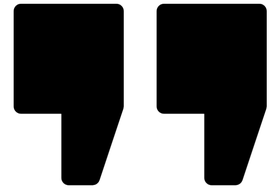
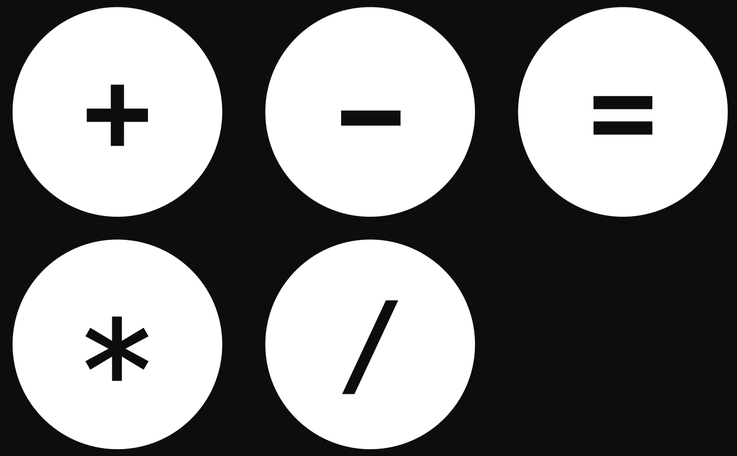
Within the **DEFINED CONTEXT**, something is **Atomic**, **IF** it **WILL HAPPEN** in its **ENTIRETY**, without anything else **HAPPENING SIMULTANEOUSLY** in the **SAME CONTEXT**.



