Logic Design

Tutorial 2

Amr Keleg

Faculty of Engineering, Ain Shams University

August 10, 2021

Contact: amr_mohamed@live.com

Outline

1 Conversion of a decimal floating value to any numbering system
- Sheet 1 - Question 13

2 Two's complement

3 BCD and other representations

4 Unsigned binary multiplication

5 Binary and Gray code

Outline

Q13) Do the following conversion problems:
b- Calculate the binary equivalent of 2/3 out to eight places. Then convert from binary to decimal. How close is the result to 2/3?

Q13) Do the following conversion problems:
b- Calculate the binary equivalent of 2/3 out to eight places. Then convert from binary to decimal. How close is the result to 2/3?

Remember that to represent 2/3 as a decimal value, we need infinite number of digits (i.e: 0.666666666...6666666...)
The fact of needing infinite number of digits is also present in different bases other than decimal.

Finding the binary equivalent of $2/3$ is a similar idea to how we convert integer decimal values to any base.

Value	Value * base	Value * base $>=$ 1?
$\frac{2}{3}$	$1\ \frac{1}{3}$	1

Finding the binary equivalent of $2/3$ is a similar idea to how we convert integer decimal values to any base.

Value	Value * base	Value * base $>=$ 1?
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0

Finding the binary equivalent of 2/3 is a similar idea to how we convert integer decimal values to any base.

Value	Value * base	Value * base >= 1?
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0
$\frac{2}{3}$	$1\ \frac{1}{3}$	1

Finding the binary equivalent of $2/3$ is a similar idea to how we convert integer decimal values to any base.

Value	Value * base	Value * base $>=$ 1?
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0

Finding the binary equivalent of $2/3$ is a similar idea to how we convert integer decimal values to any base.

Value	Value * base	Value * base $>=$ 1?
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0
$\frac{2}{3}$	$1\ \frac{1}{3}$	1

Finding the binary equivalent of $2/3$ is a similar idea to how we convert integer decimal values to any base.

Value	Value * base	Value * base >= 1?
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0
$\frac{2}{3}$	$1\ \frac{1}{3}$	1

Note: Here we write the values from top to bottom and not from bottom to top.

$(\frac{2}{3})_{10} = (0.10\ 10\ 10\ 10\ \ldots)_2$

Finding the binary equivalent of $2/3$ is a similar idea to how we convert integer decimal values to any base.

Value	Value * base	Value * base >= 1?
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0
$\frac{2}{3}$	$1\ \frac{1}{3}$	1
$\frac{1}{3}$	$\frac{2}{3}$	0
$\frac{2}{3}$	$1\ \frac{1}{3}$	1

Note: Here we write the values from top to bottom and not from bottom to top.
$(\frac{2}{3})_{10} = (0.10\ 10\ 10\ 10\ \ldots)\ _2$
By taking only the first 8 places: $0.10101010\ _2 = 0.6640625_{10}$

Q13) Do the following conversion problems:
a- Convert decimal 27.315 to binary.

$27_{10} = 11011_2$ (As in tutorial 1)
$0.315_{10} = ??$

Value	Value * base	Value * base >= 1?
0.315	0.630	0

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1
0.260	0.520	0

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1
0.260	0.520	0
0.520	1.040	1

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1
0.260	0.520	0
0.520	1.040	1
0.040	0.080	0

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1
0.260	0.520	0
0.520	1.040	1
0.040	0.080	0
0.080	0.160	0

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1
0.260	0.520	0
0.520	1.040	1
0.040	0.080	0
0.080	0.160	0
0.160	0.320	0

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1
0.260	0.520	0
0.520	1.040	1
0.040	0.080	0
0.080	0.160	0
0.160	0.320	0
0.320	0.640	0

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1
0.260	0.520	0
0.520	1.040	1
0.040	0.080	0
0.080	0.160	0
0.160	0.320	0
0.320	0.640	0
0.640	1.280	1

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1
0.260	0.520	0
0.520	1.040	1
0.040	0.080	0
0.080	0.160	0
0.160	0.320	0
0.320	0.640	0
0.640	1.280	1
0.280	0.560	0

