Summary

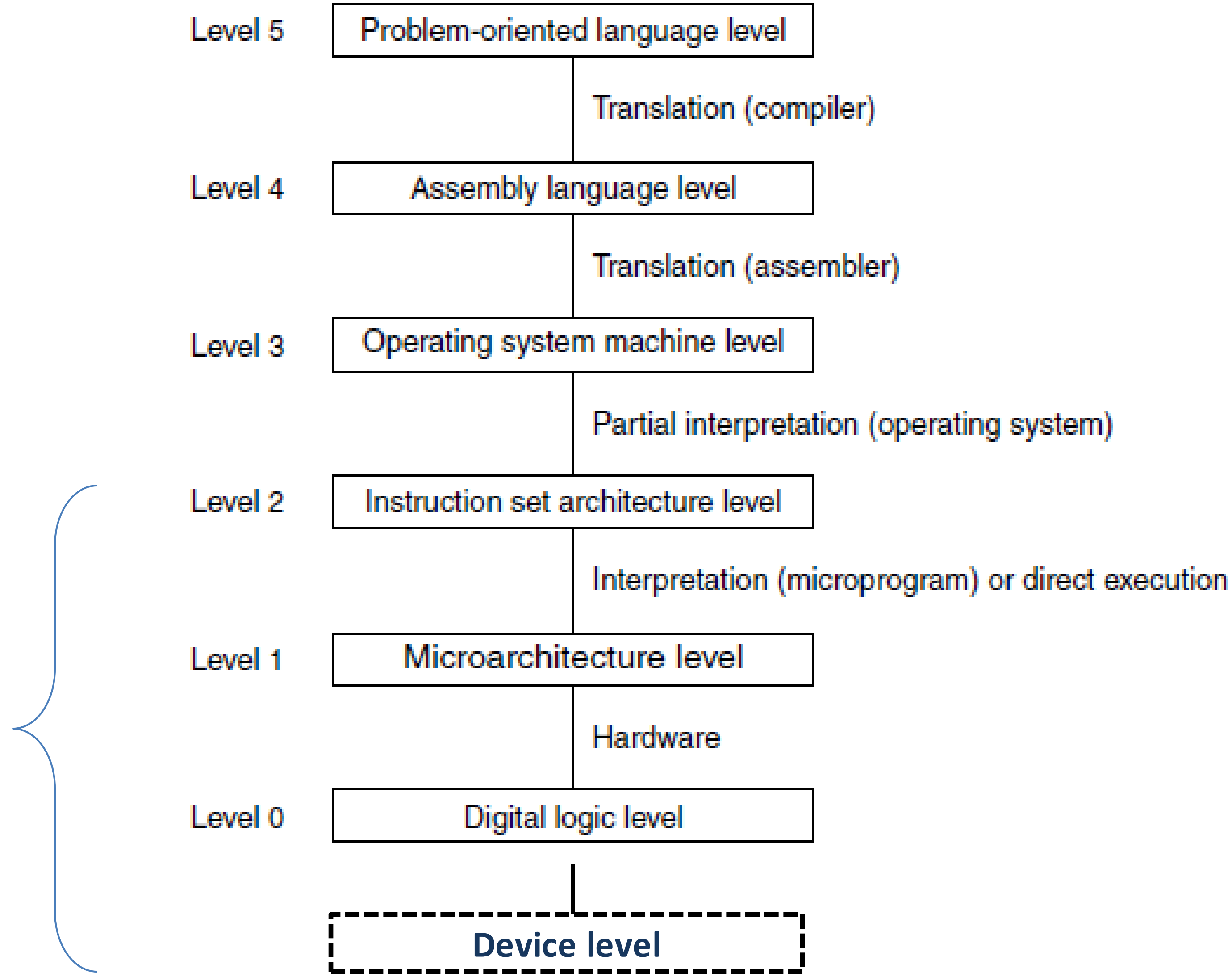
Machine can understand 0 or 1, so we use high level programming languages to communicate with machines.

Two ways to execute high level machine code:

Translation and interpretation

Hardwired or microprogram are two ways used in CPU’s to implement the control unit.

