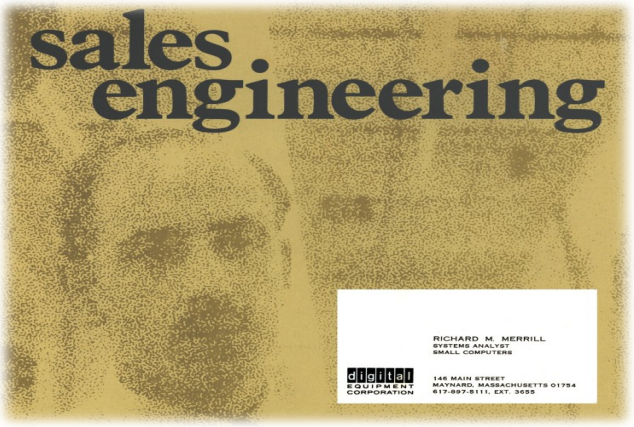
Rick Merrill – FOCAL Notes and Background



as told to: Bruce Ray
Wild Hare Computer Systems, Inc.
2024-Apr-25

FOCAL's Creator - Richard M. Merrill

The Instrumentation Lab (now Draper) hired me as an undergraduate in 1960 to work in the machine shop part time. Then in graduate school with a security clearance the lab hired me as a Research Assistant to do my thesis on a computer strain analysis of the PIGA (Pendulous Integrating Gyroscopic Accelerometer). I got time on Project MAC and on the computers at Apollo Lab.

I decided to take a Master's Degree from MIT and work for a new computer consulting group called Adams Associates. We were to build an incremental compiler for the PDP-6. I worked on the "Eval" portion which was helpful when building FOCAL. Then Adams had me write a fancy debugger for the TX-2 at Lincoln Labs where I worked with Larry Roberts and Ivan Sutherland.

I learned about DEC through an article on the development of DEC-TAPE and interviewed and was hired in Maynard in June of 1966 where I received a low, 4-digit badge number. I was actually in the "Marketing" department where I helped customers configure PDP-8 and LINC-8 systems for various jobs including data acquisition systems for anti-ballistic missiles in the Kwajalein islands.

A Navy Doctor came to the Maynard mill dressed in Navy Whites. His goal was to make a computer model of "The Bends" using two six-foot bays of a LINC-8. Today his product is a wrist-wearable unit that helps Navy Divers decompress 6x faster and more safely.

The Birth of FOCAL

I took demo software to the show Federation of American Societies for Experimental Biology (FASEB). I showed electroencephalograph (EEG) data on a LINC-8 computer. If you are a biologist, this is great stuff. We could zoom in on the brain waves while the output would display and the tapes spin -- It was a great demo!

A doctor in white scrubs dismissed the demo with a wave of his hand sat down and said, "Let's see your machine add two and two." I thought: It's not programmed for that. Not a good answer! I could load a different operating system; bring in the desk calculator program; remember the cryptic codes that made it go -- and it still was going to give us 3.999! So I toggled in two assembly codes and single stepped through 2+2=>4. Our guest said, reading the binary display, "Two to the zero, one, two - that's four!". "Well," he said, getting up, "That's better than they could do last year!" I learned from this that it is more important to do the customer's job than a demo.

Back in Maynard I took an editor, a floating point package and the specs for the JOHNNIAC Open Shop System (JOSS) and ideas from the Massachusetts General Hospital Utility Multi-Programming System (MUMPS) and wrote an interpreter that would do the user's job on the spot and interactively.

I remember thinking that I wanted a single letter for each command not only for the simple branch table but also for debugging. I did not want to have to remember an octal code for each command but to be able to see in the code the letter itself that DDT could display. I eschewed pre-compiling as a waste of time.

