Introduction to Digital Design

Prof. Dr. Berna Örs Yalçın

Istanbul Technical University
Faculty of Electrical and Electronics Engineering
Office Number: 2318
E-mail: siddika.ors@itu.edu.tr

Goals and objectives

- The goal of this course is to:
  - provide a good understanding of the digital systems.
  - introduce the basic building blocks of digital design including combinational logic circuits, combinational logic design, arithmetic functions and circuits and sequential circuits.
  - Showing how these building blocks are employed in larger scale digital systems
- Having successfully completed this course, the student will:
  - acknowledge the importance of digital systems.
  - Design a digital circuit given a Boolean function.
  - Get familiar with typical combinatorial (adders, decoders, multiplexers, encoders) and sequential (D flip-flops, counters, registers, shift registers) components.
  - Understand how larger systems are organized.

References

- Text Books :
  - Digital Design, M. Morris Mano, Michael D. Ciletti,
  - Logic and Computer Design Fundamentals, 4/E, M. Morris Mano and Charles Kime , Prentice Hall, 2008.
- Slides and all anouncements : ninova

Grading

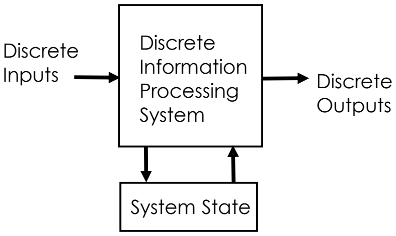
- 1st Midterm - % 25
  - 6th week
- 2nd Midterm – % 25
  - 11th week
- 5 Homeworks - % 10
- Final Exam - % 40

What is a Digital System?

- One characteristic:
  - Ability of manipulating discrete elements of information
- A *set* that has a finite number of elements contains discrete information
- Examples for discrete sets
  - Decimal digits {0, 1, …, 9}
  - Alphabet {A, B, …, Y, Z}
  - Binary digits {0, 1}
- One important problem
  - how to represent the elements of discrete sets in physical systems?

DIGITAL & COMPUTER SYSTEMS - Digital System

- **Takes a set of discrete information inputs and discrete internal information (system state) and generates a set of discrete information outputs.**

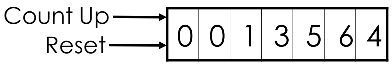


Types of Digital Systems

- **No state present**
  - **Combinational Logic System**
  - **Output = Function(Input)**
- **State present**
  - **State updated at discrete times**
    => Synchronous Sequential System
  - **State updated at any time**
    =>Asynchronous Sequential System
  - **State = Function (State, Input)**
  - **Output = Function (State) or Function (State, Input)**

Digital System Example:

A Digital Counter (e. g., odometer):



Inputs: **Count Up, Reset**

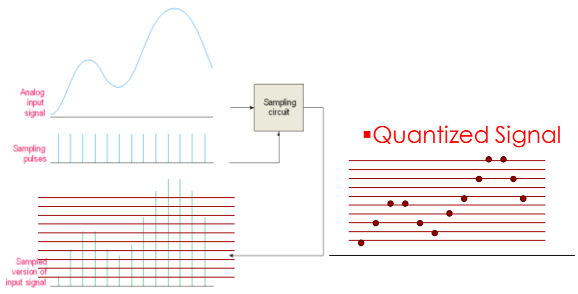
Outputs: **Visual Display**

State: **"Value" of stored digits**

Analog – Digital Signals

- **The physical quantities in real world like current, voltage, temperature values change in a continuous range.**
- **The signals that can take any value between the boundaries are called analog signals.**
- **Information take discrete values in digital systems.**
- **Binary digital signals can take one of the two possible values: 0-1, high-low, open-closed.**

Conversion of Analog Signals to Digital Signals



How to Represent?

- In electronics circuits, we have electrical signals
  - voltage
  - current
- Different strengths of a physical signal can be used to represent elements of the discrete set.
- Which discrete set?
- Binary set is the easiest
  - two elements {0, 1}
  - Just two signal levels: 0 V and 4 V
- This is why we use binary system to represent the information in our digital system.

Binary System

- Binary set {0, 1}
  - The elements of binary set, 0 and 1 are called "binary digits"
  - or shortly "bits".
- How to represent the elements of other discrete sets
  - Decimal digits {0, 1, …, 9}
  - Alphabet {A, B, …, Y, Z}
- Elements of any discrete sets can be represented using groups of bits.
  - 9 → 1001
  - A → 1000001

How Many Bits?

- What is the formulae for number of bits to represent a discrete set of *n* elements?

