Appunti di Architettura

Architettura degli Elaboratori (prof. Fersini) - CdL Informatica Unimib - 23/24

Federico Zotti

05 Mar 2024

Indice

1 Sistemi Numerici

1.1 Introduzione

I calcolatori utilizzano, a differenza di noi, il **sistema binario**. Questo perché la corrente elettrica può rappresentare solo due stati: acceso (*1*) e spento (*0*).

Sono stati definiti degli **standard di codifica**: regole che vengono utilizzate nella rappresentazione dei dati in formato binario.

Con il termine **bit** definiamo l'**unità di misura dell'informazione**. Combinando tra loro più bit si ottengono strutture più complesse. In particolare:

- Nybble (o nibble): 4 bit
- Byte: 8 bit
- Halfword: 16 bit
- Word: 32 bit
- Doubleword: 64 bit

Dati k bit, il numero di configurazioni ottenibili è pari a 2^k.

Una **rappresentazione** è un modo per descrivere un'entità. Bisogna distinguere l'entità (o valore) e la sua rappresentazione.

1.2 Vari sistemi numerici

Il **sistema numerico decimale**:

- Usa 10 cifre
- È un **sistema posizionale**: ogni cifra assume un valore diverso a seconda della posizione che occupa all'interno del numero

Il **sistema romano** invece non è posizionale (il valore della cifra non dipende dalla sua posizione).

Nei sistemi numerici posizionali un valore numerico N è caratterizzato dalla seguente rappresentazione:

$$N = d_{n-1}\, d_{n-2} \dots d_1\, d_0\,,\, d_{-1} \dots d_{-m}$$

$$N = d_{n-1} \cdot r^{n-1} + \dots + d_0 \cdot r^0 + d_{-1} \cdot r^{-1} + \dots d_{-m} \cdot r^{r-m}$$

$$N = \sum_{i=-m}^{n-1} d_i \cdot r^i$$

Dove: - d è la singola cifra - r la radice o base del sistema - n numero di cifre della parte intera (sinistra della virgola) - m numero di cifre della parte frazionaria (destra della virgola) - N è la rappresentazione del numero

1.2.1 Sistema Binario

Un **byte** è una sequenza di 8 bit consecutivi. Il bit *"più a sinistra"* è chiamato **MSB** (most significant bit) ovvero il bit che rappresenta la cifra con il valore più grande. Il bit *"più a destra"* è chiamato **LSB** (least significant bit) ovvero il bit che rappresenta la cifra con il valore più piccolo.

2 Rappresentazione di numeri interi con segno

2.1 Operazioni aritmetiche

2.1.1 Somma

La somma di due sequenze di bit è la somma tra i bit di pari ordine:

- $0 + 0 = 0$
- $0 + 1 = 1$
- $1 + 0 = 1$
- $1 + 1 = 0$ con riporto 1 sul bit di ordine superiore
- $1 + 1 + 1 = 1$ con riporto 1 sul bit di ordine superiore

Dunque la somma è definita su 3 elementi:

- I due addendi
- Il riporto (carry)

2.1.2 Sottrazione

La sottrazione di due sequenze di bit è la sottrazione tra i bit di pari ordine:

- $0 - 0 = 0$
- $1 - 0 = 1$
- $1 - 1 = 0$
- $0 - 1 = 1$ con prestito 1 dal bit di ordine superiore

Dunque la sottrazione è definita su 3 elementi:

- Minuendo e sottraendo
- Il prestito (borrow)

2.2 Modulo e Segno

Supponiamo di avere a disposizione 1 Byte per rappresentare numeri sia positivi che negativi.

Con il metodo del **modulo e segno** utilizzeremo:

- I primi 7 bit per il valore assoluto del numero
- Il bit più a sinistra *(MSB)* per indicare il segno (1 se il numero è negativo, 0 se è positivo)

Con n bit totali si possono rappresentare i numeri interi nell'intervallo

$$\left[-(2^{n-1}-1), +(2^{n-1}-1)\right]_{10}$$

I problemi di questa rappresentazione sono che esistono due diverse rappresentazioni dello 0 e che un bit viene "speso" solo per il segno.

2.2.1 Somma

Confronto i bit di segno dei due numeri:

- Se i bit di segno sono uguali:

  - Il bit di segno risultante sarà il bit di segno dei due addendi
  - Eseguo la somma bit a bit (a meno di overflow)

- Se i bit di segno sono diversi:
  - Confronto i valori assoluti dei due addendi
  - Il bit si segno risultate sarà il bit di segno dell'addendo con valore assoluto
  - Eseguo la somma bit a bit

2.2.2 Sottrazione

> Dalle slide #todo-uni

2.3 Complemento a 1

Questo metodo si basa sull'**operazione di complemento**. Con complemento si intende l'operazione che associa ad un bit il suo opposto.

Il metodo del **complemento a 1** è molto semplice:

- Se il numero da codificare è **positivo**, lo si converte in binario con il metodo tradizionale
- Se il numero è **negativo**, basta convertire in binario il suo modulo e quindi eseguire l'operazione di complemento sulla codifica binaria effettuata

Anche in questo caso sussiste il problema delle due diverse rappresentazioni dello 0.

2.4 Complemento a 2

Il complemento a 2 è un altro metodo di codifica usato per rappresentare i numeri interi sia positivi che negativi. È basato sul complemento a 1.

A differenza del complemento a 1 i numeri negativi dopo aver complementato il numero vengono incrementati di 1.

Dati n bit si possono rappresentare i numeri interi nell'intervallo

$$\left[-(2^{n-1}), +(2^{n-1}-1)\right]_{10}$$

2.4.1 Conversione da CA2 a decimale

Se il numero è positivo (MSB = 0), si converte in base decimale usando il numero binario puro. Se il numero è negativo (MSB = 1), si applica l'operazione di CA2 a questo valore ottenendo la rappresentazione del corrispondente positivo, si converte il risultato come numero in binario puro e si aggiunge il segno meno.

2.4.2 Somma

> Dalle slide #todo-uni

2.4.3 Sottrazione

> Dalle slide #todo-uni

2.5 Shift

> Dalle slide #todo-uni

2.6 Rappresentazione Eccesso 2^{n-1}

> Dalle slide #todo-uni