

FPGA and ASIC Forth Market



SVFIG Forth day

November 16, 2024

Christopher Lozinski

EE M.Sc. 9/24

lozinski@PythonLinks.info

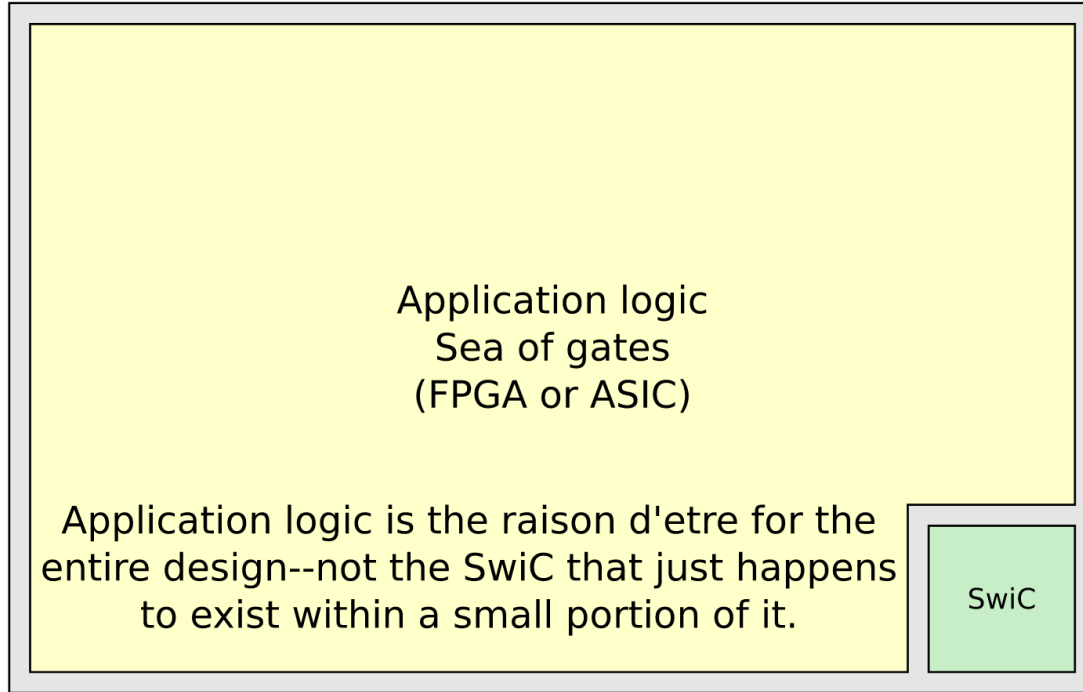
[@PythonLinks@Mastodon.Social](https://mastodon.social/@PythonLinks)

[Forth Discord Server](#)

[Pico-Ice Discord Server](#)

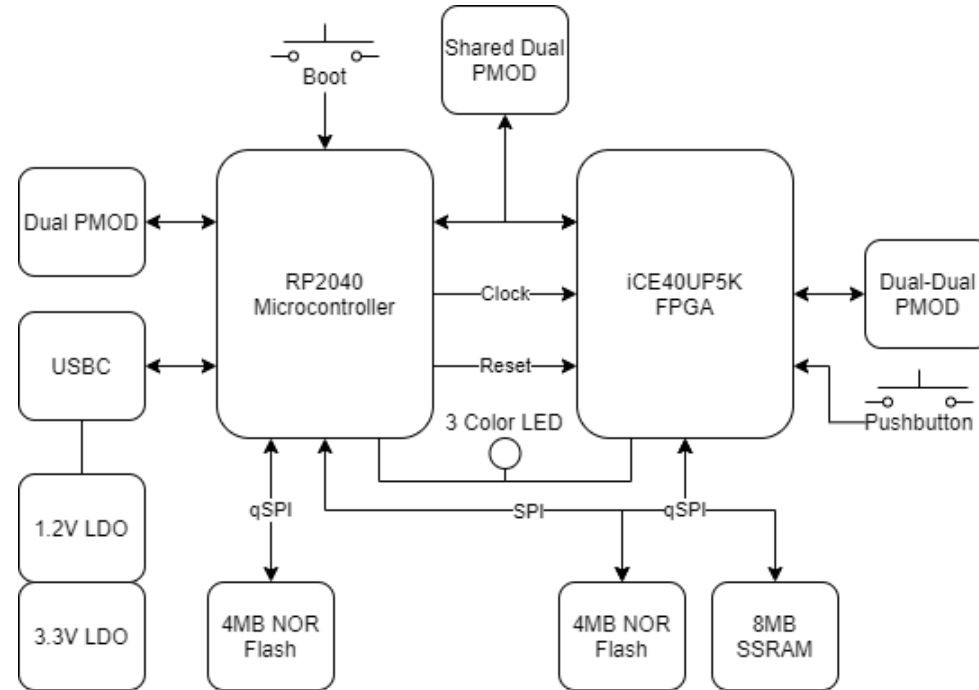
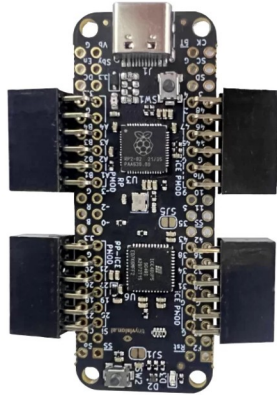
[Hana-1 Discord Channel](#)

Forth Soft Core on FPGA



When the finite state machine gets too complex, engineers move to a soft core cpu. Usually RISC-V, but not Forth.

