Unit 5: Data Representation

Prashant Gautam
M.Sc. CSIT

Binary, Octal & Hexadecimal Number Systems and their conversions

Number system

- ***A number system defines how a number can be represented using distinct symbols.***

- A number can be represented differently in different systems.

- For example, the two numbers $(2A)_{16}$ and $(52)_8$ both refer to the same quantity, $(42)_{10}$, but their representations are different.

Number system can be categorized as

1. Decimal number system
2. Binary number system
3. Octal number system
4. Hexadecimal Number System

- Each number system is associated with a base or radix
- The decimal number system is said to be of base or radix10
- A number in *base r* contains r digits 0,1,2,...,r-1
- Decimal (Base 10): 0,1,2,3,4,5,6,7,8,9

System	Base	Symbols	Used by humans?	Used in computers?
Decimal	10	0, 1, … 9	Yes	No
Binary	2	0, 1	No	Yes
Octal	8	0, 1, … 7	No	No
Hexa-decimal	16	0, 1, … 9, A, B, … F	No	No

The decimal system (base 10)

- The word **decimal** is derived from the Latin root **decem**(ten). In this system the **base b = 10** and we use ten symbols.

S = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}.

Binary system (base 2)

- The word binary is derived from the Latin root **bini** (or two by two).

- In this system the **base b = 2** and we use only two symbols,

  S = {0, 1}

- The symbols in this system are often referred to as **binary digits** or **bits.**

The hexadecimal system (base 16)

- The word **hexadecimal** is derived from the Greek root hex (six) and the Latin root **decem** (ten).
- In this system the **base b = 16** and we use sixteen symbols to represent a number.
- The set of symbols is

S = **{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}**

- The symbols A, B, C, D, E, F are equivalent to 10, 11, 12, 13, 14, and 15 respectively.
- The symbols in this system are often referred to as **hexadecimal digits.**

The octal system (base 8)

- The word octal is derived from the Latin root octo (eight).
- In this system the base $b = 8$ and we use eight symbols to represent a number.
- The set of symbols is:

$S = \{0, 1, 2, 3, 4, 5, 6, 7\}$

Common Number Systems

System	Base	Symbols	Used by Humans?	Used in Computers?

Quantities / Counting

Decimal	Binary	Octal	Hexa-decimal

Decimal	Binary	Octal	Hexa-decimal

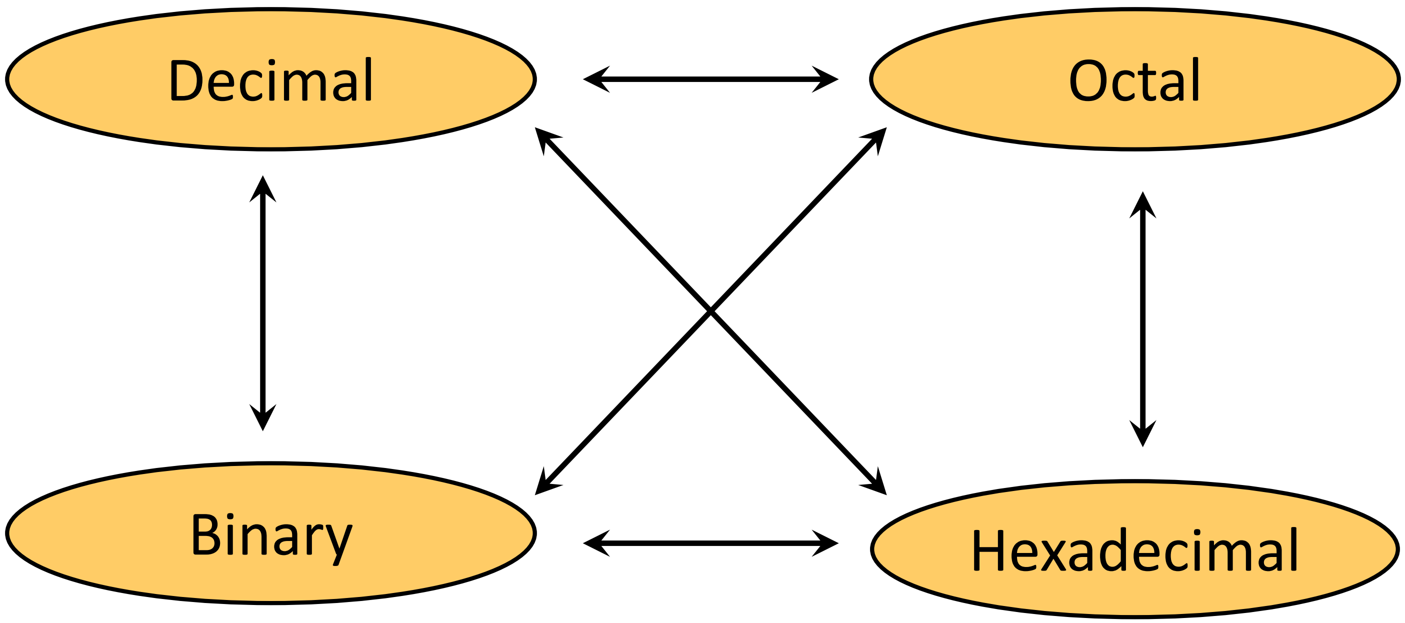
Quantities / Counting

Decimal	Binary	Octal	Hexa-decimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7

