Atmel's ARM®-based Microcontroller Excels in Real Time

White Paper

2325 Orchard Parkway, San Jose, CA 95131

Rev. 1744A–03/01

Abstract

Architectural features of Atmel's AMR7TMDI™-based AT91 microcontrollers enhance the real-time processing capacity of the core, offering a total system solution that is ideal for the most exacting real-time applications.

We all know the inconvenience of signal breakup on mobile phones, and downloaded video going into freeze frame. These are just two examples of electronic systems being unable to cope with events in real time. End users don't like these situations, won't put up with them, and will select information appliances that do not malfunction in this way. With the market exploding for such appliances – mobile communicators, PDAs, solid-state music players, and a host of Web-based devices – the stakes are extremely high.

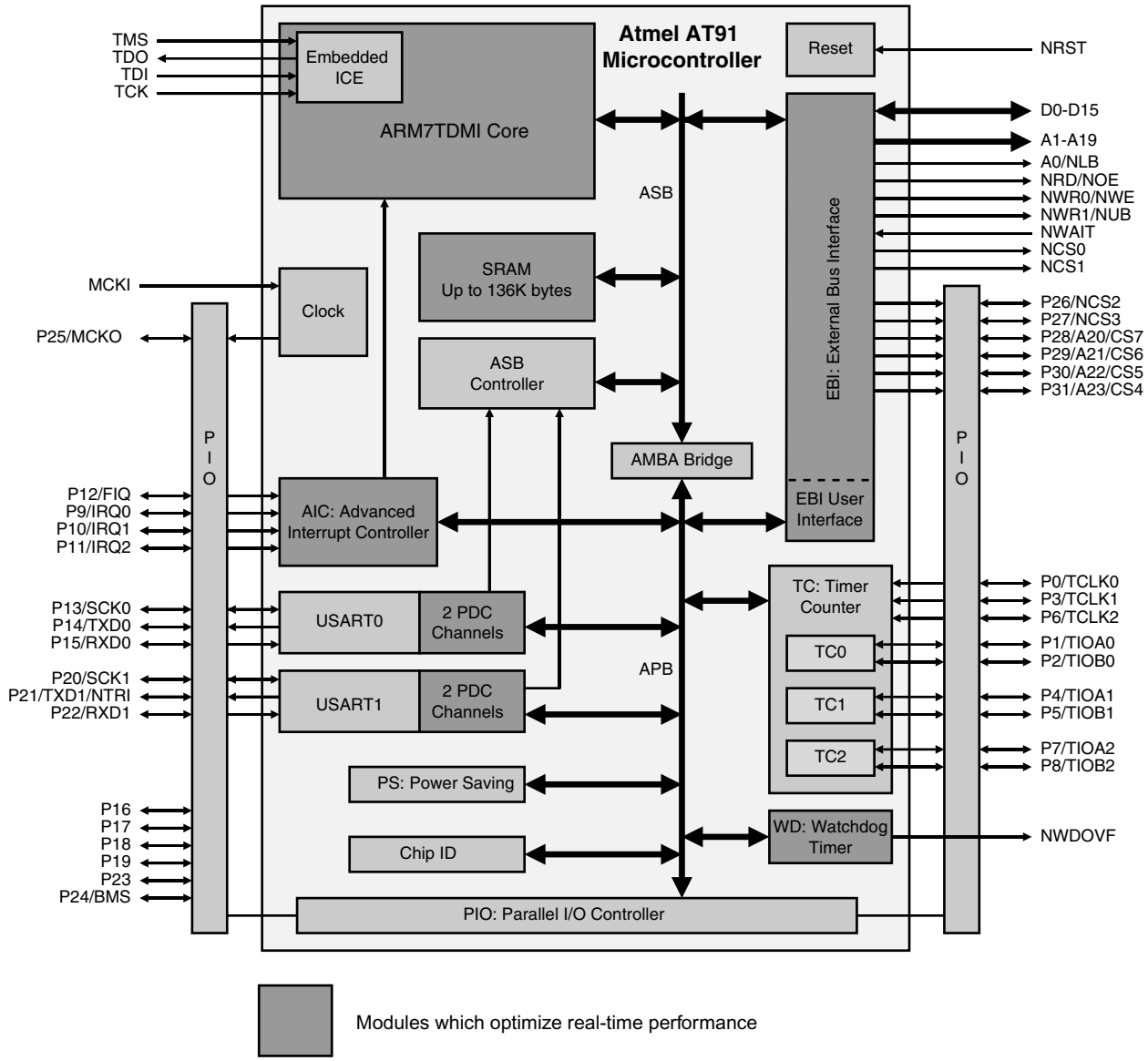
This puts pressure on the designers of the ICs that drive real-time applications to get their act together. Such ICs must be designed, from the outset, to cope with the tight performance and response time constraints imposed by real-time operation. These constraints are many, but the two most important are high instruction and data throughput, and low-latency, deterministic interrupt handling.