Value	Value * base	Value * base >= 1?
0.315	0.630	0
0.630	1.260	1
0.260	0.520	0
0.520	1.040	1
0.040	0.080	0
0.080	0.160	0
0.160	0.320	0
0.320	0.640	0
0.640	1.280	1
0.280	0.560	0
0.560	1.120	1

Value	Value * base	Value * base >= 1?
0.560	1.120	1
0.120	0.240	0
0.240	0.480	0
0.480	0.960	0
0.960	1.920	1
0.920	1.840	1
0.840	1.680	1
0.680	1.360	1
0.360	0.720	0
0.720	1.440	1
0.440	0.880	0
0.880	1.760	1

Value	Value * base	Value * base >= 1?
0.880	1.760	1
0.760	1.520	1
0.520	1.040	1

c) How to convert to hexadecimal first?

c) How to convert to hexadecimal first?

Value	Value * base	Value * base >= 1?
0.315	5.040	5

c) How to convert to hexadecimal first?

Value	Value * base	Value * base >= 1?
0.315	5.040	5
0.040	0.640	0

c) How to convert to hexadecimal first?

Value	Value * base	Value * base >= 1?
0.315	5.040	5
0.040	0.640	0
0.640	10.240	10 (A)

c) How to convert to hexadecimal first?

Value	Value * base	Value * base >= 1?
0.315	5.040	5
0.040	0.640	0
0.640	10.240	10 (A)
0.240	3.840	3

c) How to convert to hexadecimal first?

Value	Value * base	Value * base >= 1?
0.315	5.040	5
0.040	0.640	0
0.640	10.240	10 (A)
0.240	3.840	3
0.840	13.440	13 (D)

c) How to convert to hexadecimal first?

Value	Value * base	Value * base >= 1?
0.315	5.040	5
0.040	0.640	0
0.640	10.240	10 (A)
0.240	3.840	3
0.840	13.440	13 (D)
0.440	7.040	7

c) How to convert to hexadecimal first?

Value	Value * base	Value * base >= 1?
0.315	5.040	5
0.040	0.640	0
0.640	10.240	10 (A)
0.240	3.840	3
0.840	13.440	13 (D)
0.440	7.040	7
0.040	0.640	0

c) How to convert to hexadecimal first?

Value	Value * base	Value * base >= 1?
0.315	5.040	5
0.040	0.640	0
0.640	10.240	10 (A)
0.240	3.840	3
0.840	13.440	13 (D)
0.440	7.040	7
0.040	0.640	0

$$(0.315)_{10} = (0.5\ 0A3D7\ 0A3D7\ 0A3D7\ 0A3D7\ ....)_{16}$$

Outline

Outline

How to perform this operation: $99_{10} + 99_{10}$?

$$
\begin{array}{ccc}
\hline
9 & 9 & \\
 & & + \\
9 & 9 & \\
\hline
\end{array}
$$

$$
\begin{array}{cccc}
1 & 1 & & \\
\hline
 & 9 & 9 & \\
 & & & + \\
 & 9 & 9 & \\
\hline
1 & 9 & 8 &
\end{array}
$$

$$\begin{array}{ccc} \hline 1 & 1 & \\ & & + \\ 1 & 1 & \\ \hline \end{array}$$

$$\begin{array}{cccc} 1 & 1 & & \\ \hline & 1 & 1 & \\ & & & + \\ & 1 & 1 & \\ \hline 1 & 1 & 0 & \end{array}$$

Outline

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)		

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)		

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)		

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)	0010,0101 (37)	0010,0110 (38)

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)	0010,0101 (37)	0010,0110 (38)
d) 1010,1010 (170)		

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)	0010,0101 (37)	0010,0110 (38)
d) 1010,1010 (170)	0101,0101 (85)	0101,0110 (86)

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)	0010,0101 (37)	0010,0110 (38)
d) 1010,1010 (170)	0101,0101 (85)	0101,0110 (86)
e) 1000,0101 (133)		

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)	0010,0101 (37)	0010,0110 (38)
d) 1010,1010 (170)	0101,0101 (85)	0101,0110 (86)
e) 1000,0101 (133)	0111,1010 (122)	0111,1011 (123)

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)	0010,0101 (37)	0010,0110 (38)
d) 1010,1010 (170)	0101,0101 (85)	0101,0110 (86)
e) 1000,0101 (133)	0111,1010 (122)	0111,1011 (123)
f) 1111,1111 (255)		

