CMP september 2000 www.embedded.com volume 13, number 10

Embedded Systems PROGRAMMING

Open Source:
Is It Worth the Price?

Analysis with **Sequence Enumeration**

Configuration Management Tips

Souped-up **Z80s**

How to do **Hartley**

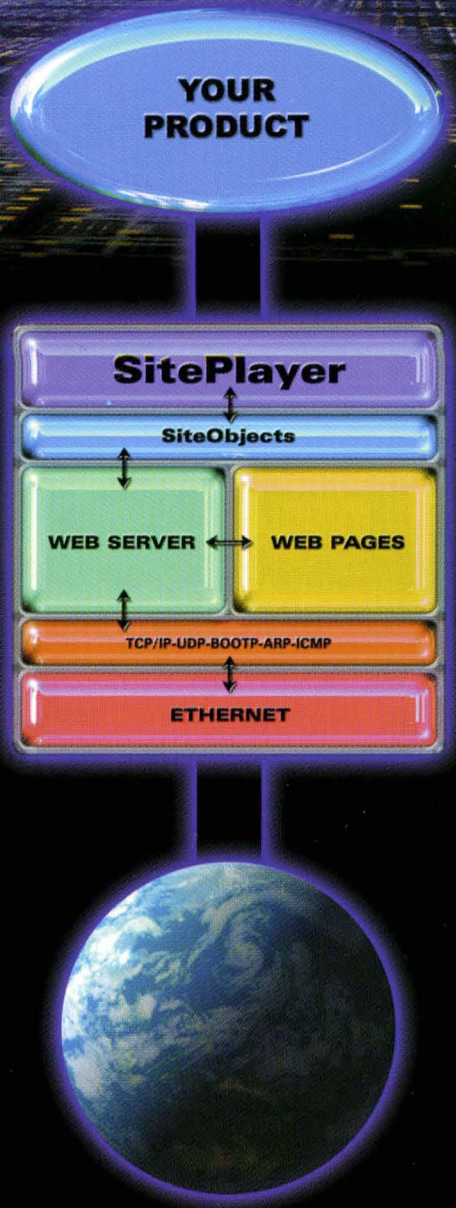
When no **RTOS** is Needed



Internet Appliance Design:
IPv6 in Your Future

SitePlayer™

EMBEDDED INTERNET SERVER



You've got a device. You want your device on the web. You don't have a lot of time. You don't want a lot of headaches. You don't want to spend a lot of money. You just want it to work.

You want it to be directly usable, with simple installation for your customers. No gateways, extra hardware or special wiring required. You just want Ethernet with installation as tough as plugging into a hub and browsing away.

Welcome to the world of SitePlayer. SitePlayer is a web server coprocessor module, complete with ARP, ICMP, BOOTP, UDP, TCP/IP and HTTP protocols, web page storage, 10baseT Ethernet interface, all in one square inch!

You create the look and feel of your device using standard web page tools. SitePlayer even gives you a library of graphical switches, knobs and LEDs to give your product a professional look.

During development, simply download your new pages to SitePlayer over Ethernet. Updates in the field are easy.

Need space for fancy graphics, MP3s, video clips, animations, embedded user's manuals, or multiple languages? SitePlayer supports optional flash memory cards for multi-megabyte web sites.

Communication from your device to SitePlayer is easy. A serial port is all you need. That's it! Minimal code to write and you are up and running with a world wide graphical user interface. Don't believe it? Just buy the SitePlayer developer kit for only $199.95 and see for yourself. SitePlayer chip sets are available for high volume applications.

Whether you need to make 10 or 10 Million devices, SitePlayer is the simplest, fastest, most cost effective way for you and your customers to get your device on the web.

WWW.SITEPLAYER.COM

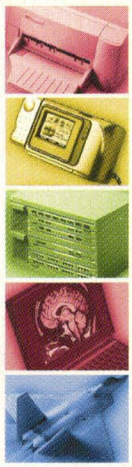
NETMEDIA INC. 10940 N STALLARD PLACE TUCSON ARIZONA 85737 t520.544.4567 f520.544.0800 SALES@NETMEDIA.COM

SitePlayer, the SitePlayer logo, and NetMedia are trademarks of NetMedia, Inc.

WindRiver®

Diab™ C/C++/FastJ™, RTA, visionICE, visionPROBE, SingleStep Solutions™, SNiFF+,™ and Development Boards. Separately, they're Best in Class. Market Leaders. Powerful. Open. Scalable. Now unified under Wind River, these tools form complete solutions that lift your development project and make it soar, no matter what operating system you choose, all the while backed by Wind River's world class sales and support organization. To help your next embedded project take off, call 1-800-545-WIND or browse http://embeddedtools.windriver.com and register for a free tools evaluation.

www.windriver.com



How smart things think™

Lift and Soar.
Embedded Development Tools from the New Wind River.

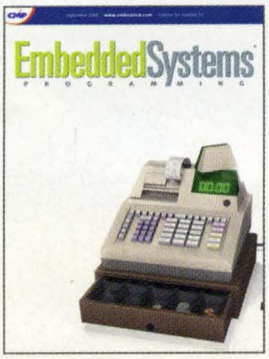
Wind River tools are available on these architectures:
ARM
Hitachi
IBM PowerPC
MIPS
Mitsubishi M32R
Motorola PowerPC
Motorola ColdFire/68K
Motorola MCORE/DSP

© 2000 Wind River Systems, Wind River logo, How Smart Things Think, Diab, FastJ, SNiFF+ and SingleStep Solutions are trademarks and registered trademarks of Wind River Systems, Inc. All other marks belong to their respective owners.

EmbeddedSystems
PROGRAMMING

contents

SEPTEMBER 2000



COVER

Will the open source movement prove to be profitable, or will you end up getting short-changed?

Cover illustration by Rupert Adley.

50

New Twists on the Z80

Despite being a quarter-century old, the 8-bit Z80 and its derivatives continue to be popular with embedded system designers. With two new derivatives emerging within the last year, it's a good time to take a fresh look at the architecture. This article considers the architectural choices from the perspective of an RTOS implementer or user.

BY JEAN LABROSSE

116

Sequence Enumeration

Sequence enumeration is a technique for analyzing systems. It can be used to turn a complex set of requirements into an easily implemented state machine.

BY ROB OSHANA

128

Doing Hartley Smartly

Think the fast Fourier transform is fast enough? Think again. Meet the fast Hartley transform.

BY ROBERT SCOTT

146

Get by Without an RTOS

Too many simple systems use a commercial RTOS. All that's sometimes needed is a way to implement event-driven and periodic functions. Here's an easy way to do just that.

BY MICHAEL MELKONIAN

166

Configuration Management Tips

Whether you're just getting started with version control techniques or an old pro, this article has some good suggestions for you to improve your overall plan.

BY ALLAN FOLZ

internet appliance design

69 **CONNECTING...**
Internet Working
The Internet Protocol (IP) is the glue that holds an internet together. In this month's column, Michael presents the IP layer implementation that is the core of his μC/Net stack.
BY MICHAEL BARR

Cover Story

78 Open Source Point/Counterpoint: Are Open Source and Innovation Compatible?
According to the popular wisdom of the moment, open source software is the way of the future. But can the purveyors of free software really innovate?
BY NIALL MURPHY

Counterpoint

Open Source is Already Delivering
One need not speculate about the benefits of open source software. Just look to its past accomplishments. The future holds more of the same. —p. 88
BY BILL GATLIFF

95 Ready or Not: IPv6 is Coming
The World Wide Web is no longer just workstations, servers, and PCs. It's now also PDAs, cell phones, and automobiles. The old protocols of the Internet cannot handle these millions of devices. IPv6 is coming and you're going to have to be ready.
BY STEPHEN HARPSTER

109 Embedded Internet Tools

departments

5 **#INCLUDE**
Embedded Ubiquity
BY MICHAEL BARR

7 **PARITY BIT**

11 **NEWS VECTORS**

189 **NEW PRODUCTS**

199 **BOOK REVIEW**
Doing Hard Time
BY JEFFREY TAYLOR

204 **RECRUITMENT**

211 **MARKETPLACE**

216 **ADVERTISER INDEX**

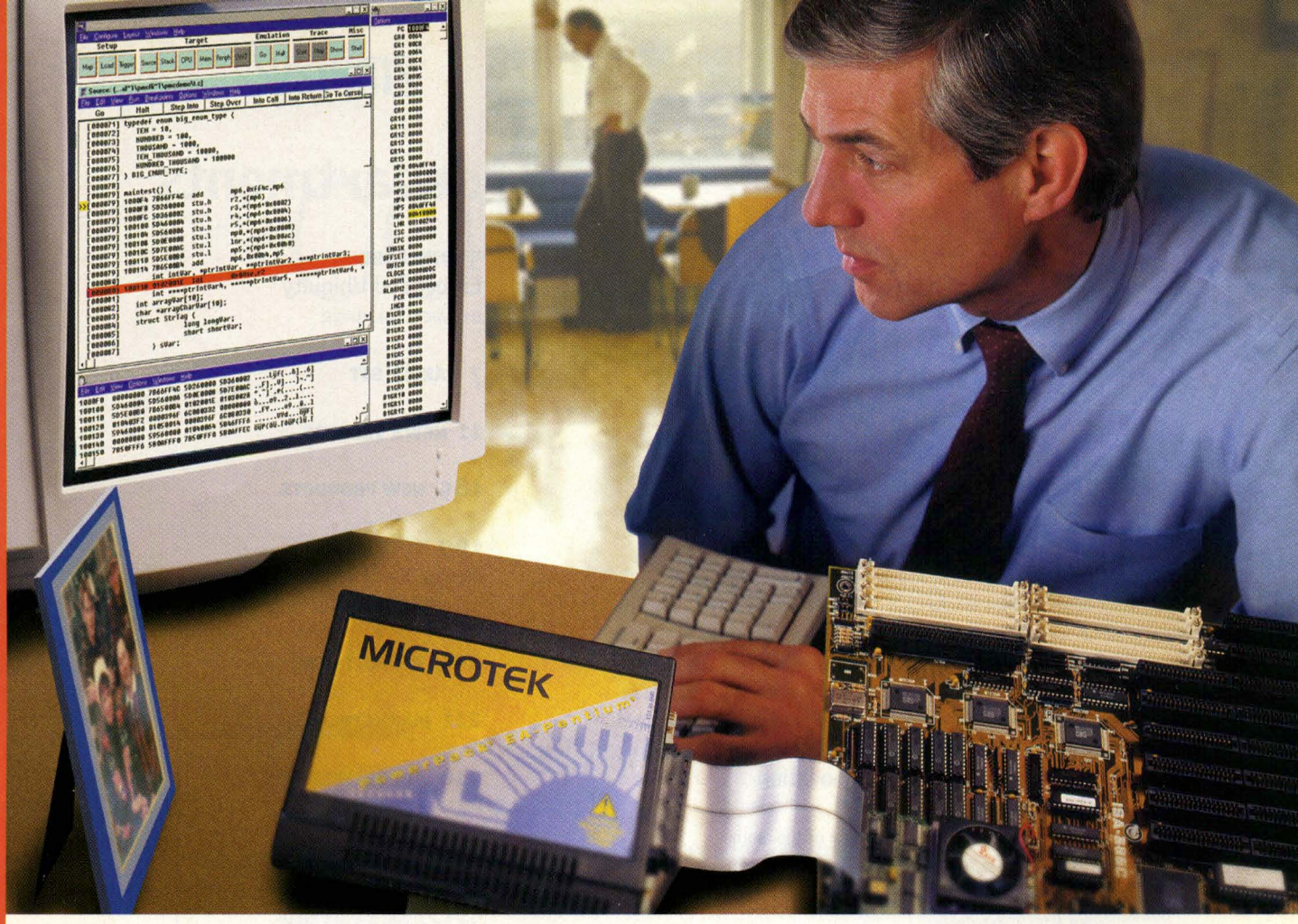
columns

17 **PROGRAMMER'S TOOLBOX**
The Topic That Would Not Die
Locked in battle with his twin nemeses—Dell and minimization—Crenshaw pines for the good old days and gains some mathematical ground.
BY JACK W. CRENSHAW

113 **PROGRAMMING POINTERS**
Numeric Literals
Every literal has a type. It may not be obvious, and it may vary across platforms, but the standard does specify it precisely.
BY DAN SAKS

179 **SPECTRA**
Field-Oriented Control
Three-phase motors require proper care and feeding. Field-oriented control makes this easier.
BY DON MORGAN

219 **BREAK POINTS**
Repair or Replace
An engineer's adventures in paradise: Jack shares what he's learned from watching engineers make do in the Bahamas.
BY JACK G. GANSSLE



"Now, I have the tools to keep my project on time."

I remember when I budgeted for my project. I refused to spend money on debug tools. I didn't think I would need them. The money I saved would make me a hero.

Now, my project is one month, two months ... six months overdue. Nobody thinks I'm a hero. What would I give to get some of that time back?

Microtek debug tools can make time. They speed up development and testing, saving you precious time at the end of the project, where you need it most!

Microtek emulators offer the debug features you need to track down and correct software, hardware, and system integration issues. They can find the errors software debuggers cannot see.

First, emulators can debug before the operating system is functioning. If an unexpected issue is affecting boot up, an emulator can find it. They are also operational after a hard crash. This is significant, because software debuggers lose debug information. Microtek emulators keep track of the last 128 KB of bus cycles, allowing you to sift through and find the problem.

Finally, if there is a subtle issue during the integration of hardware, Microtek EA emulators are capable of providing a clock-cycle-by-clock-cycle trace that allows you to view each signal and determine whether it was in the correct state. Microtek's tools are clearly superior to software debug solutions.

URGENT!

If you are developing a Pentium class target using a reference design or off-the-shelf board, please take a moment to speak with our staff. They will insure your design is on track, and tool friendly if debug is needed in the future.

There has never been a project that couldn't use more time when it's critical ... like in final testing when everything has come together ... or in final debug, when everything is going down in flames.

Wouldn't it be great to deliver your next project on time? Next time, be sure to put a Microtek emulator in your project plan from the start.

It will help take the knot out of your stomach. And it will improve your company's bottom line!

Additional interfaces: CAD/UL® and Windriver Tornado II®

MICROTEK
IN-CIRCUIT EMULATORS

1 (800) 886-7333

Phone (503) 533-4463
Fax (503) 533-0956
Email – info@microtekintl.com

Financing Available

Pentium® III • Celeron® • Pentium® II • Pentium® • Intel486™ • Intel386™EX/CX/SX • 683xx/HC16 • 8051/52/25

Pentium, Intel486 and Intel386 are regjstered trademarks of Intel Corp. © 1998 Microtek International

www.microtekintl.com

Embedded Systems
PROGRAMMING



Michael Barr

EDITORIAL DIRECTOR
Lindsey Vereen, lvereen@cmp.com

EDITOR IN CHIEF
Michael Barr, mbarr@cmp.com

MANAGING EDITOR
Felisa Yang, fyang@cmp.com

EDITOR AT LARGE
Alex Wolfe, awolfe@cmp.com

EDITORIAL ASSISTANT
Timothy Sullivan, tsullivan@cmp.com

CONSULTING TECHNICAL EDITORS
Jack G. Ganssle
Jerome L. Krasner, PhD

CONTRIBUTING EDITORS
Jack W. Crenshaw
Larry Mittag
Don Morgan
Dan Saks

PUBLISHER
Eric Berg, (415) 278-5220

NATIONAL SALES MANAGER
Stephen Corrick, (323) 512-4100

EASTERN REGIONAL SALES MANAGER
Damon Graff, (781) 839-1285

EASTERN SALES REPRESENTATIVE
Jared Grimm, (781) 839-1286

CALIFORNIA SALES MANAGER
Andres Diaz, (415) 278-5274

CALIFORNIA SALES ASSOCIATE
Molly Bruns, (415) 278-5298

WESTERN ACCOUNT EXECUTIVE
Sam Louis, (415) 278-5223

PRODUCTION COORDINATOR
Matthew Powell

CIRCULATION MANAGER
Jennifer Schuler

CIRCULATION ASSISTANT
Michael Falabella, mfalabel@cmp.com

CIRCULATION DIRECTOR
Susan Harper

SUBSCRIPTION CUSTOMER SERVICE
Toll free: (877) 676-9745
(847) 559-7307
Fax: (847) 291-4816
Omeda Communications
610 Academy Drive
Northbrook, IL 60062
esp@omeda.com

Back issues may be purchased on a prepaid basis through: Miller Freeman, 1601 West 23rd St., Suite 200, Lawrence, KS 66046; (800) 444-4881; (785) 841-1631

REPRINTS
Stella Valdez, (916) 983-6971

ONLINE PRODUCTION COORDINATOR
Billy Biondi, wbiondi@cmp.com

PRESIDENT/CEO, CMP MEDIA INC.
Gary Marshall

EXECUTIVE VICE PRESIDENTS
Regina Starr Ridley John Russell
Steve Weitzner Tony Uphoff

SENIOR VICE PRESIDENT/ GLOBAL SALES & MARKETING
Bill Howard

SENIOR VICE PRESIDENT/ BUSINESS DEVELOPMENT
Pam Watkins

VICE PRESIDENT/ELECTRONICS
Donna J. Esposito

EMBEDDED/DSP GROUP DIRECTOR
Mike Flynn, (415) 278-5251

PRESIDENT/ELECTRONICS
Steve Weitzner

Consumer Audit
BPA

CMP

Visit our Web site at
www.embedded.com

Embedded Ubiquity

By virtually all accounts, the future of computing lies in the overlapping space between the Internet and embedded. As embedded system developers, we find ourselves right in the middle of this crowded intersection. Without us, there could neither be people trading stocks from their cellphones today nor speculation about a tomorrow-world with hundreds or thousands of smart devices per human inhabitant. Yet somehow, embedded systems have never caught the full attention and imagination of those outside our community.

Some long-overdue attention is finally coming our way. A recent issue of *Communications of the ACM* focused specifically on the topic of networked embedded systems ("Embedding the Internet," May 2000). One author, David Tennenhouse, stated that "the computer science research community now enjoys a rare and exciting opportunity to redefine its [40 year old] agenda and establish new goals that will propel society beyond interactive computing...to increased human productivity and quality of life."

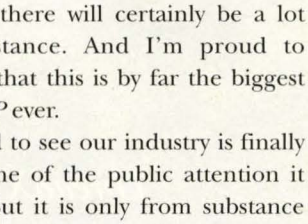
It seems computer science researchers are beginning to realize that further research into one-on-one human-computer interaction will produce diminishing returns. The most complicated problems we'll face tomorrow are related to the management of arrays of networked embedded devices. To be useful to human communities, such systems must eventually manage their own networks and data and make a variety of decisions automatically.

Consumers and Wall Street are also beginning to notice embedded systems a bit more these days. Again, the attention is focused most heavily on networked devices. Within the past year, several mainstream publications have run cover stories showcasing future uses of embedded technology in the home. However, the excitement in the consumer space centers primarily on new capabilities for home appliances we already know and love—microwaves that read UPG codes from packaging and use that information to download cooking instructions, for example.

Wall Street's limited attention to our space has been largely misguided. To date, the largest "embedded plays" have been the Linux-related IPOs. But those aren't the folks who really stand to gain from the current explosion of uses for embedded CPUs. No organization would build any commercial product if they thought the cost of developing it would rival or exceed the profits on the sales of the thing. So it remains the case that the folks who will really profit in the long term will be the ones making the proverbial "nickel for every unit sold." The biggest money is still very much in the processors and peripherals (whether silicon or IP) and in those software components that produce royalties. Services will always play second fiddle.

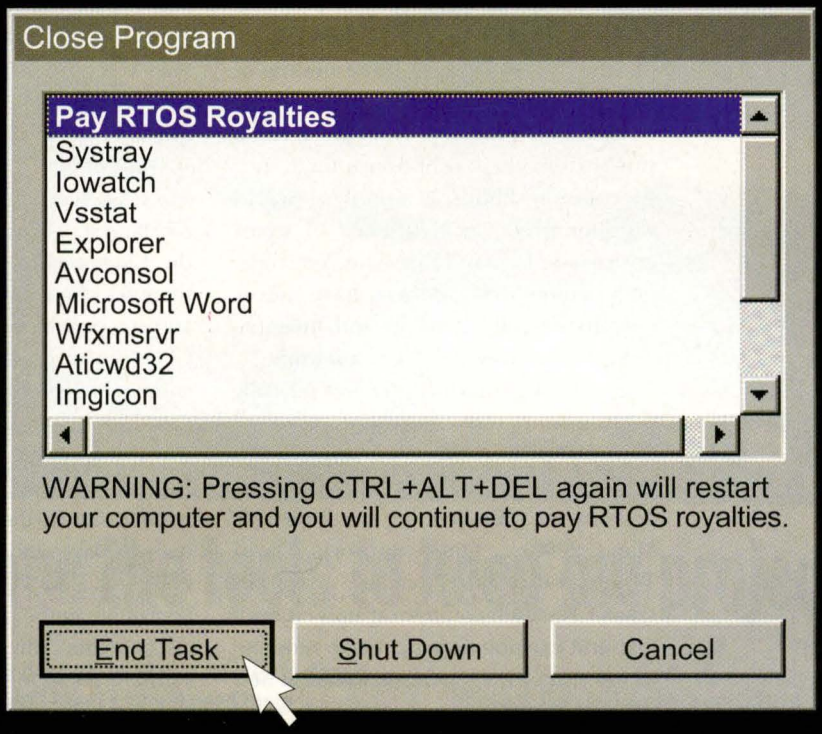
Myself, I'm looking forward to this month's Embedded Systems Conference in San Jose. There may not be as much glitz there as on Wall Street nor as much gadgetry as at the Consumer Electronics Show, but there will certainly be a lot more substance. And I'm proud to announce that this is by far the biggest issue of *ESP* ever.

I'm glad to see our industry is finally getting some of the public attention it deserves. But it is only from substance and hard work that ubiquitous computing might eventually be born.



mbarr@cmp.com

Wouldn't It Be Nice If It Were This Easy To Eliminate RTOS Royalties?



It Is!

ThreadX® and INTEGRITY™ Royalty-free RTOS Solutions
EEMBC - Best Optimizing Compilers
MULTI® 2000 World-Class IDE

Green Hills
• SOFTWARE, INC. •

805.965.6044 • sales@ghs.com • www.ghs.com



Organized Dissent

In their article, "Portable Vector Math Libraries" (June 2000, p. 104), William Wright and James Metzger describe an approach to programming for portability across multiple platforms and vector libraries. They were kind enough to mention the Vector, Signal, and Image Processing Library (VSIPL) effort. However, their mention of this effort included some factual inaccuracies which we would like to take this opportunity to correct.

The article concludes its section on VSIPL by stating that "no implementations of the library are widely available." Actually, implementations of the library are available on embedded multicomputer hardware from Mercury, SKY, and CSP. MPI Software Technology has an implementation available for the PowerPC G3 and G4 processors that is also distributed with hardware from DY 4 Systems. Links to the Web sites for these vendors can be obtained from the "users and products" section of the VSIPL Web site (*www.vsipl.org*). In addition, an open source version of the library is available for download from the site. This library, written by Randy Judd of the U.S. Navy Space and Naval Warfare Systems command, should compile on most platforms that have an ANSI C compiler. The open VSIPL standard and multiple commercial implementations free the user from having to develop his own API, and then to update it for each new target architecture.

Using VSIPL has several advantages over the approach advocated by the authors. VSIPL has a well-defined, portable interface, designed with substantial input from major embedded and workstation computing vendors and users, and supported on all the platforms listed above. Regression testing of the library is supported directly by the freely available VSIPL test suite. As the authors correctly point out, individuals defining their own API are required to write and support these tests themselves. Memory alignment issues for vector libraries are handled by the VSIPL implementation without intervention from the user. In the area of complex data types, the approach advocated by the authors will work as long as the library expects the real and imaginary portions of the data to be stored in an alternating (interleaved) format. However, VSIPL also supports a split format for complex data, which is advantageous on vector processors such as the Motorola G4 PowerPC.

The authors state that a major advantage of designing your own vector library API is that one can "choose the sacrifices that are appropriate in terms of execution speed and memory usage to achieve portability." The VSIPL standard supports such trade-offs by partitioning the functionality of the library into well-defined profiles. The Core-Lite profile is focused on vector signal processing and includes FFT and FIR filter operations. The full Core profile includes matrix algebra operations for more complex signal processing requirements. Users can write their programs using the functionality profile that best suits their needs. In addition, the participation of multiple vendors in defining the standard has resulted in a design that does much to ensure implementors can offer high performance VSIPL libraries on a variety of architectures.

James Lebak
jlebak@ll.mit.edu

Mark Richards
mark.richards@gtri.gatech.edu

Randy Janka
randall.janka@gtri.gatech.edu

Co-chairs, VSIPL Forum

Gainful PID

I read Don Morgan's column "The PID Filter" (July 2000, p. 125) with great interest and enthusiasm, but found it short on ways to tune or adjust the gain parameters for a particular system. I agree that the implementations found in software are many and varied (for example, the integral "gain" parameter is specified in a reciprocal form in some implementations that I've seen). However, I have a concern about Mr. Morgan's presentation.

The formulation presented in the first column on p. 127 is missing a definite integral symbol [with limits, say 0 to t] between the "K_i" and the "$e[t]$"—a typo perhaps since the pseudo-code presentation in the second column of the same page clearly shows that error is summed. On the other hand, in the third paragraph of the first column, he makes the statement: "An interesting aspect of this sort of system is immediately evident in this relationship: when the error goes to zero, the output goes to zero." Sorry, but those last five words are simply *not* true. Indeed, the fact that it is not true is why high integral gains can lead to severe oscillations capable of harm to both the equipment and people.

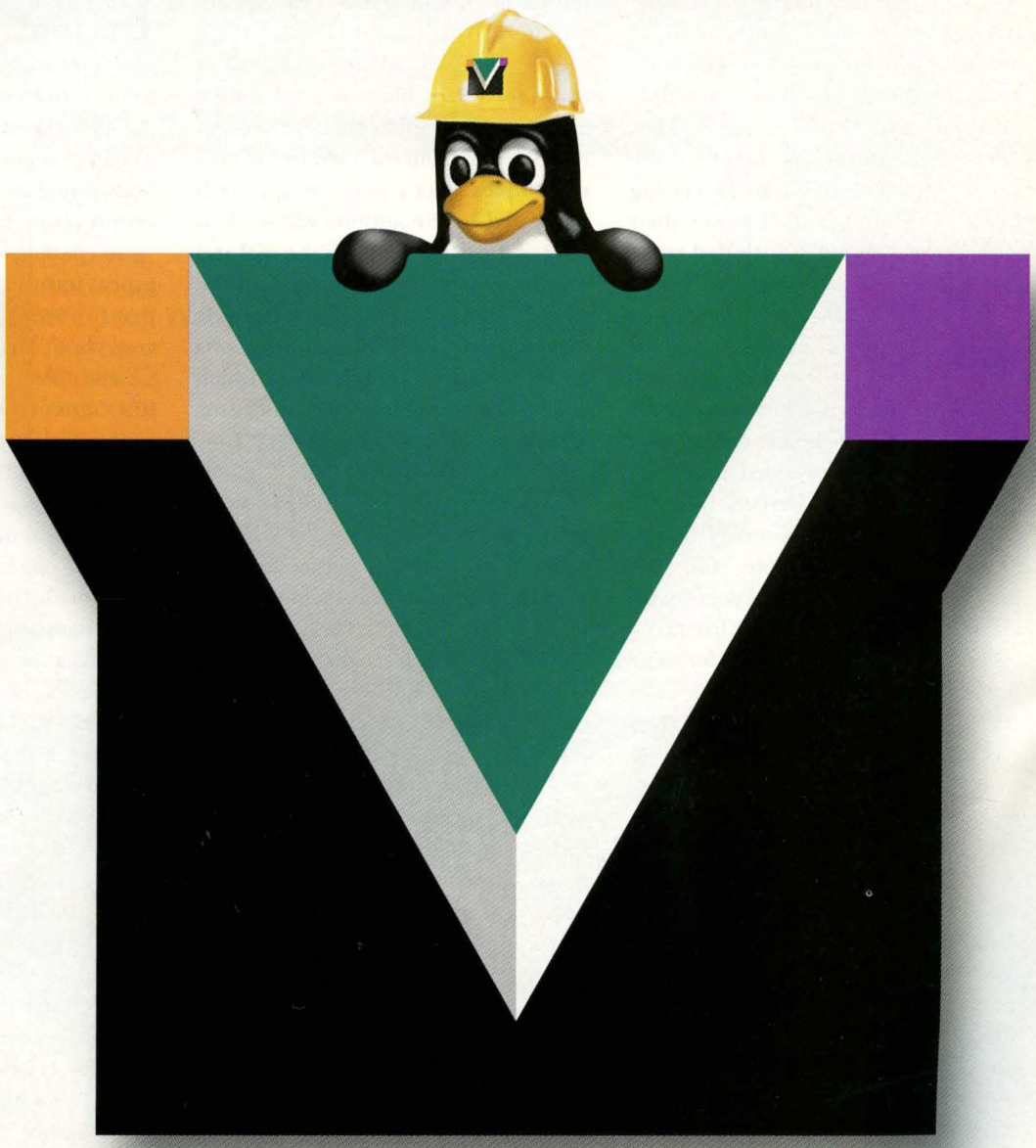
I'd like to encourage Mr. Morgan to revisit the PID controller in one of his future articles. Use of this algorithm is very common. Understanding how to set it up and successfully tune it is not.

William L. Schultz
schultz@sea-us.net

WHAT'S ON YOUR MIND?

Embedded Systems Programming gladly welcomes feedback from readers. Please send any question or comments to Editor in Chief Michael Barr at mbarr@cmp.com. We reserve the right to edit all letters for clarity and length.

MontaVista...



Embedded Linux®.

- Embedded Linux that is 100% pure, royalty-free, open source.
- Embedded Linux with the broadest hardware support.
- Embedded Linux for design, development and deployment.
- Embedded Linux leadership measured by what counts...customers.

Visit us at the Embedded Systems Conference, September 24 – 28, 2000
See Hard Hat Linux running on PowerPC, x86, StrongARM and MIPS architectures.



MONTAVISTA™
SOFTWARE

490 Potrero Avenue
Sunnyvale, CA 94085
tel: 408-328-9200
fax: 408-328-3875
www.mvista.com

The Embedded Linux® Experts.

What's just as important as getting to market first?



Looking good doing it.

The success of your product relies on two things: high image quality and quick time-to-market. Cadence® SPW and VCC multimedia solutions ensure both. Our innovative technology allows you to design systems with the best video quality, whether for wireless networks, wired networks, or storage systems. You can easily optimize digital video, HDTV, digital cameras, and other imaging systems using our customer proven libraries or your own. Success isn't all about good looks, it's only half the equation. For details, email systems@cadence.com or call 800.746.6223. Please visit us at **www.cadence.com/systems**

Cadence System-Level Design

cadence

how big can you dream?™

© 2000 Cadence Design Systems, Inc. All rights reserved. In the U.S. and numerous other countries, Cadence and the Cadence logo are registered trademarks and "how big can you dream?" is a trademark of Cadence Design Systems, Inc. All others are properties of their respective holders.

Altera: High Hopes for Programmable Logic Cores

In an attempt to broaden the market for advanced processor technologies, Altera is introducing a chip architecture that combines programmable logic with processor cores. The company believes that the flexibility of their product will allow it to tap into the ASIC, ASSP, and stand-alone embedded processor markets all at once. Altera predicts that these markets could add up to a $50 billion business opportunity by 2003. Their predictions for the traditional PLD industry fell to a less promising $5 billion in market potential.

"We believe the trend away from a fixed system-on-a-chip methodology will accelerate dramatically," said Rodney Smith, Altera's president and CEO.

Altera has formed licensing agreements with MIPS and ARM to provide the processors for their new project. The company will also be integrating its own NIOS processor with their PLD architecture.

Three years ago, Motorola abandoned a similar project to incorporate processor cores into its Pilkington FPGA architecture.

CoWare Garners Financial Support

Sony, STMicroelectronics, and ARM invested a total of $14 million in CoWare, a company that produces tools and methodologies that aim to cut design time for system-on-a-chip projects. All three investors are long-term CoWare customers and partners. Not surprisingly, they had good things to say about the company. They also emphasized the role CoWare would play in their own projects.

"CoWare is a key ingredient in our platform-based design strategy," said Alan Dutheil, corporate vice president of strategic planning at STMicroelectronics. Takeo Minomiya, president of Sony's semiconductor company was similarly committed to CoWare's future: "Through our investment in CoWare, we aim to attain the means to continually improve our SOC design process."

Guido Arnout, CoWare's president and CEO reciprocated the positive sentiments. "These companies are leaders in the three most critical components for SOC success—semiconductors, systems, and IP," he said.

Before this round of funding, CoWare received nearly $14 million in venture capital.

Worldwide Shipments of Embedded HW Development Tools Exceeds $455 Million in 1999

Led by rapid gains in the acceptance and use of codevelopment tools, the world market for embedded hardware development tools totaled $455.1 million in 1999. ICE/JTAG shipments generated the greatest revenue, followed by logic analyzers, codevelopment tools, device programmers, and ROM emulators.

The Electronics Market Forecasters Group recent study of the hardware and codevelopment tools marketplace has analyzed the market dynamics behind the growth of these specific market segments.

The emerging use of system-on-chip custom silicon in advanced embedded designs, hardware-software integration, and debugging problems created by complex designs has resulted in a 64% increase in codevelopment tools shipments from the previous year.

North America consumed 53.7% of hardware development tool shipments followed by a 24.6% European consumption and a 21.7% Asian consumption.

Worldwide shipments of embedded hardware and codevelopment tools

Tools	1997	1998	1999
ICE/JTAG	148.5	159.5	172.3
Logic analyzers	92.8	107.8	130.8
Device programmers	65.9	64.7	68.2
ROM emulators	7.9	8.1	8.2
Codevelopment tools	25.1	46.1	75.6
Total	**340.2**	**386.2**	**455.1**

NAMES IN THE NEWS

GoAhead Software promoted **JAN E. CLAESSON** to executive vice president responsible for sales, account management, client services, and business development. **KEN HOSAC** was promoted to vice president of business development and **SHANNON SWIFT** was promoted to vice president of human resources and corporate services. **ALFRED J. STEIN** and **GARY KENNEDY** have joined ZF Linux Devices' Board of Directors. Stein is the former chairman and CEO of VLSI Technology. Kennedy most recently served as president/CEO and board member of Orbit Semiconductor.

Merger Forms New Interactive Television Technology Company

OpenTV Corp., which provides technology for digital television systems, completed a merger with Spyglass Inc, an Internet consulting company. The new company, which will continue on under the name OpenTV, aims to provide internet technology and services for the cable and satellite television industry. OpenTV's notion seems to be that the new company's combination of television software experience and broadband Internet expertise will allow it to take advantage of the oft-predicted convergence of the two media. In addition, Spyglass brings a U.S. presence to OpenTV, which, in its previous incarnation, was largely an international organization.

More Merger News

LynuxWorks, known until recently as Lynx Real-Time Systems, merged with Integrated Software and Devices. Under the terms of the agreement, LynuxWorks will acquire ISD Corp. through a stock swap transaction whereby newly issued LynuxWorks shares will be exchanged for all oustanding shares of ISD Corp.

The motivation behind the deal was the expectation that it would enable LynuxWorks to offer a broader range of scaleable embedded Linux products to the marketplace. In addition, LynuxWorks will now have Linux support for a wider variety of processors, including those from ARM and MIPS.

Reza Soliman-Noori , chairman and CEO of ISD, will serve as vice chairman of the combined companies. Dr. William Hogan will continue as president of LynuxWorks and Art Swift of ISD will become CEO of LynuxWorks. All will report to Inder Singh, present CEO and chairman of LinuxWorks.

Virata Acquires Agranat

Citing the value of key Web technologies and network management software, Virata Corp. announced that it has acquired Agranat Systems in exchange for 492,000 shares of Virata common stock, including assumed options.

"By integrating Agranat software into our DSL products, we hope to make technologies like DSL and broadband wireless easier to maintain and deploy," said Charles Cotton, Virata's CEO.

To support Agranat's existing customer base, Agranat employees will operate as a separate business out of their Maynard, MA office. The new unit will be managed by Ian Agranat.

Briefly noted...

The OSE RTOS from **Enea OSE Systems** now supports the MPC7400 PowerPC Microprocessor. ✳ **Virtio** announced its formal launch, as well as the introduction of a Web site for on-line virtual prototyping of embedded systems product. ✳ **eMation**, a provider of Internet infrastructure, acquired **Intuitive Technology Corp.**, a supplier of Internet technology for the manufacturing industry. ✳ **IKOS Systems**' VLE emulator was used by MIPS to verify multiple operating systems for the MIPS64 Kc 64-bit processor core. ✳ **Sky Computers**, a subsidiary of Analog Corp., named **Carnation Technology** of Beijing the exclusive distributor for its embedded computer systems in the People's Republic of China. ✳ **Hewlett-Packard** inked a seven-year multimillion dollar licensing agreement with **Delphi Automotive Systems** to embed HP's ChaiVM virtual machine and ChaiAWT graphics display software in a mobile multimedia product now under development. ✳ **Puma Technology**, an Internet software company, wants to change its name to **PUMATECH**. The company will put in a request to this effect at its annual meeting of stockholders in December. ✳ An alliance has been formed between **ZF Linux Devices** and **Phoenix Technologies**, in which the PhoenixBIOS will be bundled with the MachZ PC-on-a-chip. ✳ **Wind River Systems** has partnered with **ZF Linux** to provide the VxWorks RTOS for use in embedded applications with ZF's MachZ. **LynuxWorks** has also partnered with ZF Linux, this time to bundle LynuxWorks' BlueCat Linux and the LynuxWorks development tools with the ZF MachZ. ✳ In an effort to enable broader connectivity in its Internet appliances tools, **I-Logix** joined the Bluetooth Special Interest Group. ✳ An agreement between **inSilicon** and **Intoto** calls for cooperation in the marketing and distribution of Intoto's USB- and IEEE-1394-based products for use with inSilicon's semiconductor technology. ✳ **Nematron** and **Optimation Technology** have signed a letter of intent to merge by means of a stock transaction.

THE NUMBERS GAME

CENTURA SOFTWARE announced net revenue of $10.1 million and a net loss of $4.8 million for the fiscal quarter ended June 30. **AMD** reported sales of over $1.17 billion, operating income of over $250 million, and net income of over $207 billion for the quarter ended July 2. AMD's board of directors has authorized a 2-for-1 stock split. Stockholders of record on August 7 will receive one additional share of common stock for each share of AMD common stock held on that date. **STMICROELECTRONICS** announced that net revenues for the second quarter reached over $1.8 billion. Sales of differentiated products were just under $1.2 billion. **ZF LINUX** received $15 million in funding, which will be used for marketing and to begin volume production against a backlog of orders. This round of funding was led by Sands Brother Venture Capital LLC and included additional investments from National Semiconductor, Samsung, and TAT Capital Partners, among others. **SCENIX** also recently closed a $42.1 million equity investment. Primary participants included Cisco Systems, Dell Computer, JatoTech Ventures, and MSD Capital. **MICROCHIP TECHNOLOGY** reported net sales of $157.7 million for the first quarter of fiscal 2001 ended June 30.

Let's Work Together

San Jose McEnery Convention Center • Booth # 5089A
Embedded Systems Conference
September 26-28, 2000
Visit us at the
EMBEDDED SYSTEMS CONFERENCE



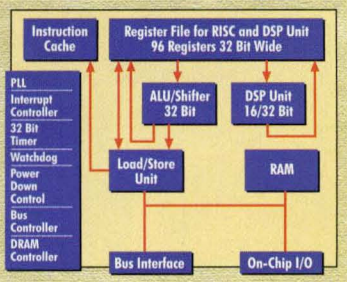








The Integrated RISC/DSP Solution



You need a fast RISC architecture. You want DSP support without the silicon overhead or software complexity of a dual-core design. Well, now you've found it. The new E1-32XS processor from Hyperstone. RISC and DSP working together. Delivering a fast time-to-market solution with the high power efficiency - 3.6 GOPS/Watt - demanded by today's consumer electronic products. We're already working together with many leading names in the industry. Shouldn't we be working with you, too?



RISC + DSP = hyperstone

USA: Phone: +1 323 888 1505 Fax: +1 323 888 1527 Europe: Phone: +49 (7531) 98030 Fax: +49 (7531) 51725 Asia/Pacific: Phone: +852 2951 0146 Fax: +852 2341 9314
www.hyperstone-ag.com

Green Hills Software Delivers...

...High Performance

ThreadX® RTOS
C/C++ Compilers
MULTI® IDE



Audio/Video Node Controller

...Fault Tolerance

INTEGRITY™ RTOS
C/C++ Compilers
MULTI® IDE



Wireless Network Controller

...Minimal Footprint

ThreadX® RTOS
C/C++ Compilers
MULTI® IDE



Virtual Jukebox

Royalty-Free Solutions for Embedded Development

Looking for a single source, fully integrated solution for all your embedded software development needs? Look no further! Green Hills Software delivers more than just tools; it's a leading edge provider of royalty-free solutions for 32/64-bit applications.

Fault-Tolerant RTOS
EEMBC Best Optimizing Compilers
Best of Class Tools

Green Hills Software's ThreadX® and INTEGRITY™ Real-Time Operating Systems are fully integrated with its market leading C, C++, EC++, Ada 95, and FORTRAN Compilers and MULTI® Integrated Development Environment. This combination provides a total development and run-time solution that addresses both deeply embedded and high-reliability applications.

Call or E-mail for a Free Demo CD



Green Hills
• SOFTWARE, INC. •

805.965.6044 • sales@ghs.com • www.ghs.com

RTOS • Compilers • Complete IDE





Only $3,200*

Logic*Wave*

- Debug designs with 8- and 16- bit microcontrollers
- 100 MHz state analysis and 250 MHz timing analysis
- 128 K memory per channel
- 34 channels of logic analysis

Doing what comes naturally. That's the idea behind Logic*Wave*, Agilent Technologies' new logic analyzer. It may be the easiest logic analyzer you've ever used. How? For starters, it's PC-based, for familiar, simple operation. So you won't need to waste hours learning—and relearning—the interface. You can draw the trigger easily by selecting a trigger condition from a drop-down menu. And its low cost means you can actually afford one of your own. Because a logic analyzer isn't very helpful if you have to share it with ten other engineers.





Experience Logic*Wave* for yourself. Visit our web site for a FREE download of the actual interface software. Or call us to get a FREE demo on CD-ROM. And talk to another engineer who'll answer your questions about Logic*Wave's* ease of use. Although, after you've used it once, you probably won't have any.

www.agilent.com/find/bi

1-800-452-4844, Ext. 6918

Agilent Technologies
Innovating the HP Way

*U.S. list price.

© 2000 Agilent Technologies ADGPP008/ESP



Jack W. Crenshaw

The Topic That Would Not Die

Boy, when I predicted that this series was going to be the longest-running ever, I wasn't just whistling Dixie, was I? I had no idea it would drag out as long as it has. I'm determined, however, to make a very large dent in the topic of minimization of a one-dimensional function this month. I want to get on with it, so I won't spend a lot of time philosophizing. I do, however, have a couple of thoughts that I need/want to share. I'll try to make them as brief as possible.

First, you may recall that in my last column, I ranted pretty loudly about my experience with Dell Computers. In the interest of fairness, I think I should offer a view from the other side.

No, Dell didn't come back and address all of my complaints. No, they didn't offer to take the computer back or give me a new one. Dell's service record remains as rotten as ever; I haven't heard from them at all. I did hear from two other disgruntled Dell customers, whose stories are as bleak as my own. One has had three video boards replaced. When he asked Dell to give him a different brand of board, one that actually works, they refused. Their position: you ordered Brand X, you get Brand X. Apparently, Dell is content to keep replacing the Brand X board forever. Meanwhile, this reader is stuck with a $3,000 boat anchor.

Another reader, however, took me to task for being unfair to Dell. Considering that he had similar experiences with Gateway, I guess I have to agree. It's not that Dell is a bad company to deal with, relatively speaking. It's that they're all bad.

I've been pondering this matter in great detail, and I think I've figured out what's going on. See whether you agree.

Back in the '40s, when I was a kid, manufacturers built stuff to last. The metal was thick and strong, the parts over-engineered, and the designs intended to allow maintenance, essentially forever. My parents had an old GE refrigerator, the kind with the cylindrical evaporator on top. That sucker would freeze a milk bottle as solid as steel if you turned the thermostat far from "min." Likewise, my wife and I had a portable Kenmore dishwasher that ran like a train for years. It was given to us used, and we handed it down to someone else. The only thing that stopped either one of those appliances was that the rubber gaskets rotted with age, and we couldn't find replacements. We had an old '46 Buick which was once sideswiped by a truck. It knocked about 100 pounds of caked mud out from under the car, but we were never able to find a single mark where the truck hit it. One night, in a drive-in theatre, a new Mercury sideswiped the same car. It ripped off his entire front-end grille, bumper, the works. We felt only the slightest bump, and again could find no damage.

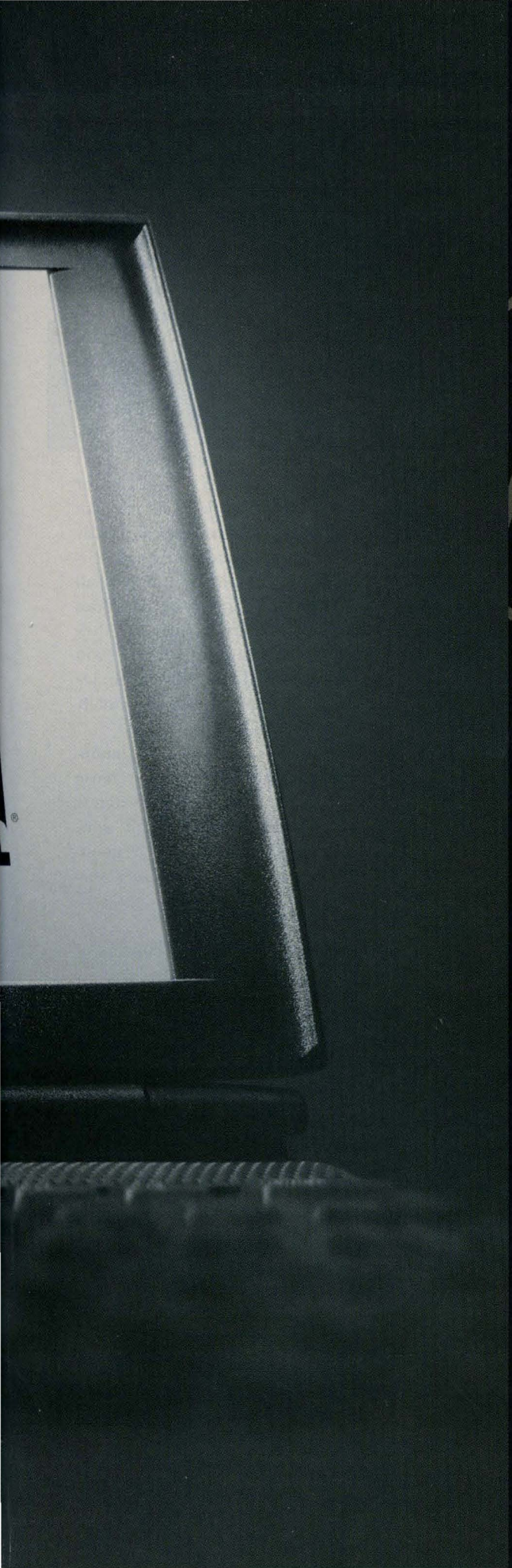
Then someone noticed that people weren't buying Buicks, refrigerators, or dishwashers anymore. Most folks already had one, and the products were so good, they saw no reason to "upgrade." In the '50s, some bean-counter invented a new concept. It's called *planned obsolescence.*

Locked in battle with his twin nemeses—Dell and minimization—Crenshaw pines for the good old days and gains some mathematical ground.

Around that time, I worked for two companies, GE and IBM, whose consumer products groups had what were arguably among the best engineers who ever earned a degree. Their marching orders were simple: design the product so it will fail the day after the warranty runs out. At first, every once in awhile they would screw up and actually produce a quality product. But over the years they got better, meaning that the products got worse. Today, thanks to computerized modeling tools such as Nastran and its kin, an engineer can design a part with a safety factor of 1.001, and just enough material in it so that it will barely hold together, assuming



©2000 Motorola, Inc. Motorola, DigitalDNA, The Heart of Smart, the DigitalDNA logo and the stylized M logo are trademarks or registered trademarks of Motorola, Inc. The PowerPC name is a trademark of International Business Machines Corporation and is used by Motorola, Inc. under license therefrom. 3Com is a registered trademark of the 3Com Corporation. All other logos are used by permission, and are trademarks or registered trademarks, of their respective companies.



Looking for a smart solution for your next-generation router, switch, server, modem, or desktop device? There's only one name to know: DigitalDNA from Motorola. DigitalDNA is chips, systems, software and ideas — embedded solutions that help smart companies create smart products. And you'll find it in leading-edge technology like 0.10µm copper interconnect CMOS-based solutions. The world's fastest copper interconnect SRAMS. And a wide portfolio of highly scalable PowerPC cores. So if you're ready to go big time with your networking or computing design, we're happy to help.

www.digitaldna.motorola.com

Digital DNA™
from Motorola

THE HEART OF SMART™

It doesn't take long to discover that if you take care of the people too well, they multiply like rabbits. The people plow under your croplands and build houses and stores on them. Next thing you know, everyone's starving, the coffers are empty, your army is in rebellion, and you're lucky to escape alive.

LISTING 1 Brent's method, Fortran version

```
      double precision function fmin(ax,bx,f,tol)
      double precision ax,bx,tol
      external f
      double precision f
      integer iter
c
c  an approximation  x  to the point where  f  attains a minimum  on
c  the interval  (ax,bx)  is determined.
c
c   input..
c
c   ax     left endpoint of initial interval
c   bx     right endpoint of initial interval
c   f      function subprogram which evaluates  f(x)  for any  x
c   in the interval  (ax,bx)
c   tol    desired length of the interval of uncertainty of the final
c   result (.ge.0.)
c
c   output..
c
c   fmin  abcissa approximating the point where  f  attains a
c   minimum
c
c   the method used is a combination of  golden  section  search  and
c   successive parabolic interpolation.  convergence is never much slower
c   than  that  for  a  fibonacci search.  if  f  has a continuous second
c   derivative which is positive at the minimum (which is not  at  ax  or
c   bx),  then  convergence  is  superlinear, and usually of the order of
c   about  1.324....
c   the function  f  is never evaluated at two points closer together
c   than  eps*abs(fmin)+(tol/3), where eps is  approximately  the  square
c   root  of  the  relative  machine  precision.  if  f  is a unimodal
c   function and the computed values of  f  are  always  unimodal  when
c   separated  by  at least  eps*abs(x)+(tol/3), then  fmin  approximates
c   the abcissa of the global minimum of  f  on the interval  ax,bx  with
c   an error less than  3*eps*abs(fmin)+tol.  if  f  is  not  unimodal,
c   then fmin may approximate a local, but perhaps non-global, minimum to
c   the same accuracy.
c   this function subprogram is a slightly modified  version  of  the
c   algol  60 procedure  localmin  given in richard brent, algorithms for
c   minimization without derivatives, prentice-hall, inc. (1973).
c
c
```

Listing 1 continued on p. 24

that it's never stressed. That approach is what passes for good engineering in consumer circles.

In fact, I contend that some products were never intended to work. I'm talking about the fairly cheap ones that come in blister packs, and *look* very much like the products you used when you were younger. The manufacturer counts on your buying it based on pleasant memories. It's not until you get it home and unwrap it that you discover it's a fake; it doesn't actually *do* anything. The vendor counts on your not deeming it worth the effort to return the junk. My favorite example is one of those circle-drawing compasses like we all used in grade school. The new ones look just like the old ones, but they won't hold a setting. What? You expected to actually draw a circle? You expected the stapler to actually drive staples? Silly you.

Nevertheless, until recently, manufacturers at least made some pretense of trying to keep their customers happy. If your product didn't work right, they'd try to make it good. They'd replace it, fix it for you, sometimes even offer you extra perks to keep you happy.

Now we move on to the concept of acceptable loss. Years ago, in the early days of personal computing, I got a game for my TRS-80 called Hammurabi, the ancestor of all the Sim City sorts of games. In Hammurabi, you're the king of Babylon, and it's your job to run the country so that it prospers, along with you. I reckon everyone who's ever played the game starts out resolving to be the world's most benevolent king. No starvation, no lack for the citizens of my country. I'm going to be remembered for keeping the populace safe, happy, and well fed.

It doesn't take long, playing such a game, to discover that if you take care of the people too well, they multiply like rabbits. Immigrants move in from outside to take advantage of your largesse. The people

Any other way of developing software is MANUAL LABOR.



Why slave over writing, validating, debugging and documenting your code...when you can actually enjoy designing it?

We feel your pain. That's why we created Rhapsody®. It's the only visual UML compliant real-time application software development environment that simultaneously integrates and automates analysis, design, implementation and test.

Our exclusive Model-Code Associativity guarantees that your model and code are always in sync. Change your model, your code changes. Or–you won't believe this–change your code, your model changes. And you won't even break a sweat.

Frankly, Rhapsody is amazing. It automatically generates readable, deployable, production-quality C, C++ and Java®. Not just code frames, but ALL the code. You can say goodbye to the tedium of manually developing makefiles, state machines, communication infrastructures and the like. You can also move from one RTOS to another with the push of a button. Yes!

With Rhapsody's unique design-level debugging you can visualize, easily detect and instantly correct logic and programming errors. In fact, you can actually test your application as you build it. And, wonder of wonders, you can import and reuse your legacy code. How's that for flexible?

Rhapsody users are already reducing their development cycle by at least 30%. C'mon, what are you waiting for?

Rhapsody will change the way you work. Forever.

Visit us at www.ilogix.com to download a free copy of Rhapsody.

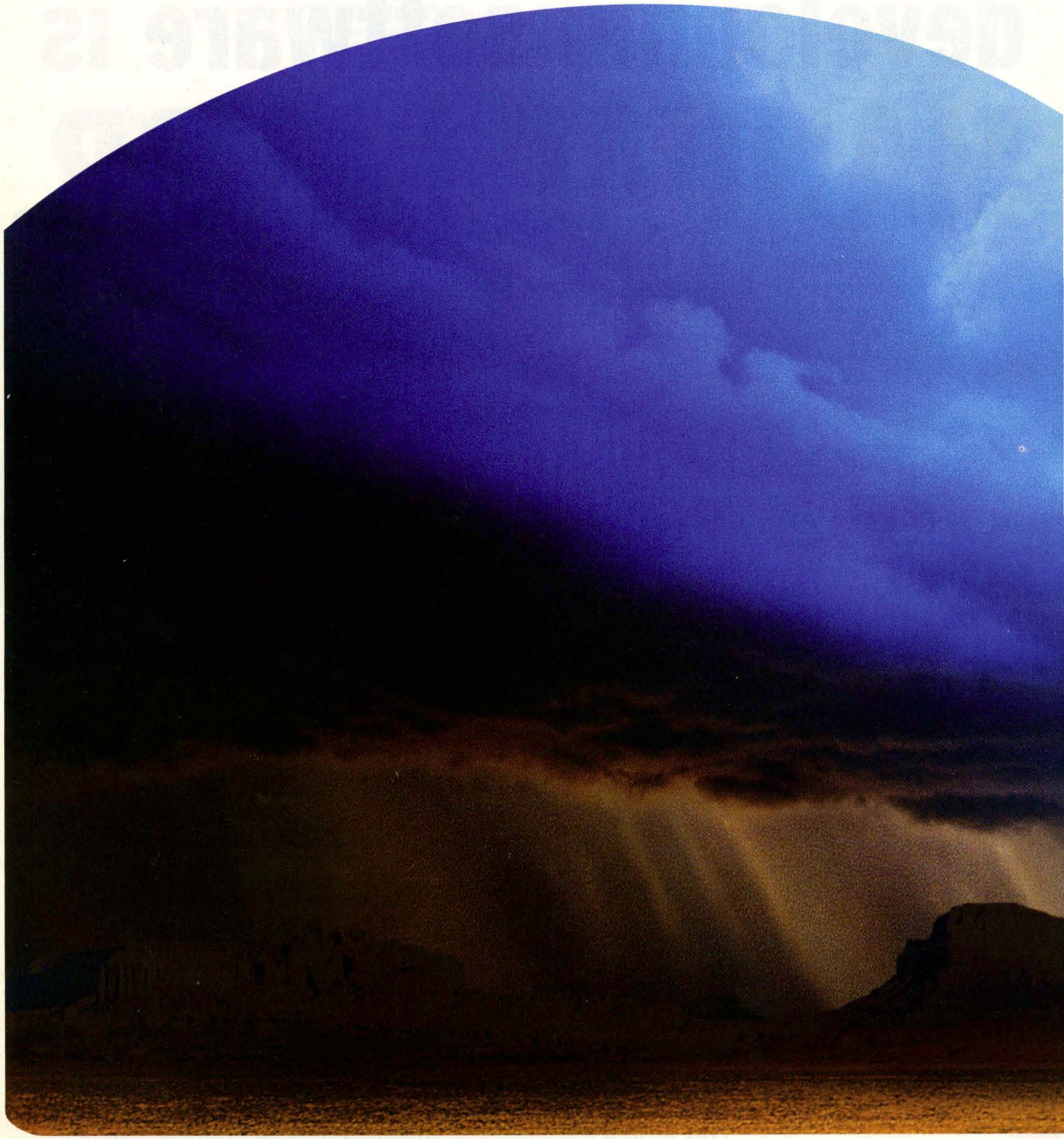
Rhapsody®

© 2000 I-Logix Inc. Java is a registered trademark of Sun Microsystems, Inc.

I-Logix®

Visit us at Embedded Systems Conference 2000, Booth# 1914

WindRiver®



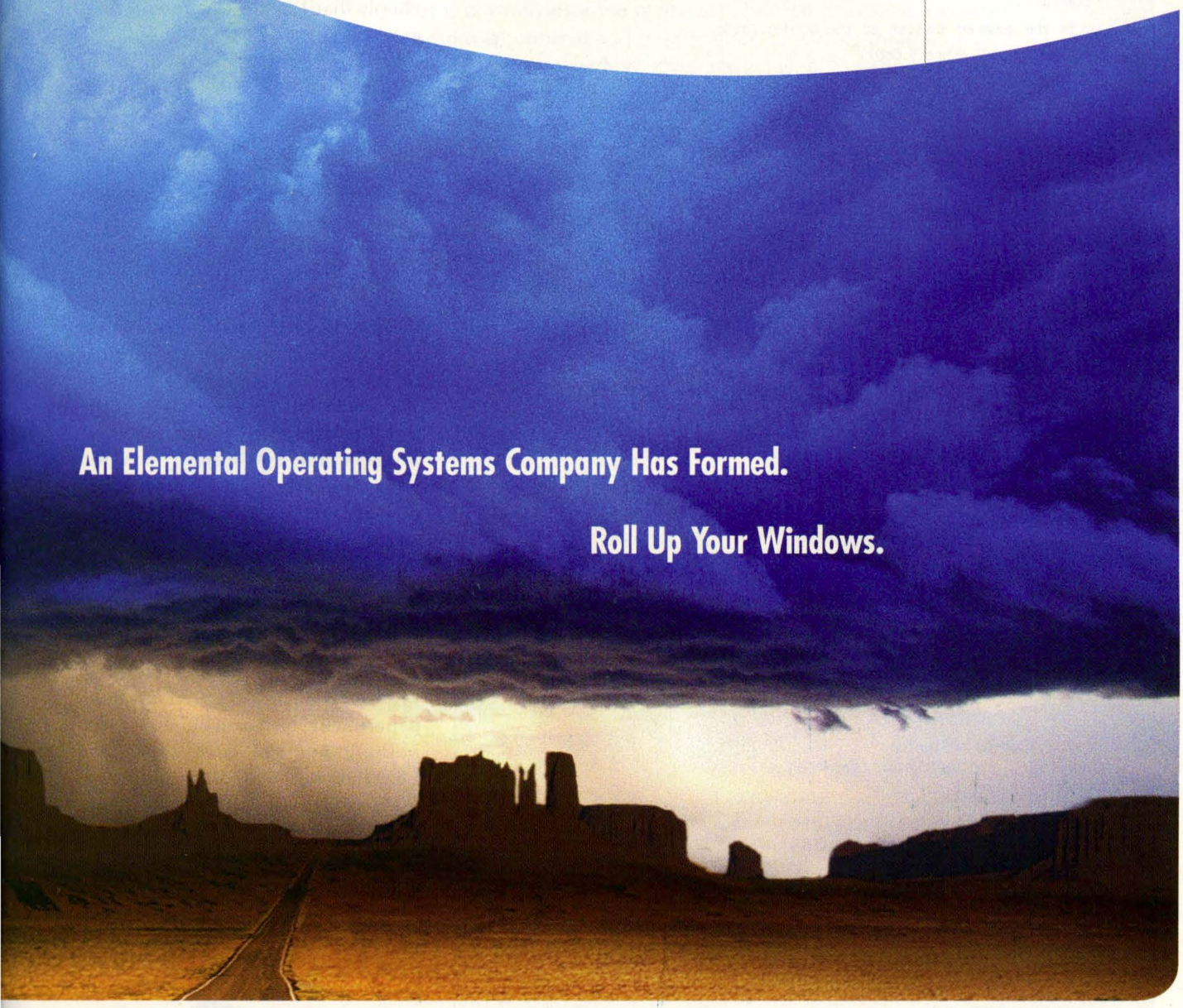
© 2000 Wind River Systems, Wind River, the Wind River logo, How Smart Things Think, ISI, DIAB SDS, Doctor Design and TakeFive Software are trademarks and registered trademarks of Wind River Systems, Inc.

There's a new force sweeping across the landscape, ushering in a new era of embedded capability. An era of operating systems and development tools that are balanced, strong and consistent, like nature itself. What's behind this? The merger of Wind River Systems and Integrated Systems, Inc., including DIAB SDS, Doctor Design and TakeFive Software. Separately, these companies were leading the way in software for smart devices. Together, they will offer unprecedented expertise, drawing from a vast pool of resources to craft targeted solutions for their clients. So behold the power, and put it to work for you. Call Wind River at 1-800-545-WIND today.



www.windriver.com

How smart things think™



So what does this have to do with Dell, Gateway, and other vendors? Simply that I believe we're entering a whole new era of planned disgruntlement. The grandchildren of the beancounters who invented planned obsolescence are now following in their ancestor's footsteps.

LISTING 1, cont'd. Brent's method, Fortran version

```
      double precision  a,b,c,d,e,eps,xm,p,q,r,tol1,tol2,u,v,w,fu,fv,fw,
     2 fx,x,tol3
      double precision  dabs,dsqrt,d1mach
c
      iter=0
c     c is the squared inverse of the golden ratio
      c=0.5d0*(3.0d0-dsqrt(5.0d0))
c
c     eps is approximately the square root of the relative machine
c     precision.
c
   10 eps = 1.0d-12
      tol1=eps+1.0d0
      eps=dsqrt(eps)
c
      a=ax
      b=bx
      v=a+c*(b-a)
      w=v
      x=v
      e=0.0d0
      fx=f(x)
      fv=fx
      fw=fx
      tol3=tol/3.0d0
c
c     main loop starts here
c
      xm=0.5d0*(a+b)
      tol1=eps*dabs(x)+tol3
      tol2=2.0d0*tol1
c
c     check stopping criterion
c
      do while(dabs(x-xm).gt.(tol2-0.5d0*(b-a)))
        p=0.0d0
        q=0.0d0
        r=0.0d0
        if (dabs(e).gt.tol1) then
c
c     fit parabola
c
          r=(x-w)*(fx-fv)
```

Listing 1 continued on p. 28

plow under good croplands and build houses and stores on them. Next thing you know, everyone's starving, the coffers are empty, your army is in rebellion, and you're lucky to escape alive. As hard a lesson as it is, to survive the game you must allow a certain percentage of your people to be sick, poverty-stricken, or dead. It's the only way the country can prosper.

So what does this have to do with Dell, Gateway, and other vendors? Simply that I believe we're entering a whole new era of *planned disgruntlement*. The grandchildren of the beancounters who invented planned obsolescence are now following in their ancestor's footsteps. What's more, they all know how to run what-if games using Microsoft Excel.

I can see it now. Someone runs a scenario that involves cutting the quality of a product. The simulation shows that a certain percentage of customers are going to get lemons, and making them happy is not going to be easy. The vendor runs a good risk of losing them permanently. But, wonder of wonders, the spreadsheet says we don't care. If the company is growing, as most are, it's gaining more customers than it's losing. What's more, the cost of paying decent wages to support people who actually support can run higher than the cost of of losing customers.

I'm willing to bet that lurking within a company like Dell, a whole department of beancounters does nothing but run such scenarios. Like Hammurabi, they optimize the tradeoff between lost customers and the cost of keeping them. They have optimized the cost of producing a quality product against the cost of lost sales. The spreadsheet program says, right there in black and white, that it's cheaper in the long run to let the irate customer remain irate. It's cheaper to let them walk away cursing than it is to spend time and energy trying to fix their problem.

SmartStack™

The lightest solution for complete Internet connectivity

e◇device TECHNOLOGY
The Internet enabler

SmartStack™ Technology



Integrate e-mail, FTP and Web-based control functionality in Industrial Equipment, Office Equipment, Home Appliances, Consumer Electronics

Application examples



- **Remote reading** of utility meters,
- **Remote maintenance** of industrial, office and home equipment,
- **Industrial control** and programmable logic controllers
- **Communication** with vending machines, automatic teller machines, point-of-sale terminals, home security systems, surveillance devices, air conditioning, heating and sprinkler systems.





Features

- **Direct Internet connectivity** to any equipment through any telephone line and any Internet Service Provider. No need for an extra modem or a gateway PC;



- **Ability to select protocols** (POP, SMTP, HTTP, FTP) depending on target application (send or retrieve e-mails, download files, display a web page);



- **Availability of simple interfaces** with the client application. The OEM can connect its device to the serial port of the **SmartStack**™ reference design, or integrate the chips (DSP, Flash, DAAs) within its design using the standards connectivity tools provided by eDevice, or port its application (for small applications only) on the DSP used for **SmartStack**™.



Advantages

- **Complete connectivity**. The modem and the Internet protocols are provided together. The client only needs to connect its application to **SmartStack**™;

- **Minimal cost and footprint**. As the complete solution (modem software, Internet protocols stack, client application or interface with client application) fits in a single modem DSP, there is no hardware cost or space added compared to a simple modem connectivity;

- **Minimal integration**. The real time integration between the modem and the Internet protocols stack is done by **SmartStack**™. **SmartStack**™ also offers standard interfaces to connect the client application to the Internet protocols.

DAvE's Back!

DAvE 2.0 – The Digital Application virtual Engineer from Infineon Technologies.

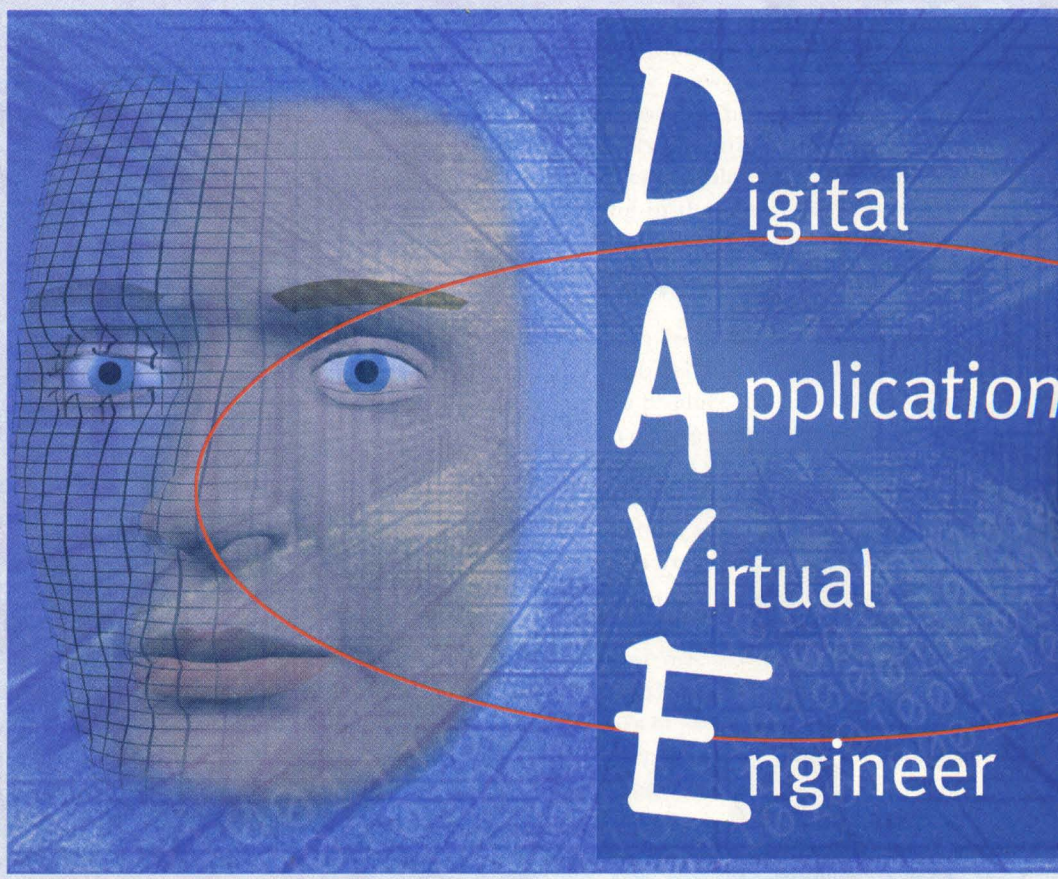
DAvE Who?

DAvE stands for Digital Application virtual Engineer and is Infineon Technologies' code generator for their range of 8-, 16- and 32-bit Microcontrollers. It provides initialization, configuration and driver code to ease programming for beginners as well as experts. DAvE comes with an easy to use windows user Interface running on Windows 95/98 and Windows NT. DAvE is now available in version 2.0 free of charge from Infineon Technologies.

Working with DAvE means that you can select your controller, click on the peripherals of your controller, select the functionality of your controller and tell DAvE to generate the appropriate code for you. The only work that remains for you to do is to write the application itself. Even then, DAvE will generate detailed documentation for your project.

The tooltip help provides detailed help on every disabled control – even giving reasons for why it's disabled. This will increase the ease of system configuration and allows developers to understand the controller in it's full functionality. Having multiple dialog boxes that can be kept open at the same time provides easy visual display of dependencies between various peripherals of a controller.

A register explorer also allows you to follow up changes of the configuration in the settings of registers. The explorer gives one-click access to the complete range of registers within the controller that the engineer is working with. The register-view provides information on the register address, the reset value, as well as the current value based on the configuration.

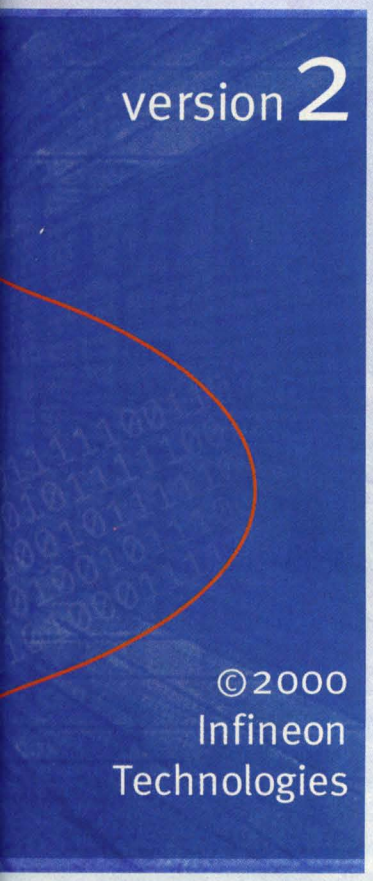


What's new in DAvE 2.0?

- The entirely new program source code enables DavE 2.0 to follow new and innovative object-orientated concepts
- The new MDI (Multiple Document Interface) User interface allows all configuration masks to be open at the same time
- It's programmable via VBA (Visual Basic for Applications) like Script language
- DAvE comes with SDK (Software Development Kit) toolchain, including Dialog editor, Template generator, Setup wizard, Deployment wizard and complete documentation. The SDK supports the Add-In interface as well as the derivative implementation interface
- Software Interface for 3rd party tool integration
- New register viewer, scaleable window showing register address, reset value, current value (due to configuration) as well as highlighted bits affected by the last change in settings
- New File Explorer
- New Setup that allows you to install DavE 2.0 with only the necessary components completely on the hard disk
- Setup includes automatic web update for components
- Add-in interface that allows complete access to the DAvE database from integrated components and features seamless integration of additional capabilities
- Support of more than 20 Infineon derivatives
- Support of the new 32-bit TriCore architecture
- Improved help system
- Enhanced code generation
- Improved error handling. Possible problems are shown

in the documentation file that is generated by DAvE.

Every controller that is supported by DavE 2.0 comes with the complete original documentation and DAvE 2.0 also offers direct links to related topics in the manual. Inside each configuration mask, any additional entries to the manual can be selected that are directly related to the selected mask settings.



The Tool Connection

DavE 2.0 offers a software interface allowing tool partners to directly communicate with DAvE and it's database. Via the interface, 3rd party tools can receive information about the generated code and the latest configuration. The software interface is already supported by many Infineon tool partners such as Keil, Tasking, Greenhills, Hitex, pls, nohau, Lauterbach and Appliware.

The DAvE SDK – Develop your own Add-Ins

DAvE 2.0 includes a complete Software Development Kit (SDK). This allows the developer to implement his own Add-Ins, as well as derivatives accessing the complete database of DAvE – includes manuals and SFR definitions.

Implementations are based on a VBA-like scripting language. The SDK comes with a dialog editor for custom DAvE controls as well as a template generator that develops the complete Add-In framework for the developer. A packaging wizard wraps the final product into a ready-to-go setup. The SDK is fully documented and an API reference is also additionally provided.

DAvE's On-Line

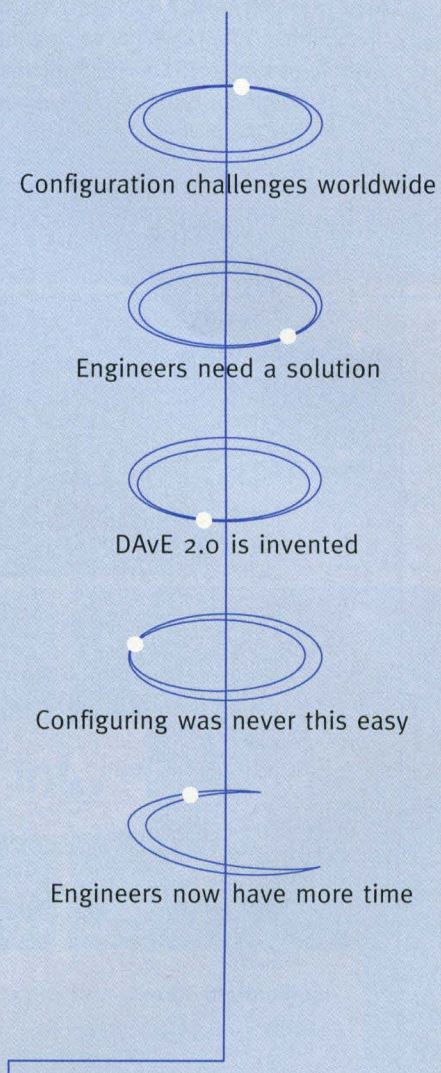
The amount of available derivatives for DAvE are constantly growing. To keep DAvE up-to-date, Infineon keeps the latest implementations on the DAvE website available for download. Go to www.infineon.com/dave to get the latest product information, a FAQ list, updates, news, new Add-Ins and discussions on DAvE.

The website also provides an ordering number for DAvE. Just write an eMail or FAX to obtain DAvE. For comments, suggestions and questions write to:
DAvE@infineon.com

©2000 Infineon Technologies AG

{PERPETUAL THINKING PROCESS}

INFINEON CYCLE



DAvE will help you compare and evaluate the different members of Infineon's 8-, 16- and 32-bit families of microcontrollers and help you find the right chip for your embedded control application.

www.infineon.com/dave

Infineon technologies

Never stop thinking.

Visit us at Embedded Systems Conference 2000, Booth# 1020

In case you hadn't noticed, I seem to have a tiger by the tail in this area of minimization. It was never my intention to re-invent the wheel, or write a whole new algorithm. After all, mathematicians have been minimizing functions for quite a long time now, and they should be pretty good at it.

LISTING 1, cont'd. Brent's method, Fortran version

```
        q=(x-v)*(fx-fw)
        p=(x-v)*q-(x-w)*r
        q=2.0d0*(q-r)
        if (q.le.0.0d0) then
          q=-q
        else
          p=-p
        endif
        r=e
        e=d
      endif

      if ((dabs(p).ge.dabs(0.5d0*q*r)).or.(p.le.q*(a-x))
     2       .or.(p.ge.q*(b-x))) then
c
c   a golden-section step
c
        if (x.ge.xm) then
          e=a-x
        else
          e=b-x
        endif
          d=c*e
      else

c
c   a parabolic-interpolation step
c
        d=p/q
        u=x+d
c
c    f must not be evaluated too close to ax or bx
c
        if (((u-a).lt.tol2).or.((b-u).lt.tol2)) then
          d=tol1
            if (x.ge.xm) then
              d=-d
            endif
          endif
        endif
c
c    f must not be evaluated too close to x
c
Listing 1 Continued on p. 32
```

In the end, the goals of departments called Tech Support and Customer Service are not what we laypeople think they are. They're not there to give support. Like the compass that can't draw a circle, they only give the *appearance* of offering support. And their goal is to do so while spending the minimum amount of money and using the least amount of resources. End result: the company is paying minimum wage to an inadequate and understaffed department. The people who deal with the complaints have only one goal: get the customer off the phone so we can move on to the next victim, er, valued customer.

What about minimization?

In case you hadn't noticed, I seem to have a tiger by the tail in this area of minimization. It was never my intention to re-invent the wheel, or write a whole new algorithm. After all, mathematicians have been minimizing functions for quite a long time now, and they should be pretty good at it. I thought that all I had to do was to pick up a canned technique called Brent's method, explain it to you, and move on. Silly me.

Therein lies yet another story. (Have you noticed I have a lot of stories? That's the price of advanced years, I'm afraid.) Years ago, when I first started working with computers, we didn't have much access to other folks' canned algorithms. If you needed a sine or cosine routine, you dug around until you found someone else's, or you wrote your own. At the time, I was working on things I thought were pretty important (helping mankind walk on the Moon), and I was hoping someone else would provide the algorithms I needed. They didn't.

A few years later, I went to work for IBM. At last, I thought, I'm in the company where the action is. Surely the company that practically invented (or at least popularized) the



could this be the only board design you'll ever need?

intel

the new intel® scalable performance board design fits multiple processors for multiple purposes. which gives you the key to survival in today's economy: flexibility. thanks to intel, you can design a single board around the 810 or 440bx chipsets, then plug in different intel® pentium® III or celeron™ processors depending on the end use of the board. the result: time and money aren't spent designing a unique board for each processor. and, even better, one universal board can serve a wide range of customers and applied computing applications—ranging from communications infrastructure and appliances to pos terminals to industrial pcs. because in the surge economy, if you're not flexible, you're history. download the design guide ➜ *http://developer.intel.com/design/info/070.htm*

© 2000 Intel Corporation. Intel and Pentium are registered trademarks and Celeron is a trademark of Intel Corporation. All rights reserved.

Microsoft Windows Powered

The Microsoft Windows Embedded Family.

The Microsoft® Windows® Embedded Family of operating systems offers a comprehensive software platform for bringing the next generation of intelligent and connected devices to market faster.

Choose from the widest range of embedded OS technologies that integrate seamlessly with Windows and the Web. Tap into a worldwide pool of more than 5 million Win32®-trained developers familiar with award-winning development tools, like Visual Studio®, to help develop your embedded applications.

With three OS's to choose from, there's one that's right for your next project.

*While supplies last. Online orders have a fee of $12.95 which includes shipping and handling.

