shelf parts (Small or Medium Scale Integration).
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;
- 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 like 1-bit adder, D FF, NAND gates, 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 arrays;
• More sophisticated components (macro cells) can be constructed from base cells;
• Masks needed only for metal layers (connection wires).

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 field;
• No custom masks needed.
SSI/MSI Components

- Small parts with fixed, limited functionality;
- e.g. 74HC00 CMOS series (more than 100 parts);
- Resource (e.g., power, board area, manufacturing cost etc.) is consumed by package but not silicon;
- Seldom a viable option.

Three Viable Technologies

- Standard Cell ASIC;
- Gate Array ASIC;
- FPGA.
Comparison of Technologies

• Area or 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
• Power
• Cost
• Best 2 out of 3?

Cost

• Types of cost
  – NRE (Non-Recurrent Engineering) cost - one-time, per-design 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 Technologies

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

- Trade-off between optimal use of hardware resources and design effort/cost;
- No single best technology – depends on the application.

View - Different System Representations

- 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 a block diagram.
- Physical view
  - Add more info to structural view - component size, component locations, routing wires;
  - Layout of a printed circuit board, for instance.
Structural and Physical Views

• Level of abstractions
  – Transistor level;
  – Gate level;
  – Register transfer (RT) level;
  – Processor level.

• Characteristics of each level
  – Basic building blocks;
  – Signal representation;
  – Timing representation;
  – Behavioral representation;
  – Physical representation.
Summary

<table>
<thead>
<tr>
<th></th>
<th>Typical Blocks</th>
<th>Signal Representation</th>
<th>Time Representation</th>
<th>Behavioral Description</th>
<th>Physical Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>transistor</td>
<td>transistor, resistor</td>
<td>voltage</td>
<td>continuous function</td>
<td>differential equation</td>
<td>transistor layout</td>
</tr>
<tr>
<td>gate</td>
<td>and, or, xor, flip-flop</td>
<td>logic 0 or 1</td>
<td>propagation delay</td>
<td>Boolean equation</td>
<td>cell layout</td>
</tr>
<tr>
<td>RT</td>
<td>adder, mux, register</td>
<td>integer, system state</td>
<td>clock tick</td>
<td>extended FSM</td>
<td>RT level floor plan</td>
</tr>
<tr>
<td>processor</td>
<td>processor, memory</td>
<td>abstract data type</td>
<td>event sequence</td>
<td>algorithm in C</td>
<td>IP level floor plan</td>
</tr>
</tbody>
</table>

View vs. Abstraction

- View and abstraction are two independent aspects;
- Combined in a Y-chart.
Main 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 layout of a netlist.
• Circuit extraction
  – Determine the wire resistance/capacitance/inductance.
• Other
  – Derivation of power grid and clock distribution network, assurance of signal integrity etc.

Verification

• Check whether a design meets the specification and performance goals;
• Concerns the correctness of the initial design and the refinement processes;
• Two aspects
  – Functionality;
  – Performance (timing).
Methods of Verification

• Simulation
  – Spot check: cannot verify the absence of errors;
  – Can be computationally intensive.
• Timing analysis
  – Just check delay.
• Formal verification
  – Apply formal math techniques to determine properties;
  – E.g., equivalence checking.
• Hardware emulation

Testing

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

- Electronic Design Automation (EDA) software can automate some tasks;
- Can software replace the 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.
Additional Tasks

- Large design targeting FPGA
  - Design partitioning;
  - More verification.
- Large design targeting ASIC
  - Thorough verification;
  - Testing;
  - Physical design.