

Ferro-51

Twenty-five years after the debut of the original, a new '51 MCU is making headlines. Tom describes Ramtron's VRS51L3074 FRAM-enhanced '51 MCU and prepares you for what's to come.

By the time you read this, I'll have given my "Is ARM the '51 of Tomorrow?" presentation at the ARM Developers' Conference in Santa Clara, California. Can ARM match the popularity and longevity of the venerable '51? Is it MCU déjà vu all over again?

Leaving those questions for another day, this month's topic draws from one of the punch lines in my presentation. In one sense, ARM can't be the '51 of tomorrow because, ta-da, the '51 is the '51 of tomorrow.

That's right. While there's no doubt that low-cost flash memory 32-bit MCUs are taking off, the idea that they will replace 8-bit MCUs overnight is ludicrous. Rather, my take is that while the 8-bit MCU growth rate may be topping out, the 8-bit story is far from over.

One thing both ARM and the '51 do uniquely share is a multisourced architectural bandwagon. That's important because it increases the variety of specialized parts designers have to choose from. Case in point is this month's topic, yet another "new" '51 making its debut fully a quarter century after the original.

MEMORY MAPPING

As I was researching last month's article on alternative memory technologies, I came across an interesting press release on Ramtron International's web site. As I mentioned in that column, when it comes to MRAM, FRAM, and the like, I think embedded applications are a real market, one that's overlooked in all the Holy Grail hype of

replacing stand-alone DRAM and flash memory. Thus, it's no surprise that "Ramtron Introduces the First FRAM-Enhanced 8051 MCU" was right up my alley and definitely got my attention.

Ramtron's been in business selling FRAM memory chips for a long time, but I didn't know they made '51s. Ah ha, it turns out they're new to the MCU game, buying into the market with their recent (August 2005) acquisition of Goal Semiconductor, a niche supplier of mixed-signal '51s.

Imagine an MCU that gets by with just FRAM instead of the motley collection of flash memory, RAM, EEPROM, ROM, etc. that clutters today's chips. Better yet, FRAM would eliminate all the granularity issues (e.g., code versus data storage) associated with today's fragment-

ed multimemory setups. Faster, lower power, nonvolatile, cheaper, and more flexible: that's what I'm talking about!

Sounds good, but that's not what the new Ramtron VRS51L3074 microcontroller is. The press release is careful to say that the '3074 is just the first step along the way. Witness the fact it still has flash memory (64 KB) and SRAM (4 KB + 256 bytes) along with the 8 KB of FRAM (see Figure 1). Indeed, under the hood, I'm pretty sure you're looking at separate MCU and FRAM chips rather than true unification of FRAM and CMOS processing on a single die.

Based on the specs of Ramtron's very SRAM-like 1-Mb chip that I covered last month, I expected to find a similar FRAM on the '3074. Thus, I was a bit surprised to find that the integrated FRAM on the '3074 isn't nearly as capable as the stand-alone part.

First the good news. Once all is said and done, access to the on-chip FRAM is via the '51's MOVX instruction (i.e., the same instruction used to access the on-chip SRAM as well as external memory and I/O devices). There's no need for the multi-instruction software bit-banging or programming routines typically required by flash memory or EEPROM. A bit of software is required to set up and enable the FRAM; but after that, a MOVX is all it takes.

Last month's 1-Mb FRAM chip did a decent job of delivering on the promise of next-generation memories by offering the speed and read/write symmetry near that of an SRAM. However, the FRAM on the '3074 isn't especially fast or sym-

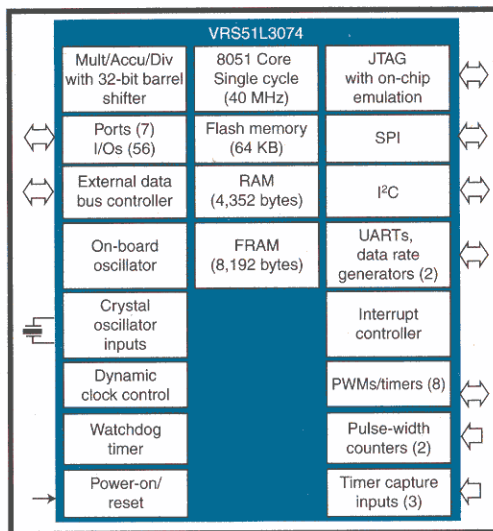


Figure 1—Ramtron makes their move into the MCU business with an assist from Goal Semiconductor, which they acquired last year. The headline feature that sets their VRS51L3074 apart from the crowd is 8 KB of FRAM, but this '51 is also notable for its performance features.

metric, with minimum access times of 1.9 and 2.6 μ s for reads and writes, respectively. Note that the MOVX is interlocked so that the processor stalls until the FRAM read or write completes.

