

M I C R O P R O C E S S O R

www.MPRonline.com

THE INSIDER'S GUIDE TO MICROPROCESSOR HARDWARE

ATMEL'S CUSTOMIZABLE MCUS

Metal-Programmable Gates Add Flexibility to ARM-Based Microcontrollers

By Tom R. Halfhill {10/29/07-01}

These days it's easy to make Atmel's marketing people cringe. Just refer to their new Customizable Atmel Processors (CAP) as "structured ASICs." Then get ready to duck. So disenchanted has the industry become with the once-bright promise of structured

ASICs that any association is as dreaded as clock skew in a processor's timing chain. (See our two-part series in *MPR 7/2/07-01* and *MPR 7/9/07-01*, "Structured ASICs: Dead or Alive?") But Atmel doesn't believe in cache-flushing the baby with the bath water. Atmel's CAPs invert the structured-ASIC formula to preserve the good aspects (design flexibility, rapid turnaround) while avoiding the bad aspects (complex design and verification, insufficient advantages over FPGAs and standard-cell ASICs).

Instead of offering a blank slate of programmable metal encompassing nearly the whole chip, Atmel's CAPs are fundamentally ARM7- or ARM9-based microcontrollers with the usual integrated peripherals and I/O interfaces. As Figure 1 shows, only about 10% to 20% of the chip is reserved for a metal-programmable block. Indeed, the chips are capable of operating as fully functional microcontrollers, even without using the programmable block—although that would be unwise, because CAPs inevitably cost more than conventional microcontrollers. However, by using the programmable block to integrate additional peripherals, application-specific logic, or even multiple processor cores, customers can transform these off-the-shelf parts into the near equivalent of a custom ASIC.

For instance, if a motor-control application needs more pulse-width modulators (PWM) than the two or four 16-bit PWM channels supplied on CAPs as standard equipment, customers can use the programmable block to add more—perhaps dozens more. (Atmel speaks of "PWM farms.") If one

10–100MB/s Ethernet controller or two Full-Speed USB host controllers aren't enough, simply add more controllers. If a secure application requires hardware-assisted cryptography, add a DES or AES accelerator. And so on. It's even possible to supplement the built-in ARM7 or ARM9 processor core with a second ARM processor, or to add multiple 8051-compatible processor cores.

CAPs provide another alternative for bridging the gap between standard parts and ASICs. They're priced a little higher than standard microcontrollers, but they are customizable—without the huge engineering costs and time-to-market lag of developing an ASIC. CAPs have fewer metal-programmable gates than fully structured ASICs, but they require much less development work and verification. They aren't as flexible as FPGAs, but they run faster, leak less power, and cost about the same in volume. Although CAPs aren't for everyone, they fill an important niche between FPGAs and ASICs.

Exploring Gate-Level Programmability

Microcontroller catalogs from major vendors offer hundreds of standard parts, often with minor variations in features. Two related parts might be identical in every way except for one integrated peripheral or I/O interface. In fact, many of these chips are built on identical dies, with bond-out options determining which features the vendors expose to customers. Yet despite this variety, some customers can't find a standard part that precisely fits their needs. They end up paying for