operand operation operand result

operand operation result



i++

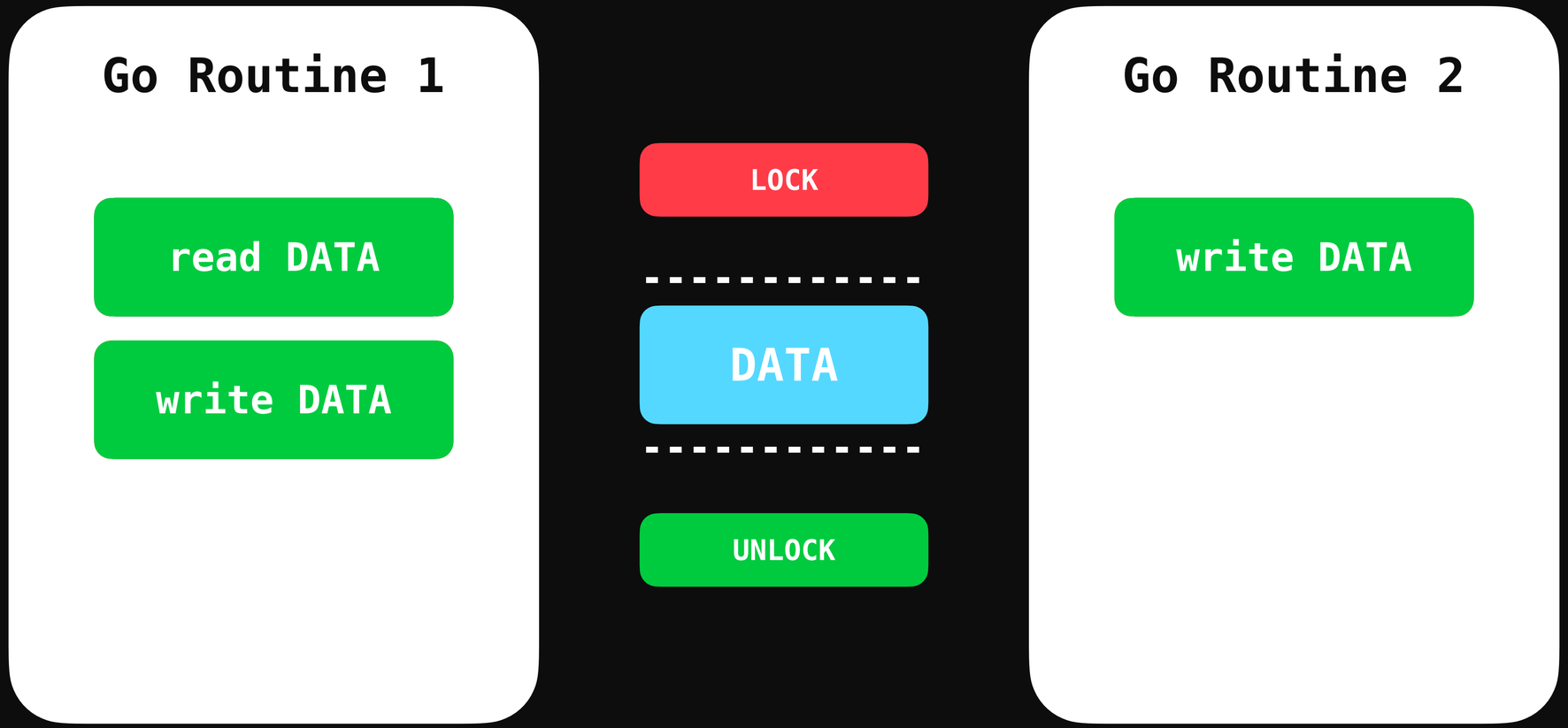
get value of **i**

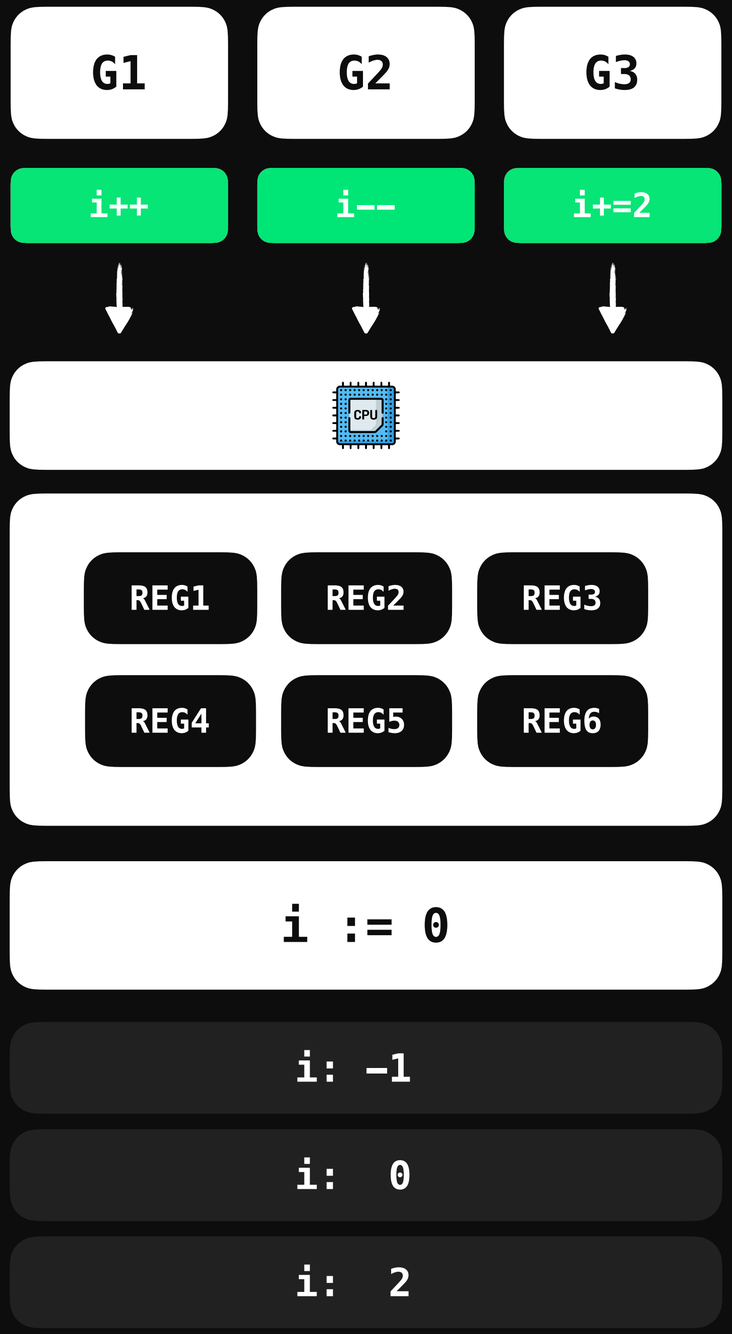
increment value of **i**

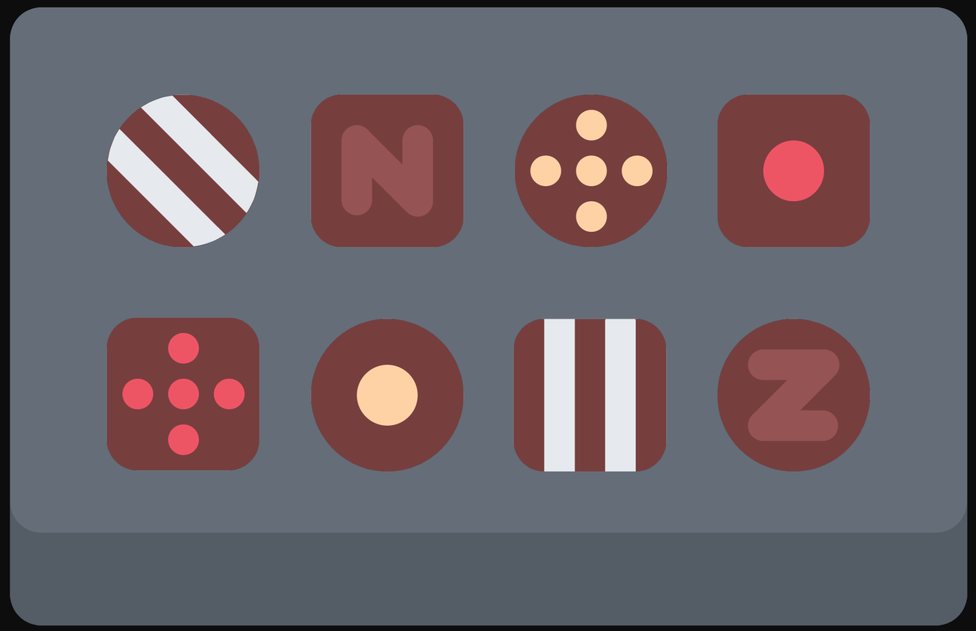