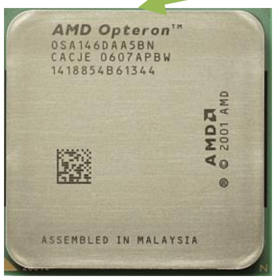
Summary
Current Multilevel Machines



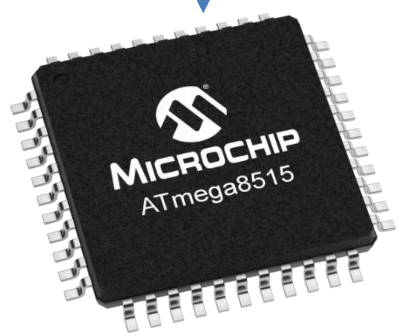
Summary

Different Architectures

X86, AVR, ARM



Microprogrammed control unit



Hardwired control unit

1) Data Representations
2) Boolean Algebra
3) Logic gates

Data Representations

Math (review)

Number Systems

1. Decimal {0,1,…9}
2. Octal {0,1,…7}
3. Hexadecimal {0,1,…10(A),…15(F)}
4. Binary {0,1}

Floating Point Representation

Data Representations

Math

Laws of exponents

For any number a:

$a^m.a^n = a^{n+m}$ (same base but different powers)

$a^1 = a$

$a^0 = 1$

$a = a.1$

Examples:

$10^0 = 1$

$10^1 = 10$

$100 = 10.10 = 10^1.10^1 = 10^2$

$1000 = 10.10.10 = 10^1.10^1.10^1 = 10^3$

Data Representations

Math

Positive and negative numbers

Examples:

Positive numbers: +1 (1), +200,(200) +45(45)

Negative numbers: -1, -200, -45

Zero: is not positive or negative

-(-5) = 5

Data Representations

Math

Addition

carry out

```
  1
 1 8 9
+9 4 8
-------
     7
```

9+8 = 17

Data Representations

Math

Addition

carry out

```
 1 1

  1 8 9
 +9 4 8
-------
    3 7
```

$8+4+1 = 13$

Data Representations

Math

Addition

```
carry out
 1  1
  1 8 9
 +9 4 8
--------
 1137
```

$1+9+1 = 11$

Data Representations

Math

Addition

```
  8 1 4
+ 9 9 7
 -------
  ????
```

Data Representations

Math

Subtraction

Example:

$4 - 9 = 4 + (-9) = -5$

$-9 - 4 = (-9) + (-4) = -13$

Subtraction can be considered addition of a negative number

Example:

If you have 9 dollars and you owe someone 4 dollars, you really only have 5 dollars

$9 - 4 = 9 + (-4) = 5$

Data Representations

Math

Subtraction

Example:

4 – 4 = 4 + (-4) = 0

4 + positive number = 0 ????

Is it possible to replace -4 with a positive number and get zero ?

Data Representations

Decimal

Decimal system:

Symbols or digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Example:

224 = two hundred twenty four

= 200 20 4

$= 200 + 20 + 4$

$= 2x100 + 2x10 + 4$

$= 2x(10x10) + 2x10 + 4$

$= 2x10^2 + 2x10^1 + 4x1$

$= 2x10^2 + 2x10^1 + 4x10^0$

Data Representations

Decimal

Decimal system:

Symbols or digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Example:

390 =

Data Representations

Octal

Octal system:

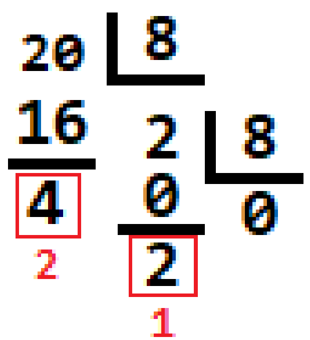
Symbols or digits: 0, 1, 2, 3, 4, 5, 6, 7

Example:

$20\ (20_{10}) = 24_8$

Converting decimal to octal

1. Continually divide by 8 until the number is greater than or equal zero and less than 8
2. At each step write down the reminder ($0 \leq$ reminder < 8)
3. At the end write out the remainders in the reverse order



$$20_{10} = 24_8$$

1 2

Data Representations

Octal

Octal system:

Symbols or digits: 0, 1, 2, 3, 4, 5, 6, 7

Example:

100 (100_{10}) = $(144)_8$

Data Representations

Octal

Converting octal to decimal

$(a_n \ldots a_3 a_2 a_1 a_0)_8 = a_n x 8^n + \ldots + a_3 x 8^3 + a_2 x 8^2 + a_1 x 8^1 + a_0 x 8^0$

Example:

$24_8 = 2X8^1 + 4X8^0 = 20$

$100_8 = 1X8^2 + 0X8^1 + 0X8^0 = 64$

Example:

$108_8 = (?)_{10}$

Data Representations

Hexadecimal

Hexadecimal systems:

Symbols or digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(10), B(11), C(12), D(13), E(14), F(15)

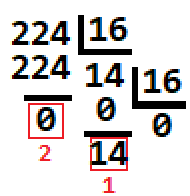
Example:

$$224_{10} = E0_{16}$$

$$224_{10} = EX16^{2} + 0X16^{0} = 14X16^{2} + 0X16^{0}$$

Converting decimal to hexadecimal

1. Continually divide by 16 until the number is greater than or equal zero and less than 16
2. At each step write down the reminder ($0 \leq$ reminder < 16)
3. At the end write out the remainders in the reverse order



$$224_{10} = \overset{1}{E}\overset{2}{0}_{16}$$

Data Representations

Hexadecimal

Hexadecimal systems:

Symbols or digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(10), B(11), C(12), D(13), E(14), F(15)

Example:

$100_{10} = (64)_{16}$

Data Representations

Hexadecimal

Converting hexadecimal to decimal

$(a_n \ldots a_3 a_2 a_1 a_0)_8 = a_n x16^n + \ldots + a_3 x16^3 + a_2 x16^2 + a_1 x16^1 + a_0 x16^0$

Example:

$E0_{16} = EX16^1 + 0X16^0 = 14X16^1 + 0X16^0 = 224_{10}$

$ABC_{16} = (?)_{10}$

Data Representations

Binary

Binary systems:

Symbols or digits: 0, 1

Example:

$7_{10} = 111_2$

$8_{10} = 1000_2$

$9_{10} = 1001_2$

$10_{10} = 1010_2$

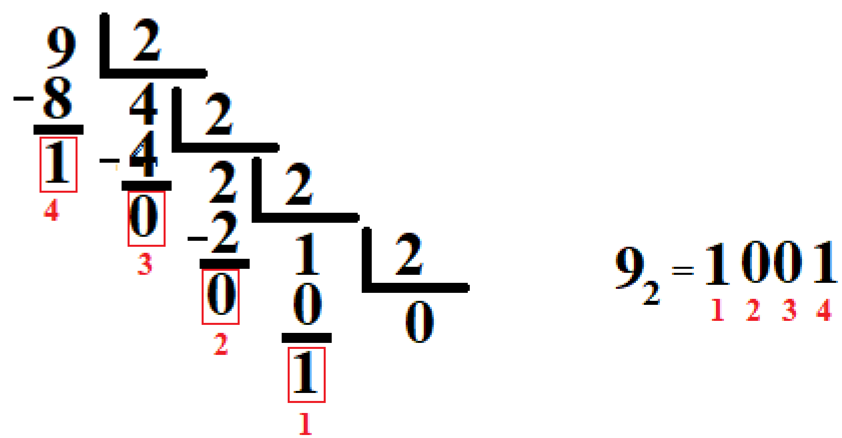
$100_{10} = 1100100_2$

Data Representations

Binary

Converting decimal to binary

1. Continually divide by 2 until the number is greater than or equal zero and less than 2
2. At each step write down the reminder (0 ≤ reminder < 2)
3. At the end write out the remainders in the reverse order



Data Representations

Binary

Example:

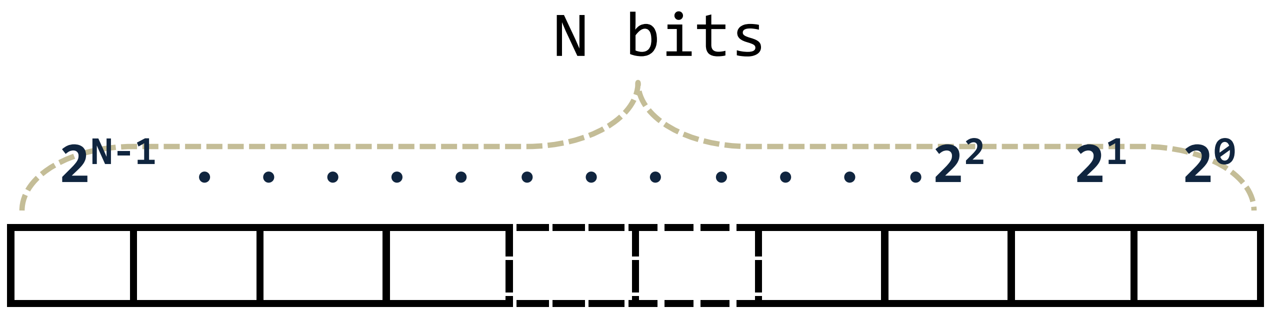
$224_{10} = 11100000_2$

Converting binary to decimal

$(a_n \ldots a_3 a_2 a_1 a_0)_2 = a_n x 2^n + \ldots + a_3 x 2^3 + a_2 x 2^2 + a_1 x 2^1 + a_0 x 2^0$

Data Representations

Binary



Each cell 0 or 1 = 2 states

- 2^N quantities can be represented by N bits (number of different combinations)
- Covers numbers between 0 and 2^N - 1

Data Representations

Binary

Example:

3 bits

	2^2	2^1	2^0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

$\boxed{2^3 = 8}$

$\boxed{0}$

$\boxed{1}$

$\boxed{2^3-1 = 7}$

Data Representations

Binary

Binary system is used in computers

- **Binary devices are easy to design (circuits)**
- **Two voltage levels (on, off)**

Data Representations

Binary addition :

$\begin{array}{r} 0 \\ +0 \\ \hline 0 \end{array}$	$\begin{array}{r} 0 \\ +1 \\ \hline 1 \end{array}$	$\begin{array}{r} 1 \\ +0 \\ \hline 1 \end{array}$	$\begin{array}{r} 1 \\ +1 \\ \hline 10 \end{array}$

Example:

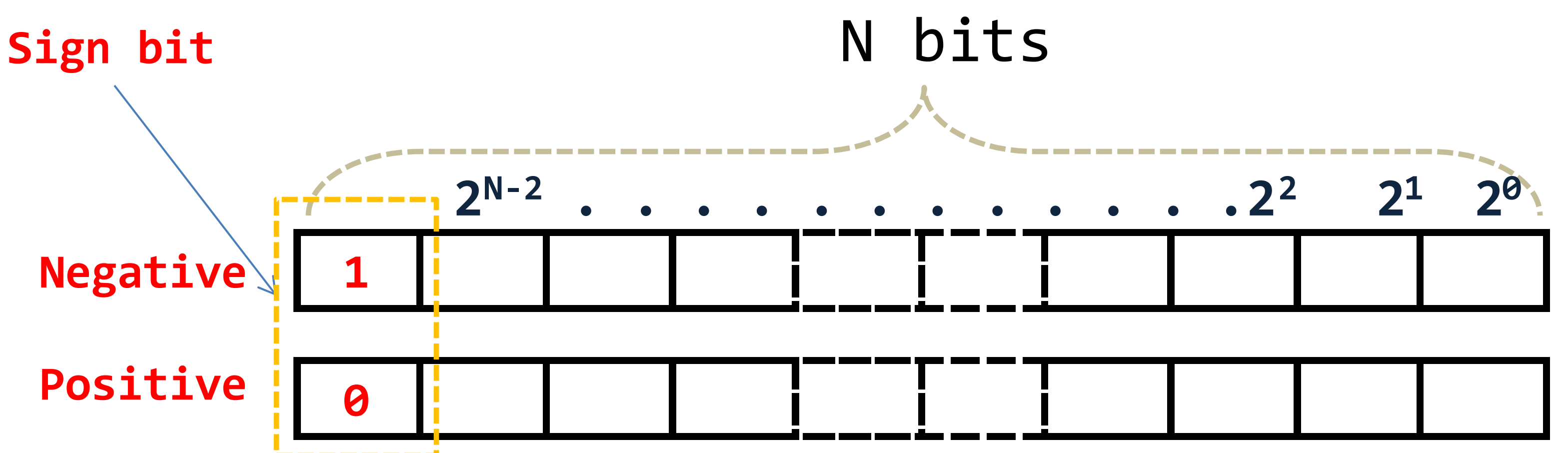
```
 carry out
  01000

+101101
 001010
-------
 110111
```