Open Source Pico-Ice Hardware



My Thesis

OLD ICE40LP-1k	NEW ICE40UP5K	
3K App Words 5K System Words Pseudo dual port $32 * 4\text{kbits} = 128\text{Kbits}$ PSRAM $8\text{k} * 16$ bit words.	10**14 words Single Port 10**14b words Single Port	$30 * 4\text{kbits} =$ 120Kbits of PSRAM 10**14 words Single Port 10**14 words Single Port $4 * 16\text{Kbits} = 1\text{MBit}$ SPRAM

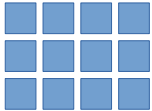
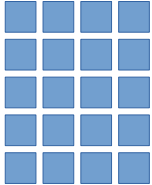
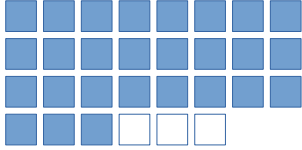
Required Resources

	Stack Depth	LUTs	SPRAMs	BRAMS	Max Frequency (Mhz)	Total Memory (Words)
J1a	15/17	1162		30	?	8192
Mecrisp Ice J1a	32	2141		30	19.45	7680
Hana 1 on BRAMS	256	1183		30	21.60	7168
Hana 1 on SPRAM	256	1043	1-3	3	25.18	16K - 42K

J1 Vs Mecrisp Vs Hana 1

	J1	Mecrisp Ice 16bit PSRAM	Hana 1 16 bit SPRAM
LUTs	1162	2141	1043
Instructions	16	22	26
Shifters	2	5	6
Stack Depth	15/17	32	512
Data Bus		Yes	Yes
SPI		Bit Banged	Shift Register
Timer		Yes	Yes
Max Frequency		17.45 Mhz	23.12 MHz

How is That Possible?

OLD ICE40LP-1k	NEW ICE40UP5K	
 3K App Words  5K System Words Pseudo dual port $32 * 4\text{kbits} = 128\text{Kbits}$ PSRAM $8\text{k} * 16$ bit words.	 10**14 words Single Port 10**14b words Single Port	$30 * 4\text{kbits} =$ 120Kbits of PSRAM 10**14 words Single Port 10**14 words Single Port $4 * 16\text{Kbits} = 1\text{MBit}$ SPRAM

Hana 1, $\frac{1}{2}$ the RISC-V Size

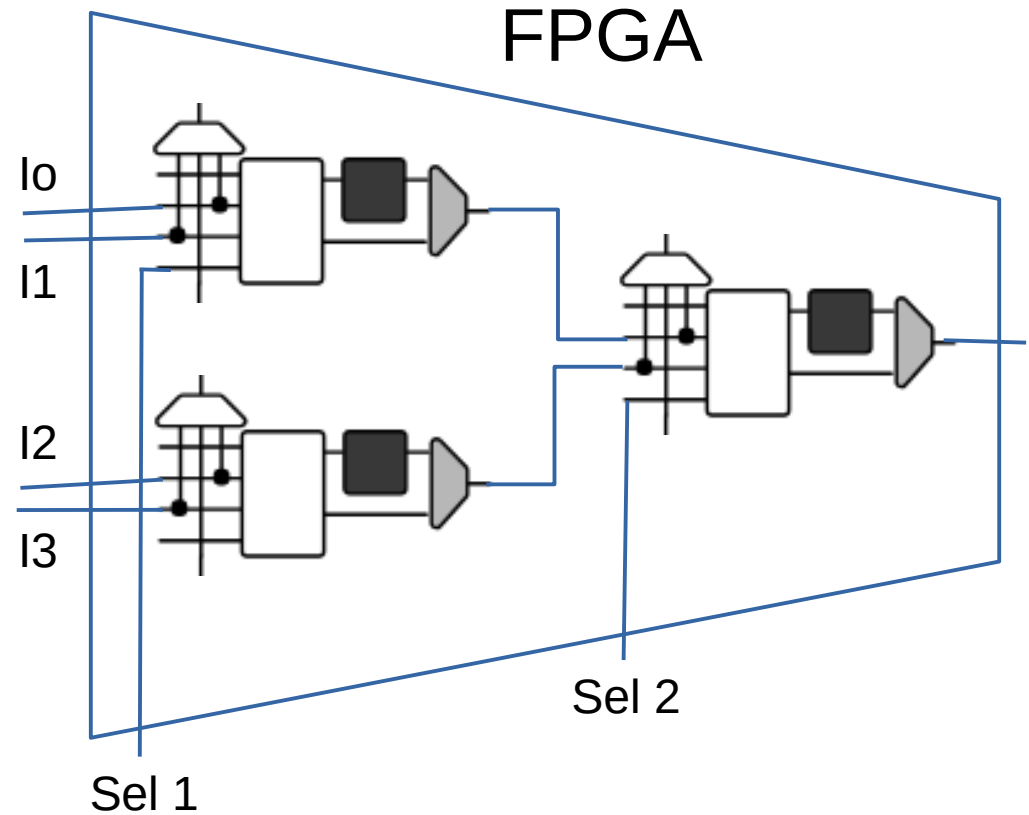
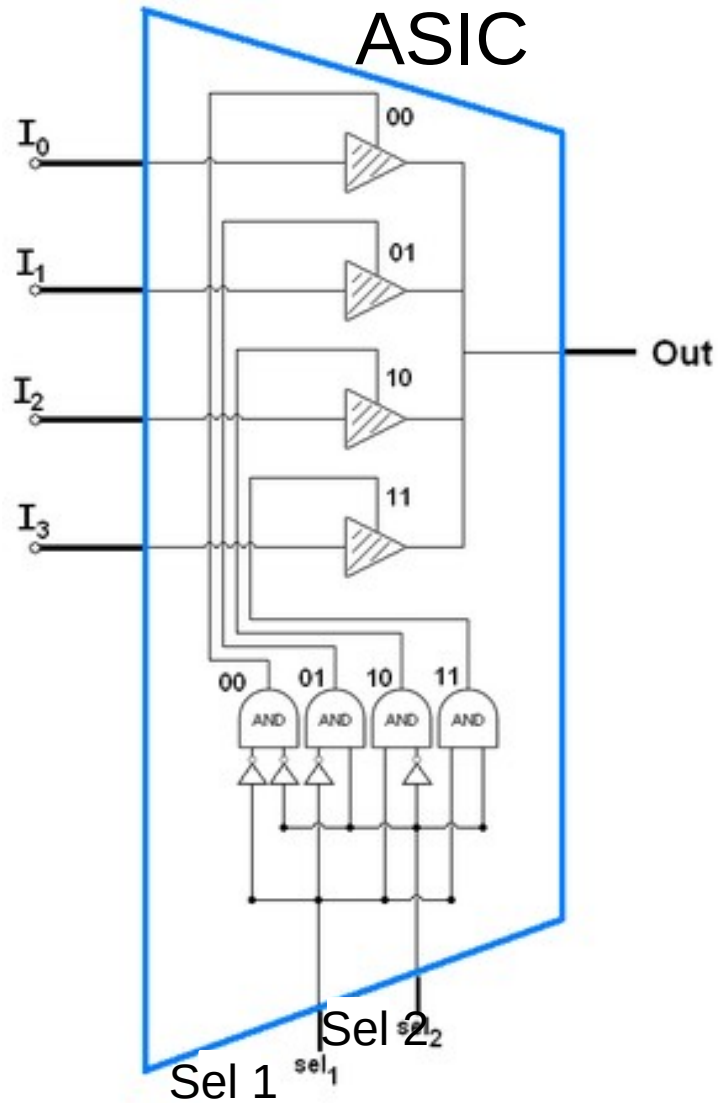
1162 LUTs SwapForth J1a