It's not for everyone.

Only those interested in faster time to market.

The Microsoft Windows Embedded Family consists of:

Windows CE 3.0

The modular, real-time OS, perfect for small footprint 32-bit devices. Supports multiple processor architectures? Yes. Built-in network connectivity? Yes. Instant-On? Yes. Build your devices quickly with Windows CE Platform Builder 3.0.

Microsoft® Windows® CE

Windows NT® Embedded 4.0

The full-featured embedded OS designed for systems based on PC architectures. Supports thousands of device drivers and Windows-based applications. Enables the richest applications and services for your embedded solution.

Microsoft® Windows NT® Embedded 4.0

Windows 2000 with the Server Appliance Kit

Creating dedicated server appliances? Take advantage of Windows 2000 performance, reliability, and security. Use the Server Appliance Kit with predefined templates and headless server support to build your dedicated servers.

Microsoft® Windows 2000

Get time on your side.
Come visit us at ESC WEST, booth #808 and pick up your FREE Windows Embedded Evaluation Kit*.
Or visit www.microsoft.com/embedded/eval and order online.

Where do you want to go today?® Microsoft®

That was then, this is now. Nowadays, companies all over the world write packages for solving math problems in Fortran, C, and other languages. On the Internet, applied mathematicians in academia pass around algorithms for solving all manner of complex problems.

LISTING 1, cont'd. Brent's method, Fortran version

```
          if (dabs(d).lt.tol1) then
            if(d.le.0.odo) then
              u+x-tol1
            else
              u=x+tol1
            endif
          else
            u=x+d
          endif
            fu=f(u)
          iter=iter+1
c
c     update  a, b, v, w, and x
c
          if (fx.le.fu) then
            if (u.ge.x) then
              b=u
            else
              a=u
            endif
          endif
          if (fu.gt.fx) then
            if ((fu.gt.fw).and.(w.ne.x)) then
              if ((fu.le.fv).or.(v.eq.x).or.(v.eq.w)) then
                v=u
                fv=fu
              endif
            else
              v=w
              fv=fw
              w=u
              fw=fu
            endif
          else
            if (u.ge.x) then
              a=x
            else
              b=x
            endif
            v=w
            fv=fw
            w=x
            fw=fx
Listing 1 continued on p. 36
```

"Giant Brain" would have researched the math algorithms needed to solve virtually any problem, and kept canned versions handy to give to their engineers.

If such a group existed at IBM, it's news to me. I think it's a case where the cobbler's children have no shoes. Far from getting the best software IBM had to offer, we IBM programmers got to be the beta testers for the rest of the company. We didn't even get the same quality that customers did. As usual, I ended up writing my own algorithms.

My next job was with Northrop Corporation, which had a small IBM mini (the 1130, my only favorite IBM computer). Ironically enough, that computer came with a document called the IBM Scientific Subroutine Package (SSP). IBM did have a collection of tools; it's just that I had to leave IBM to find it. Though written in the worst kind of spaghetti-code Fortran II, the algorithms were pretty good, and to this day, I still use many of them.

That was then, this is now. Nowadays, companies all over the world write packages for solving math problems in Fortran, C, and other languages. On the Internet, applied mathematicians in academia pass around algorithms for solving all manner of complex problems. Authors like Press et al. write books like *Numerical Recipes,*[1] which must contain the best algorithms of each kind. Surely I can be forgiven for assuming that I'd be able to pick up one such algorithm and present it to you in the form of a robust and general-purpose minimum-finder. I thought that in Brent's method I'd found one.

Alas, it was not to be. Not only did I not like the method, I couldn't find the one thing I absolutely insist on in this column: a cogent explanation of how it works. Who could have guessed this turn of events?

Nevertheless, like it or not, I seem to be at that branch point in this

LANTE IBM ECAMPUS.COM CALICO COMMERCE HEWLETT-PACKARD RECRUITSOFT.COM ROCKWELL SOFTWARE
IXL **HEWLETT-PACKARD** EZENIA! **ERICSSON** TRELLIX **ALCATEL** AUCTIONROVER.COM **LANTE** BOOKSONLINE
ARTS **J.D. EDWARDS** COMMERCE ONE **ICG COMMUNICATIONS** FOOD.COM **DELOITTE CONSULTING** IXL SUN
MS TICKETWEB **TRELLIX** MICROSOFT **ECAMPUS.COM** PEOPLESOFT **ROCKWELL SOFTWARE** LANTE DELOITT
EZENIA! **PEOPLESOFT** J.D. EDWARDS LANTE **AUCTIONROVER.COM** ERICSSON ALCATEL **IBM** BOOKSONLIN

THE INTERNET DEPENDS ON SOFTWARE FROM COMPANIES LIKE THESE.

COMPANIES LIKE THESE DEPEND ON RATIONAL.

.COM **NETWORK ASSOCIATES** EZENIA! **MICROSOFT** FOOD.COM **SUN MICROSYSTEMS** ALCATEL HEWLETT-PA
FT **IXL** ERICSSON **FOOD.COM** SUN MICROSYSTEMS RECRUITSOFT.COM **CALICO COMMERCE** COMMERCE ONE
FTWARE **COMMERCE ONE** IBM **TICKETWEB** MICROSOFT **RECRUITSOFT.COM** TRELLIX **EZENIA!** J.D. EDWARDS
NSULTING **CASE PARTS** AUCTIONROVER.COM **BOOKSONLINE.COM** CASE PARTS **CONTEXT INTEGRATION** ECAM

Companies creating software in the electronic, Internet-driven economy are faced with a modern e-software paradox: how do you create software faster – in Internet time – and achieve the high quality you need to satisfy customers? The answer is e-development from Rational® Software. Thousands of companies around the world increase their software development speed *and* quality by applying Rational's software engineering best practices, unified tools and services. In fact, 47 of the "Fortune e-50" use Rational products, and *The Wall Street Journal* recently ranked Rational Software as the number one company in a survey of "Companies Wall Street Analysts Liked Best." Want to build software faster *and* better? Visit us at www.rational.com/success.

Rational®
the e-development company™

© 2000 Rational Software Corporation. All trademarks are the property of their respective owners.

nohau

In-Circuit Emulators

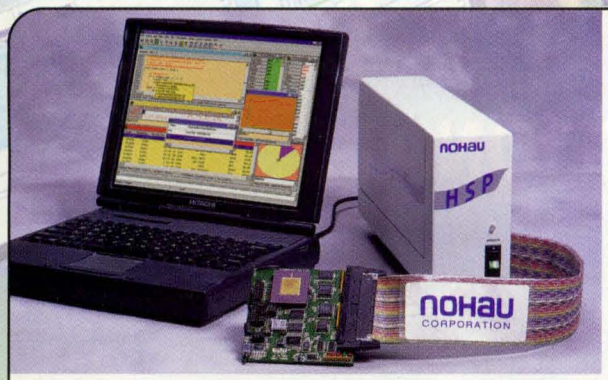


www.nohau.com/newflyers/st10.pdf & c166.pdf

Infineon C166 and ST Microelectronics ST10: Nohau has the broadest and fastest line of C166 and ST10 emulators in the world. Up to 80 MHz with no cycle stealing from the bondout CPU. You are assured that your programs always run full speed as intended with no CPU stalling. The Shadow RAM displays writes in real-time at all processor speeds.

These emulators are hand held portable models allowing operation anywhere you and your laptop can go: or share in your lab. Connection to your LPTx: port or via an ISA card for versatility. Emulation memory can be mapped down to a granularity of 2 bytes allowing wrapping around any peripheral. This easily accommodates your targets in various stages of development.

Nohau provides personality modules to adapt to new derivatives as they are introduced so you never have to wait for support. You can start development using the emulator in stand-alone mode before the target chips are available. These emulators are designed and Made in the USA.



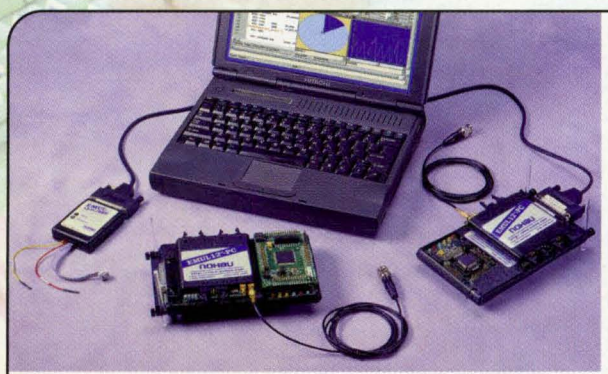
www.nohau.com/newflyers/8051.pdf, c500.pdf, & dallas.pdf, 251.pdf & xa.pdf

8051 and Infineon C500 Families: Nohau is the world's largest 8051 emulator manufacturer. Nohau has more bondout experience than anyone else. This assures you of trouble free single-chip emulation with processors from Intel, Dallas, Philips, Infineon, Atmel and others. Nohau supports nearly all 8051 derivatives. Our website at www.nohau.com/newflyers/support.pdf gives you the complete list of all devices supported.

Nohau offers a wide range of options to suit any need and budget you have including rentals. Nohau technical support is legendary and with reps everywhere, you always get fast answers to get you developing sooner.

Nohau supports the Philips XA and the Intel and Temic 251 family if you use these upgrade paths from the 8051. If you use the Infineon C500 and C166 families together in your projects, both emulators use the same user interface. This eases moving from one family to the other hence increasing productivity.

Nohau works seamlessly with all popular compiler and RTOS developers.



www.nohau.com/newflyers/hc12.pdf

Motorola HC12: Finish your developments on time with the most complete emulators supporting the B32, BC32, D60, DA128, DG128 and more! Your new emulator is obsolete-proof since future derivatives will be easily handled with economical personality modules.

The pipeline is fully decoded preventing false triggering on fetched, yet unexecuted instructions. The ports are reconstructed using Nohau's own FPGA design which provides true single chip operations. No chip resources are used by the emulator and full speed, full feature operation is assured. The optional trace board allows triggering on user-specified events to increase productivity. FLASH and EEPROM programming is supported for appropriate chips.

Nohau has a BDM emulator for the above parts plus the HC12A4, HC16 and the 683xx family. Nohau completes its Motorola support with full emulators for the HC16 and 683xx, and the HC11. See the website www.nohau.com/newflyers for more information or call your local Nohau representative.

Cut Your Development Time!

with Nohau in-circuit emulators.



www.nohau.com/newflyers/m16c.pdf

Mitsubishi M16C: Nohau suppoorts this family with a new emulator which has the ability to program the FLASH in your target CPU. The photo illustrates this setup. Like most Nohau products, this emulator is handheld.

The pipeline is fully decoded and the optional trace board displays cycles filtered by the triggers. The entire 1M space is emulated with a granularity of 2 bytes. Shadow RAM displays all writes in real-time in several numerical or graphical formats. The emulator operates in single-chip or expanded mode.

It operates to 16 MHz and with a Vcc of 2.7 to 5.5 volts, it will easily adapt to your target. The triggers can be set and the trace viewed without stealing cycles from the CPU thanks to the internal housekeeping controller. All Nohau emulators operate in this fashion giving you the truest emulation possible.

To speed up your M16C project, contact Nohau today for the latest information.

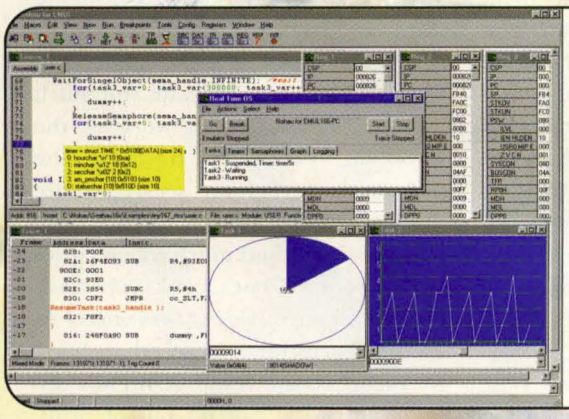


www.nohau.com/newflyers/xa.pdf

Philips XA and others: Nohau supports many other families including the XA, 251, 196, 296, HC11, HC16, 683xx. More are under current development. They all use the same Seehau interface for continuity and the same Nohau quality that assures you of a faster development process. For data sheets on these products, browse www.nohau.com/newflyers. For information on how to use an emulator in your development, call Nohau or your local rep.

Check other areas of the Nohau website for application notes, useful links to our partner's websites, new product announcements and software upgrades.

Nohau has shipped over 18,000 emulators since 1986 and is the world's largest supplier of 8 and 16 bit emulators. Nohau reps and offices are everywhere offering the best in customer service. Nohau is headquartered in Silicon Valley USA and has major offices in Sweden, Germany, China and Great Britain.



Seehau: Nohau's new user interface: Seehau provides you with a sophisticated yet friendly debugging environment. For a complete description see the website www.nohau.com/newflyers/seehau.pdf. Call Nohau for the latest demo CD or call your Nohau rep for a personal demonstration at your site.

Seehau can operate over TCP/IP for remote debugging, has integral RTOS support, multiple configurable windows, a MACRO scripting language including an integrated Visual Basic (Sax Basic), and much more.

Data writes can be displayed in many numerical and graphical formats and the trace can be set and viewed in real-time. A built-in editor allows modifying code, then calling your compiler and then displaying any error messages - all within one environment. Seehau runs under Windows NT/95/98. Nohau offers no cost upgrades from the web and they can be retrofitted to existing Nohau emulators.

Your Action List:

- ☐ Visit our website for app notes and "The Designer's Guide to In-Circuit Emulation"
- ☐ Call your local Nohau rep to see how they can help you complete your projects faster.
- ☐ Nohau reps offer complete development solutions for easy integration of projects.
- ☐ Call Nohau for a free evaluation period.

www.nohau.com

Nohau Corporation
51 East Campbell Avenue
Campbell, California 95008
Email: sales@nohau.com

Tel: (888) 886-6428
Tel: (408) 866-1820
Fax: (408) 378-7869
Web: www.nohau.com

You've heard me say, in the past, that I was not a great fan of the Numerical Recipes series. Popular as it is, I always get the feeling that I'm getting a canned subroutine, converted over from old Fortran code by people who didn't understand it all that well.

LISTING 1, cont'd. Brent's method, Fortran version

```
            x=u
            fx=fu
          endif

c
c     update tolerances for next loop
c
          xm=0.5d0*(a+b)
          tol1=eps*dabs(x)+tol3
          tol2=2.0d0*tol1
      enddo

c
c  end of main loop
c
      fmin=x
      return
      end
```

study. I absolutely will not be deterred from giving you an algorithm this month. If it has to be one I've invented on the spot, that's what it has to be.

More on Brent and Press

Now that I've taken on, in recent months, Microsoft, Mathsoft, and Dell, can it be that I'm about to take on Press et al. as well? Yeah, I guess it can. You've heard me say, in the past, that I was not a great fan of the *Numerical Recipes* series. Popular as it is, I always get the feeling that I'm getting a canned subroutine, converted over from old Fortran code by people who didn't understand it all that well.

Don't get me wrong; the scope of coverage of *Recipes* is monumental. Press et al. give more subroutines to solve more problems than I am likely to write in several lifetimes. I've always stood in awe over the sheer volume of algorithms that are covered in the book. Whenever I want to learn more about certain problems, I still go to *Recipes* first. Still, one is left with that nagging question: how can one or a few people be experts in so many different areas? Obvious answer: they can't. When I want a truly authoritative explanation of an algorithm, I generally find that I must look elsewhere.

I thought I was alone in my reaction to *Recipes*. A reader pointed out that I was not, and referred me to the Web site: *math.jpl.nasa.gov/nr/*.

To spare you the suspense of looking it up, I'll quote the title: "Why not use *Numerical Recipes*?" and the first line, "We have found *Numerical Recipes* to be generally unreliable."

Pretty strong words. Author W. van Snyder of NASA-JPL goes on to quote several specialists who share the feeling that *Recipes* gives one just enough information to get into trouble. One of the experts quoted is my hero, Lawrence F. Shampine. Shampine is arguably the world's leading authority on numerical integration algorithms. Whatever he says, you can take to the bank.

The general flavor of the Web site article is that while *Recipes* may be a good source for serviceable canned subroutines to solve certain problems, they are not the ones an expert in that particular area would use.

Understand, I don't intend this as any great indictment of the *Recipes* books. Press et al. have never presented the routines as the most powerful or effective of their kind. Their goal seems to be to cover as much ground, in as many areas, as possible. It would be asking a lot to expect them to not only give you algorithms that work in so many different areas *and* represent the current state of the art in every one. My only problem is that the current state of the art, for this particular area, is exactly what I was looking for.

Please don't get the idea that, though I haven't been with you much lately, I haven't been working the problem. I've been traveling the Information Highway seeking more background on Brent's method. I'm afraid I didn't find much. *Recipes* gives as reference Brent's book, *Algorithms for Minimization Without Derivatives*.[2] As we've discussed before, that book seems to be the result of Brent's dissertation work. But it's out of print, and my perpetual request to Amazon.com has not turned up a copy.

In search of more explanation of Brent's method and how it works, I turned to Alan Miller's Web site *www.ozemail.com.au/~milleraj/*.

Now, if you're looking for someone who really is an expert in every

www.vci.com For more information.

Time for NT Embedded, Windows 2000, or Windows CE As Part of Your Next Product?

How will you handle:

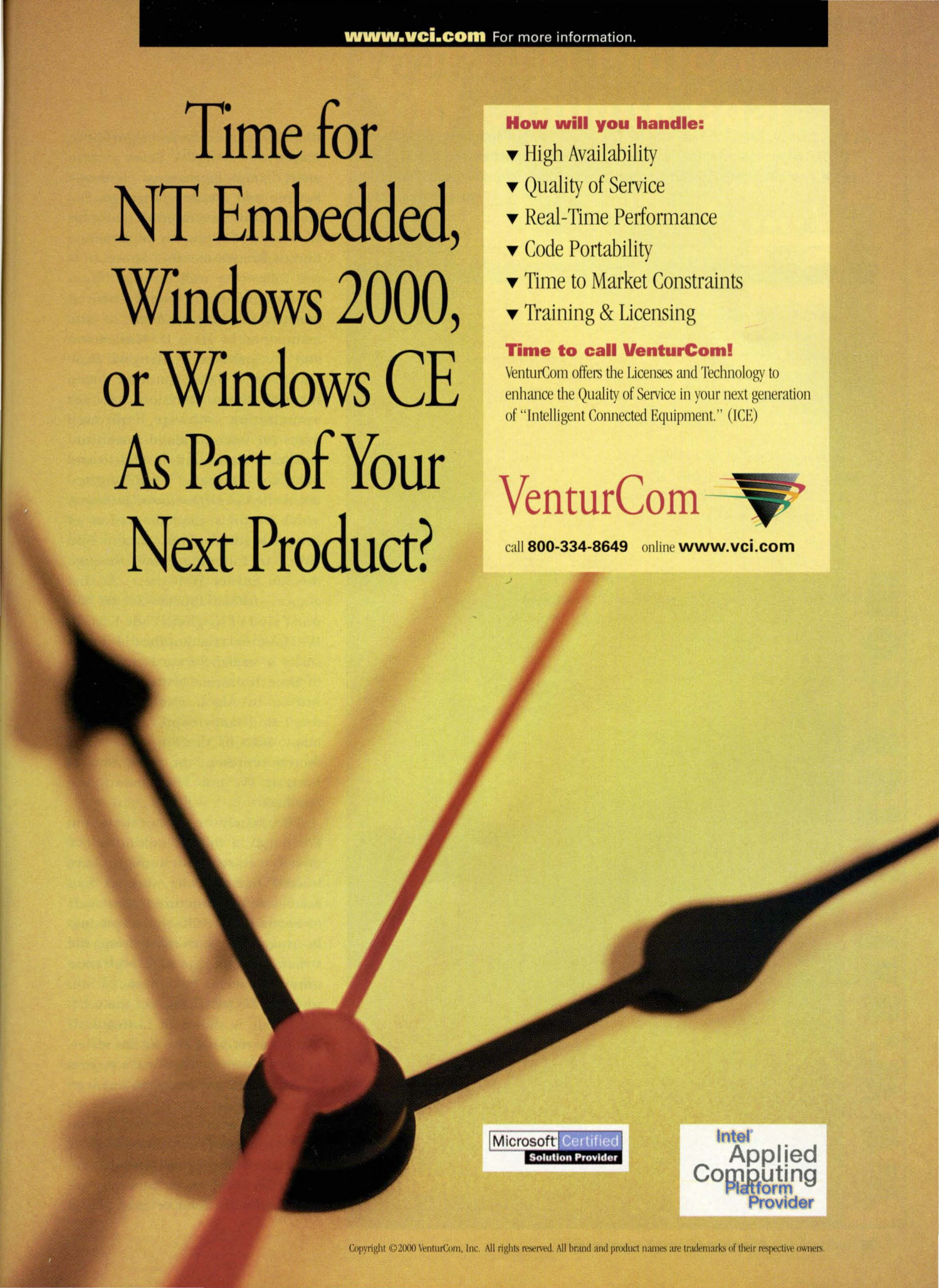
- High Availability
- Quality of Service
- Real-Time Performance
- Code Portability
- Time to Market Constraints
- Training & Licensing

Time to call VenturCom!

VenturCom offers the Licenses and Technology to enhance the Quality of Service in your next generation of "Intelligent Connected Equipment." (ICE)

VenturCom

call **800-334-8649** online **www.vci.com**



Microsoft Certified Solution Provider

Intel Applied Computing Platform Provider

Copyright ©2000 VenturCom, Inc. All rights reserved. All brand and product names are trademarks of their respective owners.

At first glance, fmin.f appears to be the worst kind of spaghetti code Fortran IV. However, take another look. It's really a straightforward translation of Brent's original code, which was written in Algol. (No wonder we don't find many copies of it!)

LISTING 2 Functions for golden section search

```
// swap two double-precision floats
void swap(double &x1, double &x2){
      double temp = x1;
      x1 = x2;
      x2 = temp;
}

// divide an interval according to the golden section
double gold(double x0, double x2){
      static const double g = 2-golden;
      return x0 + g *(x2 - x0);
}

// shrink an interval by dividing one side of it
void gold_shrink(double &x0, double &x1, double &x2,
                 double &y0, double &y1, double &y2){
      double x = gold(x0, x1);
      double y = f(x);
      if(y > y1){
        x0 = x;
        y0 = y;
      }
      else{
        x2 = x1;
        y2 = y1;
        x1 = x;
        y1 = y;
      }
      swap(x0, x2);
      swap(y0, y2);
}

// Find the minimum of a function using golden section search
double gold_search(double &x0, double &x2, double Eps){
      double eps = Eps * (abs(x2-x0));
      double x1 = gold(x0, x2);
      swap(x0, x2);
      double y0 = f(x0);
      double y1 = f(x1);
      double y2 = f(x2);
      while(abs(x2-x0) > eps)
        gold_shrink(x0, x1, x2, y0, y1, y2);
      return (x0+x2)/2.0;
}
```

single area of numerical algorithms, Alan's your man. I've never seen so many implementations (mostly Fortran and the new language, F—more on this next month). Sadly, his Web site doesn't list an implementation of Brent's method. However, if you follow the trail through "Nonlin. programming," you'll arrive at *plato.la.asu.edu/guide.html*, a site maintained by Hans D. Mittlemann and P. Spellucci of Arizona State University. Take another step through their "decision tree for optimization software," through paths for unconstrained linear and nonlinear optimization, and lo and behold, there's *netlib.org/cgi-bin/netlibfiles.pl?filename=/go/fmin.f*, which is their implementation of Brent's method. (You can also find `fmin.f` by following a trail through the van Snyder Web site.) At first glance, `fmin.f` appears to be the worst kind of spaghetti code Fortran IV. However, take another look. It's really a straightforward translation of Brent's original code, which was written in Algol. (No wonder we don't find many copies of it!) The many `GOTO`s in the Fortran version merely represent the limitations of Fortran IV, not Mittlemann and Spelluci.

Fortunately, we are not limited to Fortran IV. I took a look at `fmin.f` myself, to see what it might have looked like had its authors had access to the structured constructs of Fortran 77 or 90. Not surprisingly, since it was derived from the structured language Algol, the code unravels nicely. For those of you who like Fortran, a copy of the modified code is shown in Listing 1. If I've done nothing else in this series, at least I've given the world a version of Brent's algorithm that one can actually read.

Comparing the Fortran code from `fmin.f` with the C code from *Recipes*, I was struck by two things. First, they are almost identical; even the same variable names are used in most places,

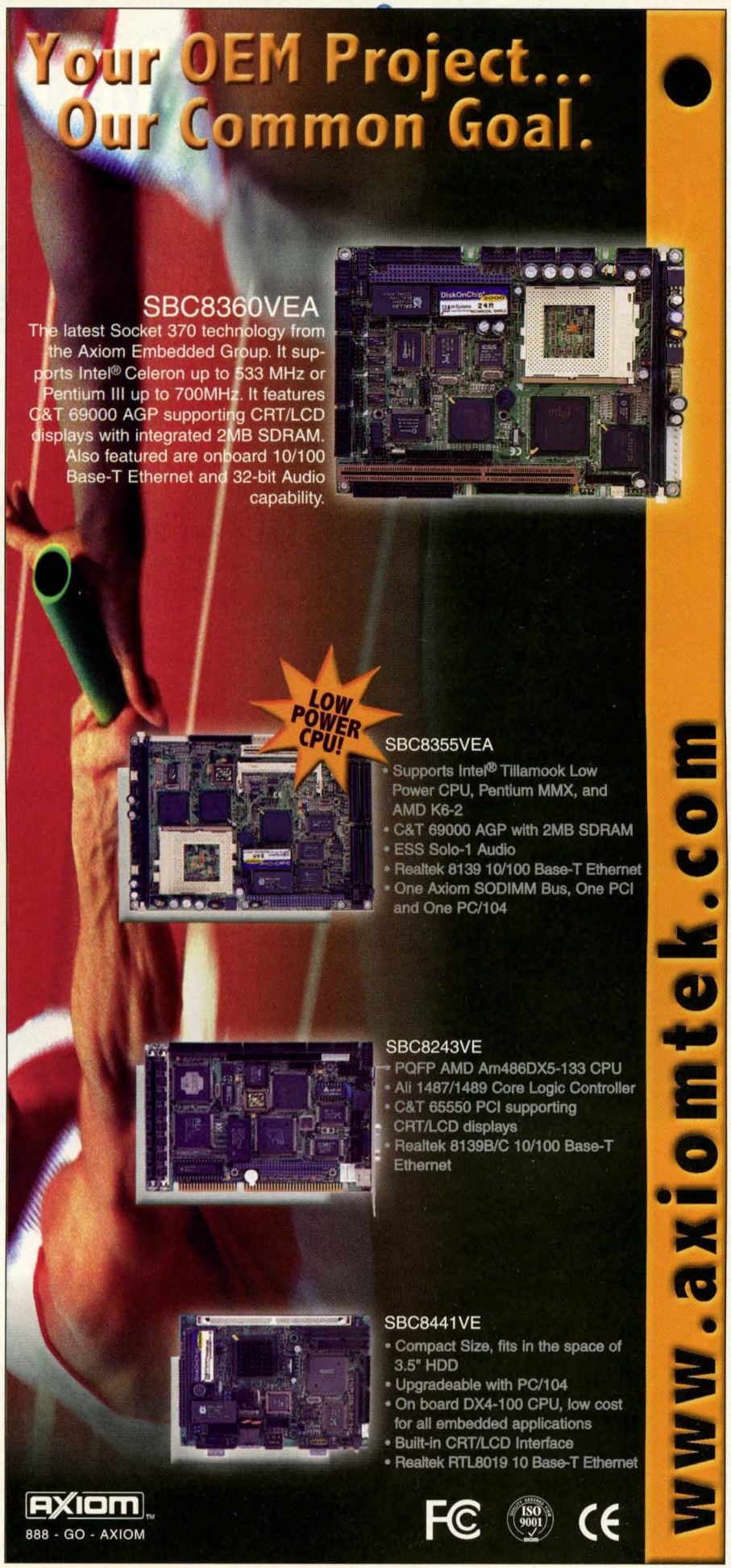
and the same operations are done in pretty much the same places. Somewhat surprisingly, though, they are not *exactly* identical. Nor do they perform identically. On the same problem, my structured version of `fmin.f` took 11 iterations to converge to double-precision accuracy. The *Recipes* version, `brent.c`, took 19 iterations to converge to only single-precision accuracy.

At this point in my trek, I am completely bummed out. I suppose, for the sake of completeness, I should have been spending my summer vacation comparing the two programs and trying to discover the reason why the Fortran version converges nearly twice as fast. I didn't. To be brutally honest, I don't really care anymore. I've seen all of Brent's method I want to see.

Can we improve on Brent?

When I began this odyssey, it was certainly not my intention to try to invent the world's greatest second-order minimization algorithm. And I must say, I feel pretty nervous about attempting it. This is not the kind of thing one does casually. A lot of users out there want good algorithms, and if my offering turns out to have one or more fatal flaws, you can be sure I'm going to be hearing from them in droves. I'm sure to make van Snyder's hit list. However, at this point, I don't see that I have much of an alternative. My goal has always been to give you a method that not only works, but that I can explain to you rationally. If I can't explain Brent, I'll explain Crenshaw.

There's also no way I can spend the kind of time Brent presumably spent on his dissertation project. To come up with the perfect algorithm, one should run it on as many different kinds of problems as possible, looking for the cases that give it trouble, and then patching it up to deal with them. I don't have time to do that here.



So I'll make a deal with you. Let's do this thing together. I'll give you an algorithm that seems to work on a limited number of cases. Feel free to try it out and let me know how it works.

So I'll make a deal with you. Let's do this thing together. I'll give you an algorithm that seems to work on a limited number of cases. Feel free to try it out and let me know how it works. If you find problems with it, get back to me, and we'll fix it together. Between us, maybe we'll arrive at that killer solution, after all. Deal?

LISTING 3 Functions for parabolic fit

```
// estimate a minimum via parabolic fit
double para_fit(double x0, double x1, double x2,
                double y0, double y1, double y2){

      double m0 = (y1 - y0)/(x1 - x0);
      double m1 = (y2 - y1)/(x2 - x1);
      double c  = (m1 - m0)/(x2 - x0);
      return (x0 + x1 - m0/c)/2.0;
}

// shrink an interval using parabolic fit
void para_shrink(double &x0, double &x1, double &x2,
                 double &y0, double &y1, double &y2){

      double x = para_fit(x0, x1, x2, y0, y1, y2);
      double y = f(x);

      if(x > x1){
        swap(x0, x2);
        swap(y0, y2);
        swap(x, x1);
        swap(y, y1);
      }

      if(y > y1){
        x0 = x;
        y0 = y;
      }
      else     {
        x2 = x1;
        y2 = y1;
        x1 = x;
        y1 = y;
      }
      swap(x0, x2);
      swap(y0, y2);
}
```

Getting serious

Okay. We've pussyfooted around this issue for long enough. Let's put this sucker to bed. Before doing so, however, maybe I should review the bidding. We've already seen, months ago, an algorithm that can reliably find a minimum of a single variable by means of bisection. Bisection using the golden ratio turned out to work a little better than halving intervals.

The problem with any bisection method is that it takes awhile to get the solution. We can expect each step to improve our solution by just half a bit. For improvement by a factor of, say, 10,000 (typical for minimum-finders; solution to machine accuracy is not feasible), that means around 24 to 26 iterations. Not outrageously inefficient (did you catch the fact that `brent.c`, from *Recipes*, took 19 on my sample problem?), but naturally we'd like to do better. A better approach is to use a quadratic method. Such methods involve fitting a quadratic through the three smallest points, and using that to estimate the minimum. A quadratic method doubles the number of significant digits for each iteration. Once it gets near a solution, it homes in like a hound dog after a possum. To the 10^4 accuracy, we found a solution to our sample problem in seven iterations (nine function evaluations, counting setup of the initial conditions).

Each approach has its advantages. The bisection method is the workhorse, the bubble sort of minimum finders. It may be slow, but it's sure. One nice feature of bisection methods is that they don't really care if the function in question is well-behaved or not. For example, the golden section search will cheerfully find the minimum of:

$$f(x) = |x| \quad (1)$$

Such a function will drive a quadratic method crazy, because its derivative is discontinuous. Like most higher-order

ONE GIANT LEAP FOR MANKIND.
(AND MILLIONS OF LITTLE STEPS FOR THE ENGINEERS)



30 years ago the American Space Program landed a man on the moon for the first time, and millions watched it happen live. The Infineon Technologies Space Program isn't quite so glamorous – but we do have our moments. Today, our development tools partners are working together to make the design-in process for our 8- and 16-Bit Microcontrollers, TriCore and DSP products easier for you to work with and use in your next application. For more information on our tools, turn to the next 3 pages, or visit us on our web site:

www.spacetools.com

Stop by the Infineon Technologies Booth (#1020) at the Embedded Systems Show, and visit these 8-, 16- Bit, TriCore and Carmel DSP Tools Partner's Exhibits:

Company	Booth #	Company	Booth #	Company	Booth #
ACE Associate Compiler Experts bv	2614	dSPACE Inc.	6015	Lauterbach, Inc.	345
Advantech Technologies, Inc.	5069	Embedded Power Corporation	128	Logical Systems Corp.	5015
Agilent Technologies	528	Emulation Technology, Inc.	2107	Mentor Graphics	820
Anna Technology	1046	emWare	5046	Nohau Corporation	536
Archimedes Software	2022	Fonix	8095	Phytec America llc.	5091
Ashling Microsystems	7007	Grammar Engine, Inc.	1522	pls Development Tools	7019
Avocet Systems Inc.	116	Green Hills Software	636	Signum Systems	1308
BP Microsystems Inc.	246	HI-TECH Software llc.	145	TASKING, Inc.	222
Byte Craft Ltd.	119	Hitex Development Tools	930	Telenetworks	2814
Cadence Design Systems, Inc.	2701	IAR Systems	344	TQ Components	7019
CAN in Automation	2044	iSystem	2709	US Software	1536
Ceibo, Inc.	121	Ixxat Automation GmbH	8087	Vector Software	5090
CMX Systems, Inc.	2206	Keil Software	1646	Wind River	316, 830

Hands-On Training Classes

Our classes and seminars cover popular topics such as Embedded Technology Updates, Shorten-Time-To-Market, Embedded Networking (CAN, CANopen, DeviceNet), Embedded Internetworking and Microcontroller Architecture training classes.

EMBEDDED SYSTEMS ACADEMY

Consulting

We offer application-specific consulting services. Our expertise covers several microcontroller architectures including their tool chains and many embedded networking protocols, especially CAN (Controller Area Network) and CANopen.

FREE Technical Documents

Our web site offers a wide collection of technical documentation, including example programs and answers to Frequently Asked Questions about microcontrollers and embedded networking.

www.esacademy.com

50 Airport Parkway
San Jose, CA 95110

Tel.(877) 812-6393
Fax.(408) 249-4473

info@esacademy.com

TRACE32®

IN-CIRCUIT-EMULATORS & DEBUGGERS

ONE SYSTEM FITS ALL!



- *Bond-Out Emulators and OCDS Level 1 & 2 Debuggers*
- *Support for Tri-Core, UTAH, EGOLD, C161..C167 and C500 family*
- *Download your free Simulator Debugger for TriCore*

Embedded Systems Conference West
San Jose, CA
Sept. 26-28, 2000

electronica 2000
Munich, Germany
Nov. 21-24, 2000

Lauterbach Datentechnik Gmbh
Fichtenstraße 27
D-85649 Hofolding, Germany
Fax: ++49-8104-8943-49
Phone: ++49-8104-8943-0
e-mail: info@lauterbach.com

Lauterbach, Inc.
4, Mount Royal Ave
Marlborough, MA 01752
Fax: 508-303-6813 · Phone: 508-303-6812
e-mail: info_us@lauterbach.com

LAUTERBACH

http://www.lauterbach.com

Infineon Technologies 8-, 16-, and 32-Bit Hardware & Software Development Tools Support

KEIL SOFTWARE

Development Tools

µVISION 2

C51 C251 C166

IDE, Compiler, Debugger, Real-time OS & more...

Keil Software develops, manufactures, and distributes software development tools for the 8051, 251, and 166 microcontroller families. Development tools include C compilers, assemblers, real-time kernels, debuggers & simulators, integrated environments, and evaluation boards. Our web site provides up-to-date information about our development tools, evaluation software, updates, application notes, and example programs.

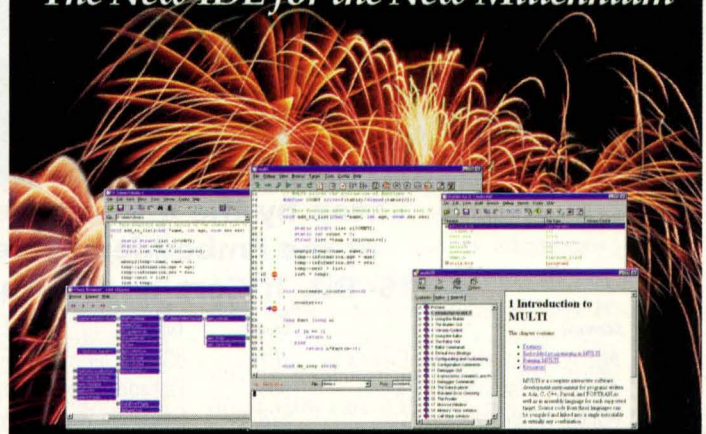
In America:
Keil Software, Inc.
16990 Dallas Parkway, Suite 120
Dallas, TX 75248-1903
✆ 800-348-8051 / 972-735-8052

In Europe and Asia:
Keil Elektronik GmbH
Bretonischer Ring 15
D-85630 Grasbrunn - Munich
✆ ++49 89 456040-0

www.keil.com

MULTI 2000

The New IDE for the New Millennium



The Best Compilers • The Best Debuggers

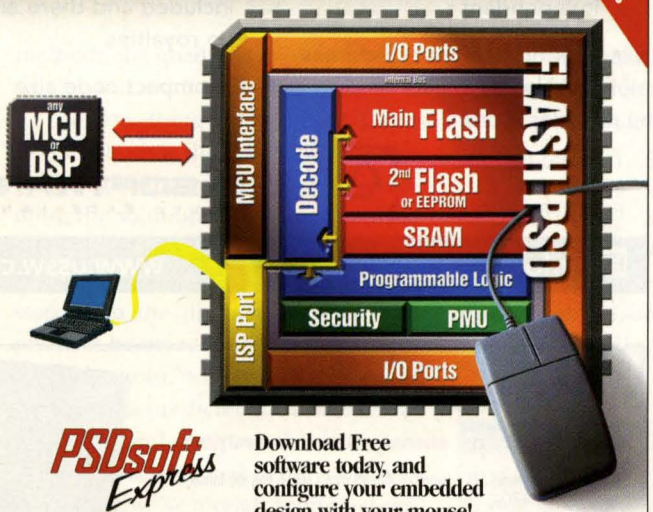
Now, all wrapped up in a brand new, powerful, easy to use IDE. Once you try MULTI 2000, you won't want to struggle with "old fashioned" tools ever again. Contact us now for a Free Evaluation CD and start the millennium with a bang!

Green Hills SOFTWARE, INC.

US Headquarters: 805.965.6044
European Headquarters: +44 (0)1494 429336
Email: sales@ghs.com • www.ghs.com

Flash ISP/IAP

Now for 16-bit MCUs and DSPs



PSDsoft Express

Download Free software today, and configure your embedded design with your mouse!

Waferscale now offers 16-bit versions of our popular *Easy*FLASH™ PSDs. The 16-bit PSD4000 series integrates 4Mb of Flash program store, 256Kb of concurrent Flash, 64Kb of Scratch-Pad SRAM, programmable logic, and advanced in-system programming capabilities onto a single chip. The new devices are configurable with the same easy to use, point and click development/programming software... **PSDsoft Express**. The software guides you through the entire design from configuring the MCU interface and selecting the PSD, to programming the code and firmware into the PSD...All in less than 2 hours! Visit the URL below today to learn more about the PSD4000 series, and to download your FREE copy of PSDsoft Express.

www.waferscale.com/inf/c167.html

wafersale.com

Waferscale-USA
Tel. 800-832-6974
Fax 510-657-5916
info@waferscale.com

Waferscale-EUROPE
Tel. 33-1-69320120
Fax 33-1-69320219
wsifrance@wsiusa.com

Waferscale-ASIA
Tel. 82-2-761-1281
Fax 82-2-761-1283
james@mail.wsiasia.co.kr

State-of-the-Art Emulation



Don´t waste Your time ...

Hitex Development Tools provides proven solutions for most popular microcontrollers, including debugging tools, in-circuit emulators and analyzers for C167/ST10, TriCoreTM, 8051, 251, C500, MSP430, 68HC08/11/12, x86 and Pentium®.

For more detailed information visit us at: **www.hitex.com.**

... with the wrong tools

hitex
DEVELOPMENT TOOLS

Hitex Germany
Tel.: +49-721-9628-0
Fax: +49-721-9628-149
Hitex USA
Tel.: +1-800-45-HITEX
Fax: +1-408-733 6320
Hitex UK
Tel.: +44-24-76692066
Fax: +44-24-76692131
Hitex Asia
Tel.: +65- 566-7919
Fax: +65-563-7539

Infineon Technologies 8-, 16-, and 32-Bit Hardware & Software Development Tools Support

Infineon · PHILIPS · ST · ARM · AMD · MOTOROLA · MICROCHIP · ATMEL · intel · Z · ARC · Triscend · MITSUBISHI ELECTRONIC DEVICE GROUP · NEC · HITACHI

Join the Leading Semiconductor Companies and dozens of their Tools Partners in the Largest Hands-On Embedded Training Events in the World!

More than just a conference or exhibition, the Embedded University is your chance to evaluate dozens of Microcontrollers and their tool chains for leading applications-in actual computer labs.

Select from these times and locations:

Embedded U Germany: September 12-15 *Sindelfingen, Germany*
Embedded U, UK: May 16 & 17 *ESS Show, London*
Embedded U, USA: May 30, 31 and June 1 *Mission Collage, Santa Clara*

Go On-Line Today for the Latest Class Schedules:

www.embeddedU.com

Axel Wolf von Infineon Technologies hält einen CAN Kurs. Dabei benutzt er den C167 CR, Keil-Compiler und den Nohau-Emulator. Eine zweite Labsession zeigt anschließend den Tasking Compiler und den Hitex Emulator.



Ingenieure installieren und testen einen Hiware Compiler in einer USB Labsession von STMicroelectronics.



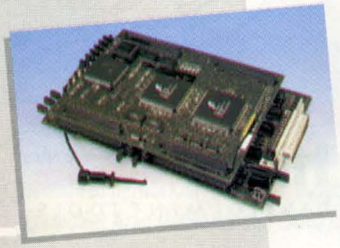
EMBEDDED UNIVERSITY
Hands-On-Training with Embedded Chips and Development Tools

PRIMUS INTER PARES

EMUL166™-PC

In-Circuit Emulator for Siemens C166 & C500 Family

Nohau's EMUL166-PC supports the Infineon Technologies C166 family of microcontrollers, including C167CR, C167SR, C165, C163, C161RI and their associated Xperipherals. The EMUL166-PC emulator system consists of a card which connects to the PC's printer port (LPTx:), and a pod (containing the controller, memory and logic) and a cable connecting the two. The pod consists of two PCB boards with the ability to connect to a third board for tracing capabilities. The optional trace board can record and trigger on internal ROM and external buses. External trigger in/out is also available.



EMUL166-PC Features:

- Emulates at maximum chip speed (33MHz)
- High-level support for all popular C-compilers
- Set hardware and software breakpoints
- 256K to 1M external pod memory available
- Customizable register windows
- Optional trace board

EMUL51™-PC supports the Infineon Technologies C500 family including the C505C, C505CA and C515C.

nohau CORPORATION
51 E. Campbell Ave.
Campbell, CA 95008-2053
TEL: (408) 866-1820
FAX: (408) 378-7869
E-Mail: sales@nohau.com
URL: http://www.nohau.com

USNET® EMBEDDED TCP/IP

- Processor and RTOS independent
- Supports Network Address Translation (NAT)
- ANSI C source code is included and there are no royalties
- Compact code size

TEL: 1-800-356-7097
503-844-6614
FAX: 503-844-6480

U S SOFTWARE
EMBEDDED EXCELLENCE

info@ussw.com www.ussw.com

PHYTEC

Stick it in!

phyCORE Modules
NEW GENERATION SINGLE BOARD COMPUTERS



- Insert-ready sub-mini SBCs (small as 47x55 mm.) supporting the Infineon C161xx & C167Cx
- 32 KB to 2 MB external SRAM & Flash (controller-dependent)
- FlashTools enable on-board in-system (ISP) programming
- RS-232, RS-485, & CAN interfaces; ADC; Chip-Select signals
- Low EMI design achieved via GND circuitry, 6 to 8 layer PCB, by-pass capacitor grid and short signal traces achieved via small fottprint and use of 0402 SMD passive components
- Controller signals extend to high-density Molex (0.625 mm.) header pins on two sides of the board
- Available in Rapid Development Kits including Development Board, AC adapter, serial cable and SPECTRUM CD-ROM with eval software tools (Keil, TASKING), FlashTools, electronic documentation and demos

www.phytec.com (800) 278-9913

Infineon Technologies 8-, 16-, and 32-Bit Hardware & Software Development Tools Support

ASHLING

THE DEVELOPMENT SYSTEMS COMPANY

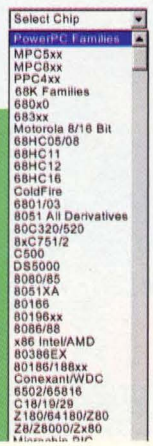
Need Embedded Development Tools for TriCore?

Go to
www.ashling.com/inftools



Power Tools for Embedded Systems Development

QuickTool Finder



C Compilers
Assemblers
Simulators
RTOS
In-Circuit Emulators
BDMs

www.avocetsystems.com

One Click
One Source
One Manufacturer
Every Tool

AVOCET SYSTEMS, INC. (800) 448-8500

The Universal Minimodul



TQM167U

with Infineon technologies C167CR or C167CS

up to 2 MB SRAM
up to 2 MB Flash
EEPROM
2 serial Interface
2 CAN Interface
7-Segment Display
all Microcontroller Pins available
Dimension only **44 x 54 mm²**
Evaluation & Application Board available

The Key To The Embedded World

http://www.tq-group.com
Email info@tq-group.com
US: justist@ixxi.com

TQ components

methods, the quadratic method is more twitchy and more easily confused.

Another thing I like about bisection methods is the criterion we use to halt. Beginning with two points straddling the solution, a bisection method pulls both ends inward. We don't really have to test the solution itself; we stop when the distance between end points is below some epsilon distance.

By contrast, with quadratic methods we get a predicted location for the minimum and compare that with previous values. This makes me a bit nervous, since we've already seen how certain combinations of inputs can yield the same predicted minima, even though neither is anywhere close to the true value.

I also noticed that quadratic methods tend to hang on to end points longer than bisection methods. Bisection methods have no choice; that range is going to be reduced by roughly 25% (a bit more with the golden section) no matter what. Quadratic methods don't really care which points they use, as long as they're roughly centered on the solution.

The thing that attracted me to Brent's method was that it seems to be a near-optimal mix of golden section and quadratic searches. If it can get what it regards as a good quadratic fit, that's what it will use. If not, it rejects the step and uses a golden section instead.

But what makes me nervous about Brent's method—other than that I can't get anyone to explain it to me—is that it will sometimes use quadratics exclusively, if it's satisfied with certain conditions. Because I've seen the quadratic methods hang on to end points too long, I suspect that the interval between the end points doesn't decrease as rapidly as I'd like.

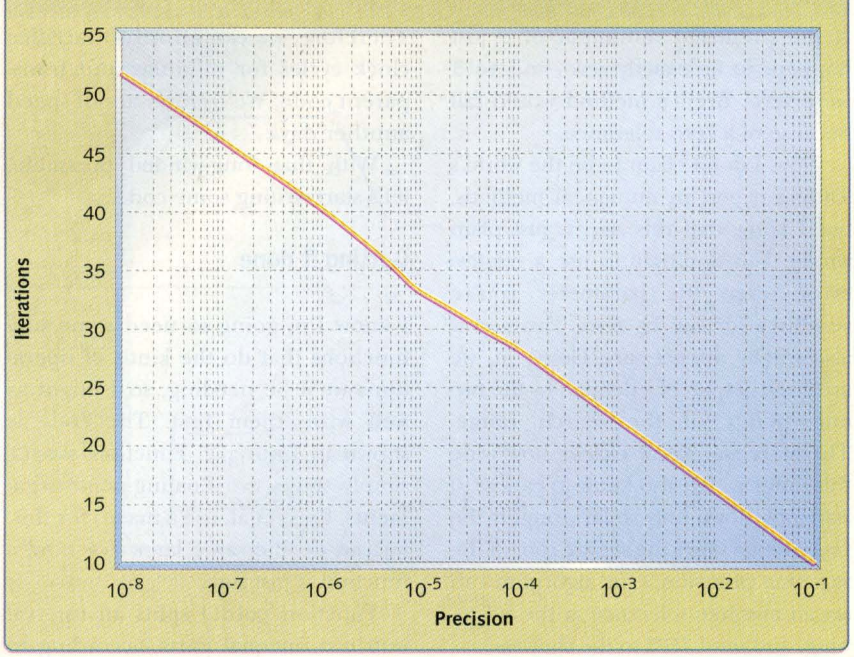
How do we make the two points draw together? I haven't a clue. Short of forcing bisection to bring them together, I'm not sure we can guarantee that the bracketing interval ever gets small. And if it doesn't get small at a rate faster than bisection would produce, we have no hope of speeding the convergence. However, I'd feel more comfortable with an occasional forced bisection step, regardless of how well the quadratic fit is working.

How do we make the two points draw together? I haven't a clue. Short of forcing bisection to bring them together, I'm not sure we can guarantee that the bracketing interval ever gets small.

LISTING 4 A test driver for quadratic search

```
void main(void){
  double x0 = 0;
  double x2 = 1;
  double x1 = gold(x0, x2);
  double y0 = f(x0);
  double y1 = f(x1);
  double y2 = f(x2);
  gold_shrink(x0, x1, x2, y0, y1, y2);
  for(int i=0; i<12; i++){
    para_shrink(x0, x1, x2, y0, y1, y2);
    cout << x0 << ' ' << x1 << ' ' << x2 << endl;
  }
  cout << k << endl;
}
```

FIGURE 1 Golden section, iteration vs. accuracy



The bottom line is this: we want a method that combines the nicest features of both bisection and quadratic fit. Brent's method is such a method, but certain details, such as the halt criterion and the possibility of using quadratics exclusively, are cause for concern.

Picking nits

To summarize, the things that bother me about Brent's method are:

- It can and will finish exclusively with quadratic fits
- We cannot guarantee that the bracketing interval is small

In addition to these concerns, I've found two other things to complain about. In both the C and Fortran versions of Brent's method, I see that a small epsilon distance is used—`ZEPS` in *Recipes*, `EPS` in `fmin.f`. (Mittlemann and Spelluci use a machine-language function to find the machine epsilon. I just stuck a reasonable number in its place.) This parameter is later used to set a value for `tol3`. Brent's method needs this epsilon value because it's looking for a certain maximum *relative* (not absolute) error in x. See the line:

```
tol1=eps*dabs(x)+tol3
```

If the predicted minimum of x just happens to be exactly zero, and `tol3` were zero, Brent's method would fail because of a zero tolerance.

Now, I don't claim to be the world's leading expert in numerical methods, but I know that this step is just plain wrong. You don't try to use a relative error when the parameter you're searching for can be zero. The step is also wholly unnecessary. Going in, we are given a span of x defined by the two end points of the search range. Therefore we know exactly how wide this range is. We also know to expect to narrow it down only so far: roughly the input range times the square root of the machine precision. Our algorithm can set an absolute tolerance at the beginning, and send `ZEPS` to the showers.

The bottom line is this: we want a method that combines the nicest features of both bisection and quadratic fit. Brent's method is such a method, but certain details, such as the halt criterion and the possibility of using quadratics exclusively, are cause for concern. Instead of trying to psychoanalyze Brent and his algorithm, let's simply write one that uses the same notions, but does things our way. If it turns out looking very much like Brent's method, we should not be too surprised. I suspect, however, that it will also display significant differences, one of which I just mentioned.

One last thought before we dive in: it occurrs to me that one reason algorithms like `brent.c`, `fmin.f`, and in fact most of the algorithms in *Recipes* seem so inscrutable is that their authors wanted them to be time-efficient. At this point, I don't care about that. I'd like to have numerical efficiency, but I don't care about counting clock cycles. Therefore, we can afford to sacrifice clock cycles for modular and transparent code. We'll worry about speed another day.

With that long-winded preamble, let's start writing some code.

Getting it done

I know I'm going to need some subfunctions that do the kinds of operations we'll be needing, so I might as well write them first. The code is shown in Listing 2. Function `swap()` simply swaps two floating-point arguments. Press et al. use a macro for this; we can do the same later. For now, a function is just fine.

Function `gold()` splits an interval into two unequal parts, according to the golden section. The function uses a new constant, `golden`, which I've added to my standard file `constant.cpp`:

```
double golden =
  sqrt(5.0)+1.0)/2.0;
```

This is the famous golden ratio, Φ. Numerically, it's approximately 1.618. Subtracting it from 2 gives 0.382, which is the value Press et al. call `CGOLD`, and Mittlemann and Spelucci call `'c'`. Don't be thrown by the fact that they describe it as "the squared inverse of the golden ratio." Φ shows funny behavior when squared or inverted.

The function `gold_shrink()` in Listing 2 takes one step of the golden section search. By that, I mean it begins with three points p_0, p_1, and p_2, and subdivides the interval on the "left," $p_0..p_1$. It then replaces either p_0 or p_1 and p_2, and sets up for the next iteration. Part of this setup is to exchange p_0 and p_2, so we shouldn't make any assumption that, say, $x_0 < x_2$. What seems to be the "left" side of the interval can really be left or right, on alternate intervals.

As long as we're this close, we might as well write a full-blown golden section search routine. That's the one shown in Listing 2 as `gold_search()`. As you can see, function `gold_shrink()` does almost all of the work, so `gold_search()` only needs to set things up properly, and decide when to quit. Note my use of an absolute error as a halt criterion.

Figure 1 shows the performance of the algorithm. Note that it took 28 function evaluations for a 10^4 improvement in accuracy, just about exactly what I predicted (this count includes setup evaluations of $f(x)$). The slope is also about six iterations per decade of improvement, again exactly what we might expect. Though this result was obtained for a specific problem, you can take the slope value to the bank.

On to parabolas

Now that we remember how to do the golden section search, let's see what we can do with parabolic fits. In my column "A Simulator's Paradise," (April 2000, p. 11), I worked out the formulas for fitting a parabola through three points, and estimating the minimum. The math goes:

$$x_{min} = \frac{1}{2}\left(x_1 + x_2 - \frac{m_0}{c}\right) \quad (2)$$

where:

$$c = \frac{m_1 - m_0}{x_2 - x_1} \quad (3)$$

and m_0 and m_1 are the two slopes:

$$m_2 = \frac{y_1 - y_0}{x_1 - x_0} \quad (4)$$

$$m_1 = \frac{y_2 - y_1}{x_2 - x_1} \quad (5)$$

Note again that the algorithm doesn't really care whether the xs increase to the left or the right, so anything we do in terms of swapping ends will not affect the result.

Using Equations 2 through 5, we can write a function that estimates the minimum by means of parabolic fit. This is shown as `para_fit( )` in Listing 3.

Looking at this code, I think, "Gee, it sure doesn't bear much resemblance to Brent's `p`, `q`, `r`, `x`, `w`, `v`, `d`, and `e`, does it?" Never mind. I'm sure the two parabolic fits are equivalent—it's pretty hard to get that part wrong—but I'm not currently inclined to prove the equivalence. I understand what I've got, and that's enough for me.

Once we have an estimated x_{min} from the parabolic fit, we still have to decide what to do with it. Clearly, if it's outside the interval $x_0..x_2$, it's no good and we must discard it. However, this should never happen. Our fundamental assumption is that:

$$y_0 > y_1 < y_2 \quad (6)$$

We assume that this is true going into the algorithm, and we take steps to

I'm sure the two parabolic fits are equivalent—it's pretty hard to get that part wrong—but I'm not currently inclined to prove the equivalence. I understand what I've got, and that's enough for me.

FIGURE 2 Quadratic convergence

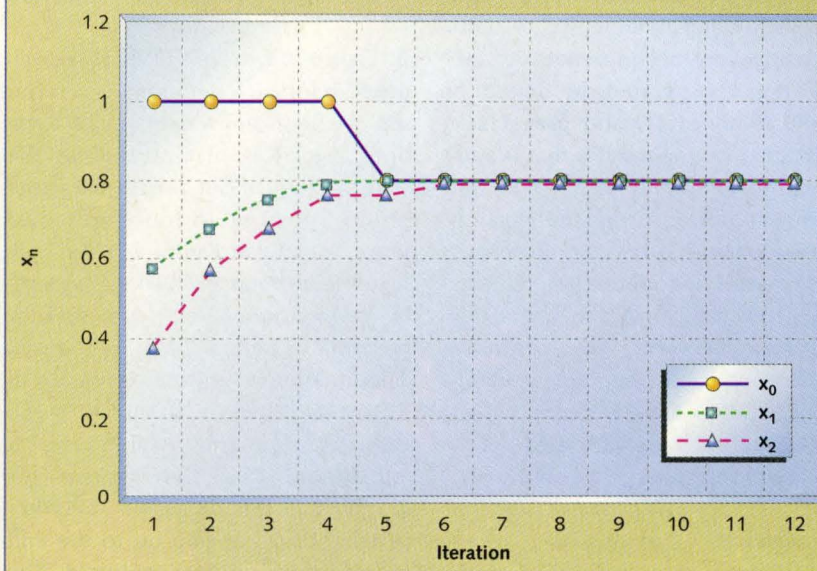
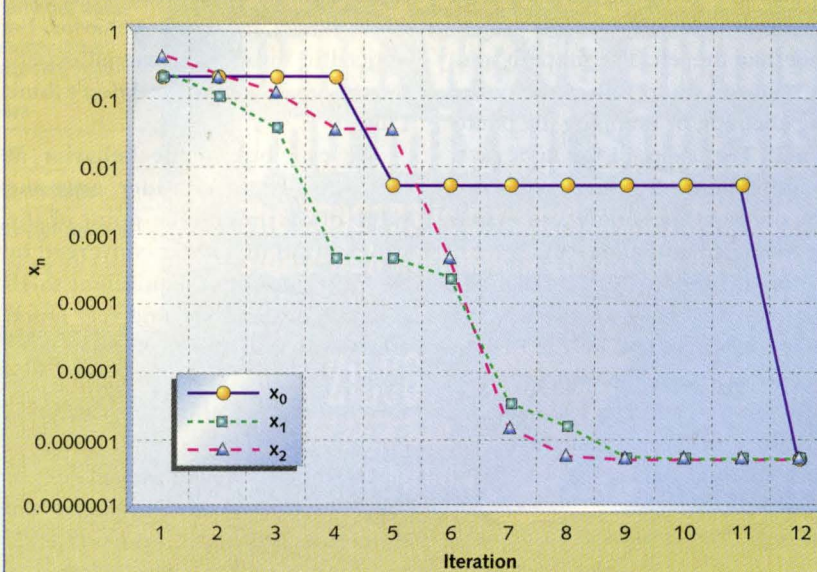


FIGURE 3 Homing in



I'm willing to accept more golden section steps, even if it means more function evaluations, if that's what it takes to force the end points of the range to pull together in a smoother fashion.

keep it so at each stage. As long as Equation 5 is satisfied, x_{min} has no choice but to lie between x_0 and x_2.

Next we should decide how to adjust the points and the interval they represent for the subsequent step. When you think about it, the situation is identical to that presented by the new point generated by function `gold()`. Both `gold()` and `para_fit()` generate a new value of x that is supposed to be an improvement over the previous one. The decision-making process doesn't care what logic was used to come up with this new point, it only needs to deal with it. There is a difference, however. `gold_shrink()` always knows that the value coming out of `gold()` will be between x_0 and x_1. This isn't the case with `para_fit()`, so additional logic is required. Otherwise, we can write a function completely analogous to `gold_shrink()`. I've given it the incredibly imaginative name `para_shrink()`.

In writing `para_shrink()`, I tinkered with alternate ways to deal with the situation in which the output of `para_fit()` lies to the "right" of x_1, rather than the left. The simplest solution seems to be to reflect the entire thing about p_1, by swapping the points around. The swap of p_0 and p_2 is straightforward. To make the rest come out right, we should also swap p (the estimated minimum) and p_1.

I haven't shown the equivalent of `gold_search()` because, unlike the golden section search, we can't count on a parabolic search to work without a little help. Nevertheless, I've written a main program to test the method. That code is shown in Listing 4.

How well does it work?

The results of a run on our test case are shown in Figures 2 and 3. The test function, by the way, is the one we've been using all along, which is:

$$f(x) = \cos(2\pi x^2) \tag{7}$$

In Figure 2, I've shown the three values of x, for each iteration. As you can see, we have converged in 12 iterations, which is nice. Including the setup and initial call to `gold_shrink()`, we do a total of 16 function evaluations. Looking at Figure 2, it certainly seems that things are working smoothly, and we are essentially converged by the seventh iteration. But look again. The situation is shown more clearly in Figure 3, where I've included the error from the known solution, on a logarithmic scale. Here we can see graphically, the quadratic method's penchant for hanging on to the end points far longer than it should. The value of x_0 seems to stick at one for four steps. Not visible in Figure 2, but plain in Figure 3, is the fact that it sticks again for a full seven steps, right after that. This is the behavior I'd noticed in my earlier, manual experiments with quadratic fits. I don't think I like it.

We can look at the behavior in two ways. If we consider only the value of x_1, the middle point of the trio, we find that is has converged by the ninth iteration, equivalent to 13 function evaluations, and very much consistent with the performance of `fmin.f` for the same problem. But the interval's failure to shrink is bothersome.

On the other hand, recall that in the golden section method, it's the difference between x_0 and x_2 that we use to determine when we've converged, and the best estimate of the minimum is the average of the two. We can't use either of these constructs for the quadratic case, which I suppose is why Brent's method uses a rather tortuous criterion for convergence. I see nothing in Brent's method to prevent this tendency to hang onto wide intervals, and that bothers me.

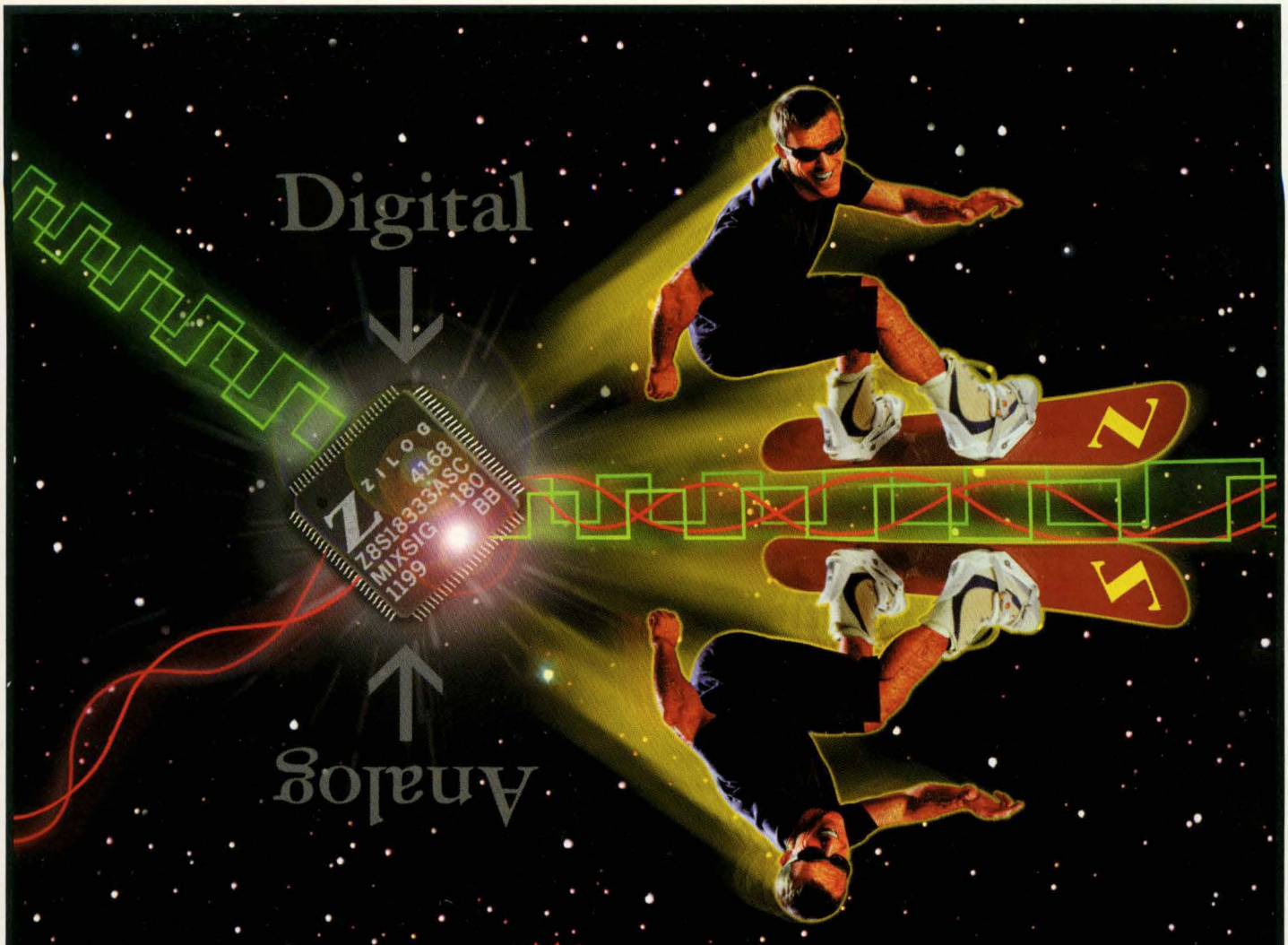
My position on the matter is simple: I want that interval between x_0 and x_2 to shrink in a smooth and systematic way, and I'm not going to be satisfied until it does. I think we can make it do just that by forcing an occasional golden section step. Press et al. imply that if the function is well-behaved, Brent's method will drive directly to the minimum using only quadratic steps. Indeed, it does exactly that, which is precisely why it makes me uncomfortable. I'm willing to accept more golden section steps, even if it means more function evaluations, if that's what it takes to force the end points of the range to pull together in a smoother fashion.

I'd like to pursue these issues further this month, but I'm out of both time and space. I think you'll agree, though, that we've made quite a dent in the problem. Next month, we'll consider ways to force occasional golden section steps. We'll also address all the other issues, such as alternate end criteria. See you then. **esp**

Jack W. Crenshaw is a senior principal design engineer at Alliant Tech Systems Inc. in Clearwater, FL. He did much early work in the space program and has developed numerous analysis and real-time programs. He holds a PhD in physics from Auburn University. Crenshaw enjoys contact and can be reached via e-mail at jcrens@earthlink.net.

Resources

1. Press, William H. et al. *Numerical Recipes in C*. Cambridge: (Cambridge Univ. Press, 1992)
2. Brent, *Algorithms for Minimization Without Derivatives*. Englewood Cliffs, NJ: Prentice-Hall, 1973.



"Now available with an Embedded Web Server Software Suite for Internet applications."

Z80S183 Features

• 8-ch 10-bit 500Ksps ADC	• 2K SRAM
• 10-bit 1 us DAC	• 6 low power modes
• 100% Z80/180 code compatibility	• 1K boot ROM w/virtual loader
• 1 Megabyte address space	• 8-ch Programmable Output Generator / improved PWM
• Real Time Clock / Calendar w/alarm	• Watch Dog Timer
• New ZDI Debug Interface	• Power On Reset
• 2 Enhanced UARTs	• Low Power PLL oscillator
• 2 High speed DMA controllers	• Economical 100-VQFP package
• 2 16-bit Counter-Timers	• DC to 33 Mhz @ 5 V +/- 10%
• 32-bits General Purpose I/O	• DC to 20 Mhz @ 3.3 V +/- 10%

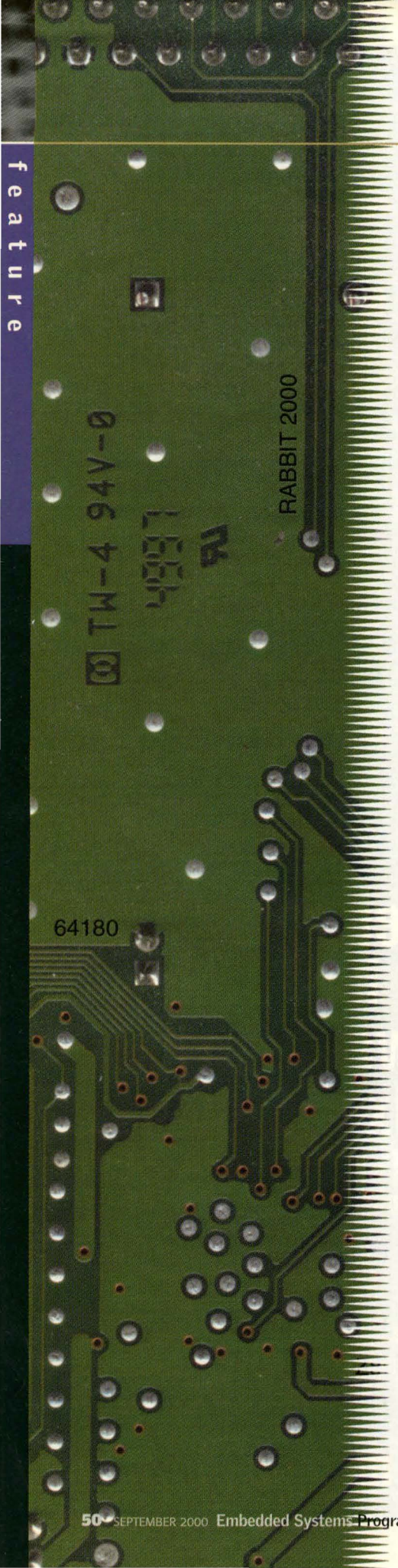
Copyright© 2000 ZiLOG, Inc. All Rights Reserved.

To speed your design integration time, we're giving you mixed signals.

The Z80S183 is a mixed signal Z80 with a super-rich peripheral set all on one chip.

ZiLOG extreme connectivity©

www.zilog.com/Z183



JEAN J. LABROSSE

New Twists on the Z80

Despite being a quarter-century old, the 8-bit Z80 and its derivatives continue to be popular with embedded system designers. With two new derivatives emerging within the last year, it's a good time to take a fresh look at the architecture. This article considers the architectural choices from the perspective of an RTOS implementer or user.

Introduced by ZiLOG in the early '70s, the Z80 was an impressive processor for its time. The original Z80 (a 2MHz part) was a souped-up version of the then-popular Intel 8080 processor. While compatible with the 8080, the Z80 offered many new features: new instructions, new addressing modes, two index registers, faster execution, high performance peripheral chips, and a clever interrupt scheme. Soon, Z80s were available in higher speed grades (4MHz and 6MHz). For many years, the Z80 was the king of 8-bit processors.

In 1976, ZiLOG introduced the Z280, which was an enhanced Z80, but it turned out to be a big failure.

Around 1985, Hitachi introduced the 64180 microprocessor, which was an improved Z80. ZiLOG second-sourced the processor under the Z180 name. This 64-pin DIP processor executed most instructions with fewer clock cycles than the Z80 and contained a number of on-chip peripherals such as a pair of UARTS, a clocked serial I/O port, an on-chip interrupt controller, two DMA channels, two 16-bit timers, and an on-chip memory management unit allowing the '180 class processor to extend the address space from 64K to 1,024K. Although called an MMU, this device only implemented memory banking (described later).

In the mid '90s, ZiLOG announced the Z380 but, like the Z280, it never really went anywhere. Today, a number of companies are selling Z80-derived processors, including AB-Semicon, Kawasaki, LSI, NEC, Toshiba, and VAutomation.

Towards the end of last year, ZiLOG announced a new generation of

The Z80 architecture is simple to learn and still quite powerful. And, like any popular platform, there is a lot of valuable legacy software floating around for it.

Z80 derivatives. Around that same time, a new company called Rabbit Semiconductor introduced its own new generation of Z80 derivatives. ZiLOG's offering is called the eZ80 and is expected to hit the street in the second half of 2000. Rabbit Semiconductor's new processor is called the Rabbit 2000; it started shipping in November 1999. Rabbit Semiconductor was founded by its parent company Z World Inc. Z World has been around since 1981 and has been using the Z80 and Z180/64180 extensively in its board-level controllers.

Annual volumes of Z80 flavors are reported to be at around 200 million units. I asked both ZiLOG and Rabbit why they would introduce new products based on a 25-year-old architecture. I got the same response from both: a lot of engineers (about 100,000), have used these chips at some point in their career, and are already familiar with the architecture. The Z80 architecture is simple to learn and still quite powerful. And, like any popular platform, there is a lot of valuable legacy software floating around for it. Finally, it's quite difficult and expensive to introduce completely new architectures. Both ZiLOG and Rabbit claim that their new processors rival 16-bit processors in performance.

The original

The register model of the Z80 is shown in Figure 1. The Z80 can only address 64K of memory but has a separate 64K I/O address space. The Z80 has two sets of 8-bit registers but only one is visible at any given time. Most 8-bit operations are performed on the A register. The F register contains the six CPU flags (**S**ign, **Z**ero, **H**alf carry, **P**arity/o**V**erflow, **N**egative, and **C**arry). The flags are automatically changed based on the result of the current operation. Registers B, C, D, E, H, and L are general-purpose 8-bit registers that can be used during 8-bit operations. These last registers can be combined in pairs to form 16-bit registers: BC, DE, and HL. The BC, DE, and HL registers can be used in 16-bit arithmetic operations or can be used as pointers.

The I register is used to hold the upper 8-bit address of the interrupt vector table. When an interrupt is generated, the CPU uses the I register and an 8-bit value provided by the interrupting device to form a 16-bit interrupt vector address. Determining the source of an interrupting device and vectoring to the appropriate interrupt service routine (ISR) is thus handled in hardware.

The R register is incremented on each instruction and is used to provide a refresh address for dynamic memory during a refresh cycle that is buried in each instruction cycle. However, it is unlikely that anyone would use dynamic memory nowadays because static memory is more appropriate for a Z80 design. Both the IX and IY registers are 16-bit pointers. Finally, SP is the stack pointer and PC is the program counter. On average, a Z80 processor executes an instruction every 10 to 12 clock cycles. Newer Z80 derivative processors cut this to two to five cycles per instruction, depending on the architecture.

Real-time kernels

A real-time kernel is software that manages the use of a microprocessor or microcontroller to ensure that all time-critical events are processed as efficiently as possible. A real-time kernel can help simplify your design because it allows your project to be divided into multiple independent elements called tasks. A task is a program that competes for CPU time and is generally written as an infinite loop as shown in Listing 1.

LISTING 1 Task code

```
void TaskCode (void)
{
 while (1) {
  Do something useful;
  Wait for event or time to expire;
 }
}
```

LISTING 2 Letting the kernel manage your tasks

```
void main (void)
{
   .
   .
  OSTaskCreate(Task1Code, TopOfStack1, Priority1);
  OSTaskCreate(Task2Code, TopOfStack2, Priority2);
   .
   .
}
```

With most real-time kernels, each task is given a priority based on its importance and its own stack space. You allow the kernel to manage each task by calling a function provided by the kernel, as shown in Listing 2.

When you design a product using a real-time kernel you split the work to be done into tasks, each responsible for a portion of the problem solution. A real-time kernel also provides valuable services to your application such as time delays, system time, message

HANNAH BLAIR

passing, synchronization, mutual-exclusion, and more.

Most commercial real-time kernels are preemptive. A preemptive kernel ensures that the highest-priority task that is ready to run is given control of the CPU. When an ISR makes a higher priority task ready to run, the higher priority task will be given control of the CPU as soon as all nested interrupts complete. The execution profile of a system designed using a preemptive kernel is illustrated in Figure 2. As shown, a low priority task is executing (1). An asynchronous event interrupts the microprocessor and the processor vectors to an ISR (2). The microprocessor services the event (3) and calls a service, provided by the kernel, that causes a higher priority task to become ready for execution. Most likely, the higher priority task was waiting for the ISR to occur. Upon completion of the ISR, the kernel is invoked once more. This time, the kernel notices that a higher priority task is now ready to run and thus, the processor cannot return to the interrupted task. Instead, the kernel will resume the high priority task (4). The higher priority task executes (5) until it again needs to be signaled by another occurrence of the ISR or another task. A kernel service is again invoked to have the high priority task wait for an event. If the event did not occur, the kernel resumes execution of the lower priority task (6).

FIGURE 1 Z80 registers

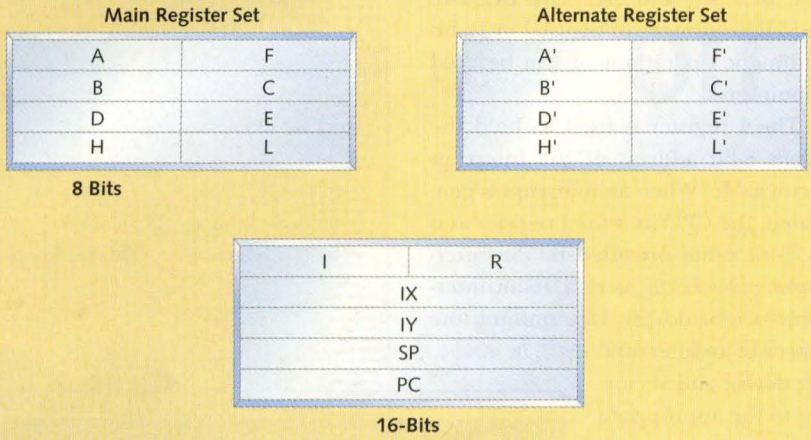


FIGURE 2 Execution profile of a preemptive kernel

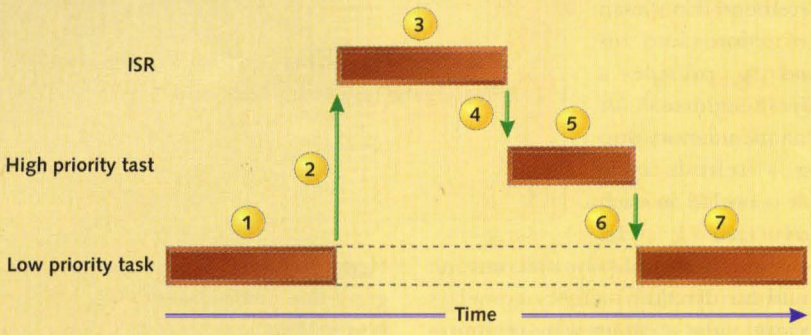
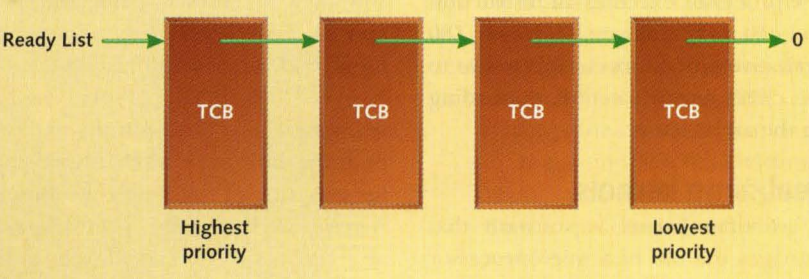


FIGURE 3 List of tasks waiting to be executed



A kernel thus ensures that time critical tasks are performed first. Furthermore, execution of time-critical tasks are deterministic and are almost insensitive to code changes. In fact, you can often add and change low priority tasks without affecting the responsiveness of your system to high priority tasks.

In order to manage tasks, the kernel needs to maintain internal data structures. One such data structure is called the task control block (TCB). Each task is assigned a TCB by the kernel. A TCB contains the "state" of a task (ready or waiting), the current location of the task's top-of-stack (TOS), priority, and other kernel-specific elements.

