Avraham “Abe” Bernstein | CV-Lite

Email: *Avraham DOT Bernstein PLUS cv AT gmail DOT com*
Tel/Whatsapp: *+972.54.641-0955*
City: *Jerusalem 9727433 ISRAEL*
Time Zone: *UTC +02:00/+03:00 (winter/summer)*
Shabbat Observant: *Not accessible electronically nor engaging in any business activities from Fri. evenings (Jm. time) beginning 1 hour before sunset until Sat. night 1 hour after sunset, nor on Jewish holidays*
WWW: *https://www.avrahambernstein.com*
Linkedin: *https://www.linkedin.com/in/avrahambernstein/*
Last Update: *2025-12-28*

1. Summary

Senior software architect with over 40+ years of **INNOVATION** in:

- algorithm design
- compiler construction
- source code refactoring
- obfuscation tools
- multiple patents and inventions

Worked at an **expert level** in many industries including:

- cybersecurity anti-reverse engineering
- automotive
- internet TV
- accessibility
- bioinformatics

Expert generalist and autodidact polymath who thrives on technically challenging projects. Adept at design of:

- domain-specific languages
- building MVPs and prototypes for CTO teams

I have solved many problems that initially had ambiguous problem specifications. I was able to refine the problem to a commercially useful and testable solution. I prefer to work with and to mentor small teams. I am searching for position as a contractor or consultant.

Core Skills & Tools

- **Languages:** C, Python, Jinja2, Pyexpander, Bash & Posix CLI Commands, HTML, XML, Markdown, WASM
- **Technologies:** srcML (acquired commercial license), Beautiful Soup, Linux, ELF, Misra C, GCC, Clang, Zydis, Pandoc, Obsidian, YAML & PKL
- **Domains:** Compiler Design, Domain Specific Languages (DSL), Code Refactoring and Obfuscation (= anti-reverse engineering), Cybersecurity, Reverse Engineering, Embedded Systems, Accessibility, Automotive Software, Factory Automation, Bioinformatics, Network Protocols

2. Inventions (Reverse Chronological Order)

1. Showed how srcML combined with Python Beautiful Soup could refactor "C" source code for automotive software updates, and for cybersecurity obfuscation.

2. Invented Linux cybersecurity techniques, especially to minimize the attack surface of `.so` (DSO) files that supply no formal exports, efficiently shroud system calls and strings and constants and randomize the stack, and thus are extremely difficult to reverse engineer. See short technical description.

3. Patented FLASH techniques for implementing S/W updates on small FLASH devices, and small RAM.

4. Invented a technique for *dynamically* loading a large static FLASH database into an embedded system that could be larger than RAM that does not violate any Misra C restrictions, with the help of preprocessing tools such as *Jinja2* or *Pyexpander*.

5. Invented a simple technique to obfuscate photographs using GIMP filters while remaining easily recognizable by a young child, but not recognizable by photographic database software.

6. Invented highly accurate fuzzy logic classifier used to discover rooted *Android* devices.

7. Showed how trivial Virtual Machine (VM) attacks could disrupt *Digital Rights Management (DRM)* protection, and how the QEMU VM could disrupt the cryptographic nonce mechanism which allowed subscriber IDs to be shared by a confederacy of pirates.

8. Invented a generic font modification technique in order to make them understandable by dyslexics.

9. Invented a technique that enabled the blind to "see" and navigate digital maps and to explore mathematical functions on standard PCs and smartphones with a sound card and the addition of a consumer grade graphics tablet via the use of custom S/W that runs on a standard web browser with SVG support.

10. Invented *bioinformatic* PCR algorithms that (1) can accurately detect the "Ct" of an *assay* with high levels of *inhibition* via the use of an AI threshold algorithm instead of the classic functional analysis technique, and (2) can trivially clean the *systematic noise* from an *assay*, especially useful when the *assay* contains significant *inhibition*. Note that the project took only 2 months, even though I had zero formal background in microbiology and genetics. I received intensive mentoring from domain experts. I saved the client's project from bankruptcy.

11. Invented a network protocol for hybrid cable modems with a dialup upstream and RF downstream, where the "edge router" controlled access to both the dialup and RF networks, which enabled the edge router's ARP table to be dynamically modified when a modem logged on and off.

12. Designed a super efficient cable modem network laboratory which multiplied by a factor of 24 how many modems could be attached to a single PC via multiplexing with a dynamically programmable layer-2 switch.

13. Designed a GCC port for Forth-like CPUs that effectively allowed an unlimited number of registers.

14. Designed Domain Specific Languages (DSL) for Quality Assurance (QA) projects that allowed them to be tested with scripts. Examples included satellite communications, client movie players, and milspec testing laboratories. The scripts exploited flaws in the satellite software design. The satellite scripts were used to implement sanity tests on the developer's desktop prior to code check-in which significantly reduced QA testing. The mil-spec scripts replaced 3+ meters of test documentation, and allowed the system engineer to write simple *ad hoc* tests.

