Digital Circuits

ECS 371

Dr. Prapun Suksompong

prapun@siit.tu.ac.th

Lecture 2

Office Hours:
BKD 3601-7
Monday 1:30-3:30
Tuesday 10:30-11:30

ECS371.PRAPUN.COM

Announcement

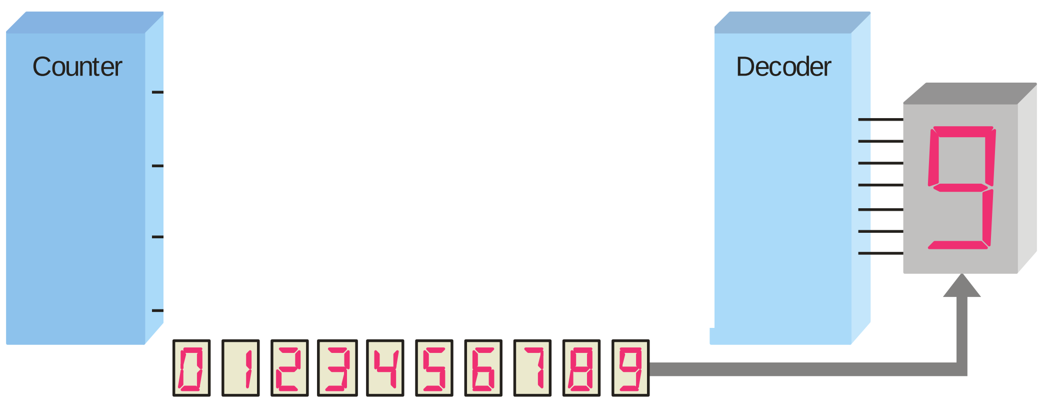
- **The fact that I will end the class on time does NOT mean that I will leave the room immediately after the lecture.**
  - **I will stay to answer questions.**
- **Reading Assignment:**
  - **Chapter 1: 1-1, 1-2**
  - **Chapter 2: 2-1, 2-2, 2-3**
  - **Skip 2.4**
  - **Chapter 2: 2-5, 2-6**
  - **Skip 2-7 to 2-12**
  - **Chapter 3: ALL**

Binary Counting

A binary counting sequence for numbers from zero to fifteen is shown.

Notice the pattern of zeros and ones in each column.

Digital counters frequently have this same pattern of digits:



0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1
10	1 0 1 0
11	1 0 1 1
12	1 1 0 0
13	1 1 0 1
14	1 1 1 0
15	1 1 1 1

Binary Addition

The rules for binary addition are

0 + 0 = 0	Sum = 0, carry = 0
0 + 1 = 1	Sum = 1, carry = 0
1 + 0 = 1	Sum = 1, carry = 0
1 + 1 = 10	Sum = 0, carry = 1

When an input carry = 1 due to a previous result, the rules are

1 + 0 + 0 = 01	Sum = 1, carry = 0
1 + 0 + 1 = 10	Sum = 0, carry = 1
1 + 1 + 0 = 10	Sum = 0, carry = 1
1 + 1 + 1 = 11	Sum = 1, carry = 1

Representation of Negative Numbers

- Digital Logic represents numbers as *n*-bit binary numbers, with fixed *n*.
- Some important operations:
  1. **1's complement**: Change all 1s to 0s and all 0s to 1s.
  2. **2's complement**: Add 1 to the LSB of the 1's complement.
     - If the addition produces a result that requires more than *n* digits, we throw away the extra digit(s).
  - If a number *D* is complemented twice, the result is *D*.
- An alternative method of finding the 2's complement: Change all bits to the left of the least significant 1.
  1. Start at the right with the LSB and write the bits as they are up to and including the first 1.
  2. Take the 1's complements of the remaining bits.

Signed Binary Number

- A signed binary number consists of both sign and magnitude information.
  - The sign indicates whether a number is positive or negative
    - In a signed binary number, the left-most bit (MSB) is the **sign bit**.
    - 0 indicates a positive number, and
      1 indicates a negative number
  - The magnitude is the value of the number.
- There are three forms in which signed integer (whole) numbers can be represented in binary:
  1. sign-magnitude,
  2. l's complement,
  3. and 2's complement.
- Of these, the 2's complement is the most important

Signed Binary Number

(1) Sign-Magnitude Form

- The magnitude bits are in true (uncomplemented) binary for both positive and negative numbers.
- Negate a number by changing its sign.

(2) 1's Complement Form

- A negative number is the 1's complement of the corresponding positive number.

There are two possible representations of zero, "+0" and "-0", but both have the same value.

	Sign-Magnitude	1's Complement	2's Complement
000	0	0	0
001	1	1	1
010	2	2	2
011	3	3	3
100	-0	-3	-4
101	-1	-2	-3
110	-2	-1	-2
111	-3	-0	-1

Signed Binary Number (2)

(3) 2’s Complement Form

- A negative number is the 2's complement of the corresponding positive number.
- The weight of the sign bit is given a negative value.
- Decimal values are determined by summing the weights in all bit positions where there are 1s and ignoring those positions where there are zeros.
- Has only one representation of zero.
- Zero is considered positive because its sign bit is 0.

	Sign-Magnitude	1’s Complement	2’s Complement
000	0	0	0
001	1	1	1
010	2	2	2
011	3	3	3
100	-0	-3	-4
101	-1	-2	-3
110	-2	-1	-2
111	-3	-0	-1

2's Complement Representation (con't)

- The number of different combinations of n bits is 2^n
- For n bit 2's complement signed numbers, the range is

$$-\left(2^{n-1}\right) \text{ to } +\left(2^{n-1}-1\right)$$

- Has one extra negative number
  - This number does not have a positive counterpart.
- To convert n-bit 2's complement number into m-bit one:
  - If $m > n$, append m-n copies of the sign bit.
    - This is called ***sign extension***.
  - If $m < n$, discard n-m leftmost bits
    - The result is valid only if all of the discarded bits are the same as the sign bit of the result.