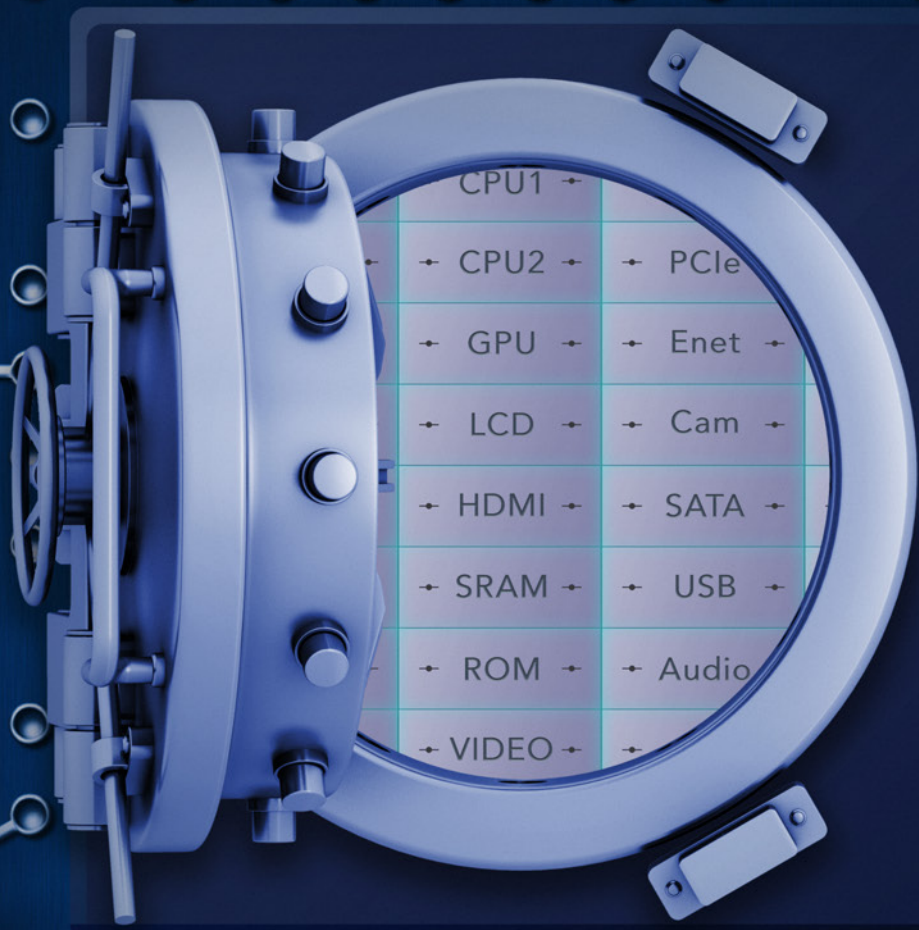


Embedded COMPUTING DESIGN

Connecting Silicon, Software, and Strategies for Intelligent Systems

NOVEMBER 2015 #7
VOLUME 13
EMBEDDED-COMPUTING.COM

LOCKING
DOWN
IoT EDGE
PROCESSORS
PG. 24



PLUS

RESEARCH REVIEW
AN INDUSTRY-UNIVERSITY
DEV TOOLS COLLABORATION
PG. 5

HOW-TO GUIDE
PWM CONTROL WITH MICROCHIP'S
CURIOSITY DEV BOARD
PG. 16

Annapolis Micro Systems

The FPGA Systems Performance Leader

WILDSTAR OpenVPX Ecosystem

FPGA Processing Boards

1 to 3

Altera Stratix V or
Xilinx Virtex 6 or 7
FPGAs per Slot

Input/Output
Modules
Include:

Quad 130
MSps
thru
Quad 550
MSps A/D
1.5 GSps thru
5.0 GSps A/D
Quad 600
MSps D/A
Dual 1.5
GSps
thru
4.0 GSps D/A

1 to 40 Gbit
Ethernet
SDR to FDR
Infiniband



Open VPX Storage

Up to 8 TBytes Per Slot

4 - 8 GBytes
Per Second

GEOINT,
Ground Stations,
SDR, Radar,
Sigint, COMINT,
ELINT, DSP,
Network
Analysis,
Encryption,
Image
Processing,
Pattern Matching,
Oil & Gas
Exploration,
Financial and
Genomic
Algorithms,

Open VPX Switch

1 to 40 Gbit
Ethernet
SDR to FDR
Infiniband

Chassis
4, 6 or 12 Slot
Up to 14G

**High Performance Signal and Data Processing
in Scalable COTS FPGA Computing Fabric**

190 Admiral Cochrane Drive, Suite 130, Annapolis, Maryland USA 21401
wfinfo@annapmicro.com USA (410) 841-2514 www.annapmicro.com



Why companies should focus on their core competency

By Rory Dear, Technical Contributor

rdear@opensystemsmedia.com

When something is outside a company's core competency it's time to outsource in order to save time, money, and risk.

"To outsource, or not to outsource?" is the key dilemma facing decision makers from start-ups to SMEs to corporate behemoths. In smaller scale organisations, the temptation to want to do everything themselves may well stem from their inception; when start-up funds were tight the last burden the fledgling CEO wanted to incur were additional expenses. The attraction of taking on learning a new task himself or delegating internal staff to do the same was naturally strong, and I speak from personal experience in describing how hard it can be to move away from that mindset.

I started in the embedded industry on the bottom rung of the proverbial ladder. As time progressed I quickly earned numerous promotions, and whilst performing new tasks I continued to simultaneously undertake my (technically) previous roles. Additionally, as new responsibilities were generated – tasks well outside my primary remit – I foolishly, enthusiastically, blindly volunteered myself. I describe this not as some nostalgic self-contemplation, but to draw you to the undeniable parallel this dilemma has at both the individual and organisational level. I eventually found myself working unsustainably long hours to encompass this unmanageable workload I had thrust upon myself. Far more importantly than the overall volume of work was the increasing detraction from my own core competency, which began to impact my personal success and has a parallel effect at the organisational level.

One poignant example I come across daily is companies involved in electronics often blindly insist on undertaking electronic design in-house. This may well stem from the same "we'll do everything" attitude of the earlier cited start-up CEO, and in fairness this is understandable – after all, he set out to create an electronics company and this is an electronics task, isn't it? Perhaps once upon a time, when the technology and tools were that much simpler, this would hold true. The reality today is electronics has exploded into an unfathomable number of highly specialist areas, most notably (for us) an entire ecosystem of embedded computing. The danger comes when those outside of that specific skills sphere, through lack of understanding of the complexities involved, have the temptation to attempt to undertake such design alone.

Doing so necessitates pushing inexperienced personnel up a very sharp learning curve and leaving them to the wolves to overcome the numerous challenges long since understood and resolved by existing specialists with decades of training and experience. It involves significant cost – be that for hiring new employees or training existing ones and buying the latest tools – time, and, of course, risk. With project competition greater than ever these are the three fundamental aspects one rarely has room to play with. Interestingly, the primary objection to outsourcing such work, sometimes alongside pride, is cost.

The cost differential of outsourcing may appear vast due to those evaluating not including their employees' time as cost, nor appreciating that if internal staff take twice as long, they effectively cost twice as much – and a factor of two is usually understating the difference. It's also easy to exclude the costs of financing that learning curve and investing in the latest tools that may well only be for a single project. Far more importantly, what if it all goes wrong?

A comparison I like to use is your family vehicle's braking system. Most of us won't mechanically touch our cars at all, myself included, but some fancy themselves as budding hobbyist mechanics. Very few have such confidence in their success that they're willing to place their family's lives at risk by undertaking work on the brakes. I've no doubt that my cognitive abilities could eventually learn how to successfully undertake brake repair, but the issue is that I can't commit to the time to become an expert, and unless I was I would never place my family's lives at risk through failure. Replace "brake repair" with "electronic design" and "family" with "project" and I hope I've made my point.

Beyond avoiding the risk of losing important customers, outsourcing frees up employees to focus on their own core competencies. Maximum benefit and success at individual and company level is derived from dedicating time to your strengths. Trying to save your business money by avoiding outsourcing may end up costing you far more than that project ever would. **ECD**

Departments

3 Tracking Trends *Rory Dear, Technical Contributor*

Why companies should focus on their core competency

5 Research Review *Monique DeVoe, Managing Editor*

An industry-university development tools collaboration

7 IoT Insider *Brandon Lewis, Assistant Managing Editor*

So, how do you *make* money in IoT?

31 Editor's Choice

Web Extras

» Amazon launches IoT platform, embedded responds with off-the-shelf dev kits

By Brandon Lewis, Assistant Managing Editor
[opsy.st/AmazonIoT](#)

» The search for a simpler, more cost-effective way to implement USB 3.0 functionality

By Gordon Lunn and Lee Chee Ee, FTDI Chip
[opsy.st/USB3FTDI](#)



DOWNLOAD THE APP

Download the
Embedded Computing Design app:
iTunes: [itun.es/iS67MQ](#)
Magzter: [opsy.st/ecd-magzter](#)

Silicon

10 Microcontroller-based FPGAs hit the mark *By Ted Marena, Microsemi*

Software

13 The theory behind product line engineering *By Curt Schwaderer, Editorial Director*

16 Look Ma, no code! PWM control in under five minutes with Microchip Curiosity development board *By Brandon Lewis, Assistant Managing Editor*

20 Testing the test: How to use coverage metrics for more effective embedded software testing *By John Thomas, LDRA Ltd.*

Strategies

24 Ensuring trust in IoT with Trusted Platform Modules and Trusted Brokered IO *Q&A with Steve Hanna, Infineon and Trusted Computing Group; Stefan Thom, Microsoft and Trusted Computing Group*

24



An industry-university development tools collaboration

By Monique DeVoe, Managing Editor

mdevoe@opensystemsmedia.com

Development tools are important in all stages of the embedded design process, and research is no exception. Limited budgets can create extra challenges for universities, and the lack of inexpensive, but high-quality tools is a challenge faced by many researchers, including Dr. Luciano Ost, University of Leicester (<http://le.ac.uk>) Lecturer, Embedded Systems and Communications Research Group.

"The lack of electronic design automation (EDA) tools combining model flexibility, and fast and accurate evaluation of performance, power, and reliability is one of the major challenges currently faced by embedded researchers," Ost says, adding that even expensive, commercially available tools don't often meet modeling and simulation needs for emerging technologies.

Free, open source tools – especially those available to the research community – can be highly beneficial for students learning the embedded ropes and for researchers looking to advance the field. When companies and universities work together to share tools and knowledge in an open source format, good things can happen for the future of embedded computing.

Imperas (www.imperas.com) provides tools and solutions for embedded software development, debug and test, and has been working with universities around the world with its Imperas University Program, which provides academic and research institutions access to tools and virtual platform models.

"The Imperas University Program encourages participation in the embedded systems community in three ways: use on research projects, use in the classroom, and sharing of virtual platform models through the Open Virtual Platforms (OVP) Library," says Duncan Graham, University Program Manager at Imperas.

The OVP initiative (www.ovpworld.org) provides researchers access to a library of more than 150 CPU models and more than 200 peripheral and platform models, as well as Extendable Platform Kits (EPKs) – all open source and distributed using the Apache open source license. Libraries like this are very helpful to researchers like Ost who are working with the cutting edge of processing technology.

"The description of processors – i.e., register or gate-level – is rarely available to universities, and commercial licenses are

quite expensive," Ost says. "Having free tools with different state-of-the-art processor models allows the exploration of new system architectures."

Imperas launched OVP in 2008, providing free access for academic users to the model libraries in addition to APIs, the OVPSim simulator, and to the OVP Forum for technical questions and discussions for academic users. Users can also share their models, platforms, and tools they develop. In 2010, the Imperas University Program formalized and expanded the OVP program, providing access to Imperas' Multiprocessor/Multicore Software Design Kits (M*SDK) and QuantumLeap Parallel Simulation Accelerator tools.

Ost has found the tools provided through the Imperas University Program to be helpful for his research. He and his research team focus on multi- and many-core embedded systems, specifically runtime management techniques for performance, energy efficiency, and reliability improvement that must be conducted at the application, operating system, or architectural level. He uses system modeling tools and languages like SystemC as well as virtual platform simulation tools like Imperas' OVPSim, which help Ost and his team work with increasingly complex processors and software architectures.

"The high-speed simulation and debugging capabilities of Imperas tools facilitate us to extend FreeRTOS to support novel runtime techniques as well as to promote instruction-driven performance and power models," Ost says.

Ost also notes that free tools also facilitate collaborative projects between universities and external industry partners. In collaboration with the Federal University of Rio Grande do Sul (UFRGS) in Brazil, a fast and flexible fault injector framework, called OVPSim-FIM, was developed on the basis of OVPSim and presented at DFT 2015 (www.dfts.org). OVPSim-FM enables users to identify and critical soft errors in multi- and many-core systems at an early design phase. **ECD**

Read about the FlexTiles project developed through Imperas University Program and Imperas's future plans at opsy.st/ImperasUniversity.

ADVERTISER INDEX

- 9 **Anaren** – Join the evolution
- 2 **Annapolis Micro Systems, Inc.** – WILDSTAR OpenVPX ecosystem
- 21 **ATP Electronics, Inc.** – ATP Ruggedized IoT NAND flash storage solutions
- 19 **COMMELL Systems Corporation** – 5th gen Intel Core i7/i5/i3 SBC
- 23 **Digital Voice Systems, Inc.** – AMBE+2 Vocoder chip delivers high quality voice at low cost
- 12 **Embedded World** – Get up to speed with the latest developments in your industry
- 29 **Sheldon Instruments** – Improving inter-processor communication with software framework
- 27 **Technologic Systems** – Superior embedded solutions
- 25 **Toradex** – Engineering resources at your finger tips
- 32 **WinSystems, Inc.** – Thinking beyond the board
- 30 **WITTENSTEIN High Integrity Systems** – Watchdog strategies for RTOS enabled embedded systems



Get your free digital edition at
embedded-computing.com/emag



Subscriptions:
embedded-computing.com/subscribe
subscriptions@opensystemsmedia.com



2015 OpenSystems Media®
© 2015 Embedded Computing Design
All registered brands and trademarks within
Embedded Computing Design magazine are the
property of their respective owners.
iPad is a trademark of Apple Inc., registered in the U.S.
and other countries. App Store is a service mark of Apple Inc.
ISSN: Print 1542-6408, Online: 1542-6459



Embedded Computing Design Editorial/Creative Staff

Rich Nass, Brand Director
rnass@opensystemsmedia.com
Curt Schwaderer, Editorial Director
cschwaderer@opensystemsmedia.com
Monique DeVoe, Managing Editor
mdevoe@opensystemsmedia.com
Brandon Lewis, Assistant Managing Editor
blewis@opensystemsmedia.com

Rory Dear, Technical Contributor
rdear@opensystemsmedia.com
David Diomedé, Creative Services Director
ddiomedé@opensystemsmedia.com
Konrad Witte, Senior Web Developer
kwitte@opensystemsmedia.com
Chris Rassiccia, Graphic Designer
crassiccia@opensystemsmedia.com

Sales Group

Tom Varcie, Sales Manager
tvarcie@opensystemsmedia.com
(586) 415-6500
Rebecca Barker, Strategic Account Manager
rbarker@opensystemsmedia.com
(281) 724-8021
Eric Henry, Strategic Account Manager
ehenry@opensystemsmedia.com
(541) 760-5361
Kathleen Wackowski, Strategic Account Manager
kwackowski@opensystemsmedia.com
(978) 888-7367

Asia-Pacific Sales

Elvi Lee, Account Manager
elvi@aceforum.com.tw

Regional Sales Managers

Barbara Quinlan, Southwest
bquinlan@opensystemsmedia.com
(480) 236-8818
Denis Seger, Southern California
dseger@opensystemsmedia.com
(760) 518-5222
Sydele Starr, Northern California
[sstarr@opensystemsmedia.com](mailto:ssstarr@opensystemsmedia.com)
(775) 299-4148
Twyla Sulesky, Strategic Account Manager California
tsulesky@opensystemsmedia.com
(408) 779-0005

Reprints and PDFs

Contact: Wyndell Hamilton, Wright's Media, whamilton@wrightsmmedia.com, 281-419-5725

EMEA

Rory Dear, Technical Contributor
rdear@opensystemsmedia.com
James Rhoades-Brown – Europe
james.rhoadesbrown@husonmedia.com

Christian Hoelscher, Account Manager – Europe
christian.hoelscher@husonmedia.com
Gerry Rhoades-Brown, Account Manager – Europe
gerry.rhoadesbrown@husonmedia.com

OpenSystems Media Editorial/Creative Staff



John McHale, Group Editorial Director
Military Embedded Systems
PC/104 and Small Form Factors
PICMG Systems & Technology
VITA Technologies
Signal Processing Design
jmhale@opensystemsmedia.com

Joe Pavlat, Editorial Director
PICMG Systems & Technology
jpavlat@opensystemsmedia.com

Jerry Gipper, Editorial Director
VITA Technologies
jgipper@opensystemsmedia.com

Steph Sweet, Creative Director
Joann Toth, Contributing Designer
Lisa Daigle, Assistant Managing Editor
Military Embedded Systems

PC/104 and Small Form Factors
ldaigle@opensystemsmedia.com

Sally Cole, Senior Editor
Military Embedded Systems
scole@opensystemsmedia.com

Brandon Lewis, Assistant Managing Editor
Industrial Embedded Systems
PICMG Systems & Technology
Signal Processing Design
blewis@opensystemsmedia.com

Jennifer Hesse, Managing Editor
VITA Technologies

Joy Gilmore, E-cast Manager
jgilmore@opensystemsmedia.com

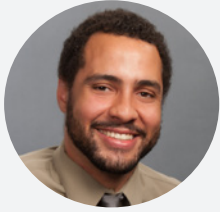
Mariana Iriarte, Associate Editor
Military Embedded Systems
miriarte@opensystemsmedia.com

Corporate

opensystemsmedia.com

Patrick Hopper, Publisher
phopper@opensystemsmedia.com
Rosemary Kristoff, President
rkristoff@opensystemsmedia.com
John McHale, Executive Vice President
jmhale@opensystemsmedia.com
Rich Nass, Executive Vice President
rnass@opensystemsmedia.com

Wayne Kristoff, CTO
Emily Verhoeks, Financial Assistant
Headquarters – ARIZONA:
16626 E. Avenue of the Fountains, Ste. 201
Fountain Hills, AZ 85268
Tel: (480) 967-5581
MICHIGAN:
30233 Jefferson, St. Clair Shores, MI 48082
Tel: (586) 415-6500



So, how do you *make* money in IoT?