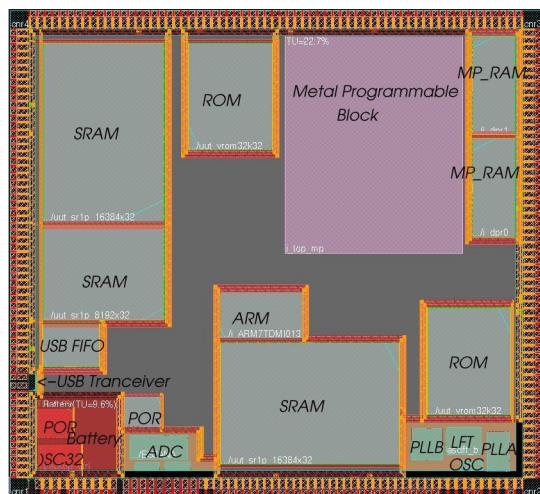


Figure 1. Customizable Atmel Processor (CAP) die photo. Unlike conventional microcontrollers, CAPs reserve a region of metal-programmable gates for user customization. Unlike structured ASICs, the customizable region occupies only 10–20% of the chip's total area. The rest of the chip contains an ARM7 or ARM9 processor core plus common microcontroller peripherals and I/O interfaces. Atmel calls the customizable region a Metal-Programmable Cell Fabric (MPCF) or "metal-programmable block." Some engineers prefer the term "mask-programmable block," because the block requires custom masks for final fabrication. Unlike FPGA logic, the block is not field programmable or reprogrammable.

features they don't want or altering their designs to compensate for features they can't get in a single chip.

Atmel isn't the first company to recognize that a partially customizable microcontroller would be the answer to designers' prayers. One challenge is finding the best mix of integrated features and gate-level programmability. Another challenge is finding the best technology for that programmability. A famous early startup that tackled those challenges was

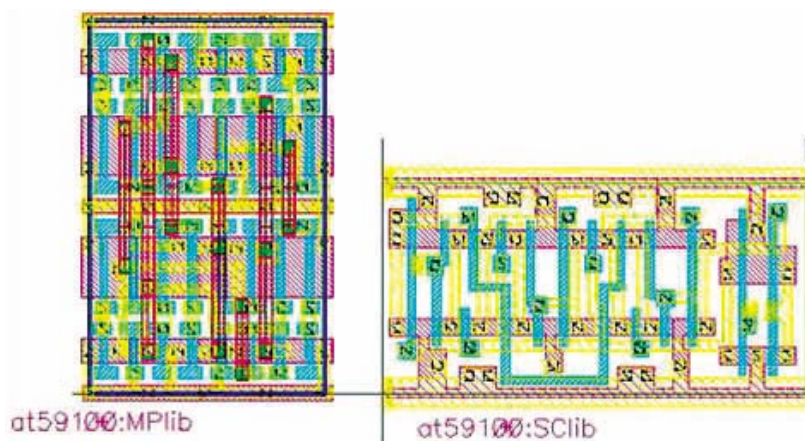


Figure 2. Size comparison of Atmel's metal-programmable gates with standard-cell gates. Both these D-type flip-flops (metal programmable, left, and standard cell, right) are implemented in 0.13-micron CMOS. Overall, metal-programmable gates are a little larger than standard cells and use about 10–15% more power.

Triscend, founded in 1997 by former Xilinx employees. Triscend's microcontrollers combined a 32-bit ARM7 or 8-bit 8051-compatible CPU core with some integrated peripherals and field-programmable logic. Customers could optimize the chips for specific applications by implementing additional soft peripherals and function units in the fabric. Unfortunately, Triscend and its products vanished soon after Xilinx acquired the company in 2004. (See *MPR 3/15/04-02*, "Xilinx Reconfigures Triscend.")

That same year, fabless-semiconductor startup Stretch announced its S5000 family, which integrates a Tensilica Xtensa V 32-bit processor core with field-programmable logic. However, Stretch's fabric is mainly intended for implementing application-specific extensions to the Xtensa core, not for adding microcontroller peripherals. (See *MPR 4/26/04-01*, "Stretching Performance.") A year later, STMicroelectronics announced its first hybrid chip of this type, the STW21000. More like Triscend's chips, the STW21000 combines the integrated peripherals and I/O interfaces of a standard microcontroller with a relatively small block (150,000 gates) of field-programmable logic. The STW21000's CPU core is an ARM926. (See *MPR 4/4/05-01*, "ST's Reconfigurable GreenField.")

This brief history of combining programmable logic with fixed functionality is by no means exhaustive, but it outlines the background behind Atmel's latest effort. The point is that the need for user-level customization is real, even if the ideal formula is elusive. Atmel's twist is to use customizable metal layers for the programmable block instead of a field-programmable fabric.

