


| | |
|---|---|
|  SASTRA <small>SAKSHI ACADEMY OF SCIENCE, ARTS, TECHNOLOGY & RESEARCH UNIVERSITY</small> <small>WISDOM BETTER KNOWLEDGE BETTER FUTURE</small> | COMPUTER ORGANIZATION AND ARCHITECTURE Course Code: CAP403R01 Semester : I / MCA Dr. K. Geetha/SAP/CSE/SOC/SASTRA |
|---|---|

1.4 Arithmetic for Computers: Addition and Subtraction

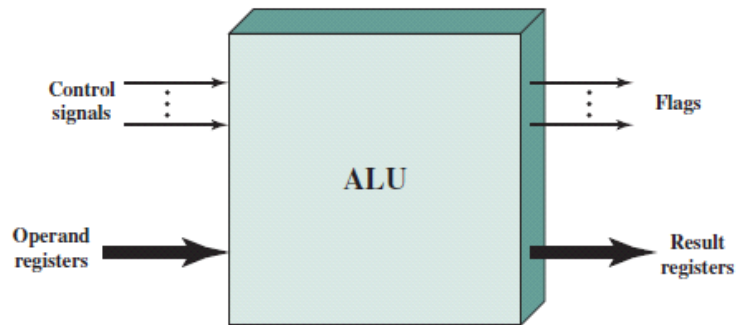


Figure 10.1 ALU Inputs and Outputs

10-2 Addition and Subtraction

There are three ways of representing negative fixed-point binary numbers:
 signed-magnitude
 signed-1's complement
 signed-2's complement.

| | | | |
|-----|---|----------|------------------|
| +18 | = | 00010010 | |
| -18 | = | 10010010 | (sign magnitude) |

Sign Magnitude

$$A = \begin{cases} \sum_{i=0}^{n-2} 2^i a_i & \text{if } a_{n-1} = 0 \\ -\sum_{i=0}^{n-2} 2^i a_i & \text{if } a_{n-1} = 1 \end{cases} \quad (10.1)$$

| | | | |
|------------------|---|----------|------------------|
| +0 ₁₀ | = | 00000000 | |
| -0 ₁₀ | = | 10000000 | (sign magnitude) |

Twos- complement representation

Consider an n -bit integer, A , in two's complement representation. If A is positive, then the sign bit, a_{n-1} , is zero. The remaining bits represent the magnitude of the number in the same fashion as for sign magnitude:

$$A = \sum_{i=0}^{n-2} 2^i a_i \quad \text{for } A \geq 0$$

For negative numbers

Two's Complement

$$A = -2^{n-1} a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i$$

Table 10.2 Alternative Representations for 4-Bit Integers

| Decimal Representation | Sign-Magnitude Representation | Two's Complement Representation | Biased Representation |
|------------------------|-------------------------------|---------------------------------|-----------------------|
| +8 | — | — | 1111 |
| +7 | 0111 | 0111 | 1110 |
| +6 | 0110 | 0110 | 1101 |
| +5 | 0101 | 0101 | 1100 |
| +4 | 0100 | 0100 | 1011 |
| +3 | 0011 | 0011 | 1010 |
| +2 | 0010 | 0010 | 1001 |
| +1 | 0001 | 0001 | 1000 |
| +0 | 0000 | 0000 | 0111 |
| −0 | 1000 | — | — |
| −1 | 1001 | 1111 | 0110 |
| −2 | 1010 | 1110 | 0101 |
| −3 | 1011 | 1101 | 0100 |
| −4 | 1100 | 1100 | 0011 |
| −5 | 1101 | 1011 | 0010 |
| −6 | 1110 | 1010 | 0001 |
| −7 | 1111 | 1001 | 0000 |
| −8 | — | 1000 | — |

Sign extension

In sign-magnitude notation, this is easily accomplished: simply move the sign bit to the new leftmost position and fill in with zeros.

| | | | |
|-----|---|------------------|---------------------------|
| +18 | = | 00010010 | (sign magnitude, 8 bits) |
| +18 | = | 0000000000010010 | (sign magnitude, 16 bits) |
| -18 | = | 10010010 | (sign magnitude, 8 bits) |
| -18 | = | 1000000000010010 | (sign magnitude, 16 bits) |

For twos complement representation

Instead, the rule for twos complement integers is to move the sign bit to the new leftmost position and fill in with copies of the sign bit. For positive numbers, fill in with zeros, and for negative numbers, fill in with ones. This is called sign extension.

| | | | |
|-----|---|------------------|----------------------------|
| -18 | = | 11101110 | (twos complement, 8 bits) |
| -18 | = | 1111111111101110 | (twos complement, 16 bits) |

For floating-point operations, most computers use the signed-magnitude representation for the mantissa.

