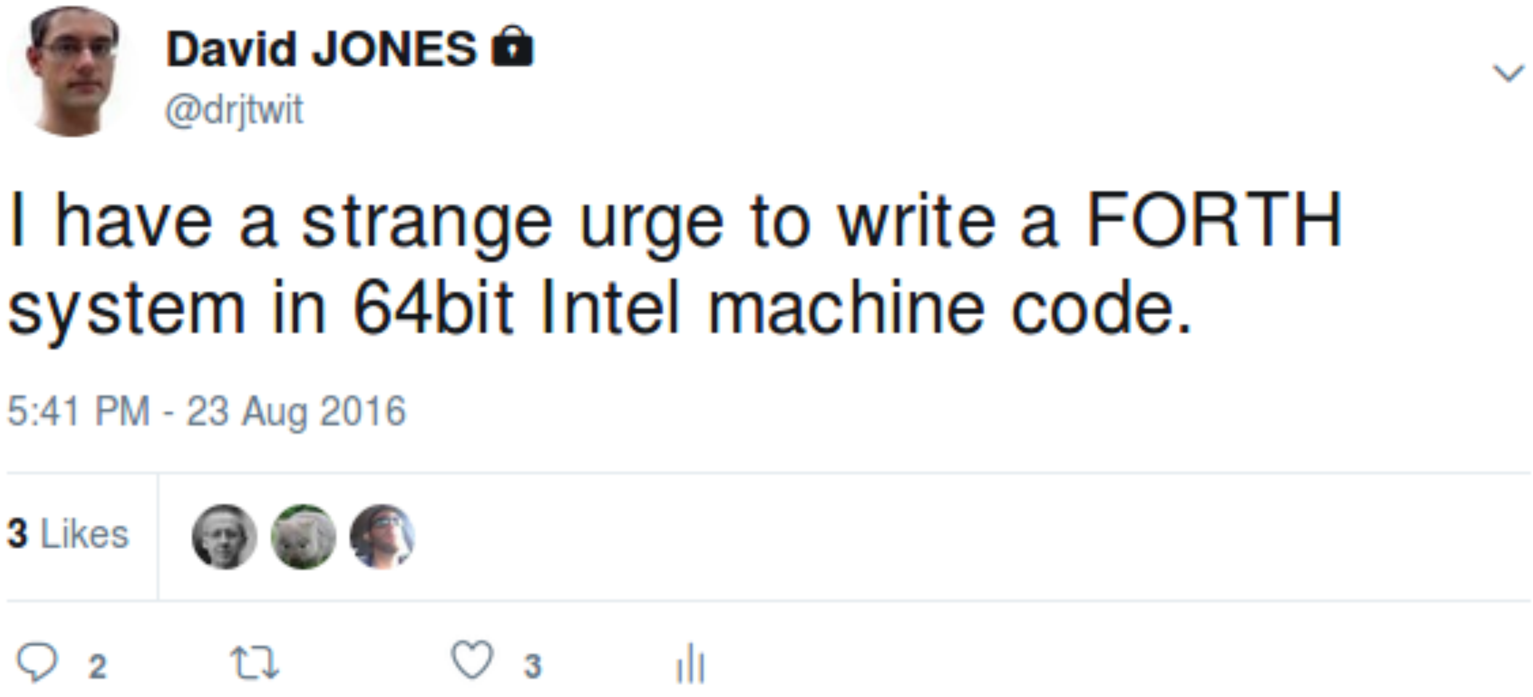
what i learned building Forth in 64-bit Intel assembly