Advantages: much denser, faster logic, because conventional gates require less silicon than the lookup tables of reprogrammable gates. Conventional gates also have fewer transistors, so they leak less current when unclocked. In fact, as Figure 2 shows, Atmel's metal-programmable gates are nearly as compact as standard-cell ASIC gates. But there are disadvantages to Atmel's approach, too. Metal-programmable gates are defined by custom mask layers during fabrication, so they aren't reprogrammable at the gate level. Once the die is cast, customers can't change the chip without a re-spin. And to make the devices economical, customers must order them in large enough quantities to offset the cost of manufacturing and packaging.

It's interesting that Atmel, also an FPGA vendor, would choose metal-programmable gates instead of field-programmable gates for its customizable microcontrollers. Skeptics might wonder whether Atmel fears cannibalizing its FPGA business—relatively small though it is. More likely, Atmel

IP Block for Customizable Atmel Processors (CAP)	IP Supplier	Approximate Gate Count
AMBA Advanced Peripheral Bus (APB) IP		
Real-Time Clock	Atmel	3.5k
RS-232, RS-485, ISO-71816, IRDA Interfaces	Atmel	8k
Serial Synchronous Controller (TDM, I ² S, AC97)	Atmel	12k
Audio AC97 Controller Component v2.3	Atmel	15k
Two-Wire Interface (master and slave)	Atmel	5k
Serial Peripheral Interface (SPI)	Atmel	6k
SD Card / MMC Card Host Controller	Atmel	11k
Control-Area Network 2.0B + 8 Mailboxes	Atmel	40k
Parallel I/O (32 bits)	Atmel	11k
Timer / Counter	Atmel	11k
Pulse Width Modulator (PWM)	Atmel	18k
Data Encryption Standard (DES), 133MHz	Atmel	20k
Advanced Encryption Standard (AES), 128/196/256	Atmel	110k
Secure Hash Algorithm (SHA1)	Atmel	50k
AMBA Advanced High-Speed Bus (AHB) IP		
AHB / APB bridge	Atmel	2k
External SRAM / Flash Controller	Atmel	17k
Error-Correction Controller (ECC) for HSMC3	Atmel	20k
SDRAM Controller	Atmel	10k
ZBT RAM Controller	Atmel	10k
STN/TFT LCD Controller + DMA + AHB Interface	Sidsa	40k + DPR 256K x 16 bits + DPR 256K x 32 bits
Ethernet MAC 10/100Mb/s (MII + RMII) + DMA	Cadence	35k + external PHY
USB Full-Speed Host / DMA / AHB Interface	Synopsys	55k + external PHY + internal PHY*
LCD Graphical OS GUI	Amulet	45k + DPR 2K x 16 bits

Table 1. Soft macros available for CAPs. Atmel's macros are license- and royalty-free to CAP customers; third-party macros cost extra. Atmel notes that CAPs protect IP better than FPGAs do, because the logic is carved into metal and harder to steal. (*The internal PHY is a hard macro already integrated on chip. ARM7-based CAPs have one Full-Speed USB PHY, which is available if a customer uses the USB host controller in the metal-programmable block. ARM9-based CAPs have a High-Speed PHY and two Full-Speed PHYs on chip for the USB host controller.)

grasps the drawbacks of field-programmable logic for this purpose. Actually, Atmel already sells FPGAs with integrated hard processor cores and memory. Those devices, which Atmel calls Field-Programmable System-Level Integrated Circuits (FPSLIC), are similar to the processor-integrated FPGAs from the leading programmable-logic vendors, Altera and Xilinx. Atmel's CAPs are closer to structured ASICs than to these FPGAs, but they are really a different class of device altogether.

Looking Good Against FPGAs

One of Atmel's challenges was deciding how much area on the chip to reserve for customizing. Too many gates would make CAPs indistinguishable from structured ASICs. Not enough gates would make them inadequate for usefully customized designs. Ultimately, Atmel decided to reserve enough room to duplicate the programmability of a fairly large FPGA. This decision puts CAPs on a stronger footing against rival solutions that combine a standard microcontroller with an FPGA, as well as with FPGAs that integrate a hard 32-bit processor core.

