Sprockell

A **s**imple **proc**essor in Has**kell**

Just like real hardware?

- No…
  - pipelining
  - caches
  - hyperthreading
  - …
- So really, a simple processor!

Source?

- https://github.com/martijnbastiaan/sprockell
- Including wiki, docs, ...
- Referred to as source

- Report found issues at github!
  - (You can even submit pull requests ;-))

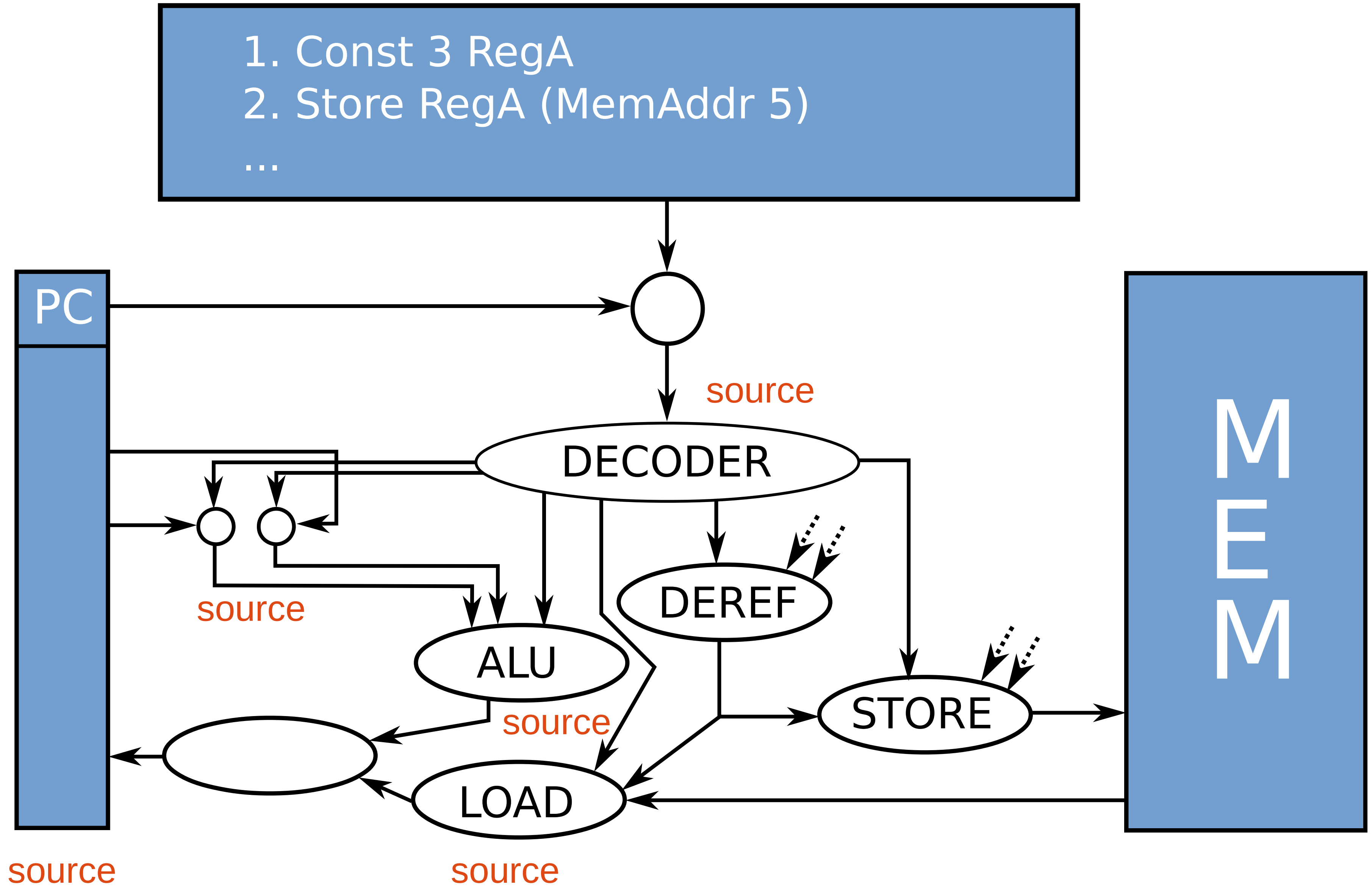
Source layout

- src/Sprockell/
  - TypesEtc.hs
  - Sprockell.hs
  - System.hs
  - Components.hs

