SNS COLLEGE OF TECHNOLOGY

An Autonomous Institution

Coimbatore-35

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

19ECB212 – DIGITAL SIGNAL PROCESSING

II YEAR/ IV SEMESTER

UNIT 4 – FINITE WORD LENGTH EFFECTS

TOPIC – FIXED POINT AND FLOATING POINT ARITHMETIC

RADIX NUMBER SYSTEM

- The two major methods of representing binary numbers are Fixed Point Representation and Floating Point Representation.
- In Fixed Point Representation the digits allotted for integer part and fraction part are fixed and so the position of binary point is fixed. Since the number of digits is fixed it is impossible to represent too large and too small numbers by fixed point representation.
- Therefore the range of numbers that can be represented in fixed point representation for a given binary word size is less when compared to floating point representation

RADIX NUMBER SYSTEM

- In Floating Point Representation the binary point can be shifted to desired position so that number of digits in the integer part and fraction part of the number can be varied. This leads to larger range of number that can be represented in floating point representation
- In Fixed Point Representation there are three different formats for representing negative binary fraction numbers. They are

1. Sign-Magnitude Format
2. One’s Complement Format
3. Two’s Complement Format

SIGN MAGNITUDE FORMAT

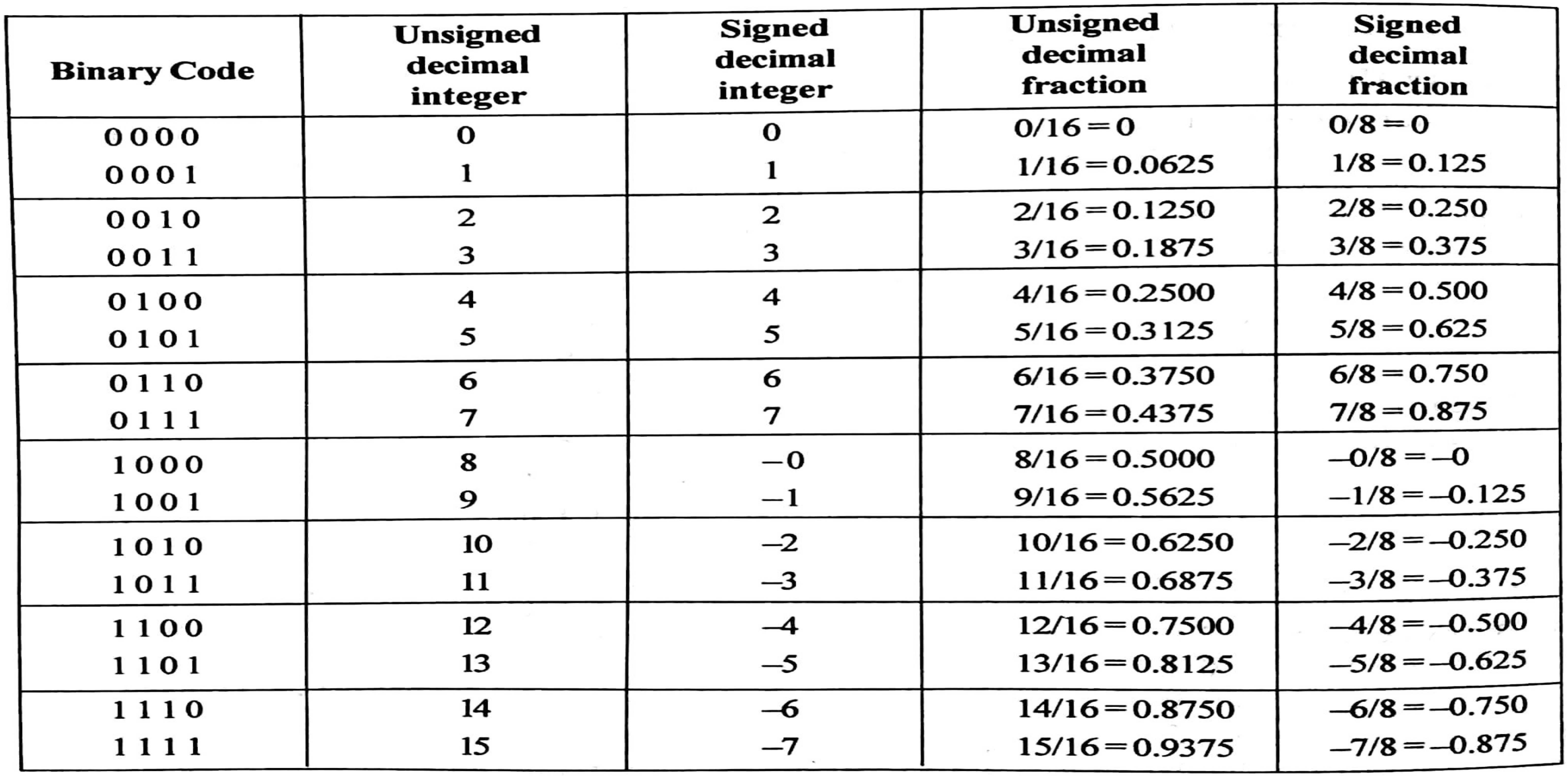
- In Sign magnitude format the negative value of a given number differ only in sign bit (i.e., digit d_0) The sign digit d_0 is zero for positive number and one for negative number

$$\therefore \text{ Negative binary fraction number, } N_N = (1 \times 2^0) + \sum_{i=1}^{B} d_i\, 2^{-i}$$

- The range of decimal fraction numbers that can be represented in B –bit fixed point Sign – magnitude format is

