Binary → Unsigned :

0 0 0 1

× 2^3 2^2 2^1 2^0

0 0 0 1

∴ 1

there's no -ve 0 (-0) in math so binary 1000 → unsigned is 0

* 2's complement is always 1 value smaller than 1's complement *

Represent the number -92₁₀ in...

(i) 8-bit signed magnitude

signed = -92₁₀

unsigned = 92₁₀

2 | 92
2 | 46 0
2 | 23 0
2 | 11 1
2 | 5 1
2 | 2 1
2 | 1 0
2 | 0 1

unsigned ⇒ 01011100

~~unsigned = 01000011~~ X

signed = 11011100 * ✓

(1 = negative)

(ii) 8-bit 2's complement

1's = ~~0~~ 01011100 (-ve) (10100011+1 = 10100100)

2's = 10100011 + 1

10100100 *

Convert the following numbers to 6-bit 2's complement notation & add them :

(i) -16 + 11 (ii) 16 - 11

16 → 010000

11 → 001011

-16 → 101111 + 1 = 110000

-11 → 110100 + 1 = 110101

(i) 110000 + 001011 = 111011

(ii) 010000 + 110101 = (1) 000101

* 2's complement can discard the extra bit

Because...

1's → 2's need +1 already save the process to create memory to store the discarded 1

Binary arithmetic with 8-bit 2's complement :

(i) 33 + 92

2 | 33
2 | 16 1
2 | 8 0
2 | 4 0
2 | 2 0
2 | 1 0
2 | 0 1

2 | 92
2 | 46 0
2 | 23 0
2 | 11 1
2 | 5 1
2 | 2 1
2 | 1 0
2 | 0 1

100001 + 1011100

1's : 011110 + 0100011

2's : 011111 + 0100100

(ii) 33 - 92 ⇒ 33 + (-92)

33 2's = 1011111

92 2's = 0100100

-92 2's = 1100100

1 1 1 1
1 0 1 1 1 1 1
+ 1 1 0 0 1 0 0
1 1 0 0 0 0 1 1

CORRELTION : 33 ⇒ 00100001

92 ⇒ 01011100

-92 ⇒ 10100011+1 → 10100100

(i) 00100001 + 01011100 = 01111101

(ii) 00100001 + 10100100 = 11000101

4-bit binary number	signed	1's	2's
0000	0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7
1000	0	-7	-8
⋮	⋮	⋮	⋮
1111	-7	0	-1

* n = number of bits

	range
unsigned	$0 \longrightarrow 2^n - 1$
signed 1's	$-\left(\frac{2^n}{2} - 1\right) < 0 < \left(\frac{2^n}{2} - 1\right)$
2's	$-\left(\frac{2^n}{2}\right) < 0 < \left(\frac{2^n}{2} - 1\right)$

* overflow = out-of-range

↳ How computer detect:

ANS: • + 2 positive number results in -ve number

• + 2 negative number results in +ve number

What are the ranges of number that can be represented in a computer if it's using...

(i) 4-bit 2's complement representation

ANS: $-8 < 0 < +7$

(ii) 6-bit 2's complement representation

ANS: $-32 < 0 < 31$

(iii) 8-bit 2's complement representation

ANS: $-128 < 0 < 127$

Use binary arithmetic with 8-bit 2's complement representation to calculate 92 + 92. result of operation accepted?

ANS: 92 → 01011100

```
  01011100
+ 01011100
----------
  10111000  (overflow)
```

↳ results of operation is NOT ACCEPTED.

92+92=184

184 is outside the range of numbers that can be represented in a computer using 8-bit 2's complement representation.