Like their stand-alone memory chip, the '3074 FRAM does support Burst modes that can increase throughput somewhat (see Table 1). But Burst mode has some gotchas that can trip up unwary designers. Once you put the FRAM in Burst mode, you must read only consecutive addresses or write consecutive addresses. Mixing reads and writes or skipping addresses isn't allowed.

Furthermore, there is a timing constraint limiting the idle time allowed between each transfer in a burst. If you dally too long, the FRAM automatically exits Burst mode. Although the idle time allowed is somewhat programmable (four choices), it does require that you guarantee your software can meet the required deadline. For example, this may require disabling interrupts during a burst transfer or at least extra software to correctly restart an interrupted burst.

The scary part is that if you make any mistakes (e.g., mixing reads and writes, nonsequential address, missing the burst timeout deadline) the FRAM can lose data, and there's no interrupt or error flag to let you know that happened. If you must use Burst mode, it's there, but be careful.

If there's a bright side, the relatively slow access time in conjunction with ever-improving endurance specs virtually eliminates concerns about wear out. The '3074

FRAM retention time spec is a whopping 45 years! Although I couldn't find any retention time or endurance specs for the flash memory in the preliminary datasheet, I presume they're similar to those of other flash memory MCUs (e.g., ten(s) years, thousand(s) cycles).

MIPS MASTER

The original '51 was rather pokey, maybe hitting 1 MIPS with a tail-

wind. By contrast, like practically all '51s these days, the '3074 offers good performance by a combination of architectural turbocharging (fewer clocks per instruction) and more revs (i.e., faster clock up to 40 MHz). I figure a typical application program mix would average three to four clocks per instruction for an honest 10+ MIPS.

The '3074 features a 40-MHz oscillator on-chip that, with 2% precision, should be accurate enough to eliminate the need for crystal in many applications. In addition, an external 4- to 40-MHz crystal can be connected, and you can dynamically switch between the two. In turn, the chosen oscillator feeds a 16-level power-of-two prescaler (i.e., divide ratio = 1, 2, ... 32,768), which allows software to control the throttle depending on the task at hand. There's even a feature that can automatically hit the gas (i.e., set clock divide ratio to one) when an interrupt occurs.

For number crunching, the '51's built-in MUL and DIV instructions are standouts at just two clocks (i.e., 50 ns at 40 MHz). But the math capability of the '3074 goes way beyond just speeding up those native instructions. As you can see in Figure 2, the chip integrates a 16/32-bit arithmetic coprocessor and barrel shifter that can significantly accelerate math-intensive applications (e.g., DSP loops). There is a bit of software overhead (preferably ASM

Mode	FRAMCLK[1:0] = 00 (Sysclk/2)		FRAMCLK[1:0] = 01 (Sysclk/4)	
	Read	Write	Read	Write
Normal	1.9 μ s	2.6 μ s	3.6 μ s	4.9 μ s
Burst*	1.1 μ s	0.4 μ s	2.3 μ s	0.7 μ s
Read burst*	0.7 μ s	0.4 μ s	1.6 μ s	0.7 μ s

Table 1—FRAM has a lot of potential, but the implementation on the '3074 could be better (i.e., faster with symmetric read/write times). Burst modes help, but impose system timing constraints that may trip up the unwary.

for speed) to drive the coprocessor, but ultimately, it returns answers as fast as you can feed it operands. It's no DSP, but the math coprocessor does deliver signal processing capability far beyond that of most 8-bit MCUs.

Another embellishment is a second DPTR (data pointer) register to bypass the well-known bottleneck imposed by the originals single pointer. For example, block memory operations can use one of the data pointers for the source address and one for the destination rather than having to coerce a single DPTR into serving double duty.

There are also additional instructions that streamline access to the all-important special function registers (SFRs) that control the various I/O ports. In the interest of avoiding any possible software compatibility issues that might arise with the new opcode (which is a NOP for regular '51s), this feature can be dynamically enabled and disabled by software.

Speaking of I/O, '3074 upgrades fall into two categories. First, be reassured the historic '51 I/O features are all there and can work the same as before.