Data Representations

Binary, negative numbers:

- Sign-magnitude representation
  1. uses the most significant bit(sign bit)
  2. <u>zero</u> for positive numbers and <u>one</u> for negative numbers



$$-(2^{N-1} - 1) < \text{numbers} < +(2^{N-1} - 1)$$

Data Representations

Binary, negative numbers:

- Sign-magnitude representation

Example:

4 bits

1	1	1	1	-7
0	1	1	1	+7

$$-(2^{4-1} - 1) \leq \text{numbers} \leq +(2^{4-1} - 1)$$

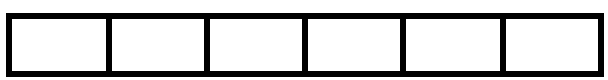
$$-7 \leq \text{numbers} \leq +7$$

Data Representations

Binary, negative numbers:

- Sign-magnitude representation
  1. Two zeros, ± 0
  2. Difficult to perform addition and subtraction

Question: what is sign-magnitude representation of ±10 in 6 bits?



±10 ?

Data Representations

Binary, negative numbers:

- Two's-complement notation

  negative number is obtained by inverting each bit of the number(1->0 and 0->1) and adding 1 to the result

Example:

$$24_2=11000$$
$$00111$$
$$+1$$
$$-24_2=1000$$

Data Representations

Binary, negative numbers:

- Two's-complement notation
  negative number is obtained by inverting each bit of the number(1->0 and 0->1) and adding 1 to the result

Example:

$24_{10}=11000_2$ $-24_{10}=01000_2$

$11000_2+(-11000_2) = 0$

$11000 + 01000 = 100000$

Data Representations

- Two's-complement notation
  1. The sign of a number is determined by examining the high bit, 1 for negative numbers and 0 positive numbers
  2. Retrieving the original number by applying twice two's-complement technique
  3. One representation of 0
  4. No need to design a new hardware for number subtraction e.g. 3-13 = 3 + (-13)
  5. Can represent values from $-(2^{N-1})$ to $(2^{N-1}-1)$

