Piazza FAQ Week 1

Q: Why does multiplying by 2 and adding convert a binary number to decimal?

The key concept here is the concept of place value: the way we write/express numbers. We all learned it as kids but now we are experts and hence looking at numbers is like breathing and any new convention raises confusions, this is natural :)

A number in base-N written as

$$(a_n a_i \dots a_1 a_0)_N$$

where each digit a_i is a symbol from 0-N-1 (N symbols, with a symbol for nothing!); This means, that read right to left, each digit represents a value (place value!)

$$(a_n a_i \dots a_1 a_0)_N = a_n \times N^n + a_i \times N^i \dots + a_1 \times N^1 + a_0 \times N^0 \quad (1)$$

For example, let us consider most familiar system to us, the decimal (base-10) system

$$(1729)_{10} = 1 \times 10^3 + 7 \times 10^2 + 2 \times 10^1 + 9 \times 10^0$$

Please note that right hand side just expresses the value of the number on the left-hand side. Decimal system being most convenient for us, we used decimal representation. We might as well represent the value of this number by drawing 1729 stones/lines as we used to do as kids, write it in Roman as MDCCXXIX etc. Bottomline is, we know the value denoted by a number $(1729)_{10}$ is well-defined in place value system.

When we say we are converting a number from binary to decimal. The number/value remains the same, we just change the notation from base-2 to base-10.

For example:

$$(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

To reiterate, the right-hand side merely represents the value of the number $(1101)_2$ based on the place value system. We can write that value in most familiar form, in decimal as

$$(1101)_2 = (13)_{10}$$

Note that $(1101)_2 = (13)_{10}$ are same numbers! They represent the same value (think of counting stones!) They are just written in different bases and knowing the rules of writing a number in a given base-N as expressed in Eq.1, we can convert between bases.

Addendum: The inverse problem of converting a base-10 notation (decimal) into base-2 number (binary) has been discussed in labs. This is also based on the key concept of Eq.1. for an explanation, please see this link (Thanks to a student posted note @258 on Piazza!)

https://math.stackexchange.com/questions/2463620/why-does-the-method-of-converting-from-decimal-binary-by-taking-remainder-work

Q: How is hexadecimal number system related to binary?

Binary system is the most fundamental unit for machine logic and memory. It represents a bit (a transistor state: ON/OFF or 0/1). Other more convenient units are 8-bit (recall old Nintendo games!) or 16-bit = 1 byte memories. To represent states for a 8-bit or a 16-bit system, *octal* and *hexadecimal* systems were developed subsequently.

Let us look at hexadecimal representation as an example. We recall our discussion of Eq.1, for a number in hexadecimal i.e. base-16 representation, we need 16 symbols. Now 0-9 are easily known to us from decimal, for the rest, we use 10=A, 11=B, 12=C, 13=D ,14=E and 15=F.

With these conventions, which are all just application of general principle Eq.1, we now can see that the largest number which a single symbol can represent in hexadecimal system is $(\mathrm{F})_{16} = (15)_{10} = (1111)_2$

Hence a 4-bit number can be represented by only one symbol in hexadecimal! This makes reading 1 byte much easier. Therefore, a 16-bit number or a byte can be represented by only 4 symbols in hexadecimal!

$$(1101010111001111)_2 = (1101\ 0101\ 1100\ 1111)_2 = (\mathrm{D5CF})_{16}$$

Q: How do we convert negative/rational/irrational/complex numbers to binary?

Since these concepts are not directly needed in the course, please Google following keywords to see.

Negative numbers:
Keywords: 1's complement, 2's complement.

Check out this video!

https://www.youtube.com/watch?v=lKTsv6iVxV4

Rational/Irrational Numbers:

A note here: a computer will represent a number only with finite precision! Hence an irrational number with infinite digits, will be approximated only to the extent a variable memory allows, hence it will be truncated. Thus rational and irrational numbers go into the same class, are represented by what are known as *floating point* representation.

Keyword: floating point representation in binary

Check out this video!

https://www.youtube.com/watch?v=PZRI1IfStY0

Complex Numbers:

As you will learn, complex numbers can in most simplest terms, treated as a tuple of 2 numbers, real and imaginary parts. And every operation can be defined in terms of those two numbers. Hence a simple representation of complex number is just using a tuple of two real binary numbers. However an interesting piazza link was shared where a paper describes another method. See @169.