Digital Electronics and Microprocessors

Class 18

CHHAYADEVI BHAMARE

Digital Arithmetic: Operations and Circuits(Chapter 6)

Arithmetic operations

- Binary addition
- Representing Signed Numbers
- Addition and subtractionin the 2’s complement systems.
- Multiplication and division of binary numbers
- BCD addition
- Hexadecimal Arithmetic

Digital Arithmetic: Operations and Circuits(Chapter 6)

Arithmetic Circuits

- Design of a full adder
- Parallel binary adder
- Complete parallel adder with registers
- Carry propogation
- IC arithmetic circuits (parallele adder, cascading of parallel adder, ALU IC etc)

Binary Addition

- Binary numbers are added like decimal numbers.
- In decimal, when numbers sum more than 9 a carry results.
- In binary when numbers sum more than 1 a carry takes place.
- Addition is the basic arithmetic operation used by digital devices to perform subtraction, multiplication, and division.

4 different cases for binary addition

$$0 + 0 = 0$$

$$1 + 0 = 1$$

$$1 + 1 = 10 = 0 + \text{carry of } 1 \text{ into next position}$$

$$1 + 1 + 1 = 11 = 1 + \text{carry of } 1 \text{ into next position}$$

Example

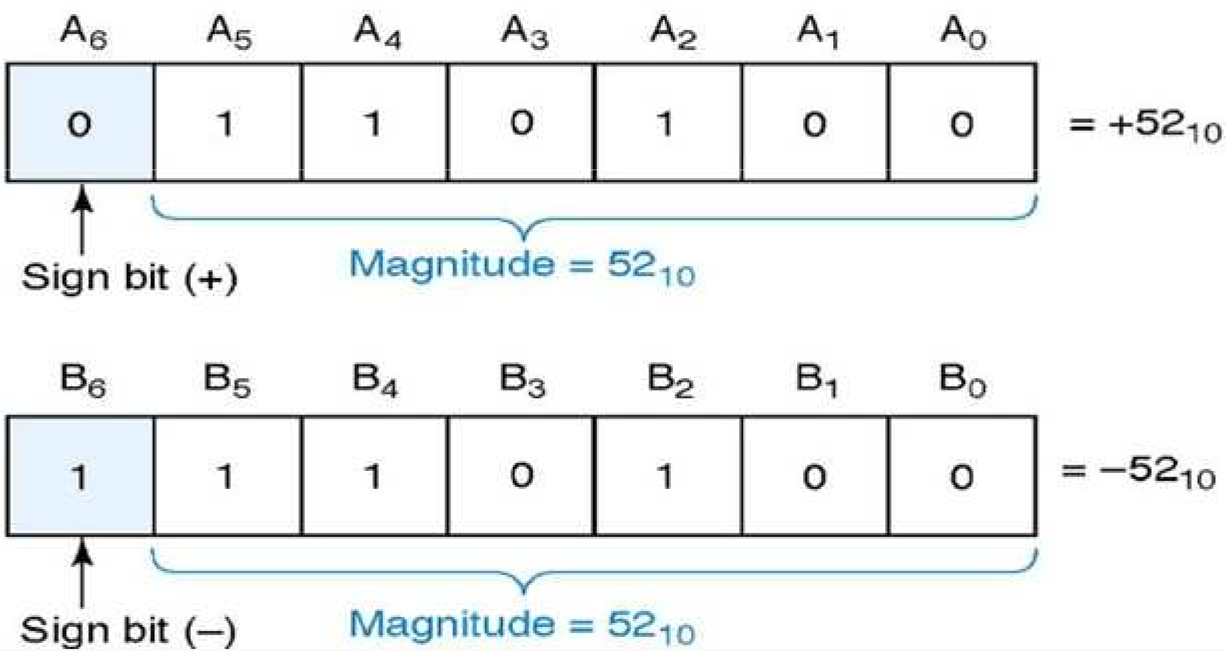
$$\begin{array}{r} 011\,(3) \\ +110(6) \\ \hline 1001(9) \end{array} \qquad \begin{array}{r} 1001(9) \\ +1111(15) \\ \hline 11000(24) \end{array} \qquad \begin{array}{r} 11.011(3.375) \\ +10.110(2.750) \\ \hline 110.001(6.125) \end{array}$$

Representing Signed Numbers

- Since it is only possible to show magnitude with a binary number, the sign (+ or –) is shown by adding an extra “sign” bit.
- A sign bit of 0 indicates a positive number.
- A sign bit of 1 indicates a negative number.
- The 2’s complement system is the most commonly used way to represent signed numbers.

Considerations: representing both positive and negative numbers; efficient computation.

