Reed Electronics Group

MICROPROCESSOR REPORT

www.MPRonline.com

THE INSIDER'S GUIDE TO MICROPROCESSOR HARDWARE

# ARM's Smallest Thumb

## *New Cortex-M0 Is ARM's Tiniest Processor Core for MCUs*

*By Tom R. Halfhill {3/2/09-01}*

The name says it all: Cortex-M0. As in "M-Zero." Unless ARM starts naming its processors with negative numbers, the new Cortex-M0 will always be the smallest member of the growing Cortex-M family. Announced February 23, it's a 32-bit synthesizable processor core for microcontrollers and deeply embedded applications.

The Cortex-M0's minimum usable configuration is a mere 12,000 gates. To put that in perspective: it's about one-third the size of the ARM7TDMI hard macro—itself a small processor core, designed in the mid-1990s when everything was smaller. Even the base configurations of customizable processor cores from ARC International, MIPS Technologies, and Tensilica aren't this tiny. Anything tinier is probably an 8- or 16-bit processor.

Why get small, when other CPUs keep getting bigger? Because ARM is continuing its quest to lure developers away from 8- and 16-bit processors. As ARM reduces the size, cost, and power penalties, a 32-bit design looks better and better. The Cortex-M0 can deliver more performance, larger memory addressing, and upward compatibility with the world's most popular 32-bit embedded-processor architecture. However, to achieve its small size, the Cortex-M0 makes some compromises. It won't run 32-bit ARM software, because it executes mostly 16-bit Thumb and Thumb-2 instructions. Competing vendors, less intent on replacing 8- and 16-bit processors, have elected to preserve 32-bit code compatibility and higher performance.

ARM is pitching the Cortex-M0 for the next generation of small MCUs and deeply embedded SoCs, especially as those chips absorb more analog functions. Mixed-signal designs usually require larger-scale fabrication processes to accommodate the analog elements, so a smaller processor core is desirable. In particular, ARM sees lucrative opportunities in mixed-signal ASICs, ASSPs, sensors, and actuators. Examples include MEMS pressure sensors; shock detectors for hard-disk drives in mobile systems; battery-strength monitors; wireless peripherals; touchpad controllers; GPS receivers; and inertial sensors for airbags and stability-control systems in automobiles.

Other things being equal, a processor with fewer gates uses less dynamic power and leaks less static power, too. When fabricated in a generic 90nm CMOS process, the Cortex-M0 sips only 15 microwatts (0.015mW) of dynamic power per megahertz. That's while running Dhrystone 2.1. The maximum worst-case clock frequency in a 90nm-G process is about 270MHz, so typical dynamic power at top speed is a mere 4.0mW. (Core voltage in this process is 0.9V.)

If developers implement the Cortex-M0 in an ultralow-leakage 0.18-micron process, ARM offers a power-management kit and cell library that enable a special deep-sleep mode. In this slumber, the processor stops its clock and shuts down the main power rail, drawing only nanoamps of current. Yet wakeup is almost instantaneous, because a separate low-power domain preserves the state of the processor's registers. Although 8- and 16-bit processors can use even less power, the Cortex-M0 will be attractive 32-bit bait for developers.

### A Twig On the Family Tree

First, let's put the Cortex-M0 into context. The new processor joins the ARM Cortex-M family below the Cortex-M3

and alongside the Cortex-M1. All are 32-bit synthesizable processor cores. Whereas the Cortex-M3 implements the ARMv7-M ISA, the Cortex-M0 and Cortex-M1 implement a subset of this ISA called ARMv6-M. To conserve memory, ARMv6-M supports ARM's 16-bit Thumb and Thumb-2 instructions, but only a few 32-bit instructions. Therefore, the Cortex-M0 is upward compatible with the Cortex-M3. However, code written for the Cortex-M3 or other ARM processors will not run on the Cortex-M0 without modification.

All Cortex-M processors have three-stage instruction pipelines, without such frills as dynamic branch prediction, speculation, or superscalar execution. To preserve strict deterministic behavior in real-time systems, all are cacheless cores that access program code and data in main memory or tightly coupled memory (TCM). These days, 32-bit processors just don't get much simpler than this.

