CS61C Spring 2015 Discussion 0 – Number Representation

1 Unsigned Integers

If we have an n-digit unsigned numeral $d_{n-1}d_{n-2}\ldots d_0$ 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 binary, decimal, and hex we just let r be 2, 10, and 16, respectively.

Recall also that we often have cause to write down large numbers, and our preferred tool for doing that is the IEC prefixing system:

- Kilo- (Ki) $= 2^{10} \approx 10^3$
- Mega- (Mi) $= 2^{20} \approx 10^6$
- Giga- (Gi) $= 2^{30} \approx 10^9$
- Tera- (Ti) $= 2^{40} \approx 10^{12}$
- Peta- (Pi) $= 2^{50} \approx 10^{15}$
- Exa- (Ei) $= 2^{60} \approx 10^{18}$
- Zetta- (Zi) $= 2^{70} \approx 10^{21}$
- Yotta- (Yi) $= 2^{80} \approx 10^{24}$

1.1 We don't have calculators during exams, so let's try this by hand

1. Convert the following numbers from their initial radix into the other two common radices:

   (a) 0b10010011

   (b) 0xD3AD

   (c) 63

   (d) 0b00100100

   (e) 0xB33F

   (f) 0

   (g) 0x7EC4

   (h) 437

2. Write the following numbers using IEC prefixes:

   (a) 2^{16}

   (b) 2^{34}

   (c) 2^{27}

   (d) 2^{61}

   (e) 2^{43}

   (f) 2^{47}

   (g) 2^{36}

   (h) 2^{58}

3. Write the following numbers as powers of 2:

   (a) 2 Ki

   (b) 256 Pi

   (c) 512 Ki

   (d) 64 Gi

   (e) 16 Mi

   (f) 128 Ei

2 Signed Integers w/ Two's Complement

- Two's complement is the standard solution for representing signed integers.
  - Most significant bit has a negative value, all others have positive.
  - 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.

2.1 Exercises

For the following questions assume an 8 bit integer. Answer each question for the case of a two's complement number and unsigned number.

1. What is the largest integer? The largest integer + 1?
2. How do you represent the numbers 0, 1, and -1?
3. How do you represent 17, -17?
4. What is the largest integer that can be represented by *any* encoding scheme that only uses 8 bits?
5. Prove that the two's complement inversion trick is valid (i.e. that x and $\overline{x} + 1$ sum to 0).
6. Explain where each of the three radices shines and why it is preferred over other bases in a given context.

3 Counting

Bitstrings can be used to represent more than just numbers. In fact, we use bitstrings to represent *everything* inside a computer. Two key facts to remember about binary counting:

- With n bits, we can count or represent 2^n things. [Why? Because we have 2 possible values for the each bit (0 or 1) $\implies 2 \cdot 2 \cdots 2$ (n times) $= 2^n$ different bitstrings]
- Equivalently, to count or represent k things, we need $n = \lceil \log_2 k \rceil$ bits.

3.1 Exercises

1. If the only values a variable can take on are $0, \pi$ or e, what is the minimum number of bits needed to represent it?
2. If we need to address 3 TiB of memory and we want to address every byte of memory, how long does an address need to be?
3. If the only value a variable can take on is e, how many bits are needed to represent it?