By Brandon Lewis, Assistant Managing Editor

blewis@opensystemsmedia.com

A recent InfoWorld article proclaims “The Internet of Things is not paying the rent.” In it, Adobe’s VP of Mobile Matt Asay cites data from VisionMobile and McKinsey & Co. to point out that “less than 10 percent of IoT developers are making enough to support a reasonably sized team,” and that “developers need to get real about what they’re selling and to whom,” which “probably involves a ‘dull’ enterprise-facing business.” This begs the question, how do you make money in the IoT?

In a column last year on hardware commoditization I discussed the idea of “IoT-as-a-Service,” wherein Internet of Things companies could potentially transition away from one-off IoT platform sales and into business models that allow for accretive growth by means of data and feature monetization. In this cloud-based approach, companies could establish service plans or provide additional features to end users similar to how your cell phone or cable company operate, generating recurring streams of income

that continue to flow after the initial platform sale (or perhaps, giveaway) to help offset ongoing maintenance, service, and support costs. Furthermore, this paradigm permits a new way of thinking about the product development life cycle, as rather than offering a portfolio of hardware platforms each with different features engineered into individual SKUs, software can be utilized to enhance or reduce functionality on a given platform (or set of platforms) by turning capabilities on or off.

However, one setback of this model is that it relies on services and licensing fees as the primary source of revenue generation. With the Internet breeding a generation of developers and consumers that expect things for free or nearly free, how do you ensure ROI? In addition, while the smartphone, cable, and utilities markets have matured to the point where providers can afford front-end revenue hits on hardware in lieu of lucrative service payouts over time, in the fledgling IoT it’s hard to rely on commitments to long- or even short-term commitments at the expense of a large upfront payday. Especially if you’re a small IoT startup, asking a group of angel investors to risk bankrolling today’s IoT devices in exchange for the uncertain promise of tomorrow’s data- and software-driven dollars seems like a prayer.

This leaves IoT developers at a crossroads, as although the increasing amount of value and a more economic approach to electronic system design is now rooted in software, capitalizing on that value has largely been restricted to traditional, hardware-centric ROI models (Figure 1).

Management and control are key to monetizing IoT

By its very nature, the IoT is based on the real-time or near-real-time delivery of software and services, so getting capabilities to the end user is not the issue. Rather, the problem is one of control and management – control that ensures IP can’t be stolen or reverse engineered so potential users are able to take advantage of features for free, and management that facilitates the distribution of software and services in such a manner that data or feature utilization can be monitored for appropriate billing (Figure 2).

Business Model Versatility

Traditional Sales Models

- ✕ Product Sale
- ✕ Update/upgrade sale
- ✕ Maintenance
- ✕ Consumable re-fill
- ✕ Renewal

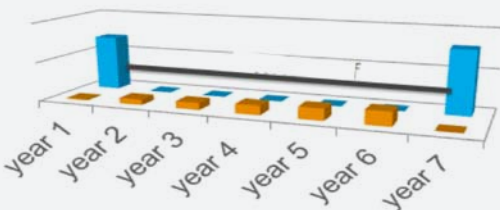


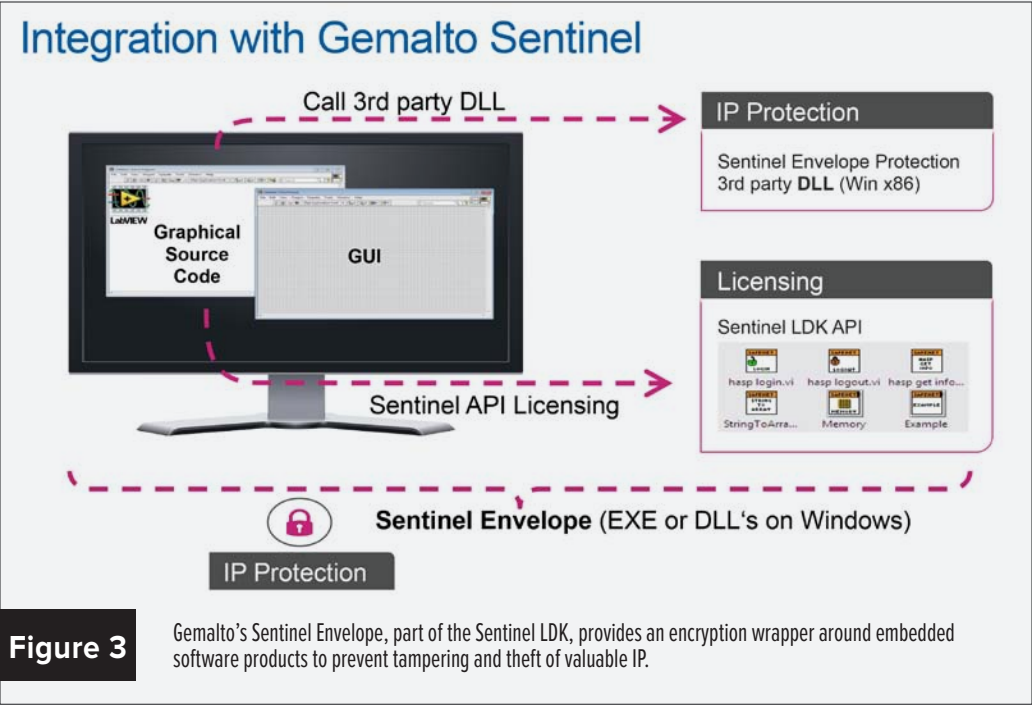
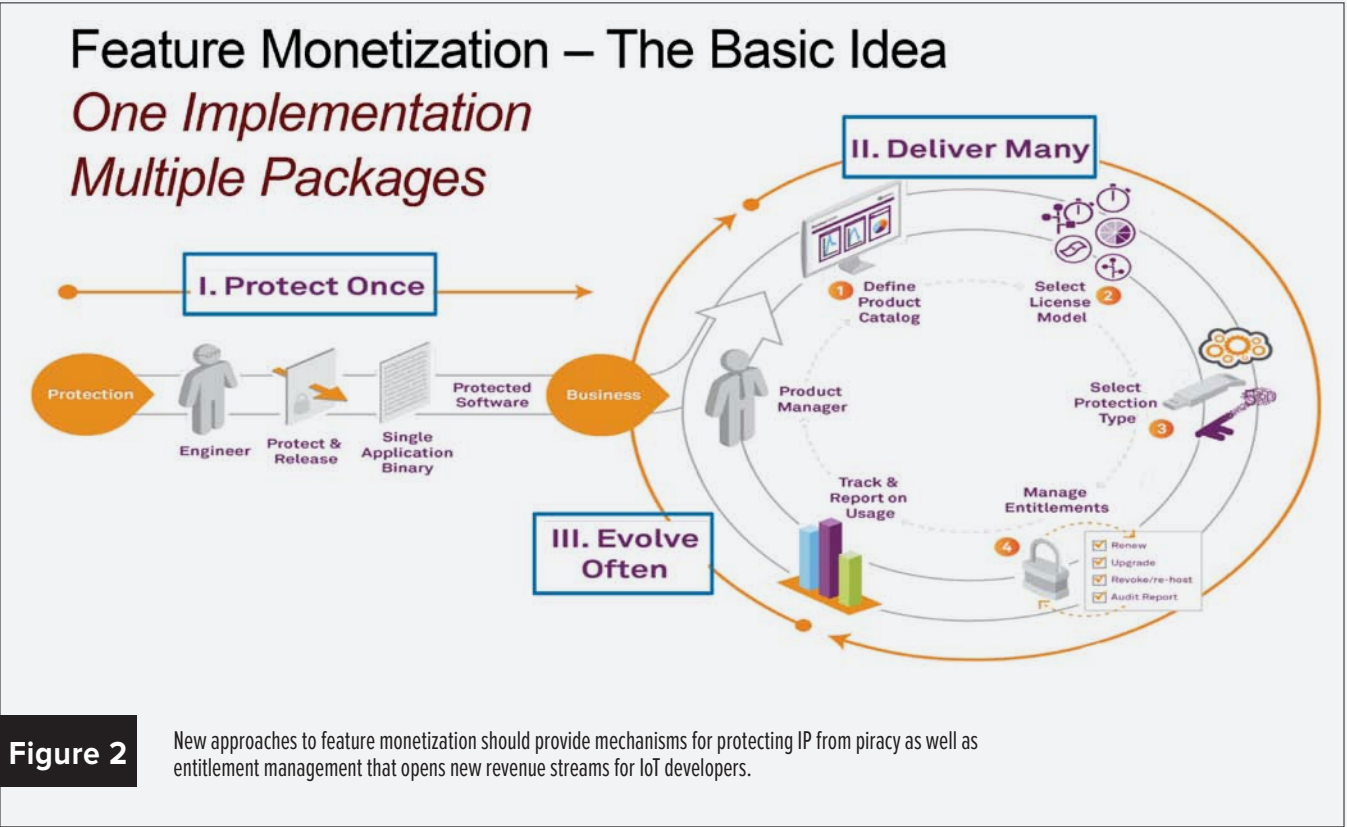
Figure 1

Traditional product-centric business models limit the earning capacity of Internet of Things (IoT) solutions as they are typically earmarked by a single income phase followed by extended periods of service and maintenance costs.



In a meeting with Aurelius Wosylus, Director of Business Development at Gemalto (www.gemalto.com) earlier this summer I was introduced to the company's Sentinel Licensing Development Kit (LDK) that addresses these challenges in multiple ways. From a security and IP protection perspective, the Sentinel LDK represents an evolution of Gemalto's

licensing protection products into a single "Cross-Locking" technology suite that allows developers to implement a combination of hardware and/or software-based encryption. In software, keys are exchanged using the Sentinel Envelope, a wrapper that uses code obfuscation and system-level anti-debugging, among additional features, to control access



to executables, data link libraries (DLLs), and other software data files so they can only be decrypted by authorized parties (Figure 3). Hardware-based encryption technologies such as AppOnChip can also be utilized for applications that require authentication via hardware tokens.

Once software IP is secure, developers are able deploy software features to target devices with confidence, enabled by the Entitlement Management System (EMS) built into the Sentinel LDK. A web-based solution built on an SAP backend, the

Topology of Monetization of Features using Remote Activation

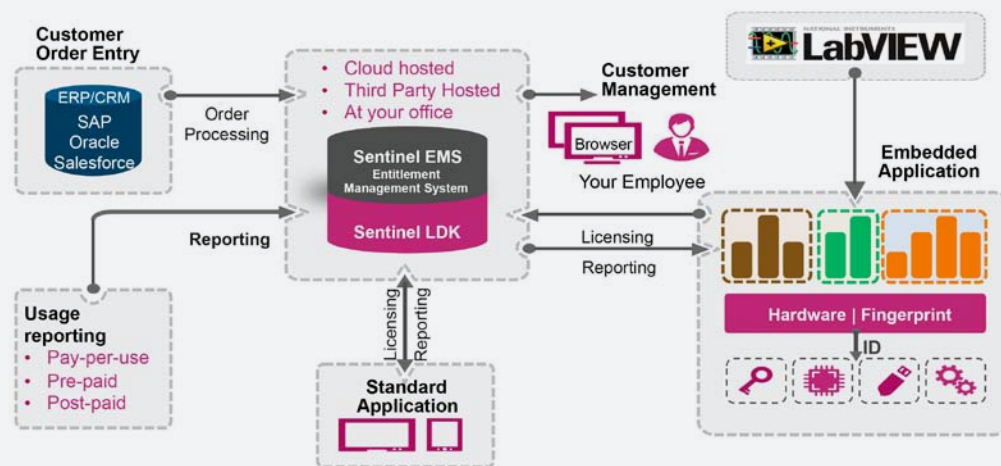


Figure 4

The EMS features of Gemalto's Sentinel LDK permit flexible remote software licensing through reporting that can be used to support pre-pay, post-pay, and pay-per-use models.

EMS allows software updates to be remotely pushed to target users and devices while also retrieving usage metrics that make flexible licensing models possible, including pay-per-use, pre-pay, and post pay. In addition, usage data can also provide insight into the behavior users exhibit when interacting with various products, which in turn can be leveraged for future R&D, marketing, and so on. Figure 4 shows an example software monetization architecture using the Sentinel LDK and

embedded industry the general consensus is that while consumer IoT is currently driving about 80 percent of IoT's hype, the "enterprise/industrial IoT" will amount to 80 percent of its actual value. However, nebulous projections like this don't answer the core question of how IoT companies will survive, whatever the flavor. Software monetization strategies are a big step toward getting black figures on the bottom line sooner rather than later. **ECD**

National Instruments' LabVIEW application.

With control and management technologies serving as the foundation, developers can take a new approach to product definition that reduces hardware dependency and enables innovative software packaging and licensing models, many of which can provide recurring revenue streams.

A pragmatic approach to the bottom line

In my travels around the

JOIN THE EVOLUTION.



Learn more

Get "mobile smart" in 3 easy steps:

- 1 Get your **AIR for Wiced Smart** dev kit at your distributor of choice. (See our website for a current list.)
- 2 Develop your wireless link and basic app using our exclusive **Atmosphere** development tool.
- 3 With our AIR for Wiced Smart module on board, proceed in record time to a prototype and final, mobile-app development!



Evolve to app-based control with AIR for Wiced Smart!

If you're ready to evolve from fixed control panels populated with dials, buttons, keypads, and LCD displays to mobile-app based control of your embedded product – check out Anaren's AIR for Wiced Smart module, featuring Broadcom's Wiced Smart Bluetooth® chip (BCM20737). Not only does our small-footprint, SMT, and pre-certified all-in-one module save you the time, effort, and trouble of designing your own radio... It's supported by our industry-exclusive Atmosphere development ecosystem that lets you develop your basic embedded code and app code in one, easy-to-use development tool – for a far speedier product development cycle and time-to-market. Follow the steps at left to join the evolution, right now!

www.anaren.com/AIRforWiced
800-411-6596
In Europe: 44-2392-232392

Anaren®
What'll we think of next?™





Microcontroller-based FPGAs hit the mark

By Tedarena

FPGA vendors have been purposely pushing a growing divide in FPGA architectures. The main suppliers have gravitated to either SoC FPGAs with high-performance application processors or offer low-end FPGAs with no processor. Applications such as server farms and high-performance computing require ever-growing increases in performance. To address this, many SoC

FPGAs are incorporating ARM A-class application processors, which are well-suited for those applications. The reality for a multitude of other applications is that these processors are more than what is required. A-class SoC FPGAs are often too expensive, require too much software support burden because of the required operating system, and feature higher power consumption. Additionally,

the raw compute performance is often not required.