- 140 LUTs Remove the Memory Multiplexer
- 512 LUTs Use BRAMs for two 16 deep 16 bit stacks

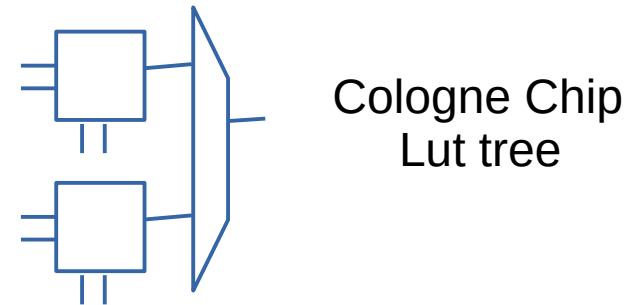
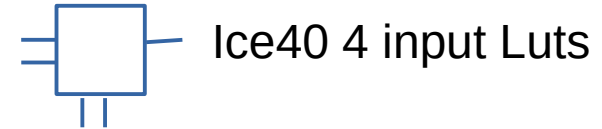
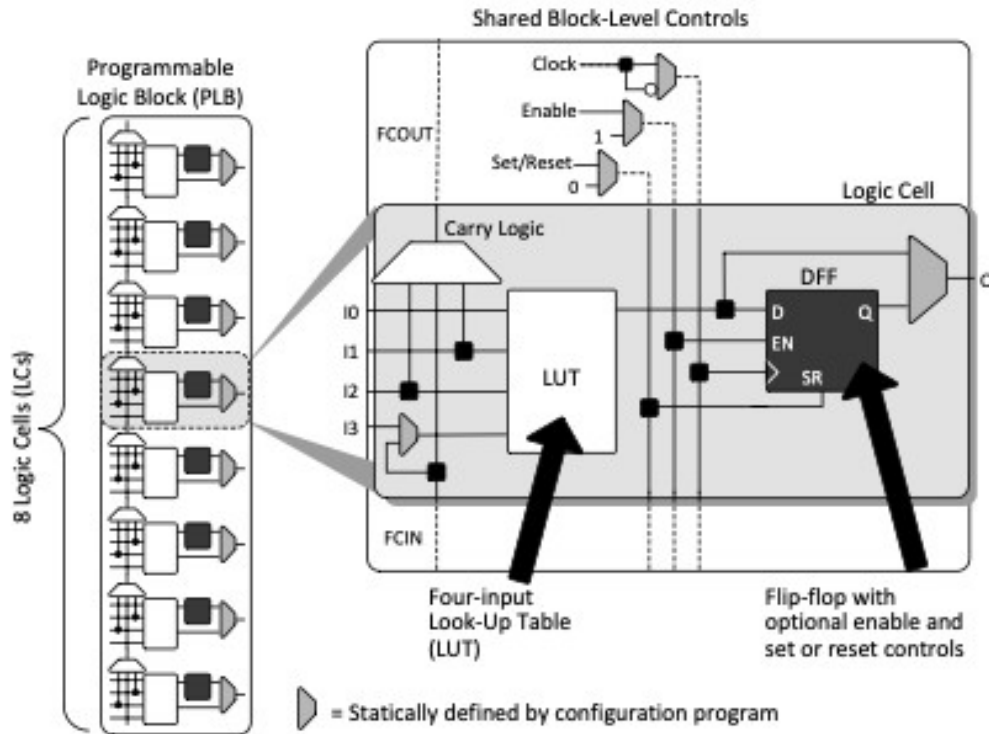
500 LUTs The size of the Bit Serial RISC-V
Serve CPU.