Decimal	Binary	Octal	Hexa-decimal
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Conversion among Bases

- Possibilities

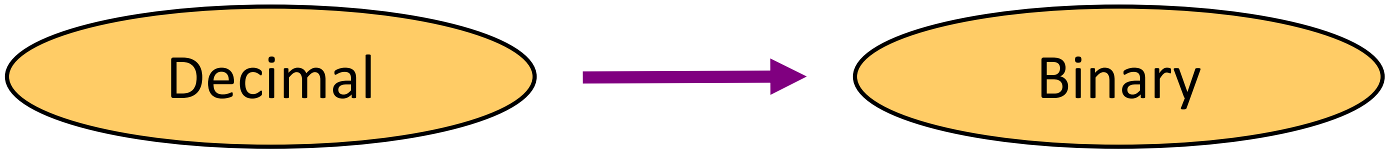


- Example

$$25_{10} = 11001_2 = 31_8 = 19_{16}$$

Base

Decimal to Binary



- Technique
  - Divide by two, keep track of the remainder
  - First remainder is bit 0 (LSB, least-significant bit)
  - Second remainder is bit 1 and so on

Example (Decimal to Binary)

$125_{10} = ?_2$

2	125	1
2	62	0
2	31	1
2	15	1
2	7	1
2	3	1
2	1	1
	0	

$125_{10} = 1111101_2$

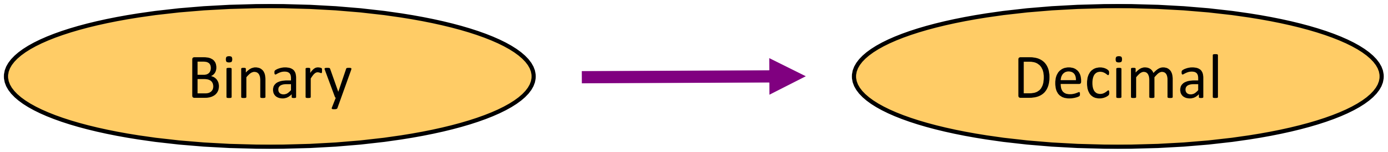
Example (Decimal to Binary)

$0.6875_{10} = ?_2$

		integer		*fraction*
$0.6875 \times 2 =$	1.3750	1	+	0.3750
$0.3750 \times 2 =$	0.7500	0	+	0.7500
$0.7500 \times 2 =$	1.5000	1	+	0.5000
$0.5000 \times 2 =$	1.0000	1	+	0.0000

$0.6875_{10} = 0.1011_2$

Binary to Decimal



- Technique
  - Multiply each bit by 2^n, where n is the “weight” of the bit
  - The weight is the position of the bit, starting from 0 on the right
  - Add the results

Example (Binary to Decimal)

1 0 1 0 1 1

$$1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$32 + 0 + 8 + 0 + 2 + 1$$

$$43_{10}$$

$$101011_2 = 43_{10}$$

Example (Binary to Decimal)

$$1 \quad 1 \quad . \quad 1 \quad 1$$

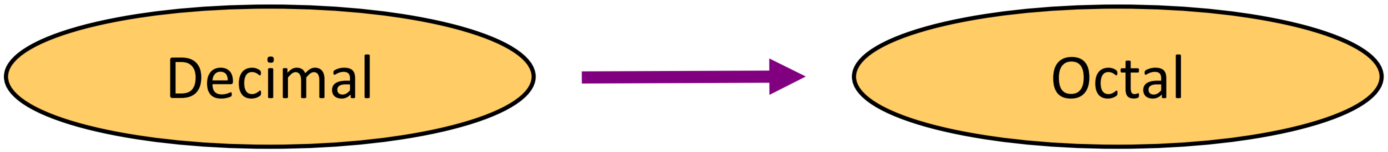
$$1 \times 2^{1} \quad + \quad 1 \times 2^{0} \quad + \quad 1 \times 2^{-1} \quad + \quad 1 \times 2^{-2}$$

$$2 \quad + \quad 1 \quad + \quad 0.5 \quad + \quad 0.25$$

$$3.75_{10}$$

$$11.11_{2} = 3.75_{10}$$

Decimal to Octal



- Technique
  - Divide by eight
  - Keep track of the remainder

Example (Decimal to Octal)

$125_{10} = ?_8$

8	125	5
8	15	7
8	1	1
	0	

$125_{10} = 175_8$

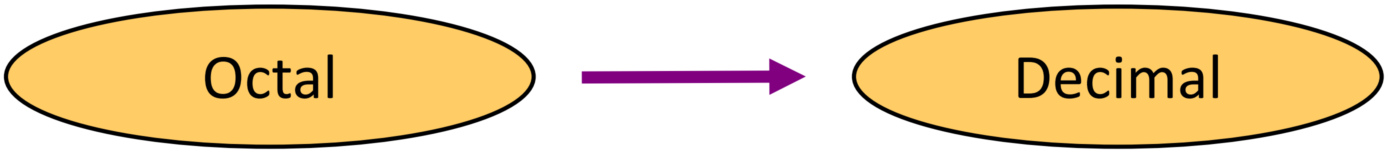
Example (Decimal to Octal)