Because each task is assigned a priority, the kernel must also keep track of tasks in order of priority. The kernel maintains what's called a Ready List. The ready list tells the kernel which tasks are able to execute. The kernel always chooses the highest priority task that is ready to run. Only one task can execute on the processor at any given time, and thus, the other tasks that are ready will have to wait until

Shorten your day.



Haven't you got better things to do than integrate someone else's microprocessor core? Thought so.

Wouldn't you appreciate a 32-bit microprocessor you can configure yourself, so it integrates into your system like a dream? Thought so again.

But then you usually wake up.

Not this time. When you configure the ARC, you automatically generate the tool chain, the interfaces and code that allows the ARC core to integrate smoothly and seamlessly into your system-on-chip. Leaving you time to design new features not glue logic.

ARC also provides the ARChitect configuration and customization tool, as well as full hardware source code, so future changes will be a breeze – not a breakdown.

The ARC user-configurable processor makes it easier and quicker, from beginning to end. Which is why our customers – and your competitors – are getting to market faster.

Download a free demonstration of the ARC processor configuration tool at www.arccores.com

It won't just make your day shorter. It'll make your day.

microprocessors by

ARC

Start thinking.

NORTH AMERICA: ARC Cores Inc, 6862 Santa Teresa Blvd, San Jose, CA 95119 Tel: 408 360 2120 Fax: 408 360 2139
EUROPE: ARC Cores Ltd, ARC House, Waterfront Business Park, Elstree Road, Elstree, Herts, WD6 3BS, UK Tel: +44 (0) 20 8236 2800 Fax: +44 (0) 20 8236 2801
www.arccores.com info@arccores.com

FIGURE 4 Tasks waiting on a semaphore

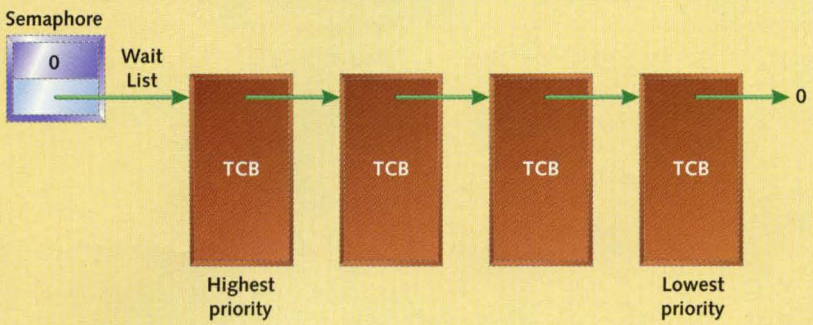
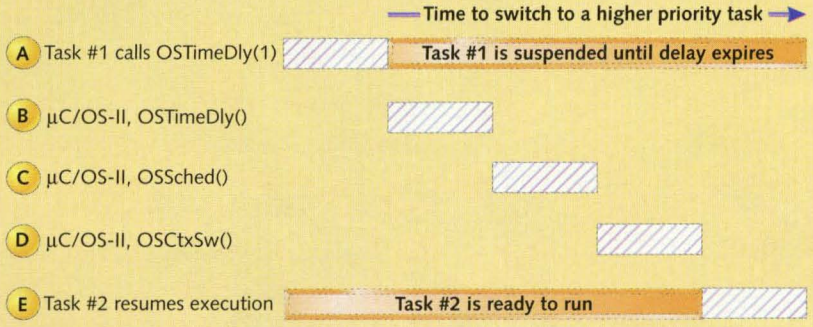


FIGURE 5 A task suspending itself



the current (running) task is no longer able to run. Figure 3 shows how TCBs can be organized in the ready list. The singly linked list is only shown to demonstrate the concept. Kernel designers have devised more efficient ways to maintain a ready list.

The kernel also maintains a number of Wait Lists. A task can wait on kernel objects such as a semaphore, mailbox, message queue, or pipe. Each of these objects contains state information. For example, a semaphore contains a value indicating whether the semaphore is available or not, as well as a list of tasks waiting on the semaphore, provided the semaphore is currently owned by a task. Figure 4 shows a list of tasks waiting on a semaphore. Again, the singly linked list is shown mainly to illustrate the concept.

Figure 5 shows the execution profile of a service, `OSTimeDly()`, provided by an kernel that I developed a few years ago, called µC/OS-II. In the example, Task 1 is executing and calls `OSTimeDly()` to suspend execution of Task 1 for one system tick. A system tick is generally created by a timer chip that interrupts the CPU and occurs at a fixed interval (10ms to 100ms, depending on the application). A task calling `OSTimeDly()` is placed on a list of tasks waiting for time to expire. `OSTimeDly()` calls `OSSched()`, which is another kernel function that finds the next most important task that is ready to run. Once found, `OSSched()` calls `OSCtxSw()`, which performs a context switch to this task. What's important to note here is that these functions take time to execute, so the processor should execute these as quickly as possible.

The kernel performs a context switch when it determines that the current task is no longer the most important task to execute. A processor's context generally consists of the volatile state of a processor. For the Z80, the context consists of the following registers: AF, BC, DE, HL, AF′, BC′, DE′, HL′, IX, IY, SP, and PC. In other words, if you want to stop execution of one task and resume execution of another task, you need to save the context of the first task onto its stack and restore the context of the task you wish to execute from its stack.

A context switch for the Z80 is shown in Figure 6. It is assumed that the kernel maintains pointers to the TCB of the running task (task to suspend) and the new task (task to resume). A context switch starts by saving the processor's context onto the running task's stack (1). The processor's SP register (new TOS) is then saved in the TCB of the task to suspend (2). The task's TOS is retrieved from the new task's TCB and placed in the processor's SP register (3). Finally, the processor registers are restored by popping them from the stack (4). At this point, the new task code continues execution as if it had never been suspended.

Listing 3 shows the pseudo-code for the above operation. Note that it's assumed that the PC register is already on the TOS of the task to suspend. A context switch takes about 426 clock cycles on a Z80 (17µs at 25MHz).

Kernel requirements

Kernels can be designed to work with just about any processor. In fact, a kernel requires very few features from a processor. However, to run a preemptive kernel, a processor must be:

- Able to disable and enable interrupts
- Able to access a large stack area
- Able to load and store the processor's stack pointer
- Able to save and restore the processor's context onto/from the stack
- Able to easily manipulate pointers

The Z80 and its derivatives support all of these requirements.



Forward-thinking

companies
don't start over

they start
smarter with

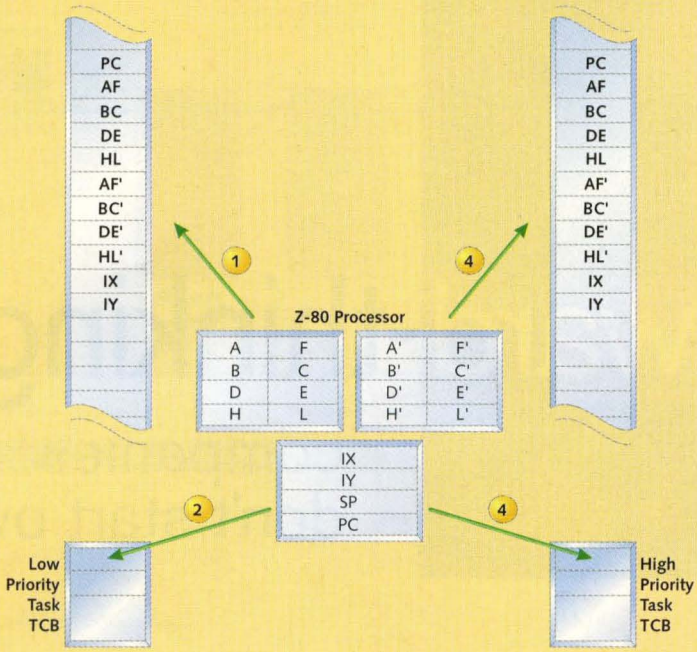
design reuse.

Recreating designs that have been done already isn't very creative. And when it comes to designing a complex chip, it's certainly not practical. That's why it makes sense to leverage high quality IP blocks through design reuse whenever possible. Synopsys can help you to do just that with our best practice methodologies, industry-leading tools and partnerships, and high-quality commodity IP library. Our design reuse solutions can help your team get more accomplished and move designs into production faster than with any other approach. So talk to us about how to start smarter on your next big project. When it comes to design reuse, Synopsys has everything you need to leverage the best of what's gone before.

www.synopsys.com/design_reuse

SYNOPSYS®

FIGURE 6 Context switch on a Z80 processor



The kernel disables interrupts to protect critical sections of code. Thus, the processor must thus have instructions to disable and enable interrupts. The Z80 provides two instructions: `DI` and `EI`, respectively. What if the programmer had interrupts disabled before calling a function provided by the kernel? Unfortunately, the Z80 doesn't provide an easy way to preserve the state of the interrupt disable flag and restore it upon completion of the critical section. Because of this, interrupts on the Z80 are always enabled after leaving a critical section. Fortunately, most of the time, this is the way you would want it. The Rabbit 2000 (described later) is the only Z80 derivative that can preserve the state of the interrupt disable flag.

The Z80 provides a number of registers that can be used to indirectly address memory, making it a natural for pointer manipulation.

A kernel needs RAM to store data structures and to maintain a stack for each stack. On a Z80, a kernel such as µC/OS-II requires as little as 1K of RAM for internal structures (task stacks excluded). Where code space (ROM) is concerned, a minimal configuration for µC/OS-II would require about 3K. Other operating systems may require additional RAM and/or ROM.

Hitachi 64180

As previously mentioned, the Hitachi 64180 processor was introduced in the mid '80s. Hitachi has stopped promoting the 64180 in the U.S. and seems to be concentrating on the H8 and SH processor families. ZiLOG has been pushing many Z180 derivative products, all based on Hitachi's original design, having different complements of on-chip peripherals.

Apart from its on-chip peripherals, a 180 class processor adds value by integrating an on-chip MMU. Again, although called an MMU, the 180-class processors only implement memory banking. Figure 6 shows how the MMU translates a 64KB logical address space into a 1024KB physical memory space. A '180 class processor can only address 64KB of memory at any given time. The 64KB logical address space is split into three areas: Common Area 0, Bank Area, and Common Area 1. The size of each of these can be adjusted with a granularity of 4KB. The MMU is configured through three 8-bit I/O ports, which Hitachi calls registers—CBAR, BBR, and CBR.

Common Area 0 is designed for ISRs and other code that cannot be banked, such as kernel services. This region always starts at address 0x0000 in the logical address space and always translates to physical address 0x00000.

The starting address, and thus the size, of the other areas is established by the common/bank area register, or CBAR. The upper nibble (four bits) of the CBAR sets the starting address in the logical address space of Common Area 1. In other words, if the upper nibble of the CBAR is set to 0x8 then Common Area 1 starts at 0x8000. The lower nibble of the CBAR sets the starting address of the Bank Area. For example, if the lower nibble of CBAR is set to 0x4 then the Bank Area starts at 0x4000. To achieve the layout of the logical address space shown in Figure 7, the CBAR would be initialized to 0x84 in the startup code and would not be changed at run time.

Common Area 1 is generally used to hold data (RAM). The common base register, or CBR establishes the starting address of Common Area 1 in the physical address space. The mapping of logical address to physical address for Common Area 1 is given by:

```
PhysicalAddress = (CBR << 12) +
LogicalAddress;
```

Your application can change the value of the CBR at run time and thus point to a different physical block of memory in the physical address space. However, the CBR is generally set once at startup.

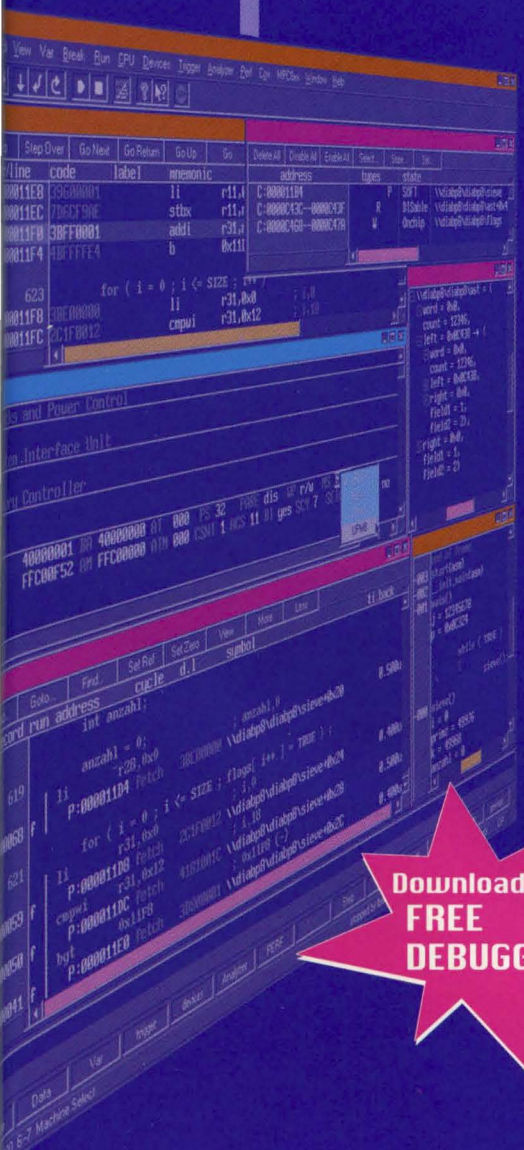
Finally, we get to the bank base register, or BBR. Like the CBR, this register allows your application to view a "window" of physical memory (could

TRACE32®

IN-CIRCUIT-EMULATORS & DEBUGGERS

ONE IDE SERVES ALL!

TRACE32-PowerView



TRACE32-ICD
the highly cost effective
In-Circuit Debugger

TRACE32-FIRE
the Fully Integrated Risc
Emulator at highest speeds

TEST THE BEST!

- *Professional tools exceed highest demands*
- *The ultimate in technology and speed*
- *Greatest variety of supported CPUs (ARM, C167, H8S, PowerPC, SH2, ST10 and more)*
- *Unique modularity and open concept*
- *Test and compare our comfortable tools by downloading our simulators*

Embedded Systems
Conference
San Jose, CA
Sept. 26-28, 2000

Download
FREE
DEBUGGER!





Lauterbach, Inc.
4, Mount Royal Ave
Marlborough, MA 01752
Fax: 508-303-6813
Phone: 508-303-6812
e-mail: info_us@lauterbach.com

Lauterbach, Inc.
13256 SW. Hillshire Drive
USA-Tigard, OR 97223
Fax: 503-524-2223
Phone: 503-524-2222
e-mail: info_us@lauterbach.com

LAUTERBACH

Free simulators available in the Internet:

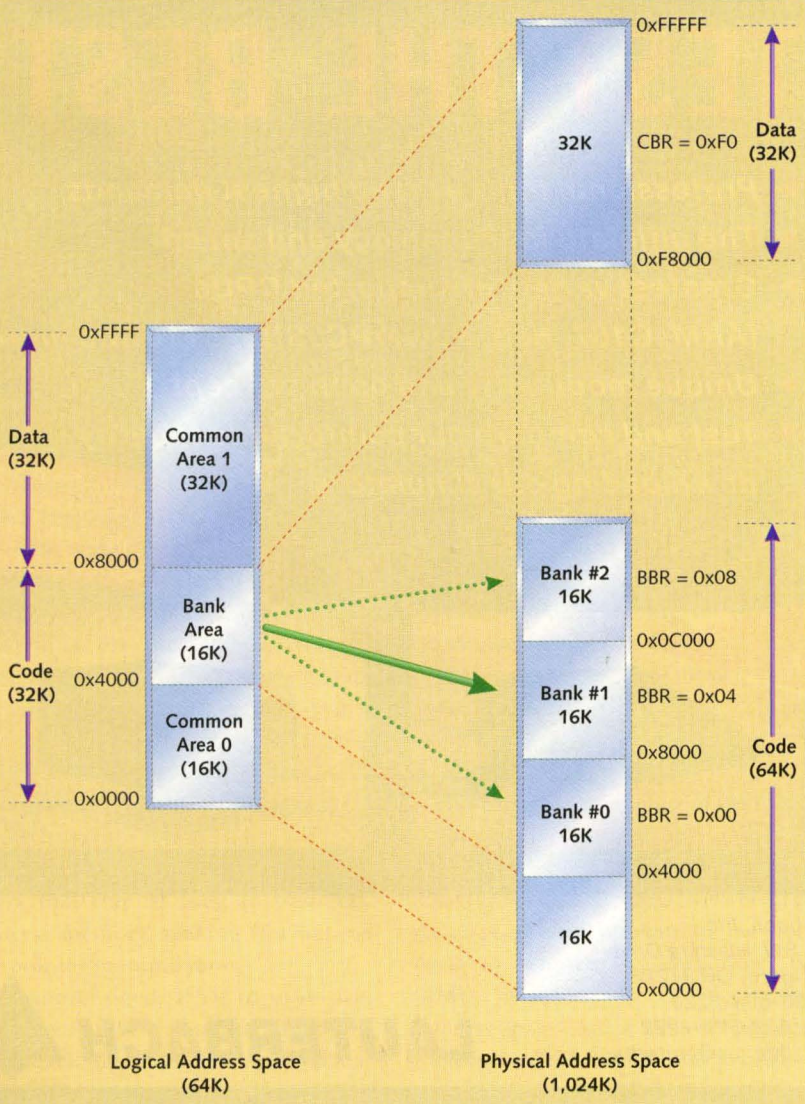
http://www.lauterbach.com/download.html

LISTING 3 Context switch pseudocode for a Z80

```
Z80_Context_Switch:
; Save the current task's context
  PUSH main register set
  PUSH secondary register
  Save SP into the TCB of the task to suspend;

; Restore the new task's context
  Load SP from the TCB of the task to resume;
  POP secondary register set
  POP main register
  RET
```

FIGURE 7 Logical to physical memory translation of the 180-class processor's MMU



be code or data). The BBR can be changed at run time and thus allows you to view different chunks of code or data. The mapping of logical address to physical address for the Bank Area is given by:

```
PhysicalAddress = (BBR << 12) +
LogicalAddress;
```

As shown in Figure 7, logical address 0x4000 would map to physical address 0x0C000 if the BBR contains 0x08. You should note that steps in value of the BBR are determined by the bank size. If you setup the Bank Area to have 16KB, values of the BBR are in steps of 0x04 since each increment represents 4KB.

The C compiler I've used sets the BBR to the proper value if you compile your code with the banking option. From a kernel's perspective, the kernel needs to save the BBR in addition to the other Z80 registers during a context switch. This means that the context switch code for the 180-class processors would look as shown in Listing 4. A context switch for µC/OS-II on the 180-class processors takes about 400 clock cycles (16µs at 25MHz).

Rabbit 2000

Last fall, Rabbit Semiconductor introduced the Rabbit 2000 processor. From its name, it's not obvious that this is a Z80/180 derivative processor, but it is. Like the 180-class processors, the Rabbit can address up to 1MB of code and data. Rabbit Semiconductor added new instructions that substantially improve 16-bit operations, allow alternate registers to be referenced explicitly, allow an offset to be added to the SP and HL registers (making stack manipulation much more efficient), allow the interrupt disable flag to be saved and restored to/from the stack, and more. In short, it corrects a number of deficiencies in the Z80/180.

The Rabbit goes through code mostly at two clock cycles per byte with

Your proud partner in that amazing new thing we're not supposed to talk about yet.

The most exciting new products are the ones you can't tell anyone about yet. Products that can give their creators a huge market advantage. Intuitive, adaptive, "smart" products that'll change the way people live, work and play. Made possible with DigitalDNA™ from Motorola. DigitalDNA technology is chips, systems, software and ideas — embedded solutions that are helping smart companies create smart products. Does your company have an idea for one? We can keep a secret.

Contact us at www.digitaldna.com.

DigitalDNA™
from Motorola

THE HEART OF SMART™

©2000 Motorola, Inc. Motorola, DigitalDNA, The Heart of Smart, the DigitalDNA Logo and the Stylized M Logo are trademarks or registered trademarks of Motorola, Inc.

LISTING 4 Context switch pseudocode for a 180-class processor, in banking mode

```
H64180_Context_Switch:
; Save the current task's context
  PUSH main register set
  PUSH secondary register
  Get BBR and PUSH
  Save SP into the TCB of the task to suspend;

; Restore the new task's context
  Load SP from the TCB of the task to resume;
  POP and set BBR
  POP secondary register set
  POP main register
  RET
```

very few extra cycles needed. The exception is the 16-bit multiply instruction, which requires 12 cycles. This is much better than the original Z80, which requires a minimum of three or four cycles per byte with many instructions requiring multiple extra cycles. Rabbit Semiconductor advertises the Rabbit 2000 to be four times faster than a Z80 when executing compiled C code on systems that have equal memory access times. This is a due to a combination of improvements:

- Every instruction executes in fewer clock cycles
- New instructions improve the quality of compiler generated code
- Memory interface improvements permit higher clock speeds for the same memory access time

Finally, the Rabbit 2000 contains a number of I/O devices on chip: 40 parallel I/O ports, four asynchronous serial ports, numerous timers, a watchdog timer, and interfaces for memory chips.

The register model for the Rabbit is shown in Figure 8 and is nearly identical to the Z80 except for the addition of the following registers (shown in bold in Figure 8): XPC, IP, STACKSEG, DATASEG, SEGSIZE.

The EIR is the same as the I register on the Z80 and thus contains the upper byte of an interrupt vector table. The IIR replaces the R register on the Z80. However, the IIR is used to point to an interrupt vector table specific to internally generated interrupts.

The IP register is the interrupt priority register. It contains four 2-bit fields that hold a history of the processor's interrupt priority. In fact, the Rabbit supports four levels of processor priority. The Z80 and 180-class processors have only two: maskable and non-maskable. What I find particularly useful is that you can save and restore the current interrupt priority level on the stack. This makes it a natural for critical sections as shown in Listing 5.

Memory management on the Rabbit is similar to the 180-class processors as shown in Figure 9, but it's more flexible. There are four registers in the Rabbit MMU: SEGSIZE, DATASEG, STACKSEG, and XPC. In Figure 9, I assumed a product might have 128KB of ROM and 64KB of RAM. The SEGSIZE register is used to establish the boundary of the Root Segment, Data Segment, and Stack Segment. I decided to have 24KB of code that is non-bankable and a data segment of 32KB. The Rabbit makes it easy to have multiple stacks since the stack segment can be placed on any 4KB page boundary beyond physical address 0x0D000. With the memory configuration shown, we could have 64 stacks each having 512 bytes and thus have each stack segment hold the stack of eight tasks. Of course, we could split up the stack differently between tasks because some tasks may require more or less RAM than others.

The XPC is similar to the BBR register of the 180-class processors but sets the bank size to 8KB. The 8KB area act as a page rather than as a bank. This scheme is similar to the 80x86 real mode except that the page is 8KB instead of 64KB and the amount that the page can slide is 4KB instead of the 80x86's 16 bytes. There are no limits regarding the 8KB size of the page because the compiler generates code to change the XPC register when the code passes the 4KB mark (that is, halfway through the page). This is done with a "long jump" instruction that changes the XPC register and the PC. The net effect is that there are no gaps in the utilization of memory as is the case of banked schemes when functions do not quite fill up a bank. The entire process is handled by the compiler and is quite transparent to the programmer. The only caveat is that no single statement can generate more than 4KB of code—an unlikely possibility. With the configuration shown in Figure 9, I could have 24KB of root code and 104KB of extended code for a total of 128KB of code. The root code is shown in the diagram as taking 24KB. In fact, the root code region can be made smaller or larger. Root code enjoys a slight advantage over extended code in that subroutine linkage is a little faster, a factor that is most important for very short subroutines. In a typical Rabbit application, most of the code would reside in extended code space.



Capture the technologies that you want.

On the platform that you need.

In a timeframe you never thought possible.

Design the product of your dreams.

When you start with phoenix™ technologies anything is possible.

Visit Phoenix, booth #142 at the Embedded Systems Conference, San Jose, CA Sept. 24 - 28, 2000.

FIGURE 8 Rabbit 2000 register model

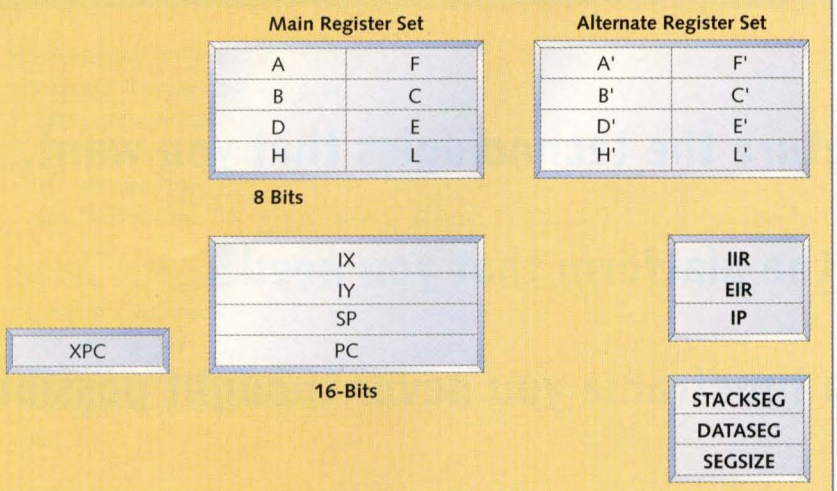
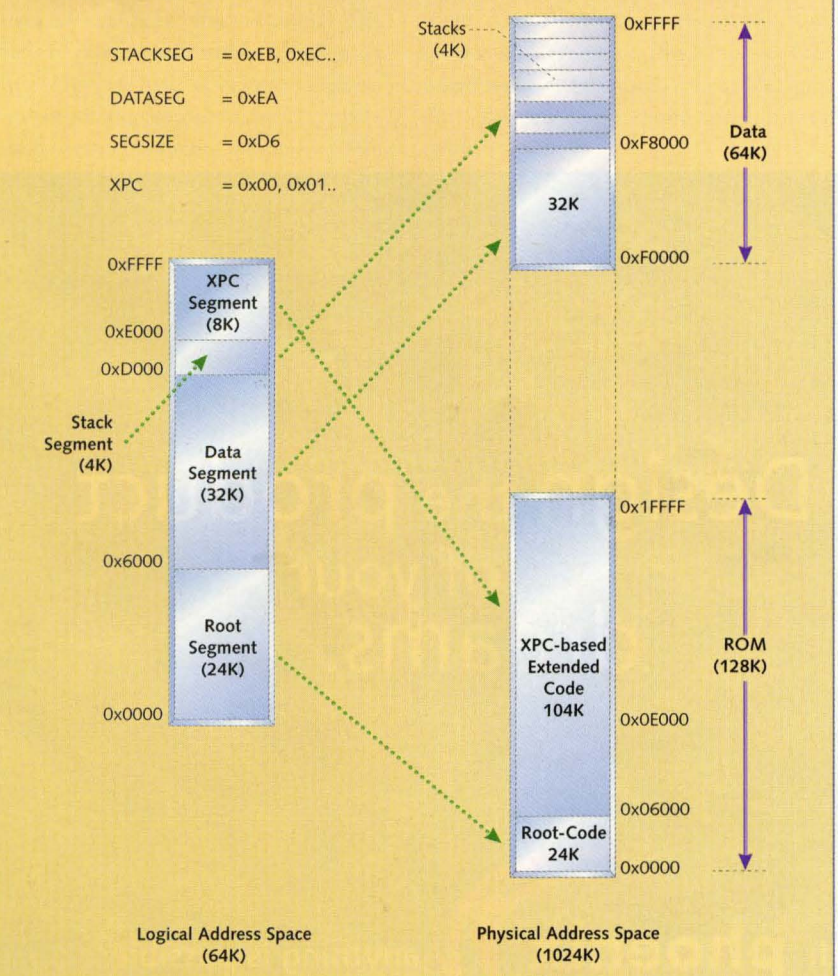


FIGURE 9 Logical to physical memory translation of the Rabbit 2000 MMU



LISTING 5 Handling of a critical section on the Rabbit

```
; Enter critical section:
  PUSH IP
  IP  1
;
; Critical section
;
; Exit critical section:
  POP IP
```

The code for mapping of logical address to physical address for the stack segment is:

```
PhysicalAddress = (STACKSEG << 12)
+ LogicalAddress;
```

This equation is valid only when the logical address is between 0xD000 and 0xDFFF. The STACKSEG register is most likely maintained by the kernel and not the compiler. The code for mapping of logical address to physical address for the XPC segment is:

```
PhysicalAddress = (XPC << 12) +
LogicalAddress;
```

Extended code is visible in the 64KB logical memory space in an 8KB window at the top of memory from 0xE000 to 0xFFFF. The value of the XPC register is maintained by the compiler.

A context switch on the Rabbit is similar to the Z80 and 180-class processors, as shown in Listing 6. You will notice that the other two MMU registers (SEGSIZE and DATASEG) are not saved during a context switch because, most likely, they will not be changed dynamically. A context switch for µC/OS-II on the Rabbit takes about 330 clock cycles (13µs at 25MHz).

ZiLOG eZ80

Late last year ZiLOG announced their fourth-generation Z80 derivative. ZiLOG's marketing literature indicates that eZ80s will be available

The RTOS for Internet Appliances, Systems-On-A-Chip, and all other Embedded Systems



EmProve
Virtual Application Development and Prototyping (VAD-P™) environment

RTXC™
Internet Connected Multitasking Real-Time OS

BeaconSuite™
- Compilers
- Linkers & Locators
- Debuggers
- Simulators

Q.E.D.™
- JTAG Debugger
- 186 /386 ICE

The shortest distance between two points includes the ground we've already covered for you.

We have covered that ground with the most useful, flexible, scalable RTOS you can imagine for DSP, 32-, 16-, and 8-bit embedded applications.

It's what we do for a living - building it right so that you don't have to. You *could* write your own d.o.s. (dumb old scheduler), and *then* try your hand at integrating some file, graphics, and network components.

But why?

As a proven OS, **RTXC** has a pre-emptive scheduled multitasking real-time kernel with an extensive set of services and subsystems built in to assure top performance.

For example, **RTXCnet** includes a full array of networking components - from TCP/IP or PPP to web browser/server. All RTXC components are integrated to kernel resources and architected to fully utilize pre-emptive multitasking. We blow the competition's polling-based architectures away - and so will you! Just pick the networking components you need and RTXCnet arbitrates synchronization and interactions between your application, network stack and kernel.

RTXCfile32, our MS-DOS compatible file management subsystem can be used on any embedded system that uses RTXC - without regard to the target processor. We have drivers for CD ROM, floppy disks, hard disks, PCMCIA memory cards, and FLASH or RAM disks. Our drivers can be tailored to meet your most demanding file-oriented storage device needs.

Our **RTXCgraphics** subsystem endows your development effort with an existing portable graphic user interface library. Video device support is superlative, and fully supports the Rapid Application Development Standard.

Supported Processors

AMD: x86xx Family, ELAN 3/4/520C, K6
ARM: 6, 7/7T, 710, 9T, StrongArm
Hitachi: H8/300 Family
IBM: PPC 4/6
Infineon: 80C16x Family
Intel: 80x86/386/486 Real Mode, 80x86/386/486/Pentium Family Protected Mode
Mitsubishi: M16C
Motorola: DSP, StarCore, M.CORE, MPC500, ColdFire 52xx/53xx, 68K Family, CPU 32 Family, 68HC16 Family, 68HC12 Family, and 68HC11 Family
Philips: 80C51XA
ST Microelectronic: ST10
Texas Instruments: 320C3x/5x/20x/54x

www.embeddedpower.com

Embedded Power CORPORATION

U.S. 281-561-9990
email: info@embeddedpower.com
Europe: +44 (0) 1256 474448
email: eurosales@embeddedpower.com
© 2000 Embedded Power Corporation
Trademarks are the property of their respective holders.

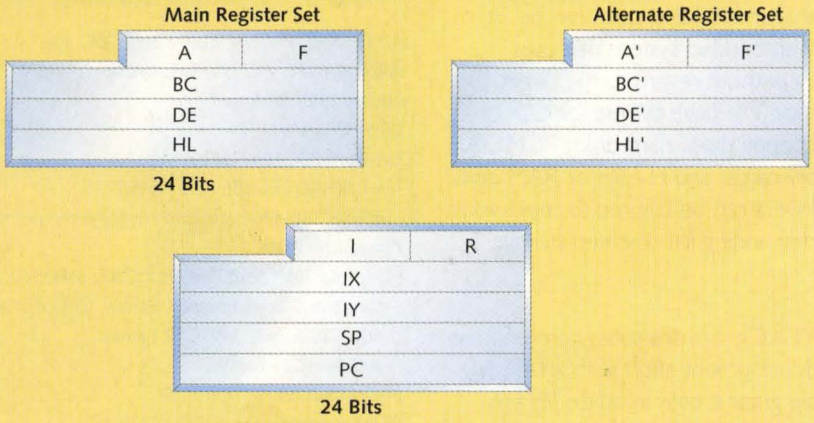
The eZ80 has a more convenient addressing scheme than either the 180-class processors or the Rabbit 2000 and offers a larger linear address space (16MB vs. 1MB).

LISTING 6 Context switch on the Rabbit 2000

```
Rabbit_Context_Switch:
; Save the current task's context
  PUSH main register set
  PUSH secondary register
  Get XPC    and PUSH
  Get STACKSEG and PUSH
  Save SP into the TCB of the task to suspend;

; Restore the new task's context
  Load SP from the TCB of the task to resume;
  POP and set STACKSEG
  POP and set XPC
  POP secondary register set
  POP main register
  RET
```

FIGURE 10 eZ80 register model in ADL mode



at speeds of up to 80MHz. ZiLOG claims that the eZ80 will execute code four times faster than a standard Z80 operating at the same clock speed. The eZ80 offers binary compatibility modes with the Z80 and the Z180. ZiLOG added a few new instructions to the architecture but nothing spectacular.

Some eZ80 derivative processors will be available with a 40-bit multiply-and-accumulate (MAC) engine. Information on the MAC is very preliminary. The MAC will act as a coprocessor and will be accessed using block I/O operations. eZ80 chips are not currently available but should be by the time you read this article.

The eZ80 has an interesting mode called the Address and Data Long (ADL) mode. In this mode, most of the eZ80 registers are 24 bits, as shown in Figure 10. The eZ80 is thus able to address up to 16MB of memory linearly without resorting to banking schemes. In addition, the address space can contain either code or data. The eZ80 only has an 8-bit wide data bus. Thus, obtaining a 24-bit address will require slightly more execution time than the other Z80 derivatives. I believe that this is a small price to pay to get the convenience that the eZ80 gives you.

As you would expect, the 24-bit registers can also be used to contain data. This allows you to perform certain 24-bit arithmetic operations. Another nice feature of the eZ80 is its inclusion of the ZiLOG Debug Interface. This is a two-pin interface that allows debugging without requiring an in-circuit emulator.

Listing 7 shows the pseudo-code for the context switch of the eZ80. I excluded saving the MAC registers for sake of comparison. You may notice that a context switch is identical to the Z80. However, in ADL mode, the 24-bit registers will be appropriately pushed and popped. Note that it's assumed that the 24-bit PC register is already on the top of the stack of the task to suspend. I was able to get an advanced copy of the list of instruction execution times for the eZ80, enabling me to determine how long a context switch takes to execute. To save and restore all the registers, I came up with a context switch time of 163 clock cycles (about 6.5µs at 25MHz).

Head to head

The eZ80 has a more convenient addressing scheme than either the 180-class processors or the Rabbit 2000 and offers a larger linear address space (16MB vs. 1MB). From a performance standpoint, one could make a case that the 180-class processors and the Rabbit would be faster (at the same clock speed) than the eZ80 because you generally only need to load the memory bank once and use a 16-bit address to access a number of

• Best Practices • Software Development • Configuration Management • System Definition • System Testing • Best Practices •

Software Development • Configuration Management • System Definition • System Testing • Best Practices • Software Development

Configuration Management • System Definition • System Testing • Best Practices • Software Development • Configuration Management • System Definition • System Testing



Rational ClearCase. Change. Automated.

Simplify change management and make time for the things that really matter. When you spend all day keeping track of changes, you lose time for the important things. Like writing code and updating Web content. Rational ClearCase® can help by automating many of the time consuming tasks associated with developing software and building e-business applications, such as parallel development and change management. It even manages code and content so you're always sure the correct version of your Web site is posted. Plus, it integrates with other Rational analysis, design, development and testing tools to provide the industry's only complete development lifecycle solution. No wonder IDC ranked Rational ClearCase 1st in Software Configuration Management Market Leadership* for the 4th year in a row. Visit **www.rational.com/change3 for your free IDC report.**

Rational®
the e-development company™

© 2000 Rational Software Corporation. Rational, Rational logo, Rational the e-development company and Rational ClearCase are trademarks or registered trademarks of Rational Software Corporation.
*Application Life-Cycle Management Forecast and Analysis 2000-2004

LISTING 7 Context switch on the eZ80

```
eZ80_Context_Switch:
; Save the current task's context
  PUSH main register set
  PUSH secondary register set
  Save SP into the TCB of the task to suspend;

; Restore the new task's context
  Load SP from the TCB of the task to resume;
  POP secondary register set
  POP main register set
  RET
```

adjacent memory locations. Furthermore, the compiler takes care of most of the memory banking work for you. However, from a programmer's standpoint, I sure like the linear addressing scheme of the eZ80.

I like the Rabbit's multiple interrupt priority scheme and especially the fact that you can save and restore the state of the interrupt level on the stack. This means that if you have interrupts disabled before making a call to the kernel, interrupts would be disabled upon return from the kernel. None of the other processors provide this feature.

Table 1 shows the execution time (in clock cycles) for some instructions that a kernel would use as well as the execution time for a context switch (assuming µC/OS-II) for each processor. The 180-class processor is faster than the Z80 by about 5%. The Rabbit is about 25% faster than a Z80. The eZ80 is more than twice as fast as the Z80 and assumes ADL mode. However, the execution times for the eZ80 resulted from preliminary data that is subject to change.

A context switch is certainly not representative of the true performance of a processor and should thus not be considered, by any means, to be an extensive benchmark. A context switch only measures how fast registers can be saved and restored to/from the stack.

It remains to be seen whether ZiLOG will deliver on its promise of an 80MHz part. If they do, one has to wonder about the memory speed required to feed this device. Some of the eZ80 instructions are single cycle. This means that at 80MHz, the eZ80 would read or write from/to memory in less than 12.5ns (in fact, much less if you consider propagation delays, setup, and hold times). It's my understanding that the fastest flash chips have a 35ns access time. This would mean that, at 80MHz, you would need to add wait-states to stretch memory accesses by at least four clock cycles. This would defeat the whole idea of getting a fast processor. If you include op-code fetches, it's easy to see how a 5-byte instruction could take at least 20 clock cycles (not five). To obtain single-cycle byte reads and writes on commercially available 35ns flash chips, I believe that ZiLOG only needs to come out with 20MHz parts (at best). Of course, your design could boot from slower flash but run out of fast, but expensive, static RAM if you need to get every ounce of performance out of the eZ80.

The Z80 has always been a decent processor to use with a kernel. It offers great addressing modes, plenty of registers and a good set of instructions. We seem to always ask more of the processors we choose and hopefully, these new generations of Z80 processors will help quench this need. **esp**

Jean Labrosse is a senior technical staff member at Dynalco Controls in Fort Lauderdale, FL. Dynalco is a manufacturer of electronics equipment that serves the oil and gas industry. He has a master's degree in electrical engineering and has been designing embedded systems for many years. Labrosse is the author of two books published by R&D Technical Books: MicroC/OS-II, The Real-Time Kernel *and* Embedded Systems Building Blocks, Complete and Ready-to-Use Modules in C. *Contact him at JLabrosse@Dynalco.com.*

TABLE 1 Head-to-head chip comparison

Instruction	OpCode	Z80	64180/Z180	Rabbit 2000	eZ80
Enable interrupts	EI	4	3	2	1
Disable interrupts	DI	4	3	2	1
Multiply	MLT	N/A	17	12	5
Load memory pointed to by IX with contents of register	LD (IX + d), r	19	14	10	4
8-bit register move	LD r,s	4	4	2	1
Load A with memory pointed to by HL	LD A, (HL)	7	6	5	2
Load SP with contents of HL	LD SP, HL	6	4	2	1
Push 16-bit register on stack	PUSH dd	11	11	10	4
Pop 16-bit register from stack	POP dd	10	9	7	4
Increment memory pointed to by IX	INC (IX + d)	23	18	12	6
Call function	CALL nn	17	16	12	7
Jump	JP nn	10	9	7	6
Return from function	RET	10	9	8	6
Return from interrupt	RETI	14	13	12	7
µC/OS-II context switch	N/A	426	400	330	163

PLANET MICROCHIP



12-bit ADCs: Less than $2.40. Op Amps: Below 40¢. System Supervisors: About 39¢. Complete Design Solutions: Priceless.

At Microchip, we understand the value of a complete signal chain solution. That's why we've launched analog ICs for both the analog *and* digital designer. No other company gives you the flexibility of choice: world-class 12-bit ADCs stand-alone or embedded in the popular 8-bit PICmicro® MCU. *FREE* tools such as the innovative FilterLab™ software help ease the job of filter design. Add rail-to-rail Op Amps and more than a hundred System Supervisor Options – now you've got end-to-end signal chain support.

12-bit A/D Converters	Sample Rate	Channels	Interface	INL	DNL	THD (typ)	Package	Price
MCP320X Family	100 ksps	1, 2, 4, 8	SPI	±1 LSB	±1 LSB	–80 db	PDIP SOIC TSSOP	From $2.37**
Op Amps	**Channels**	**GBWP**	**I_Q (typ)**	**V_{OS} (max)**	**Op Voltage**	**Output**	**Package**	**Price**
MCP60X Family	1,2, 4, 1+ reset	2.8 MHz	230µA	±2mV	2.7V – 5.5V	Rail-to-Rail	PDIP SOIC TSSOP	From $0.39**
Precision System Supervisors	**V_{CC} Range**	**Reset Voltages**	**Reset**	**Output**	**T_{RPU} (typ)**	**I_{DD} (typ)**	**Package**	**Price**
MCP1XX/8XX Families	1.0V – 5.5V	2.7V – 4.85V	Active Low Active High	Push-Pull Open Drain	350ms	45µA	TO–92 SOT–23–3 8 SOIC	From $0.39**

All products specified –40° to +85° C

** USD 1000s, FOB USA

All this backed by the same great technical service you've come to expect from Microchip. Visit our website or call to request the new Analog Design Pack CD-ROM featuring FilterLab, and put our specs to the test. Analog devices from Microchip. One more reason to explore the Universe of Embedded Control.

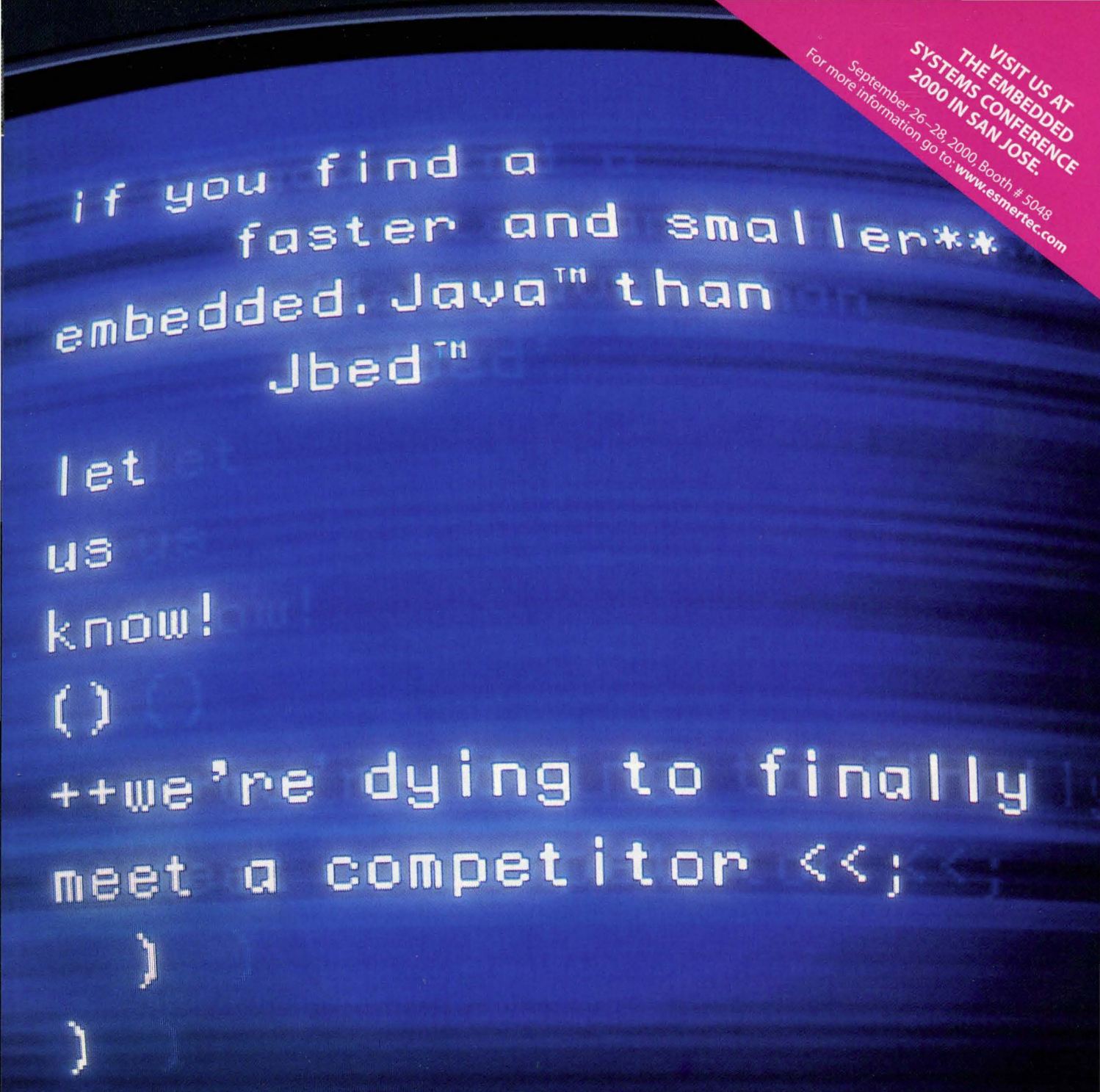
1-888-MCU-MCHP 1-888-628-6247

microchip.com

Explore the Universe of Embedded Control

MICROCHIP In Control™

The Microchip name and logo, PIC, PICmicro, and The Embedded Control Solutions Company are registered trademarks and Microchip In Control and FilterLab are trademarks of Microchip Technology Inc. in the USA and other countries. SPI is a trademark of Motorola. © 1999 Microchip Technology Inc. All rights reserved. All other trademarks are the property of their respective owners.



VISIT US AT THE EMBEDDED SYSTEMS CONFERENCE 2000 IN SAN JOSE.

September 26–28, 2000 Booth # 5048

For more information go to: www.esmertec.com

Then again, we are very pleased to see that Jbed™ is alone at the top. After all there's a good reason for this: the Jbed Combined RTOS/Java™ Virtual Machine eliminates your concerns about the size, performance and cost of running a JVM in an embedded, let alone real-time, demanding system. Why? Firstly, Jbed IDE allows you to prototype your tasks, so you know you can meet your scheduling deadlines and unused processor load percentage. Secondly, you can download new or modified Java classes to your fielded, running systems without needing to reset, while the Jbed Admission-Facility continues to ensure that your deadlines are met. And last, but certainly not least, Java byte-code is complied on your PC or on your running target by our Target Bytecode Compiler into optimised machine code – never interpreted.

By the way: if you are concerned about running hand-held applications on the KVM™, use our new Jbed Micro Edition CLDC™. **info@esmertec.com, www.esmertec.com**

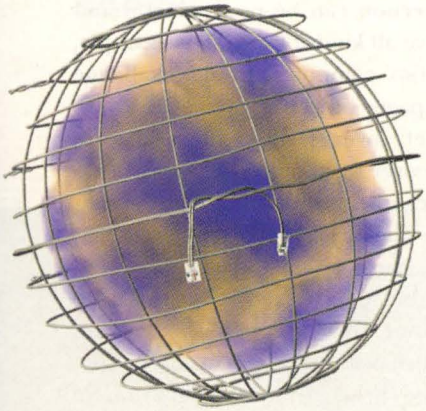
esmertec

your partner for embedded real-time Java™

Jbed is a trademark of esmertec, inc. - Java and all Java-based trademarks are registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. All other trademarks are the property of their respective owners.



Michael Barr



Internet Working

Internet Appliance Design

in this issue

78 Open Source Point/Counterpoint: Are Open Source and Innovation Compatible? Open Source is Already Delivering

95 Ready or Not: IPv6 is coming

109 Embedded Internet Tools

The Internet Protocol is just that: the protocol upon which the Internet is built. It is also the IP of TCP/IP and UDP/IP fame. Several types of network interfaces may lie below the IP layer (serial, Ethernet, ATM, and so on) and a variety of upper-layer protocols may run on top of it (TCP, UDP, and so on). IP is, therefore, the one constant, the glue that holds the network together. This month's column is about the IP layer and my implementation of it. But first, I need to make a quick correction and give my stack a name.

An outdated copy of the source code for one address resolution protocol (ARP) function managed to slip through the cracks and into my column,"Mid-Year's Resolutions," (July 2000, p. 43). The misprinted function was named `NetArpAddEntry()` and shown in that month's Listing 5. The call to `NetArpInit(n)` shown there should have been to `NetArpFlush()`. I had later changed the name of the cache initialization function, for clarity's sake. It seems to make more sense to flush a cache more than once than to re-initialize it. The parameter `n` passed to `NetArpInit()` should not have appeared in the magazine either. That was something I had added for my own testing of the stack. To allow two UDP/IP stacks to share the same Win32 process space it was necessary that they each have their own ARP cache.

I've decided to name the open source UDP/IP stack I've been developing µC/Net. This name sounds nice =for two reasons. First, with this stack I'm trying to bring internet connectivity to low and mid-range microcontrollers. Also, the name µ/Net works well alongside the µC/OS-II operating system, to which I'm currently porting the stack.

IP freely

As we've seen, the presence of a network interface like an Ethernet controller in a product makes it possible to send packets of data from one computer to another. To send the data, however, a destination hardware address must be attached to the packet before it is streamed out onto the network. ARP provides a mapping service between the hardware address of the destination computer and its IP address. So despite the fact that they are not part of the IP layer, both a network interface and ARP are required for network communications.

The IP layer's main role is to provide for the routing of packets from one IP address to another. In other words, it can connect any machine on the Internet with any other, regardless of physical location. One of the communicating machines may be connected to an Ethernet LAN at a company in China, while the other could be a laptop in Maine that has just connected with a modem and a PPP link. Because of IP, their physical locations and underlying network interfaces don't matter. The IP layer software on each machine and similar software within routers and switches on the Internet backbone make this end-to-end communication possible.

LISTING 1 Data structures and constants for the IP layer

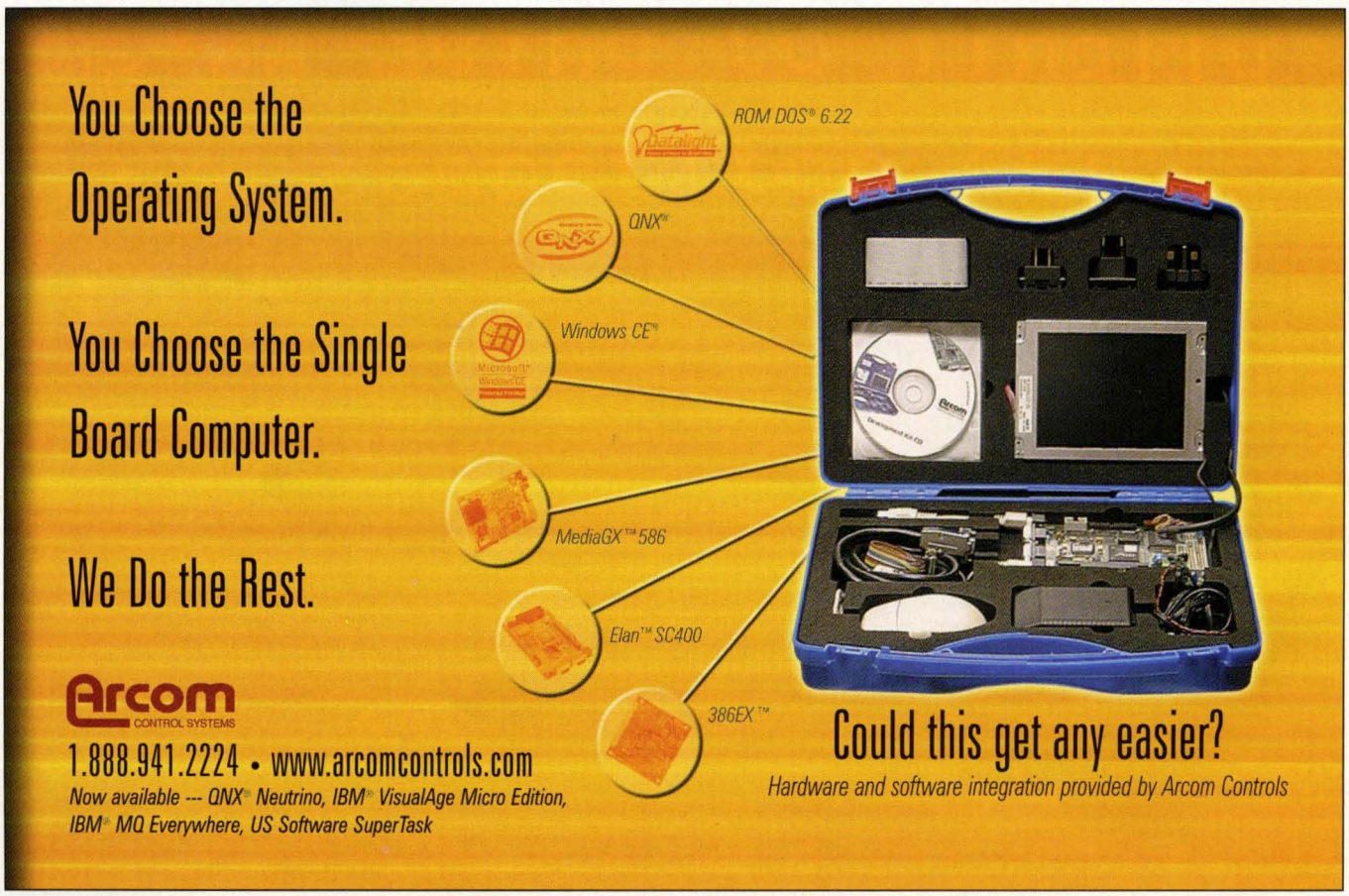
```
/*
 * IP Header
 */
typedef struct
{
   INT8U    ver_hlen;    /* Header version and length (dwords). */
   INT8U    service;     /* Service type. */
   INT16U   length;      /* Length of datagram (bytes).*/
   INT16U   ident;       /* Unique packet identification. */
   INT16U   fragment;    /* Flags; Fragment offset. */
   INT8U    timetolive;  /* Packet time to live (in network). */
   INT8U    protocol;    /* Upper level protocol (UDP, TCP). */
   INT16U   checksum;    /* IP header checksum. */
   INT32U   src_addr;    /* Source IP address. */
   INT32U   dest_addr;    /* Destination IP address. */

}
NetIpHdr;
```

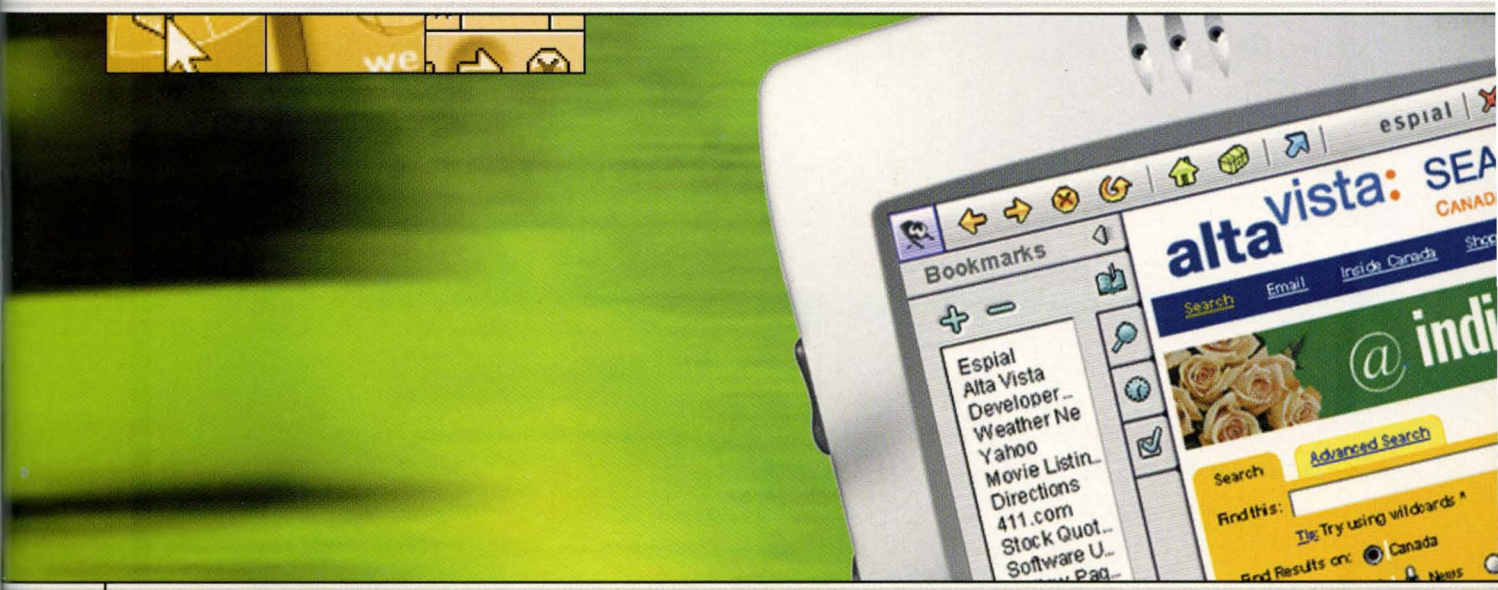
Listing 1 continued on p. 72

The result of all this is a logical connection between two computers. This connection can be used to send and receive all kinds of data. For example, the two computers could exchange Web pages via HTTP over TCP, SNMP packets over UDP, telnet sessions, voice over IP, or any other type of data that can be sent in a stream of one or more digital packets.

Because so many types of data can be exchanged over IP, the Internet Protocol is full of features that are only used by a subset of connected hosts. In fact, the protocol itself is overdue for an upgrade from the current version, called IPv4, to the more capable IPv6. (For more on IPv6 and the coming changeover, see the article by Stephen Harpster in this month's Internet Appliance Design section.) Since IPv6 is not yet widely deployed and even many of the IPv4 features are not necessary



ESPIAL®



infrastucture software for the pervasive internet ...say again?

Our vision at Espial® is to move technology beyond the PC by harnessing the full potential of the internet. Access to information anytime, anywhere, for anyone – the industry is buzzing about it, but who can actually offer it?

Well, between our **service management software**, our **super-small internet applications** and our **development tools**, we meet the needs of service providers, OEMs and developers – in other words, everybody involved with getting **service-based smart devices** into the hands of consumers. Contact us to learn more about our complete software solution.

BUILD A SMART FUTURE™

espial.com

© 2000 Espial Group Inc. Espial is a registered trademark of Espial Group Inc. The Espial logo and 'Build a smart future' are trademarks of Espial Group Inc. All rights reserved.

LISTING 1, cont'd. Data structures and constants for the IP layer

```
#define IP_VER_HLEN     0x45
#define IP_HEADER_LEN   sizeof(NetIpHdr)
#define IP_DATA_LEN     (MTU - IP_HEADER_LEN)

/*
 * IP Packet
 */
typedef struct
{
  NetIpHdr        ipHdr;
  unsigned char   data[IP_DATA_LEN];
}
```

LISTING 2 A function to send an IP packet

```
int
NetIpSnd(NetIpPkt * pIpPkt, INT16U len)
{
    INT16U  ident;

    /*
     * Assign a unique ID to the outgoing packet.
     */
    // Enter critical section here.
    ident = gNextPacketID++;
    // Leave critical section here.

    /*
     * Fill in the remaining IP header fields.
     * (Some fields were pre-filled by the UDP layer.)
     */
    pIpPkt->ipHdr.ver_hlen    = IP_VER_HLEN;
    pIpPkt->ipHdr.service     = 0x00;
    pIpPkt->ipHdr.length      = htons(len);
    pIpPkt->ipHdr.ident       = htons(ident);
    pIpPkt->ipHdr.fragment    = 0x00;
    pIpPkt->ipHdr.timetolive = 0x10;

    /*
     * Compute and fill in the IP header checksum.
     */
    pIpPkt->ipHdr.checksum = NetIpChecksum((INT16U *) pIpPkt, IP_HEADER_LEN);

    /*
     * Forward the IP packet to the network interface driver.
     */
    return (NetPhySnd(htons(PROTO_IP), (unsigned char *) pIpPkt, len));
}
```

for a lot of applications, my implementation of the IP layer includes only a sort of bare-minimum of functionality to satisfy IPv4-style communications. Most commercial TCP/IP stacks for sale in the embedded systems marketplace are more complete.

To get a feel for what is and isn't supported, let's look at the data structures and constants defined in Listing 1. Of particular note here is that the `NetIpHdr` structure defines a header of a fixed size. In truth, the length of this header may be extended in 32-bit increments to support some less commonly used IP features. However, µC/Net never uses any of the special features that these extensions allow. So variable length IP headers are not generated by the stack and it will reject any incoming IP packets that have a header of length other than 20 bytes.

The first field in any IP header, `ver_hlen`, gives the protocol version (4 for IPv4) in the first nibble and header length (five 32-bit words for the fixed size header) in the second nibble. All of the IP packets sent by µC/Net will, therefore, have this field set to the constant value 0x45. Similarly, it is a requirement that all packets received by the stack have that same value in `ver_hlen`.

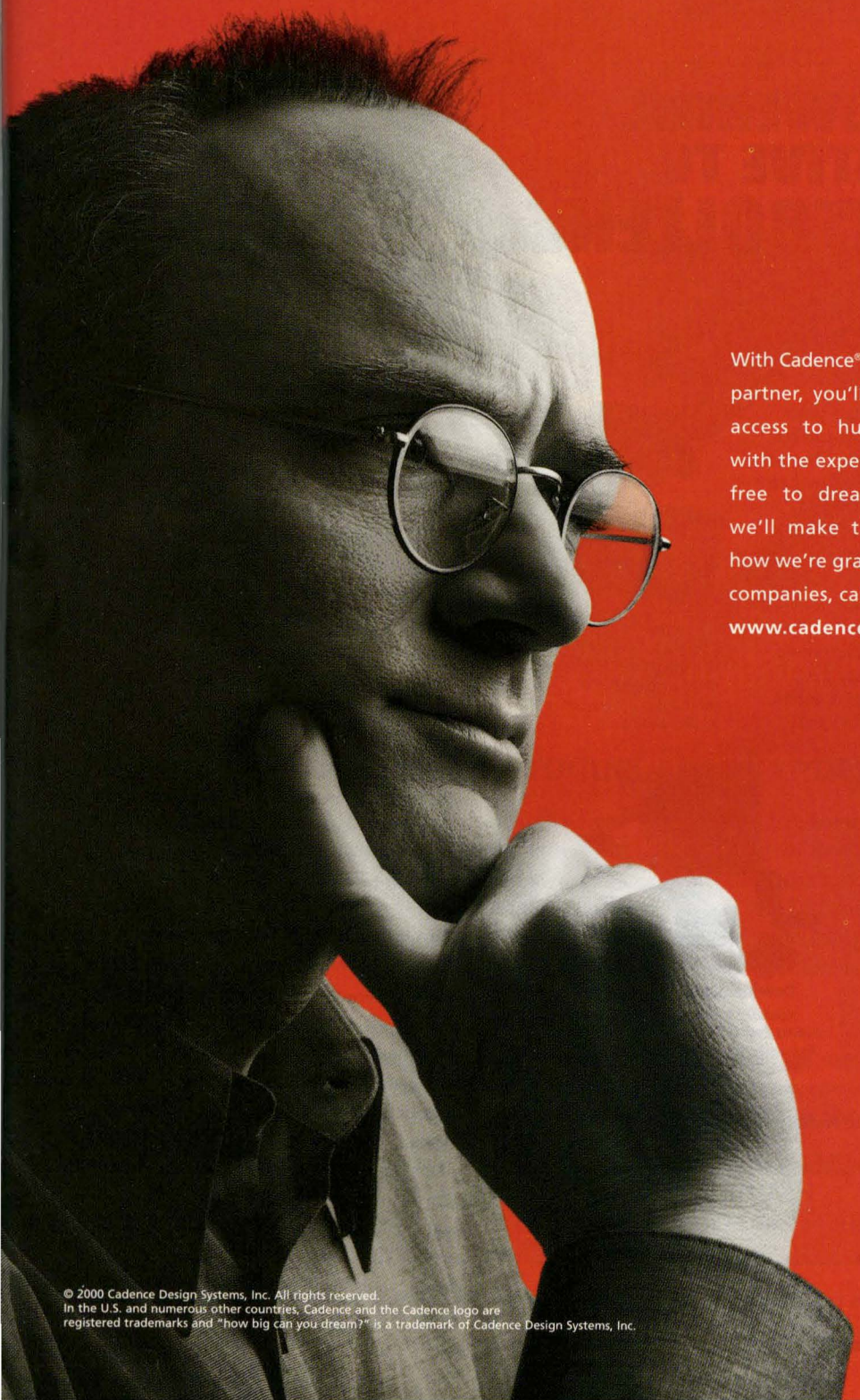
In addition to this restriction, several other features of IP are either unsupported by µC/Net or the support is significantly scaled back. The only one of these that is likely to affect you is their lack of support for fragmentation and reassembly.

Fragmentation and reassembly

Ethernet frames have a defined maximum size payload of 1,500 bytes each. This is termed the maximum transmission unit, or MTU, of the physical network. Included within that payload are the contents of the IP header, the UDP (or TCP) header, and the actual data. If all of the pack-

It takes time to put together an expert design team.

Sometimes as long as 24 hours.



With Cadence® as your Design Services partner, you'll have quick and easy access to hundreds of designers with the expertise you need. So feel free to dream up those ideas—we'll make them happen. To see how we're granting wishes for other companies, call 800.746.6223 or visit **www.cadence.com/24hours**

Cadence

cadence®

how big can you dream?™

© 2000 Cadence Design Systems, Inc. All rights reserved.
In the U.S. and numerous other countries, Cadence and the Cadence logo are registered trademarks and "how big can you dream?" is a trademark of Cadence Design Systems, Inc.

Unfortunately, the implementation of fragmentation and reassembly is extremely cumbersome. Adding just this one feature to the IP layer would approximately double the amount of code (and code complexity) of the entire µC/Net stack!

ets of data you want to send and receive fit within that maximum size—as is the case in the majority of embedded applications requiring only limited connectivity—µC/Net will do the job handily. If, however, you want to send even a single 1,501-byte frame, you'll run into trouble.

A more complete IP layer implementation would include a feature called fragmentation and reassembly. The idea that underlies this feature is simple: payloads too large to be sent over the physical network all at once should be split up across multiple frames (fragmented) and reassembled at the receiving end. The IP header field `fragment` can be used by the stacks on the two ends to coordinate this process.

Unfortunately, the implementation of fragmentation and reassembly is extremely cumbersome. Adding just this one feature to the IP layer would approximately double the amount of code (and code complexity) of the entire µC/Net stack! For that reason, I've made the decision not to support fragmentation and reassembly at all at this time. If the UDP layer attempts to send a packet that is too large, the IP layer will simply truncate the data to `IP_DATA_LEN` bytes and send that instead. If a remote computer fragments and then sends a large packet to a computer running µC/Net, the incoming data will appear to the client or server application as though it were actually multiple unrelated packets of information, rather than one really large one. (A poorly coded application could get confused by this, so beware.)

Sending and receiving

By now you must be wondering what IP features I have implemented. Ultimately, the IP layer is just responsible for sending and receiving IP packets. Listing 2 shows the code for `NetIpSnd()`, which does the sending. The IP packets resulting from this function call should make sense to any IPv4 system to which they are sent. That's the beauty of a more limited implementation of the protocol:

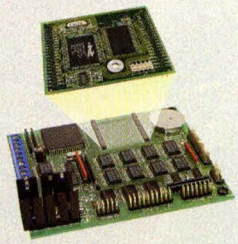
A MORE POWERFUL ALTERNATIVE TO MICROCONTROLLERS



1.90" x 2.30" In Size

$29 qty 100
$39 qty one

Design your custom microprocessor-based board around the Rabbit 2000™ Core Module. The core module is based on the new Rabbit 2000 microprocessor, and can handle programs of up to 50,000 C statements. Benefit from easy and rapid development with this low-cost solution!



Make the Rabbit 2000™ Core Module the heart of your custom board design. A low-cost manufacturing license is also available.

RABBIT 2000™ CORE MODULE FEATURES

- 40 I/O pins
- 4 serial ports
- 7 timers
- Battery-backed time/date clock
- 128K-512K SRAM
- 256K flash
- Clock up to 25.8MHz
- Slave port
- Remote cold boot
- Excellent math processing



Development Kit only $169

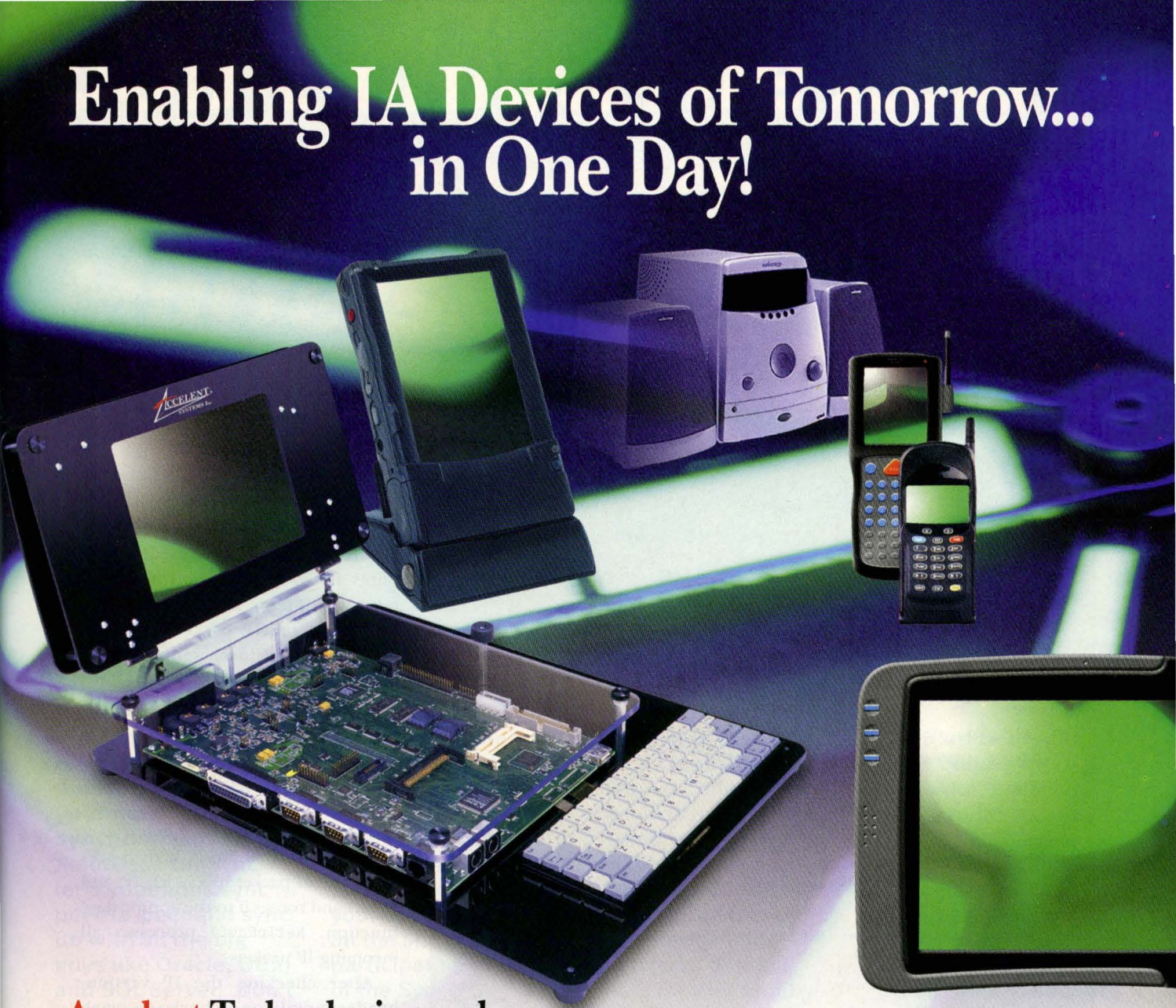
Includes core module, prototyping board, complete Dynamic C® software development system (not a trial version), PC serial cable for real-time debugging, and documentation on CD-ROM

RABBIT Semiconductor

Order the Rabbit 2000™ Core Module Kit online at
www.rabbitsemiconductor.com
or call 530.757.8400

2932 Spafford Street, Davis, CA 95616 • Fax 530.757.8402

Enabling IA Devices of Tomorrow... in One Day!



Accelent Technologies and Products for Fastest IA Product Development

As the leading enabler of IA devices, Accelent Systems can drastically accelerate the IA product development cycle. The power of the Accelent solution is in the underlying technologies that have been developed for devices running on Windows® CE or Linux for ARM® *(including Intel® StrongARM®)* processors. Accelent has packaged its technologies and system software into development and reference platforms, such as WebPad and Pocket PC among others, that allow you to focus on your application. As such, you are able to develop a more complete, reliable IA product in a fraction of the time and cost. That's why we are known as the Fastest IA Solutions Provider™ – FSP™!

Test drive our technology by visiting the ***FSP One Day Design Center***™ at ***www.accelent.com*** or call us with your project requirements and we'll give you ***free operational test code in One Day.*** Qualified visitors will receive a ***free FSP Atomic Clock***** *(as a symbol of our precision solutions)* and will be entered into a monthly drawing for an ***AudioRamp iRad Internet Stereo*** that was developed using our solutions!



ACCELENT SYSTEMS Inc. ***Fastest IA Solutions Provider***

Phone: 330-864-2300 ▪ Fax: 330-864-8674 ▪ www.accelent.com

See a demo at the Embedded Systems Conference, booth #8055

All trademarks are the property of their respective owners.

*While supplies last, limited quantity per project.

LISTING 3 A function to receive an IP packet

```
int
NetIpRcv(NetIpPkt * pIpPkt)
{
    INT16U  checksum;

    /*
     * Check IP header version and length.
     */
    if (pIpPkt->ipHdr.ver_hlen != IP_VER_HLEN)
    {
        /*
         * Unsupported header version or length.
         */
        return (NET_ERROR);
    }

     /*
     * Move the IP header checksum out of the header.
     */
    checksum = pIpPkt->ipHdr.checksum;
    pIpPkt->ipHdr.checksum = 0;

    /*
     * Compute checksum and compare with received value.
     */
    if (checksum != NetIpChecksum((INT16U *) pIpPkt, IP_HEADER_LEN))
    {
                    return (NET_ERROR);    //Bad checksum
    }

    /*
     * Route the packet to the appropriate Layer 3 protocol.
     */
    switch (pIpPkt->ipHdr.protocol)
    {
      case PROTO_UDP:
        return (NetUdpRcv((NetUdpPkt *) pIpPkt,
                          ntohs(pIpPkt->ipHdr.length) - IP_HEADER_LEN));
#if defined (NET_TCP_EN)
      case PROTO_TCP:
        return (NetTcpRcv((NetTcpPkt *) pIpPkt,
                          ntohs(pIpPkt->ipHdr.length) - IP_HEADER_LEN));
#endif

      default:
      return (NET_ERROR);                    // Unsupported protocol
    }
}   /* NetIpRcv() */
```

it's small, easily coded and understood, and yet—because it implements a subset of the full protocol—can communicate with any system on the network.

Some fields of the IP header are filled in by the UDP layer above. Specifically, the UDP layer fills in the IP header's source and destination IP addresses, which it combines with the UDP header and payload to compute a UDP checksum. The UDP sending function then passes the entire packet to `NetIpSnd()`. After filling in the remaining fields of the IP header, `NetIpSnd()` computes its own checksum (of the IP header contents only) and passes the completed IP packet on to the network driver, to be sent out over the physical network.

The `NetIpRcv()` function, shown in Listing 3, handles incoming IP packets. This function is called from within the context of a task that's part of the network driver. The driver task becomes active whenever a new packet arrives over the network. It then sees what type of packet it is (ARP, IP, and so on) and routes it to the appropriate function. `NetIpRcv()` processes all incoming IP packets.

After checking the IP version, header length, and checksum, each incoming IP packet is routed to the layer above. If it is a UDP packet, `NetUdpRcv()` is called. If it is a TCP packet and TCP support is included, `NetTcpRcv()` is called instead.

Next month I'll tell you everything you ever wanted to know about the UDP layer that lies above this one. And, of course, I'll share my implementation with you. In the meantime, stay connected... **esp**

Michael Barr is the editor in chief of Embedded Systems Programming. *He holds BS and MS degrees in electrical engineering from the University of Maryland. He is the author of the book* Programming Embedded Systems in C and C++ *(O'Reilly & Associates). Michael can be reached via e-mail at mbarr@cmp.com.*

Don't Cloud the Issue, Synchronization Matters!



Why should your mobile apps be uninformed when all your data can be synchronized? PointBase UniSync let's your apps sync peer-to-peer and sync up with all the big guys like Oracle, DB2, and SQL Server. Don't be fooled. A few database companies will cloud the issue with foggy messages about data ubiquity, when some can't even sync with themselves!

Sidestep all the fog and mist about synchronization control. PointBase UniSync provides an intuitive and complete pure Java API that lets your application call all the shots. We are a participating member of the SyncML Consortium so we employ all the synchronization standards today and we know what's coming tomorrow.

Lightning fast even on a cloudy day! From mobile/wireless devices to lap-tops to servers, PointBase UniSync allows you to create global data management and synchronization networks that get data moving rapidly from hubs to spokes to all corners of the enterprise ... and back again.

It's hard to predict what clouds will do. Your data, however, needs to be secure and needs to move to where it's needed in a predictable way. PointBase UniSync offers full encryption and conflict resolution capabilities to support enterprise data synchronization around the block and around the world.

PointBase

*Managing data **anywhere, anytime.***

***NEW:** Free Eval copy of PointBase UniSync for building powerful data synchronization platforms and networks. Call us at: (877) 238-8798 Toll Free*

***NEW:** Free Eval copy of PointBase Embedded Server 3.3 pure Java database. www.pointbase.com/esp*

***NEW:** Cut through the fog with your own copy of our Synchronization White Paper. www.pointbase.com/wps*

NIALL MURPHY

Open Source Point/Counterpoint:

Are Open Source and Innovation Compatible?

According to the popular wisdom of the moment, open source software is the way of the future. But can the purveyors of free software really innovate?

Members of the open source movement contend that they have found a better way of writing and distributing software. Because the cost of software has increased as a fraction of total development cost, the movement is appealing to embedded developers. However, I question whether we are going to see much more from this movement than the limited benefits already seen.

Over the years, the free software movement has shifted from a set of principles based on building software independent of a commercial community—the original GNU ("GNU's Not Unix") approach—to building software that could blend with commercially supported offerings. Most recently, the movement has shifted again—from an engineering philosophy to a business model (climaxing in several well-publicized stock market IPOs)—and this has transformed open source from a guiding principle into a marketing policy.

While the open source movement, in its many guises, has undoubtedly made valuable contributions to the engineering community, I contend that in its present form and direction it has little more to offer either as a business model or as a software construction model. I further contend that open source offers little that has not appeared before.

