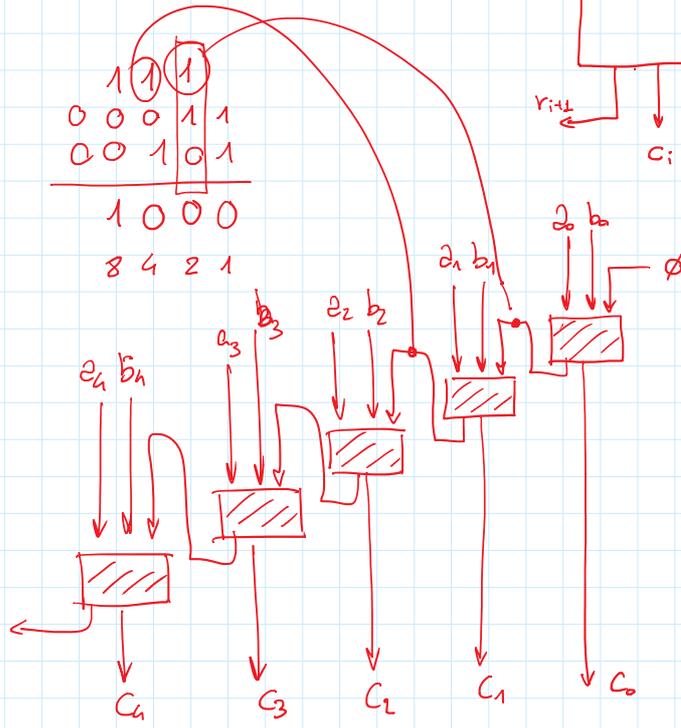


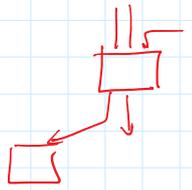
Rappresentazione numeri in binario

a
b

| | | | | |
|-------|---|---|---|---|
| 1 | 1 | 1 | | |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| <hr/> | | | | |
| 1 | 0 | 0 | 0 | |
| 8 | 4 | 2 | 1 | |



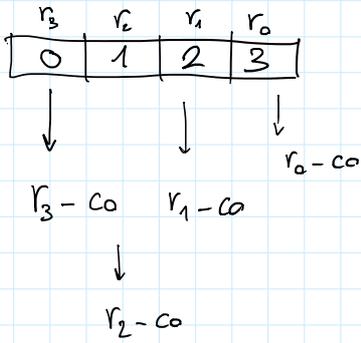
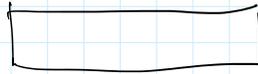
rete combinatoria



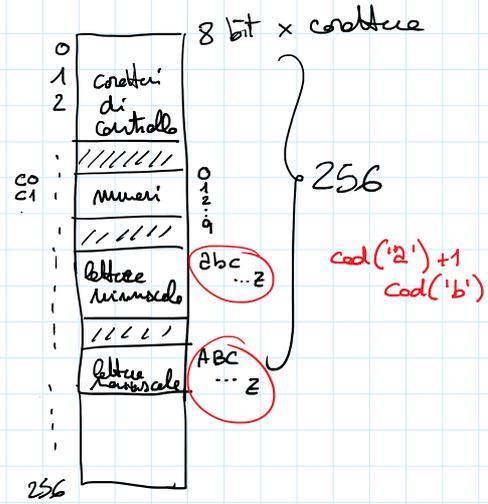
Rappresentazione dei caratteri (stringhe)

char c = 'a';

bit



codifica ASCII



Numeri in virgola mobile

IEEE (754)



123,456

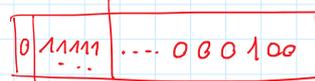
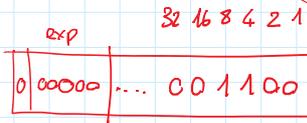
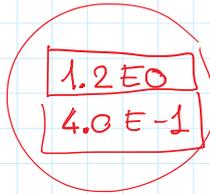
1.23456 E 2

