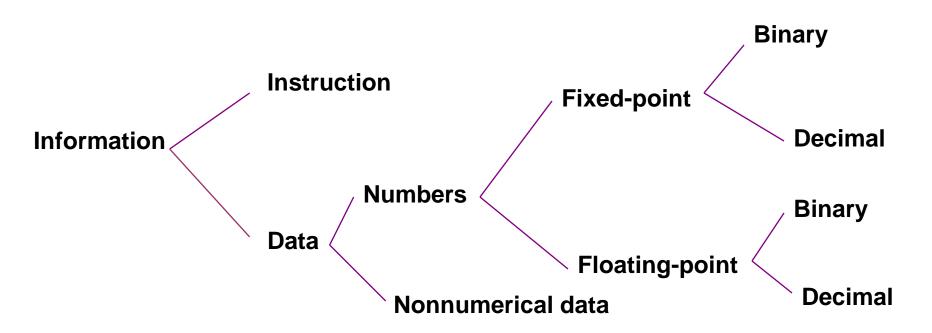
vaniable stonge vaniable stonge representation (Floating Point)

Lecture - 3

The Basic Information Types



1

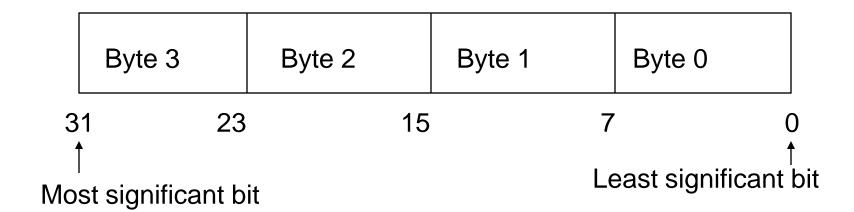
Word Length

Bits	Name	Description
1	Bit	Logic variable, flag.
8	Byte	Smallest addressable memory item, Binary-coded decimal digit pair.
16	Halfword	Short fixed-point number. Short address, Short instruction.
32	Word	Fixed or floating point number, Memory address, Instruction.
64	Double word	Long instruction, Double-precision floating point number.

Some Instruction formats of the Motorola 680X0 microprocessor series



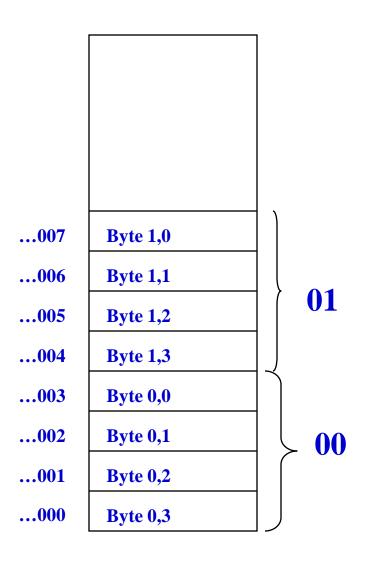
Byte Storage



Indexing convention for the bits and bytes of a word

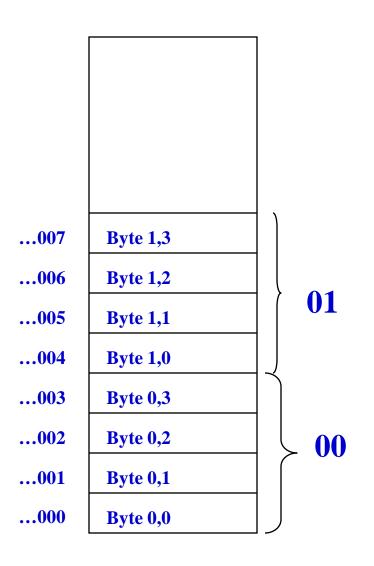


Big-Endian Storage Method



- Suppose a word W_i is represented as B_{i, 3} B_{i, 2} B_{i, 1} B_{i, 0}
- Most significant byte B_{i, 3} of W_i is assigned the lowest address and the least significant byte B_{i, 0} is assigned the highest address.
- It is so named because it assigns highest address to byte 0 of a word.





- Suppose a word W_i is represented as B_{i, 0} B_{i, 1} B_{i, 2} B_{i, 3}
- Most significant byte B_{i, 3} of W_i is assigned the highest address and the least significant byte B_{i, 0} is assigned the lowest address.
- It is so named because it assigns lowest address to byte 0 of a word.

M

Tags

- It is a technique of determining the type of a word.
- This is done by associating with each information word a group of bits, called a tag, that identifies the word's type.