The first and most powerful member of the Cortex-M family was the Cortex-M3, announced in 2004. It has a Harvard bus architecture (separate I/O buses for instructions and data), a 32-bit fast multiplier, a 32-bit fast divider, and an optional memory-protection unit. Last year, ARM released an enhanced version of this processor with faster clock speeds, configurable debug logic, new power-saving features, and compatibility with third-party fault-tolerance technology. (See *MPR 5/12/08-01*, "Fault Tolerance for Cortex-M3.")

The second processor to join this family was the Cortex-M1 in 2007. It's almost identical to the Cortex-M3 but is optimized for implementation in FPGAs. In fact, it's the first and only processor core that ARM sanctions for deployment in programmable-logic devices. Owing to inherent limitations of programmable logic, the Cortex-M1 is slower than the Cortex-M3. (See *MPR 3/19/07-01*, "ARM Blesses FPGAs.")

Which brings us to the new Cortex-M0. It's similar to the Cortex-M1 but even simpler. It drops the Harvard bus architecture in favor of a von Neumann architecture—a unified I/O bus for instructions and data. Indeed, the Cortex-M0 doesn't even support TCMs. It fetches code and data directly from main memory over its single AMBA AHB-Lite 32-bit interface. (The Cortex-M1 and Cortex-M3 have two AHB-Lite interfaces.)

Only 56 instructions comprise the Cortex-M0 instruction set. All but six are 16 bits long. The instruction set includes 35 original Thumb instructions inherited from the ARM7TDMI, plus 21 Thumb-2 instructions introduced with the ARMv7-M ISA in 2004. (See *MPR 11/29/04-01*, "ARM Debuts Logical V7.") Most Thumb-2 instructions in the Cortex-M0 are intended for system software, not application software. They help provide upward compatibility with the Cortex-M3. Essentially, the Cortex-M0 is a subset of the Cortex-M3, which has the same 56 instructions plus 99 more. Remarkably, the Cortex-M0 has barely half as