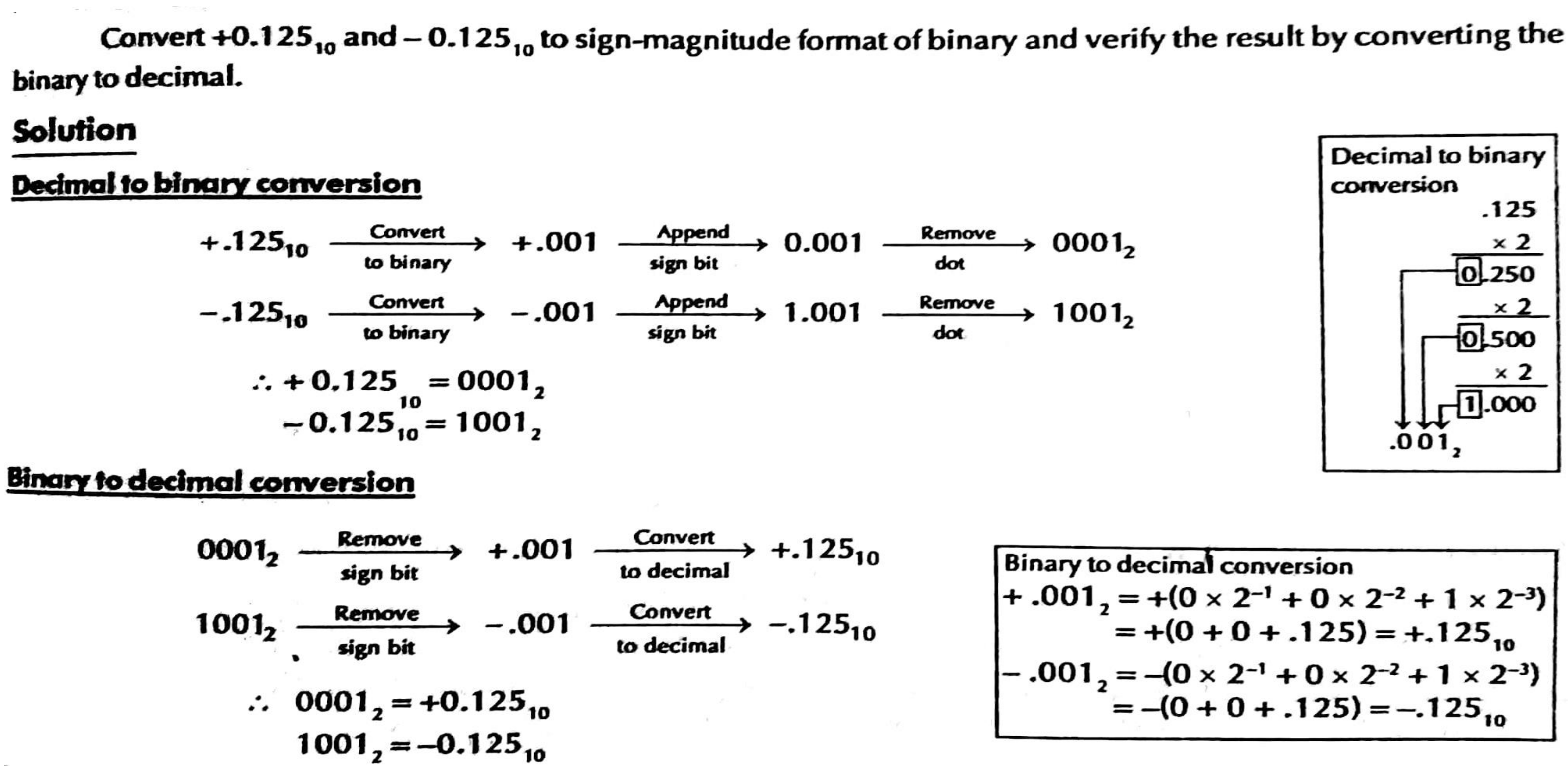
$$-\left[1-2^{-(B-1)}\right] \text{ to } +\left[1-2^{-(B-1)}\right] \quad ; \text{with step size } = \frac{1}{2^{B-1}}$$

BINARY REPRESENTATION OF DECIMAL NUMBERS



Binary Code	Unsigned decimal integer	Signed decimal integer	Unsigned decimal fraction	Signed decimal fraction
0000	0	0	0/16 = 0	0/8 = 0
0001	1	1	1/16 = 0.0625	1/8 = 0.125
0010	2	2	2/16 = 0.1250	2/8 = 0.250
0011	3	3	3/16 = 0.1875	3/8 = 0.375
0100	4	4	4/16 = 0.2500	4/8 = 0.500
0101	5	5	5/16 = 0.3125	5/8 = 0.625
0110	6	6	6/16 = 0.3750	6/8 = 0.750
0111	7	7	7/16 = 0.4375	7/8 = 0.875
1000	8	–0	8/16 = 0.5000	–0/8 = –0
1001	9	–1	9/16 = 0.5625	–1/8 = –0.125
1010	10	–2	10/16 = 0.6250	–2/8 = –0.250
1011	11	–3	11/16 = 0.6875	–3/8 = –0.375
1100	12	–4	12/16 = 0.7500	–4/8 = –0.500
1101	13	–5	13/16 = 0.8125	–5/8 = –0.625
1110	14	–6	14/16 = 0.8750	–6/8 = –0.750
1111	15	–7	15/16 = 0.9375	–7/8 = –0.875

SIGN MAGNITUDE FORMAT



Convert $+0.125_{10}$ and -0.125_{10} to sign-magnitude format of binary and verify the result by converting the binary to decimal.

Solution

Decimal to binary conversion

$$+.125_{10} \xrightarrow[\text{to binary}]{\text{Convert}} +.001 \xrightarrow[\text{sign bit}]{\text{Append}} 0.001 \xrightarrow[\text{dot}]{\text{Remove}} 0001_2$$

$$-.125_{10} \xrightarrow[\text{to binary}]{\text{Convert}} -.001 \xrightarrow[\text{sign bit}]{\text{Append}} 1.001 \xrightarrow[\text{dot}]{\text{Remove}} 1001_2$$

$$\therefore +0.125_{10} = 0001_2$$

$$-0.125_{10} = 1001_2$$

Decimal to binary conversion

$$\begin{array}{r} .125 \\ \times\ 2 \\ \hline \boxed{0}.250 \\ \times\ 2 \\ \hline \boxed{0}.500 \\ \times\ 2 \\ \hline \boxed{1}.000 \end{array}$$

$$.001_2$$

Binary to decimal conversion

$$0001_2 \xrightarrow[\text{sign bit}]{\text{Remove}} +.001 \xrightarrow[\text{to decimal}]{\text{Convert}} +.125_{10}$$

$$1001_2 \xrightarrow[\text{sign bit}]{\text{Remove}} -.001 \xrightarrow[\text{to decimal}]{\text{Convert}} -.125_{10}$$

$$\therefore\ 0001_2 = +0.125_{10}$$

$$1001_2 = -0.125_{10}$$

Binary to decimal conversion

$$+.001_2 = +(0 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3})$$
$$= +(0 + 0 + .125) = +.125_{10}$$

$$-.001_2 = -(0 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3})$$
$$= -(0 + 0 + .125) = -.125_{10}$$

