ECE

09/07 November 2007 B53585

Information & Know-how for Embedded Engineers

The Complementary Magazine to

www.Embedded-Control-Europe.com

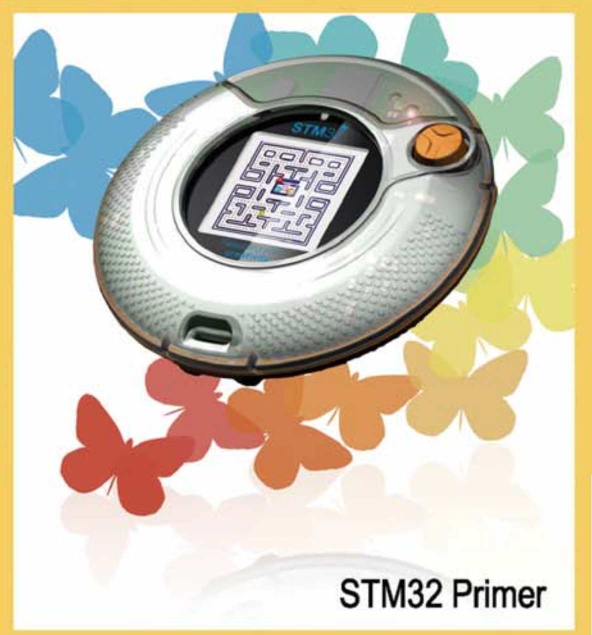
STM32

SPECIAL FEATURES:

- **Microcontrollers & DSPs**
- **Software Development**
- **Embedded Connectivity**

COVER STORY:

ARM Cortex-M3 core-based MCUs with ultra-low-power standby



STM32 Primer

SEMICONDUCTOR
CENTRAL
LPC2101F
691.11
ZPG060
2-Y
OP27G
#0548
P3207
AD7799
BRUZ
#0543
LINX
FAIRCHILD SEMICONDUCTOR
freescale
National Semiconductor
Panasonic
ember
TOSHIBA
Infineon
NEC
intel
MAXIM
DIODES
IXYS
Jennic
CIRRUS LOGIC
NXP
TEXAS INSTRUMENTS
International IOR Rectifier
ALTERA
XILINX
ANALOG DEVICES
MICROCHIP
ZETEX
Atmel
SHARP
intersil
OSRAM Opto Semiconductors
ZiLOG
SILICON LABS
Micro

Dear Readers,

One special feature of this issue of ECE Magazine is about Software Development and of course Open Source plays a major role in the arena.

23 Common Misconceptions about Open Source Software, is the headline of an article which summarizes some common misconceptions about Open Source software and explains the background. This includes important topics like licensing, economics, scalability, community, warranty, support and – last, but not least – costs. While Open Source software is considered as "free" it is not necessarily free of costrs. As with proprietary software, there is a cost to the production. At this point, there is no difference to traditional development processes. It is true that you can download many Open Source software packages for free, but there will be production cost to newly developed parts of the software. Don't expect Open Source developers to work for you gratis. You can just expect that when they already have some part of the software ready, they will possibly ship it under an Open Source license. You can just hire Open Source developers to work on a special project to complete certain features. Technically, there is no difference to the traditional/proprietary software development process since there is a requirements definition, analysis, development/ implementation, testing, deployment, even warranty as stated in the contract. ***Read more in the article starting on page 28.***

A new approach to combining the Open Source and proprietary software models has just been announced by QNX. Like many commercial software vendors, QNX does not want to relinquish all control over its intellectual property and give it away for free. Thus, the company has introduced a new software model that integrates Open Source and proprietary software products in new ways. Building on its Eclipse experience, QNX has started to publish source code for key parts of its runtime products, including the QNX Neutrino microkernel, and will develop those products in the open, for anyone to follow. Non-commercial development licenses for the full version of the QNX development suite, which includes the QNX Momentics development tools and the QNX Neutrino RTOS, are available for free. Partner licenses are also available at no charge for anyone looking to add their products to the QNX ecosystem. As a result, anyone interested in QNX technology can now cooperate on development for the benefit of the community as a whole. At the same time, by publishing its QNX Neutrino RTOS source code, QNX is inviting others to take QNX technology down new open or commercial development paths. Traditional open source communities are, in some sense, open to anyone who wants to follow community rules of behaviour and licensing. This QNX community is very similar, but the laws of intellectual property — and the limitations that QNX places on the use of its copyrighted and patented software products — gives this community more of a commercial feel. This community consists only of QNX licensees. That isn't open source, but it is a realistic modification of open source rules to create an open development community for QNX software, which is used in commercial products worldwide. Outside of the community of QNX licensees, QNX proprietary software is published but it isn't open. ***Read more in the article artsting on page 24.***

Yours sincerely
Jürgen Hübner
(Editor)

Cover Photo
STMicroelectronics

ARM Cortex-M3 core-based MCUs with ultra-low-power standby PAGE 6

This article describes how the STM32 ARM Cortex-M3 core-based microcontrollers provide low power modes and features that mitigate the impacts of leakage on battery powered applications, where static current may be a major contributor to consumption.

Advances in distributed video processing technology PAGE 12

Surveillance cameras are nowadays widely used. With modern 32-bit RISC/DSP architectures, such digital network cameras can almost be built with only one chip.

Common misconceptions about open source software PAGE 28

Open source software is surrounded by many misconceptions relating to issues such as cost, licensing, economics, scalability, community, warranty, support, quality, free software, distribution, compatibility, security, source code, profit, business strategy, volunteers, control, copyright and freedom.Many common misconceptions about open source software are identified below. All are basically wrong, for reasons which are explained.

Affordable embedded security with cryptographic memories PAGE 34

This article looks at security in embedded systems, reviews the trends characterised by trade-offs between security and cost, and introduces cryptographic memories, an innovative technology for embedded security at low cost.

Embedded development for the rest of us PAGE 38

One of the first platforms to support .NET Micro Framework is the Digi Connect ME. The Connect ME is an embedded serial-to-Ethernet module powered by an ARM7TDMI processor running at 55MHz.

ARM Cortex-M3 core-based MCUs with ultra-low-power standby

By Jean-Michel Gril-maffre, STMicroelectronics

This article describes how the STM32 ARM Cortex-M3 core-based microcontrollers provide low power modes and features that mitigate the impacts of leakage on battery powered applications, where static current may be a major contributor to consumption.

■ Requirements for increased computing power and more integrated functions are driving a growing number of applications from 16-bit to 32-bit microcontrollers. This is true for battery-powered applications, which benefit from the lower voltage supply, as well as the high performance and small die size achieved by 32-bit devices based on advanced CMOS processes. However, deep submicron technologies also have an important drawback: their much higher leakage is a major issue, especially for the limited power resources of a battery-powered application. To enable migration, new 32-bit microcontrollers, including general-purpose devices, must provide very efficient ultra-low-power modes for long term standby. The leakage can be defined as the remaining continuous current in a CMOS gate in static state (no switching activity). It has several root causes, and increases with each new technology shrink. Its three main contributors are sub-threshold, gate, and junction tunneling leakage.

Sub-threshold leakage is linked to the threshold voltage diminution that is required with the decreasing voltages used in each new technology. Gate leakage is induced by the scaling of the gate oxide thickness that is needed to reduce the “short channel” effect. Junction tunneling leakage is induced by the electric field across a reverse biased p-n junction (tunneling of electrons). Leakage increases as temperature rises mainly due to the exponential increase of the sub-threshold leakage over temperature. Without any switching activity, the quiescent current of a 32-bit microcontroller using an advanced process can still be limited to a few µA at ambient temperature. However this leakage will rise with temperature and can even exceed 1 mA at 125°C. For this reason, it is very important to take into account the leakage at the maximum application temperature.

Even though several techniques exist to limit the leakage of a digital library (increase poly length above minimum allowed by the technology, increase gate oxide thickness on transistors), such modifications impact the propagation time in the digital cells. Using such a library in the entire core logic would prevent the device from achieving high performance in run mode. The added dilemma for 32-bit devices is that, from a structural point of view, the main contributors to leakage current in a microcontroller are digital logic and memories. In addition to the increasing leakage due to technology shrink, both the digital gate count and average memory size have increased dramatically in subsequent generations of 8-bit, 16-bit, and new 32-bit microcontrollers. As a result, leakage is a major problem for all general-purpose microcontrollers using the latest semiconductor technology, and is a particular consideration for battery-powered applications with their limited power resources. Static current consumption becomes the main contributor to average current as soon as the average runtime becomes very low compared to the standby time. Given the energy level provided by a battery, a quick estimate of the application lifetime (not including non-linearity of the battery capacitance described by the Peukert law) is:

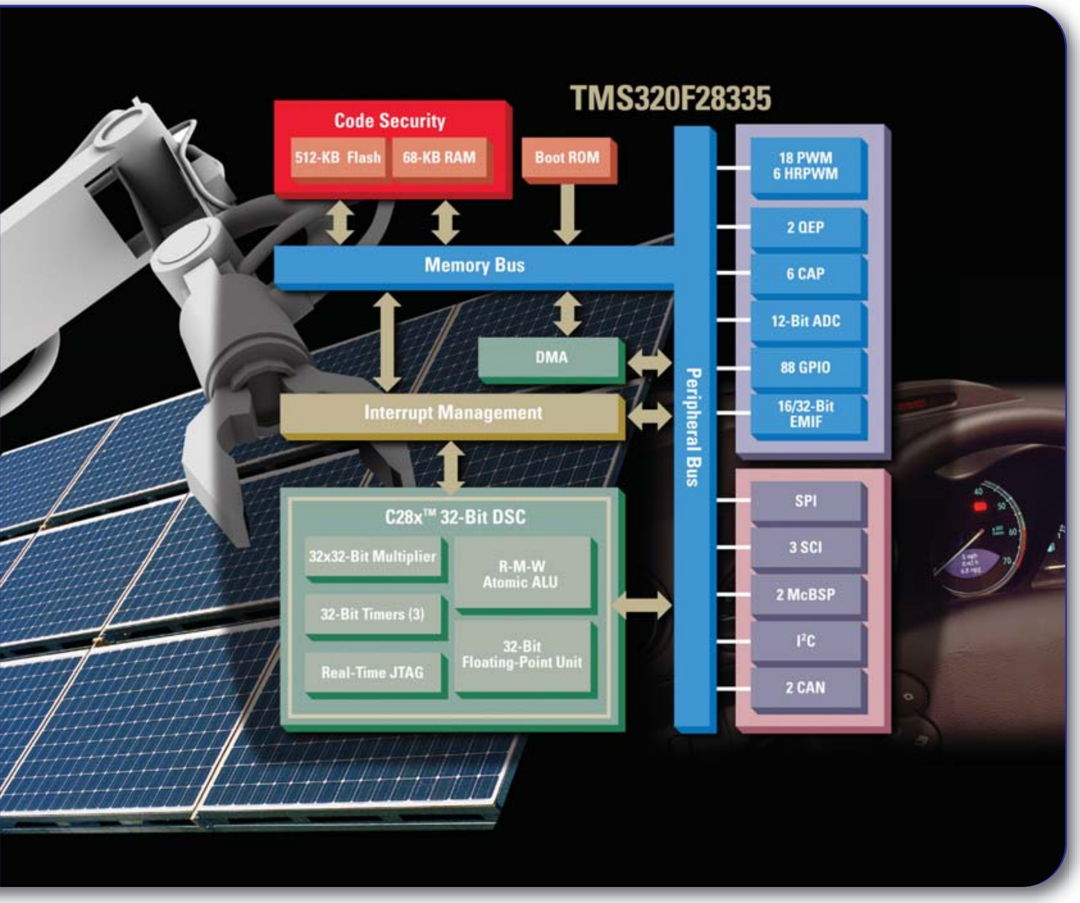
$$Tapp = \frac{Eb}{Irun - (Irun - Istdby) * Trs}$$

Irun: *run current (mA)*
Istdby: *standby current (mA)*
Eb: *Battery capacity (mA.H)*
Trs: *relative time spend in standby mode from 0 to 1*

For example, if the specific ultra-low-power standby mode was not available on the STM32 128Kbyte flash microcontroller, the average current could be significantly impacted: typical run current at 72MHz with all peripherals enabled is only 36mA (0.5mA/MHz) thanks to the ARM Cortex-M3 architecture and low power design techniques. However, due to the use of advanced processes, the leakage current starts to be significant at 55°C. Thanks to a very low power voltage supervisor and regulator, static current is still limited to 50µA at 55°C. This is negligible compared to the run consumption. However if the application runs only one minute a day, the static current becomes the

World's First Floating Point Digital Signal Controllers

Enable Greener Industrial Applications



Applications

- AC drives and servos
- Robotics & computer numerical control machines
- Industrial Digital Power Supplies
- Alternative energies
- Driver assistance radar systems (blind spot detection, ACC)

Features

- 150-MHz / 300-MFLOPS performance
- Up to 512-KB on-chip flash
- Up to 68-KB RAM
- 12-bit ADC @ 12.5 MSPS
- 6-channel DMA
- - 40°C to + 125°C temperature range
- Prices starting at $13.30 for 1KU

Benefits

- 50% average performance boost over previous DSC's
- Simpler software development
- Greater precision
- Greater dynamics
- Software compatible with fixed-point C28xTM controllers

New TMS320F2833x Controllers Boost Performance up to 200% and Reduce Development Time with Floating Point.

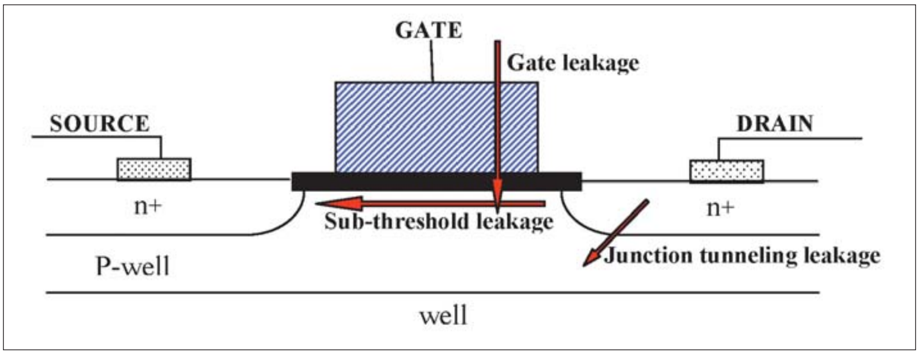
The break-through series of TMS320F2833x controllers target applications that require the faster code development and performance of a floating-point processor with the integration of an advanced controller. By using F2833x controllers, inverters will more efficiently convert solar energy from photovoltaic panels, variable-speed AC drives will operate more efficiently, servo motor controllers will have greater precision and driver assistance radar systems will go for better detection. The new F2833x floating-point controllers will increase performance by an average of 50 percent over the previous leading digital signal controllers while operating at the same 150-MHz clock rate. Some algorithms, such as Fast Fourier Transform (FFT) will see a 200 percent improvement over an equivalent 32-bit fixed-point implementation.

Texas Instruments
Technology for Innovators
TMS320C2000™ Digital Signal Controllers
Embedded Controllers

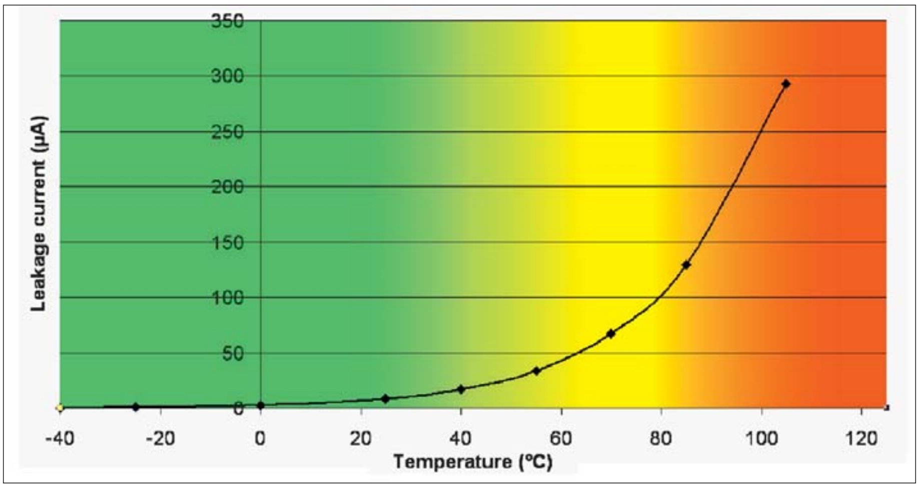
Learn more about TI's NEW Floating-Point Digital Signal Controllers, register for advance product updates and training at www.ti.com/floatingpointDSCe

1860A0

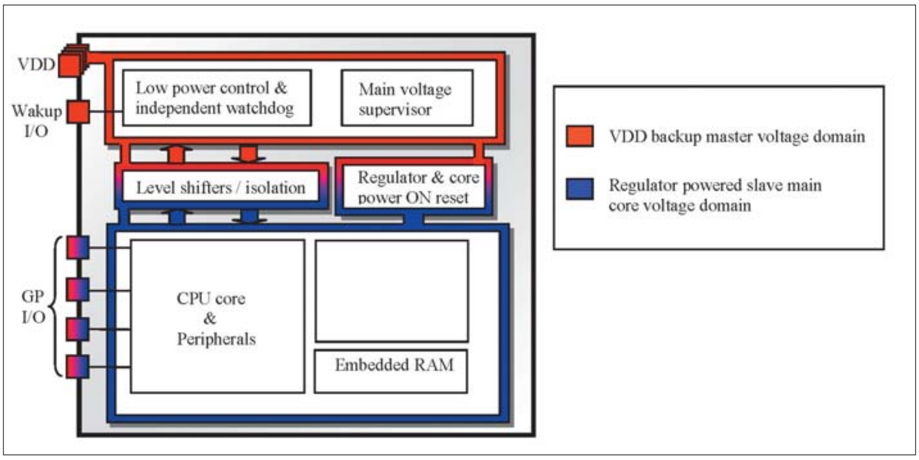
Technology for Innovators™ Texas Instruments



Leakage currents in a CMOS transistor



Leakage increase with temperature



Backup and core voltage implementation

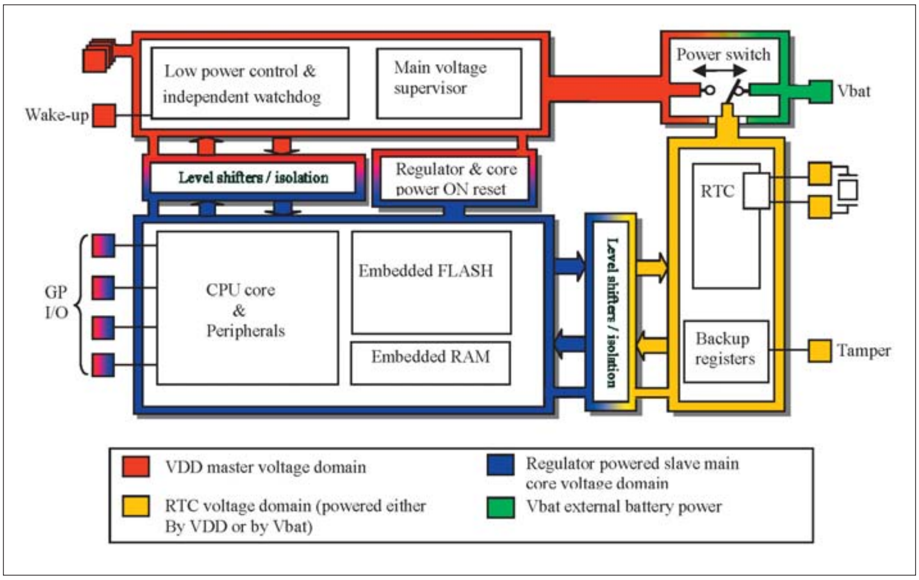
main contributor to consumption (64%). To address this problem, the designers of the STM32 went to great lengths to enable a true low-power standby by implementing an embedded regulator, independent voltage domains, and integrated power switches at the architecture level. This implementation enables low-power modes that can optimise battery life depending on the application.

The total consumption of a microcontroller is the addition of dynamic power (switching activity of CMOS gates) and static current (leakage and static analog consumption). Stopping the product clocks, thereby eliminating all dynamic consumption is obviously not a sufficient low-power standby for a battery-powered application where static current can be a main contributor to consumption. Even decreasing the core voltage when clocks are stopped is of little help in providing an efficient standby mode. To achieve ultra-low-power standby mode, most of the core logic (and memories) must be powered off. To do this, two voltage domains can be implemented at device level, which can be powered by the internal regulator: a small "always on" voltage domain for low-power control, and a "main core" voltage domain with all other functions powered through a switch in order to shut it down in standby mode. As a result, the main core voltage domain can be focussed on processing performance as the leakage (static current) design constraints are mainly important in the "always on" voltage domain.

However, in this implementation the internal regulator must be kept "on" in standby mode, implying a significant quiescent current. For this reason, it is better to stop the embedded regulator in order to reach an ultra-low standby supply current. The STM32 follows this type of dual-domain implementation with VDD backup master voltage domain and slave main core voltage domain. VDD backup master voltage domain is based on thick oxide high voltage transistors focusing on very low static current. With the high voltage transistors, it is directly powered by the main VDD. It includes low-power mode control and a very low-power watchdog, associated low-power RC oscillator and an optimised gate count. Slave main core voltage domain includes all other functions (CPU core, most peripherals, and memories) kept at lower voltage, focusing on high performance and low dynamic power. This implementation allows the STM32F103 to offer a safe, very low-power standby mode with only a 2µA typical current at 3.3V. The remaining 2µA current is the consumption of the accurate voltage supervisor that monitors the main supply voltage to ensure that the standby mode is as reliable as the run mode. Since leakage can be kept very low, increase of this standby current with temperature is very limited: 2.4µA at 85°C 3.3V for a typical device.

Dynamic functions can also be implemented in the master voltage domain. For example, the STM32 includes an independent ultra-low-power watchdog that is available in standby mode with a total added consumption (dedicated RC oscillator and watchdog digital consumption) of only 1µA at 3.3V for a typical device. This feature can prevent application failure in case of an unexpected entry in standby mode.

Separating the voltage domains inside the microcontroller silicon implies many specific design constraints. Full wake-up logic and analog must be implemented in the backup voltage domain making it difficult to offer a large number of possible wake-up sources. Isolation between voltage domains during power down must be implemented (all signals coming from core voltage are floating). The specific sequence for clock source stops and voltage power down/power on must be robust. The main core voltage logic needs a dedicated reset for example. Timing constraints between volt-



Simplified schematic of STM32 voltage domains implementation

age domains must be taken into account specifically because both domains are nearly independent for voltage and process, but not for temperature. This implies that more cases have to be checked during timing analysis (backup domain with worst voltage and process and main core voltage with best process best voltage for example).

Some security features like watchdog function must be implemented in the backup domain in order to protect the application from unexpected standby mode entry. Keeping the ratio of useful I/Os versus total number of I/Os is also required to offer the performance of a 32-bit product in small packages. On the STM32, the main core voltage regulator does not need external decoupling capacitors. This is why no extra power pin is lost on the package because of this dual power implementation. However, in exchange for this silicon design complexity, the STM32 gains a true ultra-low-power standby that will help application developers optimise battery consumption in their applications. As a result of the dual power domain implementation, STM32 provides two different low power modes: stop mode and standby mode. Both function with the voltage supervisor "on" to protect the application in case of a voltage drop. In stop mode, the low-power regulator is kept on but clocks are stopped. It provides very fast restart time on internal RC (<10µS) and retains the software context. Typical current at ambient temperature is 15µA (3.3V). However this mode does not mitigate the problem of leakage, which increases exponentially with temperature. In standby mode the regulator is off in order to provide a 2µA current at ambient temperature (3.3V) and very little increase with temperature increase (2.4µA at 85°C for a typical device). However, restart from standby implies that software content is lost: RAM, core and most peripheral register contents are lost. Restart from standby is nearly equivalent to a restart from reset for the software.