Sign-magnitude system



Representing Signed Numbers

- In order to change a binary number to 2's complement it must first be changed to 1's complement.
  - To convert to 1's complement, simply change each bit to its complement (opposite).
  - To convert 1's complement to 2's complement add 1 to the 1's complement.

1's complement system

- Change each 0 to 1, and each 1 to 0.
- Example

(45)	1	0	1	1	0	1	original binary number
	↓	↓	↓	↓	↓	↓	
(-45)	*0*	*1*	*0*	*0*	*1*	*0*	complement each bit

```
    1 0 1 1 0 1 (45)
+   0 1 0 0 1 0 (-45)
____________________
    1 1 1 1 1 1
```

Add one to this result, get zero.

2's complement of a binary number:

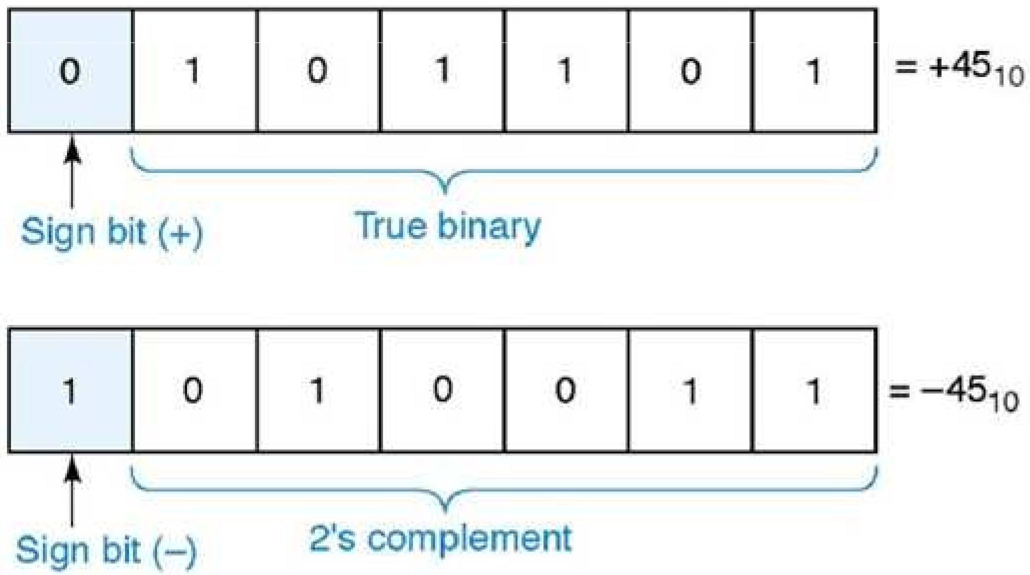
- **Take the 1's complement of the number**
- **Add 1 to the least-significant-bit position**

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

A positive number is true binary with 0 in the sign bit.
A negative number is in 2's complement form with 1 in the sign bit.

example



Addition in the 2's Complement System

- Perform normal binary addition of magnitudes.
- The sign bits are added with the magnitude bits.
- If addition results in a carry of the sign bit, the carry bit is ignored.
- If the result is positive it is in pure binary form.
- If the result is negative it is in 2's complement form.

Addition in the 2's-complement system

- Case I: Two Postive Numbers.

```
+9  →  | 0 |  1001   (augend)
+4  →  | 0 |  0100   (addend)
-----------------------------
       | 0 |  1101   (sum = +13)
```

Sign bits

- Case II: Positive Number and Smaller Negative Number

```
+9  →  | 0 |  1001  (augend)
-4  →  | 1 |  1100  (addend)
-----------------------------
     1 | 0 |  0101
```

Sign bits

This carry is disregarded; the result is 01001(sum=+5)

Addition, cont.

- **Case III: Positive Number and Larger Negative Number**

```
-9 →  10111
+4 →  00100
      11011     (sum = -5)
```

Negative sign bit

- **Case IV: two negative Numbers**

```
-9 → 10111
-4 → 11100
   1 10011
```

Sign bit

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

Addition, cont.

- Case V: Equal and Opposite Numbers

$$\begin{array}{lcr} -9 & \rightarrow & 10111 \\ +9 & \rightarrow & 01001 \\ \hline 0 & & 100000 \end{array}$$

Disregard; the result is 00000 (sum = +0)

Subtraction in the 2's Complement System

- The number subtracted (subtrahend) is negated.
- The result is added to the minuend.
- The answer represents the difference.
- If the answer exceeds the number of magnitude bits an overflow results.

Arithmetic Overflow

- When two positive or two negative numbers are being added, an overflow could occur if there is a carry happening to the sign-bit position.
- Overflow can occur when the minuend and subtrahend have different signs.

Multiplication of Binary Numbers

- This is similar to multiplication of decimal numbers.
- Each bit in the multiplier is multiplied by the multiplicand.
- The results are shifted as we move from LSB to MSB in the multiplier.
- All of the results are added to obtain the final product.