Indeed, the pinout for the 44-pin version of the MCU ('3174) is the same as that of many traditional '51s. However, going beyond compatibility, there are also options that enhance the traditional I/O functions with new capabilities.

For example, the original '51 required using a general-purpose timer as a bit rate generator for its built-in UART. By contrast, not only does the '3074 provide dedicated bit rate generators, but the

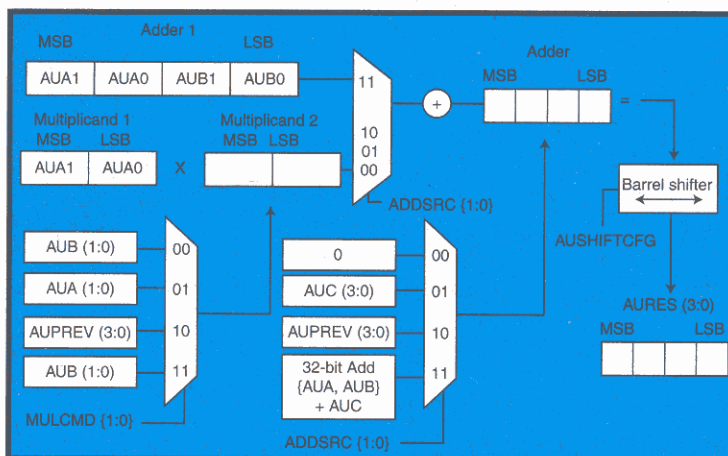


Figure 2—With a combination of 32-bit multiplier, adder, and barrel shifter, the '3074A built-in math coprocessor turns this '51 into a mini-me DSP.

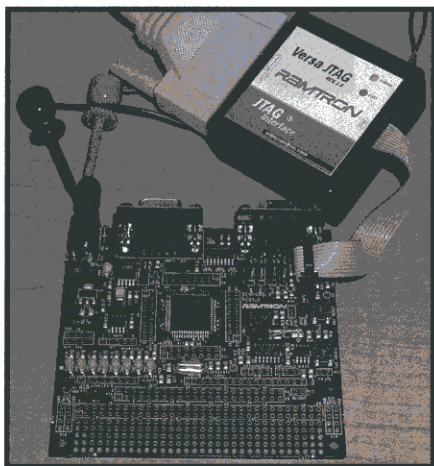


Photo 1—The '3074 evaluation kit is relatively standard fare, but with a notable FRAM bias. In addition to the FRAM on the MCU itself, the board includes three stand-alone serial FRAM chips.

general-purpose timers also feature new modes as well. For example, timers that are limited to 8- or 16-bit operation in a traditional '51 can be chained together to deliver 24, 32, and even 48 bits of resolution. Another upgrade supplements the general-purpose (i.e., parallel) I/O lines with a pin monitor function that can detect a change (edge or level sense) on any and all pins of interest.

The second class of I/O enhancements includes new features not found on the original '51. These include SPI (enhanced to support transfers up to 32 bits in length) and I²C (both Master and Slave modes), as well as even more timing and counting capability in the form of an eight-channel PWM and a two-channel pulse-width counter. The PWM features 16-bit resolution and can be used for generic timing if PWM capability isn't required. The pulse-width counter feature works with the general-purpose timers (i.e., T0 and T1) to boost event timing and gating flexibility by allowing the count start and stop conditions to be programmably defined in various ways. For instance, you could configure the timer to start counting on a rising edge of one pin and stop counting on the falling edge of a different pin.

FREE "C"

Given the maturity and popularity of the '51, one thing you can be sure of is that there's no shortage of development tools. Even as a new player, Ramtron has fairly decent support,

including packages from major suppliers like Keil and American Raisonance. Better yet, over the years, a variety of low-cost and even free alternatives have emerged that belie the traditional truism that you get what you pay for. With Ramtron's \$99 '3074 evaluation board in hand, let's see how far freeware can take you (see Photo 1).

The board itself is rather minimalist and straightforward with little more than the '3074 chip, RS-232 transceivers for its twin serial ports, some LEDs, and a connection for a JTAG debugger. Somewhat ironic considering the '3074's built-in FRAM, the only notable add-ons are a trio of the company's stand-alone FRAMs, including a processor companion chip and SPI and I²C variants.^[1] It's a veritable FRAM traffic jam! That seemingly odd situation is explained by the fact the same board is used for evaluating the various FRAM chips and non-FRAM variants of the MCU.