Choosing the best mode for an application can have a large impact on battery life. Here are some basic tips to consider when selecting a mode: 1. Check if the microcontroller state in standby is compatible with application requirements (for example: I/Os standby state, wake-up sources). 2. Consider the impact on battery life of the "worst case" temperature conditions under which application functionality must be guaranteed. 3. Check what the restart from standby time is, and if it is fast enough for the application restart time requirements. 4. Check if there is a saving in energy consumption in standby compared to stop mode. Between two events, is the standby consumption plus the restart from standby consumption less than the consumption in stop mode.

These questions are application-dependent. Estimating the restart time from standby mode includes the time from wake up to reset vector fetch, which depends on hardware (regulator startup time, clock source startup time around 40µS in STM32) and the time needed by the software to restore the application context. Typically the software must check the wake-up source(s), recover context information from backup registers and re-configure the microcontroller functions used by the application. Because of this software- dependent restart from standby, the energy lost during this wake-up phase is also application-dependent. One practical way to estimate this energy loss is to

produce a given amount of wake-ups in a time frame (software going back in standby mode just after the wake-up) and compare the average current consumption when no wake-up is generated.

In order to optimise the restart time from standby mode, the developer must not forget to optimise the initialisation phase added by the compiler and reduce it as much as possible (RAM initialisation should be removed for example). The real time clock feature is a common requirement for battery- powered applications. Moreover, core voltage shut off implies losing the complete program context and is nearly equivalent to a product restart from reset. Implementing a backup register bank for application restart allows the recovery of minimum context required for program execution.

Integrating these functions directly inside the microcontroller can be done in a backup domain. However, the RTC function is typically supposed to be available over an extended period of time (years) while the main application, even if battery-powered, is often based on a rechargeable battery. Creating a third power domain for the RTC and offering a dedicated pin for its power supply allows the use of a small coin cell dedicated only to this function, while the main application is supplied by another main supply source. This way the coin cell power is only used by the RTC and associated oscillator, and not by the other functions, such as the voltage supervisor which is still available in standby mode. However, this implementation is not optimal as the coin cell is always used to provide the power to the RTC and backup registers even when the main power supply is available. A smart alternative that is implemented in the STM32 is to extend RTC battery life by adding a power switch to provide current to the RTC and backup registers from the main supply when it is available and from the battery when the main power is not available. The switch command is provided by the main voltage supervisor with a specific latching mechanism. When the voltage drops below the VDD low threshold, the switch changes the RTC and backup registers power source to external VBAT power. If VDD rises above the VDD high threshold, the switch automatically selects VDD as the power source for this dedicated voltage domain.

One additional advantage with this implementation is that extra dynamic power consumption resulting from software read/write access to this specific voltage domain (through level shifters) never implies extra consumption on the coin cell. In run mode current is always taken from the main supply. Thus, coin cell minimum battery life can be directly calculated based on RTC consumption and the coin cell energy. On STM32 with a typical RTC current of 1.4μA (ambient temperature 3.3V) the minimum battery lifetime when using a CR2032 battery is close to 20 years. However, if the main power is present most of the time, the life can be much longer and a coin cell with smaller capacity can be used. The implemented RTC and backup registers are, of course, available in standby mode. Thus, the RTC can be the source of the wake-up from standby and some key values can be saved in the backup registers before entering standby mode. This implementation significantly increases the complexity of the microcontroller design. It requires more complex isolation between voltage domains; robust power switch design, correctly adjusted to expected consumption (internal RTC voltage domain is not present on I/Os to avoid reducing the number of general purpose I/Os available in small packages, so no decoupling capacitor can be added). It also requires consideration of different startup scenarios without added static consumption on Vbat. For example Vbat rising when VDD is not present must not lead to an unexpected state (there must be no consumption in this state because the coin cell may be soldered to the application during production phase and consumption would cause an unnecessary depletion of its energy level). The RTC voltage domain must be designed to tolerate a significant voltage drop below the VDD minimum threshold before switching on Vbat.

In spite of some new application considerations linked to context loss in a standby state, ultra-low-power standby and multi-voltage power architectures like that of the STM32 can be effective solutions that allow an application to function in a high-performance run mode, while mitigating the impact of static consumption in standby mode. In addition, thoughtful integration of standalone functions, like the RTC in the STM32, can enable fast and efficient development of battery-powered applications and optimum use of application power supplies. ■

For more information please go to www.st.com/stm32

Product News

■ TI expands C64x DSP families

Texas Instruments announces the availability of the TMS320C6452 digital signal processor designed to optimize price and performance for process intensive multi-channel infrastructure and medical imaging systems. Simultaneously, TI announces the TMS320C6455 DSP at 1.2-GHz, the world's fastest single core DSP. The new code compatible 900 MHz C6452 DSP allows designers to quickly and easily migrate their designs from the widely deployed C641x based products.

News ID 819

■ Keil: evaluation board for STM Cortex-M3 MCUs

Keil announces the MCBSTM32 Evaluation Board, which provides easy access to all functions of the STM32 microcontroller family from ST Microelectronics. The STM32 family is based on the Cortex-M3 core from ARM which is specifically designed for embedded applications. The Keil board provides interfaces to most on-chip peripherals of the STRM32F103 including UART, CAN, USB, and SPI with SD memory card connector.

News ID 706

■ Freescale: ColdFire MCUs for low-end 32-bit systems

Freescale Semiconductor has added 10 highly integrated microcontrollers to the company's 32-bit ColdFire portfolio. These latest ColdFire MCU additions provide developers with a broad range of cost-effective, low-power connectivity options, coupled with comprehensive software tools. The expanded ColdFire MCU lineup includes extensions to the MCF5223x Ethernet MCU family, the MCF5221x USB MCU family and the MCF51QE ultra-low-power MCU family.

News ID 720

■ STM: 3rd party development kits for STM32 MCUs

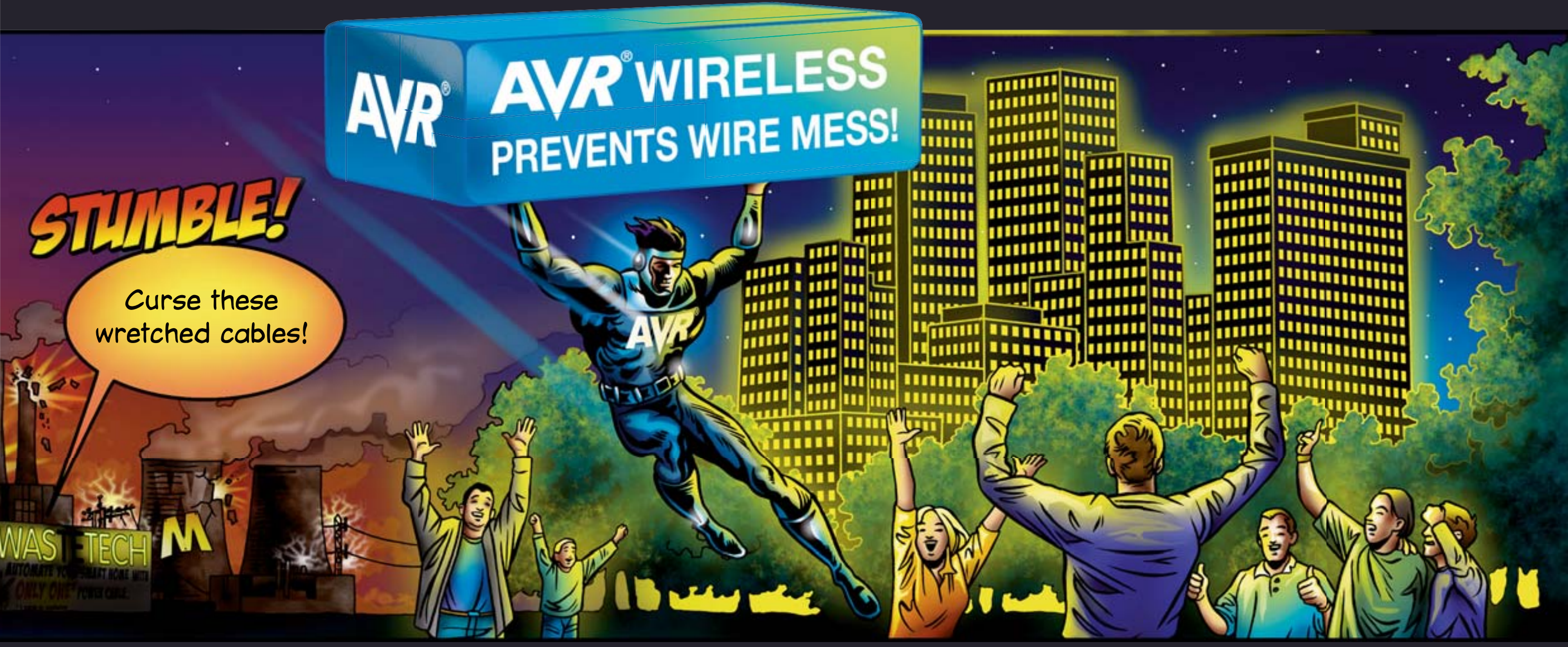
STMicroelectronics announces four evaluation and development kits, from Hitex, IAR, Keil and Raisonance, designed to support its STM32 microcontroller, which is based on the ARM CortexT-M3 core. All the kits are based on proven tool solutions that are well known to application developers working with ARM core-based devices.

News ID 835

■ Zilog: 32-bit high security single chip solution

Zilog unveiled its new 32-bit high-security, high-performance Zatara single chip solution for the rapidly growing secure transactions marketplace. ZataraSeries ARM ASSP was built from the ground up to deliver high level of security and safety to protect the integrity of EFT POS transactions.

News ID 755



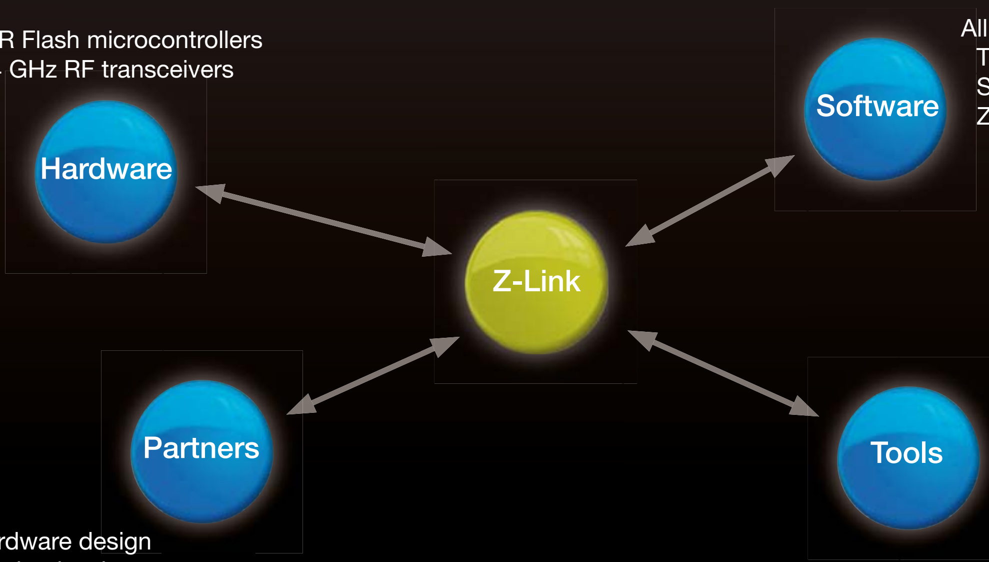
Go wireless with AVR® Z-Link®

The Z-Link Solution

Atmel®'s AVR Z-Link product line provides the best IEEE 802.15.4 compliant and ZigBee™ certified solution available. Based on the AVR RF family of radio transceivers, AVR microcontrollers, free software, reference designs and development kits, the solution enables deployment of large wireless networks with very low energy consumption.

RF radio transceivers provide the widest link budget available with -101 dBm receive sensitivity and 3 dBm transmit power. Combined with picoPower™ technology used by the AVR microcontrollers, the chipset achieves very low power consumption.

The AVR family of 8-bit RISC microcontrollers which features up to 256 KB Flash memory and rich peripherals, can easily handle both wireless communication and your main application.

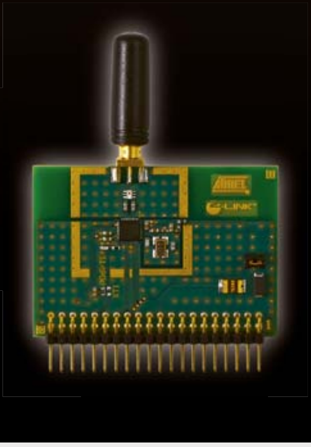


AVR Flash microcontrollers
2.4 GHz RF transceivers

All free!
Transceiver Access Toolbox
Software MAC
ZigBee Stack

Hardware design
Stacks development
Wireless gateways
System software

Evaluation kits
Demo kits
One toolchain for all AVRs



Get more at: www.atmel.com/AVRman

Advances in distributed video processing technology

By Peter Duchemin, for Hyperstone

Surveillance cameras are nowadays widely used. With modern 32-bit RISC/DSP architectures, such digital network cameras can almost be built with only one chip.

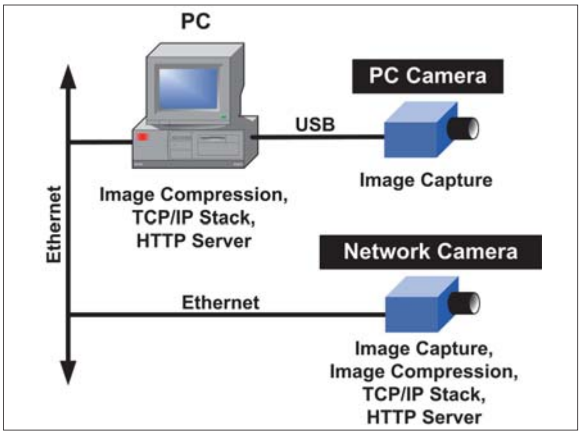

PC
PC Camera
USB
Image Capture
Ethernet
Image Compression, TCP/IP Stack, HTTP Server
Network Camera
Ethernet
Image Capture, Image Compression, TCP/IP Stack, HTTP Server

Figure 1. The difference between a PC camera and a network camera

■ Consumer webcams are available for little cost in almost any computer shop. But most of these are simple passive cameras that just do digital image-capture and deliver these images to a PC, typically by means of a USB cable connection. It is only the PC that is able to process the images, such as compressing them in the JPEG format, and making them available, for example, on the connected TCP/IP network running an HTTP server to which remote users can connect with web browsers. These so-called “webcams” should more precisely be renamed “PC-cams” as they do not really include the “web”.

Real network cameras on the other hand do not need a PC for image compression and attachment to a network (figure 1). Apart from a CMOS or CCD image sensor and, for example, an Ethernet network interface, these types of cameras employ a high-performance CPU such as the Hyperstone hyNet S, which can compress images, encapsulate them in TCP/IP frames and deliver them over the network. Video streams can be provided to attached web browsers by means of the HTTP protocol. Alternatively e-mails with attached images could be sent, or video streams could be stored as files to a remote FTP server.

Whatever the network interface is, the image compression is a very important requirement. Image compression, for example the JPEG format, can reduce the data sizes of images by a factor of 20 to 30, depending on the required image quality and also on ambient light conditions. This is important for bandwidth considerations. It is obvious that if the image compression takes place on the camera, then more images can be delivered in a given time frame over a given interface. For example: a VGA image (640 x 480 pixels) with 16-bit colour information per pixel has a raw size of 600 kBytes. An image sensor which delivers image sequences of up to 30 frames per second (fps) would then produce almost 150 MBit/s of data continuously. Even for 100 MBit/s Ethernet this amount of data is too much to transport in raw format. In compressed format such as M-JPEG the required bandwidth can easily reduce to 5 MBits/s. Other compression algorithms, such as MPEG4, can reduce the required bandwidth even lower to 1 MBit/s or less, but are not appropriate for all industries or applications due to lower image quality.

Intelligent digital network cameras as described have been known for quite some time, and are available from a range of different manufacturers. However, these used to be very expensive when compared to simple passive webcams. Nowadays this difference is much less significant as new generations of single-chip solutions offer the performance of a sophisticated network camera for the historic cost of a simple webcam. The Hyperstone hyNet S chip concentrates all necessary functional blocks and interfaces on one piece of silicon (figure 2). Most popular digital CMOS sensors can be attached directly to the hyNet S via the YUV interface. Where CCD sensors offered better image quality and higher sensitivity in the past, modern CMOS sensors can offer almost the same features today, but at a fraction of the cost. The CMOS image sensor is configured via a serial I2C channel and continuously delivers clocked image data directly to the CPU’s memory by means of a DMA process. This happens autonomously in the background, so that, in the meantime, the CPU can undertake other tasks.

Once an image has been completely stored to memory it is passed on to the internal video compression unit (VPU) which compresses the image in JPEG format. As this happens with hardware assistance the compression time is very small, typically 20 to 30 ms per image. From here on, higher software protocol layers deliver images over different interfaces. As well as the standard 100 MBit/s Ethernet interface, various wireless interfaces can be attached to the hyNet S chip: via the PCI host interface standard WiFi modules can be connected; over serial ports (UART, SPI), in addition Bluetooth and ZigBee modules can be directly addressed.

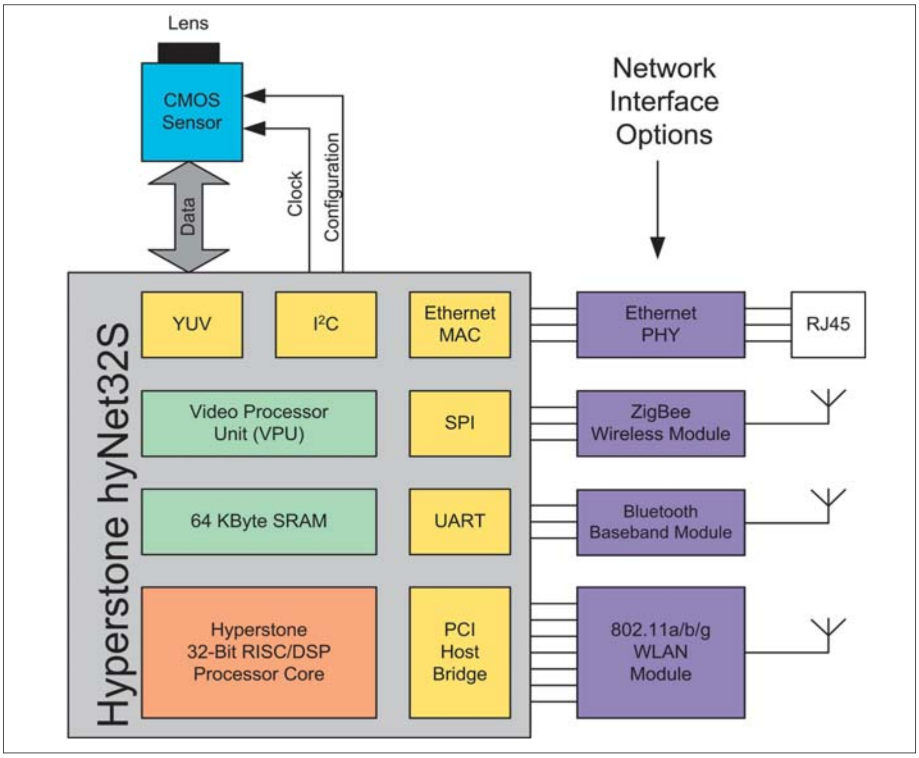


Figure 2. Hyperstone hyNet S architecture and interface options

The critical role in this is being played by the firmware, which has to control all on-chip processes when the task is to continuously deliver a compressed video stream to network clients without interruptions or deadlocks. It has to ensure that image-capture and compression processes are carefully synchronised with each other and at the same time asynchronous image requests from network clients can be served on a continuous basis. It is therefore very important to understand that only careful design of DMA and software task priorities in the firmware enable this task to be really achieved.

Apart from the classic surveillance tasks, where video streams are watched or stored on a continuous basis, a modern intelligent camera can do many more things with captured images; operations that were mostly only possible on PCs attached to the camera in the past. One example is the analysis of the raw images right after being captured. For example, a more intelligent surveillance process would transfer images only after a change in the monitored scenario has taken place. If the camera watches a doorway a motion detection algorithm applied to the images could start compression and image transfer only when a person moves through the doorway. This again reduces the necessary bandwidth on the network.

In a similar fashion the camera could store sequences of images on some local buffer memory. If an alarm occurs, e.g. triggered by a sensor, the camera captures images for a period of time but can otherwise be in power save or sleep mode. When additionally equipped with a wireless interface this also enables efficient battery operation and complete autonomy from local connections. The images from the alarm situation can then be transferred to a central access point on request, which does not even require a fast interface. A ZigBee wireless interface with data rates as low as 100 kbit/s are sufficient, as the video would not need to be streamed in real time.

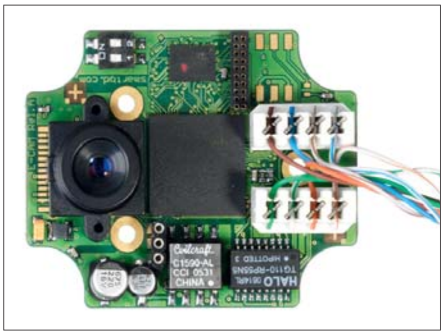


Figure 3. A Power-over-Ethernet (PoE) camera based on the Hyperstone hyNet S IC

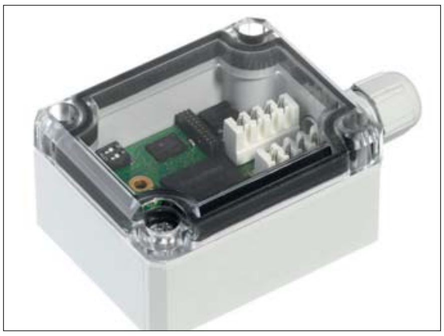


Figure 4. Hyperstone hyNet S camera in a protective housing (IP65)

Another application is pattern recognition, where barcodes, data matrix codes or labels can be recognised on the camera. A specific pattern could start the camera again to capture live images to report an error or alarm situation. Combined with an RFID reader such a camera could also take pictures of any goods that pass a control station. This would enable to detect and document damaged items. Last but not least, with shape recognition algorithms objects can be identified, such as reversed bottles in vending machines, where wrong items can automatically be rejected or sorted out.

In traffic control applications, intelligent cameras can count cars or recognise number plates of moving cars and perform automatic billing on toll roads, or collect fees in parking lots without the need to manually open a barrier when entering or leaving the site. In access control applications, face recognition algorithms can identify persons authorised to pass a gate to a secured area. In this case, image data can be pre-processed on the camera and sent to an authorisation server on the back-end of the network, getting the response in a fraction of a second.

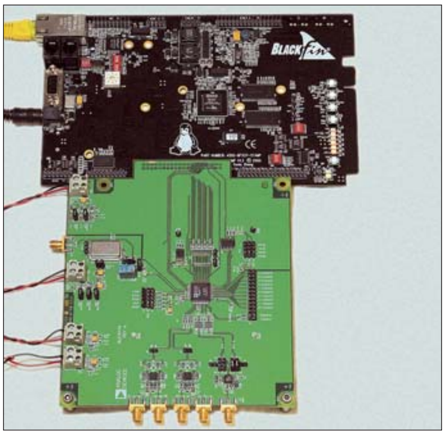
If the network camera has an Ethernet interface it does not necessarily need to have a separate power supply cable. With the Power-over-Ethernet (PoE) standard according to IEEE 802.3af, the wires of the Ethernet cable can also be used to transfer the necessary energy to operate the camera, once again reducing installation costs. This requires of course PoE-enabled Ethernet switches which can deliver up to 13W of power per network segment. A typical PoE network camera based on the hyNet S chip (figure3) requires around 1.5W so that up to 8 cameras can be operated on one Ethernet cable segment. By supplying both power and the network interface over one cable it is also much easier to put the camera in a protective housing, e.g. for outdoor or heavy duty operations (figure 4).