I gathered the code for the editor and the floating point package (probably developed by Henry Burkhart) and got them to assemble together eliminating duplicate symbols and other bookkeeping issues. This was the top-down stage. Then I realized that my branch tables used full word addresses and would therefor flow across the page ends, wasting no space because of local page sizes. So I began coding from the bottom-up stage with SORTJ, PUSHJ, POPJ, ETC.

I acquired the "JOSS Apologetica" which was really helpful in distinguishing the commands so that there was no overlap. Then I found that the semicolon could fore-shorten commands in generally useful ways. Adding buffered interrupts came later and made a huge difference.

This program became known as Formula Calculator (FOCAL). I had initially thought to call it "FORGE" but found prior use in obscure nooks and crannies. So we held a naming committee that decided to call it FOCAL and every one was happy with that. Ken Olsen then told legal department to make sure it was internationally trademarked.

FOCAL was the first interrupt driven, fully interactive software for minicomputers. It was the first to self-start. It was the first to have user configuration (the transcendental functions and extended memory were optional). It was the first to use hash code storage. It was re-entrant. It was the first "mini-computer" to have a dedicated multi-user system of four users at the same time.

The multi-user system flowed out of the full-interrupt driven code. I realized I did not care about "time sharing" but about "computer I/O sharing". So ANY interrupt service would cause the next user to become activated. I worried needlessly that a compute-bound code would tie up other users BUT there was never a single complaint about that.

I presented how FOCAL worked at Digital Users (DECUS) meeting and received a standing ovation. People asked for my autograph! It was my "15 minutes of fame." Little did I realize that my "moment" would last longer. Thousands of school kids, teachers, scientists, and engineers would cut their teeth on FOCAL.

Versions of FOCAL were eventually created for PDP-5s, PDP-7s, PDP-9s, PDP-10s, PDP-11s, PDP-12s, PDP-15s, and helped my company sell $10M of computers per year at the peak of mini-computer sales.

Post FOCAL

After PDP-8 FOCAL I consulted on the design of the PDP-11, including codifying the assembly language instructions, register references, and indirect references. Porting FOCAL to the PDP-11 was a follow-on project, then working on the VT-100 terminal and a PDP-11 Unibus to IBM 360 interface.

FOCAL Anecdotes

A Sneaky Bug

Only one bug in FOCAL was ever reported from the field through the chain of command. Individuals reported other bugs directly to me. A student at Sudbury high school had written a program in FOCAL to calculate the digits of PI. When he checked his first printout against the documentation he found two digits transposed as if the computer were dyslexic. He was understandably miffed. However, no one could repeat that bug!

We were running four users on a PDP-8 with a full 16K with teletypes at 110 baud when we got a Tektronix display that ran 2400 baud. Then I saw the problem happening at random! Suspecting what the issue was I asked Dave Plumber to set me up with his PDP8 simulator on the PDP6. Then I set the interrupt frequency to match the word fetch frequency of the PDP8 - in other words interrupt at each instruction: bingo, A dyslectic computer. We needed to move the IOF instruction to the start of the output routine. Next week we would be running 4-users on the brand new PDP-8/I never tested with 4 users in Maynard to a big show in Anaheim, CA.

FOCAL Saves The Day!

Once in Anaheim I loaded the program and started running the PDP-8I with 1,2,3, and 4 users when suddenly the output from each user terminals all started showing on TTY #1 which was of course running in the lowest memory bank. Within a few minutes Louis Klotz and I figured out that the interrupt was simply clearing the high memory bits, EVEN when the instruction was IOF! With prints all over the floor Louis and his team made a green wire fix that night and the next day we were up and running with a waiting line of people wanting to try four users on a PDP-8/I.

Suddenly the lights went out, the computer fans sighhhhed and all users the cavernous show room went AWWWW with a few other choice words thrown in. The loudspeakers announced that they apologized for the outage and power would be restored in fifteen minutes exactly. The wait was to let the big lights cool enough to restart!

