Lecture 4
Binary Arithmetic

Binary Arithmetic

- Addition
- Subtraction
- Complements – 1's and 2's

Binary Addition

(a)

$$\begin{array}{r} 0 \\ +\,0 \\ \hline 0 \end{array}$$

(b)

$$\begin{array}{r} 0 \\ +\,1 \\ \hline 1 \end{array}$$

(c)

$$\begin{array}{r} 1 \\ +\,0 \\ \hline 1 \end{array}$$

(d)

$$\begin{array}{r} 1 \\ +\,1 \\ \hline 1\,0 \end{array}$$

Carry Bit

Binary Addition Examples

(a)
```
   1011
+  1100
-------
  10111
```

(b)
```
   1010
+   100
-------
   1110
```

(c)
```
   1011
+   101
-------
  10000
```

(d)
```
    101
+  1001
-------
   1110
```

(e)
```
   10011001
+    101100
-----------
   11000101
```

Binary Complement

(1s Complement) Operation

1 → 0

0 → 1

Example

1 1 0 0 1 0 1 1 0

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

0 0 1 1 0 1 0 0 1

Two's Complement

The Two's complement of a binary number is obtained by first complementing the number and then adding 1 to the result.

1001110

0110001 ← **One's Complement**

+ 1

0110010 ← **Two's Complement**

Binary Subtraction

Binary subtraction is implemented by adding the Two's complement of the number to be subtracted.

Example

Two's complement of 1001

```
  1101          1101
 -1001         +0111
               -----
               10100
```

If there is a carry then it is ignored. Thus, the answer is 0100.

Basic Digital Arithmetic

- Signed Binary Number: A binary number of fixed length whose sign (+/-) is represented by one bit (usually MSB) and its magnitude by the remaining bits
- Unsigned Binary Number: A binary number of fixed length whose sign is not specified by a bit. All bits are magnitude and the sign is assumed +.

Signed Binary Numbers I

- Sign Bit: A bit (usually the MSB) that indicates whether a number is positive(=0) or negative (=1).
- Magnitude Bits: The bits of a signed binary number that tell how large it is in value.
- True Magnitude Form: A form of signed binary whose magnitude bits are the TRUE binary form (not complements).

Signed Binary Numbers II

- 1s Complement: A form of signed binary in which negative numbers are created by complementing all bits.
- 2s Complement: A form of signed binary in which the negative numbers are created by complementing all the bits and adding a 1 (1s Complement +1).

Unsigned Binary Arithmetic

- Sum: Result of an Addition Operation of two (or more) binary numbers (operands).
- Carry: A digit (or bit) that is carried over to the next most significant bit during an N Bit addition operation.
- The carry bit is a 1 if the result was too large to be expressed in N bits.

Basic Rules (Unsigned)

- One Bit Unsigned Addition

0 + 0 =	0 0
1 + 0 =	0 1
1 + 1 =	1 0

1 + 1 + 1 = 1 1

CIN A B COUT SUM

True Magnitude Form

- 5 Bit Numbers Negative = S=1

+25 = 011001

-25 = 111001

-12 = 101100

+12 = 001100

Note sign bit = MSB = S = 0

Same as +25 with S=1

True magnitude

2's complement of a binary number:

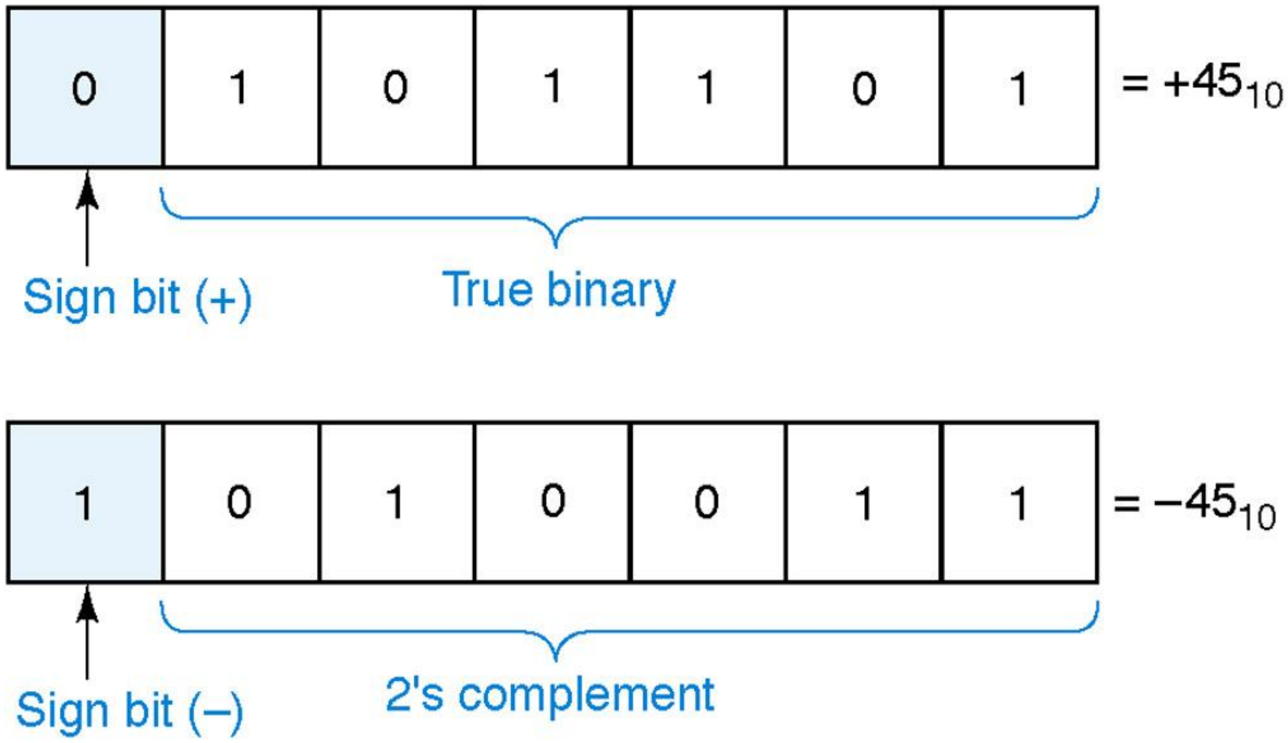
– Take the 1's complement of the number
– Add 1 to the least-significant-bit position