1150 LUTs Typical small RISC-V size.

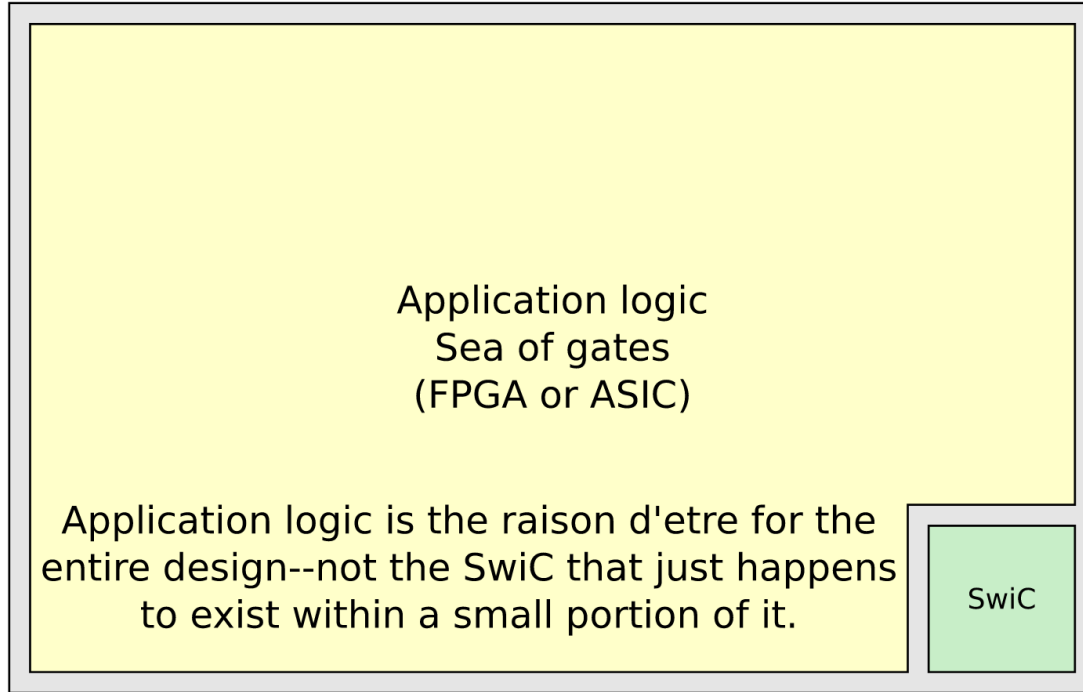
Multiplexers



Programmable Logic Block



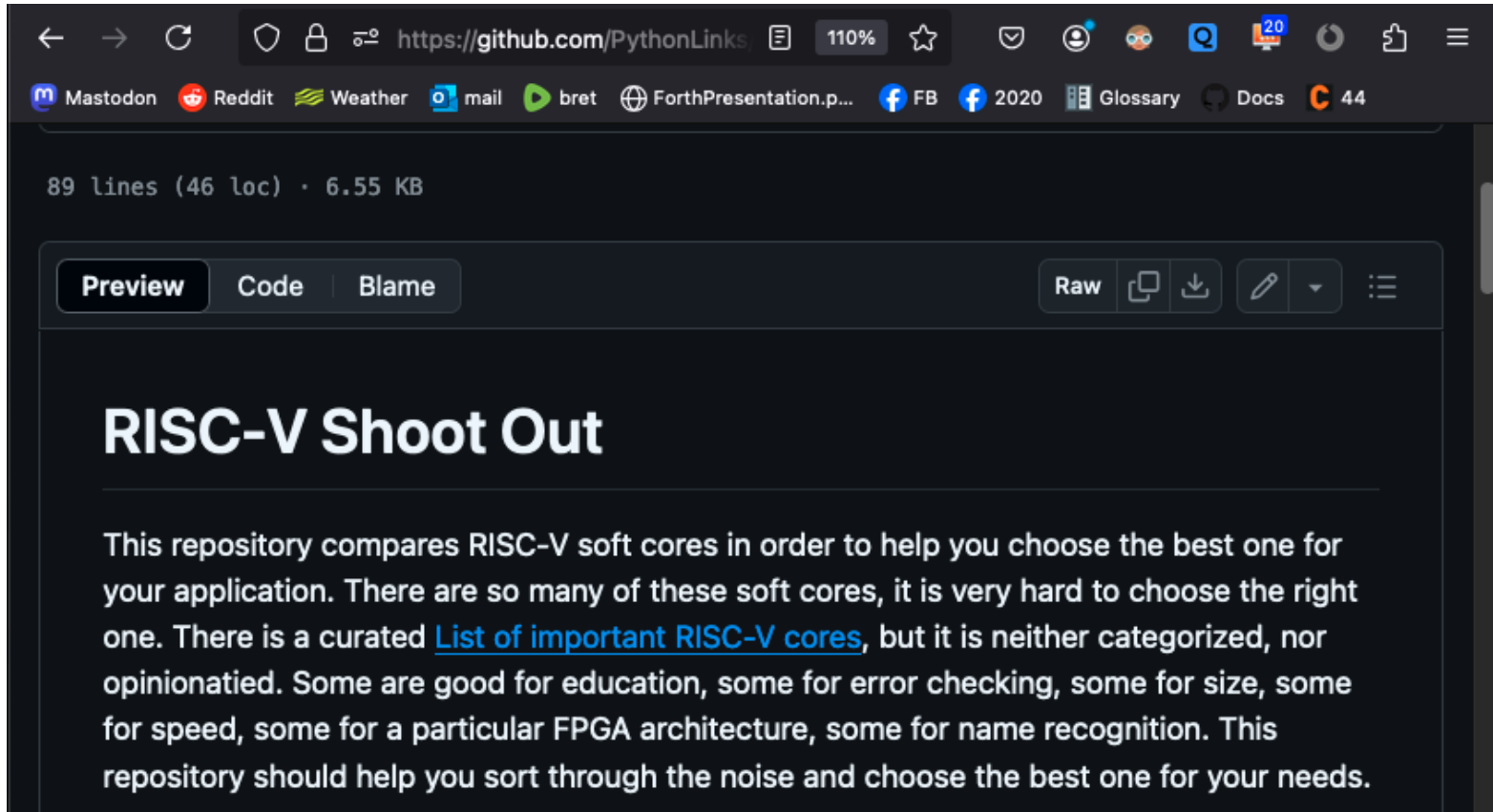
Everyone Wants RISC-V on FPGA



When the finite state machine gets too complex, engineers move to a soft core cpu. Usually RISC-V, but not Forth.

Lots of RISC-V Soft Cores

<https://github.com/PythonLinks/risc-v-shoot-out>



The screenshot shows a web browser displaying the GitHub repository page for 'risc-v-shoot-out' by the user 'PythonLinks'. The browser's address bar shows the URL 'https://github.com/PythonLinks/risc-v-shoot-out' and the page is zoomed in at 110%. The browser's toolbar includes various icons for navigation and social media. Below the browser window, the repository's file statistics are shown: '89 lines (46 loc) · 6.55 KB'. The repository's main content area has a dark theme and features a 'Preview' tab selected, with 'Code' and 'Blame' tabs also visible. To the right of the tabs are buttons for 'Raw', 'Copy', 'Download', 'Edit', and a menu icon. The main heading of the repository is 'RISC-V Shoot Out'. Below the heading, a paragraph of text describes the repository's purpose: 'This repository compares RISC-V soft cores in order to help you choose the best one for your application. There are so many of these soft cores, it is very hard to choose the right one. There is a curated [List of important RISC-V cores](#), but it is neither categorized, nor opinionated. Some are good for education, some for error checking, some for size, some for speed, some for a particular FPGA architecture, some for name recognition. This repository should help you sort through the noise and choose the best one for your needs.'

89 lines (46 loc) · 6.55 KB

Preview Code Blame Raw Copy Download Edit

RISC-V Shoot Out

This repository compares RISC-V soft cores in order to help you choose the best one for your application. There are so many of these soft cores, it is very hard to choose the right one. There is a curated [List of important RISC-V cores](#), but it is neither categorized, nor opinionated. Some are good for education, some for error checking, some for size, some for speed, some for a particular FPGA architecture, some for name recognition. This repository should help you sort through the noise and choose the best one for your needs.