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)	0010,0101 (37)	0010,0110 (38)
d) 1010,1010 (170)	0101,0101 (85)	0101,0110 (86)
e) 1000,0101 (133)	0111,1010 (122)	0111,1011 (123)
f) 1111,1111 (255)	0000,0000 (0)	0000,0001 (1)

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)	0010,0101 (37)	0010,0110 (38)
d) 1010,1010 (170)	0101,0101 (85)	0101,0110 (86)
e) 1000,0101 (133)	0111,1010 (122)	0111,1011 (123)
f) 1111,1111 (255)	0000,0000 (0)	0000,0001 (1)

- An 8-bit number $+$ its 1's complement $=$ 255

Q14) Obtain the 1s and 2s complements of the following binary numbers and find the corresponding decimal equivalent.

Number	1's complement	2's complement
a) 0001,0000 (16)	1110,1111 (239)	1111,0000 (240)
b) 0000,0000 (0)	1111,1111 (255)	1,0000,0000 (256)
c) 1101,1010 (218)	0010,0101 (37)	0010,0110 (38)
d) 1010,1010 (170)	0101,0101 (85)	0101,0110 (86)
e) 1000,0101 (133)	0111,1010 (122)	0111,1011 (123)
f) 1111,1111 (255)	0000,0000 (0)	0000,0001 (1)

- An 8-bit number + its 1's complement = 255
- An 8-bit number + its 2's complement = 256

Outline

Q16) Perform subtraction on the given **unsigned** binary numbers using the 2s complement of the subtrahend. Where the result should be negative, find its 2s complement and affix a minus sign.

For two **unsigned** values, to subtract B from A (A-B):

- Make sure the two values are represented in the same number of bits and pad with zeros if needed.
- Find the two's complement of B
- Add A and the two's complement of B
- In the case of an overflow bit, ignore it.
- In the case of no overflow, compute the two's complement of the result.

a) 10011 - 10010

a) 10011 - 10010

$A = 10011$
$B = 10010$

a) 10011 - 10010

$A = 10011$
$B = 10010$
Two's complement of $B =$ One's complement of $B + 1$
$= 01101 + 1 = 01110$

a) 10011 - 10010

$A = 10011$
$B = 10010$
Two's complement of B = One's complement of $B + 1$
$= 01101 + 1 = 01110$

$A - B = A +$ Two's complement of $B = 10011 + 01110$

$$
\begin{array}{cccccc c}
1 & 1 & 1 & 1 & & & \\
\hline
 & 1 & 0 & 0 & 1 & 1 & \\
 & & & & & & + \\
 & 0 & 1 & 1 & 1 & 0 & \\
\hline
\mathbf{1} & 0 & 0 & 0 & 0 & 1 &
\end{array}
$$

$$
\begin{array}{cccccc c}
1 & 1 & 1 & 1 & & & \\
\hline
 & 1 & 0 & 0 & 1 & 1 & \\
 & & & & & & + \\
 & 0 & 1 & 1 & 1 & 0 & \\
\hline
\mathbf{1} & 0 & 0 & 0 & 0 & 1 &
\end{array}
$$

Since there is an overflow, (The two numbers were represented in 5 bits but we needed a 6th bit), then:

- Ignore the overflow
- The result of the subtraction is the value after ignoring the overflow
- 10011 - 10010 = 00001

c) 1001 - 110101

c) 1001 - 110101

$A = \mathbf{00}1001$
$B = 110101$

c) 1001 - 110101

$A = \mathbf{00}1001$
$B = 110101$
Two's complement of $B =$ One's complement of $B + 1$
$= 001010 + 1 = 001011$

c) 1001 - 110101

A = **00**1001
B = 110101
Two's complement of B = One's complement of B + 1
= 001010 + 1 = 001011