101101	binary equivalent of 45
010010	complement each bit to form 1's complement
+ 1	add 1 to form 2's complement
010011	2's complement of original binary number

Representing signed numbers using 2's complement form

- If the number is positive, the magnitude is represented in its positional-weighted binary form, and a sign bit of 0 is placed in front of the MSB.
- If the number is negative, the magnitude is represented in its 2's complement form, and a sign bit of 1 is placed in front of the MSB.

example


0
1
0
1
1
0
1
= +45₁₀
Sign bit (+)
True binary
1
0
1
0
0
1
1
= −45₁₀
Sign bit (–)
2's complement

Example

- Represent each of the following signed decimal numbers as a signed binary number in the 2's-complement system. Use a total of five bits including the sign bit.

  (a) +13 (b) –9 (c) +3 (d) –2 (e) -8

Addition in the 2’s-complement system

- Case I: Two Postive Numbers.

$$
\begin{array}{llll}
+9 \rightarrow & 0 & 1001 & \text{(augend)} \\
\underline{+4 \rightarrow} & \underline{0} & \underline{0100} & \underline{\text{(addend)}} \\
 & 0 & 1101 & \text{(sum = +13)}
\end{array}
$$

Sign bits

Addition, cont.

- Case II: Positive Number and Smaller Negative Number

+9 → 0 1001 (augend)

Sign bits

-4 → 1 1100 (addend)

1 0 0101

This carry is disregarded; the result is 0101(sum=+5)

Addition, cont.

- Case III: Positive Number and Larger Negative Number

Negative sign bit

$$
\begin{array}{rl}
-9 \rightarrow & 10111 \\
\underline{+4 \rightarrow} & \underline{00100} \\
 & 11011 \quad \text{(sum = -5)}
\end{array}
$$

Addition, cont.

- Case IV: two negative Numbers

$$
\begin{array}{r}
-9 \rightarrow 10111 \\
\underline{-4 \rightarrow 11100} \\
1\ 10011
\end{array}
$$

Sign bit

This carry is disregarded; the result is 10011(sum =-13)

Negative Result Example

- 2s Complement Negative Result (65-80)

+65 =	0 100 0001	100 0001
-80 =	1 101 0000 (2s C.)	+ 011 0000
		111 0001
	Invert	000 1111
	Add 1	+ 1
	Final Result = -15	0000 1111 = 15(Neg.)

Addition, cont.

- Case V: Equal and Opposite Numbers

$$
\begin{array}{l}
-9 \rightarrow \ 1 \quad 0111 \\
\underline{+9 \rightarrow \ 0 \ \ 1001} \\
0 \quad 1 \quad\ \ 0000
\end{array}
$$

Disregard; the result is 0000(sum = +0)

Subtraction in the 2’s-complement System

- The procedure for subtracting one binary number(the subtrahend) from another binary number(the minuend)
  - Negate the subtrahend. This will change the subtrahend to its equivalent value of opposite sign.
  - Add this to the minuend. The result of this addition will represent the difference between the subtrahend and the minuend.

Addition and Subtraction of BCD and Excess-3 Code

Unsigned Numbers BCD Addition

Use binary arithmetic to add the BCD digits:

8	1000	Eight
+5	+0101	Plus 5
13	1101	is 13 (> 9)

3	
+5	
8	OK (< 9)

If result is > 9, it must generate a carry and be corrected!

To correct the digit, add 0110 in the result.

8	1000	Eight
+5	+0101	Plus 5
13	1101	13 (is > 9)
	+0110	so add 6 (always, for results > 9)
	carry = 1 0011	giving 3 + carry
	0001 | 0011	Final answer (two digits)

We try to avoid subtraction!
Replacing it with addition!

The adder circuit utilizes the resulting carry bit by sending it as carry-in to the next digit

Add 2905_{BCD} to 1857_{BCD} showing carries and digit corrections.

1010	1	0	1	0
1857	0001	1000	0101	0111
+2905	+ 0010	1001	0000	0101
	0100 ($4<9 \rightarrow +0$)	10001 ($17>9 \rightarrow +6$)	0110 ($6<9 \rightarrow +0$)	1100 ($12>9 \rightarrow +6$)
Add 6 or 0	+ 0000	0110	0000	0110
4762	0100	1 0111	0110	1 0010

Excess-3 Code

A BCD Code formed by adding 3 (0011) to its true 4-bit binary value.

Excess-3 is a self-complementing code:

A negative code equivalent can be found by inverting the binary bits of the positive code

Inverting the bits of the Excess-3 digit yields 9's Complement of the decimal equivalent.

Example : Excess -3 code of decimal 4 is 0111. (0100 + 0011 = 0111)

(4) = 0111

(-4) = 1000 (inverting the bits) which is Excess -3 code of decimal 5.

It is 9's complement of the decimal equivalent. (9 – 4 = 5)

Excess-3 Examples

- 3 = 0011 + 0011 = 0110 = 6 in E3.
- 1 = 0001 + 0011 = 0100 = 4 in E3.
- If we complement 1 = 1011 in E3, this is the code for an 8.
- 9's Complement of 1 = (9 – 1) = 8 (SelfComplement)