$1.23456 \times 100 = 123.456$

Singola precisione
doppia

| | | |
|---|---|---------|
| 0 | 2 | 123456 |
| 0 | 3 | .123456 |

$$\begin{array}{r} .1.2 + \\ .0.4 \\ \hline 1.6 \end{array}$$



- 1) 0000001
- 2) 11111110
- 3) 1 111

3 pezzi
allineamento mantisse

1.2
4

Soluzione
1.6



Operazioni particolari (+ veloci di quelle generali)

$$x * y = z$$

$$123 \times 10^2$$

$$\boxed{123} \mid 00$$

$$123 \overset{10^2}{/} 100 = 123$$

← 2 pos

$$38 \mid 2$$

$$38_{10} = \begin{array}{cccccccc} 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \end{array}_2$$

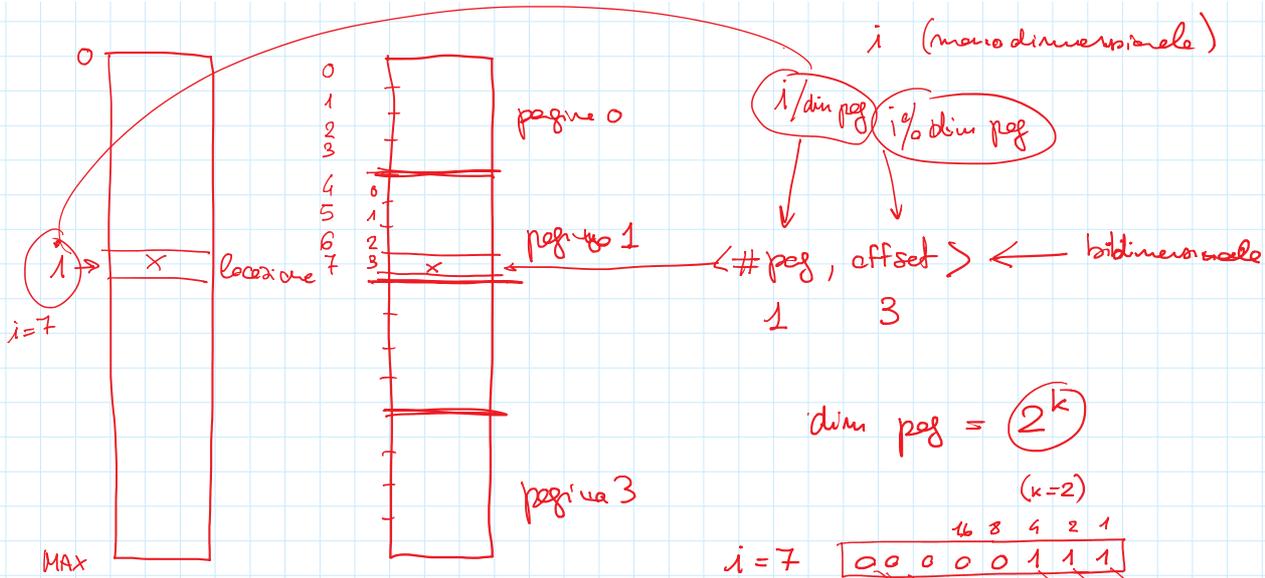
SHIFT (1)

$$000000010_2$$

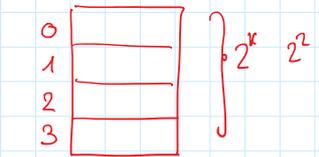
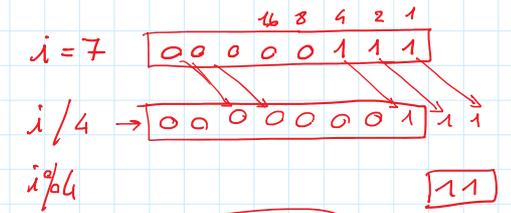
$$38 \mid 4$$

$$\begin{array}{cccccccc} 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \end{array} \mid 0$$

$$\begin{array}{cccccccc} & & & & 8 & & & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{array} \mid 10$$

