Number bases and basics

16 October 2024 10:17

Why do we use binary in a computer system?

Provides the greatest opportunity for noise immunity

How many bits does one decimal symbol takes?

Approximately ≈ 3.3 bits

What conclusion can be drawn from this?

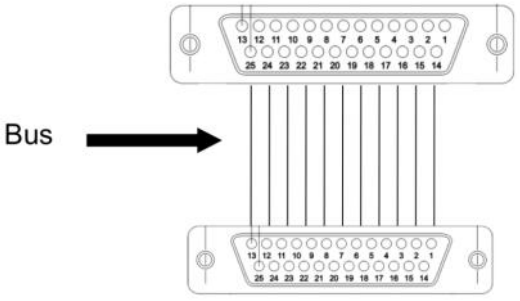
Hexadecimal and octal are more convenient than decimal when describing values on a bus

Electrical Circuit Representation

TTL (transistor-transistor logic)

0V - 0.8V represents 0

2.4V - 5V represents 1



Parallel bus

Collection of wires communicating a value between sub-circuits

A Brief History of Processor Representation

Intel 4004 (1971)
The first commercially available microprocessor had a word length of 4 bits

MOS 6502 and Zilog Z80 (~1975)
Popular 8-bit processors in early personal computers, such as the Apple II (6502) and TRS-80 (Z80)

Motorola 68030, PowerPC G4, and Intel Pentium (upto 2004)
Processor at the era of 32-bit architecture, being widely used in personal computers and workstations

IBM PowerPC G5 and x86-64 (e.g., Intel Core 2 and AMD64, introduced in 2003–2004)

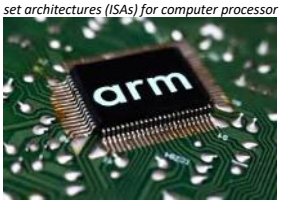
The transition to 64- bit, offering higher performance, larger addressable memory space, and greater efficiency in modern computing

Modern Processors (2020s onwards)
Processors like:
- *Intel Core i9*
- *AMD Ryzen*
- *Apple M-series*

continue to evolve the 64-bit architecture with
- *multi-core designs*
- *advanced parallel processing*
- *coprocessors for AI*
- *Coprocessors for machine learning*

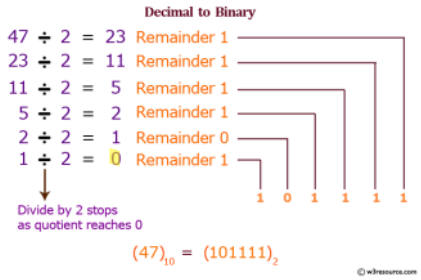
and be informed by design elements of ARM and RISC-V

ARM (stylised in lowercase as arm, formerly an acronym for Advanced RISC Machines and originally Acorn RISC Machine) is a family of RISC instruction set architectures (ISAs) for computer processor

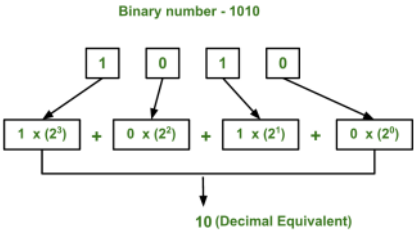


Converting between bases:

1. *Decimal to binary*



2. *Binary to decimal*

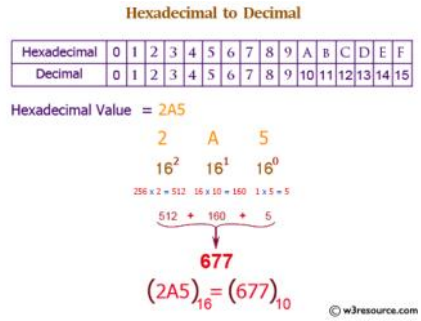


3. *Decimal to hex*

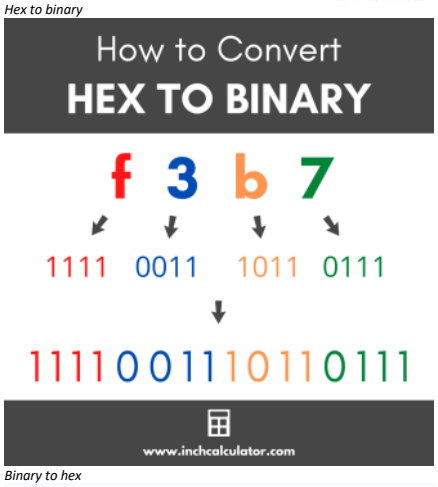
It may be easier to convert into binary first and then into the required base

