TIP #1: OBJECT CREATION IS BAD

Excessive object instantiation (especially of short-lived objects) will cause poor performance. This is because object churn causes frequent young generation garbage collections, and young generation garbage-collection algorithms are mostly of the "stop-the-world" type.

Microcoding: Less Is More or More Is Less

So now we have a GC-less decoder for parsing the IA-32 instruction stream, but we have not discussed what such a decoder should decode to. The IA-32 architecture is not a fixed-length instruction system; instructions range in length from a single byte to a maximum of 15 bytes. Much of the complexity in the set is down to the plethora of memory-addressing modes that can be used for any given operand of an instruction.

The initial highest level of factorization splits each operation into four stages:

Input operands
: Loading the operation's data from registers or memory.

Operation
: Data processing on the input operands.

Output operands
: Saving the operation's results out to registers or memory.

Flag operations
: Adjusting the flag register bits to represent the result of the operation.

This factorization into operands and operations allows us to separate the simplicity of the operation from the complexity of its operands. An operation such as `add eax,[es:ecx*4+ebx +8]` is initially factorized into five operations:

```
load eax
load [es:ecx*4+ebx+8]
add
store eax
updateflags
```

It is immediately clear that `load [es:ecx*4+ebx+8]` is a far from simple operation, and easily could be factorized into a number of smaller elements. In fact, for this addressing format alone there are:

- Six possible memory segments
- Eight possible index registers
- Eight possible base registers