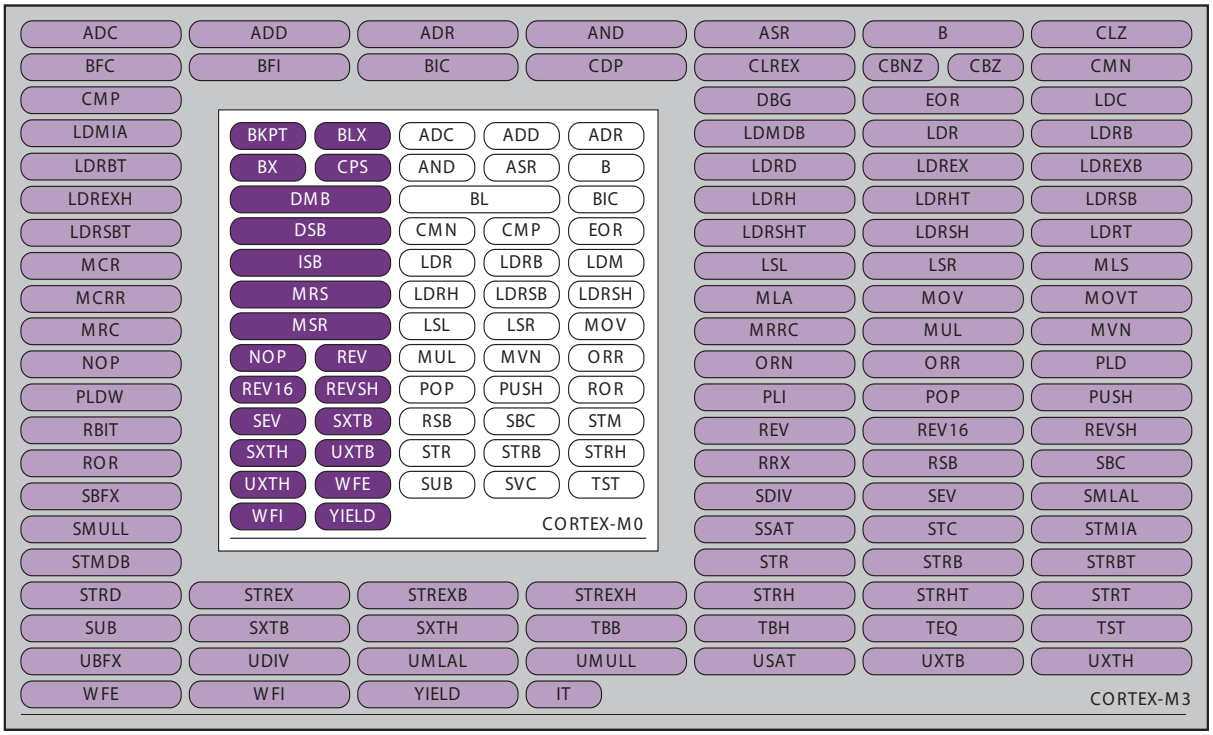


**Figure 1.** ARM Cortex-M0 instruction set. The Cortex-M0 supports the 56 instructions in the central box of this figure, including 35 original Thumb instructions from the ARM7TDMI. These 56 instructions are a subset of the 155 instructions in the Cortex-M3. Therefore, code written for the Cortex-M0 will run without modification on the Cortex-M3, but not vice versa. Developers can write all user code with Thumb instructions—including interrupt handlers, which the ARM7TDMI cannot execute in Thumb mode.

many instructions as an 8-bit 8051 processor. Figure 1 illustrates the Cortex-M0 and Cortex-M3 instruction sets.

### Trimming Features to Save Gates

To save more gates, the Cortex-M0 lacks a fast 32-bit divider and allows developers to choose from two 32-bit multipliers: a larger, fast one that multiplies 32-bit integers in one clock cycle or a smaller, slower one that takes 32 cycles. There's no option for a memory-protection unit. Another missing option is the integrated real-time trace logic offered for the Cortex-M3. Table 1 summarizes the features of all three members of the Cortex-M family plus the synthesizable version of ARM's old standby, the ARM7TDMI.

One thing on which ARM didn't skimp: interrupt control. This feature, vital for many embedded systems, is a weakness of the aging ARM7TDMI-S. The older processor requires an external interrupt controller and responds relatively slowly to interrupts. It also lacks a nonmaskable interrupt (NMI). The Cortex-M family modernizes interrupt handling with a nested vectored interrupt controller (NVIC) closely coupled to the core. The NVIC supports many more interrupts (1 to 32 in the Cortex-M0), multiple levels of interrupt priorities (four in the Cortex-M0), NMIs, and shorter interrupt latencies. The NVIC is configurable, so developers can save gates by implementing only as many interrupts as they need.

Comparing interrupt latencies among different processors is as tricky as comparing their throughput and power consumption, because it depends on what's included in the calculation. For Cortex-M processors, the interrupt latencies in Table 1 include priority detection, nesting the interrupts, copying corruptible registers to the stack, branching to the interrupt-service routine, and starting to execute the first instruction of the routine. The Cortex-M NVIC performs all these functions in hardware. Other vendors may specify lower interrupt latencies in hardware, but their processors may need to execute some of these functions in software, which is slower.

With any core that offers a degree of configurability, the vendor's minimum gate count can be misleading. But ARM promises that the 12,000-gate base configuration of the Cortex-M0 isn't like buying a car without wheels. Those 12,000 gates include an NVIC with one interrupt line, the 32-cycle multiplier, and the AHB-Lite bus. The only thing not included in the gate count is an example SRAM interface (well under 1,000 gates), which developers will probably replace with their own memory interface. Otherwise, the core is fully functional and occupies only 0.04mm$^2$ of silicon when fabricated in a 90nm-G process. ARM estimates dynamic power consumption at 15 microwatts per megahertz. Figure 2 is a block diagram of the processor.