ONE'S COMPLEMENT FORMAT

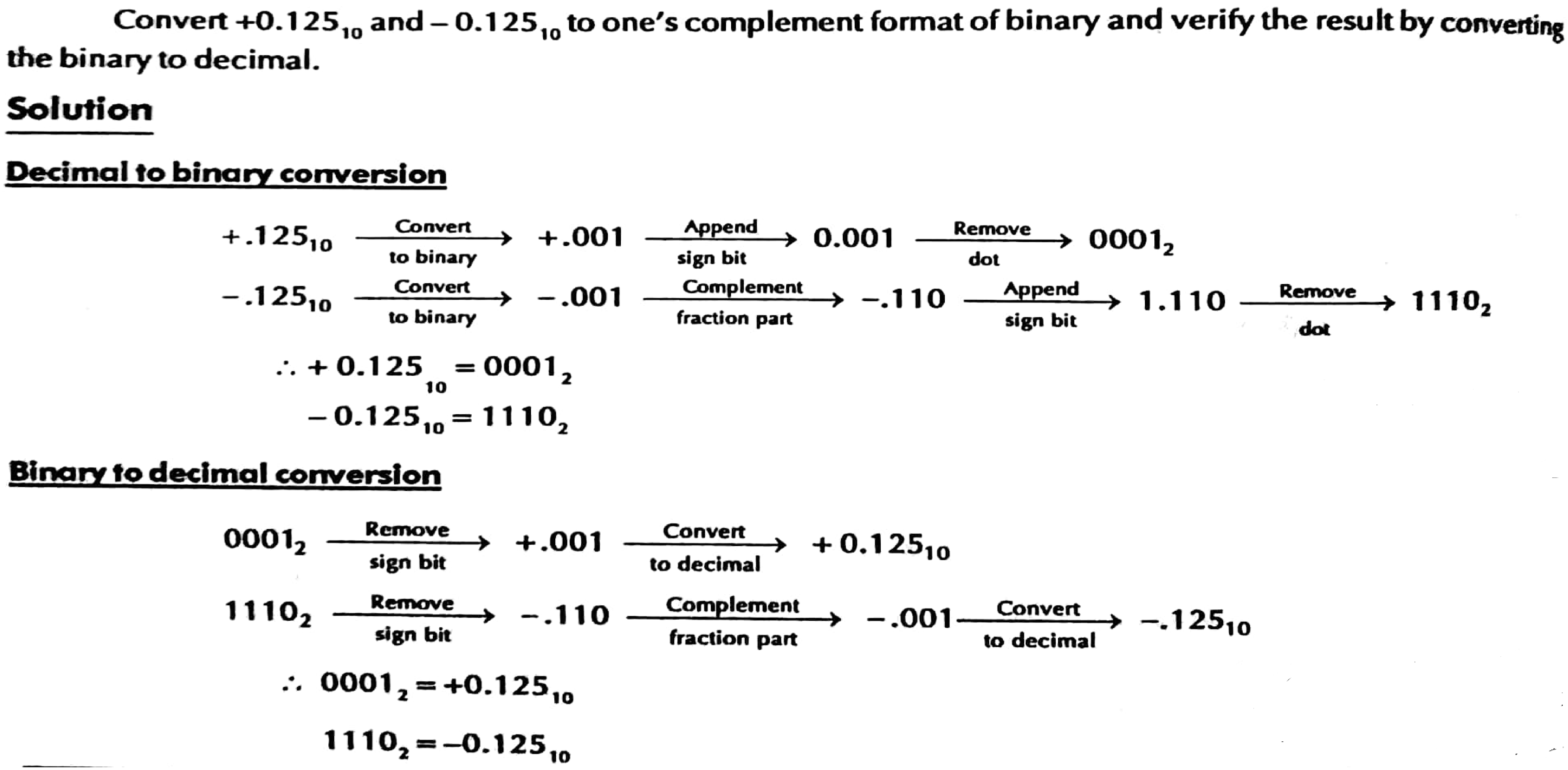
- The positive number is same in all the formats of fixed point representation and it is given by $(0 \times 2^0) + \sum_{i=1}^{B} d_i\, 2^{-i}$
- In one's complement format the negative of the given number is obtained by bit by bit complement of its positive representation. The complement of a digit d_i can be obtained by subtracting the digit from one

$$\text{Complement of } d_i = \overline{d_i} = (1 - d_i)$$

- From eqn $(0 \times 2^0) + \sum_{i=1}^{B} d_i\, 2^{-i}$ if we set the sign bit to one and replace d_i by $(1- d_i)$

$$\left.\begin{array}{l}\therefore\ \text{Negative binary fraction} \\ \quad \text{number in one's complement}\end{array}\right\} N_{1c} = (1 \times 2^0) + \sum_{i=1}^{B} (1 - d_i)2^{-i}$$

ONE'S COMPLEMENT FORMAT



Convert $+0.125_{10}$ and -0.125_{10} to one's complement format of binary and verify the result by converting the binary to decimal.

Solution

Decimal to binary conversion

$$+.125_{10} \xrightarrow[\text{to binary}]{\text{Convert}} +.001 \xrightarrow[\text{sign bit}]{\text{Append}} 0.001 \xrightarrow[\text{dot}]{\text{Remove}} 0001_2$$

$$-.125_{10} \xrightarrow[\text{to binary}]{\text{Convert}} -.001 \xrightarrow[\text{fraction part}]{\text{Complement}} -.110 \xrightarrow[\text{sign bit}]{\text{Append}} 1.110 \xrightarrow[\text{dot}]{\text{Remove}} 1110_2$$

$$\therefore +0.125_{10} = 0001_2$$

$$-0.125_{10} = 1110_2$$

Binary to decimal conversion

$$0001_2 \xrightarrow[\text{sign bit}]{\text{Remove}} +.001 \xrightarrow[\text{to decimal}]{\text{Convert}} +0.125_{10}$$

$$1110_2 \xrightarrow[\text{sign bit}]{\text{Remove}} -.110 \xrightarrow[\text{fraction part}]{\text{Complement}} -.001 \xrightarrow[\text{to decimal}]{\text{Convert}} -.125_{10}$$