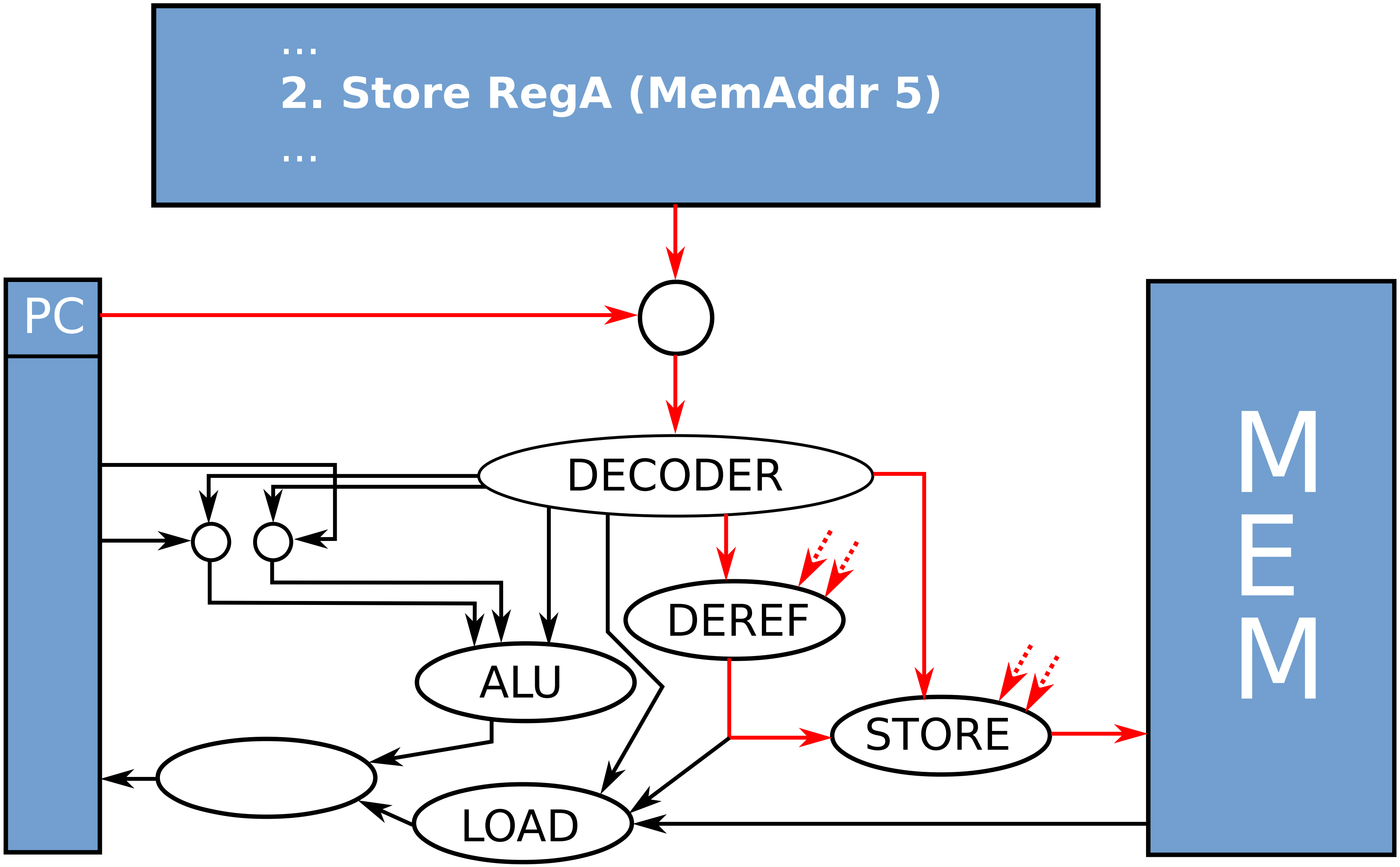
From simple..