Example: use the 8-bit representation to obtain 3-13

Data Representations

Example: use the 8-bit representation to obtain 3-13 = -10

$3_2 = 00000011$

$13_2 = 00001101$

$(-13)_{10} = (11110010+1)_2=11110011_2$

$3_{10} - 13_{10} =(00000011+11110011)_2=11110110_2= (-10)_{10}$

$(-10)_{10} = 11110110_2$

$-(-10)_{10} = (00001001 + 1)_2 = (00001010)_2 = +10_{10}$

Data Representations

Binary to Hexadecimal and vice versa

1. Divide the binary number into sets of 4 digits starting from right to left

2. Use the hexadecimal equivalent of every 4 digit binary number from right to left

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

Data Representations

Binary to Hexadecimal and vice versa

Example :
1101010011010

			1	1	0	1	0	1	0	0	1	1	0	1	0

0	0	0	1	1	0	1	0	1	0	0	1	1	0	1	0

1 A 9 A

1A9A

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

Data Representations

Binary to Hexadecimal and vice versa

Example :
FBC9

F				B				C				9			
1	1	1	1	1	0	1	1	1	1	0	0	1	0	0	1

1111101111001001

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

Data Representations

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

Example: memory

```
11B4:0100  0F 00 B9 8A FF F3 AE 47-61 03 1F 8B C3 48 12 B1   ........Ga.
11B4:0110  04 8B C6 F7 0A 0A D0 D3-48 DA 2B D0 34 00 A3 11   .........H.
11B4:0120  00 DB D2 D3 E0 03 F0 8E-DA 8B C7 16 C2 B6 01 16   ..........
11B4:0130  C0 16 F8 8E C2 AC 8A D0-00 00 4E AD 8B C8 46 8A   ..........
11B4:0140  C2 24 FE 3C B0 75 05 AC-F3 AA A0 0A EB 06 3C B2   .$.<.u....
11B4:0150  75 6D 6D 13 A8 01 50 14-74 B1 BE 32 01 8D 8B 1E   umm...P.t.
11B4:0160  8E FC 12 A8 33 D2 29 E3-13 8B C2 03 C3 69 02 00   ....3.)...
11B4:0170  0B F8 83 FF FF 74 11 26-01 1D E2 F3 81 00 94 FA   .....t.&..
```

Data Representations

Basic concepts

1 bit : Basic unit of information in a computer (0 or 1)

1 byte: Eight bits

Word : A contiguous group of bytes.

- The precise size of a word is machine-dependent.
- e.g.

  16 bits, 1 word = 2 bytes

  32 bits, 1 word = 4 bytes

  64 bits, 1 word = 8 bytes

Data Representations

Floating Point Representation

Central Processing Unit (CPU):

- **A**rithmetic **L**ogic **U**nit (ALU)

Performs integer arithmetic operations such as addition, subtraction, and logic operations such as AND, OR, XOR, NOT and so on.

- **F**loating **P**oint **U**nit (FLU/FPU)

performs floating point operations.

- **R**egisters

  Local fast memories

- **C**ontrol **U**nit

Data Representations

Floating Point Representation

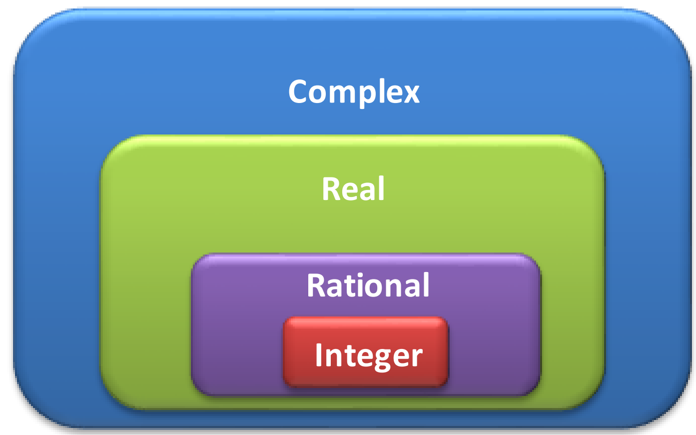
- Complex numbers

  3-5i
- Real numbers

  √5
- Rational numbers

  4/3
- Integer numbers

  -5, 0, 3



Extremely large and small values:

Sun/Mass : 1.989×10^{30} kg

Electron/Mass : 9.11×10^{-31} kg

Floating point → Fraction × Base ^{power}

Data Representations

Floating Point Representation

Decimal fractions to Binary conversion

Example :

$3.14_{10} = 3 + 0.14 = 3 + 14/100 = 3 + (10+4)/100 = 3 + 10/100 + 4/100$
$= 3 + 1/10 + 4/100 = 3\times10^{0} + 1\times10^{-1} + 4\times10^{-2}$

$$3.14_{10} = 3\times10^{0} + 1\times10^{-1} + 4\times10^{-2}$$

$$512.123 = 5\times10^{2} + 1\times10^{1} + 2\times10^{0} + 1\times10^{-1} + 2\times10^{-2} + 3\times10^{-3}$$