$$\therefore 0001_2 = +0.125_{10}$$

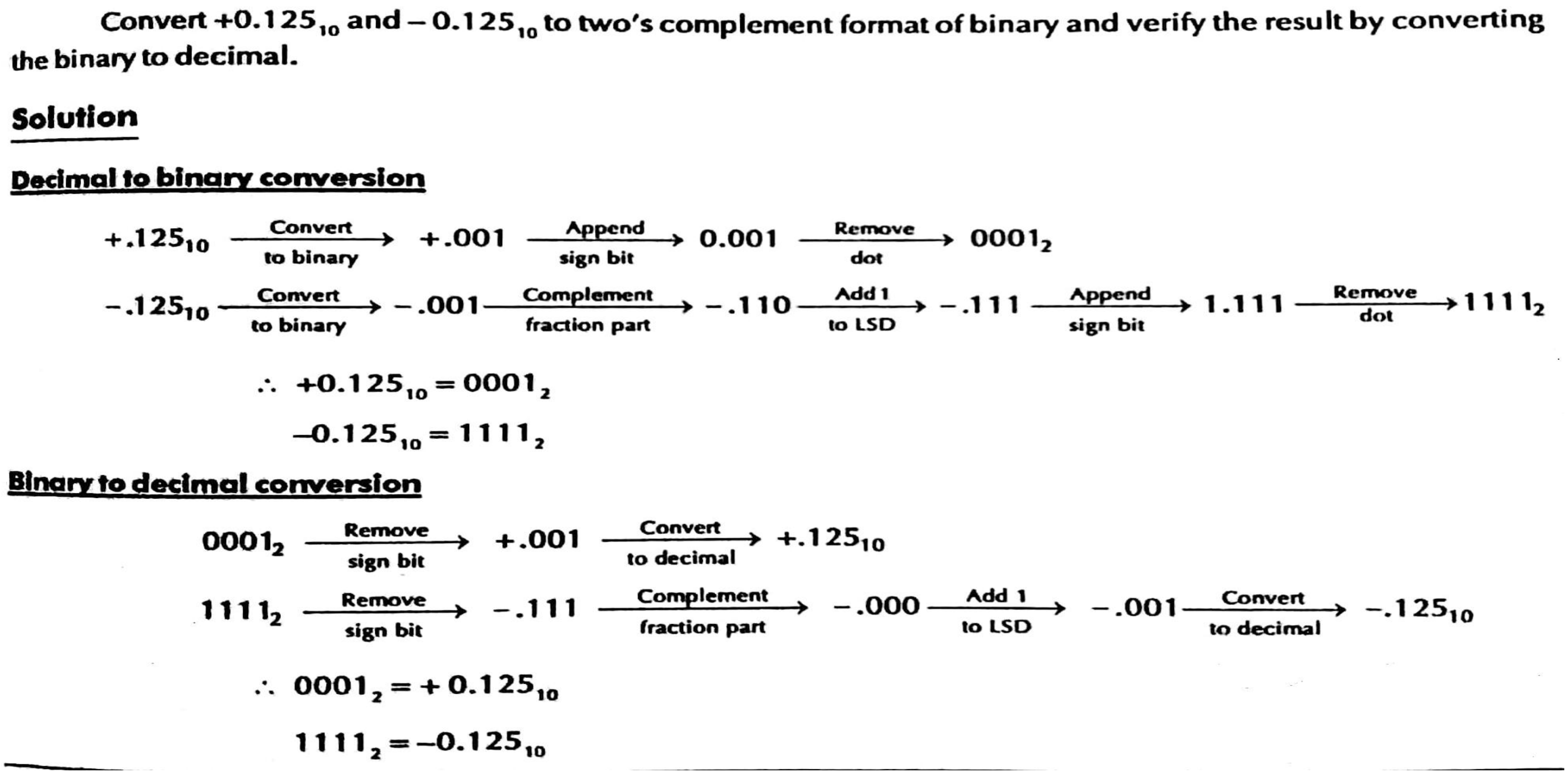
$$1110_2 = -0.125_{10}$$

TWO'S COMPLEMENT FORMAT

- The positive number is same in all the formats of fixed point representation and it is given by $(0 \times 2^0) + \sum_{i=1}^{B} d_i 2^{-i}$
- In two's complement format the negative of the given number is obtained by taking one's complement of its positive representation and then adding one to the least significant bit. If we add $1x2^{-B}$ the we get two's complement format for negative numbers

$$\therefore \left.\begin{array}{l}\text{Negative binary fraction}\\ \text{number in two's complement}\end{array}\right\} N_{2c} = (1 \times 2^0) + \sum_{i=1}^{B} (1 - d_i)2^{-i} + (1 \times 2^{-B})$$

TWO'S COMPLEMENT FORMAT



Convert $+0.125_{10}$ and -0.125_{10} to two's complement format of binary and verify the result by converting the binary to decimal.

Solution

Decimal to binary conversion

$$+.125_{10} \xrightarrow[\text{to binary}]{\text{Convert}} +.001 \xrightarrow[\text{sign bit}]{\text{Append}} 0.001 \xrightarrow[\text{dot}]{\text{Remove}} 0001_2$$

$$-.125_{10} \xrightarrow[\text{to binary}]{\text{Convert}} -.001 \xrightarrow[\text{fraction part}]{\text{Complement}} -.110 \xrightarrow[\text{to LSD}]{\text{Add 1}} -.111 \xrightarrow[\text{sign bit}]{\text{Append}} 1.111 \xrightarrow[\text{dot}]{\text{Remove}} 1111_2$$

$$\therefore \; +0.125_{10} = 0001_2$$

$$-0.125_{10} = 1111_2$$

Binary to decimal conversion

$$0001_2 \xrightarrow[\text{sign bit}]{\text{Remove}} +.001 \xrightarrow[\text{to decimal}]{\text{Convert}} +.125_{10}$$

$$1111_2 \xrightarrow[\text{sign bit}]{\text{Remove}} -.111 \xrightarrow[\text{fraction part}]{\text{Complement}} -.000 \xrightarrow[\text{to LSD}]{\text{Add 1}} -.001 \xrightarrow[\text{to decimal}]{\text{Convert}} -.125_{10}$$

