Discussion Section 1

Sean Huang
January 22, 2021
About Me

• 4th-year graduate student
  – Advised by Prof. Bora Nikolic
• Working in the Berkeley Wireless Research Center (BWRC)
• Researching a framework for designing and generating PLLs for all purposes
About This Discussion Section

- Quick review of relevant points from each week
- Some example problems about those topics from lecture and homework
About This Discussion Section

• Quick review of relevant points from each week
• Some example problems about those topics from lecture and homework

– We’ll be doing these together!
Last-Minute Logistics

• Email me at sehuang@berkeley.edu
• Office Hours Fridays 2PM–3PM (14:00–15:00), Zoom link to be posted on the website
  – If this time doesn’t work, let me know through email and we can try to set up another time
Process and Frequency Scaling

- Transistors getting smaller, faster, cheaper
Process and Frequency Scaling

IBM 1401 (1959)
Process and Frequency Scaling
Process and Frequency Scaling
Process and Frequency Scaling

8 transistors per card!

Berkeley
UNIVERSITY OF CALIFORNIA
Process and Frequency Scaling

Each card ≈ a 7400 chip

8 transistors per card!
Process and Frequency Scaling

Each card ≈ a 7400 chip

8 transistors per card!
Process and Frequency Scaling

• Dennard Scaling
  – Reducing dimensions and supply voltage yields performance improvements
    • Decreases gate capacitance (increased speed, lower power)
    • Constant power density (more devices, less power per device)
Process and Frequency Scaling

• **Dennard Scaling**
  – Reducing dimensions and supply voltage yields performance improvements
    • Decreases gate capacitance (increased speed, lower power)
    • Constant power density (more devices, less power per device)

• **Frequency Scaling**
  – Keep supply voltage constant, decrease size
    • Speed increases quadratically! 😊
    • Power increases cubically! 😞

• “Power wall” at 3–4GHz
  – Same old tradeoff not worth it anymore
Process and Frequency Scaling

• Moore’s Law
  – Still trying to cram more transistors on the same package (chiplets)
  – More transistors = more functionality?
Process and Frequency Scaling

• Moore’s Law
  – Still trying to cram more transistors on the same package (chiplets)
  – More transistors = more functionality?

• Frequency Scaling
  – Actually dead
  – Worked up to the power wall, no clear way to get past this
  – Most modern processor clock speeds pretty much constant across generations
The Pareto Optimal Frontier

• Tradeoffs
  – Power vs. Performance
  – Cost vs. (Pretty much everything)
  – Time-to-market vs. Performance
• Every engineering project is balancing compromises
  – How much should we sacrifice to meet specifications?
• Pareto optimal frontier
  – Best possible outcome of tradeoff space
Digital Logic Design
Combinational Logic

- Output only depends on current input
  - “Memoryless”

- Truth Table
  - List all possible inputs
  - Define output behavior for each

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
### Proof By Truth Table

#### Truth Table 1

| C | B | A | A|B | (A|B)&C | Out  |
|---|---|---|---|---|--------|------|
| 0 | 0 | 0 | 0 | 1 | 0      | 1    |
| 0 | 0 | 1 | 1 | 0 | 1      | 0    |
| 0 | 1 | 0 | 1 | 0 | 1      | 0    |
| 0 | 1 | 1 | 1 | 0 | 1      | 0    |
| 1 | 0 | 0 | 0 | 1 | 0      | 1    |
| 1 | 0 | 1 | 1 | 1 | 0      | 0    |
| 1 | 1 | 0 | 1 | 1 | 0      | 0    |
| 1 | 1 | 1 | 1 | 1 | 0      | 0    |

#### Truth Table 2

| C | B | A | A|B | !(A|B) | !C | ![A|B] | !(A|B) & !C | Out  |
|---|---|---|---|---|-------|---|-------|------------|------|
| 0 | 0 | 0 | 0 | 1 | 1     | 1 | 0     | 0          | 1    |
| 0 | 0 | 1 | 1 | 0 | 1     | 1 | 0     | 0          | 1    |
| 0 | 1 | 0 | 1 | 0 | 1     | 1 | 0     | 0          | 1    |
| 0 | 1 | 1 | 1 | 0 | 1     | 1 | 0     | 0          | 1    |
| 1 | 0 | 0 | 0 | 1 | 0     | 0 | 0     | 0          | 0    |
| 1 | 0 | 1 | 1 | 0 | 0     | 0 | 0     | 0          | 0    |
| 1 | 1 | 0 | 1 | 0 | 0     | 0 | 0     | 0          | 0    |
| 1 | 1 | 1 | 1 | 0 | 0     | 0 | 0     | 0          | 0    |
Sequential Logic

- What if we need memory?
- Registers (flip-flops) store information
  - Output updates to equal input depending on load signal
    - This is not instant
  - Output is held steady until next load edge
- Clock (clk) signal is typically used for synchronizing registers
Register Transfer Level (RTL)

- All digital designs can be abstracted as RTL
  - Mix of combinational and sequential logic blocks
- There are limitations for synchronous designs
  - No combinational loops
- HDL Languages (Verilog, VHDL)
  - Use this abstraction to describe systems
  - Follow similar division of designs
    - Combinational section
    - Sequential section

Not to be confused with Resistor-Transistor Logic
ASIC vs. FPGA
ASIC vs. FPGA

Application-Specific Integrated Circuit (ASIC)
- Customizable layout
- Control over entire chip layout (at design)
- Optimized for few applications

Field-Programmable Gate Array (FPGA)
- Array of pre-placed general logic blocks
  - Look-up tables, Registers, Multiplexer, Memory, DSP accelerators, Combinational Logic, Interconnect
- Can be reprogrammed on the fly to implement hardware even after manufacture
# ASIC vs. FPGA

<table>
<thead>
<tr>
<th>ASIC</th>
<th>FPGA</th>
</tr>
</thead>
</table>
| Area-efficient  
  - Only place what you need | Generality  
  - Some logic blocks won’t be used for every application |
| Inflexibility  
  - Can only configure as far as designed to  
  - Usually only a few applications | Flexibility  
  - Can change hardware connections on the fly  
  - Many potential applications |
| Design turnaround time in months/years | Design turnaround time in minutes/hours |
| High Fixed Cost (NRE)  
  - Design and Verification  
    - Only have 1 shot to get the chip right  
    - Many hours of engineering work to create each iteration | Low Fixed Cost (NRE)  
  - Just need to program some existing FPGAs  
  - Low risk as redesign is quick |
| Low Manufacturing Cost  
  - Once design verified, can mass produce ASICs at very low cost  
    - Area optimized for application, so higher yield | Moderate Manufacturing Cost  
  - Need to source FPGAs from a third party  
    - Need more area for generality, more cost per die  
    - Less tolerance for manufacturing defects because of larger die |
| Need to sell a lot of chips to offset NRE cost! | Good for low volume production or need for flexible implementation |
Simulation

- **RTL Simulation**
  - Quick verification of logical function
    - Used more in early stages of design to verify idea works
  - Ideal behavioral models

- **Gate-level Simulation**
  - More “real” effects taken into account (e.g. timing)
    - Used in later design stages to verify actual chip still works
  - Models each gate with library of parameters

- **Circuit Level Simulation (SPICE, Spectre)**
  - Lowest level simulation, taking everything possible into account
Simulation

Simulation is your friend!
(But also your worst enemy)

- SPICE is a tool, not an oracle
- Sometimes (rarely) the simulation is wrong!
  - Incorrect setup
  - Bad model
- Have an idea for what to expect
  - Do hand calcs before simulating
  - Know what the system **should** do
LTSpice Installation and Tutorial