On the other end of the spectrum, low-end FPGAs with no hard microcontroller or processor can be limiting for numerous designs. Low-end FPGAs only offer soft microcontrollers and there is no accompanying peripherals or subsystem. If peripherals are required then

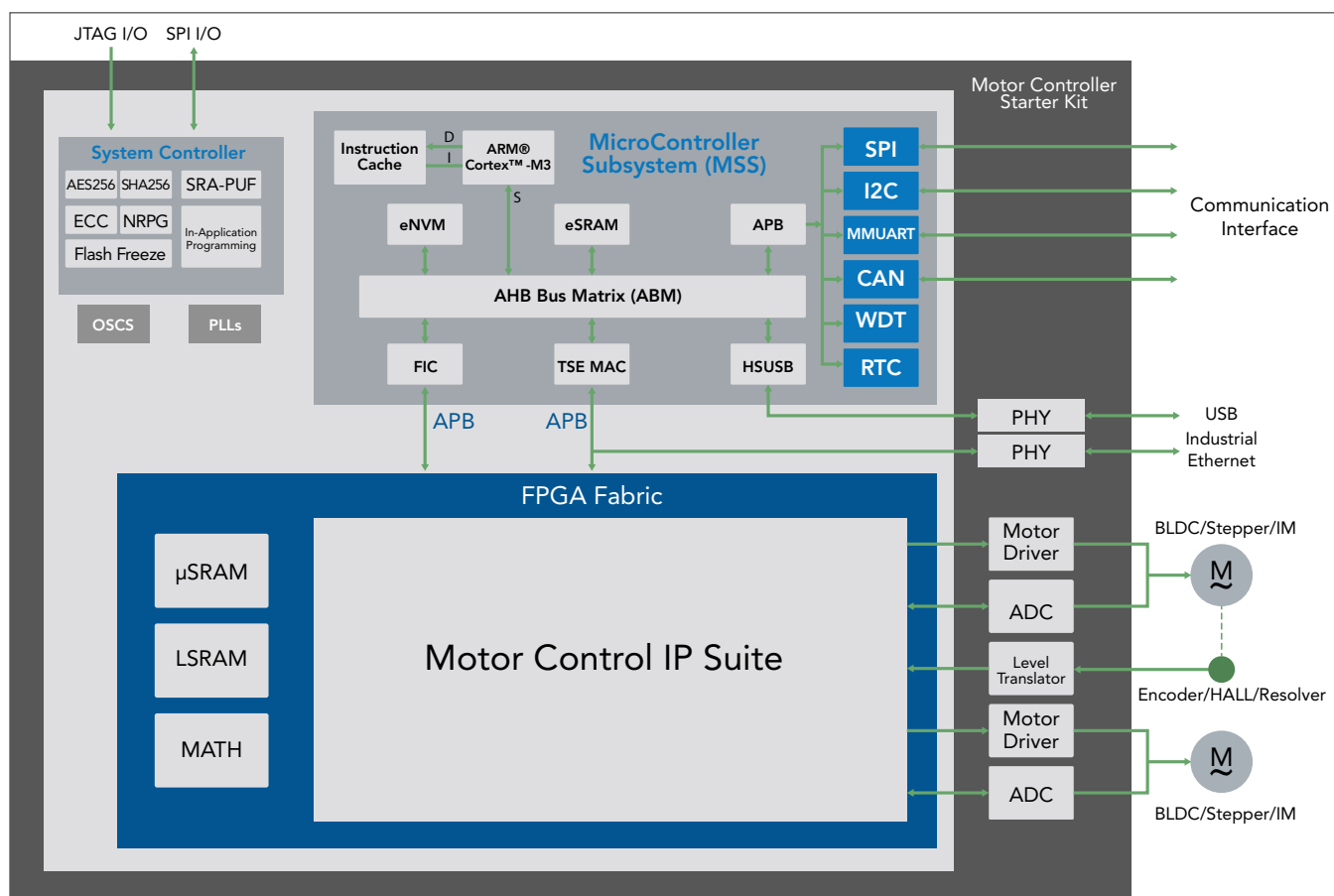


Figure 1 Block diagram of a multi-axis motor control design.

they must be created by using FPGA gates. Low-end FPGAs with a soft microcontroller run very slow, are inefficient due to the FPGA gates needed to create the microcontroller and subsystem, offer no security, and are developed on proprietary architectures.

For a broad class of applications, a hardened microcontroller-based SoC FPGA is often the more effective solution. A large amount of 32-bit microcontroller architectures are produced each year because there are numerous applications that can use them. Imagine if there was a microcontroller with a common subsystem of components and it also included a configurable block that could implement hardware acceleration tasks or other logic functions? An ARM Cortex-M3 SoC FPGA can be viewed as a microcontroller with configurable hardware acceleration. The hardware acceleration and implementation of logic functions are two key features where the FPGA fabric excels. Combined, an ARM Cortex-M3 and an FPGA fabric allow an ideal division of labor for many tasks in a wide variety of applications.

Partitioning design examples with processor and FPGA fabric

A microcontroller is ideal for slower-speed serial tasks because of the architecture and requirement to access memory for instructions. An FPGA fabric is ideal for parallel-processed functions that are more time critical. When partitioning a design in this manner, it becomes clear which functions each component should implement. For example, one of the main challenges of controlling multiple motors is the requirement for a deterministic response of the motor control loops. To ensure a reliable design, each motor must be serviced within a tight, deterministic time with no wide timing variation. Because the multi-axis motor control algorithm is a time-critical function, it should reside in the FPGA fabric. The FPGA is ideal to implement the control loops with tight deterministic timing. Figure 1 is a block diagram of a multi-axis motor control design. The bulk of the motor control algorithm is in the FPGA fabric, while the slower speed interfaces are connected to the ARM Cortex-M3.

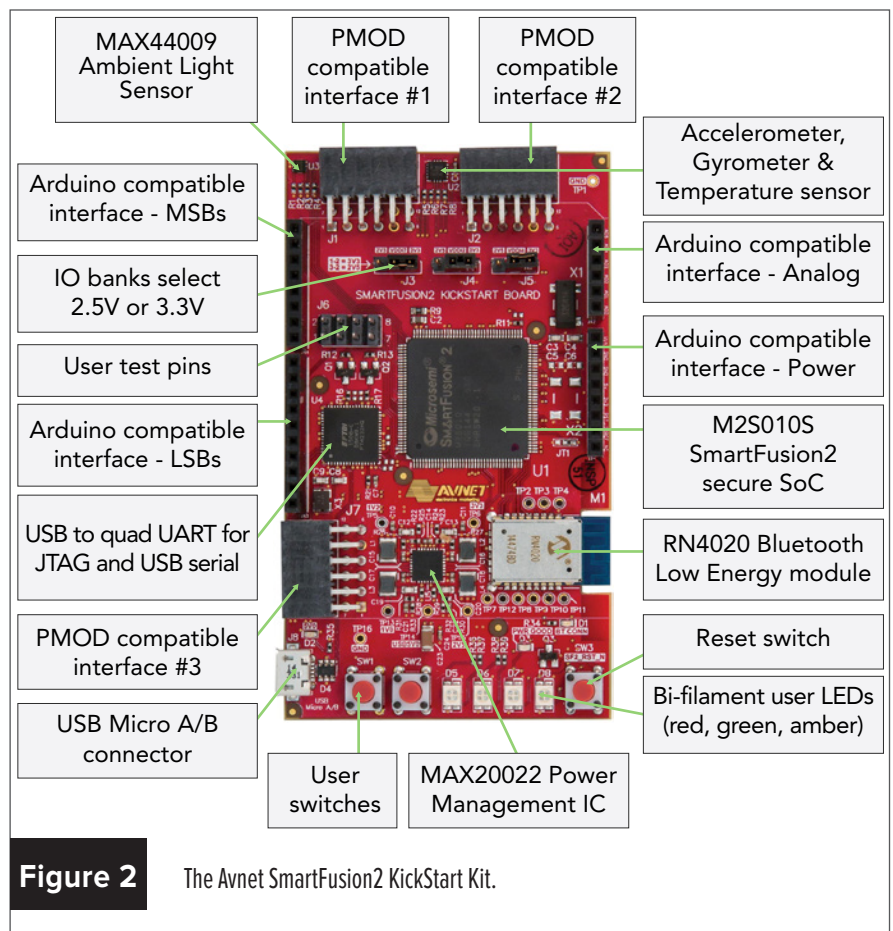


Figure 2 The Avnet SmartFusion2 KickStart Kit.

The motor control algorithm is not the only required function. Often one or more communications interface and control I/Os are required for a complete motor control design. These interfaces are not high-performance oriented and are ideal for a microcontroller such as the Cortex-M3 to implement. The communication interfaces could be CAN bus, SPI, UART, or other control buses.

Adding data security to a design

The addition of data security requires a certificate and key exchange, as well as implementing encryption and decryption algorithms on the data. There are multiple data security handshake exchanges and support for multiple formats is often required. The exchange handshake is typically not time-critical, but each exchange algorithm has different strings that need to be parsed, various verification protocols, and multiple rounds of certificate checking. With all of these procedures and the lack of time criticality, it is ideal to implement the certificate and key exchange in a microcontroller such as a Cortex-M3. The appropriate code can be called and executed depending on

the handshake exchange requested. Once the keys are exchanged and both devices are trusted, the data communication will need to be encrypted and decrypted. There are many wired and wireless communication speeds and protocols, and, depending on the link speed, performance is usually needed for the data encryption and decryption. When performance throughput is needed, it is recommended to implement it in the FPGA fabric. The transmitting device would implement an encryption algorithm such as AES 256, triple DES, RSA, or similar, and the receiving device would implement the same decryption algorithm. If the data link performance is not critical, the Cortex-M3 could implement the encryption and decryption. However, the FPGA fabric is ideally suited when higher throughput is needed.

Adding custom peripherals to a design

Although processors and microcontrollers offer common peripherals, numerous designs require custom interfaces. Many medical, industrial, and embedded designs often need to

add additional interfaces, and open bus interface connectors address this issue. Two of the more popular peripheral bus connectors are the peripheral module (Pmod) and the Arduino shield. There are many peripheral designs that leverage these connectors and a SoC FPGA is ideal to bridge between these custom peripherals and the rest of the design. A microcontroller-based SoC FPGA can be uniquely leveraged when additional peripherals are required. The combination of the microcontroller and the FPGA fabric can implement bridging, acceleration functions, communication protocol management, I/O expansion, and control logic. One example is a design that needs to add a custom display. To interface to the display will require interface control logic, image manipulation and also register configurations. The ARM Cortex-M3 can address the latter and do other house-keeping tasks, while the FPGA is best suited for interfacing to the display and modifying the image as necessary.

Avnet Electronics recently created a hardware kit to demonstrate the power that a microcontroller-based SoC can bring to the broad market (Figure 2). A number of peripheral options can be used with this board because it has both Pmod connectors and an Arduino shield connector set. Other key features include onboard peripherals, Bluetooth Low Energy (BLE), USB, and a host of sensors and switches. The board features the Microsemi SmartFusion2 SoC FPGA, which includes a hard ARM Cortex-M3 microcontroller and comprehensive subsystem. An HDL and C-code reference design, Windows-based software GUI, and Android app are included with the architecture for added flexibility. The Windows GUI can interface to all of the onboard components, and the reference design examples enable the ability to add on an Arduino shield or Pmod peripherals. When connecting to an Android phone using the Windows GUI, a full data security demonstration can be seen. This data security design implements the key

exchange as well as encrypting the data between the SmartFusion2 FPGA and an Android smartphone.

There are a wide variety of general purpose applications that are ideally implemented in a microcontroller-based SoC FPGA that enables solutions such as bridging, I/O expansion, hardware acceleration, protocol management, and board initialization to be addressed. Using a hardware solution with a reference design and a software GUI allows design ideas to quickly become reality.

Ted Marena is Director of SoC/FPGA Products at Microsemi.

Microsemi

www.microsemi.com

[@MicrosemiSoC](https://twitter.com/MicrosemiSoC)

www.linkedin.com/company/microsemi

www.youtube.com/user/MicrosemiCorp

Nuremberg, Germany
23 – 25.2.2016



embeddedworld

Exhibition & Conference

... it's a smarter world

Get up to speed with the latest developments
in your industry!

embedded world is THE meeting place for the international embedded
community – secure your crucial knowledge advantage now!

Trade fair organizer

NürnbergMesse GmbH

T +49 9 11 86 06-49 12

visitorservice@nuernbergmesse.de

Conference organizer

WEKA FACHMEDIEN GmbH

T +49 89 2 55 56-13 49

info@embedded-world.eu

embedded-world.de

Media
partners

elektroniknet.de

computer-automation.de

ENERGIE
& TECHNIK
Solutions for a Smarter World

DESIGN &
ELEKTRONIK
KNOW-HOW FOR ENTWICKLER
MEDIZIN-elektronik.DE

Elektronik
FACHMEDIUM FÜR INDUSTRIELLE ANWENDER UND ENTWICKLER
Elektronik
automotive
FACHMEDIUM FÜR PROFESSIONELLE ANWENDER UND ENTWICKLER

Markt & Technik
Computer &
Automation
MEDIZIN-elektronik
Spezialwissen für Fachleute in der Medizintechnik

NÜRNBERG MESSE



The theory behind product line engineering

By Curt Schwaderer, Editorial Director

cschwaderer@opensystemsmedia.com

At first glance, product development may seem like the most complex component of the product life cycle. However, everything developed from the silicon, circuit boards, and software have features, options, and derivatives that need to be identified, productized, and maintained. Managing the myriad of variations within a product line is often called product line engineering (PLE). In this column, we'll explore PLE in more detail and look at a company with a unique paradigm and solution for lowering complexity and increasing reliability within product lines.

In order to be competitive in today's environment, companies must deliver a product line, not just a single product with a mentality that "one size fits all."

What is product line engineering?

In general, product line engineering (PLE) refers to the practice of creating an underlying architecture (both hardware and software) that describes the base platform. The architecture describes the base commonality across the product line as well as planned variations. PLE focuses on the process of engineering new products so it is possible to reuse product assets and flexibly apply a subset of variations for a specific product with minimal cost and time spent.

A Carnegie Mellon SEI study in 2003, among many others, has shown that PLE can have several benefits including higher productivity within the organization, higher quality products, faster time-to-market, and lower labor requirements.

Challenges implementing efficient PLE

PLE is not something a "brute force" process can solve. For example, a rather simplistic embedded system with

a circuit board that can be purchased with two CPU options and software with three add-on features yields six possible product variations. With each feature addition, two additional product variations are created. Adding connectivity variations to this example creates an "M x N x O" variations problem. With each additional variable, the product variations can quickly grow out of control, causing duplication, increased development and test time, and even less reliable products.

There is significant complexity to managing all these variations. The example above was simple. Applying the same principles to autonomous vehicles that have a variety of options for radars, cameras, sensors, and electronic steering makes this a tremendous challenge.

The theory behind PLE

Systems and software PLE creates and maintains a portfolio of related products using a shared set of engineering assets and an efficient means of production.

I talked with Dr. Charles Krueger, CEO of BigLever Software (www.biglever.com) about their approach to PLE. Dr. Krueger says their PLE solutions take a "factory view" of the engineering process.

"Think of an analogy of a factory," Dr. Krueger says. "Assembling and producing requirements and specifications is one dimension. Think about the things coming into the factory: the traditional artifacts like technical specifications, subsystem designed, bills of material, software, user documentation, calibration data, test cases, certifications... The list is long. We want to get really good at reusing these artifacts."

Dr. Krueger cites a familiar example of the user's manual in the glove box of an automobile.

"Open [the manual] up and it's full of things that say 'if you have this kind of option, do this; otherwise do that.'" Dr. Krueger says. "Most of the time I have no idea if that option is on my car. If PLE can customize the product [auto] down to the user's manual in the glove box so we don't have to guess at these if/then statements, things are cleaner and easier."

Two key abstractions

BigLever uses two key abstractions in their PLE solution: a feature catalog and a bill of features.

These abstractions are key to corporate success in managing the PLE process.

The key to success here is, at a corporate level, everyone in the organization buys into the utilization and management of the feature catalog for a product line and a bill of features must exist for each unique product within the product line.