$$\therefore \; 0001_2 = +0.125_{10}$$

$$1111_2 = -0.125_{10}$$

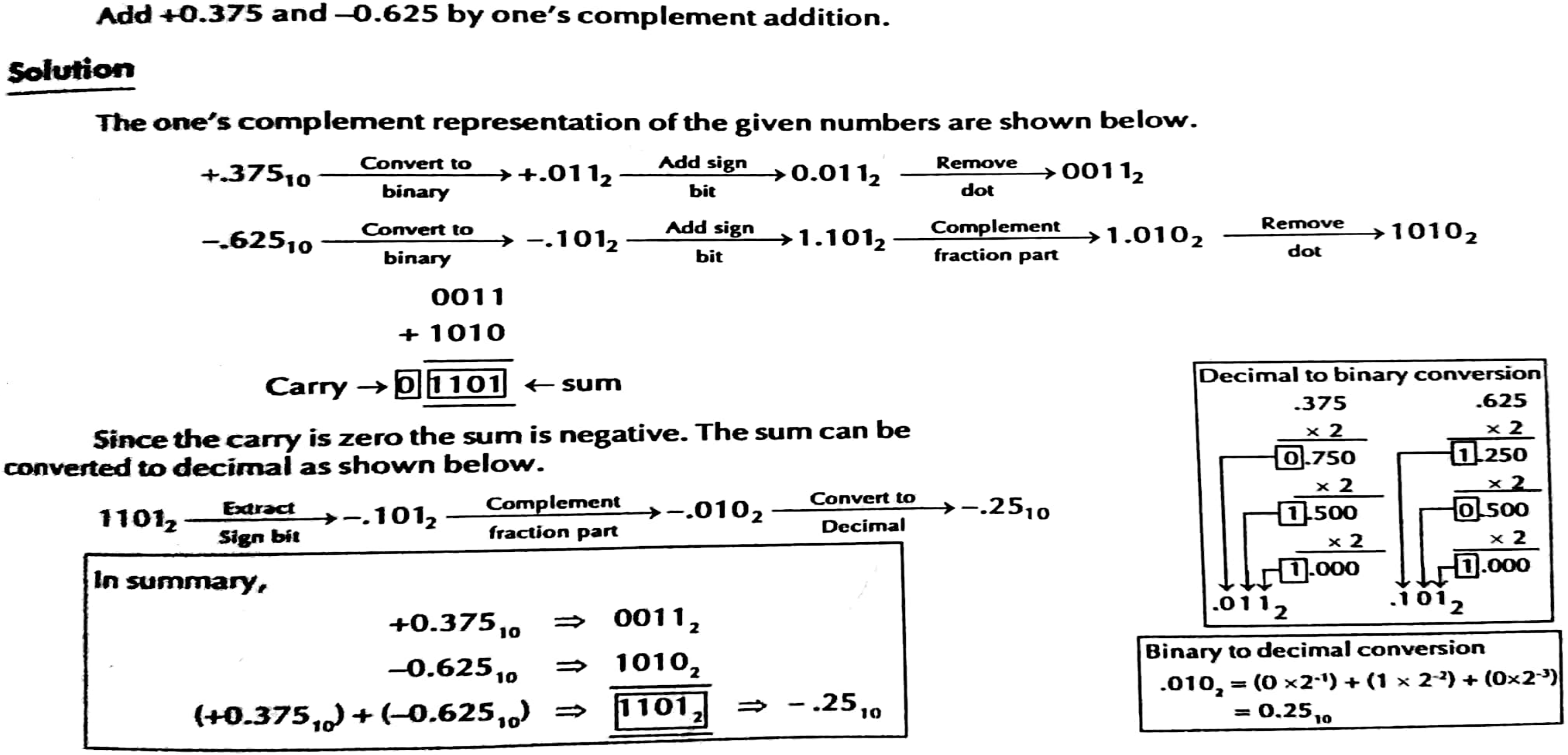
TYPES OF ARITHMETIC IN DIGITAL SYSTEMS

- The types of arithmetic in digital systems generally depends on the representation of the binary numbers. Hence the arithmetic can also be classified into two broad classes:
- Fixed Point arithmetic and Floating Point arithmetic
- The fixed point number system has three types of representation for negative numbers. They are sign-magnitude arithmetic, One’s complement arithmetic and Two’s complement arithmetic
- The sign magnitude arithmetic is generally avoided in digital systems due to inherent difficulty in handling negative numbers during additions.

ONE'S COMPLEMENT ADDITION

- In one's complement addition the numbers are represented in one's complement format and then the addition is performed
- The carry generated in addition is added to the least significant digit (LSD) to get the actual sum
- If the carry is zero after addition then the sum is negative and if the carry is one then the sum is positive
- Two examples of one's complement addition one with positive sum and the other with negative sum

ONE'S COMPLEMENT ADDITION



Add +0.375 and –0.625 by one's complement addition.

Solution

The one's complement representation of the given numbers are shown below.

$$+.375_{10} \xrightarrow[\text{binary}]{\text{Convert to}} +.011_2 \xrightarrow[\text{bit}]{\text{Add sign}} 0.011_2 \xrightarrow[\text{dot}]{\text{Remove}} 0011_2$$

$$-.625_{10} \xrightarrow[\text{binary}]{\text{Convert to}} -.101_2 \xrightarrow[\text{bit}]{\text{Add sign}} 1.101_2 \xrightarrow[\text{fraction part}]{\text{Complement}} 1.010_2 \xrightarrow[\text{dot}]{\text{Remove}} 1010_2$$

$$\begin{array}{r} 0011 \\ +\ 1010 \\ \hline \text{Carry} \rightarrow \boxed{0}\,\boxed{1101} \leftarrow \text{sum} \\ \hline \end{array}$$

Since the carry is zero the sum is negative. The sum can be converted to decimal as shown below.

$$1101_2 \xrightarrow[\text{Sign bit}]{\text{Extract}} -.101_2 \xrightarrow[\text{fraction part}]{\text{Complement}} -.010_2 \xrightarrow[\text{Decimal}]{\text{Convert to}} -.25_{10}$$

In summary,

$$+0.375_{10} \Rightarrow 0011_2$$

$$-0.625_{10} \Rightarrow \underline{1010_2}$$

$$(+0.375_{10}) + (-0.625_{10}) \Rightarrow \underline{\underline{\boxed{1101_2}}} \Rightarrow -.25_{10}$$

Decimal to binary conversion

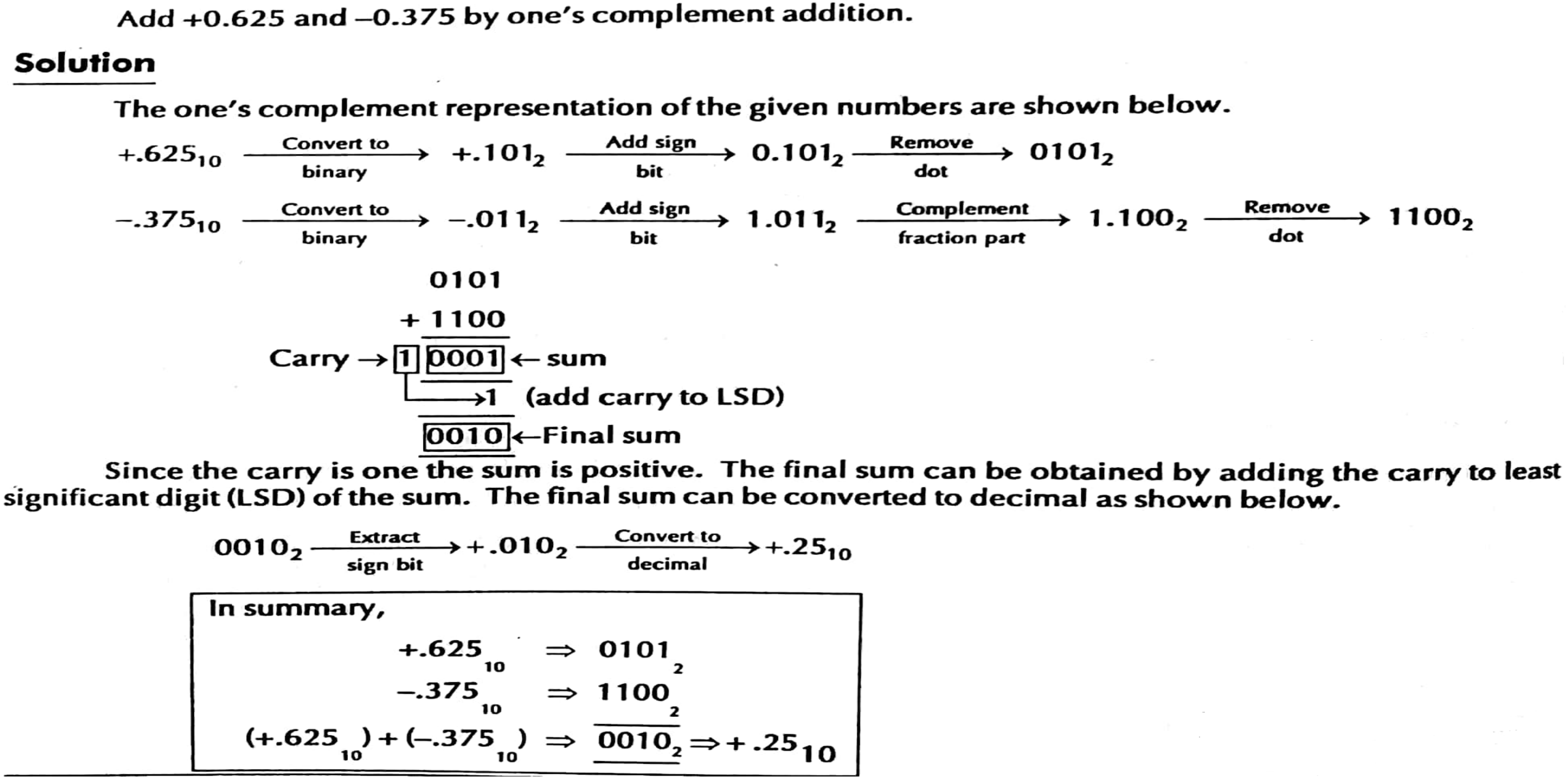
$$\begin{array}{ll} .375 & .625 \\ \times\ 2 & \times\ 2 \\ \hline \boxed{0}.750 & \boxed{1}.250 \\ \times\ 2 & \times\ 2 \\ \hline \boxed{1}.500 & \boxed{0}.500 \\ \times\ 2 & \times\ 2 \\ \hline \boxed{1}.000 & \boxed{1}.000 \\ .011_2 & .101_2 \end{array}$$