A - B = A + Two's complement of B = 001001 + 001011

$$
\begin{array}{cccccccc}
 & & 1 & & 1 & 1 & & \\
\hline
 & 0 & 0 & 1 & 0 & 0 & 1 & \\
 & & & & & & & + \\
 & 0 & 0 & 1 & 0 & 1 & 1 & \\
\hline
\mathbf{0} & 0 & 1 & 0 & 1 & 0 & 0 &
\end{array}
$$

$$
\begin{array}{ccccccccc}
 & & 1 & & 1 & 1 & & \\
\hline
 & 0 & 0 & 1 & 0 & 0 & 1 & \\
 & & & & & & & + \\
 & 0 & 0 & 1 & 0 & 1 & 1 & \\
\hline
\mathbf{0} & 0 & 1 & 0 & 1 & 0 & 0 &
\end{array}
$$

Since there is no overflow, (The two numbers were represented in 6 bits and we didn't need a 7th bit), then:

- Find the two's complement of the result without the overflow bit
- Add a negative sign to the two's complement
- 001001 - 110101 = (-) Two's complement (010100) = (-) (101011 + 1) = - 101100

Outline

Assume we have 4 bits to represent numbers on our system.

- We can represent 16 unsigned values [0, 15].

Assume we have 4 bits to represent numbers on our system.

- We can represent 16 unsigned values [0, 15].
- What if we want to represent negative values as well?

Assume we have 4 bits to represent numbers on our system.

- We can represent 16 unsigned values [0, 15].
- What if we want to represent negative values as well?

Table 1.3
Signed Binary Numbers

Decimal	Signed Magnitude
+7	0111
+6	0110
+5	0101
+4	0100
+3	0011
+2	0010
+1	0001
+0	0000
−0	1000
−1	1001
−2	1010
−3	1011
−4	1100
−5	1101
−6	1110
−7	1111
−8	—

Table 1.3
Signed Binary Numbers

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

How to add two numbers in signed two's complement representation?
Example:
$(+7) + (+7) = ??$

How to add two numbers in signed two's complement representation?

Example:

$(+7) + (+7) = ??$

$(0111) + (0111) =$

How to add two numbers in signed two's complement representation?

Example:

$(+7) + (+7) = ??$

$(0111) + (0111) = (1110)$ which is

How to add two numbers in signed two's complement representation?

Example:

$(+7) + (+7) = ??$

$(0111) + (0111) = (1110)$ which is -2 !!

How to add two numbers in signed two's complement representation?
Example:
$(+7) + (+7) = ??$
$(0111) + (0111) = (1110)$ which is -2 !!
Therefore we need to have another bit to handle this overflow:

How to add two numbers in signed two's complement representation?
Example:
$(+7) + (+7) = ??$
$(0111) + (0111) = (1110)$ which is -2 !!
Therefore we need to have another bit to handle this overflow:$(\mathbf{0}0111) + (\mathbf{0}0111) = \mathbf{0}1110$

How to add two numbers in signed two's complement representation?
Example:
$(+7) + (+7) = ??$
$(0111) + (0111) = (1110)$ which is -2 !!
Therefore we need to have another bit to handle this overflow:$(\mathbf{0}0111) + (\mathbf{0}0111) = \mathbf{0}1110$

- The result is positive since the sign bit is equal to zero
- The value is $8 + 4 + 2 = 14$ (Makes sense now)

What about (-7) + (-7)?

What about (-7) + (-7)?

+7 = (00111)
-7 = (11001)

What about (-7) + (-7)?

+7 = (00111)
-7 = (11001)

(11001) + (11001) = 1**1**0010 (Ignore the overflow).

The result is 10010.

What about (-7) + (-7)?

+7 = (00111)
-7 = (11001)

(11001) + (11001) = **1**10010 (Ignore the overflow).

The result is 10010.

- The result is negative since the sign bit is equal to one
- Its two's complement is 01110
- The value is - (8 + 4 + 2) = -14 (Makes sense now)

Outline

Q17) Convert decimal +49 and +29 to binary, using the signed 2's complement representation and enough digits to accommodate the numbers. Then perform the binary equivalent of (+29) + (-49), (-29) + (+49), and (-29) + (-49). Convert the answers back to decimal and verify that they are correct.

Q17) Convert decimal +49 and +29 to binary, using the signed 2's complement representation and enough digits to accommodate the numbers. Then perform the binary equivalent of (+29) + (-49), (-29) + (+49), and (-29) + (-49). Convert the answers back to decimal and verify that they are correct.

