Automotive 32-bit Microcontrollers

A new family of Atmel® AVR® low-power 32-bit microcontrollers provides higher processing performance, improved accuracy, and more power efficiency for automotive applications. These enhancements enable implementation of new product-differentiating features such as advanced control algorithms, voice control, and capacitive touch sensing (see figure 1). The Atmel AVR UC3C 32-bit microcontroller includes innovations such as a peripheral event system, precision clocking, and high-performance peripherals. Integrated features—such as secure Flash memory, hardware-based safety mechanisms, the ability to interface directly with analog sensors, and a configurable software framework—significantly accelerate application development.

32-bit Performance for Advanced Processing

The difference in efficiency between 32- and 8-/16-bit systems is substantial: a generic 32-bit multiple/accumulate requires four multiplications and four additions on a 16-bit processor with additional overhead for data accessing. Thus, a single 32-bit multiplication could require about 20-40 cycles on a 16-bit processor. On a 32-bit UC3C processor this operation requires only a single cycle supported with a 32-bit pipeline for rapid data access.

The availability of an integrated FPU also simplifies application development. The implementation of complex algorithms in particular requires less effort and the wider dynamic range maintains the highest precision. Implementing complex algorithms using 32-bit floating-point instructions not only increases system accuracy and efficiency but can significantly accelerates development as well. A wide variety of applications can benefit from the use of a floating-point unit, including motor control and audio applications.

The UC3C 32-bit microcontroller instruction set is an efficient mix of 16- and 32-bit instructions that allows C compilers to balance performance and code density. The UC3C’s architecture has been optimized for managing real-time...
events common to embedded systems while minimizing processing latency (see figure 2). The UC3C microcontroller includes a wide variety of state-of-the-art peripherals and interfaces—such as CAN and LIN—required by automotive control modules (ECU), while also ensuring reliable operation across the entire automotive temperature range in compliance with the AECQ100 specification.

5V I/O, Power Supply, and Peripherals

The reduction of I/O voltage to lower power consumption and increase stability while process technologies shrink is welcome in many applications such as consumer electronics. However, harsh conditions in the automotive environment still require 5V I/O technology to support the myriad of 5V-based components available on the market. Switching to 3.3V I/O and components is not feasible due to EMC and EMI constraints.

Figure 2: With a high-speed bus matrix and DMA controllers to minimize processing latency, the Atmel AVR UC3C 32-bit microcontroller is designed for high performance.
Atmel AVR UC3C 32-bit automotive-grade microcontrollers can be powered either by a 3.3V or a 5V supply and generally support 5V I/O. This has been achieved by moving to a modified 0.18-micron process technology, which can support higher I/O voltage levels in a reliable and cost-effective manner without any complex and expensive voltage conversion.

In addition to supporting 5V I/O, the UC3C has been designed with a wide range of high-performance peripherals required by automotive applications:

- **ADC**: 16 channels with 12-bit resolution at up to 1.5M samples/second; dual sample and hold capabilities; built-in calibration; internal and external reference voltages
- **DAC**: four outputs (2 x 2 channels) with 12-bit resolution; up to 1M sample/second conversion rate with 1us settling time; flexible conversion range; one continuous or two sample/hold outputs per channel
- **Analog comparator**: four channels with selectable power vs. speed; selectable hysteresis (0.20mV and 50mV); flexible input selections and interrupts; window compare function by combining two comparators
- **Timer/Counter**: multiple clock sources (five internal and three external); rich feature set (counter, capture, up/down, PWM); two input/output signals per channel; global start control for synchronized operation
- **Quadrature decoder**: integrated decoder supports direct motor rotation detection
- **Multiple interfaces**: includes a two-channel, two-wire interface (TWI), master/slave SPI, and full-featured USART that can be used as an SPI or LIN
- **Fully integrated USB**: built-in USB 2.0 transceivers support low (1.5Mbps), full (12Mbps) and on-the-go modes; included in the AVR Software Framework are production-ready drivers for various USB devices (mass storage, HID, CDC, audio), hosts (mass storage, HID, CDC), and combined function devices

Achieving Higher System Throughput with a Peripheral Event System

Managing peripherals by the CPU can become a major system bottleneck, especially as the number of peripherals and their operating frequencies increase. With high sampling rates across multiple channels, interrupt overhead and data processing can consume a large percentage of the processor’s available clock cycles. If the CPU load needs to manage a single SPI port even at a low data rate of 1.2Mbps, this would require 53% of the processor’s capacity. In addition, the interrupt latency increases and introduces jitter.

The AVR UC3C architecture utilizes a new peripheral event system, which allows CPU-independent handling of inter-peripheral signaling through an internal communication fabric that interconnects all peripherals. Rather than triggering an interrupt to tell the CPU to read a peripheral or port (see figure 3), the peripheral instead manages itself by directly transferring data to the SRAM for storage—all without requiring any action by the CPU. From a power perspective, only those blocks that are part of the conversion are active. The CPU is free to execute application code or conserve power in idle mode during the entire event.