Binary to decimal conversion

$$.010_2 = (0 \times 2^{-1}) + (1 \times 2^{-2}) + (0 \times 2^{-3})$$

$$= 0.25_{10}$$

ONE'S COMPLEMENT ADDITION



Add +0.625 and −0.375 by one's complement addition.

Solution

The one's complement representation of the given numbers are shown below.

$$+.625_{10} \xrightarrow[\text{binary}]{\text{Convert to}} +.101_2 \xrightarrow[\text{bit}]{\text{Add sign}} 0.101_2 \xrightarrow[\text{dot}]{\text{Remove}} 0101_2$$

$$-.375_{10} \xrightarrow[\text{binary}]{\text{Convert to}} -.011_2 \xrightarrow[\text{bit}]{\text{Add sign}} 1.011_2 \xrightarrow[\text{fraction part}]{\text{Complement}} 1.100_2 \xrightarrow[\text{dot}]{\text{Remove}} 1100_2$$

$$\begin{array}{r} 0101 \\ +\ 1100 \\ \hline \text{Carry} \rightarrow \boxed{1}\ \boxed{0001} \leftarrow \text{sum} \\ \rightarrow 1 \quad \text{(add carry to LSD)} \\ \hline \boxed{0010} \leftarrow \text{Final sum} \end{array}$$

Since the carry is one the sum is positive. The final sum can be obtained by adding the carry to least significant digit (LSD) of the sum. The final sum can be converted to decimal as shown below.

$$0010_2 \xrightarrow[\text{sign bit}]{\text{Extract}} +.010_2 \xrightarrow[\text{decimal}]{\text{Convert to}} +.25_{10}$$

In summary,

$$+.625_{10} \Rightarrow 0101_2$$

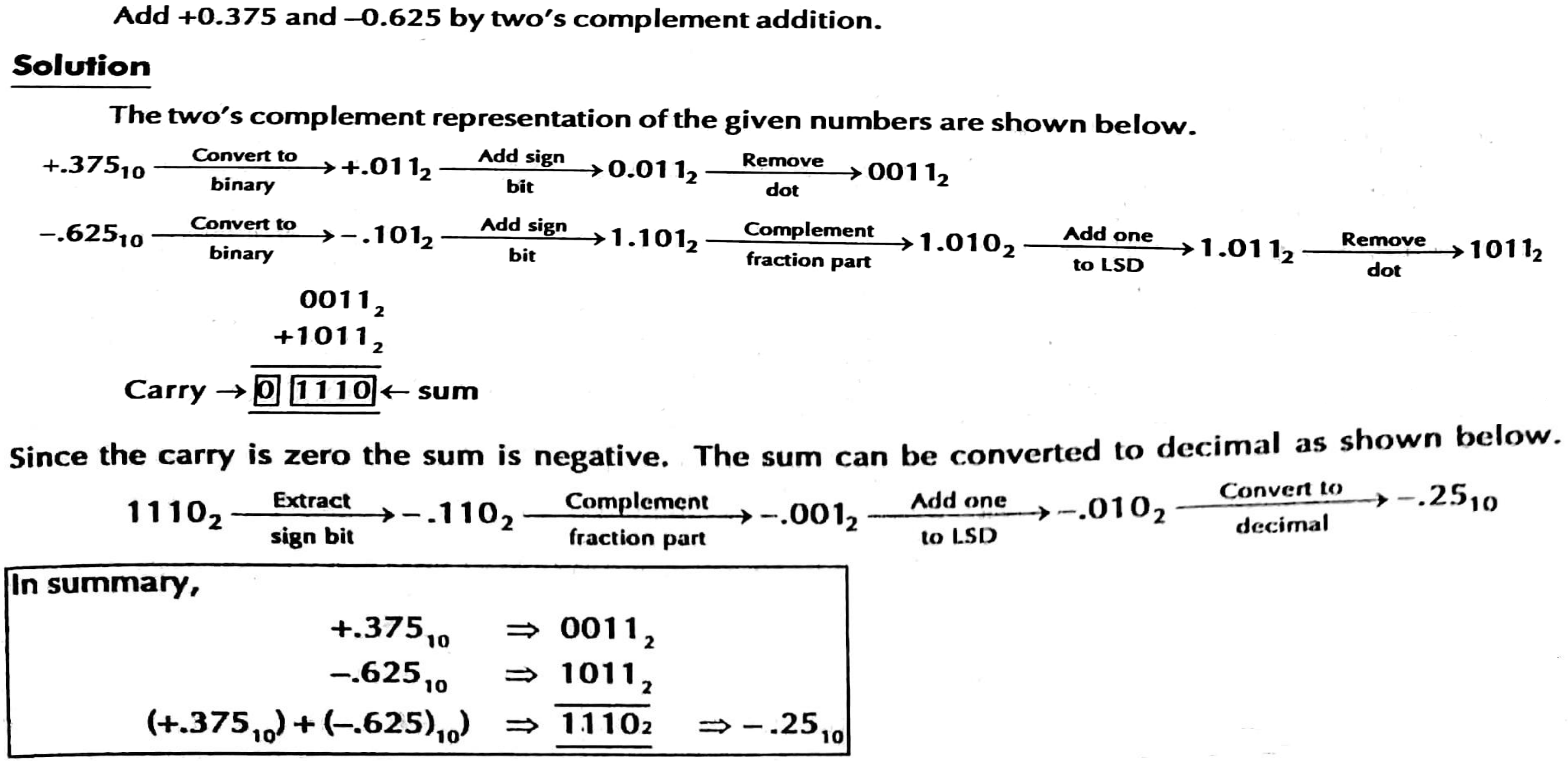
$$-.375_{10} \Rightarrow 1100_2$$

$$(+.625_{10}) + (-.375_{10}) \Rightarrow \overline{0010}_2 \Rightarrow +.25_{10}$$