- $49_{10} = 110001_2$
- $29_{10} = \mathbf{0}11101_2$

Q17) Convert decimal +49 and +29 to binary, using the signed 2's complement representation and enough digits to accommodate the numbers. Then perform the binary equivalent of (+29) + (-49), (-29) + (+49), and (-29) + (-49). Convert the answers back to decimal and verify that they are correct.

- $49_{10} = 110001_2$
- $29_{10} = \mathbf{0}11101_2$
- Add two bits, one for sign and one for overflow:
- $49_{10} = \mathbf{00}110001_2$
- $29_{10} = \mathbf{00}011101_2$

Q17) Convert decimal +49 and +29 to binary, using the signed 2's complement representation and enough digits to accommodate the numbers. Then perform the binary equivalent of (+29) + (-49), (-29) + (+49), and (-29) + (-49). Convert the answers back to decimal and verify that they are correct.

- $49_{10} = 110001_2$
- $29_{10} = \mathbf{0}11101_2$
- Add two bits, one for sign and one for overflow:
- $49_{10} = \mathbf{00}110001_2$
- $29_{10} = \mathbf{00}011101_2$
- $-49_{10} = \mathbf{11}001111_2$
- $-29_{10} = \mathbf{11}100011_2$

A] $(+29) + (\text{-}49) = 0001{,}1101 + 1100{,}1111$

A] $(+29) + (\text{-}49) = 0001{,}1101 + 1100{,}1111 = 0{,}1110{,}1100$

A] (+29) + (-49) = 0001,1101 + 1100,1111 = 0,1110,1100

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **1**110,1100.

A] (+29) + (-49) = 0001,1101 + 1100,1111 = 0,1110,1100

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **1**110,1100.
- This is a negative value (Sign bit = 1).

A] $(+29) + (-49) = 0001,1101 + 1100,1111 = 0,1110,1100$

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **1**110,1100.
- This is a negative value (Sign bit $= 1$).
- Its magnitude is 0001,0100 which is $16 + 4 = 20$

A] (+29) + (-49) = 0001,1101 + 1100,1111 = 0,1110,1100

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **1**110,1100.
- This is a negative value (Sign bit = 1).
- Its magnitude is 0001,0100 which is 16 + 4 = 20
- The result is $(-20)_{10}$

B] (-29) + (49) = 1110,0011 + 0011,0001

B] (-29) + (49) = 1110,0011 + 0011,0001 = 1,0001,0100

B] (-29) + (49) = 1110,0011 + 0011,0001 = 1,0001,0100

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **0**001,0100.

B] (-29) + (49) = 1110,0011 + 0011,0001 = 1,0001,0100

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **0**001,0100.
- This is a positive value (Sign bit = 0).

B] (-29) + (49) = 1110,0011 + 0011,0001 = 1,0001,0100

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **0**001,0100.
- This is a positive value (Sign bit = 0).
- Its magnitude is 0001,0100 which is 16 + 4 = 20

B] (-29) + (49) = 1110,0011 + 0011,0001 = 1,0001,0100

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **0**001,0100.
- This is a positive value (Sign bit = 0).
- Its magnitude is 0001,0100 which is 16 + 4 = 20
- The result is $(+20)_{10}$

C] (-29) + (-49) = 1110,0011 + 1100,1111

C] (-29) $+$ (-49) $=$ 1110,0011 $+$ 1100,1111 $=$ 1,1011,0010

C] (-29) + (-49) = 1110,0011 + 1100,1111 = 1,1011,0010

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **1**011,0010.

C] (-29) + (-49) = 1110,0011 + 1100,1111 = 1,1011,0010

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **1**011,0010.
- This is a negative value (Sign bit = 1).

C] (-29) + (-49) = 1110,0011 + 1100,1111 = 1,1011,0010

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **1**011,0010.
- This is a negative value (Sign bit = 1).
- Its magnitude is 0100,1110 which is 64 + 8 + 4 + 2 = 78

C] (-29) + (-49) = 1110,0011 + 1100,1111 = 1,1011,0010

- ALWAYS IGNORE ANY BITS OTHER THAN THE bits that we need (8 bits in this case).
- Result is **1**011,0010.
- This is a negative value (Sign bit = 1).
- Its magnitude is 0100,1110 which is 64 + 8 + 4 + 2 = 78
- The result is $(\text{-}78)_{10}$