15. Invented a DSL for a VLSI toolchain that clock-accurately emulated the CPU in "C" including its instruction pipeline. The "C" emulator ran 100 times faster than the electrical (e.g. VERILOG) simulator. Therefore the DSL enabled instruction set sanity tests to be run immediately every time the architecture changed. The DSL solved one of the most painful problems in designing the assembler manually, namely detection of when the pipeline was broken which required programmer insertion of NOP instructions. For many months the VLSI architects changed the pipeline restrictions on a regular basis. Therefore DSL included a restriction language. It allowed the assembler to be built within 2 hours multiple times a week. The DSL was expanded to include a disassembler. Eventually the toolchain formed the backbone of a GUI debugger. The debugger incorporated programmable simulation of I/O ports. The debugger enabled the building of accurate applications many months before the physical CPU was produced (i.e. "taped out").

16. Invented a DSL which described an automated sintered metal blade production factory in complete detail. At that time commercial quality object oriented (OO) languages did not yet exist. But the factory contained about 50 object classes, e.g. workstations, stands, pallets, automatic guided vehicles, stacks, conveyors, cranes, etc. Interrupt routines were required for tool sharpening maintenance, oiling of parts that stayed too long in storage, etc. The DSL included a scheduling language. We designed an object oriented language (in Pascal) which completely described the factory. The factory was operational within a year. The factory

engineer could make gross changes to the configuration file, while clerks had individual GUI screens to make spot changes. Originally I had zero knowledge of factory design, but after about 2 month of mentoring from an industrial engineer, we began to design the software.

17. Invented an application for quadriplegics that enabled them to access a PC with a CRT screen via a telescopically extended light pen (800 mm vs 5 mm distance) and via a sip-and-puff accessibility switch. The subject wears a head band to which the pen was attached. Designed a virtual screen interface which only popped up an individual virtual key when the pen hovered over it, leaving 95% of the screen still visible. The first user was a quadriplegic polio victim who was able to type 30 characters per minute, and became a financially independent book editor.

18. Invented many small CS algorithms over the years: hardened (minimized collisions) Adler-32 and new Adler-64 checksum; cryptographic quality key wrapper implemented in registers; *invertible* cross Hamming Weight transformation; cryptographic quality FFT *uniformly distributed* Hamming Weight-like primitive; binary expandable hash table (size `2^N`) that can grow without rehashing

3. Work Experience

2025: Independent: Founder & Principal Engineer Compiler & Obfuscation Tools Development. Download brochure.

2022-25: Aurora Labs, Tel Aviv: Senior Software Architect in CTO office for Digital Automotive Industry

2021: Morphisec, Beer Sheva: Senior Software Architect: Anti-Reverse Engineering Modifications to *Linux x64* Libc Kernel implemented with the help of the Zydis x64 disassembler

2021: Qedit, Tel Aviv: Consultant: WASM Cybersecurity for Financial Industry

2017-20: *Argus Cyber Security* (now PlaxidityX), Tel Aviv: Senior Researcher for Digital Automotive Industry

2013-17: Viaccess-Orca (subsidiary of Orange FR), Ra'anana: Cybersecurity Obfuscation Manager for Internet TV industry

2017 part-time: Independent: Dyslexic Accessibility

2015 part-time: Canary Mission, Jerusalem: Consultant: Internet Security "Hygiene"

2013 part-time: *NVT*, US (defunct): CTO: Agritech Startup for Cassava Production in Nigerian Jungle

2012: *Telequest*, Jerusalem: VP R&D: Automated Vehicle Navigation To Find Optimal Routes in City Traffic

2011: Syntezza Bioscience, Jerusalem: Consultant: Bioinformatic algorithms

2005-10: *NDS* (now Synamedia), Jerusalem: Senior Researcher: Internet TV Industry

2002-03: *Virtouch*, Jerusalem (defunct): VP R&D: Blind Accessibility

1999-2004: *Vyyo*, Jerusalem (defunct): S/W Group Leader: RF wireless cable modem industry

1998: *Fourfold*, Jerusalem (defunct): Software architect: modified GCC compiler for massively parallel CPU for a Forth-like CPU with unlimited registers

1991-97: *Pitkha*, Jerusalem (defunct): CEO: Domain Specific Language (DSL) Architect

1990-91: Iscar-Matkash, Tefen: Software Architect: Factory Automation

1988-89: *Cubital* (subsidiary of Scitex), Herzliya:

- S/W R&D: Early 3D printing (*stereo-lithography*)
- Inventor: Quadriplegic Accessiblity for PCs

...1983-84: Elta IAI, Ashdod: junior embedded programmer: Lavi fighter plane

...1977: Ontario Energy Board (OEB), Toronto

- While attending the MBA program at the Univ. of Toronto (see below), I was an intervenor in the *ECAP77* hearings on marginal cost pricing for electricity at Ontario Hydro. I was the first public interest intervenor

in the history of the OEB to be awared costs. I published an oped about the hearings in Canada's then paper of record The Globe and Mail.

4. Education

1979: *York University*, Toronto Canada: **MA Economics**, minor in Applied Mathematics

1973-78: *University of Toronto*, Canada: **BA** Undergraduate School of Arts & Sciences

- I did graduate studies at the Rotman Graduate School of Management (MBA program) and Graduate School of Engineering where credits were applied to my *MA Economics* above. After the Ontario Energy Board (OEB) hearings immediately above in 1977, I switched focus to engineering. I extraordinarily passed my economic compulsory examinations before I even started the York program, so the school allowed to me to take graduate level courses anywhere I chose.

5. Teaching & Mentorship

Part-time instructor and mentor in various technical domains. Strong advocate of pairing with domain experts to accelerate onboarding and innovation.