$0.6875_{10} = ?_8$

		integer		*fraction*
$0.6875 \times 8 =$	5.5000	5	+	0.5000
$0.5000 \times 8 =$	4.0000	4	+	0.0000

$0.6875_{10} = 0.54_8$

Octal to Decimal



- Technique
  - Multiply each bit by 8^n, where *n* is the "weight" of the bit
  - The weight is the position of the bit, starting from 0 on the right
  - Add the results

Example (Octal to Decimal)

7 2 4

$$7 \times 8^2 + 2 \times 8^1 + 4 \times 8^0$$

$$448 + 16 + 4$$

$$468_{10}$$

$$724_8 = 468_{10}$$

Example (Octal to Decimal)

4 3 . 2 5

$4 \times 8^{1} \quad + \quad 3 \times 8^{0} \quad + \quad 2 \times 8^{-1} \quad + \quad 5 \times 8^{-2}$

$32 \quad + \quad 3 \quad + \quad 0.25 \quad + \quad 0.0781$

35.3281_{10}

$43.25_{8} = 35.3281_{10}$

Decimal to Hexa-Decimal

Decimal → Hexa-Decimal

- Technique
  - Divide by 16
  - Keep track of the remainder

Example (Decimal to HexaDecimal)

$1234_{10} = ?_{16}$

16	1234	2
16	77	13=D
16	4	4
	0	

$1234_{10} = 4D2_{16}$

Hexa-Decimal to Decimal

Hexa-Decimal → Decimal

- Technique
  - Multiply each bit by 16^n, where n is the “weight” of the bit
  - The weight is the position of the bit, starting from 0 on the right
  - Add the results

Example (HexaDecimal to Decimal)

A B C

$$A \times 16^2 + B \times 16^1 + C \times 16^0$$

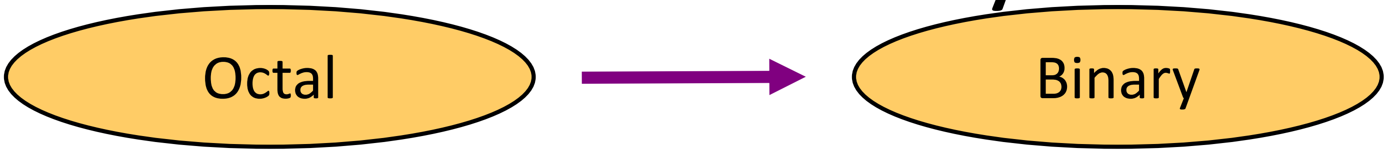
$$10 \times 16^2 + 11 \times 16^1 + 12 \times 16^0$$

$$2560 + 176 + 12$$

$$2748_{10}$$

$$ABC_{16} = 2748_{10}$$

Octal to Binary



- Technique
  - Convert each octal digit to a 3-bit equivalent binary representation

Octal - Binary Table

Octal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

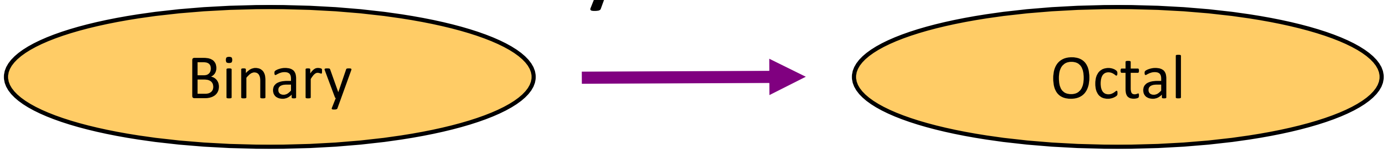
Example (Octal to Binary)

$705_8 = ?_2$

7	0	5
↓	↓	↓
111	000	101

$705_8 = 111000101_2$

Binary to Octal



- Technique
  - Group bits in threes, starting on right
  - Convert to octal digits

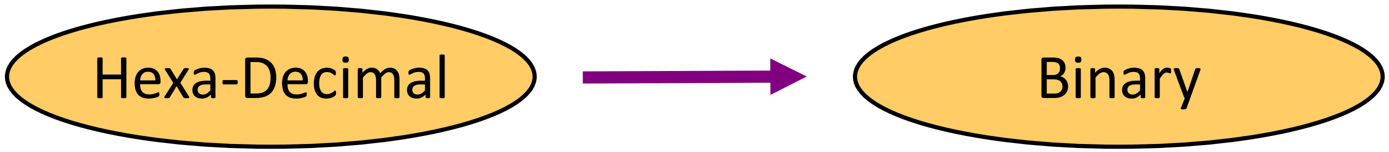
Example (Binary to Octal)

$1011010111_2 = ?_8$

$$\begin{array}{cccc} 001 & 011 & 010 & 111 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 1 & 3 & 2 & 7 \end{array}$$

$$1011010111_2 = 1327_8$$

Hexa-Decimal to Binary



- Technique
  - Convert each hexadecimal digit to a 4-bit equivalent binary representation

Hexa-Decimal to Binary

Hexa-Decimal	Binary	Hexa-Decimal	Binary
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