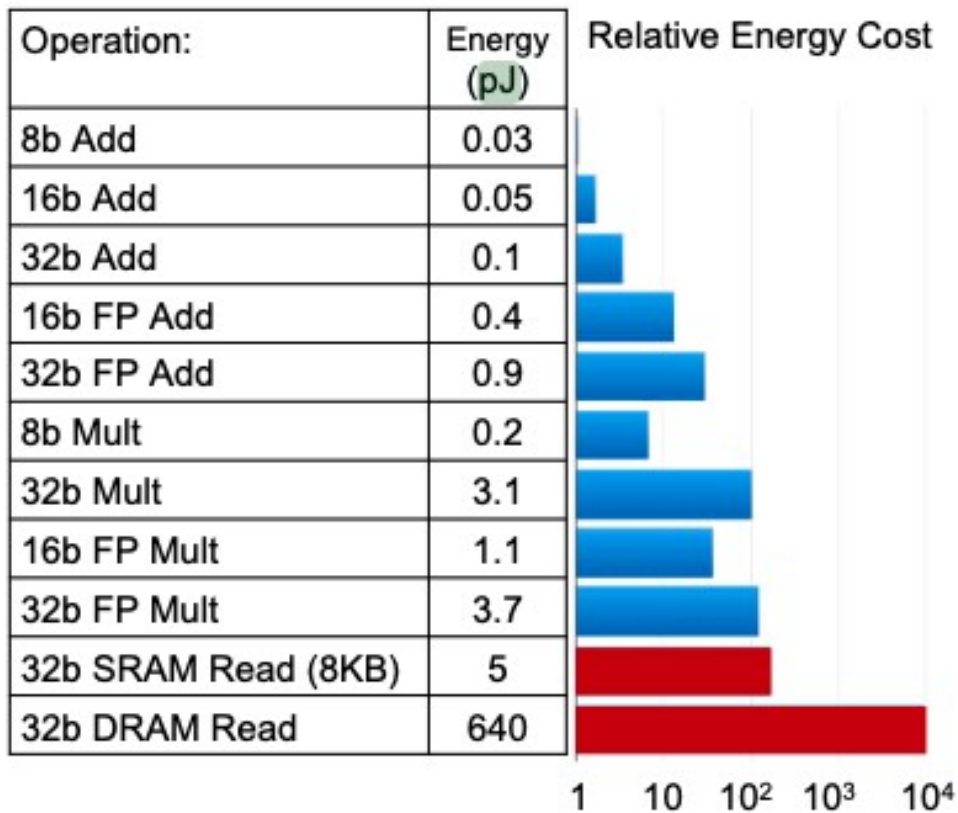
RISC-V Python Vs Forth

Memory Requirements

<u>MicroPython's</u> minimum requirements are 256KByte of ROM and 16KB RAM.	Mecrisp nucleus requires 6.5KBytes + Application space No ROM. On ICE40.
Snek requires 32KBytes ROM + 2 K Byte RAM	11 kBytes of flash and 512 bytes of RAM on MSP430
The Lua interpreter takes 282KBytes and the Lua library takes 470KBytes.	
eLua requires 256KBytes of ROM, + 32 KBytes of RAM.	

ROM is slower than RAM

POWER



Memory access is **orders of magnitude** higher energy than compute

Power For Memory Access



On Chip 5pJ

Off Chip 640 pJ



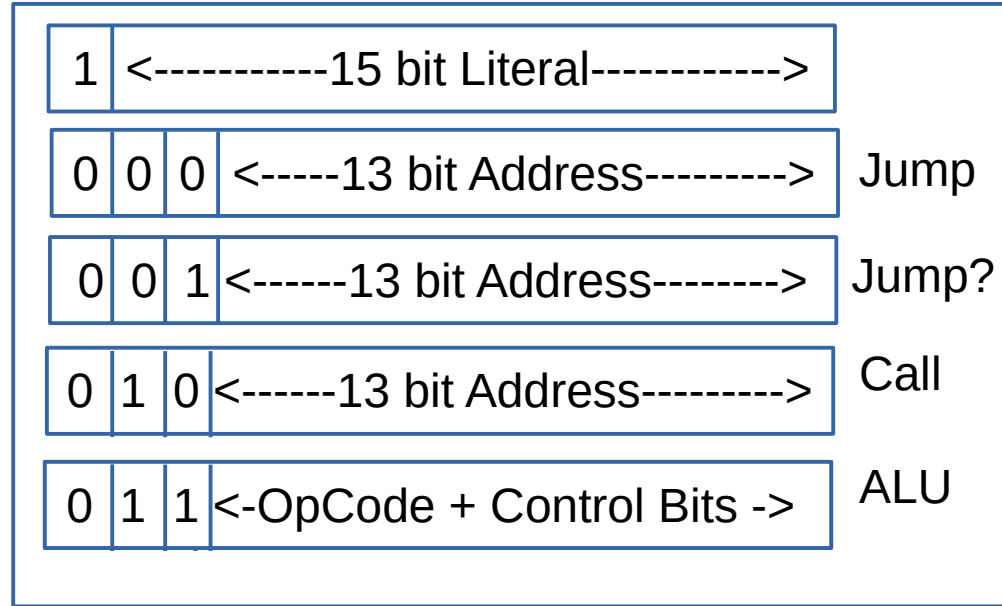
Memory Required

It is hard to compare soft core CPUs which are all very different. Best to compare the same application running on both a register and a stack machine.

Register Machine Version	Stack Machine Version
J1 Video application on 32 bit and 32 register Microblaze in C 16380 Bytes	J1 Video application on J1 in Forth. 6349 Bytes

J1 Vs RISC-V Memory

J1



RISC-V

3 * 5 bits for two operands and a results register and 7->17 bits OpCode = 22->32 bits

Fun7	Reg1	Reg2	Fun3	Result	OpCode
------	------	------	------	--------	--------

Stack Vs Register Machine Memory

Stack Machines



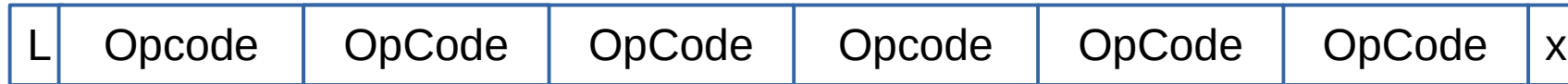
6-8 Bits for a Stack Machine Instruction. Eg: [MicroCore](#) [UXN](#)

L = Literal



3 OpCodes in 16 bits. Eg [EP16](#), [b16](#)

5 opCodes in 32 bits. Eg [EP32](#)