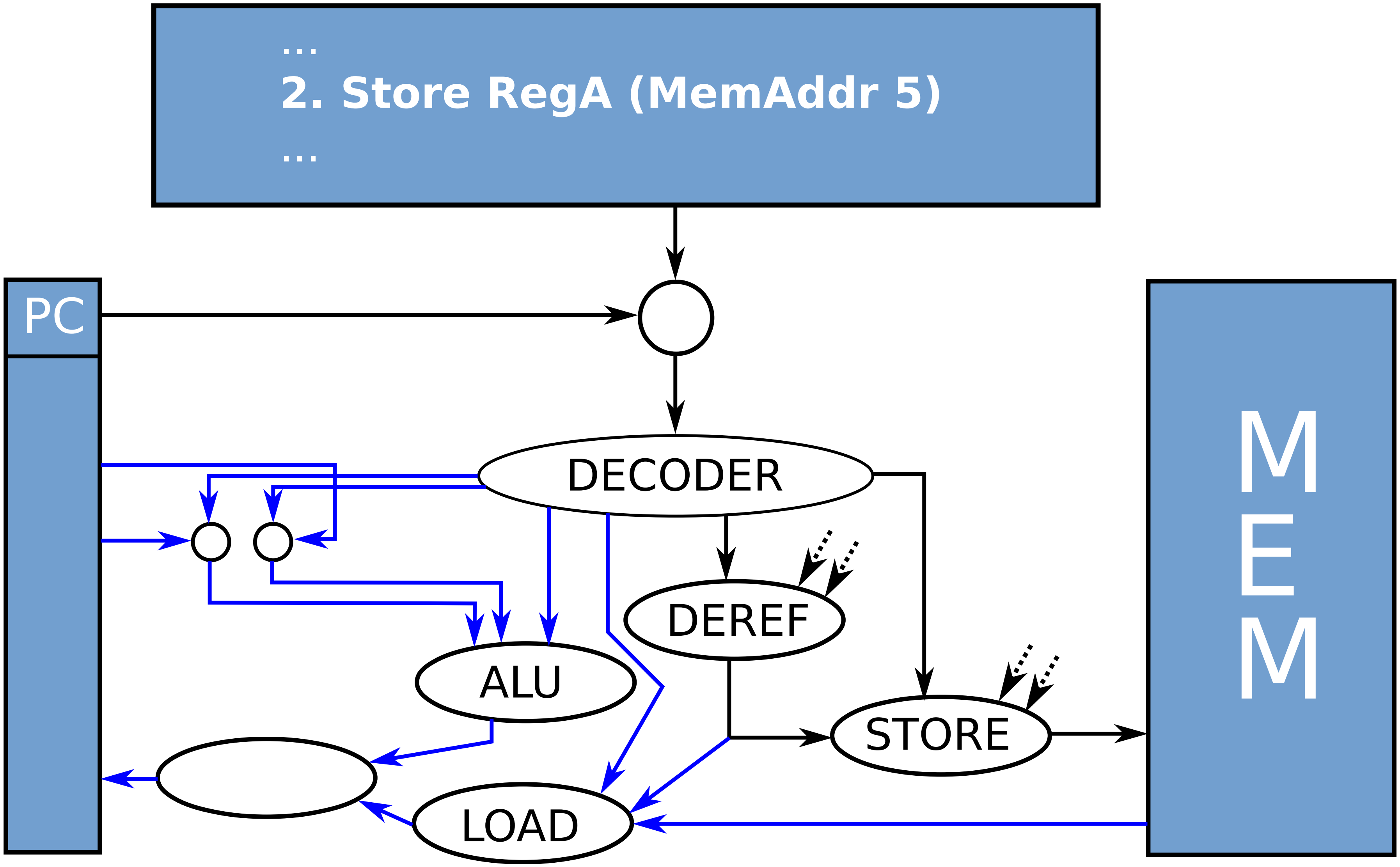
..to possibly magic

- src/
  - Demo*.hs
  - Tests.hs



* Not all components are shown


...
2. Store RegA (MemAddr 5)
...
PC
DECODER
DEREF
ALU
STORE
LOAD
MEM


...
2. Store RegA (MemAddr 5)
...
PC
DECODER
DEREF
ALU
STORE
LOAD
MEM

????

Decoder emits defaults

- Store: to register Zero
- Deref: from register Zero
- ALU: XOR with zero
- …
- source

Challenge

- Write a (useful) instruction which employs multiple components at once
  - (+optimizer?)