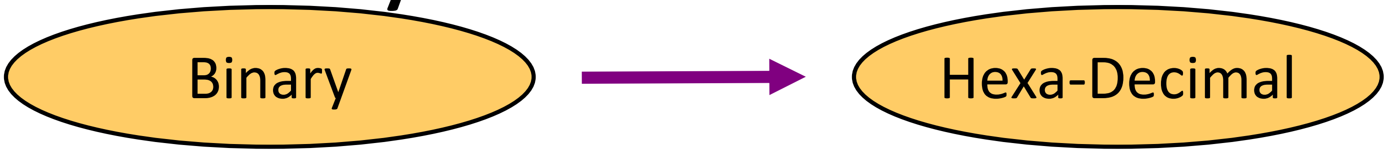
Example (Hexa-Decimal to Binary)

$10AF_{16} = ?_2$

1	0	A	F
↓	↓	↓	↓
0001	0000	1010	1111

$10AF_{16} = 1000010101111_2$

Binary to Hexa-Decimal



- Technique
  - Group bits in fours, starting on right
  - Convert to hexadecimal digits

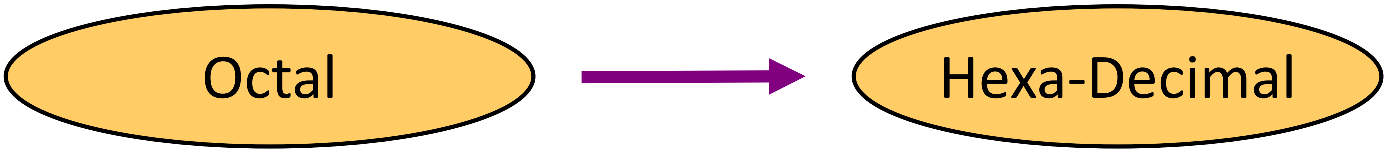
Example (Binary to Hexa-Decimal)

$1011010111_2 = ?_{16}$

$$\begin{array}{ccc} 0010 & 1101 & 0111 \\ \downarrow & \downarrow & \downarrow \\ 2 & D & 7 \end{array}$$

$$1011010111_2 = 2D7_{16}$$

Octal to Hexa-Decimal



- Technique
  - Convert Octal to Binary
  - Regroup bits in fours from right
  - Convert Binary to Hexa-Decimal

Example (Octal to Hexa-Decimal)

$1076_8 = ?_{16}$

1 0 7 6

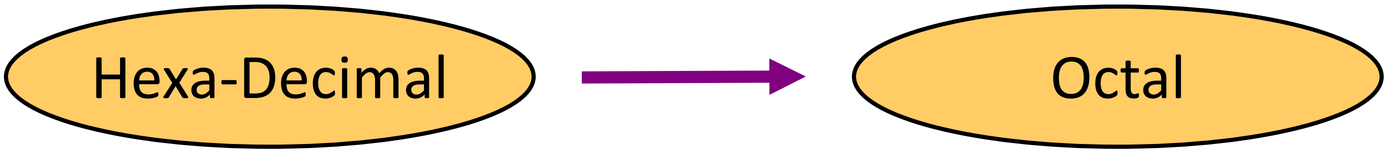
001 000 111 110

0010 0011 1110

2 3 E

$$1076_8 = 23E_{16}$$

Hexa-Decimal to Octal



- Technique
  - Convert Hexa-Decimal to Binary
  - Regroup bits in three from right
  - Convert Binary to Octal

Example (Hexa-Decimal to Octal)

$1F0C_{16} = ?_8$

1	F	0	C
↓	↓	↓	↓
0001	1111	0000	1100

000	001	111	100	001	100
↓	↓	↓	↓	↓	↓
0	1	7	4	1	4

$1F0C_{16} = 17414_8$

2.1.2 Binary Arithmetic

Binary Addition

- Rules for binary addition

	1	1	1	1		1	1	
	1	1	0	1	.	1	0	1
+	0	1	1	1	.	0	1	1
1	0	1	0	1	.	0	0	0

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10 i.e. 0 with a carry of 1

Binary Subtraction

- Rules for binary subtraction

$$\begin{array}{rcccccccc}
 & 0 & 1 & 10 & 1 & & 1 & 10 & \\
 & \cancel{1} & \cancel{0} & \cancel{1} & \cancel{0} & . & \cancel{0} & \cancel{1} & 10 \\
- & 0 & 1 & 1 & 1 & . & 1 & 1 & 1 \\ \hline
 & 0 & 0 & 1 & 0 & . & 0 & 1 & 1
\end{array}$$

0 – 0 = 0

1 – 1 = 0

1 – 0 = 1

0 – 1 = 1, with a borrow 1

Binary Multiplication

```
        1 0 1 1 1
   x    1 0 0 1 1
   ------------------
        1 0 1 1 1
      1 0 1 1 1
    0 0 0 0 0
  0 0 0 0 0
1 0 1 1 1
---------------------
1 1 0 1 1 0 1 0 1
```

Binary Division

```
110 | 1 0 1 1 0 1 | 0111.1
      0 0 0
      -------
      1 0 1 1
        1 1 0
      ---------
        1 0 1 0
          1 1 0
        ---------
          1 0 0 1
            1 1 0
          ---------
              1 1 0
              1 1 0
            ---------
              0 0 0
```

2.1.1 Representation of Signed Numbers, Floating Point Number

Signed Binary Numbers

- Two ways of representing signed numbers:
- 1) Sign-magnitude form, 2) Complement form.
- Most of computers use complement form for negative number notation.
- 1’s complement and 2’s complement are two different methods in this type.