Outline

Outline

Q18) Represent the decimal number 6,248 in:

- a) BCD

[1] https://en.wikipedia.org/wiki/Aiken_code

Q18) Represent the decimal number 6,248 in:

- a) BCD: Represent each decimal **digit** as a 4-bit binary number.
  $6{,}248_{10} = 0110,\ 0010,\ 0100,\ 1000_{BCD}$
- b) excess3 code

[1] https://en.wikipedia.org/wiki/Aiken_code

Q18) Represent the decimal number 6,248 in:

- a) BCD: Represent each decimal **digit** as a 4-bit binary number.
  $6{,}248_{10} = 0110,\ 0010,\ 0100,\ 1000_{\text{BCD}}$
- b) excess3 code: Similar to BCD but add three to each digit before converting to binary
  $6{,}248_{10} = 1001,\ 0101,\ 0111,\ 1011_{\text{Excess-3 code}}$
- c) 2421 code[1]

[1] https://en.wikipedia.org/wiki/Aiken_code

Q18) Represent the decimal number 6,248 in:

- a) BCD: Represent each decimal **digit** as a 4-bit binary number.
  $6{,}248_{10} = 0110,\ 0010,\ 0100,\ 1000_{\text{BCD}}$
- b) excess3 code: Similar to BCD but add three to each digit before converting to binary
  $6{,}248_{10} = 1001,\ 0101,\ 0111,\ 1011_{\text{Excess-3 code}}$
- c) 2421 code[1]: Instead of considering the weights of the 4 bits to be 8, 4, 2, 1 make them 2, 4, 2, 1
  $6{,}248_{10} = 1100,\ 0010,\ 0100,\ 1110_{\text{2421 code}}$
- d) 6311 code

[1] https://en.wikipedia.org/wiki/Aiken_code

Q18) Represent the decimal number 6,248 in:

- a) BCD: Represent each decimal **digit** as a 4-bit binary number.
  $6{,}248_{10} = 0110,\ 0010,\ 0100,\ 1000_{\mathrm{BCD}}$
- b) excess3 code: Similar to BCD but add three to each digit before converting to binary
  $6{,}248_{10} = 1001,\ 0101,\ 0111,\ 1011_{\text{Excess-3 code}}$
- c) 2421 code[1]: Instead of considering the weights of the 4 bits to be 8, 4, 2, 1 make them 2, 4, 2, 1
  $6{,}248_{10} = 1100,\ 0010,\ 0100,\ 1110_{\text{2421 code}}$
- d) 6311 code: Instead of considering the weights of the 4 bits to be 8, 4, 2, 1 make them 6, 3, 1, 1
  $6{,}248_{10} = 1000,\ 0011,\ 0101,\ 1011_{\text{6311 code}}$

[1] https://en.wikipedia.org/wiki/Aiken_code

Outline

Q1) Convert the following numbers into BCD and hence carry out the BCD addition:

- b) $(398)_{10} + (198)_{10}$

Q1) Convert the following numbers into BCD and hence carry out the BCD addition:

- b) $(398)_{10} + (198)_{10}$
- $(398)_{10} = (0011,\ 1001,\ 1000)_{\mathrm{BCD}}$
- $(198)_{10} = (0001,\ 1001,\ 1000)_{\mathrm{BCD}}$

$$
\begin{array}{rrr}
0011 & 1001 & 1000 \\
 & & + \\
0001 & 1001 & 1000 \\
\hline
\end{array}
$$

0011	1001	1000
		+
0001	1001	1000
		10000

(result is more than 9, Add 6 to the value)

$$
\begin{array}{rrr}
0011 & 1001 & 1000 \\
 & & + \\
0001 & 1001 & 1000 \\
\hline
 & & 10000
\end{array}
$$

(result is more than 9, Add 6 to the value)

$$
\begin{array}{r}
+ \\
0110 \\
\hline
\end{array}
$$

0011	1001	1000
		+
0001	1001	1000
		10000

(result is more than 9, Add 6 to the value)

	+
	0110
1	0110

$$
\begin{array}{rrr}
 & 1 & \\
0011 & 1001 & 1000 \\
 & & + \\
0001 & 1001 & 1000 \\
\hline
\end{array}
$$

$$
\begin{array}{rrr}
 & 1 & \\
0011 & 1001 & 1000 \\
 & & + \\
0001 & 1001 & 1000 \\
\hline
 & 10011 & 0110
\end{array}
$$