Sure enough the lights came on, and the fans went wwwoooOOO, BUT the ONLY CHEERING came from the Digital booth because each and every user was back to exactly where they were at power out! No reboot required! It was a Big score for core that day!

Alan Kotok Bug

To run four users FOCAL was loaded into memory bank zero; then put unique code into banks 1,2,3. But the binary loader wouldn't let me reset the PC to bank 0 to patch the binary loader to self-start! So I found that Alan Kotok had written the binary loader; it needed one more instruction in the 256 word loader! I bet Alan $1 that he couldn't save one more instruction. He not only took the bet and of course fixed the bug, but Alan went further and issued an ECO to make sure the NEW binary loader was distributed thereafter

Smart Keyboard

Digital Equipment Corporation (DEC) hired a brilliant Swiss professor from Ecole Polytechnic Federal de Lucerne (EPFL) and DEC assigned me to help Jean-Daniel Nicoud with whatever he needed. I learned how to wire breadboard circuits and wrote a bootstrap loader if software were ever to be developed. Many people such as Stan Olsen, Ken Olsen, and others had ideas about the configuration. Ken wanted a handle such as the ones he knew would be on the back side of any small TV. Stan wanted a keyboard that could attach to the screen and be carried with one hand the way old telephones could mate with the desk set. That was a feature that was in all the old desktop telephones. Ken had the model shop put together a TV-case with sides which folded out like a flower for assembly, service, or repair. I put a working video processor designed by Professor Nicoud into first one then another of these and called it 'smart keyboard" or Smaky.

Then for Thanksgiving Digital gave out a turkey to each employee. As I walked out through the building 5 parking lot carrying my 15 lb. turkey home to my wife and twins I realized that 15 lb. was too heavy for a "portable computer." One manager wanted to sell computers to do inventories for grocery stores and others. He held a big meeting then asked me what I "wanted." I was stunned and unprepared to be the next "Steve Jobs".

I had the Smaky design implemented on a printed circuit board. I never got to "populate" the board but sent a couple of boards to EPFL. Professor Nicoud had designed the layout so that each module was on a separate section. I showed it to a Dick Clayton who asked "how much did that cost?" I had no idea. A couple of other DEC employees wanted to take Smaky out of DEC and they courted me to help do that by moving to the edge of Bolton where they had found a house that "we" could use to develop the product. They did not stay with DEC long.

Now Smaky was not based on a PDP (DEC) computer. I did not have a decent assembly language tool for it much less a compiler. I could have written my language, FOCAL, for the 8080 as I had for the PDP-8 and again for the PDP-11. I feared that no one else would support such a processor nor the assembly language, chip testing and so on. And then I was 'fired'. Of course in DEC that meant you could go find another job in the company. My next product was the GIGI, which incorporated a computer in a keyboard and had advance graphics. The Education Sales group had sponsored the

development of the GIGI. I wrote an editor for it that dealt with different character sizes, graphics with wrap-around text, and a plethora of other features that enabled things like 3D chemistry graphics.

FOCAL Does Cosmetics?

FOCAL on a PDP-8 as seen in Scientific American was used by a cosmetic company whose name I redact from this remembrance. The task was to control a simple robot which ever so gently scooped lanolin from one vat into another mixing unit just as the "old master" used to do. High pressure injection of liquids simply would NOT do!

One day a hinge pin in the robot came loose and the lanolin ladle dumped the product directly on top of the PDP-8 computer! Oil penetrated all the computer modules! The Digital Equipment Corporation (DEC) Field Service man unplugged the computer and carefully removed each individual module, dipped same in trichloroethylene (which was legal in those days), blow dried the module with a hair dryer, and replaced it into the backplane.

He came to the core memory module of tiny magnets interwoven with tiny wires and now full of sheep oil (lanolin). I can just imagine him shake his head then replace the memory unit with a brand new one. He ran the diagnostics and pronounced the computer as "All is well again." He explained, "Now you have to reload your software."