RISC-V

3 * 5 bits for two operands and a results register and 7->17 bits OpCode = 22->32 bits [Risc-V](#)



[Second Generation Stack Computer Architecture](#)

Forth BNF Parser

Program	Implementation Languages	Output	Source Lines	All Lines	Remark
bnfparse	Forth	-	14	16	only Rule.pm (not Lex.pm)
DCG	Prolog	Prolog	68	226	
mop	Perl	-	156	291	
Gray	Forth	bin. Forth	473	754	
kwParsing	Python	bin. Python	1691	2883	If you want an infix Interpreter the Forth BNF parser is only 16 Lines of code.
Coco/R	Modula-2, Coco/R	Modula-2	4005	5106	
mlyacc	ML, mlyacc	ML	4395	6352	
rdp	C, rdp	C	4947	6519	
bison	C	C	7806	11258	
ell	Modula-2, Cocktail tools	Modula-2, C	8712	11384	
ANTLR	C, ANTLR	C	18577	23318	

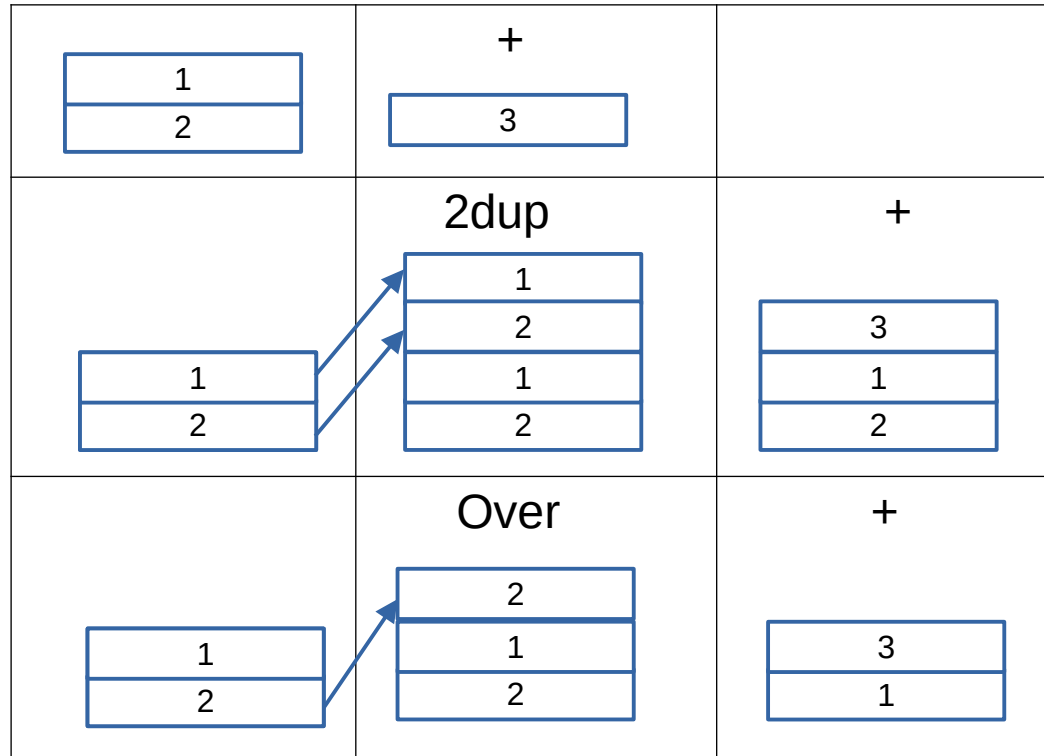
Faster Stack Instructions

Stack based languages such require fewer instructions on stack based computers.

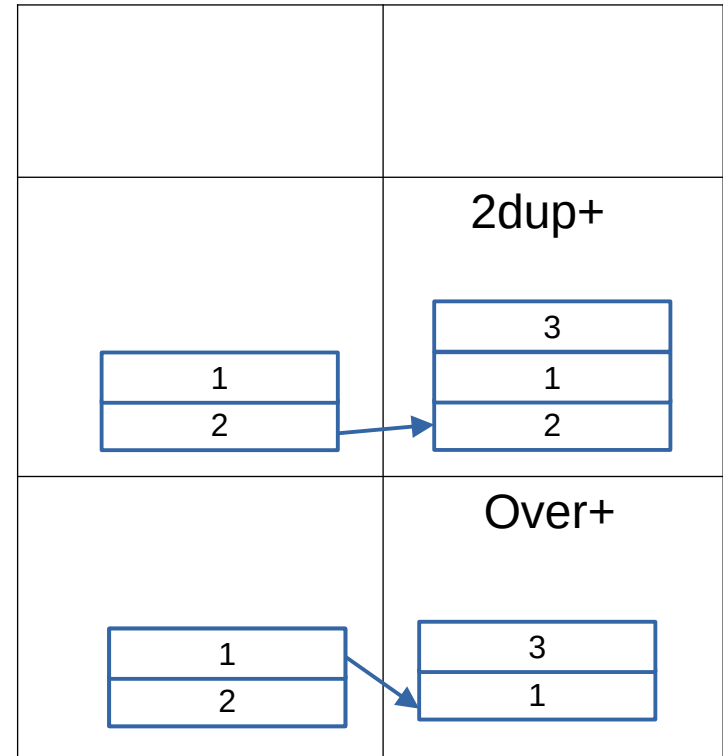
Operation	Stack effect	Forth CPU	Arm C
+	(x1 x2 -- sum)	+ 1 clock Cycle	LDM pt2nd!, {r0} ADD pstop, r0, pstop 2 instructions, 3 cycles
Swap	(x1 x2 – x2 x1)	Swap 1 clock cycle	LDR r0, [PT2ND] STR PSTOP, [PT2ND] MOV PSTOP, r0 3 instructions, 5 clock cycles
Rotate3	(x1 x2 x3 – X3 X1 X2)	Rot3 1 clock cycle	LDM pt2nd!, {r0, r1} STMDB pt2nd1 {r1, PSTOP} MOV PSTOP, r0 3 instructions, 7 clock cycles