The feature catalog is a corporate-wide asset that lists the available features for the product line. Features describe things that are optional within the product family. Product managers can choose to include or exclude any feature in the feature catalog for a given product within the product family.

The “bill of features” is a description the product manager puts together that details exactly what features go into the making of a specific product within the product line.

The BigLever “Gears product configurator” takes the bill of features description and takes the asset descriptions that come from the supply chain to output a specific product asset subset for a specific version of the product.

This includes the product bill of materials, test cases, feature specifications, documentation, and anything else relevant to the included features for this specific product. The Gears configurator differentiates between assets needed for internal production and those that are included in the end product. For example, software source code and test cases are identified as assets needed to build the features included in the product, but only the executable binary and no test cases are included in the final product delivery.

Applying factory point of view to PLE

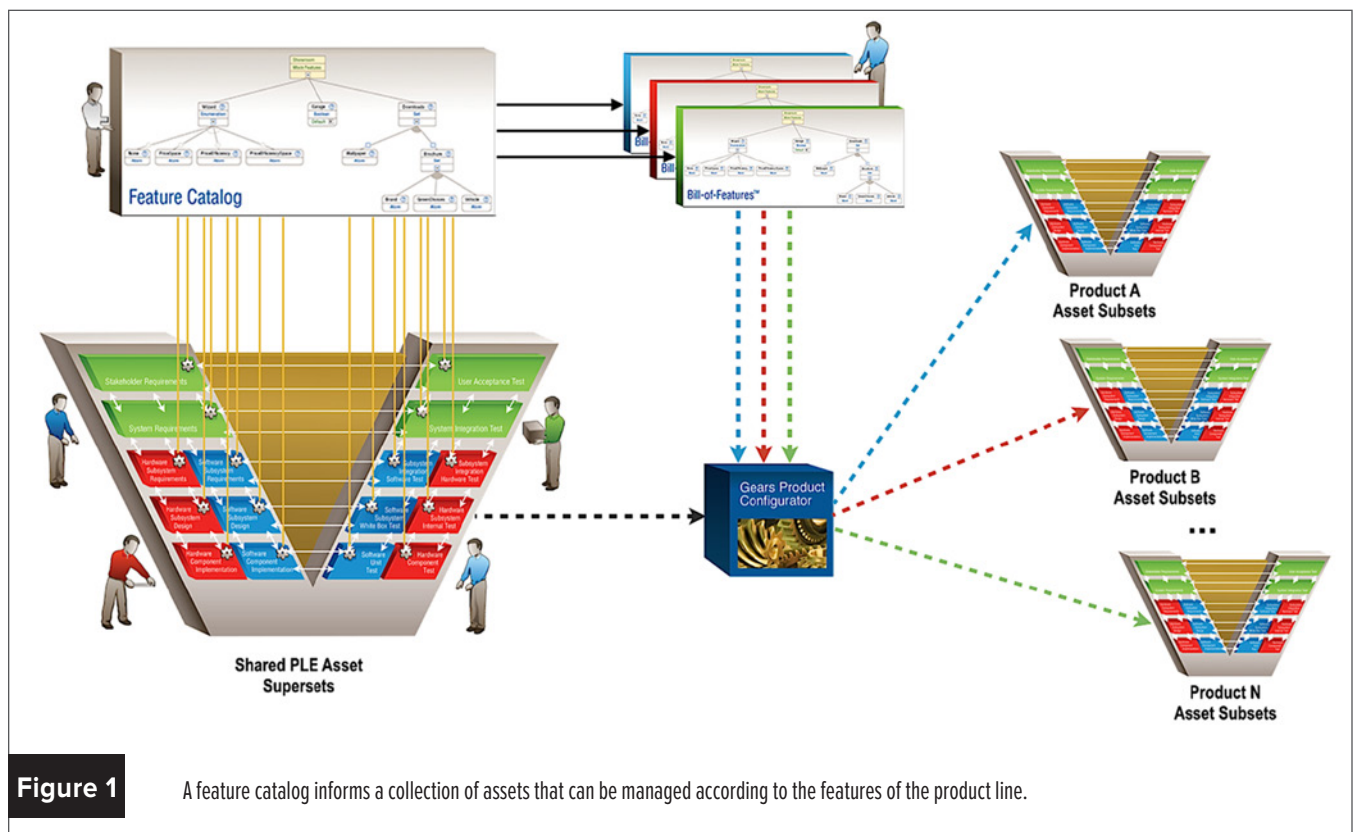
Figure 1 illustrates the factory-based PLE approach as applied to the V-model product life cycle. The V-model demonstrates the relationships between each phase of the development cycle and its associated testing phase.

The diagram shows a feature catalog that informs a collection of assets that can be managed according to the features of the product line. These may be a long list of engineering artifacts – bill of materials,

descriptions, software, system requirements, test cases, and wiring harness instructions are some of the common artifacts in an automotive product.

The feature descriptions go into the PLE process and are used to engineer the feature profiles that cause the products to come out. You’ll notice in the diagram the comprehensive nature of the feature catalog as it feeds into both the development engineering and testing phases of the V-model.

In order to create a specific product, we need a way to filter all these assets. How do we know what to do within the configurator? This is where the bill of features specifications come in. If the feature list includes a feature, the configurator will include that in the assets, based on the bill of features. The configurator uses the bill of features to associate the pieces within the feature profiles to know what to include/exclude from the assets. This drives every engineer working on their part of the system to think about how their assets get used on all the variants



of these products within the family. Engineers become skilled at specifying these feature-based variations, and that results in efficiencies up and down the V-model.

"We want all the engineers to create features by accepting the single source of truth throughout the product family," Dr. Krueger says. "Anywhere there is variability the organization needs to make sure there is a single, common understanding of what the varying features are."

In order for the process to work, there must be an organizational view of PLE. Dr. Krueger mentions when they begin working with a company often, they are changing the fundamentals of how the organization does their engineering. The BigLever goal is to elevate the organization to this factory point of view. The new engineering mindset is they are creating artifacts that get put into the factory. But the training permeates across all disciplines of the organization in order to be effective.

The first level of decision involves the feature catalog – what features do the organization want to support? Orchestration between engineering, product management and executive management is critical here.

Once the feature catalog is set and the proper bill of features development is understood, engineers can be turned loose. The key concept here is to respect the feature catalog and develop accordingly. All experts are still doing their jobs, but they are doing so realizing there are different flavors defined by the feature catalog.

The second important discipline involves a portfolio team. This is the team that is deciding which specific features are to be included for a given product instance. These people are assembling according to the feature catalog and, in some cases, clustering by sub-families of these features.

The configurator operates on each of the bill of features and a unique V model pops out and pulls out the right assets automatically so every feature specified

to be in the product will be included. Once this process has been followed, the automation of assembly and production enables additional efficiencies.

At the end of the day, the PLE process can bring an enormous increase in efficiencies, resulting in competitive advantage from the process itself in addition to the feature set.

PLE and Agile development

The V-model example may cause the impression that PLE is only applicable to the waterfall mode, but Dr. Krueger says that PLE is development-agnostic.

"Agile processes still have assets [definition of done, user stories, sprint reviews], and these assets can be handled in the same way within the factory view of PLE," Dr. Krueger says. "The development of each feature may be coming from a user story, but the asset result is the same."

Real-world factory view PLE examples

The automotive market is finding a significant benefit with a factory view of PLE. General Motors faces one of the most complex PLE challenges in terms of product complexity, multiple options, and high reliability. They call this the "Mega-scale product line engineering" problem. GM produces on average one vehicle every three seconds somewhere around the world. Each one is considered a member or variant of one single big family of vehicles in the GM organization. They are adopting BigLever solutions to address the massive feature variants their product line entails.

In the military market, Lockheed Martin and Navy have adopted a "fix it once" initiative. They have 100 ships and it's possible that a defect might go out and be found on a single ship. Once found, there is a cycle to fix it. Later, this same defect may pop up on another ship and the same effort happens. PLE can facilitate a paradigm where you find it once, fix it once, then apply the fix across the entire product line that includes the feature with the defect. This is an extremely valuable part of the solution.

The Lockheed Martin integrated weapons system is deployed in more than 100 ships. By managing a single "family of ships" with variants as defined by a bill of features, they can leverage the factory view of the PLE process. They reported a \$47M cost avoidance per year using the BigLever PLE solution and dramatically increasing the productivity by using the Gears approach.

Sound PLE also results in competitive advantage. Lockheed bought in and got good at this solution, and as a result their productivity has increased dramatically. By leveraging the efficiencies in the process, they can deploy projects at lower cost and deliver the same or a superior feature set. In addition, they have capacity to continue advancing and upgrading their capabilities.

In another example, General Dynamics teamed with BigLever to create the winning proposal for the U.S. Army Live Training Transformation "Consolidated Product Line Management." In this product line of training systems, each soldier has a personal area network connected to a larger network of up to 2,000 soldiers. This can be seen as an interesting Internet of Things (IoT) application. They are monitoring all these data points during maneuvers to detect patterns of activity of individuals and units that they can use as learning and training information to train soldiers on how to be effective and stay alive in the field. Using the BigLever PLE solution, the Army study produced audited numbers that project \$600M in cost avoidance over 15 years.

What about small/medium businesses (SMBs)?

The examples cited were from very large organizations with big budgets, but Dr. Krueger claims the factory view of PLE can still be cost effective for start-ups and small businesses alike.

"The pricing scales by the number of people. And smaller organizations are typically faster to adopt the process because there is less to change and fewer existing blocks to overcome," Dr. Krueger says. "In SMB organizations, saving time for a few key people can make a world of difference in the organizations' growth potential." **ECD**



Look Ma, no code! PWM control in under five minutes with Microchip Curiosity development board

By Brandon Lewis, Assistant Managing Editor

Goal: The following describes how to build a simple PWM for motor control applications on the Microchip Curiosity Development Board using the MPLAB X Integrated Development Environment (IDE) and MPLAB Code Configurator (MCC) software. We begin with a basic bring up of Curiosity's onboard PIC16F1619 8-bit MCU running a "Hello, World!" program, and continue to set up a PWM using a Timer based on an FOSC/4 clock. All of this is done using the Microchip MCC without ever writing a single line of code.

Difficulty Level: Beginner

Hardware required:

- > Microchip Curiosity Development Board (Outfitted with PIC16(L) F161X MCUs – Used here is the PIC16F1619)
- > USB 2.0 A-Male to Mini-B Cable

Software required:

- > Microchip MPLAB X IDE version 3.05 or later (Used here is MPLAB X IDE version 3.10; Download from www.microchip.com/pagehandler/en-us/family/mplabx/)
- > MPLAB XC Compiler (Used here is MPLAB® XC8 Compiler version 1.35; Download from www.microchip.com/pagehandler/en-us/devtools/mplabxc/home.html)
- > MPLAB Code Configurator version 2.25 plugin (Download

from www.microchip.com/mymicrochip/filehandler.aspx?ddocname=en576270)

Recommended platforms:

- > Windows (x86/x64)
- > Linux 32-bit and Linux 64-bit (32-bit compatibility libraries required)
- > Mac (Versions 10.X)

Support/documentation:

- > www.microchip.com/curiosity
- > www.microchip.com/support

Cost estimate:

- > Microchip Curiosity Development Board – \$20 (Before tax, shipping & handling. Available from multiple distributors and www.microchipdirect.com.)
- > USB 2.0 A-Male to Mini-B Cable – \$5 (Rough estimate. Available from multiple distributors.)

Powering the board and setting up a project

After downloading and installing the necessary software on your computer (installation instructions can be found at www.microchip.com/curiosity), plug the USB 2.0 Mini cable into the J2 connector located on the bottom of your Curiosity board. Doing so will illuminate the green LED D2 located near the center of the board to indicate that Curiosity is powered on, as well as LED D1 in the bottom left corner signifying that a +3.3V power supply is available on the board. If you

haven't already, move the shunt jumper to the right two pins of the J12 jumper to connect a +5V power supply (the left two pins connect +3.3V).

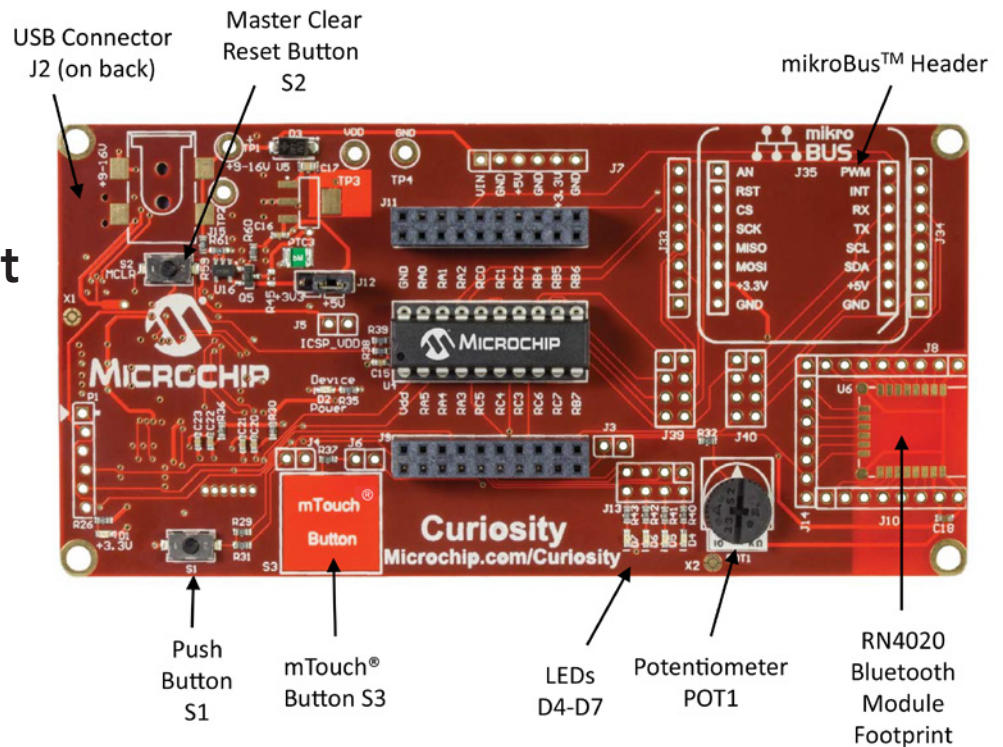
You're now ready to program. Open the Microchip MPLAB X IDE from your applications folder, and click "New Project" in the upper left corner of the interface, or select it from the "File" drop-down menu. Doing so will open a "New Project" window.

From the Microchip Embedded folder, double click "Standalone Project." This will direct you to the next step in the project wizard, which prompts you to select a device. The Curiosity development board comes stock with the 8-bit PIC16F1619 microcontroller, so select it by either typing the model number into the "Device" dropdown, or by selecting "Mid-Range 8-bit MCUs (PIC10/12/16/MCP)" in the "Family" field and then opening the Device dropdown and scrolling until you find the PIC16F1619. Hit "Next."

You'll be prompted to select a debug header, but you can leave that with its default settings, so click Next again.

In the next step of the Wizard, you'll be asked to select a tool, so navigate to "Curiosity – FN: XXXXXX" under the "Microchip Starter Kits" folder, and double click. This will generate a popup window that allows you to give

Microchip Curiosity Development Board



the board a human-friendly name. For the purposes of this exercise, just call it "Curiosity." Hit Next.

At this point we have to select a compiler. You should have already downloaded and installed Microchip's XC8 compiler, either the free or the Pro version. The Pro version offers some enhanced optimization features, and if you'd like to try it out Microchip offers a free 60-day trial. But for the purposes of this project, the free version will work just fine. Select the XC8 compiler from the "Compiler Toolchains" folder. Hit Next.

Now you're going to name your project. Call it CuriosityPWM. Leave all the default settings as they are, and click "Finish." A project folder should now appear in a column on the left of the MPLAB X interface.

Before getting started on the actual application, you have to configure the project to support low-voltage programming, which allows you to program the MCU without a 12 V power supply. To do this, simply right-click on your project, and select "Properties" from the subsequent dropdown. A popup window will appear that has a list of "Categories" on the left. Under the "Conf: [default]" parent, click "Starter Kit (PKOB)." The

options on the right side of the window will change, revealing a dropdown next to the heading "Option categories." Choose "Program Options" from that menu, and then check the box that says "Enable Low Voltage Programming." This may already be checked, but just highlight it anyway and click "Apply." Hit "OK."