Fully loaded, the Cortex-M0 does grow larger—to about 24,000 gates. This configuration includes the single-cycle multiplier, the maximum number of NVIC interrupts (32), a wakeup interrupt controller, a system timer, and single-wire debug logic with the maximum number of watchpoints and breakpoints. (Table 2 lists all the configuration

| Feature | ARM Cortex-M0 | ARM Cortex-M1 | ARM Cortex-M3 | ARM ARM7TDMI-S |
|---|---|---|---|---|
| Architecture Width | 32 bits | 32 bits | 32 bits | 32 bits |
| ISA | ARMv6-M | ARMv6-M | ARMv7-M | ARMv4T |
| Instr. Lengths | 16 bits | 16 bits | 16 bits | 16 / 32 bits |
| Thumb Instructions | Thumb-1 Thumb-2 | Thumb-1 Thumb-2 | Thumb-1 Thumb-2 | Thumb-1 |
| Pipeline | 3 stages | 3 stages | 3 stages | 3 stages |
| Memory Architecture | von Neumann | Harvard | Harvard | von Neumann |
| L1 Cache | — | — | — | — |
| Tightly Coupled Memory (TCM) | — | 0K–1024K (code) 0K–1024K (data) | 0K–1024K (code) 0K–1024K (data) | Requires external controller |
| 32-Bit Multiplier* | 1 cycle or 32 cycles | 1 cycle or 32 cycles | 1 cycle | 1 cycle |
| 32-Bit Divider | — | — | Yes | — |
| Memory Protection Unit (MPU) | — | — | Optional (8 regions) | — |
| System Interface | 1 x AHB-Lite 32 bits | 2 x AHB-Lite 32 bits | 2 x AHB-Lite 32 bits | 1 x 32 bits |
| Nested Vectored Interrupt Controller | Yes | Yes | Yes | — |
| Ext. Interrupts* | 1–32 | 1–32 | 1–240 | 2 |
| Interrupt Priorities | 4 | 4 | 256 | 2 |
| NMI | Yes | Yes | Yes | — |
| Interrupt Latency | 16 cycles | >20 cycles | 12 cycles | 24–42 cycles |
| Inter-Interrupt Latency | 6 cycles | 9 cycles | 6 cycles | 24 cycles |
| Real-Time Trace | — | — | Optional | Optional |
| Breakpoints* Watchpoints* | 4 or 2 2 or 1 | 4 2 | 8 or 4 2 or 1 | 2 2 |
| Core Freq (Max)* (IC Process) | ~270MHz 90nm G | ~174MHz Virtex-5 | ~270MHz 90nm G | 245MHz 90nm G |
| Core Size* | 12k gates | n/a | 33k–60k gates | 48k gates |
| Dhrystone 2.1 | 0.9Dmips / MHz | 0.8Dmips / MHz | 1.25Dmips / MHz | 0.95Dmips / MHz |
| Power (90nm G) | 0.015mW / MHz | n/a | 0.047mW / MHz | 0.06–0.09mW / MHz |

**Table 1.** Four 32-bit ARM processor cores suitable for microcontrollers and deeply embedded applications. Three-stage pipelines and minimal features make these synthesizable processors some of the lowest-gate-count and most power-stingy cores in the industry. The ARM7TDMI-S is the synthesizable version of ARM's best-selling processor since 1995. The Cortex-M family, announced in 2004, offers better performance, lower power, and superior interrupt handling. (*Configurable feature. n/a: not applicable.)

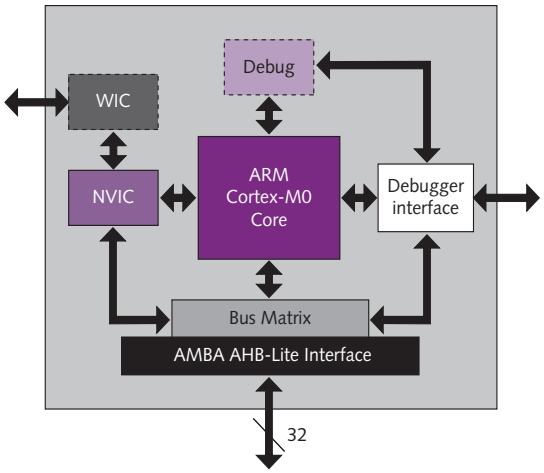


**Figure 2.** ARM Cortex-M0 block diagram. Dotted lines indicate optional features. Most elements are configurable, including the wakeup interrupt controller (WIC), nested vectored interrupt controller (NVIC), and the number of breakpoints and watchpoints supported by the debug logic. There's only one AHB-Lite interface, unlike the dual interfaces on other Cortex-M processors.

options.) When synthesized with the Artisan Advantage library for a 90nm-G process, the fully configured Cortex-M0 will occupy 0.08mm$^2$ of silicon. ARM estimates dynamic power consumption at 17 microwatts per megahertz. Those are still pretty small numbers.