Nothing new

The three basic tenets[1] of the open source movement are that software be made available in source code form, legally modifiable for any purpose, and redistributable. Individually these tenets have been part of the commercial software landscape. For example, the RTOS market is supplied by several vendors who choose to supply the source

I question whether we are going to see much more from [the open source] movement than the limited benefits already seen.

code of their product, albeit to paying customers. Modifications of that software are allowed and sometimes necessary to make the RTOS suitable for a specific application. However, a buyer is not typically allowed to give away or sell the resulting code as a new RTOS.

Free redistribution is where members of the open source community believe that they are doing something radical. This is also where they try to turn business madness (giving one's product away) into a business model. But is this really all that new? Let's look at it from a business perspective. Consider that there are some customers who are prepared to pay a reasonable amount for your software and that there are other potential customers who are not willing to pay the price. Some products benefit greatly from the network effect,[2] which says that the more popular the product, the greater the value of any individual item. This is true for telephones, since it means that I can call a greater number of people on the phone that I purchased, and it is true for operating systems, since their popularity dictates the number of applications, and the amount of expertise, that are available.

It is the network effect that makes it sometimes worthwhile for a business to give its product away for free to everyone, or at least to those who would not otherwise purchase it. (Even hardware costs can be subsidised, as is currently done in the cellular phone and satellite television markets.) Many companies have taken advantage of this in the shareware marketplace. It has also worked to commercial software developers' advantage, in a more subtle way, as the result of software piracy. In a sense, free redistribution and unlitigated piracy are one and the same business model. (It's interesting to think about how these models are already competing in the operating system market for PCs.)

Follow the money

Even if the open source model cannot introduce anything new, perhaps it will still be a success. Maybe the combination of the three properties described above will work more than any one or two of them individually.

Success, of course, can be defined in more than one way. Success can be defined as "makes money," but I would prefer to define success in this arena as "results in a better product, or better value, to the user." By the first definition, a number of Linux companies have succeeded in making large amounts of money from their IPOs. After the fashion of a host of other dot-com start-ups, they have managed to trade potential for cash. Like many of the dot-com companies, some will fail to live up to their potential. However, with a bit of luck these companies will contribute a bunch of free software to the community before they go bankrupt.[3]

Unfortunately, I am not sure that even this benefit will accrue to the programming community. It is significant that the major open source companies are all leveraging already existing open source products, which were originally written with no commercial motivation. I contend that these companies will fail to ever truly innovate. Innovation requires a level of risk, and the returns will never justify the risk when the playing field has been levelled by an open source philosophy. Even the most successful open source products that already exist t>nd to be imitations of successful commercial products.

In this way open source companies are dodging the two most expensive parts of a product's life. The definition of requirements that is necessary to define a new market is often the part that requires the most vision and innovation. The second, and financially larger, cost is marketing. Today, demand for Linux would be minimal had commercial Unix not already established the definition of a good operating system and market awareness of the problems Unix can solve. Marketing costs are often 10 times development costs, sometimes more.

Open source companies claim that they are putting back into the community by releasing their improvements as open source. Cygnus was, at one point, doing 50% of all new development on the GNU compiler toolchain.[4] They should be applauded for this, certainly, but the statistic overlooks the fact that they were not contributing 50% of the effort of defining the C language, nor of marketing its benefit. When the open source community is asked to produce products including the costs of definition and of marketing, I believe that the model will fall asunder.

In "Giving It Away," Red Hat's Robert Young considers how they've succeeded at selling their brand of Linux, when the customer can get the same software for free elsewhere.[5] He compares this to Evian selling bottled water. Brand recognition counts for more than what the product contains (some bottled waters have even been found to be of lower quality than the local tap water). This model will thrive in the pure commodity market, but the engineering community is faced with challenges that will only be solved with real research and development effort. Young's model will not solve those problems, just repackage the solutions that already exist.

Not only are these new, well-funded companies unlikely to innovate themselves, their commercial interests may stifle some efforts coming from academia or elsewhere. Consider if I were to independently write a cross compiler

RUPERT ADLEY

Know Your Rights

One of the more confusing aspects of the open source phenomenon is the proliferation of different source code licensing schemes. If you are considering using software developed by others within your products, you'll probably want to have an intellectual property lawyer read the individual license agreements in detail and summarize your rights. However, this quick guide to the licensing terminology should be enough to get you started.

Proprietary Software

In the most traditional licensing model for commercial software, no customer is allowed to examine the source code for the product. The one exception to this is a customer willing to pay an additional, often exorbitant, source code license fee. But even if you do buy the source code for such a product, you cannot publish that code or otherwise cause it to fall into the possession of anyone outside the licensed group. And your rights to modify the source code may be restricted.

Shareware

With shareware, the traditional model is somewhat reversed. The software is distributed for free, and may even be passed on to others. Users of the shareware product are honor-bound to pay the developer's registration fee, whatever it may be. A variant on this licensing scheme, termed crippleware, will not function fully until the registration fee is paid. Freeware is shareware with no registration fee. The source code is typically not included with any of these.

Free Software

Free software, on the other hand, is all about access to the source code. The free software movement (*www.fsf.org*) is as much a political organization as anything else. Under the free software licensing model, it is your right to use the software, modify it, and redistribute it in any way you like. It's even okay for you to charge for your distribution. However, these broad rights are conditioned upon your commitment to provide similar access to your modifications and to never narrow the licensing rights as a condition of distribution.

Proponents of free software generally believe that all information, especially source code, has a right to be free. Therefore, they mean free software in the "free speech" sense, not as in "free lunch." They advocate attachment of a *copyleft*, which says that "anyone who redistributes the software, with or without changes, must pass along the freedom to further copy and change it."

Public Domain Software

Public domain software is just as free as free software, but less restrictive. It can be used, modified, or redistributed in any way. However, you are free to make changes to the software and keep those changes proprietary. You can even choose to charge for the original code or a derivative, without providing access to the source code at all. In that way, anyone can make use of public domain software in any way, without consulting a lawyer beforehand.

Unless otherwise indicated, all of the source code published in *ESP* is placed into the public domain by its author(s). In other words, while copying the text of the article is a copyright infringement, using the source code in your work is perfectly legal.

continued on p. 82

and release it as open source. A few years ago, if its quality were superior to anything else available for free, the free software community would embrace this new product and it would thrive. Now, following a spate of multi-million dollar IPOs I would face competition from companies with endless financial resources, who already had large investments in the established open source cross compiler GCC. Similarly, internal innovations will be unlikely since they could immediately be improved upon by competitors, once the source had been released. David has grown up and become Goliath.

Commercial pursuits

The IPO market is probably not the best place to look for typical performance of open source companies, since it is so speculative in nature. Let us look instead at some of the established companies that have made a business decision to release their flagship product with an open source license. Netscape (released their browser Mozilla), Zinc (open sourced their graphics toolkit), and Inprise (released the Interbase database as an open source product) all spring to mind. Netscape and Zinc were bought by other companies within months of the open source releases, and Inprise narrowly escaped being taken over by Corel. I am not suggesting that releasing their software as open source caused a collapse, but would rather suggest that the open source release was a last gasp attempt to save a threatened product or company.

Other companies have provided open source releases of products that were not so close to their core business, to encourage the sales of their core products or services. For those companies, the open source package is often loss leader, and in many cases releasing the source itself does not really benefit the community because the product is so inextricably linked to other products from that same company. In the case of AOL, they released one version of their Instant Messaging (IM) client as open source, but then discovered that this

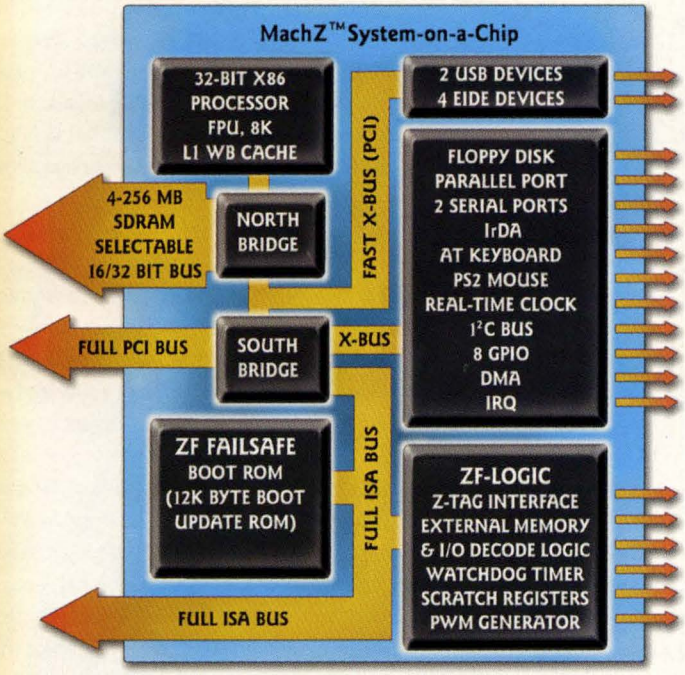
Finally
a *real* x86 System on a Chip



(actual size)

Introducing the "FailSafe MachZ" for the next generation of embedded systems

You can design the MachZ system into virtually anywhere your imagination can take you. It's very small and at a 1/2 watt it actually runs cool to the touch. Think about it: you could embed this little power house in a system that is completely sealed. Consider the possibilities, consider the facts. Then put your engineering brain to work.



Ultra Low Power <500mW@100MHZ

Auto-Boot FailSafe System allows crash-immune Internet upgrades – FailSafe Boot, Flashless Operation, Dual WD-Timer, Z-tag 2M-bit/s Flash download, Scratch Registers

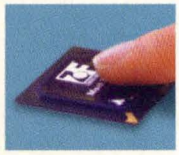
Fully PC Compatible runs all x86 code natively

Easily Interface HomePNA, Ethernet, Modems, Graphics Controllers, PC/104, MPEG, XDSL, ISDN, IrDA, Flash, CAN, GPS

Includes BIOS and Linux O/S or WindRiver VxWorks RTOS

phoenix technologies

Oh, and did we mention that it runs cold?



388 BGA Runs at a cool 1/2 Watt not a typical 8 to 10 Watts

FailSafe EMBEDDED

ZF LINUX DEVICES™

800.683.5943 • www.zflinux.com

Visit us at Embedded Systems Conference 2000, Booth# 2002

American RAISONANCE



Introducing American Raisonance. Your new supplier of development tools for the Philips XA, 8051, and ST6 microcontroller families.

XA

Packages available with assembler, compiler, simulator and utilities for the Philips XA. Web distributed for immediate installation. One year technical support included in purchase.

8051

Available individually or combined with XA tools. Source Code compatible with Keil 8051 tools for easy migration to XA.

ST6

Free assembler and simulator. Optimizing compiler including RIDE and utilities for program development.

CAN

For CAN development, American Raisonance also distributes IXXAT CAN products and libraries for CAN-Open and DeviceNet.

Call today for introductory pricing!

877-315-0792

www.amrai.com

Know Your Rights (cont'd. from p. 80)

GNU General Public License (GPL)

Since 1984, the goal of the GNU project has been to develop a complete UNIX environment that is licensed as free software. Although some of the code involved is public domain, the vast majority is licensed under GPL (*www.fsf.org/copyleft/gpl.html*). GPL is a specific implementation of copyleft. This is analogous to copyright law, in which there is a general right that is implemented in various ways in different contracts and print and electronic publications.

GPL prohibits proprietary patents related to modifications of the software, prohibits royalties, and requires that the same terms be attached when redistributing the software or a derivative of it. Of course, anyone can create software and then license it under these same terms. Use of the GPL language is not restricted to GNU-related projects. (Their copyleft is not copyrighted.)

The popular GNU compiler and associated tools are licensed under GPL. This means that anyone making a new and improved GNU compiler must give their new source code back to the community. However, it is important to note that this does not mean that software built with the compiler must also become free. It is legal to use a free software tool to produce proprietary software.

GNU Lesser General Public License (LGPL)

The LGPL (*www.fsf.org/copyleft/lesser.html*) is used to license free software so that it can be incorporated into both free software and proprietary software. In other words, it is a weaker sibling of GPL. The rules are basically the same, with one major exception: the requirement that you open up the source code to your own extensions to the software is removed. So while LGPL components remain free software, they can be included within a larger proprietary software package.

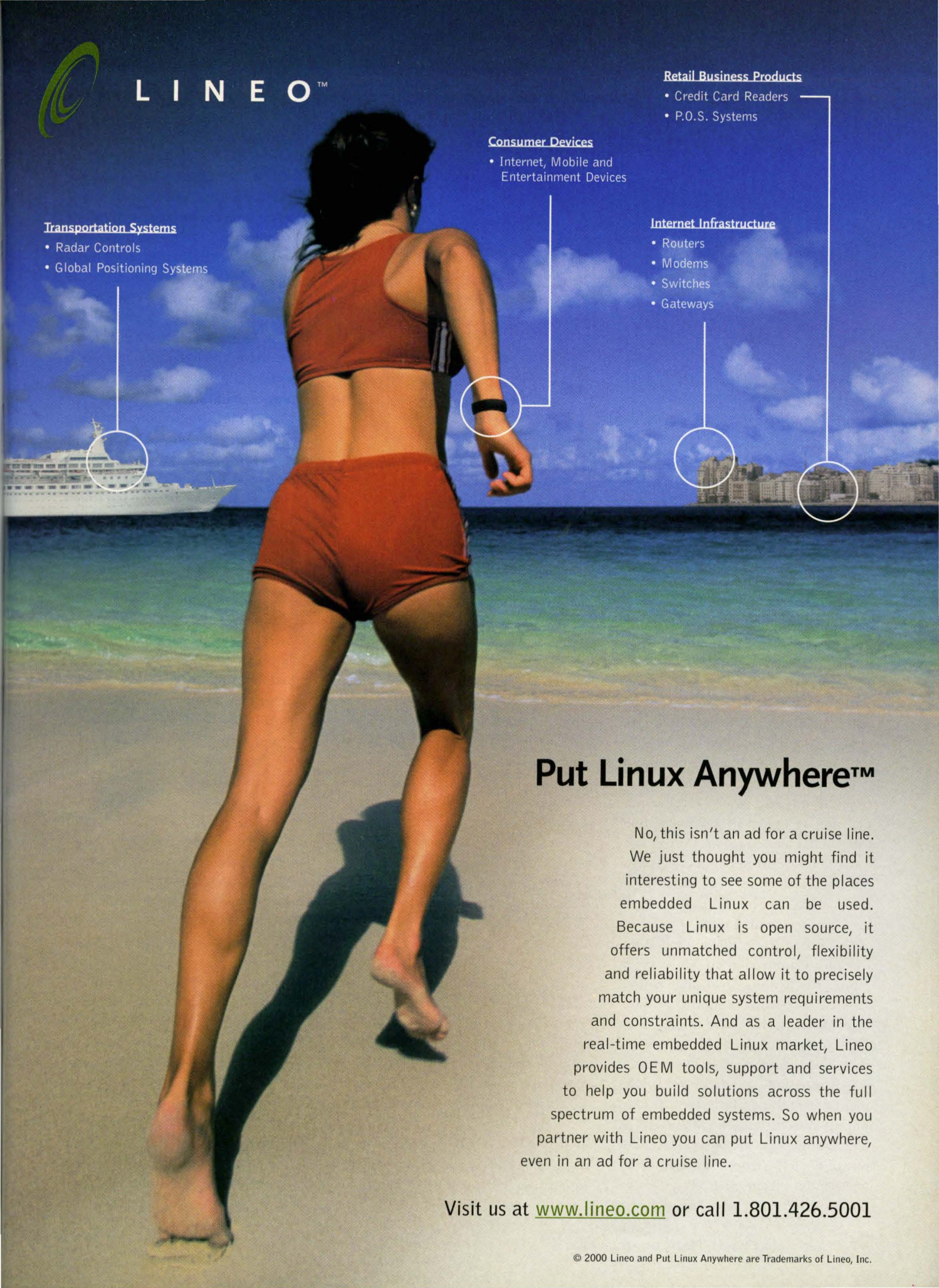
The real downside of the GPL, particularly for embedded developers, is that it's designed to discourage the creation of proprietary software and to encourage free software. If you wanted to build your firmware around a GPL package or library, you'd be forced to give away the source code to your firmware. But this is not a problem with an LGPL package, like the GNU C library, which can be legally included as part of proprietary software.

Open Source

Unfortunately, this is where things get confusing. Although we've framed our debate with those words, there is no clear definition for "open source software" and no standard license. Many companies are using the term open source these days, but in far different ways. And while the idea is similar to that of free software (you can generally still use, modify, and redistribute the software), there is far less emphasis on the right of the source code to be free.

While not as true with respect to Linux, many open source companies seem to be unwilling to give up central control of their software. (Free software has no owner.) What's important about open source software, particularly for embedded developers, is that its licensing terms are more like LGPL than GPL. In other words, you are typically free to add your own proprietary software to the open source code and produce a proprietary result. The free software movement doesn't much like this, but is otherwise more in alignment with the newer open source movement than in opposition to it.

continued on p. 84



Put Linux Anywhere™

No, this isn't an ad for a cruise line. We just thought you might find it interesting to see some of the places embedded Linux can be used. Because Linux is open source, it offers unmatched control, flexibility and reliability that allow it to precisely match your unique system requirements and constraints. And as a leader in the real-time embedded Linux market, Lineo provides OEM tools, support and services to help you build solutions across the full spectrum of embedded systems. So when you partner with Lineo you can put Linux anywhere, even in an ad for a cruise line.

Visit us at www.lineo.com or call 1.801.426.5001

© 2000 Lineo and Put Linux Anywhere are Trademarks of Lineo, Inc.

Know Your Rights (continued from p. 82)

Community Source

Sun Microsystems' community source licensing program (*www.sun.com/981208/scsl/principles.html*) was launched to give Sun and its customers the simultaneous advantages of both free and proprietary software. Users of the software have full access to Sun's source code and can do their research and development work before paying licensing fees. They can even modify the code for any purpose. However, users are asked to register at the time of code download and are ultimately required to negotiate licensing terms with Sun prior to launching any commercial product based on that code.

Sun felt this licensing model was a better alternative than making its formerly proprietary software into free software. By keeping licensing fees in the equation, Sun hopes to reap some benefits from all this giving away of its most valuable code. (To date they have made the source code for Solaris, StarOffice, and many Java technologies available on their Web site.) By retaining their role as a central authority on the software and a provider of compatibility testing, they hope to reduce chaos and source code fragmentation in the market.

—*Michael Barr*

provided Microsoft, Yahoo!, and others with a competitive advantage. They immediately released another incompatible (closed source) version. Any developer who was depending on support for the open source version was left high and dry. According to the open source philosophy, it should be straightforward for the user community of that software to support it themselves. In this case, however, the source was useless unless it was compatible with AOL's closed-source server software. Software from commercial organizations will always get caught up in these battles, leaving third-party developers as incidental victims.

At the Spring 2000 Embedded Systems Conference in Chicago, a panel discussion on open source seemed to revolve around one question: "How do we make money from open source?" Making money was neither the primary objective of those who started the open source movement nor of those who made the most important contributions. So this is a bit like asking: "How do we make money out of poetry?" True, some people do make money from poetry but if everyone involved starts to focus solely on what makes money, we are not likely to see much new, innovative poetry (though there would be an incredible number of publishers selling "Best Loved Verses" compendiums).

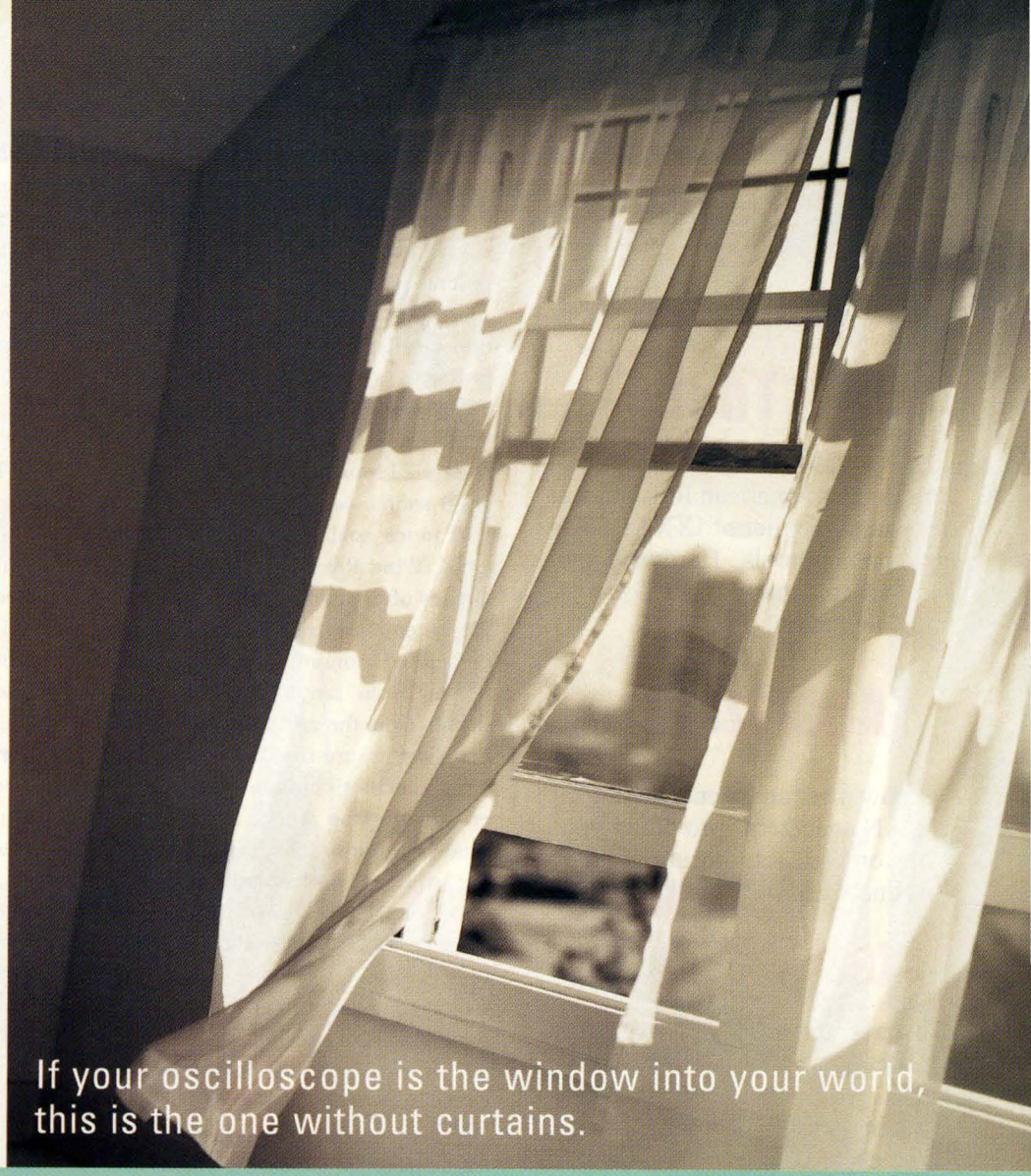
Encouraging embedded engineers to release code as open source rarely has benefit unless the hardware design is also included. There is no evidence that this is happening and it is unlikely to start. The chances of me being able to use an open source alarm system as a starting point for the commercial alarm system that I wish to develop are slim. When it comes to end-user applications with narrow markets, competition dictates that releasing designs becomes commercially foolhardy. So the only software that the OS community is likely to receive is the material like operating systems and compilers which are not application-specific. This means that the opportunities for embedded engineers to share the majority of the code they write will always be very limited.

Embedded Linux

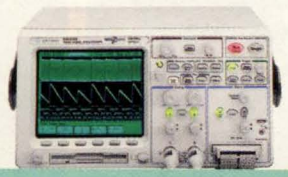
The greatest desire for third-party software within the embedded community is for an operating system. Linux vendors and a number of market commentators are trying to convince us that embedded Linux will become the preferred choice.[6] The fact that it is not a real time operating system can be overcome in a number of ways. One technique involves running Linux, including multiple processes, as a low-priority task within a more conventional RTOS.[7] Another is to embed the RTOS in Linux as a device driver, since device drivers can handle interrupts and generally have better real-time properties than Linux processes.

What is the motivation for these architectures? The phrase "easy portability of application software and programming skills from desktop Linux" gets tossed around a lot. The last time I heard similar logic was in the early stages of Windows CE marketing. It was as bogus then as it is now. Desktop-style applications can only be used in a tiny minority of embedded designs; admittedly, this refers to large volume applications like set-top boxes. Sharing a skill set is bogus too. The vast majority of embedded software developers have never written a single program to the Unix API (nor the Win32 API, I might add) and asking them to learn one of these monsters in order to use an RTOS embedded within it is putting a huge onus on the engineer. Will programmers experienced in Linux on desktop machines now be able to turn their hand to embedded systems programming? This would assume that, apart from the knowledge of a particular API, there are no skills necessary in the embedded domain that are not already known to desktop programmers. This is simply not the case, and to suggest otherwise is an insult to the engineers who've spent years honing their skills as embedded and real-time specialists.

In the proposed real-time Linux environment, the developer will also need to learn how to communicate between Linux processes and RTOS tasks, as well as knowing both inter-process and inter-task communications mechanisms. The developer will also have to handle trade offs between putting certain application



If your oscilloscope is the window into your world, this is the one without curtains.



from **$2,295***

Agilent 54600 Series Oscilloscopes

- 2-channel, 4-channel, 2-scope +16 logic channels
- 60-100 MHz
- 2 MB MEGAZoom Deep Memory
- 32 levels of gray scale high-definition display
- Flexible triggering (Edge, Pulse width, Pattern, I^2C)

Since the advent of mixed analog and digital designs, it seems you need more channels, more memory and triggering capability than ever to clearly see your signals. Now you can see them as never before, on the 54600 Series Oscilloscopes from Agilent Technologies.

On our Agilent 54622D model, you can watch analog and digital signals interact simultaneously, through a combination of two scope channels and 16 logic channels. And all of our 54600 Series scopes offer our MegaZoom® Deep Memory technology with 2 MB of memory per channel, so you can easily look deep into MCU-based designs. In fact, the combination of deep memory and our patented new high-definition display system uncovers signal subtleties other scopes can't begin to show you.

As for getting your tough measurement questions answered, at Agilent Technologies you can talk to another engineer just by calling our toll-free number. And of course, we'll understand if you don't just take our word for it when we tell you how these scopes can improve how well you see into your design. You'll probably want to see the scope in action. No problem. You can arrange a demo of any Series 54600 scope with no obligation. Or visit our web site for an on-line demo. Either way, we think you'll enjoy the view.

www.agilent.com/find/megazoom

1-800-452-4844,** Ext. 6847

Agilent Technologies
Innovating the HP Way

* U.S. list price

©2000 Agilent Technologies ADEP3466010/ESP **In Canada, call 1-877-894-4414, Ext. 6847.

American RAISONANCE



CAN Tools

Introducing American Raisonance. Your new supplier of IXXAT CAN development tools. Hardware and software to simplify your CAN implementation!

Analyzers

We now offer an extensive line of CAN analyzers, providing you with complete monitoring of your CAN network.

Interface Cards

We have the CAN interface card you need, whether it's ISA, PCMCIA, parallel port or USB. Utilizing your PC, 80320 or a C165 every card comes with:

- CAN BUS Monitor
- User manual
- Download and flash programs
- VCI Virtual CAN Interface

Protocols

CANopen Master/Slave software. Delivered in C source code and comes with example programs and technical support.
DeviceNet Master/Slave software. Delivered in MASM/C source code and comes with example programs and technical support.

877-315-0792

www.amrai.com

functionality into a Linux process where it may have soft real-time targets, and putting it in a hard real-time task. In short the architecture has taken a solution (an RTOS) and divided it into two problems (a soft real time system, and an integrated RTOS).

The real motivation behind these real-time Linux architectures is to allow open source vendors to play catch-up with more established RTOS vendors. QNX, Wind River Systems, and others already offer any functionality (such as file systems and networking) that the embedded community may want to borrow from Linux. These products, written from the ground up for embedded systems, are free from a legacy of design decisions influenced by the server and desktop world. A number of smaller vendors have gathered under the banner of embedded Linux in the hope of upsetting the applecart for these larger vendors, and benefit to the tune of a few apples themselves. Unfortunately for these cidermakers, the strength of open source will always lie in its disruptive influences—it will never create new markets, just scare the big cats in already established markets. In the desktop OS market there was an undeniable monopoly, but the RTOS market is much more diversified and there is not the same dire need to shake up a single player with 95% of the market. The very fact that the RTOS market is reasonably fair and open means that open source software will not have the same appeal that it had when competing with Microsoft.

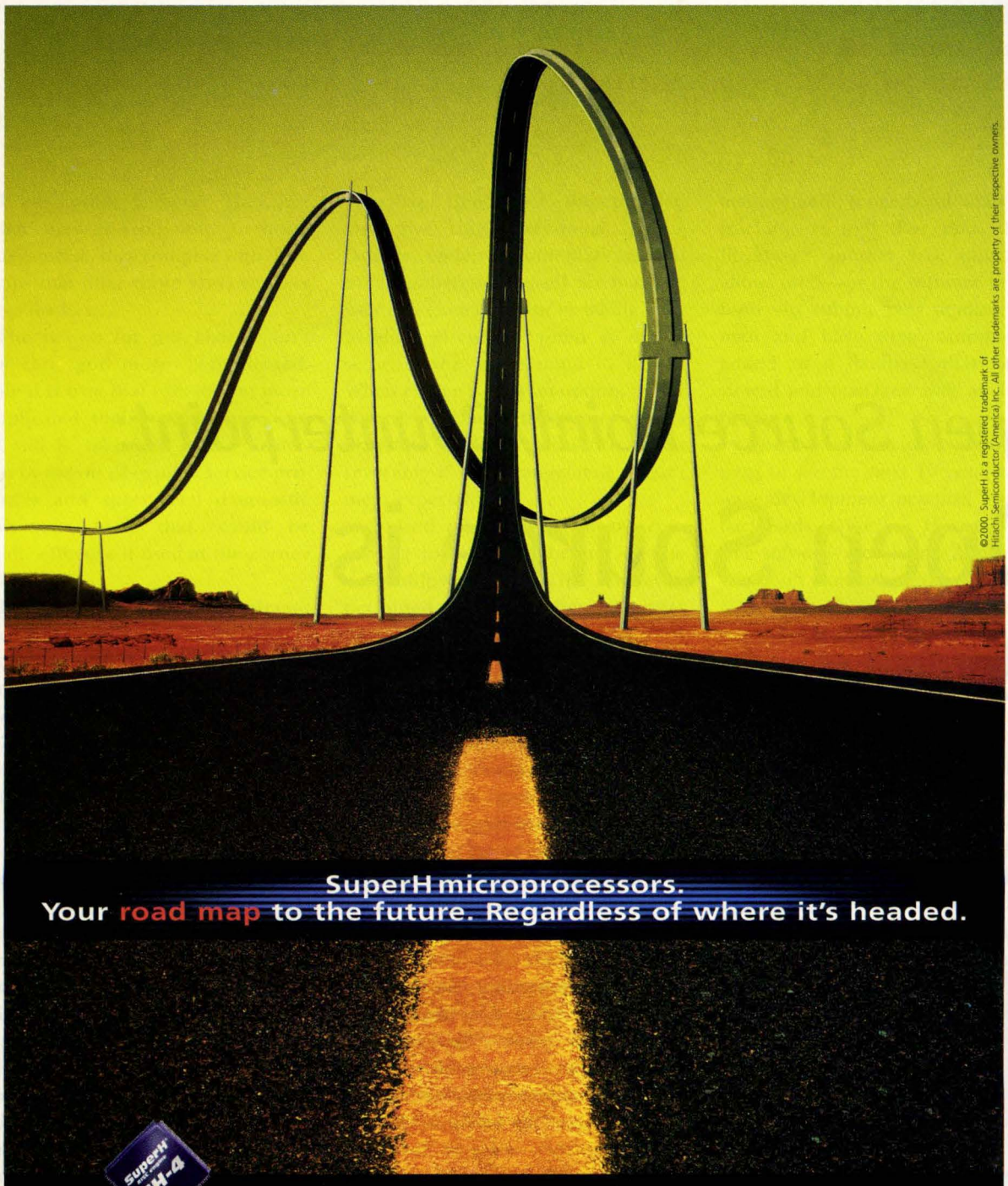
Moral high ground

The open source community claims to have the moral high ground over commercial developers. Bertrand Meyers disputes this and argues that much of the high-minded attitude is misplaced.[8] I tend to agree. Saying that you are writing software for the good of the world, and for no personal gain, becomes harder to swallow once you've become known throughout the engineering world as a multi-millionaire. Those who've made big money from the efforts of the open source developers will be compromised by their newfound responsibilities as owners of major corporations. And those who did not make money out of the recent spate of public offerings will be less likely to offer large chunks of their programming time for free next time around.

While some may try to convince us that open source vendors are the wave of the future, I think we will look back on this idea as a quaint notion that happened to provide the embedded community with a good cross compiler. I doubt this new way of doing business will profit any of us in the long run.

References

1. "Licensing Models of Open Source Computing" available at *www.eg3.com/open/license.htm*.
2. Shapiro, Carl and Hal R. Varian. *Information Rules: A Strategic Guide to the Network Economy*. Harvard Business School Press.
3. "Can Linux Billionaires Carry the Free Software Torch" by Andrew Leonard available at *www.salon.com/tech/feature/1999/12/23/linux_ipos/*.
4. "Future of Cygnus Solutions: An Entrepreneur's Account," by Michael Tiemann, included in *Open Sources: Voices from the Open Source Revolution*, edited by Chris DiBona, O'Reilly and Associates, 1999.
5. "Giving It Away: How Red Hat Software Stumbled Across a New Economic Model and Helped Improve an Industry" by Robert Young, included in *Open Sources: Voices from the Open Source Revolution*.
6. Crenshaw, Jack, "And Another Thing..." *Embedded Systems Programming*, March 2000, p. 15.
7. Epplin, Jerry, "Linux as an Embedded Operating System," *Embedded Systems Programming*, October 1997, p. 96. This article is available online at *www.embedded.com/97/fe39710.htm*.
8. Meyers, Bertrand, "The Ethics of Free Software," *Software Developer*, March 2000, p. 59. This article is available online at *www.sdmagazine.com/features/2000/03/f4.shtml*.







Voice



Video



Data

What's the fastest way to market? On board a SuperH® RISC SH-4, 32-bit MPU core. Everything you need to design your Internet appliance is mapped out and ready to drive off the lot. With SH-4- based chips from Hitachi and ST you get voice, graphics, data, and video in a high-performance, low power consumption environment. Then we throw in a comprehensive IP toolkit, unparalleled upward compatibility, and a price that's unbeatable. So no matter what the future has in store, the SH-4 lets you hit the road running. Check out www.SuperH.com today.

HITACHI
Semiconductor



BILL GATLIFF

Open Source Point/Counterpoint:

Open Source is Already Delivering

One need not speculate about the benefits of open source software. Just look to its past accomplishment. The future holds more of the same.

I wish to provide a counterpoint to Niall Murphy's views on the present and future of "open source" software for embedded systems. Mr. Murphy makes some good points but has overlooked much of the value of the open source movement to embedded systems designers.

Open source business models

The key concern of the open source business model, as I see it, is how to remain profitable when users of your product have significantly more rights than under other models. Because the end user can obtain your product or service elsewhere, a revenue stream is not assured unless users find that you are a better choice than a competing supplier. The solution: be better, just like a plumber or race car driver has to be better than the competition if they want to stay in business. Simple enough.

Mr. Murphy is correct that this model doesn't break any new ground. But the concept clearly hasn't been practiced in software industry until recently, and its practice still isn't widespread.

However, it's my opinion that because the open source model works for the good of the end user, and because the alternative is so bad, success of the open source movement is without doubt. As if to help motivate us, history provides examples of what the alternative outcome is. And it isn't pretty.

Think back to the late 1970s, before automobiles with computer-controlled engines were popular. At that time, garages and repair shops were almost as frequent as gas stations, and, through documentation and basic ingenuity, these shops could accomplish anything from serious diagnostics to complete engine overhauls. Prices were kept reasonable because competition was everywhere.

Today, primary automobile maintenance is performed almost exclusively by factory-authorized service centers, also known as dealerships, at prices that clearly are not competition-driven.

Most open source projects actually demonstrate how product development must be managed in order for the software industry as a whole to graduate from its current chaotic state.

And the corner garages? They just change tires and oil now. To make matters worse, they compete with dealerships that offer those same services as loss leaders.

The reason for this change isn't that cars got more complicated. While it is true that cars *did* get more complicated, their sophistication was, and still is, managed at the dealerships by means of explicit service procedures and specialized diagnostic equipment—tools that would be equally effective if used at the corner garage.

The reason all those corner garages were reduced to tire and oil changers is because the major automobile manufacturers created "closed source" products (the electronic engine controller), and then controlled access to essential product information and training in order to create advantage for their own service centers. The major players in the software industry have clearly followed this same strategy, and have arrived at exactly the same conclusion: it works well for them.

Even the parallel in product release cycles is disturbingly similar. Consider this: in the auto industry, the typical way to stoke a bottom line when profits from existing products start to sag is to update the product line. By forcing the market to transition to this year's more profitable models, and phasing out support for previous models, the process renews itself.

Does any of this sound familiar? And don't get me started on the similarities between automobile leasing and the latest software distribution model: application servers.

The exit from this scenario, as proposed by the free software movement, is to do what the automobile and software industries currently won't do: offer basic freedoms to the end user, and give those freedoms "teeth" (source code, documentation, and unencumbered licenses) so that the user can exercise them by taking your product elsewhere when it makes sense in the user's mind to do so. When choice is truly an option, power doesn't consolidate unless users say it's okay first. And the decision is always revocable if the arrangement doesn't meet expectations.

Indeed, often only one or two controlling interests are behind all the dealerships in town. This isn't even permitted in some other industries like commercial radio.

Development

Mr. Murphy asserts that open source has little to offer engineering as a software construction model. Quite to the contrary, most open source projects actually demonstrate how product development must be managed in order for the software industry as a whole to graduate from its current chaotic state. The idea that open source would lead the way into the future of our industry is a profound one indeed.

To prove my point, consider the open source industry's most popular version management system, a product called CVS.[1] CVS's client-server capability and non-locking development model make it the perfect tool for large, geographically diverse teams, which has always been the way of business for open source development. As the software industry as a whole makes increasing use of outsourcing and telecommuting, it's good to know that the open source community is already there showing them how to do it.

But a large, distributed development effort needs more than version management. It needs automated bug tracking as well. For that, there's Jitterbug,[2] another free utility that allows users—or the software product itself—to submit bug reports via e-mail and have them automatically added to a database that can be viewed and managed with an HTML browser.

These examples show that if you want to see the next 10 years of software development practice, just look backwards at the last 15 years of the free software movement. And in case you don't want to set up and use these tools yourself, there's SourceForge, a free service that will supply them for you—but only if you're writing free software.[3]

Innovation

A free software innovation that is currently in the works is EL/IX, a Red Hat-sponsored project to specify a common Linux-derived RTOS API that can be implemented by any RTOS vendor, free or otherwise.[4]

Although counterintuitive at first, both open source and commercial RTOS vendors stand to gain a lot in the success of this effort, because their EL/IX-compliant products become drop-in replacements for other EL/IX-based components. This makes it easier for a vendor to unseat an established competitor in situations where their product is clearly a better choice.

But consumers of EL/IX-related products stand to gain the most from EL/IX, because the cost of changing an EL/IX-compliant component for another is significantly reduced. When EL/IX hits the streets, RTOS vendors will finally be forced to market themselves based on their true strengths and weaknesses, instead of winning designs by pedigree,

I participate in the free software movement because I don't need, nor do I want, the assistance of a "closed source" license in order to be successful.

because they won't have the inertia of the installed user base they have today.

Academia

Mr. Murphy seems concerned that the success of the open source movement spells the end of contributions to the free software community from academia and other nonprofit sources. Again, the evidence doesn't support his fears.

Much free software was, and still is, written and maintained by members of the academic community, including instructors, graduate students, and undergraduates. One of the most profound examples of this is the Coda File System, which is the heir-apparent to the popular Network File System (NFS) as an approach for distributed data management on an enterprise scale. Coda is free software.[5]

Although technically still under development, Coda has been deployed at Carnegie Mellon University and elsewhere, and is maintained by members of the computer science staff there. The number of graduate students whose master's degrees are based on their contributions to Coda isn't listed anywhere, but is likely a significant number.

Yet another example is Mosix, a novel clustering solution that lets networked Unix-like computers share processors and other resources without user or application intervention.[6] Like Coda, Mosix is in routine use although it is still under active development, and it is also a product of academia.

For something with more of an embedded influence, there's the LART project, a StrongArm-based system that runs Linux.[7] Among its strengths are that it offers about 250 MIPS on a measly one watt of power. And yes, schematics are included.

Free software morality

Unfortunately, I must agree with Mr. Murphy on one point—many free software and open source community members do mistakenly claim the moral high ground over their commercial brethren.

I participate in the free software movement because I don't need, nor do I want, the assistance of a "closed source" license in order to be successful—professionalism, skill, and proper marketing are enough. Releasing a product as free software demonstrates my commitment to that and isn't a morality issue at all.

My position on this subject is probably common, but my situation isn't—I don't have to convince a corporate boardroom to adjust their way of doing business. I only have to encourage a handful of clients, who are usually progressive enough to go along willingly after I explain the alternatives (which occasionally include finding another supplier). Go easy on the people that don't have such a handy lever to pull.

When you use open source software, you support those who wish to deal with you forthrightly and who allow you complete access to information needed to evaluate their strengths and weaknesses. When you choose free or open source software over a closed source alternative, you demonstrate your preference for vendors who want to portray their capabilities in the most unambiguous way possible—as source code.

If you think you gain a certain moral advantage by participating in this process, you've missed the point entirely. Free software isn't about giving users their rights back, it's about not taking them away in the first place!

Embedded code

Mr. Murphy states: "Encouraging embedded engineers to release code as Open Source rarely has benefit unless the hardware design is included."

This statement isn't altogether true, but it is clear that providing the source code for an OTP or masked-ROM embedded system is of little benefit.

Or is it? Advanced users and analysts could use the source code to track down bugs and compare features of competing products. And in cases of product obsolescence (planned or otherwise), that source code could also be used by motivated third parties to populate replacement devices.

Furthermore, new coders would get an example of how a particular problem was solved. So open source software also provides a valuable and inexpensive training opportunity. How can this not be of benefit?

Unfortunately, both the FSF and the Open Source Initiative are understandably short on ideas for how to protect user's rights related to embedded systems that *are* field upgradable, and I have personally run headlong into this problem in my own work.

Furthermore, the liability issues that arise from an end user's incorrect programming of a piece of critical hardware should not be underestimated. But some products, like factory automation equipment, are built with this explicit purpose in mind (save the "incorrect programming" part, perhaps), so working out these details in an embedded product is not without precedent nor indication of reward.

And finally, there's the debugging opportunity that showing your code to the world offers. End users of technical products are often technically competent, and I always get beneficial feedback and ideas for improvements after my code (embedded and otherwise) hits the 'Net.

OSE

– Conventional RTOS? ...sorry!



OSE THE NEW GENERATION RTOS!

SAFER
OSE is the world's only RTOS that is safety certified to the demanding specifications of IEC 61508 . OSE is also being certified according to the stringent DO 178-B.

FASTER
Streamlined for extreme reliability and speed, OSE's kernel and TCP/IP stacks blow away the competition. From the RTOS to the tools, OSE sets tomorrow's standards for small size and high performance.

TOUGHER
As the only fault-tolerant RTOS, OSE supports mission-critical real-time systems, allowing complete non-stop recovery from hardware and software failures AND hot swaps – critical for high-availability functionality.

LONGER-LASTING
OSE is heterogeneous, scalable, and distributable, protecting your investment by allowing your application to grow from one CPU to hundreds.

MORE POWERFUL
OSE's kernel offers automatic supervision, dynamic reconfiguration and integrated error handling, letting you focus on your core competency: designing applications.

SIMPLER
OSE's powerful ultra-efficient message-based architecture lets you write nearly every bit of application code using only eight system calls.

PROVEN
Millions of products worldwide are already taking advantage of OSE , including the top brands in telecommunications and process control. OSE is the RTOS of the future.

www.enea.com

ENEA OSE SYSTEMS
5949 SHERRY LANE , SUITE 625, DALLAS TX 75225. PHONE: 214-346-9339. FAX: 214-346-9344. EMAIL: info@enea.com

Okay, so if the free software model is so great, how does somebody practice it in a way that feeds a family?

Embedded Linux is bogus

Thanks, Niall, for saying that.

But seriously, we've seen all this before, with Windows CE, with Java, and countless ideas before them, including C. One would think that we would have learned to tune this stuff out already.

Embedded Linux is clearly a great idea in some situations, and a lousy choice in others. We're hearing so much about it now because some are trying to find an alternate means to market dominance than by the simplest, most obvious way: being better than the alternatives.

This tends to happen when the ratio of investment capital to good sense is what it is right now. As with all previous examples, in another six months or so Embedded Linux will assume its proper place *alongside*, and not in place of, other tools developers can use to solve problems. Until then, we'll just have to suffer through the normal progression of things.

The cost of free software

To the casual observer, free software can legally be had at no cost. But to the hardcore embedded users who need to tailor products before use, the cost of free software can't be ignored.

In my own experience, it is often about the same cost to download, learn, adapt, and use for the first time a free utility like newlib (a C runtime library)[8] as to purchase a similar commercial offering.

So why do I bother? First, note that I didn't include the "learn, adapt, and use" steps for the commercial offering. Those steps are often still necessary, and in many cases made more difficult by a lack of source code or experienced user community to consult for help.

Second, with open source software, the setup cost isn't repeated when I change processors or architectures, because free software tends to be more portable than commercial equivalents. So the time spent up front is really an investment that pays dividends on the next project.

And finally, the only way to make sure that future newlibs will be available is by showing an interest in the current one. That isn't always enough, but it is certainly a start. And when that's all you can offer, it's clearly better than nothing, and the true free software community will be grateful for it.

Making money

Okay, so if the free software model is so great, how does somebody practice it in a way that feeds a family? Here's my recipe.

First, find someone—hopefully your current employer—who wants you to perform a software technology service, say, develop an application. Embedded or otherwise, free or not, isn't important yet.

Second, gain the client's approval to build the product using free software. The product itself isn't required to be free for this step to work, but keep in mind that certain free software can only be used to construct free software. Check licenses carefully.

Next, build the product. Avoid skipping this step.

Finally, repeat this process starting at the first step, using your new skills as a developer of similar applications, and/or as someone skilled in tailoring and using the tools that went into the product. Season to taste.

This approach is working for me, anyway.

Free software

The truth is that the free software movement is a long overdue course correction that reverses the software technology industry's progression towards a state that holds the rights of software vendors in higher regard than the rights of software consumers.

Furthermore, products of the free software movement provide models that demonstrate how software should be designed, managed, and marketed in the coming years.

By my count, that's two heaping helpings of positive change. All for the price of, well, free software.

References

1. *www.sourcegear.com/CVS*
2. *samba.org.au/jitterbug*
3. *sourceforge.org*
4. *sourceware.cygnus.com/elix*
5. *www.coda.cs.cmu.edu*
6. *www.mosix.cs.huji.ac.il*
7. *www.lart.tudelft.nl*
8. *sourceware.cynus.com/newlib*

The Last Word

Niall Murphy responds

Mr. Gatliff has made some good points about the contribution that open source has made. However, I remain unswayed in my thoughts on its prospects for the future. In starting his argument with an analogy of car dealerships, he points out that the dealership model makes it difficult for the small corner garage to compete. This model is not in the best interests of consumers, but it is in the best interests of the large dealerships. Surely the point that should be derived from this is that big business serves consumer interest less well, and since open source has now become big business, we are not going to see them work to the advantage of the small guy as much as in the past.

One of the examples Mr. Gatliff uses for the future prospects of embedded Linux is the prospect of a common API for reduced versions of Linux. If Red Hat is the only player in the EL/IX game, then EL/IX has nothing to offer, since no one else is

I predict that in five years time most embedded developers will not find themselves using any more open source products than they do today....

going to be a guest at a Red Hat party. And why should they, they are all trying to innovate in ways that make their product different enough to be able to make money out of it. Did I say different? Oops, I meant to say proprietary. Unless the other major Linux players buy into EL/IX, it is more likely to cause fragmentation of Linux than to prevent it, and so far Red Hat has not convinced any major players to publicly come on board.

On the question of whether embedded software for custom hardware can be released as open source and whether there is an advantage for the consumer, Mr. Gatliff makes the valid point that having the source may allow a technical customer to track down a bug more quickly, or more accurately. This is a model that will find favor with few. Are we encouraging suppliers to push the onus of finding bugs onto the consumer, rather than leaving it with the supplier? I wonder where that will end. Maybe you do not need to document a device's behavior if the customer can merely read the source to figure out how it works. In any case the debate on releasing this kind of device-specific code is futile—in the real world it is not happening, mainly due to concerns of company secrecy and competition.

From Mr. Gatliff's description you would assume that the open source products mentioned were assumed to be of higher quality than the commercial equivalent, or that they preceded the commercial competition in some way. But if you look at the Web site for the Coda file system, you will find it described as an experimental file system. In more detailed documentation it states "the rough edges, which inevitably come with research systems, are slowly being smoothed out." While this is a straightforward description of a product in its early stages, it is apparent that this is not something you can depend on for industrial strength development.

Mosix is another product mentioned, but the latest version is 0.97.6. Can you take risks on a product which is not even considered a 1.0 release? These two cases underline a common problem with open source development. They have great difficulty getting releases out the door. Open source projects tend to be so engineering-driven and technology-oriented that the feature list grows endlessly. This leads to situations like Mozilla, which has been waiting two years for a release since it became open source. At the time of this writing the next release date is uncertain.

I will finish with a glance into my crystal ball. While I am genuinely thankful to the free software movement for the tools they have made available so far, I predict that in five years time most embedded developers will not find themselves using any more open source products than they do today, and the number of such products, which can reasonably compete with fee-based products, will not have significantly increased. **esp**

Niall Murphy has been writing software for user interfaces and medical systems for ten years. He is the author of Front Panel: Designing Software for Embedded User Interfaces. *Murphy's training and consulting business is based in Galway, Ireland. He welcomes feedback and can be reached at nmurphy@panelsoft.com.*

Bill Gatliff is an independent consultant specializing in free software tools and techniques for embedded systems development. He is a member of the Embedded Systems Conference Advisory Board and a frequent contributor to Embedded Systems Programming magazine. Bill welcomes comments via e-mail to bgat@open-widgets.com.



BIG FEATURES
SMALL FOOTPRINT



U S Software products are compatible with:

x86	M*Core
386/486PM	ColdFire
680x0/683xx,	MIPS
DragonBall EZ	C166
PowerPC	CR16
ARM	8096/196
StrongARM	68HC11
SH 1, 2, 3, 4	68HC16
SPARCLite	80251
i960	Z80/180
NEC V85x	

See our website for additional microprocessors we support.

With thousands of embedded applications and 25 years in the business, U S Software continues to build cutting-edge, yet compact development tools.

Our software has a history of acclaimed performance and features:

- Compact – Even with industry-leading innovations our small footprint saves you valuable space.
- Fast – Optimized design ensures quick, efficient performance.
- Customizable – Source code is included to give you complete control.*

In the beginning there was silicon. Then, there was U S Software. And throughout, our engineers have crafted software and provided technical support designed to ensure your success.

Stop by booth 1536 and learn about U S Software's new voice over IP technology

- **USNET®** Embedded TCP/IP protocols, Internet Access Package, SNMP and Embedded Webserver.
- **SUPERTASK!™** Multitasking RTOS development suite. ITRON 2.x/3.x API available.
- **USFILES®** Windows/DOS compatible file system with CompactFlash and CD-ROM support.
- **NETPEER™** Distributed networking platform with comprehensive desktop interface including GUI, database, multimedia and more.
- **SUPERPEG™** Embedded GUI for SuperTask!®
- **IRPRO™** IrDA protocol stack with SIR and FIR

*Netpeer™ is delivered as a library.

800-356-7097

WWW.USSW.COM

U S SOFTWARE®
EMBEDDED EXCELLENCE

7175 NW Evergreen Pkwy. Suite 100 Hillsboro, OR 97124
TEL: 503-844-6614 • FAX: 503-844-6480 • EMAIL: info@ussw.com • www.ussw.com

STEPHEN HARPSTER

Ready or Not: IPv6 is Coming

The World Wide Web is no longer just workstations, servers, and PCs. It's now also PDAs, cell phones, and automobiles. The old protocols of the Internet cannot handle these millions of devices. IPv6 is coming and you're going to have to be ready.

The Internet, and many smaller, private networks, use the Internet Protocol version 4, IPv4, as the network layer upon which other protocols, such as TCP, reside. Developed in the late '70s, it became apparent by the early '90s that IPv4 was reaching the end of its useful life.

The Internet Software Consortium's annual survey of Domain Name Servers indicates incredible growth on the Internet. The latest numbers represent an annual growth rate of over 65%, and that includes only those systems that are publicly advertised. Private nodes are not included in this count. In fact, it was the fear of running out of IP addresses that first prompted the look into a new version of IP.

Changes

The need for a new set of Internet protocols was first described in RFC 1287, "Towards the Future Internet Architecture," published in December 1991.[1] By early 1996, the first proposed standards for IP version 6, IPv6, were published.

Four major changes were made in version 6:

- Larger address space
- Simplified header
- Flows
- Authentication and privacy

Larger address space

The most highly publicized change is the larger address space. IPv4 addresses are 32 bits in length. The number of connected networks has continuously doubled

FIGURE 1 Internet hosts identified by DNS

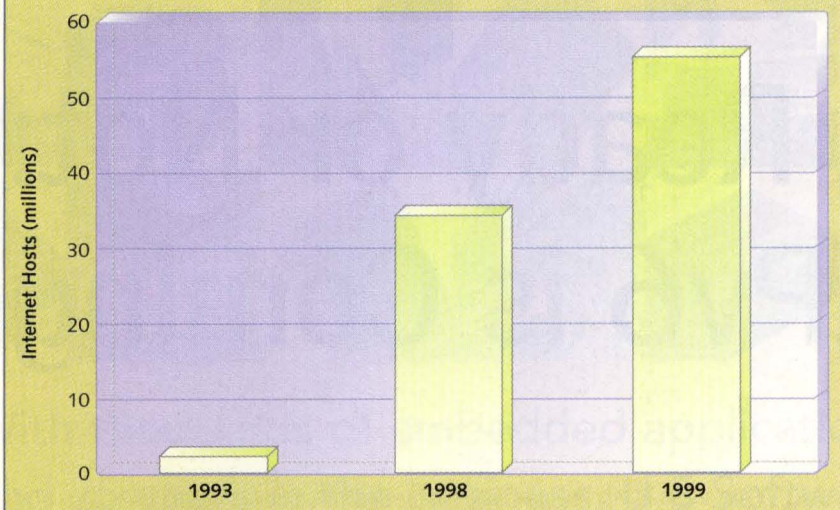


TABLE 1 IPv4 vs. IPv6: sockets API structures and constants

Feature	IPv4	IPv6	Comments
Protocol family. First argument in socket().	AF_INET PF_INET	AF_INET6 PF_INET6	Address and protocol families are used interchangeably
Address family	AF_INET	AF_INET6	In sockaddr family field
Inet address structure	struct sockaddr_in	struct sockaddr_in6	Note: sockaddr_in6 is larger than sockaddr
Generic address structure	struct sockaddr	struct sockaddr_storage	Only when used to allocate storage
IP address structure	struct in_addr	struct in6_addr	
Loopback address	INADDR_LOOPBACK	in6addr_loopback IN6ADDR_LOOPBACK_INIT	Constant can only be used for structure initialization
Wildcard address for binding listeners and receivers	INADDR_ANY	in6addr_any IN6ADDR_ANY_INIT	Constant can only be used for structure initialization
Name to address	gethostbyname()	getaddrinfo()	
Address to name	gethostbyaddr()	getnameinfo()	
Free data structures returned by getaddrinfo()		freeaddrinfo()	getaddrinfo() is multithread safe
Report errors		gai_strerror()	Errors from getaddrinfo() and getnameinfo()
String to address	inet_addr()	inet_pton()	Added family parameter
Address to string	inet_ntoa()	inet_ntop()	Added family parameter
Socket option for TTL	IP_TTL	IPv6_UNICAST_HOPS	Set TTL/hop limit
Get reserved port	rresvport()	rresvport_af()	Used by rcmd()
Execute a command on a remote host	rcmd()	rcmd_af()	Added family parameter

in less than 12 months for much of the past 10 to 15 years. With the explosion of Internet devices, PCs, PDAs, and even cell phones, the Internet is expected to run out of IPv4 address spaces sooner than later. In contrast, IPv6 has 128-bit addresses.

One way to understand how many addresses 128 bits represents is to think of a book that describes every possible Internet node address with 100 node descriptions per page. If each page was 0.1mm thick and was printed double-sided, the book to describe all possible IPv4 addresses would be 2,000 meters thick.

This may sound like a big book, but the book describing all possible IPv6 addresses would be $2x10^{16}$ light-years thick. To date, the farthest sight seen by the Hubble telescope is less than $2x10^{13}$ light-years.[2]

You may recall that IPv4 uses a "dotted decimal" notation. For example, 123.45.67.89. IPv6 addresses are instead represented by hexadecimal 16-bit quantities separated by colons, for example, FF02:0:0:0:0:1:200E:8C6C (with the exception noted below). A double colon (::) can be used as a type of shorthand for a string of zeroes. The address FF02::1:200E:8C6C is, therefore, equivalent to the previous IPv6 address. The double colon can only be used once in an address.

There is also the notion of an *IPv4-compatible address*, that has the 96 high-order bits set to zero. This can be represented as 0:0:0:0:0:0:0:123.45.67.89 or ::123.45.67.89.

Whenever there is a case where a colon would cause confusion, a literal IPv6 address can be quoted using square brackets. For example, a URL may look like http://[fec0::55:a00:20ff:fe90:58f8]:80/index.html.

Simplified header

IPv6 headers consist of eight equal length fields. This is in contrast to IPv4, which has variable length fields, some of which are optional. IPv6 handles options by placing them in sepa-

rate extension headers that follow the actual IP header. These changes make IPv6 packets more efficient to route since there are fewer fields to examine and process.

Flows

A "flow" is defined by RFC 1883 as a sequence of packets that require special handling by the intervening routers.[3] Flows may be used to control quality of service to support "real-time" applications like voice and video.

Authentication and privacy

Although security was considered when IPv4 was designed, e-commerce has stressed the need for built-in security. IPv6 uses two extension headers for security: the authentication header described in RFC 1826[4], and the encapsulating security payload described in RFC 1827[5].

The authentication header provides strong integrity and strong authentication for packets. The encapsulating security payload provides confidentiality by encrypting the payload of the packets.

Autoconfiguration

IPv6 addresses are longer than IPv4 addresses, and much more awkward for people to work with. One of the goals of the IPv6 standards was to simplify the management of addresses.

Toward that end, IPv6 offers autoconfiguration, which gives IPv6 machines plug-and-play connectivity. Two types of autoconfiguration are available: stateless and stateful.

With *stateless autoconfiguration*, hosts automatically compute their own IP address by combining the network prefix obtained by the local router with their own Ethernet MAC address. Stateless autoconfiguration is described in RFC 2462.[6]

Because some people view the use of a MAC address as an invasion of privacy, a new draft proposes a way for hosts to compute their own IP address based on a random number.[7] Using a

LISTING 1 Using sockaddr_in6

```
struct sockaddr_in6     sin;
struct hostent          *hp;
...
(void) memset((void *) &sin, sizeof(sin), 0);
sin.sin6_family = hp->h_addrtype;
sin.sin6_flowinfo = 0;
sin.sin6_port = htons(port);
(void) memcpy((void *) &sin.sin6_addr, (void *) &hp->h_addr, hp->h_length);
if (connect(sock, (struct sockaddr *) &sin, sizeof(sin)) == -1)
{
      ...
}
```

LISTING 2 Using a) gethostbyname() vs. b) getaddrinfo()

a)

```
int
myconnect(char *host, int port)
{
     struct sockaddr_in dest;
     struct hostent *hp;
     int ret;
     int sock;

     hp = gethostbyname(host);
     if (hp == NULL ||
        hp->h_addrtype != AF_INET ||
        hp->h_length != 4 {
            /* handle error */
     }
     dest.sin_family = AF_INET;
     dest.sin_port = htons(port);
     bcopy(hp->h_addr,
            &dest.sin)_addr, 4);
     sock = socket(AF_INET,
            SOCK_STREAM, 0);
     if (sock == -1) {
            /* handle error */
     }
     ret = connect(sock,
            (struct sockaddr *)
             &dest, sizeof(dest));
     if (ret == -1) {
            /* handle error */
     }
     return(sock)
}
```

b)

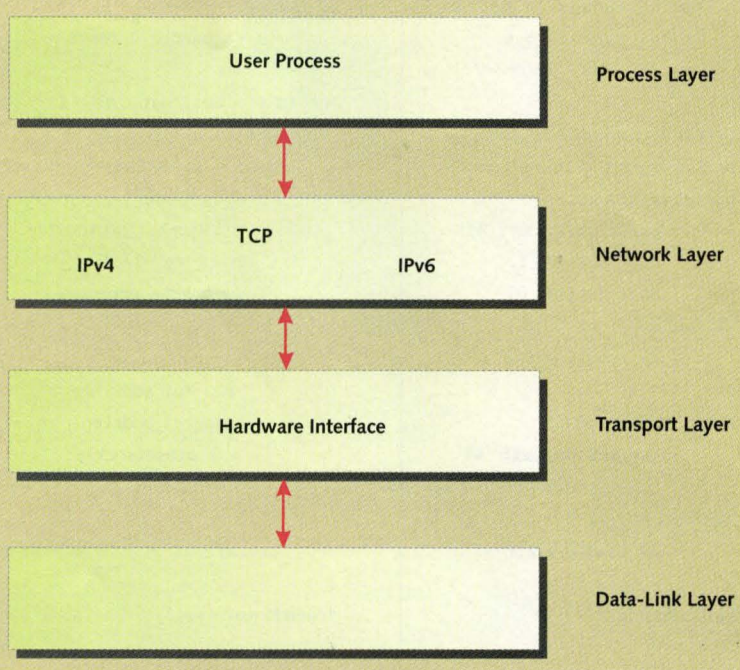
```
int
myconnect(char *host, int port)
{
     struct addrinfo *res;
     struct addrinfo *aip;
     int error;
     int sock;

     error = getaddrinfo(node,
              service, NULL, &res);
     if (error != 0) {
              /* handle error */
     }
     for (aip = res; aip != NULL;
          aip = aip->ai_next) {
          sock = socket(aip->ai_family,
                  aip->ai_socktype,
                  aip->ai_protocol);
          if (sock == -1) {
             /* handle error */
           }
          if (connect(sock,
              aip->ai_addr,
              aip->ai_addrlen) == -1) {
                 close(sock);
                 continue;
          }
          break;
     }
     freeaddrinfo(res);
     return(sock);
}
```

LISTING 3 Using inet_ntop()

```
new_sock = accept(sock, (struct sockaddr *) &faddr, &faddrlen);
if (IN6_IS_ADDR_V4MAPPED(&faddr.sin6_addr))
{
     /*
      * Print an IPv4 address.
      */
     struct in_addr ina;
     IN6_V4MAPPED_TO_INADDR(&faddr.sin6_addr, &ina);
     (void) printf("Connection from %s/%d\n",
       inet_ntop(AF_INET, (void *) &ina, addrbuf, sizeof(addrbuf)),
       ntohs(faddr.sin6_port));
}
else
{
     /*
      * Print an IPv6 address.
      */
     (void) printf("Connection from %s/%d\n",
       inet_ntop(AF_INET6, (void *) &faddr.sin6_addr, addrbuf, sizeof(addrbuf)),
       ntohs(faddr.sin6_port));
}
```

FIGURE 2 Dual-stack protocol stack



random number makes it harder for eavesdroppers and other information collectors to identify machines.

Stateful autoconfiguration is done with the DHCPv6 protocol.[8,9] This is similar to the current DHCP protocol that is used to parcel out addresses and service information to individual machines.

One advantage of using DHCPv6, instead of stateless autoconfiguration, is that DHCPv6 gives you the ability to configure dynamic updates to DNS. Another advantage is security. With stateless autoconfiguration, any node can connect to your network and start running. With DHCPv6 servers, you can maintain tighter control over which nodes can be configured for the network. The only nodes that can be configured through the DHCP server are those that you have explicitly authorized.

Converting from IPv4

RFC 2553, which includes portions of POSIX 1003.1g, defines the new sockets API for IPv6.[10,11] The good news is that the new API is protocol-independent. It doesn't matter if your protocol is IPv4, IPv6, or IPv*n*—the same code will work with no changes.

This means you can code for IPv6 today, even if you still only use IPv4! The API, and your application, will still work. When you enable IPv6 in your network, the applications should continue to run with no changes.

The changes described in RFC 2553 for IPv6 cover four areas:

- Core socket functions
- Address data structures
- Name-to-address translation functions
- Address conversion functions

Core socket functions

The original core socket functions, those that set up and tear down connections and send and receive packets, were designed to be protocol-independent. These functions do not have to change for IPv6, but they do require


COMPETENCE
180C
Send Mail: Building a Gateway is that simple
Send
Quote
Address
Attach
Options
Spelling
Save
Security
Stop
To
NEO@matrix.com
Priority:
Highest
Subject:
Building a Gateway is that simple
Dear NEO,
You've heard all the TALK.. But the truth is..
There is ONLY ONE OSGi compliant embedded server software designed for Gateways that has "working" JINI implementation and UPnP and WAP and DIRECTORY service for security and network management
So ONLY ONE Question Remains.. Why haven't you called ProSyst?
If you are in the game of device automation then we have the key to the Gateway.
Its simple. Its all we do, and the ONLY ONE who does it RIGHT..NOW
Visit www.prosyst.com or call us 678-366-4607
Regards,
Keith
Sign up for our Early Mover Program and get free development tools!
See a demonstration at the Embedded Systems show San Jose Sept 25-27!
ProSyst USA, Inc.
555 North Point Center East
4th Floor, Suite 450
Alpharetta, GA 30022
http://www.prosyst.com
Keith Sciulli
Marketing Director
Phone 678 366 4607
Fax 678 366 4655
k.sciulli@prosyst.com
OSGi
ProSyst
www.prosyst.com

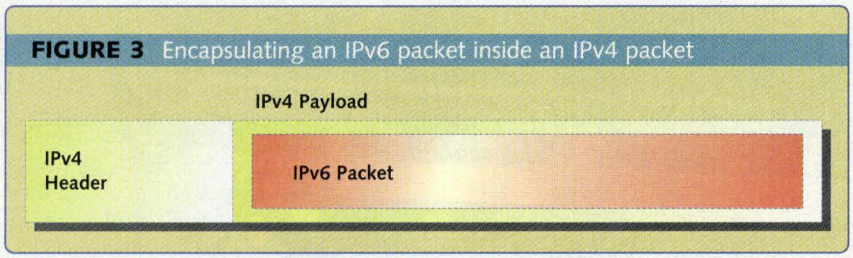


FIGURE 3 Encapsulating an IPv6 packet inside an IPv4 packet

RTOS Win32® with REAL realtime

Phar Lap Software™

making our mark on the embedded WORLD

TNT EMBEDDED TOOLSUITE® 10.0

- Edit, compile, link and debug with MS Developer Studio
- TCP/IP - Winsock API featuring SNMP, WinInet and MicroWeb Server
- Knowledgeable technical support
- Comprehensive engineering services and training programs
- Realtime GUI
- Extensive sample code and printed documentation
- Small footprint

Call for a FREE GUI demo

TEL: (617) 661-1510
FAX: (617) 876-2972
www.pharlap.com
info@pharlap.com

Phar Lap Software, Inc.
60 Aberdeen Ave.
Cambridge, MA 02138

©2000 Phar Lap Software and TNT Embedded ToolSuite are registered trademarks of Phar Lap Software, Inc.
Win32 is a registered trademark of Microsoft Corporation

some new IPv6 data structures, which are described in the next section.

Note that this applies only if you are using a sockets API, which may not be the case in an embedded system. Similarly, the application code changes I'm about to describe will only make sense if you're using a sockets or sockets-like networking API. If you are using such an API, you may be able to take advantage of a "socket scrubber" developed at Sun (*www.sun.com/software/solaris/ipv6*).

Address data structures

A protocol-specific data structure is defined for each protocol that the socket functions support. You normally cast the protocol-specific data structure to `sockaddr` when calling the core socket functions. For example, in IPv4, the protocol-specific data structure is `sockaddr_in` which you cast to `sockaddr` when calling `connect()`. The protocol-specific data structure for IPv6 is `sockaddr_in6`.

IPv6 also has a new address family, `AF_INET6`, a new protocol family, `PF_INET6`, and a new structure for holding a single IPv6 address, `in6_addr`. These are used in a way that is similar to IPv4's `AF_INET`, `PF_INET`, and `in_addr`.

Name-to-address translation functions

The most common function for translating names to addresses in IPv4 is `gethostbyname()`. It's still retained for backwards compatibility, but is inadequate for IPv6. A common complaint has been that `gethostbyname()` is not thread-safe, but a bigger problem is that there's no way to specify anything about the types of addresses desired (IPv4, IPv6, and so on). Adopted from POSIX 1003.1g, `getaddrinfo()` solves these problems.

There's also a function to perform the reverse. Given a node name, `getnameinfo()` looks up an IP address.

Address conversion functions

Typically, the functions `inet_addr()` and `inet_ntoa()` are used to convert

Make Your Embedded Platform Soar!

with Treck Inc.

Real-Time Internet Protocols for Embedded Systems

We know that schedules are always tight, and that you do not have time to "port" Internet protocols into your embedded environment (let alone write them from scratch). That does not stop marketing people inside your company from asking for Internet functionality in your new product line that is rolling out in the next few weeks. Now you have someone to turn to.

High quality, high performance Internet protocols are what makes us the #1 choice for designers of embedded systems who need Internet connectivity. We design and implement our products specifically for embedded systems. That is why we are three times faster than our fastest competitor. Of course our Internet protocols do not require any porting to your specific platform. This means that you will never need to change a single line of our ANSI "C" code. You do not need an RTOS to use our Internet protocols, but if you have one, we will certainly work with it.

We are very committed to supporting you in your development effort. We know that you do not have time to wait for someone to call days later to get a support answer. That is why we provide the "finest" support in the industry. You can always count on Treck Inc. to help you every step of the way.

Our base product (Treck Real-Time TCP/IP), includes Sockets, TCP, UDP, ICMP, IP, Ethernet, ARP, SLIP and Ping. Treck also provides support for PPP, DHCP, BootP, FTP, and Telnet. All of our Internet products arrive at your door in source code form and are always royalty free.

Call us today to see how Treck Inc. can help you!

Treck Inc.

(800)340-6648 or (513)688-0553

or visit our web page at www.treck.com



CIRCLE # 27 ON READER SERVICE CARD

In reality, "dual-stack" is a misnomer. Actually, only one IP module handles both IPv4 and IPv6.

an IPv4 address between binary and text form. The analogous IPv6 functions are called `inet_pton()` and `inet_ntop()`. `inet_pton()` converts a text address to its binary equivalent; `inet_ntop()` does the reverse, converting a binary address to printable text. Since both of these functions have an parameter for specifying the address family, they can be used to convert both IPv4 and IPv6 addresses.

6bone

The 6bone is an IPv6 testbed. It's run as a world-wide informal collaborative project with oversight from the NGtrans (IPv6 Transition) working group of the IETF. There are 462 sites in 42 countries.

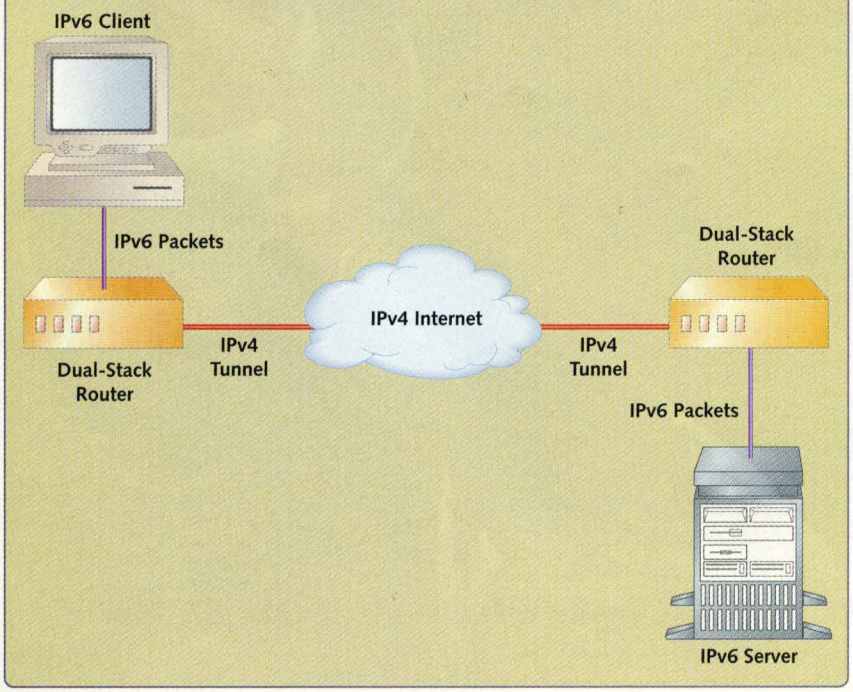
You can use the 6bone for testing purposes, or to gain experience with IPv6 before releasing your product commercially. To join the 6bone, you will need the following:

- A router that supports IPv6
- A 6bone site you can connect to
- A nameserver that support IPv6 AAAA records

You can find more information about the 6bone at *www.6bone.net*.

The easiest way to connect a single workstation to the 6bone is to use Freenet6 by Viagènie. Freenet6 is a server that will generate a configuration for you to download for your specific operating system. After downloading and installing the configuration, you're on the 6bone. See *www.freenet6.net* for more information.

FIGURE 4 A tunnel



The state of IPv6 today

When the IETF started working on IPv6, one of the requirements was to allow for a slow migration from IPv4 to IPv6. Obviously, an abrupt cutover from one protocol to another would be impossible. To make IPv6 successful, it would have to work on the Internet simultaneously with IPv4. RFC 1933, "Transition Mechanisms for IPv6 Hosts and Routers," defines several methods for deploying IPv6 devices.[13]

Dual-stack nodes

A dual-stack node supports both IPv4 and IPv6 simultaneously. When communicating with an IPv6 node, or another dual-stack node, the dual-stack node will use IPv6. When communicating with an IPv4 node, the dual-stack node uses IPv4. Both protocols are handled gracefully, unbeknownst to the applications.

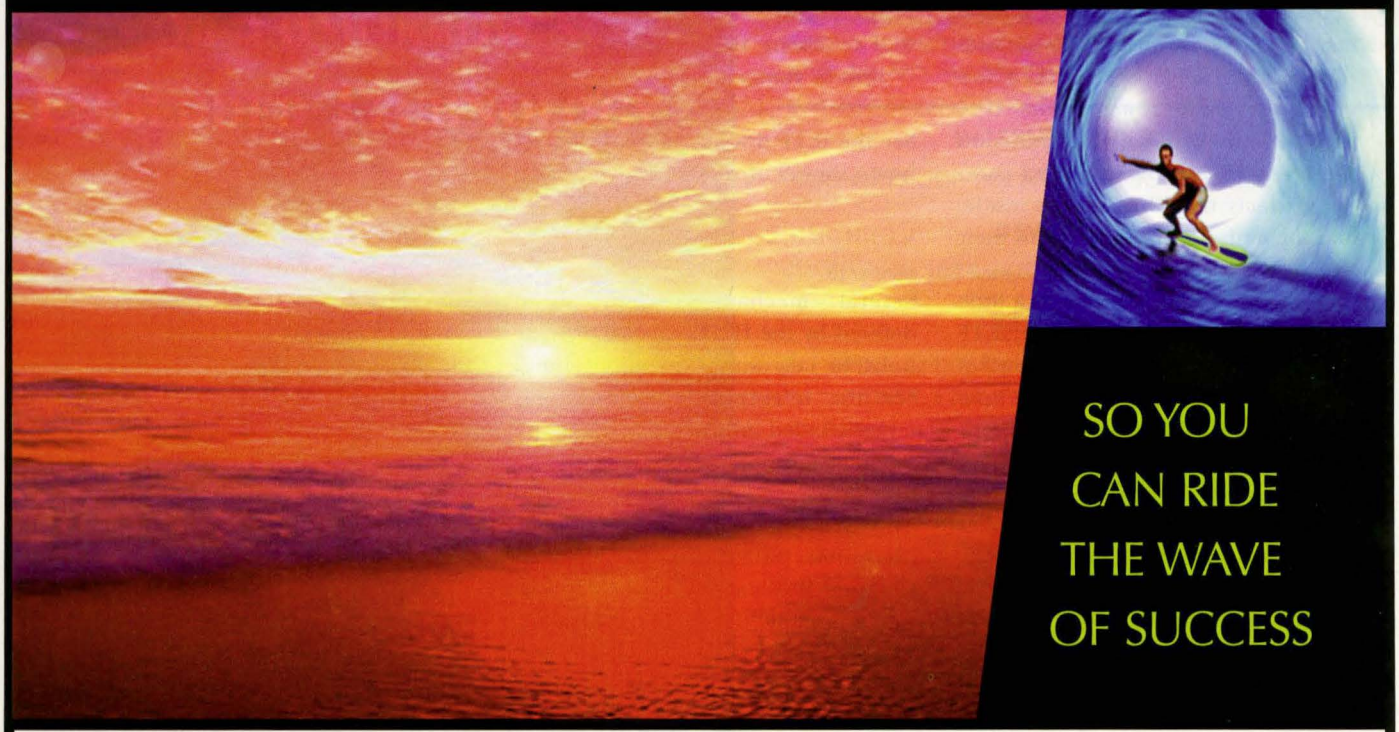
In reality, "dual-stack" is a misnomer. Actually, only one IP module handles both IPv4 and IPv6. The socket layer, based on the AF_INET or AF_INET6 parameter, knows which protocol is being used. If IPv4 is being used, it constructs an IPv4-mapped address, where the 80 high-order bits are zero, the next 16 bits are 0xffff, and the 32 low-order bits are the IPv4 address, for example, ::FFFF:123.45.67.89. Thus, the IP module always receives a 128-bit address from the upper layer.

When the IP module receives an IPv4-mapped address, it assembles the IPv4 headers. When it receives a non-IPv4-mapped address, it assembles the IPv6 headers. Either way, the same module does the work of both protocols, so there really is only one stack.

Tunnels

What happens if two IPv6 nodes are separated by an IPv4 network? How do they communicate? By using dual-stack routers, a *tunnel* can be dug across the IPv4 network. Tunnels are dug by encapsulating an IP packet within the payload of another packet.

WE CALM THE SEAS OF TECHNOLOGY



SO YOU CAN RIDE THE WAVE OF SUCCESS

WHY PACIFIC SOFTWORKS ?

- **Protocols:** TCP/IP, UDP/IP, DHCP, BOOTP
- **Applications:** FTP, TFTP, TELNET, DNS, SMTP, POP3, RPC
- **Internet:** PPP, SLIP, CSLIP, CHAP, PAP
- **Web:** Web Micro Browser, Web Server, Graphics Toolkit/GUI
- **Network Management:** SNMP v1/v2/v3, MIB Compiler
- **Routing:** RIP/RIP2, OSPF, BGP4, NAT
- **Security:** SSL Support
- **RTOS:** ThreadX™

Since 1982, Pacific Softworks has been delivering the quickest and most stable Internet Protocols available in the industry. We're proud of our reputation as the leading supplier of protocol software.

Royalty Free or Royalty based licensing
End-to-End support from the Protocol Specialists
Ported and Tested Compatibility with any Processor, Kernel or RTOS
Flexible and Scalable solutions to protect your investment
Demonstrated reduction of time to market
Leading the industry with next generation products and solutions
World-wide presence
...and, most importantly, we have Happy Customers!

Visit our website for more in-depth information on Fusion, the "off-the-shelf" FastTrack and other leading embedded solutions. www.pacificsw.com , e-mail: sales@pacificsw.com

Pacific Softworks provides a single supplier solution ensuring that today's successes are the building blocks for tomorrow.