Hello, World!

You're ready to roll, and can start by running a basic "Hello, World!" program just to make sure everything is working properly on the board. From the top navigation menu, go to Tools -> Embedded -> MPLAB Code Configurator. Several windows will populate to the right of the "Projects" section with project and device resources, a pin manager, and a main dashboard.

The first thing you'll want to do is make sure that low-voltage programming is also enabled on the MCU, and to do this just select "System Module" from the Project Resources window, navigate to the "Registers" tab that appears in the main dashboard, scroll down to "Register: CONFIG2," and change the LVP field from "High-voltage on MCLR/VPP must be used for programming" to "Low-voltage programming enabled."

Now, add a GPIO from the Device Resources window viewable under your project resources by finding "GPIO::GPIO" and double clicking it. This will add a GPIO to your project, and open the Pin Manager window on the far right. Open up the port for LED D7 that will run Hello, World! by closing the lock icon on the PIC16F1619's RC5 GPIO pin that controls that LED.

From here, select "GPIO Module" in the Project Resources window so we can configure LED D7 to illuminate when the application code is generated and flashed to Curiosity. When you have that selected, RC5 configuration options will appear in the PinManagerPaneViewer in the center dashboard, and here you will want to check the box labeled "Output" as we're going to be outputting to LED D7, as well as the "Start High" box to indicate that the pin should start running on high (Figure 1).

Now, hit "Generate" at the top of the main dashboard to generate the Hello, World! code. A popup will appear advising you that MCC hasn't detected a main.c file, and ask if you would like to generate one. Click Yes, then hit the "Make and Program Device" button (code box with downward arrow) in the MPLAB X top toolbar. The code will load

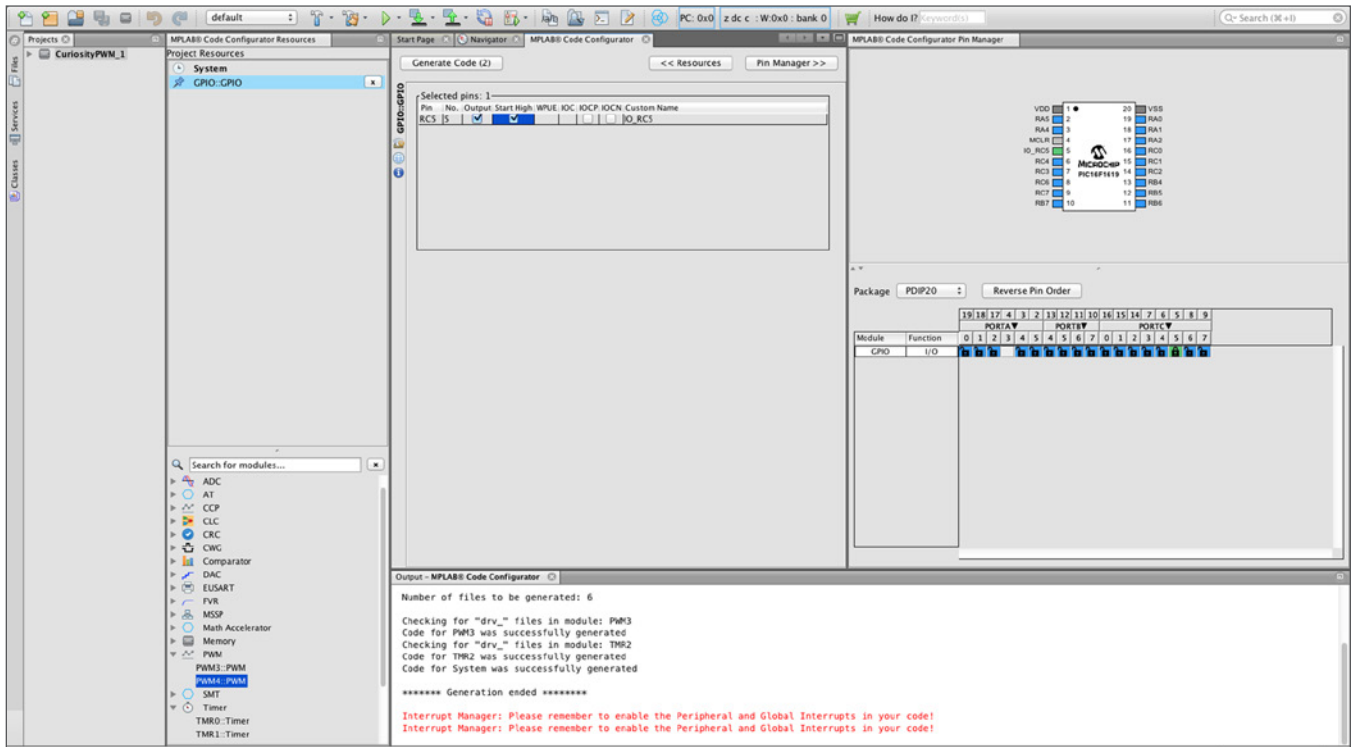


Figure 1

After adding the RC5 GPIO pin to your project, select "Output" and "Start High" to configure LED D7 to run the "Hello, World" application.

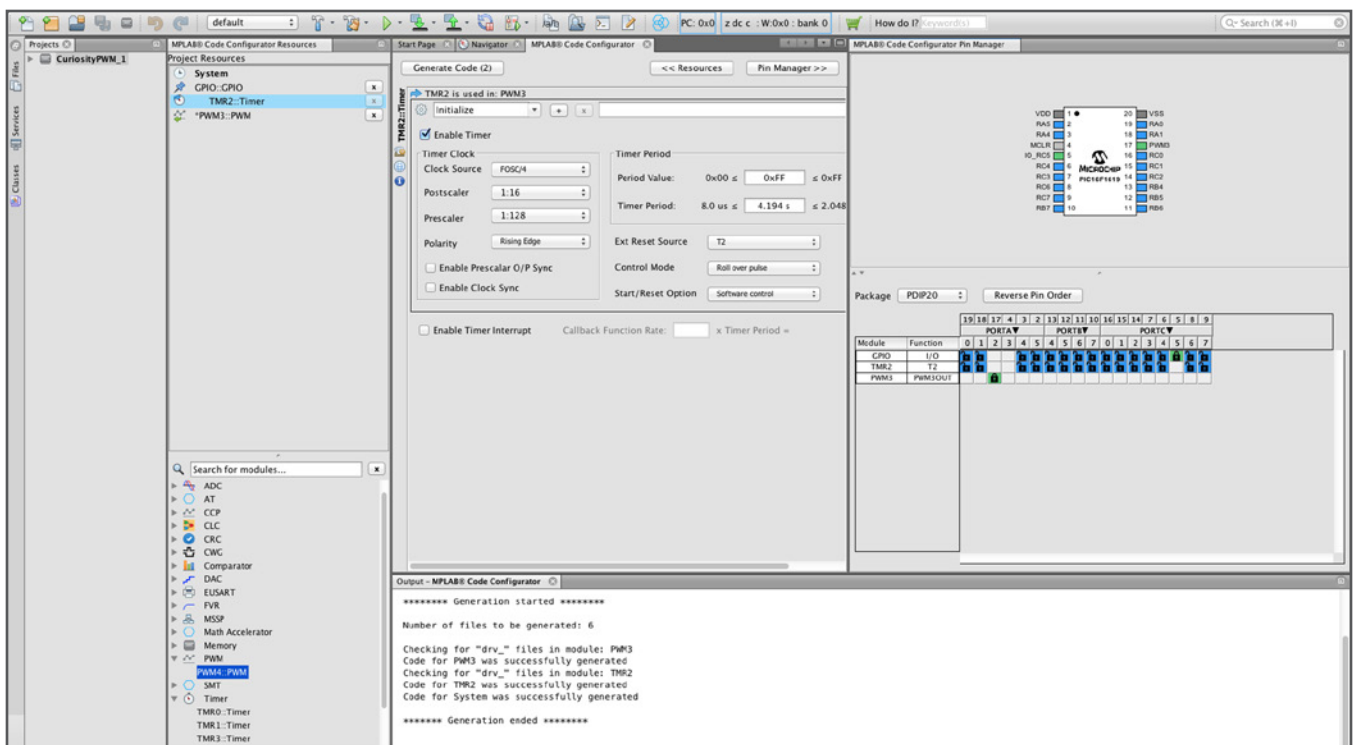


Figure 2

The timer period will vary significantly based on the parameters of your motor control application, but for the purposes of having our PWM visibly report on LED D6, we will configure ours to more than 4 seconds.

to Curiosity, and LED D7 will be statically lit in a matter of seconds.

PWM control

With confidence that Curiosity's hardware and all of your software are working correctly, you can now move forward with building the PWM. You'll be able to see the PWM in action by pulsating LED D6 in conjunction with the PWM's timer period, so it's time to get configuring.

In the "Device Resources" window, which should be located in the lower left of the MPLAB X interface, add the PWM3 peripheral to your project by double clicking. This will automatically add the Timer 2 (TMR2) peripheral to your project as well.

Now that the appropriate resources have been added to your project, you'll want to make sure that PWM3 is using TMR2 as the clock source. It should default to TMR2, but just to be sure, select PWM3 from the Project Resources window, and under the Easy Setup tab make sure that the "Enable PWM" checkbox is marked and that Timer2 is populated in the "Select a Timer" field. We won't need to adjust anything else in the PWM peripheral for our purposes, so just leave all the values at their default settings.

At this point, toggle over to the TMR2 peripheral under Project Resources where you'll be able to manipulate the clock and timer settings that will govern our PWM. Make sure that the "Enable Timer" checkbox is ticked.

Since the PIC16F1619 system clock defaults to FOSC (the system clock and oscillator can be changed by selecting the "System Module" under Project Resources), leave the TMR2 clock source at FOSC/4. What you do want to modify, though, in order to observe the PWM function operating on LED D6 are the prescaler and postscaler ratios, which help define the ratio of timer overflows to pulses. These ratios, along with other timer settings and clock configurations will obviously vary based on the control application, but for the purposes of this build just max them out to 1:128 and 1:16, respectively, in order to maximize the timer period so that the PWM pulses can be observed on the Curiosity board with the naked eye.

In the "Timer Period" field, enter a value of "4 s" to push the timer period up further, and leave everything else at its default setting. This will give us a PWM period of roughly 260 ms – on the high side for many control applications, but you get the idea (Figure 2).

Back in the Pin Manager view on the right, you will now notice that several resources have been added, among them the PWM3 and TMR2. To enable the PWM we'll need to open up the RA2 pin by clicking the PMW3 lock there. You don't need to link TMR2 since the PMW3 is already using that as a baseline.

Generate code. Make and build program. Watch for LED D6 to pulsate. At this point, LED D7 should still be statically lit from the previous application, with LED D6 pulsating next to it. If you want to turn LED D7 off at this point, deselect the RC5 lock in the MCC Pin Manager, regenerate the code, and remake/rebuild the application.

Conclusion

If you haven't noticed, not only have we not written a single line of code to generate a working PWM control application, but we also haven't checked a datasheet once. This is possible because MCC loaded in all of the PIC16F1619 MCU and Curiosity board parameters when we configured our project, saving time that would have been spent poring through PDFs.

If you're interested in exploring additional capabilities, Curiosity includes a footprint for the RN4020 Bluetooth module, as well as application-specific add-on boards from MikroElektronika (www.mikroe.com/ click) called "Clicks" that connect through the mikroBUS socket and can also be configured using MCC. Additionally, a 3.0 beta version of the MCC is now available on the Microchip website (www.microchip.com/mcc) that adds additional algorithm libraries and networking stacks to enable more complete builds with minimal coding effort. **ECD**

For more detailed instructions, read an extended version of this article or watch a video demonstration at opsy.st/MicrochipPWMDemo.



Advanced and reliable IPC products

New

5th Gen Intel® Core™ i7/i5/i3 SBC

LE-37E 3.5" SBC **LP-174 Pico-ITX**




100x72mm

Qseven CPU Module

QE-E70

- Intel® J1900/N2930/E3845
- DDR3 2 or 4GB on board
- VGA, DVI, LVDS
- Four PCIe2.0 x 1
- USB3.0, USB2.0
- SDIO, SATAII, HD Audio



Intel® Celeron® J1900, N2930 & Atom™ E3845 SBC

LE-37D 3.5" SBC **LP-173 Pico-ITX**




100x72mm

LV-670 Mini-ITX **LN-D70 Nano-ITX**




120x120mm

TI OMAP-L138 DSP+ARM SBC

COMMELL DSP-L138 SBC



- LVDS, SATA, USB2.0, 10/100Mb Ethernet, Stereo Codec
- 128M DDR, RS232 Console, 8GB SPI flash, SD slot
- Ready for U-Boot, TI Linux DVSDK, TI DSP & SYS BIOS

PCIe Mini Card

MPX-2515 **MPX-24794G2** **MPX-210D2**





- CAN 2.0B Card
- 32-bit GPIO Card
- Dual Giga LAN
- MCP-2515
- CY8C24794
- Intel® I210
- MCP-2551
- USB 2.0
- IPMI, MCTP
- Windows SDK
- Windows driver
- Windows driver

www.comsell.com.tw

General Information: info@commell.com.tw

Welcome to be commell Distributor



Testing the test: How to use coverage metrics for more effective embedded software testing

By John Thomas

Timing is everything in software development. Just think about the timing of when a software defect is found. Found after a release, it can be a disaster, compromising people's safety and costing millions. Found before a release, the defect can be reduced to merely an annoyance.

This is why software testing has become an integral part of the development life cycle. Indeed, according to a 2012 survey by the Fraunhofer Esk Institute, not only is testing an important part of the software development process, for most embedded systems developers it is the most difficult part as well.

Testing can be difficult for many reasons, but one of the most fundamental challenges is measuring progress. Failure to track progress during testing with reliable metrics can waste time and result in a lack of software quality. It is tempting to try to ignore metrics and aim for testing everything, yet this

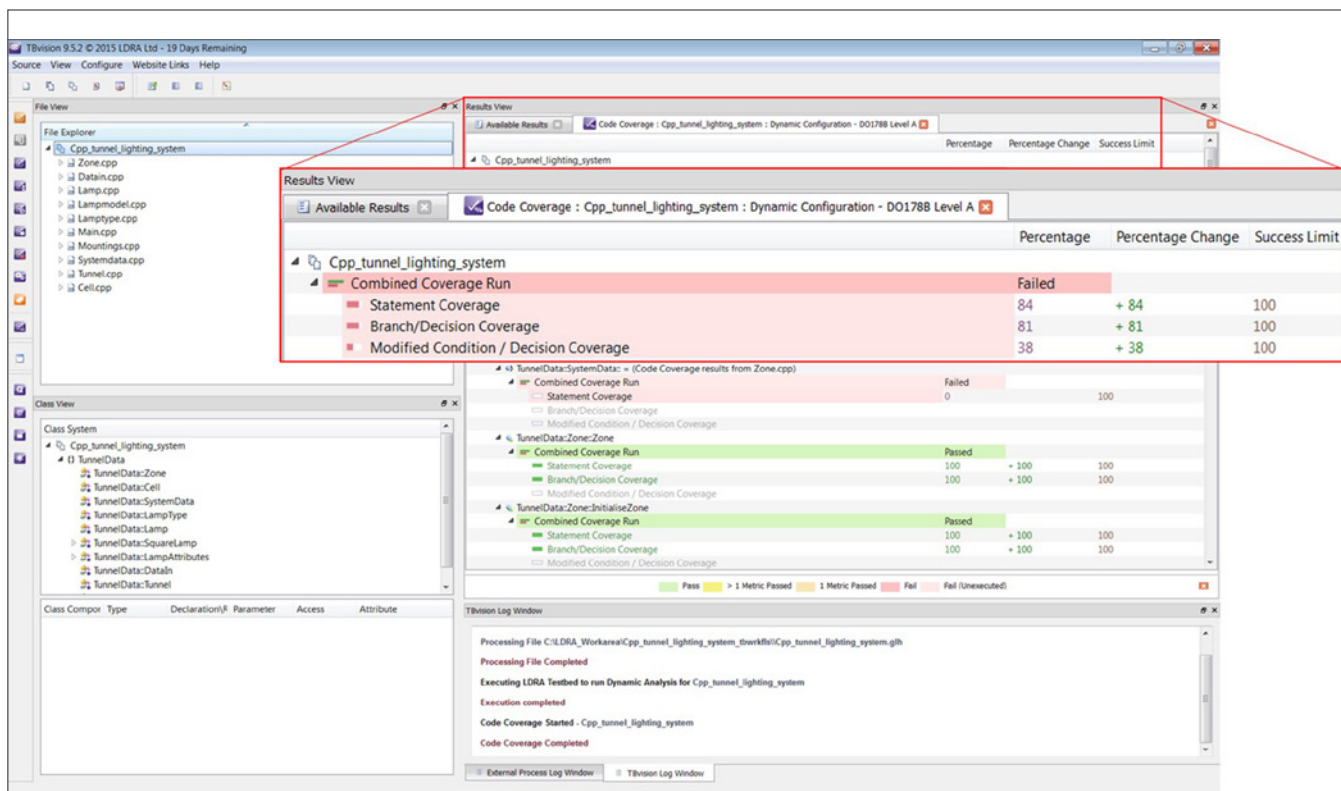


Figure 1

LDRA's TBvision code coverage results are displayed inline with system/file/function name to give a detailed overview of which aspects of the system meet the expected code coverage levels or metrics.

approach is dangerous since software testing has the potential to become an endless undertaking. Glennford Myers demonstrated this in 1976 in his book "Software Reliability: Principles and Practices," when he showed that a 100-line program could have as many as 1,018 unique paths. In the modern software development world, software can grow as large as millions of lines of code. This makes completely exhaustive testing infeasible.