Puzzled, the cosmetic company engineer replied, "We don't have to. The program is always in there." So the DEC field service had to explain that the memory was replaced and no longer had their program.

They had a paper tape copy of their FOCAL language application, but nowhere could they lay a hand on a copy of the FOCAL-69 operating software. Digital had to go back to their "Bomb Vault" to find a copy for them.

The PDP-8 and FOCAL had been operating flawlessly for 12 years!

FOCAL is a Blast

An aluminum refinery is controlled by FOCAL running on a PDP-8 computer and FOCAL. Bauxite ore is dumped into a vat of molten aluminum and it is heated by running an electric current from a large, 6 inch diameter solid carbon rod - called "THE ARC". The electric current reading is sent to the PDP-8 running FOCAL to tell the computer what the temperature is. The FOCAL program then runs a lead-screw motor that moves the rod up or down to maintain the perfect vat temperature 24 x 7.

But the arc is powerful and wants to pull the rod down. What if the arc shorts out to the vat? There is no circuit breaker from here to Niagara Falls! What if the motor fails? What if there's a bug in FOCAL? What if the computer dies?! The backup to those parts is Klaxon alarm and a stick of dynamite that blows the rod up and out of the vat!

As far as I know, the dynamite has never been used.

Rick Merrill FOCAL Artifacts

Computer History Museum

Rick Merrill has donated original FOCAL-related items to the Computer History Museum, Mountain View, California, including:

1. FOCAL language source line printer listing (PDP-8 computer PAL assembler), blue computer binder
2. DEC-08-AJBA-DL, FOCAL Technical Specifications, 1968, paper manual
3. DECUS FOCAL-17, "FOCAL: How To Write New Subroutines and Use Internal Routines", Doug Wrege, 1970, paper manual
4. PS/8 FOCAL, 1971, OMSI Development Group, paper manual
5. DEC-12-AJAA-LA, FOCAL-12 Listing, 1971, paper document
6. "focal point" 4-page newsletter, newsprint size
7. Westwood Public Schools, Project Local, TSS-8 FOCAL 'thank you' letter to Rick Merrill
8. 4-page Digital sales brochure "Sales Engineering", 1968, Rick Merrill's picture on cover
9. Richard M. Merrill business card, 1966, Digital Equipment Corporation
10. DEC-08-XJFA-D, FOCAL Demonstration Programs, 1970, paper manual (only copy known to exist)
11. "focal, a new conversational language", orange-cover brochure, 36-pages
12. "FOCAL Party Line", paper brochure, 6 pages, 1969
13. "FOCAL Lets You Tame a Computer", 4 page brochure, 1969
14. pdp8/i-pdp8/L FOCAL, language summary reference card, 8-pages, 1969
15. pdp9-pdp9/L FOCAL, language summary reference card, trifold, 1969
16. pdp15 FOCAL, language summary reference card, trifold, 1970
17. samples of FOCAL user complements received by Rick Merrill, personal memorabilia;
18. "Rick Merrill – FOCAL Notes_and_Background", (this document), 2024

www.ComputerHistory.org

Computer History Museum Software Preservation Project

Bruce Ray, Wild Hare Computer Systems, Inc., converted Rick Merrill's original FOCAL items to digital PDF-file format for inclusion in a new FOCAL section of the Computer History Museum's Software Preservation Project website.

www.SoftwarePreservation.org

Wild Hare Computer Systems Archives

The DEC [Digital Equipment Corporation] section of Wild Hare Computer Systems' computer history preservation web site contains digital copies of Rick Merrill's FOCAL items plus additional artifacts reflecting FOCAL's significant part of computer history. These include subsequent FOCAL derivatives running on different types of computers and modern open-source FOCAL implementations.

www.NovasAreForever.org

Part 1 of Rick Merrill's FOCAL donations to the Computer History Museum

Part 2 of Rick Merrill's FOCAL donations to the Computer History Museum