

CIRCUIT CELLAR®

THE MAGAZINE FOR COMPUTER APPLICATIONS

SILICON UPDATE

Tom Cantrell

Twenty Years Ago, Today

Tom takes us back to the dawning of the DSP. Follow him as he marks important moments in DSP development leading up to today's modern DSP chips.



Though many are familiar with the story of the microprocessor, few may recall the early days of DSP. [1] It's illuminating (and timely, being roughly the twentieth anniversary) to take a look back. Frankly, it's also amusing.

For instance, Intel is known for being first in micros, and we know they were responsible for a number of other important inventions, such as DRAM, EPROM, and flash memory. However, they also came up with a number of other firsts that turned out to be lasts, including LCD watches, bubble memory, and, let's not forget, the DSP.

What? You never designed in the venerable

Intel 2920 (see Figure 1) introduced in 1979? This puppy was by definition a signal processor (actually, a digital analog computer), seeing as the inputs and outputs were analog!

Actually, you can dimly discern the underpinnings of modern DSPs in the 2920 if you look close enough. For instance, it had a "very-short, long-instruction-word" architecture with each 24-bit instruction comprised of separate parallel operations for ALU, barrel shifter (left shift 2 to right shift 13), and analog conversion (successive approximation done in software). The 2920 also featured conditional execution (*loads*, *adds*, and *subs*), which is now all the rage.

The good news is the 2920 was a single-chip DSP, with all program and data memory on-board. The bad news is there wasn't much of it—192 × 24-bit EPROM program memory and 40 × 25-bit RAM.

With a maximum of 192 instructions, the 2920 didn't even have a branch instruction per se. Instead, you would just string out your code (inlining they call it now) and end it with the EOP instruction, which would reset the program counter to

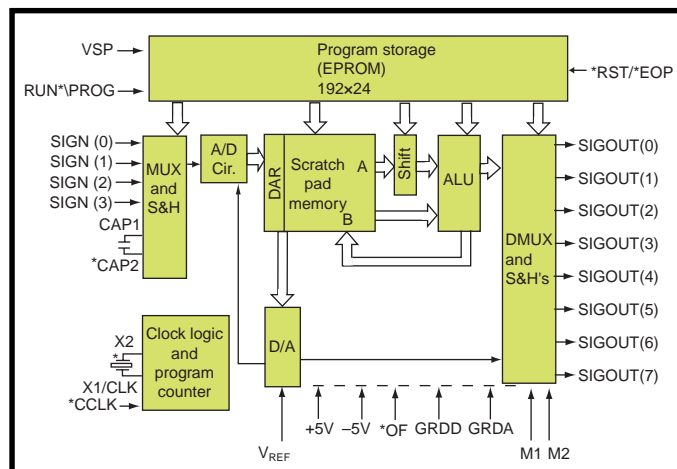


Figure 1—Introduced in 1979, the Intel 2920 was one of the first DSP chips. With only a 192-instruction EPROM and 40-word data RAM, it was perhaps a case of "too little, too soon," and acceptance was lukewarm at best.

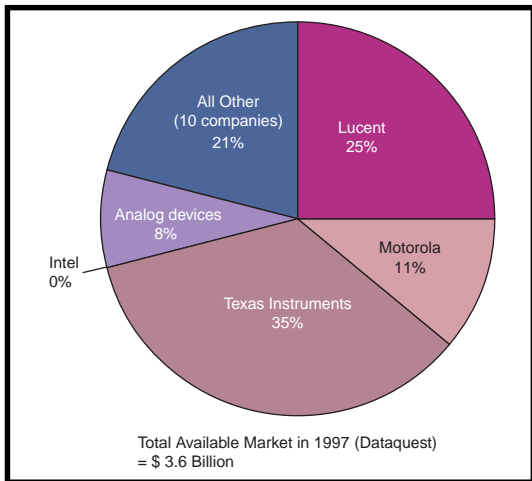


Figure 2—Motorola and Lucent didn't need a DSP to figure out that number two plus number three is greater than number one, and the StarCore joint venture was the result.

zero. There weren't any inner loops in 2920 apps, because there weren't any outer loops—just “the loop.”

Lack of a multiplier was a rather key omission, so if you sprung for the most expensive 2.5-MIP version of the part, you might be able to handle up to 10 kHz. Need more bandwidth? Make your program shorter. Needless to say, there was no C compiler. Even those used in dealing with the baroque programming called for by other Intel chips did plenty of head scratching over lines of code like *ADD DAR, KM2, R0, CND6*. Guess it was a good thing you only had to keep 192 of them straight.