1's Complement

- 1's complement of a binary number is obtained by subtracting each digit of that binary number from 1.
- Example

$$\begin{array}{r} 1\ 1\ 1\ 1 \\ -\ 1\ 1\ 0\ 1 \\ \hline 0\ 0\ 1\ 0 \end{array}$$

(1's complement of 1101)

$$\begin{array}{r} 1\ 1\ 1\ .\ 1\ 1 \\ -\ 1\ 0\ 1\ .\ 0\ 1 \\ \hline 0\ 1\ 0\ .\ 1\ 0 \end{array}$$

(1's complement of 101.01)

2's Complement

- 2's complement of a binary number is obtained by adding 1 to its 1's complement.

- Example

	1	1	1	1			1	1	1	.	1	1
-	1	1	0	0		-	1	0	1	.	0	1
	0	0	1	1			0	1	0	.	1	0
+				1		+						1
	0	1	0	0			0	1	0	.	1	1

(2's complement of 1100) (2's complement of 101.01)

9's Complement

- 9's complement of a decimal number is obtained by subtracting each digit of that decimal number from 9.

▪ Example

$$\begin{array}{rcccc} & 9 & 9 & 9 & 9 \\ - & 3 & 4 & 6 & 5 \\ \hline & 6 & 5 & 3 & 4 \end{array}$$

(9's complement of 3465)

$$\begin{array}{rcccccc} & 9 & 9 & 9 & . & 9 & 9 \\ - & 7 & 8 & 2 & . & 5 & 4 \\ \hline & 2 & 1 & 7 & . & 4 & 5 \end{array}$$

(9's complement of 782.54)

10’s Complement

- 10’s complement of a decimal number is obtained by adding 1 to its 9’s complement.

- Example

$$\begin{array}{rrrrr} & 9 & 9 & 9 & 9 \\ - & 3 & 4 & 6 & 5 \\ \hline & 6 & 5 & 3 & 4 \\ + & & & & 1 \\ \hline & 6 & 5 & 3 & 5 \end{array}$$

(10’s complement of 3465)

$$\begin{array}{rrrrcrr} & 9 & 9 & 9 & . & 9 & 9 \\ - & 7 & 8 & 2 & . & 5 & 4 \\ \hline & 2 & 1 & 7 & . & 4 & 5 \\ + & & & & & & 1 \\ \hline & 2 & 1 & 7 & . & 4 & 6 \end{array}$$

(10’s complement of 782.54)

$$7 - 5 = 2$$

minuend subtrahend difference

Subtraction using 9’s complement

- Obtain 9’s complement of subtrahend
- Add the result to minuend and call it intermediate result
- If carry is generated then answer is positive and add the carry to Least Significant Digit (LSD)
- If there is no carry then answer is negative and take 9’s complement of intermediate result and place negative sign to the result.

Example

1) 745.81 – 436.62

$$
\begin{array}{r}
7\ 4\ 5\ .\ 8\ 1\\
-\ \ 4\ 3\ 6\ .\ 6\ 2\\
\hline
3\ 0\ 9\ .\ 1\ 9
\end{array}
\xrightarrow{\text{9's complement}}
\begin{array}{r}
7\ 4\ 5\ .\ 8\ 1\\
+\ 5\ 6\ 3\ .\ 3\ 7\\
\hline
1\ 3\ 0\ 9\ .\ 1\ 8\\
+\ 1\\
\hline
3\ 0\ 9\ .\ 1\ 9
\end{array}
$$

Example

2) 436.62 - 745.81

$$
\begin{array}{r}
4\,3\,6\,.\,6\,2 \\
-\;7\,4\,5\,.\,8\,1 \\
\hline
-\;3\,0\,9\,.\,1\,9
\end{array}
\quad \xrightarrow{\text{9's complement}} \quad
\begin{array}{r}
4\,3\,6\,.\,6\,2 \\
+\;2\,5\,4\,.\,1\,8 \\
\hline
6\,9\,0\,.\,8\,0
\end{array}
$$

9's complement

$$-\;3\,0\,9\,.\,1\,9$$

As carry is not generated, so take 9's complement of the intermediate result and add ' – ' sign to the result

Subtraction using 10’s complement

- Obtain 10’s complement of subtrahend
- Add the result to minuend
- If carry is generated then ignore it and result itself is answer
- If there is no carry then answer is negative and take 10’s complement of result and place negative sign to the result.

Example

1) 745.81 – 436.62

$$
\begin{array}{r}
745.81 \\
-\ 436.62 \\
\hline
309.19
\end{array}
\xrightarrow{\text{10's complement}}
\begin{array}{r}
745.81 \\
+\ 563.38 \\
\hline
1309.19
\end{array}
$$

Ignore the carry

Example

2) 436.62 - 745.81

	4	3	6	.	6	2	
-	7	4	5	.	8	1	10's complement →
-	3	0	9	.	1	9	

	4	3	6	.	6	2
+	2	5	4	.	1	9
	6	9	0	.	8	1

10's complement

-	3	0	9	.	1	9

As carry is not generated, so take 10's complement of the intermediate result and add ' – ' sign to the result

Subtraction using 1’s Complement

- Obtain 1’s complement of **subtrahend**
- Add the result to **minuend** and call it intermediate result
- If carry is generated, then answer is positive and add the carry to Least Significant Bit (LSB)
- If there is no carry, then answer is negative and take 1’s complement of intermediate result and place negative sign to the result.