California Corporate Headquarters:
tel: 800.541.9508 fax: 805.499.5512
European Headquarters:
tel: 44.1494.432.735 fax: 44.1494.432.728
Japan/Asia-Pacific Headquarters:
tel: 81.3.5669.7722 fax: 81.3.5669.7723

Pacific Softworks
our wave... your Success.

Visit us at Embedded Systems Conference 2000, Booth# 242

The dual-stack router on one end of the communication takes IPv6 packets from the sender, encapsulates them within IPv4 packets, then forwards the packets across the IPv4 Internet. Another dual-stack router on the other side receives the IPv4 packets, extracts the IPv6 packets inside, and forwards the IPv6 packets to their proper destination.

Configuring tunnels can be a little tricky. To make life easier, the IETF draft, "IPv6 Tunnel Broker," describes a way to automatically manage tunnels. A tunnel broker is a dual-stack machine that creates, modifies, or deletes tunnels for you. To dig a tunnel, you simply give the tunnel broker the following information:

- The IPv4 address of the tunnel's client side
- A nickname to be used for the DNS registration of the global IPv6 addresses assigned to both sides of the tunnel
- The client function: standalone host or router

See the IETF draft or more details on how a tunnel broker works.[14]

Communicating with IPv4 machines

Several mechanisms have been defined by the IETF for IPv6 applications to talk with IPv4 applications. See the referenced documents for more detailed information.

SIIT[15]

The Stateless IP/ICMP Translation Algorithm (SIIT) translates between IPv4 and IPv6 packet headers in separate translator "boxes" in the network.

NAT-PT[16]

A combination of Network Address Translation (NAT) and Protocol Translation (PT) can be used to provide transparent routing. This does not mandate dual-stacks or special purpose routing requirements on end nodes.

TCP/UDP Relay[17]

This mechanism translates packets at the TCP layer. A transport relay translator box sits between an IPv6-only node and an IPv4-only node, handling the protocol translation.

DSTM[18]

The Dual Stack Transition Mechanism (DSTM) provides a method to assign temporary global IPv4 addresses to IPv6 nodes. It also uses dynamic tunnels within an IPv6 network to carry IPv4 traffic.

6to4[19]

This mechanism uses an IPv4 network

development tools
C/C++/EC++ Compilers
True Time Simulator
Real Time Debugger
I/O Simulation and Stimulation
Real Time Kernel
Peripheral Builder
BDI BDM / JTAG Interfaces
Motorola
HC05, HC08, HC11, HC12, HC16, M•CORE, M68xxx/3xx, PowerPC
Philips
8051XA
STMicroelectronics
ST7, ST16, ST19
NEW! PANTA
INTUITIVE USER INTERFACE
HIWARE
Innovation for your Success
is now
metrowerks
A Motorola Company
EMBEDDED SYSTEMS CONFERENCE
Sept 26-28, San Jose, CA
visit us at booth #436
Hiware USA
5808 Brown Rock Trail
Austin, TX 78749
USA
Ph. (512) 301 4500
Fax (512) 301 0957
info_us@hiware.com
Hiware
Riehenring 175
4058 Basel
Switzerland
Ph. +41 61 690 7500
Fax +41 61 690 7501
info@hiware.com
Hiware France
1 Rue de la Haye Le Dôme
95731 Roissy CDG Cedex
France
Ph. +33 1 41 51 19 72
Fax +33 1 41 51 19 73
info_fr@hiware.com
www.metrowerks.com
www.hiware.com

SOLUTIONS TODAY... FOR TOMORROW'S EMBEDDED TECHNOLOGY

Let EBSnet help you build your next embedded device!

- All Source Code is Provided
- 100% ANSI "C"
- Royalty Free
- Competitive Prices

Prompt Competent Support

- Consulting
- Satisfaction Guaranteed!

For More Information, Visit Our Web Site

www.ebsnetinc.com

1-800-428-9340

P.O. Box 873 • 39 Court Street
Groton, MA 01450
Phone: 978 448 9340 • Fax: 978 448 6376

http://www.ebsnetinc.com • email: sales@ebsnetinc .com

EMBEDDED TCP-IP NETWORK STACK

The EBSnet Cross Development System provides comprehensive TCP/IP protocols and network applications for embedded CPUs. RTIP 3.0-MPC provides support for Motorola microprocessors including PowerPC, MPC860, 68K, 68360, and Coldfire. Systems are also available for 808x, 80186, 386, ARM, SparcLite, Mips, Hitachi, Mitsubishi, Philips, and Infineon.

INTERNET PROTOCOLS

SUPPORTS UDP, TCP, ARP, RARP, BOOTP, ICMP, IGMP, DNS

DEVICE DRIVERS

Ethernet, 100 Base-T, Uart, PCMCIA based, PCI based

KERNEL DRIVERS

AMX®, CMX®, SMX®, Nucleus®, RtKernel®, Pharlap®, TNT®, ETS®, TNTRT®, PSOS®, RTXC®, RTPX, Polled (no kernel), UCOS®, ECOS®, VRTX®, Threadx®

PORTING LAYER

Ports easily to any realtime OS/CPU

DIAL-UP

SLIP, CSLIP, PPP, modem support

APPLICATION PROTOCOLS

SNMP, DHCP, NFS, FTP, TFTP, Telnet, Mail (SMTP, POP3, IM AP), RIP, and Virtual/Memory File Systems

WEB SERVER

Full featured embedded Web Server (with diskless option)

EMBEDDED WEB BROWSER

WEBSTER, EBSnet's Embedded Web Client, is a complete tool for developing WEB browser interfaces for interactive internet appliances and other embedded systems that wish to use HTML to present a user interface.

WEBSTER is written in portable C++, will port to any target that can support the PEG Graphics Library and uses standard socket calls to access the network.

WEBSTER is compatible with EBSnet's RTIP but can be standalone.

ERTFS-DOS COMPATIBLE FILE SYSTEM

Comprehensive, portable, high performance DOS compatible file I/O for embedded systems.

ebSnet Inc

Y2K

as a unicast point-to-point link layer. IPv4 sites are assigned a unique IPv6 address prefix. Packets are then encapsulated using this prefix and transmitted over IPv4. No configured tunnels are required. 6to4 is typically implemented almost entirely in border routers.

6over4[20]

This RFC allows isolated IPv6 nodes, located on a physical link that has no directly connected IPv6 router, to run IPv6 by using an IPv4 multicast domain as their virtual local link. The IPv6 hosts do not require IPv4-compatible addresses or configured tunnels in this approach.

Do your systems need to support both IPv4 and IPv6, or just IPv6? That depends. If your systems will run in a private network, IPv6 alone is enough. The extra memory required by the IPv4 portion of the stack can be eliminated if your systems will never talk to IPv4-only applications. Even if your applications connect to the Internet, they can use tunnels to communicate with other IPv6-only applications. The dual-stack can be relegated to the routers connecting your private network to the Internet.

In practice, the memory consumed by the IPv4 portion of the stack is relatively small, approximately 15KB for most implementations. Including it makes a nice insurance policy, especially since the majority of the Internet will still be IPv4 for a while. With the dual-stack approach, you will be guaranteed that your applications will work today and tomorrow, regardless of the mix of IPv4 and IPv6 machines on the network.

Not if, but when

The big question these days is not if IPv6 will be widely deployed, but when. IPv6 is already being rolled out in Japan and parts of Europe, where IPv4 addresses are increasingly difficult to obtain. In North America, where IPv4 addresses remain more plentiful, IPv6 will deploy more slowly. Nevertheless, it's coming in the next few years. Will your systems be ready for the change? **esp**

Stephen Harpster is the engineering manager for IPv6 at Sun Microsystems Inc. He earned his BS in computer science at the Georgia Institute of Technology in 1982, and his MS in computer science/electrical engineering at Northwestern University in 1987. You can reach him via e-mail at stephen.harpster@sun.com.

References

All of the RFCs and drafts can be retrieved for free from the IETF at *www.ietf.org*.

1. RFC 1287, "Towards the Future Internet Architecture," D. Clark, L. Chapin, V. Cerf, R. Braden, and R. Hobby, December 1991.
2. IPv6 and the Future of the Internet, Sun Microsystems, Inc., 1999. Available at *www.sun.com/software/white-papers/wp-ipv6/ipv6wp.pdf*.
3. RFC 1883, "Internet Protocol, Version 6 (IPv6) Specification," S. Deering and R. Hinden, December 1995.
4. RFC 1826, "IP Authentication Header," R. Atkinson, August 1995.
5. RFC 1827, "IP Encapsulating Security Payload (ESP)," R. Atkinson, August 1995.
6. RFC 2462, "IPv6 Stateless Address Autoconfiguration," S. Thomson and T. Narten, December 1998.
7. Draft-ietf-ipngwg-addrconf-privacy-01, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6," T. Narten and R. Draves, October 1999.
8. Draft-ietf-dhc-dhcpv6-15, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," J. Bound, M. Carney, and C. Perkins, May 2000.
9. Draft-ietf-dhc-dhcpv6exts-12, "Extensions for the Dynamic Host Configuration Protocol for IPv6," J. Bound, M. Carney, and C. Perkins, May 2000.
10. RFC 2553, "Basic Socket Interface Extensions for IPv6," R. Gilligan, S. Thomson, J. Bound, and W. Stevens, March 1999.
11. IEEE, "Protocol Independent Interfaces," IEEE Std. 1003.1g, 1997.
12. Porting Networking Applications to the IPv6 APIs, Sun Microsystems, Inc., October 1999. Available at *www.sun.com/software/solaris/ipv6/porting_guide_ipv6.pdf*.
13. RFC 1933, "Transition Mechanisms for IPv6 Hosts and Routers," R. Gilligan and E. Nordmark, April 1996.
14. Draft-ietf-ngtrans-broker-02, "IPv6 Tunnel Broker," A. Durand, P. Fasano, I. Guardini, and D. Lento, October 1999.
15. RFC 2765, "Stateless IP/ICMP Translation Algorithm (SIIT)," E. Nordmark, February 2000.
16. RFC 2766, "Network Address Translation – Protocol Translation (NAT-PT)," G. Tsirtsis and P. Srisuresh, February 2000.
17. Draft-ietf-ngtrans-tcpudp-relay-00, "An IPv6-to-IPv4 Transport Relay Translator," J. Hagino and K. Yamamoto, January 2000.
18. Draft-ietf-ngtrans-dstm-01, "Dual Stack Transition Mechanism (DSTM)," J. Bound, L. Toutain, and H. Afifi, March 2000.
19. Draft-ietf-ngtrans-6to4-05.txt, "Connection of IPv6 Domains via IPv4 Clouds without Explicit Tunnels," B. Carpenter and K. Moore, May 2000.
20. RFC 2529, "Transmission of IPv6 over IPv4 Domains without Explicit Tunnels, " B. Carpenter and C. Jung, March 1999.

Additional IPv6 resources

This set of web pages provides good generic information on IPv6:
playground.sun.com/pub/ipng/html/

This document outlines the business and technical case for IPv6:
www.ietf.org/internet-drafts/draft-iab-case-for-ipv6-05.txt

This is a geographic map that gives an idea of worldwide deployment:
noc.ntua.gr/~adamo/6bone/

The IPv6 Forum is a world-wide consortium of Internet vendors:
www.ipv6forum.com/

These are slides from a good tutorial:
www.viagenie.qc.ca/en/ipv6forum.pdf



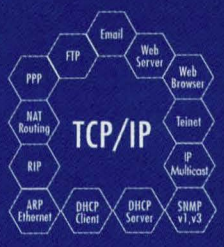
Without a truly complete and modular embedded solution you might not be as competitive as you think.

PROVEN SOLUTIONS: Market needs and competitive situations will continue to change. That's why we have provided solutions over the last ten years that are modular so you can quickly adapt and easily meet these challenges with a minimum of engineering overhead.

interniche
technologies, inc.

Why work with a multitude of companies for your deeply embedded software needs? Interniche solutions include everything you need for building Internet connectivity, embedded Web and DHCP servers, Browsers, routing and more. Today's design needs demand a broad range of solutions. Solutions that are CPU and RTOS independent, and modular so the design-in costs are low.

What's more, we offer more royalty free applications than anyone else. And our solutions are supported by the engineers who write them. That adds up to truly complete solutions. Solutions that let you go the distance.
So follow the lead of companies like ARM, Intel, Ericsson and Green Hills Software, Inc. and design in Interniche solutions today.



Solutions for Building Embedded Web & DHCP Servers, Routing & more.

For more information call 1-408-257-8014. Email sales@iniche.com or go to iniche.com today. In Europe call 46-40-45-85-49.

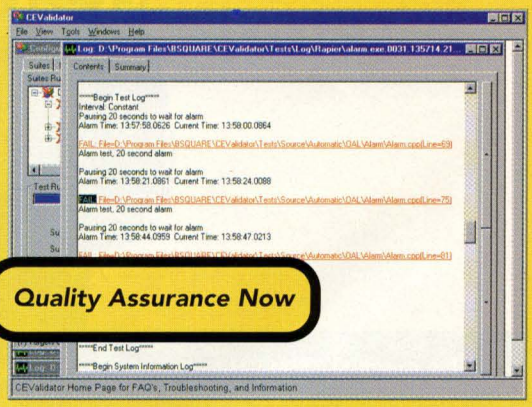


For every era there's product testing

BSQUARE's QA products and services make your product testing painless.

Put your quality assurance testing on autopilot with BSQUARE's CEValidator™ family of test automation products. Run and record systematic and comprehensive sets of tests for any Windows® CE-based project with the click of a mouse. Complete, thorough testing is at your fingertips. Want us to test it for you? BSQUARE is also your source for custom testing and QA packaged services for all your Windows-based development needs. The QA process has never been faster, easier–or safer.

BSQUARE has helped companies like Motorola®, JVC® and Symbol® Technologies get great products to market fast. **Test us out!**
Download free evaluation versions at www.bsquare.com/qa



Quality Assurance Now

© 2000 BSQUARE Corporation. All rights reserved. BSQUARE is a registered trademark of BSQUARE Corporation.
All other names, product names, and tradenames are trademarks or registered trademarks of their respective holders.

BSQUARE Solutions

- Platform development and connectivity
- QA tools and services
- OS licensing and tools, training, end-user software
- Engineering services and support
- Solutions for set-top boxes, Web pads, Windows-based terminals, mobile & wireless devices and more

Innovating tomorrow's devices™

bSQUARE

www.bsquare.com



Embedded Internet Tools

TCP/IP stack

CMX-MicroNet is a TCP/IP stack that will work with virtually all 8- and 16-bit processors. It uses between 3.5KB and 17KB of ROM, depending on the processor and implementation. Users can choose between single and multiple socket implementations. CMX-MicroNet can operate by itself or in conjunction with an RTOS. It supports TCP, PPP, UDP, SLIP, IP, and HTTP Web server. It starts at $5,500 and is available now.

CMX Systems
Framingham, MA
(508) 872-7675
www.cmx.com

Java accelerator

JVX is a Java accelerator designed to work with any processor, microcontroller, or RTOS, as well as with a Java Virtual Machine (JVM). It consists of hardware and software semiconductor IP technology that can be integrated into new or existing mobile platforms. It requires 15,000 logic gates and is portable to any semiconductor design environment. Without necessitating changes to the microcomputer instruction set or system architecture, the JVX technology interfaces to the processor in existing systems, preserving all the capabilities of the original system and adding the ability to execute Java code at near compiled speeds. It interfaces with the JVM either through a JIT interface or within the JVM's interpreter loop. This allows the CPU to switch between execution of Java and native code transparently.

InSilicon
San Jose, CA
(408) 894-1900
www.in-silicon.com

Database manager

The Empress RDBMS, a database management system, now supports BlueCat Linux. It places no limits on storage capacity and is able to store and retrieve almost any type of data, from microsecond timestamp to variable-length text to image and sound. The system is also configurable, offering options such as in-memory configuration and read-only file systems. The entire Empress SQL engine, contained on a single floppy, can run entirely in-memory. A second floppy contains the BlueCat Linux OS. Pricing varies based on the platform, the number of processors, and the number of users. A free demo is available for download at the company's Web site.

Empress Software
Greenbelt, MD
(301) 220-1919
www.empress.com

Real-time Linux networking

RTnet is a modification of the Linux networking subsystem that provides real-time networking over standard IP networks. It supports the IP, ICMP, and UDP protocols; it provides a sockets implementation for use by real-time tasks; it uses a programmable interface that is nearly identical to the standard sockets interface; and it works with Linux 2.2x kernels in connection with either RTAI v. 1.3 or RTLinux v. 2.3. It is available now at no cost at the company's Web site. Although the company has done thorough testing, they say that it should still be thought of as a beta version.

Lineo
Lindon, UT
(801) 426-5001
www.lineo.com

Java application environment

OSE Personal JRuntime is an application environment designed for building network-connectable applications for consumer devices. It enables designers to use Java in a manner similar to C/C++, while taking advantage of Java capabilities. The OSE Personal JRuntime includes a number of class library packages and an implementation of the Java Virtual Machine that spawns the Java application threads. The Java Virtual Machine ensures that code runs across a variety of operating systems and microprocessors. In addition, a Java-OSE API will handle the interaction between Java threads and OSE processes. OSE Personal JRuntime is available now.

Enea OSE Systems
Dallas, TX
(214) 346-9339
www.enea.com

Embedded CAM family

The first product in a family of embedded Content Addressable Memories (CAMs) is Custom-Touch NetCAM, a compiler that generates ternary CAMs that can be embedded into IC designs for use in network applications such as routers and switches. The NetCAM compiler provides characterized timing and power information, which enables designers to make trade-offs during the design cycle based on their needs. CAMs generated by the NetCAM compiler are

equipped for pattern searches. They have sizes ranging from 1KB to 32KB and word widths up to 64 bits. NetCAM is available now. It cost $200,000 for a single configuration package and $750,000 for a single, node-locked compiler license that can generate any configuration.

Virage Logic
Fremont, CA
(877) 360-6690
www.viragelogic.com

TCP/IP stack

Nucleus NET v. 4.3 is an upgrade of the scaleable TCP/IP stack for the Nucleus RTOS. Nucleus TFTP Client and Nucleus RIP II—two modules once sold separately—are now part of Nucleus NET. Developers can exclude unwanted modules and develop projects ranging from simple remote monitoring devices to Web-browsing applications. The stack's basic protocols include TCP, UDP, IP, and ICMP. Licenses for Nucleus NET begin at $14,495 and are provided with source code and no royalty fees.

Accelerated Technology
Mobile, AB
(334) 661-5770
www.atinucleus.com

Voice over various

The aXs family of board-level products addresses the system requirements for carrier-class wireless and wireless network applications. This includes echo cancellers, trunking or media gateways, and broadband access gateways for voice over DSL, voice over cable, and voice over wireless local loop. The set of products in the family is the aXs.500 series of Packet-Voice Processing Engines. The series realizes up to 1024 voice channels of processing on a single-width PCI mezzanine card module, depending on the application. These board-level products are preconfigured with voice compression, fax relay, echo cancellation, telephony processing, and packet processing functionality. Announcements regarding application-specific products will be made during the third quarter of 2000.

Spectrum Signal Processing
Burnaby, BC, Canada
(604) 421-5422
www.spectrumsignal.com

Development tools

The CAD-UL and US Software Solution is a package of development tools designed for use in the design of x86 processor-based embedded Internet devices such as factory floor controllers, medical instrumentation, set-top boxes, gaming consoles, and communications gateways. It includes US Software's SuperTask! RTOS, USNET Web Server, USNET TCP/IP Protocol Suite, the CAD-UL XDB SuperTask! (a debugger designed specifically for use with the SuperTask RTOS), and the CAD-UL C/C++ compiler system. CAD-UL XDB SuperTask! is also available as a standalone product. Debugger pricing is $1,900 for Real Mode and $2,800 for Protected Mode. For the entire package, pricing is $11,900 for Real Mode and $14,000 for Protected Mode. All versions are available now.

U S Software
Hillsboro, OR
(503) 844-6614
www.ussw.com

CAD-UL
Scottsdale, AZ
(480) 945-8188
www.cadul.com



why would you want to network a thermostat?



ure = 72 degrees • power consumption = 4 kilowatts per hour • regulate = auto • high set point temp = 72 degrees • low set point te
r drop = 4.2 degrees • fan = low • energy discount = 18% • light = on • switch = off • filter efficiency = low • current mode = he

We don't make thermostats. We provide networking software that allows thermostats and many other everyday products to communicate, making them smarter. For example, our software enables thermostats to be remotely managed. That means utility providers can eliminate brownouts and reduce the need to build new power plants. And for customers it means less power consumption, which results in lower utility bills. So maybe the question should be, why wouldn't you network a thermostat? To find out how emWare can help make your products better, visit us at www.emware.com/thermostat3

to conserve energy.

emWare® ANY DEVICE, ANY NETWORK™

eti ALLIANCE™

e-smart™

device networking software that makes products better

go to www.emware.com/thermostat3 or call toll-free 1.877.436.9273

Copyright © 2000 emWare, Inc. All rights reserved. The ETI logo, e-smart and the e-smart logo are trademarks of emWare, Inc. All other trademarks are the property of their respective owners.


The Future
Is Ahead
of Schedule
Convergence. It's happening. Now. How do you get there? Trillium. We provide the software and support to enable you to develop equipment for the convergence of telephony, Internet, wireless and cable networks. Call us at +1-310-442-9222 or visit www.trillium.com. Now. Because the future is ahead of schedule ... and so is Trillium.
TRILLIUM®
The World's Source for Communications Software™
Trillium Digital Systems, Inc. Los Angeles, California
www.trillium.com/ESP



Dan Saks

Numeric Literals

The C language provides several different kinds of constants: *integer constants* such as `10` and `0x1C`, *floating constants* such as `1.0` and `6.022e+23`, and *character constants* such as `'a'` and `'\x10'`. C also provides *string literals* such as `"ouch!"` and `"\n"`. C++ provides the same kinds of constants and literals, except the C++ standard just calls all of them *literals* instead. I do, too.

Every literal in C and C++ has a type. For example, `10` is obviously an integer. But is it an `int` (which is `signed` by default) an `unsigned int`, or a `short` or `long` variation thereof? Do you even care?

If you're a C programmer, you might not care. Part of me hopes you'd care, but I'm not absolutely sure you need to. Many C programmers apparently get by quite well without knowing the exact types of constants.

If you're a C++ programmer, you really should care. C++ supports function name overloading, and the exact type of a literal just might determine which function is the best match for a particular call. For example, given:

```
int f(int);
unsigned int f(unsigned int);
long int f(long int);
```

which of these functions does `f(10)` call? The answer is:

```
int f(int);
```

because the precise type of `10` is `(signed) int`.

Forms of integer literals

An integer literal takes different forms:

- A decimal integer literal (base 10) is a non-zero digit followed by zero or more decimal digits
- A hexadecimal integer literal (base sixteen) is `0x` or `0X` followed by a sequence of one or more hexadecimal digits
- An octal integer literal (base eight) is the digit `0` followed by a sequence of zero or more octal digits

For example, the decimal integer literal `63` can also be expressed as the hexadecimal integer literal `0x3F` or as the octal integer literal `077`.

Any integer literal may have a suffix that influences its type:

- `U` or `u` specifies that the literal has an `unsigned` type
- `L` or `l` specifies that the literal has a `long` type
- `LL` or `ll` specifies that the literal has a `long long` type. (This is available in the most recent edition of Standard C, C99, but not in earlier versions of C or in C++)

For example, `63u` has type `unsigned int`. `0x3FL` has type `(signed) long int`. An integer literal suffix may combine `U` with either `L` or `LL` (in either upper or lower case and in either order), so that both `63ul` and `0x3FLU` have type `unsigned long int`.

An integer literal's suffix influences the literal's type, but does not determine it. The type also depends on the literal's value.

Every literal has a type. It may not be obvious, and it may vary across platforms, but the standard does specify it precisely.

Types of integer literals

In general, an integer literal's type is the smallest integer type no smaller than `int` that can hold the literal's value and still satisfy the constraints imposed by the form and suffix. For example, the type of a decimal integer literal with no suffix is the smallest (the first) of the following types that can represent its value: `int`, `long int`, and in C99, `long long int`. Because the maximum values of the types vary across platforms, the exact types for many integer literals also vary across platforms.

Both the C and C++ standards allow considerable slack in the range of values for integer types. The maximum value for an `int` must be at least $2^{15}-1$, and the maximum value for a `long int` must be at least $2^{31}-1$. In practice, this means that an `int` must

In C++, the only form of a floating literal is a decimal floating literal, in which all of the digits (in both the significant part and the exponent part) are decimal digits. C99 also provides hexadecimal floating literals, in which the digits of the significant part are hexadecimal digits.

TABLE 1 The type of integer literals in C99 and C++

Suffix	Decimal Literal	Hexadecimal Literal
None	int	int unsigned int
	long int	long int unsigned long int
	long long int	long long int unsigned long long int
U or u	unsigned int	unsigned int
	unsigned long int	unsigned long int
	unsigned long long int	unsigned long long int
L or l	long int	long int unsigned long int
	long long int	long long int unsigned long long int
LL or ll	long long int	long long int unsigned long long int
Both U and L (in either case and either order)	unsigned long int unsigned long long int	unsigned long int unsigned long long int
Both U and LL (in either case and either order)	unsigned long long int	unsigned long long int

Lighter areas apply only to C99

occupy at least 16 bits and a `long int` must occupy at least 32 bits.

On any platform, `int` and `long int` may occupy more than their minimum storage requirement, as long as a `long int` occupies at least as much storage as an `int`. An unsigned integer type must occupy the same amount of storage as its corresponding signed integer type. For instance, a 16-bit platform might use 16 bits for `int` and `unsigned int` and 32 bits for `long int` and `unsigned long int`. A 32-bit platform might use 32 bits for all four types, `int`, `unsigned int`, `long int`, and `unsigned long int`.

Table 1 describes the rules for determining the type of an integer literal in C99 and C++. A literal whose value can't be represented in any of the listed types produces a compile error.

The rules in Table 1 lead to some potential portability problems. A decimal integer literal whose value is not greater than 32,767 (= 2^{15}–1) has type `int` on every standard C and C++ environment. No surprise there. However, any decimal integer literal whose value is greater than 2^{15}-1 but not greater than 2^{31}-1 has type `long int` on a 16-bit platform, but just plain `int` on a 32-bit platform.

For example, given the overloaded functions:

```
int f(int);
unsigned int f(unsigned int);
long int f(long int);
unsigned long int
    f(unsigned long int);
```

calling `f(32768)` calls `f(long int)` when compiled using C++ for a 16-bit platform, but it calls `f(int)` when compiled for a 32-bit platform. On the other hand, if you write the call as `f(0x8000)` (`0x8000` is 32,768 as a hexadecimal literal), it calls `f(unsigned int)` on every platform.

Floating literals

Floating literals offer a few little surprises, too. The type of a floating literal such as `1.0` or `6.022e+23` is `double`, not `float`. Appending the letter `F` or `f` to a floating literal makes its type `float`. Appending the letter `L` or `l` to a floating literal makes its type `long double`. For example, `1.0F` has type `float`, and `6.022e+23L` has type `long double`. Combining `F` and `L` (in either case and in either order) in a floating literal is a syntax error.

In C++, the only form of a floating literal is a decimal floating literal, in which all of the digits (in both the significant part and the exponent part) are decimal digits. C99 also provides hexadecimal floating literals, in which the digits of the significant part are hexadecimal digits.

In a decimal floating literal, the exponent part is optional. If present, it begins with `E` or `e` followed by decimal digits representing a power of 10. In a hexadecimal floating literal, the exponent part is mandatory. Since `E` and `e` are valid hexadecimal digits, a hexadecimal floating literal uses `P` or `p` to mark the beginning of the exponent part, following by decimal digits representing a power of two. For example, `0x1.FFFFFEp127f` is a floating constant whose value is `0x1.FFFFFE` multiplied by 2^{127} (decimal). The `f` at the end of the literal specifies that the literal's type is `float`.

Neither the form nor value of a floating literal affects its type. The suffix letter (or absence thereof) completely determines the type.

Character and string literals

Integer and floating literals are essentially the same in both C and C++. The differences lie in C's added support for `long long` integer types and hexadecimal floating literals. In contrast, character and string literals behave slightly differently between C and C++. Those differences are my topic for next time. **esp**

Dan Saks is the president of Saks & Associates, a C/C++ training and consulting company. He is also a consulting editor for the C/C++ Users Journal. *He served for many years as secretary of the C++ standards committee. With Thomas Plum, he wrote* C++ Programming Guidelines *(Plum Hall). You can write to him at dsaks@wittenberg.edu.*

PUT THE INTERNET IN



AND YOU'LL BE AMAZED AT WHAT COMES OUT



YOU ALREADY KNOW THAT CAPTURING INFORMATION FROM NON-NETWORKED DEVICES MEANS CONNECTING THEM TO THE INTERNET. BUT DO YOU KNOW HOW EASY IT IS? DO YOU KNOW THAT WITH OUR DEVICE SERVERS, YOU CAN ADD INTELLIGENCE

WITH DEVICE SERVER TECHNOLOGY™ YOU CAN INTERNET ENABLE ANY DEVICE.

TO ANY DEVICE AND TAP INTO A WHOLE WORLD OF POWER AND INFORMATION? WELL YOU CAN. YOU CAN INCREASE THE LIFE AND USEFULNESS OF ALL YOUR LEGACY SYSTEMS — WITHOUT HAVING TO MODIFY THEM. YOU CAN INTERNET-ENABLE THE DEVICES YOU HAVE TODAY AND EVEN THE ONES YOU'LL HAVE TOMORROW. WHAT'S MORE, YOU CAN IMMEDIATELY BEGIN ACCESSING AND SHARING INFORMATION ACROSS NETWORKS AND COMMUNICATING MORE EFFECTIVELY THAN EVER BEFORE. AND BY COMBINING OUR DEVICE SERVERS TOGETHER WITH OUR ADVANCED SOFTWARE OR OTHER REVOLUTIONARY APPLICATIONS, YOU'LL HAVE A COMPLETE INTERNET-ENABLED, TURN-KEY SOLUTION. SO THERE'S NO NEED FOR BULKY AND EXPENSIVE PC STATIONS. THIS IS THE POST PC ERA. WELCOME TO IT.

INTELLIGENT INTERNET CONNECTIVITY

By utilizing Device Server Technology™ you can make all your devices more intelligent and valuable, allowing them to interact dynamically and independently with other devices, data servers and clients anywhere via the Internet. And you can do it all quickly and affordably — without any device modifications.

ACCESS From easy installation to easy access, Device Server Technology enables new levels of connectivity and communication.

MANAGE Device servers are managed by using standard web browsers, so there's no need to modify the device.

CONTROL Our unique Software Developer's Kit enables you to write your own software and modify or enhance the devices' performance over the Internet.

From hardware to software, Lantronix provides complete turn-key solutions. We're ready to become your partner in the Post PC era. For more information regarding our Device Server Technology, please visit our website at **www.deviceserver.com** or call us at **877-462-9681.**

LANTRONIX

INTELLIGENTLY LEVERAGE THE INTERNET.

ROB OSHANA



feature

Sequence Enumeration

Sequence enumeration is a technique for analyzing systems. It can be used to turn a complex set of requirements into an easily implemented state machine.

The criticality of correct, complete, and testable requirements is a fundamental tenet in software engineering. Both functional and financial success are affected by the quality of requirements. So what is a requirement? It may range from a high-level abstract statement of a service or of a system constraint to a detailed mathematical and functional specification. Requirements are needed for several reasons:

- To specify external system behavior
- To specify implementation constraints
- To serve as reference tool for maintenance
- To predict future changes
- To characterize responses to unexpected events

The system designer must understand requirements and be able to organize them. A technical background and an understanding of the user are both required. Before design can start, each requirement must be understood in terms of significance and priority in the solution strategy. Because both the developer and the customer must understand the requirements, they are usually written in natural language. But natural language is a poor medium for communicating requirements.

The use of natural language to specify complex requirement has at least two problems: those of ambiguity and inaccuracy. Many words and phrases have dual meanings and can be altered depending on the context in which they are used. A word that means one thing to one person can mean something entirely different to someone else. For example the interpretation of the word "bridge" can be completely different depending on whether you are a dentist, a civil engineer, an electrical engineer, or someone who just retired early.

Other disciplines require a more exact language for communicating specifications. For example, an architectural blueprint or a circuit board schematic are formal specifications that define what is needed but not necessarily the implementation

Successful designs are usually the result of significant rethinking and reworking. Several iterations of design is the norm, rather than the exception. Design alternatives should be considered.

details of how to get there. Software requirements should be the same way. A software requirements specification should not contain implementation details. These specifications should describe in sufficient detail what the software will do, but not how. A good set of requirements should be:

- Correct—it meets the need
- Unambiguous—have only one possible interpretation
- Complete—it covers all the requirements
- Consistent—there should be no conflicts between requirements
- Ranked for importance
- Verifiable— a test case can be written
- Traceable—referring to requirements should be easy
- Modifiable—it should be easy to add new requirements

Trying to define a large multidimensional capability for a complex embedded system within the limitations of the linear two-dimensional structure of a document becomes almost impossible. At the other end of the scale, the use of a programming language is too detailed. This is nothing more than "after the fact specification" which is just documenting what was implemented rather than what was required

Developing a good set of requirements to specify a system can be difficult. In many cases the stakeholders don't know what they really want. The various stakeholders also tend to express requirements in their own terms and may have conflicting requirements that need to be resolved. Organizational and political factors may also influence the system requirements. Requirements may also change during the analysis process as new stakeholders emerge. The term "requirements engineering" has emerged to address the transformation by which vague and, often unrelated, customer requirements are transformed into the detailed and precise requirements needed for system implementation.

Successful designs are usually the result of significant rethinking and reworking. Several iterations of design is the norm, rather than the exception. Design alternatives should be considered. In general, designs should get simpler rather than more complex.

It can be difficult to specify the total behavior of a complex system because of the total number of possible uses of the system. But this is precisely what needs to be done in order to ensure completeness and consistency in our designs.

FIGURE 1 Simple soda machine

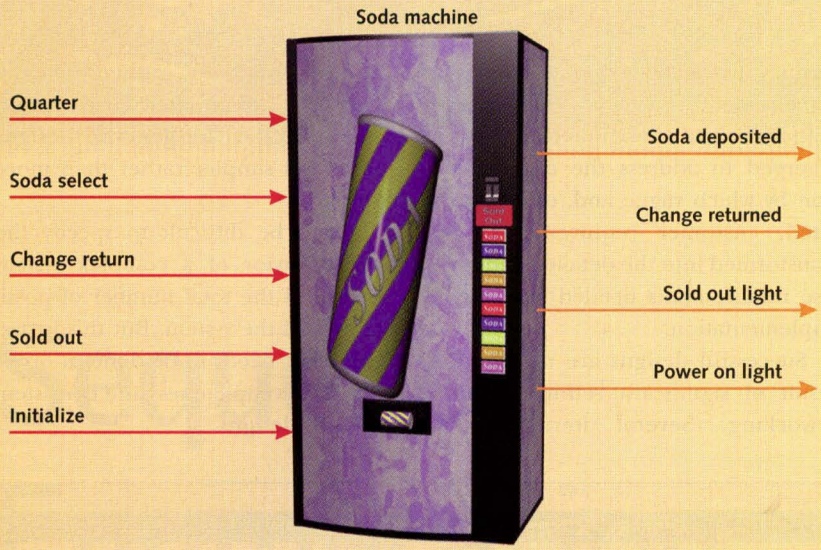


FIGURE 2 Natural language requirements for the soda machine

1. The soda machine will produce a soda after 75 cents has been entered. The machine will only accept quarters.
2. When the "Change Return" button is pressed, the available change in the input tray is returned.
3. When there is no more soda in the machine, the "Sold Out" light will be illuminated.

RUPERT ADLEY

FIGURE 3 Soda machine black box



You always believed there were more intelligent embedded tools out there.



People doubted their existence – yet you continued to search – and now you've found them.

COSMIC C compilers are fast, efficient, reliable, and produce the tightest object code available. Cosmic Software's embedded development tools offer portability for a complete line of microcontrollers. All toolkits include IDEA, our intuitive IDE that provides everything you need in a single, seamless Windows framework.

Add ZAP, our non-intrusive source-level debuggers and minimize your test cycle too. Want proof of their existence?

Download a free evaluation copy of our development tools at *www.cosmic-software.com* or call Cosmic today.

Cosmic supports the Motorola family of microcontrollers: **68HC05, 68HC08, 68HC11, 68HC12, 68HC16, 68300 and STMicroelectronics' ST7 Family.**

You were right.

COSMIC Software

E-mail: c-tools@cosmic-us.com
www.cosmic-software.com

Phone: US 781 932-2556
France 33 1 4399 5390
UK 44 01256 843400
Germany ...49 0711 4204062
Sweden 46 31704 3920

The hard part is being able to determine all the possible types of interaction with the system. One common technique for describing the system behavior is use cases. Use cases are textual narrative descriptions of the processes in an enterprise or system. They are used to explore and elicit requirements and can help determine which types of interaction to expect between the actors (external stimuli to the system) and the system. But use cases may not always combine to describe complete and consistent behavior.

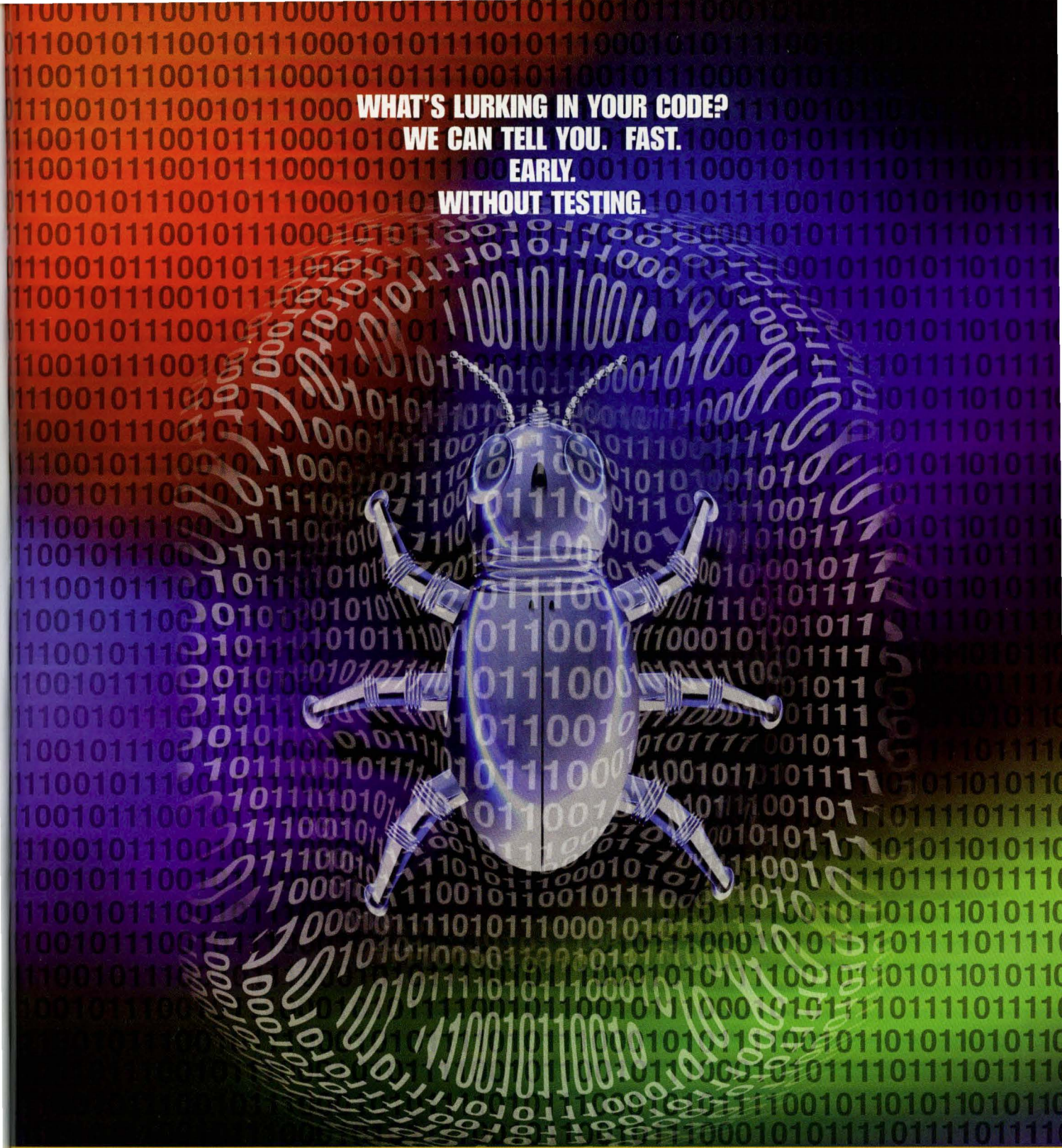
Use cases can be used to explore the problem domain and to perform front-end domain analysis. Other techniques can then be used to fully specify the solution strategy. One technique that has worked well in our embedded system project is called sequence enumeration.

Sequence enumeration

Sequence enumeration is a theoretically sound and highly practical approach to describing the external (or black box) behavior of a system. Sequence enumeration is a way of specifying and mapping stimuli and responses of an embedded system. This approach considers all permutations of input stimuli. Sequence enumerations consist of a list of prior stimuli and current stimuli as well as a response for that particular stimuli given the prior history. Equivalent histories are used to map certain responses. This technique maps directly to a state machine implementation. The strength of sequence enumerations is that the technique requires the developer to consider the obscure sequences that are usually overlooked.

As an example, I will consider the simple soda machine shown in Figure 1. A set of natural language requirements for this system is shown in Figure 2.

To start with, a use case can be developed that describes one type of interaction with the system. A use case



Crash-causing bugs can cost you. In time, money, and customer satisfaction. Reasoning's InstantQA℠ is a breakthrough Automated Source Code Inspection Service that finds bugs traditional QA testing misses. In under a week, without test cases, and without taxing your in-house resources. And because InstantQA finds defects in source code, you'll achieve higher software quality earlier, while reducing development costs and shortening your development cycle. Find out what's lurking in your code today with a **FREE TRIAL!**

Visit www.reasoning.com/emtrial.html
or call 1-888-429-4111 for more information.

REASONING

HELPING THE WORLD BUILD GREAT SOFTWARE™

Enumeration starts by evaluating a single stimulus to the system and determining the appropriate response for each stimulus.

TABLE 1 Soda machine stimuli and response abbreviations

Stimulus	Abbreviation	Response	Abbreviation
Initialize	I	Soda deposited	SD
Quarter	Q	Change return	CHR
Soda select	SS	Sold out light	SOL
Change return	CR	Power on light	LIGHT
Sold out	SO		

TABLE 2 Enumeration level 1

Line Number	Enumeration Level	Stimuli Sequence	Response	Equivalence	Requirement
1	1	I	LIGHT		4
2		Q	null	illegal	none
3		SS	null	illegal	none
4		CR	null	illegal	none
5		SO	null	illegal	none

TABLE 3 Enumeration levels 1 and 2

Line Number	Enumeration Level	Stimuli Sequence	Response	Equivalence	Requirement
1	1	I	LIGHT		4
2		Q	null	illegal	none
3		SS	null	illegal	none
4		CR	null	illegal	none
5		SO	null	illegal	none
6	2	I I	null	I	4
7		I Q	null		1
8		I SS	null	I	D1
9		I CR	CHR	I	2
10		I SO	SOL		D2

is a story about the usage of a system told from the end user's perspective. It consists of:

- Preconditions—what must be available and done before the performance of the use case
- Description—the story
- Exceptions—exceptions in the normal flow of the story
- Illustrations—figures to help understand the use case
- Postconditions—the state of the system and actors after the use case has been performed

Use cases are desirable because they:

- Analyze and improve functional requirements
- Model the functionality of the system as soon as possible
- Allow the customer to understand the operation of the system

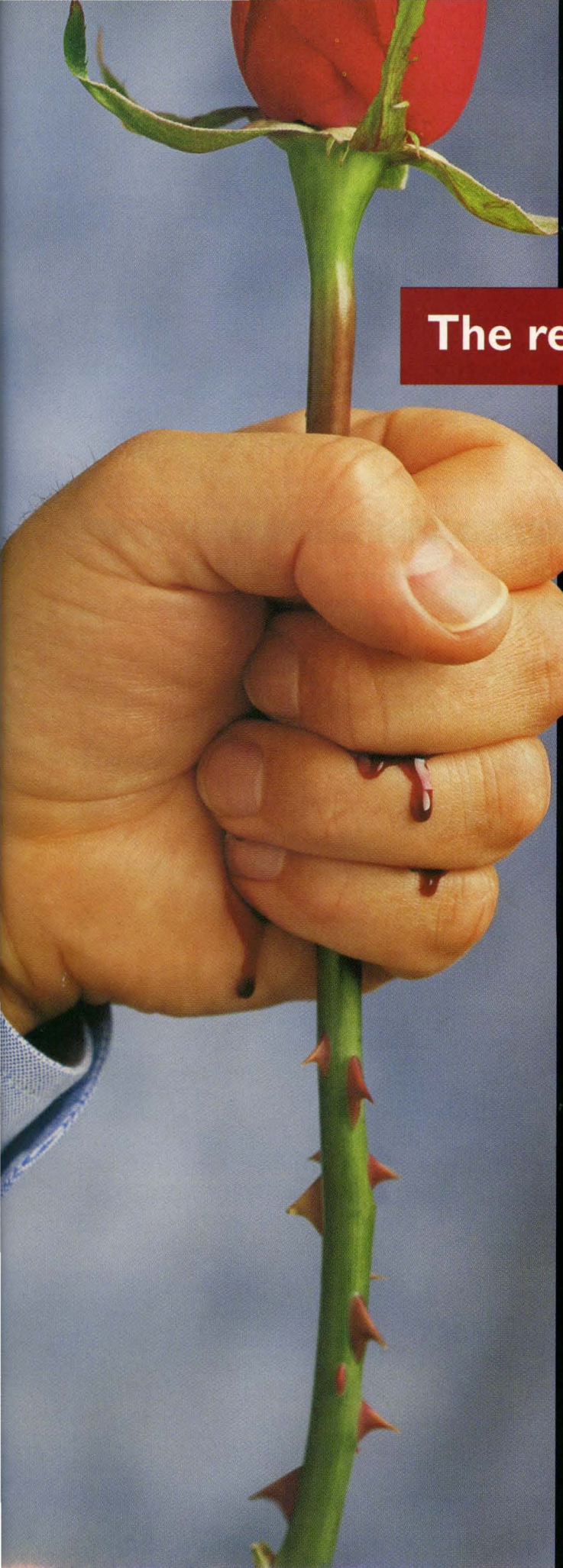
Use cases must specify the most important functional requirements. A use case depicts a typical way of using the system but nothing more. In general, a use case should not try to define all possible ways of performing a task. Certain important exception conditions are described in "secondary" use cases or in exceptions within a use case. A simple use case for the soda machine can be as follows:

- Precondition: Machine is ON and there is one soda in the machine
- Description: A user walks up to the machine and deposits two quarters. He then presses the soda button. Nothing comes out because three quarters are required to get a soda. The user hits the change return to get his change. He then leaves to get another quarter. He returns with three quarters, deposits the quarters and selects a soda. This is the last soda in the machine so the "Sold Out" light comes on
- Exceptions: None
- Postconditions: There is no soda left in the machine and the "Sold Out" light is ON

Other use cases can be developed depending on the various business cases for this system. Once we have a general idea of what the system should do (something use cases are helpful for determining), we can then perform a sequence enumeration to more completely map all possible combinations of stimuli to responses.

To start with, it's a good idea to represent the system as a "black box," showing the stimuli and the responses that are possible. The black box for the soda machine is shown in Figure 3. Table 1 lists the abbreviations used for each stimulus and response in the system enumeration.

A sequence enumeration can be done in a spreadsheet. Enumeration starts by evaluating a single stimulus to the system and determining the appropriate response for each stimulus. Table 2 shows the first level enumeration for the system. As the table shows, each stimulus is separately evaluated. The table shows that if the Initialize stimulus (I) occurs (the power is applied and the system is initialized), the Power on light response (LIGHT) is produced, as required by requirement 4. If any other stimulus occurs, the response is null and it is considered an illegal sequence (you cannot select a soda if the power is not on, for example).



Rational Rose for real-time?

The result could be painful.

More and more development teams agree Real-time Studio® is the better system and software modeling solution.

It makes perfect sense. If you're developing real-time embedded systems, you need a software modeling solution that can support your entire team and will actually fit your existing process. Make no mistake — reach for Real-time Studio. This is the one and only software modeling solution with an object-based repository enabling complete, up-to-date, shared views of your system. Finally, your entire development team can communicate and collaborate — from start to product delivery. Developed exclusively for real-time systems, Real-time Studio is non-intrusive, flexible and features automatic code generation for C, C++, and Java. Today, it's the proven solution for hundreds of developers and development teams. And the list keeps growing. Get started on the fastest path to the right product. Visit **www.artisansw.com/real** for more information and a demo of Real-time Studio, plus you can register for our upcoming Real-time UML seminar.

ARTiSAN SOFTWARE

Real-time Studio®
www.artisansw.com/real

No developer was actually harmed or risked injury in the making of this ad.
Rational Rose is a registered trademark of Rational Software Corporation.
Real-time Studio is a registered trademark of Artisan Software Tools, Inc.

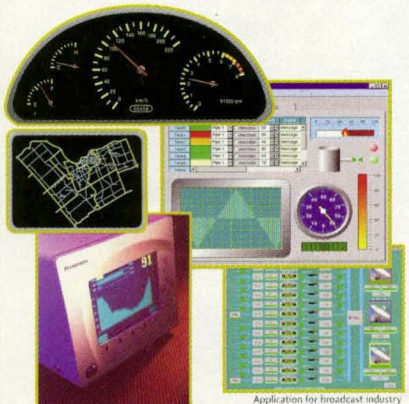
visual solutions for the real world

The **TILCON Real-Time Developer (TRTD)** is the ideal solution for custom GUI/MMI development in a variety of industries, such as: embedded device interfaces, medical displays, SCADA systems, virtual instruments and navigation systems.

TRTD will drastically cut development time. Innovative architecture provides a clean separation between application code and windowing systems. With its rich set of objects and ability to test interfaces without writing code, **TRTD** quickly moves your project from concept, to prototype, to product.

The **TRTD** API provides dynamic creation and editing of all objects. The API is identical across all platforms, including:

- NT Embedded
- Windows 2000/NT/95/98
- Windows CE
- RTX (VenturCom Inc.)
- LynuxWorks
- Linux: Red Hat, SuSE, Lineo
- QNX® Photon microGUI®
- QNX Neutrino®
- QNX X Free86
- new platforms coming soon



Courtesy of Physiometrix Inc. Patient State Analyzer PSA 4000™ FDA Approved

Application for broadcast industry Courtesy of Crystal Computer Corporation

www.tilcon.com

TILCON

tel: 800-665-5928 (toll free in North America) or 613-226-3917 | fax: 613-226-3631
e-mail: infonews@tilcon.com

Copyright 2000 TILCON Software Ltd. All rights reserved. Product and company names are registered trademarks or trade names of their respective companies. Ref: #801

Once the sequence enumeration is complete, it is a fairly simple step to develop the finite state machine that represents the specified behavior.

The result from performing this first level enumeration is that the system must be initialized (the I stimulus has occurred) before anything else can be done. We must remember that this event occurred. Therefore, the I stimulus is used in the next level of enumeration.

The level 2 enumeration uses all stimulus sequences that were not illegal or did not have a shorter equivalent sequence in the previous enumeration level than the first stimulus in the sequence. So, as Table 3 shows (lines 6 through 10), the first stimulus is I and the second stimulus is an enu-

TABLE 4 Complete enumeration for soda machine

Line Number	Enumeration Level	Stimuli Sequence	Response	Equivalence	Requirement
1	1	I	LIGHT		4
2		Q	null	illegal	none
3		SS	null	illegal	none
4		CR	null	illegal	none
5		SO	null	illegal	none
6	2	I I	null	I	4
7		I Q	null		1
8		I SS	null	I	D1
9		I CR	CHR	I	2
10		I SO	SOL		D2
11	3	I Q I	null	I	4
12		I Q Q	null		1
13		I Q SS	null	I Q	D1
14		I Q CR	CHR	I	2
15		I Q SO	SOL, CHR	I SO	D2
16		I SO I	null	I	4
17		I SO Q	CHR	I SO	D3
18		I SO SS	null	I SO	D4
19		I SO CR	CHR	I SO	D5
20		I SO SO	null	I SO	D6
21	4	I Q Q I	null	I	4
22		I Q Q Q	null		1
23		I Q Q SS	null	I Q Q	D4
24		I Q Q CR	CHR	I Q Q	2
25		I Q Q SO	CHR, SOL	I SO	D2
26	5	I Q Q Q I	null	I	4
27		I Q Q Q Q	null	I Q Q Q	1
28		I Q Q Q SS	SD	I	1
29		I Q Q Q CR	CHR	I	2
30		I Q Q Q SO	CHR, SOL	I SO	3

FIGURE 4 Derived requirements for the soda machine

D1: Selecting a soda before the correct change has been entered will result in no response from the soda machine.

D2: A "Sold Out" stimulus before the last soda is selected will cause the "Sold Out" light to be illuminated.

D3: Whenever the "Sold Out" light is illuminated, any change deposited into the machine will automatically be returned.

D4: The soda select button will be ignored whenever the "Sold Out" light is on or whenever correct change has not been received.

Aim to Deliver the Best Product on the Market?

So Do We. In Fact We Already Have.

The best development environment anywhere ASPEX is a fully integrated, completely open development environment offering you the tools to develop, integrate and optimize embedded software for today's complex products. If you've found other development environments slow, buggy and prone to crashes, you'll find ASPEX just the opposite.

A toolkit for today Mixed processors, multiple processors, systems-on-chips. Intensely increased integration. That's today's embedded software reality and ASPEX was designed from the inside out to seamlessly support it all. Make it smaller, make it cheaper, make it faster, make it better. ASPEX can make it happen.



Superior debugging capabilities All this multiprocessing and integration vastly increases the complexity of software debugging. But ASPEX can handle it. From a single event-driven debugger, ASPEX lets you simultaneously debug multiple target systems whether the processors are all the same or of different architecture.

Easy to learn, easy to use ASPEX provides a uniform, graphical interface that tightly integrates all tools and lets you manage projects graphically instead of learning cryptic command strings. Our unique Extended Target Visibility feature actually lets you see more target resources. Our Trace and Analysis feature helps you track down hard to find problems.

ASPEX supports ARM, Analog Devices, DSP Group, Intel Strong ARM and Texas Instruments.

Let ASPEX give you the tools you need to meet the challenges you face. The tools to develop, integrate and optimize embedded software for today's increasingly complex products. We're experts in embedded development tools and it shows in every aspect of ASPEX. Test drive ASPEX for yourself and experience features our competition hasn't even dreamed of. Download your free, fully functional evaluation copy of ASPEX at www.allant.com.

ALLANT

Allant Software Corporation

1280 Civic Drive, Suite 206, Walnut Creek, CA 94596 • 925.944.9690 • fax 925.944.9612 • email info@allant.com • www.allant.com

The technique of sequence enumeration can be scaled to larger systems by using abstractions for the stimuli and responses.

FIGURE 5 Canonical sequences for the soda machine

Canonical Sequences (five states)

I
I Q
I SO
I Q Q
I Q Q Q

FIGURE 6 State data for the soda machine

State Data

SYS_STATUS	Bool {ON, OFF}
COIN_COUNT	Enum {0, 1, 2, 3}
SODA_AVAIL	Bool {TRUE, FALSE}

meration of each of the possible stimuli to the system).

As an example, line 6 states that if the system receives an Initialize stimulus (I), followed by another Initialize stimulus, the system reacts as if only a single Initialize stimulus was applied. In this case, the Equivalence column is set to I to indicate that the sequence I I is equivalent to the sorter sequence I in terms of behavior.

Line 7 describes that the sequence of Initialize followed by the depositing of a single quarter (I Q) produces no response and has no equivalent sequence (another piece of behavior that the system needs to remember). Line 8 describes that the sequence of Initialize followed by a Soda Select stimulus (SS) produces no response and is equivalent to the sequence I in terms of behavior. In other words, if you press the soda select button after the power is applied and you have not entered the correct amount of money, your request will be ignored. Each sequence in this level that is not illegal or does not have a shorter equivalent sequence must be remembered and used in the next enumeration level. In this example, lines 7 and 10 must be remembered.

This process continues until all behavior is eventually mapped to a shorter equivalent sequence (it must always end or the software system may never terminate). Table 4 shows the complete enumeration for the soda machine. The process finally terminates in lines 26 through 30. Line 28 describes the condition where the Initialize stimulus, followed by three Quarter stimuli, followed by a Soda Select stimulus (I Q Q Q SS) will produce the Soda Deposited response (SD). This sequence also has an equivalent state of I, meaning that after a soda is deposited, the system does not need to remember that a soda has been deposited and effectively starts again from the Initialize state.

You may have noticed that the enumeration tables have a "Requirement" column that is used to trace the requirement as we define the behavior. This is a simple easy way to trace all requirements to behavior. You will also notice the derived requirements (indicated as D1, D2, and so on). A derived requirement is a lower level requirement that is determined to be necessary for a higher level requirement to be met. In general, a derived requirement is more specific and directed toward some sub-element of the project. Derived requirements often occur through the process of analysis. The derived requirements for this system are shown in Figure 4.

A couple of important conclusions ca be drawn from this process. We can say that this process produces a set of behaviors, that is:

- Complete: the enumeration considered, in effect, every possible combination of stimuli to the system for an infinite length of possible stimuli. This ensures that we have considered every possible behavior condition. The requirement trace also ensures that we have covered all possible requirements
- Consistent: because we have considered every combination of stimuli only once, we have ensured consistency. In other words we have not mapped a stimuli combination to more than one possible output sequence
- Correct: the mapping from stimuli to responses have been properly specified in the judgement of the domain experts

Implementation

Once the sequence enumeration is complete, it is a fairly simple step to develop the finite state machine that represents the specified behavior. To determine the number of states in the finite state machine, simply count the lines of enumeration that are not illegal and do not have equivalent sequences. These are referred to as the canonical sequences. In Table 4, these are lines 1, 7, 10, 12, and 22. These are also listed in Figure 5. State data can be invented that will encapsulate the behavior described in these canonical sequences. Figure 6 shows the state data that I invented to encapsulate this system's behavior.

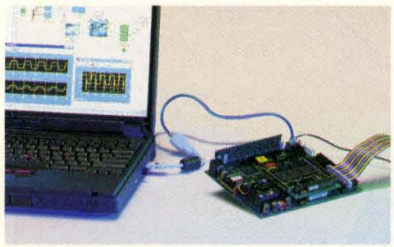
Based on the state data and the behavior described by the enumeration sequences, the state machine for the soda machine is easily produced (Figure 7). Notice the five states corresponding to the five canonical states produced during the enumeration process.

The technique of sequence enumeration can be scaled to larger systems by using abstractions for the stimuli and responses. These abstractions can be decomposed and expanded at lower levels of behavior definition, as more and more details of the system are exposed. In effect, what we have

NEW
PC-IN-THE-LOOP PROTOTYPING PRODUCTS

Don't just predict your real-time system's performance. Prove it.

With the new PC-in-the-Loop™ products, you can accurately test the performance of real-time embedded system designs on a low-cost PC long before your final target hardware is available. Simulink code generation products convert block diagrams to real-time C code for rapid prototyping. You can then go directly to your embedded target with our production-quality code.



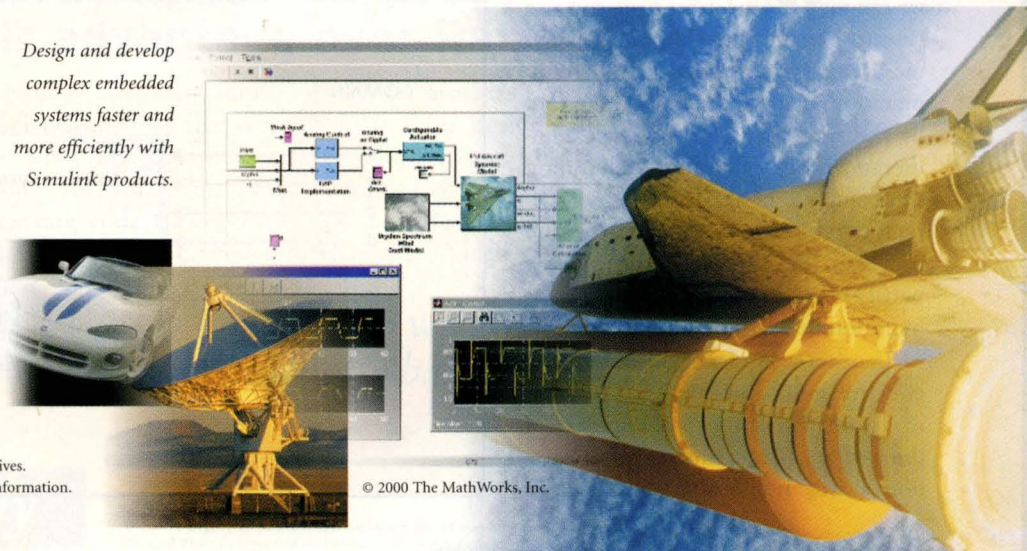
New PC-in-the-Loop products enable real-time rapid prototyping of embedded designs on standard x86 PC hardware in any form factor.

Learn how PC-in-the-Loop products enable rapid prototyping right on your own desktop.

To get your Embedded Systems Technical Kit, for pricing information, or to buy online now, visit www.mathworks.com/espc.

SIMULINK®

Design and develop complex embedded systems faster and more efficiently with Simulink products.



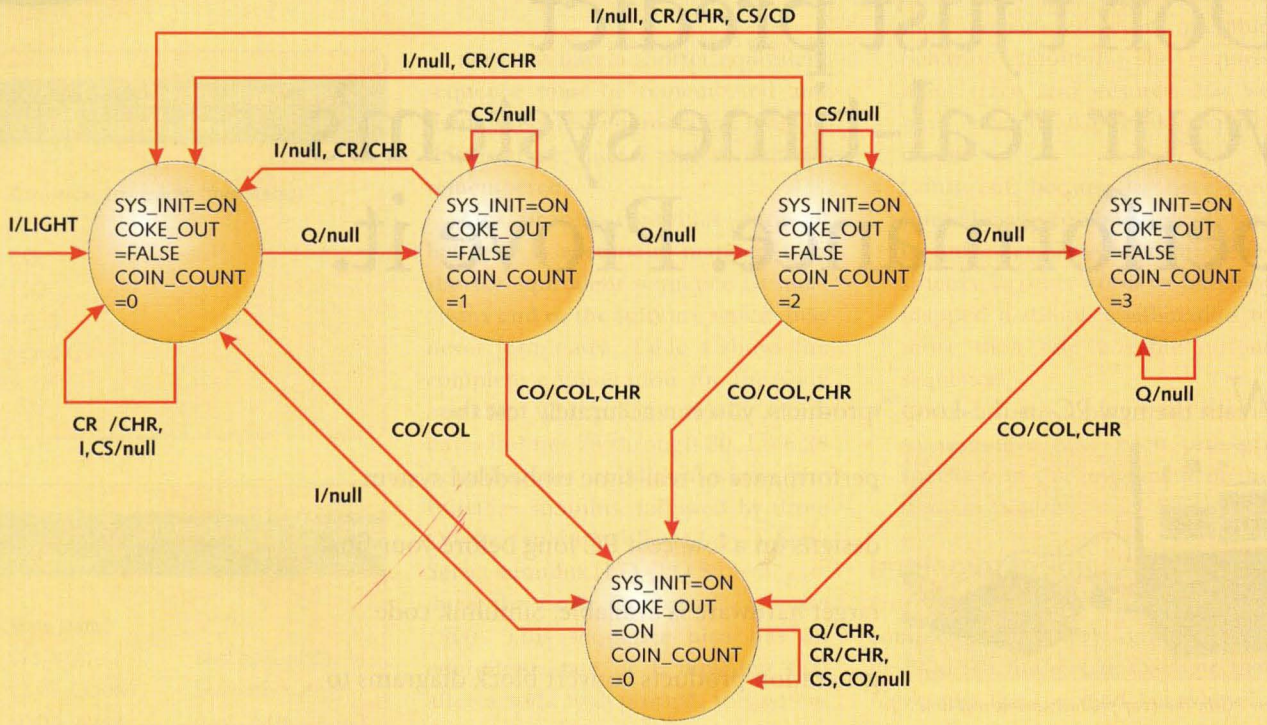
The MATH WORKS Inc.

Visit www.mathworks.com/espc or call 508-647-7000

We have a worldwide network of international representatives.
Visit our Web site at www.mathworks.com/eur for more information.

© 2000 The MathWorks, Inc.

FIGURE 7 State machine for the soda machine system



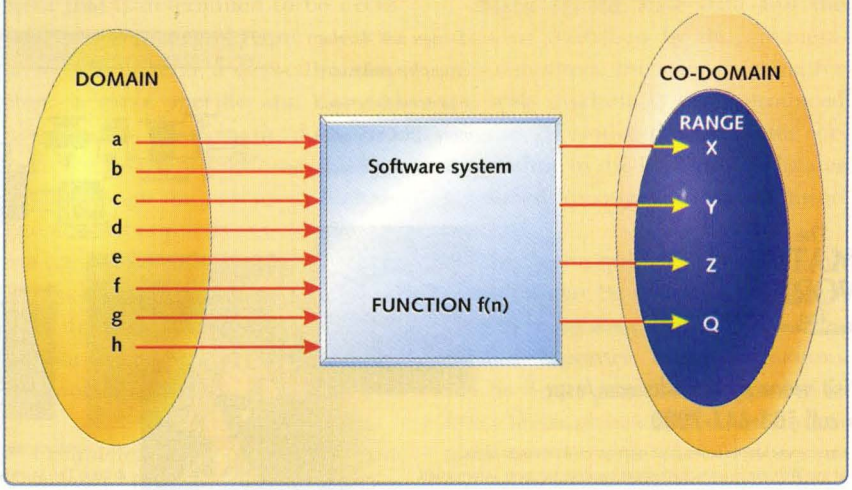
done is define a rule for a function for this software system. This process is rooted in the mathematics of function theory, which maps a domain (valid inputs or stimuli to a system) to a range (the correct responses) within a co-domain (all possible responses). See Figure 8.

Referring back to the use case described earlier, you will notice that we can map the behavior from this use case to parts, but not all, of the enumeration. Certain parts of the enumeration describe the story being told by the use case. The sequence enumeration goes a step further, defining the complete behavior that was originally described in the story told in the use case. This is where the synergy between use cases and sequence enumeration becomes apparent.

Use cases and scenarios are an effective tool to decide what we have to do for the various stakeholders of the system. The sequence enumeration drives home the completeness and consistency that will give your developers a clear picture of what to do in all circumstances of use, as well as resolve any inconsistencies between the stakeholders. **esp**

Rob Oshana is an engineering manager in the Software Development Systems group at Texas Instruments. He has over 18 years of experience in the management and development of real-time embedded systems. He also teaches requirements and design courses at Southern Methodist University. His e-mail address is oshana@airmail.net.

FIGURE 8 A function maps a domain to a range within a co-domain



There is nothing Rational about paying for UML Modeling.



We hate to say it, but Rhapsody Modeler has an unfair advantage over Rational Rose

It's free. That's right, free. Of course, we also think Rhapsody Modeler is great for other reasons, but free is a pretty darn good reason to start with. So what makes Rhapsody Modeler the right choice for you? You can analyze, design, document and generate code frames for your applications using UML. It's available in C, C++ and Java®. It provides a compatible upgrade to Rhapsody Validator for design validation and Rhapsody Developer for validation and full production code generation without losing any of your data. And – as if that's not enough – if you use Rose, your models can be read directly into Rhapsody Modeler. Need we say more? Oh yeah, it's free.

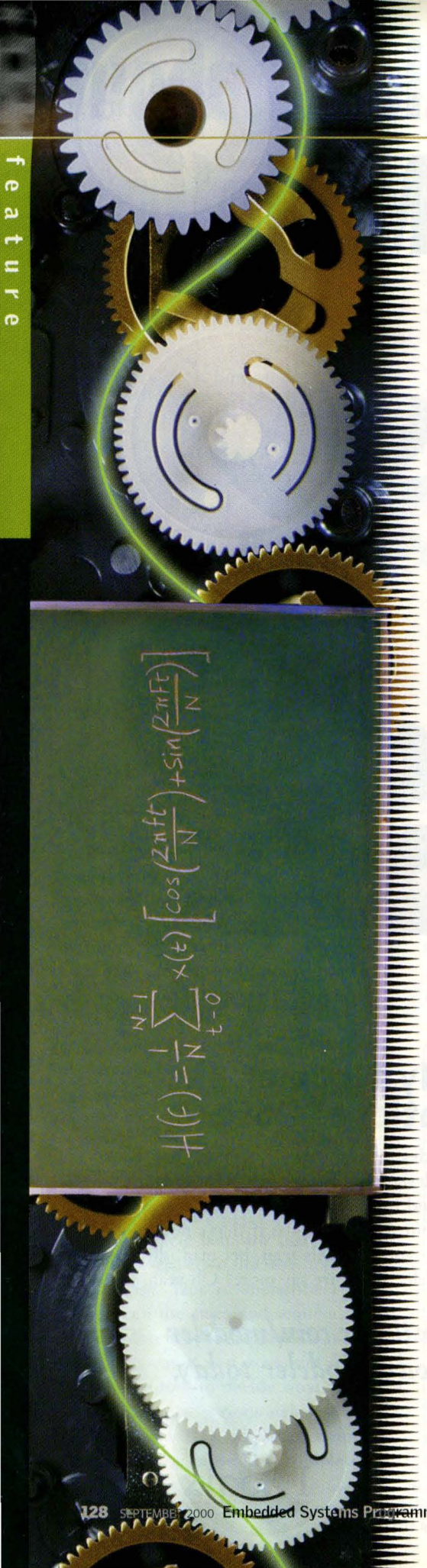
So what are you waiting for? Go to www.ilogix.com/modeler and sign up for your free copy of Rhapsody Modeler today.

Rhapsody®

© 2000 I-Logix Inc. Java is a registered trademark of Sun Microsystems, Inc. Rational Rose is a registered trademark of Rational Software Corporation.

I-Logix®

Visit us at Embedded Systems Conference 2000, Booth# 1914



ROBERT SCOTT

Doing Hartley Smartly

Think the fast Fourier transform is fast enough? Think again. Meet the fast Hartley transform.

Transforming a series of samples taken in the time domain into a frequency domain representation is useful in a wide variety of signal processing applications. This includes direct analysis of the frequency spectrum as well as operations like convolutions, which are used in digital filtering. The usual tool for performing this transformation is the Fourier transform, or to be more specific, the fast Fourier transform (FFT). But for most applications, there is an even faster method called the Hartley transform. This article reviews some published data on the fast Hartley transform (FHT) and then develops an optimized implementation of this algorithm.

A typical FFT application starts with a series of samples taken in the time domain. For example, suppose you want to determine the vibration characteristics of an automobile engine. You could attach an accelerometer to the engine block and sample the output of the accelerometer at regular intervals. Suppose you take 4,096 samples over a one-second period. These 4,096 samples would then be transformed, using an FFT, into an array of 4,096 elements where each element specifies the vibrational energy at a particular frequency. The frequencies represented in the FFT array run from 0Hz to 4,095Hz in steps of 1Hz. So if you needed to determine the total energy in the frequency band from 200Hz to 300Hz, you could add up the amplitudes of FFT entries 200 through 300. Although this example is described using the FFT, the FHT works just as well.

When I first saw an article about the Hartley transform, I must admit that I didn't give it serious consideration. The formulae looked similar to the ones for the Fourier transform, except that the imaginary, *i*, was missing. I thought there must be some catch. But then, years later, I ran across

What could be so magical about correlating against a sine function that was shifted by an arbitrary amount? Well, the answer is "nothing."

that article again and decided to delve into it to see what the story was. I was so impressed with what I learned that I immediately wrote a little freeware utility for general aviation enthusiasts—a PC program to measure the RPM of an airplane engine based on the sound of that engine. The program (available at *www.wwnet.net/~rscott/rpmsound.html*) uses the Hartley transform to find the peaks in the frequency spectrum. The pattern and location of the frequency peaks determine the RPM.

Hartley transform defined

The Hartley transform was first proposed in 1942 by Ralph Hartley. This is the same fellow, by the way, who invented the well-known Hartley oscillator circuit. Just as in the Fourier transform, the Hartley transform starts with a sequence of samples in the time domain. Let $X(t)$ for $t = 0 \ldots N\text{–}1$ be such a sequence. The Hartley transform of this sequence is another sequence, $H(f)$ for $f = 0 \ldots N\text{–}1$ given by:

$$H(f) = \frac{1}{N}\sum_{t=0}^{N-1} X(t)\left[\cos\left(\frac{2\pi ft}{N}\right) + \sin\left(\frac{2\pi ft}{N}\right)\right] \quad (1)$$

If, for some reason, you want to undo this transform, the formula is almost identical, except for the absence of the $1/N$ factor:

$$X(t) = \sum_{f=0}^{N-1} H(f)\left[\cos\left(\frac{2\pi ft}{N}\right) + \sin\left(\frac{2\pi ft}{N}\right)\right]$$

The first thing that made me suspicious about these formulae was that the sum of the cosine and the sine of an angle is just another phase-shifted sine function (shifted by 45 degrees, to be exact). What could be so magical about correlating against a sine function that was shifted by an arbitrary amount? Well, the answer is "nothing." In fact, it is possible to define a whole class of reversible transforms based on other phase-shifted sine functions. But the specific sine function used in the Hartley transform exhibits a certain symmetry between the transform and its inverse. And when it comes to implementing the transform, we will see that we end up taking pure sines and cosines anyway and not their sums.

Comparisons with the FFT

In a 1988 article in *Byte*,[1] Mark O'Neill gave formulae similar to the ones shown above; but note the typo in his Equation 5, which I have corrected. In addition to introducing the Hartley transform, O'Neill went on to develop the fast Hartley transform, based on a 1984 article by Ronald Bracewell in *Proceedings of the IEEE*.[2]

FIGURE 1 A single FHT butterfly

The results of recursion Equations 3 and 4 are combined in this data-flow diagram. Blue represents terms with a cosine factor. Red represents terms with a sine factor. Black represents terms with no trig factors.

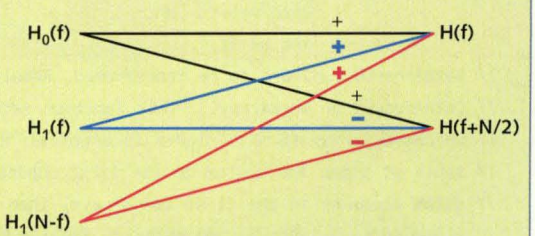
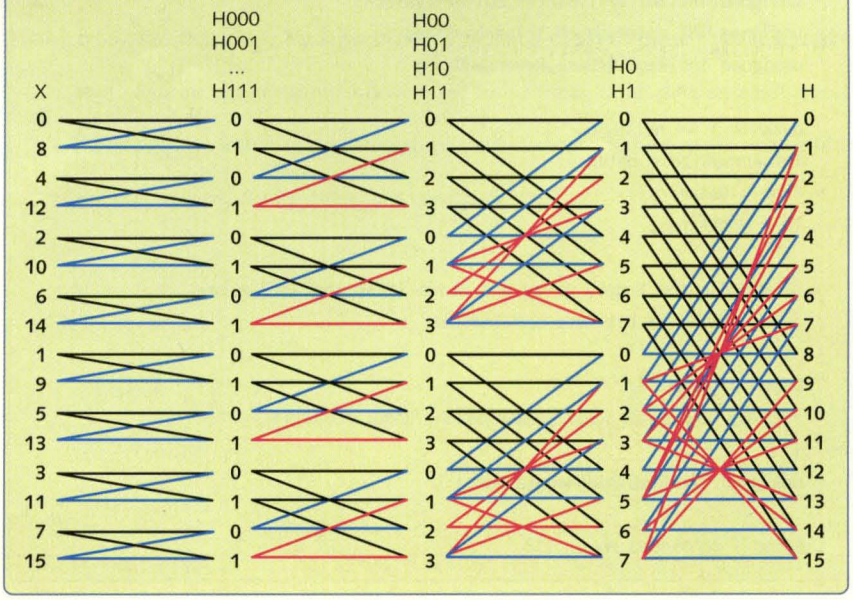


FIGURE 2 FHT data flow

A 16-point sequence, X, generates its Hartley transform, H. The lines represent instances of the single butterfly calculation from Figure 1. Lines have been ommited for null terms.



HANNAH BLAIR

LISTING 1 Working implementation of FHT

```
// Fast Hartley Transform - optimized for speed
#include <math.h>
typedef float DataType;

void InitTrigTable(DataType *TrigPtr, int nBits)
// Pre-compute sine function for first quadrant only
{
  unsigned int dataNdiv4,i;
  DataType RadianIncrement;

  dataNdiv4 = 1 << (nBits-2);
  RadianIncrement = 3.141592653589793238 / 2 / dataNdiv4;
  for(i=0; i<dataNdiv4; i++)
    TrigPtr[i] = sin( RadianIncrement * i );
}

DataType *FastHartleyTransform( DataType *DataArray,
                  DataType *WorkArray,
                  DataType *TrigPtr,
                  int nBits )
 // DataArray[0...N] are to be transformed. Results will be in either
 // DataArray[] or WorkArray[]. This function returns a pointer to
 // whichever array it is. TrigPtr is a pointer to a pre-computed
 // table of sines for angles in the first quadrant. If there are
 // dataN elements in the first two arrays, then there are dataN/4
 // elements in TrigPtr[], and nBits is the log base 2 of dataN.
 // dataN must be a power of 2 and be at least 16.
{
  DataType *FromArray,*ToArray,*SwapTemp;
  DataType *From,*To,*FromRetrograde;
  DataType *To_ArrayEnd,*To_QuadrantEnd;
  DataType *SinPtr,*CosPtr;
  unsigned int ButterflyBlocks,ButterflyWidth;
  unsigned int dataN,baseN,TrigArraySize;
  unsigned int RegularBase,ReversedBase;

  dataN = 1 << nBits;
  TrigArraySize = dataN/4;
  From = DataArray;
  To = WorkArray;

  // Reorder the From[] array and place it in the To[] array...
  // according to bit-reversed indices.

  baseN = dataN/16;
  for(RegularBase=0; RegularBase<baseN; RegularBase++)
  {
  unsigned int RemainingRegularBase;
```

Listing 1 continued on p. 132

While the Fourier transform works in the complete generality of complex-valued sequences, the Hartley transform only works with real-valued sequences. This should not be a problem for most real-world applications, which are limited to real-valued time domain samples anyway. To show that nothing is lost in using the Hartley transform, O'Neill gave the following formulae for the real and imaginary parts of the Fourier transform in terms of the Hartley transform:

$$F_{real}(f) = H(f) + H(N-f)$$
$$F_{imaginary}(f) = H(f) - H(N-f) \qquad (2)$$

It turns out that using the Hartley transform to compute the Fourier transform is faster than computing the Fourier transform directly. This is an important use of the Hartley transform because the Fourier transform has immediate natural interpretation while, as far as I know, the Hartley transform does not. Each complex value in the Fourier transform represents the amplitude and phase of a particular frequency component in the frequency spectrum of a time domain sequence. Often only the amplitude is interesting, so applications will then form the array of magnitudes of the complex-valued Fourier series. The implementation of the Hartley transform described here provides for a direct calculation of this array of amplitudes.

Note that Equation 2 leads to out-of-range indices when $f = 0$, since there is no $H(N)$. The interpretation of the indices must, therefore, be extended modulo-N. Therefore, $H(N)$ takes the value $H(0)$. And, in fact, this modulo-N interpretation of the indices is supported by the formulae that define the Hartley transform, which give the same values if you add N to the indices.

