Digital logic

Lecture 1.
Introduction

Doru Tadinca

Department of Computers
Politehnica University of Timisoara
Outline

Principles and practice

Analog versus digital

Digital devices

Electronic aspects of digital devices
  Logic levels, invalid levels and noise margins

Integrated circuits

Digital-Design Levels
Administrative

- Instructor: Doru Todinca, room B622
- e-mail: doru.todinca@cs.upt.ro
- Labs are mandatory and will count 50% in the final mark
- The lecture is also mandatory
- Examination: written exam, counting 50% of the final grade
Outline

Principles and practice

Analog versus digital

Digital devices

Electronic aspects of digital devices
   Logic levels, invalid levels and noise margins

Integrated circuits

Digital-Design Levels
Textbook

- The fourth edition was published in 2006
- If not specified otherwise, my presentations are entirely based on John Wakerly’s book
- in the sense that figures, tables, definitions, examples, etc, from third edition are used for these presentations
- Handouts will be enough for your exam, but Wakerly’s book may be useful.
Principles and practice

- Most of the principles that you learn now will continue to be important in the future.
- Maybe some principles will be applied in ways that have not yet been discovered!
- Practice changes much faster, sometimes even before you start working in the field.
- For sure many practical things will change through your career.
- Wakerly: “Treat practice material as a way to reinforce principles.
- and as a way to learn design methods by examples”.
- These things are valid not only for Digital Logic, but for most things that you study in college!
Outline

Principles and practice

Analog versus digital

Digital devices

Electronic aspects of digital devices
  Logic levels, invalid levels and noise margins

Integrated circuits

Digital-Design Levels
Analog versus digital

- Analog signals (produced by analog devices) can take any value over a continuous range of values (of voltage, current, or other metric).
- We model a **digital signal** as taking at any time only two discrete values.
- We call these two values **0** and **1**, **LOW** and **HIGH**, **FALSE** and **TRUE**, negated or asserted, etc.
- In reality digital signals do take values over a continuous range of voltages, currents, etc, but we ignore their analog behaviour.
- **Digital abstraction**: we associate a range of analog values with a logic **0** value and another range of analog value with a logic **1**.
- The range of values associated to **0** logic and the range of values associated to **1** logic are separated by a range of invalid (undefined) values.
Analog versus digital: advantages of digital devices

- Reproducibility of results:
  - A properly designed digital device always obtains the same results (outputs) for the same set of inputs.
  - For an analog circuit, this is not always true, because its outputs can vary with temperature, power supply, aging, and other factors.

- Ease of design: Digital, or logic design is logic, no special math needed (e.g. calculus).

- Flexibility and functionality: once a problem is in digital form, we can follow a set of logical steps and solve it.

- Programmability:
  - Much of digital design is done using Hardware Description Languages (HDLs).
  - HDLs are used for modeling, simulation and synthesis.
  - The use of HDLs in digital design will increase even more in the future.

- Speed: digital circuits are very fast.
Advantages of digital devices

- Economy: the cost of digital circuits decreases, making mass production very effective
- Steadily advancing technology: when designing a digital system, we know that there will be a faster, cheaper, better technology in the future, and can anticipate it (e.g. by providing expansion sockets)
- Digital devices replaced analog devices in many domains, in the last decades:
  - still pictures (cameras)
  - video recordings: digital versatile discs (DVDs)
  - audio recordings: compact discs (CDs)
  - automobile carburetors
  - the telephone system
  - mobile phones
  - traffic lights
  - movie effects
  - and many more!
Outline

Principles and practice

Analog versus digital

Digital devices

Electronic aspects of digital devices
  Logic levels, invalid levels and noise margins

Integrated circuits

Digital-Design Levels
There are two types of digital devices: *combinational devices* and *sequential devices*

**Definition**

**Combinational devices**: their outputs depend only on the current *input combination* (i.e., the combination of their input values)

**Definition**

**Sequential devices**: their outputs depend on the current input combination *and* the *sequence* of past inputs.
Digital devices: sequential circuits

- Sequential devices have *states*, or *memory*, i.e., they store values.
- Usually the state of a sequential device can be changed only at certain time moments, determined by a “clock” input signal.
- The most basic sequential circuit is called *flip-flop*.
- The state of a flip-flop can be either 0 or 1.
- Or, we can say that a flip-flop stores either a 0 or a 1.
- Flip-flops are built from combinational circuits (from gates).
- In general, a sequential device consists of flip-flops and combinational devices.
- That’s why we will study first combinational circuits, then sequential circuits.
Combinational circuits: gates

