

A Third World Embedded Engineer Responds to "RISC-V: They Should Have Known Better"



▮ August 16, 2026 | ▮ RISC-V Article

Dmitry Grinberg published a long piece explaining his distaste for RISC-V, you can read his article here: [RISC-V: They Should Have Known Better - Dmitry.GR](#). It went to the front page of Hacker News and it started a good argument on Lobsters. It is the most substantial criticism the architecture has had in a while and though I switched my entire stack away from STM32 and ARM to RISC-V and did a video on it about a year ago [Goodbye STM32 ARM – Meet the CH32 RISC-V Chips That Replaced It!](#), part of me is infuriated because so much of what he said seems like a biased perspective.

Look, I am not going to defend the ISA committee, RISC-V international denied me membership to their golden tower. On the architecture itself, the compressed store offsets really are strange, Zicsr really should not be a separate thing you have to remember to ask for, I have hit every one of these and I have written a book thats about 80% complete about hitting them on the CH32V003, which is one of the very "RV32E" type chip he mentions.



Maybe I should say where I am writing from, because it changes which parts of this argument look important from my perspective.

I work out of Trinidad and Tobago, a small island nation off the coast of Venezuela. When I want a development board I am not clicking through to next day delivery, I am checking whether the seller ships here at all, what customs will do to it (if I get it at all), and what the total lands at in TT dollars. "Free Shipping" from Digikey, Mouser or any US or European manufacturer doesn't apply to me. I pay anywhere from US \$60 to US \$200 to ship one dollar chips that people everywhere else get free shipping on. In fact a well known PCB company who reached out to me considering sponsorship turned me down solely based on shipping to my location. Have a look here:

Hi Armstrong,

Thank you for sharing. That's very interesting feedback, and I appreciate the detailed explanation.

Before we discuss the sponsorship scope, may I ask where you are currently located? Are you based in the United States or in **Trinidad** and Tobago?

Shipping to **Trinidad** and Tobago can be very expensive. If the project would need to be shipped there, I'm afraid we would not be able to support or collaborate on it due to our limited sponsorship budget.

Looking forward to your reply.

2026-06-03

The students I want to teach are in the same position, and so are the ones in Nigeria and Bangladesh and everywhere else the people in the industry does not think about when it writes its blog posts. From that position, the difference between a ten cent part and a one dollar part is not a rounding error and it is not a detail you get to wave past on the way to the interesting discussion about encodings. It is the difference between a class of thirty students each having their own chip and a class of thirty students watching one demo board if any at all. Instruction set elegance is a thing you can afford to care about once the hardware is already on your desk. Whether the hardware can get to your desk at all comes first. That is why the paragraph most people scrolled past is, to me, the most important one in the article.

Grinberg missed that part that RISC-V creates a space for the other 99% outside of "the world" (which in this space "world" is mainly the US and Europe) and it has nothing to do with architecture.

He Derives the Requirements and Lands on RV32EC

Before the interrupt arithmetic, before the encoding complaints, he does something careful. He asks what a cheap microcontroller core is actually for. His answer is that it sits inside a larger chip prodding registers and configuring hardware blocks, in an "MP3 player, an SD card, a USB stick". The real work is done by custom silicon around it. From that he derives what such a core needs. Low interrupt latency a small die area and good code density, because the code lives in ROM or SRAM and both are expensive per byte. No hardware divider, possibly not even a multiplier, since you are not doing much arithmetic. No privilege separation, because nothing untrusted ever runs there.

Then he writes the line himself:

"But," you might say, "you just described RV32IC (or RV32EC)!"

And earlier, plainly:

I am 100% sure that RISC-V will own the cheap-as-dirt single-use microcontroller space eventually.

So the most credible RISC-V critic of the month sat down, worked out from first principles what a cheap microcontroller core should be, arrived at the instruction set a ten cent chip implements, and stated that this segment is going to be RISC-V's.

He derives the case for the chip and then spends the rest of the article annoyed that the chip exists.

This is almost satirical.

His quarrel is with whether that outcome was earned. That is a real question and I understand why it bothers him. It is not, however, a question that affects anybody deciding what to learn on, because the chip is on the shelf either way.

Where I Actually Disagree, Strongly.

Simply put, the things a high-end CPU needs are diametrically opposed to the things a small cost-saving microcontroller core needs.

The conclusion he draws is that no single ISA can serve both ends, and that RISC-V fans are fooling themselves, in theory the premise is true. The conclusion does not follow, and I can show you why from three parts sitting on my desk as we speak.

