**­­­Soluzioni Esercizi Numeri Binari**

TEMPLATE TABELLA CALCOLI

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# **Esercizio 1­­­­­**

Conversione numero 22

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 22 / 2 | 11 | 0 |
| 11 / 2 | 5 | 1 |
| 5 / 2 | 2 | 1 |
| 2 /2 | 1 | 0 |
| 1/2 | 0 | 1 |
|  |  |  |

22 🡪 010110

Conversione numero 30

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 30 / 2 | 15 | 0 |
| 15 / 2 | 7 | 1 |
| 7 / 2 | 3 | 1 |
| 3 / 2 | 1 | 1 |
| 1 / 2 | 0 | 1 |
|  |  |  |

30 🡪 011110

Somma :

22 + 30 🡪

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 |

TRABOCCO!! Perché il range di valori è [-25, 25) , quindi [-32,+31], e la somma di 22 + 30 = 52, che chiaramente supera il range di valori rappresentabili con il complemento a due con 6 bit a disposizione

Sottrazione:

22 – 30 ->

Faccio il complemento a 2 di 30 🡪

011110 🡪 inverto i bit, 100001 🡪 aggiungo 1, 100010 : questo è -30

A questo punto, posso effettuare la somma tra 22 e -30 :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 |

Quindi 22 – 30 = 111000 = -32 + 1 \* 24 + 1 \* 23 = -32 + 16 + 8 = **-8**

# **Esercizio 2**

115,273

Convertiamo come prima cosa 115 a binario

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 115 / 2 | 57 | 1 |
| 57 / 2 | 28 | 1 |
| 28 / 2 | 14 | 0 |
| 14 / 2 | 7 | 0 |
| 7 / 2 | 3 | 1 |
| 3 / 2 | 1 | 1 |
| 1 / 2 | 0 | 1 |

115 in binario : 1110011 = 1 \* 26 + 1 \* 25 + 1 \* 24 + 1\* 21 + 1\*20 = 64 + 32 + 16 + 2 + 1 = 115

Convertiamo la parte decimale 0,273 in binario

|  |  |  |
| --- | --- | --- |
| **Numero** | **Prodotto** | **Parte Intera** |
| 0,273 \* 2 | 0,546 | 0 |
| 0,546 \* 2 | 1,092 | 1 |
| 0,092 \* 2 | 0,184 | 0 |
| 0,184 \* 2 | 0,368 | 0 |

La parte decimale approssimata a 4 cifre binarie è : 0100 = 1\*2-2 = 0,25 (chiaramente approssimando il valore non è proprio uguale e viene approssimata a 0,25)

Uniamo quindi la parte intera e la parte decimale:

1110011,0100

Dobbiamo quindi spostare la virgola fino al primo “1” a sinistra:

1,1100110100 \* 26

Quindi il nostro esponente è 6 🡪 127 + 6 = 133

Portiamo 133 in binario:

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 133 / 2 | 66 | 1 |
| 66 / 2 | 33 | 0 |
| 33 / 2 | 16 | 1 |
| 16 / 2 | 8 | 0 |
| 8 / 2 | 4 | 0 |
| 4 / 2 | 2 | 0 |
| 2 / 2 | 1 | 0 |
| 1 / 2 | 0 | 1 |

133 in binario : 10000101

Possiamo quindi formare il nostro numero binario a virgola mobile con precisione singola a 32 bit:

0 10000101 1100110100000000… (fino a 32 bit)

# **Esercizio 3**

Conversione numero 8: 01000

Conversione numero 7: 00111

Somma 8+7 🡪

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 |

01111 = 1 \* 23 + 1 \* 22 + 1\* 21 + 1 \* 20 = 8 + 4 + 2 + 1 = 15

Non c è trabocco! Perché il range di valori è [-24, 24) , quindi [-16,+15] e la somma di 8 + 7 = 15, rientra quindi nel range di valori rappresentabili da 5 bit

Sottrazione 8 – 7 🡪

Faccio il complemento a 2 di 7 🡪

00111 🡪 inverto i bit, 11000 🡪 aggiungo 1, 11001 : questo è -7

P.S. : Per verificare che sia -7, si può calcolare a partire dal valore binario:   
  
11001 🡪 -16 + 1 \* 23 + 1 \* 20 = -16 + 8 + 1 = -7

A questo punto, posso effettuare la somma tra 8 e -7 :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 |

Quindi 8 – 7 = 00001 = 1 \* 20 = 1

# **Esercizio 4**

75,252

Convertiamo come prima cosa 75 a binario

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 75 / 2 | 37 | 1 |
| 37 / 2 | 18 | 1 |
| 18 / 2 | 9 | 0 |
| 9 / 2 | 4 | 1 |
| 4 / 2 | 2 | 0 |
| 2 / 2 | 1 | 0 |
| 1 / 2 | 0 | 1 |

1001011 = 1 \* 26 + 1 \* 23 + 1 \* 21 + 1 \* 20 = 64 + 8 + 2 + 1 = 75

Convertiamo la parte decimale 0,252 in binario

|  |  |  |
| --- | --- | --- |
| **Numero** | **Prodotto** | **Parte Intera** |
| 0,252 \* 2 | 0,504 | 0 |
| 0,504 \* 2 | 1,008 | 1 |
| 0,0,08 \* 2 | 0,016 | 0 |
| 0,016 \* 2 | 0,032 | 0 |

