CSCC85 Spring 2006: Tutorial 0 Notes

Yani Ioannou

January 11th, 2006

"There are 10 types of people in the world, those who understand binary, and those who don't."

Contents

1 Number Representations **1**
1.1 Base b to Decimal . . . 2
1.2 Horner's Rule . . . 2
1.3 Decimal to Base b . . . 2
1.3.1 Integer numbers . . . 2
1.3.2 Fractional numbers . . . 3
1.4 Converting between binary, hexadecimal and octal . . . 3
1.4.1 Binary to Hexadecimal . . . 3
2 Signed Number Representations **4**
2.1 Signed Magnitude . . . 4
2.2 One's compliment . . . 4
2.3 Two's compliment . . . 4
2.3.1 The Exception . . . 6

1 Number Representations

We write a number of base (*radix*) r as

$$(\ldots x_3\, x_2\, x_1\, x_0\, .\, x_{-1}\, x_{-2} \ldots)_b\,.$$

The most common radix we encounter is base 10, decimal, which is often assumed to have arisen from humans 10 digits (fingers/thumbs). However many other radix have been used historically, often as a matter of convenience. Even today we often use different radix without knowing it, for example the common angular measure of degrees ($^\circ$), minutes ($'$) and seconds($''$), it is also used in time for hours and minutes. Similarly we use radix 12 for the hours in a day and months in a year.

Base 2 (*Binary*), base 8 (*Octal*) and base 16 (*Hexadecimal*) have come into widespread use with the advent of digital computers, all of which (except for a few early base-3 and base-10 designs), have been binary. This is of course where our interest in the subject lies.

1.1 Base b to Decimal

For any base b we can find the decimal (base 10) representation of the number through the simple polynomial expansion

$$\dots x_3b^3 + x_2b^2 + x_1b^1 + x_0b^0 + x_{-1}b^{-1}x_{-1}b^{-2}\dots_b = \sum_n x^n b^n. \tag{1}$$

For example to convert for the following base 2 (*binary*) number to decimal,

$$\begin{aligned} 1101.01_2 = & \quad (1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0 + 0\times 2^{-1} + 1\times 2^{-2})_{10} \\ = & \quad (8+4+0+1+0+\tfrac{1}{4})_{10} \\ = & \quad 13.25_{10}. \end{aligned}$$

1.2 Horner's Rule

We can use Horner's rule to re-write the integer part of the polynomial in equation 1 as

$$\begin{aligned} 1101_2 = & \quad (1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0)_{10} && (2)\\ = & \quad (((0\times 2+1)\times 2+1)\times 2+0)\times 2+1 && (3)\\ = & \quad 13_{10}. && (4) \end{aligned}$$

This form is much easier for humans to calculate, and you can use this method to quickly convert binary to decimal (or in fact any base).

1.3 Decimal to Base b

1.3.1 Integer numbers

To convert the integer part of a decimal number into base b, we use the remainder of a division by the target base b. For example to convert the decimal number 1022 into binary,

$$\begin{aligned} 1022/2 = & \quad 511 \quad \text{r } 0 \\ 511/2 = & \quad 255 \quad \text{r } 1 \\ 255/2 = & \quad 127 \quad \text{r } 1 \\ 127/2 = & \quad 63 \quad \text{r } 1 \\ 63/2 = & \quad 31 \quad \text{r } 1 \\ 31/2 = & \quad 15 \quad \text{r } 1 \\ 15/2 = & \quad 7 \quad \text{r } 1 \\ 7/2 = & \quad 3 \quad \text{r } 1 \\ 3/2 = & \quad 1 \quad \text{r } 1 \\ 1/2 = & \quad 0 \quad \text{r } 1 \\ \Rightarrow 1022_{10} = & \quad 1111111110_2, \end{aligned}$$

note that we read the binary digits bottom to top.

1.3.2 Fractional numbers

To convert the fractional part of a decimal number we instead repeatedly multiply by the target base b, and take the result's integer parts as the digits. For example to convert the decimal number 0.25 into binary,

$$\begin{aligned} 0.25 \times 2 =& \quad \mathbf{0}.5 \\ 0.5 \times 2 =& \quad \mathbf{1}.0 \\ \Rightarrow 0.25_{10} =& \quad 0.01_2, \end{aligned}$$

note that in this case we read the digits top to bottom. However the fractional part will not always converge to 0, in general a fraction in one base might not have an exact representation in another. For example,

$$\begin{aligned} 0.2432 \times 2 =& \quad \mathbf{0}.4864 \\ 0.4864 \times 2 =& \quad \mathbf{0}.9728 \\ 0.9728 \times 2 =& \quad \mathbf{1}.9456 \\ 0.9456 \times 2 =& \quad \mathbf{1}.7824 \\ 0.7824 \times 2 =& \quad \mathbf{1}.5648 \\ 0.5648 \times 2 =& \quad \mathbf{1}.1296 \\ 0.1296 \times 2 =& \quad \mathbf{0}.2592 \\ 0.2592 \times 2 =& \quad \mathbf{0}.5184 \\ 0.5184 \times 2 =& \quad \mathbf{0}.0368 \\ 0.0368 \times 2 =& \quad \mathbf{0}.0368 \\ &\vdots \\ \Rightarrow 0.2432_{10} \approx& \quad 0.011110_2, \end{aligned}$$

1.4 Converting between binary, hexadecimal and octal

Although binary is useful in understanding binary computers, it is also needlessly tedious for humans to work with. Because Octal (base 2^3) and Hexadecimal (base 2^4) are simply grouping of binary numbers, it is often much easier to convert binary numbers into these and work with them.