The metal-programmable blocks on CAPs range in size from 250,000 to 500,000 routable gates. Because those are masked gates, nearly as dense as standard cells, Atmel estimates that an FPGA would need one million to two million reprogrammable gates to implement an equivalent amount of logic. According to Atmel's measurements, a mix of common microcontroller peripherals implemented in an FPGA would

consume about 68% more dynamic power than the same peripherals in a CAP. More significantly, the FPGA leaks about twice as much static power as a CAP—partly because FPGA gates are less efficient and require more transistors, and partly because large FPGAs are usually manufactured in the latest (leakier) fabrication processes. Atmel fabricates CAPs in low-leakage 0.13-micron CMOS. Atmel has measured static leakage for an ARM7-based CAP at 274 microamps at 1.256V (or 344 μ W).

Prices for CAPs are similar to those for FPGAs of similar size. Atmel's CAPs are priced at \$5 to \$13 in volumes of 50,000 to 100,000 units. Xilinx sells Spartan-3E devices with 250,000 to 500,000 system gates for about \$3 to \$5 in the same volumes. Of course, FPGA system gates aren't equivalent to the denser gates of CAPs. A Spartan-3E with one million system gates costs about \$10, still within the price range of CAPs—but the Spartan lacks an ARM processor and Atmel's built-in peripherals. Even as FPGA prices fall, Atmel can promote the higher performance and lower power consumption of CAPs.

In comparisons with conventional 32-bit microcontrollers, Atmel's best selling point is gate-level programmability. ARM-based microcontrollers without metal-programmable logic are available for as little as \$1, although most of them cost several dollars. All things considered, Atmel's CAPs aren't expensive for 32-bit microcontrollers with their unique capabilities.

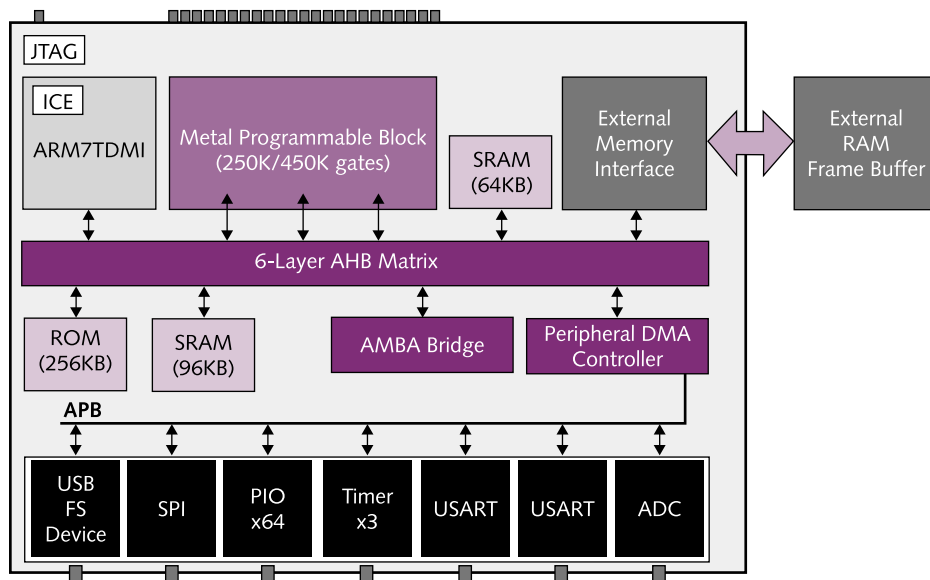


Figure 3. CAP7 block diagram. CAP7 devices have an ARM7TDMI processor, and CAP9 parts have an ARM926EJ-S. Integrated peripherals and I/O interfaces vary across the product line, but they are typical of 32-bit microcontrollers. Higher-end parts have High-Speed USB and 10–100Mb/s Ethernet.