If you have ever used the Fourier transform to process real-valued data, you may have been aware of a sense of wasted effort. To begin with, you prob-

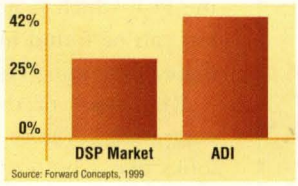
Visit us and our DSP Collaborative partners at the Embedded Systems Conference, San Jose, **Booth #1817**

THANKS TO YOU, WE'RE THE FASTEST-GROWING DSP SUPPLIER.



WE'RE SUPPORTING OVER 10,000 CUSTOMERS WHO DEVELOP THE WORLD'S HOTTEST PRODUCTS.

In this new era of signal processing, Analog Devices is proving that the DSP supplier who can simplify your design challenge is the one you choose most often. The 30 years of high-performance analog expertise we design into every DSP is enabling applications like broadband ADSL modems, wireless Internet appliances, home theater, voice-over-network gateways, and digital motor control. We recognize that these signal processing systems demand more than just DSPs, they require applications know-how, algorithms, and development tools. Above all, we know that partnerships with customers like you are responsible for our success. And for that we say thank you.



Our focus on applications that demand high-performance analog and DSP has resulted in ADI DSP sales growth of nearly 2x the overall market.

ANALOG DEVICES

www.analog.com

DSP SOLUTIONS THAT MAKE YOUR DESIGN CHALLENGE EASIER™

LISTING 1, continued Working implementation of FHT

```
  unsigned int Regular,Reversed;
  int BitsToMove,nibble;
   static const unsigned char bitrev16[] =
    {0,8,4,12,2,10,6,14,1,9,5,13,3,11,7,15};

  ReversedBase = 0;
  RemainingRegularBase = RegularBase;
  BitsToMove = nBits - 4;
  while(BitsToMove)
  {
   if(BitsToMove >= 4)
   {
    ReversedBase = (ReversedBase<<4)
     + bitrev16[RemainingRegularBase & 15];
    RemainingRegularBase >>= 4;
    BitsToMove -= 4;
   }
   else //(less than 4 bits to move...)
   {
    ReversedBase = (ReversedBase<<1)
     + (RemainingRegularBase & 1);
    RemainingRegularBase >>= 1;
    BitsToMove--;
   }
  }
  ReversedBase <<= 4;
  Regular = RegularBase;
  for(nibble=0; nibble<16; nibble++)
  {
   Reversed = ReversedBase + bitrev16[nibble];
   if(Regular <= Reversed)
   {
    To[ Regular ] = From[ Reversed ];
    To[ Reversed ] = From[ Regular ];
   }
   Regular += baseN;
  }
 }

// Perform the first two rounds of FHT at once

 From = WorkArray;
 To = DataArray;
 To_ArrayEnd = To + dataN;
 while(To != To_ArrayEnd)
 {
   DataType a,b,c,d;
  To[0] = (a=From[0] + From[1]) + (b=From[2] + From[3]);
```

Listing 1 continued on p. 134

ably stuffed your real-valued sequence into the real parts of an array of complex numbers and zeroed out the imaginary parts. Having all those zeroes using up space but providing no information always bothered me. Then, when the Fourier transform was done, the resulting array of complex numbers had a lot of redundancy. In particular, you never had to look past the midpoint of the array (the so-called Nyquist frequency) because the frequency domain data in the second half of the array was a kind of mirror image of the first half. The complex-valued $F(N-f)$ is just the complex conjugate of $F(f)$. (This is only true for the Fourier transform of real-valued sequences. In general, the Fourier transform has no such redundancy.) When I saw how the power of the Fourier transform was being "wasted" on real-valued sequences, I finally understood how something like the Hartley transform could be more efficient.

Butterflies

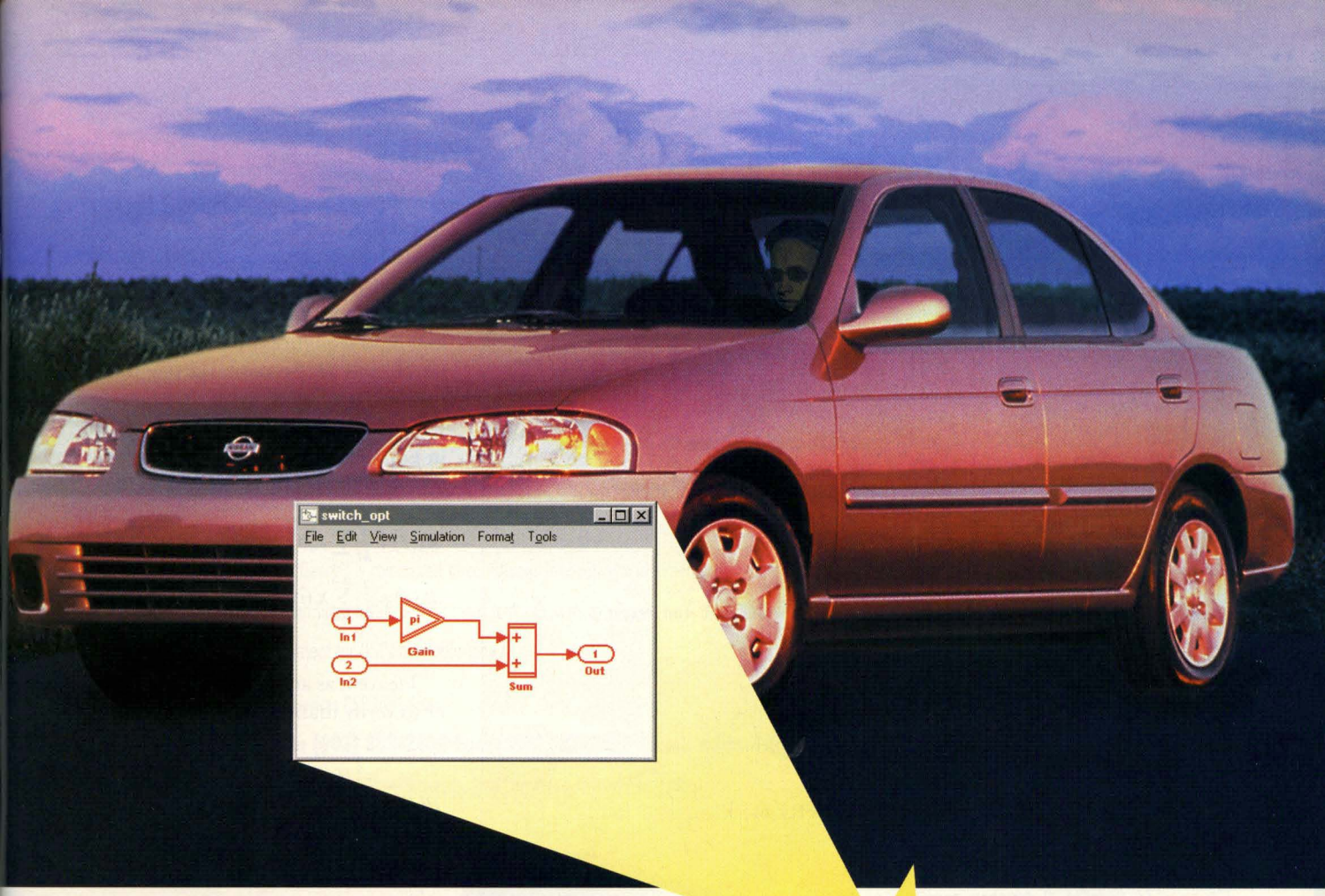
Now let's get down to business and see how the Hartley transform can be implemented efficiently as the FHT. The development is similar to that of the FFT, which is not presented here but can be found in any textbook on digital signal processing.

The heart of the algorithm is a computation whose data-flow diagram looks like a butterfly. But unlike the symmetric butterfly of the FFT, the FHT butterfly, as shown in Figure 1, is a bit ugly. It has a vestigial third wing.

From now on we will be assuming that N is a power of two. Since N is even, the time sequence, $X(t)$, which appears in the definition of the Hartley transform (Equation 1), can be divided into two intertwined sequences, X_0 and X_1, given by:

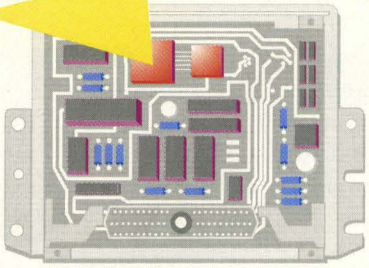
$$X_0(t) = X(2t)$$
$$X_1(t) = X(2t+1)$$

That is, sequence X_0 consists of the even-indexed values from sequence X and sequence X_1 consists of the odd-



dSPACE Code Hits the Road

The world's lowest emission car runs with TargetLink code



Featuring a novel fuel mix control, the new Nissan Sentra CA fulfills California's strict Zero Emission Vehicle laws (P-ZEV). A decisive factor for the swift time to market was the development team's trust in TargetLink, dSPACE's production code generator.

TargetLink offers the following advantages:

- Highly efficient code
- Intuitive user interface
- Complete integration in MATLAB®/Simulink®

"TargetLink is highly reliable. With dSPACE we reduced our development time for the air/fuel ratio controller to 60%."
Shigeaki Kakizaki, Nissan Motor

TargetLink is also being used by DaimlerChrysler, Ford, Honda, TRW, Unisia Jecs, VDO, Visteon, Volvo Truck.

dSPACE Inc.
22260 Haggerty Road · Suite 120
Northville MI 48 167
Tel: (248) 344-0096 · Fax: (248) 344-2060
info@dspaceinc.com · www.dspaceinc.com

dSPACE

LISTING 1, continued Working implementation of FHT

```
  To[1] = (c=From[0] - From[1]) + (d=From[2] - From[3]);
  To[2] = a - b;
  To[3] = c - d;
  To += 4;
  From += 4;
 }

// Start general butterfly calculations with round 3...

 FromArray = DataArray;
 ToArray = WorkArray;
 ButterflyWidth = 4;      // first two rounds are done...
 ButterflyBlocks = dataN/8; // these parameters are set for round 3.
 while( ButterflyBlocks )
 {
  From = FromArray;
  To = ToArray;
  To_ArrayEnd = ToArray + dataN;
  while(To != To_ArrayEnd)
  { // This while loop does a single butterfly block
    DataType CSTerm;

     // Special 0 degrees case...
   To[0]  = From[0] + From[ButterflyWidth];
   To[ButterflyWidth] = From[0] - From[ButterflyWidth];

     // ..then the rest of the first quadrant...
   To_QuadrantEnd = To + ButterflyWidth/2;
   SinPtr = TrigPtr;
   CosPtr = TrigPtr + TrigArraySize;  // beyond end of trig array
   FromRetrograde = From + ButterflyWidth + ButterflyWidth;
   while(1)
   {
    From++; To++; FromRetrograde--;
    if( To == To_QuadrantEnd ) break;
    SinPtr += ButterflyBlocks;
    CosPtr -= ButterflyBlocks;
    CSTerm = *CosPtr * From[ButterflyWidth] +
             *SinPtr * FromRetrograde[0];
    To[0]  = From[0] + CSTerm;
    To[ButterflyWidth] = From[0] - CSTerm;
   }

     // Special 90 degrees case...
   To[0]  = From[0] + FromRetrograde[0];
   To[ButterflyWidth] = From[0] - FromRetrograde[0];

     // ..then the rest of the second quadrant...
   To_QuadrantEnd = To + ButterflyWidth/2;
```

Listing 1 continued on p. 138

indexed values. As before, we will consider all indices to be interpreted modulo-N. But since X_0 and X_1 are defined in terms of $2t$, they actually repeat modulo $N/2$. So consider X_0 and X_1 as sequences of length $N/2$ and let H_0 and H_1 be their Hartley transforms. Let M stand for $N/2$. Then, by the definition of the Hartley transform in Equation 1 we have:

$$H_0(f) = \frac{1}{M}\sum_{t=0}^{M-1} X_0(t)\left[\cos\left(\frac{2\pi ft}{M}\right) + \sin\left(\frac{2\pi ft}{M}\right)\right]$$

$$H_1(f) = \frac{1}{M}\sum_{t=0}^{M-1} X_1(t)\left[\cos\left(\frac{2\pi ft}{M}\right) + \sin\left(\frac{2\pi ft}{M}\right)\right]$$

I leave it as an exercise for the reader to verify that:

$$H(f) = \frac{1}{2}\left\{\begin{array}{l} H_0(f) + H_1(f)\cos\left(\frac{2\pi f}{N}\right) \\ \quad + H_1(N-f)\sin\left(\frac{2\pi f}{N}\right)\end{array}\right\} \quad (3)$$

(If you are reading the *Byte* article, note that this equation is incorrectly shown with N_s instead of N.) It took me about half an hour to verify this equation with paper and pencil, using only the trig identities:

$$\sin(A)\cos(B) = \frac{\sin(A+B) + \sin(A-B)}{2}$$

$$\sin(A)\sin(B) = \frac{\cos(A-B) - \cos(A+B)}{2}$$

$$\cos(A)\cos(B) = \frac{\cos(A-B) + \cos(A+B)}{2}$$

If you want to develop an appreciation of the basis of the Hartley transform, I recommend performing the exercise.

Keeping in mind that the Hartley transforms on the right side of Equation 3 repeat modulo $N/2$, we can recognize the following symmetry:

$$H\left(f + \frac{N}{2}\right) = \frac{1}{2}\left\{\begin{array}{l} H_0(f) - H_1(f)\cos\left(\frac{2\pi f}{N}\right) \\ \quad - H_1(N-f)\sin\left(\frac{2\pi f}{N}\right)\end{array}\right\} \quad (4)$$

Equations 3 and 4 taken together

VisualDSP – The Best DSP Development Environment For The Best DSPs Delivered.

You want to build the most advanced digital signal processing applications – that's why you are working with DSPs from Analog Devices. To get the most out of them, you need the best development environment – VisualDSP is that product!

With VisualDSP, you have a completely integrated, easy-to-use project management environment that will allow you to control and manage your projects efficiently from end-to-end, and make the most of the superior advantages only Analog Devices DSPs can offer.

Analog Devices is committed to continuous innovation and superior DSP performance via the White Mountain line of DSP development tools. Give us a call and we'll fetch the best development environment for DSPs available today.

Analog Devices DSP Tools Sales

Phone: (603) 883-2430 Fax: (603) 882-2655 Email: aditools@analog.com

Web: http://www.analog.com for more information today.

WHITE MOUNTAIN DSP

The Design Center for Analog Devices DSP Tools



TABLE 1 FHT execution times for 75MHz Pentium

Points (N)	float	double
256	0.48ms	0.62ms
2,048	7.0ms	9.1ms
16,384	0.10s	0.13s
131,072	1.4s	1.8s

define the complete FHT butterfly computation, as diagrammed in Figure 1. These two equations express the Hartley transform of length N in terms of two smaller (half-length) Hartley transforms. This is the beginning of a recursive process. At each step in the recursion we recognize that all of the Hartley transforms from the previous step can be further broken down into Hartley transforms of half the length. The process continues until we have only Hartley transforms of length 1, which can be trivially computed from Equation 1.

At this point I will abandon mathematical rigor and proceed by example alone. Suppose $N = 16$. Figure 2 shows a data-flow diagram that illustrates the recursive decomposition of a Hartley transform of length 16. Although the data-flow diagram is computed from left to right, we shall analyze it from right to left. On the far right, the numbers 0 to 15 represent the values of $H(0)$ through $H(15)$, which is the Hartley transform of the original sequence $X(0)$ through $X(15)$.

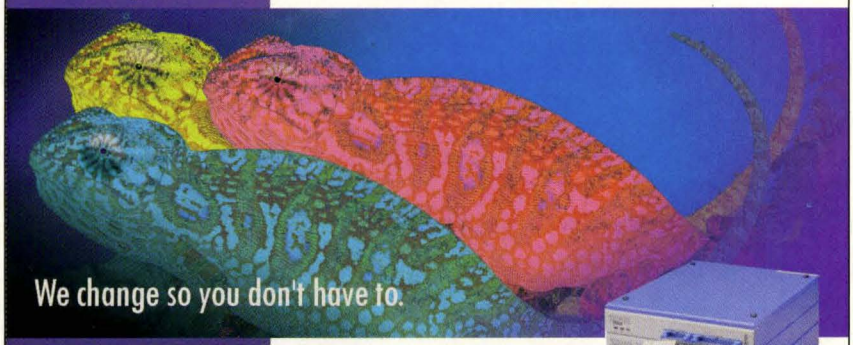
If you let f run from 0 to 7 in Equations 3 and 4, you can see how the 16 values of H can be computed from the two smaller Hartley transforms, H_0 and H_1, each of length 8. The right-most column of data-flow lines in Figure 2 illustrates this process as the overlay of eight instances of the butterfly diagram from Figure 1. The lines have been omitted where the corresponding trig factor is zero. In the column of numbers to the left of the final butterfly diagrams, the two sets of indices, 0 through 7, represent the value of H_0 and H_1 with H_0 on top. For example, the first three instances of Equations 3 and 4 in this column of butterflies is:

$$H(0) = 1/2\{H_0(0) + H_1(0)\cos(0) + H_1(0)\sin(0)\}$$
$$H(8) = 1/2\{H_0(0) - H_1(0)\cos(0) - H_1(0)\sin(0)\}$$
$$H(1) = 1/2\{H_0(1) + H_1(1)\cos(\pi/8) + H_1(7)\sin(\pi/8)\}$$
$$H(9) = 1/2\{H_0(1) - H_1(1)\cos(\pi/8) - H_1(7)\sin(\pi/8)\}$$
$$H(2) = 1/2\{H_0(2) + H_1(2)\cos(\pi/4) + H_1(6)\sin(\pi/4)\}$$
$$H(10) = 1/2\{H_0(2) - H_1(2)\cos(\pi/4) - H_1(6)\sin(\pi/4)\}$$

Note that the indices of H_0 and H_1 have been reduced modulo 8, since they are Hartley transforms of length 8.

The decomposition then continues with the next column to the left, showing how H_0 and H_1 are each repre-

Adaptable In-Circuit Emulators



iC1000
8 bit
Emulator



iC2000
8,16,32 bit
Emulator



iC3000
BDM, JTAG
Emulator



winIDEA
Software
Toolset



Preserve your investment in time and capital with universal emulators from iSYSTEM.

A complete line, from low-cost BDM/JTAG to high-end full-function ICEs.

Support for over 400 micro-controllers with a simple swap of the POD or software setup.

Driven with an intuitive development environment integrating all popular compilers.

Thousands in use world-wide.

Call for your free demo CD.

Support for 68HC05 08 09 11 12 16, 68K, 683xx, MCORE, MPC, 8051, 80x86, C500, Z80/180, CR16, PIC, ST7, and many more.

America 1 (888) 543-5300
Europe 49 (8131) 70610
Scandinavia 46 (40) 459571
Asia Pacific 82 (2) 2645-0386

iSYSTEM
www.isystem.com

Visit us at Embedded Systems Conference 2000, Booth #2709





Trouble getting your development off the ground?

Start your DSP product development with Blackhawk™

Blackhawk™

by EWA Inc.

DSP Enabling Technologies™

© 2000 EWA

For more information please visit: ***www.blackhawk-dsp.com***
or phone: ***1-877-983-4514***

WIN THESE PRIZES:
Digital Camcorder
Digital Camera
DVD Player
www.blackhawk-dsp.com/contest101.shtml

LISTING 1, continued Working implementation of FHT

```
    while(1)
    {
     From++; To++; FromRetrograde--;
     if( To == To_QuadrantEnd ) break;
     CSTerm = *SinPtr * FromRetrograde[0] -
              *CosPtr * From[ButterflyWidth];
     To[0]  = From[0] + CSTerm;
     To[ButterflyWidth] = From[0] - CSTerm;
     SinPtr -= ButterflyBlocks;
     CosPtr += ButterflyBlocks;
    }
    From += ButterflyWidth;
    To += ButterflyWidth;
   }
   ButterflyWidth <<= 1;
   ButterflyBlocks >>= 1;
   SwapTemp = FromArray;  // Swap "from" and "to" arrays
   FromArray = ToArray;
   ToArray = SwapTemp;
  }   // end of while( ButterflyBlocks ) loop
  return FromArray;
}

void PowerSpectrum(DataType *power, DataType *dht, int nBits)
 // converts Hartley transform dht[0...N-1] into power spectrum
 // power[0...N/2]. The power spectrum array is one entry longer than
 // half as long as the DHT because it is symmetrical after the
 // mid-frequency anyway. It is OK to call with power=dht and use
 // only the first N/2+1 elements of dht on return.
{
  unsigned int N,i,NminusI;

 N = 1 << nBits;
 for(i=0; i <= N/2; i++)
 {
  NminusI = (i==0)? 0 : (N-i);
  power[i] =
   sqrt( (dht[i] * dht[i] + dht[NminusI] * dht[NminusI]) / 2 );
 }
}

void ScaleHartleyTransform(DataType *data, int nBits)
{
  unsigned int i,N;
  DataType Nrecip;
 N = 1<<nBits;
 Nrecip = 1 / (DataType)N;
 for(i=0; i<N; i++)
  data[i] *= Nrecip;
}
```

sented in terms of Hartley transforms of length 4. H_0 is decomposed into H_{00} and H_{01}, while H_1 is decomposed into H_{10} and H_{11}. The numbers 0 through 3 are indices into these smaller Hartley transforms.

The recursion process continues all the way back to the first column, where we would have 16 Hartley transforms of length 1. Equation 1 shows that a Hartley transform of a time sequence of length 1 is just the original time sequence itself. So the first column represents values from the original time sequence $X(t)$. The numbers in that column are the indices into the original time sequence, $X(t)$. But how did they get into that strange order?

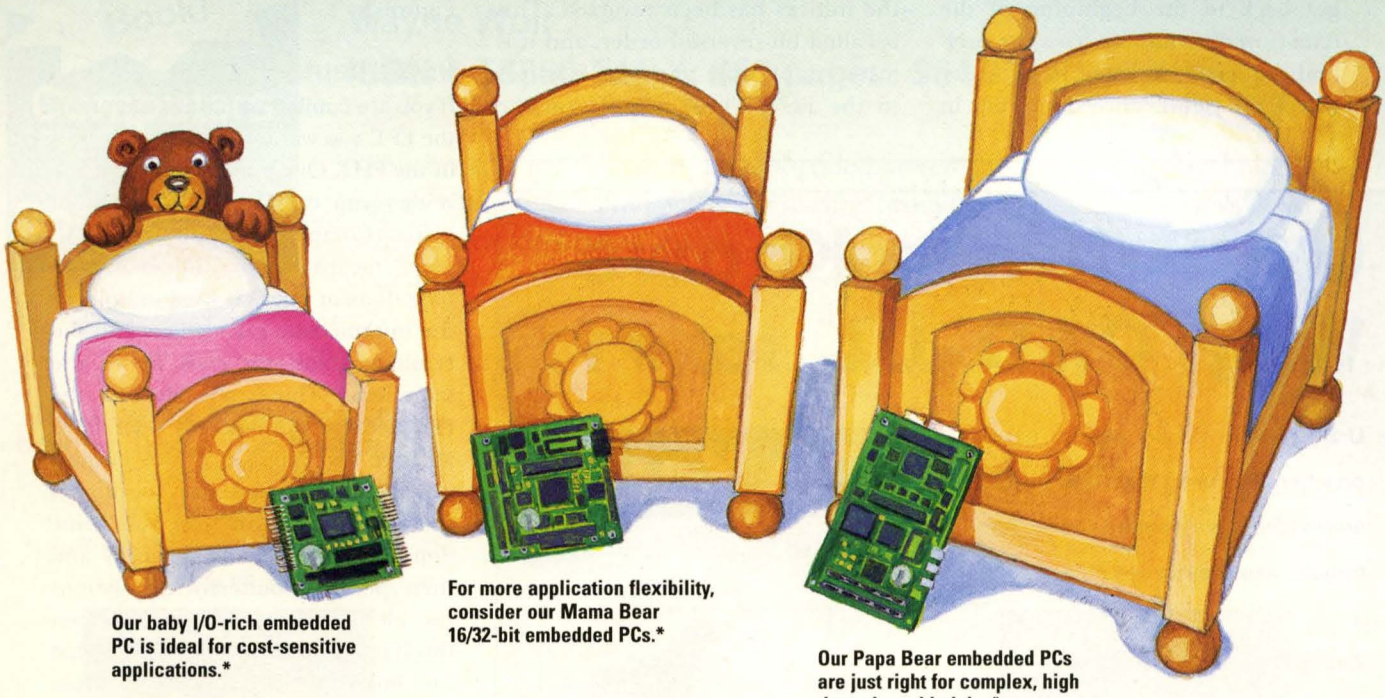
To answer this question we must determine exactly which time sequence is transformed into each of the Hartley transforms in Figure 2. Starting again at the right, we know that the full Hartley transform is derived from the original time sequence $X(0)$ through $X(15)$ in normal order. Going back one step in the recursion and recalling how the smaller Hartley transforms in Equations 3 and 4 were defined, we see that H_1 is the Hartley transform of the sequence $X(0)$, $X(2)$, ..., $X(14)$ and H_1 is the Hartley transform of the sequence $X(1), X(3), \ldots, X(15)$. Purely as a computational convenience, we place the time sequence for H_1 after the time sequence for H_0. We have effectively sorted the sequence, X, according to the least-significant bit of its indices.

In going from Hartley transforms of length 8 to Hartley transforms of length 4, we further refine the sort on X based on the next-least-significant bit of its indices. That is:

H_{00} is based on $\{X(0), X(4), X(8), X(12)\}$
H_{01} is based on $\{X(2), X(6), X(10), X(14)\}$
H_{10} is based on $\{X(1), X(5), X(9), X(13)\}$
H_{11} is based on $\{X(3), X(7), X(11), X(15)\}$

At each stage in this progressive sort, the results of the previous sort are preserved. Therefore when we

One Size Doesn't Fit All



Our baby I/O-rich embedded PC is ideal for cost-sensitive applications.*

For more application flexibility, consider our Mama Bear 16/32-bit embedded PCs.*

Our Papa Bear embedded PCs are just right for complex, high throughput, big jobs.*

Same is True with Embedded PCs

It's no fairy tale. You shouldn't settle for less than an optimum fit between your application needs and an embedded PC. If, for example, you need Linux operation, a V40 processor isn't appropriate. Or, if DOS is your choice, you may not need Pentium® power. Our *reasonable integration approach* and top notch engineering and technical support can help you make the optimum choice and assist you through application development. The result: a just right match of your requirements for PC compatibility and whatever specific mix of I/O works best for you, *all on one board*. Select from I/O like 12-bit A/D and D/A converters, 82C55 digital I/O and Ethernet. We include pre-installed firmware to run your favorite operating system from poweron so you don't have to struggle with boot loaders. Our Micro/sys embedded PC story is platform reality, not story book fancy. For more details, call or log onto our Web site today.

*** Please see our Web site for complete specifications of each board for your application.**

Visit our Web site at www.embeddedsys.com or call (818) 244-4600 today!

MICRO/SYS

©2000 Micro/sys. All brand and/or product names listed are registered trademarks of their respective owners.

3730 Park Place, Montrose, CA 91020 Voice (818) 244-4600 Fax (818) 244-4246 info@embeddedsys.com

If you are familiar with the structure of the FFT, you will recognize similarities in the FHT.

get back to the beginning of the data-flow diagram, we have the original sequence, *X*, listed in an order where the significance of the bits in the indices has been reversed. This is called bit-reversed order, and it is just as applicable to the FFT as it is to the FHT. Thus, to perform the FHT, we must first reorder the sequence, *X*, into bit-reversed order. Then we perform the butterfly computations as they are diagramed in Figure 2.

Implementation

If you are familiar with the structure of the FFT, you will recognize similarities in the FHT. One important difference is that you cannot do the FHT "in place." Computing the transform "in place" means that your initial array of time domain values is used to hold all the intermediate calculations and the resulting transform ends up in the same array. This is possible with the FFT because the butterfly operations all have two inputs and two outputs which are in the same array location as the inputs. You can load your time domain sequence into an array and then perform butterfly calculations two elements at a time. The reason this is not possible with the FHT is that the butterfly operation takes three inputs to produce two outputs. If the outputs are immediately stored in the original array, then an input to some other butterfly operation will get overwritten before it is used.

When we calculate the FHT, we must have two arrays. Each round of calculations uses one array for input (the `FromArray` in the code) and the other array for output (the `ToArray` in the code). Which array the final transform ends up in depends on whether you had to do an even or an odd number of butterfly calculation rounds. In any case, this inability to do a transform in place contradicts the opening claim of O'Neill's article that only half the memory is needed for the FHT. It is true that the FHT uses only real-valued arrays instead of the complex-valued arrays needed by the FFT, and so the arrays use half as much memory. But you need two of those arrays with the FHT and only one array with the FFT, therefore the memory usage is a wash.

In general, each round of calculations requires $N/2$ evaluations of

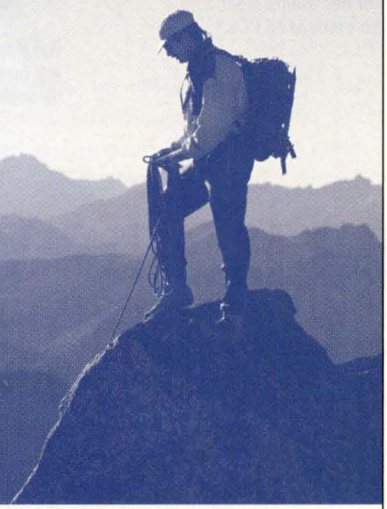
U S Software Knows Why CAD-UL is #1.

U S Software knows who to count on when they need a partner for x86 embedded development tools. And it looks like they're not alone.

Do You?

According to VDC, CAD-UL is the number one revenue generator of x86 embedded development tools worldwide.

Why? Because we're committed to providing a seamless development solution from the day you start writing your code to the day you finish testing it. And, oh yeah, CAD-UL tools work!



U S SOFTWARE
EMBEDDED EXCELLENCE

- Real-Time Operating System
- Embedded TCP/IP Protocol Suite
- Embedded File System
- Distributed Computing

www.ussw.com
(800) 356-7097

CAD-UL

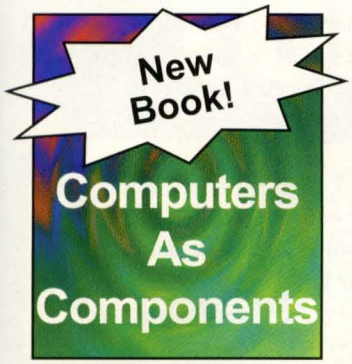
- C/C++ x86 Compiler Systems
- Symbolic Debugging Systems
- IDE: CAD-UL Workbench
- Code Coverage Tools

www.cadul.com
(877) GO CAD-UL

Morgan Kaufmann Publishers

be an expert

www.mkp.com



Computers as Components: Principles of Embedded Sytems Design

Wayne Wolf

ISBN 1-55860-541-X; 650 pages; $64.95

"This book is the first to bring embedded systems technology and techniques together under a single cover. The author provides a practical overview of the many interrelated issues that must be addressed during the complete design cycle of an embedded computer system."

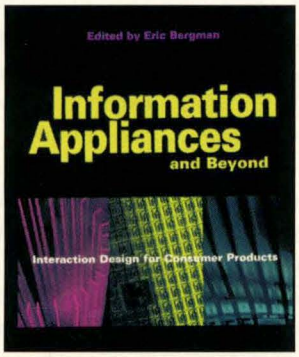
—Randolph E. Harr, Director of Research, Advanced Technology Group (ATG), Synopsys Inc.

Information Appliances and Beyond: Interaction Design for Consumer Products

Edited by Eric Bergman

ISBN 1-55860-600-9; 384 pages

$44.95



"This book is both visionary and practical: future consumer electronics, toys, and games need great usability or customers will abandon them. 'My new cell phone is so complicated that I need a two-week training class!' Not!"

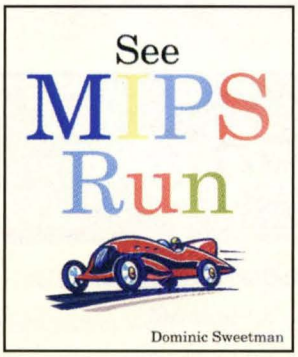
—Jakob Nielsen, Usability Guru, Nielsen Norman Group

See MIPS Run

Dominic Sweetman

ISBN 1-55860-410-3; 512 pages

$49.95



". . . this book is the best combination of completeness and readability of any book on the MIPS architecture . . ."

—from the foreword by John L. Hennessy

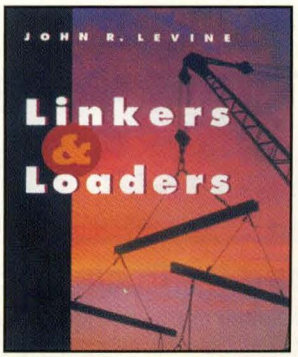
It's all here, from the nuts and bolts of a programming reference to the big picture that only a true expert can deliver.

Linkers & Loaders

John Levine

ISBN 1-55860-496-0; 272 pages

$36.95



Linkers and loaders are among the most ubiquitous software elements, but until now, there has been no comprehensive book addressing the ins and outs of these compile-time and run-time processes. Create faster, cleaner code; save space and enhance performance; learn to use linkers and loaders to their greatest advantage.

MK®

An Imprint of Academic Press
A Harcourt Science & Technology Company

Web: www.mkp.com Email: orders@mkp.com
Phone: 800.745.7323/407.345.3800
Fax: 800.874.6418/407.345.4060
Look for our titles in your local bookstore!

Equations 3 and 4. Each such evaluation involves a sine, a cosine, two multiplications, and two additions. To make things quick, the trig values can be in pre-computed lookup tables. Additionally, we recognize that in each butterfly calculation, the terms in Equation 3 are the same as the terms in Equation 4, except for a difference in signs. Referring to the example of N=16, this allows us to calculate terms $H(0)$ through $H(7)$ and get $H(8)$ through $H(15)$ almost for free. This also means that we only need values of sine and cosine for angles from zero to just under 180 degrees. In fact, I choose to implement the smallest possible trig lookup table, namely, the values of $\sin(x)$ in the first quadrant (zero to 90 degrees). The values of sine in the second quadrant as well as the values of cosine in both quadrants will come from this table using the trig identities:

$$\sin(x+\pi/2)=\sin(x)$$
$$\cos(x)=\sin(\pi/2-x)$$
$$\cos(x+\pi/2)=-\cos(x)$$

Thus to perform a Hartley transform of length N, we only need to precompute the $N/4$ values of $\sin(x)$ in the first quadrant. In order to simplify accessing this table, I have separate code for the first and second quadrants. The third and fourth quadrants are done as a byproduct using the symmetry noted in Equation 4.

Let's jump right to the heart of the implementation. The inner-most loop is essentially:

```
while(in first quadrant)
{
   From++; To++; FromRetrograde--;
   SinPtr += ButterflyBlocks;
   CosPtr -= ButterflyBlocks;
   CSTerm = *CosPtr *
        FromCButterflyWidth]
        + *SinPtr * FromRetrogradeCO];
   ToCO] = FromCO] + CSTerm;
   ToCButterflyWidth] = FromCO] -
        CSTerm;
}
```

This has been extracted and simplified from Listing 1. The actual implementation of Equation 3 is in the calculation for `ToCO]`. The calculation for `ToCButterflyWidth]` shows the use of the symmetry of Equation 4. As you can see, `CSTerm` only needs to be calculated once for each pair of Hartley transform values. Both `SinPtr` and `CosPtr` point into the pre-computed sine lookup table, but they run in opposite directions to give the sine and cosine. The values for `ButterflyWidth` and `ButterflyBlocks` change for each round. `ButterflyWidth` is the index offset between the two outputs of the butterfly. It starts at four for round three and doubles in each round. `ButterflyBlocks` is the number of Hartley transforms on the output side of each round. It is cut in half after each round, ending up at one for the last round.

I should point out here that the code actually computes double the Hartley transform at each row because we ignore the factor of 1/2 in Equations 3 and 4. This makes our final result too large by a factor of N. In many applications this error is irrelevant since only relative values are used. But if correct scaling is required, a separate scaling function, `ScaleHartleyTransform()`, is provided in the code. Without calling this function you actually get the reverse Hartley transform.

You will notice that two of the three terms in Equation 3 involve indices that increment as the target index increments. But the third term in that equation uses a backwards-running index. This has been called the "retrograde index" in the literature. It is what I call the ugly third wing of the

KEIL SOFTWARE
C51 Version 6

Attention Engineers...

Keil Software C51 is the leading C compiler development suite for the 8051 family of microcontrollers. The features in Version 6 help you write and test your embedded applications faster. Call us to get the latest CD-ROM and **FREE** evaluation tools.

Keil C51 Benefits

- µVision2 IDE & debugger speed software development and application testing
- Web-based updates keep your tools current
- Training at our facility or yours shortens the learning curve
- Answers to over 1,000 questions are available around the clock on our web site

Upgrade Today...

New Features in V6

- 32-bit programs work with Windows 95/98/NT/2000 and support long file names
- Three new optimizer levels help shrink program size up to 25%
- Integrated source browser
- Complete device database sets all compiler, assembler, and linker options for you
- Kernel-aware debugging

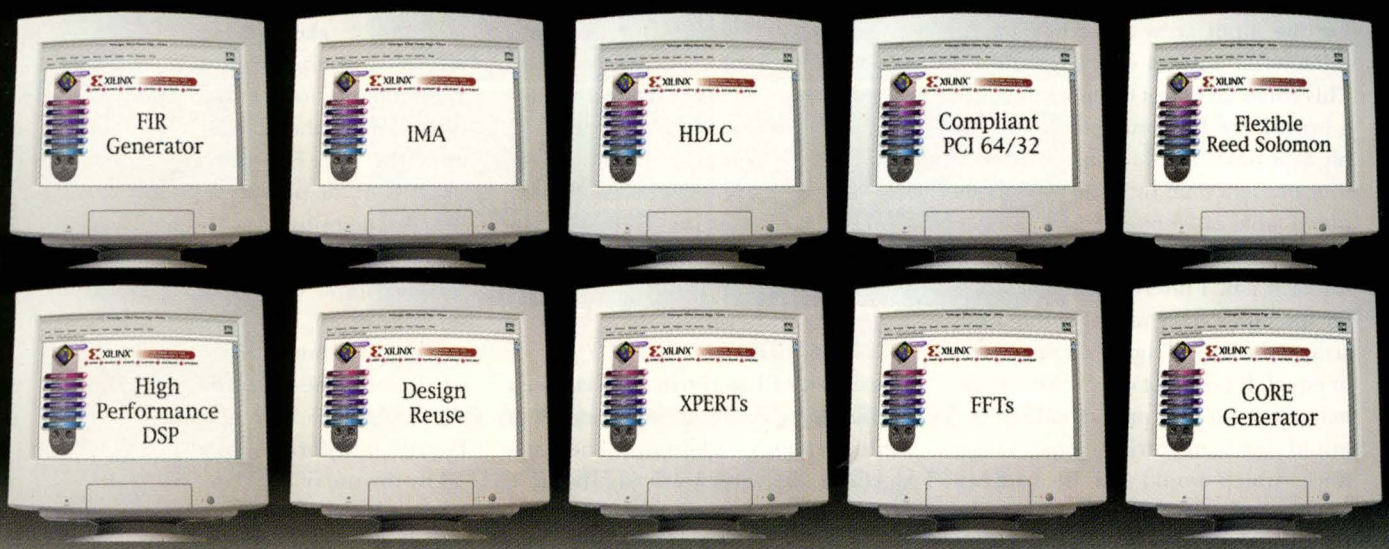


www.keil.com

Keil Software, Inc.
1501 10th Street, Suite 110
Plano, TX 75074

Toll-Free 800-348-8051
Phone.....972-312-1107
FAX.....972-312-1159

IP that clicks for everyone.



IP Center™ Xilinx IP Center is the only web portal that brings together cores, reference designs, design reuse tools, and design services. Everything the system designer could wish for.

SmartSearch™ engine

Get to the cores and design tools you want fast, and access our AllianceCORE™ and XPERTs Partners with the industry's best search engine. SmartSearch means easy access, and easy downloads.

VIRTEX-E

Complete solutions

Supporting our powerhouse programmable logic products like Virtex™ and Spartan™ FPGAs, the Xilinx IP Center offers you the total solution you need.

SPARTAN-II FPGA

Try before you buy

At the Xilinx IP Center, you can try the most efficient IP solutions available today before you buy. It's just one more reason why Xilinx is the industry leader in IP. Visit ***www.xilinx.com/ipcenter*** today and get the IP that clicks for you.

XILINX®
The Programmable Logic Company℠

www.xilinx.com

© 2000 Xilinx, Inc., All rights reserved. The Xilinx name and Xilinx logo are registered trademarks, AllianceCORE, IP Center, SmartSearch, Spartan, Virtex and XPERTs are trademarks and The Programmable Logic Company is a service mark of Xilinx, Inc.

FHT butterfly. It is represented by `FromRetrograde` in the code.

Bit reversal and special cases

The transform code starts by permuting the input array according to the bit reversal pattern described earlier. This could take a lot of time if done in a brute-force fashion, peeling off one bit at a time. I have chosen to do it four bits at a time by using a pre-computed table of 16 bit-reversed nibbles. I generate all N bit-reversed pairs and, for each pair I interchange and transfer the input array values to the other array if the pair of indices is increasing or equal. If I did not check the relative magnitude of the paired indices, we would be transferring most values twice, which would not be bad but would waste some time. Note that the bit reversal could have been done in-place, but the other work array was available so I used that instead. Because of the assumptions made in this optimized bit reversal, the implementation is limited to transforms of sequences whose length is at least 16.

The first two rounds of the FHT are so simple that I do them all at once as a special case. They involve only degenerate trig values (0 or 1). Also, the first computation of every set of butterflies is special because it has only two terms and no trig lookups. I have, therefore, structured the code to compute the butterflies for zero degrees and 90 degrees as special cases.

Interface details

In order to use the FHT code, the calling program must first generate a trig table of the appropriate size. The function `InitTrigTable()` is provided for just that purpose. Thereafter the client code may call `FastHartleyTransform()` by providing the appropriate parameters. Notice that the client code is responsible for allocating all the arrays, and that the resulting transform may be in either the original input array or the work array. Also note that no static variables are used, so this implementation is re-entrant. The return value of `FastHartleyTransform()` is a pointer to the correct result array.

As noted earlier, unlike the Fourier transform, the Hartley transform has no natural interpretation as a frequency spectrum. The most natural use for the Hartley transform is as a means to develop the Fourier transform, as shown by Equation 2. The magnitude of the first half of the Fourier transform is calculated by the `PowerSpectrum()` function.

Performance

Classically, FFT performance has been evaluated by counting the number of multiplications, additions, and subtractions that are involved. In these terms, the FHT does very well. If we disregard the simplicity of the first two rounds, we have each round requiring N multiplications and $2N$ additions or subtractions. The number of rounds is $\log_2 N$. By comparison, the FFT requires $2N$ multiplications and $7N/2$ additions or subtractions in each round. In both cases these numbers assume that no redundant operations are performed.

But this may not tell the whole story. On modern processors with floating point capability, the time required to do arithmetic may not dominate the overall running time. Time spent in computing indices and other "overhead" operations may actually take more time than all the multiplications, especially in some of the demonstration programs that are given as examples of a Hartley implementation (as in O'Neill[1]).

The implementation in Listing 1 is meant as a real working implementation, not just a demonstration. I admit that the cost-savings of performing the first two rounds as a special case does become a smaller percentage of the whole as the number of points in the sequence increases. But the savings gained by minimizing the number of function calls and parameters passed in the inner-most loop is significant for transforms of any length.

If the Hartley transform is used only as a means to develop the Fourier transform, the cost of converting to the Fourier transform using Equation 2 should be taken into account in performance comparisons. This will add $2N$ additions to the total number of operations if the FHT is used to generate the FFT. However, if the goal is to develop the power spectrum, which is the magnitude of the complex Fourier transform values, then there is no additional cost to the FHT. The operations in the `PowerSpectrum()` function amount to no more than what would be required to perform the magnitude calculations on the Fourier transform values.

By the use of a `typedef`, the code in Listing 1 can be made to operate on either doubles or floats. Table 1 contains some benchmarks run on a 75MHz Pentium.

Doing the job quickly

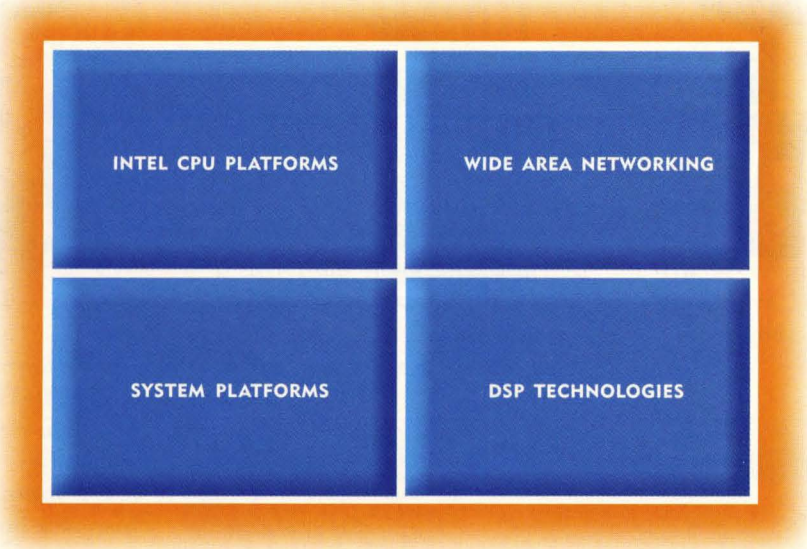
Whenever real-valued sample sequences need to be transformed into the frequency domain, the fast Hartley transform does the job in about half the time of the fast Fourier transform. The FHT algorithm is simple enough to code that there is little reason to resort to third-party libraries. With direct control over the implementation you can achieve optimizations that surpass the typical demonstration implementation. **esp**

Robert Scott is the president of Real-Time Specialties, which provides consulting services particular to embedded systems in the automotive industry. He can be reached at rscott@wwnet.net.

The code in this article is available at www.embedded.com.code/htm.

References

1. O'Neill, Mark A, "Faster Than Fast Fourier," *Byte*, April 1988, p. 293.
2. Bracewell, Ronald N., "The Fast Hartley Transform," Proceedings of the IEEE, vol. 72, no. 8, pp. 1010-1017, 1984.



HERE'S A NO-NONSENSE MESSAGE ABOUT OUR TOTAL TELECOM SOLUTION. DON'T BE FOOLED BY ITS SIMPLICITY.





Only RadiSys offers the total telecommunications solution.

A solution this complete is hard to pull off. A lot of vendors offer different parts of telecom solutions. Only RadiSys has the vision and technical expertise to deliver the total, integrated solution across both PCI and CompactPCI architectures. RadiSys has acquired Texas Micro, and is now positioned to redefine how the market is buying OEM embedded telecommunications systems. Now you can let us put the total solution together. Download our comprehensive white paper on SS7, and find out how simple it is to get the total solution from one source at www.radisys.com, or call 1-877-837-6859.

INTEL CPU PLATFORMS
State of the art Pentium,® PII, PIII, IXP and StrongArm solutions

SYSTEM PLATFORMS
Enterprise and carrier class CompactPCI and PCI system enclosures and backplanes

WIDE AREA NETWORKING
Field-proven frame relay, SS7, ATM, T1/E1, X.25, IP and other protocols and interfaces

DSP TECHNOLOGIES
Advanced TI C6x voice processing solutions-hardware and algorithms

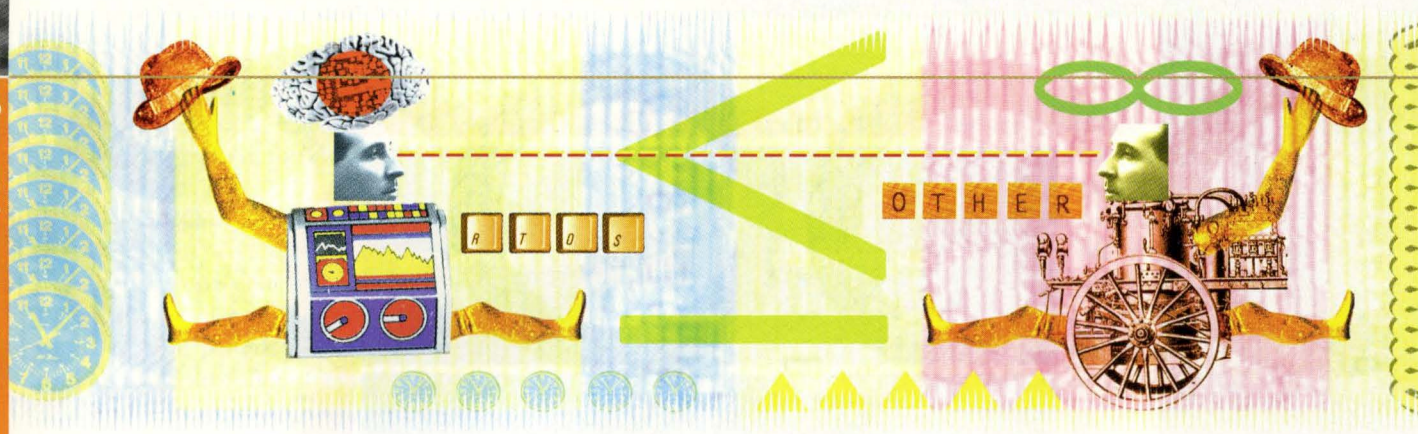
Download FREE white paper today!

www.radisys.com/SS7

Intel® Applied Computing Platform Provider

RadiSys.

MICHAEL MELKONIAN



Get by Without an RTOS

Too many simple systems use a commercial RTOS. All that's sometimes needed is a way to implement even-driven and periodic functions. Here's an easy way to do just that.

It's a fact of life that many embedded systems survive perfectly well without a multitasking real-time operating system (RTOS). I have always wondered if I could find a single criterion that would tell me if it would be an advantage or a liability to include an RTOS for a particular project.

How hard is it to guess if an RTOS is hidden inside an embedded box simply by looking at it, using it, and reading the owner's manuals? Do you think your VCR has an RTOS hidden inside? What about your car? And your mobile phone? (Maybe, just as some car models proudly state their engine size and the number of valves on the back, an embedded system should clearly state "DRIVEN BY XOS. Number of tasks: 24. RAM size: 2MB.")

Chances are your guess will be wrong. A really complicated, large system may use an infinite loop plus a couple of hard working interrupts. On the other hand, a smaller system with less than 20,000 lines of C code may have a full-blown commercial RTOS. Let's consider two extreme cases. First, consider a primitive software-driven refrigerator. Our refrigerator software simply monitors and controls the temperature and handles the door-open switch and the lamps. Even a hard-core advocate of operating systems would hardly insist on procuring a suitable RTOS.

At the other end of the scale, consider something much more complex (and not normally viewed as an embedded system), something probably sitting in a close proximity to you right now. A personal computer. I guess only a few really brave individuals would proclaim that a proper multitasking operating system on the desktop is an unnecessary luxury. But wait! You only need to look back a decade or so. Back then, the world's PCs weren't driven by multitasking operating systems. In fact, DOS simply provided some software interfaces to the hardware, some memory management functions, and a few other bits and pieces.

Somewhere in the middle, between a fridge and a desktop PC is a fine line between a "yes we need one, let's get the sales rep" and "no, let's just get a few guys and start coding."

Our fridge can become more complex because marketing now wants a keypad so the user may manually enter defrost schedules. Suddenly, we need to provide a real-time clock, a display, and a keypad. Since we now have a display, it might be wise to provide some cool graphics to entertain the users as they fetch beers from the fridge. Marketing also wants to make the software compatible with the next generation of voice-activated fridges.

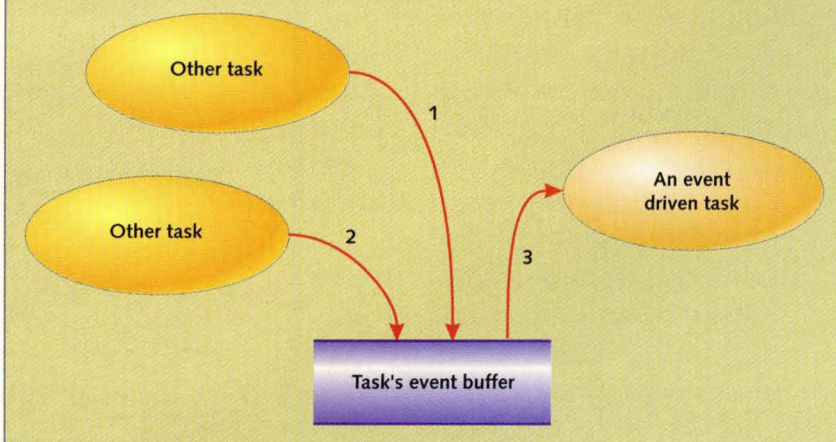
For small to medium embedded products, there is no simple answer to the "yes" or "no" question. The following drawbacks should be considered if you choose to include an RTOS:

- If you decide to buy a commercial off-the-shelf product, the one-off or per-site royalty costs can be considerable
- Your RTOS will require extra resources
- Time-critical aspects of the system may be negatively affected. For example, interrupt latencies may make your high-speed comms more difficult
- It is never easy to port an RTOS, no matter what ads in electronic and software magazines claim. Take into account the time and effort involved in porting the RTOS you choose
- If you choose a pre-emptive RTOS, remember that the power of pre-emptive multitasking comes at a price. Make sure that all the engineers you are about to hire understand that a single thread of code may be interrupted and execution passed to another task, that data shared between tasks must be protected from simultaneous access, and so on
- Many in-circuit emulators and debuggers refuse to work reliably under an RTOS. You should be prepared to calm down angry engineers who are trying to hit an elusive breakpoint while the kernel code keeps rescheduling away

LISTING 1 A sample main control loop

```
int main(void)
{
  Init_All();
  for (;;)
  {
    IO_Scan();
    IO_ProcessOutputs();
    KBD_Scan();
    PRN_Print();
    LCD_Update();
    RS232_Receive();
    RS232_Send();
    TMR_Process();
  }
  // should never ever get here
  // can put some error handling here, just in case
  return (0);  // will keep most compilers happy
}
```

FIGURE 1 Event-driven task: data/event flow diagram



BEN FISHMAN

LISTING 2 Input event structure

```
typedef unsigned int word;

typedef struct
{
    word InPtr;                                  /* Head of buffer */
    word OutPtr;                                 /* Tail of buffer */
    word Count;                                  /* Counter */
    EVENT_TYPE Store[BUFFER_SIZE];  /* Actual data buffer space */
}  INPUT_EVENT_QUEUE_TYPE;
```

LISTING 3 Input event queue implementation

```
#define OK       0
#define EMPTY    0xFFFF
#define FULL     0xFFFF

INPUT_EVENT_QUEUE_TYPE EventBuf;

void InitEventBuffer(void)
{
    EventBuf.InPtr = 0;
    EventBuf.OutPtr = 0;
    EventBuf.Count = 0;
}

static word GetEvent(EVENT_TYPE *event)
{
if (EventBuf.Count)
{
    EventBuf.Count--;
    // copy from the buffer
    memcpy(event,&EventBuf.Store[EventBuf.OutPtr],sizeof(EVENT_TYPE));
    if (EventBuf.OutPtr >= BUFFER_SIZE - 1)
    {
        EventBuf.OutPtr = 0;
    }
    else
    {
        EventBuf.OutPtr++;
    }
    return (OK);
    }
    return (EMPTY);
}
```

Listing 3 continued on p. 150

On the other hand, if you decide to take a short cut and not use an RTOS in a system where you really need one, things will be much worse. You will find that your software is getting more and more clumsy, the system keeps falling over and hangs in the most unexpected places, until finally you call it a day and start again from the scratch, using a few pieces of code you can salvage from the mess.

To find the right balance, it's important to realize that a system without an RTOS can exhibit multitasking behavior. It seems to me that often the main reason behind a decision to port an RTOS where it is unnecessary is the lack of understanding that by using some simple means and some not-so-complex code, an efficient, fast, and reasonably balanced system can be built. Moreover, should an RTOS be required some time in the future, this is no longer a painful exercise, as the system can be engineered as self-contained tasks, even in the absence of a "true" RTOS. In the following sections, I will describe such a system.

Main control loop

Our system exhibits a set of neat features, which are normally attributed to the existence of an RTOS:

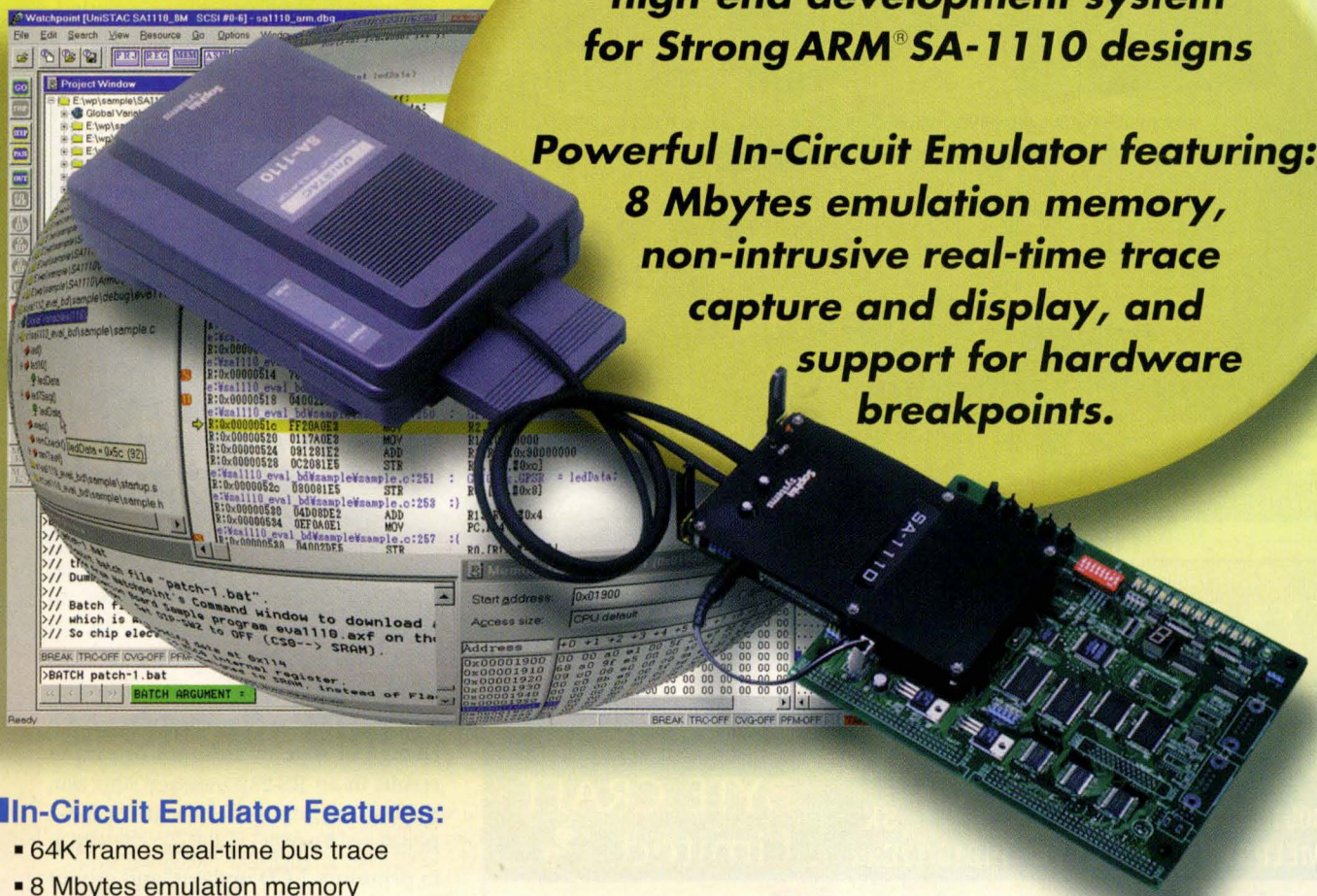
- The software is broken into standalone, well-defined tasks. It is perfectly valid to say that a particular function belongs to a particular task, and the same can be said about data structures
- Event-driven tasks are supported. These tasks have input event queues and execute only when a suitable "trigger" event arrives. Otherwise, these tasks are idle and consume very little processing time
- All tasks may send event messages to event-driven tasks
- Periodic tasks (that is, tasks that do not require a trigger to run) can be

UniSTAC™ for Strong ARM®

Sophia systems™

UniSTAC™ is a full-featured, high-end development system for Strong ARM® SA-1110 designs

Powerful In-Circuit Emulator featuring: 8 Mbytes emulation memory, non-intrusive real-time trace capture and display, and support for hardware breakpoints.



In-Circuit Emulator Features:

- 64K frames real-time bus trace
- 8 Mbytes emulation memory
- 4 hardware breakpoints
- Unlimited software breakpoints available
- Program performance analysis and code coverage

Sophia's Target Interface

- Sophia original connector-without removing CPU
- Adapters also available for BGA256

Advanced GUI source level debugger Watchpoint®

Sophia's powerful high-level language debugger hosted on Windows®95/98,WindowsNT®

Sophia Sytems solutions:

Hitachi	SH-4,SH7751,SH7065,SH7615,SH7709/09A SH7729,SH7055,SH-1,SH-2,SH-3,SH7040 SH7050,SH7011/16/17,H8S/2000,H8/300H,H8/300
ARM	ARM7TDMI®, ARM9TDMI®
Intel	StrongARM®, Pentium®, 80C186/188XL
Fujitsu	FR30, FR40
IBM	PowerPC603e,PowerPC403,PowerPC740/750
Mitsubishi	M16C Family, M32R/E
NEC	V850,V831,V832,V850E,VR4000/5000
Motorola	MPC8xx,680x0,683x0, MPC740/750/755
Panasonic	MN101C00,MN102L00,MN10300
Zilog	Z80H,Z80CMOS

Watchpoint® is registered trademarks of Sophia Systems Co.,Ltd.
UniSTAC™ is a trademark of Sophia Systems Co.,Ltd.

StrongARM® is registered trademarks of Advanced RISC Machines.
All other trademarks and registered trademarks are the property of their respective holders and should be treated as such.

URL:http://www.sophia.com/

Sophia Systems Co.,Ltd.
Telephone: 408 - 467 - 9911 Fax: 408 - 467 - 9910 Postal Address: 2107 N.First Street Suite 250 San Jose, Ca 95131
E-mail: sales@sophia.com

LISTING 3, cont'd. Input event queue implementation

```
word PutEvent(EVENT_TYPE *event)
{
     if (EventBuf.Count < EH_BUF_SIZE)
     {

        EventBuf.Count++;
        // copy to the buffer
        memcpy(&EventBuf.Store[EventBuf.InPtr], event, sizeof(EVENT_TYPE));
        if (EventBuf.InPtr >= BUFFER_SIZE - 1)
        {
            EventBuf.InPtr = 0;
        }
        else
        {
            EventBuf.InPtr++;
        }
        return (OK);
     }
     return (FULL);
}
```

executed at a pre-defined speed. Depending on the requirement, this speed can be either exactly defined or relative to the speed of execution of the main control loop
- Some basic means of inter-task communications are provided. These include stopping and restarting tasks, slowing them down, and speeding them up
- Software timers provide a convenient method of performing a variety of duties that require exact timing (for example, flash a cursor at a fixed rate). The software timers can be one-shot or run forever

In addition, as our system is not pre-emptive (tasks cannot be interrupted by another task) we have the luxury of not worrying about protecting our data with semaphores/mutexes. All of our tasks relinquish control only when the entry function of the task returns.

As an example, let's consider an embedded system with a keypad, an LCD, and an RS-232 port that runs some comms. The system also has some I/O and a parallel printer. Each change of state of an input or output results in an RS-232 message sent out, a printout, and an LCD update. Received RS-232 messages can result in printouts, LCD updates, and output status updates. We may have to start a flash pattern on a particular lamp as a result of:

- An input or output becoming active (fridge door open)
- Keypad entry (defrost schedule entered),
- Comms message received (the owner is e-mailing us to cool the beer down just before he arrives)

Let's have a quick look at our main control loop in Listing 1. Nothing new or original here. Three things are immediately obvious:

- Each function called in our infinite

PROductivity Software

C COMPILERS SUPPORTING:

8051 68HC05 68HC08 COP8™ SX
MELPS740/M38000 Z8 PIC16/17Cxx

BYTE CRAFT Limited

Features:
Optimizing C Compiler
Integrated Development Environment
Built-in Macro Assembler
BClink Linker
Runs under: Windows 95/98/NT.
Also available for DOS, HP-UX and Sun Solaris.

Byte Craft Limited
421 King Street North
Waterloo, Ontario
Canada N2J 4E4
Tel: 1-519-888-6911
Fax: 1-519-746-6751
Email: info@bytecraft.com

http://www.bytecraft.com

COP8 is a trademark of National Semiconductor Corporation

Don´t gamble with success ...



Hitex deals you winning cards!

USB Agent - USB Analyzer

The USB Agent is a full-featured USB-Protocol analyzing instrument ...
A development tool that captures, analyzes and intelligently displays all USB-signals. Right performance - Right price!

In-Circuit Emulators for ...

MX430 family:
For MSP430 microcontrollers, these emulator systems support all members of this TI family of low-power microprocessors, including the ultra-low-power MSP430x11x derivatives. These instruments are proving to be very popular.

C166 family: **DProbe167 and DBox167**
A modular family of powerful in-circuit emulator systems for the Siemens C16x variants featuring the most advanced adaptation technology: PressOn. Our C161 system with a 64 K trace buffer module is available for less than $3,500. Also supports ST-10 microcontrollers.

68HC12 family: **DProbeHC12**
For 68HC12 processors; supports maximum speed and all operating voltages; plug and play (no jumpers, no switches); additional BDM interface cable; flash programming built-in; PlugOn trace with 32k frames. Move up to super advanced features with DBoxHC12.

8051 family: **MX51/AX51**
True real-time emulation for more than 500 variants from all manufacturers. Including derivatives like Atmel 89S8252, Intel 8x931x and Siemens C505C.

and more:
251, 68k, CPU-32, 80x86, 68HC11, Pentium®Processor

hitex
DEVELOPMENT TOOLS

Hitex USA
710 Lakeway Dr., Suite 280
Sunnyvale, California 94086

	Tel.:	(800) 45-HITEX
	Tel.:	(408) 733-7080
	Fax:	(408) 733-6320
	E-mail	info@hitex.com
Hitex Germany	Tel.:	(0721) 9628-133
	Fax:	(0721) 9628-149
Hitex UK	Tel.:	(01203) 69 20 66
	Fax:	(01203) 69 21 31
Hitex Asia	Tel.:	(65) 74 52 551
	Fax:	(65) 74 54 662

www.hitex.com

Advanced technology that brings results...

"The best emulator that I ever used!"

Benefit from our world-wide support:

Austria Tel: +43-1-259727/20 Belgium Tel: +31-77-3078438 PR China Tel: +86-10-62383376 Czech Republic Tel: +420-2-66316661 Denmark Tel: +45-43-424742 France Tel: +33-1-39611414 India Tel: +91-22-8520817 Israel Tel: +972-3-7657666 Italy Tel: +39-0434-633500 Malaysia Tel: +603-7167591 Netherlands Tel: +31-77-3078438 Pakistan Tel: +92-51-822075 Poland Tel: +48-22-7556983 Russia Tel: +7-812-3252792 Singapore/Indonesia Tel: +65-7496168 South Korea Tel: +82-2-645-0386 Spain/Portugal Tel: +34-91-6401234 Sweden/Norway/Finland Tel: +46-87-404580 Switzerland Tel: +41-1-3086666 Taiwan Tel: +886-2-26983456 Turkey Tel: +90-312-4472700 PBX

Visit us at Embedded Systems Conference 2000, Booth# 930

How do we achieve different execution speeds if the functions are called from the same control loop?

loop represents an independent task
- Each of these tasks must return in a reasonable time, no matter what thread of code is being executed
- We have no idea at what frequency our main loop runs. In fact, the frequency is not constant and can significantly change with the changes in system status (as we are printing a long document or displaying a large bitmap, for example)

So what happens inside the functions called from our infinite loop?

The majority of tasks in our system are event-driven tasks. They do not execute until a suitable message is received. Each of these tasks has a dedicated input event queue. For example, `IO_ProcessOutputs` is an event-driven task. It handles the state of outputs and does nothing if there are no state changes to be performed on the outputs. However, if an output needs to be turned on, an event message is sent to this task. In our system, three tasks will send event messages to the `IO_ProcessOutputs`:

- The input scanner (`IO_Scan`) task, when input state change dictates an output state change
- The RS-232 receive task, when an RS-232 message is received, requesting an output to be turned on/off
- The keypad scanner task, (`KBD_Scan`), when an entry is completed with the request to turn on/off an output

The handling of the events inside `IO_ProcessOutputs` is exactly the same, no matter which of the three tasks has sent the event. (The event-driven task structure is described in the next section.)

Other tasks in our system are periodic. They run without a trigger event. Some need to run faster, others slower. For example, we may need to scan the inputs at a much faster rate than the LCD update. How do we achieve different execution speeds if the functions are called from the same control loop?

I have also promised to provide some simple means of inter-task communications. For example, we may want to stop scanning the inputs after a particular keypad entry and restart the scanning after another entry. This would require a call from a keypad scanner to stop the I/O scanner task. We may also want to slow down the execution of some tasks depending on the circumstances. Let's say we detect an avalanche of input state changes, and our RS-232 link can no longer cope with sending all these messages. As a solution, we would like to slow down

LISTING 4 Sending an event to a task

```
// somewhere in RS232 module
OUTPUT_EVENT_TYPE OutputEvent;
OutputEvent.NewState = 1;          // new state - on
OutputEvent.Number = 1;            // turn on output number one
OUTPUT_PutEvent(&OutputEvent);     // inject an event
```

LISTING 5 Example of an event-driven task

```
// this task is called from main control loop
void IO_ProcessOutputs(void)
{
     word ret;
     OUTPUT_EVENT_TYPE OutputEvent;

     // our normal execution counter processing goes here..
     // ..
     // end of execution counter processing

     if ((ret = OUTPUT_GetEvent(&OutputEvent) != EMPTY)
     {
        // buffer not empty
        // process OutputEvent   turn on/off the required output
        IO_OutputStateChange(OutputEvent.Number, OutputEvent.NewState)
     }
}
```

Shrink Your Designs 66% With 44-Pin Chip Scale COP8™ OTP

National's COP8SGR/SGE Microcontroller Packs 40 I/Os in 1/3 Area of 44P TQFP

- COP8SGR: 32K OTP with 512 Bytes RAM
- COP8SGE: 8K OTP with 256 Bytes RAM
- Full Duplex USART
- 2 Analog Comparators
- User Selectable WATCHDOG, Clock Monitor, and Power-on-reset Options
- Four 16-Bit Timers (3 Multifunction, 1 Free Running)
- Patented EMI Reduction Technology (-20 db)
- Designed for High-noise Immunity
- Absolute Code Security
- 8 High-current Outputs
- Multiple Fail-safe Features
- **Fully Compatible With National's COP8.COM Internet Connectivity Solution!**

Ideal for Small Board Designs, Portable Appliances, Cordless Phones, and Security Systems

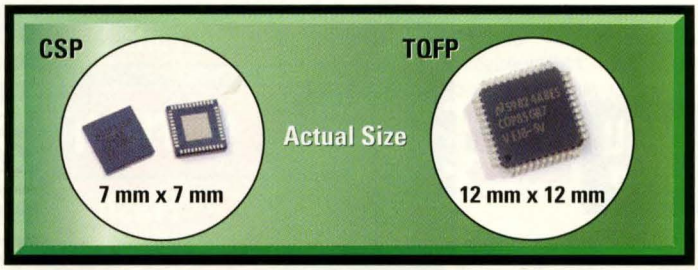
National Backs All Products With Complete Technical Support and Proven High Volume Manufacturing Expertise.

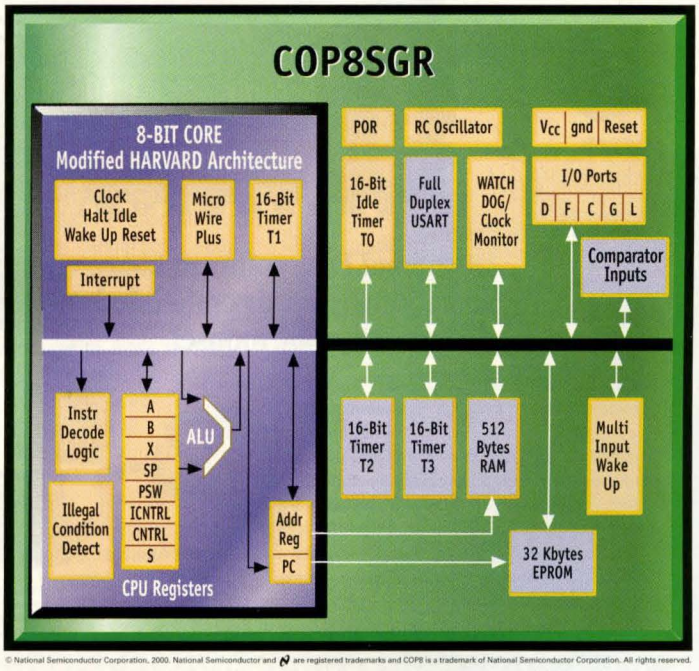
For More Information on COP8SGR/SGE:
www.national.com/cop8
1-800-272-9959

Free CD-ROM Data Catalog Available at:
freecd.national.com

The COP8SGR Series of Microcontrollers is Supported with a Full Set of Development Tools Including:

- Real-time Emulation and Debugging Toolset for $299.00
- Integrated Development Toolset Combining C-Compiler, Assembler, Linker, Librarian, and Debugger in Windows Environment





© National Semiconductor Corporation, 2000. National Semiconductor and ℕ are registered trademarks and COP8 is a trademark of National Semiconductor Corporation. All rights reserved.

National Semiconductor

In our implementation, each event-driven task has a single input queue, implemented as a ring buffer.

the I/O scanner task from the RS-232 sending task. All this is achieved using the proven and reliable, if extremely unoriginal, technique of execution counters, and is described in the Periodic Tasks section.



In addition to all these features, we need to perform a variety of small but important duties. For example, we want to dim the LCD exactly one minute after the very last key was pressed. We also want to flash a cursor on the LCD at a periodic, fixed and exact frequency. Since dedicating a separate task to each of these functions is definitely overkill, we handle them through software timers. Rather than being explicitly called from the main control loop, the function to turn the cursor on/off is called indirectly from `TMR_Process` task, which is the only non-user-defined task in the main control loop.

Event-driven tasks

Figure 1 shows the concept of an event-driven task. In our implementation, each event-driven task has a single input queue, implemented as a ring buffer. Two functions are provided, PutEvent, to be used by any task when an event needs to be inserted in the queue, and GetEvent, to extract the message. GetEvent will be used exclusively by the task itself and will not be called by other tasks. See Listing 2.

Please note that the `EVENT_TYPE` structure is unique for each task. In other words, the task itself determines the format of events it expects to receive. For example, in the `IO_ProcessRequests` task, we would want to include the number and the new state of the output. In the printer task we could simply construct the `PRN_EVENT_TYPE` as a buffer large enough to contain a single null-terminated string. Since `EVENT_TYPE` is likely to be different for each task, the user will have to define a unique structure, based on `INPUT_EVENT_QUEUE_TYPE`, for each event-driven task. Moreover, each task will have it's own `GetEvent`, `PutEvent`, and an initialization function.

A simple ring buffer structure allows us to achieve asynchronous reads and writes to the buffer, and to

Drawbacks to Web Applications

by Adam Kolawa

Are Web-based applications the development solution everyone has been waiting for? Not surprisingly, the answer is "no". The chief problem with Web applications is that they are stateless. To be sure that they are viewing or testing the right page, developers must make sure their state variables are properly configured for each page--a tedious and time-consuming task.

The second major problem with Web-based applications is a significant usability problem. In a client-server application, when you write the client you control user movement through menus. By providing certain menu items on certain pages, you restrict the way users can move through the application and ensure that they will take a logical path. On the Web, browsers typically contain a Back button and Forward button, which seem convenient to the user but can get a dynamic site out of sync. Programmers have tried various methods to work around this obstacle, but with little success.

A third problem is the issue of how to refresh dynamic Web site pages with updated information from the server. If you have small areas of the screen that are supposed to be interactive and are constantly changing, you face the challenge of providing the user with an updated display and connecting to the server. At first, developers tried to use HTML frames or applets to solve the problem, but frames are not portable and applets are too slow to download.

Developers have also tried to use Java scripts as a solution. The problem is that when the client is executed, its actions will not matter unless the server can connect to it. After users modify the page and everything looks right, they need to click the Submit button, which tells the server to perform the actions they wanted. Without this functionality, all the information is lost. In client-server applications, however, users can immediately register to the server what they are doing up until the moment an application dies.

This debate also raises the issue of speed. Certain applications (for example, sales databases) require quick page changes, which are not possible with a browser.

So is client-server dead? The answer is definitely "no". Client-server is useful for developers who are building speed-critical applications that need to provide interactivity and ease of use. But client-server has been replaced by Web development in areas in which proprietary clients can be replaced by browsers.

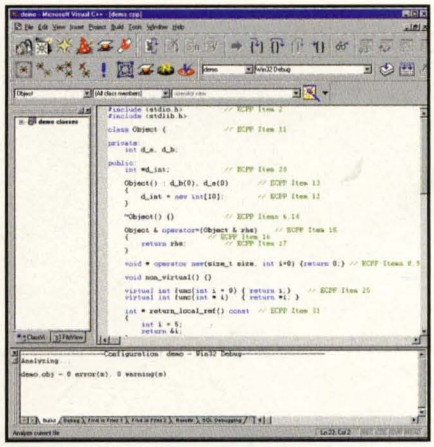
Adam Kolawa, Ph.D., is Chairman and CEO of ParaSoft. You can reach him at ak@parasoft.com

ParaSoft®
codewizard®

CodeWizard® prevents C/C++ errors and slashes your debugging time

"CodeWizard is awesome. I am recommending our management to purchase at least 5 copies....ParaSoft has a lot to offer to the world of C++ development." ***-Piyush Shaw, Arrowsmith Technologies, Inc.***

CodeWizard® is a tool that helps you prevent errors by automatically performing static analysis on C/C++ code. The first thing you'll notice about CodeWizard is that it cuts down on the thing you hate the most: debugging. Preventing errors with CodeWizard is painless and saves you time in the long run.



CodeWizard enforces over 120 sophisticated C/C+ coding standards.

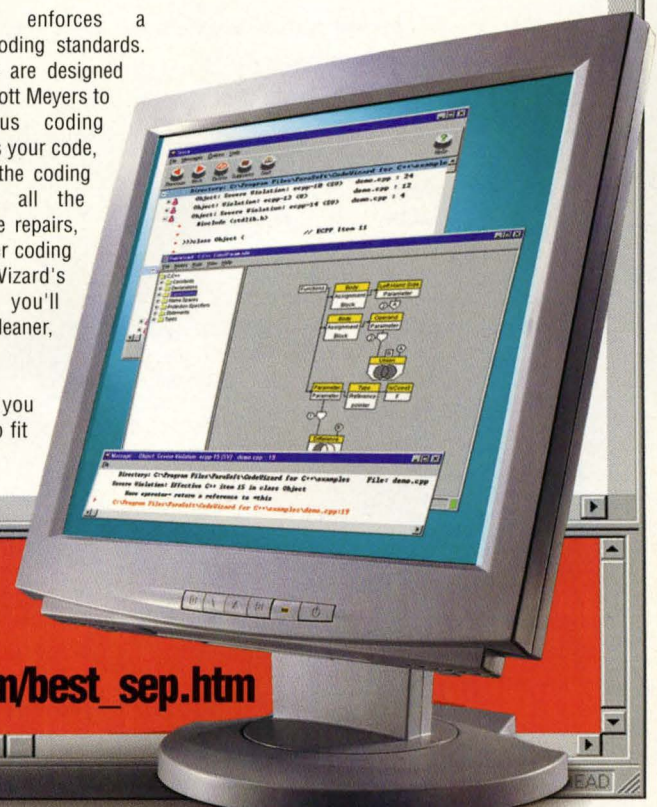
CodeWizard automatically enforces a sophisticated set of C/C++ coding standards. These language-specific rules are designed by industry experts such as Scott Meyers to help you avoid dangerous coding constructs. CodeWizard parses your code, clearly displays violations of the coding standards, and gives you all the information you need to make repairs, including suggestions on better coding constructs. Follow CodeWizard's coding suggestions, and you'll immediately begin to write cleaner, more reliable code.