CH32V003. This is the cheap "RV32EC" with sixteen registers, no multiplier, no divider, machine mode only, 2KB of SRAM, 16KB of flash, ten cents, it's EXACTLY the core he specified. I shipped two products with these, one is a bin monitor that has a ToF sensor, an LED and an air tag. The other is an agricultural product for a client that opens and closes a door at a certain time. It also makes a good throw away part, as I show case in my whistle switch [Clap Switch Is Dead. Here's the RISC-V Powered Whistle Switch!](#) and which in my view is the BEST part to replace the overpriced, outdated Arduino [Did Arduino Q Ruin Arduino? - Here's how to Switch to RISC-V with the CH32V003.](#)

CH32H417. A dual core MCU that is unmatched in performance to price point and is at the higher end of the MCU line of things. It has a QingKe V5F at 400 MHz alongside a V3F at 144 MHz, 896KB of SRAM, 960KB of flash. USB 3.2 Gen1 with an integrated 5 Gbps transceiver, 100M Ethernet MAC and PHY, a SerDes isolated transceiver, a 500 MB/s high speed interface, SDMMC, a camera interface, a display controller, a graphics accelerator etc etc. I got a web browser running on this thing [I Built a Web Browser on a RISC-V Microcontroller \(No Linux\)](#) Quantum entropy based GAN cat generation [Schrödinger's De/Motivational Quantum Cat: GAN Image Generation on CH32 RISC-V Microcontroller](#) and real-time facial recognition [Real Time Facial Recognition on The Edge With CH32H417 RISC-V MCU](#) in under 150KB of ram. I got a host of other projects running but those are just some I got time to record and put up.

Baochip. A VexRISC-V with an MMU built around a stack thats open from silicon to os [Baochip-1x: A Mostly-Open, 22nm SoC for High Assurance Applications « bunnies blog](#), that runs Xous [betrusted-io/xous-core: The Xous microkernel](#) designed by legendary hardware hacker "bunnie" Huang , a Rust microkernel with real process isolation. Privilege separation, the exact



thing he says the cheap end does not need and therefore does not get. In addition to Xous it also supports operating systems like SEL4 [ytQsob/bao1x](#)

[seL4: seL4_port to baochip-1x](#) and Linux [pkoscik/baochip-linux: An attempt to boot mainline Linux on a stock Dabao board](#). I wrote the bare metal C SDK for the chip [ArmstrongSubero/dabao-sdk: Bare metal C SDK for the Baochip-1x RISC-V SoC](#) and it was of course the chip inside the badge of DEFCON 34 [The New Defcon Badges Pack a Unique Open Source Chip That Doubles as a Security Key](#) | [WIRED](#) this year.

I can also point to the NES emulator I wrote for the \$1 ESP32C3 RISC-V based chip [NES Emulator on \\$1 ESP32-C3 RISC-V Microcontroller](#), or experimenting with Linux on the Orange Pi RV2 [OrangePi RV2 5 Minute Unboxing and Setup](#) | [RISC-V Ubuntu Linux](#) that takes 5 minutes to setup and has been running since the day I boot it up.

Point is I could go on and on about how diverse and accessible currently shipping RISC-V parts are, but then we'll be straying too much from the topic at hand.

I linked all those to say this, that all these parts all have the same base instruction set and I gained expertise in all in under a year and under US \$100 across the entire stack, from disposable silicon to PC level, of course minus data center compute.

For under US \$100 including shipping I was able to explore an entire vertical stack using one architecture. Due to the AI race the OrangePi RV2 has now gone up in price but at release it cost \$30 and shipped free. For about 7 dollars I got 50 CH32V003s with a debugger, the CH32H417 board is \$20 on analog lamb and uses the same cheap (and official) debugger for the CH32V003 and the Baochip Dabao board (which I wrote a book about by the way check it out here ([The Dabao Book - Payhip](#)) was \$9.50 on crowd supply when I bought it, two with shipping from crowd supply cost me \$35, under \$100 in total. A debugger for an ARM part alone a Segger J-Link costs about \$600, though I guess for that \$100, and add another \$100 to ship, so about \$200 I could get an EDU edition J-link and no chips or boards. Yaay.

Back to RISC-V, across all these parts, the base set is the same. So that means the same register model, same calling convention, same toolchain. Yes the extensions differ, but the thing is what I learned writing assembly on the ten cent CH32V003 part did not stop being true on any of the others. A dual core MCU, an SBC running Linux or an advanced custom security chip



running a real operating system. My skills were transferrable to the point that in each case within a few hours I had toolchains setup, could focus on

my applications and when debugging I felt at home. All I need to work with them is the ISA manual and a C compiler.

Now price the same journey on the other side, forget x86-64 and that duopoly, patent minefield, with multi-thousand dollar debug probes; we'll take a look at ARM.

The equivalent to the CH32V003 is the Cortex-M0 is ARMv6-M so something like an STM32F030, step it up we have a Cortex-M7 which is ARMv7-M, to get an MMU in a part for Linux or SEL4 and Xous, you're looking at an application processor like the ARMv8-A. These are different Arm profiles with significantly different privilege, exception, and system models, so moving up the stack involves substantially more relearning than simply enabling another RISC-V extension. Trust me I've used them all.