Other trends are the reduction of mechanical parts in digital cameras. Should a certain area need to be monitored, where motor cameras have been used in the past in order to do panning or zooming of image areas, today these functions can also be replaced by image processing algorithms in software. For example a wide angle lens on a sensor with very high resolution (megapixels) covers a large area and a piece of software directs a view window of e.g. VGA size within this area allowing digital panning and zooming controlled from a remote location. This also requires a powerful CPU like the Hyperstone hyNet S. ■

Single digital signal controller handles advanced RFID applications

By David Katz, Glen Ouellette, Rick Gentile, Giuseppe Olivadoti,
Analog Devices

This article describes how the seemingly disparate functions in a RFID system - signal conversion and network connectivity - are executed by a single Blackfin processor.



An MxFE evaluation board and a Blackfin ADSP-BF537 STAMP development platform for RFID reader applications

■ RFID technology is transforming many existing applications, from inventory control to fast checkouts at the supermarket, and enabling many new applications. The RFID signal chain starts with small tags attached to the units of interest which convey information in the form of a bit stream to an RFID reader that detects tag presence in a specific area, and reads their information. At the back end, a server-based system maintains and updates the tag database, generating alerts or initiating other information-based processes within the enterprise. Most RFID readers currently employ more than one processor to satisfy application requirements. Typically, a signal processor is interfaced to an analog-to-digital converter (ADC) and a digital-to-analog converter (DAC). Then a network processor communicates with a server for information storage and retrieval.

RFID technology enables many new types of applications by allowing concurrent monitoring of multiple items, without requiring a person to touch each one (with a hand-held barcode scanner, for example). The kinds of applications that can take advantage of this automated identification include diverse areas such as inventory control, logistics management, surveillance, and toll collection. Today, the ubiquitous merchandise-oriented universal product code (UPC), a one-dimensional (1D) barcode, graces nearly everything available for public purchase. The barcode contains relevant information about the item to which it is attached, perhaps including the items suggested retail price and/or the place and date of manufacture. 1D and 2D barcodes can also be used to track shipment details for an item.

RFID technology replaces the UPC with an EPC (Electronic Product Code), in the form of a stream of bits. At a minimum, an EPC allows the same type of information contained in a barcode to be collected automatically and accessed remotely, with minimal human intervention. In addition, an EPC can include much more information relating to unique identifying characteristics of the tagged item, even if there are many identical items. Moreover, unlike a conventional barcode, it does not matter in which direction the items are facing, or what the ambient lighting conditions are - the items can be still be detected and tracked. RFID uses radio-frequency (RF) transmissions of bit streams to communicate with, identify, classify, and/or track objects. Each object has its own RFID tag (also known as a transponder). The overall system employs a tag reader, a subsystem that receives RF energy from each tag. The reader has embedded software that manages the interrogation, decoding, and processing of the received tag information, and it communicates with a storage system that houses a tag database and other relevant information.

The RFID reader provides the connectivity between individual tags and the tracking/management system. Available in a variety of form factors, it is typically small enough to be mounted on a counter, tripod, or wall. Depending on the application and operating conditions, there may be a multiplicity of readers to fully service a specific area. In a warehouse, for example, a network of readers can ensure that 100% of all pallets are queried and logged as they pass from point A to point B. Overall, the reader provides three main functions: bidirectional communication with the tags to isolate individual ones; initial processing of received information; and connection to the server that links the information into the enterprise. The RFID reader must deal with multiple tags within the field of interest.

The primary challenge is that collisions will occur when many readers send out queries and multiple tags respond at the same time. The most common way to avoid this problem is to use some form of time-division multiplexing algorithm. Readers can be set to interrogate at different times, while tags can be configured to respond after a random time interval. An RFID tag consists of an IC chip holding unique information (such as EPC data) about the object to which the tag is affixed, an antenna (usually a printed circuit pattern) for receiving RF energy from the reader and for transmitting

Visit us at www.Zilog.com/z8encoremc
ARE YOU INTO IT?

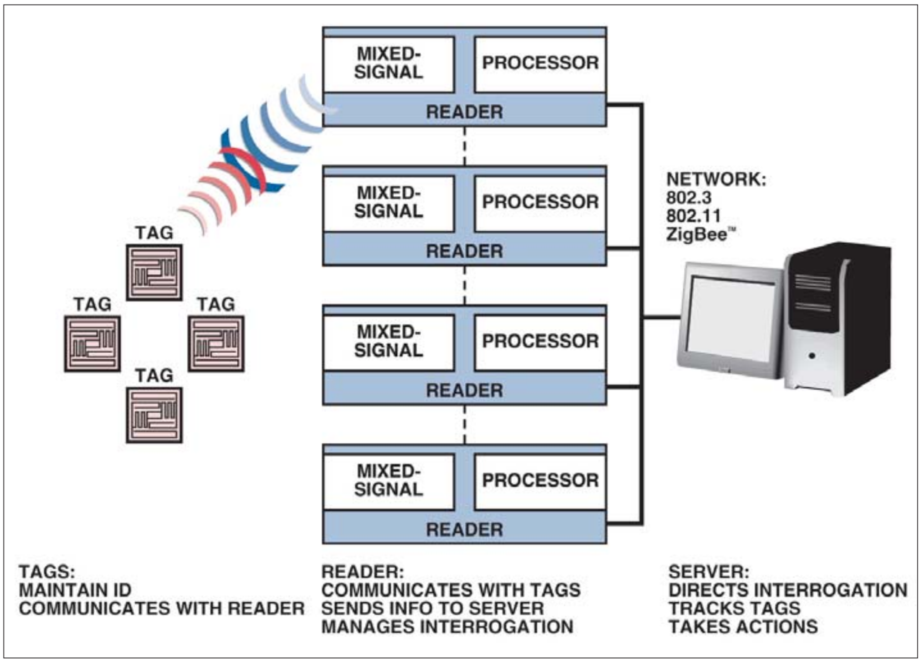


Figure 1. Simplified representation of an RFID system

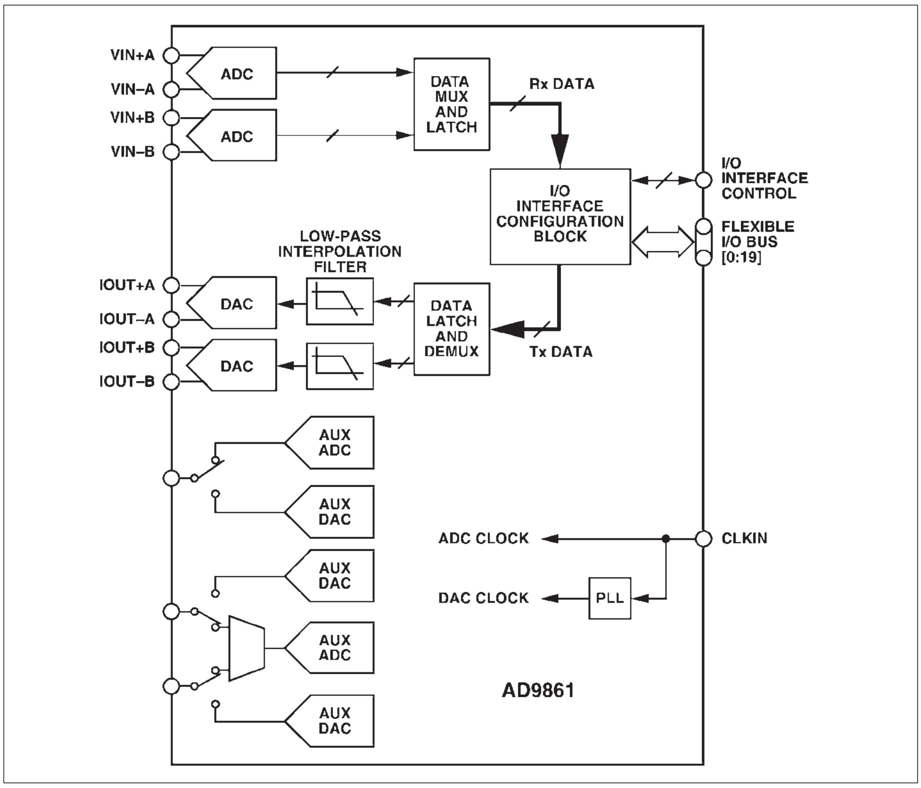


Figure 2. Block diagram of a representative MxFE IC, the AD9861

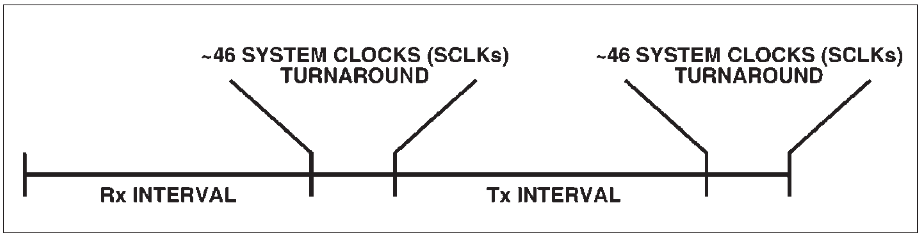

~46 SYSTEM CLOCKS (SCLKs) TURNAROUND
~46 SYSTEM CLOCKS (SCLKs) TURNAROUND
Rx INTERVAL
Tx INTERVAL

Figure 3. Illustration of Tx/Rx sequence for RFID reader with a single ADC/DAC interface

information, and some kind of housing that envelops the tags components. The distance from the tag to the reader, an important system variable, is directly influenced by the tag technology.

Passive tags are powered exclusively by RF energy sent from the reader, they do not have an integrated battery, so they can be inexpensive, mechanically robust, and quite small (e.g., about the size of a thumbnail). Passive tags have a limited reader-to-tag range, however, because the received power depends on their physical proximity to the RFID reader. The range of the link is also affected by the RF frequency chosen. Low-frequency (LF) tags commonly utilise the 125 kHz to 135 kHz portion of the spectrum; since their range is constricted, they are mainly used for access control and animal tagging. High-frequency (HF) tags, mostly operating in the 13.56-MHz band, allow a range of a couple of feet. UHF tags, on the other hand, operate at frequencies from 850 MHz to 950 MHz and have a considerably longer range - 10 feet or more. Moreover, because of the potentially wider bandwidth available, a reader can interrogate many of these tags at a time, as opposed to the one-on-one tag-reading process at lower frequencies. This trait helps minimise the need for multiple readers in a given zone, making UHF tags very popular in industrial applications for inventory tracking and control.

Like passive tags, semi-active tags reflect (rather than transmit) RF energy back to the tag reader to send identification information. However, these tags also contain a battery that powers their ICs. This allows for some interesting applications, such as when a sensor is included in the tag to monitor real-time attributes, such as temperature or humidity

Active tags go one step further, by powering both the tag IC (along with any sensors) and the RF transmitter, using an integrated battery. Being self-powered, they can operate over a much larger reader-to-tag range (up to 100+ meters), which also translates into allowing goods to move past the reader much faster than in the case of passive or semi-active tag systems. In addition, active tags can carry much more product information than just an EPC code.

The three elements of the RFID reader software architecture are: the back-end server interface, the middleware, and the front-end tag reader algorithms. RFID readers often contain a networking element - wired Ethernet (IEEE 802.3), wireless Ethernet (IEEE 802.11 a/b/g), or ZigBee (IEEE 802.15.4), - that connects single RFID-read events to a central server. The central server runs a database application, with functions that include matching, tracking, and storage. In many applications, an alert function is also present (the re-order trigger, for supply chain and inventory management systems, or an alert to a guard, for security applications). Incidentally, a reader built around a high-performance embedded processor that runs μClinux (such as Blackfin) has a substantial advantage over one that does not when communicating with a back-end server. The presence of a robust TCP/IP stack and the availability of SQL database engines greatly reduce an otherwise major integration burden in the development process.

In RFID terms, middleware is the software translation layer between the front-end RFID reader and the back-end enterprise system. The middleware filters the data from the reader and ensures that it is free of multiple reads or bad data. In early RFID systems, the middleware ran on the server, but the filtering of RFID data is now often performed on the reader before sending it through the enterprise network. This degree of increased functionality is another advantage embedded processors bring to this application space. The systems filter- and transform-intensive signal processing, occurring in the front end of the reader, requires a device with the kind of strong signal-processing performance typically associated with Blackfin processors.

For communicating with a tag, the mixed-signal front-end (MxFE) IC forms the interface of interest. MxFE devices are general-purpose, intermediate-frequency subsystems that include A/D and D/A converters, low-noise amplifiers, mixers, AGC circuitry, and programmable filters. Output streams of I&Q data connect directly to processor parallel ports. Analog Devices MxFE IC family members constitute the highest performance narrow-band receivers available, well-suited to RFID—and other—applications.

Blackfin processors provide connectivity to both wired and wireless networks. Some processors, such as the ADSP-BF536 and ADSP-BF537, have a 10-Base-T/100-Base-T Ethernet MAC on chip. On the wireless side, all Blackfin processors can connect directly to both 802.15.4 ZigBee and IEEE 802.11 chipsets via the SPI and SPORT peripherals. Line-speed transfers can be obtained without consuming the entire processor bandwidth. In addition, Blackfin processors include a parallel peripheral interface (PPI), which can connect directly to ADCs and DACs such as those mentioned above. Some Blackfin processors include two PPIs, which can expand system functionality even further—allowing a camera to be connected to an RFID reader, for instance.

For RFID applications, a single PPI is often sufficient because of the way the RFID reader interrogates tags. First, the PPI is configured in transmit mode, and the processor sends a digital sequence to a DAC. The transmitted sequence is converted to an analog signal, which is then up-converted and sent out to excite/wake up local RFID tags, which then respond. Simultaneously, the PPI is reconfigured as a receiver in a small number of processor system clock pulses, as shown in figure 3. In this way, a down-converted RF signal can be sampled by an ADC and brought into the Blackfin directly. In the figure, the time between each receive (Rx) and transmit (Tx) interval is measured in system clock cycles. The elapsed time allows for the transmitted signal to reach the tag and for the tag to transmit a response. In some RFID applications, a Blackfin processor alone can act as the server - for example, when large data stores and database manipulations are not necessary. For instance, imagine an elderly person wearing a bracelet with a tag that could be monitored within the house. If no signs of activity were noted within a specified time interval, the monitoring agency could alert registered friends or relatives. The most common application of RFID is asset management, which benefits by reduction of lost inventory, elimination of incorrect deliveries, improvement in distribution logistics, and lessening of stock-outs - as the result of being able to track a pallet's movement through the warehouse. An

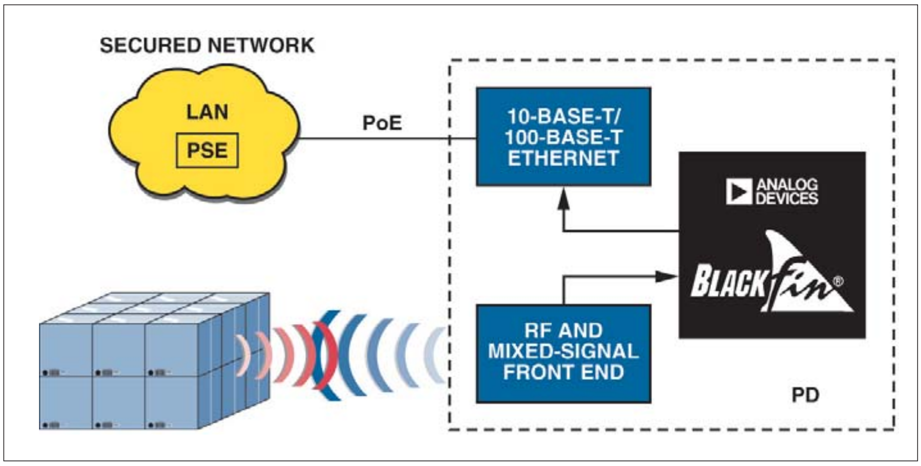


Figure 4. Example of a PoE-based RFID asset-tracking system

RFID system in a large warehouse can track a container-laden pallet's movement from the time the pallet enters the warehouse to the time it leaves. Such a system relies on fixed RFID readers placed throughout the warehouse and at points of inbound/outbound shipping.

As a means of simplifying wired infrastructure, Power-over-Ethernet networks (PoE) are ideal for these types of applications. IEEE 802.3a/f PoE deals with networked systems in low-power applications. PoE has a recommended maximum cable length of 100 meters, which is suitable for many embedded RFID applications. A network processor supporting embedded RFID applications must have the performance and integration to handle a multilayer IP stack, in addition to the RFID acquisition software. The ADSP-BF537 Blackfin processor - which includes a 10-Base-T/100-Base-T Ethernet MAC - is a good example. Many Ethernet PHY devices provide a status pin with the capability to interrupt upon a status change. Blackfin processors support this feature within the interrupt functionality to yield a robust, power-efficient system. For applications such as a forklift-mounted scanner or a portable hand-held scanner, where wired or PoE operation is not possible, wireless protocols like IEEE 802.11b/g allow RFID readers to connect to a wireless access point, as shown in figure 5. Blackfin processors connect to 802.11 chipsets via either serial or parallel interfaces.

The µClinux operating system is a popular choice for facilitating network connectivity - which is the largest software component of the reader - and the critical requirements of robustness and standards compliance. The software components that make up a Blackfin RFID reader are available on the Blackfin.µCLinux.org website. This includes drivers necessary to interface to the mixed-signal, front-end IC, as well as a DMA driver that is very useful in moving data through a system. The µClinux-based network stack and SQL database engines are also available. When reading RFID tags, it is essential to ensure that real-time requirements are met. Since the µClinux scheduler is not strictly real-time, it can be replaced with the ADEOS real-time scheduler, which safely holds off µClinux interrupts until the real-time critical processing is finished. The front-end reader software can execute from the ADEOS domain in real time, while the middleware and back-end server interface can run in the traditional µClinux environment. ■

Product News

■ NEC: 32-bit flash MCUs for access control and POS systems

NEC introduces two additions to their V850ES lineup of 32-bit all flash microcontrollers. The new V850ES/Hx3 and V850ES/Jx3 MCUs execute up to 69 Dhrystone MIPS at clock speeds of 32 megahertz, and are well-suited for next-generation access control products; point-of-sale systems; industrial controllers and drivers; heating, ventilation and air-conditioning systems; computer peripherals; home appliances; factory automation equipment; metering devices; medical and process analytics instruments; and test and measurement equipment.

News ID 760

■ Digi-Key: full line card of Luminary Micro MCUs available

Digi-Key announces that the full line card of Luminary Micro's Stellaris family of Cortex-M3-based microcontrollers is now in stock. The availability of these products is in keeping with the signing of a global distribution agreement which was announced in June. Luminary Micro's entire product portfolio is available through Digi-Key, encompassing 51 Stellaris microcontrollers, along with compact, versatile and easy-to-use Stellaris evaluation and development kits that provide everything an embedded developer needs to be up and running in 10 minutes or less.

News ID 710

■ Microchip: 8-bit MCU family with up to 64KB of Flash

Microchip announces eight new members of the PIC18F 8-bit microcontroller family that take advantage of Microchip's latest process technology developments. The new devices' low-power nanoWatt technology features include power-managed modes, an operating voltage range of 1.8V to 3.6V and efficient on-chip peripherals. The devices also feature a precision internal oscillator that will support full speed (16 MIPS) operation from a 3V supply. In addition, the new family offers full code, pin-out and tool compatibility with Microchip's wide portfolio of 8-bit microcontrollers.

News ID 761

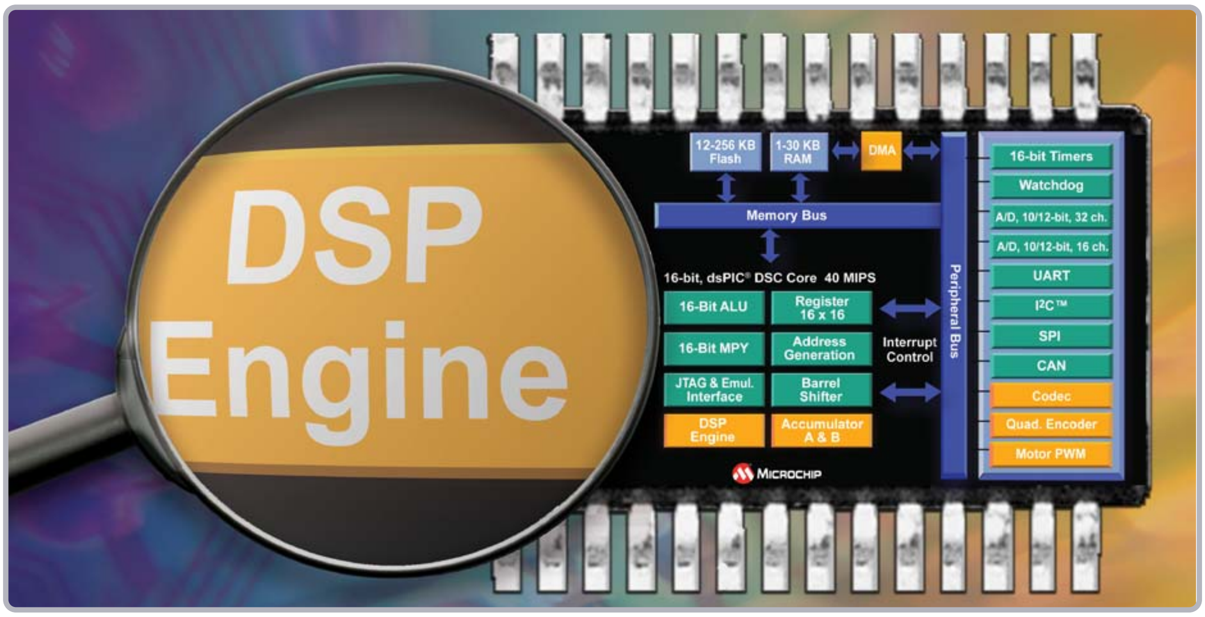
■ TI: starter kit for floating-point digital signal controllers

Texas Instruments announces the F28335 eZdsp starter kit for its recently introduced floating-point TMS320F2833x digital signal controllers. The F28335 eZdsp starter kit is a stand-alone tool that provides developers a platform to easily develop software for the F2833x controllers and comes complete with an F28335 target board, TI's Code Composer Studio Integrated Development Environment (version 3.3), universal AC power adapter and USB cable. The target board features the 150MHz/300 MFLOPS F28335 DSC with 512KB flash and 68KB RAM, 128Kx16 off-chip SRAM.

News ID 816

More information about each news story is available on www.embedded-control-europe.com/ece_magazine
You just have to type in the "News ID". —

Look at the Easiest Way to Add DSP to Your MCU Application



Microchip's dsPIC® Digital Signal Controllers integrate a true DSP engine in their 16-bit microcontroller architecture to help you add advanced functionality with ease

You do not need to learn DSP! Extensive libraries and graphic digital filter design tools make digital signal control easy!

Easy migration! dsPIC digital signal controllers are pin, peripheral, instruction set and tools compatible with Microchip's PIC24 microcontrollers.