store value of **i**

INDIVISIBLE

UNINTERRUPTIBLE


Go Routine 1
read DATA
write DATA
LOCK
DATA
UNLOCK
Go Routine 2
write DATA

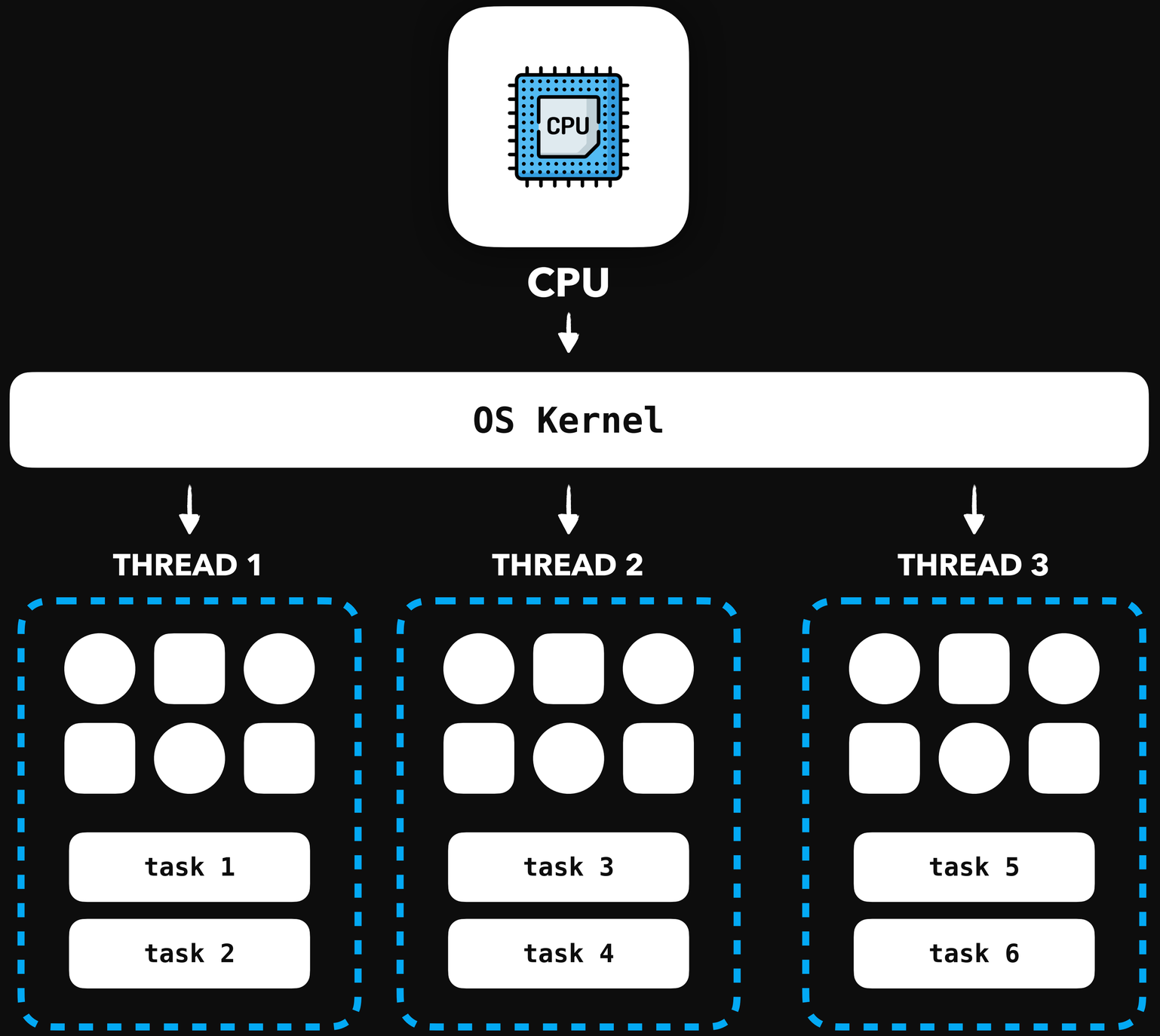

G1
G2
G3
i++
i--
i+=2
CPU