- The simplest combinational circuits are called gates.
- This is because they control the flow of digital information: they allow or not to pass certain information from inputs to output.
- Gates have one single output and one or more inputs.
- Of course, inputs and output take analog values, but we interpret them digitally (0 or 1).
- There are three fundamental gates (see figure 1), from which any other gate can be obtained:
  1. AND gate
  2. OR gate
  3. NOT gate, or inverter
Combinational circuits: gates

Figure 1: Digital devices: (a) AND gate, (b) OR gate and (c) NOT gate, or inverter

Figure 1 shows the symbols of the three fundamental gates, and their behaviour: all input combinations and the resulting outputs.
Digital gates and truth tables

A gate’s behaviour can be expressed more compactly using the truth table (see figure 2)
The figure shows also the functions realized by the three gates: $X \text{ AND } Y$, $X \text{ OR } Y$, and $\text{ NOT } X$
Inverting gates

(a) X NAND Y

(b) X NOR Y

We can combine an AND gate and a NOT gate, obtaining a NAND gate

The circle on the gate symbol is called inversion bubble, and it means that the output of the gate is negated

Which means that, instead of function X AND Y, the gate implements the function NOT(X AND Y)

Similarly for NOR gate, the function is NOT(X OR Y)
Logic values, logic signals and gates. Definitions

- A logic value, 0 or 1, is called a *binary digit*, or *bit*.
- If more than two values are needed, then we can add more bits.
- With *n* bits we have $2^n$ different values.
- When we discuss electronic logic circuits we use LOW and HIGH for 0 and 1.
- LOW: a signal is in the range of algebraically lower values, which is interpreted as logic 0.
- HIGH: a signal is in the range of algebraically higher values, which is interpreted as logic 1.
- Association between 0 and LOW and 1 and HIGH is arbitrary, and is called *positive logic*.
- The opposite association, i.e., 0 to HIGH and 1 to LOW is called *negative logic*. Normally we use positive logic.
The operation of a combinational circuit is fully described by a truth table that lists all combinations of input values and the output value(s) produced by each input combination.

For a combinational circuit with $n$ inputs, the truth table has $2^n$ lines.

The behaviour of a sequential circuit can be described by a state table.

The state table specifies next state and the output as function of its inputs and current state.
Logic values, logic signals and gates. Definitions

- An **AND gate** produces a 1 output if and only if (iff) all its inputs are 1. Otherwise its output is 0.
- It means that, if at least one input is 0, the output of an AND gate is 0.
- The output function of an AND gate with inputs $X$ and $Y$ is denoted $X \text{ AND } Y$ or $X \cdot Y$.
- An **OR gate** produces a 1 output if and only if one or more inputs are 1.
- It means that an OR gate produces a 0 output iff all inputs are 0.
- The function of an OR gate with inputs $X$ and $Y$ is denoted $X \text{ OR } Y$ or $X + Y$. 
Logic values, logic signals and gates. Definitions

- A **NOT gate** (an inverter) produces an output value that is the opposite of the input value.
- It means, when the input is 0, the output is 1; when the input is 1, the output is 0.
- The function of the NOT gate with input X is $NOT \ X$, denoted also $\bar{X}$, or $X'$. We will prefer the notation $X'$.
- We can combine AND and NOT to obtain the **NAND gate**, with the function $NOT(X \ AND \ Y)$, or $\overline{X \cdot Y}$, or $(X \cdot Y)'$, or $X \ NAND \ Y$.
- The output of a NAND gate is the opposite of an AND gate: a 0 iff all inputs are 1.
- A **NOR gate** is obtained by combining an OR and an inverter.
- The function of a NOR is denoted $NOT \ (X+Y)$, or $\overline{X+Y}$, or $(X + Y)'$, or $X \ NOR \ Y$.
- The output of a NOR gate is the opposite of an OR gate: a 0 iff one or more inputs are 1.
Representations of a digital device

1. The “black-box” representation (fig 4):
   - Minimum amount of detail: only the number of inputs and outputs
   - It does not describe the functioning of the device (how it responds to input signals)

2. The truth table: for combinational circuits (fig 5)

3. Logic diagram (logic circuit) (fig 6): we will learn how to obtain the logic circuit from the truth table

4. Timing diagram (fig 7):
   - Contains the time dimension of the circuit’s behaviour
   - It shows how the circuit might respond to the time-varying input signals
   - It shows also that the logic signals do not change instantaneously from logic 0 to logic 1: the signals have a slope, visible on the oscilloscope, but not in simulation
   - Also, we can see that there is a lag between an input change and the corresponding output change (circuit’s delay): it can be visualized by simulation
Representations of a digital device

Figure 4: Black-box representation of a 3-input, 1-output logic circuit

Table 3-2
Truth table for a combinational logic circuit.

