

1 Pre-Check

This section is designed as a conceptual check for you to determine if you conceptually understand and have any misconceptions about this topic. Please answer true/false to the following questions, and include an explanation:



1.1

Depending on the context, the same sets of bits may represent different things.

~~True.~~ What it refers to also depends on the base. The same set of bits represents different things, in, for example, decimal, binary, and hexadecimal.

1.2

It is possible to get an overflow error in Two's Complement when adding numbers of opposite signs.

~~True.~~ For example, you have a 0111, then when you add another 1 to this set, you get an overflow which represents -8. *False. Overflow errors only occur when the correct result of the addition falls outside the range of $[-2^{n-1}, 2^{n-1}-1]$. Addition numbers of opposite signs will not result such occasions.*

1.3

If you interpret a N bit Two's complement number as an unsigned number, negative numbers would be smaller than positive numbers.

~~False.~~ The first number of Two's complement is 1 in negative number, but 0 in positive number. So when interpreted into an unsigned number, negative number would be larger than positive number.

1.4

If you interpret an N bit Bias notation number as an unsigned number (assume there are negative numbers for the given bias), negative numbers would be smaller than positive numbers.

~~True.~~ Positive numbers are after negative numbers.

2 Unsigned Integers

2.1

If we have an n -digit unsigned numeral $d_{n-1}d_{n-2}\dots d_0$ in radix (or base) r , then the value of that numeral is $\sum_{i=0}^{n-1} r^i d_i$, which is just fancy notation to say that instead of a 10's or 100's place we have an r 's or r^2 's place. For the three radices binary, decimal, and hex, we just let r be 2, 10, and 16, respectively.

Let's try this by hand. Recall that our preferred tool for writing large numbers is the IEC prefixing system:

$$\begin{array}{llll} \text{Ki (Kibi)} = 2^{10} & \text{Gi (Gibi)} = 2^{30} & \text{Pi (Pebi)} = 2^{50} & \text{Zi (Zebi)} = 2^{70} \\ \text{Mi (Mebi)} = 2^{20} & \text{Ti (Tebi)} = 2^{40} & \text{Ei (Exbi)} = 2^{60} & \text{Yi (Yobi)} = 2^{80} \end{array}$$

- (a) Convert the following numbers from their initial radix into the other two common radices:

$$\begin{array}{ll} 1. \text{0b10010011} & 147_{10} \quad 93_{16} \\ 2. 63 & 0b00111111_2 \quad 3F_{16} \\ 3. \text{0b00100100} & 36_{10} \quad 24_{16} \\ 4. 0 & 0b0_2 \quad 0x0_{16} \\ 5. 39 & 0b00100111_2 \quad 27_{16} \\ 6. 437 & 0b001101100101_2 \quad 1B5_{16} \\ 7. \text{0x0123} & 0b000100100011_2 \quad 29_{10} \end{array}$$

- (b) Convert the following numbers from hex to binary:

$$\begin{array}{ll} 1. \text{0xD3AD} & 0b11010011010110101101 \\ 2. \text{0xB33F} & 0b1011001100111111 \\ 3. \text{0x7EC4} & 0b0111110111001000 \end{array}$$

- (c) Write the following numbers using IEC prefixes:

$$\begin{array}{llll} \bullet 2^{16} = 64 \text{ Ki} & \bullet 2^{27} = 128 \text{ Mi} & \bullet 2^{43} = 8 \text{ Ti} & \bullet 2^{36} = 64 \text{ Gi} \\ \bullet 2^{34} = 16 \text{ Gi} & \bullet 2^{61} = 2 \text{ Ei} & \bullet 2^{47} = 128 \text{ Ti} & \bullet 2^{59} = 512 \text{ Pi} \end{array}$$

- (d) Write the following numbers as powers of 2:

$$\begin{array}{lll} \bullet 2 \text{ Ki} = 2^{11} & \bullet 512 \text{ Ki} = 2^{19} & \bullet 16 \text{ Mi} = 2^{24} \\ \bullet 256 \text{ Pi} = 2^{38} & \bullet 64 \text{ Gi} = 2^{36} & \bullet 128 \text{ Ei} = 2^{67} \end{array}$$

3 Signed Integers

3.1

Unsigned binary numbers work for natural numbers, but many calculations use negative numbers as well. To deal with this, a number of different schemes have been used to represent signed numbers, but we will focus on two's complement, as it is the standard solution for representing signed integers.