The key to CodeWizard is that you can customize the program to fit your development schedule.

With a few clicks, you can tell CodeWizard to enforce only the standards that are most important to you at a particular stage of your project. Using the RuleWizard feature, you can point and click to build your own sophisticated coding standards. RuleWizard guides you through the coding standard building process and lets you know when your coding standard is ready for enforcement by CodeWizard.

You won't have any problem adding CodeWizard to your arsenal of development tools, because it installs directly into Windows and UNIX development environments. In Microsoft Developer Studio, you'll begin testing with a single click of the CodeWizard icon. On UNIX platforms, CodeWizard is a wrapper around the compiler. Learn a few simple commands, and you'll be on your way to cleaner code.

If you would like to prevent errors with a customizable tool that conforms to your development cycle, download CodeWizard today at www.parasoft.com/prevention_sep.htm, or call (888)305-0041 for more information.



ParaSoft®

Download your FREE DEMO at:

www.parasoft.com/best_sep.htm

ParaSoft is a registered trademark, CodeWizard and Insure++ are registered trademarks of ParaSoft, and C++Test is a trademark of ParaSoft.

LISTING 6 Execution counter processing

```
volatile unsigned int LCD_ReloadValue = LCD_TASK_DEFAULT_SPEED;
volatile unsigned int LCD_ExecCounter = LCD_TASK_DEFAULT_SPEED;

void LCD_Process(void)
{
#ifdef EXACT_TIMING
   disable();   // disable interrupts for a short instance
#endif
   if (LCD_ExecCounter == TASK_DISABLED)
   {
#ifdef EXACT_TIMING
         enable();
#endif
         return;
   }
#ifdef EXACT_TIMING
      // note contention possible with interrupt routine
   if (LCD_ExecCounter)
#else
   if (--LCD_ExecCounter)
#endif
   {
#ifdef EXACT_TIMING
         enable();
#endif
         return;
   }
   // if we reach this point, we are about to run our task
   // reload the execution counter
   LCD_ExecCounter = LCD_ReloadValue;
#ifdef EXACT_TIMING
   enable();
#endif
   // application task code follows
   // ...
   // end of function
}

// and this is our fixed time interrupt routine...
interrupt void TIMER_IRQ_10ms(void)
{
     // other stuff
#ifdef EXACT_TIMING
     if ((LCD_ExecCounter != TASK_DISABLED) && (LCD_ExecCounter))
       --LCD_ExecCounter;
#endif
     // other stuff
}
```

store up to `BUFFER_SIZE` entries. Any other task, or the task itself, can inject events of `EVENT_TYPE` into the input ring buffer. The events will be inserted at the position of `OutPtr`, which will grow until the "owner" task executes and reads events from the buffer. When the task extracts events, the position of `InPtr` is adjusted. When `OutPtr` and `InPtr` are equal, the buffer contains no unprocessed events. The "Count" member contains the number of unprocessed events in the buffer.

Listing 3 shows the implementation of `InitEventBuffer`, `GetEvent`, and `PutEvent`. It's quite simple to imagine how other tasks would activate the outputs. All they need to do is to construct an event of `OUTPUT_EVENT_TYPE` and call `OUTPUT_PutEvent`, as shown in Listing 4.

With all other tasks happily using the `OUTPUT_PutEvent` function, the only thing we need to do is implement our output controller task. This is straightforward, as shown in Listing 5.

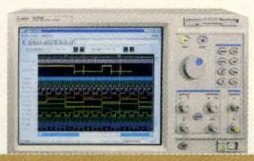
Note that by simply changing "if" to "while" in the function above we can significantly change the behavior of the task. Instead of processing one event per iteration of the task, it will process all the events in the buffer. Another valid idea would be to implement a mixture of two methods, where a task extracts a maximum of X events at a time. Even better if X can be changed by other tasks. Changes to Listing 5 to implement these ideas are straightforward, and I leave them to the reader.

Periodic tasks

Nothing is stopping us from calling any function from the main control loop. However, we have to consider two things first:

- We don't want to call our tasks too often. Why would we call a watchdog refresh function every 20ms if





16702B Logic Analysis System

- Large touch screen flat panel display
- Knobs and dedicated hot keys
- Built-in 40X CD-ROM drive
- 100BaseT/10BaseT Ethernet
- Supports up to 400 MHz state speed/32M depth

We've given the learning curve a new direction.

Our job is to make your job easier. That's why our latest logic analyzer employs a huge 12.1 inch touch screen that's easy to see and use, transforming your task into much less of one. It also utilizes hot keys and front panel knobs that allow you to instantly access frequently used windows, and break many functions down into just a couple steps. Add to that Eye Finder technology that automatically adjusts for the optimal set-up and hold conditions on high speed buses.

And modules from your 16500 can be used in the Agilent Technologies 16702B Logic Analyzer, which supports future generation modules as well. Meaning you'll be performing effortless logic analysis for years to come.

To find out more, contact us. We'll send you a free video that will show you just how simple it really is.

www.agilent.com/find/LAS-DSD
1-800-452-4844*, Ext. 7076

Agilent Technologies
Innovating the HP Way

©2000 Agilent Technologies ADEP3464004/ESP *In Canada, call 1-877-894-4414, Ext. 7077

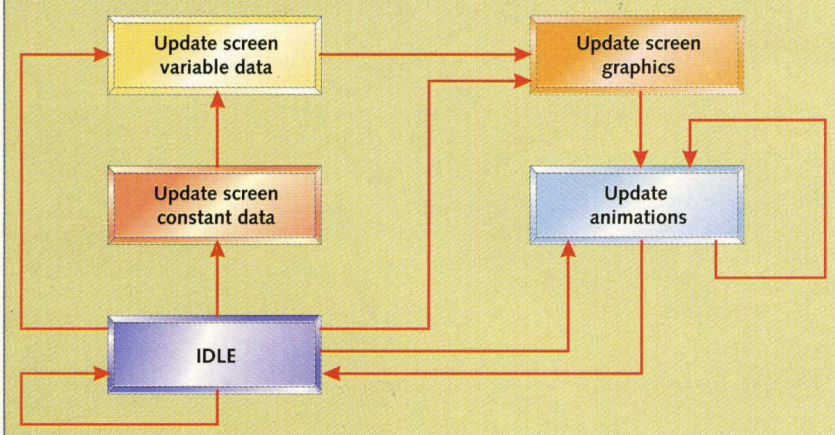
The reason for using a variable rather than a constant for reload value is that this helps us to dynamically manipulate the execution speed of the task.

LISTING 7 A paced memory checking task

```
void MEMORY_Check(void)
{
      static unsigned int State = 0;
      static unsigned int Csum;
      byte *Ptr;
      // the usual exec counter processing goes here

      if (State == 0)
      {
        // zero the csum
        Csum = 0;
      }
        // derive the memory pointer
        Ptr = (void *)(RAM_START + State*100);  // do CRC 100 bytes at each
                                                // iteration
      AddToCsum(&Csum, Ptr, 100);
      if (++State == 100)
      {
        if (Csum != GetRAMCsum())
        // error   csum mismatches   do something
           Panic();
        State = 0;
      }
}
```

FIGURE 2 State-driven LCD update task—the state transition diagram



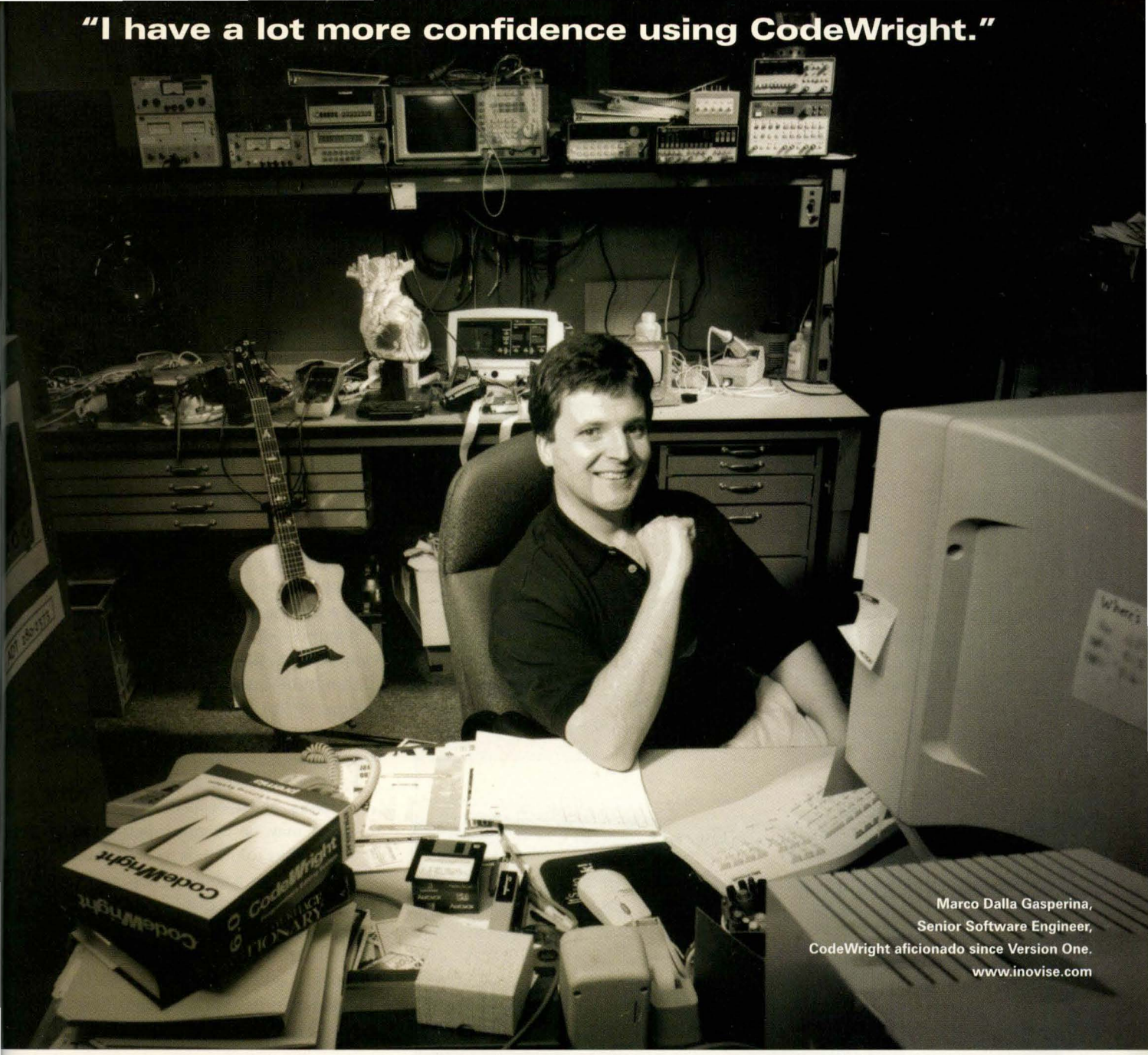
we really only need to do it every second?

- We don't want to delay the execution of other tasks for too long

In regards to the first problem, a mechanism is available to help slow down the tasks. This can be done in terms relative to the main control loop, or in absolute terms (we will call it exact timing). In both cases, we need two variables per task. One is called an *execution counter* and the other *reload value*. The execution counter counts down from reload value. When it reaches zero, the task is called, otherwise the entry function to the task exits without further processing. The reason for using a variable rather than a constant for reload value is that this helps us to dynamically manipulate the execution speed of the task. See Listing 6.

Some points about the code:

- In the exact timing system, the execution counter is decremented in a fixed frequency interrupt routine, while in relative timing system the task itself decrements the counter. In an exact frequency system, we know how frequently our task will execute (minus the latency of other tasks). By setting `LCD_TASK_FREQUENCY` to 100, and using a 10ms interrupt, we know that our task will execute every second plus the execution time of other tasks that may have been called before ours, but after the LCD tasks execution counter was decremented to 0
- The `volatile` keyword will prevent optimization by the compiler. Some optimizing compilers will otherwise (in an exact timing system) assume that the `LCD_ExecCounter` will always be set to a non-zero value, especially if the 10ms interrupt handler is coded in a different file
- `TASK_DISABLED` will be `#defined` to the largest possible unsigned integer. Setting the execution counter



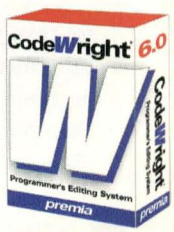
CodeWright®

The Programmer's Editing System™

Marco is one of the team of programmers bringing early low cost detection of heart disease to reality. Inovise Medical is developing software that uses waveforms from a standard EKG test to provide a graphical view of the size and location of damage to a patient's heart.

The editor behind the code: CodeWright. When precision counts the most.

Download an evaluation copy at **www.starbase.com/embedded** or call **(800) 491-6784**.



$299
Single User
Multi-User License available
Windows 32-bit platforms

starbase

Premia, a Starbase company

The periodic tasks must be written in a manner that will guarantee that they return in a reasonable time. This is not always easy.

LISTING 8 A periodic LCD update task

```
word WinTaskState = WIN_TASK_IDLE;

void Window_Update(void)
{
     //.. reload counters go here
     switch (WinTaskState)
     {
        case WIN_TASK_IDLE:
           return;
        case WIN_TASK_CONST:
           UpdateWinConst();
           WinTaskState++;
           break;
        case WIN_TASK_VARS:
           UpdateWinVars();
           WinTaskState++;
           break;
        case WIN_TASK_GRAPH:
           UpdateWinGraph();
           WinTaskState++;
           break;
        case WIN_TASK_ANIMATIONS:
           if (UpdateWinAnim())
           {
              // all work done, back to idle state..
              WinTaskState = WIN_TASK_IDLE;
           }
           // if not all animations updated, stay in this state for
           // a little while..
           break;
        default:
        // panic

        }
}
```

to TASK_DISABLED disables the task until further intervention. This can be done from any other task. For example, an important event can disable our printing process until further notice (a simple form of inter-task communications)

- Our 10ms interrupt is undoubtedly doing a fair bit of processing in the exact timing system. However, with a samll number of tasks, it's hard to believe that the processor will be struggling to do the comparisons and pre-decrements. It will be a good idea to set this interrupt to a lower priority than more critical interrupts in your system (or allow it to be pre-empted by other interrupts). It is debatable if some neater mechanism should be introduced (such as delta queue) to prevent too many counters decremented from an interrupt routine. As the number of tasks for the system of our size rarely exceeds 20 to 30, the delta queue mechanism may be unnecessary (see Software Timers for discussion of delta queues)
- In exact timing systems, care should be taken to avoid preemption of the task by a fixed-time interrupt routine. The exact mechanism is CPU- and compiler-specific. In some cases, it may not be a problem, as 16- or 32-bit reads and writes may be atomic operations. The easiest solution is to disable all interrupts for a short period in the section of code dealing with the execution counter (as shown in Listing 6, by using "disable" and "enable" function calls). This is acceptable in the vast majority of cases

The periodic tasks must be written in a manner that will guarantee that they return in a reasonable time. This is not always easy. For example, consider a background memory checking task. If the amount of memory to check is large, we will not want to do the whole check in one iteration, as it may take too long. A useful approach to such tasks is to build them as finite state machines, with each iteration taking it to the next state. The processing in each state is limited to the longest permissible time-slice. Sample code for memory checking function is shown in Listing 7.

This piece of code will execute 100 times to check 10,000 bytes of RAM. We control how often the memory check is done using the usual execution counter and reload value technique. Another variation of the same idea is shown in Listing 8. The task is doing nothing until somebody sets the `WinTaskState` to any non-idle state. Up to four following iterations will then update relevant portions of information in the window. For example, to repaint the entire window, we would set `WinTaskState` to `WIN_TASK_CONST`, which will update all the constant information, all variable information, and all graphics and animations in the window. After this update the window update task will become idle again until the next trigger via a `WinTaskState`. Conveniently, if we only want to update the animations, we can set the `WinTaskState` to `WIN_TASK_ANIMATIONS` (in this case, we will not repaint the constants, variables, and static graphics on the screen). Figure 2 shows the LCD update task's state transition diagram.

Software timers

The software timers add a "real" multitasking behavior to our system. There are hundreds of actions that require events to happen at a fixed time once or periodically. Some examples are:

- A blinking cursor
- An output that needs to be periodically switched on/off
- The need to show a user message that will disappear or change after a fixed period
- The need to light up the keypad or the LCD and turn it off after a period of inactivity
- A short reset pulse to the printer
- A flashing lamp

Do we really want to dedicate periodic tasks to these little, yet important, functions? Most of these will also require exact timing, making our 10ms interrupt work very hard. Our

The software timers add a "real" multitasking behavior to our system.

LISTING 9 Expired timer data structure and ring buffer

```
typedef struct
{
    word  ID;   /* Timer ID */
    dword Parameter;   /* Parameter */
    void (*TimeoutHandler)(word Id, dword Parameter);
} EXPIRED_TIMER_EVENT_TYPE;

typedef struct
{
    byte InPtr;   /* Head of buffer */
    byte OutPtr;  /* Tail of buffer */
    word Count;   /* Number of expired timers in the buffer */
    EXPIRED_TIMER_EVENT_TYPE Store[MAX_TIMERS];   /* Actual data buffer space
*/
} EXPIRED_TIMER_BUF;
```

LISTING 10 Application interface of the software timer module.

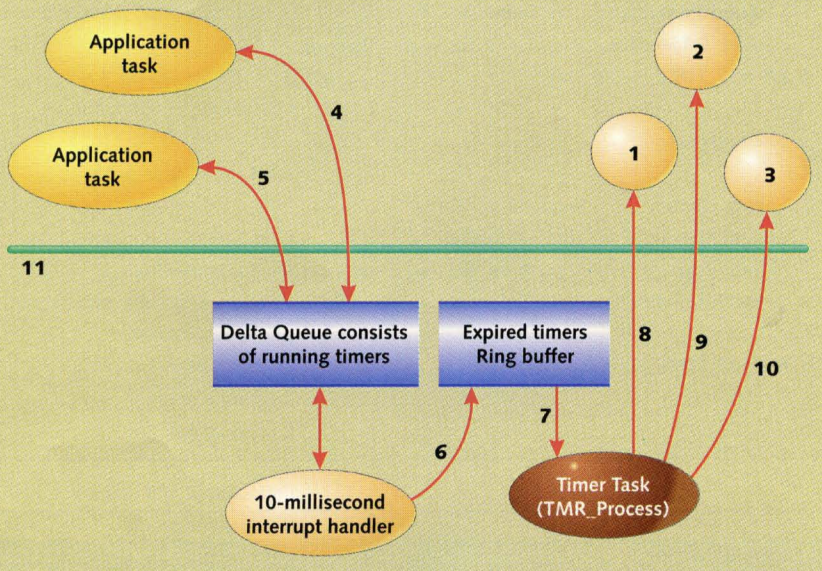
```
word TMR_InstallTimeoutHandler(word timer_handle, void(*timeout_func)(word, dword))
word TMR_Start(word timer_handle, word timeout, dword parameter);
word TMR_Stop(word timer_handle);
```

LISTING 11 Software timer task

```
EXPIRED_TIMER_EVENT_TYPE *Timer;
void TMR_Process(void)
{
   if ((Timer = TMR_GetEvent()) != INVALID_TIMER)
   {
      if (Timer->TimeoutHandler != NULL)
      {
         // call the user timeout function indirectly..
         Timer->TimeoutHandler(Timer->ID, Timer->Parameter);
      }
   }
}
```

Decrementing hundreds of timer counters at each 10ms interrupt seems like a bad idea. Instead, we will use the well-known technique of a delta list.

FIGURE 3 Software timers module—data/event flow diagram



Notes:
1,2,3 - Application-provided timeout handler functions (e.g. blinking cursor)
4,5 - Application tasks have read and write access to the delta queue via calls to TMR_Start and TMR_Stop
6 - Write via TMR_PutEvent
7 - Read via TMR_GetEvent
8,9,10 - Application timeout handler functions are called directly from TMR_Process
11 - Thick green line separates application code from system code

main control loop will become long and messy. So we must find a neater solution.

You may have noticed a call to `TMR_Process` in the main control loop. This is the only non-user-defined task in our system. In our implementation, the `TMR_Process` task itself is an event-driven task, and works exactly as the user-defined event-driven tasks described in this article.

Internally, the timer task could use something like the structures in Listing 9 to define the timer task input event queue. We use the `Parameter` field to add some flexibility to our module. The application may choose to use it for a variety of purposes, or simply ignore it.

The software timer module in Listing 10 provides the three essential functions for use by the application tasks.

Before calling `TMR_Start` and `TMR_Stop` functions, a user-defined timeout function for each timer must be installed. This is done by calling `TMR_InstallTimeoutHandler` with two parameters: a unique timer ID and a pointer to a timeout function. After this, a timer can be started and stopped by using the `TMR_Start` and `TMR_Stop` functions.

The implementation of `TMR_Process` is simple. Internally, it will use the `TMR_PutEvent` and `TMR_GetEvent` functions, familiar to us from the discussion of the event-driven tasks. See Listing 11.

A data/event flow diagram in somewhat simplified RT/SASD notation is shown in Figure 3.

You have already guessed that the 10ms interrupt is, in our case, the only provider of the timers into the expired timers ring buffer. As we can have hundreds of software timers running at any given time, we should implement this part of our system in a slightly more intelligent way. Decrementing hundreds of timer counters at each 10ms interrupt seems like a bad idea. Instead, we will use the well-known technique of a delta list. The timers are inserted in the delta queue based on their time-out values. Only the timer that is next due to expire is decremented in the interrupt routine. For example, if we have timers that are set to expire in 10, 60, and 200 clock ticks, our delta queue looks like:

10
50 (equals 60 minus 10)
140 (equals 200 minus 60)

And the only one decrementing is the timer with 10 ticks left.

The slight drawback of this technique is the fact that we need to do a bit more processing when we add or remove a timer to/from the delta queue. Plus, there is a tricky case of two timers due to expire at the same time (some delta queue implementations cheat and add an extra tick to avoid this situation).

So what happens when the 10ms interrupt routine decrements the head of the delta queue to zero? Of course, it doesn't call the timeout handler directly, as this would take a large, application-determined amount of time and can't be executed with interrupts disabled. Instead, it removes it

We don't have to remind you how important it is to get your design into the market quickly and efficiently. But we would like you to know that our wide range of SoC verification tools excel at doing just that.

There's ModelSim®, for example, the leading tool for mixed language simulation. XRAY®, the only product to support debugging of multiple cores. Seamless® Co-Verification Environment, the industry standard by far. And a host of other hardware, software, IP and consulting solutions designed to help you increase productivity, reduce steps and slash verification time across every SoC development stage.

SoC VERIFICATION
TOOL SET

Mentor Graphics®

So be sure to start with our tools on the front end. Then you won't have to worry about the other one. For more information, call 1-800-547-3000, or visit www.mentor.com/soc to get our new online SoC newsletter.

Our SoC verification tools can save the only thing more valuable than your time.



©2000 Mentor Graphics Corporation. Seamless, XRAY, ModelSim and Mentor Graphics are registered trademarks of Mentor Graphics Corporation.

The point where a proper RTOS becomes more of an aid than an impediment is hard to define and depends on the specific system.

from the delta queue and simply calls the `TMR_PutEvent` with the ID and parameter of the expired timer. This inserts the expired timer into the expired timer ring buffer.

I will leave the full implementation of the software timer module to the reader. An exercise in delta queue coding always proves to be a good brainteaser. There are some interesting aspects to be taken care of. For example, when the application calls `TMR_Stop`, we must make sure that we handle the removal of the timer from the delta queue (if the timer is running) or the expired timers ring buffer (if it has expired but the `TMR_Process` hasn't pulled it out yet).

With all the complexity hidden in the software timer module, the application code implementing a blinking cursor looks quite simple indeed. See Listing 12.

Tidying up

The simple and well-known methods described in this article are sufficient to build reasonably complex and well-balanced applications. However, it would be nice to tidy up a number of unstructured elements in the examples in this article. Some of the things we don't like:

- Having too many globals called `XXX_ExecutionCounter` and `XXX_ReloadValue`. It would be much better if they were grouped in a single data structure
- We don't like the fact that each task in the very beginning does exactly the same processing of the execution counters. It would be nice to handle it in a single piece of code
- We would like to refer to the tasks by their unique identifiers. Instead of writing `IOSCAN_ExecutionCounter = TASK_DISABLED`, it is much nicer to write `OS_Stop(TASK_ID_IO_SCAN)`
- We do not want to #define values for reload values for each task. We would rather read them from a tidy data table
- When tasks have to be slowed down or speeded up, for modularity, rather than directly manipulating the reload/execution values for the task, it is much better to do this via a generic function call. For example, to halve the execution speed of the I/O scanner task:

```
OS_SetReloadVal(TASK_ID_IO_SCAN,
  OS_GetReEntryValue(TASK_ID_
  IO_SCAN)*2))
```

This can be done by introducing a uniform task structure, which would include common aspects of the task functionality. All RTOSes no matter how big or tiny, have a task table in one form or another. And it is not surprising that the result of our little discussion may evolve into something reminiscent of a "real" RTOS.

Indeed, we were talking about tasks, input queues, sleep and idle states—concepts normally found in RTOS manuals. So, with a little effort, the reader may create his own, "almost real" RTOS, with some help from the discussions and examples in this article.

Simple, yet complex

Reasonably complex real-time embedded applications can be built without a "true" multitasking RTOS. Unfortunately, the point where a proper RTOS becomes more of an aid than an impediment is hard to define and depends on the specific system.

LISTING 12 Example of implementation of a blinking cursor

```
// somewhere in the initialization section of the application code
TIMER_InstallTimeoutHandler(TIMER_ID_CURSOR_BLINK, CursorBlink);
// start the 330 ms timer   blink around 3 times per second
TimerStart(TIMER_ID_CURSOR_BLINK, 33, 0);
// end of initialization

// this is the actual timeout handler function
// parameter is  the next state of the cursor   on or off
void CursorBlink(word handle, dword parameter)
{
     // do the actual drawing / erasing on the LCD
     CursorOnOff((byte)parameter);
     // restart the timer. next time it will expire with the
     // third argument in the opposite phase
     TimerStart(TIMER_ID_CURSOR_BLINK, 33, parameter ? 0 : 1);
}
```

Michael Melkonian is a software developer with MIL/DKS Building Automation in Sydney, Australia. In the past 10 years, Michael was involved in embedded software development for a number of companies in a variety of fields. You can contact him at mindflow@tpgi.com.au.

Special thanks to Jim Chase, for kindly reading and reviewing this article.

Recommended reading

Comer, Douglas and Timothy Fossum. *Operating System Design—The XINU Approach*. New York: Prentice Hall, 1988.

I stopped at a yardsale this afternoon and found a terrific packet-voice processing engine.

I hope you checked to see if it's preconfigured with echo cancellation functionality.



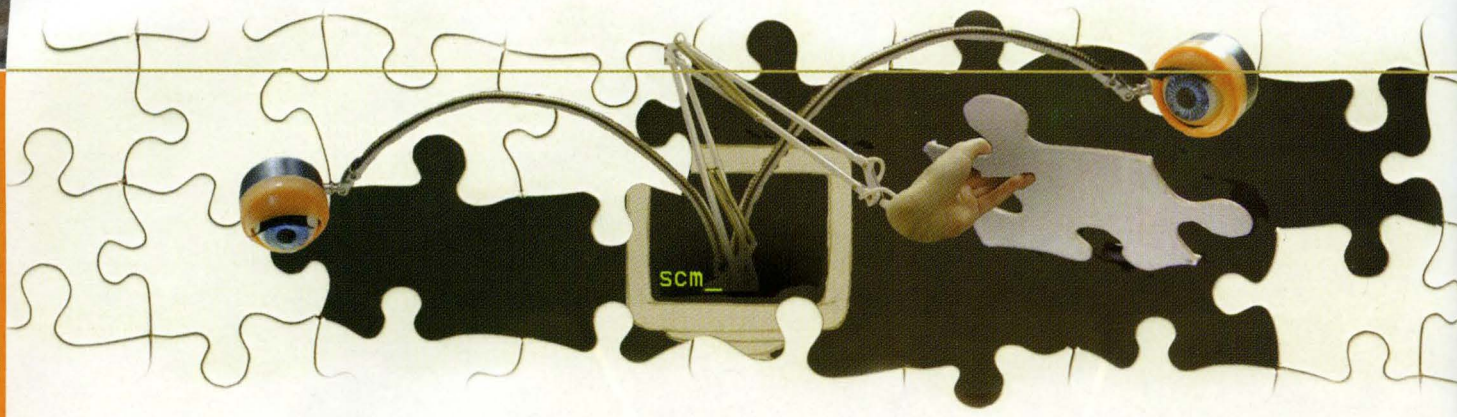
Welcome to embeddedtechnology.com. Your kind of community.

An entire community devoted to embedded computer design? In a word, yes. It's an online community brought to you by VerticalNet, a company dedicated to helping people conduct business as easily as possible. You'll have access to the latest industry news, extensive career opportunities, and a thriving marketplace where industry professionals buy and sell every product imaginable. To visit, simply go to www.embeddedtechnology.com. The resources are there for you. All you have to do is take advantage of them.

embeddedtechnology.com a VerticalNet® community

VerticalNet

ALLAN G. FOLZ



Configuration Management Tips

Whether you're just getting started with version control techniques or an old pro, this article has some good suggestions for you to improve your overall plan.

Many organizations have adopted the use of source configuration management (SCM) tools. While having the right tool for the job is the mantra of professional mechanics everywhere, knowing how to use the tool is equally important. Hammers, screwdrivers, and wrenches are largely self-explanatory; such is not the case with SCM tools. This article will review how SCM delivers higher quality software and aids developers working together; show how it should be used to gain maximum advantage; and how it fits into the bigger CM picture.

SCM and the development effort

When used properly, SCM tools are great aids in debugging software and delivering quality products that inspire confidence. SCM provides a mechanism whereby every product can be characterized as a collection of known inputs. The ramifications of that mechanism are two-fold. First, it provides the concept of a control set when debugging. Second, it provides the ability to know, with 100% confidence, which products are affected by a newly discovered bug.

In science, the fundamental necessity for performing any test or experiment is a control group. Through the use of a control group, scientists are able to isolate a single variable from the set of all variables that could cause an observable change, and then measure the change that said variable actually causes.

As an embedded engineer, I have had countless experiences where a new bug is discovered when trying out some feature that is in the process of minor tweaking. Often, the question asked is, "did something on the hardware break or has my software introduced a new bug?" Pre-development (or prototype) hardware is error-prone and very sensitive. When things go wrong or new bugs are discovered, you can't be sure if the hardware is somehow to blame or the software is at fault. This is particularly

Another useful debugging technique CM provides is the ability to try out and test prior executables for the presence of a newly discovered bug.

true when a unit stops responding altogether. By having the ability to back up and try a previous, "known-good" executable, you can quickly determine whether the hardware or software is at fault.

Another useful debugging technique CM provides is the ability to try out and test prior executables for the presence of a newly discovered bug. If the bug can be characterized as existing in one executable (Y), but not in the executable immediately prior (X), you know the bug resulted from the set of changes that made executable Y different from executable X.

For software already in production, once a bug has been located, the affected executables must be identified, and, depending on the circumstances, customers notified. This problem is somewhat similar to batch manufacturing processes. If a defect is found, all affected products must be located and repaired or replaced. As a result, manufacturers use product ID codes to track all products that are of the same batch. SCM tools make use of version numbers and release labels to track and identify products. This ability may be a non-issue for teams and organizations that develop a single product from a single code base. However, for engineers developing multiple, closely related products, this capability is extremely useful. It allows them to use a single code base as much as possible while still being able to know exactly which source modules were used to create any given product. When a bug is found in a source module, the engineers are able to know which executables are affected.

Another great advantage of SCM is the way in which it aids and encourages developers to work together. Because SCM tools can maintain multiple environments for reading and editing files, they allow developers to work on a set of files without immediate concern for the work of other individuals on the team. While a developer is doing the work of editing and compiling code, only the changes he makes will be reflected in his development environment. This allows every engineer on a team to independently make changes to the same file at the same time. Once each of them has completed making and testing all the modifications that need to be made, they can one by one put their completed work back into the SCM system. The SCM will automatically synchronize all orthogonal changes. Wherever a conflict arises, the SCM will stop the automatic synchronization and prompt the user for manual intervention.

Another manner in which SCM assists developers working together is that it allows each team member to view the other members' work. When one developer is about to make changes to a file, he can immediately see if another developer is working on the same file. Once the other developer has finished making changes, and added them back into the SCM database, the first developer can easily compare the new version to the old version and see the exact nature of the changes.

Maximizing effect

Like any tool, SCM should be used on the problems it was designed to fix and in the manner is was designed to operate. To truly gain the benefits mentioned in the preceding paragraphs, SCM must be used in a planned, organized method. What follows is a compilation of common sense practices to using SCM that are often obvious only in retrospect.

What should be under version control?

- Source files
- Makefiles
- Scripts
- Associated documentation

The first question any SCM user has to answer is: "what should be version controlled?" Strictly speaking, anything that is an input to the process of creating your product should be version controlled. However, I would add the following caveat: tools from outside the organization that are not expected to change over the course of the project should not be version controlled. For example, your compiler, linker, and debugger are static over the course of the average project. Therefore, I would not put them under SCM. I do suggest that they be archived in some fashion, but they will not need the amount of rigor that an SCM tool provides.

There is a gray area for tools that are developed inside the organization, but by other teams or groups. You are the one best able to evaluate the other groups in your organization. What level of trust do you place in their products and procedures? The easiest solution is when the other group is using SCM (they should be) and you can get to their code. In this case it should be adequate to merely add their source files to your workspace in some fashion such that there is no doubt as to which version of their product was used to assist in creating your product.

Intermediate products should not be placed in SCM. The most obvious intermediate products are object files. Though they are inputs to the linker, they are products of the source files and compiler. The litmus test for when something is an intermediate product is whether it can be

HANNAH BLAIR

The first rule of using an SCM tool is to do all development work within the provided workspace.

recreated from one day to the next, given identical inputs and tools. In more direct terms, if one has to check out a file to be able to reproduce a build, something is being incorrectly version controlled.

Some items that may not be obvious, but nevertheless should be placed under SCM are the scripts used to invoke the tools. Likewise any command-line options passed to the tools should be placed in directive files passed to the tool, which can then be placed in the SCM database. This leads to the first rule of maintaining a build process:

Source + Tools = Product

The build process should be cast in stone with no room for deviation. Any and all command-line options should be in some form of file passed in with the command-line. The steps of calling each tool and script should be in batch files as much as possible. Ideally, one command typed at the command-line will execute every step necessary for creating the executable. Anything that cannot be automated in some kind of script should be fully documented in the CM plan (discussed later). The test of a good build process is: "can an engineer not on the development team recreate the product your team is developing?"

As a matter of completeness, any project-related documentation should also be placed under version control. Just as reviewing the evolution of source code over the course of a project is useful, reviewing the evolution of such things as requirements, designs, policies, and practices can also be useful. By placing these documents in the SCM database, the ability to review where the project has been from a managerial perspective is greatly facilitated.

Working in the SCM workspace

- Always work within the workspace
- Only use one workspace per unit of work
- Keep the workspaces synchronized with the codeline

The first rule of using an SCM tool is to do all development work within the provided workspace. Many engineers new to SCM prefer to stick to their old habits of ad hoc SCM and develop their files in their own subdirectories. Once their work is complete, they check out the files they changed, copy their personal files to the workspace, and check everything back in. This approach is flawed in many ways. Part of the power of SCM is the history of a file it records as the file incrementally changes through the course of development. I usually suggest that anything that compiles is ready to be checked in. Anyone inclined to develop outside the SCM workspace is also going to be very reluctant to check in anything until it is 100% complete. The next flaw with such an approach is the risk to the project should the guilty developer be unexpectedly unavailable at a critical time. If the developer has his work spread across multiple subdirectories using various home-brew back-up naming schemes, there is little chance another developer will be able to step in and figure out what is the "correct" version with which to proceed.

Next, each developer should use a unique workspace for each unit of work: problem report/change request (PRCR), bug fix, or feature. If a developer is working on two separate PRCRs he should be using two separate workspaces. If two developers are working on the same unit of work, they should again have their own workspaces. Any exchange of files should only be done through the SCM system via check-in. Sharing a workspace either between developers or among PRCRs can introduce confusion as to the state of files. The SCM tracks development by check-ins from the workspace. When multiple development efforts are taking place in a workspace, the ability to make distinctions as to which check-in was for which PRCR is compromised. In effect, the development efforts become coupled.

Lastly, developers should always keep their workspace synchronized with the codeline. I can attest that constantly re-synchronizing with an ever-changing mainline is difficult. It is like trying to hit a moving target, but to not do so is an invitation for error. It is quite a waste to spend half a day debugging some problem that has already been fixed and made part of the latest codeline. Worse still, is to write code that works within the constraints of some other module that has been redesigned or even removed. Furthermore, to forego merging and synchronizing until all the development work is complete doesn't remove the necessity of synchronization; it only puts it off until the end, when the workspace and the codeline are most divergent.

Growing the SCM database

- Always have a mainline codeline
- Branch when appropriate
- Design the codeline before the project starts

Two approaches to codeline development are the *mainline model* and the *promotion model*. The mainline approach has a single branch that evolves forever. It is the ultimate destination for most (not necessarily all) changes made in all the other branches. By virtue of that fact, the mainline approach creates a common point from which all new development will sprout. All developers know the mainline is the most up-to-date, and the best place to begin new work.

The Embedded Software Solution That Wins.

SUPERIOR INTEGRATED DEVELOPMENT ENVIRONMENT

Internet and Multimedia Applications 1
High Performance Java™ VM, Internet Browsers, E-mail, MPEG, GUI, MP3, Third-Party Technology

Integrated Modular Connectivity 2
IEEE 1394, 802.11, USB, Fast Ethernet, Bluetooth, IXP 1200, ATM, WAP

Reliable, Secure, Proven OS–9® RTOS 3
Protected, process-based with threads support
Outstanding, integrated development tools

Expert Consulting and Support Services 4
Leading design, development and implementation
Total commitment to customer satisfaction

If you're in a race to market with a new product and you want to finish first, make sure you get the embedded software solution that goes the distance.



Get to market first™

MICROWARE®



To order a FREE evaluation CD go online to www.microware.com/first
Or call: 888–642–7609 (U.S. and Canada) • 515–223–8000 (International)

© Copyright 2000 Microware Systems Corporation. All Rights Reserved. Java and all Java based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc.

JAVA

Branching should be used only when incompatible policy exists for user's needs.

Conversely, the promotion model abandons every branch once it is no longer needed. This creates a moving point for beginning new development—something that may be difficult to communicate. It also creates extra difficulty for developers to stay synchronized with the codeline. If the branch they were working from becomes obsolete, they must relocate all their work to some other branch.

I also suggest that the mainline always have a fully working and tested build as its latest release. This gives all developers a known-good baseline to start whatever development effort they need. This is something that is impossible with the promotion model as the branch in which to start new development is always changing.

Branching is, perhaps, the most controversial topic in SCM because everyone seems to do it differently. SCM tools support branching in a myriad of different ways, and organizations use branching in still a myriad more. My advice for branching is to branch only when absolutely necessary. Maintaining branches is a lot of work. Propagating changes from a branch to the mainline and back again is a never-ending chore. It is most prudent to keep all this effort to a minimum by just keeping branching to a minimum.

Branching should be used only when incompatible policy exists for user's needs. For example, the product release group wants a check-in policy that is very controlled and rigorous. The marketing research group wants a check-in policy that allows frequent, easy check-ins of nearly anything that compiles. While these groups may be working with the same source modules, their needs are radically different. Another example is that of separate development teams using a common code base and creating common products, but for different markets. They may be working on the same product, but a Web-based Vehicle Registration System for the Commonwealth of Kentucky is going to have some radically different code than that for the State of Indiana.

Don't copy when you should be branching. I have seen some engineers think they can avoid the effort and hassle of branching by just copying. Copying has all the maintenance requirements of branching without the automatic merging support pro-

PromICE
The Leader in Memory Emulation
•Ultra fast downloads.
•Support for any memory.
•Works with many debuggers.
•Trace & code coverage.
Grammar Engine Inc.
Call Toll Free:
1-800-776-6423
www.promice.com

32/64-Bit CPU Development Boards
• ARM
• MIPS
• Power PC
• M-Core
• PCI
Cogent Modular Architecture (CMA) - a series of flexible development platforms offering advanced on-board I/O, PCI expansion and interchangeable CPU Modules for a variety of 32/64-Bit architectures.
CMA
COGENT MODULAR ARCHITECTURE
COGENT
Cogent Computer Systems, Inc.
Tel: 401-295-6505 • www.cogcomp.com

THREADX

Being The Smallest Is not Enough.

FREE *Evacuation Kit*™ for pSOS users

Contact Us Today

ARM

MIPS

Hitachi

PowerPC

ColdFire

Processors Supported

- Win32 NEW!
- ARC
- ARM
- StrongARM
- Thumb
- PowerPC
- ColdFire/68K
- x86
- Hitachi SH
- MIPS
- NEC V8xx
- TinyJ
- SPARC
- M-Core
- TriCore
- SHARC
- TMS320C6x
- TMS320C54x

Imagine an RTOS that functionally scales to your needs, giving you a footprint as small as 3 Kbytes. Now imagine as well as having the smallest kernel, this RTOS has a host of other features to make you feel comfortable...no run-time royalties, better real-time responsiveness, lower run-time costs.

Open your eyes to ThreadX, the leading royalty-free, source-code RTOS. ThreadX offers you all the outstanding qualities you need to develop the best embedded products on the market today. Its unique picokernel architecture ensures the size of the RTOS is directly proportional to what you use.

ThreadX also includes must-have features, such as preemption-threshold technology, shown by independent research to give you the best of pre-emptive and non-preemptive scheduling models. With processor support for most processor families, clear and concise documentation, and such extensive ThreadX-aware debugger integration, you will be up and running in no time!

Call **888.THREADX** or visit **www.threadx.com** and find out more about ThreadX, the RTOS that makes you comfortable.



the comfortable RTOS

eL

Express Logic, Inc.

tf **888.THREADX** • *fx* **858.613.6646** • **www.expresslogic.com**

FIGURE 1 Sample release note

```
Release V1.4D_02
Filename        Previous Version    Current Version    Feature ID
Added:
   none
Changed:
   analog.h          19                  21          Phone
   audcntrl.c        17                  20          Loudness AUTables
   audcntrl.h        17                  19          AUTables
   audcurve.c        20                  25          AUMenu VolLmt Phone
   audcurve.h        22                  29          Loudness AUTables
   display.h         34                  39          AUMenu AUTables
   executive.h       28                  30          VolLmt
   hardwr_def.h      25                  28          RmtKB Phone
   keybrd.h          24                  26          RmtKB
   rds_tuner.c       32                  35          <mainline>ATCPS
   rds_tuner.h       32                  34          ATCPS
   relay_cntrl.c     20                  22          Phone
   system.h          37                  40          Phone AUTables
   theft_detrnt.c    8                   10          Phone
   uart_decod.h      19                  21          Phone
Removed:
   lword_math.c      4                   -           AUTables
   lword_math.h      4                   -           AUTables

This release has:
 <mainline> : Fixed an improper call to EE_CopyBlock that slipped in the last
release
   RmtKB     : Changed the key scanning logic to be active low
   Loudness  : Toggle loudness on button press, instead after 2s hold
               set/clear LDN enable bit based upon user setting
   AUMenu    : VOL is last in the audio mode cycle
               first press of the audio button always brings up BASS menu
               holding the audio mode button 2s no longer resets all the audio
               settings
   VolLmt    : Added volume limiting at power-on and also an absolute max in EE
   ATCPS     : auto-preset tunes to PS1 when complete, and restores to PS0 when
               canceled
               fixed preset overwriting when radio was powered-on
               added VOICE & MUSIC predefined settings for AM band
   Phone     : added cell phone mute functionality
               fixed theft problem that wouldn't allow certain display modes to
               time-out
   AUTables  : remapped the audio tables per audio engineer's spec's
               add a Bass limiting table as a function of percent volume
               made the Bass boost/cut DSP filter constant an EE parameter

Note: files were merged in the order as they appear in the Feature ID list.
=============================================================================

Allan Folz
23 Oct 98
```

vided by the SCM tool. If a bug is found and fixed for the Indiana project, and it applies equally to Kentucky and 28 other states, the effort of propagating the bug fix is very high when it must be done manually as compared to an automatic merge facility provided by most SCM tools.

One practice that requires a lot of branching, but is well worth the effort is that of the Branched Development Model (BDM). When using BDM, a branch is created for every development effort. All developers have complete freedom for check-in and check-out in their development branches. Once their work is complete, they notify the build boss (discussed shortly) that their work is ready for merge. The build boss then merges all the development branches en masse back to the mainline for release. Often this merge effort takes place in still another branch before it merges into the mainline. This practice is very beneficial when multiple developers are making many small changes to the same sets of files. BDM also helps in supporting the practice of always having a mainline with a known-good release as the latest version.

Highbrow SCM

- Designate a "build boss"
- Make use of change packages
- Releases and release notes

The *build boss* is the person responsible for all the operational SCM decisions. He "owns" the mainline and is responsible for any check-ins that occur to it. He ensures that the build and release process is working and all products can be reproduced. Likewise he is responsible for creating releases. Should a release fail, it is his responsibility to track down the offending files, contact the responsible developer(s), and work out a solution. The build boss enforces the SCM policy, and when developers request to circumvent stated policy,



There is only one conclusion.

Only one RTOS provides all the benefits you need. Accelerated Technology combines a level of service that is unmatched, an affordable pricing model and open source benefits with a vast, tightly integrated embedded product line offering.

Nucleus MNT - Windows-based rapid prototyping environment
Nucleus EDE - intuitive embedded development environment based on Microsoft Developer Studio™
Nucleus PLUS - robust, scalable, multitasking real-time kernel
Nucleus NET - complete TCP/IP networking protocol stack
Nucleus UDB – portable source level debugger
SurroundView and Nucleus ProView - revolutionary profiling tools that introduce a new level of application and OS monitoring
Nucleus WebServ - a tightly integrated, internet-enabling embedded web server
Nucleus WebBrowse - a compact and tightly integrated embedded web browser
Nucleus GRAFIX - portable embedded graphical user interface

These Nucleus embedded products support a wide range of processors:

- MCF5206, 5307 • M•CORE
- PPC40x, 5xx, 60x, 7xx, 8xx, 82xx
- 680x0, 683xx
- ARM6/7/9, AEB, Atmel 40400, CL7110, 7111, 7209, Samsung SNDS100
- SA100-285, SA1100, SA1110 • ARC
- SH1, SH2, SH3, SH4, SH3-DSP, H8S/2000, H8/300H
- IDT RC3081, RC4640/50, R5000, RC32364
- LSI LR 33000, 40xx, 410x, 64008, Lexra 4180, NEC 41xx, 4300, 5000, NKK 4650, Toshiba TX3904, 3927
- x86 RM/PM • TriCore • C167

For additional information on the complete Nucleus product line and how it can greatly diminish your time-to-market needs, royalty-free, please contact us at:

Phone: 1.800.468.6853
Address: 720 Oak Circle Dr. E. Mobile, AL 36609
Email: info@atinucleus.com
Internet: www.atinucleus.com

NUCLEUS

All You Need in an RTOS. Royalty Free.

All in all, the build boss is responsible for the smooth operation of the SCM system. His value cannot be understated and every development team should have one.

he decides if it is warranted. All in all, the build boss is responsible for the smooth operation of the SCM system. His value cannot be understated and every development team should have one.



Our inspiration.

Powerful tools. Great productivity. Integrated price.

Paradigm introduces all the tools you'll need for x86 integration in one package.

Paradigm C++ is alone in offering a complete integrated development environment that includes all the tools you need to get your x86 embedded application jump started. Editing, project management, debugging, compiler, assembler, version control and more, all fully integrated into the powerful Paradigm C++ development environment.

If you are tired of wasting time on non-integrated tools, then Paradigm C++ is where you want to be. Download a copy of Paradigm C++ from **http://www.devtools.com/pcpp** and see the future of x86 development tools today.

Paradigm

Paradigm Systems
3301 Country Club Road
Suite 2214
Endwell, NY 13760

1-800-537-5043

Phone **607-748-5966**
Fax **607-748-5968**
info@devtools.com
http://www.devtools.com



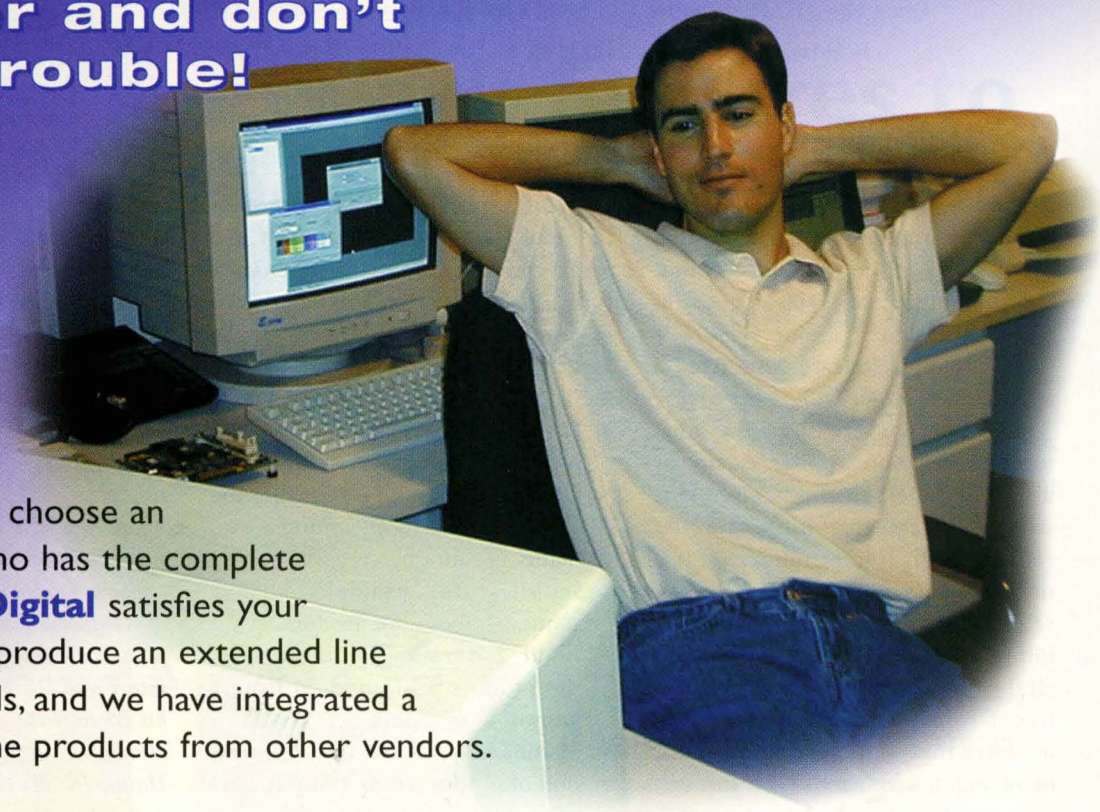
A *change package* is a logical grouping of all the files that were modified through the course of some unit of development work (for example, PRCR). Rarely does any but the most trivial bug fix require modification of only one file. Rather, the bug fix spans multiple files that are inter-dependent to get a working executable. Trying to build with three of the four files that were changed will likely not even compile. Since using the modified versions of the affected files is an all-or-nothing proposition, it makes sense to track them as a single unit of change. This creates the capability to undo the entire change should a problem or question concerning the PRCR ever arise. This concept is directly supported by such SCM tools as Continuus/CM, ClearGuide, and others. Of course it can be manually supported by any SCM user through such mechanisms as release notes (discussed next).

A *release* is a formalized build that creates a milestone product. The product itself can be for internal consumption only, or for delivery to a customer. Typically a release involves applying a label to all versioned files. The label provides a means for getting a specific version of a file by name rather than number. This is a convenience for developers by logically grouping specific versions of all the files that create a build. The manual alternative would be to keep a list of the version number for every file needed to build a given executable.

The *release notes* are a simple ledger created after a version label is applied to the source code. It lists all the files that were modified, the version used in the previous release, the version used in the current release, the order in which branches were merged into the mainline, and a brief description of the reason for the change, or a reference to the PRCR that the work was performed against.

Most Programmers Want ...

Integrated tools that work together and don't cause trouble!



It makes sense to choose an **RTOS** vendor who has the complete solution. **Micro Digital** satisfies your requirement. We produce an extended line of embedded tools, and we have integrated a broad range of fine products from other vendors.

KERNEL - Low latency, hard-real-time, preemptive, multitasking, powerful API.

FILE - High-performance (up to 14 MB/sec) FAT 12/16/32 file system with all the drivers you are likely to need. Contiguous files supported.

NETWORK - Fully compliant TCP/IP stack with over 20 Ethernet drivers and 10 application-level protocols, including a capable web server and an HTML 4.0 browser with a very small footprint (200 KB). We also offer a USB host stack with 11 drivers.

USER INTERFACE - Text windows; graphics library; GUI with window builder, font capture, and image capture utilities, Unicode, and over a dozen drivers; 3D animation library with high-speed drivers for all common graphics boards.

TOOLS - Full support for popular compilers, linkers, locators, and debuggers with kernel awareness.

SPECIALTY - Dynamic load module support, x86 protected mode environment, NT console application loader, DOS emulator, C++ support package, boot loader and more.

SUPPORT - Excellent support is as close as your phone or the Internet and is provided by the programmers who wrote the code. No quibbling policy: we either explain it or we fix it. We also offer training and contract help.

TERMS - No royalties, source code included, 30-day free trial.

FREE EVALUATION KIT - Call today for details, **800-366-2491** or visit our website, **www.smxinfo.com/espa.htm**

µd

Micro Digital Inc

Embedded Software Outfitters

25 years of excellence

smx®

714-437-7333 **www.smxinfo.com/espa.htm** **800-366-2491**

Once the effort of using the SCM tool has been reigned in, it is time to take a careful look at the high-level concepts and how your organization is supporting them.

The release notes do not contain any unique information; rather, they summarize in one convenient location everything that makes this release different from the last one. Figure 1 shows a sample Release Note from a hypothetical project.

The first line identifies the label to which this note applies. Again, by using the label, all the components that make up this build can be retrieved with 100% confidence. It is not necessary to keep track of the relevant version number of every individual file. The next section lists all the files that make this release different from the previous one. Any modules that were added, removed, or changed are listed with: the version number they had in the previous release, the version number for the current release, and a list of all branches from which the changes were merged (this project uses the BDM). The third section is a high-level description of what was fixed or otherwise changed and why. Notice the Feature ID is an implementation of the Change Package concept. If something proves wrong with the work done for the Phone effort, it is possible to undo the change for each of the seven files that were affected by Phone. A new release would then be performed to ensure a working mainline is always preserved. The last two lines identify a point of contact and a date of creation. The point of contact is usually the build boss. He is the one most intimately familiar with the release since he is the one actually going through the process of building and releasing the software.

The big picture

This article has largely been about techniques for using your SCM tool efficiently and effectively. However, the tool itself is less than half the whole configuration management picture. Configuration management encompasses the following seven aspects:

- Identification: identifying the input components
- Control: the policy for allowing changes to the input components
- Status accounting: recording vital statistics (for example, change requests) about the product
- Audit and review: verification of the product being derived from a known set of inputs
- Manufacture/build: the construction of product from its input components
- Process management: ensuring the CM policy is being designed and followed
- Team work: managing the interactions among multiple developers

Once the effort of using the SCM tool has been reigned in, it is time to take a careful look at the high-level concepts and how your organization is supporting them. Different SCM tools support different concepts at varying levels. For example, RCS is heavily centered on the identification and control aspects, but has minimal built-in audit and review capabilities. Chances are, the concepts that the tool you are using most strongly supports are the ones your organization most strongly supports. However, effort should be made to supplement any configuration management aspects not inherently part of your tool's paradigm.

The best place to start the rest of your configuration management effort is the CM plan. The CM plan gathers in one location all the policy decisions for implementing configuration management on your project. Once the policy decisions have been made, a foundation exists to create the procedures which will be used over the course of the development effort. A full discussion of a CM Plan is outside the scope of this article. However, I suggest reading "Configuration Management (CM) Plans: The Beginning to Your CM Solution" (by Nadine Bounds and Susan Dart) as an excellent place to start further research. **esp**

Allan Folz is a staff engineer at Logikos Inc. a software out-source and consulting company located in Indiana. He has developed software for a variety of applications in the automotive industry in both product-intent and R&D settings. Allan received his BS in electrical engineering from Rose-Hulman Institute of Technology in Terre Haute, IN. He encourages correspondence and can be reached via e-mail at afolz@logikos.com.

References

Bolinger, Don and Tan Bronson, "Applying RCS and SCCS," 1995. Available at *www.oreilly.com/catalog/rcs/noframes.html*.

Bounds, Nadine and Susan Dart, "Configuration Management (CM) Plans: The Beginning to Your CM Solution," 1993. Available at *www.sei.cmu.edu/legacy/scm/abstracts/abscm_plans.html*.

Dart, Susan. "Concepts in Configuration Management Systems." Available at *www.sei.cmu.edu/legacy/scm/abstracts/abscm_concepts.html*.

Weber, Darcy Wiborg, "Change Sets Versus Change Packages: Comparing Implementations of Change-Based SCM," 1997. Available at *www.continuus.com/developers/developersACEG.html*.

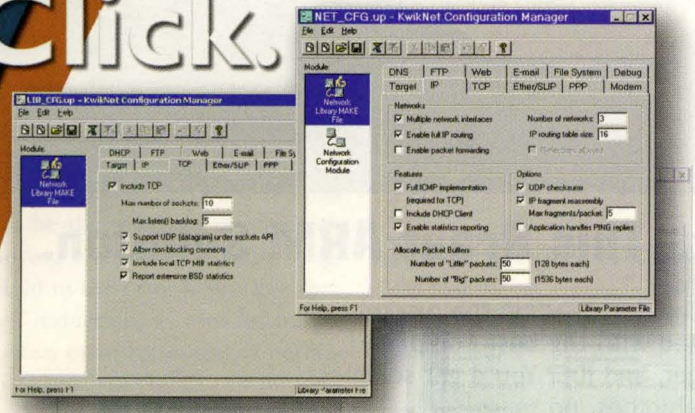
Wingerd, Laura and Christoher Seiwald, "High-level Best Practices in Software Configuration Management," 1998. Available at *www.perforce.com/perforce/bestpractices.html*.

KwikNet™

Adding network features to your embedded products?

KwikNet makes it quicker, easier, and less expensive.

Just Point and Click.



You will also appreciate ***KwikNet's****:*

- portability to any RTOS with the minimum required features
- modular ANSI C code which has been tested on a wide range of target processors with 12 popular compilers

And if you add ***KwikNet's*** *Web Server option, you'll enjoy even more benefits.*

After more than 20 years of leadership in providing real-time multitasking kernels for mission-critical embedded applications, **KADAK** brings the same reliability and ease of use to TCP/IP stacks and web servers. With **KwikNet**.

When you use **KwikNet**, there is no need to spend months of time and incur the expense to manually create and test your TCP/IP stack.

We've already tested it.

And **KwikNet's** Configuration Builder lets you speed through the configuration complexities inherent in any TCP/IP stack!

Plus, as with any product from **KADAK**, you are given a royalty-free site license including source code.

Find out all that it can do for you! Visit us at www.kadak.com

KwikNet TCP/IP Stack and Web Server

KADAK RTOS and Network Products

KADAK Products Ltd.
206-1847 W. Broadway
Vancouver, BC, Canada V6J 1Y5
Tel: (604) 734-2796 Fax: (604) 734-8114
Email: amxsales@kadak.com

KwikNet and AMX are trademarks of KADAK Products Ltd.

We Put Our Heads Together



And Came Up With This Great ARM Solution.

ThreadX

- No Royalties
- Fastest RTOS for ARM
- Smallest memory requirements
- Complete ThreadX source code
- Preemption-threshold™
- Picokernel™ architecture
- Optimized ARM interrupt handling
- IRQ and FIQ interrupt support

MULTI 2000

- Best IDE for ARM Development
- Best ARM Debugger
- Best ARM Compiler
- Seamless ARM/Thumb interworking
- CodeBalance™ tuning
- ThreadX-aware Debugging
- Run-time Event Analyzer for ThreadX
- ThreadX C Library Integration
- Complete ARM simulation

What if the best RTOS in the market combined with the best Compiler, Debugger, and IDE? You'd get seamless integration, no hitches, no snags, and no surprises.

www.ghs.com/armsolutions

Well now it's happened. Express Logic's ThreadX® kernel and Green Hills Software's MULTI® 2000 IDE have come together to create the industry's most powerful development and run-time solution for deeply embedded applications.

With ThreadX you get a royalty-free, source-code RTOS, and state-of-the-art features like preemption-threshold and picokernel design. The advanced MULTI 2000 IDE gives you an Optimizing C/C++ compiler, ThreadX-aware debugging, complete ARM simulation, and the run-time EventAnalyzer for ThreadX. Together, ThreadX and MULTI 2000 are easily the best ARM solution available today.

For a free demo CD of the MULTI 2000/ ThreadX solution, call Green Hills Software at 805.965.6044 or visit us on the web at www.ghs.com/armsolutions.

Best support for the ARM Processor

ARM6
ARM7
ARM7M
ARM7TDMI
ARM7500fe
ARM8
ARM9
ARM9E
StrongARM

Other Processors Supported:

- Win32 • PowerPC • X86
- ColdFire/68K • Hitachi SH
- MIPS • NEC V8xx • SPARC
- M-Core • TriCore

Green Hills SOFTWARE, INC.

From the brains of

Express Logic, Inc. and Green Hills Software, Inc.

ph. 805.965.6044 • sales@ghs.com • www.ghs.com/armsolutions



Don Morgan

Field-Oriented Control

Last month, we looked at how a three-phase motor works. Basically, it's a system in which an element (rotary or linear) can be moved by manipulating an associated magnetic field. Not only can we move it, we can move it as we wish.

The magnetic field that we control is in the stator of the motor. This is the shell of a rotary motor that surrounds the rotor (usually the element that moves). In a linear motor it is most likely to be the platform upon which the moving element rests. The magnetic field in the rotor is set up either through induction, as with an induction motor, or with the use of permanent magnets. For the rest of this column, we will refer only to a rotary device but the same information may be used to control a linear motor.

In a three-phase motor, the currents are controlled in each of the phase windings in such a way as to establish a magnetic field in the rotor—if this is not a permanent magnet motor—and cause the rotor to align with the field flux. Then, by properly controlling the currents in the stator field, we produce a vector that leads the shared magnetic field of the stator and rotor, which causes the rotor, and ultimately the shaft, to move. In a three-phase motor, the stator is an electromagnet made with a winding for each phase on a soft iron casting. In each winding, current may flow in a forward (positive) or reverse (negative) direction; this results in six unique steps or pole alignments. We control the amount of current that flows by either pulse width modulation (PWM) or analog means. The resolution of control actually depends on the resolution of the positioning feedback device, the current feedback, and the update rate.

The technique I have just described is called *field-oriented control.*

PWM versus analog

A PWM-driven system controls the current in the windings of a motor by switching the voltage across each winding off and on at a fixed frequency, but with a duty cycle that varies according to the amount of current the controller commands. This is not the only type of amplifier available.

Three-phase motors require proper care and feeding. Field-oriented control makes this easier.

Another popular way of accomplishing the same end is to use an analog amplifier. Unfortunately, analog amplifiers are usually expensive and large compared to PWM amplifiers. In an analog amplifier, the transistors operate in their active region. They are neither fully on nor fully off. Instead, they dissipate power based upon the instantaneous voltage drop across them and the current they are supplying to the motor. These currents and voltages can be quite high.

PWM drives have an advantage in that the transistors only ever assume two states: fully on or fully off. When they are off, current flow is negligible and so, therefore, is heat. When they are on, the voltage drop across each device is only on the order of a couple of volts, so the power they dissipate is at a minimum. For this reason, PWM drives can be small, and therefore cheaper than analog.

Either way, the field-oriented control we describe is the same. It is only the last stage, in which the output values are either sent to an analog amplifier or converted to duty cycle varying pulses, that changes. Envision the output as a digital-to-analog converter (DAC), and you will have an idea of what is going on. It is only what occurs after the output at the DAC that changes.

Stator frame

In a static frame—that is a frame in which we hold everything still for a moment—we see that two currents are involved in the motion of the rotor. One current, i_d, is associated solely with the stator flux. This is the current that induces a magnetic field in the rotor of an induction motor and, held constant, causes the rotor to align with it. With that current alone, we would have a device called a "stepper," as its motion can be controlled by indexing the stator flux in a circular fashion. I will describe a technique shortly that does just that. This produces very little

torque, however. The only torque it does produce results from the motion of the flux to the next indexed step.

The second current is 90 degrees out of phase with the first and is called the i_q, or quadrature, current. This current produces a flux that either leads or trails the stator flux. If it trails the stator motion, the motor is a generator. If it leads, we have torque, and thus, a motor. The amount of i_q current controls the amount of torque we have.

For more torque, you need more i_q. For more rotor flux, you need more i_d. Knowing these two values, you can solve for the vector that represents the amount of torque in a motor. By feeding i_d, i_q, and your command values to your PID controller, you can generate result values that may be used to correct the motion of position, velocity, or torque in your device.

But there's a catch...

Unfortunately, the motor is static; it's not in motion. The stator flux is constantly moving when the motor is in motion. In a three-phase motor these values are constantly changing. Even if the motor is simply moving in a single direction at a constant speed with no acceleration or deceleration, the currents in each winding are constantly changing according to the angular position of the rotor in each electrical revolution. This does not mean that the i_q and i_d change. (You will see in the following mathematical expression that they will be constant.) It means that in order to keep the flux moving at a constant rate, the currents in each winding must vary continuously.

At this point, let's go back and look at those three-phase currents. Remember, these transforms go both ways. We must use the forward transform to convert the currents and position for input to the PID controller and as reverse transforms to convert them back to individual phase values for output to the amplifier.

The Clarke transform

We have two sources of information about the motor that we can use in our PID controller. One consists of the position of the rotor returned from the encoder or a similar device and the other is the current in the windings. Both are important. In order to produce values we can use to control or correct the motors operation, we need to know the current status. We need the currents and position.

Because trying to control the motor using the individual phase currents is a complicated affair, we use the Clarke transform to change the reference of the three-phase currents, $i_a(t)$, $i_b(t)$, and $i_c(t)$ to currents in the two-phase orthogonal stator axis: i_α and i_β. This conversion is illustrated in Figure 1. Now, referring to the star-connected motor in Figure 2, we have:

$$i_a(t)+i_b(t)+i_c(t)=0$$

$$V_a(t)+V_b(t)+V_c(t)=0$$

$$\varphi_a(t)+\varphi_b(t)+\varphi_c(t)=0$$

which denote currents, voltages, and flux linkages. Please note that the final relationship maintains the balance of currents, voltages, or flux linkages as explained by Kirchoff's Law, that is, their sum will be zero. Anytime there is a current, voltage, or flux in one phase there must be corresponding currents, voltages or fluxes in the other two to balance it. Both the forward and reverse Park and Clarke transforms may be applied to currents, voltages, or linkages in exactly the





For developers who practice

safe embedded relationships.

TASKING

The Embedded Communications Company™

To reduce development costs and speed time to market, you need a supportive, reliable embedded software partner with the right tools for the job. That partner is TASKING.

With TASKING, you'll find development tools for leading DSPs, including DSP56xxx and CARMEL. You'll have the best tools for leading 8-bit, 16-bit and 32-bit microprocessors and microcontrollers, including PowerPC, 68xxx, TriCore, C166, ST10, M16C, XA, 8051, 196/296 and more. And you'll save even more time and money with integrated RTOS and embedded Internet solutions, as well as expertise from our Embedded Applications Services group.

With our installed base of over 100,000 licensed users, world-class support and worldwide presence, you'll know you made the safe choice for your company.

US: +1-781-320-9400
Germany: +49-7152-97991-0
Japan: +81-3-3457-6831
Netherlands: +31-33-4558584
Italy: +39-02-6698-2207
UK: +44-1280-706686

www.tasking.com/mfbuyers

same way. In the application described here, we use the currents we sense in the phase windings to compute new voltages for the PWM.

With this in mind, the following relationship exists between a homopolar and three-phase system:

$$y = \frac{2\pi}{3} \tag{1}$$

$$\begin{bmatrix} i_\alpha(t) \\ i_\beta(t) \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & \cos(\gamma) & \cos(2\gamma) \\ 0 & \sin(\gamma) & \sin(2\gamma) \end{bmatrix}\begin{bmatrix} i_a(t) \\ i_b(t) \\ i_c(t) \end{bmatrix} \tag{2}$$

which may also be written as:

$$i_{sa} = i_a \tag{3}$$

$$i_{s\beta} = \frac{1}{\sqrt{3}} i_\alpha + \frac{2}{\sqrt{3}} i_b \tag{4}$$

with the inverse as:

$$\begin{bmatrix} i_a(t) \\ i_b(t) \\ i_c(t) \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & 0 \\ \cos(\gamma) & \sin(\gamma) \\ \cos(2\gamma) & \sin(2\gamma) \end{bmatrix}\begin{bmatrix} i_\alpha(t) \\ i_\beta(t) \end{bmatrix}$$

$$i_\alpha = \frac{2}{3} i_\alpha - \frac{1}{3} i_a (i_b - i_c)$$

$$i_\beta = \frac{2}{\sqrt{3}} (i_b - i_c)$$

$$i_o = \frac{2}{3} (i_a + i_b + i_c)$$

Though these two versions appear different, they are exactly the same. The real difference lies in what they communicate to the software engineer. The first, Equation 2, suggests a convolution and very likely a tighter program. The second, Equations 3 and 4, looks like an in-line operation. Generally, matrix coding and using the multiply-accumulate (MAC) instruction will produce cleaner, clearer code. The matrix notation makes it easy to see how this and the next step combine into one equation.

The Clarke transformation

We can use the relationship we described involving the i_d and i_q currents both to reflect the state of the currents in the motor and to control the field in the rotor and the torque. In general, these values may change for various conditions such as starting, accelerating, phase advance, and field weakening. But if the motor is simply to maintain a uniform speed and torque, they may not.

In order to move our homopolar currents to a static frame suitable for a PID controller, we must use some sort of feedback device, such as an encoder, to detect the position of the rotor. The corresponding transform is known as the Clarke transform, but most people will know it as a simple coordinate transformation matrix:

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix}\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}$$

The encoder gives us the θ. Using the transform, we can create static (slowly moving) values from the instantaneous (and changing) homopolar values based on the rotor's position in the electrical revolution.

Algorithmically, this is commonly done with a lookup table. Encoders have a constant number of counts per revolution. If we divide a circle by this number, we will have the smallest arc we can measure. We can then make a table based on the sine and cosine of the angle this arc subtends. So each time we wish to update the amplifier that supplies power to the motor—a period usually called the servo update rate—we must get the current encoder position and use it to transform the i_d and i_q values to meaningful i_α and i_β values.

What do the leading silicon vendors know about BIOS?

They know that the right BIOS is key to the success of embedded designs—and that configurability is key to the right BIOS.

That's why **AMD**, **Intel**, and **STMicro** ship General Software's Embedded BIOS pre-installed on their embedded platform evaluation boards.

With over 400 configuration options, Embedded BIOS offers the advanced configurability you need to run your custom target environment without editing the core BIOS source code.

Contact us today for detailed information and a free sample BIOS binary for your standard reference design.

Embedded BIOS™

ADAPTATION KIT:
Full source code automatically configured with over 400 parameters using BIOStart™ expert system

CORE BIOS FEATURES:
ROM/RAM/Flash disks, Setup system, console re-direction, manufacturing mode, WinCE loader, configurable PCI, integrated debugger, modular callouts to chipset, board, and CPU-level modules

CHIPSETS:
ALI—Aladdin V, Finali
AMD—186, SC300, SC400, SC520
INTEL—386EX, 430HX/TX, 440BX, 810, 840
NSC—Geode GXm, GXlv
STMicro—STPC family

IDEAL FOR:
Windows 95/98/CE/NT Embedded, DOS, Linux, and all x86-based operating systems

GENERAL™ SOFTWARE

www.gensw.com • sales@gensw.com • 800-850-5755 • 425-454-5755

© 2000 General Software, Inc. All rights reserved. General Software, the GS Logo, and Embedded BIOS are trademarks of General Software Inc.

IAR Embedded Highway
Get There Faster!



IAR Embedded Workbench™

A superior C cross compiler package offering advanced embedded C/C++ optimization techniques

- Supports the following MCUs:

Atmel	AVR, AT89
Hitachi	SH, H8S, H8/300H, H8/300
Intel	8051, 80x96, 80251
Microchip	PICmicro 16/17 and 18
Mitsubishi	M16C/60/20, MC80, 740
Motorola	6811, 6812, 6816
National	COP8, CR16
NEC	V850, 78K0, 78K4
Sharp	SM6000, SM8500
TI	MSP430
Zilog	Z80, Z180

and more... for complete list, check www.iar.com

- Different Architectures, One Solution: The same user friendly environment applies for all targets.

IAR visualSTATE®

A UML-compliant graphical state machine tool generating micro-tight code for any embedded MCU.

- Very small code footprint (A breakthrough patented technology)
- Table driven code (compact & fast)
- Uniquely suited for low-end 8/16-bit MCUs
- Generates MCU & Compiler independent code (works for 8/16/32/64-bit MCUs)
- 100% formal verification of state model (a second patent)
- Allows state level debugging, breakpoint setting, variable/action watch and graphical animation of model
- Static & dynamic analysis with regression testing
- Automatically generates complete design documentation in 'rtf' format.

IAR MakeApp™

The Device Driver Wizard. Automatically generates fully tested C initialization code for MCUs

- Visual configuration of device drivers.
- Supports the following targets

Atmel	AVR
Hitachi	SH, H8S/300H, H8/300
Mitsubishi	M16C
Toshiba	TX RISC

for up-to-date target support check www.iar.com

- Checks for illegal settings & resource conflicts

IAR App. Development Services

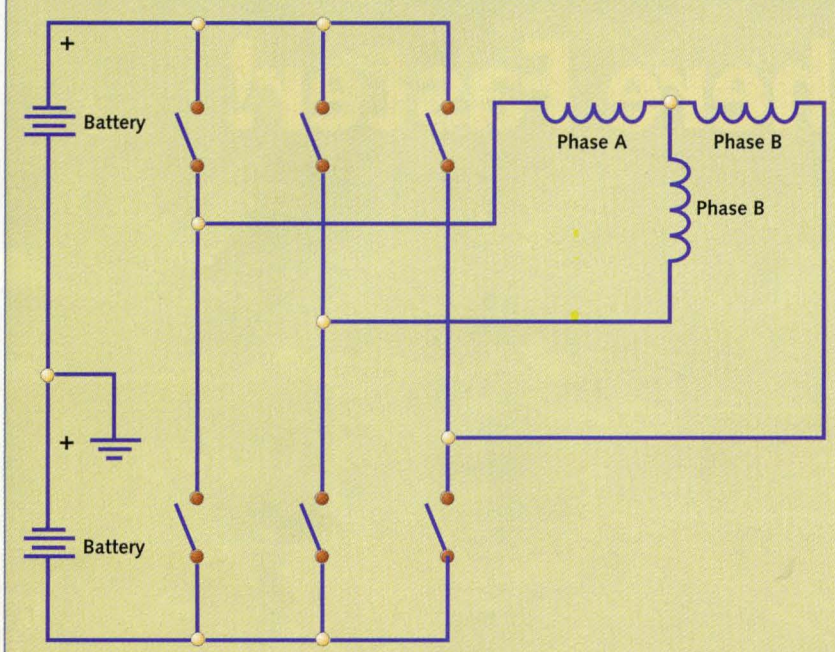
- Contact IAR to speed your next embedded development project.

For a free demo & more information, please contact IAR Systems.
Email: info@iar.com, Tel: 1-800-427-8868 Or visit **www.iar.com**

IAR Systems (US HQ & West Coast), One Maritime Plaza, San Francisco, CA 94111 Tel: 415-765-5500, Fax 415-765-5503
IAR Systems (East Coast) 2 Mount Royal, Marlborough, MA, 01752, Tel: 508-485-2692, Fax 508-485-9126

IAR SYSTEMS
From idea to target.
www.iar.com

FIGURE 1 Three phase stator and power bridge



Of course, we may write this matrix in an algebraic form as well:

$$i_{sd} = i_a * \cos(\theta_e) + i_\beta * \sin(\theta_e)$$

And the reverse transform may be expressed:

$$i_a = i_{sd} * \cos(\theta) - i_{sq} * \sin(\theta)$$

$$i_\beta = i_{sd} * \sin(\theta) + i_q * \cos(\theta)$$

$$\begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} I_d \\ I_q \end{bmatrix}$$

A short aside

Now, we can combine both transforms into one matrix. This fact is a great deal easier to see and understand when we view the transforms as matrices:

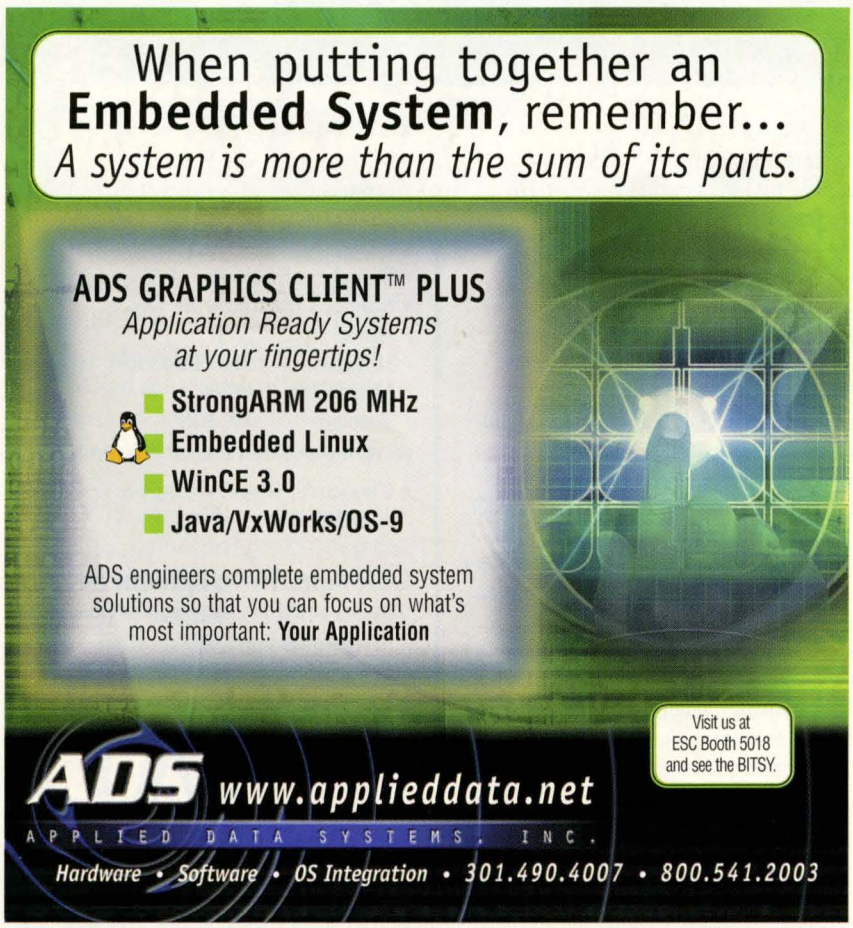
$$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \frac{2}{3} \begin{bmatrix} \cos(\theta) & \cos(\theta - \gamma) & \cos(\theta - 2\gamma) \\ -\sin(\theta) & -\sin(\theta - \gamma) & -\sin(\theta - 2\gamma) \end{bmatrix} \begin{bmatrix} I_a \\ I_b \\ I_c \end{bmatrix}$$

Even though this looks efficient, take care. Depending upon the architecture of your processor, it can end up being more work because of the extra sinusoidal computations of position.