And at the top of that range the gap is not even about learning curves. There is no Cortex-M microcontroller with an integrated USB 3.0 SuperSpeed PHY. The nearest dual core Arm part is an STM32H747, which is a fine chip and does not have one. If you need USB 3.0 you leave the microcontroller class entirely: an i.MX 8 or an RK3xxx, which means Cortex-A. You want an MMU, Linux, DDR, a PMIC, and a board you are not laying out in two layers. Or you keep the M7 and add an external bridge chip.

The H417 evaluation board is around twenty dollars. The H747 in TFBGA240 carries a twenty week manufacturer lead time, chip only, costs about the same, before you have anything to plug in, and Mouser asks for ID before you can order, Digikey has also been known to deny people parts depending on where they are and their name as Hussein Ali, well known Youtuber from NorthridgeFix describes [Starlink Repair - Digi-key refused my order.](#). Oh and it's about US \$60-100+ to ship to my location. I can pick up H417s on the official WCH store on Aliexpress with free shipping and no verification hullabaloo. We haven't even started talking about the Cortex-A parts that have MMUs or their debugging tools and ecosystem fragmentation.

The Boundary Is Not Technical

Here is the part that undercuts his framing most directly, and it has nothing to do with encodings. He treats the gap between a small core and a large



one as an architectural fact, something that falls out of opposed requirements. On ARM chips it is not an architectural fact. It is a PRODUCT

boundary, and it is enforced by licensing. Has anyone tried adding an MMU to a Cortex-M? The physical tradeoffs are real, the difference is that with RISC-V, the ISA owner does not decide for you where that boundary must be drawn. If you want virtual memory on ARM you license a Cortex-A instead, which is a different core family, a different profile, a different negotiation, and a different royalty. There is no incremental path. there is a wall, with a sales team on the other side of it.

Compare what happened with Baochip. The RISC-V privileged specification defines supervisor mode and Sv32 paging as optional things an implementation may provide. VexRISC-V is an open core, somebody added an MMU to it. bunnie built a chip around it and runs a microkernel with real process isolation on it that me in Trinidad a country who's name does not even come up in ISA circles can experiment with at low cost and teach to other people in the region.

That's what freedom looks like.

Nobody asked permission, nobody signed anything, nobody pays a royalty per unit shipped and anybody can learn down to the RTL the silicon is built on. So when Grinberg in his article lists privilege separation among the things the cheap end does not need and therefore does not get, it is describing a property of ARM's product segmentation and attributing it to instruction set design. On RISC-V it is a checkbox in the privileged spec, you leave it off in a ten cent part because it costs area you do not want to spend, and you turn it on when you do, and the instruction set underneath is the same either way.

That is the real difference between the two ecosystems, and it is why "one ISA cannot serve both ends" reads differently depending on which side you are standing on. On one side the ends are separated by physics and cost, on the other they are separated by physics, cost, and a contract.

The Thing He Calls Fragmentation

Before I close I want to address his stance on fragmentation. He is not wrong that the extension mechanism fragments the standard. Zcb splitting off from C is annoying and Zicsr not being implied by the base is annoying. Vendors adding proprietary interrupt hardware does fragment things further, I learned

But that mechanism is the answer to his own opening question. The reason one instruction set can sit in a ten cent part with sixteen registers and also in a chip running a protected multi-process operating system is precisely that the small part is not carrying the large part's baggage. There is no compromise core in the middle serving both badly, which is what "diametrically opposed requirements" would normally force. Fragmentation and scalability are the same property, you do not get one without the other and whether the tradeoff was worth it is a fair argument and I do not think it has an obvious answer.

What I do think is that he is right about the important part, and right in a way that favours the thing he is criticising. RISC-V is not going to take the cheap microcontroller space because its encoding is elegant. It is going to take it because the part costs ten cents, and because the ladder above it is the same instruction set all the way up. It is going there because an embedded engineer in a 3rd world country can shine a cheap LED and see the transistors in the silicon, [Infra-Red, In Situ \(IRIS\) Inspection of Silicon « bunnies blog](#) and get 50 chips with a debugger and free development tools for the price of a cup of coffee and shipped free. It also means that world class engineers can design MMUs onto chips that the gate keepers will never give a license for.

He writes that this will happen "not due to its ISA design, but despite it," and he means it as a mild indictment. Read it from here and it is not one. Winning on price and availability is not a lesser way to win. It decides who is in the room. An architecture that arrives in my country at ten cents a part, with an open toolchain and no license to negotiate, puts embedded systems within reach of people who were previously going to watch somebody else's demo board and consume their products without ever being able to match what they have access to. That's the power of freedom, openness and is democracy in its truest sense.

That is a better reason than elegance. and I want to tell Mr Grinberg, that the word privilege he tosses around in his article also extends beyond the ISA depending on where you are in the world.

Nuff said.



Armstrong Subero is an embedded systems engineer and published author with Apress/ Springer. He builds the [Rover](#) RISC-V education platform from Trinidad and Tobago.

Share this post:

[Facebook](#)

[Twitter](#)

[Pinterest](#)

[Copy Link](#)

Posted on: August 16, 2026