Though the 2920 never sold worth beans, it was arguably the first commercial DSP. However, there were other contenders, including the then semi-major player AMI (until it was laid low by the mid-80s video-game blowout). They announced their S2811 signal-processing peripheral in 1978 but had process problems that ultimately stopped the show. Meanwhile, NEC announced their own UPD7720 in 1980 but, no doubt pre-occupied with DRAMs and such, failed to get the tools together or mount an effective promotion effort.

In terms of actual delivery, the first place ribbon should probably go to Bell Labs who disclosed their DSP1 at the same time as NEC. It's not completely surprising that they came up with a good chip, considering their legacy of names like Nyquist, Bode,

and Shannon on the payroll. Talk about interesting chat around the water cooler! Of course, it also helps to get designed in when headquarters (AT&T) happens to own the phone business. The DSP1 quickly found its way into a variety of telecommunication equipment apps and arguably sparked the first wave of digitization of the network.

BIG BANG DSP

Those of you who subscribe to the hard-copy version of *Circuit Cellar* already know a bit about StarCore,

the brand new DSP architecture jointly developed by heavyweights, Lucent and Motorola (see “DSP Doings” in *Circuit Cellar* 109).

To refresh, it was in June of 1998 when the alliance was first announced, and just about a year ago that the team disclosed the initial technical details of the SC140 core.

This is an interesting deal from many angles. On the business front, the premise is simple enough—combine the forces of the number two and number three DSP suppliers to go after number one, TI (see Figure 2). Yet StarCore isn't a full-fledged spinoff (As far as I can tell, the staff members remain employees of their respective companies.), but is more than a mere collaboration with, for example, a completely separate facility in Atlanta, Georgia.

Though developing the core jointly,

each partner will use it to develop their own chips, even as both companies get cross-licenses for the others existing DSPs (the Motorola 56K and Lucent '16x, respectively). In addition, Lucent gets a license to Motorola's M-Core embedded RISC, itself a bit of a skunk works project within Motorola.

In addition to the core itself, StarCore is responsible for delivering the tools (i.e., compilers, ect.). At the same time, they're supporting third-party tool suppliers, such as Green Hills. Furthermore, like M-Core, StarCore is purportedly available for licensing by other chip suppliers, although, to the best of my knowledge, no such deals have been announced.

Technically, StarCore (see Figure 3) finesses a whizzy-yet-practical take on the VLIW concept, which witnessing the success of the TI 'C6X, and now with the blessings of Intel with Merced (a.k.a., 'Itanium), it is clearly the heir apparent to superscalar for performance-at-any-price processors.

As important as the architectural rocket science, the implementation addresses pragmatic issues, such as no one wants to carry around a cell phone that needs a car battery. Clever logic design like activity-driven distributed decoding, combined with the latest low-voltage process, remarkably cuts power consumption to less than 1 mW per MHz at a mere 0.9 V. Meanwhile, 16-bit instructions (packed into 128-bit long instruction words) offer the promise of reasonable code density.

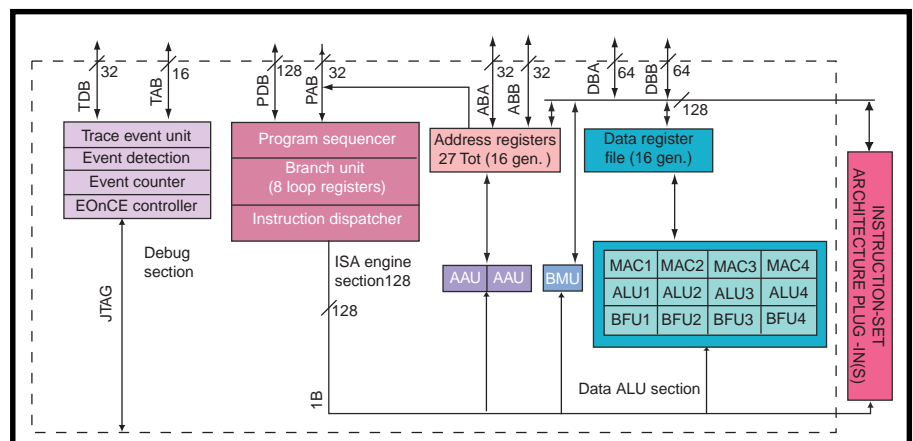


Figure 3—StarCore is a silicon black hole that swallows signals at up to 4.8 GBps.

