

design Offloading CPU Boosts Microcontroller Performance And Cuts Power

FAQs

William Wong
Embedded/Systems/Software Editor

FREQUENTLY ASKED QUESTIONS

ED ONLINE 18448

Why should designers offload a microcontroller's CPU?

Performance and power consumption are the typical reasons for adding offload capability to a microcontroller. Traditionally, any hardware peripheral offloads the CPU. This includes devices like universal synchronous/asynchronous receiver/transmitters (USARTs) and serial peripheral interfaces (SPIs). But these fixed-function devices generally operate in a standalone fashion unless linked by the CPU or other mechanisms such as direct memory access (DMA).

What options are there for offloading a CPU?

There are three options: a co-processor, DMA, and event handling.

Co-processors come in many forms, from programmable processors that may share the CPU's memory space to simple state machines. They usually perform functions more efficiently than a general-purpose CPU. DMA and event systems are simpler co-processors.

DMA controllers are a form of co-processors, but they tend to be dedicated to moving data from one spot to another. They may perform limit data manipulation, though positional data manipulation, such as skipping through arrays, is common.

Event systems typically are configurable state machines that link a peripheral to one or more peripherals or memory areas. For example, a comparator transition might initiate an analog-to-digital converter (ADC) capture that might in turn initiate a DMA transfer of the data, all without CPU intervention. The communication usually uses communication channels that are independent of the CPU to reduce overhead and to allow the event system to operate when the CPU may be in sleep mode.

What are the performance implications?

Offloading the CPU can provide significant performance benefits because of more efficient implementations. A graphics engine can be 100 times faster, while DMA can often move data in one or two clock cycles.

What are the power implications?

Offload support can be more efficient when it comes to power requirements. But significant savings can occur if the CPU can operate in a sleep or lower-speed, reduced-power mode while the necessary offload support remains powered. This may reduce overall power consumption by 40% or more. The ability to reduce the CPU operation will depend on the application.

What are the advantages of co-processors?

Performance. Co-processors can accelerate many applications by a significant amount. Algorithms can be implemented in dedicated hardware. For example, a crypto engine may be 1000

times faster than the same algorithm implemented on an 8-bit CPU.

What are the disadvantages of co-processors?

Cost. A co-processor may add a thousand gates. The impact will depend on the percentage of die size that must be allocated to the co-processor.

What advantages does DMA offer?

DMA usually has a simpler interface and a smaller footprint than a co-processor. DMA implementations are typically most efficient for moving blocks of data.

What disadvantages does DMA offer?

Not all applications move a lot of data. DMA may reduce CPU memory bandwidth or require more dedicated or higher-performance interconnects than a lone CPU.

What are the advantages of events?

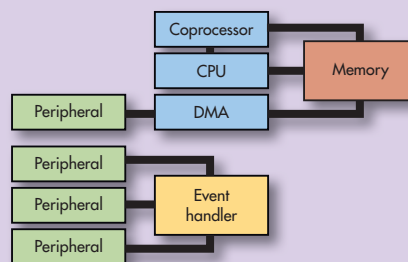
An event system usually has a smaller footprint than DMA. It can handle faults or events without generating interrupts or inducing latency in the system. Also, event handling is predictable.

What are the disadvantages of events?

The cost of an event system can grow exponentially if all devices involved are fully interconnected. On the other hand, not all interconnections may make sense in a system.

What methods are preferred in different classes of microcontrollers?

Traditionally, higher-end, 32-bit microcontrollers featured co-processors and DMA controllers. Fewer 8- and 16-bit devices included these more complex adjuncts. This is changing, though, as even 8-bit platforms gain both.



The event-handler control flow is usually independent of other control units when dealing with peripherals. The CPU, DMA, and coprocessor typically share memory, though some microcontrollers have dedicated memory or cache for some of these subsystems.

product Q&As

What applications are best served by co-processors?

Co-processors address applications that require customization and can be addressed by configurable or programmable co-processors. Fixed-function co-processors such as crypto engines fit with applications that use their functionality, such as secure communication for this example. Yet the range of programmability can vary significantly with co-processors. For instance, crypto engines are used to operate on blocks of data and are more akin to DMA in complexity from a programmer's view, but the operations they perform can be quite complex.