1.4.1 Binary to Hexadecimal

To convert a binary number into Hexadecimal we simply group the number into groups of 4 bits (nibbles), and pad the left with 0s. Then by inspection we can easily find the corresponding hexadecimal digits. For example to convert 1011110_2 to hexadecimal,

$$\begin{aligned} &(\,\mathbf{0}101 \quad 1110\,)_2 \\ \Rightarrow &(\,5 \quad 14\,)_{10} \\ \Rightarrow &(\,5 \quad E\,)_{16}. \end{aligned}$$

2 Signed Number Representations

Although we have discussed number bases, we haven't seen how to represent signed numbers in alternative basis.

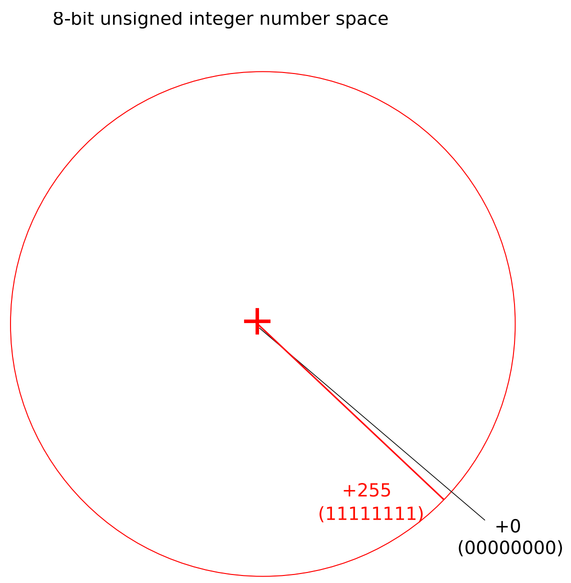

Figure 1: Unsigned 8-bit integer number space

2.1 Signed Magnitude

In the decimal number system we commonly use the *signed magnitude* method to represent a signed number; i.e. we simply augment the magnitude of the number (e.g. 10) with a sign (+ or $-$). This representation however does have its drawbacks - namely there are two possible representations of the number zero $- +0, -0$, while intuitively zero has no sign.

2.2 One's compliment

One's compliment is a form of signed magnitude for binary (or radix 2) numbers where the sign is simply represented by the most significant bit (MSB). To obtain the 1's compliment of a binary number we simply take the binary NOT of it. For example,

$$\begin{aligned} -75_{10} &= -(01001011_2) \\ &= 10110100_2 \end{aligned}$$

2.3 Two's compliment

Two's compliment addresses the problems of one's compliment (2 representations of zero, ones compliment numbers can't be worked with directly as expected). To obtain the 2's compliment of a

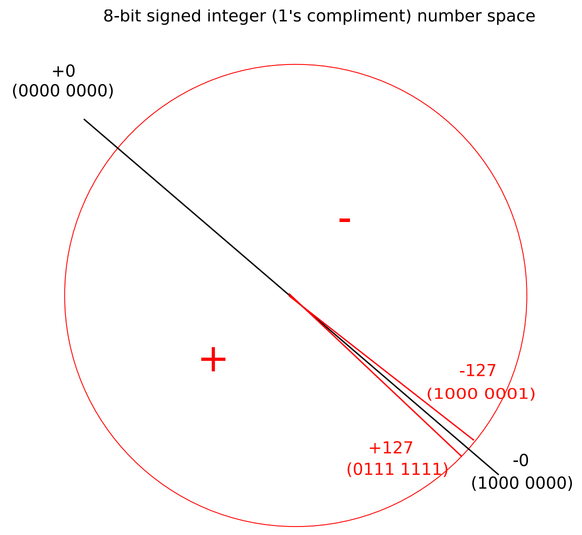

Figure 2: Signed (1's compliment) 8-bit integer number space

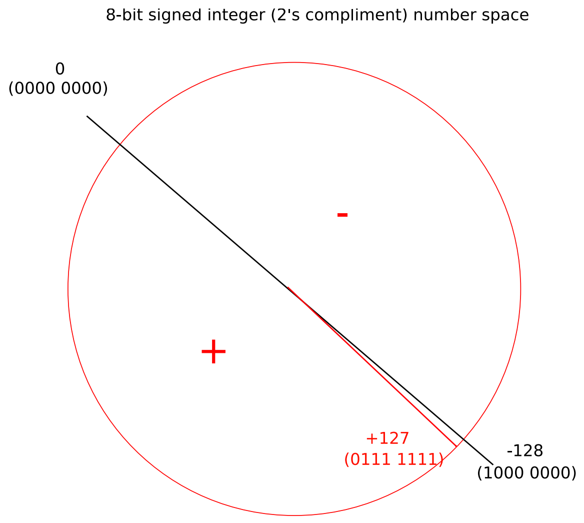

Figure 3: Signed (2's compliment) 8-bit integer number space

number we simply perform 1's compliment (i.e. NOT the binary number) and add 1. For example,

$$\begin{aligned} -(01011101)_2 =& \quad (10100010)_2 \quad \text{(1's compliment)} \\ (10100010)_2 + 1_2 =& \quad (10100011)_2 \quad \text{(2's compliment).} \end{aligned}$$

2.3.1 The Exception

There is a single exception to the 2's compliment rule, namely the most negative number (i.e. $10\ldots0_2$) whose 2's compliment is always itself. For example for the 8-bit example 10000000

$$\begin{aligned} -(10000000)_2 =& \quad (01111111)_2 \quad \text{(1's compliment)} \\ (01111111)_2 + 1_2 =& \quad (10000000)_2 \quad \text{(2's compliment).} \end{aligned}$$

Intuitively this is because there is not positive equivalent of that number (remember the negative range is one more than the positive range with 2's compliment numbers). For example, in an 8-bit 2's compliment signed integer representation the range is $-128\ldots+127$, where -128 has the representation 10000000.