Producing a CAP is practically a turnkey process, from the customer's point of view. Customers need no ASIC design experience. Atmel provides a development board with either an Altera or Xilinx FPGA, according to the customer's preference. Using standard FPGA development tools, customers develop, simulate, test, and verify the custom portion of their hardware design on the development board. When the design is finished, customers submit it to Atmel, which has its own tools for converting the FPGA implementation into a metal implementation. From that point onward, Atmel handles all the place-and-route, prototype fabrication, and volume production.

Atmel maintains an inventory of prefabricated CAP wafers in advance of customer orders. These wafers are complete except for the four additional metal layers required to implement the customer's design. Upon receiving an order, Atmel finishes the wafers, dices them, packages the chips, and delivers them to the customer. Turnaround time is 8–10 weeks. (While waiting for the first silicon, customers can start developing software on the FPGA system.) If a customer needs to make last-minute revisions to a design, only one or more of the custom mask layers must change. In contrast, respinning an ASIC usually requires making more new masks, which quickly becomes expensive. (A complete new mask set at this geometry would cost about \$500,000.)

Atmel reduces the customer's workload in other ways as well. The company offers a library of peripherals and I/O interfaces as drop-in soft macros—the same preverified macros that Atmel offers ASIC designers. Atmel's macros are license- and royalty-free. In addition, a few third-party companies offer licensable intellectual property (IP) for CAPs. Table 1 lists some of the IP available.

Compared with millions of dollars in nonrecurring engineering (NRE) costs to design an ASIC, Atmel's CAPs look like a steal. For \$150,000, Atmel shoulders all the design services from the point of receiving the customer's finished FPGA implementation. That flat fee includes final verification and manufacturing samples. For volume production of CAP7 devices (those based on the ARM7TDMI processor), customers pay \$5 to \$7 in 50,000-unit quantities, depending on whether the chip has 250,000 or 450,000 programmable gates. For volume production of CAP9 devices (based on the ARM926EJ-S), customers pay \$13 to \$15 in 100,000-unit quantities, depending on whether the chip has 250,000 or 500,000 programmable gates.

Only an ASIC manufactured in larger volumes could amortize its much higher NRE costs and compete in price with a CAP. A structured ASIC would be cheaper to develop than a conventional ASIC and potentially less expensive to manufacture in volume than a CAP, but it would still have higher NRE costs to amortize before breaking even. At the other end of the spectrum, a single-chip FPGA design—or a two-chip microcontroller/FPGA solution—would be more economical than a CAP only in smaller quantities. Overall, CAPs look like a good middle-of-the-road alternative to ASICs and FPGAs, thanks to their combination of low NRE costs, relatively low unit prices, and relatively large degree of customizability.

CAPs Are Microcontrollers, Too

The main selling point of CAPs is that they're customizable, but they are also fully featured microcontrollers. In all, Atmel offers nine different parts. Three have an ARM7TDMI processor, and six have an ARM926EJ-S. The approximate number of metal-programmable gates they can accommodate is 250,000, 450,000, or 500,000. There are several other distinguishing features, including clock speeds (80–200MHz), SRAM (32–160KB), and ROM (32–256KB), and the numbers and types of integrated peripherals, I/O interfaces, and packages. Of course, these differences also influence their power consumption and prices.

Figure 3 is a block diagram of a typical CAP7 part. Note the features common to many 32-bit microcontrollers, such as USB, SPI, real-time clocks, watchdog timers, and USARTs. Some CAPs have more advanced features, such as High Speed USB, 10–100Mb/s Ethernet, an LCD controller, and DES/AES crypto accelerators. Even without a metal-programmable block, these devices would be competitive microcontrollers.

The metal-programmable block is invaluable for adding more peripherals or even additional processor cores—whatever is needed to customize the chip for a special application. With as many as 500,000 programmable gates, ambitious things are possible. At the recent ARM Developers Conference in Silicon Valley, Atmel showed an example developed by partner Amulet Technologies: an LCD interface and GUI engine. This intriguing design allows an ARM7-based CAP to control user-friendly embedded systems that otherwise would require a more powerful microcontroller.