The JTAG debugger dongle itself gets marked down by virtue of the fact it uses a printer port interface with the PC. That was once a common hack, but it has fallen out of favor with the rise of USB and the fall of PCs equipped with parallel ports. No problem you say, I'll just use a USB or serial-port-to-parallel-port adapter. Sorry, Charlie. Those are designed to talk to printers, and the manual cautions that they won't work. Fortunately, my desktop PC has a parallel port, and luckily I didn't even have to twiddle any BIOS (EPP, ECP, etc.) or driver settings (port addresses or interrupts) to get the thing to work.

Along with evaluation versions of the pro tools (Keil, etc.), the kit comes with the bits and pieces of software that allow you to cobble together a freeware solution. Compared to the elegant (and pricey) one-screen-does-all IDEs, it may seem a bit of a hack, but I actually found the old-school solution quite serviceable.

The low-budget lash-up comprises three components: a programmer/debugger from Ramtron, a GNU-based "Small-Device C Compiler" (SDCC) suite, and an editor of your choice. (I installed the freeware "Crimson" editor that came on the CD.)

Ramtron's JTAG software packages

flash memory programming and software debugging in a single program, although each function is relatively stand-alone. The programming portion handles basic flash memory operations and is really simple. The streamlined interface offers one-click programming (i.e., erase, write, and verify), while the ability to flash individual blocks and multiple files supports customization (e.g., serial number and calibration data).

Once you burn a program into flash memory, the debugger provides the critical link between your source code and the relatively sophisticated debug hardware built into the '3074 chip. There are six hardware breakpoints that work in the usual fashion (i.e., stopping the processor when a particular address is accessed). In addition, there's a "value"

```

ORG 2000h
Start:
  MOV  A, #10 ; Load ACC with 10
Loop:
  DEC  A
  JNZ  Loop
  LCALL Function ; Call a function
  JMP  Start
ORG 3000h
Function:
  RET ; Dummy function

```

Branch	Loop repeat	Loop skip
1	0x2003	0x2003
2	0x2003	0x2005
3	0x2003	0x2008
4	0x2003	0x2003
5	0x2003	0x2005
6	0x2003	0x2008
7	0x2003	0x2003
8	0x2003	0x2005
9	0x2003	0x2008
10	0x2003	0x2003
11	0x2005	0x2005
12	0x2008	0x2008
13	0x2003	0x2003
14	0x2003	0x2005
15	0x2003	0x2008
16	0x2003	0x2003
17	0x2003	0x2005
18	0x2003	0x2008
19	0x2003	0x2003
20	0x2003	0x2005
21	0x2003	0x2008
22	0x2003	0x2003
23	0x2005	0x2005

Table 2—The '3074's built-in debug hardware includes breakpoints and a low-budget real-time trace capability. As shown here, the latter has the option of tracing into or around tight loops.

breakpoint option that stops the processor based on the value of a data item. Although a “value” breakpoint consumes two breakpoint slots, it can be a real timesaver compared to stopping on every access.

There’s even a measure of real-time trace capability built into the chip. This is something I’ve seen on bigger and newer chips, but I can’t recall seeing it on a ‘51. It tracks branches

(i.e., J MPS, but unfortunately not CALLS or RETS) nonintrusively (i.e., unlike software “non-real-time” trace schemes) and has the option of skipping repeated entries (i.e., tight loops). Yes, it’s kind of clunky just showing the raw addresses (see Table 2). Maybe a future revision could link those raw addresses back to labels in the source code or even provide a “Step Backwards” button. Nevertheless, like the “value” breakpoints, this mini-me real-time trace is a feature that could come in really handy when you’re scratching your head over some intermittent “How in the heck did I end up here?” bug.

As a C compiler that supports “small devices” like the ‘51, ‘68, and PIC, I can say SDCC has its heart in the right place. Although not fully ANSI-compatible, it does have specific provisions for things like I/O, interrupts, and in-line assembler that reflect a sensitivity to the needs of memory-constrained real-time applications. For example, while the compiler fully supports 32-bit floating-point math, it also offers streamlined versions of library routines (e.g., printf) that minimize the associated bloat.

One thing that you don’t get with SDCC (and don’t pay for either) is a fancy IDE. Frankly, I think a lot of the popular commercial setups have feature-creeped beyond the needs of simple 8-bit applications. On the other hand, the old-school command-line setup (i.e., DOS box) used by SDCC, although capable enough in the hands of an expert, seems increasingly dated.

