CS61C Fall 2014 Discussion 0 – Number Representation

1 Unsigned Integers

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 binary, decimal, and hex we just let r be 2, 10, and 16, respectively.

Recall also that we often have cause to write down unreasonably large numbers, and our preferred tool for doing that is the IEC prefixing system: Ki $= 2^{10}$, Mi $= 2^{20}$, Gi $= 2^{30}$, Ti $= 2^{40}$, Pi $= 2^{50}$, Ei $= 2^{60}$, Zi $= 2^{70}$, Yi $= 2^{80}$.

1.1 We dont have calculators during exams, so lets try this by hand

1. Convert the following numbers from their initial radix into the other two common radices: 0b10010011 = 147 = 0x93 , 0xD3AD = 0b1101 0011 1010 1101 = 54189 , 63 = 0b0011 1111 = 0x3F , 0b00100100 = 36 = 0x24 , 0xB33F = 0b1011 0011 0011 1111 = 45887 , 0 = 0b0 = 0x0 , 39 = 0b0010 0111 = 0x27 , 0x7EC4 = 0b0111 1110 1100 0100 =32452 , 437 = 0b0001 1011 0101 = 0x1B5

2. Write the following numbers using IEC prefixes: 2^{16} = 64 Ki , 2^{34} = 16 Gi, 2^{27} = 128 Mi, 2^{61} = 2 Ei, 2^{43} = 8 Ti, 2^{47} = 128 Ti, 2^{36} = 64 Gi, 2^{58} = 256 Pi.

3. Write the following numbers as powers of 2: 2 Ki $= 2^{11}$, 256 Pi $= 2^{58}$, 512 Ki $= 2^{19}$, 64 Gi $= 2^{36}$, 16 Mi $= 2^{24}$, 128 Ei $= 2^{67}$.

2 Signed Integers

Unsigned binary numbers work to store 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.

2.1 Sign and Magnitude and One's complement

Both of these schemes are relatively simple conceptually, but have been replaced by cleverer representations. Why? Both schemes were abandoned because they have relatively complicated rules of arithmetic, as well as both a positive and a negative 0.

- Most significant bit tells you the sign: 1 if negative, 0 if positive.
- Positive values can be treated just like unsigned integers.
- To invert the sign of a sign and magnitude number flip the MSB.
- To invert the sign of a one's complement number flip all the bits.

2.2 Biased Notation

- Like an unsigned int, but offset by $-(2^{n-1} - 1)$, where n is the number of bits in the numeral. Aside: Technically we could choose any bias we please, but the choice presented here is extraordinarily common.
- Formally, if we have an n-bit biased notation number with bits $d_{n-1}d_{n-2}\dots d_0$, then the value of the numeral is $-(2^{n-1} - 1) + \sum_{i=0}^{n-1} 2^i d_i$.
- Just one zero, but it's not at 0b0.
- Addition is a little weird, but not overwhelmingly so.

2.3 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.
- Only one 0, and it's located at 0b0.

2.4 Exercises

For the following questions assume an 8 bit integer. Answer each question for the case of a sign and magnitude number, a one's complement number, a biased notation number, and a two's complement number.

1. What is the largest integer? The largest integer + 1?
   (a) [Sign and Magnitude:] 127, -0
   (b) [One's Complement:] 127, -127
   (c) [Biased Notation:] 128, -127
   (d) [Two's Complement:] 127, -128
2. How do you represent the numbers 0, 1, and -1?
   (a) [Sign and Magnitude:] 0b0000 0000 or 0b1000 0000, 0b0000 0001, 0b1000 0001
   (b) [One's Complement:] 0b0000 0000 or 0b1111 1111, 0b0000 0001, 0b1111 1110
   (c) [Biased Notation:] 0b0111 1111, 0b1000 0000, 0b0111 1110
   (d) [Two's Complement:] 0b0000 0000, 0b0000 0001, 0b1111 1111
3. How do you represent 17, -17?
   (a) [Sign and Magnitude:] 0b0001 0001, 0b1001 0001
   (b) [One's Complement:] 0b0001 0001, 0b1110 1110
   (c) [Biased Notation:] 0b1001 0000, 0b0110 1110
   (d) [Two's Complement:] 0b0001 0001, 0b1110 1111
4. What is the largest integer that can be represented by *any* encoding scheme that only uses 8 bits? There is no such integer. For example, you could use biased notation with an arbitrarily large bias.
5. Prove that the two's complement inversion trick is valid (i.e. that x and $\overline{x}+1$ sum to 0). Note that for any x we have $x+\overline{x} = $ 0b1...1. A straightforward hand calculation shows that $0\text{b}1\ldots1 + 0\text{b}1 = 0$.
6. Explain where each of the three radices shines and why it is preferred over other bases in a given context. 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 with, as we'll see later in the course.

   Hexadecimal numbers are a convenient shorthand for displaying binary numbers, owing to the fact that one hex digit corresponds exactly to four binary digits.

3 Counting

Bitstrings can be used to represent more than just numbers. In fact, we use bitstrings to represent *everything* inside a computer. And, because we don't want to be wasteful with bits it is important that to remember that n bits can be used to represent 2^n distinct things. To reiterate, n bits can represent up to 2^n distinct objects.

3.1 Exercises

1. If the value of a variable is $0, \pi$ or e, what is the minimum number of bits needed to represent it. 2
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? 42 bits
3. If the only value a variable can take on is e, how many bits are needed to represent it. 0