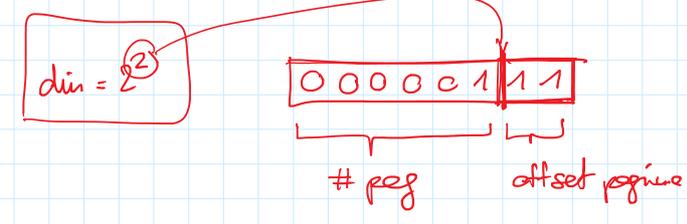


$dim\ pag = 2^k$
($k=2$)



#bit?
x rappresenta

$$\lg_2(4) = \lg_2(2^2) = 2$$



```
for(int i=0; i<N; i++) { ... }
```

$N = 2^k$
 $\nexists k \mid N = 2^k$

| | | | | | |
|-------|-------|-------|-------|-------|-------|
| $k=2$ | b_3 | b_2 | b_1 | b_0 | |
| | 0 | 1 | 2 | 3 | |
| 0 | 0 | 0 | 0 | 0 | $i=0$ |
| 1 | 0 | 0 | 0 | 1 | $i=1$ |
| 2 | 0 | 0 | 1 | 0 | $i=2$ |
| 3 | 0 | 0 | 1 | 1 | $i=3$ |
| 4 | 0 | 1 | 0 | 0 | $i=4$ |

$N=2^k$
 $< 2^k$
 fino a che $b_k = 0$

```
for(int i=0; i<3; i++)
```

| | | | | |
|-------|-------|-------|-------|------------|
| b_3 | b_2 | b_1 | b_0 | |
| 0 | 0 | 0 | 0 | $i=0$ ✓ |
| 0 | 0 | 0 | 1 | $i=1$ ✓ |
| 0 | 0 | 1 | 0 | $i=2$ ✓ |
| 0 | 0 | 1 | 1 | $i=3$ ✗ No |

devo arrivare
 fino a 3
 escluso
 $b_3 \ b_2 \ b_1 \ b_0$
 $0 \ 0 \ 1 \ 1$

$x_3 \ x_2 \ x_1 \ x_0$ iterazione corrente ($i++$)

mi fermo a $b_i = x_i \ \forall i \in [0, 3]$

```
for(int i=0; i<N; i++) {
    me[i] = mb[i];
}
```

```
for(int i=N-1; i>=-1; i--)
    me[i] = mb[i];
```

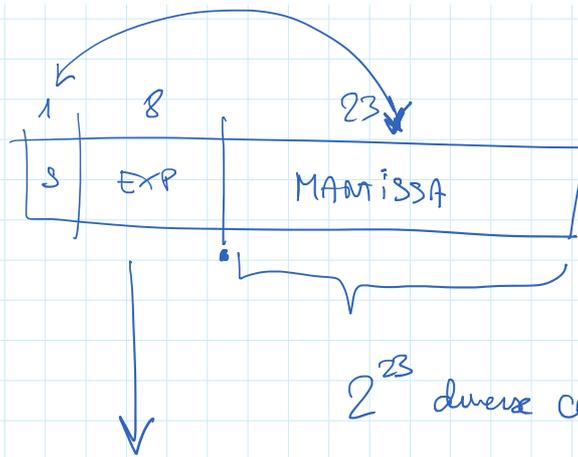
```
for(int i=0; i<3; i++)
```

```
for(int i=2; i>=-1; i--)
```

| | |
|--------|-----------|
| $i=-3$ | $me[i+1]$ |
| $i=-2$ | |
| $i=-1$ | |
| $i=0$ | ✗ |

| | |
|--------|------|
| $i=2$ | ✓ |
| $i=1$ | ✓ |
| $i=0$ | ✓ |
| $i=-1$ | ✗ No |

| | | | |
|-------|-------|-------|-------|
| b_3 | b_2 | b_1 | b_0 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |



$$[0, 2^{23} - 1]$$

2^8 combinations

$2^8 - 1$ senza segno

$$(-2^7) \dots (2^7 - 1)$$

11

mercoledì 21 settembre 2016 13:59

