# Minutes

10 March 2015

* Rants about prescriptivism and writing style
* Macro for generating code table automatically:
  + I showed you work I did in setting up macros for writing assembly code in bulk
  + Frustratingly, macros cannot be used as operands in an assembly instruction in NASM
  + You noticed some problems with the code I've written:
    - 'lodsw rax' is invalid because lodsw only works on ax
    - My divide code wouldn't work properly on negatives because I didn't sign extend the numerator. I forgot which instruction you mentioned, but CQO can do this
* Instruction set definition comments
  + You answered my questions on the instruction set definition doc on google drive.
  + We spoke about where immediate values (ie., constants) should be in code.
    - Immediate values should stay word-aligned because some architectures don't even support misaligned addressing, and ones that do typically have a penalty
    - If an instruction has a const operand, the next 16bit word will contain a reference to the constant. The format of the reference remains to be seen. The options are:
      * Offset in a data section somewhere
        + Not great because constants might be quite far away from where they're being used
      * Relative displacement from instruction
        + This is what we settled on
        + The next 16bit word after the instruction is a relative offset to a 64-bit value. So the constant has address pc+mem[pc+2]
      * Right after the instruction, double quad word aligned
        + In which case we can access them at address (pc+2+8)&~7 to round up to the next dq word.