Entrepreneurial Profile

Chuck Moore And Forth

by Karl Lunt

Software consultant, developer of the Forth programming language, chip designer, and co-founder of Forth, Inc.; Chuck Moore has covered a lot of ground in the last 25 years.

Chuck spent the early '60s as a free-lance programmer, what we now call a consultant. He developed custom business and scientific software for the main-frame computers of the day, moving from site to site and into different fields as the market demanded.

He found writing business programs lucrative, but he enjoyed the scientific software more. "I did scientific programming, analyzing data, moving widgets," he recalls. But those main-frames were slow and primitive compared to today's PCs.

"The compile time was just too great. The Columbia University main-frame gave me one compile per day, and it took half an hour to compile my program," he says. "That kind of computer time was so expensive."

And when Chuck moved to the mini-computers, the problems got worse. "Most computers of those days had 8K of memory. The system fit into 4K, and my program had to fit in 4K. I had to use paper tape as mass storage, and the mechanics of feeding tape became tedious."

Adding to that, Chuck had to work on a variety of different machines, with many types of peripherals and mass storage. Operating systems varied, as did editors, assemblers, and compilers. Finally, the difficulties in doing his job well became too great. Chuck started designing a tool to help him write software more effectively.

An early version of what would become Forth served as an interface between Chuck's software and the system running on the host machine. "A lot of those early systems used FORTRAN," he recalls. When he arrived at a new job, "...I would show them this eight-inch long deck of punched cards and explain that these were I/O routines that I needed for my program. FORTRAN in those days had very inefficient I/O routines, so they usually bought it and let me install my system."

Once installed, this early system would do most of what today's Forth run-time kernel does. It could not, however, do sophisticated math operations. Therefore, whenever Chuck's program had to compute an algebraic expression, it would invoke a system FORTRAN subroutine to do the work.

But this was only a partial solution. To eliminate the time-consuming loading operations, he needed to have his new tool memory-resident at all times. He came up with a short list of other features, then started writing what would be the prototype Forth. "That first system had an interpreter, compiler, editor, and assembler, and it all ran in 4K on an IBM 1130," he remembers.

(In Leo Brodie's marvelous Forth primer, *Starting Forth*, Chuck explains why he chose the name. He saw his creation as a fourth-generation language and wanted to call it FOURTH, but the IBM 1130 only allowed five-character file names. So he called it FORTH instead, "a nicer play on words anyway.")

He first used Forth on a business project for a company called Mohasco in upstate New York. "That is where I learned COBOL and inventory control. I couldn't really use Forth very much on that project, because they wanted compatibility with COBOL."

On his next project, he worked for the National Radio Astronomy Observatory (NRAO) in Virginia. The job involved collecting and analyzing data from their telescope at Kitt Peak Observatory, near Tucson, Arizona. At the time, collected data had to be analyzed off-site, forcing astronomers to endure a considerable delay before they could see results.

"I went out on a limb and said that I could collect data more quickly using my new Forth system," Chuck says. NRAO told him to go ahead, so Chuck started on the first large scientific Forth program. "It only took me about a week to move my Forth system to a new machine," he recalls. Before long, he had Forth running on NRAO's PDP-11.

When finished, his program let the astronomers collect their data and process it on-site, giving them faster access to their results. The program was quite a hit with visiting astronomers. "They would use the system and want to take it home with them," says Chuck. Since they usually had a different computer system than the PDP-11, Chuck would have to convert Forth to run on their system, then send them a magnetic tape of the program.

Even though Forth was popular with the astronomers, it did not set well with management; they wanted the program done in FORTRAN. Eventually the environment became unfriendly, and Chuck decided to leave in 1973.

He had by now reached a point familiar to many Midnight Engineers. "There were several dozen copies of Forth running on about a dozen different types of computers. All of the systems involved astronomy, except one that did astrology." Chuck begin to think about starting his own company to develop custom Forth applications.

He had done the NRAO job from their offices in Charlottesville, Virginia. Support for his software at the Kitt Peak Observatory had been provided by Elizabeth Rather. Chuck and Elizabeth decided to team up and start Forth, Inc. as an applications house. They borrowed $5000 from Elizabeth's

boss at Kitt Peak and headed to Manhattan Beach, California, where Elizabeth's husband had a job at the time.

"We had no contracts and no computers," Chuck recalls. "Computers were not easy to come by in those days. Eventually, we got a Nova mini-computer with a disc cartridge drive instead of paper tape."

The two programmers started by selling application software to astronomy contacts. Chuck would usually go on-site to install the software, and considers his trip to the Royal Greenwich Observatory in England as his most exotic installation.

Business was good enough to provide a living, and eventually they developed Forth applications in other fields. "Forth, Inc. did a heart monitor that you strapped to your waist; it collected data for later analysis. We also did the 200-foot long wing jig for rewinging the C5-B cargo transport plane.

"Forth, Inc. also did the Saudi Arabian King Kalid airport security system. It took a couple of years to write, even in Forth. We got the job after two other companies had failed to produce any results at all."

An important customer in those early days was Art Gravina of Cybek. Art sold business systems running on minicomputers. Chuck promised Art that Forth would give him increased performance over his BASIC software. Art ended up making millions with his Forth-based Cybos system. While most minicomputers could only support three or four users at a time, Art's system handled 32 users on a single Nova machine. "This proved you could make money selling Forth," Chuck claims. "Just don't call it Forth."

Chuck claims that his most enjoyable project of those early years was the software he wrote for a Hollywood movie company. "We did the motion control program for a movie called *Battle Beyond the Stars,* a modern version of *Seven Samurai.*" The camera had to be precisely controlled through 14 axes of motion; the math routines were particularly challenging. The whole project took just two months to complete.