TWO'S COMPLEMENT ADDITION

- In two's complement addition the numbers are represented in two's complement format and then the addition is performed
- The carry generated in addition is discarded
- If the carry is zero after addition then the sum is negative and if the carry is one then the sum is positive
- Two examples of one's complement addition one with positive sum and the other with negative sum

TWO'S COMPLEMENT ADDITION



Add +0.375 and –0.625 by two's complement addition.

Solution

The two's complement representation of the given numbers are shown below.

$$+.375_{10} \xrightarrow[\text{binary}]{\text{Convert to}} +.011_2 \xrightarrow[\text{bit}]{\text{Add sign}} 0.011_2 \xrightarrow[\text{dot}]{\text{Remove}} 0011_2$$

$$-.625_{10} \xrightarrow[\text{binary}]{\text{Convert to}} -.101_2 \xrightarrow[\text{bit}]{\text{Add sign}} 1.101_2 \xrightarrow[\text{fraction part}]{\text{Complement}} 1.010_2 \xrightarrow[\text{to LSD}]{\text{Add one}} 1.011_2 \xrightarrow[\text{dot}]{\text{Remove}} 1011_2$$

$$\begin{array}{r} 0011_2 \\ +1011_2 \\ \hline \text{Carry} \rightarrow \boxed{0}\ \boxed{1110} \leftarrow \text{sum} \\ \hline \end{array}$$

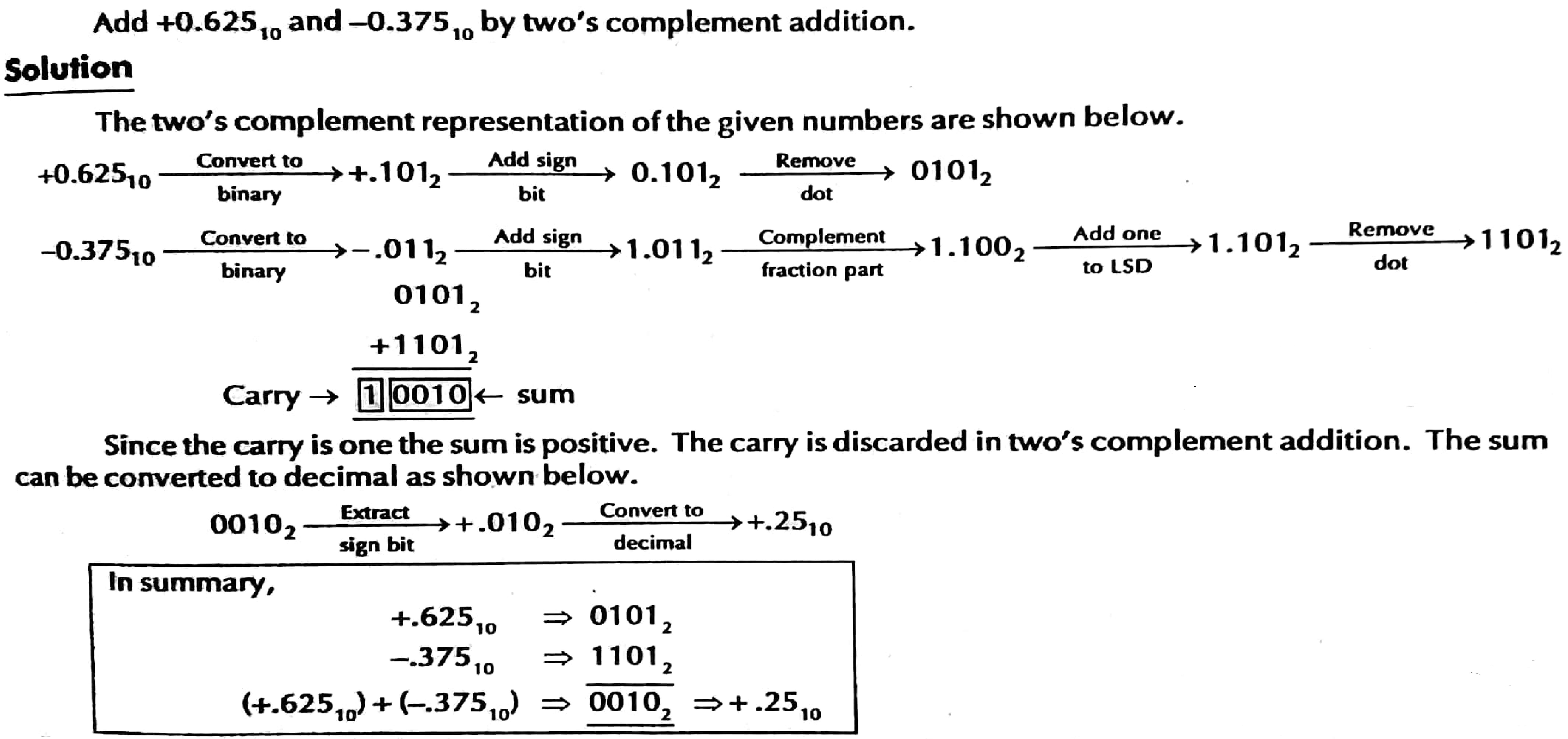
Since the carry is zero the sum is negative. The sum can be converted to decimal as shown below.

$$1110_2 \xrightarrow[\text{sign bit}]{\text{Extract}} -.110_2 \xrightarrow[\text{fraction part}]{\text{Complement}} -.001_2 \xrightarrow[\text{to LSD}]{\text{Add one}} -.010_2 \xrightarrow[\text{decimal}]{\text{Convert to}} -.25_{10}$$

In summary,

$$\begin{array}{rcll} +.375_{10} & \Rightarrow & 0011_2 & \\ -.625_{10} & \Rightarrow & 1011_2 & \\ (+.375_{10}) + (-.625)_{10}) & \Rightarrow & \overline{\underline{1110_2}} & \Rightarrow -.25_{10} \end{array}$$

TWO'S COMPLEMENT ADDITION



Add $+0.625_{10}$ and -0.375_{10} by two's complement addition.