(result is more than 9, Add 6 to the value)

		1	
	0011	1001	1000
			+
	0001	1001	1000
		10011	0110
(result is more than 9, Add 6 to the value)			
			+
		0110	

$$
\begin{array}{rrr}
 & 1 & \\
0011 & 1001 & 1000 \\
 & & + \\
0001 & 1001 & 1000 \\
\hline
 & 10011 & 0110
\end{array}
$$

(result is more than 9, Add 6 to the value)

$$
\begin{array}{rrr}
 & & + \\
 & 0110 & \\
\hline
1 & 1001 & 0110
\end{array}
$$

$$
\begin{array}{rrr}
 \overset{1}{0011} & 1001 & 1000 \\
 & & + \\
 0001 & 1001 & 1000 \\
\hline
\end{array}
$$

$$
\begin{array}{rrr}
1 & & \\
0011 & 1001 & 1000 \\
 & & + \\
0001 & 1001 & 1000 \\
\hline
0101 & 1001 & 0110
\end{array}
$$

(result is less than 9, Do thing)

$$
\begin{array}{rrr}
1 & & \\
0011 & 1001 & 1000 \\
 & & + \\
0001 & 1001 & 1000 \\
\hline
0101 & 1001 & 0110 \\
\end{array}
$$

(result is less than 9, Do thing)

$$
\begin{array}{rrr}
\hline
0101 & 1001 & 0110 \\
\end{array}
$$

Outline

Outline

How to perform the following **UNSIGNED binary** multiplications:

- $10011_2 * 1_2 =$

How to perform the following **UNSIGNED binary** multiplications:

- $10011_2 * 1_2 = 10011_2$

How to perform the following **UNSIGNED binary** multiplications:

- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 =$

How to perform the following **UNSIGNED binary** multiplications:

- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 = 10011\mathbf{0}_2$
  Explanation:
  $10011_2 = (2^4) + (2^1) + (2^0)$
  $10_2 = (2^1)$

How to perform the following **UNSIGNED binary** multiplications:

- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 = 10011\mathbf{0}_2$
  Explanation:
  $10011_2 = (2^4) + (2^1) + (2^0)$
  $10_2 = (2^1)$
  $10011_2 * 10_2 = ((2^4) + (2^1) + (2^0)) * (2^1) = 2^5 + 2^2 + 2^1$
- $10011_2 * 100_2 =$

How to perform the following **UNSIGNED binary** multiplications:

- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 = 10011\mathbf{0}_2$
  Explanation:
  $10011_2 = (2^4) + (2^1) + (2^0)$
  $10_2 = (2^1)$
  $10011_2 * 10_2 = ((2^4) + (2^1) + (2^0)) * (2^1) = 2^5 + 2^2 + 2^1$
- $10011_2 * 100_2 = 10011\mathbf{00}_2$

- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 = 100110_2$
- $10011_2 * 100_2 = 1001100_2$

- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 = 100110_2$
- $10011_2 * 100_2 = 1001100_2$
- $10011_2 * 111_2 =$

- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 = 100110_2$
- $10011_2 * 100_2 = 1001100_2$
- $10011_2 * 111_2 =$
  $10011_2 * 1_2 + 10011_2 * 10_2 + 10011_2 * 100_2$

- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 = 100110_2$
- $10011_2 * 100_2 = 1001100_2$
- $10011_2 * 111_2 =$
  $10011_2 * 1_2 + 10011_2 * 10_2 + 10011_2 * 100_2 = 10011 + 100110 + 1001100 = 1000,0101_2$

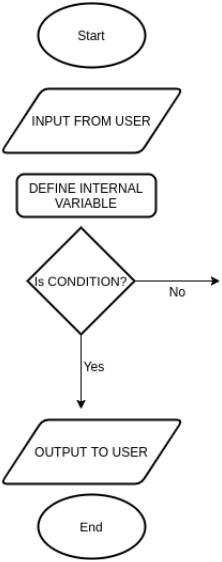
- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 = 100110_2$
- $10011_2 * 100_2 = 1001100_2$
- $10011_2 * 111_2 = 10011_2 * 1_2 + 10011_2 * 10_2 + 10011_2 * 100_2 = 10011 + 100110 + 1001100 = 1000,0101_2$
- Note: Convert the values to decimal and check that the result is correct.