REG1
REG2
REG3
REG4
REG5
REG6
i := 0
i: -1
i: 0
i: 2

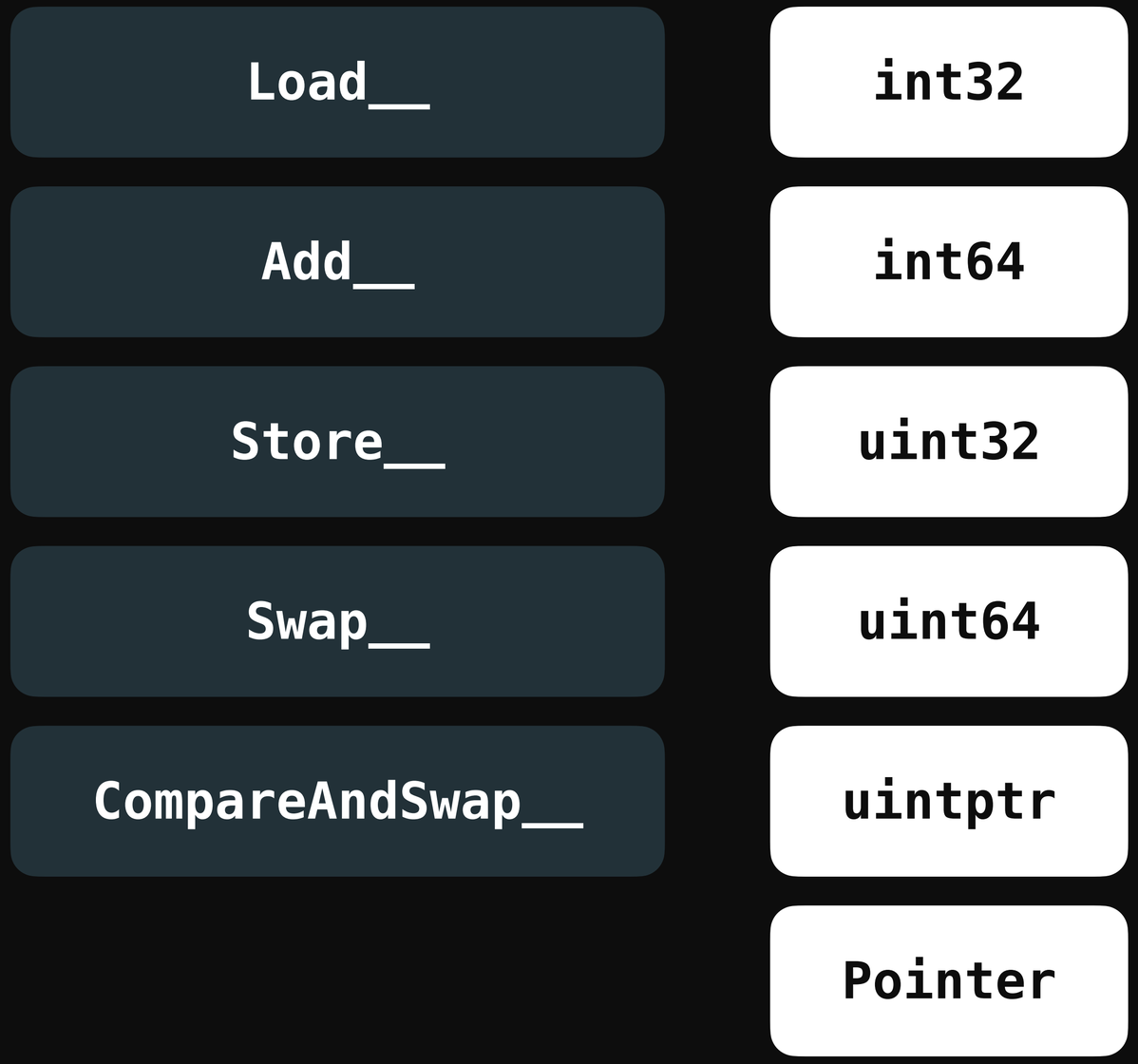


EMPTY REGISTER

FULL REGISTER

- store intructions
- store results
- manipulate data


CPU
CPU
OS Kernel
THREAD 1
THREAD 2
THREAD 3
task 1
task 2
task 3
task 4
task 5
task 6


Load__
Add__
Store__
Swap__
CompareAndSwap__
int32
int64
uint32
uint64
uintptr
Pointer

atomic.Value

Store()

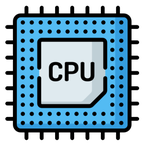
Load()