David Jones
drj@pobox.com
@drjtwit
github.com/drj11

In The Beginning

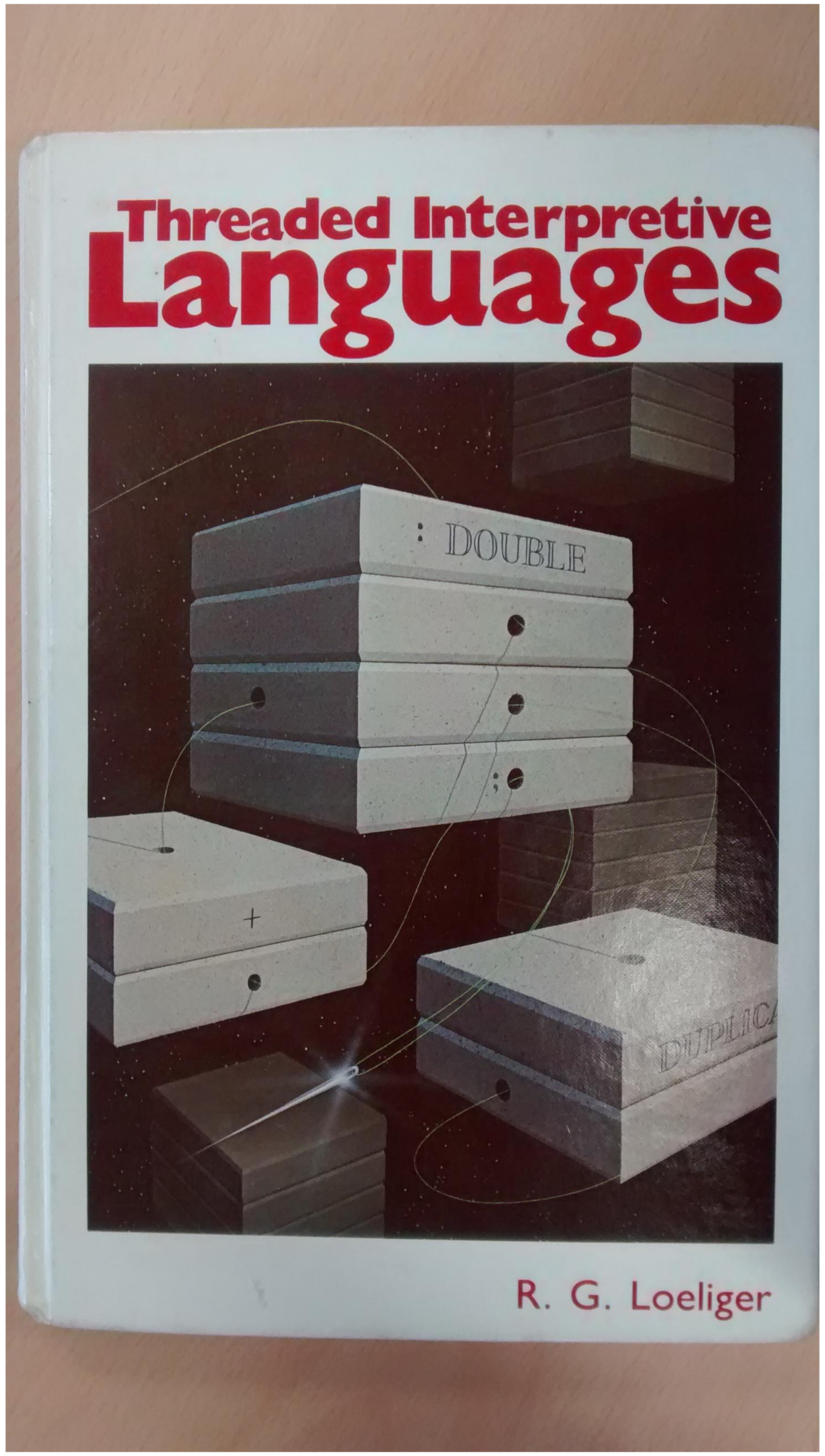


David JONES
@drjtwit

I have a strange urge to write a FORTH system in 64bit Intel machine code.

5:41 PM - 23 Aug 2016

3 Likes



SixtyForth

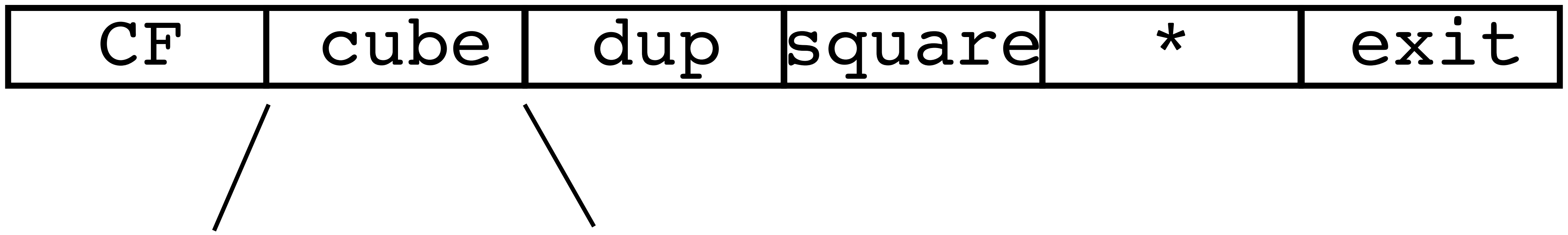
search: sixtyforth drj11

https://github.com/drj11/sixtyforth

*

a threaded word

```
: cube  dup square * ;
```



CF	cube	dup	square	*	exit

```
  1 cell
= 64 bits
= 8 bytes
```

executing cube

```
: square  dup * ;
             ^
: cube  dup square * ;
                  ^
5 cube .
```

data stack

5	5	5

return stack

CUBE+2	SQUARE+1

*

A Forth VM

(after Conklin and Rather)