$23_{10} = 16 + 4 + 2 + 1$
$= 10111_2$
$= 27_8 \quad (010\ 111_2)$
$= 17_{16} \quad (0001\ 0111_2)$

4. *Hex to decimal*



5. *Hex to binary*

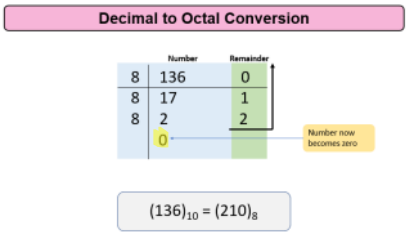


6. *Binary to hex*

To convert binary numbers into hexadecimals, you only have to make 4-bit groups and convert directly each group:

1011 0011 0101 (binary)
B 3 5 (hex)

7. *Decimal to Octal*



Signed Magnitude Representation

The MSB is an indicator, i.e., indicates it is negative

The value of the remaining bits are calculated as before

1101_{2SM}:	**1**	**1**	**0**	**1**
Position:	**negative?**	**2^2**	**2^1**	**2^0**
=	**negative?**	**4**	**2**	**1**
Value:	**-1 x**	**(1 x 4 +**	**0 x 2 +**	**1 x 1)**
=	**-5_{10}**			

Two's Complement Properties

Can represent more negative numbers than positive

The range is asymmetric

$1000_{2TC} = -8$

$0111_{2TC} = +7$

For N bits, the range is -2^{N-1} to $(2^{N-1}-1)$

Unique zero

Signed magnitude has both +0 and –0

Forming Two's Complement

To form the bit pattern for a negative number in two's complement:

Take the representation of the positive number

Invert the bits, ensuring there are enough bits for the MSB to be the sign

Add 1, ignoring any overflow

What is the two's complement bit pattern for -9?

$+9_{10} = 1001_2$
Ensure enough bits: 00001001_2
Invert: 11110110_2
Add 1: 11110111_2
$-9_{10} = 11110111_{2TC}$

Addition in Two's Complement

Addition is almost the same as we've seen previously

Just as before but ignore any overflow

```
 00001001 = +9      00001101 = +13     00000101 = +5
+11110111 = -9     +11110111 = -9     +11110111 = -9
 00000000 = 0       00000100 = 4       11111100 = -4
 11111111           11111111               111
```

Fixed Point Representation

Add columns to denote the fractional parts of a value

10.11_2	**=**	**1**	**0**	**1**	**1**
Position:		**2^1**	**2^0**	**2^{-1}**	**2^{-2}**
	=	**2**	**1**	**0.5**	**0.25**
Value	**=**	**(1 x 2 +**	**0 x 1 +**	**1 x 0.5 +**	**1 x 0.25)**
	=	**2.75_{10}**			

Floating Point Representation

Scientific notation allows small and large values to be written succinctly
The speed of light is 3x108 m/s ≈ 299,792,458 m/s
Floating point uses the same principles as scientific notation
Express numbers as (mantissa x 10exponent)
The mantissa represents the detail of the value to a certain precision
The exponent represents the magnitude of the value

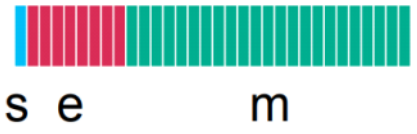
IEEE Floating Point

Floating point representation trades precision for range

- Extending range at the expense of reducing precision

IEEE Standard 754 is widely used and specifies levels of binary precision

- Single precision (using 32 bits)
- Double precision (using 64 bits)
- Quad precision (using 128 bits)



For example, single precision uses 1 bit for the sign (s), 8 bits for the exponent (e) and 23 bits for the mantissa (m)