![Figure 3. A peripheral event system allows peripherals to manage themselves without involving the CPU. For example, without a peripheral event system (left), a timer can trigger a conversion on one of the ADCs, which then passes the result to DMA where it is stored in memory—all without requiring any action by the CPU (right).](image)

In addition, the peripheral event controller allows a more deterministic response compared to a CPU-based, interrupt-driven event controller, because the latency is fixed to 3 cycles, i.e., 33ns when operating at 66MHz. This enables precise timing of events without jitter, resulting in constant sample rates for ADCs and DACs.

Protecting IP

With manufacturing frequently outsourced to overseas locations, code protection becomes a critical design consideration as software represents more and more of a company’s intellectual property (IP). The following factors become important in IP protection:

- **Protection of program code**: It needs to be ensured that the internal Flash memory can be locked to protect code from being read or copied.
Partial locking of code: System development can be significantly accelerated by making application code available from third-party developers in a locked section of the Flash memory.

Cost of programming devices: Microcontrollers must either be programmed before they are assembled on boards or programmed in-device. Preprogramming creates a challenge in logistics, as devices must be programmed in a trusted facility and transported to the manufacturing site. Programming in-system allows the most recent code to be added during the manufacturing stage or in the field. The AVR UC3C comes with USB drivers that support the Device Firmware Upgrade (DFU) class to allow devices to be programmed over the system’s USB port. As an alternative, a boot loader can be used to allow in-system programming using the CAN interface.

The AVR UC3C 32-bit microcontroller architecture includes the Atmel FlashVault™ code protection technology. FlashVault allows the on-chip Flash to be partially programmed and locked, creating secure on-chip storage for software IP and boot loader operation.

System Safety

Safety is a primary condition for any automotive application. Systems must be able to recover quickly from clock failure. For example, motor control systems must be able to shut systems down upon detection of clock failure to prevent severe damage to the motor or operator. To achieve this, the UC3C devices incorporate a main clock failure detection functionality that immediately switches over to the internal 115kHz RC oscillator in case of malfunction. The system may either continue operation using the backup clock (while triggering an alarm that the primary clock has failed) or perform any necessary shutdown operations to put the system in a fail-safe condition.

Another important safety feature is a window watchdog timer. Microcontrollers commonly support a watchdog timer for verifying that a system is running properly. If the application code is corrupted, the watchdog timer will not be “kicked” and a software reset is triggered to alert the system that it has failed to start-up. A problem with simple watchdog timers is that they are set and enabled by software. If the code that sets the watchdog timer is not being executed, the fail-safe interrupt will not be triggered. A more reliable failure detection mechanism is a windowed watchdog timer (see figure 4) that defines a time window. The watchdog must be kicked within the defined timing interval.

System Development

Today, evaluating a microcontroller architecture requires a complete development environment, including an evaluation kit (see figure 1), a software development environment with compiler and debugger, as well as a comprehensive set of application examples, drivers, and services. Atmel simplifies system development with the AVR Software Framework, which supports a variety of optimized interface drivers,
peripheral firmware, and application code, including extensive motor control algorithms, capacitive touch drivers, advanced digital signal processing algorithms (i.e., FFTs and filters such as band-pass, high-pass, and low-pass), commonly used audio and image codecs such as MP3, speech recognition engines, display drivers, and FAT12/16/32 file systems, to name a few.

For automotive systems, the support with LIN and CAN software stacks, as well as with operating systems such as OSEK, and MCAL layers for the Autosar environment is mandatory. Model-based approaches for the development of automotive applications are becoming more and more popular, and these require additional support of design environments such as MATLAB/Simulink.

Atmel AVR microcontrollers support real-time trace, enabling full system operation visibility. In addition, updates with new features are available every quarter.

The intuitive GUI-based Atmel AVR Studio® is the industry’s most complete development environment for 8- and 32-bit applications, offering full compiler and debugger support for all AVR microcontrollers. Since peripherals are configured using the AVR Software Framework, migration between different AVR devices is truly seamless. The AVR Studio includes the AVR Software Framework with both available free of charge or on a royalty basis.

Atmel also supplies a wide range of hardware-based tools for in-system programming, debugging, and evaluation. The AT32UC3C-EK evaluation kit (see figure 1) provides access to the extensive capabilities of the UC3C architecture with out-of-the-box simplicity. The evaluation kit also supports Atmel QTouch® capabilities.

Automotive Applications

The extended processing capacity of the AVR UC3C paves the way to a wide range of applications for embedded automotive systems:

- **Car Audio Applications**: Many automotive designs have a fragmented functional architecture—i.e., there is a separate chip for USB, one for CAN, another for graphics, one for each audio decoder supported, etc. A microcontroller such as the UC3C—with peripheral integration and extended processing capacity—allows an entire system architecture to be consolidated onto a single chip (see figure 5).

- **Body Controller**: The door control module includes the functions for mirror positioning, window lifter operation, LED backlighting with a dimming function for the indicators, and interfaces for different types of sensors and switches to control the window lifter and the mirror positioning (see figure 6). The interconnection to the vehicle network can be implemented either through a LIN and/or CAN interface.