3 EASY STEPS TO GET STARTED...

1. **FREE 16-bit web seminars**
2. **FREE product samples**
3. **Exclusive development tool discounts available at www.microchip.com/easyDSC**

Selected 16-Bit Digital Signal Controllers

Device	Pins	RAM (KB)	Flash (KB)	Features Include
dsPIC33FJ12GP201	18	1	12	True DSP Engine Rapid, Predictable Interrupt Response 10/12-bit ADC, 32 ch. with 1.1 MSPS 8 ch. DMA Programmable PLL with Internal Oscillator Peripheral Pin Select 8x Motor PWM 2x CAN
dsPIC33FJ12MC201	20	1	12	
dsPIC33FJ12GP202	28	1	12	
dsPIC33FJ12MC202	28	1	12	
dsPIC33FJ64GP506	64	8	6	
dsPIC33FJ64MC506	64	8	6	
dsPIC33FJ128GP706	64	16	12	
dsPIC33FJ128GP708	80	16	128	
dsPIC33FJ128MC708	80	16	128	
dsPIC33FJ256GP710	100	30	256	
dsPIC33FJ256MC710	100	30	256	

Purchase and program your 16-bit dsPIC digital signal controllers and related development tools at **www.microchipdirect.com**

Check out our web site for special pricing on the 16-bit 28-pin Development Kit and MPLAB® ICD 2 In-Circuit Debugger!

Visit www.microchip.com/easyDSC!

microchip DIRECT
www.microchipdirect.com

Now Pb-free!
RoHS Compliant

MICROCHIP
www.microchip.com/easyDSC

A new debugging tool for analysis of embedded real-time systems

By John Carbone, Express Logic

TraceX, a new analysis tool, paints a picture of the system that standard debuggers cannot. It enables developers to see interrupts, context switches and other system events without time-consuming instrumentation of code and examination of resulting data.

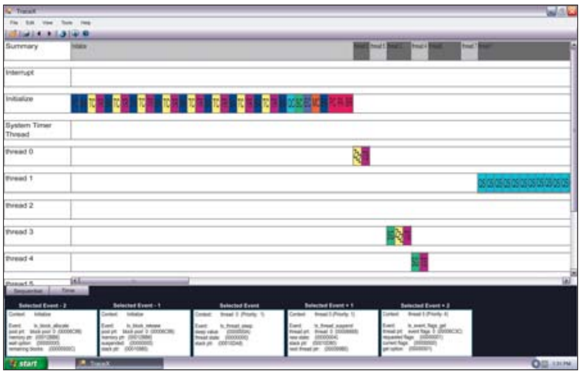


Figure 1. TraceX offers a graphical view of real-time system events. In this example, you can see the initialisation process and early execution of the standard ThreadX demonstration program.

■ Leading-edge real-time operating systems (RTOSs) offer powerful multitasking features such as the ability to make rapid context changes between threads and support for many thread priority levels. While this makes it possible to more easily provide real-time control, these capabilities create the potential for considerable complexity in the way that threads share resources. As a result, real-time applications are often difficult to understand and even more difficult to optimise.

With its new TraceX tool, Express Logic has made a major improvement in the way that developers can visualise and understand the behavior of real-time systems. Tools such as TraceX provide the capability to see clearly the occurrence of system events such as interrupts and context switches that occur out of view of standard debugging tools. The ability to identify and study these events and pinpoint their timing in the context of the overall systems operation enables developers to identify bugs in less time and optimise multitasking behavior.

Real-time programmers have long understood the importance of system behavior to the functionality and performance of their applications. The conventional approach to address these issues is to instrument the code by leaving “bread crumbs” that will generate data on system behavior when the code reaches certain stages such as toggling an I/O pin, using printf, setting a variable or writing a value to a file. Inserting such responses can require a considerable amount of time, especially when you consider that the instrumentation code often does not work exactly as expected the first time around and has to be debugged. Once that part of the program is verified, the instrumentation code needs to be removed and its removal also needs to be debugged.

Since much of the instrumentation process is manual, the process is time-consuming and prone to additional errors. Besides instrumenting the code, the developer also needs to find a way to interpret the data that is generated. Due to the volume of information generated by the instrumentation code, gaining an understanding of what system events have transpired, and in what sequence, can be challenging in itself.

A few RTOS vendors have made efforts to address these concerns by developing tools that assist in capturing and interpreting these system events. Generally, these tools provide a graphical display of a captured event log, and assist in giving developers visibility into the behavior of their system. One weakness of many of these tools, though, is that they are typically available only as an element of an overall integrated toolset which is often very expensive to buy and may duplicate other tools already in use. Most system event analysers also tend to be inflexible in the way that they manage the trace buffer that stores system events. They typically write to only one specific trace buffer and the buffer cannot be turned off or back on by the application, thereby risking the loss of events of interest, or saturating the buffer with useless clutter.

Most of these programs present trace events graphically, on multiple lines, representing the various threads in the program, and system routines such as interrupt handlers, initialization code, etc. and the user may have to do a considerable amount of scrolling to see all of the captured events. And, since developers generally get information on events by clicking on the event and viewing a pop-up window with key event information, they can only view one event at a time.

The new TraceX system event viewer (figure 1) from Express Logic provides developers with a new approach to system and application event viewing that avoids these weaknesses, common in similar products. Designed to work with Express Logic ThreadX RTOS, TraceX collects a database of system and application events on the target system during runtime. These events include thread context switches, preemptions, suspensions, terminations and systems interrupts. The user has the opportunity to log any desired application events using an application programming interface (API) provided with TraceX.

KEIL™
An ARM® Company

Microcontroller Development Tools

ARM Microcontroller Solution

ARM-powered Microcontrollers – available from many silicon vendors; offer high computing performance along with rich peripherals. Turn ARM Microcontrollers into your solution for cost-sensitive powerful applications – **with Keil Development Tools**.

C/C++ Development Kit

The RealView **Microcontroller Development Kit** (MDK) is the complete software development environment for ARM7/9 and Cortex-M1/M3.

MDK is easy to learn and use, yet powerful enough for the most demanding embedded ARM application. The integrated Device Database® simplifies tool configuration and includes more than 250 ARM-powered Microcontroller variants for your embedded project.

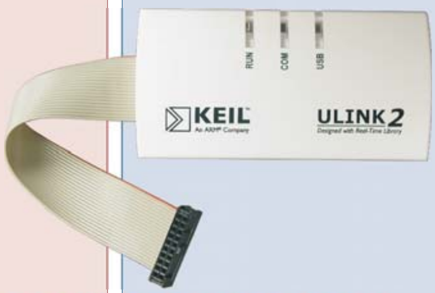
RealView® Microcontroller Development Kit

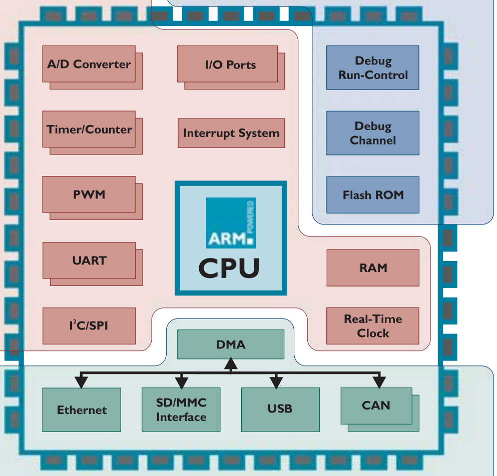
- RealView C/C++ Compiler
- RTX RTOS Kernel Library
- µVision Device Database & IDE
- µVision Debugger & Analysis Tools
- Complete Device Simulation

Examples and Templates

JTAG Debugger

ULINK2® connects to the JTAG or 2-wire debug interface and supports on-the-fly debugging and Flash programming.





RealView® Real-Time Library

- RTX RTOS Source Code
- TCPnet Networking Suite
- Flash File System
- USB Device Interface
- CAN Interface

Examples and Templates

RTOS and Middleware

The RealView **Real-Time Library** (RL-ARM) solves the real-time and communication challenges of your ARM project and expands **MDK** with essential components for sophisticated communication and interface peripherals.

Learn more about RealView **MDK**, **RL-ARM**, and **ULINK2**. Download a free evaluation version from www.keil.com/demo or call **+49-89-456040-0**.

Embedded World, Nuremberg 26. - 28. February 2008, Booth 10-229

Cx51

Keil Cx51 is the de-facto industry standard for all classic and extended 8051 device variants. C51 Version 8.5 includes the latest devices such as XC800, ADE7169, and C8051F4xx - F6xx.

More information: www.keil.com/c51

C166

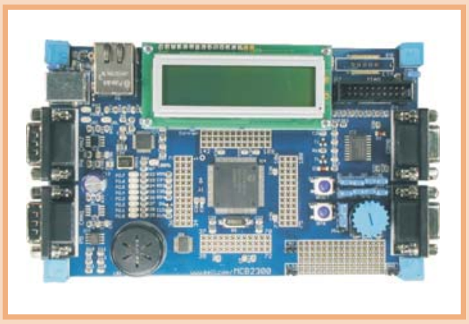
Keil C166 is the complete software development environment for Infineon C166, XC166, XC2000 and ST Microelectronics ST10 with debug and programming support via ULINK2.

More information: www.keil.com/c166

ARM

RealView MDK combines the best-in-class ARM C/C++ Compiler, the genuine Keil µVision IDE/Debugger/Simulator, and the royalty-free RTX RTOS Kernel.

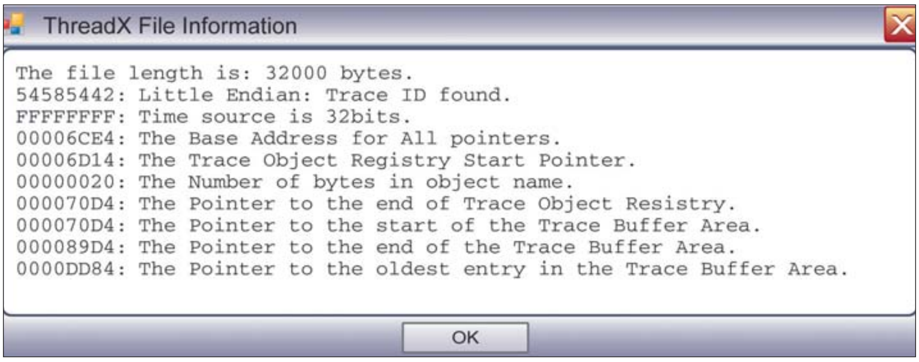
More information: www.keil.com/arm



Keil MCB evaluation boards come with code size limited tools and extensive example projects that help you get up and running quickly with your own embedded application.

More information: www.keil.com/boards

www.keil.com



```
The file length is: 32000 bytes.
54585442: Little Endian: Trace ID found.
FFFFFFFF: Time source is 32bits.
00006CE4: The Base Address for All pointers.
00006D14: The Trace Object Registry Start Pointer.
00000020: The Number of bytes in object name.
000070D4: The Pointer to the end of Trace Object Resistry.
000070D4: The Pointer to the start of the Trace Buffer Area.
000089D4: The Pointer to the end of the Trace Buffer Area.
0000DD84: The Pointer to the oldest entry in the Trace Buffer Area.
```

Figure 2. TraceX can display information about the circular buffers used to capture ThreadX event data during execution..

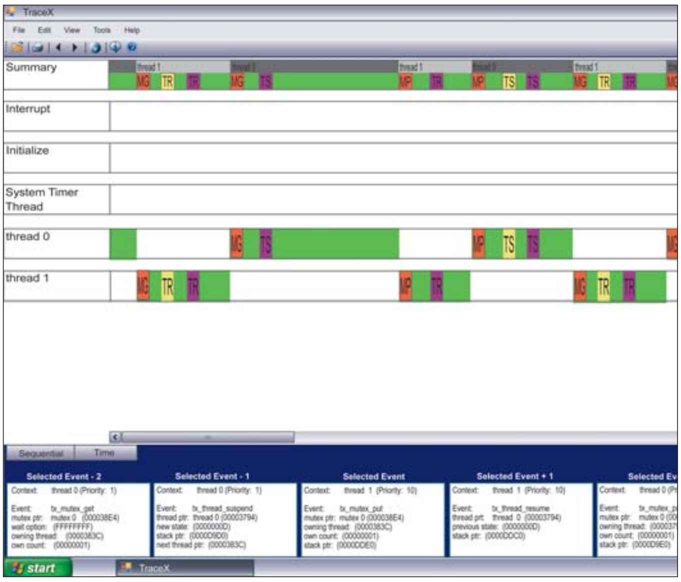
Events are logged in the database under program control with time-stamping and active thread identification so they can be displayed later in the proper time sequence. To make events available for sequential viewing, trace information is stored in a circular buffer on the target system with buffer size determined by the application. A circular buffer enables the most recent "n" events to be stored at all times and to be available for inspection in the case of a system malfunction or other significant event (figure 2).

Event logging may be stopped and started by the application program dynamically, such as when an area of interest is encountered. This avoids cluttering the database and using up target memory when the system is performing correctly. To enable developers to hone in on specific threads, a system event analyser should make it possible to use multiple trace buffers and to switch between them when necessary. The trace information may be uploaded to the host for analysis at any time, either when encountering a breakdown or after the application has finished running. With TraceX, once the event log is uploaded from the target memory to the host, it displays the events graphically on the horizontal axis which represents time. The various application threads and system routines, to which events are related, are listed along the vertical axis and the events themselves are presented in the appropriate row. All events are presented in the top summary row which provides developers with a handy way to obtain a complete picture of system events without scrolling.

Events are represented by color-coded icons, located at the point of occurrence along the horizontal timeline, to the right of the relevant thread or system routine. The axes may be expanded to show more detail about events or collapsed to show more events. When an event is selected, detailed information is provided for that event on the bottom of the screen including the context, event, thread pointer, new state, stack pointer, and next thread point. Information is presented not only for the current event, but also for the two events preceding and the two events following the current event.

An example of the bugs that can be solved more quickly and easily with TraceX is the classic priority inversion problem. Priority inversions arise because RTOSs employ a priority-based preemptive scheduler that ensures the highest priority thread that is ready to run actually runs. The scheduler may preempt a lower priority task in mid-execution to meet this objective. Problems can occur when high and low priority tasks share resources, such as a memory buffer. If the lower priority task is using the shared resource when the higher priority task is ready to run, the higher priority task must wait for the lower priority task to finish. If the higher priority task must meet a critical deadline, then it becomes necessary to calculate the maximum time it might have to wait for all its shared resources in determining its worst-case performance. Priority inversions are difficult to identify and correct. Their symptom is normally poor performance, but poor performance stems from many potential causes. Just as troublesome is the potential that the priority inversion might not be noticeable in testing, which could cause the application to be non-deterministic.

With the system event analyser, it is possible to relatively easily identify and correct priority inversions. The trace buffer clearly identifies which thread is running at any point in time. So it is easy to go back in time and determine whether a higher level priority thread is ready to run. The next step is typically determining the resource blockage causing the priority in-



Selected Event - 2
Context: thread 0 (Priority: 1)
Event: tx_mutex_get
mutex ptr: mutex 0 (000038E4)
wait option: (FFFFFFFF)
owning thread: (0000383C)
own count: (00000001)

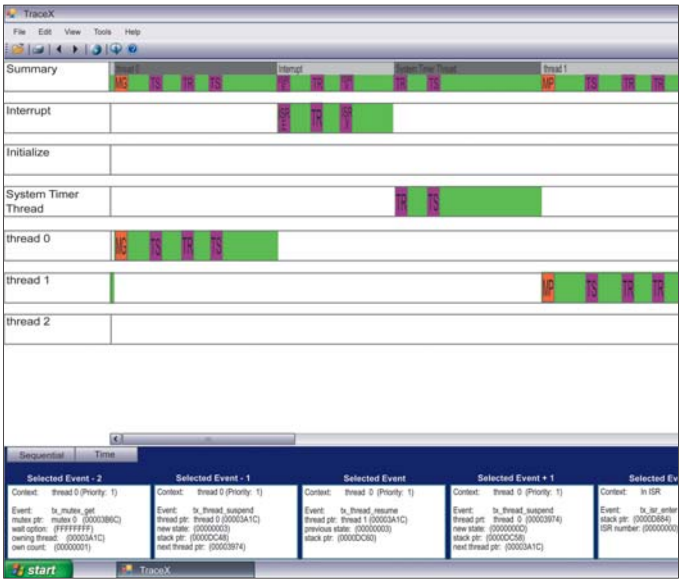
Selected Event - 1
Context: thread 0 (Priority: 1)
Event: tx_thread_suspend
thread ptr: thread 0 (00003794)
new state: (0000000D)
stack ptr: (0000D9D0)
next thread ptr: (0000383C)

Selected Event
Context: thread 1 (Priority: 10)
Event: tx_mutex_put
mutex ptr: mutex 0 (000038E4)
owning thread: (0000383C)
own count: (00000001)
stack ptr: (0000DDE0)

Selected Event + 1
Context: thread 1 (Priority: 10)
Event: tx_thread_resume
thread prt: thread 0 (00003794)
previous state: (0000000D)
stack ptr: (0000DDC0)

Selected Ev
Context: thread 0 (Pr
Event: tx_mutex_p
mutex ptr: mutex 0 (00
owning thread: (00003?
own count: (00000001)
stack ptr: (0000D9E0)

Figure 3. This display shows a simple priority inversion, where Thread_1 holds a resource that is needed by Thread_0, which is higher in priority. Thus, Thread_0 is delayed by a lower priority thread.



Selected Event - 2
Context: thread 0 (Priority: 1)
Event: tx_mutex_get
mutex ptr: mutex 0 (00003B6C)
wait option: (FFFFFFFF)
owning thread: (00003A1C)
own count: (00000001)

Selected Event - 1
Context: thread 0 (Priority: 1)
Event: tx_thread_suspend
thread ptr: thread 0 (00003A1C)
new state: (00000003)
stack ptr: (0000DC48)
next thread ptr: (00003974)

Selected Event
Context: thread 0 (Priority: 1)
Event: tx_thread_resume
thread ptr: thread 1 (00003A1C)
previous state: (00000003)
stack ptr: (0000DC60)

Selected Event + 1
Context: thread 0 (Priority: 1)
Event: tx_thread_suspend
thread prt: thread 0 (00003974)
new state: (0000000D)
stack ptr: (0000DC58)
next thread ptr: (00003A1C)

Selected Ev
Context: In ISR
Event: tx_isr_enter
stack ptr: (0000D884)
ISR number: (0000000

Figure 4. In this system snapshot, ThraceX shows a complex priority inversion with mutex priority inheritance. Thread_0 is still being delayed while it waits for a resource owned by a lower priority thread Thread_1. However, priority inheritance prevents the mid-priority Thread_2 from running during the priority inversion situation.

version. The normal process is to cycle back on the higher priority thread to identify the last point in time at which it was blocked. Clicking on this event will identify the mutex or semaphore on which the high-priority thread is blocked, and can be used to track the ownership of the resource and the lower priority event that has take control of the semaphore.

The simple priority version shown in figure 3 is identified by clicking on the MP event (Mutex Put) in the thread 1 line. Selected Event -2 shows a planned priority inversion. Thread 0 has suspended on mutex 0 which is owned by the lower priority thread 1. This could be an error if the developer did not know that this priority inversion was possible. More likely, this is a typical case of different priority threads competing for the same resource (protected by mutex 0).

Figure 4 shows a complex priority inversion with mutex priority inheritance revealed by clicking on the TR icon (Thread Resume) in the thread 0 time line. Selected Event -2 shows the same priority inversion problem as the previous example. In this case, the mutex is setup for priority inheritance so that when thread 0 attempts to get the mutex, thread 1 temporarily inherits the priority of thread 0. The effect of this is that although thread 5 becomes ready during the priority inversion window (inside the System Timer Thread at the TR icon), it does not run until after the priority inversion is cleared and thread 0 finishes its processing.

While most developers will begin using TraceX in order to understand and correct problems, a potentially broader benefit is derived from using the tool to analyse and improve system-level application performance. As a general rule, the greater the proportion of time spent on the application and the less spent on system-level tasks such as context switches, the faster the application will run. The system event analyser makes it easy to see at a glance how much time is devoted to system activity. The developer can easily drill down on specific events for diagnostic purposes.

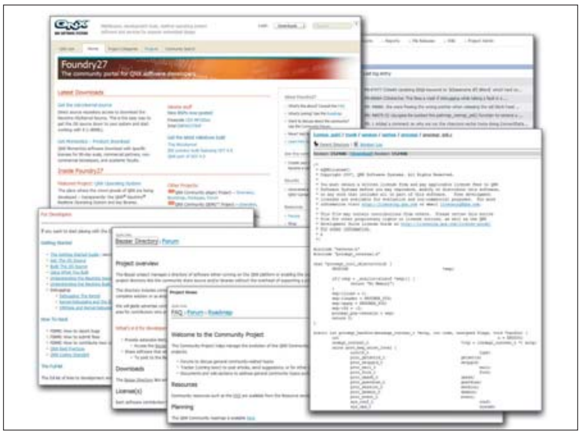
For example, with TraceX, you might see a large number of interrupts caused by the receipt of a data packet. The simplest and most common approach is to simply add each incoming data packet to a queue. As the number of packets received increases, the amount of time required to interrupt the current thread and throw the packet into the queue becomes substantial. The system event analyser makes it easy to visualise the overhead involved in putting packets into the queue. This information helps developers realise the need to look at alternative solutions such as setting a semaphore whenever the queue is empty. Then instead of interrupting the thread every time a packet arrives, it is interrupted only when there are no packets in the queue.

TraceX also provides the opportunity to look at how the setting of priorities affects performance. When threads are set at a relatively high number of priority levels, there is typically a lot of switching between threads to keep the highest priority thread that is ready running. Developers challenge performance by assigning priorities without thinking about the volume of context changes they are willing to tolerate. Context changes are transparent to traditional debugging tools so developers usually have no way to determine the impact of priorities. TraceX reveals context switches and makes it easy to understand their impact on performance. Invariably one of the first things developers notice when they begin using such tool is the larger than expected number of context switches. They are typically surprised at the amount of time these switches consume and they then can modify their programs to use a smaller number of priorities so that context switches occur less frequently. Notably, without a tool such as TraceX, developers would be unaware of many of the inefficiencies in their system. ■

New approach to combining open source and proprietary software models

By Lawrence Rosen, Rosenlaw & Einschlag

QNX has introduced a new hybrid software model in an effort to improve how embedded software is developed and distributed. The aim is to provide the benefits of open source while allowing embedded developers to profit from their derivative works.



The Foundry27 web portal allows developers to participate in the development of QNX products and to share code with other community members.

■ Today the rate of change to software and hardware is so rapid, and software so complex, that vendors and customers alike struggle to keep up. Often, software vendors are their own worst bottleneck, as they work to enhance their existing products while also attempting to satisfy new, and often divergent, customer needs. Meanwhile, the embedded developers who use these software products often know exactly what features they want; many would make the modifications themselves if allowed to do so. Many of these developers would also welcome opportunities to share the results of their development efforts — just as they would in an open source project.

For me, open source is usually part of the answer. I have been involved for a long time in the open source community, and have served as a lawyer for organizations like the Open Source Initiative and the Apache Foundation. The goals of open source — built upon licences that promise freedom to use, copy, modify, and distribute software — are part of my nature. They are also becoming part of the nature of the entire software industry.

However, a pure open source approach does not work in all cases. A case in point: QNX Software Systems. Like many commercial software vendors, QNX does not believe that relinquishing all control over its intellectual property and giving it away for free would best serve the interests of its customers. Thus the company has introduced a new software model that integrates open source and proprietary software products in new ways. This model represents a step forward toward openness of embedded software development, and it gives customers significantly greater flexibility to adapt QNX technology for their own purposes.

Technology companies implement their fundamental business strategies through licensing their intellectual property. It is a subtle task. If a company gives too much away through overly generous grants of copyrights or patents, then its competitors and customers get a free ride on its products. But if the company makes restrictions on use too tight and complicated, it discourages customers from taking full advantage of its products. This is where QNX is looking to innovate, with a new blend of transparent development and accessible licences for embedded developers. It is an enablement strategy that combines the benefits of an open source development model with the sustainability of a royalty-based business model for commercial projects.

QNX already serves as a major contributor to open source software projects, including the Eclipse C/C++ Developer Tools (CDT), which is based on code that QNX donated to Eclipse. In fact, QNX runtime technology and development tools contain a variety of open source components, many of which are available for free use. The company has also released board support packages (BSPs) under the Apache Licence, Version 2.0 (Apache 2.0). This license gives developers the option of offering their derivative works — BSPs in this case — for free or for a fee. It does not force developers to publish their derivative source code, yet it provides a framework for open, cooperative development.