To grasp the breadth of today's microprocessor spectrum, consider that 8,737 fully configured Cortex-M0 cores would fit on the die of Intel's future "Tukwila" Itanium server processor, which will measure 699mm$^2$. The base configuration of the Cortex-M0 could be replicated 17,475 times on a Tukwila die. These two processors could hardly be more different, yet it's likely that someday they will work side by side—one powering a server, the other controlling a disk drive in the server.

| Cortex-M0 Blocks | Descriptions |
|---|---|
| Base Core<br>Slow Multiplier<br>NVIC (1 Interrupt)<br>AHB-Lite Bus (1 x 32 Bits) | Smallest usable configuration = 12k gates, with 32-cycle multiplier, 0.04mm$^2$ area (90nm-G), 15µW / MHz dynamic power. |
| Fast Multiplier | Single-cycle multiplier |
| NVIC<br>(16 Interrupts)<br>(32 Interrupts) | Nested vectored interrupt controller |
| WIC | Wakeup interrupt controller |
| System Timer | Includes extensions for RTOS |
| Single-Wire Debug<br>(1 Watchpoint, 1 Breakpoint)<br>(2 Watchpoints, 4 Breakpoints) | Another option is 5-pin JTAG; no real-time trace available. |
| FULL CONFIGURATION | Largest configuration = 24k gates, 0.08mm$^2$ area (90nm-G), 17µW / MHz dynamic power. |

**Table 2.** ARM Cortex-M0 configuration options. Even when fully configured, this processor is about half the size of an ARM7TDMI-S and about one-third the size of the ARM7TDMI hard macro. Notice that dynamic power consumption for a full configuration is only slighter greater than for the base configuration. However, static power leakage tends to grow proportionally with the gate count and may exceed dynamic power by a factor of 5x–10x in a 90nm-G CMOS process.

## Other Cores Are Larger

More relevant are comparisons with other small embedded-processor cores. At 48,000 gates, the venerable ARM7TDMI-S is two to four times larger than the Cortex-M0. The Cortex-M3 weighs in at 33,000 to 60,000 gates, depending on the configuration. ARC's preconfigured ARC 605 processor is about 30,000 gates with a bus interface, debug logic, and flip-flop register file. The MIPS32 M4K processor is about 37,500 gates, depending on the configuration. The base configuration of Tensilica's Diamond Standard 106Micro core is 20,000 gates but omits some features included with the Cortex-M0, such as a 32-bit multiplier and AHB-Lite bus bridge.

These gate counts, obtained from vendors, assume that developers optimize their synthesis for minimum area, not for maximum clock speed. The difference can be immense. For instance, MIPS says the M4K can vary from 37,500 gates to 116,000 gates, depending on the type of optimization. That's not unusual. Other major factors influencing the size of a synthesizable processor are the configuration options, the cell library used for synthesis, and the target fabrication process. Table 3 compares the Cortex-M0 with similar processor cores from ARC, MIPS, and Tensilica.

The other cores in Table 3 aren't necessarily direct competitors with the Cortex-M0. ARC, MIPS, and Tensilica haven't pared down their instruction sets or taken other drastic measures to reduce gate counts, as ARM has. The ARC 605, MIPS M4K, and Tensilica 106Micro aim for somewhat higher performance while preserving software compatibility with other processors in their families.

In some cases, developers can take matters into their own hands. Many processor cores from ARC, MIPS, and Tensilica have a configurable ISA, which the Cortex-M0 does not. Developers striving to conserve every speck of silicon can modify the ISA to cut corners themselves, sometimes in unusual ways. For instance, it's possible to trim the register file or implement registers in SRAM instead of in flip-flops. ARC goes further than other vendors by allowing developers to tinker with the Verilog model of the core by using the ARChitect processor-configuration tool. Adventurous developers can even modify the Verilog output from ARChitect, although ARC doesn't encourage it and won't support such changes. Several years ago, an ARC customer went to the extreme of *removing* instructions from the base instruction set—which had only about 30 instructions to begin with.