The same manner as the multiplication of decimal numbers.

```
    1001      ← multiplicand = 9₁₀
    1011      ← multiplier=11₁₀
    ----
    1001
   1001
  0000
 1001
 -------
 1100011        final product = 99₁₀
```

Multiplication in 2’s complement system

- If the two numbers to be multiplied are positive, they are already in true binary form and are multiplied as they are.

- When the two numbers are negative, they will be in2’s-complement form. Each is converted to a positivenumber, and then the two numbers are multiplied. The product is kept as a positive number and is given asign bit of 0.

- When one of the number is positive and the other isnegative, the negative number is first converted to a positive magnitude by taking its 2’s complement. The product will be in true-magnitude form, should bechanged to 2’s complement form and given a sign bitof 1.

Binary Division

- Exercise for you

Binary Division

- This is similar to decimal long division.
- It is simpler because only 1 or 0 are possible.
- The subtraction part of the operation is done using 2's complement subtraction.
- If the signs of the dividend and divisor are the same the answer will be positive.
- If the signs of the dividend and divisor are different the answer will be negative.

BCD Addition

- When the sum of each decimal digit is less than 9, the operation is the same as normal binary addition.
- When the sum of each decimal digit is greater than 9, a binary 6 is added. This will always cause a carry.
- BCD subtraction – a more complicated operation – is not discussed in the book.

Sum Equals 9 or Less

$$\begin{array}{rrl} 5 & 0101 & \leftarrow \text{BCD for 5} \\ \underline{+4} & +\underline{0100} & \leftarrow \text{BCD for 4} \\ 9 & 1001 & \leftarrow \text{BCD for 9} \end{array}$$

$$\begin{array}{rrl} 45 & 0100\ \ 0101 & \leftarrow \text{BCD for 45} \\ +\underline{33} & +\underline{0011\ \ 0011} & \leftarrow \text{BCD for 33} \\ 78 & 0111\ \ 1000 & \leftarrow \text{BCD for 78} \end{array}$$

Sum greater than 9

$$
\begin{array}{rrl}
6 & 0110 & \leftarrow \text{BCD for 6} \\
\underline{+7} & +\ \underline{0111} & \leftarrow \text{BCD for 7} \\
+13 & 1101 & \leftarrow \text{invalid code group for BCD}
\end{array}
$$

Sum greater than 9

$$
\begin{array}{rrl}
 & 0110 & \leftarrow \text{BCD for } 6 \\
+ & \underline{0111} & \leftarrow \text{BCD for } 7 \\
 & 1101 & \leftarrow \text{invalid sum} \\
 & \underline{0110} & \leftarrow \text{add } 6 \text{ for correction} \\
\underbrace{0001}_{1} & \underbrace{0011}_{3} & \leftarrow \text{BCD for } 13
\end{array}
$$

Sum greater than 9

47	0100	0111	← BCD for 47
+35	+ 0011	0101	← BCD for 35
82	0111	1100	← invalid sum in first digit
	1 ←	0110	← add 6 to correct
	1000	0010	← correct BCD sum
	$\underbrace{1000}_{8}$	$\underbrace{0010}_{2}$	

Perform this Hex addition

$$\begin{array}{r} 59 \\ +\underline{38} \\ 97 \end{array}$$

Ans

$$
\begin{array}{rrrl}
 & 1 & & \\
 & 0101 & 1001 & \leftarrow \text{BCD for 59} \\
+ & 0011 & 1000 & \leftarrow \text{BCD for 38} \\
\hline
 & 1001 & 0001 & \leftarrow \text{perform addition} \\
 & & 0110 & \leftarrow \text{add 6 to correct} \\
\hline
 & \underbrace{1001}_{9} & \underbrace{0111}_{7} & \ \text{BCD for 97}
\end{array}
$$

BCD addition procedure

1. Using ordinary binary addition, add the BCD code groups for each digit position.
2. For those positions where the sum is 9 or less, no correction is needed. The sum is in proper BCD form.
3. When the sum of two digits is greater than 9, a correction of 0110 should be added to that sum to get the proper BCD result. This case always produces a carry into the next digit position, either from the original addition (step 1) or from the correction addition.

Hexadecimal Arithmetic

- Hex addition:
  - Add the hex digits in decimal.
  - If the sum is 15 or less express it directly in hex digits.
  - If the sum is greater than 15, subtract 16 and carry 1 to the next position.

Hex addition Examples

$$\begin{array}{r} 58 \\ +\underline{24} \\ 7C \end{array} \qquad \begin{array}{r} 58 \\ +\underline{4B} \\ A3 \end{array} \qquad \begin{array}{r} 3AF \\ +\underline{23C} \\ 5EB \end{array}$$