This White Paper explains how Atmel has optimized the real-time performance of its AT91 range of ARM7TDMI-based microcontrollers. The AT91 family takes full advantage of the high data and code bandwidth inherent in the ARM core, and adds features such as a large on-chip SRAM, advanced interrupt controller and peripheral data controllers to maximize data throughput, and minimize the response time to interrupts.

AT91 Architecture

The ARM7TDMI core has been designed for high instruction and data throughput. All internal buses are 32 bits wide, and its central feature is a banked 32-bit register file for single-cycle load/store operations. Most data processing instructions fetch the operand(s) from the register file, process them, and store the result in the destination register in a single clock cycle. The 3-stage-pipelined decode/execute unit gives the highest possible instruction bandwidth. Its integral hardware multiplier and barrel shifter enable the ARM core to carry out many tasks that would otherwise invoke the speed penalty of an external DSP unit.

Figure 1. The AT91R40807, highlighting the features that enhance real-time performance



SRAM

Atmel exploits this high instruction and data bandwidth by providing the AT91 series with up to 136K bytes of embedded SRAM. The ARM core's Von Neumann architecture makes this SRAM available for any combination of code and data. Its direct connection to the high-speed 32-bit ASB means that code and data are accessed in a single clock

cycle with no wait states. This enables the AT91 system to take advantage of the full 32-bit ARM code set, and handle data up to 32 bits wide as fast as possible.

External Bus Interface (EBI)

The proprietary external bus interface (EBI) carries this single-clock-cycle-access philosophy to external memories and peripheral devices in the AT91 address space. The EBI gives access to 8- or 16-bit devices in a single clock cycle, with programmable wait states. Its Atmel-patented early read protocol optimizes its performance with certain types of memory, notably fast RAM. In particular it enables compressed 16-bit Thumb instructions to be accessed in a single clock cycle.

Peripheral Data Controller (PDC)

The transfer of data between on- or off-chip memories and serial peripheral devices such as USARTs is optimized by the peripheral data controller (PDC) channels. These enable blocks of data to be transferred, at maximum speed, with no processor intervention. They are tightly integrated into the architecture of each peripheral, enabling a transfer to be triggered directly by the activity of the peripheral. During such data transfers, either the processor is freed to handle other tasks, or it can be put into idle mode to save power. One PDC channel is dedicated to each direction of data transfer to or from each peripheral, giving maximum autonomy and parallelism to these operations.