Paradoxically, it's possible to reduce power consumption by *adding* instructions to customizable processors, even while increasing the gate count. Custom instructions optimized for specific applications can execute tasks more quickly and efficiently, allowing the processor to spend more time in a lower power state.

Overall, though, ARM's Cortex-M0 is the smallest 32-bit synthesizable processor core we've seen. More

important, even the minimum 12,000-gate configuration is fully functional, including an interrupt controller and AHB-Lite bus. With a mere 24,000 gates fully loaded, it's hard to beat. Only 8- and 16-bit processor cores will significantly undercut the Cortex-M0.

### Deep-Sleep Mode Saves Power

In some embedded systems, the CPU spends most of its time in a low-power idle state, waiting for an interrupt. Then it wakes up, performs a brief flurry of tasks, and goes back to sleep. The most important attributes for this kind of duty cycle are very low power consumption while sleeping and the ability to awaken quickly after an interrupt. An example is an IEEE 802.15 "Zigbee" node. A wireless Zigbee sensor/controller that governs a thermostat or servo motor might be active less than 1% of the time.

For these applications, developers can implement the Cortex-M0 in ways that support some special power-saving modes. The best example is a state-retention deep-sleep mode. Special flip-flops retain the processor's state information in shadow registers while stopping the clock and shutting off the main power rail. A separate low-power domain preserves the state of the shadow registers during sleep. State retention eliminates the need to copy the registers to nonvolatile external memory before the processor goes to sleep. Of course, it also eliminates the need to restore the registers from external memory on wakeup. Therefore, the processor awakens almost instantly.

To use this deep-sleep mode, developers must synthesize the Cortex-M0 with ARM's Artisan Metro 180ULL (ultralow-leakage) cell library and use ARM's Power Management Kit, another block of physical intellectual property (IP). The target process is 0.18-micron CMOS, and the core voltage can vary from 1.62V to 1.8V. In this implementation, the Cortex-M0's wakeup controller (WIC) acts as the middleman between the power manager and the regular interrupt controller (the NVIC). Figure 3 illustrates this arrangement.

ARM says the power savings can be dramatic. Whereas ordinary standby mode will draw only about 0.10 microamps—itself a very small amount of current—the drain in deep-sleep mode is estimated at less than 50 nanoamps. Consider that a typical Zigbee node could be powered by a CR2032 lithium battery, which is about the size of

### Price & Availability

ARM's Cortex-M0 processor core is available for licensing now. ARM doesn't publicly disclose licensing fees or royalties. Deliverables include the Verilog model of the core and synthesis scripts. ARM's Keil software-development tools are extra. For more information:

- *www.arm.com/products/CPUs/ARM-Cortex-M0.html*

For information about the ARC International ARC 605 processor:

- *www.arc.com/configurablecores/arc600/605.html*

For information about the MIPS Technologies MIPS32 M4K processor:

- *www.mips.com/products/processors/32-64-bit-cores/mips32-m4k/*

For information about the Tensilica Diamond Standard 106Micro processor:

- *www.tensilica.com/products/diamond/di_106micro.htm*

| Feature | ARM Cortex-M0 | ARC ARC 605 | MIPS MIPS32 M4K | Tensilica 106Micro |
|---|---|---|---|---|
| Architecture | ARMv6-M | ARCompact | MIPS32 | Xtensa 7 |
| Arch. Width | 32 bits | 32 bits | 32 bits | 32 bits |
| Configurable ISA | — | Yes | Optional | Preconfigured |
| Instr. Lengths | 16 / 32 bits | 16 / 32 bits | 16 / 32 bits | 16 / 24 bits |
| Pipeline | 3 stages | 5 stages | 5 stages | 5 stages |
| L1 Cache (I / D) | — | — | — | — |
| Instr. RAM | — | 1K–512K | 0–4GB | 0–128K |
| Data RAM | — | 2K–256K | 0–4GB | 0–128K |
| 32-Bit Multiplier | 2 options | 3 options | Configurable | Yes |
| 32-Bit Divider | — | — | Configurable | — |
| Memory Management | — | — | MMU (fixed mapping) | Memory Protection Unit (MPU) |
| System Interface | 1 x AHB-Lite 32 bits | AHB, AXI, or BVCI 32 bits | MIPS SRAM, COP2 32 bits | Xtensa PIF 32 bits |
| Ext. Interrupts | 1–32 | 0–28 | 6 | 15 |
| Interrupt Priorities | 4 | 2 | 1 | 2 |
| NMI | Yes | Yes | Yes | Yes |
| Debug Module | Yes | Yes | Yes | Yes |
| Core Freq (Max) (IC Process) | ~270MHz (90nm G) | 380MHz (90nm G) | 200–414MHz (90nm) | 400MHz (90nm G) |
| Core Size (Base) | 12k gates | 30k gates | 37.5k gates | 20k gates |
| Dhrystone 2.1 | 0.9Dmips / MHz | 1.3Dmips / MHz | 1.6Dmips / MHz | 1.22Dmips / MHz |
| Power (typical) (IC Process) | 0.015mW / MHz (90nm G) | 0.03mW / MHz (90nm LP) | 0.04–0.15mW / MHz (90nm G) | 0.044–0.054mW / MHz (90nm G) |
| Power Efficiency | 60Dmips / mW | 43Dmips / mW | 10–40Dmips / mW | 23–28Dmips / mW |

**Table 3.** Feature comparison of the ARM Cortex-M0, ARC 605, MIPS32 M4K, and Tensilica 106Micro processor cores. In this group, the Cortex-M0's strengths are its small size, low power, versatile interrupt features, and high code density. The other cores have advantages, too, such as deeper instruction pipelines (enabling higher clock speeds) and tightly coupled memories for faster access to code and data. The M4K and 106Micro have simple memory-management units. The M4K is also available as a prehardened core, which can save significant design time. As always, take the vendors' core-size and power-consumption estimates with a large grain of salt. These numbers vary widely, depending on such factors as the core configuration, synthesis optimization, cell library, layout, and fabrication process.

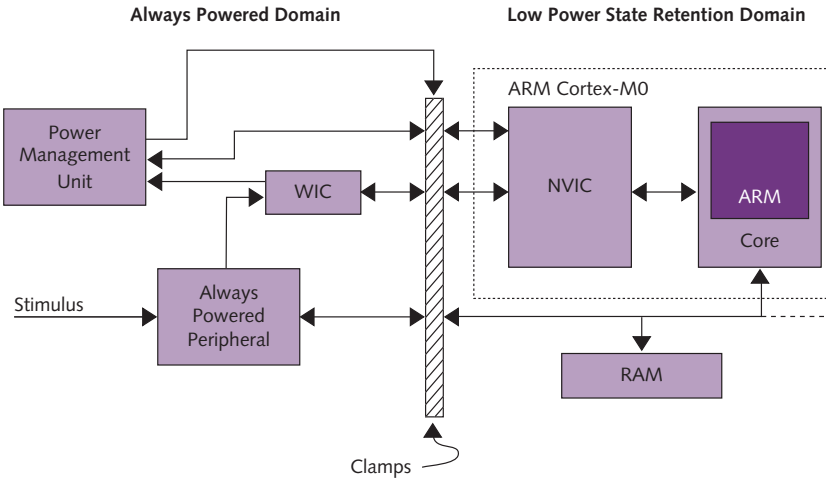


**Figure 3.** Low-power state retention in the Cortex-M0. To enable a low-power deep-sleep mode, developers must use a special 0.18-micron design flow and cell library that create two independent power domains. At left, the ARM power manager and wakeup interrupt controller (WIC) are always powered up, waiting for an interrupt from the powered peripheral. On an interrupt, the WIC signals the Cortex-M0's nested vectored interrupt controller (NVIC) to bring the processor up to speed. There's no need to dump or restore registers to nonvolatile external memory with this implementation, because special flip-flops retain the processor's state in shadow registers during sleep.

a U.S. 25-cent piece and rated at 230 milliamp-hours. (These batteries are commonly found in small electronics products and cost a few dollars.) ARM estimates that the Cortex-M0 would draw so little power while sleeping that a CR2032 battery could last 30 years—if self-discharge or chemical leakage doesn't kill it first. Figure 4 shows a simulated power profile using this implementation.