Data Representations

Floating Point Representation

Decimal fractions to Binary conversion

Example :

$$110.11_2 = (1\times2^2 + 1\times2^1 + 0\times2^0 + 1\times2^{-1} + 1\times2^{-2})_{10}$$
$$= (4 + 2 + 0.5 + 0.25)_{10}$$
$$= 6.75_{10}$$

$$0.75_{10} = (0.11)_2$$
$$0.75 = 0.75 \times 2 = 1.5$$
$$= 0.5 \times 2 = 1.0$$

Data Representations

Floating Point Representation

Decimal fractions to Binary conversion

Example :

$0.6_{10} = (0.100110011001\ldots\ldots)_2$

$0.6 = 0.6 \times 2 = 1.2$

$= 0.2 \times 2 = 0.4$

$= 0.4 \times 2 = 0.8$

$= 0.8 \times 2 = 1.6$

$= 0.6 \times 2 = 1.2$

. . . .

$0.125_{10} = (0.001)_2$

$= 0.125 \times 2 = 0.25$

$= 0.25 \times 2 = 0.5$

$= 0.5 \times 2 = 1.0$

Data Representations

Floating Point Representation

Normalized Floating Point Numbers

Floating point(FP)

Base =2

$$(-1)^S \times 1.F \times 2^E$$

S = sign (0-positive number , 1-negetive number)

F = fraction (fixed point number)

called mantissa or significand

E = exponent (positive or negative integer)

Data Representations

Floating Point Representation

Normalized Floating Point Numbers

Floating point(FP)

Base =2

$$(-1)^S \times 1.F \times 2^E$$

$$100101.101_2 = (-1)^0 \text{ x } 1.00101101_2 \text{ x } 2^5$$

S = 0, F = 00101101, E = 5

Data Representations

Floating Point Representation

Normalized Floating Point Numbers

Floating point(FP)

Base =2

$$(-1)^S \times 1.F \times 2^E$$

-0.001_2 = ?

Obtain S, F, and E value ?

Data Representations

Floating Point Representation

Normalized Floating Point Numbers

Floating point(FP)

Base =2

$$(-1)^S \times 1.F \times 2^E$$

$110000.10110011001100110 1_2 = ?$

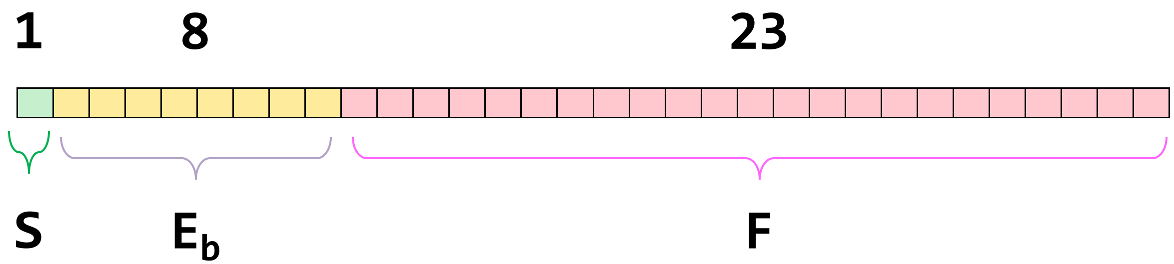
Obtain S, F, and E value ?

Data Representations

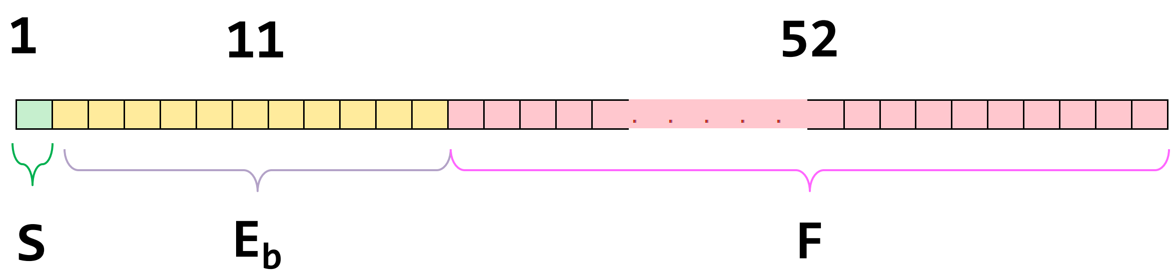
Floating Point Representation

IEEE 754 standard

Single precision numbers (32bits)



