

3.1 Number representations and codes

Name: _____ Class: _____ Date: _____ Total: 23 marks

KEY WORDS binary • hexadecimal • two's complement
• overflow

Objective

Convert between binary, hexadecimal and decimal; negate in two's complement; detect overflow by sign reasoning – and verify every conversion by converting back.

1 Key knowledge and worked examples

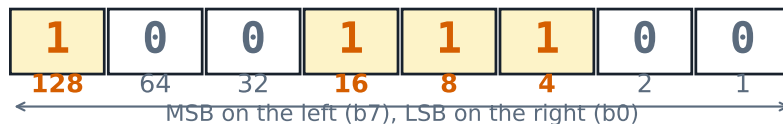
■ Positional conversion

Integer conversion is repeated division by the base, collecting remainders; hexadecimal groups binary in fours from the right. Every conversion checks by converting back or re-adding the weights.

Worked example. $156 = 10011100_2 = 9C_{16}$. Check by weights: $128 + 16 + 8 + 4 = 156 \checkmark$. Hex check: $9C = 9 \times 16 + 12 = 156 \checkmark$.

$$128 + 16 + 8 + 4 = 156$$

hex: $1001\ 1100 = 0x9C$



■ Two's complement: invert and add one

To negate an N-bit number: invert every bit, add 1. -5 in 8 bits: $00000101 \rightarrow 11111010 + 1 = 11111011$.

The **range** in N bits is $-2^{N-1} \dots 2^{N-1} - 1$; the extreme value $1000\dots 0$ is its own negative – negating it is **overflow**.

■ Overflow detection: like signs in, unlike sign out

Adding two same-sign numbers that yields the opposite sign is overflow – the result left the range. Adding unlike signs never overflows. **Worked example:** $0111\ 0000$ (112) + $0000\ 1000$ (8) = $0111\ 1000$ (120) fine; but $0111\ 0000 + 0001\ 0000$ (16) = $1000\ 0000$ (-128 as signed) – two positives made a negative: overflow ✓ (the true sum 128 is out of the +127 range).

2 Practice

2.1 Convert 200 to binary and to hex; check by re-adding the weights. [3]

2.2 Write -46 in 8-bit two's complement, and verify by adding it to +46. [3]

2.3 Add the 8-bit signed values $0101\ 1010$ and $0100\ 0110$. Give the result as signed decimal and state whether overflow occurred, with the reason. [4]

3 Exam-style questions

3.1 A sensor reports angles in 10-bit two's complement covering $-512^\circ \dots +511^\circ$.

State the binary for $+200^\circ$ and for -200° . Then add $+400^\circ$ and $+200^\circ$ as 10-bit values, give the signed result, and explain the overflow using the sign rule and the range. State the largest positive angle that can safely be doubled. [7]

3.2 An 8-bit system computes $X - Y$ as $X + (-Y)$ in two's complement.

Compute $90 - 30$ through the full binary working (convert, negate, add), verify the decimal result, and state why the subtraction method needs no special borrow logic. Then compute $90 - (-90)$ and explain any anomaly. [6]