- $10011_2 * 1_2 = 10011_2$
- $10011_2 * 10_2 = 100110_2$
- $10011_2 * 100_2 = 1001100_2$
- $10011_2 * 111_2 =$
  $10011_2 * 1_2 + 10011_2 * 10_2 + 10011_2 * 100_2 = 10011 + 100110 + 1001100 = 1000,0101_2$
- Note: Convert the values to decimal and check that the result is correct.

Thus, binary multiplication depends on two operations:
Shift-left (multiplication by 2) and Binary-addition.

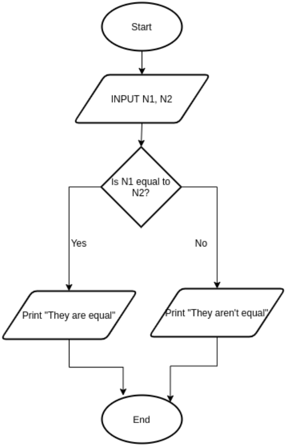
Outline

The main blocks are:



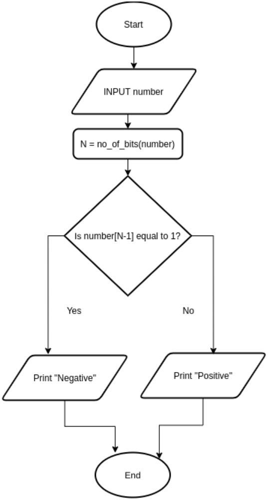
Example 1: Allow the user to input two values and check whether they are equal to not.

Example 1: Allow the user to input two values and check whether they are equal to not.



Example 2: Allow the user to input a **Signed 2's complement binary number** and check whether it's positive or negative.

Example 2: Allow the user to input a **Signed 2's complement binary number** and check whether it's positive or negative.



Outline

Q4) Write an algorithm for unsigned binary numbers multiplication. You can draw a flow chart.

Q4) Write an algorithm for unsigned binary numbers multiplication. You can draw a flow chart.

Note: For binary non-floating numbers:

- A shift left operation adds a zero to the right of the number (e.g: 101 becomes 1010).

Q4) Write an algorithm for unsigned binary numbers multiplication. You can draw a flow chart.

Note: For binary non-floating numbers:

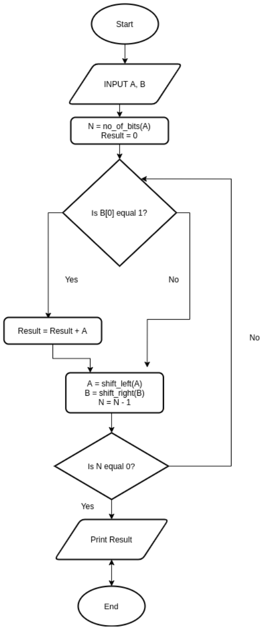
- A shift left operation adds a zero to the right of the number (e.g: 101 becomes 1010).
- A shift right operation drops the least significant bit (e.g: 101 becomes 10).

Q4) Write an algorithm for unsigned binary numbers multiplication. You can draw a flow chart.

Note: For binary non-floating numbers:

- A shift left operation adds a zero to the right of the number (e.g: 101 becomes 1010).
- A shift right operation drops the least significant bit (e.g: 101 becomes 10).

Note: Assume that we want to compute (A * B) where A, B are two binary numbers of the equal length.


Start
INPUT A, B
N = no_of_bits(A)
Result = 0
Is B[0] equal 1?
Yes
No
Result = Result + A
No
A = shift_left(A)
B = shift_right(B)
N = N - 1
Is N equal 0?
Yes
Print Result
End

Outline

Outline

Table 1.6
Gray Code

Gray Code	Decimal Equivalent
0000	0
0001	1
0011	2
0010	3
0110	4
0111	5
0101	6
0100	7
1100	8
1101	9
1111	10
1110	11
1010	12
1011	13
1001	14
1000	15

https://en.wikipedia.org/wiki/Gray_code