Advantages:

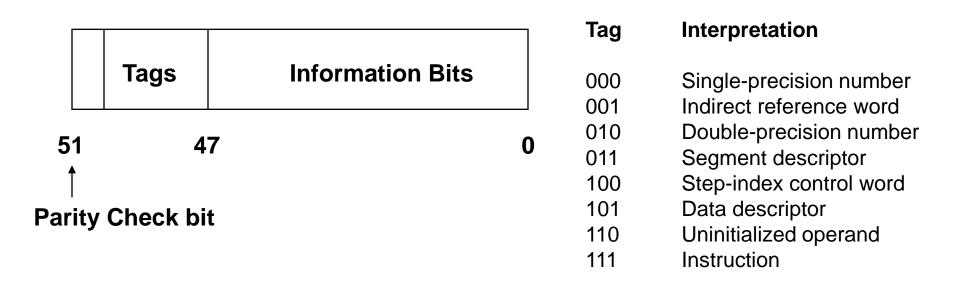
- It simplifies instruction specifications.
- Tag inspection permits the hardware to check for software errors.

Disadvantages:

- They increase memory size.
- Add system hardware costs without increasing computing performance.



Tags



Tagged-word format of the B6500/7500 Series

м

Error Detection and Correction

- The parity bit is appended to an *n*-bit word $X = (x_0, x_i, \dots, x_{n-1})$ to form the (n+1)-bit word $X^* = (x_0, x_i, \dots, x_{n-1}, c_0)$.
- Bit c_0 is assigned the value 0 or 1 that makes the number of ones in X^* even.

Even Parity:

 $c_0 = x_o \oplus x_i \oplus \ldots \oplus x_{n-1}$

Odd Parity:

 $c_0' = x_0 \oplus x_i \oplus \dots \oplus x_{n-1}$

×

Error Detection and Correction

- Fig. 3.20 Page 166
- Let, B receive the word $X' = (x'_0, x'_i, \dots, x'_{n-1}, c'_0)$.
- Then, $c_0^* = x'_0 \oplus x'_i \oplus \dots \oplus x'_{n-1}$
- If, $c'_{\Omega} \neq c^*_{\Omega}$, the received information contains an error.
- If, $c'_0 = c^*_0$, there is no single-bit error.

Factors to select a number representation

- The number types to be represented; for example, integers or real numbers.
- The range of values (number magnitudes) likely to be encountered.
- The precision of numbers, which refers to the maximum accuracy of the representation.
- The cost of the hardware required to store and process the numbers.

м

Fixed Point Binary Numbers

- The unsigned binary fixed-point format takes the form $b_N...b_1b_0.b_{-1}b_{-2}...b_M$ where each b_i is 0 or 1, representing the number $\sum b_i 2^i$ where M <= i <= N.
- This is a positional notation in which each digit has a fixed-weight according to it's position relative to the binary point.

The signed binary number is represented as

$$X_{n-1}X_{n-2}X_{n-3}...X_2X_1X_0$$

| Sign Magnitude

■ Using n-bit number format we can represent all integers N with magnitude |N| in the range $0 <= |N| <= 2^{n-1}$.

M

Signed Numbers

3 types of representations of a signed number:

Sign magnitude

+75=01001011

-75=11001011

1's complement

+75= 01001011

-75 = 10110100

2's complement

+75= 01001011

-75 = 10110101



Signed Numbers

- Sign magnitude number employ the positional notation for magnitude and simply change the sign bit to represent + or -.
- In both complement code x_{n-1} retains the role as sign bit, but remaining bits no longer form a simple positional code when the number is negative.
- The advantage of the 2'scomplement code is that subtraction can be performed by logical complementation and addition only. 2's complement addition and subtraction can be implemented by a simple adder for unsigned numbers.
- Multiplication and division are more difficult to implement of 2's complement code.

w

Signed Numbers

Decimal Representation	Sign Magnitude	1's Complement	2's Complement
+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	1000	1111	0000
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001

.

Decimal Codes

BCD (Binary coded decimal)

- In BCD format each digit d_i of a decimal number is denoted by a 4-bit equivalent b_{i,3}b_{i,2}b_{i,1}b_{i,0}.
- BCD is a weighted (positional) number code where each b_{i,j} has the weight 10ⁱ2^j.

ASCII

The 8-bit ASCII code represents the 10 decimal digits by a 4-bit BCD field and the remaining 4-bits of the ASCII word have no numerical significance.