Example

1) 68.75 – 27.50

$$
\begin{array}{r}
68.75 \\
-\ 27.50 \\
\hline
+\ 41.25
\end{array}
\qquad
\xrightarrow{\text{1's complement}}
\qquad
\begin{array}{r}
01000100.1100 \\
+\ 11100100.0111 \\
\hline
100101001.0011 \\
+1 \\
\hline
00101001.0100
\end{array}
$$

Example

2) 43.25 - 89.75

	43.25		00101011.0100
-	89.75	1's complement →	+ 10100110.0011
-	46.50		11010001.0111
		1's complement	00101110.1000

As carry is not generated, so take 1's complement of the intermediate result and add ' – ' sign to the result

Subtraction using 2’s Complement

- Obtain 2’s complement of subtrahend
- Add the result to minuend
- If carry is generated, then ignore it and result itself is answer
- If there is no carry, then answer is negative and take 2’s complement of result and place negative sign to the result.

Example

1) 68.75 – 27.50

68.75		01000100.1100
- 27.50	2's complement →	+ 11100100.1000
+ 41.25		$\boxed{1}$00101001.0100

Ignore Carry bit

00101001.0100

Example

2) 43.25 - 89.75

$$
\begin{array}{rr}
 & 43.25 \\
- & 89.75 \\
\hline
- & 46.50
\end{array}
\qquad \xrightarrow{\text{2's complement}} \qquad
\begin{array}{rr}
 & 00101011.0100 \\
+ & 10100110.0100 \\
\hline
 & 11010001.1000 \\
 & \\
 & 00101110.1000
\end{array}
$$

2's complement

As carry is not generated, so take 2's complement of the intermediate result and add ' – ' sign to the result

Signed Binary Numbers

- To represent negative integers, we need a notation for negative values.
- It is customary to represent the sign with a bit placed in the leftmost position of the number since binary digits.
- The convention is to make the sign bit 0 for positive and 1 for negative.
- Different methods of representations (example):

Signed-magnitude representation:	10001001
Signed-1's-complement representation:	11110110
Signed-2's-complement representation:	11110111

- ... he three representations.

Signed Binary Numbers

Table 1.3
Signed Binary Numbers

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

Representation of negative number in 2's complement form

- Express -65.5 in 12 bit 2's complement form.

2	65	1
2	32	0
2	16	0
2	8	0
2	4	0
2	2	0
2	1	1
	0	

$0.5 \times 2 = 1.0$

So, result in 12-bit binary is as follows:

$65.5_{10} = 01000001.1000_2$

For negative number, we have to convert this into 2's complement form

$-65.5_{10} = 10111110.1000_2$

Signed Binary Numbers

- Arithmetic addition
  - The addition of two numbers in the signed-magnitude system follows the rules of ordinary arithmetic. **If the signs are the same**, we add the two magnitudes and give the sum the common sign. **If the signs are different**, we subtract the smaller magnitude from the larger and give the difference the sign if the larger magnitude.
  - The addition of two signed binary numbers with negative numbers represented in signed-2's-complement form is obtained from the addition of the two numbers, including their sign bits.
  - A carry out of the sign-bit position is discarded.
- Example:

+ 6	00000110	− 6	11111010
+13	00001101	+13	00001101
+ 19	00010011	+ 7	00000111
+ 6	00000110	− 6	11111010
−13	11110011	−13	11110011
− 7	11111001	− 19	11101101

Signed Binary Numbers

- Arithmetic Subtraction
  - In 2's-complement form:

1. Take the 2's complement of the subtrahend (including the sign bit) and add it to the minuend (including sign bit).
2. A carry out of sign-bit position is discarded.

$$(\pm A)-(+B)=(\pm A)+(-B)$$
$$(\pm A)-(-B)=(\pm A)+(+B)$$

- Example:

(– 6) – (– 13) → (11111010 – 11110011)

→ (11111010 + 00001101)

→ 00000111 (+ 7)

Representation of Floating-Point Number

BINARY REPRESENTATION OF FLOATING-POINT NUMBERS

Converting decimal fractions into binary representation.

Consider a decimal fraction of the form: 0.d1d2...dn

We want to convert this to a binary fraction of the form:

0.b1b2...bn (using binary digits instead of decimal digits)

Algorithm for conversion

Let X be a decimal fraction: $\mathbf{0.d_1d_2..d_n}$

i = 1

Repeat until X = 0 or i = required no. of binary fractional digits {

    Y = X * 2

    X = fractional part of Y

    B_i = integer part of Y

    i = i + 1

}

EXAMPLE 1

Convert 0.75 to binary

X = 0.75   (initial value)

X* 2   = 1.50. Set   b1 = 1, X = 0.5

X* 2   = 1.0.   Set   b2 = 1, X = 0.0

The binary representation for 0.75 is thus

0.b1b2 = 0.11b

Let's consider what that means...