Addition and Subtraction

1. Unsigned Integers

4 bit = 0 to $2^{(4-1)} = 0$ to 15

OVf : if actual result exceeds the range (above) larger than largest

UVF: If actual result falls less than the range (below) smaller than the smallest

Addition - 01- OVf, 10- UVF, C_{n+1} and C_{n-1} are different

1. (0111+0110)
2. (1000+0111)
3. (1111+1010)
4. (0100+0011)
5. (1000+1001)

Overflow in Integer Arithmetic

Using 2's-complement representation, n bits can represent values in the range

$-2n-1$ to $+2n-1 - 1$.

For example, the range of numbers that can be represented by 4 bits is -8 through $+7$

OVERFLOW RULE: If two numbers are added, and they are both positive or both negative, then overflow occurs if and only if the result has the opposite sign.

| | |
|--|---|
| $\begin{array}{r} 1001 = -7 \\ + 0101 = 5 \\ \hline 1110 = -2 \end{array}$ <p>(a) $(-7) + (+5)$</p> | $\begin{array}{r} 1100 = -4 \\ + 0100 = 4 \\ \hline 10000 = 0 \end{array}$ <p>(b) $(-4) + (+4)$</p> |
| $\begin{array}{r} 0011 = 3 \\ + 0100 = 4 \\ \hline 0111 = 7 \end{array}$ <p>(c) $(+3) + (+4)$</p> | $\begin{array}{r} 1100 = -4 \\ + 1111 = -1 \\ \hline 11011 = -5 \end{array}$ <p>(d) $(-4) + (-1)$</p> |
| $\begin{array}{r} 0101 = 5 \\ + 0100 = 4 \\ \hline 1001 = \text{Overflow} \end{array}$ <p>(e) $(+5) + (+4)$</p> | $\begin{array}{r} 1001 = -7 \\ + 1010 = -6 \\ \hline 10011 = \text{Overflow} \end{array}$ <p>(f) $(-7) + (-6)$</p> |

Figure 10.3 Addition of Numbers in Twos Complement Representation

When the actual result of an arithmetic operation is outside the representable range, an **arithmetic overflow** has occurred.

When adding **unsigned numbers**, a **carry-out of 1** from the MSB - indicates an overflow
 when adding **signed numbers** - value of the carry-out bit from the sign-bit position is not an indicator of overflow.

For example, if we add $+7$ and $+4$, the sum vector is 1011 , which is the representation for -5 , an incorrect result.

In this case, the carry-out bit from the MSB position is 0. If we add -4 and -6 , we get $0110 = +6$, also an incorrect result. In this case, the carry-out bit is 1.

SUBTRACTION RULE: To subtract one number (subtrahend) from another (minuend), take the twos complement (negation) of the subtrahend and add it to the minuend.

| | |
|--|---|
| $\begin{array}{r} 0010 = 2 \\ +1001 = -7 \\ \hline 1011 = -5 \end{array}$ <p>(a) $M = 2 = 0010$ $S = 7 = 0111$ $-S = 1001$</p> | $\begin{array}{r} 0101 = 5 \\ +1110 = -2 \\ \hline 10011 = 3 \end{array}$ <p>(b) $M = 5 = 0101$ $S = 2 = 0010$ $-S = 1110$</p> |
| $\begin{array}{r} 1011 = -5 \\ +1110 = -2 \\ \hline 11001 = -7 \end{array}$ <p>(c) $M = -5 = 1011$ $S = 2 = 0010$ $-S = 1110$</p> | $\begin{array}{r} 0101 = 5 \\ +0010 = 2 \\ \hline 0111 = 7 \end{array}$ <p>(d) $M = 5 = 0101$ $S = -2 = 1110$ $-S = 0010$</p> |
| $\begin{array}{r} 0111 = 7 \\ +0111 = 7 \\ \hline 1110 = \text{Overflow} \end{array}$ <p>(e) $M = 7 = 0111$ $S = -7 = 1001$ $-S = 0111$</p> | $\begin{array}{r} 1010 = -6 \\ +1100 = -4 \\ \hline 10110 = \text{Overflow} \end{array}$ <p>(f) $M = -6 = 1010$ $S = 4 = 0100$ $-S = 1100$</p> |