Decimal Codes

Excess-Three Code

- The excess-three code can be formed by adding 0011₂ to the corresponding BCD number.
- The advantage of the excess-three code is that it may be processed using the same logic used for binary codes.
- Some arithmetic operations are difficult to implement using excess-three code, mainly because it is a nonweighted code.

.

Decimal Codes

Excess-Three Code

■ If two excess-three numbers are added like binary numbers, the required decimal carry is automatically generated from the high-order bits. The sum must be corrected by adding +3.

```
1000 = 5
+ 1100 = 9
Carry 1 0100 Binary sum
+ 0011 Correction
0111 = 4 Excess-three sum
```

.

Decimal Codes

Two-out-of-Five

- Each decimal digit is represented by a 5-bit sequence containing two 1s and three 0s. There are exactly 10 distinct sequence of this type.
- It is single-error detecting code, since changing any one bit results in a sequence that does not correspond to a valid code word.
- It uses 5 rather than 4 bits per decimal digit.

м

Decimal Codes

Decimal Digit	BCD	ASCII	Excess-three	Two-out-of-five
0	0000	00110000	0011	11000
1	0001	00110001	0100	00011
2	0010	00110010	0101	00101
3	0011	00110011	0110	00110
4	0100	00110101	0111	01001
5	0101	00110101	1000	01010
6	0110	00110110	1001	01100
7	0111	00110111	1010	10001
8	1000	00111000	1011	10010
9	1001	00111001	1100	10100

٠,

Decimal Codes

Hexadecimal Numbers:

- Uses 16 digits, consisting of decimal digits 0, 1, ..., 9 augmented by the six digits A, B, C, D, E and F, which have the numerical values 10, 11, 12, 13, 14 and 15, respectively.
- Useful for representing long binary numbers.

M

Need for Floating-Point Number

- The range of number represented by a fixed-point number is insufficient for many applications, particularly, when very large and very small numbers are required.
- Scientific notation allows to represent such numbers using relatively few digits.
- For example, it is easier to write 1.0 x 10¹⁸ than
 1 000 000 000 000 000

Basic Format of Floating-PointNumber

- Three numbers are associate with a floating-point number: a mantissa *M*, an exponent *E* and a base *B*. The mantissa *M* is also referred to as the significand or fraction in the literature.
- A real number is represented as $M \times B^E$.
- Example: 1.0×10^{18} where 1.0 = mantissa, 10 = base and 18 = exponent.

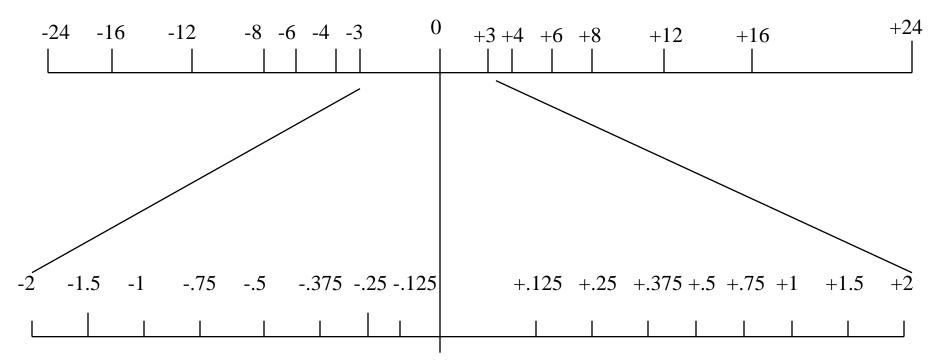
Representation of Floating-Point Number

- A floating point number is represented as a word (M,E) consisting of a pair of signed fixed-point numbers: M, which is usually a fraction or integer and E, which is an integer.
- Since, B is constant, it is not stored but is simply built into the circuit that process the number.
- The number of digits in M determines the precision of (M,E); B and E determines the range.

М.

Example of Floating-Point Number

- Let M and E are both 3-bit, sign magnitude numbers. Then M and E each can assume the values ±0, ±1, ±2, ±3. Let B=2.
- All binary words of the form (M, E) = (x00, xxx) represent 0, where x denotes either 0 or 1.