In the binary representation

$0.b_1b_2...b_m$

b_1 represents 2^{-1} (i.e., 1/2)

b_2 represents 2^{-2} (i.e., 1/4)

...

b_m represents 2^{-m} $(1/(2^m))$

So, 0.11 binary represents

$2^{-1} + 2^{-2} = 1/2 + 1/4 = 3/4 = 0.75$

EXAMPLE 2

Convert the decimal value 4.9 into binary

Part 1: convert the integer part into binary: 4 = 100 b

Part 2.

Convert the fractional part into binary using multiplication by 2:

$X = .9*2 = 1.8.$	Set $b_1 = 1$, $X = 0.8$
$X*2 = 1.6.$	Set $b_2 = 1$, $X = 0.6$
$X*2 = 1.2.$	Set $b_3 = 1$, $X = 0.2$
$X*2 = 0.4.$	Set $b_4 = 0$, $X = 0.4$
$X*2 = 0.8.$	Set $b_5 = 0$, $X = 0.8$,

which repeats from the second line above.

Since X is now repeating the value 0.8,

we know the representation will repeat.

The binary representation of **4.9** is thus:

100.1110011001100...

COMPUTER REPRESENTATION OF FLOATING-POINT NUMBERS

In the CPU, a 32-bit floating point number is represented using IEEE standard format as follows:

SIGN | EXPONENT | MANTISSA

where SIGN is one bit, the EXPONENT is 8 bits, and the MANTISSA is 23 bits.

- The ***mantissa*** represents the leading significant bits in the number.

- The ***exponent*** is used to adjust the position of the binary point (as opposed to a "decimal" point)

- The mantissa is said to be **normalized** when it is expressed as a value between 1 and 2. I.e., the mantissa would be in the form 1.xxxx.

- The leading integer of the binary representation is not stored. Since it is always a 1, it can be easily restored.

- The "SIGN" bit is used as a sign bit and indicates whether the value represented is positive or negative

  (0 for positive, 1 for negative)

- If a number is smaller than 1, normalizing the mantissa will produce a negative exponent.

- But 127 is added to all exponents in the floating point representation, allowing all exponents to be represented by a positive number.

Example 1. Represent the decimal value 2.5 in 32-bit floating point format.

2.5 = 10.1

In normalized form, this is: $1.01 * 2^1$

The mantissa: M = 01000000000000000000000
(23 bits without the leading 1)

The exponent: E = 1 + 127 = 128 = 10000000
The sign: S = 0 (the value stored is positive)

So, **2.5 = 0 10000000 01000000000000000000000**

Example 2: Represent the number -0.00010011 in floating point form.

$$0.00010011 = 1.0011 * 2^{-4}$$

Mantissa: M = 00110000000000000000000 (23 bits with the integral 1 not represented)

Exponent: E = -4 + 127 = 01111011

S = 1 (as the number is negative)

Result: **1 01111011 00110000000000000000000**

Exercise 1: represent -0.75 in floating point format.

Exercise 2: represent 4.9 in floating point format.

Complements
(Summary)

Complements

- There are two types of complements for each base-*r* system: the radix complement and diminished radix complement.
- **Diminished Radix Complement - (r-1)'s Complement**
  - Given a number *N* in base *r* having *n* digits, the (*r*–*1*)'s complement *of N* is defined as:

    $(r^n – 1) – N$
- **Example for 6-digit <u>decimal</u> numbers**:
  - 9's complement is $(r^n – 1) – N = (10^6 – 1) – N = 999999 – N$
  - 9's complement of 546700 is 999999–546700 = 453299
- **Example for 7-digit <u>binary</u> numbers:**
  - 1's complement is $(r^n – 1) – N = (2^7 – 1) – N = \underline{1111111} – N$
  - 1's complement of 1011000 is <u>1111111</u>–1011000 = 0100111
- **Observation:**
  - Subtraction from $(r^n – 1)$ will never require a borrow
  - Diminished radix complement can be computed digit-by-digit
  - For binary: 1 – 0 = 1 and 1 – 1 = 0

Complements

- 1's Complement (*Diminished Radix* Complement)
  - –All '0's become '1's
  - –All '1's become '0's

Example $(10110000)_2$

□ $(01001111)_2$

If you add a number and its 1's complement…

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

Complements

- Radix Complement

The r's complement of an n-digit number N in base r is defined as $r^n - N$ for $N \neq 0$ and as 0 for $N = 0$. Comparing with the $(r - 1)$'s complement, we note that the r's complement is obtained by adding 1 to the $(r - 1)$'s complement, since $r^n - N = [(r^n - 1) - N] + 1$.

