Digital System Design

Analog
- time varying signals that can take on any value across a continuous range of voltage, current or other metric

Digital
- signals are modeled with two states, 0 or 1
- underneath it is all analog
Digital logic levels

TTL logic levels with noise margins

Sending system:
VOL: output low voltage
VOH: output high voltage

Receiving system:
VIL: input low voltage
VIH: input high voltage
IC Basics
Comparison of Switching Standards

5-V TTL
Standard TTL: ABT, AHCT, HCT, ACT, Bipolar HC, AHC, AC, LV-A

5-V CMOS
Rail-to-Rail 5 V

3.3-V LVTTL
LVT, LVC, ALVC, LV-A, ALVT

2.5-V CMOS
AUC, AVC, ALVC, LVC, ALVT

1.8-V CMOS
AUC, AVC, ALVC, LVC

Is $V_{OH}$ higher than $V_{IH}$?
Is $V_{OL}$ less than $V_{IL}$?

D | R | 5TTL | 5CMOS | 3LVTTL | 2.5CMOS | 1.8CMOS
--- | --- | --- | --- | --- | --- | ---
5TTL | Yes | No | Yes* | Yes* | Yes*
5 CMOS | Yes | Yes | Yes* | Yes* | Yes*
3 LVTTL | Yes | No | Yes | Yes* | Yes*
2.5 CMOS | Yes | No | Yes | Yes | Yes*
1.8 CMOS | No | No | No | No | Yes

* Requires $V_{IH}$ Tolerance
Why digital?

- Reproducibility of results
  - Outputs of analog circuits vary with temperature, power supply voltage, component aging
- Ease of design
  - Logic vs calculus, e.g. digital vs analog computer
- Flexibility and functionality
- Programmability – tools
- Speed
- Economy
- ... and the technology keeps improving!
Digital System Design

System
- orderly interconnection of parts into a meaningful whole
- interactions and interconnections define the system

What are the system components?
... depends entirely on your point of view
Closed Digital System

Components/Interfaces

Board

Programmable Logic devices

Integrated circuits

Gates

Transistors

IC Manufacturing process

Device physics

Digital System Design

Simple systems can be designed by one person using *ad hoc* methods.

Real-world systems are designed by teams:
- require a systematic modular design methodology
  - decompose system to define components to be designed
  - define information needed and produced
  - define relationships between components
    - dependencies, sequences
- EDA tools used
A Simple Design Methodology

- Requirements and Constraints
- Design
  - Functional Verification
    - OK? (Y: Synthesize, N: Return to Design)
    - OK? (Y: Post-synthesis Verification, N: Return to Design)
    - OK? (Y: Physical Implementation, N: Return to Design)
  - OK? (Y: Synthesize, N: Return to Design)
- Physical Implementation
  - Physical Verification
    - OK? (Y: Manufacture, N: Return to Design)
- Manufacture
- Test
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
Hierarchical Design

Design

Functional Verification

OK? Y N

Architecture Design

Unit Design

Unit Verification

OK? N Y

Integration Verification

OK? N Y
Digital System Design

Design process ...

- Start with a word problem
- Functional specification
  - Define inputs, outputs, control lines
  - Describe circuit function; “what” not “how”
- Block diagram
  - Pictorial
  - Major functional modules and interconnections
The Design Process ... specifications

<table>
<thead>
<tr>
<th>word problem</th>
<th>Design a circuit to select one of two inputs depending on the value of a control variable.</th>
</tr>
</thead>
<tbody>
<tr>
<td>functional specification</td>
<td>Design a system where the output will be if $=$ $;$ if $=$ $$.</td>
</tr>
<tr>
<td>block diagram</td>
<td></td>
</tr>
</tbody>
</table>
Functional vs Operational block diagrams
Formal Specifications  ... the “how”

Abstractions ... vs Representations ...
Truth tables  
Boolean equations  
Timing diagrams  

Structured logic device descriptions  
- State tables  
- State diagrams  
- HDL code  

Gate level circuit diagram  
MSI level circuit diagram  
Transistor level circuit diagram  
Structured logic device descriptions  
- State tables  
- State diagrams  
- HDL code
The Design Process ... formal specifications

Truth table

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

the output will be if $A = 0$; if $A = 1$.

Boolean equations
The Design Process...

Gate level circuit diagram

MSI level circuit diagram

Transistor level circuit diagram

Copyright © 2000 by Prentice Hall, Inc.
Digital Design Principles and Practices, 3/e
The Design Process ... HDLs

Abel program

module mux2
  title 'two input multiplexer'
  MUX2 device 'P16V8'
  A, B, S  pin 1, 2, 3;
  Z  pin 13 istype 'com';
  equations
  WHEN S == 0 THEN Z=A; ELSE Z=B;
end mux2
library IEEE;
use IEEE.std_logic_1164.all;

entity mux2 is
  port (A, B, S : in STD_LOGIC;
        Z : out STD_LOGIC);
end mux2;

architecture mux2_arch of mux2
begin
  Z <= A when S = '0' else B;
end mux2_arch;
The Design Process ... HDLs

Verilog program -- *behavioural*

```verilog
module mux2 ( input A, B, S, output Z );

Assign Z = S ? A : B ;

endmodule
```
The Design Process ... then what

- Simulate
- Test ... repeat above steps as necessary
- Build prototype
- Test ... repeat above steps as necessary
- Final build

... and most importantly
The Design Process

Documentation

• Functional specification
• Block diagram
• Timing diagram
• Structured logic device description
• Schematic diagram
• Circuit description *in words*
  • Explain how the circuit works
Alternate solutions

Performance
- Semiconductor material
- Number of gates, types
- Circuit structure, layout
- Size (physical)

Cost
- Minimize components (number)
- Minimize board space
- Use programmable logic devices
- Minimize manufacturing costs (large runs)
- Use commercially available components (small runs)
Terms

I make a distinction between:

**Analysis**
logic diagram $\rightarrow$ formal description
$\equiv$ *representation* $\rightarrow$ *abstraction*

**Synthesis**
formal description $\rightarrow$ logic diagram
$\equiv$ *abstraction* $\rightarrow$ *representation*

**Design**
informal description $\rightarrow$ representation