Number Systems and Base Conversion

Positional Number Systems

A way to represent numbers where **the position of each digit determines its value**. The value of a digit depends on two factors:

- The digit itself (what number it represents, e.g., 0, 1, 2, ...).
- The position of the digit relative to a fixed reference point, usually the radix point (like a decimal point in base-10).

The position determines the power of the base (or radix) that the digit is multiplied by.

Base / Radix

The base (or radix) of a number system defines **how many unique digits** (symbols) can be used to represent numbers.

- Base-10 (Decimal): Digits are $\{0, 1, 2, \ldots, 9\}$.
- Base-2 (Binary): Digits are $\{0, 1\}$.
- Base-8 (Octal): Digits are $\{0, 1, 2, \ldots, 7\}$.
- Base-16 (Hexadecimal): Digits are $\{0, 1, 2, \ldots, 9, A, B, C, D, E, F\}$.

Radix Point

The radix point is a general term for what we call the decimal point in base-10 numbers. It separates the integer part of the number (to the left) from the fractional part (to the right).

Binary Numbers

The binary number system, or base-2, is important to computing because **it directly represents the on-off states of electronic switches** in hardware. Unlike the decimal system (base-10) with digits 0 through 9, binary has only two digits: 0 and 1.

Arithmetic Operations in Binary

- **Binary Addition**: Follows the same rules as decimal (base-10) addition.

```
   111    (carry)
  01011   (11 in decimal)
+ 01101   (13 in decimal)
-------
  11000   (24 in decimal)
```

- **Binary Subtraction**: Note that the borrow adds the radix *r* to the minuend digit. In this case, each borrow in Binary Subtraction will add 2 to the minuend digit.

```
   0122   (borrow)
   1010   (10 in decimal)
-   011   (3 in decimal)
-------
   0111   (7 in decimal)
```

- **Binary Multiplication**: Notice that since the multiplier digits in base-2 are always 1 or 0, you can assume that the partial products are equal to either a left-shifted copy of the multiplicand or 0.

```
   1011   (11 in decimal)
×    10   (2 in decimal)
-------
   0000   (partial products)
+ 1011
-------
  10110   (22 in decimal)
```

- **Binary Division**: Follows the same steps as long division in decimal.

```
       1 1 1 1   (15 in decimal)
   ___________
10 ) 1 1 1 1 0   (30 in decimal)
   - 1 0 | | |
   ----- | | |
       1 1 | |
     - 1 0 | |
     ----- | |
         1 1 |
       - 1 0 |
       ----- |
           1 0
         - 1 0
         -----
```

Converting Base-*r* to Decimal

- Commonly referred to as weighted conversion because the value of each digit in the base-*r* number is "weighted" by its positional value, which is a power of the radix *r*.
- Positions to the left of the radix point (.) are non-negative powers, and positions to the right are negative powers.

Example 1: $(11.3) = (?)_{10}$

$(11.3)_4 = 1 * 4^1 + 1 * 4^0 + 3 * 4^{-1}$

value	weight	result
1	4^1	4
1	4^0	1
3	4^{-1}	0.750

Add the terms:

$4 + 1 + 0.750 = 5.750$

base-10 = 5.750

Converting Decimal to Base-*r*

- **Whole Numbers (Repetitive Reverse Division)**: Repeatedly dividing by the radix *r* and recording the remainder. Repeat until the result is 0.
    - Remainders are read from bottom to top.

Example 1: $(135)_{10} = (?)_5$

Base	Num	Rem		
5	135	-		
	27	0	^	(27 r. 0)
	5	2	|	(5 r. 2)
	1	0	|	(1 r. 0)
	0	1	|	(0 r. 1)

base-5 = 1020

- **Fractions (Repetitive Multiplication)**: Multiply the decimal fraction by the radix *r*, and record the integer part at each step. Repeat until the remainder is 0.000 or the result is at an acceptable accuracy (usually set by company standard or course professor).
    - Integers are read from top to bottom.