Meanwhile, key components of the copyrighted and patented technology at the heart of QNX runtime software remain available only to QNX licensees, as are certain value-added features of the QNX developer tools. Nothing in this new QNX business model changes that rule.

However — and this is what is new — the new model offers more visibility into the QNX development process and it grants developers more freedom to modify and share licensed copies of QNX software. Building on its Eclipse experience, the company has started to publish source code for key parts of its runtime products, including the QNX Neutrino microkernel, and will develop those products in the open, for anyone to follow. Non-commercial development licences for the full version of the QNX development suite, which includes the QNX

Momentics development tools and the QNX Neutrino RTOS, are available for free. Partner licenses are also available at no charge for anyone looking to add their products to the QNX ecosystem.

The company is, in effect, creating an open source community within its community of RTOS, middleware, and development tool licensees. As a result, anyone interested in QNX technology can now cooperate on development for the benefit of the community as a whole. At the same time, by publishing its QNX Neutrino RTOS source code, the company is inviting others to take QNX technology down new open or commercial development paths. It has even created opportunities so that commercial developers can implement QNX technologies on target operating systems other than the QNX Neutrino RTOS.

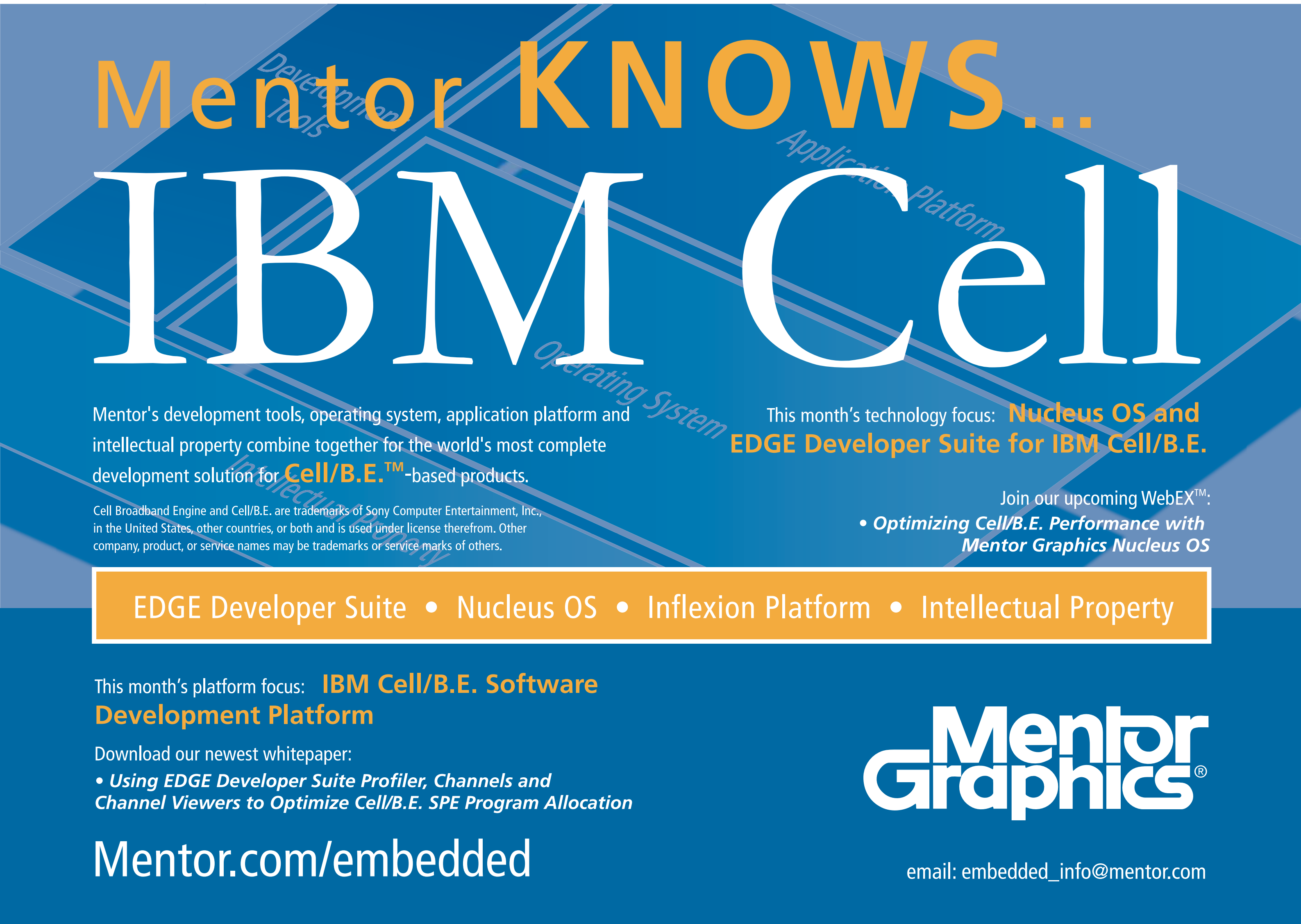
Traditional open source communities are, in some sense, open to anyone who wants to follow community rules of behavior and licensing. This QNX community is very similar, but the laws of intellectual property — and the limitations that the company places on the use of its copyrighted and patented software products — give this community more of a commercial feel. Anyone can join, and they can become QNX licensees (for free) as long as they promise not to license their QNX or derivative work software to third parties who are not also QNX licensees, or unless they get a commercial distribution licence from QNX.

This community consists only of QNX licensees. That is not open source, but it is a realistic modification of open source rules to create an open development community for QNX software, which is used in commercial products worldwide. Outside this community of licensees, QNX proprietary software is published but it is not open. Within the QNX community, developers enjoy the benefits they would find in an open source development environment while being able to leverage advantages associated with proprietary products.

Open source software thrives when a community of users and developers cooperates to develop new solutions for the entire community to share. To help encourage the growth of such a community, QNX is launching a web portal called Foundry27. Using this portal, anyone can access information from the company and from others in the community about QNX products and services (including all published source code). As with most open source development projects, free registration is required to get write privileges for wikis and forums.

The company does not mandate that Foundry27 be the only development and distribution vehicle for QNX-related products. Licensees may participate in other academic development labs or commercial and non-commercial projects, as long as all the participants are themselves licensed to use the QNX development suite. Coordination at the development portal is encouraged but not required.

By downloading the development suite and applying for a license key, developers will gain access to a free copy of most QNX development tools and runtime software — including repository download rights for published source code. Depending upon the software licence(s) that they qualify for, developers can use that software to prototype target systems, extend hardware support for the QNX Neutrino RTOS, develop new applications, and many other purposes. Members of the QNX community can also exchange their software solutions with any other QNX licensees, as specified in the new software licences.



The new hybrid software model divides software products into three classes: 1. A small (and growing smaller) class of patented or copyrighted proprietary software that is based on unpublished source code. Soon this will be limited to certain value-added tools and some middleware products. 2. A large (and growing larger) class of published source code for proprietary software components that are available for the creation and sharing of derivative works. 3. A large (and hopefully growing much larger) collection of source code published under open source licence terms, or that has been made available for free from other members of the QNX community.

Deciding what software goes into what class is a balancing act. If the company claims too many intellectual property rights, it will limit the ultimate success of the community that it hopes to empower. The balance will be maintained by the commitment to publish more and more of its software over time, and by its promise to allow its customers and the development community greater licensing freedom with QNX software. QNX Neutrino RTOS runtime technologies and the QNX Momentics development tools are not open source in the way that the open source definition requires, and do not claim to be. But the QNX approach to enabling the sharing of derivative works within the community is open source, and is familiar to anyone who has used open source software. This shared source code will help developers create new applications and share them with others. It will also help companies that build target systems, and companies that create new tools for the QNX development suite, to make even more capable products.

When developers download the QNX Momentics development suite, they can choose from one of three QNX licences, the first two of which are free of charge. Non-commercial end users — Licensees may receive the QNX development suite, which includes the QNX Momentics development tools and the QNX Neutrino RTOS software, under a royalty-free Non-Commercial End User Licence Agreement (EULA), for certain evaluation and limited development purposes, including prototyping. The EULA is also designed for academic faculty to train their students. Community partners — for companies who wish to offer their own products and services to QNX customers, the company now offers its technology partners the Partner Software Licence Agreement (PSLA) at no charge. Commercial customers — for companies that create commercial products with QNX Neutrino RTOS software, the company provides the Commercial Software Licence Agreement (CSLA).

This licence is not free. It includes important warranties and indemnities appropriate for commercial software. Licensees will need to execute a separate OEM Licence Agreement or Runtime Licence Agreement to manufacture and distribute target systems that embed the QNX Neutrino RTOS software. All these licences allow developers to develop derivative works of QNX software that can be distributed to other licensees, and they enable a community where each developer can benefit from the efforts of other developers. ■

Product News

■ Vector: easy optimization of ECU-specific parameters

The new version 6.5 of the CANape measurement and calibration tool from Vector simplifies optimization of ECU-specific parameters in vehicles. In addition to extended bus and protocol support for FlexRay, ECU developers also benefit from convenient display of Simulink software models that are used. Direct access to internal parameters of FlexRay ECUs via XCP on FlexRay can now also be obtained with PDU support based on an OEM-specific description file.

News ID 676

■ pls and HighTec: tool chain for TriCore derivatives

pls and HighTec announced a complete development environment for Infineon's TriCore microcontroller family and the TC1796 Starter Kit "EasyRunTC1796". Essential component parts of the "Tricore System Development Platform", which was designed particularly for industrial applications, are the Code::Blocks IDE and the already established platform independent TriCore compiler system, based on GNU technology, from HighTec.

News ID 822

■ Lauterbach: debugging tool supports Freescale i.MX27

TRACE32 PowerTools of Lauterbach now support the Freescale i.MX27. The i.MX27 is a high-performance, low-power chip processor, which is based on a 400-MHz ARM926EJ-S core. The i.MX27 can be used in DVD players, digital cameras, and other portable multimedia devices. The new TRACE32 PowerTools have been enhanced with new processor architecture for Freescale i.MX27 to support efficient debugging at C and C++ levels over the chip-integrated debugging interface.

News ID 687

■ Great Western: RTOS for for Microchip 16-bit MCUs

Great Western Microsystems has been appointed by AVIX-RT as it's European distributor for the new AVIX-RT RTOS, specifically developed for Microchip sixteen bit microcontrollers of PIC24 and dsPIC families. The AVIX RTOS is available for purchase from The Debug Store, the Online store of Great Western Microsystems.

News ID 789

■ SMSC: enhanced suite of MOST network analysis tools

SMSC has enhanced its OptoLyzerG2 30xx family, the Network Analysis platform for MOST25 and MOST50 networks. Various new features are now available with the new V1.4.0 release of the OptoLyzer Suite's graphical user interface. An improved file feeder enhances recorded trace file analysis. For example, a trace file can now be imported into the OptoLyzer Suite and High Level Protocols such as AMS, MOST High and MAMAC are now supported in this use case. In addition, the data can be fed into the Viewer, Recorder, Graph and Watch components.

News ID 677

■ Wind River: Linux platform comes with 64-bit application support

Wind River announces the availability of the next generation of Wind River Linux, based on the 2.6.21 Linux kernel. Wind River will provide users with 64-bit application support, including tools for both kernel and user space debugging across all supported architectures. Wind River Linux will include an advanced cross-build system that incorporates a structured framework for managing device software components as independent 'layers.'

News ID 752

■ NI: library offers new edge-detection algorithms

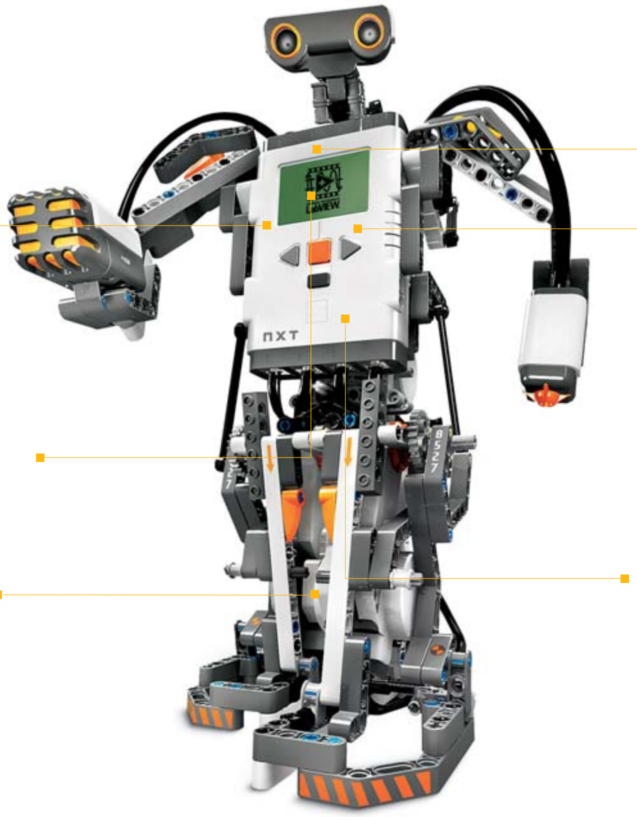
National Instruments announces the newest version of the NI Vision Development Module, its library of image processing and machine vision functions for multiple programming languages, including NI LabVIEW and LabWindows/CVI as well as Microsoft C, C++, Visual Basic and .NET.

News ID 768

NEW VERSION AVAILABLE

NI LabVIEW.

Limited Only by Your Imagination.



Build and program robots with LEGO® MINDSTORMS® NXT using software powered by NI LabVIEW

Communicate via multiple protocols including Bluetooth

Graphically program concurrent, real-time applications

Develop your human machine interface (HMI) display

Independently control multiple servo motors

Target 32-bit microprocessors and FPGAs

Real-Time and Embedded | Signal Processing | High-Performance Test | Industrial Control

PRODUCT PLATFORM

LabVIEW Real-Time Module

LabVIEW FPGA Module

LabVIEW Microprocessor SDK

NI CompactRIO Embedded Hardware Platform

When the LEGO Group needed parallel programming and motor control tools intuitive enough for children, it selected graphical software powered by NI LabVIEW. With LabVIEW graphical system design, domain experts can quickly develop complex, embedded real-time systems with FPGAs, DSPs, and microprocessors.

>> Expand your imagination with technical resources at **ni.com/imagine**

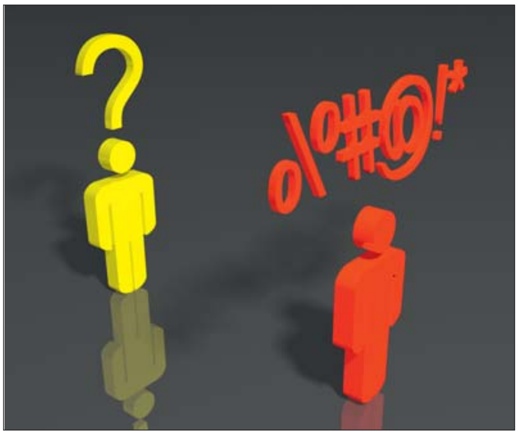
Austria	43 662 457990-0
Belgium	32 (0) 2 757 0020
Czech Republic	420 224 235 774
Denmark	45 45 76 26 00
Finland	358 (0) 9 725 72511
France	33 (0) 1 57 66 24 24
Germany	49 89 7413130
Hungary	36 23 501 580
Ireland	353 (0) 1867 4374
Israel	972 3 6393737
Italy	39 02 41309277
Netherlands	31 (0) 348 433 466
Norway	47 (0) 66 90 76 60
Poland	48 22 3390150
Portugal	351 210 311 210
Russia	7 495 783 6851
Slovenia/Croatia	386 3 425 42 00
Spain	34 (91) 640 0085
Sweden	46 (0) 8 587 895 00
Switzerland	41 56 2005151
UK	44 (0) 1635 523545

NATIONAL INSTRUMENTS™

Common misconceptions about open source software

By Roland Stigge, Philosys

Open source software is surrounded by many misconceptions relating to issues such as cost, licensing, economics, scalability, community, warranty, support, quality, free software, distribution, compatibility, security, source code, profit, business strategy, volunteers, control, copyright and freedom.Many common misconceptions about open source software are identified below. All are basically wrong, for reasons which are explained in this article.



1. "Open source software is gratis."
While open source software is considered as free in the free speech sense, it is not necessarily free in the free beer sense. As with proprietary software, there is a cost to the production. At this point, there is no difference to traditional development processes. It is true that you can download many open source software packages for free, but there will be production costs for to newly developed parts of the software. Do not expect open source developers to work for you gratis. You can just expect that when they already have some part of the software ready, they will possibly ship it under an open source licence. You can just hire open source developers to work on a special project to complete certain features. Technically, there is no difference from the traditional/proprietary software development process since there is a requirements definition, analysis, development/ implementation, testing, deployment, even warranty as stated in the contract.

2. "You don't need a licence for open source software, it is in the public domain."
While software which is in the public domain certainly qualifies as open source, most open source software is currently not in the public domain. It is actually copyrighted and your right to use, modify and (re-)distribute it comes from a licence. Typical (and according to the open source definition) officially approved open source licences can be found on the respective open source initiatives page.

3. "Open source software is the opposite of commercial software."
The opposite of open source software is proprietary software. Open source software can be commercial, as seen with the big projects like Linux where millions of copies are sold on the server, desktop and embedded market. So there is no contradiction between open source and commercial software. Although there might be a contradiction between actual proprietary and open source marketing strategies.

4. "There is a difference, to the licences and technical development, between commercial and community open source projects."
While different projects might have different characteristics in their development and marketing procedures (e.g., developed mainly in-house in a closed organisation or admitting official committers from outside), their open-source-ness is defined by the licence which should be one of the OSI-certified licences for it to count as an open source project. It is true that some commercial vendors provide commercial versions of their projects (or open-source effectively just a part of the project under an open source licence) which are proprietary or at least not really open source according to the definition (licence etc.). This cannot seriously be considered as open source. In fact, it is only a way to fool potential clients into the open source hype. It still leads to vendor lock-in and all the problems that open source wanted to prevent in the first place. It is also true that projects can be quite different in planning, development and day-to-day community involvement. But the outcome, a product that is shipped under an open source licence that enables you to use, modify and re-distribute the package, is always the same.

5. "There is no warranty to open source software."
While typical open source licences include warranty disclaimers, it is possible to get warranty by contract with the software author or a separate support company And yes, this probably won't be gratis. The reason for the typical open source licence to disclaim warranty is its freeness: consider a consumer who would get warranty with the open source licence. The licence would allow him to redistribute the software package wherever he wants under the same licence. The problem of warranty fulfilment, especially the legal and financial aspects, potentially to the rest of the world, is often not possible for the producer, even for large companies. But by additional contracts, companies will sell you warranty. It is even normal that for contractual software development, the producer must provide a certain minimum warranty.

2007 Portable Power Design Seminar

5 locations in Europe Please register today at a location near you.

Schedule

Dec 3rd	Munich	Germany
Dec 4th	Paris	France
Dec 5th	Birmingham	UK
Dec 6th	Copenhagen	Denmark
Dec 7th	Eindhoven	Netherlands

To register, and for additional information, visit

www.ti.com/portable-power-ece

HIGH-PERFORMANCE ANALOG >> YOUR WAY™

Technology for Innovators™

Texas Instruments

Misconception 1: Open source software is gratis

This does not influence the regular open source licence.

6. "There is no support available for open source software."

As with warranty, you can easily get a contract from software vendors and support companies. It is even typical for commercial open source companies to sell support contracts, which are an essential income source. Examples are the big Linux distribution vendors like Novell, RedHat and Canonical.

7. "Open source has bad quality. It can't be good if it is gratis."

This is the typical attitude from the old times where cost and quality where directly related. This connection is completely gone with open source development models. It does not mean that good quality is an intrinsic feature of all open source projects, but there is some evidence to the tendency for good quality in famous open source projects. The bigger and more famous the project is, the more distributions (Linux or other) include it and the more users will know it. These highly tested programs (like Linux and Apache) tend to be impressively stable, and when problems occur, they tend to be fixed within only a few days (sometimes, reportedly, within minutes!). Open source software can be, and in the widely distributed cases must be, developed in a peer review fashion, as in science. This cannot be achieved in typical proprietary environments where only a small number of people can review the source code. Further, quality assurance measures from traditional software development processes can be applied to open source projects as well. And for the big open source projects like Linux, Perl, Debian etc., it has been done for many years.

8. "There is a difference between free software and open source."

There are slight differences between the free software definition and the open source definition. While the Free Software Foundation focuses on the simplicity of the definition (using, distributing, modifying and distributing changes), the Open Source Initiative elaborates further on political aspects like the prevention of discrimination of persons, groups or fields of endeavour. But finally, the most important points of both views are the same. For software to be considered open source or free software, it must be licensed under a certain licence, e.g. GPL, LGPL, BSD or Artistic Licence. Since most software packages are licensed under one of these common licences, they are automatically considered both open source and free software.

9. "Open source is only a small part of the whole software industry."

This is a typical proprietary-software-centered view with measures from the respective business models. If you count the volume of sold software licences, proprietary software will probably have the biggest share (with open source lacking accounting in terms of money and number of licences). But since open source software licences are typically not sold like proprietary ones, this kind of measurement cannot be employed here. The software market is actually highly penetrated by open source software, with open source products often being first choice because they are the industry standard. Famous examples are BIND (DNS server), Apache (web server), and the Java, Perl and Python programming languages.

There is also no lack of open source projects for all kinds of tasks. At sourceforge.net, there are currently registered more than 155,000 projects (not all of which are assumed to be active right now, but one can get an idea of the dimension). The Debian GNU/Linux distribution contains more than 11,500 software packages which are maintained and usable right now. Even more traditional software vendors embrace the open source development and distribution process. For example, MacOS X is partially based on open source (Darwin, Mach, BSD, etc.) and even Microsoft is in the process of registering their own open source licences at the Open Source Initiative. All this can be considered as some evidence that open source is not just a small part of this still fast growing industry.

10. "Open source software is made by volunteers, being students or working in their spare time, they are not paid for their open source work."

In fact, in many open source projects, there is only a small percentage of committers who are completely volunteers, as seen in the Linux and Apache projects.

11. "Open source software comes from universities or private entities."

In the early years (roughly before 2000), universities, students and volunteers were heavily involved in open source projects. This has changed in recent years. There are several reasons for this. Universities realised the possibility of making a profit from their work, mainly by patents, licences, co-operations with commercial entities. Therefore, the pressure not to provide open source code from the results found in scientific processes increased. At the same time, project members left university and already knew how advantageous working with open source software could be and founded their own projects, often done in startup companies like Thawte. Now, the software industry actually drives a big share of the open source development itself as seen in project participant lists of the famous open source projects.

12. "You can buy open source licences for using it per CPU."

Famous examples of companies who would like

Application Type	Example Package(s)
Operating System	GNU/Linux, FreeBSD, NetBSD, OpenBSD
Graphical User Interface	X.org, GNOME
Word processing	OpenOffice.org, LaTeX, DocBookXML
Audio	Audacity, Ardour, Hydrogen
Video	Cinelerra, MPlayer
Web Server	Apache
Web Browser	Firefox
Email Client	Thunderbird, Evolution
Graphics	Inkscape, Gimp, Dia

to give you this impression are vendors like RedHad, Novell and MontaVista. They sell perseat licences that seem to prevent you from using a big part of an open source operating system on more systems than you have paid for. Internally, most of these systems are open source software that you can actually use and share freely. Just make sure not to be confused by a complicated system with added components that are not open source. The real open source part of these products is licensed to be freely redistributable, and reusable.

13. "When we open-source our software, we do not have control of our copyright or we lose our copyright."