Solution

The two's complement representation of the given numbers are shown below.

$$+0.625_{10} \xrightarrow[\text{binary}]{\text{Convert to}} +.101_2 \xrightarrow[\text{bit}]{\text{Add sign}} 0.101_2 \xrightarrow[\text{dot}]{\text{Remove}} 0101_2$$

$$-0.375_{10} \xrightarrow[\text{binary}]{\text{Convert to}} -.011_2 \xrightarrow[\text{bit}]{\text{Add sign}} 1.011_2 \xrightarrow[\text{fraction part}]{\text{Complement}} 1.100_2 \xrightarrow[\text{to LSD}]{\text{Add one}} 1.101_2 \xrightarrow[\text{dot}]{\text{Remove}} 1101_2$$

$$\begin{array}{lrl} & 0101_2 & \\ & +1101_2 & \\ \hline \text{Carry} \rightarrow & \boxed{1}\boxed{0010} & \leftarrow \text{sum} \end{array}$$

Since the carry is one the sum is positive. The carry is discarded in two's complement addition. The sum can be converted to decimal as shown below.

$$0010_2 \xrightarrow[\text{sign bit}]{\text{Extract}} +.010_2 \xrightarrow[\text{decimal}]{\text{Convert to}} +.25_{10}$$

In summary,

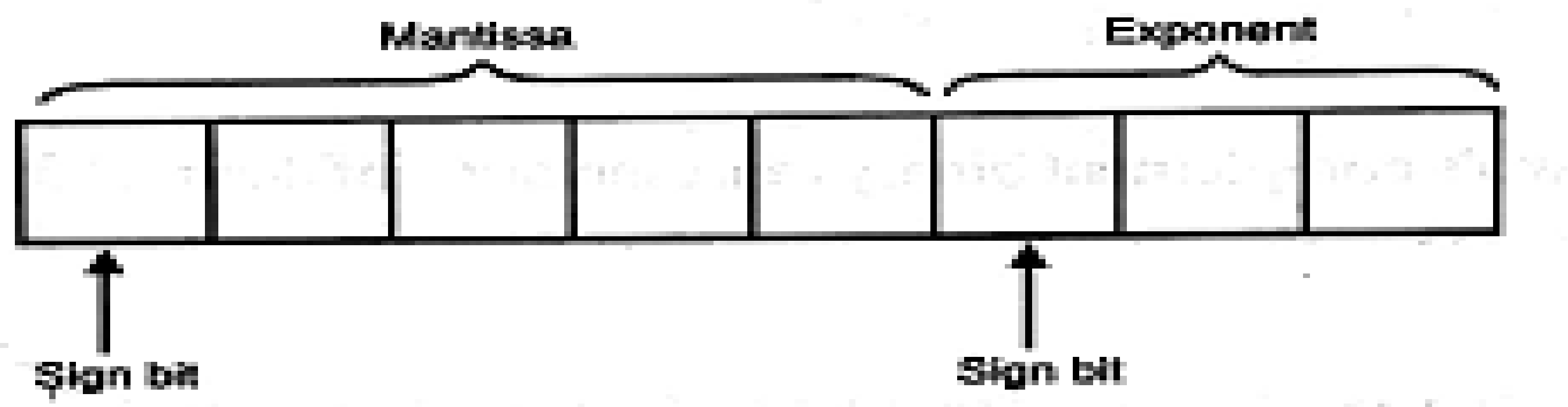
$$\begin{array}{rcll} +.625_{10} & \Rightarrow & 0101_2 & \\ -.375_{10} & \Rightarrow & 1101_2 & \\ \hline (+.625_{10}) + (-.375_{10}) & \Rightarrow & \underline{0010_2} & \Rightarrow +.25_{10} \end{array}$$

FLOATING POINT REPRESENTATION

- The floating point representation is employed to represent larger range of numbers in a given binary word size. The floating point number is represented as
- Floating Point Number, $\mathbf{N_r = M \times 2^E}$
- M is called Mantissa and it will be in binary fraction format. The value of M will be in the range 0≤M≤1
- E is called Exponent and it is either a positive or negative integer
- In floating point representation both mantissa and exponent uses one bit for representing sign. Usually the leftmost bit in mantissa and exponent is used to represent the sign

FLOATING POINT REPRESENTATION

- “1” is the leftmost bit position represents negative sign and “0” in the leftmost bit position represents positive sign
- The floating point representation is explained by considering a five bit mantissa and three bit exponent with a total size of eight bits
- In mantissa the leftmost bit is used to represent the sign and other four bits are used to represent a binary fraction number
- In exponent the leftmost bit is used to represent the sign and other two bits are used to represent a binary integer number



FLOATING POINT ADDITION

- Floating point addition are represented in the desired floating point format
- The addition can be performed only when the exponents of both the numbers are equal
- Hence the exponent of the smaller number is changed to equal the exponent of the larger number and then addition is performed
- In floating point addition if the sum is in the unnormalized form then it has to be normalized to represent in proper (Correct) floating point format

FLOATING POINT MULTIPLICATION

- In Floating point multiplication the numbers are represented in the desired floating point format
- The product is obtained by multiplying the mantissa and adding the exponents
- The sign bits of mantissa should be added separately to determine the sign of product of mantissa
- In floating point multiplication if the product is in the unnormalized form then it has to be normalized to represent in proper (Correct) floating point format

COMPARISON OF FIXED POINT AND FLOATING POINT REPRESENTATION

S.No.	Fixed Point Representation	Floating Point Representation
1	In a b-bit binary the range of numbers represented is less when compared to floating point representation	In a b-bit binary the range of numbers represented is large when compared to fixed point representation
2	The position of binary point is fixed	The position of binary point is variable
3	The resolution is uniform throughout the range	The resolution is variable

COMPARISON OF FIXED POINT AND FLOATING POINT ARITHMETIC

S.No.	Fixed Point Arithmetic	Floating Point Arithmetic
1	The accuracy of the result is less due to smaller dynamic range	The accuracy of the result will be higher due to larger dynamic range
2	Speed of processing is high	Speed of processing is low
3	Hardware implementation is cheaper	Hardware implementation is costlier
4	Fixed point arithmetic can be used for real time computations	Floating point arithmetic cannot be used for real time computations
5	Quantization error occurs only in multiplication	Quantization error occurs in both multiplication and addition

ASSESSMENT

1. List the two methods of representing binary numbers.
2. Mention the three different formats for representing negative binary fraction numbers.
3. Define Floating Point representation.
4. Compare fixed point and floating point arithmetic
5. The fixed point number system has three types of representation for negative numbers. They are --------------, -------------- and ------------
6. Compare fixed point and floating point representation

THANK YOU