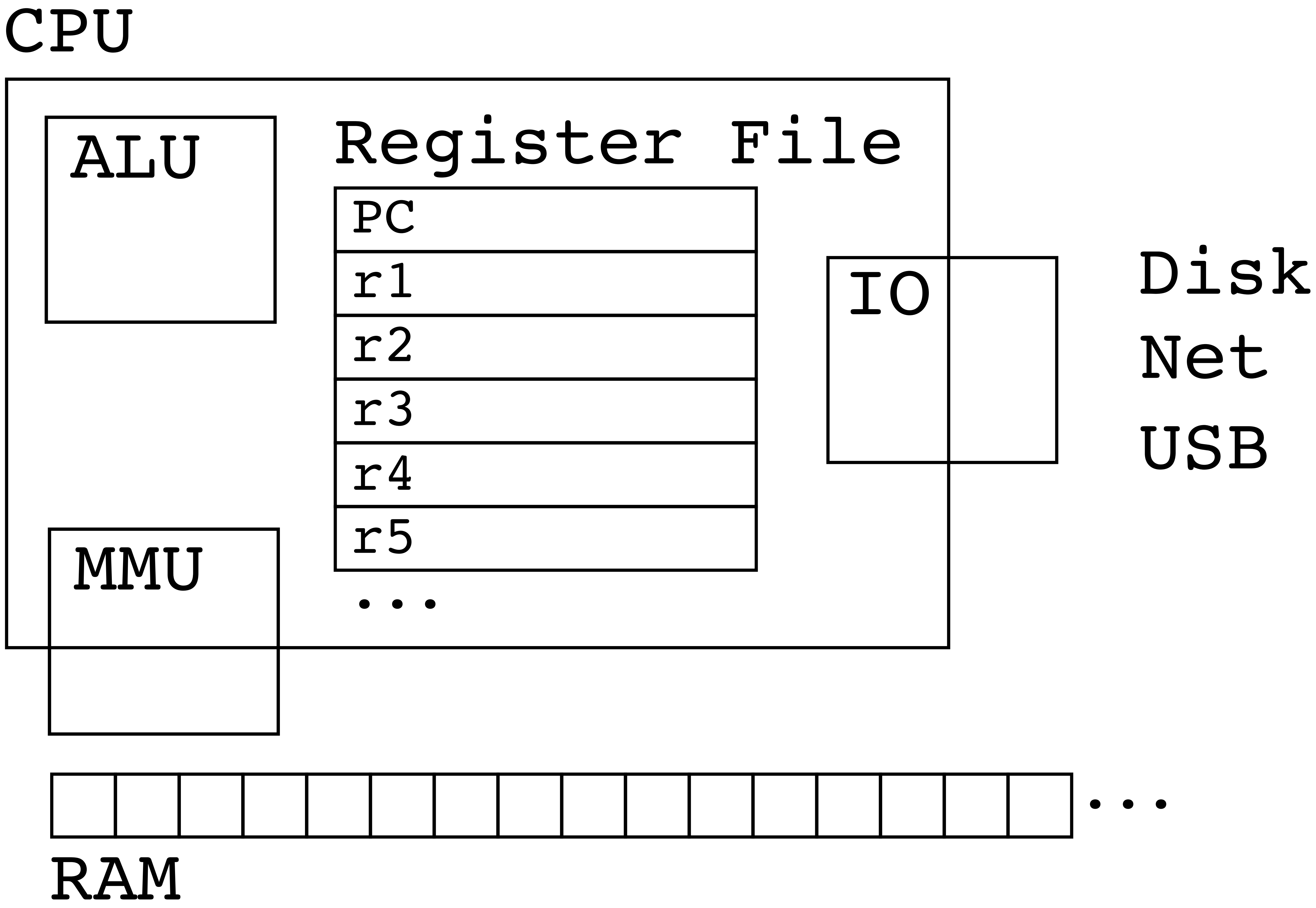
S Data Stack Pointer

R Return Stack Pointer

I Instruction Pointer

W Word Pointer

a generic computer



Intel-64 architecture

(a very simple version)

16 general purpose registers
each register is 64 bits wide

RAX	RBX	RCX	RDX
RSP	RSI	RBP	RDI
R8	R9	R10	R11
R12	R13	R14	R15

modelling the Forth VM

S	Data Stack Pointer	RBP
R	Return Stack Pointer	R12
I	Instruction Pointer	RBX
W	Word Pointer	RDX

Mostly avoiding the 6 registers used by Linux SYSCALL

RDI RSI RDX R10 R8 R9

and the two trashed by SYSCALL

RCX R11

layers

Intel Assembly (NASM)

Threaded Code (NASM)

Forth in non-portable Forth

Forth in portable Forth

key implementation ideas

execution token (xt)

xt is the code field address

code field then body

a vector of xt

Moore's discovery

Small

Sufficient

Implementable

Extensible

factoids

~ 24 different instructions used

mov jmp sub add

how many instructions there are?
nobody knows
one blog says 981 to 3683

binary is ~ 110KB

surprises

The Direction Flag

People on Stack Overflow

RCX is trashed (SYSCALL on Linux)

TCGETS IOCTL buffer size

You can't lie to a tutorial

further reading

Anjana Vakil Python bytecodes

Loeliger threaded languages

Forth programmer's handbook

Chuck Moore the early years

- end -

@drjtwit
https://github.com/drj11