- Most significant bit has a negative value, all others are positive. So the value of an n -digit two's complement number can be written as $\sum_{i=0}^{n-2} 2^i d_i - 2^{n-1} d_{n-1}$.
- Otherwise exactly the same as unsigned integers.
- A neat trick for flipping the sign of a two's complement number: flip all the bits and add 1.

- Addition is exactly the same as with an unsigned number.
- Only one 0, and it's located at 0b0.

For questions (a) through (c), assume an 8-bit integer and answer each one for the case of an unsigned number, biased number with a bias of -127, and two's complement number. Indicate if it cannot be answered with a specific representation.

(a) What is the largest integer? What is the result of adding one to that number?

1. Unsigned? $1111;1111$ (255) overflow $\rightarrow 0000;0000$ (0)
2. Biased? $1111;1111$ (128) overflow $\rightarrow 0000;0000$ (+127)
3. Two's Complement? $0111;1111$ (127) overflow $\rightarrow 0000;0000$ (-128)

(b) How would you represent the numbers 0, 1, and -1?

1. Unsigned? $0b00000000$, $0b00000001$, cannot
2. Biased? $0b01111111$, $0b10000000$, $0b01111110$
 10000000 , 10000001 , 01111111
3. Two's Complement? $0b00000000$, $0b00000001$, $0b11111111$

(c) How would you represent 17 and -17?

1. Unsigned? 00010001 , cannot
2. Biased? 10010000 , 01101110
3. Two's Complement? 00010001 , 11101111

(d) What is the largest integer that can be represented by any encoding scheme that only uses 8 bits?

$$01111111 = 2^8 - 1 = 127$$

There is no such Integer. For example. an arbitrary 8-bit mapping could choose to represent the numbers from 1 to 256 instead of 0 to 255

(e) Prove that the two's complement inversion trick is valid (i.e. that x and $\bar{x} + 1$ sum to 0).

proof Note that for any x we have $x + \bar{x} = 0b1..1$. Adding $0b1$ to $0b1..1$ will cause the value to overflow, meaning that $0b1..1 + 0b1 = 0b0..0$. Therefore, $x + \bar{x} + 1 = 0$

(f) Explain where each of the three radices shines and why it is preferred over other bases in a given context.

Normally when we need to operate only positive numbers, we use unsigned for the simple sake that it can represent more numbers than the others.

Decimal is the preferred radix for human hand calculations, likely related to the fact that humans have 10 fingers
Binary numerals are particularly useful for computers Binary signals are less likely to be garbled than higher radix signals, as there is more "distance" (voltage or current) between valid signals. Additionally, binary signals are quite

convenient to design circuits, as we'll see later in the course.

4 Arithmetic and Counting

4.1

Addition and subtraction of binary/hex numbers can be done in a similar fashion as with decimal digits by working right to left and carrying over extra digits to the next place. However, sometimes this may result in an overflow if the number of bits can no longer represent the true sum. Overflow occurs if and only if two numbers with the same sign are added and the result has the opposite sign.

- (a) Compute the decimal result of the following arithmetic expressions involving 6-bit Two's Complement numbers as they would be calculated on a computer. Do any of these result in an overflow? Are all these operations possible?

1. $0b011001 \ominus 0b000111$ ~~X~~, possible

2. $0b100011 + 0b111010$ ~~✓~~, possible

3. $0x3B + 0x06$ ~~X~~, possible converting to binary, $0b111011 + 0b000110 \rightarrow \text{overflow}$

4. $0xFF - 0xAA$ ~~X~~, possible. $0b111111 - 0b101010$

- (b) What is the least number of bits needed to represent the following ranges using any number representation scheme.

1. 0 to 256 10

2. -7 to 56 7

3. 64 to 127 and -64 to -127 8

4. Address every byte of a 12 TiB chunk of memory 44