In addition, it is often not just the software development team that needs to be satisfied with the level of testing. Customers may ask for evidence that the code was properly tested, and regulatory authorities for safety-critical industries, such as avionics, automotive, and medical fields, will want proof that there was sufficient checking for defects. Therefore, it is necessary to define a way of measuring "sufficient" testing, and it needs to be done in a way that can be measured objectively to satisfy all stakeholders in the development process.

For effective software testing, developers need to address how to measure the testing process, how to determine how much testing is enough, and how development teams can most strategically ensure that the software application they developed has been adequately tested.

What is code coverage?

Structural code coverage analysis is a way of looking at what parts of the logical structure of a program have been exercised, or "covered," during test execution. The logical structure depends on the code coverage metric being used. For example, "Entry Point" coverage looks at which function calls or "entry points" have been exercised in a test. Likewise, "Dynamic Dataflow" coverage looks at what parts of the data flow have been exercised. While different structural coverage metrics examine code from different angles, they all share a common purpose to give meaningful insight into the testing process by showing how much of the code is tested and which parts of the code have been exercised (Figure 1).

Specialized structural coverage metrics can serve special use-cases for testing,

such as analyzing data and control coupling. However, for measuring general test effectiveness, three code coverage metrics have found wide industry usage:

1. **Statement coverage (SC)** – How many statements of a program have been exercised
2. **Decision coverage (DC)** – How many branches of a decision have been exercised; this is actually a super-set of statement coverage, since for all branches of all decisions to be covered, all statements must also be covered
3. **Modified condition/decision coverage (MC/DC)** – This builds on decision coverage by making sure each of the sub-conditions of a complex decision is independently exercised in both its true and false states

These metrics have been widely recognized as ways to measure the thoroughness of testing. In particular, industries such as automotive, avionic, and industrial software have embraced these metrics in their software safety standards.

Higher criticality requires more thorough testing

Notably, these software safety standards do not mandate using statement, decision, and MC/DC coverage uniformly on all projects. Instead, each of the major industry software safety standards recommends using different levels of structural coverage depending on how critical the code is, although the level of criticality is often determined in an industry-specific way. For instance, DO-178C, the software safety standard for the avionics industry, uses the concept of software safety levels and mandates different levels of structural coverage analysis for each.

IEC 61508, a general industrial software safety standard, defines safety integrity levels (SIL) and recommends different structural coverage metrics based on each level.

In all of these standards, a common philosophy can be seen: the "safer" the code must be, the greater the thoroughness of required testing. The exact definition of what software safety means depends on the concerns, experiences, and regulatory pressures for



ATP Electronics, Inc.

ATP Ruggedized IoT NAND Flash Storage Solutions



100% tested for long-term reliability

- **Frequent, small-file write applications**
 - » ATP Advanced wear leveling & block management algorithms (Endurance Protection)
 - » ATP Life Monitor /SMART Tool
- **Read intensive applications**
 - » ATP AutoRefresh (Read Disturb Protection)
- **Power Failure Protection**
 - » PowerProtector
 - » Power Cycling RDT (Linux Based)



SATA III Products
 2.5" SSD SII Pro/M V
 M.2 2260/2242
 mSATA / Slim SATA / CFast

ATP Electronics, Inc

TEL +1-408-732-5000
 FAX +1-408-732-5055
 sales@atpinc.com
 www.atpinc.com



the particular industry, but this general principle of matching higher levels of safety required with greater levels of structural coverage required remains constant across the standards.

Testing should rise from requirements

Another commonality across industries in software safety standards is the belief that tests should emerge from requirements. Software requirements should determine the desired inputs and outputs of a test. If they don't, the tests can become a parallel set of requirements and this leads to confusion and software errors. Structural coverage cannot replace requirements as the basis of testing, since coverage metrics cannot dictate how code should behave – only that it be reachable during execution (and, given the abilities of debuggers, reachable during execution can be a flexible concept).

Although complementary, testing the effectiveness of executing code and testing the completeness of requirements are two different things. Test effectiveness, as measured in structural coverage analysis, looks at what parts of the code are exercised. Test completeness, which is sometimes called "requirements coverage," looks at whether or not the code has been tested for proper behavior for all requirements. If a software program is built according to its requirements, and if it contains no code unrelated to its requirements, then

complete testing of the requirements should cause the tests to effectively exercise all of the code. If there is code not exercised by the tests, this may be code that can be eliminated, or it may be a missing requirement, or it may be a flaw in the test. In this way, structural coverage analysis can provide feedback into the test design, software implementation, and requirements specification processes.

This relationship between exercising code and testing requirements also exists on the level of the individual requirement. While from an evidence-gathering perspective the high-level totals of how many requirements and how much code has been tested is more interesting, it is more often at the individual requirement testing level, and the structural coverage analysis of that individual requirement testing, where the most defects are identified and fixed.

Structural coverage analysis is often thought of as simply a target of achieving 100 percent of a metric, but it is essential to examine individual tests and the structural coverage resulting from them. This is especially true when the code being exercised is based on the requirement being tested. By examining the structural coverage of the code, it is possible to determine the exact behavior of the code under test and compare it to the expected behavior based on the requirement being tested.

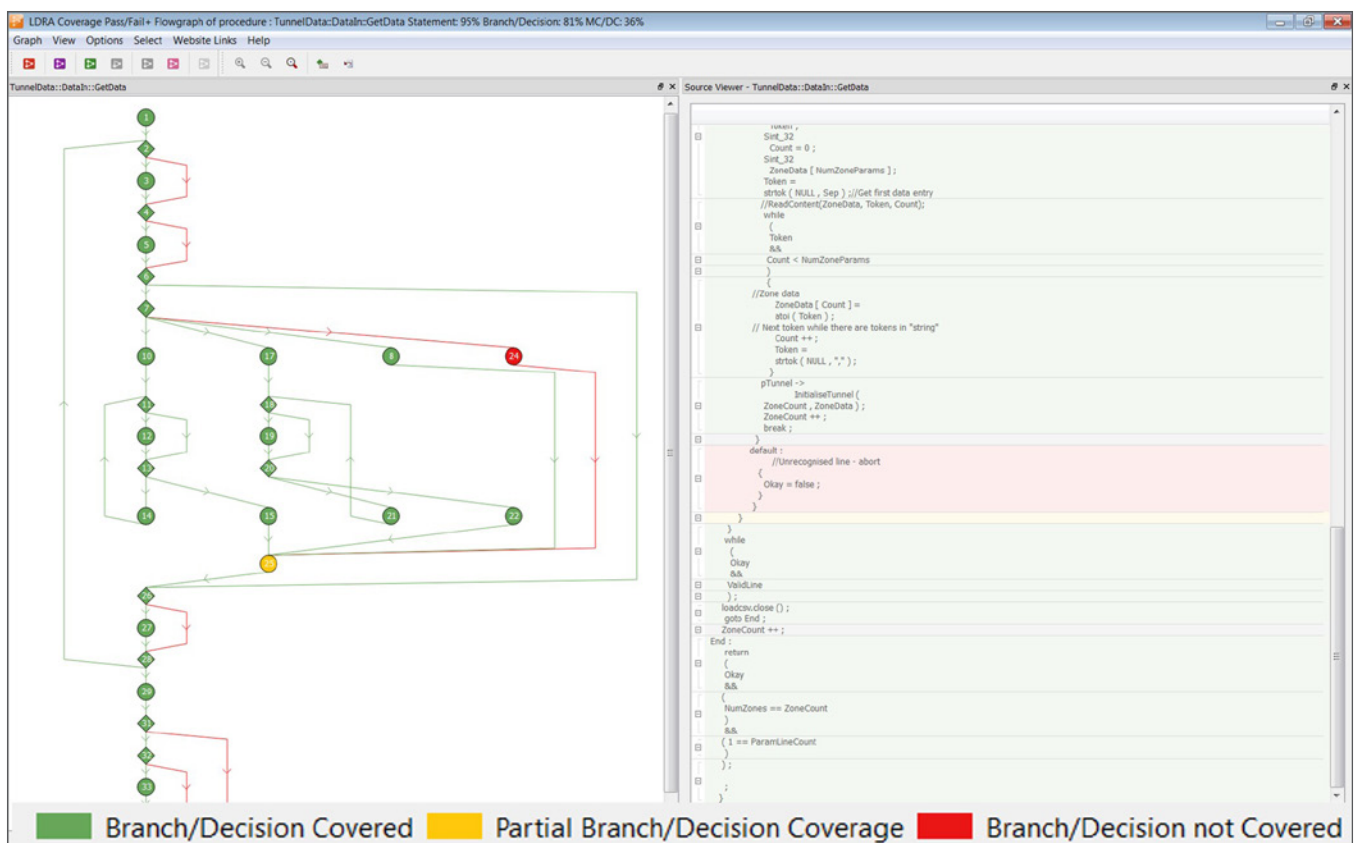


Figure 2

LDRA's TBvision gives an interactive flowgraph view of the individual procedures, so developers can focus on which procedures provide coverage and identify aspects of the code that may need further testing.

This approach reduces false negatives due to environmental factors or other parts of the code compensating for the incorrect code. In addition, if there is an incorrect behavior, structural coverage analysis often provides insight into the cause of the incorrect behavior as well.

When using structural coverage analysis to understand code behavior in this detailed manner, it is vital to be able to overlay the structural coverage analysis results on top of the analysis of the structure of the code. This overlay helps transform raw structural coverage information into a meaningful understanding of what is going on in the code (Figure 2).

Set coverage goals at unit and systems levels

Often structural coverage analysis goals might be set at both the unit and system level. Unit-level structural coverage is achieved through tests at the unit level based on requirements for that unit. On the other hand, system-level coverage goals will often start with coverage from tests on higher-level requirements. Yet if only high-level tests are used for the system-level coverage analysis, there are frequently holes in the coverage. The causes of these holes can vary. In some cases, holes in the coverage may be due to defensive programming practices required by a coding standard, but these coverage holes can be based on important functionality implemented from requirements as well.

In particular, structural coverage holes may appear when the code is based on requirements that can only be tested through conditions that are difficult or impossible to create on a high level. An example of this type of scenario is a function-level check for file-system failure. While inducing file-system failure in general may be possible, it can be highly challenging to time the file-system failure so that it occurs during that function's execution. Moreover, doing this in a repeatable way for future regression testing can be even more difficult. In situations like this, using a lower-level test that examines the code in isolation may be necessary. For this reason, structural coverage measured from higher-level tests is usually combined with structural coverage from lower-level tests when gathering metrics for achieving testing goals.

Metrics such as statement, decision, or MC/DC coverage do not guarantee that software is defect-free. As mentioned before, truly exhaustive testing can be impossible or at least infeasible. Structural coverage metrics can, however, provide a greater sense of the reliability of code and greater confidence in testing.

Since structural coverage analysis gives insight into testing activities by showing how much of the code is tested and which parts of the code have been exercised, it can be performed at the system, module, or unit level, and can be accumulated toward a testing goal. Code coverage should not be treated in isolation from requirements-based testing. Furthermore, there may be tests that need to be performed beyond structural coverage analysis. For instance, testing race conditions and integer-limit edge conditions can be valuable for detecting defects, but they may not contribute

to your structural coverage goals. Structural coverage analysis is designed to gauge the testing you have done and to guide your test planning, but it should not be taken as a goal unto itself.

Beware!

Accumulating structural coverage without understanding the tests can provide a false sense of security that can be more dangerous than inadequate testing. Structural coverage analysis is not a magic bullet, but a tool that needs to be used with intelligence and care. However, it is a tool that, when properly used, can make tests more useful and more effective and provide evidence of the testing process.

John Thomas is a Field Application Engineer for LDRA Ltd. He graduated from Rutgers University. Before working for LDRA, John worked as a developer for several companies and was previously Project Manager for Castel Enterprises, Ltd. He joined LDRA in 2011 and has worked with LDRA clients on coverage testing, target integration, and requirements traceability.

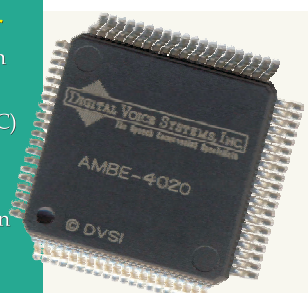
LDRA Ltd.

- www.ldra.com
- 🐦 [@ldra_technology](https://twitter.com/ldra_technology)
- in www.linkedin.com/company/ldra-limited
- ▶ www.youtube.com/user/ldraltd


Digital Voice Systems, Inc. new AMBE+2™ Vocoder chip delivers high quality voice at low cost!

High Performance Vocoder

- ✓ Voice Compression Rates from 2.0 to 9.6 kbps
- ✓ Forward Error Correction (FEC) with Viterbi Decoder
- ✓ Noise Suppression and CNI
- ✓ Tone Detection and Generation
- ✓ Half-Duplex operation




With Half-Duplex operation and DVS's latest technology the AMBE-4020™ is optimized for harsh mobile communication environments.



Ideal for land mobile radio, Voice Over IP and record/playback applications.

Compact Chip Design

- ✓ Built-in codec
- ✓ Low power consumption
- ✓ No licensing fees
- ✓ Small LQFP or BGA package
- ✓ Hardware development kit available



DIGITAL VOICE SYSTEMS, INC.
The Speech Compression Specialists

AMBE-4020™
DVS's Low Cost vocoder chip!

(978) 392-0002
www.dvsinc.com



ENSURING TRUST IN IOT WITH TRUSTED PLATFORM MODULES AND TRUSTED BROKERED IO

While connecting previously isolated devices on the Internet of Things (IoT) yields countless possibilities, it has also forced industry to reconsider how these seemingly harmless edge systems could be leveraged with malicious intent if not properly secured. In this interview with thought leaders from the Trusted Computing Group (TCG), Steve Hanna of Infineon and Stefan Thom of Microsoft discuss

why security must begin at the hardware level, and explain how the TCG's Trusted Platform Module (TPM) and Trusted Brokered IO specifications are evolving to meet the requirements of the IoT.



Steve Hanna
Senior Principal
Infineon and Trusted
Computing Group



Stefan Thom
Principal Software Development
Engineer and Platform
Security Architect
Microsoft and Trusted
Computing Group

Q Give us some background on the Trusted Computing Group (TCG).

HANNA: The TCG has been around about a dozen years, and it was originally started by a group of IT security companies – Microsoft, IBM, HP, Intel, AMD, etc. – who are our founders and who recognized at the time that something more than just software security is necessary to properly secure information technology. They had the foresight to see at the time that the level of sophistication of attacks is always growing, and we needed to provide the best security possible and do it through a standardized mechanism.

The original focus was on the PC because those were the days of Slammer and all those nasty PC malwares and worms. The focus was, “how can we build into our PCs something that is going to be

pretty much impervious to these sorts of clever malware attacks?” Yes, maybe you can crack the software and get into the operating system, but no, you’re not going to be able to get into the TPM. Then what we can do with the TPM once we have it (things like strong identity and a place to keep your cryptographic keys) and, a really special aspect of trusted computing, something that can provide confidence in the integrity of the software. Something where you can say, “how can we have confidence that this device has not been infected? How can we verify that remotely? And, how can we make sure that the secrets we really care about don’t fall into the hands of malware? That’s really a special aspect of this Trusted Computing technology – the ability to do those measurements and make sure that a particular device isn’t infected, and if it is, to detect that and be able to recover without substantial harm.