La parte decimale approssimata a 4 cifre binarie è : 0100 = 1\*2-2 = 0,25 (chiaramente approssimando il valore non è proprio uguale e viene approssimata a 0,25)

Uniamo quindi la parte intera e la parte decimale:

1001011,0100

Dobbiamo quindi spostare la virgola fino al primo “1” a sinistra:

1,0010110100 \* 26

Quindi il nostro esponente è 6 🡪 127 + 6 = 133

Portiamo 133 in binario:

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 133 / 2 | 66 | 1 |
| 66 / 2 | 33 | 0 |
| 33 / 2 | 16 | 1 |
| 16 / 2 | 8 | 0 |
| 8 / 2 | 4 | 0 |
| 4 / 2 | 2 | 0 |
| 2 / 2 | 1 | 0 |
| 1 / 2 | 0 | 1 |

133 in binario : 10000101

Possiamo quindi formare il nostro numero binario a virgola mobile con precisione singola a 32 bit:

0 10000101 0010110100000000… (fino a 32 bit)

# **Esercizio 5**

72 e 48

Conversione numero 72 🡪

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 72 / 2 | 36 | 0 |
| 36 / 2 | 18 | 0 |
| 18 / 2 | 9 | 0 |
| 9 / 2 | 4 | 1 |
| 4 / 2 | 2 | 0 |
| 2 / 2 | 1 | 0 |
| 1 / 2 | 0 | 1 |

72 = 01001000 = 1 \* 26 + 1 \* 23 = 64 + 8 = 72

Conversione numero 48 🡪

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 48 / 2 | 24 | 0 |
| 24 / 2 | 12 | 0 |
| 12 / 2 | 6 | 0 |
| 6 / 2 | 3 | 0 |
| 3 / 2 | 1 | 1 |
| 1 / 2 | 0 | 1 |

48 = 00110000 = 1 \* 25 + 1 \* 24 = 32 + 16 = 48

Somma 72+48 🡪

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

01111000 = 1 \* 26 + 1 \* 25 + 1 \* 24 + 1 \* 23 = 64 + 32 + 16 + 8 = 120

Non c è trabocco! Perché il range di valori è [-27, 27) , quindi [-128,+127] e la somma di 72 + 48 = 120, rientra quindi nel range di valori rappresentabili da 8 bit

Sottrazione:

72 – 48 🡪

Faccio il complemento a 2 di 48 🡪

00110000 🡪 inverto i bit, 11001111 🡪 aggiungo 1, 11010000: questo è -48

P.S. : Per verificare che sia -48 corretto, si può calcolare a partire dal valore binario:   
  
11010000🡪 -128 + 1 \* 26 + 1 \* 24 = -128 + 64 + 16 = -48

A questo punto, posso effettuare la somma tra 72 e -48 :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |

Quindi 72 – 48 = 00011000 = 1 \* 24 + 1 \* 23 = 16 + 8 = 24

# **Esercizio 6**

-199,523

Convertiamo come prima cosa 199 a binario

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 199 / 2 | 99 | 1 |
| 99 / 2 | 49 | 1 |
| 49 / 2 | 24 | 1 |
| 24 / 2 | 12 | 0 |
| 12 / 2 | 6 | 0 |
| 6 / 2 | 3 | 0 |
| 3 / 2 | 1 | 1 |
| 1 / 2 | 0 | 1 |

11000111 = 1 \* 27 + 1 \* 26 + 1 \* 22 + 1 \* 21 + 1 \* 20= 128 + 64 + 4 + 2 + 1 = -199 (Il primo bit è quello di segno)

Convertiamo la parte decimale 0,523 in binario

|  |  |  |
| --- | --- | --- |
| **Numero** | **Prodotto** | **Parte Intera** |
| 0,523 \* 2 | 1,046 | 1 |
| 0,046 \* 2 | 0,092 | 0 |
| 0,0,08 \* 2 | 0,184 | 0 |
| 0,184 \* 2 | 0,368 | 0 |

La parte decimale approssimata a 4 cifre binarie è : 1000 = 1\*2-1 = 0,5 (chiaramente approssimando il valore non è proprio uguale e viene approssimata a 0,5)

Uniamo quindi la parte intera e la parte decimale:

11000111,1000

Dobbiamo quindi spostare la virgola fino al primo “1” a sinistra:

1,10001111000 \* 27

Quindi il nostro esponente è 7 🡪 127 + 7 = 134

Portiamo 134 in binario:

|  |  |  |
| --- | --- | --- |
| **Numero** | **Quoziente** | **Resto** |
| 134 / 2 | 67 | 0 |
| 67 / 2 | 33 | 1 |
| 33 / 2 | 16 | 1 |
| 16 / 2 | 8 | 0 |
| 8 / 2 | 4 | 0 |
| 4 / 2 | 2 | 0 |
| 2 / 2 | 1 | 0 |
| 1 / 2 | 0 | 1 |

134 in binario : 10000110

Possiamo quindi formare il nostro numero binario a virgola mobile con precisione singola a 32 bit:

1 10000110 10001111000… (fino a 32 bit)