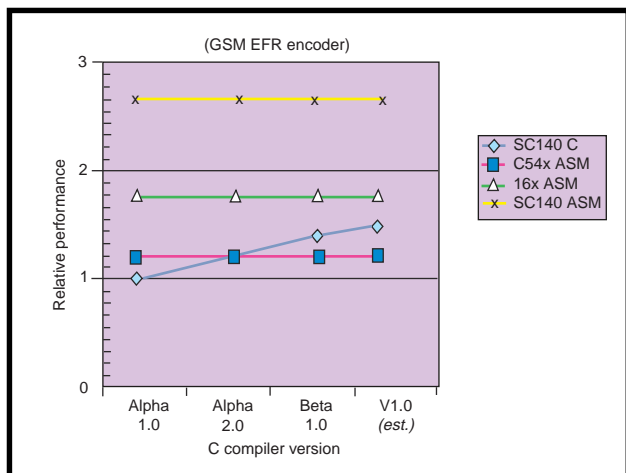


Figure 4—The StarCore compiler is improving, allowing the chip to meet or even exceed the performance of ASM on midrange DSPs. But, the 2x advantage of SC140 ASM over SC140 C puts lie to claims that C is a panacea.

COMPILER IS KING?

When it comes to VLIW, the basic premise is to replace, or at least assist, the superscalar hardware that attempts to maximize parallelism at runtime with software generated at compile time. Yes, a good compiler is always important—for a VLIW, it's everything.

There's no better way to get a compiler working than to try some real code. To that end, the StarCore folks bit the bullet and ported an actual application—a GSM EFR (enhanced full rate) Vocoder that compresses/decompresses high-quality speech to 12.2 Kbps. [2] As an international standard, C source code and validation suites are available, making a good test case for the compiler.

As shown in Figure 4, the compiler has been making steady progress through the current beta V.1.0 with projections of nearly a 50% overall speedup by the time the V.1.0 is released, which is expected in June.

Figure 4 also makes some other interesting points. One being that the combination of the SC140 and C achieves the performance of optimized assembly code on mere mortal DSPs. However, don't make the mistake of interpreting this to mean assembly language is dead. Indeed, as the figure

shows, comparing apples to apples, a hand-tuned assembly language version of the vocoder running on the SC140 is fully twice as fast as the C version.

The StarCore team decided to explore this phenomenon. After all, much of the marketing behind big-ticket DSPs alleges that ASM is history. It would be helpful (but, not absolutely necessary) if the facts of the matter didn't prove otherwise.

First, the code was modified to allow the encoder and decoder to run separately and each to handle multiple channels. After the modified code was tested on a workstation, it became the baseline StarCore compiler benchmark.

The team identified two approaches to optimization. The most common is to follow the 80/20 rule and hand code assembly language for the 20% of inner loops that execute 80% of the time and use C for control code that consumes most of the bytes but few of the cycles.

The other approach, which they call structured C, involves modifying

the source in ways that allow the compiler to generate better code. Theoretically, this is ideal because it achieves performance without sacrificing portability. However, the resulting source can become hard to read and maintain, defeating the goal of using a high-level language in the first place.

Although, a few source-level transformations were made to fully exploit the parallelism in the architecture, the team mainly relied on ASM optimization. The results, shown in Figure 5, are quite illuminating.

Bottom line and timeless truism: A little ASM goes a long way!

ONCE MORE, WITH FEELING

Now, Motorola is announcing the first chip to utilize the SC140 core, the MSC8101 Network Ready DSP (see Figure 6). Designed for network infrastructure applications (a.k.a., big iron), the 300-MHz SC140-core and now *de rigeur* huge chunk of SRAM (512KB) are just the start. The '8101 further integrates the dedicated communications processor module (CPM) and 32-/64-bit PowerPC bus found on Motorola's current high-end networking chip, the MPC8260 PowerQUICC II.

The CPM, a 32-bit RISC running at 150 MHz is no slouch, handling everything from ATM to Ethernet. Sixteen DMA channels access up to eight banks of SRAM, DRAM, EPROM, or flash memory supplemented with a dedicated pipelined SDRAM interface. There's even an Enhanced Filtering Coprocessor to streamline the innermost loops (a signal processing coprocessor for the signal processor if you will). Put any more processors on this thing, and you will need name tags to keep them all straight ("Hi, I'm EFCOP. What's your SIN?").