You still retain your copyright. Actually, with most open source licences, you need your copyright to be able to assign it to the work under consideration. Then, you can provide the software under the protection of the respective licence. What you do not have is control about what people will do with your software. But with proprietary software you also won't have this privilege. Furthermore, you won't have control over redistribution of the software. But your copyright stays assigned and keeps others from distributing the software under other licences.

14. "The main point of the open source concept is the publication of the source code."

At this point, the name open source is a bit misleading, although not wrong. It emphasises the source code aspect. Other main aspects of open source licences are the freedom to use, modify, distribute (even derivatives) of the software freely. There are even more political aspects to it, e.g. the prevention of discrimination of persons, groups or fields of endeavour (i.e. you cannot restrict the software from being used in certain areas, e.g. warfare). This means that the availability of source code is just one of several important points that must be fulfilled for a work being regarded as open source.

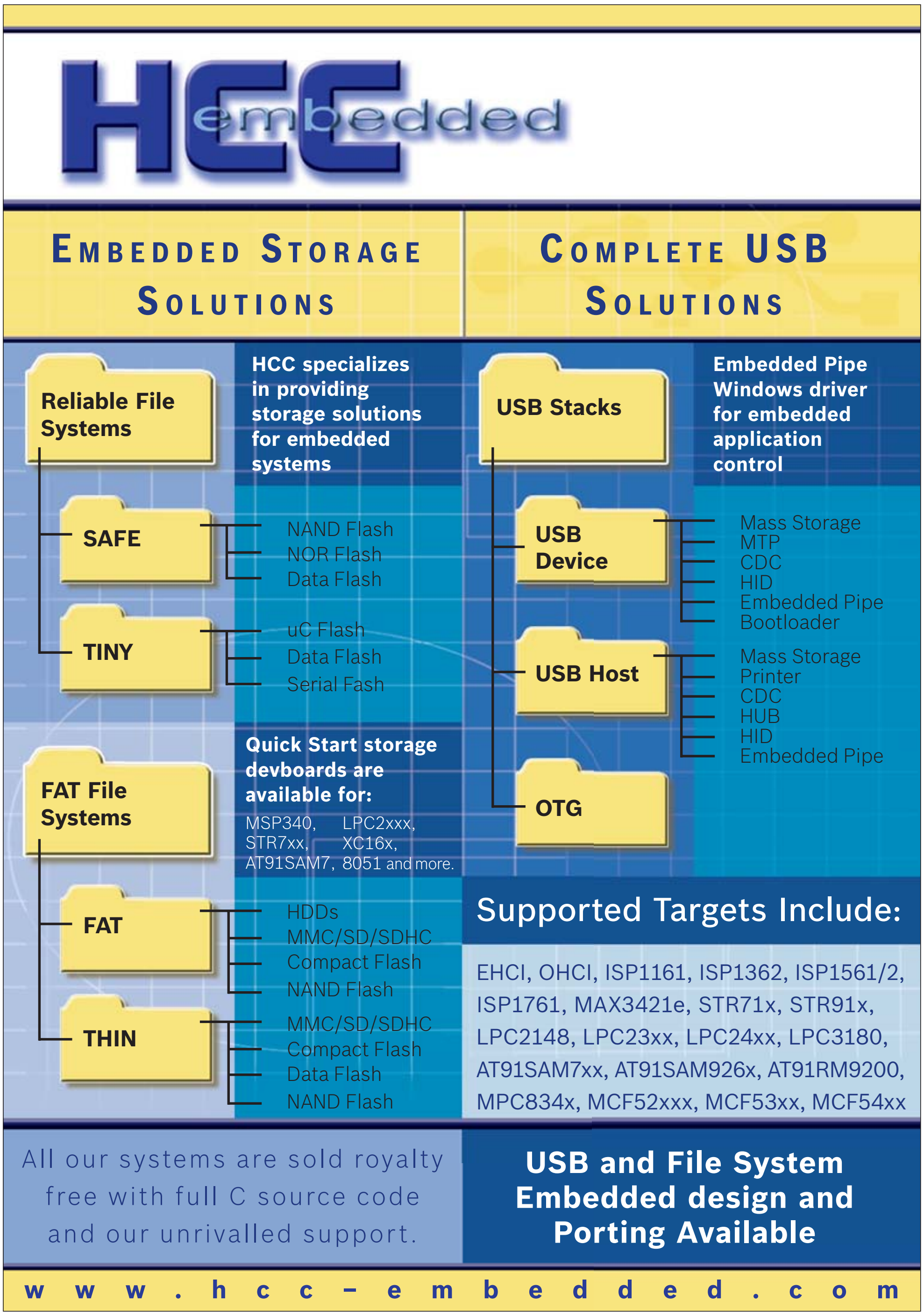
15. "Open source software is less secure."

This view is based on certain assumptions, which are probably wrong. First, there were many publicly announced security vulnerabilities for open source software in the past. This could be taken as a problem of openness. If you disclose your source code, problems, and vulnerabilities, can be found much more easily. This is correct. And it is considered a feature of open source workflows. It is the same peer review concept as is known in the sciences. This way, bugs and problems in the software can easily be identified and fixed, even without the original author or vendor. The latter is not possible in proprietary software. On the other hand, this does not mean that there are no bugs and actual or potential vulnerabilities in proprietary software, or that these are not found and exploited. Vulnerabilities in proprietary software will also be disclosed, they will not also stay proprietary.

A typical related idea from the traditional proprietary world is that if you keep your source code secret, it will be more secure since others cannot see the problems with it. This concept is also known as security by obscurity. And it is just wrong. It is just more difficult because software can be analysed, reverse engineered, etc, even without the source code. Already some years ago, a famous security expert, Bruce Schneier, stated it like this: " … In the cryptography world, we consider open source necessary for good security; we have for decades…." in Crypto-Gram, September 15, 1999.

16. "Open source programs are not compatible."

While this concern is typically raised for certain applications and protocols in comparisons of open source and proprietary versions, mostly the opposite is the case: open source has a tendency to obey open standards (or create such). The proprietary world has interest in locking customers into using certain products by certain companies to protect revenues. Therefore, they have no interest in making their software compatible by default, e.g. using open standards that other software (including open source soft-



It is possible to get warranty by contract with the software author or a separate support company

ware but also other competition) could use, or disclosing file formats and protocols.

17. "Open source software is not ready for use in many areas."

While some years ago this was certainly the case where the user and developer base for general purpose open source systems was smaller, this is not necessarily true anymore, although this concern is repeated still. Some open source examples for typical software packages are:

This is just a small current excerpt. For other applications, see the big open source software collections and sites like Sourceforge.net, Freshmeat and Debian.

18. "Making our software open source would decrease our profit."

This would be true if you base your revenue solely on software licences. While this is a problem for a few big companies in the market, this does not apply to many companies, since their core business is not copying and selling their own software but something else. Some examples follow.

If you are developing (and hopefully shipping!) a hardware product which contains software, it is not necessarily a problem for you to disclose the source code. You often do not need to change most of the open source products you are including (for free!), and the parts that you do change are adaptations to your special needs in your hardware. Sometimes this software will be driving the parts in the hardware that are a competitive advantage for you. But since you are not forced to also disclose the complete hardware wiring diagrams and design documents etc, the competition will probably not have it too easy to copy your product and take advantage of your investment. Remember that there are still the respective intellectual property laws like copyright and patents for your hardware. Also, do not forget that you are not automatically forced to disclose all the software components inside your hardware as open source. User interfaces and stand alone running software modules are common examples where you are not forced to. Another example is the case where you are basically using open source software. This is also true for people and companies that consider themselves as developers. For example, if you are running some kind of analyst company running and developing software that supports you doing your job, your revenue comes from selling your expertise, the results of your research or other services. And it can stay this way, no matter if with proprietary or open source concepts.

Even typical software developing companies can open-source their software (like, for example Alfresco). The software will be developed and deployed in customer relations like in the traditional market. The open source concept is not necessarily a problem here, but it keeps you well known in the community. Customers need specially adapted solutions. They need warranty contracts, support, and training. They won't get it from the plain open source package. The best way to get it is from you!

19. "Open source components can be integrated freely into a project."

While this can be true under certain circumstances, this can be applied only if your project follows certain preconditions. Of course, you need to obey the respective licences. And if your project mainly consists of open source components and distribution channels, this will not generally be a problem. But you can get into trouble if you mix proprietary and open source projects, linking your proprietary software with open source components where it is not allowed. For example, using GPL libraries (e.g. readline) enforces that the programs using the libraries need to have a licence compatible with this licence. You can either choose to open-source your respective program, too, or use other (open source) components instead (like, for example, LGPL'ed libraries). Also, keep in mind that in some cases, open source licences are incompatible, like for example GPL code linked to OpenSSL licensed code. Another example would be a pseudo open source licence that prevents you from using your software in certain areas like commerce. The latter would not be considered open source regarding the definition, but before using it make sure you can follow the licence terms.

20. "Open source is still a new concept in software development."

While the main part of open source software has certainly been developed from the 1990s on, the concept reaches back to previous decades (even centuries). While Richard Stallman started to develop the GNU operating system (under the GPL) at the beginning of the 80s, open sources attitudes could be found at the very beginning of large scale computing, at the latest in about 1960. Going proprietary was not a concept of software developers, and selling software as such was not a big business in the earlier decades of computing. Computers were the selling objects. Open source is a concept stolen from environments where peer review has always been an integral part of the workflow. As in the sciences. Actually, software can be considered a special kind of mathematics.

21. "If we open-source our software, there will be volunteers doing all the work."

This will only be true if volunteers are actually interested in your previous work and they can extend it while benefiting from it. Considering the already available software base in the open source world with software packages for all kinds of use, this is not necessarily the case if you have a special niche database product or graphic bitmap viewing software and now consider to open-source it. Although it might be advantageous for you anyway: people will see your expertise in a certain field and come back to you if you or your software could be helpful.

22. "The missing central control in open source projects makes scaling to large project sizes impossible."

Actually, some of the biggest software projects are open source. For example, Linux (kernel) and Debian Operating System projects include millions of lines of code, and multiple gigabytes of source code that interact successfully. Those projects have found ways to develop and progress. Certainly, this is even difficult with central control. Key points of large open source projects are distributed and networked development with massive parallel work. In the two given examples, there is a central repository available to get the software from, but actual development is not necessarily only checked into a central source code repository before the respective change is accepted. This can be done with separate subprojects, distributed branching (revision control systems like git, arch, darcs, bzr etc.) and derived projects (these so called "forks" are a kind of separately maintained branch).

23. "Open source licences are viral. If I incorporate open source code into my project, the whole project and derived works must be distributed under this licence also."

First, not all open source licences include the concept of forcing derived works to be licensed likewise. While the GPL forces derived works also to be distributed under the same licence, other licences like the BSD licence do not do this. It is a matter of choosing which project (and the associated licence) to draw code from. In the case of GPL-like viral clauses, other code in the project under consideration may just not be suitable to be distributed under the GPL (e.g., a proprietary licensed part). Here, even the GPL cannot force you to distribute the proprietary code under the GPL. But it can prevent you from conveying the resulting code (GPL-licensed code together with proprietary) at all.

You need to choose from different options. First, it is sometimes possible to approach the copyright holder(s) of the GPL-licensed code to convey the code (additionally) under an alternative licence which is compatible with proprietary code. Another option would be to license the proprietary code under the GPL which would often include a strategy change in the respective intellectual property department, or asking the copyright holder in the case of external code contributions. Sometimes, a compromise with re-licensing the whole project under a more liberal open source license, like BSD, is the only option left (except the prospect of not conveying the work at all).

Technically, there are different cases that either require licensing of derived works likewise, or not. For example, linking tightly related code together will be a clear derivation in the GPL sense. But sometimes, code is coupled less. For example the Linux kernel is licensed under the GPL version 2, but it does not require programs that are run under the control of the kernel to be licensed under the GPL also. For kernel modules (mostly device drivers), it depends on internal details. They can be licensed under a non-GPL licence but most often they are explicitly utilising parts of the kernel that only allow GPL licensed code to make use of them. This way, kernel developers are trying to force device driver authors to disclose their source code under the GPL. ■

Product News

■ Vector offers extensive FlexRay functional library

Vector Informatik extends its XL-Driver-Library for the automotive bus systems CAN, LIN and MOST by adding a FlexRay functional library. Furthermore, the Advanced FlexRay-Driver-Library offers extended functions that allow developers of FlexRay applications to utilize Vector's high-performance FlexRay interfaces in their own applications. Effective immediately, standard functions for FlexRay are now included as a component of the XL-Driver-Library for FlexRay VN3300 hardware with PCI interface and VN3600 hardware with USB interface. They enable users to send up to 128 independent frames, send in single-shot or periodic mode, receive data, null frames and error frames and so on.

News ID 777

■ pls: debug tool for Freescale's i.MX31

In addition to debug tools for various ARM7 and ARM9 derivatives, pls Programmierbare Logik & Systeme now presents the Universal Debug Engine 2.1 for Freescale's i.MX31 Multimedia Applications Processor, which is based on a standard ARM1136JF-S core. The intuitive and configurable user interface of the UDE 2.1 provides i.MX31 users with unrestricted C/C++ support, a symbol browser, freely configurable toolbars, context related menus and HTML as description language for user specific windows. The use of standard script languages guarantees a high level of automation.

News ID 788

■ Coverity: new technique of source code analysis

Coverity announces a software analysis engine based on Boolean satisfiability and will enable multiple solvers to identify software defects. This new technique of source code analysis is made possible by patent-pending technology from Coverity that creates a bit-accurate representation of a software system, where every relevant software operation is translated into Boolean values (true and false) and Boolean operators (such as and, not, or). This bit-accurate representation enables source code to be analyzed by SAT-based Solvers.

News ID 796

■ Express Logic: embedded tool for ThreadX system event analysis

Express Logic announces the release of TraceX, its first host-based embedded development tool. TraceX enables embedded developers to visualize and better understand the behavior of their real-time systems. Designed to work with Express Logic's ThreadX RTOS, TraceX collects a database of system and application 'events' on the target system during run-time.

News ID 729

■ Green Hills expands protocol options of platform for Industrial Safety

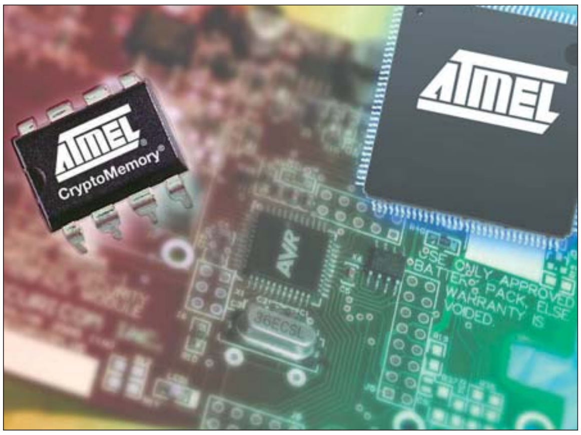
Green Hills and IXXAT have announced their collaboration to add support for CANopen, IEEE 1588 and EtherNet/IP protocols to Green Hills Software's Platform for Industrial Safety. As the backbone for communications in many industrial, medical and automotive devices today, these protocols can now execute safely and reliably when combined with Green Hills Software's RTOSes that are well known for their pedigree and experience for use in the safety and security markets.

News ID 798

Affordable embedded security with cryptographic memories

By Eustace Asanghanwa, Atmel

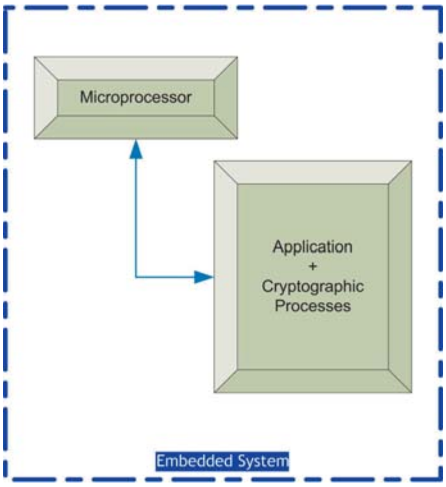
This article looks at security in embedded systems, reviews the trends characterised by trade-offs between security and cost, and introduces cryptographic memories, an innovative technology for embedded security at low cost.



■ Achieving adequate security in any system is a challenge, but more so for embedded systems. This is because modern security algorithms were developed for non-embedded systems and developed to address specific threats. As a result, many systems would deploy multiple algorithms at the same time to manage different threats. For example, during an e-commerce transaction, the PC would typically use a slower but more fitting public key algorithm like RSA to exchange encryption keys, and then use a much faster stream encryption algorithm like the Advanced Encryption Standard (AES) to encrypt the actual data. The PC is able to do this because it has with large amounts of memory to accommodate both algorithms as well as a plethora of others. Storage resources are scarce and very expensive in embedded systems, typically allowing room for just a single cryptographic algorithm thus requiring careful thought for securing the system.

Embedded systems have witnessed tremendous growth in complexity in the past two decades. Complex embedded systems today are found in life-critical applications like precision medical imaging in surgery, radar-based anti-collision systems in automobiles, aircraft control systems, missile guidance systems, and power distribution grids control. This is a long way from the last two decades, when embedded systems only performed such simple tasks as storage of subscriber codes in cable TV set-top boxes. Alongside the growth in complexity has been growth in competition among embedded systems manufacturers. Most markets for embedded systems, like defense and medicine, have just a few players in fierce competition with each other. Competition imposes downward pressures on price, thereby creating sensitivity to inclusion of such costly and extraneous features as security that do not directly contribute to the final application. However, the question of security in modern-day embedded systems is so paramount that omission is bound to lead to disastrous consequences.

Many embedded systems like precision medical imaging that use sophisticated algorithms, need security to protect the investment of the developer and also to protect the safety of the end application. The underlying intellectual property (IP) of some embedded systems cost millions of euros and many years of engineering. Without proper protection, a counterfeiter who can gain access to the IP that enables such systems can offer clones at a fraction of the price required to capture the R&D investment of the originating developer. Worse that this, the originating developer assumes the liability of low-quality clones. In addition, embedded systems, especially those for life-critical applications, require high levels of security during operation. Imagine the disaster that can result to a community from malicious interception and alteration of remote commands to its power grid embedded controller, or to passengers from unauthorized loading of the wrong firmware in an aircraft's embedded control system. The stakeholders for security embedded systems, therefore, are not only the developers of the systems but potentially everybody. The value at stake with developing or deploying embedded systems is so great that security is not optional.



A secure microcontroller shares processing power between actual application and cryptography.

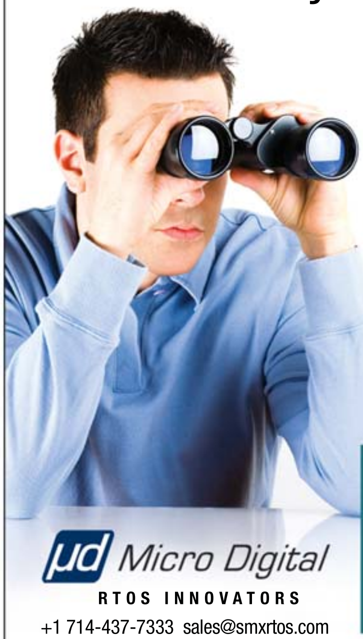
It is clear that security for most embedded systems is not an option, that embedded systems developers need to protect their investment against counterfeiters, and everybody else needs protection from a few malevolent individuals. What does the security really entail? Are there certain features to look for? For embedded systems developers, security entails making sure that only they or authorized parties manufacture the system. This way, not only do they get the chance to recapture R&D costs but they also get a chance to make sure that the end product is of proper quality.

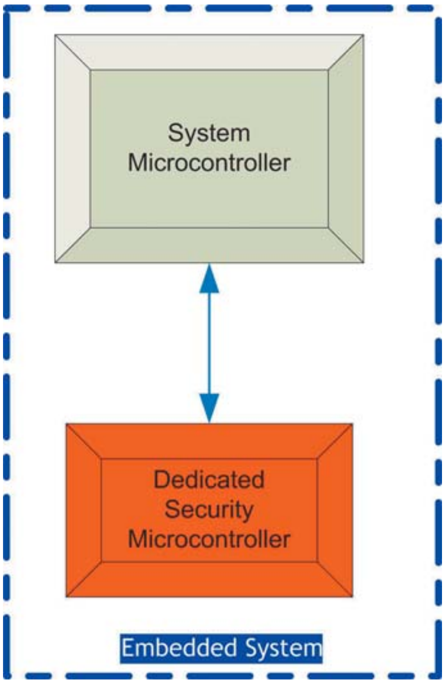
For everybody else (including the developers who want to avoid liabilities), this means making sure that the embedded system is of proper quality. It must be accessible only by trusted hosts (authenticity). For systems that operate remotely, all commands must come from trusted sources (authentication). The commands must not have been modified in transit (message integrity). The sender of the command must not later be able to claim they did not send the command (non-repudiation). Nobody must be able to access the meaning of sensitive data through encrypting it (confidentiality). It is understandable that not every embedded system will require these security features, for example a well-contained non-remote system may not require that commands be encrypted from a trusted host. However, proper security for embedded systems requires protection against all potential points of attack, meaning implementation of all applicable features.

As embedded systems have evolved in complexity over the past two decades and continue to evolve, so do embedded security strategies. Throughout this evolution, one characteristic aspect remains constant: the need to balance the level of security with deployment costs. The proper security tends not to be economically feasible in deployment thus calling for a trade-off between security and cost. With the value to protect increasing from embedded systems complexity, embedded security has evolved through several strategies. One must note here that evolution into new security strategies does not necessarily mean obsolescence of the existing strategies but an addition to available security options. Thus the security strategies in the trends discussed in this section still remain viable options.

In the 1980s, confronted by the problem of cable TV programming theft, the cable industry resorted to storing subscriber codes in smartcards instead of set-top boxes. This was an enhancement in both security and cost, in that the companies did not have to replace compromised set-top boxes as had formerly been necessary, and could change smartcards whenever they suspected security breaches. In fact, the plan was to reissue smartcards periodically, as often as every six months. The economics of replacing the cards became prohibitive. This security model did little to address any of aspects of a proper security solution except to present a slight difficulty to the hacker. Although cable TV companies still use smartcards today, the security model has improved tremendously to include full cryptographic processes.

Confronted by the problem of cheap clones, some manufacturers have resorted to obfuscation strategies to protect their embedded systems. Obfuscation methods range from strategies like storing pieces of value IP algorithms into different segments of the same memory, or into different memories altogether, to completely erasing the markings of electronic components of the embedded system in an attempt to make it difficult to guess the bill of materials. Unfortunately, fragmenting pieces of algorithms incurs engineering costs and introduces implementation risks. In addition, coun-





A dedicated secure microcontroller is an additional microcontroller for security, which adds cost.

terfeiters always find their ways around these strategies when there is profit potential. Obfuscation is like hiding candy from a child in an unlocked drawer and hoping the child will never open the drawer. It offers no particular security.

Some developers adopt a secure microcontroller as the core processing unit in their system. Secure microcontrollers have built-in cryptographic algorithms and their designs are hardened against physical attacks. Depending on the type and number of built-in cryptographic algorithms, embedded systems with secure microcontrollers offer the ability for authentication, data integrity, confidentiality, and non-repudiation, essentially all the features of a proper embedded security. The downside of using secure microcontrollers is that the cryptography takes up processing power, which detracts from the main application. Time-critical applications, like some medical images, cannot afford to give up such processing power. Alternatives are faster, but costlier, secure microcontrollers.

Instead of sharing processing resources from one microprocessor between application and cryptographic processes, some embedded systems incorporate dedicated secure microcontrollers to perform the cryptographic process. Such multi-microcontroller systems provide the needed security but are cost-prohibitive, not to mention the design-in costs of a multiprocessor operating system.