Figure 10.4 Subtraction of Numbers in Twos Complement Representation ($M - S$)

Addition and Subtraction with Signed-Magnitude Data

TABLE 10-1 Addition and Subtraction of Signed-Magnitude Numbers

| Operation | Add Magnitudes | Subtract Magnitudes | | |
|---------------|-------------------|---------------------|--------------|--------------|
| | | When $A > B$ | When $A < B$ | When $A = B$ |
| $(+A) + (+B)$ | $+(A + B)$ | | | |
| $(+A) + (-B)$ | | $+(A - B)$ | $-(B - A)$ | $+(A - B)$ |
| $(-A) + (+B)$ | | $-(A - B)$ | $+(B - A)$ | $+(A - B)$ |
| $(-A) + (-B)$ | $-(A + B)$ | | | |
| $(+A) - (+B)$ | | $+(A - B)$ | $-(B - A)$ | $+(A - B)$ |
| $(+A) - (-B)$ | $+(A + B)$ | | | |
| $(-A) - (+B)$ | $-(A + B)$ | | | |
| $(-A) - (-B)$ | | $-(A - B)$ | $+(B - A)$ | $+(A - B)$ |

Hardware Implementation

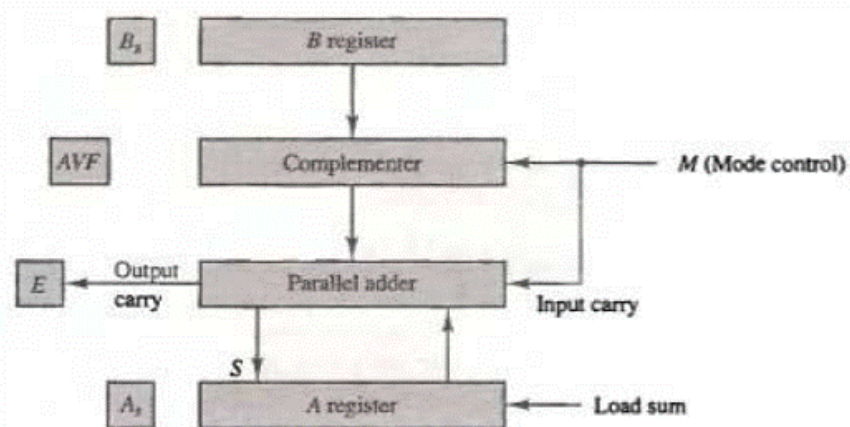


Figure 10.1 Hardware for signed-magnitude addition and subtraction.

The M signal is also applied to the input carry of the adder. When $M = 0$, the output of B is transferred to the adder, the input carry is 0, and the output of the adder is equal to the sum $A + B$.

When $M = 1$, the 1's complement of B is applied to the adder, the input carry is 1, and output $S = A + B + 1$. $= A + 2's\ c(B) = A - B$.