The TCG defined the standards for a variety of different hardware security components, such as the Trusted Platform Module (TPM), self-encrypting drives, trusted network communications, and the like. And, by defining standards for these it enables competition among the vendors that implement those standards, and it enables the vendors who make use of those standards to do so in a vendor-neutral manner (so you can build into Windows, as has been the case for the last several versions, support for the TPM and start building on top of it to create more advanced capabilities like BitLocker). That’s the sort of thing that TCG does – creating open standards for trusted computing, which is just another way to say secure computing, and we’ve adapted over the years as new forms of computing and new applications of computing have come along, such as the Internet of Things (IoT).

Q How would you define a hardware root of trust, such as a TPM?

HANNA: This is one of those things where people can argue about the nuances of the language, but the gist of it is this: a hardware root of trust is a fundamental capability upon which security has to be built. In order to have confidence in that hardware root of trust, you need to implement it in a way that's really trustworthy. That's why the TCG has defined certain standard roots of trust (a root of trust for measurement, a root of trust for storage, and so on and so forth), defined how they work, what the interfaces are, and then created the TPM certification program for a TPM implementing these certain roots of trust and meeting these certain security requirements through what's called the Common Criteria Security Evaluation, an independent and impartial evaluation of the security of the TPM.

THOM: A hardware root-of-trust also comes in different flavors. So the stuff

that we're doing on Windows for phones where Qualcomm is providing a firmware TPM, which is a software flavor of security where there is no discrete TPM card that has been inspected, obviously when you evaluate the security of a TPM with the discrete part you get a lot stronger guarantees that are enforced in silicon rather than just by some software modus that the TPM operates on.

Now, for a regular consumer, while the firmware TPM may be absolutely okay, if you're running a device in government or a government-regulated enterprise, the requirements might be much stronger in terms of tamper resistance and the possibility of attacking the root of trust. Because at the end of the day we all know that everything can be lifted, you just have to provide the right amount of money to actually get it out. If all you have to protect is your iTunes password, then a software or firmware TPM might be sufficient. If you're protecting the social security numbers of half the country, then you probably need something more. And the interesting

part here is that the TCG is working with so many different manufacturers that we have a lot different solutions that address the different market segments.

HANNA: That's one of the nice things about the TPM is that you can buy products at different levels of assurance, different levels of security, that all implement those same APIs and the same operating system can work across those different products. Whether it's a software, firmware, or hardware TPM, you're getting an increasing level of assurance, and at the high end would be a hardware-certified TPM. You just decide based on your risk tolerance what level of security is most appropriate.

Q What does a TPM architecture look like?

HANNA: Standard interfaces, but as to the internals of how the TPM is implemented, that is up to the vendor to decide. So long as it implements the standard interfaces in the standard



The advertisement features a central image of a hand holding a tablet displaying the Toradex Developer Resources website. The website shows sections for 'Product Selector', 'Hardware Resources', 'Software Resources', 'Frequent Downloads', and 'Linux'. To the right of the tablet, there are icons and text for 'Design Guides', 'Reference Designs', 'Pinout Designer', 'Premium Support', 'Complete BSPs', and 'Software Library'. Below the tablet, the text 'developer.toradex.com' is displayed. At the bottom of the advertisement, there is a section for 'ENGINEERING RESOURCES AT YOUR FINGER TIPS' with the Toradex logo and tagline 'Embedded. Computing.'. Below this, it says 'ARM® Computer on Modules Small Size. Big Performance.' and 'www.toradex.com'. On the right side of the bottom section, it says 'Starting from \$24' and shows icons for Windows, Linux, and Android. At the very bottom, the contact information for Toradex Inc. is provided: 'Toradex Inc. | 219 1st Ave S, Suite 410 | Seattle, WA 98104 | USA | seattle@toradex.com' and 'T: +1 (800) 871-6550'.

ENGINEERING RESOURCES AT YOUR FINGER TIPS

Toradex
Embedded. Computing.

ARM® Computer on Modules
Small Size. Big Performance.
www.toradex.com

Starting from **\$24**

Toradex Inc. | 219 1st Ave S, Suite 410 | Seattle, WA 98104 | USA | seattle@toradex.com | T: +1 (800) 871-6550

AMD leverages Cortex-A5, TrustZone as baseline for new Platform Security Processors

The Trusted Computing Group maintains a number of liaison relationships with other industry organizations to ensure interoperability, among them, the International Standards Organization (ISO), the Industrial Internet Consortium (IIC), and Global Platforms, the latter of which defines the Trusted Execution Environment standard that serves as the basis for ARM's TrustZone technology. The advance of ARM SoCs in platforms ranging from mobile devices and wearables to embedded systems and server environments has grown the TrustZone ecosystem to one of the tech industry's largest, as secure TrustZone IP blocks integrate with Cortex-A-class processors and extending throughout systems via the AMBA AXI bus.

After embarking on an "ambidextrous" strategy in 2014 that encompasses both x86- and ARM-based solutions, AMD has been quick to implement TrustZone technology across its portfolio, for example in its 6th generation APUs that offload security functions to dedicated Cortex-A5 cores capable of scaling to meet the size, power, and performance requirements of embedded systems. As Diane Stapley, Director of IHV Alliances at AMD notes, this platform security processor (PSP) implementation of hardware security has become a necessity, particularly as computation evolves to meet the requirements of mobile, cloud, and the IoT.

"A hardware root of trust in a CPU is the way to go," Stapley says. "Integrating a Cortex-A5 in APUs is a dedicated space for cryptographic acceleration that is fed by firmware and provides access to APIs for the hardware. One of the things that has been done with the core is on-chip dedicated buses and memory space, so when you take I/O off device or through a system there are methods for using hardware-based encryption and checksum operations. This can be used for secure transactions in the IoT space, which implies communications across what would be an open channel.

"When we looked at the A5 it was a tradeoff of space on the die and performance/power, and as you can imagine we wanted a scalable architecture," she continues. "We can use the core and modules around it for crypto- and firmware-based TPMs that can scale up and down, and it gives us a common base for our hardware root of trust, which is valuable for architects and our partners. For software partners, products based on the same core become 'write-once, run anywhere' because the APIs are common, and can be written to almost all other TrustZone products because those APIs are common as well."

The Trusted Execution Environment leveraged in AMD's PSP offerings is licensed from Trustonic, an industry partnership that emphasizes mobile security, though over the next year the company has plans to expand this functionality to its client, server, graphics, and embedded product lines under the AMD Platform Security Processor umbrella. For more information visit www.trustonic.com or the AMD website.

ways, passes the appropriate tests (including potentially security certification), then how you implement it is up to you, and that's where we enable innovation. Some companies might already have some secure processing capability, some chip that they can use for this purpose. Others might be starting from scratch.

THOM: The bandwidth of the implemented security goes from the Common Criteria-certified devices all the way to the Raspberry Pi 2, and the Raspberry Pi doesn't provide any security. However, we want to make sure that developers who sit on that platform have the ability to develop code against a TPM, even though it is not secure. So Microsoft is shipping a software TPM that has no security assurances whatsoever, but it provides all the mechanics and the interfaces to actually execute code and commands on the TPM itself. So with the software you can bring up on this really, really cheap board you can develop code based on this platform, and then when you productize it you can take the same code and put it on a platform that has a secure version of a TPM or put a discrete TPM on the Raspberry Pi and then work with that. The implementation at the bottom may change, but since the interfaces are the same you don't have to make a dependency in your code development or go back to the drawing board with your code development because the interfaces are all the same.

HANNA: It's really important for IoT in particular that we have this standardized interface in a variety of different implementations at different security and cost levels because that reflects the diversity of IoT. IoT is not just the home, it's also the smart factory, it's also transportation and self-driving cars and all these different application areas, each one of which has a different level of security that may be needed. Even within the home, the level of security you need in a light bulb versus a front door lock is probably different. But, if you can have the same APIs, the same interfaces, and the same software running on all of those, then you've reduced costs and put the security chip where it's needed for greater security.

Q How does Trusted Brokered IO play into the whole root of trust concept?

THOM: Over the last decade or so, individual companies have been doing a more or less decent job of securing their own IP on a device in terms of making sure you cannot download their software or tamper with it, but this is an individual approach for every manufacturer. The SoC manufacturers give them the means to lock the flash down or turn the debug interface off, but it's up to the developer to figure all of this out, to figure out how to secure their solution. That means that every device is handling this in a different way, so for an operating system that wants to control all of these smaller IoT devices, it's impossible to look over a range of 20 devices and understand "what's your identity, how can I interact with you, what version of software are you running, are you trusted or not" and so on.

So we took a step back and looked at the TPM, and the TPM has the same problem. We're now below the TCG library specification. When a discrete TPM manufacturer builds a chip, the TPM library to a large degree is just code that runs on that chip, which has its own identity. If the TPM manufacturer makes an update to that code and provides an updated firmware to the TPM, the question is, "do I have to change

the identity of this chip or not?" Is it still the same chip? Yes, it is, but it does something else. So the user of the product probably needs to get some notification that says there was a software update done to your device, and the device may be better suited to run in the future because it was a worthy software update, but it also could have been an attack. If somebody managed to flash bad firmware into a TPM that voided all security, then I most certainly want to know about it.

Since TPMs have the same problem and pretty much any single-chip IoT solution has the same issue on a chip level, how do we factor code into this on the lowest level? So we essentially condensed the capability of the TPM down to the absolute bare minimum, and that's a specification that we're writing right now called RTM, or roots of trust for measurement, that puts guidelines down for how identity is supposed to be done at a chip level and how immutable boot code is supposed to be done. We are working with software manufacturers and MCU manufacturers today to build prototypes of chips like this. The main drivers behind this undertaking are Microsoft and Google because we both have the same need of being able to interface with those chips and establish identities of those chips and what firmware is running in those chips. STMicro, on the other side, is involved to build the first prototype implementation of this in hardware.

What this builds is a platform foundation where, if you imagine, there is some bootrom code in the MCU and that bootrom code executes every time you turn it on – there is no way to power on around it. What this code does is take an identity from a register and hash the code that is stored in the chip into [the register], so we get an identity that consists of the hardware ID and the software. After this operation is done the hardware ID is made unavailable and then the code jumps to the actual application code that is flashed to the device.

The application can now use this compound identity with any caller and say, "look, this is my identity." It can give the hash of the identity of chip and code to the caller, but since it does not have access to the identity anymore, it cannot derive any fake identity or fake a different software on the same hardware. The caller will take the digest of the compound identity and send that to the manufacturer and say, "here is what the chip claims to be. Is this a trusted piece of hardware with a trusted piece of software on it?" That would allow the manufacturer to then say, "Yes, I recognize the software because I know how the measurement is calculated," and it will give a key back, and the key is a derivative of the compound identity specifically for the caller.

Now, the caller can essentially establish a trusted connection, because the chip would do exactly the same thing; it would be a shared derivation. And now we have a shared secret on both sides that can be used to communicate with that chip. If that chip is replaced or if the firmware is changed, the chip will no longer belong to that key, and therefore if I attack your hardware or attempt to flash something malicious into your chip, the key that I know is good that I use to talk to you no longer works for the other side. I can reestablish the connection, and I have to go through the attestation step again, but at the end of the day I have a secure channel into my piece of hardware. That's the foundation for all of this.

What we have identified, especially in the IoT world, is that if we use a generic operating system like Windows and Linux to build, say, a furnace, we'll have many GPIO pins. And let's imagine that there is a fuel control valve and there's an igniter. If I leave the policy of how these are supposed to interact with the dynamic operating system, then an attacker of the operating system can monkey in between the fuel control valve or igniter and the operating system and unhinge the policy. And if the policy

SUPERIOR EMBEDDED SOLUTIONS



DESIGN YOUR SOLUTION TODAY

CALL (480) 837-5200

www.embeddedARM.com



TS-7250-V2 Single Board Computer

Extensible PC/104 Embedded System with Customizable Features and Industrial Temps

- 800 MHz or 1 GHz Marvell PXA166 ARM CPU
- 512 MB DDR2 RAM
- 2 GB SLC eMMC Flash Storage
- PC/104 Connector
- 8k or 17k LUT FPGA
- Dual Ethernet



Available with TS-ENC720 enclosure



PC/104 Peripherals Multi-Function Daughter Cards

Our Line of PC/104 Peripheral Boards Offer Maximum Functionality at Minimum Cost.

- Digital I/O Boards
- Ultra Long Range Wireless
- Software Controlled Relays
- Multi-tech GSM, GPRS, HSPA modem boards
- Intelligent battery back-up
- Much more....

Starting at

\$165

Qty 100

\$199

Qty 1

Starting at

\$30

Qty 100

\$32

Qty 1



We've never discontinued a product in 30 years



Embedded systems that are built to endure



Support every step of the way with open source vision



Unique embedded solutions add value for our customers

Locking down shared resources in on-chip networks with NoC-Lock

As mentioned in Sidebar 1, ARM TrustZone technology has been widely adopted in the tech industry as foundation for securing systems based on a Trusted Execution Environment. Essentially, the way TrustZone works is by adding a mode bit to the operating context of the processor that provides information on whether a given instruction is run in secure mode or non-secure mode. The TrustZone instruction set can't be run by regular programs, and enables every transaction performed by the CPU to be tagged with an identifier that indicates whether or not it is secure. In its most basic form this creates the equivalent of a simple firewall, where, for example, transactions labeled with the TrustZone bit set are able to pass into specified secure areas of the chip, such as the on-chip ROM, while non-TrustZone-authorized transactions have only limited access.

However, in the context of integrated SoCs with shared resources such as memory, this architecture can become quite complex, both in terms of establishing mutual trust from a software perspective, as well as in ensuring that containerized hardware blocks remain mutually secure but protected from other domains. Drew Wingard, CTO of IP design firm Sonics Inc. explains how his company's NoC-Lock technology augments the security provisions of TrustZone to further isolate SoC blocks in such scenarios, helping minimize the risk of one compromised SoC block bringing down an entire chip (Figure 1).

"We take the bit that ARM defined in TrustZone, but we also allow security controllers that often exist on these chips to provide extra context information that also gets bundled with the transactions coming across the system to say, "this hardware block over here is participating in the streaming media container, so when you look at whether it should get access to this part of memory you should consider that carefully," Wingard says. "We can essentially provide extra tags, and then the firewall we build called NoC-Lock interrogates the nature of the transaction – is it a read or a write, which hardware block does it come from, which security domain does it believe that it's part of, and is it secure or not – and then compares that against the firewall of the program to determine whether or not access should be allowed.

"First of all, from the ground up we're implementing flexible, parameterizable, configurable containers so that you can have multiple mutually secure domains for minimizing the risk of compromising the entire system," he continues. "Secondly, we do all of our enforcement at the shared resource, so at the target rather than the initiator side, which has both a simpler software model and actually protects better against attacks. This implementation can also support protected reprogramming as well as exported or dynamically fused settings so that we can perfectly match the customer's security needs. That has the benefit of allowing them to minimize their risk profile, but it also turns out that sometimes this can simplify what a secure boot process looks like."

For more information on Sonics' innovations in IP design, security, and on-chip networks, visit www.sonicsinc.com.

is, "You shall only ignite if the gas was turned on no more than three seconds ago," it's a very important policy that needs to be enforced.

What's on the horizon for TCG and its members?

HANNA: As co-chair of the IoT subgroup, I want to point out that we just released a document called "Guidance for Securing IoT Using TCG Technologies." [1] That is an in-depth technical document that explains how to use TCG technologies for securing IoT, and we have a companion document, the architect's guide for securing IoT [2], which provides a more high level overview, a four-pager suitable for briefing management and understanding the issue of how to secure IoT from a high level. And then we've got a whole bunch of next-gen efforts. One is the Automotive Thin Profile that's specifically designed for lightweight automotive electronic control units (ECUs), but there are a whole bunch of next-gen standards that we're working on specifically in the area of IoT that address some of the challenges of IoT that have not yet been standardized – for example, software and firmware updates. There are some things that already can be done easily using the TPM to secure that, but there are other things that really need a boost in capability, so we're working actively on that.