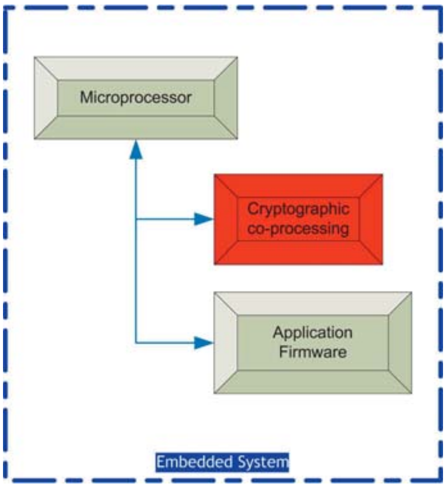
To mitigate the costs of using dedicated microcontrollers, some embedded systems resort to integrated cryptographic accelerators and co-processors. These are logic circuits within the microcontroller that perform specific cryptographic operations without taking up much microprocessor resources. These provide adequate embedded security at a high cost. Standard secure controllers with integrated security co-processors command a high price premium without promising optimality for the application.

If secure microcontrollers and standard microcontrollers with cryptographic co-processors do not provide the optimal solution at a competitive price, why not build a dedicated integrated circuit to perform both the application and security aspects of the processing unit in an embedded system? At least, this is the thinking behind the use of applicatio- specific integration circuits. The problem with this approach again is high cost. ASIC development takes over two years to the first prototype, after which it requires tremendous additional engineering efforts in characterisation and quality assurance to get to market. An ASIC solution would be ideal except that the cost is prohibitive.

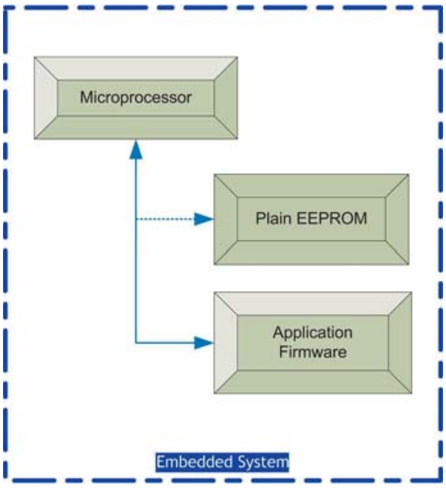
Confronted with the realities of deploying embedded securities, some embedded systems developers resort to software strategies. These entail using strong cryptographic algorithms like the Advance Encryption Standard (AES) to generate a cryptographic output called digests that are stored in a non-volatile memory like an EEPROM in the system. For instance, verification of the authenticity of a trusted host would entail generating the exact digest contained in the EEPROM. The attractiveness in this scheme lies in the low deployment costs. EEPROM memories cost about 5% of the cost of microcontrollers and require no special engineering or operating system software. The weakness in this scheme is that it does not provide the adequate security needed for embedded systems.

Contents of plain EEPROM memories can be compromised with sub-50-euro equipment. Implementers of this scheme are banking on the reputation of the cryptographic algorithms and fail to realise that the weak link in the security is the hardware itself and that the hacker sometimes does not need to decipher the digest in order to make use of it. A counterfeiter, for example, would simply copy the digests into millions of bogus systems he has made.

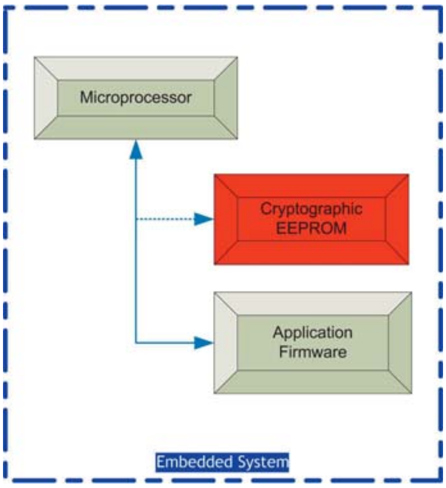
The trend in solutions for embedded security is a constant battle between adequate security and deployment costs. Fortunately, an emerging and innovative technology now provides proper embedded security at a manageable cost. This solution comes in the form of cryptographic memories. Some vendors have developed a new



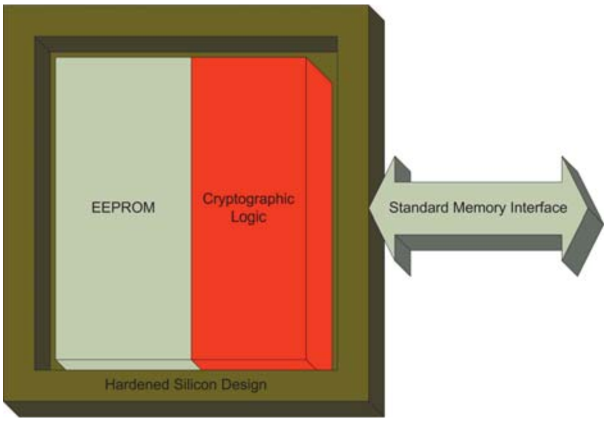
Cryptographic accelerators and co-processors add to microcontroller cost without promising optimality.



Plain EEPROM containers for cryptographic digests can circumvent security by simply reading and modifying EEPROM.



Cryptographic memory in an embedded system



Cryptographic memory results in non-volatile memory and cryptographic logic in a hardened design.

type of low-cost cryptographic memory that offers true hardware-based security for embedded systems. Cryptographic memories have a hardware-based cryptographic engine embedded in the silicon, plus multiple sets of cryptographically dynamic keys, passwords, and other physical strengthening attributes.

Cryptographic memories provide technology for an embedded system to clearly identify and be identified by a host (mutual authentication); to make sure that communication between the host and embedded system has not been modified (message integrity); to make sure that sensitive information is not viewed by unauthorised personnel during encryption (confidentiality); and to ensure that a message-issuing host can no longer claim they did not issue that message (non-repudiation). They also possess hardening features and environmental tamper monitors to withstand attacks including physical and systematic attacks. In addition, cryptographic memories provide anti-counterfeiting technology in a manner that even in the unlikely scenario of a compromised unit, the rest of the units still remain safe.

Furthermore, cryptographic memories offer technology for secure data storage where sensitive IP can be kept from prying eyes. In fact, cryptographic memories provide all the attributes of the proper security required in embedded system and only at the price range of an ordinary memory device. Moreover, since cryptographic memories offer just simple memory interfaces to embedded systems, integration is as simple as attaching a memory device with no need for additional operating system software or prototyping.

The security in cryptographic memories is determined by hardware inside the device and hardware-stored keys that generate unique cryptographic identities called cryptograms. The cryptograms are used by the device system to identify an authentic host and by the host to verify the embedded system authenticity. The keys used to create the cryptograms and other authenticating information are truly secret because they are set in hardware by the host device. Once set, fuses in the cryptographic memory are blown, rendering the keys unreadable – even by the host or device manufacturer. The authenticating information on the cryptographic memory, therefore, never sees the light of day.

During deployment, the embedded system proprietor combines their own unique, unreadable keys with unique information from each cryptographic memory, and applies cryptographic hashing functions like SHA or AES algorithms to the combined information to create a unique number, called a digest. The resulting value is so sensitive to the original information that changing even a single bit will result in a completely different result. The digest is unique to the individual device and is the basis for its digital signature. Hashes are used to create the unique device identifying numbers and keys, which are written into the device.

Once this process is complete, fuses are blown permanently locking the keys inside the device. Since the information used to create the hash is completely inaccessible, no other entity can create the same number. Session encryption keys are generated by the device for each trusted session and are always unique. The host cannot read them but must demonstrate knowledge of them as part of the challenge-response process during authentication.

To operate, the cryptographic memory enables the embedded system to establish a trusted session with a host using a random-number-enhanced mutual authentication process. The host reads the cryptogram and other identifying information from the embedded system, and combines this information with its own set of keys, knowledge of the unreadable keys buried in the cryptographic memory device and a random number. A large number, called a challenge, is created based on this information. The challenge is sent back to the device, along with the random number. The device then tries to calculate the same challenge number, based on the cryptogram, its unreadable keys and the random number it has received. If the attempt is successful, the device updates its cryptogram and declares the host authentic. The host then authenticates the embedded system by calculating a new cryptogram, and comparing it to the newly calculated cryptogram from the device. If they match, the device is authentic. Only a device possessing the keys the host expects can generate a correct cryptogram. The keys never leave the device. Only computed information, based on the keys, is transmitted. In addition, new cryptograms and session encryption keys are generated for each and every successful authentication transaction. Systemic attacks that try to exploit the information transmitted are useless in trying to defeat this type of security because the authenticating information changes with every successful authentication transaction.

Requiring a trusted link through mutual authentication also forms bases for protection against counterfeits. If label information is embedded into the cryptographic memory device, then it would take an authentic party to clone it because of prior authentication requirement. Although it would be possible for someone to steal a host reader that could read information from a cryptographic label created by that host, the information could not be used to clone fake labels because the information used to authenticate the device remains inside it. Without access to the authentication keys within the device, it is virtually impossible to create a device that can be authenticated.

In addition, the cryptograms in a cryptographic memory are dynamic. The internal non-volatile registers update themselves with a new cryptogram each time there is successful authentication. Since a different random number is used to generate each cryptogram, no two functionally equivalent operations are identical. The encrypted text for any given clear text will always be different for each encryption operation with the same device. This dynamism extends to message authentication codes, session encryption keys and cryptograms. With such dynamism, the current state of the cryptographic engine at any time maintains ties to the initial values of initially programmed secrets and cryptographic transactional history.

Cryptographic memories are laden with security features that sometimes are not available in secure microcontrollers. These include hardware authentication counters to eliminate systematic attacks, multiple security access levels and privileges, segmented memories with independent security access rights, and many more. On top of it all, these features are user configurable at time of deployment and fuses are blown to permanently lock the configuration. The use of cryptographic memories to secure embedded systems is a paradigm shift to finally deliver adequate security at a very affordable cost. Securing embedded systems may no longer be a trade-off with cost. ■

Embedded development for the rest of us

By John Leier, Digi International

One of the first platforms to support .NET Micro Framework is the Digi Connect ME. The Connect ME is an embedded serial-to-Ethernet module powered by an ARM7TDMI processor running at 55MHz.

Connect ME serial-to-Ethernet module in RJ-45 connector form factor

■ On a recent airline flight I found myself sitting next to a software engineering manager of a mid-size company. We struck up a conversation about the challenges of finding good embedded developers; it seems there are always too many embedded design projects and not enough resources to staff them. She said it was unfortunate that she could not use all her staff on embedded projects; she had a team of twelve developers, but only three of them were highly proficient in drivers, board-support packages, and boot-loaders. The other nine developers were all very good application developers, but they had no experience with low-level coding. I asked her if she had ever heard of the Microsoft .NET Micro Framework. The Microsoft .NET Micro Framework opens up embedded design projects to a new group of developers. In fact it is targeted directly at teams like those at my airline companions company – teams having some embedded experience and resources but not enough to accomplish all the projects on their roadmap, and teams including developers who have not previously been an option for embedded projects.

.NET Micro Framework has its roots in Microsoft Research and its original charter was to solve the problem of how to easily create embedded devices on low-power, small-footprint designs. With the belief that networked devices will become ubiquitous, that 32-bit processors will replace 8-bit and 16-bit processors and become the new standard, and that dozens of new protocols and architectures (like Z-Wave and flavors of Mesh networking) will continue to be defined every year, it quickly became apparent that several issues would need to be addressed. In order to design and ship the large growing number of networked embedded devices, the pool of available developers needed to grow dramatically; embedded developer productivity needed to increase significantly; and power efficiency would become even more important as new types of embedded applications were created, many of them battery-powered wireless applications.

The .NET Micro Framework was written from scratch to address all these concerns. It is not a cut-down version of Windows, or even Windows Embedded CE. Unlike Windows Embedded CE and some other embedded platforms, .NET Micro Framework does not require an MMU, allowing it to be used on ARM7 processors, in addition to ARM9 and Blackfin devices. And the memory footprint is as small as several hundred Kbytes of RAM and flash/ROM. A managed code environment on Windows Embedded CE is approximately 10 to 12 Mbytes. This large difference in memory required for a managed platform running .NET APIs means that a solution using .NET MF can be built with a lower cost of goods and less expensive bill of materials. The platform was also designed to provide a C# managed code environment. C# is an easy to learn, easy to use language that can raise the productivity of developers above that of C or even C++. The managed code environment means the developer is not spending time tracking down memory overwrites and mishandled pointers – these problems don't exist; the trade-off for managed code is that it isn't real-time; garbage collection makes the timing non-deterministic. By integrating .NET Micro Framework into Visual Studio 2005 and by using a subset of the .NET application framework, there is very little learning curve on APIs and tools. Many non-embedded developers are familiar with .NET, and even more are familiar with the Visual Studio tools.

It was first used in Microsoft SPOT (smart personal object technology) wristwatches. These SPOT watches are powered by a 27MHz ARM7TDMI processor and they receive dynamic updates of news, sports, stock prices, weather and traffic over an FM sub-carrier band. It was quickly realised that a wristwatch needed to be very power efficient – no one would want to plug a watch in for charging several times per day or once each evening. The SPOT watches proved to be an effective way to test the platform and help it mature. In February 2007, Microsoft released version 2 of the .NET Micro Framework for general embedded application development. Vendors including Digi International, Freescale, and Embedded Fusion have been shipping hardware compatible with .NET MF for several quarters, and many others are porting the platform to their products.

Several pieces comprise the .NET MF architecture. At the lowest level, sitting directly above the processor and peripherals, is the HAL (hardware abstraction layer) or hosted operating system. .NET MF can run in two modes, either directly talking to the hardware, or hosted by another operating system (perhaps an RTOS) which provides services and extensions to .NET MF. Above the HAL or hosted OS is the PAL (program abstraction layer). Unlike the HAL, the PAL is independent of the underlying hardware. Above the HAL is the CLR or common language runtime. The CLR manages memory, threading, code execution, garbage handling, serialisation, exception handling, and other services. Above the CLR are the managed libraries, including libraries for .NET and WPF (Windows presentation foundation). At the top of the stack is the developer application and libraries. This is the only layer that the embedded developer typically needs to program. The lower level drivers and CLR are implemented in C++ and typically provided by a hardware vendor such as Digi or Freescale. The user application is written in C# and links to the bootable runtime. Although there are many layers under the hood, the platform is actually very easy to learn and use.

The APIs supported in the .NET Micro Framework are a subset of the entire .NET Framework for standard desktop and server applications, much as Compact Framework (which runs on Windows Embedded CE) is a subset of the .NET Framework. Micro Framework focuses on the classes which are most applicable to embedded devices and leaves the others out. This allows developers to use existing code when practical, while not bloating the size of the Micro Framework.

A built-in hardware emulator for .NET Micro Framework is a big advantage of the platform. The default emulator can be extended by using XML, allowing the developer to create and debug an application without touching any hardware. Hardware vendors can also provide custom emulators representing their hardware, module or processor. Of course, a large part of the fun is actually seeing an application run on real hardware.

One of the first platforms to support .NET Micro Framework is the Digi Connect ME. The Connect ME is an embedded serial-to-Ethernet module powered by an ARM7TDMI processor running at 55MHz. The module includes the ARM7 processor, 2Mbytes of NOR flash, 8Mbytes of SDRAM, and an Ethernet PHY/MAC in a compact RJ-45 connector form-factor. The .NET Micro Framework support for the Digi Connect ME is implemented using a host operating system, based on Express Logic ThreadX. The host OS provides a complete IPv4 network stack that is exposed to the developer's application as a sockets interface. The Connect ME JumpStart development kit is available from Digi and includes a module and development board, along with a 90-day evaluation version of Visual Studio 2005 Professional and the Microsoft .NET Micro Framework SDK plug-in for Visual Studio. This kit provides a quick and easy way to investigate .NET MF or to build an embedded product which requires Ethernet connectivity.

Applications written to the .NET Micro Framework have access to all the hardware-independent APIs, as well as the hardware classes that the HAL supports. For the Connect ME, the HAL includes support for TCP/IP sockets, RS-232 serial and GPIO (general purpose inputs and outputs). There are four steps to create and run a simple application to read and write GPIO on the Connect ME:

1. Select Micro Framework project type in the New Project dialog in Visual Studio, and choose the Digi Connect ME template. This template generates the skeleton of a project with the required references already added. The skeleton is actually a Hello, World application that writes to the Debug output.

2. Double-click on Program.cs, the main module in the solution. Inside the Main() function, add the following lines:

```
InputPort MyInput = new
InputPort((Cpu.Pin)0,false,
InputPort.ResistorMode.Disabled);
```

```
OutputPort MyOutput = new
OutputPort((Cpu.Pin)1,false);
```

The online help for the InputPort class describes all of the parameters. The first parameter is which GPIO to use; in this instance the first GPIO pin on the Connect ME should be used. The second parameter is for something Microsoft calls a glitch filter; this filter, if implemented, can be used to smooth spikes on input state changes; the Connect ME does not use this parameter in its HAL and any value passed is ignored. The last parameter allows the HAL to configure inputs for one of three resistor modes – PullUp, PullDown, or Disabled; this value is also ignored for the Connect ME. The constructor for the OutputPort class only has two parameters. The first is the GPIO pin to use and the second parameter is the initial state for the output.

3. To read the input, simply assign it to a Bool type, bool InputState = MyInput.Read(); To write to the output, use the Write() method: MyOutput.Write(true);

4. Compile and deploy the application by selecting Start Debugging. The application will be sent over Ethernet to the flash on the Connect ME and started in SDRAM. Debug output will be sent to the Visual Studio Debug Output window.

That is all required to write, compile, deploy and debug a first application running .NET Micro Framework on an embedded target and manipulating GPIO. There is a third type of GPIO, an InterruptPort class. The Connect ME doesn't map an interruptible pin to its GPIO, so the InterruptPort is not supported, but other hardware platforms may provide that option. Similar classes, methods and properties exist for reading and writing serial and for creating socket servers and clients. All are documented in the online help and are also easy to use even for those without much previous C experience.

.NET Micro Framework succeeds in meeting its goals. With the C# language, a version of the .NET Framework targeted at embedded applications, and the powerful Visual Studio tools, there are now more options for embedded designs. Time to market is dramatically reduced when there is no boot-loader, low-level drivers, or system internals for the embedded team to implement. Connect ME customers have gone from design to working prototypes in a matter of weeks and have been ready to go to market in as few as three to six months. The .NET Micro Framework model opens up embedded programming to new developers; where before embedded development was too complicated or required special tools and knowledge, now it is approachable and appropriate for many applications. ■

Measuring RF immunity of operational amplifiers

By Gunter Langer, Langer EMV-Technik

When in use operational amplifiers are exposed to electromagnetic interference. It is important to precisely determine their limit parameters to exclude any possible EMC effects.

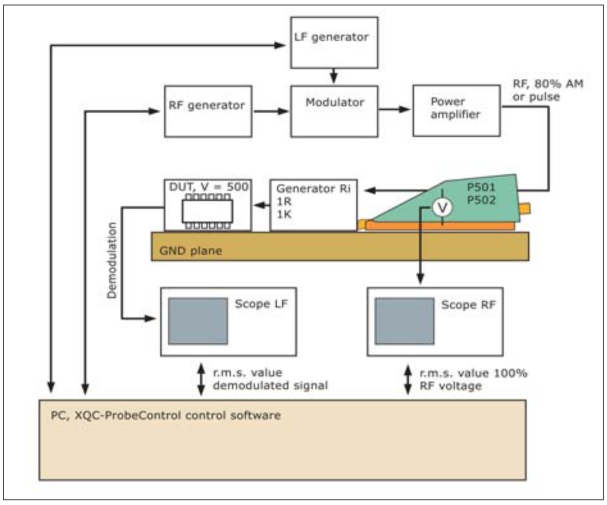


Figure 1. Block diagram of the set-up to measure the RF immunity of an operational amplifier

■ For a bipolar operational amplifier installed in automobile electronics, RF interference signals are demodulated during the EMC test procedure at radiated field strengths of more than 100V/m. Such signals cause electronic malfunctions, which can however be avoided in the planning phase by selecting an appropriate operational amplifier with higher RF compatibility, since operational amplifier ICs have different immunity to radiated interference depending on the respective IC manufacturer. IC manufacturers integrate countermeasures in ICs to increase their RF immunity, but despite these precautions the development engineer must protect the operational amplifier against excessively high RF influences. A new measuring method will assist both IC manufacturers and users by providing a tool for optimum planning of RF protection on one hand, and a tool to determine more precise useful development parameters on the other. The following example demonstrates how these EMC parameters can be determined.

The Langer company was faced with the problem of evaluating the immunity of two ASICs featuring operational amplifiers to radiated interference in a range up to 3 GHz, for automotive use. The following measuring method was developed to tackle this task. The operational amplifier circuit was operated as an inverted amplifier. Demodulation at the input base emitter diode was raised to a level easily recorded with measuring devices if the amplification is high enough. An interference generator with the required characteristics is derived according to the practical circumstances.

1. In the case of radiated interference, the RF magnetic field penetrates the existing low-impedance loops. These loops close via the distributed capacitance of the conductor runs if the frequency is high enough. A low-impedance generator was connected directly to the IC input to simulate this effect. Decoupling relative to the useful signal (negative feedback resistor) had to be provided via a decoupling capacitor. The low-impedance generator could thus be used to evaluate the EMC measures (EMC capacitors) integrated in the IC and assess their more or less satisfactory implementation in terms of RF technology.

2. An existing external series resistor increases the immunity to conducted interference. Integrated EMC capacitors that come off badly in terms of HF technology if tested with a low-impedance generator perform much better if connected through this series resistor. A high-impedance generator was used to check this characteristic.

The following generators were developed and used to perform the two measuring tasks described above: 1) a low-impedance generator, 1 ohm source resistance, and 2) a high-impedance generator, 1 kilohm source resistance, both with frequency range 10 MHz to 3 GHz. The generators were developed by modifying the

Click-for-more

Interested in more information about measuring RF immunity?

Visit our specific webpage with useful links to this topic.

http://www.embedded-control-europe.com/know-how?kid=112

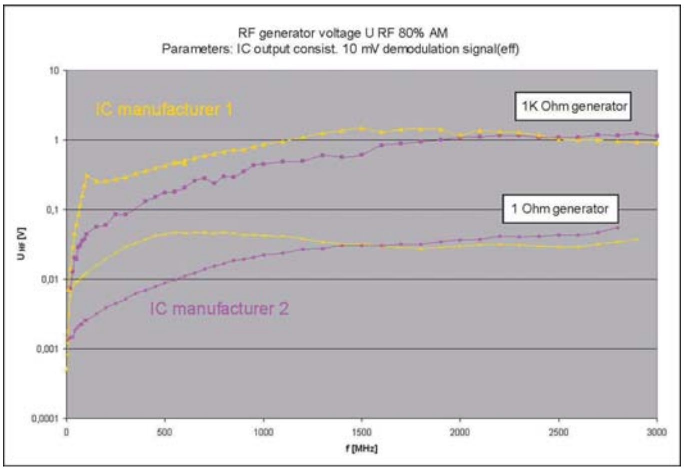


Figure 2. Results of measuring the RF immunity of two operational amplifier ASICs from different IC manufacturers

existing P501 and P502 type probes (automatic P500 probe measurement equipment for the IEC 62132-4 and DPI-BISS (6.8nF) DPI measuring method from Langer). These generators were used together with the automatic P500 measurement equipment and the associated XQC-Probe-Control control software to measure and document the frequency response characteristics (figure 1). The operational amplifier IC (DUT) is located on an adapter PCB which is let into a metal plate (GND plane). The low-impedance or high-impedance generator is placed on the metal plate and connected to the corresponding IC pin contact. The generator probe includes a voltmeter to determine the RF voltage (generator voltage UG). UG was adjusted to a sufficiently high level in the test set-up to achieve a demodulation voltage of up to 10mVeff.