- {0, 1, 2, 3}
  - 00 → 0, 01 → 1, 10 → 2, ands 11 → 3.
- {0, 1, 2, 3, 4, 5, 6, 7}
  - 000 → 0, 001 → 1, 010 → 2, ands 011 → 3
  - 100 → 4, 101 → 5, 110 → 6, ands 111 → 7.
- The formulae, then,
  - ?
  - If n = 9, then 4 bits are needed

Representation of positive numbers

- Örnek: $215_{10}=(1101\ 0111)_2 = 1{\cdot}2^7+1{\cdot}2^6+0{\cdot}2^5+1{\cdot}2^4+0{\cdot}2^3+1{\cdot}2^2+1{\cdot}2^1+1{\cdot}2^0$
- Most Signaficant Bit (MSB)
- Least Signaficant Bit (LSB)

- The largest postive number that can be represented by 8 bits is:
  - $(1111\ 1111)_2=255_{10}$
- The smallest postive number that can be represented by 8 bits is:
  - $(0000\ 0000)_2=0_{10}$

Some Bases

Name	Base	Digits
Binary	2	0,1
Octal	8	0,1,2,3,4,5,6,7
Decimal	10	0,1,2,3,4,5,6,7,8,9
Hexadecimal	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Some Numbers in Different Bases

Decimal (Base 10)	Binary (Base 2)	Octal (Base 8)	Hexa decimal (Base 16)
00	00000	00	00
01	00001	01	01
02	00010	02	02
03	00011	03	03
04	00100	04	04
05	00101	05	05
06	00110	06	06
07	00111	07	07
08	01000	10	08
09	01001	11	09
10	01010	12	0A
11	0101 1	13	0B
12	01100	14	0C
13	01101	15	0D
14	01110	16	0E
15	01111	17	0F
16	10000	20	10

Base Conversions

- From base-r to decimal is easy
  - expand the number in power series and add all the terms
- Reverse operation is somewhat more difficult
- Simple idea:
  - divide the decimal number successively by r
  - accumulate the remainders.

Example: 46_{10} to base 2

- Convert 46 to binary
  - 46/2=23 remainder 0
  - 23/2=11 remainder 1
  - 11/2=5 remainder 1
  - 5/2=2 remainder 1
  - 2/2=1 remainder 0
  - 1/2=0 remainder 1
- Put results together
  - 101110_2

Example: 46_{10} to hexadecimal (base 16)

- Convert 46 to base 16
  - 46/16=2 remainder 14
  - 2/16=0 remainder 2
- Put results together
  - $2E_{16}$

Conversion from base r to base decimal

- Convert 101110_2 to base 10

$$\begin{aligned} 101110_2 &= 1\cdot 32 + 0\cdot 16 + 1\cdot 8 + 1\cdot 4 + 1\cdot 2 + 0\cdot 1 \\ &= 32 + 8 + 4 + 2 \\ &= 46 \end{aligned}$$

Conversions between Binary, Octal and Hexadecimal

- Octal to Binary
  - 743.056_8=111 100 011.000 101 110_2
- Hexadecimal to Binary
  - $A49.0C6_{16}$=1010 0100 1001.0000 1100 0110_2
- Binary to Octal
  - 1 | 011 | 100 | 011.000 | 101 | 110 | 1_2=1343.0564_8
- Binary to Hexadecimal
  - 1 | 1010 | 0100 | 1001.0010 | 1100 | 0110 | 1_2=$1A49.2C68_{16}$

- Octal and hexadecimal representations are more compact.
- Therefore, we use them in order to communicate with computers directly using their internal representation

Binary Numbers and Binary Coding

- **Flexibility of representation**
  - **Within constraints below, can assign any binary combination (called a code word) to any data as long as data is uniquely encoded.**
- **Information Types**
  - **Numeric**
    - **Must represent range of data needed**
    - **Very desirable to represent data such that simple, straightforward computation for common arithmetic operations permitted**
    - **Tight relation to binary numbers**
  - **Non-numeric**
    - **Greater flexibility since arithmetic operations not applied.**
    - **Not tied to binary numbers**

Non-numeric Binary Codes

- **Given *n* binary digits (called <u>bits</u>), a <u>binary code</u> is a mapping from a set of <u>represented elements</u> to a subset of the 2^n binary numbers.**
- **Example: A binary code for the seven colors of the rainbow**
- **Code 100 is not used**

Color	Binary Number
Red	000
Orange	001
Yellow	010
Green	011
Blue	101
Indigo	110
Violet	111

Number of Elements Represented

- **Given *n* digits in radix *r*, there are r^n distinct elements that can be represented.**
- **But, you can represent m elements, m < r^n**
- **Examples:**
  - **You can represent 4 elements in radix r = 2 with n = 2 digits: (00, 01, 10, 11).**
  - **You can represent 4 elements in radix r = 2 with n = 4 digits: (0001, 0010, 0100, 1000).**
  - **This second code is called a "<u>one hot</u>" code.**