main.o

linker

exec

01010101010101
10001101001001



386

amd64

arm

arm64

mips

mipsle

mips64

mips64le

ppc64

ppc64le

riscv64

s390x

wasm

obj library

Go ASM

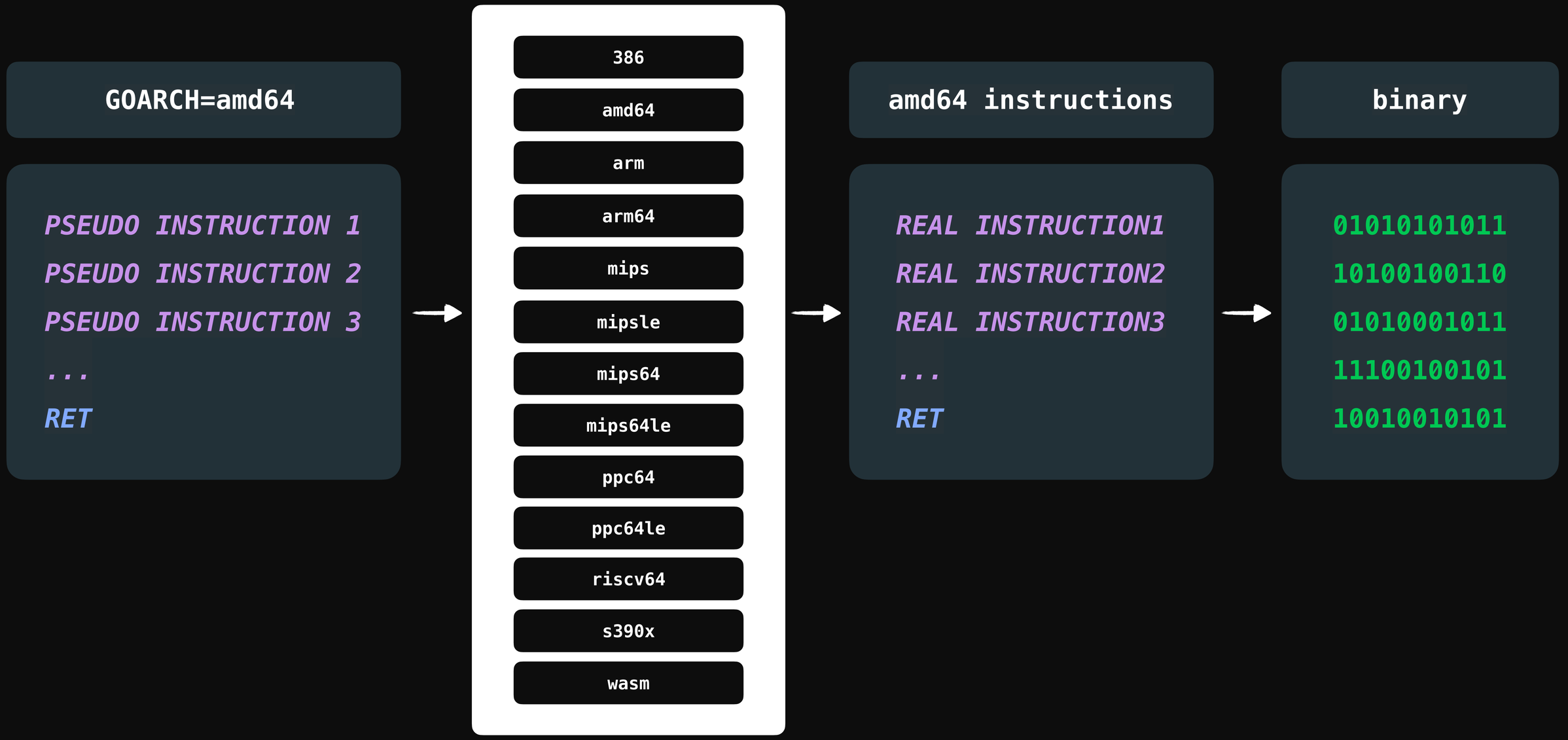
main.s

Go Compiler

$GOOS $GOARCH


Go App


go build
COMPILE
main.go
main.o
ASM
add32_amd64.s
add32.o
ASM
add64_amd64.s
add64.o
LINK
exec


GOARCH=amd64
PSEUDO INSTRUCTION 1
PSEUDO INSTRUCTION 2
PSEUDO INSTRUCTION 3
...
RET
386
amd64
arm
arm64
mips
mipsle
mips64
mips64le
ppc64
ppc64le
riscv64
s390x
wasm
amd64 instructions
REAL INSTRUCTION1
REAL INSTRUCTION2
REAL INSTRUCTION3
...
RET
binary
01010101011
10100100110
01010001011
11100100101
10010010101


64bit
32bit
0101010101101010
1010010011001011
0101000101101010
1110010010101101
0101010101101010
1010010011001011
ALU OPS: 64
ALU OPS: 32
RAM: 3.25GB
RAM: 16EB =>16B GB

- Mostly works with numbers
- atomic.Value is generic
- atomic.Value has limited context
- atomic.Value is not always atomic