Building a standard micro architecture

By Trevor Martin
Embedded Europe
(10/07/09, 03:49:00 AM EDT)

Many microcontroller architectures have a long history, with most 8 and 16 bit architectures being designed more than 20 years ago. Over the years these architectures have been re-shaped several times to incorporate new technological developments and keep pace with the industries ever-increasing demands.

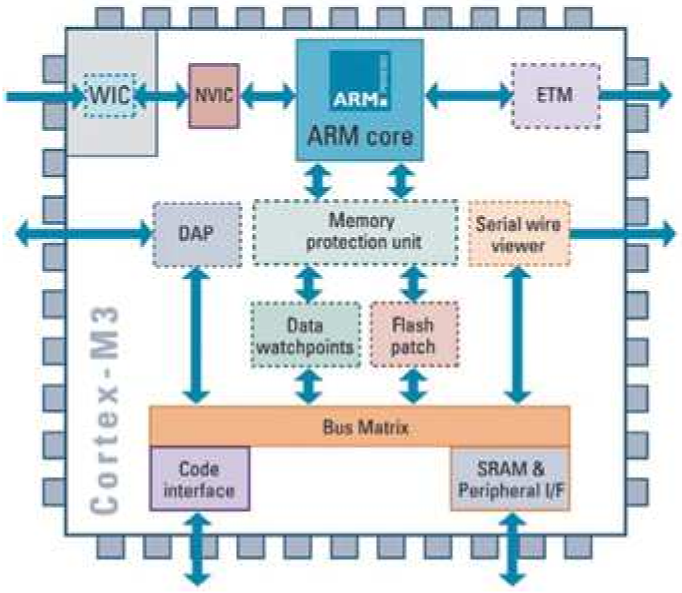
This lack of dominant microcontroller architecture has effectively prevented a common embedded microcontroller platform from developing, unlike the standards we rely on in the desktop PC world.

I have been working in the embedded systems industry for almost two decades now and it is safe to say that there is a genuine sea change under way. Over the last five years many low cost 32-bit microcontrollers have come onto the market.

This cost reduction and easy of use has been a great leveller and has made the traditional 8/16/32-bit distinction irrelevant. At the same time the number of competing 32-bit architectures has dramatically shrunk. In 1992 there were some eleven different architectures, today there are four. This is likely to shrink even further, probably down to two.

One of the big success stories of recent times has of course been ARM. The last few years has seen the widespread adoption of the ARM Cortex-Mx processor family. Many key semiconductor vendors such as Atmel, NXP, ST, TI and Toshiba now offer Cortex-based microcontrollers.

Today there are already several hundred Cortex based microcontrollers available, offering a wide ranging mix of peripherals, processing power and power consumption. Already many companies have seen Cortex as a 'safe bet' and adopted it as their standard microcontroller processor. The ARM Cortex-Mx family has been specifically designed for microcontrollers. It provides the CPU, Interrupt structure, debug and power management units in one standard 'envelope' that is common between all the different manufacturers. Learn it once use it many times, which you'll find attractive if you are anything like me.



To see a bigger version of this graphic click here.

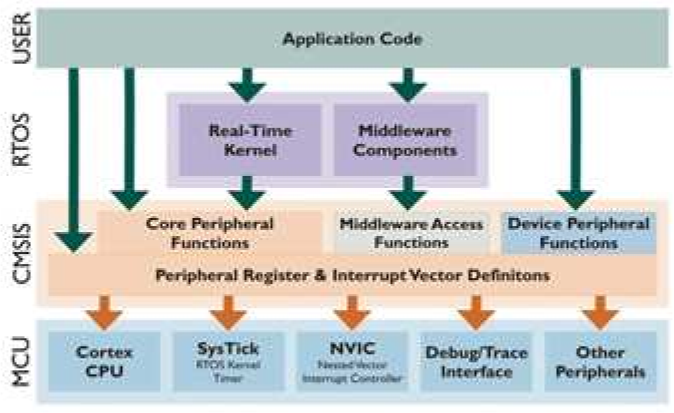
Fig 1: The Cortex-M3 micrcontroller.