Forth, Inc. had a novel approach to software maintenance. "We didn't have software maintenance contracts," Chuck notes. "We taught their programmers how to maintain the software, then cut the strings." He recalls that most of the programmers they taught ended up enjoying the language, though some had to be dragged "kicking and screaming" to learn Forth.

"After they learned how to use the language, they were delighted with how easy it was to use and what they could do with the language," Chuck says. "The programmers liked Forth because they could do anything they wanted with the language, and their system had much higher performance. Management liked Forth because they got results on-time and on-budget.

"In those early days, I had a theorem that the Forth community would double each year. This is because it usually took someone who found Forth and fell in love with it about a year to convince his best friend, so you could replicate yourself in a year."

Chuck's work at Forth, Inc. usually divided into two parts, the porting of Forth to a new computer system and the writing of Forth applications. He clearly preferred writing applications. "I always saw Forth as a tool; I didn't want to get into

the business of selling Forth," he says. "When I was at Forth, Inc., I insisted that if someone wanted a Forth system, they had to pay me to do the application, not just install Forth."

But after years of writing Forth systems for different computers, Chuck was ready for the next technological step.

Building a Forth Engine

Most Forth systems use software running on the host machine to simulate a mythical computer whose machine language is itself Forth. For example, most Forth systems imitate a Forth computer that has two stacks and two registers. Although these Forths run on many different machines, they all generally simulate the same virtual Forth computer.

Using the same model for each implementation of Forth has a big advantage; you can pattern a new Forth after previously written Forths, reducing the time it takes to port Forth to a new computer system.

However, the virtual Forth machine so ported can never execute its Forth instructions as fast as the host machine can execute its machine-language opcodes. The figure varies, based on the host's instruction set and the programmer's skill, but speed penalties of 10x or more are common.

The only way to run a Forth program at very high speeds was to find a CPU that used Forth as its machine language. Chuck wanted to see what a true Forth CPU (known as a Forth engine) could do. Since no such chip existed, and since no one would build one for him, he decided to design and build his own.

"I left Forth, Inc. in 1983 to develop the Novix chip," he recalls. "This chip was a precursor to the Harris (RTX-2000) chip, and was a silicon Forth engine." Chuck spent about six months designing the chip. Bob Murphy in Scottsdale, Arizona provided technical support in the chip design.

"I had to learn everything about the insides of computer chips. I had taken electronics courses earlier at MIT, mostly analog, but the digital world is different."

Chuck had the chip built as a gate array. He describes testing the first chip: "We plugged the chip into the breadboard circuit and it ran. It actually worked the first time, but we had some problems with the breadboard and it took us a week to realize the chip was running."

An actual silicon Forth engine was running after talking about it for five years. "The 1983 Novix chip ran at 8 MIPS; it was the fastest chip in the world running the fastest language in the world...unbeatable."

He describes the Novix chip as "successful," saying that Novix sold thousands of chips. Novix sold the patents to Harris, who created a successor, the RTX chip. Harris discontinued the RTX chip last year; Chuck says that the Forth engine took "a lot of support that a chip maker can not provide."

Chuck has since developed other, faster Forth engines similar to the Novix chip. His ShBoom chip, developed with the help of a Novix system, will be showcased at a workshop to be held at the ACM SIGForth conference scheduled for late March in Kansas City, MO.

He then used a ShBoom system to design the P20 engine, which he previewed at a presentation in Vancouver, B.C. late last year. He already plans to

use the P20, which will hit 100 MIPS, to design its own successor, the P24.

The Business Side

The early days of Forth, Inc. taught Chuck the differences between working for someone else and working for himself. "It is comfortable to work for someone else, but you work on their terms," he believes. "In my personal view, the things I have done over the last few years I couldn't have done working for someone else."

When asked about advising other Midnight Engineers in starting their own business, Chuck was quick to reply. "Go for it," he says. "You really have nothing to lose and everything to gain." Thinking back about his own startup using a borrowed $5000, he chuckles, "I still can't believe we did that."

Not surprisingly, the high points of his business career all center on writing and using Forth. "The greatest reward was all those applications," he recalls.

While he liked the technical side of his chosen career, the business side was more frustrating. Even today, the greatest obstacle is marketing a language few people know anything about. "The technical part of programming systems and applications is easy compared to selling services," he says. "Marketing has always been a major problem. Getting someone who can find markets and talk people into using Forth is a major challenge."

He and Elizabeth spent considerable time getting advice and guidance as they started their company. "We went to the Small Business Administration; we talked to a business advisor at the University of Southern California." If anything, Chuck seems disappointed that Forth, Inc. didn't go further than it did, given the powerful help it had early on.

Outside capitalization is one aspect of business startup that Chuck is adamant about. "Do not look for venture capital!" he advises. "As soon as you get involved in business to that degree, the business takes over. Whatever you wanted to do with the business will get lost in the operations." Instead of looking for venture capital, Chuck suggests, "scrape together your starting capital and go with that; don't get ambitious and try to shoot the moon."

Chuck spent some time as president of Forth, Inc. (Elizabeth Rather was the first company president), but he eventually gave the position back to Elizabeth. "In order for me to do what I wanted to do, I had to focus on the task at hand. I was really surprised at how much was involved in running the company. If that had been my task, I would not have been able to develop the applications (the company) needed."

Chuck's 25 years as consultant, chip designer and businessman have taught him a great deal about striking out on your own. He encourages others to do the same if it feels right for them, but offers this last bit of advice; **"...you won't get rich, but rich is much overrated."**

ME

Karl Lunt may be reached at 2133 186th Pl. SE, Bothell, WA 98012, 206-483-0447 or internet address karl@mav.com