# Minutes

3 March 2015

Kevin’s comments in **bold**.

* There are two problems as I see it: one is register mapping. The other one is the code table.
  + Are these related? Because you could easily do:  
    if (operand1==a && operand2==a)  
     add rax, rax  
    elif (operand1==a && operand2==b)  
     add rax, rbx  
    elif (operand1==a && operand2==c)  
     add rax, rcx  
    etc
  + This is obviously bad but I’m having trouble saying why in general (terminology)
  + **They are two separate ideas, but it only makes sense to talk about them together. The code above is worse than just storing registers in RAM. AFAIK, this and JITing are the only ways to map registers**
* Function calls in the instruction set:
  + While typing this next bit I realised I was confuse about where local variables are. Are they on a stack, or are they in some sort of list? I will just assume stack variables for this next bit.
  + **During a call instruction, registers are preserved by pushing them to the stack. Then parameters are pushed.** We didn’t talk about return address but I think it’s safe to assume that that will go on the stack too.
* Will the baseline VM be a register machine?
  + **Naturally, because it has to support the same instruction set as the register machine. It should be as similar as possible except in the details that are the subject of this project.**
* External pressures to use specific treatise style, eg., reference style
  + **APA is fine, but it’s easy to change in LaTeX. Don’t be too worried.**
* External pressures to come up with a "methodology"
  + **We can just make something up, but you’ll talk to MC about what he’s doing with his students.**
  + **Probably positivism+deduction**
* Off-agenda items:
  + Maybe we should drop floats and just use ints
  + **Not all 16 registers will be mapped, because it will make the code table explode in size**
  + **The order that the instructions are implemented in the code table might be important to performance. I can test this as part of the project. It might matter because of how the cache works (8-way caching –** I’ll research this more before next week**)** I could talk at length about how caching works in the treatise.
  + **Limitations aren’t limitations on the thing we make, it’s the limitations on the results of the study.**
* For next time:
  + I’ll think about how to make a test framework to verify the correctness of code tables
  + Will work on a macro system or script to automatically write assembly code. I said that it might be difficult to implement everything with macros because some logic about register shuffling is quite complicated (well for a macro anyway)
  + Will research about caching techniques
* Admin matter
  + This meeting will move to 11:30 – 12:30 so I can get to the programming test on time.