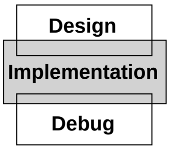
Chapter 1:Introduction

- The process of design
- Digital vs. Analog
- Digital system
- Positive vs. Negative logic
- Number systems
- Combinational vs. Sequential logic
- Truth table
- Don’t care condition
- The design process for combinational systems

The Process of Design



- Design

Initial concept: what is the function performed by the object?
Constraints: How fast? How much area? How much cost?
Refine abstract functional blocks into more concrete realisations

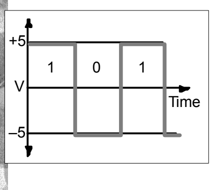
- Implementation

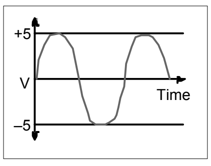
Assemble primitives into more complex building blocks
Composition via wiring
Choose among alternatives to improve the design

- Debug

Faulty systems: design flaws, composition flaws, component flaws
Design to make debugging easier
Hypothesis formation and troubleshooting skills

Digital vs. Analog

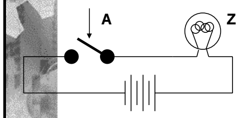




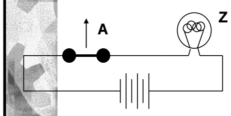
Digital:
only assumes discrete values

Analog:
values vary over a broad range continuously

Digital System



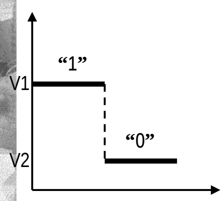
- **close switch**
  (if A is “1” or asserted)
  and turn on light bulb (Z)



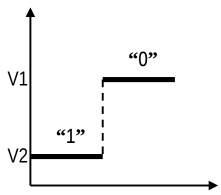
- **open switch**
  (if A is “0” or unasserted)
  and turn off light bulb (Z)

DDH1

Positive vs. Negative Logic



Positive logic



Negative logic

Number Systems

$$N=a_{n-1}r^{n-1}+ a_{n-2}r^{n-2}+ \ldots +a_2r^2+ a_1r^1+ a_0r^0$$

n is the number of digits
r is the radix or base
a_i are the coeffecients; $0 \le a_i < r$

- **Decimal:** r=10; $a_i \Rightarrow$ 0 to 9
- **Binary:** r=2; $a_i \Rightarrow$ 0 and 1
- **Octal:** r=8; $a_i \Rightarrow$ 0 to 7
- **Hexadecimal:** r=16; $a_i \Rightarrow$ 0 to 9 and A to F

Slide 5

DDH1 Dr. Daranee Hormdee, 15-Jul-03

First 16 integers

Decimal	4-bit Binary	Octal	Hexadecimal
0	0000_2	00_8	0_{16}
1	0001_2	01_8	1_{16}
2	0010_2	02_8	2_{16}
3	0011_2	03_8	3_{16}
4	0100_2	04_8	4_{16}
5	0101_2	05_8	5_{16}
6	0110_2	06_8	6_{16}
7	0111_2	07_8	7_{16}
8	1000_2	10_8	8_{16}
9	1001_2	11_8	9_{16}
10	1010_2	12_8	A_{16}
11	1011_2	13_8	B_{16}
12	1100_2	14_8	C_{16}
13	1101_2	15_8	D_{16}
14	1110_2	16_8	E_{16}
15	1111_2	17_8	F_{16}

การแปลงเลขฐาน

Binary ⇔ Octal and Hexadecimal

$$1011101010_2 = \underbrace{001}\ \underbrace{011}\ \underbrace{101}\ \underbrace{010}_2$$
$$= 1\quad 3\quad 5\quad 2_8$$

$$= \underbrace{0010}\ \underbrace{1110}\ \underbrace{1010}_2$$
$$= 2\quad E\quad A_{16}$$

การแปลงเลขฐาน (cont.)

- **To Decimal**

$7642_{10} = 7\times10^3 + 6\times10^2 + 4\times10 + 2$

$101111_2 = 1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 1\times2 + 1$

$1352_8 = 1\times8^3 + 3\times8^2 + 5\times8 + 2$

$2EA_{16} = 2\times16^2 + 14\times16 + 10$

- **From Decimal**
  - ใช้การหารสั้น

การแปลงเลขฐาน (cont.)

46/ 2 = 23	เศษ 0	produces	0
23/ 2 = 11	เศษ 1		10
11/ 2 = 5	เศษ 1		110
5 / 2 = 2	เศษ 1		1110
2 / 2 = 1	เศษ 0		01110
1 / 2 = 0	เศษ 1		101110_2

746 / 8 = 93	เศษ 2	produces	2
93 / 8 = 11	เศษ 0		02
11 / 8 = 1	เศษ 3		302
1 / 8 = 0	เศษ 1		1302_8

- Up to this point:
  - positive integers
  - unsigned numbers
- Next:
  - signed numbers

Signed Numbers

- Signed-magnitude
  - +5 → 0101, -5 → 1101;
  - +3 → 0011, -3 → 1011
  - 4-bit number: range from -7 (1111) to +7 (0111)
  - Has 2 forms for 0 → 0000 and 1000
  - Its difficulty of addition and subtraction
- Radix complement:
- One's complement
- Two's complement

Radix's Complement

Radix complement of an n-digit number D is obtained by subtracting it from r^n.

