Rappresentazione di Interi con segno

Fri, 11 Mar

Per gestire i casi di **underflow**, in cui da una sottrazione si ottiene un *numero negativo*, è sorta la necessita di creare una **convenzione** per **rappresentare** i **numeri negativi**.

Modulo e segno (MS)

Nel metodo di rappresentazione chiamato **modulo e segno**, prendendo in considerazione una serie di n bit, la si divide in *due parti*:

- Il **Most Significant Bit**

  Il MSB indica il segno del numero rappresentato: **0** se **positivo** e **1** se **negativo**.
- I restanti **n-1 bit**

  I restanti bit indicano il **valore assoluto** del numero rappresentato.

Problemi del MS

La rappresnetazione modulo e segno presenta principalmente **due problemi**:

- Una **minore capacità** di rappresentazione

  Uno degli n bit viene utilizzato per rappresentare il segno
- Una **duplice** rappresentazione del numero **zero**

  Considerando un byte: 00000000 e 10000000 sono entrambe valide rappresentazioni dello 0

Complemento a 1 (CA1)

La rappresentazione in **complemento a 1** si basa sull'*operazione del complemento*, che si traduce nell'**inversione** di tutti i **bit** della sequenza.

Per ottenere il CA1 bisogna tenere in considerazone *due casi*:

1. Se il numero da rappresentare è *positivo*, la codifica è la *normale conversione* in base due.
2. Se il numero da rappresentare è *negativo*, si effettua la codifica binaria di tale numero in **valore assoluto** e la si **complementa**.

Problemi del CA1

- Nonostante si sia eliminato il problema della *capacità* di rappresentazione *ridotta*, persiste ancora il problema della **doppia** rappresentazione dello **zero**.

  Considerando un byte, 00000000 e 11111111 sono entrambe valide rappresentazioni dello zero

Complemento a 2 (CA2)

La rappresentazione in **complemento a 2** è *un'evoluzione* di quella in *CA1*.

Per effettuare la codifica bisogna tenere conto di *due casi*:

1. Se il numero da rappresentare è *positivo*, la codifica è la *normale conversione* in base due.
2. Se il numero da rappresentare è *negativo*, si effettua la codifica binaria di tale numero in **valore assoluto**, la si **complementa** e si **aggiunge 1**.

In questo modo si *elimina* anche il problema della *duplice* rappresentazione dello *zero*.

Operazioni in CA2

Addizione

L'operazione di addizione in CA2 funziona essenzialmente come l'operazione binaria, bisogna solo tenere in consiedrazione che eventuali **carry** generati dagli **MSB** influenzerebbero il **segno** e perciò vengono **scartati**.

▼ Esempio di addizione [$+3 + (-8)$]

```
00000011 +
11111000 =
11111011 → −5
```

Sottrazione

L'operazione di sottrazione invece viene gestita come una **somma** tra il minuendo e il **CA2** del **sottraendo**.

▼ Esempio di sottrazione [$+8 - (+5)$]

```
00001000 +
11111011 =
00000011 → +3
```

▶

Shift

Si può individuare un'*ulteriore operazione*, denominata **shift**, e consiste nel **far scorrere** la sequenza di bit in una direzione. Per questo, si divide in due sotto operazioni:

- ▼ **Shift left**

  Equivale a *moltiplicare per la base* l'intero numero.

- ▼ **Shift right**

  Equivlae a *dividere per la base* l'interno numero.

- [x] ~~**Eccesso 128**~~

x+128