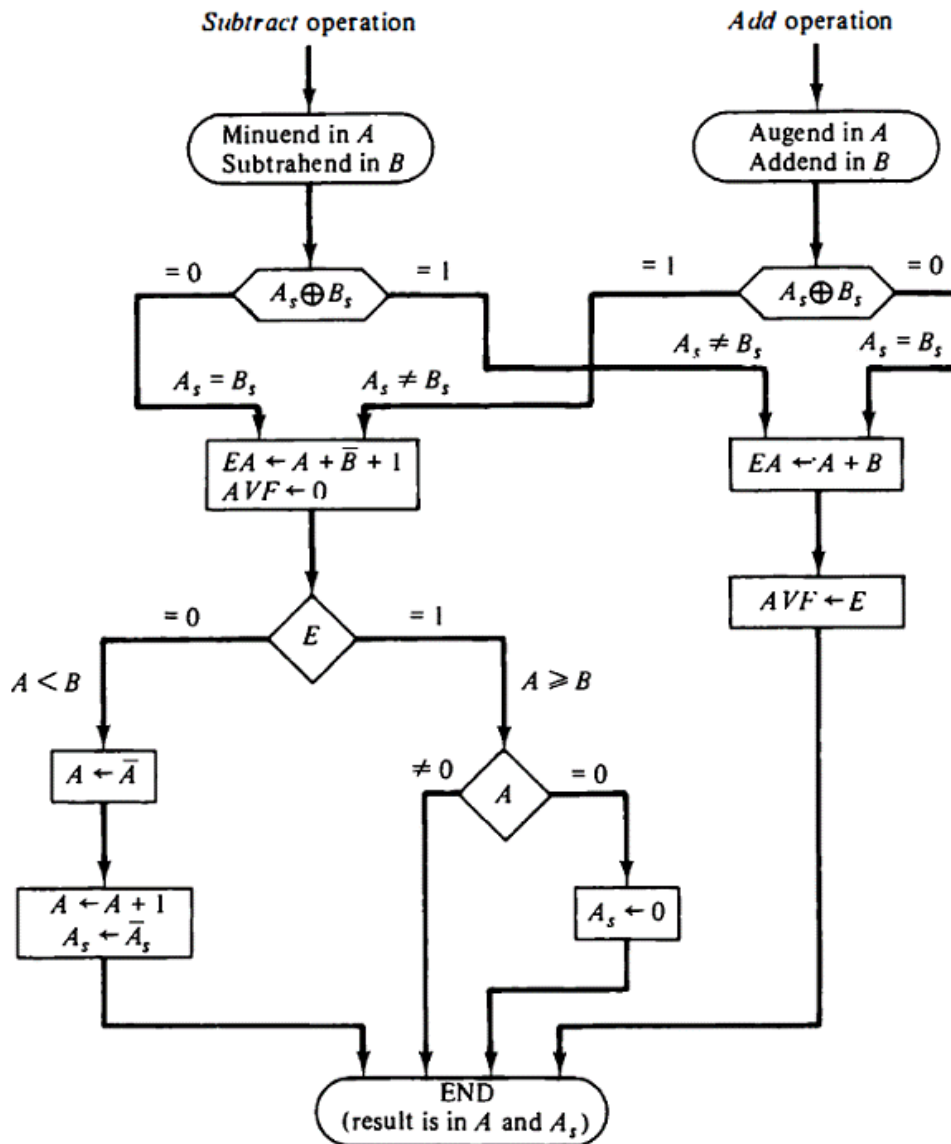
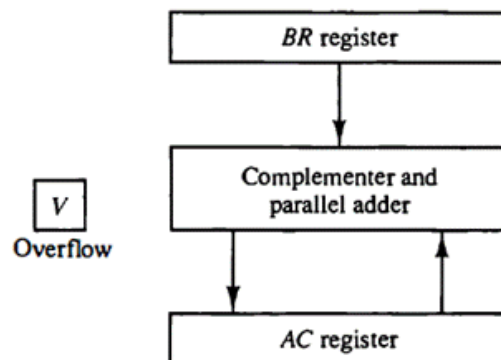


Figure 10-2 Flowchart for add and subtract operations.

Figure 10-3 Hardware for signed-2's complement addition and subtraction.



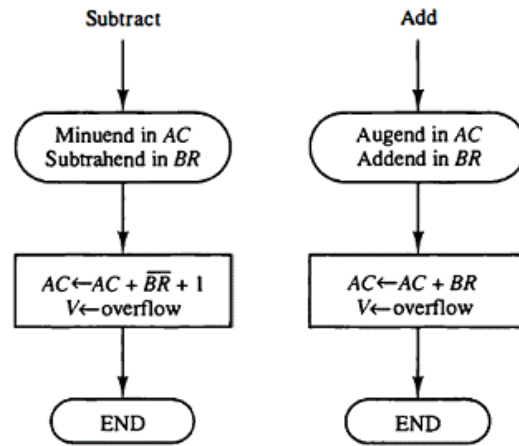


Figure 10-4 Algorithm for adding and subtracting numbers in signed-2's complement representation.