In the decimal number system, the radix complement is called the 10's complement.

Number	10's complement	9's complement
1849	8151	8150
2067	7933	7932
100	9900	9899
8151	1849	1848
0	10000 (=0)	9999

One's Complement

Two-steps:

- find the binary equivalent of the magnitude.
- complement each bit (change 0's to 1's and 1's to 0's).

Examples:

1.	5:	0101	1:	0001	0:	0000
2.	-5:	1010	-1:	1110	0:	1111

Two's Complement

Three-steps:

- find the binary equivalent of the magnitude.
- complement each bit (change 0's to 1's and 1's to 0's).
- add 1.

Examples:

1.	5:	0101	1:	0001	0:	0000
2.		1010		1110		1111
3.		+ 1		+ 1		+ 1
	-5:	1011	-1:	1111	0:	0000

Note that there is no negative zero in two's complement format.

Binary Addition & Subtraction

- a-b = a+ (-b) using 2's complement
- In 2's complement addition, the carry out of the most significant bit is ignored.
- *Overflow* occurs when the sum is out of range. For 4-bit numbers, that range is -8 ≤ sum ≤ +7.
- Example: 7-5

```
5:   0101          7:    0111
     1010          -5:   1011 +
     +  1          2: (1) 0010
-5:  1011           ignored
```


Binary Addition & Subtraction

```
-5:    1011 +    +5:    1011+     -5:    1011+
+7:    0111      -5:    0101      +3:    0011
+2: (1) 0011     0: (1) 0000      -2: (0) 1110
 ignored          ignored          ignored

 +5:   0101                   -5:   1011
 +4:   0100                   -4:   1100
   (0) 1001                     (1) 0111
      (looks like -7)               (looks like +7)
should be +9 > the range     should be -9 < the range
```


Binary Coded Decimal (BCD)

Decimal digit	8421 code	5421 code	Excess 3 code	2 of 5 code
0	0000	0000	0011	11000
1	0001	0001	0100	10100
2	0010	0010	0101	10010
3	0011	0011	0110	10001
4	0100	0100	0111	01100
5	0101	1000	1000	01010
6	0110	1001	1001	01001
7	0111	1010	1010	00110
8	1000	1011	1011	00101
9	1001	1100	1100	00011
unused	1010	0101	0000	Any of the
	1011	0110	0001	22 patterns
	1100	0111	0010	With 0,1,3,
	1101	1101	1101	4 or 5 1's
	1110	1110	1110	
	1111	1111	1111	

Combinational vs. Sequential Logic

[redacted]

Case1: a system with 2 inputs, A and B, and one output Z, which is 1 *iff* one of the inputs is 1.

Combinational logic

[redacted]

Case2: a system with one input, A, and one output Z, which is 1 *iff* both the input and the last input are 1's.

Sequential logic

The presence of feedback distinguishes between *combinational* and *sequential* systems.

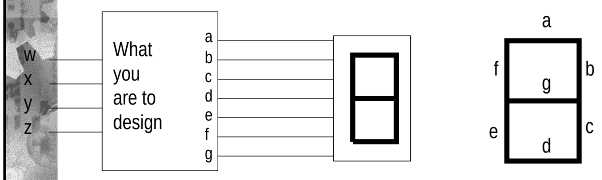
Truth Table

a system with 2 inputs, A and B, and one output Z, which is 1 *iff* one of the inputs is 1.

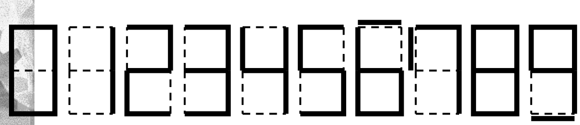
A two-input truth table

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	0

Don't Care Condition



A seven-segment display



Don’t Care Condition (cont.)

Digit	w	x	Y	z	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	X	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	x	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	x	0	1	1
-	1	0	1	0	X	X	X	X	X	X	X
-	1	0	1	1	X	X	X	X	X	X	X
-	1	1	0	0	X	X	X	X	X	X	X
-	1	1	0	1	X	X	X	X	X	X	X
-	1	1	1	0	X	X	X	X	X	X	X
-	1	1	1	1	X	X	X	X	X	X	X

The design process for combinational systems

- Represent each of the inputs and outputs
- Formalise the design specification either in the form of a truth table or of an algebraic expression
- Simplify the description
- Implement the system with the available components, subject to the design objectives and constraints

If necessary, break the problem into smaller sub-problems