The real numbers representable by a hypothetical 6-bit, floating-point format

м

Floating-Point Number

- Smallest nonzero positive number is (001, 111), denoting $1 \times 2^{-3} = 0.125$
- \blacksquare (101, 111), denoting -1 \times 2⁻³ = -0.125
- Smallest nonzero negative number is (111, 111), denoting $-3 \times 2^{-3} = -0.375$
- Largest positive number is (011, 011), denoting 3 x 2³ = 24
- Largest negative number is (111, 011), denoting -3 x 2⁻³ =
 -24

M

Floating-Point Number

- The floating point representation of most real numbers is only approximate. For example, 1.25 is approximated by (011,101) representing 1.5 or by either (001, 000) or (001, 100) representing 1.0.
- The result of most calculations with floating point arithmetic only approximate the correct result. For example, the exact result of the addition (011,001) + (011,010) = 18 which is not representable. The closest representative number of 18 is 16 (010, 011).
- Include a few extra mantissa digits termed guard digits to reduce approximation errors; the guard digits are removed automatically from the final results.

м

Normalization

- The same number can be represented in many ways. For example: 1.0×10^{18} , 0.1×10^{19} , 1000000×10^{12} etc.
- It is desirable to have a unique or normal form for each representable number in a floating point system.
- A binary number is normalized when it has one non-zero digit to the left of the decimal point.
- Example: for IEEE 754 the normalized form 1.M.
- An un-normalized number is normalized by shifting the mantissa to the right or left and appropriately incrementing or decrementing the exponent.



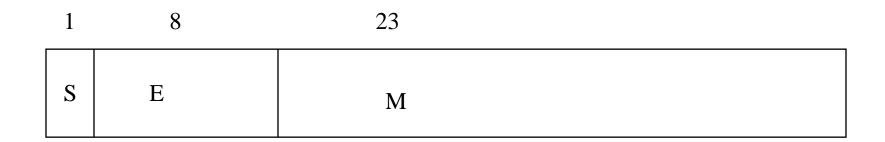
Normalization

Advantages:

- It simplifies the exchange of data.
- It simplifies the floating point arithmetic algorithm
- It increases the accuracy of the numbers that can be stored in a word, since the unnecessary leading 0s are replaced by real digits to the right of the binary point.



IEEE 754 Floating Point Number



IEEE 754 standard 32-bit floating-point number format

- S= 1 bit sign representation
- \blacksquare E = 8 bit excess-127. The actual exponent is E-127.
- M=23 bit mantissa [fraction part of sign-magnitude binary significand with hidden bit]

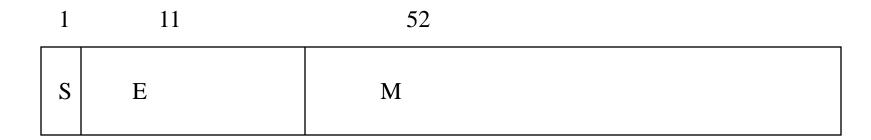
М

IEEE 754 Floating Point Number

- A real number $N = (-1)^{s}2^{E-127}(1.M)$ where, 0 < E < 255
- N=-1.5 is represented as1 01111111 10000000000.....0



IEEE 754 Floating Point Number



IEEE 754 standard 64-bit floating-point number format

■ A real number $N = (-1)^{s}2^{E-1023}(1.M)$ where, 0 < E < 2047

м

Biasing

If we use 2's complement or other notation in which negative exponents have 1 in MSB, a negative number will look like a big number. For example, 1.0×2^{-1} is represented as

```
0111111110000000000000000......
```

```
1.0 \times 2^{1} is represented as,
```

- It is desirable to represent most negative exponent as 00...000 and most positive as 11....111. This convention is called biased notation.
- The exponents are encoded in excess-*K* code where the exponent field *E* contains an integer that is the desired exponent value plus *K*. The quantity *K* is called **bias** and exponent encoded in this way is called a **biased exponent**.
- -1+127=126=011111110 and +1+127=128=10000000

M

Biasing

Exponent E	Unsigned value	Bias 127	Bias 128
11111	255	+128	+127
11110	254	+127	+126
10001	129	+2	+1
10000	128	+1	0
01111	127	0	-1
01110	126	-1	-2
00001	1	-126	-127
00000	0	-127	-128

8-bit biased exponent with bias=127 (excess-127) and bias = 128 (excess-128)

Converting from Decimal to Binary Floating Point

- What is the binary representation for the single-precision floating point number that corresponds to $X = -12.25_{10}$?
- What is the normalized binary representation for the number?

$$-12.25_{10} = -1100.01_2 = -1.10001_2 \times 2^3$$

What are the sign, stored exponent, and normalized mantissa?

S = 1 (since the number is negative)

$$E = 3 + 127 = 130 = 128 + 2 = 10000010_{2}$$

Converting from Decimal to Binary Floating Point

Overflow:

A situation in which a positive exponent becomes too large to fit in the exponent field.

Underflow:

A situation in which a negative exponent becomes too large to fit in the exponent field.