Example 1: $(0.375)_{10} = (?)_6$

```
Base   Frac    Int      Rem
6      0.375   2    |   0.250   (0.375 * 6 = 2.250)
       0.250   1    |   0.500   (0.250 * 6 = 1.500)
       0.500   3    v   0.000   (0.500 * 6 = 3.000)

base-6 = 0.213
```

Converting Base-*r* to Base-*s*

- Also called base-to-base conversion or simply radix conversion.
- Typically involves an intermediate step, where the number is first converted to decimal (base-10) and then to the target base-*s*.

Example 1: $(324)_5 = (?)_3$

Convert from base-r to decimal:

$(324)_5 = 3 * 5^2 + 2 * 5^1 + 4 * 5^0 = 75 + 10 + 4 = (89)_{10}$

Convert from decimal to base-s:

```
Base   Num   Rem
3       89     -
        29     2  ^   (29 r. 2)
         9     2  |   ( 9 r. 2)
         3     0  |   ( 3 r. 0)
         1     0  |   ( 1 r. 0)
         0     1  |   ( 0 r. 1)

base-3 = 1 0022
```

Quaternary, Octal, and Hexadecimal Numbers

Some number systems are special because their bases are a power of 2. These number systems can **serve as shorthand systems for representing binary**, which makes human interaction with binary data easier. This is done by partitioning the digits into groups. However, note that there are extra 0s added in order to do the partitioning. Be careful about this, as this can heavily affect the result of converting decimal fractions. For visibility purposes, these extra 0s are denoted in **green**.

- **Quaternary System (Base-4)**: Remember than 4 is equivalent to 2^2, hence each digit in an octal number represents 2 binary digits (bits).

```
Example 1: (10110.0111)_2 = (?)_4

number:

01 | 01 | 10   (partition into groups of 2 starting from right)
 1 |  1 |  2   (decimal / base-10 representation)

fraction:

01 | 11   (partition into groups of 2 starting from left)
 1 |  2   (decimal / base-10 representation)

base-4 = 112.12
```

- **Octal System (Base-8)**: Remember than 8 is equivalent to 2^3, hence each digit in an octal number represents 3 binary digits.

```
Example 1: (10110.0111)_2 = (?)_8

number:

010 | 110   (partition into groups of 3 starting from right)
 2  |  6    (decimal / base-10 representation)

fraction:

011 | 100   (partition into groups of 3 starting from left)
 3  |  4    (decimal / base-10 representation)

base-8 = 26.34
```

- **Hexadecimal System (Base-16)**: Each digit in hexadecimal represents 4 binary digits.

```
Example 1: (10110.0111)_2 = (?)_16

number:

0001 | 0110   (partition into groups of 4 starting from right)
  1  |  6     (decimal / base-10 representation)

fraction:

0111   (partition into groups of 4 starting from left)
  7    (decimal / base-10 representation)

base-16 = 16.7
```

Binary vs. Quaternary vs. Octal vs. Hexadecimal Numbers

Even for small numbers, binary can easily get verbose. These systems simplify these representations, making them more readable while remaining easy to convert back to binary.

Decimal	Binary	Quaternary	Octal	Hexadecimal
00	0000	00	00	0
01	0001	01	01	1
02	0010	02	02	2
03	0011	03	03	3
04	0100	10	04	4
05	0101	11	05	5
06	0110	12	06	6
07	0111	13	07	7
08	1000	20	10	8
09	1001	21	11	9
10	1010	22	12	A
11	1011	23	13	B
12	1100	30	14	C
13	1101	31	15	D
14	1110	32	16	E
15	1111	33	17	F

FAQ: What number systems will be used in the ARCH series?

? According to Sir Roger, we will have our discussions **mainly using the decimal and binary number systems**. Additionally, CSARCH1 and CSARCH2 will use quaternary and hexadecimal, with octal being the least used in real-life applications. However, do note that **all will be used in the exams**, so it would be best to familiarize yourself with all of them.