Double precision numbers (64bits)

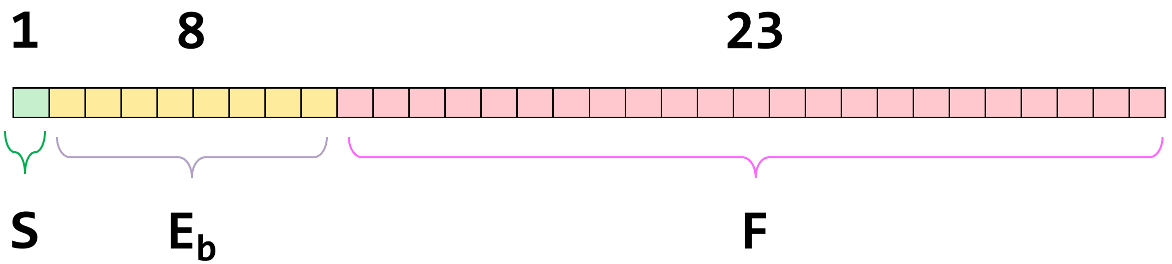


Data Representations

Floating Point Representation

IEEE 754 standard (Part F)

Single precision numbers



$1.00000000000000000000000$

$1.11111111111111111111111$

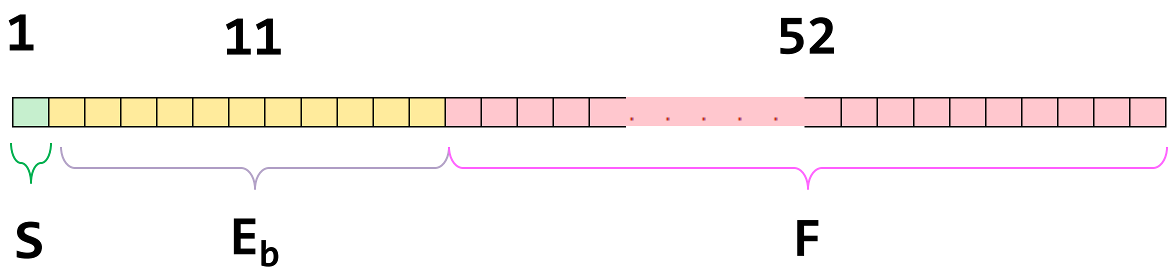
$$1 \leq 1.F \leq (2-2^{-23})$$

$$1 \leq 1.F < 2$$

Data Representations

Floating Point Representation

IEEE 754 standard (Part F)



Double precision numbers

$$1 \leq 1.F \leq (2-2^{-52})$$

$$1 \leq 1.F < 2$$

Data Representations

Floating Point Representation

IEEE 754 standard (Part E)

Single precision numbers

8

bias 127

$E_b = 127 + E$

Double precision numbers

11

bias 1023

$E_b = 1023 + E$

Data Representations

Floating Point Representation

Examples:

$-0.75_{10} = -0.11_2 = -1.1_2 \times 2^{-1}$

$2 \times 0.75 = 1.50$
$2 \times 0.50 = 1.00$
$0.75_{10} = 0.11_2$

$S = 1$
$F = 10000000000000000000000_2$
$E_b = 127 + E = 127 - 1 = 126 = 01111110_2$

1	0	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$S = 1$
$F = 1000000000000000000000000000000000000000000000000000_2$
$E_b = 1023 + E = 1023 - 1 = 1022 = 01111111110_2$

1	0	1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	. . . . .	0	0	0	0	0	0	0	0	0	0