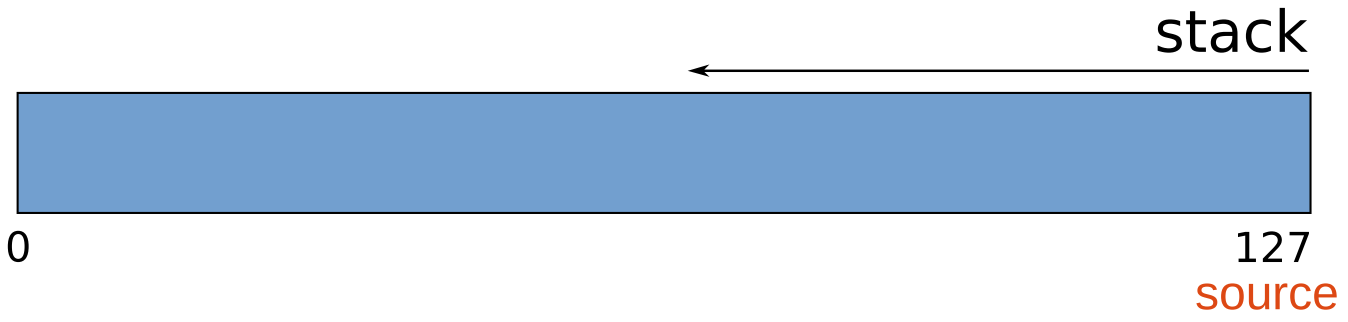
Registers

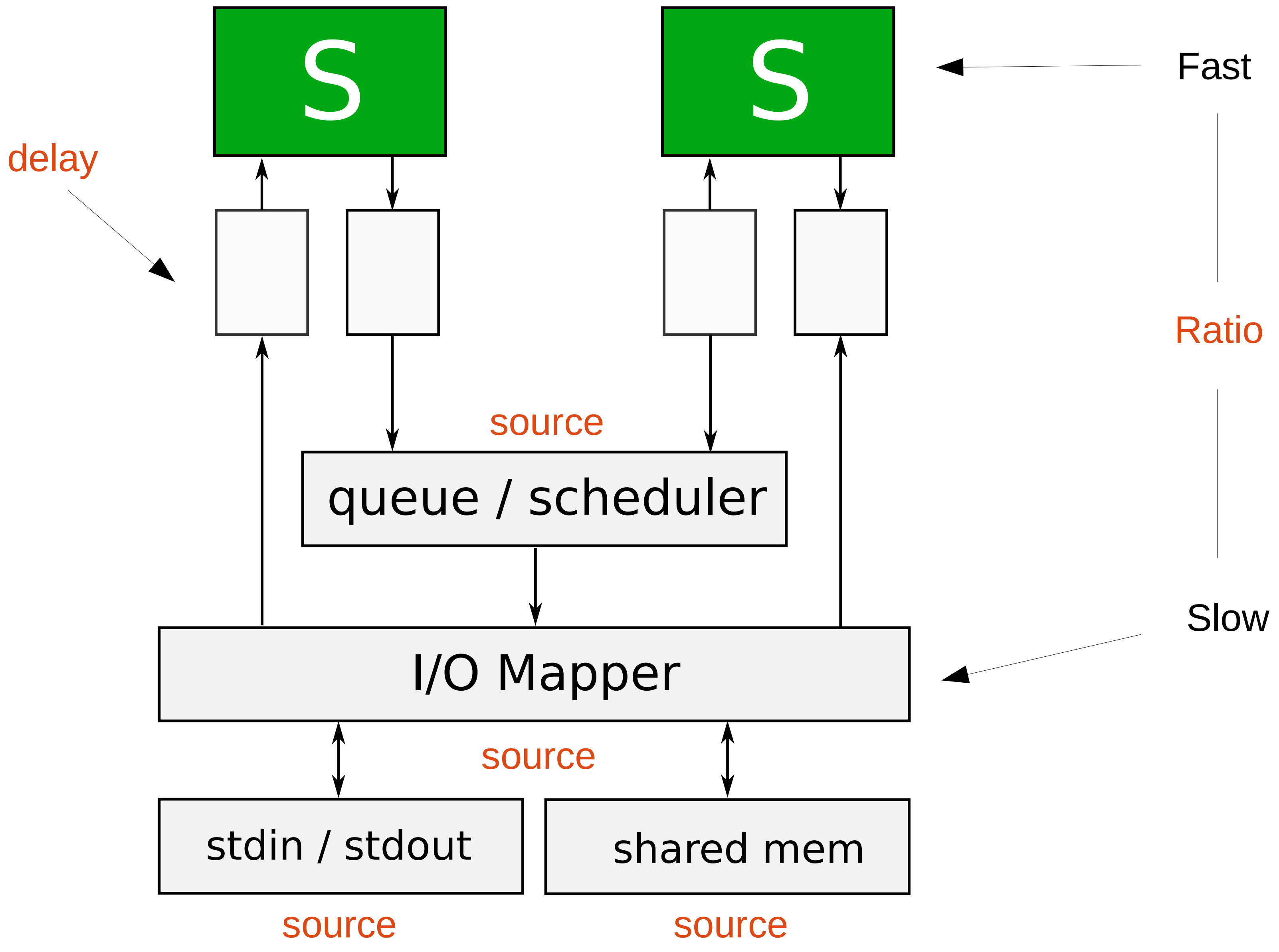
- Zero
- PC → Program Counter
- SP → Stack Pointer
- SPID → Sprockell ID
- RegA, RegB, …, RegE
- source

Instructions

- source
- wiki

Stack




S
S
Fast
delay
Ratio
source
queue / scheduler
Slow
I/O Mapper
source
stdin / stdout
shared mem
source
source

Running it

- nSprockells :: Int
- prog :: [Instruction]
- debugFunc :: SystemState → String

- *run* nSprockells prog
- *runDebug* debugFunc nSprockells prog

Both pick a seed at random

Random behaviour

- *runWithSeed*
- *runDebugWithSeed*

- Same as *runSeed* / *runDebug*, but first argument a seed

Small example

- ghc example.hs
  ./example

```haskell
import Sprockell.System

prog :: [Instruction]
prog = [
         Const 5 RegA
       , Store RegA (Addr 5)
       , EndProg
      ]


main = run 2 prog
```

Changing the Sprockell

- You're free to do it!

Type	**Read**	**Write**
Registers	!	<~
Memory	!!!	<~=

- Example: `mem <~= (addr, value)`

End notes..

- Please report bugs on github if you find one
- Detailed instructions are on the wiki