For most microcontrollers, no software peripheral interface standards exist. This forces programmers to invent solutions over and over again for the same basic problems. This conspires to slow down development and drive up costs. However this is only part of the problem and the lack of a common interface standard prevents the development of generic software components that are common in the PC world. Having no standard interface also hampers the adoption of more advanced development techniques such as object-orientated programming and other abstract techniques.

Historically, industries use standards to improve product quality and drive down costs. There are many such standards in the electronics industry and in practice they achieve wide acceptance because of the benefits they bring to the user community " i.e. you.

The diversity of embedded CPU architectures has prevented the introduction of an efficient software standard - that is until now. With the arrival of a true multi-vendor microcontroller processor, there is for the first time a real opportunity to build a common interface platform. With this in mind ARM themselves, the Cortex-Mx silicon vendors and the leading tools vendors, have defined the Cortex Microcontroller Software Interface Standard (CMSIS).

CMSIS provides a common approach for interfacing to peripherals, RTOS's and middleware components for all Cortex-based microcontrollers. CMSIS consists of two software layers; the Peripheral Access Layer (CMSIS-PAL) and the Middleware Access Layer (CMSIS-MAL).


USER
Application Code
RTOS
Real-Time Kernel
Middleware Components
CMSIS
Core Peripheral Functions
Middleware Access Functions
Device Peripheral Functions
Peripheral Register & Interrupt Vector Definitons
MCU
Cortex CPU
SysTick
RTOS Kernel Timer
NVIC
Nested Vector Interrupt Controller
Debug/Trace Interface
Other Peripherals

To see a bigger version of this graphic click here.

Fig 2: The Cortex Microcontroller Software Interface Standard " CMSIS

The Peripheral Access Layer contains symbol and address definitions along with helper functions to address the processor core registers and device peripherals. This defines a consistent way to access core peripherals and exception/interrupt vectors as well as providing a standard start-up function. The CMSIS-PAL also defines a device-independent interface for an RTOS kernel and data-trace channels for RTOS kernel-awareness in debuggers. It even supports simple printf style debugging.

The Middleware Access Layer provides common methods to access peripherals for software vendors. The CMSIS-MAL is adapted by each silicon vendor for their specific peripherals. These then provide a common interface for more complex middleware components such as communication stacks (TCP/IP and USB) and graphics libraries. Needless to say, the provision of the CMSIS-MAL vastly speeds up the introduction of middleware support for new devices - good news for anyone who has ever worked on the bleeding edge.

CMSIS has been developed exclusively for small embedded devices. Its complexity has been kept to a minimum. It does not force silicon vendors to provide identical features. Rather it defines common methods to program peripherals. This allows silicon vendors to produce CMSIS-compliant libraries that make best use of their devices peripherals. CMSIS does not prevent direct access to peripheral registers so you can still develop your own driver code, if needed.

The core CMSIS code has been kept a light as possible so as a result, the CMSIS-PAL requires less than 1Kbytes code and just 4 bytes RAM. The CMSIS-MAL replaces a proprietary middleware hardware abstraction layer so it is not really an overhead.

AND THE WINNER IS

Well everyone really. While CMSIS provides benefits to both silicon and middleware vendors, its real benefits are felt by the developer. A standard interface will be well understood by the whole community of silicon vendors, software vendors and developers. This reduces project risk since a CMSIS interface will be verified, certified and widely used. Once you have used CMSIS over several projects, your source code becomes more consistent, easier to understand and simpler to verify. Adopting the Cortex-Mx as a standard processor allows the reuse of tools across many project.

The introduction of CMSIS ups the ante by allowing code to be ported across many projects far more quickly and easily. The common programming techniques introduced by CMSIS reduce both development costs and development time. CMSIS also simplifies long term software maintenance and code reuse. It is also worth noting that when the PC arrived as a standard platform many new software products and businesses opportunities were created. Embedded microcontrollers could well use such a platform.

Trevor Martin CEng (TMartin@hitex.co.uk) is Technical Specialist at Hitex UK Ltd.

The full CMSIS specification and software libraries can be downloaded here.

Download
the
September
issue

This article appeared in September 2009 issue of Embedded Systems Design (Europe) - ESD - to an electronic version of the whole issue is available for downlaod.