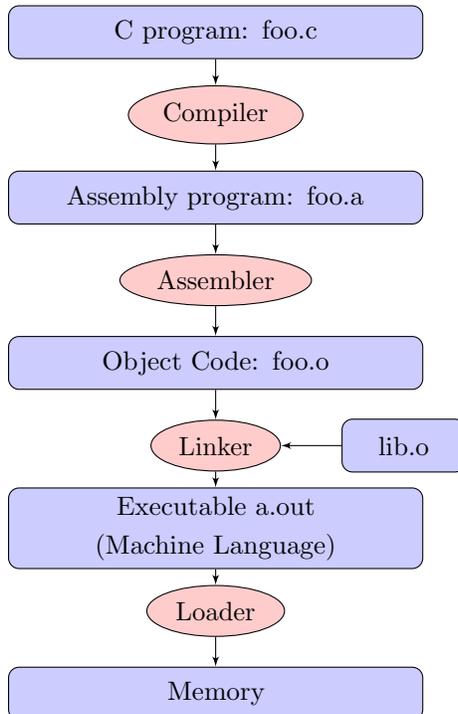


1 CALL

The following is a diagram of the CALL stack detailing how C programs are built and executed by machines:



- 1.1 What is the Stored Program concept and what does it enable us to do?
- 1.2 How many passes through the code does the Assembler have to make? Why?
- 1.3 Describe the six main parts of the object files outputted by the Assembler (Header, Text, Data, Relocation Table, Symbol Table, Debugging Information).
- 1.4 Which step in CALL resolves relative addressing? Absolute addressing?

1.5 What does RISC stand for? How is this related to pseudoinstructions?

2 Floating Point

The IEEE 754 standard defines a binary representation for floating point values using three fields.

- The *sign* determines the sign of the number (0 for positive, 1 for negative).
- The *exponent* is in **biased notation**. For instance, the bias is 127 for single-precision floating point numbers.
- The *significand* or *mantissa* is akin to unsigned integers, but used to store a fraction instead of an integer.

The below table shows the bit breakdown for the single precision (32-bit) representation. The leftmost bit is the MSB and the rightmost bit is the LSB.

1	8	23
Sign	Exponent	Mantissa/Significand/Fraction

For normalized floats:

$$\text{Value} = (-1)^{\text{Sign}} * 2^{\text{Exp}-\text{Bias}} * 1.\text{significand}_2$$

For denormalized floats:

$$\text{Value} = (-1)^{\text{Sign}} * 2^{\text{Exp}-\text{Bias}+1} * 0.\text{significand}_2$$

Exponent	Significand	Meaning
0	Anything	Denorm
1-254	Anything	Normal
255	0	Infinity
255	Nonzero	NaN

Note that in the above table, our exponent has values from 0 to 255. When translating between binary and decimal floating point values, we must remember that there is a bias for the exponent.

2.1 How many zeroes can be represented using a float?

2.2 What is the largest finite positive value that can be stored using a single precision float?

2.3 What is the smallest positive value that can be stored using a single precision float?

2.4 What is the smallest positive normalized value that can be stored using a single precision float?

2.5 Cover the following single-precision floating point numbers from binary to decimal or from decimal to binary. You may leave your answer as an expression.

- 0x00000000
- 8.25
- 0x00000F00
- 39.5625
- 0xFF94BEEF
- $-\infty$

3 More Floating Point Representation

Not every number can be represented perfectly using floating point. For example, $\frac{1}{3}$ can only be approximated and thus must be rounded in any attempt to represent it. For this question, we will only look at positive numbers.

3.1 What is the next smallest number larger than 2 that can be represented completely?

3.2 What is the next smallest number larger than 4 that can be represented completely?

3.3 Define stepsize to be the distance between some value x and the smallest value larger than x that can be completely represented. What is the step size for 2? 4?

3.4 Now let's see if we can generalize the stepsize for normalized numbers (we can do so for denorms as well, but we won't in this question). If we are given a normalized number that is not the largest representable normalized number with exponent value x and with significand value y , what is the stepsize at that value? Hint: There are 23 significand bits.

3.5 Now let's apply this technique. What is the largest odd number that we can represent? Part 3 should be very useful in finding this answer.