Amulet is a fabless semiconductor company that usually builds its Graphical OS in Silicon technology into its own chips. Porting the technology to Atmel's CAP and licensing the macro as soft IP is a new venture for the Santa Clara-based company.

Normally, an ARM7TDMI isn't powerful enough to drive an LCD by itself, because the demand for screen refresh exceeds the processor's bus bandwidth. Amulet's custom logic offloads that task from the CPU, freeing it to run application code. The Amulet LCD controller can drive passive monochrome or color LCDs with resolutions up to 640×480 pixels or color TFT LCDs with resolutions up to 800×600 pixels. It handles all refresh tasks and has an internal frame buffer sufficient for small displays. Larger screens require an external frame buffer in SDRAM. Figure 4 shows block diagrams of Amulet's LCD controller and external frame-buffer interface.

Besides controlling the LCD, Amulet's custom logic has a GUI engine that manages a graphical user interface, offloading more work from the CPU. Programmers can visually design the GUI, using standard web-design tools, then use Amulet's proprietary development tools to automatically convert the HTML into a special language, called MicroHTML, that drives Amulet's GUI engine. In all, Amulet's LCD/GUI macro block requires only 50,000 metal-programmable gates in a CAP device, leaving plenty of room for additional user customization.

More Than a Sea of Gates

Amulet's complex macro demonstrates that CAPs have a remarkable capacity for custom logic. Their equivalent of one million to two million FPGA gates far exceeds Triscend's best chips, which had only about 6,400 FPGA gates. ST's

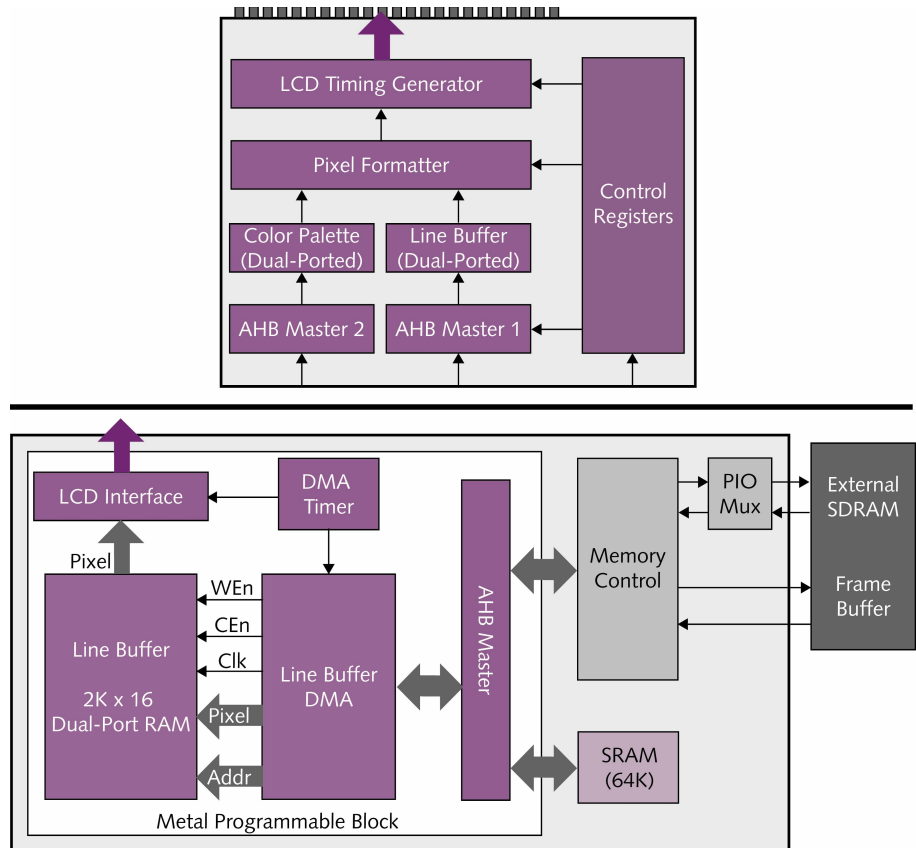


Figure 4. Amulet Technologies' Graphical OS in Silicon technology for Atmel's CAP. Amulet has ported a custom LCD controller and GUI engine to the metal-programmable portion of Atmel's CAP devices. At the top is a block diagram of the LCD controller, which can drive color screens up to 800×600 pixels. At the bottom is a block diagram of the external frame-buffer interface.