As I related in the beginning of this article, history shows that creation of the silicon DSP was fueled by (and, in turn, further fueled) the first wave of digital communications. And guess what, it is *déjà vu* all over

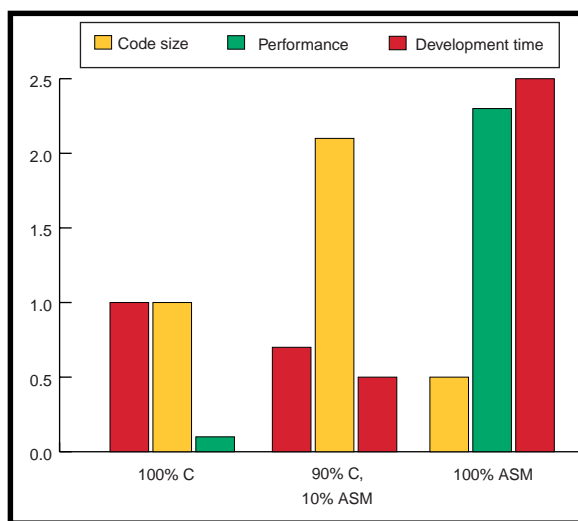


Figure 5—According to Motorola's experience with an actual application (GSM EFR Vocoder), unless you're willing to leave half your performance on the table, some DSP code (fortunately, not a lot) must be written in assembly language.

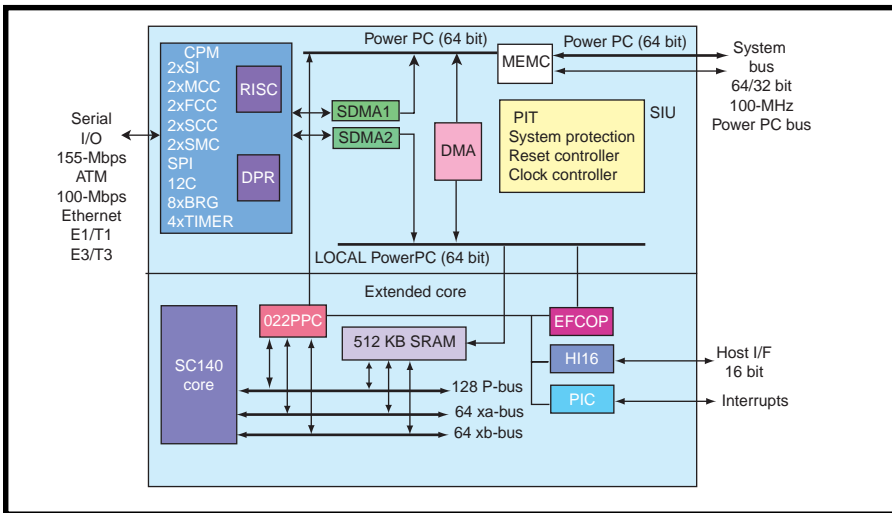


Figure 6—The MSC8101: millions of MIPS, 17 mm x 17 mm plastic package, only half a watt, and under a hundred bucks.

again for the latest wunderDSPs like the MSC8101.

Today, we're not just talking a few long lines or Central Offices, but turning every inch of wire (and Hz of wireless spectrum) into a bottomless pipe for ones and zeros.

Needless to say, that will require mucho MIPS and MACS. Fortunately, now there are incredibly powerful DSPs like the MSC8101 that designers can count on to do the heavy lifting. 📦

Tom Cantrell has been working on chip, board, and system design and marketing in Silicon Valley for more than ten years. You may reach him by e-mail at tom.cantrell@circuitcellar.com, by telephone at (510) 657-0264, or by fax at (510) 657-5441.

REFERENCES

- J. Boddie, "On the 20th anniversary of DSP1", <http://www.lucent.com/micro/dsp/dsphist.html>
- Halahmi et al, "GSM EFR Vocoder on StarCore 140", paper presented at DSP World/ICSPAT, 1999.

SOURCES

StarCore DSP
StarCore
www.starcore-dsp.com

StarCore DSP
Lucent Technologies
www.lucent.com
MSC8101 Network Ready DSP
Motorola
www.motorola.com/sps

Circuit Cellar, the Magazine for Computer Applications. Reprinted by permission. For subscription information, call (860) 875-2199, subscribe@circuitcellar.com or www.circuitcellar.com/subscribe.htm.