

Conversione 1 – 0,3 punti

Converti nel formato decimale il numero 10000000 00000000 00000000 00000000

- conversione normale da binario a decimale (unsigned):
è presente solo un 1 nel bit più significativo, siccome sono 32 bit allora quell'uno vale $1 \cdot 2^{31} = 2^{31} = 2,147,483,648$
- conversione con segno (non complemento a 2):
con questo metodo abbiamo il problema di rappresentare due zeri ovvero 0000..... e 10000000..... e questo è il caso di un uno seguito da tutti zeri quindi il risultato è -0 quindi 0
- conversione con complemento a 2:
siccome il bit più significativo è 1 il numero è negativo e devo fare il complemento a 2 al contrario ovvero prima sommando 1 e poi invertendo i bit
come prima cosa togliamo 1 ottenendo 01111....., poi inverto tutti i bit ottenendo 100000..., (ovvero la sequenza di partenza) che è il numero più piccolo rappresentabile, $-(2^{31}) = -2,147,483,648$
- conversione floating point:
il primo bit è 1 quindi il numero è negativo
l'esponente ha tutti 0 e rappresenta un esponente speciale che rappresenta il numero -0 quindi 0

Conversione 2 - 0,3 punti

Converti nel formato decimale il numero 11111111 11111111 11111111 11111111

- conversione normale da binario a decimale (unsigned):
risultato: $(2^0 + 2^1 + \dots + 2^{31}) = (2^{32}) - 1$
- conversione con segno (non complemento a 2):
il primo bit e il segno 1 = -
e poi si calcola normalmente il numero
con la conversione senza il complemento a 2 la sequenza di tutti 1 rappresenta il numero più piccolo $-(2^{31}) - 1$
- conversione con complemento a 2:
nel complemento a 2 il numero con tutti 1 rappresenta il numero 1
si può fare anche sottraendo 1 e poi invertendo tutti i bit e si ottiene 0000....1 = 1
- conversione da floating point:
il bit più significativo è 1 quindi il numero è negativo

gli 8 che seguono (11111111) sono l'esponente e quest'ultimo con otto 1 rappresenta un numero speciale, -infinito se la mantissa fosse 0, ma siccome in questo caso è diversa da 0 questo è un NaN (Not a Number)