STW21000 "GreenField" chip has only about 150,000 FPGA gates. Stretch's programmable Instruction-Set Extension Fabric is more limited in scope than Atmel's programmable metal, and Stretch prefers not to liken the fabric's capacity to FPGA-gate equivalents. On the negative side, Atmel's programmable gates require custom masks during manufacturing and aren't reprogrammable without redoing the masks and respinning the chips.

Another notable feature of Amulet's macro is that it leverages some of the communications and I/O capabilities of CAP technology. Atmel's metal-programmable block is more than a sea of gates. To make complex designs feasible, these chips have six to twelve AMBA Advanced High-Speed Buses (AHB), including six to twelve master controllers and six slave controllers. The CPU uses at least two of these bus masters to manage instruction/data transfers and DMA for peripherals. Up to four masters and four slaves are reserved for the metal-programmable block. The AHB masters aren't bound to individual buses—any master can take control of any bus, as needed. This arrangement eliminates AHB bus contentions.

Furthermore, the programmable block has up to 18 of its own interrupt lines for servicing peripherals that are

Price & Availability

Customizable Atmel Processors are available now. Atmel offers nine different parts—three with ARM7TDMI processor cores and six with ARM926EJ-S cores. Several other features distinguish these parts from each other, including the capacities of their metal-programmable blocks, which range from approximately 250,000 ASIC-equivalent gates to 500,000 gates. Atmel charges a flat fee of \$150,000 to port custom logic implemented in an FPGA to the metal-programmable block, which requires four custom layers in final fabrication. Atmel's fee includes final verification, fabrication, and manufacturing samples. Prices start at \$5 for ARM7-based CAPs in 50,000-unit quantities and at \$13 for ARM9-based CAPs in 100,000-unit quantities. For more information, see:


- www.atmel.com/products/AT91CAP/

implemented in the block, 14 dedicated peripheral-enable lines, up to 90 general-purpose I/O (GPIO) ports, an independent DMA interface, and a multiplexed connection to the on-chip USB controller. The multiplexed USB connection allows designers to add a second USB device in the programmable block. For logic that needs tightly coupled memory, the block has two 16-bit interfaces for dual-ported SRAM.

ARM7-based chips have a total of 8KB of dual-ported SRAM associated with the block. ARM9-based chips have a total of 20KB of dual-ported SRAM and 36KB of single-ported SRAM associated with the block. (Additional SRAM is attached to the AHB elsewhere on chip.)

Clearly, Atmel has invested considerable thought in this design, drawing heavily on the company's experience with ASICs and FPGAs. The main drawbacks are one-time gate-level programmability, longer turnaround times than FPGA implementations, higher unit costs than standard-part microcontrollers, and the industry's general resistance to unfamiliar technology.

Another potential drawback is that each chip's metal-programmable block is fixed in size. A design that doesn't use all the available space will cost the same to buy as a design that uses every available gate. Fortunately, the effect of wasted space on power consumption is negligible—nonexistent (or unrouted) gates don't need power, and they don't leak power, either. Nevertheless, the extra cost of CAP technology (above standard parts) argues in favor of designs that can usefully exploit 250,000 to 500,000 gates of custom logic. Designs needing only a few additional peripherals or a little custom logic might be better served by another solution.

Overall, Atmel's CAP technology offers a viable alternative to existing choices, and the chips are relatively affordable. As time goes by, they will probably look better against the rising costs of ASICs and perhaps worse against the falling costs of FPGAs. But the middle ground isn't a bad place to be. 

To subscribe to Microprocessor Report, phone 480.483.4441 or visit www.MPRonline.com