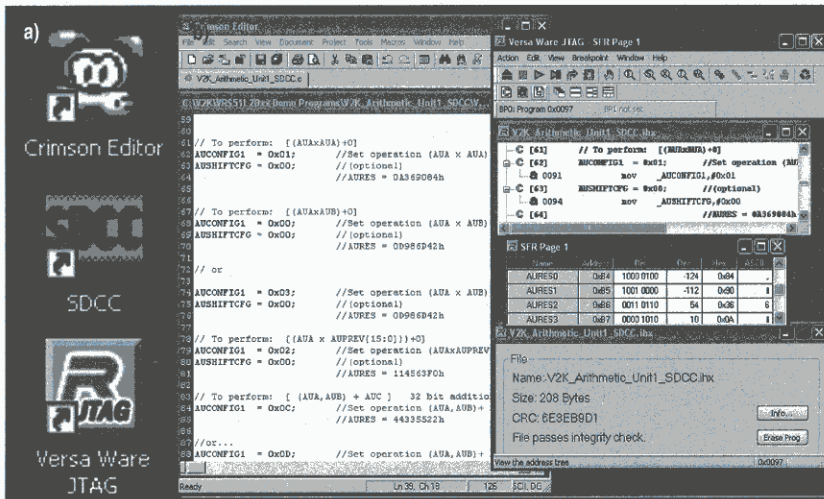


Photo 2—Combining an editor with SDCC and the Ramtron JTAG debugger software (a) would seem like a rather primitive solution, but in fact it’s surprisingly effective and easy to use (b). And with a \$0 price tag, you’re definitely getting more than you pay for.

A decent compromise is to exploit the fact that most programming-oriented editors include the capability to define editor commands and hotkeys that invoke external programs. Following the directions included with the Ramtron kit, I was able to hook SDCC into the Crimson editor for seamless point-and-click (i.e., no command line) operation (see Photo 2).

NEWS FLASH

After all is said and done, I’d say the “FRAM Enhanced” headlines for the ‘3074 kind of miss the point. First, as I described earlier, the actual FRAM functionality seems a bit disappointing, especially in light of the more SRAM-like (fast, symmetric read/write) specs for their 1-Mb memory I covered last month. Indeed, I wonder if a better stacked-die option wouldn’t just combine that chip with a de-flashed (and maybe even de-RAMed?) ‘51 MCU core.

Further irony given all the “flash is dead” hoopla is found in the fact that the flash memory on the ‘3074 actually works quite well, notably delivering the goods at 40 MHz with no wait states. Yes, it’s got the usual endurance, retention, and write speed/power limitations, but at least it’s easy to use by virtue of a built-in flash program interface (FPI) supplementing the external (i.e., JTAG) programming option. Invoking the FPI shadows in high-level flash memory functions (overlying the top 1 KB of code space) that take care of all the dirty work for in-application programming.

Thanks to the FPI, using the flash memory is as easy and arguably less prone to glitches and head-scratching than using the FRAM.

This isn’t a knock on FRAM in general. Ramtron’s stand-alone parallel and serial memory chips have really grown up and are already proving themselves in real applications. Rather, it’s just the specific implementation in the ‘3074 that

seems a bit puzzling. It is all a wakeup call that FRAM and other new age memory technologies are headed in the right direction but have a ways to go.

The popularity and longevity of the ‘51 uniquely derives from a “more the merrier” supply side—the Ramtron parts being the latest, although surely not last, addition. Putting aside the “FRAM Enhanced” headline for the moment, maybe the story worth noting is that the ‘3074 and Ramtron’s other ‘51s seem pretty darned competitive. Sure, there are still some rough edges, but I’d say their combination of FRAM know-how and mixed-signal ‘51 expertise gives Ramtron a decent chance for success. ☐

Tom Cantrell has been working on chip, board, and systems design and marketing for several years. You may reach him by e-mail at tom.cantrell@circuitcellar.com.

REFERENCE

- [1] B. Millier, “Battery-Free Nonvolatile RAM,” *Circuit Cellar* 193, 2006.

SOURCES

Crimson Editor
www.crimsoneditor.com

VRS51L3074 FRAM-Enhanced ‘51 MCU
Ramtron International Corp.
www.ramtron.com

Small-Device C Compiler
http://sdcc.sourceforge.net/