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 5: Truth table for a combinational logic circuit.
Representations of a digital device

Figure 6: Circuit diagram for the circuit described by the truth table from above

Figure 7: Timing diagram for a logic circuit
Outline

Principles and practice

Analog versus digital

Digital devices

Electronic aspects of digital devices
  Logic levels, invalid levels and noise margins

Integrated circuits

Digital-Design Levels
Figure 8: Logic levels for CMOS circuits

- Between 0.0 V and 1.5 V is the voltage interval representing logic 0 values.
- Between 3.5 V and 5.0 V is the voltage interval representing logic 1 values.
- Between the 0 logic values and 1 logic values is the interval for undefined (invalid) values: the interval 1.5 V to 3.5 V.
- The values are for the CMOS integrated circuits.
The noise margins

- The domain for logic 0 output values is smaller than the input interval for logic 0 values and included in it.
- The difference between them is the noise margin. Similar for logic 1 intervals.
- If a noise signal affects the output of a circuit, the output will be correctly recognized as a logic 0 (or 1) by the input of the next circuit if the noise signal is smaller than the noise margin.

Figure 9: Logic levels and noise margin
Logic families

- First electronically controlled logic circuits were based on relays (1930, Bell Labs)
- Eniac, first electronic digital computer was built with vacuum tubes (mid-1940s)
- Invention of semiconductor diode and bipolar junction transistor made computers smaller and faster (late 1950s)
- In 1960s: invention of integrated circuit (IC): multiple diodes, transistors and other components on a single chip
- **Definition:** “a *logic family* is a collection of different integrated circuit chips that have similar input, output, and internal circuit characteristics, but that perform different logic functions”
- Most successful bipolar logic family: transistor-transistor logic (TTL) 1960s
- Metal oxide semiconductor field-effect transistor (MOSFET, or MOS transistor): the base of CMOS family (mid 1980’s)
- CMOS technology: most used and easiest to understand!
Outline

Principles and practice

Analog versus digital

Digital devices

Electronic aspects of digital devices
  Logic levels, invalid levels and noise margins

Integrated circuits

Digital-Design Levels
Integrated circuits: the fabrication process

- “A collection of one or more gates fabricated on a single silicon chip is called an integrated circuit (IC).” [Wakerly]
- The fabrication process starts with a circular wafer that contains many replicas of the same IC (thousands replicas).
- The size of the wafer is “up to ten inches in diameter”
- Each piece (IC chip) is called a die
- Each die has pads - electrical contact points, much larger than other IC features, where the wires will be connected later
- After fabrication, the dice are tested in place on the wafer using very small probing pins to contact the pads
- Defective dice are marked
- Then, the wafer is sliced in order to produce the individual dice
- The marked dice are discarded
- Each “good” die is mounted in a package, the pads are connected to the package pins, resulting in an integrated circuit
- The packaged ICs are tested again before being sold
Integrated circuits: classification by size

1. Small-Scale Integration (SSI):
   - Contain the equivalent of 1 to 20 gates
   - Typically SSI ICs contain gates and flip-flops
   - They come in a 14-pin dual inline-pin (DIP) package (see figure 10, (a))

2. Medium-Scale Integration (MSI):
   - Contain the equivalent of about 20 to 200 gates
   - Typically contain functional building blocks: decoders, encoders, multiplexers, demultiplexer, registers, counters
   - The equivalent building blocks are used in larger ICs

3. Large-Scale Integration (LSI):
   - contain the equivalent of 200 to 1,000,000 gates or more
   - They include: small memories, microprocessors, programmable logic devices, and customized devices

4. Very Large-Scale Integration (VLSI):
   - Separation between LSI and VLSI is fuzzy and is based on transistor count
   - ICs with a few millions of transistors are VLSI
   - They include most nowadays microprocessors and memories, larger programmable logic devices and customized devices
Figure 10: Dual inline pin (DIP) packages: (a) 14-pin; (b) 20-pin; (c) 28-pin
Figure 11: Pin diagram for several SSI ICs from 7400-series
Programmable logic devices

- There are many types of ICs that can be “programmed” after manufacturing
- It means that their logic function are established after manufacturing
- Most such devices can be also re-programmed
- This is very good for the design process: e.g., if we find an error, we can change the logic functions of the IC
- First such devices were called *Programmable Logic Arrays (PLAs)*, then, after enhancements, *Programmable Array Logic (PAL) devices*
- They consist of a two-level structure of AND and OR gates with user-programmable connections
- Today, the generic name for such devices is *Programmable Logic Devices (PLDs)*
Programmable logic devices

- There are technological limitations in the capacity increase of PLDs.