What applications are best served by DMA?

Applications that receive blocks of data at high speeds such as an SPI to a ZigBee transceiver are ideal for DMA support. Typically, the processor is only involved once the entire block is transferred. DMA can address non-contiguous data, but transferring data is its primary objective. Rudimentary processing can sometimes be performed if the DMA engine can deliver information to a processing entity such as moving ADC output to an add/accumulate register.

What applications are best served by event handlers?

Event handlers are ideal for coordinating devices (see the figure). For example, the output of an analog comparator checking for an overload may trigger an immediate motor shutdown. This allows the action to be taken without the CPU running or taking extra time to perform a context switch to handle an interrupt. The interrupt handler can take hundreds of clock cycles while the event handler may take only a few. Even using the event system to handle part of a series of data-acquisition steps can reduce CPU overhead for supporting these types of actions, freeing the CPU for other duties or to operate at reduced power or clock speeds.

Are all three approaches found in one CPU?

Co-processors, DMA, and event handling can be found individually and in combination on a variety of microcontrollers. Even 8-bit platforms can utilize all three at once, though it is more common to find this level of support in 16- or 32-bit platforms.

Redefining System Performance For 8/16-bit Microcontrollers

Many 8- and 16-bit microcontrollers have difficulties in delivering the performance and functionality requirements of today's embedded applications. However, the migration to 32-bit processor cores is not necessarily the ideal solution. An efficient 8-bit CPU combined with advanced peripherals, innovative functions, and extremely low power can deliver the best combination for many modern applications.

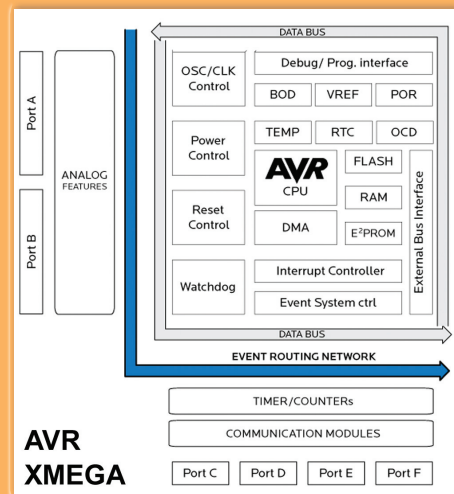
AVR® XMEGA™ IS TAKING THE CHALLENGE TO REDEFINE HOW MUCH YOU CAN ACHIEVE IN THE 8/16-BIT WORLD!

DMA ON ALL PERIPHERALS SAVES MIPS

A four-channel DMA controller enables fast, CPU-independent data transfer that significantly boosts performance. The DMA controller in XMEGA handles transfers between all combinations of data memories and peripherals. Using the DMA, the 32 MIPS delivered by the AVR CPU at 32 MHz can be dedicated to other processing tasks and improve system efficiency.

EVENT SYSTEM AVOIDS LATENCY AND SAVES EVEN MORE MIPS

Like a reflex in the human body, the innovative XMEGA Event System enables inter-peripheral communication without CPU or DMA usage. Events are routed between peripherals through a dedicated network outside the CPU, data bus, and DMA controller. The Event System enables the possibility for a change of state in one peripheral to automatically trigger actions in other peripherals. This is extremely powerful, as it allows for autonomous and predictable control of peripherals without using any interrupts or CPU resources.



MAKE IT PICOPOWER!

With AVR XMEGA, using second-generation picoPower™, battery life is further increased by additional features like true 1.6-V operation and a combined Watchdog Timer- and Brown-Out-Detector current consumption of only 1 µA. True 1.6-V operation means all functions including flash programming, EEPROM write, analog conversions, and internal oscillators are operative. In Power Down mode with RAM retention, current consumption is 100 nA. A Real-Time Clock function using a 32-kHz crystal oscillator consumes only 650 nA.

BUILT FOR SCALABILITY

Based on the solid AVR CPU platform, Atmel now offers one of the largest families of code-compatible devices, ranging from the 1-kbyte, eight-pin tinyAVR® to the 384-kbyte, 100-pin XMEGA. By standardizing on AVR MCUs, customers save investments and shorten time-to-market by reusing development tools, software, and hardware design.