THOM: Microsoft has put a lot of emphasis on the communication protocol AllJoyn, and what we're looking at now is to use AllJoyn on the chip to take, for example, the attestation data or the identity data and represent that as AllJoyn objects on the chip. Then the device manufacturer can add more capabilities and define them as AllJoyn procedures or signals that are communicated through the encrypted channel that is established with the device key. So we're coming to something that looks very much like an Arduino platform where you have an operating system and your

SonicsGN® Security Firewalls

- > SecureSoC: Sonics NoC – Lock™ Security
- > TrustZone® capable

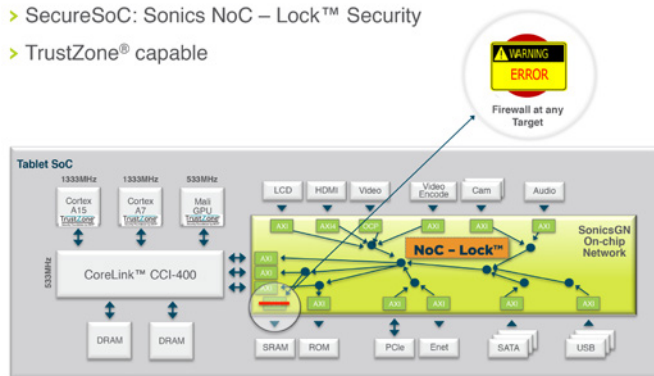


Figure 1

Sonics' NoC-Lock technology implements individual firewalls at each target destination on an SoC, such as SRAM, to minimize the risk of one compromised subsystem infecting the entire chip.

AllJoyn layer and your RTM and your root of trust on the chip, and all the app developer does is provide the AllJoyn code on top of it. Then we can connect this thing to a Windows computer or Linux computer over any serial link and the operating system can just plug this into the AllJoyn root, and any application in the vicinity or on the local machine can interact with this trusted device. If it understands how RTM works, it can essentially obtain the identity, attain the measurement, and then go establish the key it can securely work with. So RTM and AllJoyn in that manner provide a wonderful relationship together because they enable each other to bring trust to the device.

Microsoft is really making sure that where our operating system runs we have the necessary hardware to ensure secure solutions, and in return that provides value for the entire industry. There's open-source development for TPM code that then could also be picked up by alternate operating systems that want to take advantage of a hardware TPM or firmware TPM that may be present on a device. So, all in all, we're making sure that this wild beach party that is IoT today gets some adult supervision. We're putting the foundation down to actually build something trustworthy with those devices. **ECD**

References

- [1] Trusted Computing Group. "Guidance for Securing IoT Using TCG Technology, Version 1.0." 2015 TCG. <http://opsy.st/TCGloTSecurityGuidance>.
- [2] Trusted Computing Group. "Architect's Guide: IoT Security." 2015 TCG. <http://opsy.st/TCGloTArchitectGuide>.

Steve Hanna is Senior Principal at Infineon and Co-Chair of the Embedded Systems Work Group for the Trusted Computing Group.

Stefan Thom is Principal Software Development Engineer and Platform Security Architect at Microsoft and the Trusted Computing Group's member representative for IoT.

Trusted Computing Group

- www.trustedcomputinggroup.org
- 🐦 [@TrustedComputin](https://twitter.com/TrustedComputin)
- in www.linkedin.com/groups/Trusted-Computing-Group-4555624
- ▶ www.youtube.com/user/TCGAdmin

Infineon

- www.infineon.com/Internet_of_Things
- 🐦 [@Infineon4Engi](https://twitter.com/Infineon4Engi)
- 8+ plus.google.com/+Infineon4engineers
- ▶ www.youtube.com/user/InfineonTechnologies

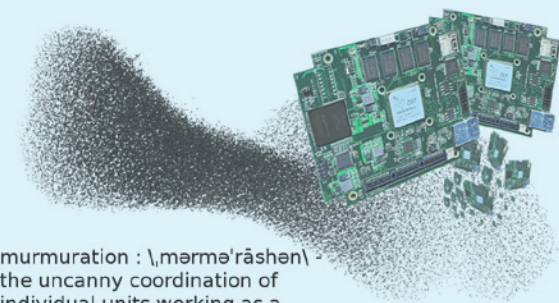
Microsoft

- www.microsoft.com/InternetOfYourThings
- 🐦 [@MicrosoftIoT](https://twitter.com/MicrosoftIoT)
- in www.linkedin.com/company/microsoft

EMBEDDED SOFTWARE EXECUTIVE SPEAKOUT

Improving inter-processor communication with software framework

Sheldon Instruments has a long history of being an embedded systems leader supplying hardware and software. We took notice of our customer needs early on and created QuVIEW, a graphical embedded DSP software programming language that was a first of its kind when introduced in 1993. Building on our years of experience with QuVIEW and other platforms, Sheldon Instruments is redefining embedded software programming development through our new software framework, Murmuration.



murmuration : \,mɜrmə'ræshən\ - the uncanny coordination of individual units working as a whole.

Murmuration is a real-time embedded software framework that enables quick, efficient and seamless communication between multiple embedded technologies. With today's embedded hardware one is likely to find solutions using multiple technologies to achieve the desired results for any given application. As is the case with FPGAs and DSPs, both will have one or more processing cores that will need to communicate through an array of complex communication interfaces. This makes Inter-Processor Communication (IPC) and data exchange a significant design challenge in modern embedded computing.

The Murmuration register map is the only a priori knowledge required between devices for communication and synchronization. Through the Murmuration registers, devices can synchronize clocks to implement deterministic processes across devices without polling or interrupts, establish inter-processor communication, as well as access device applications. The Murmuration software framework is an intuitive yet powerful way to develop on hybrid and multi-core real-time embedded systems.

Sheldon Instruments is highly committed to significant software support and building our foundations and software libraries that allow customers to more quickly integrate our hardware products into their overall platform and application.

Sheldon Instruments
sheldoninstruments.com



Watchdog strategies for RTOS enabled embedded systems

By Andrew Longhurst, Business Development Manager, WITTENSTEIN high integrity systems

A Watchdog timer is an electronic timer that is used to detect and recover from errors within embedded systems. The basic principle of the Watchdog timer is simple but effective. Within a specific time-period, the system has to notify the Watchdog that it is still operational. If the Watchdog does not receive this notification then it assumes there has been a failure and places the system into a known state. Typically, the Watchdog will reset the processor. However, for systems that are more complex the Watchdog may have to trigger a series of operations to place the system into a known safe state. Many processors support on-chip Watchdog functionality, but, for extra security, some designers prefer to use a separate discrete Watchdog component.

The challenge faced by embedded software developers is deciding when to notify the Watchdog that the system is still functional. This is more complicated when using a pre-emptive RTOS, as the software is broken down into individual Tasks operating independently. Now the designer needs to consider carefully what constitutes a working system.

Basic Watchdog protection

In a basic system, the designer may choose to have a periodic Task that simply notifies the Watchdog at the required frequency. In this scenario, the system remains available providing the Task that refreshes the Watchdog operates at the correct frequency. A complete systems crash or a failure within the Task notifying the Watchdog would cause the Watchdog to expire, placing the system into a safe state. However, if the system fails in a way whereby the Task that notifies the Watchdog remains operational but other mission critical Tasks fail to operate, the Watchdog would not place the system into a safe state.

Improving robustness

An improvement to this simple system would be to notify the Watchdog that the system is OK only if all Tasks have been active during the last Watchdog refresh time-period. In this case Tasks would register with a Monitor Task so that each time a Task runs, it informs the Monitor Task. When triggered, the Monitor Task will check that all registered Tasks have operated during the last time-period; if they have, the Monitor Task will notify the Watchdog that the system remains operational.

To manage Tasks operating at a higher or lower frequency than the Watchdog

refresh rate, the designer would need to include a time profile for all Tasks. For each Watchdog refresh period, the Monitor Task would confirm that only the expected scheduled Tasks have been active. In some systems however, knowing the Task is still operational does not provide sufficient assurance that the system is still operating correctly. In these systems the operation of critical code sections must be monitored as well as the Tasks they are found within.

To complete the temporal monitoring of events, an additional enhancement would be to monitor the response time of Interrupt Service Routines (ISR). Here the Monitor Task would measure the actual time it takes to process the response, from the time the ISR is triggered to the time when the overall operation has completed.

By monitoring the timing profile of individual Tasks, critical code sections and ISR response times, the designer has a high level of assurance that the Watchdog notification mechanism is working as expected. However, the complexity of the Monitor Task has increased significantly.

Sophisticated Task monitoring

The Checkpoints Safety Component from WITTENSTEIN high integrity systems is a software component that provides this sophisticated Task monitoring capability, ensuring the scheduling of tasks is occurring as intended.

The Checkpoints mechanism allows the user to specify timing tolerances for critical sections of code. This can be used to ensure that:

- Periodic Tasks run within tolerances.
- Sections of processing within Tasks complete.
- ISR execution to Handler Task processing completes with allowable tolerances.
- Complex functionality involving multiple tasks completes within allowable tolerances.

Individual Checkpoints can specify their own callback function or the system error hook can be activated.

- Single shot and Periodic checkpoints can be created.
- Periodic checkpoints can operate in fixed or relative timing modes.

Checkpoints is available with a full Design Assurance Pack supporting certification to IEC 61508 SIL 3, and is delivered fully integrated with either SafeRTOS® or SafeRTOS CORE.





ITTIA and Micrium

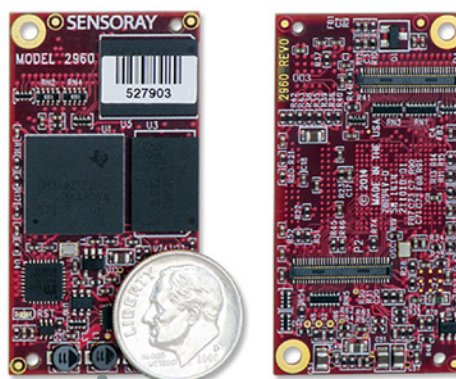
www.ittia.com | www.micrium.com
embedded-computing.com/p372056

Embedded database teams with RTOS for IoT applications

ITTIA announced ITTIA DB SQL integration with the Micrium μ C/OS-II and μ C/OS-III real-time operating systems. The database solution persistent data storage provides reliability, scalability, and shared access. In addition, the μ C/OS file system (μ C/FS) provides a portable file system that can be used with or without an operating system. This combination provides scalable capabilities for IoT applications requiring persistent and efficient data storage, retrieval, and indexing.

Compact lightweight audio/video processing engine

Sensoray's 2960 "Dragon" audio/video processing engine serves as a highly adaptable, foundational building block that Sensoray can quickly customize to customers' unique requirements. The board comes in at 4.8 cm x 2.5 cm (1.9" x 1") and a weight of 6.2 g (.22 oz). The "Dragon" is capable of performing H.264 compression on captured 1920 x 1200 video at 30 frames per second, or JPEG snapshots at up to 4096 x 3104. The board includes a controller and stream router, SD card interface, six GPIOs, USB, Ethernet, serial, and I2C interface. For device mode operation, the board can be completely powered from the USB. Controller functions include IP stream server, file server, HTTP server, and camera control server.



Sensoray | www.sensoray.com
embedded-computing.com/p373039

New embedded computing platforms featuring 6th generation Intel Core processors

Advantech has announced a range of computing platforms using 6th generation Intel Core processors. The boards offer improved CPI and graphics performance, enhanced power and feature scalability, and feature support for IoT applications from edge devices through IoT gateways and cloud. The platforms include Computer On Modules (COM), MI/O Extension Single Board Computer, industrial motherboard, and digital signage players. The portfolio has been designed with an eye toward end-to-end IoT – from cost-optimized IoT devices and smart sensor boards to higher performance gateway products through cloud servers for high performance and reliability.



Advantech | www.advantech.eu/EmbCore
embedded-computing.com/p373040



EPIC Single Board Computers
Rugged, Stackable Form Factor
Fanless -40° to +85°C Operation

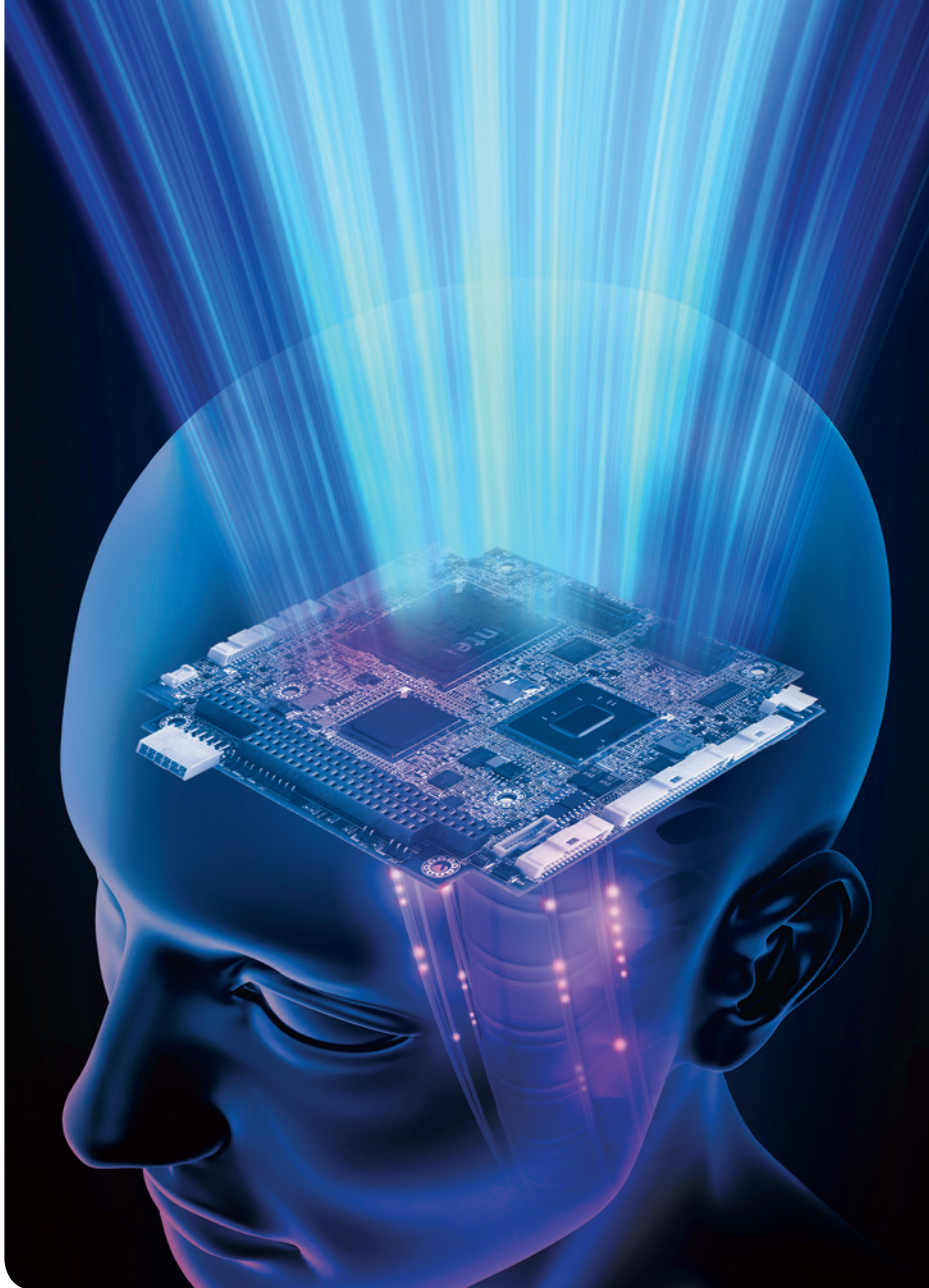


Small Form Factor Computers
Intel® Atom™ E3800 and i.MX6 CPUs
Fanless -40° to +85°C Operation



PC/104 Single Board Computers
Rugged, Stackable Form Factor
I/O Modules and Power Supplies

Single Board Computers
COM Express Solutions
Power Supplies
I/O Modules
Panel PCs



Thinking beyond the board

Sometimes our off the shelf products are not the perfect fit. Our application engineers and in house design talent are ready to develop customized solutions for your system requirements. Our stock products are accessible to use as building blocks for your next project. Calling WinSystems connects you directly with an Application Engineer who is ready to discuss customization options for firmware, operating systems, configurations and complete designs.

Team your engineers with ours to move your product from concept to reality faster.

New Functionality, Exclusive Content, Fresh Design
The NEW www.winsystems.com

715 Stadium Drive | Arlington, Texas 76011
Phone: 817-274-7553 | Fax: 817-548-1358
info@winsystems.com



 **WinSystems®**
The Embedded Systems Authority