- Two solutions have been developed:
  1. Complex PLD (CPLD): a number of PLDs and a programmable structure that connects them (fig 12 (a))
  2. Field Programmable Gate Arrays (FPGAs): a very large number of small individual logic blocks and a large, distributed interconnection structure (fig 12 (b))

- Both CPLDs and FPGAs are used for prototyping and they reduce the “time to market” of a product

- This is because of the use of HDLs (Hardware Description Languages) like VHDL or Verilog: a description of a device realized in a HDL can be synthesized and downloaded on a FPGA or CPLD chip in minutes.

- Synthesis is the process of transformation of a representation of a digital device to an equivalent lower level representation.
Programmable logic devices

Figure 12: Large PLDs: (a) CPLD; (b) FPGA
Application-Specific ICs

- The costs of a LSI chip entirely designed for a specific customer (from chip’s functions to the transistor level) are extremely high (more than 500,000 USD), being efficient only for mass production (e.g. microprocessors, buss-interface chips)

- In order to reduce the costs per chip, semicustom ICs, or application-specific ICs (ASICs) have been developed

- ASICs are “chips designed for a particular, limited product or application”

- IC manufacturers have developed libraries of standard cells, which usually contain MSI common functions like decoders, registers, counters, or LSI functions like memories

- The ASICs rely on standard cell design: the logic designer interconnects such functions using the libraries of standard cells
Printed-Circuit Boards

- ICs are mounted on printed-circuit boards (PCBs), called also printed-wired boards PWBs.
- A multilayer PCB “have copper wiring etched on multiple, thin layers of fiberglass that are laminated into a single board”
- Individual wire connections are called PCB traces.
- Most modern PCBs use surface-mount technology (SMT).
- The ICs are called SMDs (surface-mounted devices).
- They are mounted on PCB (usually by machine), with their leads bent, and held in place by a solder paste.
- Then the entire PCB is passed through an oven to melt the solder paste.
- Old DIP chips have long pins that poke through the board and are soldered on the underside.
Outline

Principles and practice

Analog versus digital

Digital devices

Electronic aspects of digital devices
  Logic levels, invalid levels and noise margins

Integrated circuits

Digital-Design Levels
Digital-Design Levels

- Digital design can be done at several levels of abstraction and representation
- Usually a designer works at a certain level of design
- It’s necessary for a designer to be able to move up or down one or two levels
- The digital-design levels are:
  1. The physical level
  2. The transistor level
  3. The logic-design level
  4. Computer design level (also called register-transfer level, or RTL)
  5. System-design level
- The level of abstraction increases from physical to system level.
Digital-Design Levels: physical level

- Deals with “device physics and IC manufacturing process” [Wakerly]
- Main responsible for the progress in IC design (speed and density) in the past decades
- *Moore’s Law*, stated by Gordon Moore (Intel founder) in 1965: “the number of transistors per square inch in the newest IC will double every year”
- In the last years the doubling of density takes almost 2 years
- We do not study this level, but it influences digital design: the reducing of transistor sizes caused the decrease of power-supply voltages
- This produced major changes in digital design of ICs:
- New research domains have emerged: low-power design, low-power testing, low-power synthesis, etc
Digital-Design Levels

- Transistor level
  - It is not the subject of the Digital Logic course
  - It will be the main topics of the *Integrated Circuits* course

- **Logic-design level**
  - It is the level of *Digital Logic* course
  - We will learn how to synthesize RTL devices (multiplexers, demultiplexers, decoders, counters, registers, etc) from gates and flip-flops

- Computer design level, or Register-transfer level (RTL):
  - Courses: *Computer Architecture, Computer Organization, Microprocessors*
  - You will use what you learn at the Digital Logic course

- System-design level
  - Courses: *Operating systems, Compilers, etc*
Digital-Design Levels representations of a multiplexer

- Specification:
  - Design a multiplexer (or MUX) circuit with two data inputs $A$ and $B$, a control input $S$ and an output $Z$
  - If $S=0$ then the input $A$ is transferred at the output $Z$ (i.e. $Z=A$);
  - If $S=1$ then the input $B$ is transferred at the output $Z$ (i.e. $Z=B$);
  - All inputs and the output are one bit wide
- The functioning of the MUX as a switch is given is figure 13
- The truth table of the MUX is shown in figure 14
- From the truth table, we derive the equation for the MUX:
  \[
  Z = S' \cdot A + S \cdot B
  \]
  - The equation reads: “$Z$ equals not $S$ and $A$, or $S$ and $B$”
- The gate-level logic diagram that corresponds to this equation is shown in figure 15, and the MSI IC in figure 16
Digital-Design Levels representations of a multiplexer

Table 1-1
Truth table for the multiplexer function.

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 13: Switch model for multiplexer function

Figure 14: Truth table for multiplexer function
Digital-Design Levels representations of a multiplexer

Figure 15: Gate-level logic diagram for multiplexer function

Figure 16: Logic diagram for a multiplexer using an MSI circuit