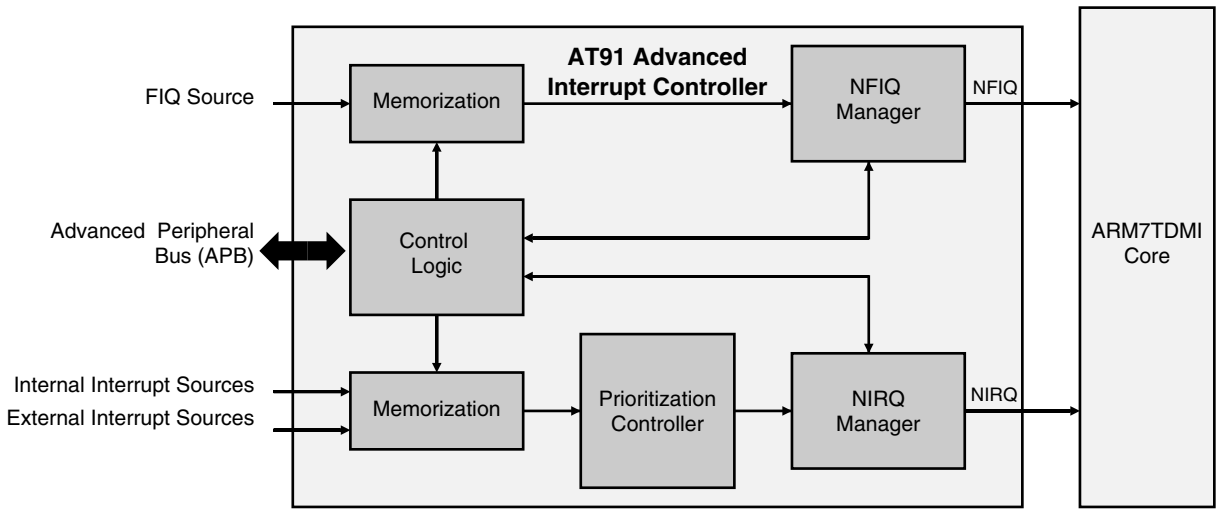
Bit Field Operation

Efficient bit handling is also important in ensuring a rapid data throughput. Many AT91 peripheral registers have the ability to set or reset individual bits in a single operation, without affecting other bits in the register. This avoids the read-modify-write cycle required in most conventional systems for bit manipulation. This saves a significant number of processor cycles, often in frequently used primitives. It is also important in interrupt enabling/disabling, as described below.

Advanced Interrupt Controller (AIC)

Real-time applications, however, demand more than just rapid data processing under ideal conditions. Real-time microcontrollers are constantly being jolted out of their current activities by interrupts. How well, and in particular how fast, they respond to these interrupts is the key to their competitivity in real-time applications. This is where Atmel's AT91 series excels.

Figure 2. AT91 Advanced Interrupt Controller Module



The ARM core has specific operating modes for normal operation and for interrupt and fast interrupt handling. Each mode has its own register bank, which means that no register saves are required when switching to or from an interrupt mode. In particular, each mode can maintain its own stack and subroutine call/returns. This gives a significant improvement in interrupt latency.

Interrupts, however, come from a number of sources, and need to be handled in priority order. Some are maskable, others are not. Atmel's AT91 architecture takes care of these issues in hardware, in the design of its advanced interrupt controller (AIC). The AIC allocates one interrupt source to each on-chip peripheral channel and provides external and software interrupts, as well as an external link for the fast interrupt source. Eight levels of interrupt priority are offered, and interrupts can be safely enabled, disabled, masked or unmasked at any time. The enabling/disabling of interrupts is atomic, thanks to the AT91's ability to set/reset individual bits in a single operation. Other interrupts do not have to be masked during this sequence, which eliminates a major programming overhead.

By setting up the address of the handler for each interrupt source in a vector table and allocating its priority during system initialization, the AIC reduces the number of instructions required to reach the required interrupt handler to only one. Together with the fast interrupt response of the ARM core, this transforms interrupt handling from a liability, as in many other microcontrollers, to an asset. By loading the handlers for high-priority interrupts into on-chip SRAM, the AT91 can provide a deterministic interrupt response time (i.e. a guaranteed response within a specified time interval whatever the circumstances). This is essential for such applications as MP-3 decoding where the output audio stream must never be broken.

Watchdog Timer (WD)

Real-time interrupt driven processing is inherently non-deterministic, so steps must be taken to avoid the situation where the processor is trapped in an endless loop by an unforeseen combination of circumstances. The AT91 provides a watchdog timer (WD) for this purpose. The WD can be programmed to generate an interrupt (or a system reset) if it does not receive, within a certain number of clock cycles, a signal indicating that an on-going operation has terminated or reached a suitable checkpoint. This activity imposes no processor overhead and can prevent system lockup.

Real-time Operating System (RTOS)

The real-time attributes of the AT91 series do not end at the hardware interface. The devices are designed to support a real-time operating system (RTOS) that makes the features described above transparent to the application programmer. The AIC gives the RTOS a real-time view of the pending interrupts, the state of all the interrupt-handling registers and the state of the interrupt signal lines to the ARM core. This enables application modules to take full advantage of the AT91's interrupt-handling features.

An RTOS enables an application to be structured as tasks, and provides services for real-time task creation/deletion, scheduling, initiation/suspension/termination and synchronisation. Tasks can communicate with each other, and see I/O devices via a standard interface. This makes it simpler and quicker to develop and debug real-time applications, and ensures that these applications derive the full benefits of the architectural features that make the AT91 series a winner for real-time systems.