- Example: Base-10

The 10's complement of 012398 is 987602
The 10's complement of 246700 is 753300

- Example: Base-2

The 2's complement of 1101100 is 0010100
The 2's complement of 0110111 is 1001001

Complements

- 2's Complement (*Radix* Complement)
  - Take 1's complement then add 1

  OR
  - Toggle all bits to the left of the first '1' from the right

Example:

Number:	1 0 1 1 0 0 0 0	1 0 1 1 0 0 0 0
1's Comp.:	0 1 0 0 1 1 1 1	
+	1	
	0 1 0 1 0 0 0 0	0 1 0 1 0 0 0 0

Complements

- Subtraction with Complements
  - The subtraction of two *n*-digit unsigned numbers $M - N$

1. Add the minuend M to the r's complement of the subtrahend N. Mathematically, $M + (r^n - N) = M - N + r^n$.
2. If $M \geq N$, the sum will produce and end carry r^n, which can be discarded; what is left is the result $M - N$.
3. If $M < N$, the sum does not produce an end carry and is equal to $r^n - (N - M)$, which is the r's complement of $(N - M)$. To obtain the answer in a familiar form, take the r's complement of the sum and place a negative sign in front.

Complements

- Example–
- Using 10's complement, subtract 72532 – 3250.

$$\begin{aligned} M &= \quad 72532 \\ \text{10's complement of } N &= \underline{+96750} \\ \text{Sum} &= \quad 169282 \\ \text{Discard end carry } 10^5 &= \underline{-100000} \\ \text{Answer} &= \quad 69282 \end{aligned}$$

- Example
  - Using 10's complement, subtract 3250 – 72532.

$$\begin{aligned} \text{M} &= \quad 03250 \\ \text{10's complement of N} &= \underline{+27468} \\ \text{Sum} &= \quad 30718 \end{aligned}$$

There is no end carry.

Therefore, the answer is – (10's complement of 30718) = – 69282.

Complements

- Example
  - Given the two binary numbers X = 1010100 and Y = 1000011, perform the subtraction (a) $X - Y$; and (b) $Y - X$, by using 2's complement.

(a)	X =	1010100
	2's complement of Y =	+0111101
	Sum =	10010001
	Discard end carry 2^7 =	–10000000
	Answer. X – Y =	0010001

(b)	Y =	1000011
	2's complement of X =	+ 0101100
	Sum =	1101111

There is no end carry. Therefore, the answer is Y – X = – (2's complement of 1101111) = – 0010001.

Complements

- Subtraction of unsigned numbers can also be done by means of the (r – 1)'s complement. Remember that the (r – 1) 's complement is one less then the r's complement.
- Example
  - Repeat Previous Example , but this time using 1's complement.

(a) $X - Y = 1010100 - 1000011$

$$\begin{array}{rr}
X = & 1010100 \\
\text{1's complement of } Y = & \pm\, 0111100 \\
\text{Sum} = & 10010000 \\
\text{End-around carry} = & +\quad\quad 1 \\
\hline
\text{Answer. } X - Y = & 0010001
\end{array}$$

(b) $Y - X = 1000011 - 1010100$

$$\begin{array}{rr}
Y = & 1000011 \\
\text{1's complement of } X = & +\, 0101011 \\
\hline
\text{Sum} = & 1101110
\end{array}$$

There is no end carry, Therefore, the answer is Y – X = – (1's complement of 1101110) = – 0010001.

Signed Binary Numbers

- To represent negative integers, we need a notation for negative values.
- It is customary to represent the sign with a bit placed in the leftmost position of the number since binary digits.
- The convention is to make the sign bit 0 for positive and 1 for negative.
- Example:

Signed-magnitude representation:	10001001
Signed-1's-complement representation:	11110110
Signed-2's-complement representation:	11110111

- Table 1.3 lists all possible four-bit signed binary numbers in the three representations.

Signed Binary Numbers

Table 1.3
Signed Binary Numbers

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

Signed Binary Numbers

- Arithmetic addition
  - The addition of two numbers in the signed-magnitude system follows the rules of ordinary arithmetic. **If the signs are the same**, we add the two magnitudes and give the sum the common sign. **If the signs are different**, we subtract the smaller magnitude from the larger and give the difference the sign if the larger magnitude.
  - The addition of two signed binary numbers with negative numbers represented in signed-2's-complement form is obtained from the addition of the two numbers, including their sign bits.
  - A carry out of the sign-bit position is discarded.
- Example:

+ 6	00000110	− 6	11111010
+13	00001101	+13	00001101
+ 19	00010011	+ 7	00000111
+ 6	00000110	− 6	11111010
−13	11110011	−13	11110011
− 7	11111001	− 19	11101101

Signed Binary Numbers

- Arithmetic Subtraction
  - 1. Take the 2's complement of the subtrahend (including the sign bit) and add it to the minuend (including sign bit).
    2. A carry out of sign-bit position is discarded.

$$(\pm A) - (+B) = (\pm A) + (-B)$$
$$(\pm A) - (-B) = (\pm A) + (+B)$$

- Example:

(– 6) – (– 13) → (11111010 – 11110011)

→ (11111010 + 00001101)

→ 00000111 (+ 7)

Binary Storage and Registers

- Registers
  - A binary cell is a device that possesses two stable states and is capable of storing one of the two states.
  - A register is a group of binary cells. A register with *n* cells can store any discrete quantity of information that contains *n* bits.

n cells → 2^n possible states

- A binary cell
  - Two stable state
  - Store one bit of information
  - Examples: flip-flop circuits, ferrite cores, capacitor
- A register
  - A group of binary cells
  - AX in x86 CPU
- Register Transfer
  - A transfer of the information stored in one register to another.
  - One of the major operations in digital system.
  - An example in next slides.