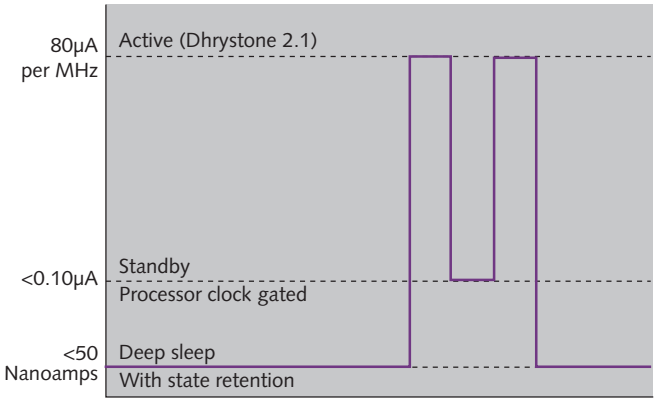


**Figure 4.** Cortex-M0 power profile. For this Dhrystone 2.1 test, ARM simulated a fully configured Cortex-M0 processor synthesized with the Metro 180ULL library. While active, the processor draws 80 microamps per megahertz. (The base configuration of the processor would draw less than 50 microamps per megahertz.) The processor awakened from deep-sleep mode, ran the benchmarks, entered standby mode during a brief lull, resumed full-speed processing, then returned to deep-sleep mode. Unlike a conventional stop-clock mode, deep sleep maintains the state of the processor's registers, allowing instant wakeup. ARM estimates that the static drain during sleep will be less than 50 nanoamps.

Deep-sleep mode uses so little power that a battery might well outlast the system. Developers will appreciate this longevity for systems that are difficult to access and service after installation. However, some systems may not require instant wakeup or may have a duty cycle with different characteristics. For those applications, the conventional stop-clock mode that saves and restores the processor's state in nonvolatile external memory might consume less power.

## 32 Bits—With an Asterisk

ARM offers comprehensive software-development tools for all its processors, and those tools have been extended to support the Cortex-M0. In 2005, ARM acquired Keil, a small company that specializes in development tools for 8- and 16-bit MCUs. Keil's expertise makes its tools a good fit for the Cortex-M0. ARM says the Cortex-M0 will allow developers to use a common tool set and design flow for embedded systems that have multiple processors. In contrast, a system with a 32-bit applications processor and an 8- or 16-bit MCU requires developers to use different tools and flows.

ARM has a point, but there's a catch. The Cortex-M family has been maligned for sacrificing full binary compatibility with other ARM processors. Because Cortex-M processors support mostly the 16-bit Thumb and Thumb-2 instructions, they can't run programs written or compiled with 32-bit instructions. They are upward compatible with other ARM processors, not downward compatible. And, as Figure 1 illustrates, the Cortex-M0 is upward compatible with the Cortex-M3, but not vice versa. For developers moving to ARM from 8- and 16-bit architectures, these limitations are irrelevant—they must rewrite their code anyway. Developers with existing 32-bit ARM code or 16/32-bit Cortex-M3 code may find a transition to the Cortex-M0 inconvenient.

Competing 32-bit processors aren't limited in these ways. Although the processors from ARC, MIPS, and Tensilica listed in Table 3 have subsets of 16-bit instructions for maximizing code density, they also support the full 32-bit (or, in the case of Tensilica, 24-bit) instruction sets as other processors from their vendors. Their binary code is upward and downward compatible, across their product lines.

By supporting mainly 16-bit instructions, Cortex-M processors can be smaller and more power efficient. The Cortex-M0 is the purest expression of that tradeoff. It's not the first 32-bit embedded processor to have a 16-bit instruction set—Hitachi's SuperH architecture is a much older example. But for developers willing to accept the tradeoffs, the Cortex-M0 is the smallest of the small.