The measurement results are shown in figure 2. The effect of the integrated RF filter capacitors (lower frequency range) improves with -increasing frequency. The measurement with the low-impedance generator shows better immunity to radiated interference up to 1.5 GHz for operational amplifier IC no.1. This result indicates that the IC has a bigger integrated EMC capacitor or better interconnection.

The line inductances determine the current distribution above 1 GHz. The 1 kilohm generator naturally provides better values. The stable RF behaviour of operational amplifier IC no.1 can also be demonstrated at higher frequencies (probably better interconnection of the capacitor). The EMC test on the installed electronics also confirmed behaviour of this operational amplifier IC. ■

Product News

■ ARM launches newCortex-A9 processors

ARM launches its new Cortex-A9 processors. The ARM Cortex-A9 MPCore multicore processor and ARM Cortex-A9 single core processor deliver performance within tight power constraints for innovative devices that deliver superior overall functionality, such as smartphones, connected mobile computers, consumer electronics, automotive infotainment, networking and other embedded and enterprise devices. The Cortex-A9 processors are compatible with other Cortex family processors and the ARM MPCore technology, thereby inheriting a rich ecosystem of OS/RTOS, middleware and applications to lower the costs associated with adopting a new processor.

News ID 817

■ Vector supports CIA447 communications protocol

A working committee of numerous automotive OEMs, module producers and communications specialists has 'under the auspices of CiA ' come to an agreement on a CANopen-based communications protocol. This enables easy integration of taxi meters, radio systems, roof bars and other electronically controlled devices in taxis, police cars and other official vehicles from different automotive OEMs. In this area, Vector offers software components for implementing the ECUs, a modified version of the CANoe.CANopen development and test tool, and consultation and project support.

News ID 736

■ Green Hills: tool suite supports OCTEON multi-core MIPS64 family

Green Hills announces that its product suite supports Cavium Networks' OCTEON multi-core MIPS64 processor family. This support includes the MULTI integrated development environment, TimeMachine tool suite, Green Hills compilers, DoubleCheck static analysis tools, μ-velOSity royalty-free real-time operating system and Green Hills Probe which are now available for the OCTEON family of multi-core MIPS64 processors.

News ID 844

■ Express Logic: RTOS and TCP/IP stack support LM3S1000/8000 MCUs

Express Logic announces that its ThreadX RTOS and NetX TCP/IP networking stack now support Luminary Micro's new Stellaris LM3S1000 Series and LM3S8000 Series ARM Cortex-M3-based microcontrollers. ThreadX and NetX for the new Stellaris devices are designed for use with the ARM/Keil RealView IDE from ARM.

News ID 743

ARC: "Energy PRO" for low power operation

ARC International enables energy efficient system-on-chip design by introducing "Energy PRO" technology for ultra low power operation. Configurability enables creation of power efficient cores. Energy PRO further reduces power consumption by as much as four fold.

News ID 681

■ SMSC extends MOST datalogger tool portfolio

SMSC announces the extension of its MOST Datalogger tool portfolio for its OptoLyzer G2 30xx family, the analysis platform for both MOST25 and MOST50 networks. The Datalogger extension for OptoLyzer G2 and OptoLyzer PowerPack, in conjunction with the OptoLyzer G2 30xx, offers an optimized set of network analysis tools to log all MOST traffic to any USB storage device that is connected to the OptoLyzer OL30xx.

News ID 805

■ Renesas: microprocessor delivers 2.8 GFLOPS for high-end graphics

Renesas Europe announces availability of the SH7780 microprocessor that delivers Pentium-class floating point performance into consumer and industrial markets. The device is ideal for applications requiring high-end graphics and so features a floating point unit tailored for that purpose. Furthermore it consumes only 2.5W, is available in a -40 to 85 degree C version, integrates PCI, has a long life time, and is low cost compared to a Pentium microprocessor of similar FPU performance. The superscalar SH7780 uses a SH-4A CPU + FPU core, with a maximum operating frequency of 400MHz. Its 64kB, four-way set-associative cache memory is divided into two 32-Kbyte areas, one for instructions and one for data

News ID 770

■ SCIOPTA opens Munich office

SCIOPTA Systems is opening an additional sales office in the Munich area. The Munich office will serve the regions of postcodes: 8xxxx, 9xxxx, 0xxxx and will also be responsible for the Austrian market. Head of this office is Mr. Thomas Winkler, who has a lot of experiences in embedded systems specially in real-time operating systems. Mr. Winkler started his career at Windriver GmbH and he successfully continued as Managing Director of Enea OSE Germany.

News ID 831

■ Silabs doubles Flash memory on small form factor MCUs

Silicon Laboratories announced the expansion of its small form factor family to include the C8051F336 family of highly-integrated 8-bit MCUs. Pin-for-pin compatible with Silicon Laboratories C8051F330 family of devices, the F336 doubles the Flash code space to 16 kB. Additional memory combined with high-performance peripherals and four-corner operation enables system designers to easily upgrade their products.

News ID 738

■ NEC: automotive gateway controller with FlexRay interface

NEC Electronics Europe announces a new microcontroller for automotive body applications. The V850E/CAG4-M is a member of NEC Electronics' 32-bit RISC microcontroller family with embedded FlexRay interface. Offering high performance, large memory and fully-fledged communication interfaces, it is especially suited for high-end gateway and other body applications.

News ID 825

■ TI: USB stick tool for wireless MCU development

Texas Instruments announces a new tool for designing embedded systems that combines ultra-low-power MSP430 microcontrollers with wireless communications. TI's new eZ430-RF2500 development tool, packaged in a convenient USB stick form factor, offers two radio frequency -enabled microcontroller target boards and a PC debugging interface that may be used to develop stand-alone wireless projects for sensing and metering, home security and automation, medical and other innovative applications.

News ID 722

■ Freescale: additions to the PowerQUICC II Pro family

Freescale announces new additions to the PowerQUICC II Pro product line. The processors integrate a broad array of technologies essential for success in storage markets, including SATA, PCI Express, USB 2.0, Gigabit Ethernet, XOR acceleration and high-performance security functionality. The storage family's MPC837xE product line targets SMB markets, supports RAID 5 and is designed for Network Attached Storage applications.

News ID 697

■ Atmel: free Linux BSP for AT91SAM9 MCUs

Atmel and TimeSys announce a free Linux Board Support Package for Atmel's ARM9-based AT91SAM9 Microcontrollers. Supporting the entire range of SAM9 products, this BSP includes Atmel's Linux kernel and drivers, BusyBox utilities for basic commands and features, and a Linux host/cross toolchain capable of rebuilding the Linux kernel and the basic packages included in the BSP.

News ID 725

■ SST: 8051-based microcontroller for mobile devices

Silicon Storage Technology announces a new addition to the company's SuperFlash-based FlashFlex family of 8-bit, 8051-compatible microcontrollers, the SST89V54RD-33-C-QIF. The new SST89V54RD is available in a 6mm x 6mm WQFN package, making it the smallest 8051-based microcontroller currently on the market. The WQFN package offers an extremely low-profile nominal package height of only 0.7mm (maximum total thickness of 0.8mm), making the new SST89V54RD well suited for height-constrained mobile applications.

News ID 694

■ Freescale: processor helps machines see, hear, speak

Freescale Semiconductor introduces the MPC8610 integrated host processor - a high performance device based on Power Architecture technology. The MPC8610 helps robots see and navigate in 3D space, enables touch screen kiosks to recognize voices and facial features, and allows cockpit controls to display images. This versatile device replaces up to four chips required by other solutions.

News ID 779

More information about each news story is available on www.embedded-control-europe.com/ece_magazine
You just have to type in the "News ID". —

ECE

Editor
Jürgen Hübner
phone +49(0)8092-2477413
fax +49(0)8092-2477429
jh@iccmedia.com

Contributing Editor
Tony Devereux
devrex@teyboyz.freeserve.co.uk

For Reader Inquiries and Address Changes please contact:
info@iccmedia.com

Sales & Marketing Director
Manfred Blumoser
phone +49(0)8092-2477411
fax +49(0)8092-2477429
mb@iccmedia.com

Claudia Mellein
cm@iccmedia.com

Christiane Lockner
cl@iccmedia.com

Sales Office - UK and USA, Benelux, Scandinavia
Blue Sky Communications
Malcolm Cameron
Kent Innovation Centre
Thanet Reach Business Park
Millennium Way
Broadstairs Kent CT10 2QQ UK
phone +44 (0)1843 609357
fax +44 (0)1843 609358
mc@blue-sky-communications.com

Sales Office - Asia
Jean Cheng, *jean@i-media.com.tw*
Vivian Hung, *vivian@i-media.com.tw*
Innovative Media Information & Services
7F-3, No. 26, Sec. 2, Ming-Quan East Rd.
Taipei 104 Taiwan
phone +886 2 2563 1186

Head Office
ICC Media

ICC Media GmbH
Rauwagnerstr. 5
85560 Ebersberg / Germany

Editorial Office UK
36a Blackacre Road
Theydon Bois
Essex, CM16 7LU

Although we make every effort to present up-to-date, accurate information, boards&solutions will not be responsible for any errors or omissions or for any results obtained from the use of such information. The magazine will not be liable for any loss caused by the reliance on information obtained on this site. Furthermore, ece does not warrant the accuracy or completeness of the information, text, graphics in this magazine. The opinions expressed in the articles are those of the authors and not necessarily the opinions of the publisher.

Ramtron: 8051-based MCU with 2-kilobit FRAM

Ramtron has launched the VRS51L3072, an 8051-based microcontroller with 2-kilobit of non-volatile FRAM memory. Ramtron has added FRAM to its fast and flexible Versa 8051s for a quick and reliable non-volatile data storage and processing system that is ideal for saving system status, data logging, and storing nonvolatile variables in a range of applications from sensors and meters to industrial controls, instrumentation and medical devices.

News ID 767

Renesas: SH7619 Ethernet MCU supported by µC/OS-II BSP

Renesas Europe and its alliance partner Embedded Office have announced the availability of a µC/OS-II Board Support Package for the M3A-HS19G59 board. The board is based on the SH7619 device that features Ethernet MAC&PHY and a 160 DMIPS SH2 RISC CPU core, as well as many other peripherals. The BSP features the realtime kernel µC/OS-II and the TCP/IP stack µC/TCP-IP. On top of the TCP/IP stack many application protocols are available, e.g.: HTTP, FTP, SNTP, POP, SMTP, DHCP, DNS and more.

News ID 717

CML: multi-mode wireless data modem for M2M and SDR

CML Microcircuits launches the CMX7143, a highly integrated and adaptable multi-mode wireless data modem built on CML's proprietary FirmASIC component technology which offers the ability to implement different wireless data schemes. Wireless data scheme flexibility is a growing market requirement; particularly sought after by Machine-to-Machine and Software Defined Radio product manufacturers.

News ID 827

Atlantik: integrated Wi-Fi and ZigBee pin-compatible modules

Rabbit Semiconductor, in sales of Atlantik Elektronik, is offering two wireless technologies directly integrated in new RabbitCore modules. The RCM4400W and RCM4510W expand on the family of pin-compatible modules already based on the Rabbit 4000 microprocessor, giving customers greater freedom to now select from serial, wired Ethernet, wireless Ethernet / Wi-Fi or wireless ZigBee / 802.15.4 as their communication link in their Industrial Control, RTU or Building Automation applications.

News ID 690

Freescale: ColdFire MCUs for Linux applications

Freescale Semiconductor has introduced a ColdFire microprocessor family designed to enable low-power, high-performance embedded systems running the Linux operating system, delivering 410 Dhrystone MIPS core performance at approximately 380 mW. Freescale's MCF5445x family includes 12 advanced microprocessors that integrate a rich set of connectivity peripherals. The MCF5445x devices include an on-chip memory management unit to support protected memory operating systems, such as the Linux OS.

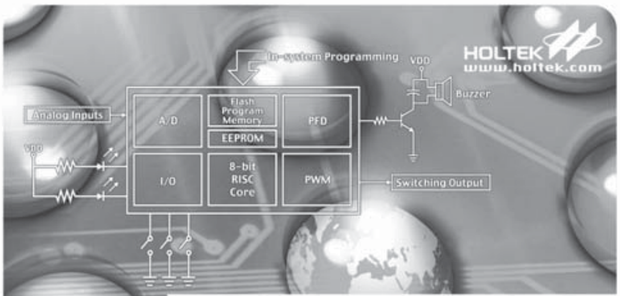
News ID 719

Digi-Key to distribute Panasonic ZigBee ready modules

Panasonic announces the appointment of Digi-Key as an authorized distributor for its latest in a line of RF modules, the PAN4555. Supporting the 802.15.4 radio standard and the ZigBee specification, the Panasonic module is designed for various sensing, monitoring and control applications for home automation and industrial control.

News ID 794

ECE
03/07
March 2007
Information & Know-how for Embedded Engineers
The Complementary Magazine to
www.Embedded-Control-Europe.com
STR750F
SPECIAL FEATURES:
Motor Control
Analog & Power
Real-Time Operating Sytems
Small Form Factor Boards
COVER STORY:
Field-oriented brushless motor control based on an ARM7 processor



Lattice: XAUI to SPI4.2 programmable bridging solution

Lattice Semiconductor announces a XAUI/HiGig/HiGig+ to SPI4.2 programmable Fabric Interface Chip solution implemented in its LatticeSCM FPGAs. The solution, which utilizes the LatticeSCM device's System Packet Interface Level 4 Phase 2 (SPI4.2) hard IP capability, and includes Lattice's 10Gigabit Ethernet Media Access Controller soft IP core and the XAUI/HiGig/HiGig+ to SPI4.2 bridge design, provides a high-performance interface between the SERDES-based XAUI standard, used in 10G Ethernet networks, and SPI4.2, a very popular parallel bus interface used by Network Processor Unit devices.

News ID 685

Nordic: wireless sensor network development kit

Nordic Semiconductor and ANT launched the ANTDKT3 wireless sensor network development kit. By using the kit, engineers avoid the design complexity traditionally associated with wireless sensor networking and are able to build a functioning 2.4GHz wireless sensor network within minutes to test their specific applications.

News ID 684

TRANGO joins MontaVista partner program

TRANGO Virtual Processors announces that it has joined the partner program of MontaVista. TRANGO offers a platform that addresses the consumer electronics and wireless markets by providing both advanced security and a richer end-user experience. The combination of TRANGO scalable security and MontaVista Mobilinux enables OEMs and ODMs to create flexible, secure, and highly competitive products.

News ID 828

NEC: 15-inch LCD modules for industrial equipment

NEC LCD introduces two new 15-inch amorphous-silicon color thin-film-transistor liquid crystal display modules with extended graphics array resolution for industrial equipment such as factory automation controllers, measuring instruments, automatic teller machine terminals, kiosks and point-of-sale systems. The NL10276BC30-32 is a standard product with brightness of 250 candelas per square meter (cd/m2), while the NL10276BC30-33 has brightness of 350 cd/m2.

News ID 815

Adeneo: engineering services on .NET Micro Framework

Adeneo announces its port of the .NET Micro Framework onto Atmel's AT91SAM9261 ARM9-based microcontroller. This port shows the ability of Adeneo to provide strong support and engineering services to Original Equipment Manufacturers looking to build products on the .NET Micro Framework.

News ID 802

Atlantik: networking SoC coprocessor family

Lantronix in sales of Atlantik Elektronik, announces a family of dedicated networking coprocessor system-on-chip solutions. The DeviceLinx XChip SoC family enables serial-to-Ethernet connectivity and web services. With the addition of the XChip family, Lantronix offers manufacturers a complete line of software-compatible device networking solutions across a product's life cycle ' from external boxes, to boards, to modules, to chips. Its small size and application-ready networking firmware make XChip an ideal solution for high-

News ID 783

■ NI: dynamic signal acquisition modules with 16 simultaneous channels

National Instruments announces the PXI-4498 and PXI-4496 dynamic signal acquisition modules, offering 16 simultaneous 24-bit analog inputs per module and IEPE constant current signal conditioning for precision measurements with microphones and accelerometers in high-channel-count systems, such as noise mapping, beamforming applications and structural vibration.

News ID 763

■ Geensys provides embedded product design services

Geensys, the merger of TNI Software and Ayrton Technology, will provide a range of embedded engineering and consulting services, embedded development tools and embedded software IP focusing on satisfying the embedded development requirements of the automotive, aerospace, defence, railway, industrial automation and telecommunications industries. Geensys will offer a range of embedded development tools covering model-driven development and requirements-centric system engineering.

News ID 716

■ ADI: CMOS dual operational amplifier consumes 20 µA

Analog Devices announces the AD8506 low-power CMOS dual operational amplifier with rail-to-rail inputs and output is designed for portable applications, including battery-powered patient monitors, remote sensors, handheld instrumentation and other mobile equipment that require precision measurement at low voltage levels with minimal power consumption. Featuring a novel circuit architecture, the AD8506 maintains high linearity by minimizing distortion through its power supplies and its inputs.

News ID 713

■ Atmel: 128K-bit serial EEPROM in 8-pin XDFN 0.40 mm package

Atmel announces a 128K-bit Serial EEPROM device in an 8-pin XDFN package with overall z-height of 0.40mm. Building upon Atmel's robust Serial EEPROM DFN portfolio, the XDFN package will accommodate the industry movement toward applications with the most stringent space constraints. The 8-pin XDFN (1.8 x 2.2 x 0.40mm) package is now offered in 2-wire Serial EEPROM (AT24) and SPI Serial EEPROM (AT25) protocols: 2-wire Serial EEPROM densities range from 1K-bit to 128K-bit and SPI densities range from 1K-bit to 16K-bit.

News ID 742

■ ADI: free spice simulation software for linear circuits

Analog Devices announces the availability of a free, downloadable circuit design environment that simplifies and speeds product design by allowing developers to capture schematics and simulate circuits for evaluation of ADI's operational amplifiers and other linear circuits. Developed by National Instruments, NI Multisim Analog Devices Edition is a new version of the NI Multisim SPICE simulation software created for component evaluation and circuit design. The new tool, loaded with over 800 ADI op amps and other linear circuits, helps shrink product development time by allowing designers to quickly and easily select, evaluate, prototype and test ADI's linear circuits in a low-risk 'virtual evaluation board' environment.

News ID 766

■ Catalyst expands supervisor family

Catalyst Semiconductor announces the addition of three new devices providing system processor power supply monitoring and reset functions. The new CAT823, CAT824 and CAT825 voltage supervisors are available in a choice of seven standard threshold levels: 4.63V, 4.38V, 3.08V, 2.93V, 2.63V, 2.32V and 2.19V. In addition to an active-low reset output included on all three devices, the CAT823 also offers a manual reset and a watchdog input, while the CAT824 provides an additional active-high reset and a watchdog input.

News ID 711

■ TI: low-power protocol for simple and small RF networks

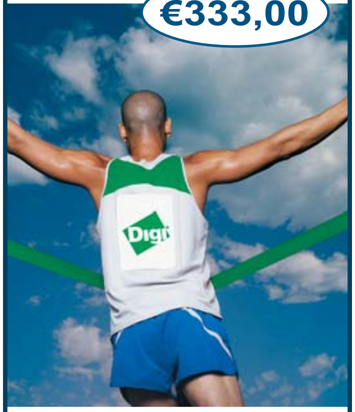
Texas Instruments announces the release of the SimpliciTI network protocol, a proprietary low-power radio frequency protocol targeting simple, small RF networks with less than 100 nodes. SimpliciTI network protocol was designed for easy implementation with minimal microcontroller resource requirements. The protocol runs out-of-the-box on TI's MSP430 ultra-low-power microcontrollers and CC110x/CC2500 RF transceivers.

News ID 701

■ Rutronik Discomp and Swissbit expand distribution contract

Discomp Elektronik, a subsidiary of Rutronik Elektronische Bauelemente, has reached a European-wide distribution agreement with Swissbit for Compact Flash products with immediate effect. Swissbit has also been manufacturing Compact Flash Cards, USB Sticks and Flash Disc Modules for industrial applications since the beginning of 2007.

News ID 689



TI: low-power ADCs for portable industrial applications

Texas Instruments introduces a pair of 16-bit analog-to-digital converters featuring 2x better linearity than competing devices (+/-1.5 LSB maximum INL). Combined with low-power operation, best-in-class temperature drift and industry-standard MSOP-8 or a 3mm x 3mm SON-8 package, the ADS8317 and ADS8326 provide an easy performance upgrade for portable, battery-powered applications including industrial data acquisition and portable medical instrumentation.

News ID 814

Adeneo and Phytec: acceleration kit for Windows Embedded CE 6.0

PHYTEC and Adeneo announce a new low cost Acceleration Kit for Microsoft Windows Embedded CE 6.0. The Acceleration Kit includes the ARM-core based phyCORE-PXA270 module and carrier board, LCD with integrated touch panel, Windows Embedded CE 6.0 Board Support Package source code, and all the contents required to enable users to set-up target hardware, build and load a Windows Embedded CE image with Platform Builder, and start developing with Windows Embedded CE .

News ID 746

Atmel: secure devices with 64-bit encryption engine

Atmel announced a family of secure products called CryptoMemory. These devices prevent product counterfeiting and/or piracy of intellectual property and OEM parts. CryptoMemory uses a 64-bit embedded hardware encryption engine, four sets of non-readable, 64-bit authentication keys and four sets of non-readable, 64-bit session encryption keys to provide a higher level of protection than products based solely on EEPROM technology.

News ID 836

NatSemi: PowerWise brand designates energy-efficient product portfolio

National Semiconductor unveiled its PowerWise brand to designate the company's line of energy-efficient, high-performance power and analog signal-path products. To system design engineers, energy efficiency is defined by a smaller energy footprint, less heat generation, and, in portable devices, longer battery life. To make it easier for these design engineers to find the most energy-optimized ICs available, National is classifying its power management and signal-path products with outstanding performance-to-power ratios and adding them to its PowerWise line.

News ID 773

ADI: dual-channel audio difference amplifier

Analog Devices announces the AD8270 and AD8273 high-speed, low distortion and precision dual-channel difference amplifiers designed for applications that require extremely fast and precise measurement without compromising signal fidelity, such as avionics, industrial process controls and high-performance audio equipment.

News ID 785

IXXAT: tools offer CANopen support

Based on the cooperation between IXXAT and KW-Software, the IEC 61131-3 compliant runtime environment ProConOS and the development environment Multiprog are now offered with CANopen support. Main feature of this solution is a full integration of the CANopen manager (master) into ProConOS and the CANopen configuration tool in Multiprog

News ID 830

QNX opens access to source code of Neutrino RTOS

QNX Software Systems announces that it is opening access to the source code of its QNX Neutrino realtime operating system under a new hybrid software licensing arrangement. QNX will make source code for its microkernel-based OS available for download. The first source release includes the code to the QNX Neutrino microkernel, the base C library, and a variety of board support packages for popular embedded and computing hardware.

News ID 708

LTC: single supervisor monitors 0.5V to 12V input range

Linear Technology introduces the LTC2917, a single voltage supervisor capable of monitoring a wide 0.5V to 12V input range. The LTC2917 can monitor 27 unique pin-selectable thresholds, ranging from a 1V drained single AA battery up to 12V industrial applications. Competitive solutions require separate supervisors for each of these different voltage thresholds. The LTC2917 VCC supply also includes a 6.2V shunt regulator for monitoring input voltages greater than 12V.

News ID 810

Advertisers Index

COMPANY	PAGE
Actron	33
ATMEL	11
Digi International	45
Digi-Key	2
Green Hills Software	5
HCC Embedded	31
Holtek	43
Hyperstone	17
Keil Elektronik	21
Lauterbach	23
Mentor Graphics	25
Mesago	44
Micro Digital	35
Microchip	19
MSC	3
National Instruments	27
SECO	35
Texas Instruments	7/29/48
Zilog	15

More information about each news story is available on
www.embedded-control-europe.com/ece_magazine
You just have to type in the "News ID". —

eZ430-RF2500
Complete Development Tool
ONLY $49