J1 Elided Words Vs ANSI Forth

ANSI Forth



J1 FORTH



Stack Machines

Stack Machine Speedup

VLIW J1 Instruction

Compatible instructions

Literal

Call

Jump (not conditional)

Stack Shuffle

Allocate Memory for Locals

Copy TOS to Locals Memory

Add Exception Handler

Single Operand + Dual Operand Instruction

RISC-V

3 * 5 bits for two operands and a results register and 7->17 bits OpCode = 22->32 bit

Fun7

Reg1

Reg2

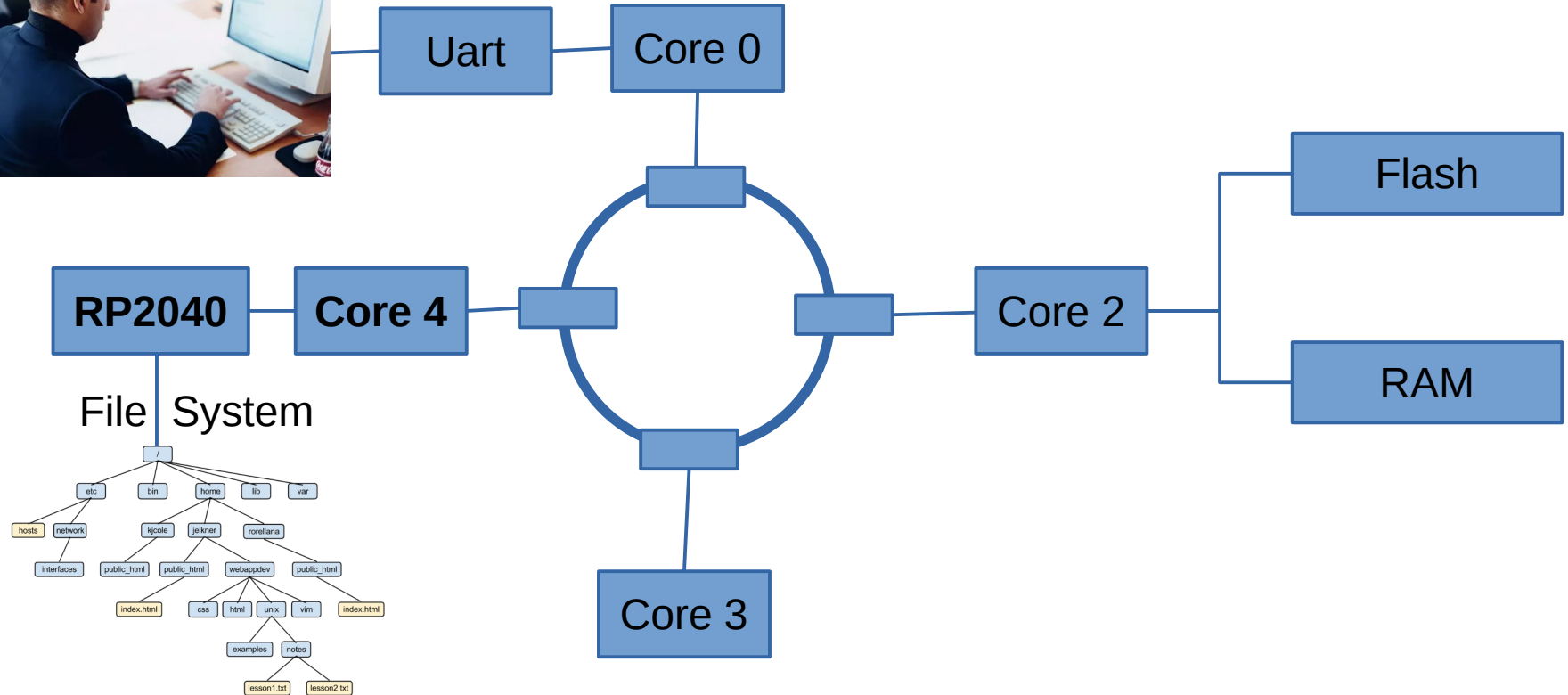
Fun3

Result

OpCode

[Second Generation Stack Computer Architecture](#)

Forth Co-Processors for Circuit Python



Questions

Christopher Lozinski

lozinski@PythonLinks.info

[@PythonLinks@Mastodon.Social](https://mstdn.social/@PythonLinks)

[Forth Discord Server](#)

[Pico-Ice Discord Server](#)

[Hana-1 Discord Channel](#)

On an FPGA, a stack machine is simpler, smaller, faster, requires less memory and less power than a 32 bit 32 register register machine.