Using the algorithms

Before we can begin to use this form of control, we need to initialize the system. Since we base our control of the current in the phase windings on the position of the rotor—to keep the torque angle 90 degrees in advance of the stator flux—we must find out precisely where the rotor is to start.

We can do this by applying a known and constant voltage vector to the stators. We do this in such a way as to cause the rotor to align with one of the poles. We use this positioning to establish our first position with the encoder and to reset the electrical position. In Table 1, you will see the eight states



Pentium® Debug: $4,995

INSPIRATION

EmProve
Virtual Application Development and Prototyping (VAD-P™) environment

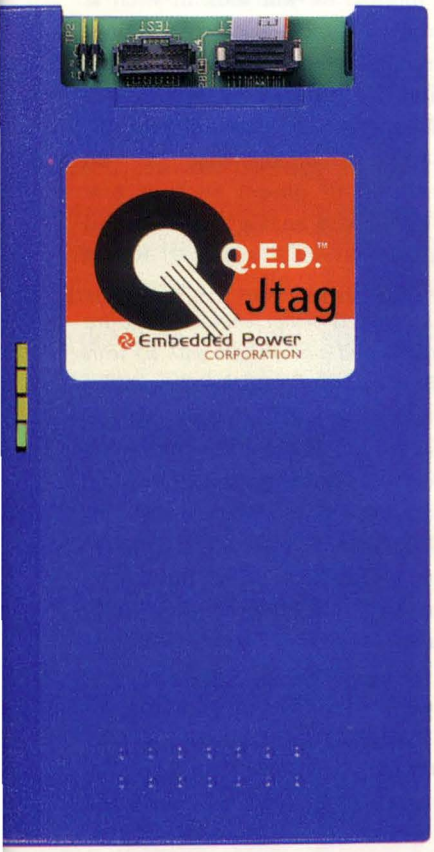
RTXC™
Royalty-free real-time OS
- TCP-IP stack
- File system
- Graphic system

BeaconSuite™
- Compilers
- Linkers & Locators
- Debuggers
- Simulators

FINAL CODE

Q.E.D.™ Jtag

Sometimes the shortest distance between two points covers new ground.



Embedded Power's Q.E.D. tradition continues - with a full JTAG debug port implementation! We bring you accurate, dependable hardware-assisted source-level JTAG debug on the Windows® NT/95/98 platforms.

Using our own VisualProbe™ or other debuggers, Q.E.D. Jtag provides:

- Kernel-aware debug with our RTXC RTOS
- Debug for all Pentium family members
- Full visibility of all internals, including hidden registers
- Support for your favorite toolchains, including CAD-UL, Borland, MetaWare® and Microsoft®
- Fast, efficient host communications. High throughput means fast debug functions and quick code downloads - even to FLASH.
- Requires NO fumbling with PC bus cards! Standard connection to your PC is via an IEEE 1284/EPP parallel port.
- Eliminates the need to CAD a JTAG port to your board. We offer a complete set of JTAG breakout adapters that make hardware connection a snap!

Stay on course through Pentium-class embedded development with Q.E.D. Jtag Debugger!

Pentium Processors	JTAG Port Pinout	Breakout Board
Pentium I	20	PGA 320
Pentium II	30	PGA 370/SLOT 1
Mobile Pentium II	30	BGA/Micro BGA Intel breakout board
Pentium II Xeon	30	PGA 370/SLOT 1
Celeron	30	PGA 370/SLOT 1

www.embeddedpower.com

Embedded Power CORPORATION

U.S. 281-561-9990
email: info@embeddedpower.com
Europe: +44 (0) 1256 474448
email: eurosales@embeddedpower.com
© 2000 Embedded Power Corporation
Pentium® is a trademark of Intel Corporation

FIGURE 2 Phase currents

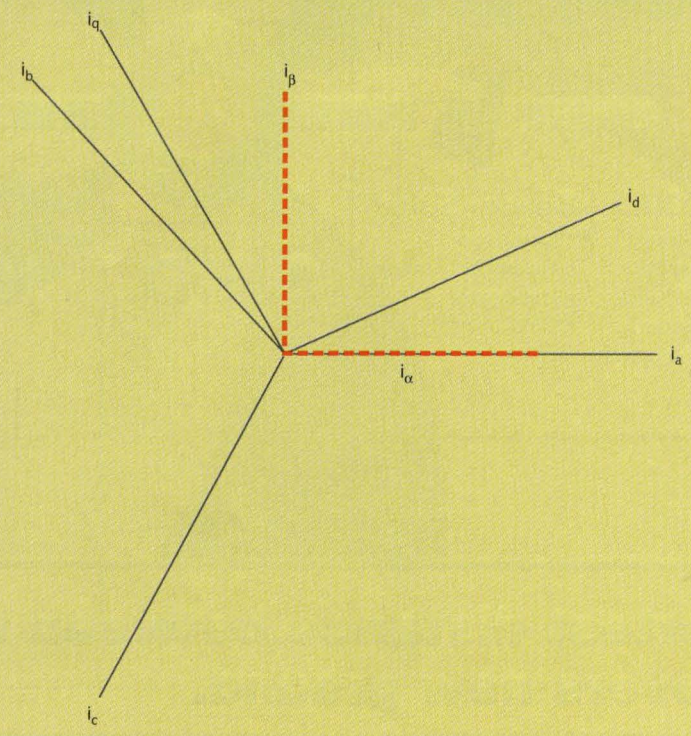


TABLE 1 Transistor firing matrix

A	B	C	VA	VB	VC
0	0	0	-	-	-
0	0	1	-	-	+
0	1	0	-	+	-
0	1	1	-	+	+
1	0	0	+	-	-
1	0	1	+	-	+
1	1	0	+	+	-
1	1	1	+	+	+

possible with three phases. In this table, the minus signs indicate when the bottom switches are all on; the plus signs mean that the top transistors are on.

Because there must be at least one minus sign and at least one plus sign in a single row for current to flow, only six of the states are workable. Therefore, to initialize the motor, we must choose one of the remaining six combinations and hold it constant. Under these conditions, current is flowing in the windings and the rotor will align with the stator flux. Although this will work on an induction motor, only the number of encoder counts per revolution is really required for initialization.

Next, we need to know how many encoder counts we have for each electrical revolution. If we do not know the number of encoder counts per electrical revolution, we may, at this point, deduce the number by "stepping" through the six combinations in the table sampling the encoder as we go.

Field-oriented control

Once we know the number of encoder counts in an electrical revolution and have reset the electrical position and captured the encoder position at that point, we may begin.

At this point the electrical angle may be shifted by 90 degrees—this moves the i_d and i_q axes apart. The rotor tends to align itself with the stator flux and the motor turns.

We must continually refresh the information about position and currents, however, or the motor will halt. These refresh points are known as *servo updates*. At these points, the encoder is read, along with the phase currents in the motor windings. We use the forward Park and Clarke transforms to compute the i_d and i_q from the phase currents in the motor windings and the electrical angle. This information, along with the user control settings, is fed to the PID controller and a new (corrected) i_d and i_q are returned. The change in notation indicates that we are actually controlling the voltages applied to the phases of the motor, as illustrated in Table 1. These values are then fed to the PWM or analog controller

This i_d and i_q are passed through the reverse transforms to give us new values for our phase currents. These values are then fed to the PWM controller or analog controller, as the case may necessitate.

Next month

Next month, we will look at what is involved in using devices other than incremental encoders to approximate position. We will see how the DSP can calculate position given quadrature sinusoids (sinusoids that are 90 degrees out of phase) such as one might have with analog encoders and resolvers. **esp**

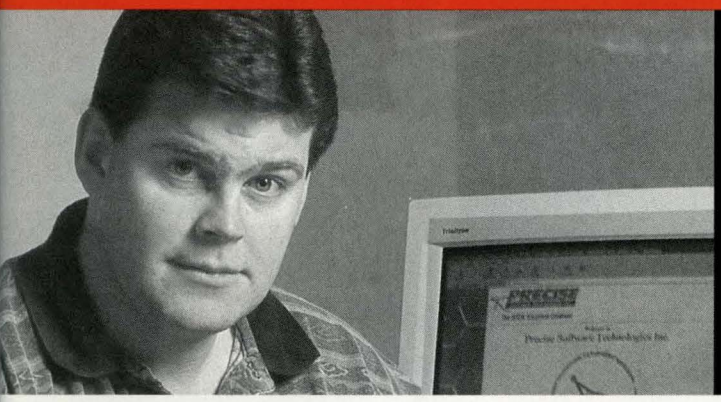
Don Morgan is senior engineer at Ultra Stereo Labs and a consultant with 25 years experience in signal processing, embedded systems, hardware, and software. Morgan has wrotten a book about numerical methods, featuring multi-rate signal processing and wavelets, called Numerical Methods for DSP Systems in C. *He is also the author of* Practical DSP Modeling, Techniques, and Programming in C, *published by John Wiley & Sons, and* Numerical Methods for Embedded Systems *from M&T.*

Resources

www.ti.com/sc/docs/apps/digital/ac_induction_motors.html#App_Notes

www.analog.com/industry/motor_control/appcode/admc401/pwm401_1.html

"When we needed to meet our customer's deadline, we knew *PRECISELY* what to do."



Steve Mills
Director, Software Development
Cadence Design Systems Ltd.

Precise Software Technologies Inc. is a leader in solutions for embedded application development, specializing in real-time embedded software development tools and real-time operating systems. And that's exactly why it's the supplier of choice for Cadence Design Systems.

When a large American company, the world's leading supplier of intelligent enterprise storage and retrieval technology, needed a way to further improve performance monitoring and fault diagnostics for its Integrated Cached Disk Array (ICDA), it turned to Cadence to build a monitoring system. The deadline was tight, and Cadence knew that to deliver on time and on budget, only the Precise embedded products would do.

"We at Cadence knew that Precise was the embedded software of choice," says Steve Mills. "And as a result of that decision, we were able to make the deadline. Our customer was able to improve its product and meet its customers' requirements for a lower-cost, higher-performance ICDA. Precise played a key role in our customer's success."

Cadence is the world's largest supplier of software products, consulting services and design services used to accelerate and manage the design of semiconductors, computer systems, networking and telecommunications equipment, consumer electronics and a variety of other electronics-based products. Precise products are used by system integrators and product developers at companies like Cadence to support the development of real-time software that is embedded in a range of applications, including Internet appliances, networking, telecommunications, avionics, consumer products, medical instrumentation, robotics and process control.

According to Mills, "The Precise embedded software and application tools simplify the development process and help us speed up our customers' time to market. Precise helps us deliver what our customers need...which is *precisely* what we want."

PRECISE™
Software Technologies Inc.

www.psti.com

Phone 613-596-2251
Fax 613-596-6713

Leadership in Embedded Internet Protocols and Real-Time Operating Systems

This little PDA went to market.



This one stayed home.



Why do some platforms and devices succeed where others fail? **Applications.** Palm™ used CodeWarrior to create the Palm OS®. Now more than **50,000 developers use CodeWarrior** to build a full range of applications for business and personal uses. Result? The Palm OS is the standard for handheld computing, and devices built on the platform are market leaders.

Want to make your platform a success? **Build it with CodeWarrior.** Get leading-edge tools and a world-wide community of more than 200,000 developers supporting your platform.

Learn how CodeWarrior can help your platform dominate the market.

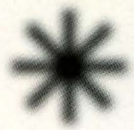
See you at ESC West in San Jose!

Visit www.metrowerks.com.

CodeWarrior

metrowerks
Software Starts Here

© 2000 Metrowerks Inc. All rights reserved. Metrowerks is a Motorola company. Metrowerks, the Metrowerks logo and CodeWarrior are trademarks or registered trademarks of Metrowerks, Inc. or its subsidiaries. Palm is a trademark and Palm OS is a registered trademark of Palm, Inc. or its subsidiaries. All other products are trademarks of their respective companies and are hereby recognized.



Tools for Embedded Developers

Software

API

An API for the MicroC/OS-II kernel allows applications to run atop the OnCore Microkernel within a protected memory management unit (MMU) partition. With this API, developers can migrate their applications immediately into an MMU-protected environment. In addition, users are able to execute Linux, POSIX, and other industry standard embedded applications in concert with their MicroC/OS-II applications on the same silicon. The MicroC/OS-II API is available now.

OnCore Systems
Half Moon Bay, CA
(650) 712-0655
www.oncoresystems.com

Debug support for OS

Run-mode debugging support is now available for applications that use the ThreadX real-time operating system and ARM9E processor. The debug mode, an enhancement to the MULTI 2000 integrated development environment, allows designers to debug their applications on a thread-by-thread basis while the application continues to run. While the application runs, Multi halts the thread whenever it hits a breakpoint and opens a source code debug window specific to that thread. Programmers can then single-step or browse through the halted thread and read memory and registers, all without disrupting program execution. Run-mode debugging for the MULTI 2000 IDE is available now.

Green Hills Software
Santa Barbara, CA
(805) 965-6044
www.ghs.com

Debugger

A set of debugging tools has been added to a development suite for the ColdFire family of 32-bit microcontrollers. The tools include a source-level simulator and a remote debugger that are almost identical. A script file to configure the debugger can be automatically generated by the simulator. The development suite as a whole is made to support the MCF5407 chip, as well as the earlier series 2 and 3 ColdFire cores. With the addition of the debugging tools, developers can create, build, and debug their programs with or without hardware. When hardware is available, the parallel port BDM debugger extends the source level debugging to the hardware itself. The debugger is available now.

Crossware
Litlington, UK
44 (176) 385-3500
www.crossware.com

DSP design software

Hypersignal RIDE is component-based DSP design software with direct support for Texas Instruments' DSK6211 and DSK6711 hardware. Using the system, designers can pull block components of code from a menu or tool bar, then connect them with a mouse to establish a data flow for the design. Run-time parameters are adjusted from pop-up boxes, and the entire algorithm is executed on the target DSK with the press of a button. RIDE also allows symbolic debugging, code profiling, and application export. It supports a variety of DSP chip families and includes drivers for a number of DSP hardware boards in addition to the ones from TI. RIDE is available now and costs between $3,995 and $9,995.

Hyperception
Dallas, TX
(214) 343-8525
www.hyperception.com

Development toolkit

CCE-SDT is a development toolkit for the ARM family of microprocessors and cores. It can be used for developing and benchmarking code, software simulation of code execution, downloading code to target by means of a debug monitor, and debugging with a JEENI emulator. To perform these tasks, CCE-SDT utilizes the following features: C/C++ compilers, assemblers, and linkers for ARM and Thumb instruction sets; an integrated development environment, including ARM Project Manager; a full source-level debugger; an instruction set simulator; and a ROM-based debug monitor. CCE-SDT is available now. It starts at $2,995.

Embedded Performance Inc.
Milpitas, CA
(408) 957-0350
www.epitools.com

Test software

Tau Logiscope v. 5.0 is a test environment for large development organizations. It reviews code and detects error-prone modules automatically. Featuring visual outputs such as call graphs, control graphs, and inheritance graphs, Tau Logiscope provides various means for giving a graphical image of manually written code. It maintains development guidelines by detecting program-

ming rule violations. It is available for Windows 98 and Windows NT 4.0.

Telelogic
Princeton, NJ
(609) 520-1935
www.telelogic.com

Development environment

Statemate MAGNUM v. 2.1 is a development environment aimed at the automotive software industry. It allows engineers to create visual electronic specifications that can be communicated to suppliers and implementation teams, along with traditional textual documentation. It is integrated with Rhapsody in MicroC, which enables suppliers to return data for comparison with original specifications, all based on a common electronic representation. In addition, interactive test vector automation capabilities let system engineers capture functional and behavioral test scenarios. It also features an automatic panel builder that allows designers to debug models by creating mock-ups of user interfaces.

I-Logix
Andover, MA
(978) 682-5995
www.ilogix.com

Abstraction layer

The pSOS Abstraction Layer enables users to run their existing pSOS applications directly on top of the Linux/RT operating system. pSOS customers who want to migrate to Linux/RT have only to recompile their code and link with a pSOS library available for Linux/RT. This will give them real-time capabilities within a true Linux kernel. These capabilities include high-resolution timers and clocks. Resource reservation can be added without modifying application code by using the TimeSys Resource Manager. New functionality is also a possibility because users are able to add new real-time tasks while their migrated pSOS apps run as they did previously. An integrated visualization and monitoring function is also available for real-time debugging. The pSOS Abstraction Layer is available now in source code form and on a royalty-free basis, as are the base components of the Linux/RT.

TimeSys
Pittsburgh, PA
(412) 681-6899
www.timesys.com

RTOS and debugger

The HiTOP language debugger now features operating system-oriented test and analysis for osCan, an OSEK VDX-compliant OS from Vector Informatik. HiTOP allows the display of information about the status

www.archimedessoftware.com
#1
technical support
in the industry!
ide
68HC05
68HC08
68HC11
68HC12
68HC16
683XX
8051
8051XA
80251
"The 8051 produces extremely compact code. The simulator drastically reduced the development cycle. I really enjoyed working with your software."
Colin Redknapp
Lockheed Martin
ARCHIMEDES SOFTWARE
1-800-338-1453
40 Lake Bellevue suite 100
Bellevue, WA 98005

Dinkum C++ for GCC/Solaris
Dinkum C++ for GCC/Linux
A COMPLETE C++ Library
as mandated by the final C++ Standard
ISO/IEC 14882:1998
Includes the Dinkum C Library
Comes ready to run with GCC 2.95.2
on either SPARC Solaris or PC Linux
Single User
Binary License for $90
Source License for $400
Dinkumware, Ltd.
Genuine Software
www.dinkumware.com
*Dinkumware & Dinkum are Registered Trademarks of Dinkumware, Ltd.
Solaris & SPARC are Registered Trademarks of Sun Micorsystems , Inc.

IF YOU USE:

ATI Nucleus™

Embedded Power RTXC™

Express Logic ThreadX™

Kadak AMX™

Precise/MQX™

Your own custom RTOS

ON:

ARM®

StrongARM

ARC

PowerPC™



Contact MetaWare for a free evaluation www.metaware.com/eval

Thanks to its object-oriented architecture, MetaWare's SeeCode™ debugger can be set to fit your project requirements. Through documented APIs, SeeCode™ interfaces with system-level tools, software/hardware execution tools, or other key elements of your own environment. The RTOS API can make SeeCode aware of your RTOS, even if it's home-grown.

MetaWare® INCORPORATED

2161 Delaware Avenue, Santa Cruz, CA 95060 USA • Toll-free: 1.877.TOOLSET (866.5738) • Fax: 1.831.429.9273

12 Rue Saulnier, 92800 Puteaux, France • Tel: 33 (0)1 47 28 86 31 • Fax: 33 (0)1 47 28 46 99

MetaWare is a registered trademark of MetaWare Incorporated. All other company and product names are trademarks or registered trademarks of their respective companies.

of operating system objects such as stack size or task-specific breakpoints. The dynamic system behavior—the so-called event trace—is logged as well. OS events such as task changes and calls of the OS nucleus are recorded in real time. Each event receives a time stamp at the time it is logged. These data enable the user to determine both the order and duration of the individual tasks and their operating system calls. Besides the dynamic behavior overview, the individual OS objects are shown with all their attributes. HiTOP can also monitor the stack for each task. A Task window offers information on status, priority, and event flags of all tasks. It's available now.

HiTex Development Tools
Sunnyvale, CA
(408) 733-7080
www.hitex.com

Vector Informatik
Stuttgart, Germany
49 (711) 13 99 96 - 0
www.vector-informatik.de

Hardware

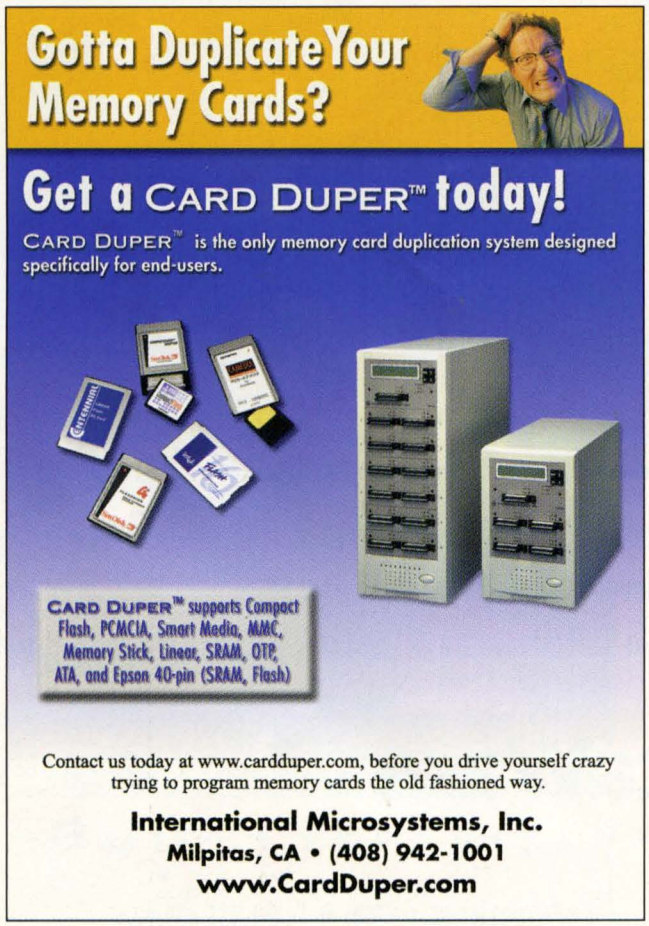
Debug tool

The Flex-BDM/68HC12/OSEK, a background mode debugger, debugs real-time embedded applications based on Motorola's OSEK/VDX RTOS. It's compatible with all members of the 68HC12 family of processors and allows developers to access OSEK-specific resources such as tasks, stacks, resources, messages, alarms, and counters. It integrates a BDM cable with PC-based, high-level debug software running on Windows NT, 2000, 98, or 95. Flex-BDM/68HC12/OSEK can program the processor's on-board flash memory without additional hardware or a separate programming voltage. A "Hot Insert" capability lets the debugger connect to the live embedded target without the system reset necessary with in-circuit emulators. Using this debugger, developers can step C code from all memory areas, read and modify target variables and on-chip registers, and configure hardware and execution breakpoints. The Flex debugger is available now.

Noral Micrologics
South Natick, MA
(508) 647-0103
www.noral.com

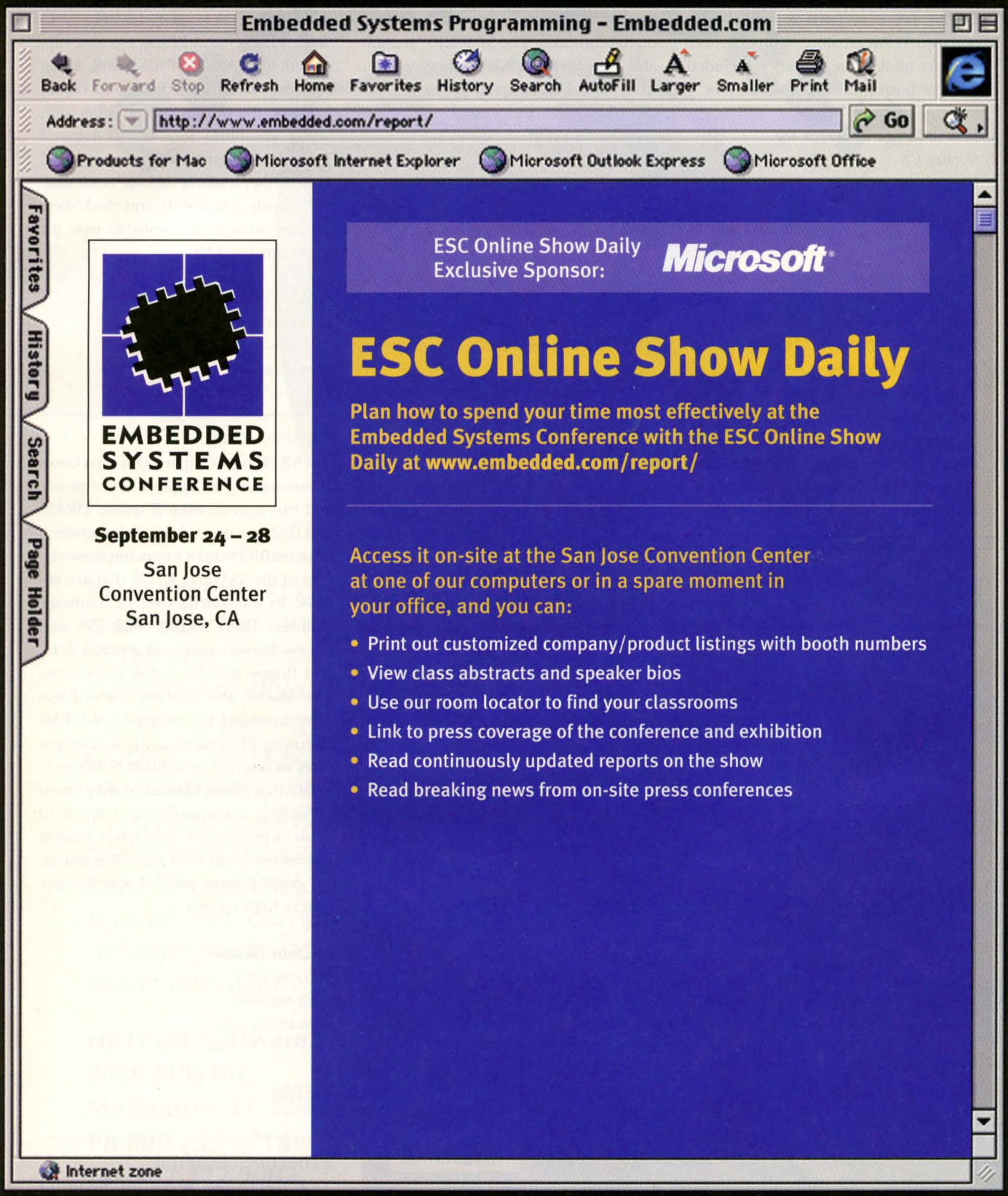
Development board

The DEV9 is a ready-to-run code and ASIC development platform. It consist of a universal motherboard and an ARM940T-based daughter board. It features 512KB flash memory; AMBA, ASB, and APB support; 128KB synchronous





www.embedded.com/report



CMP

SRAM; 512KB configurable SRAM; 512KB ROM; two 72-pin SIMM slots for DRAM; two serial ports; a parallel port; and two PCMCIA slots. The CCE-SDT software toolkit (see p. 189) is included with the board, which is available now.

Embedded Performance Inc.
Milpitas, CA
(408) 957-0350
www.epitools.com

Chips

Flash microcontroller

MMC2107 is the first member of the M-CORE flash microcontroller family. It's a 32-bit chip designed to meet the needs of distribution channel customers designing industrial control and automation applications. Utilizing an M-core microRISC CPU, it integrates both digital and analog peripherals, including a powerful 16-bit timer, communication interfaces, and a queued analog-to-digital converter. The MMC2107 is available now in 100- and 144-pin versions. They cost $15 in quantities less than 25,000.

Motorola
Austin, TX
(512) 895-6217
www.motorola.com

Programmable logic chip

The 1.5 million gate APEX EP20K1500E is a chip that combines a processor core with programmable logic. It's aimed at communications applications such as layer 3 routers and switches, wideband CDMA, and baseband signal processing, among others. It contains 51,840 logic elements and it features True-LVDS, which enables the chip's 16-channel differential signaling I/O to perform at 840MB/s. Additional I/Os, such as SSTL 3/2, GTL+, AGP, LVTTL, LVCMOS, HSTL, and PCI66 are also supported. The APEX EP20K1500E, along with the rest of the APEX chip family, features the MultiCore architecture, which combines look-up table-based logic, flexible embedded memory, and PLLs. Other features include clock multiply/divide, phase shift, and clock skew. Masked versions are available now for approximately $100.

Altera
San Jose, CA
(888) 325-8372
www.altera.com

PC-on-a-chip

MachZ PC-on-a-Chip is a chip that boots autonomously on application of power and can operate even if system DRAM and flash are unavailable. It integrates a PhoenixBIOS and a Linux implementation of the VxWorks RTOS that are tailored to the MachZ's set of hardware features. These include PCI, ISA and Access buses, serial and parallel I/O, and floppy and hard disk controllers. The MachZ also contains a set of features intended for designers of OEMs featuring PC functionality. It incorporates an internal redundant boot mechanism that allows system recovery (even if BIOS is corrupted), serial access to Flash, a proprietary chip select control mechanism, and dual watchdog timers. It's available now, and starts at $30 per unit in high volume.

ZF Linux Devices
Palo Alto, CA
(800) 683-5943
www.zflinux.com

OEMs

Ethernet terminal

The Intellicom series of operator interfaces offer direct connection to a

megatel
Small Space Solutions
Offering a Wide Range of Embedded PC Architecture
PC/II+p
PC/II+vxe
PC/II+dx
PC/II+dxe
On-board Features
• 686 to 386, 8088 or V40 Processors
• Local Bus Super VGA and Panel Video
• Up to 128MB of System Memory
• Ethernet LAN, Programmable Flash
• Serial, Parallel, USB and IrDA Ports
• 4"x4" or PC/104 form factor
• Low Power Consumption
• Operation +5 volts only (typical)
• PC/104 or ISA Bus compatible
1-888-smallpc
Tel: (416) 245-2953
Fax: (416) 245-6505
sales@megatel.ca
Embed This
megatel.ca

Personal Best



There's a feeling about achieving a goal, or simply reaching a new level of performance that's hard to put into words. It's not about winning or beating the next guy necessarily, just about knowing that you've done more than you've ever done before.

HI-TECH C users get that feeling often - the feeling of being "in the groove", the realization that not only did you meet your deadline, but you did it easily, without stress.

If you program embedded microcontrollers, and you want to raise your performance level, get HI-TECH C. World-beating performance from a tool that gets the job done without getting in your way, and backed by support our competitors can't believe - that's what *you* need to achieve your personal best. Call us today, or visit our web site for demos and information.

www.htsoft.com

HI-TECH Software
7830 Ellis Rd.
Melbourne FL 32904
Ph 800 735 5715
Fax 407 722 2902

HI-TECH
SOFTWARE

10BaseT Ethernet network. Standard features include a 4x20 character display with backlighting, a tactile feedback keypad with customizable legend, RS-232/RS-485 serial ports, eight general purpose I/Os, flash, and 128KB SRAM. Full TCP/IP source code is provided along with Dynamic C Premier, which is software consisting of an editor, a compiler, and a debugger. Ethernet drivers for the Realtek NE2000 chipset, which allows OEMs to write directly to TCP or UDP sockets, are included, as are software drivers for TCP/IP, I/O, keypad, LCD, and speakers. The Intellicom series comes in two versions. The OP6700 costs $289. The OP6600 is $219, but does not feature Ethernet connectivity. They are both available now.

Z-World
Davis, CA
(530) 757-377
www.zworld.com

Video card

The Angelo is an AGP video graphics card for VMEbus and CompactPCI baseboards. Angelo features 8MB of 100MHz video memory, which enables it to deliver resolutions of up to 1600-by-1200 at 32 bits per pixel with a refresh rate of up to 72Hz. Angelo features a 64-bit memory bus that allows the graphics processor to access video memory at up to 800MB/s. Two communications pathways exist between Angelo and the host-board: a standard 32MHz PCI bus and a 66MHz AGP port that enables the baseboard to pass video and graphics data without tying up the PCI bus. A 15-pin VGA connector is also provided for direct connection to a monitor. Angelo starts at $395. It's available now.

General Micro Systems
Rancho Cucamonga, CA
(909) 980-4863
www.gms4VME.com

Single-board computer

StingRay II+ is a Pentium III/Celeron FC-PGA single-board computer. Based on the Intel 82440BX PCI chipset, it features bus speeds of either 66MHz or 100MHz, as well as Socket 370 performance up to 850MHz with dual processing capability. On-board features include 10/100 Ethernet, Ultra DMA/33 support, ISE, Ultra Wide SCSI, SVGA, and system monitoring facilities. Using four 168-pin DIMM sockets, it supports up to 1GB of SDRAM. Interfaces for the StingRay II+ include two USB ports, a floppy disk connector, a bi-directional parallel port, and a serial port. It is available now for $2,492.

I-bus/Phoenix
San Diego, CA
(858) 974-8400
www.ibus.com

Custom C++ Code Compliance Analysis since 1984!

CODEFIX® Version 2.0

C++ CODE MAPPING

CODEFIX Version 2.0 is a programmable tool for modifying all C and C++ source code on a file or project basis.

- Obfuscation and shrouding for C++ programs.
- Insertion of code for dynamic runtime testing.
- Database generation from C++ source code.

CODECHECK® Version 9.0

SOURCE CODE ANALYST

Includes "Drop-In" Rules for Compliance analysis, Adherence to specifications, Measures of complexity, Silent error detection, Code maintainability, and Portability.

- **Maintainability** – CODECHECK identifies and measures complex, sloppy, and hard to maintain code.
- **Compliance** – CodeCheck allows your corporate coding and project specification standards to be completely automated for compliance validation.

PCYACC® Version 9.0

PROFESSIONAL LANGUAGE DEVELOPMENT TOOLKIT

Includes "Drop In" Language Engines for SQL, dBASE, POSTSCRIPT, HYPERTALK, SMALLTALK-80, C++, C, PASCAL, PROLOG, FORTRAN, COBOL, BASIC, SGML, ASN, RPG, REXX, PL1, SNA, RTF, VISUAL BASIC, SQL2, DB2, VHDL, HTML, VMRL, JAVA, ODMG-ODL/OQL, SQL3, MODULA-3, DELPHI, VBS, and ADA.

- Portable Object Oriented Classes for C++ and JAVA – Error, Symbol Table, Syntax Tree, Yacc, and Lex.

30 day Money back guarantee! Free AIR Shipping anywhere in the world!

www.abxsoft.com

ABRAXAS™ Software, Inc.
4726 S.E. Division St.
Portland, OR 97206

TEL (503) 232-0540
FAX (503) 232-0543
E-Mail: sales@abxsoft.com

To Order Call
1-800-347-5214

DOS $495, MAC $495, LINUX $495, WIN 2000 $995, UNIX $1995.

Put Linux Anywhere™



LARGE... High-Availability Servers

SMALL... Embedded Microcontrollers



...AND EVERYTHING IN BETWEEN:

- Smart appliances
- Modems
- Credit card readers
- Radar controls
- Industrial automation
- Mobile phones
- Thin clients

Whether your target design includes an embedded processor with a small footprint or is the next market-winning Compact PCI High Availability solution, Lineo delivers a Linux®-based solution to meet your needs.

Our uClinux™-based microcontrollers deliver sub-1 MB memory footprints and Lineo's HA Enabler delivers 99.999% reliability on High Availability Compact PCI servers.

So partner with Lineo and Put Linux Anywhere.

To learn more about our tools, support and services, visit us at www.lineo.com or call 1.888.463.7367

LINEO™

© 2000 Lineo, Put Linux Anywhere and uClinux are Trademarks of Lineo, Inc. Linux is a registered trademark of Linus Torvalds.

A 500 ps logic analyzer for $7,000?
Where's the glitch?



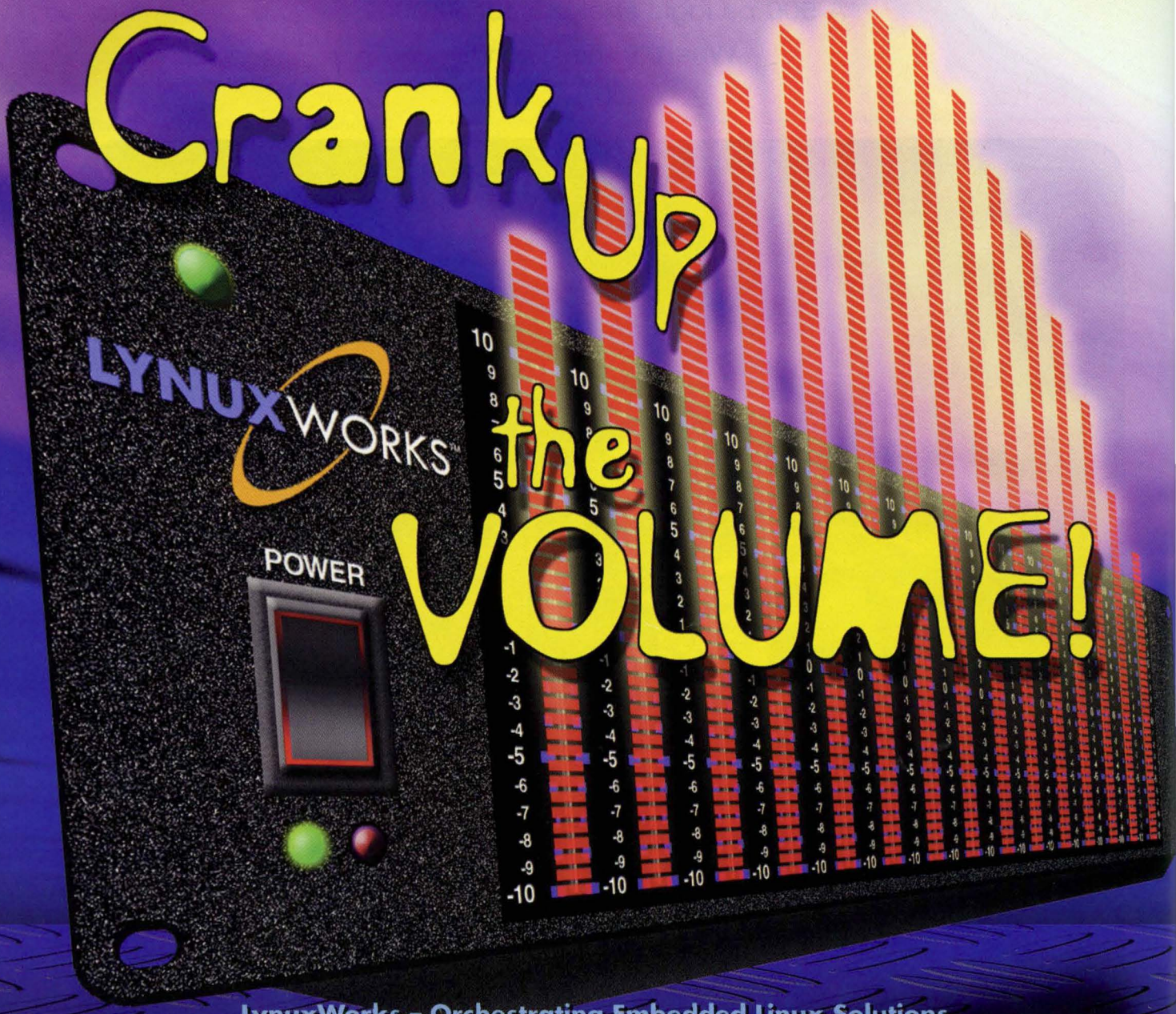


TLA600 Logic Analyzer At $7,000,* you can't afford to miss this one. Here's powerful logic analysis — through a Windows interface — that provides 500 ps timing with up to 200 MHz state on every channel through the same probe. Look for comparable timing analysis elsewhere, and you could spend nearly four times as much. See this and believe. Get your free XYZ's of Logic Analyzers by calling 800-426-2200 x3059 or visit **www.tektronix.com/TLA**

Tektronix®

*TLA600 Series starts at $7,000 (manufacturer's suggested retail price).

©Tektronix, Inc. 2000. All rights reserved. TEKTRONIX is a registered trademark, and the Tektronix logo is a trademark of Tektronix, Inc. All others are properties of their holders.



LynuxWorks – Orchestrating Embedded Linux Solutions

LynuxWorks has powered-up to help you crank up volume on your embedded computing projects. Our recent merger with ISDCorp, a premier provider of quality Linux ports to embedded customers, has enhanced our ability to deliver proven results and tangible solutions.

Our reliable, scalable products – the Linux-compatible LynxOS® real-time operating system and the open-source BlueCat™ Linux operating system – along with our professional services and network of value-added partners can help you develop better embedded computers with a faster time to market – starting today. This is what LynuxWorks orchestrates and we bring it all to you in perfect harmony. Our decade of experience and ISO9001 certification makes us the low risk, proven partner you need to make yourself be heard in the embedded Linux market.

If you're tired of the noise other companies are making, but you're ready to crank up the volume with a proven solution,

you're ready for LynuxWorks.

For more information on LynuxWorks, our product and services visit our web site www.lynuxworks.com.

LynuxWorks and BlueCat are trademarks and LynxOS and Lynx Real-Time Systems are registered trademarks of LynuxWorks, Inc.
Linux is a trademark of Linus Torvalds. All other trademarks and product names are the property of their respective holders.

JEFFREY TAYLOR

Book Review:

Doing Hard Time: Developing Real-Time Systems with UML, Objects, Frameworks, and Patterns

by Bruce Powel Douglass

Bruce Powel Douglass is a writer with an attitude. He is passionate and enthusiastic about real-time systems, safety and reliability, and direct execution of models. In Texas, to where I've just moved, attitude, passion, and enthusiasm are a part of the culture. Your mileage, as they say, may vary.

"'Safe enough' looks different at 35,000 feet."

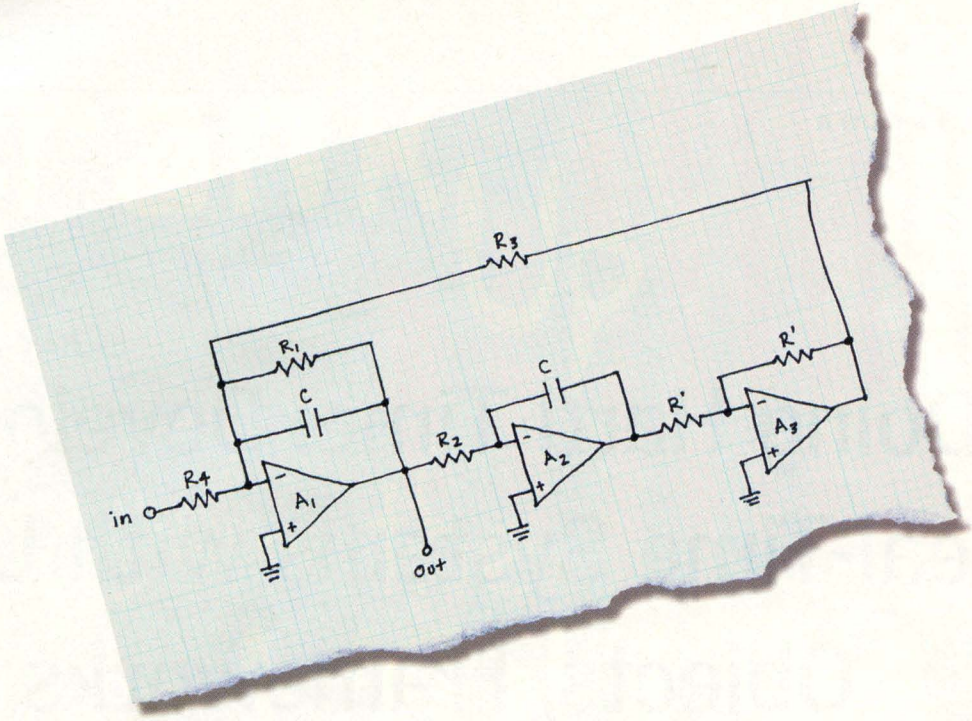
The Boeing 777 has been described as a million lines of Ada flying in tight formation. It is a completely fly-by-wire system. No mechanical or hydraulic linkages sit between the pilot's controls and the control surfaces, just those lines of code. I've got an attitude, too, about the safety and reliability of that code.

In spite of the subtitle, this book is relevant to most software developers, not just real-time developers. Even though embedded systems are the most common kind of computer (there are more microwaves and VCRs than mainframes and PCs), true real-time systems are not common. Mostly people want real fast. And they want it real soon. Most of the problems addressed by this book are present in modern applications with GUIs and Web or network connectivity. Only the sections on schedulability apply exclusively to real-time systems.

Although not included in the subtitle, process is a big part of this book. Douglass pushes process but is also realistic about it (for example, architectural design can be skipped in small systems). He lists the relevant IEEE standards and says that most can be fulfilled in a couple of pages. The purpose of process is to ask the right questions at the appropriate time, not fill a cabinet with "objective evidence" for auditors.

Douglass's flavor of process is Rapid Object-oriented Process for Embedded Systems (ROPES). If you've been following OOA/D development, there is nothing terribly radical here. He puts a heavier emphasis on safety than most treatments. He is an advocate of code generation from UML, so his usage of UML is more complete than most. UML statecharts, a kind of state diagrams derived from Harel statecharts, get a lot of coverage. This is because they fully specify the behavior, not just the interfaces. UML statecharts can be executed and debugged at the UML level. Class diagrams, on the other hand, can be used to generate header files but not executable code.

Some of his ideas have me jumping up and down, saying, "yes, that is what I've been trying to do." Others do not. For



Analog spoken here.

In this digital world, analog engineers often tell us they feel misunderstood. When the talk around the table is in ones and zeros, they're thinking in terms of currents and voltages. They say it's almost as though if you're doing analog, you're living in a different place. And speaking a different language.

If this is *your* world, we've got some good news: now you have a place you can feel right at home. We call it Planet Analog. And it's built just for you.

Planet Analog is a place on the Web devoted entirely to analog technology. A place where you can get the latest news, in-depth technical coverage and product data. Plus articles contributed from some of the finest technologists in the world. And in coming months, we'll add seminars, chats, polls, conference registration and a lot more. All under the leadership of Steve Ohr, regarded by many as the leading analog advocate in the world today.

So if you're an analog engineer, come to www.planetanalog.com for a place to call your own. You'll find we speak your language.

SPONSORED BY:

Texas Instruments

Analog Devices

National Semiconductor

EDTN NETWORK PARTNERS

EBN, EET, SBN, Embedded.com, ISD Magazine, Power Designers, Home Toys, Electronic Design, Microwaves & RF, EE Product News, Wireless Portable, Wireless System Design, PCN Alert, ChipCenter, Communications System Design, Design & Reuse, Circuits Assembly, HDI, PC Fab, Printed Circuit Design

www.edtn.com

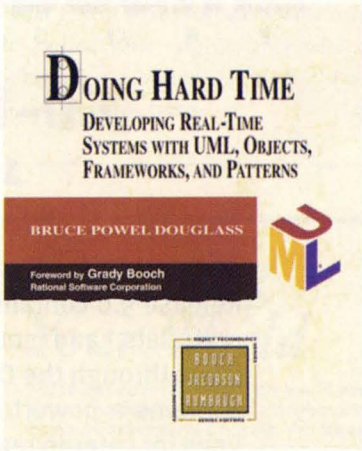
EDTN network

It's all right here.

example, I am not yet convinced of the value of the direct execution of models (for example, simulating the system inputs and watching an animation of the state diagrams). A lot depends on the quality of the tool.

UML is a language and as with any language, it takes some time to develop fluency. The book includes a whole section of statechart patterns that are worthwhile in their own right and as practice to develop fluency. Most patterns I've seen have been class diagram patterns. This is the first time I have seen patterns of temporal behavior. It's one of the ideas that made me say, "yeah, that's it."

I recommend this book with a caveat, however: it's full of cynical footnotes. For example, the footnote to the sentence "One problem that arises with environmental interaction is that the universe has an annoying habit of disregarding our opinions of how and when it ought to behave" reads "It is like teenagers in this regard." In a previous job, I sat next to Larry McAlister, one of the peer reviewers for the book,



while he was going over the proofs. I sometimes wish he or one of the editors had deleted even more of the cynical footnotes. Too much cynicism leads to apathy, not action.

One person's cynicism is another person's truth. One of the book's diagrams includes Translation Defects (for example, compiler bugs) as an input to the development process. Is this true? Unfortunately, yes. Is making this explicit useful? Only as far as naming one of the many fudge factors that go into making an estimate.

The size of this book—750 pages—is both a strength and a weakness. Reading it takes a week or two. If you read it at bedtime, it will take even longer. This is dense, deep stuff. On the plus side, the extended length allows multiple examples and views of the important points. My understanding of several key points of UML and OOA/D deepened while reading this book. Multiple views (for example, architecture, analysis, and design class diagrams) of the underlying model are included, not a single class diagram that is elaborated over time.

DOS COMPATIBLE FILE SYSTEM

ERTFS - EMBEDDED REAL TIME FILE SYSTEM

Since 1987 in Hundreds of Applications Worldwide

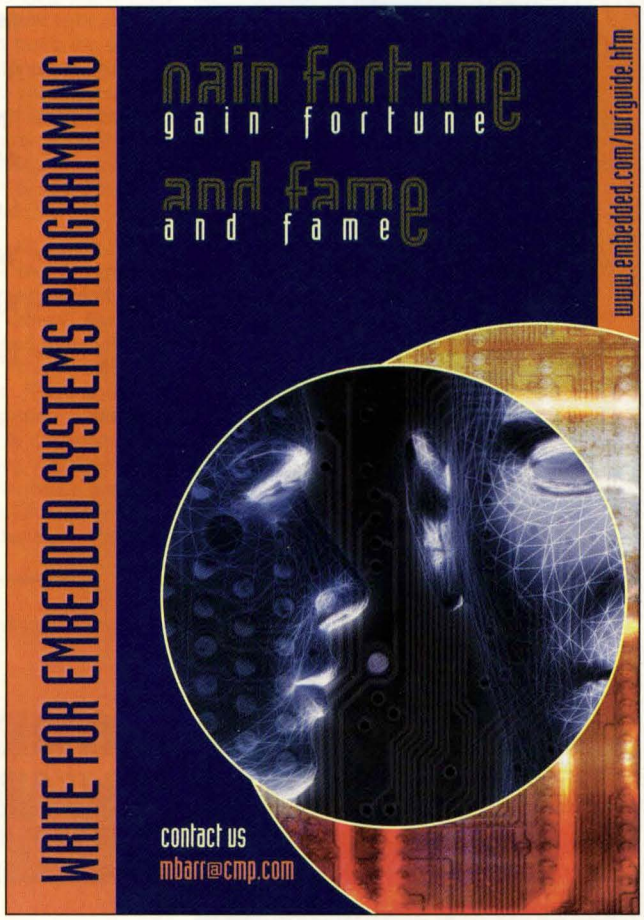
Selected as the File System for Several Major Embedded Operating Systems

DOS/Win95/FAT32 Compatible
Simple OS and CPU Porting Layer
Contiguous File Support
Realtime Extensions
Includes Support for IDE, Floppy, ROM/RAM Disk, PCMCIA, Compact Flash
CDROM Support Available
100% 'C' Source Code• Royalty Free

Only $4500^00

Embedded Systems ebs INC Software

Visit Our Web Site at:
www.ertfs.com
TOLL FREE 1 800 428-9340
Outside U.S. Call 978 448 9340
email: sales@ertfs.com





New for 2000...

EmbeddedSystems
PROGRAMMING

CD-ROM Library
1988–1999

The *Embedded Systems Programming* CD-ROM Library Release 5.0 contains all columns, features, news items, editorials, and source code from the 1988 premiere issue through the December 1999 issue. This time-saver contains a powerful text search engine and is a must-have for veteran readers and for those new to *Embedded Systems Programming*, the preeminent source of embedded development for more than 11 years.

Features Include:

- All columns, features, and source code from the premier 1988 issue through the December 1999 issue
- More than 1,200 articles—all columns and features
- A powerful text search engine
- The entire 2000 Buyer's Guide—more than 1,200 products covered in detail
- Code you can copy directly into your designs
- Windows, Unix, Linux and Mac compatibility
- All past and present author biographies
- Links to updated information on www.embedded.com

$79.95 new $29.95 upgrade

ways to order:

online
www.embedded.com

e-mail
orders@mfi.com

phone
(800) 444-4881 U.S./Canada
(785) 841-1631 other countries

Order it Online today!
www.embedded.com

The book comes with a CD-ROM that includes demos for Rhapsody, a "fully constructive UML Visual Programming Tool" and TimeWiz, an "integrated tool for timing analysis." The former is a competitor to Rational Rose. The latter performs timing and schedulability analysis. Douglass is the Chief Evangelist for i-Logix, the developer of Rhapsody. Both demos are crippled versions that will not save your work. I walked through the tutorial for Rhapsody. It looks promising. The statechart animation and debugger is by far the most interesting feature. I don't have one of the supported C++ compilers (Visual C++ 4.0 and 5.0, Borland C++ 5.0.2, Tornado 1.0, pRISM+ PPC 2.2.2, or pRISM+ 2.2.6) and was unable to try it out. The installation notes indicate that the demo version does not have statechart animation, only the high-end development package. Disappointing choice.

Both Rhapsody and TimeWiz require a significant investment of time to use properly. I suppose you could enter an already worked out project just to evaluate them. I'd want to road test something like this over the lifetime of a small project before committing a full-blown project to it. My suggestion to manufacturers is limit the size of the model, not the capabilities or use time. It isn't many projects, even toy or personal ones, that can be completely developed and the documentation updated in 30 days (or however long you can keep the program running without exiting).

Overall I like this book and intend to consult it on a regular basis. It is a book for working software developers (people who are already "doing hard time"), not undergraduate computer science students. Following its advice is a lot of work up front to save time in the long run. To appreciate it, you have to have already experienced that "quick and dirty" isn't quick.

The mark of a great book is how it holds up over the long haul. By this measure, *Doing Hard Time* is a very good book. Having read it through once, it now sits on the bookshelf next to my workstation for road testing. It's going to take a year or two of use to fully absorb. My preliminary judgment is that if you write software with multiple concurrent inputs and you are looking for a better way to build software, read this book. **esp**

Jeffrey L. Taylor has been a software engineer for over 25 years. About half of this time has been devoted to embedded and real-time systems. His current research areas are open source software, Linux, and software development. He may be reached at jeff.taylor@ieee.org.

Protecting Your World



Kidde-Fenwal is recognized the world over for excellence and leadership in the development and manufacturing of fire protection systems and commercial/industrial controls. We are also recognized for the outstanding salaries, comprehensive benefits packages and real advancement opportunities we provide our employees. Spark up a new career with Kidde-Fenwal.

Software Engineers

Experienced professionals of varied skill levels to design, develop and implement embedded and PC application software to support products in the fire detection and alarm industry. Responsibilities also include the analysis and investigation of major software engineering project tasks; the definition of design specifications and analysis to system architecture requirements; the development and application of major routines and software test plans and the resolution of difficult hardware/software compatibility and interface design considerations.

Selected candidate will have a BSEE, CSCS or equivalent and at least 3 years experience in embedded software development using C, Assembly and C++. Background using Hitachi, SGS Thomson, and Motorola 68332 processors is a plus. RTOS and object oriented design experience desired. A strong background in data communications and networking with knowledge of process instrumentation and industrial controls is helpful. Opportunities also exist for a leadership role.

We are an Equal Opportunity Employer.

Electrical Engineers

We seek a degreed (BSEE or MSEE) individual with 3 or more years of electronic design experience with a background in low power analog circuits, digital circuits, embedded microprocessor systems, printed circuit board/electronic packaging. Industrial control hardware design experience (analog and micro based) desirable. Experience with embedded software development a plus.

The successful candidate will have proven project management skills to take on a high level of responsibility for new product development in a cross functional team environment ranging from new product specification through design, test, approval, introduction and support within budgets and schedules.

Positions also available utilizing Assembly language on 8-bit microcontrollers.

We offer a competitive salary and excellent benefits, including health, dental, life and disability coverage as well as retirement and 401(k).

Qualified candidates should forward their resumes to:
Human Resources
Kidde-Fenwal, Inc.
400 Main Street,
Ashland, MA 01721
or fax to: (508) 881-6134,
or e-mail to:
jobs@kidde-fenwal.com



KIDDE FENWAL

www.kidde-fenwal.com

workingEngines[*inc]
Post-PC Era Talent Firm

placement / consulting / alliances

We strategically match, technically and culturally, highly skilled engineers with leading-edge companies who build smart devices and development tools.

Our vxJobs Practice focuses exclusively on positions requiring VxWorks expertise.

3 0 3 - 6 2 8 - 5 5 6 0
w w w . w o r k i n g E n g i n e s . c o m
w w w . v x J o b s . c o m

(Complimentary resume & interview support. All fees employer paid.)

VxWorks is a registered trademark of Wind River, Inc.

scientific.com
Software and Hardware Professionals
Nationwide opportunities available for:
Engineers/Designers
Quality Assurance/Testers/Verification

- Embedded SW
- RTOS
- ASIC/FPGA
- IC Design
- C/C++
- Firmware
- VLSI
- Java

Scientific Placement, Inc.
800-231-5920 800-757-9003 (Fax)
crs@scientific.com

Free Resume Assistance
All fees are employer paid

DESIGN/DEVELOPMENT
ENGINEERS:
NES

OPPORTUNITIES IN:
Wireless Communications (Data, PCS, Cellular, Networks, Satcom), Digital Imaging, Computers, Software, Semiconductors, Medical, CATV, Defense, Process Control, Consumer Electronics

SKILLS IN ANY OF THE FOLLOWING:
Embedded SW, OOD/OOP, C, C++, WindowsNT/98, Solaris/UNIX, JAVA, BIOS, VRTX, PSOS, DSP, MIPS, PCI, VME, Mixed Signal, ASIC/FPGA, VHDL/Verilog, Device Drivers, System Architecture, LAN/WAN, IP, Wireless Design, CDMA, GSM, Video Compression

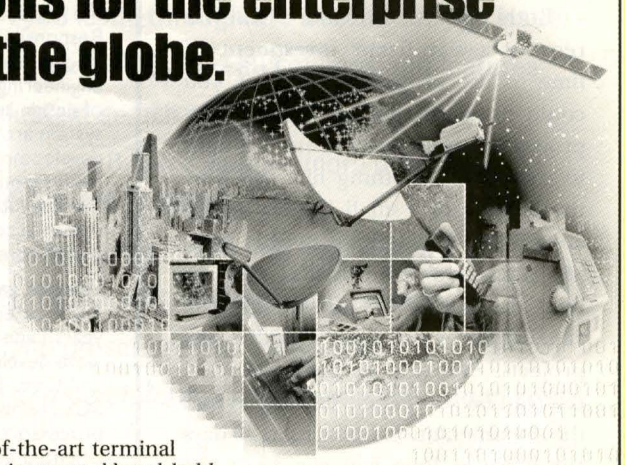
National Engineering Search® is the leading search firm placing Engineers nationwide. Contact us for immediate access to today's most exceptional engineering career opportunities! Our clients range from Blue Chips to today's newest emerging technology companies.

800/248-7020
Fax: 800/838-8789
esp@nesnet.com
See many of our current opportunities on-line at:
nesnet.com

What are you worth?
See our Online $alary Calculator!

Develop broadband solutions for the enterprise and mass markets across the globe.

Opportunities in San Diego, CA



Hughes Network Systems (HNS), The Broadband Company™, is the leader in the definition, design and deployment of next-generation, high-speed satellite delivery systems. From our dynamic world leadership position in satellite business networks, to the emergence of our consumer products like digital direct-to-home television (DirecTV) and its high-speed Internet access service product line (DirecPC), HNS is capitalizing on the explosive telecommunications market. And now we're preparing to launch Spaceway, the next generation of high-speed, broadband delivery systems that are sure to impact the future of communications.

At our facility in San Diego, we develop a broad array of state-of-the-art terminal products from our flagship DirecWay VSAT terminals to our highly integrated hand-held mobile satellite terminals. We are building the next generation of digital audio broadcast equipment.

• Software Engineers (Embedded, RTOS)
We have exciting opportunities for Software Engineers to perform software development for mobile satellite terminals. Openings require a technical Bachelor's degree or equivalent (Master's degree desired), and minimum of 2 years related experience.

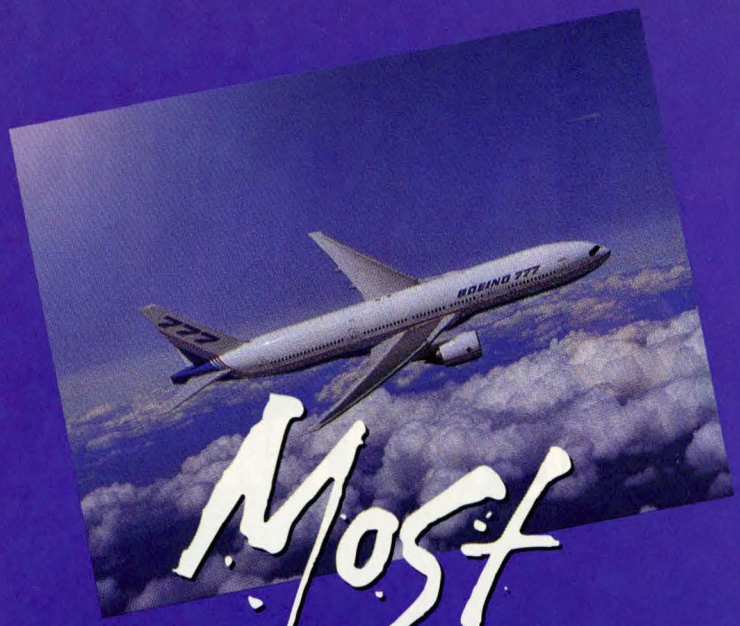
For other openings at HNS in San Diego, or for information on positions available at our Germantown, MD headquarters, visit us at **www.hns.com**

Rush your resume today to: **Hughes Network Systems, Human Resources/ MS- SD, 10450 Pacific Center Court, San Diego, CA 92121. Email (ASCII text only, no attachments please): recruitersd@hns.com. Fax: (858) 457-3863.**

HUGHES™
NETWORK SYSTEMS

HNS is an equal opportunity/affirmative action employer. We welcome and encourage diversity in the workforce.


3DO WANTS YOU.
We're looking for a few good recruits.
Engineers •
Producers •
Directors •
Designers •
Artists •
Send resumes to: Staci Taylor - resumes@3do.com
The 3DO Company, 600 Galveston Drive, Redwood City, CA 94063
www.3do.com



Most corporate ladders don't reach this high.

www.boeing.com/employment

BOEING
Forever New Frontiers

It is the policy of The Boeing Company to attract and retain the best qualified people available without regard to race, color, religion, national origin, gender, sexual orientation, age, disability, or status as a disabled or Vietnam-era veteran.

EMBEDDED SOFTWARE ENGINEER

SAN JOSE OR MASSACHUSETTS

Development of drivers for state-of-the-art measurement instrumentation. BS/EE/CS and at least 3 years' software development experience for real-time embedded multitask/multi-processing operations. C programming and Unix (Solaris) necessary, C++ and assembly (Motorola 68K) and VxWorks desirable.

Please forward your resume, including salary requirements, to:

LTX Corporation, University Avenue, Westwood MA 02090-2306; Fax: (781) 461-0185
E-mail: elizabeth_twer@ltx.com
Affirmative Action/Equal Opportunity Employer.

www.ltx.com

LTX CORPORATION

Reprints

Are Easy to Get!!!

Full of interesting articles and valuable industry news,* Embedded Systems Programming *has information you'd like to share. Reprints are great for trade shows, marketing kits, and presentations.

Contact Stella Valdez for information
916-983-6971

A CONSUMER ELECTRONIC LEADER
TAKING WIRELESS TECHNOLOGY TO THE NEXT LEVEL.



INTRODUCING WIRELESS CAREERS FROM PANASONIC.

We're Panasonic Wireless, the North American Design Center (Suwanee, GA & Calgary, Alberta, Canada) of an electronics giant. As an industry leader with a rapidly growing wireless unit, we can offer outstanding opportunities to ambitious, forward-thinking individuals. We're leading the way in digital technology with real-time embedded software and wireless/cellular projects. We have all of the excitement and advantages of a startup venture, yet with all of the rock solid support of a global electronics icon. Our headquarters is also located in Suwanee, GA. Just minutes away from the exciting, cosmopolitan city of Atlanta, we're still just far enough from the hustle and bustle to experience an outstanding quality of life.

SOFTWARE POSITIONS

Engineer I **SR Engineer**
Engineer II **Staff Engineer**

(Positions in Firmware, Protocols and Applications)

SYSTEMS & VERIFICATION

Test Engineer **Systems Engineer**

MECHANICAL

Engineer II

HARDWARE POSITIONS

Engineer I **Staff Engineer**
Engineer II **Technical Group Leader**

(Positions in RF and Logic)

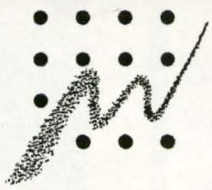
You'll enjoy a highly competitive salary, excellent career growth, and a comprehensive benefits package that includes medical/dental/vision, 401(k), tuition reimbursement, and product discounts. **For consideration, please forward your resume to recruiting@panasonicatlanta.com or fax 770-338-6001. Attn. Jobcode ES9/00.** EOE

Panasonic
Wireless Design Center
Matsushita Mobile Communications Development Corporation of U.S.A.

www.panasonic-wireless.com

Stretching the Intelligence Universe

Take the most creative minds in the industry, match them up with ***Applied Signal Technology,*** *Inc.'s years of expertise, and what you have is convergence of talent and intelligence unsurpassed anywhere on the planet today. We are creating premier digital signal processing equipment for the industrial and government sector. It's all happening right now at Applied Signal Technology, Inc. We currently have the following outstanding positions for dynamic individuals in our* ***California*** *Facility.*



Applied Signal Technology, Inc.

SOFTWARE ENGINEER

Embedded Controller Software Engineer

In this position you will be responsible for developing and maintaining embedded C software in conjunction with a telecommunications system. This assignment will involve detailed design, code and unit level test. You must be able to work as a team member with hardware and signal processing engineers to develop state-of-the-art communication signal processing equipment. This position requires a BSCS or equivalent and 3-5 years experience. Background in real-time software development with knowledge of C Programming and Unix is mandatory. Familiarity with VxWorks OS, Client/Server software architecture and device interfacing is highly desired. US Citizenship required, with the ability to obtain a security clearance..

Embedded Controller Software Engineer

In this position the candidate will be responsible for developing and maintaining embedded C software in conjunction with a telecommunications system. This assignment will involve detailed design, code and unit level test. You must be able to work as a team member with hardware and signal processing engineers to develop state-of-the-art communication signal processing equipment. This position requires a BSCS or equivalent and 3-5 years experience. Background in real-time software development with knowledge of C Programming and SCO UNIX is mandatory. Familiarity with Client/Server software architecture and device interfacing is highly desired. US citizenship required, with the ability to obtain a security clearance.

DSP Algorithm Development Engineer

In this position the candidate's responsibilities will include the design, development and implementation of advanced DSP algorithms for communication systems. This position requires a strong background in communication and DSP theory, algorithm development, and computer simulations (using C). Familiarity with demodulators, demultiplexers, modem algorithms and/or DSP chip programming desirable. MSEE or Ph.D. EE required. US citizenship required, with the ability to obtain a security clearance.

HARDWARE ENGINEER

Hardware Engineer

In this position the candidate will be a member of a small team developing telecommunication signal processing hardware. Responsibilities will include design, integration, and test of circuit cards and/or FPGAS. This position requires a BSEE/MSEE, or equivalent, with 2 years related experience. Knowledge of VHDL and C is desired. Experience with Xilinx FPGAS, digital communications, digital signal processing and an interest in understanding telecommunications systems are a plus. US Citizenship required, with the ability to obtain a security clearance.

Hardware Engineer

In this digital hardware design position, the candidate will develop deliverable, manufacturable equipment for communications and signal processing applications. The design will involve use of FPGAs, DSP (TMS320 family) processors, VME interfaces, and emulators for card debug. This position requires BS or equivalent in EE plus 2 years of related experience. Background in communications/DSP and an interest in understanding the entire processing system are a plus. US Citizenship required, with the ability to obtain a security clearance.

PROGRAM MANAGER

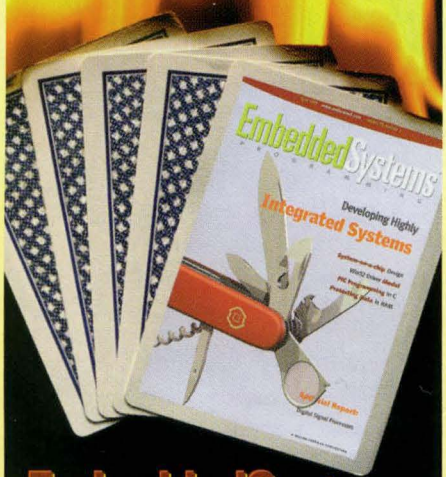
Program Manager

In this position you will be responsible for managing programs or groups of programs with staffs of 5 to 10 engineers. Your responsibilities will include project planning, staff planning, schedule and cost control, task organization and delegation, risk management, and customer interface, but may also involve direct program contributions in areas such as system engineering, test planning and documentation. You will also be involved with marketing, cost estimation and generation of proposals for new business. This position requires background in technical management of government contracts involving development of hardware, software or systems. Experience in personnel leadership/management, customer presentations, status reporting, earned value, and proposal writing is desired. Familiarity with digital signal processing and communications is a big advantage. This position requires a BS or BA degree and 3+ years of program or task management experience. US Citizenship required, with the ability to obtain a security clearance.

Please send your resume to: **Applied Signal Technology, Inc., Attention: HR/DD, 400 West California Avenue, Sunnyvale, CA 94086;**
fax to: **(408) 774-2700;** *or e-mail to:* **resume@appsig.com**
We are an Equal Opportunity Employer.

Visit us online at: **www.appsig.com**

Rent the Hottest List around!



EmbeddedSystems
PROGRAMMING

Over 55,000 Qualified Subscribers

Call Rubin Response
847-619-9800

EDS ▾ OPPORTUNITIES ▾

EDS is growing. Are you?

If you're stagnating in your present job, recent account wins have created huge new opportunities at EDS without cutting into your family life.

More than 100,000 professionals are enjoying the benefits of an EDS career. We offer excellent training, competitive salaries, and above all, room to grow. We provide cutting-edge systems and insights to top clients worldwide. If you're looking for a dynamic environment with all the opportunities you can handle, send us your resume today.

Senior Embedded Systems Software Engineers

- Embedded programming for microcontrollers using C and Assembly language including interrupt handlers and use of real-time executives
- Algorithm development, device drivers, Project Management, Formal Configuration Management, Bench development skills, J1850 and CAN

Embedded Systems Software Engineers

C or Assembler programming, Microprocessor Architecture and Interfacing experience, real-time microprocessor program development, Software Development Lifecycle, Requirements Analysis, Software Design, Coding, Unit and Integration Testing, Bench Development skills.

Windows CE Software Engineers

- 1-2 years Windows development experience using MS Visual C++
- Experience in Windows CE preferred, but will provide training if necessary

Engineering Associates

- 4-year degree in Electrical Engineering or Computer Engineering or equivalent work experience required

Positions available in North America.

Please mail, fax or email your resume, indicating position of interest, to: EDS Recruiting, Dept. 72-9722, Attn: Deborah Polvi, 700 Tower Drive, 5th Floor, Troy, MI 48098; Fax: (520) 833-9309; Email: deborah.polvi@eds.com.

EDS

EDS is a registered mark and the EDS logo is a trademark of Electronic Data Systems Corporation. EDS is an equal opportunity employer and values the diversity of its people. Copyright ©2000 Electronic Data Systems Corporation. All rights reserved.

ACHIEVE MAXIMUM CAREER VELOCITY

HERE'S A CHANCE TO PUT YOUR CAREER INTO OVERDRIVE.

Dallas, Texas
- Software/Hardware Design & Development Engineers
- ASIC Engineers
- Systems Development Engineers
- Software Test Engineers
- UNIX Systems Administrators

Milpitas, California
- Software Engineers
- Product Marketing Managers
- SQA Engineers
- Project Managers
- ASIC Design Engineers
- UNIX/NT Engineers
- Buyers

Calabasas, California
- Software/Firmware Engineers
- ASIC Engineers
- Test/Support Engineers
- Network Engineers
- Sales/Presales

Petaluma, California
- Software/Hardware Design & Development Engineers
- Software/Hardware Verification Engineers
- Software Diagnostic Engineers
- Test Engineers
- Applications Engineers
- Product Line Marketing Managers
- Program Managers

Nationwide Opportunities

Customer Service Group
- Installation Project Supervisors
- Installation Design Engineers
- Network Engineers (CNOC)
- Installation Planners
- Field Service Engineers
- Installation Quality Engineers
- Sales Support Engineers
- Technical Instructors
- CTAC Engineers

Spokane, Washington
- ASIC Design Engineers
- ASIC Simulation Engineers
- Software Engineers
- Software Test Automation Engineers
- SQA Test Engineers
- UNIX Systems Administrators

Dulles, Virginia
- Software Development Engineers-Core IP Product Development
- Network Engineers-1st and 2nd Level Support

Raleigh, North Carolina
- ASIC Design Engineers
- Component Engineers
- Software Engineers
- xDSL/voDSL Engineering Opportunities
- Software Diagnostic/ Customer Support Engineers
- Product Marketing Managers

For immediate consideration, please submit your resume, indicating position of interest and geographical preference, as well as department source code (Dept. EMS9100) to:

Alcatel
1000 Coit Road
MS HRPB6, Plano, TX 75075-5813
fax: (800) 561-4847; e-mail: recruit@alcatelusa.com

Alcatel USA is an Equal Opportunity Employer committed to diversity in the workplace.

ALCATEL

ARCHITECTS OF AN INTERNET WORLD

www.alcatelusa.com/jobs

In-Circuit Emulators

• 8051 • 80186 • 80196
• DSP 320Cxxx • TriCore • PIC

- Real-time trace
- Serial, parallel, & LAN interface
- C/C++ level Chameleon Debugger
- Multi-core debugging



FREE 2 WEEK TRIAL

www.signum.com

SIGNUM SYSTEMS
800-838-8012
805-523-9774
Fax: 805-523-9776
11992 Challenger Ct. • Moorpark, CA 93021

We can save your valuable time!

INTRODUCING! Full Featured In-Circuit Emulator



- Supports a wide range of micro-processors & micro-controllers
- Universal Architecture
- Real Time Filtering Trace
- 100MHz Bus Speed Support
- C/C++ Source Level Debugger
- Easy to Use!

Also ask about our Flash Programmer!

- Hitachi H8/H8S SH1/2/3/4
- Qualcomm MSM3000 MSM3100
- ARM ARM7/9 *"ETM support"*
- NEC V800/V850
- Motorola 68K/PowerPC
- Mitsubishi M16C/M32R
- Fujitsu SPARClite FR30/FR40
- Toshiba, Sharp, Panasonic, OKI & more !!

Yokogawa Digital Computer Corp of America
YOKOGAWA ◆
Tel: (408) 244-1932
Fax: (408) 244-1881
http://www.ydcusa.com
email: info@ydcusa.com

Please email us or visit our website for our new US office contact and full-line catalog!

ACROSSER www.acrosser.com
Tel: 886-2-8512-1616
Fax: 886-2-8512-2282
webmaster@acrosser.com

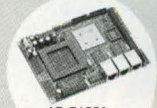
Total Embedded Solution

Embedded PC
PC/104 Module
386 / 486 / 586

Embedded OS
Embedded NT
Linux (Fully Support)

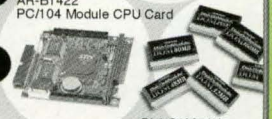
Embedded Storage
DiskOnModule

Embedded Housing
Microbox
PC/104 box



AR-B1550
Firewall CPU Card

AR-B1422
PC/104 Module CPU Card



DiskOnModule

* H/W with full embedded OS compatibility
* Flash Disk with no special socket and driver needed Only regular IDE required.
* With a complete embedded platform, Acrosser takes care all your embedded needs!

MICROPROCESSOR CORE SAVES COST AND DESIGN TIME

Design your custom board around the Rabbit 2000™ core module. Easy and rapid development, plus you save production and component costs!



$29 qty 100
$39 qty one
1.90" x 2.30"



- 40 I/O pins • 4 serial ports • 7 timers
- Battery-backed time/date clock
- 128K-512K SRAM • 256K flash
- Clock up to 25.8MHz

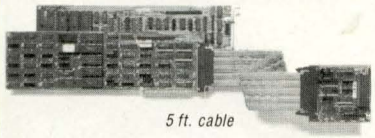
Dev. Kit Only $169
Includes RCM2020 core module, prototyping board, Dynamic C® software (not a trial version!), serial cable for real-time debugging, and documentation on CD-ROM

Z-WORLD
Call toll-free
888.362.3387
or buy online at
www.zworld.com

2900 Spafford Street, Davis, CA 95616 • Tel 530.757.3737 • Fax 530.753.5141

8051

In-Circuit Emulators



5 ft. cable

- Plug-in boards or RS-232 box.
- Choice of user interface: DOS, Windows or Borland keypress compatible. Hosted on PC's or Workstations.
- Supports virtually all derivatives of the 8051 family.
- Source-level Debugger with complete C-variable support.
- Real-time emulation speed up to 42 MHz.
- 64 bit wide, 256K deep trace with time stamp and "source line tracing."

Fax-On-Demand
Literature Request Service
408-378-2912

Call for a Free Demo Disk.
(408) 866-1820

nohau CORPORATION
51 E. Campbell Avenue
Campbell, CA 95008
Fax (408) 378-7869

TargetFFS™

Flash File System

- Re-entrant Embedded File System
- POSIX and Standard C API
- Guaranteed Integrity Across Resets
- Complete Wear Leveling
- "Thin" Driver Layer
- 100% ANSI C Source Code
- Royalty Free

BLUNK Microsystems
(408) 323-1758
www.blunkmicro.com

Fast. Reliable. Affordable.



NEEDHAM'S DEVICE PROGRAMMERS are the easiest and most cost-effective way to read, program and verify 2716 – 8 meg EPROMS. Support for Micros, Flash, EPROM, 16-bit, PLDs, Low Voltage and Mach (call for support list for specific models, or download demos from our BBS or web site). Easy to use menu driven software features on-line help, and a full-screen editor. Support for macros, read and save to disk, and split and set options.

- **Free technical support • Free software upgrades**
- **1 to 2 year warranty on all parts and labor**
- **30-day money-back guarantee • Made in the U.S.A.**
- **All models include software, on-line help, cables, and power transformers (where applicable)**

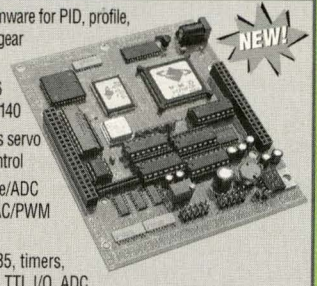
NE (916) 924-8037

NEEDHAM ELECTRONICS, INC.
4630 Beloit Drive, #20, Sacramento, CA 95838
FAX (916) 924-8065 • BBS (916) 924-8094
(Mon. – Fri. 8 am – 5 pm, PST)
http://www.needhams.com/

MotionC™ 2140

C/C++ programmable Low Cost, Standalone DSP Motion Controller

- Built-in firmware for PID, profile, electronic gear
- 4.7" x 3.8", 386EX/186 PMD MC2140
- 2- or 4-axis servo motion control
- Quadrature/ADC inputs, DAC/PWM outputs
- RS-232/485, timers, watchdog, TTL I/O, ADC



We have 30+ Low Cost Controllers with ADC, DAC, solenoid drivers, relays, PCMCIA, LCD, DSP motion control, 10 UARTs, 300 I/Os. Custom board design. Save time and money!

TERN INC.
1724 Picasso Ave., Suite A
Davis, CA 95616 USA
Tel: 530-758-0180 • Fax: 530-758-0181
http://www.tern.com
tern@netcom.com

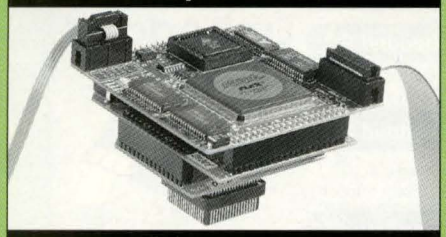
PromICE with Trace

The Leader in Memory Emulation

- Trace to pinpoint startup problems and isolate real-time bugs.
- Code Coverage to verify execution and speed up QA.
- Ultra-fast downloads via Ethernet, parallel and serial ports for Unix, Windows 95/NT and DOS.

Grammar Engine Inc.
Call Toll Free:
1-800-776-6423
www.gei.com

8051, 80C196 PIC®, AVR®



Integrated Development Systems:

Compilers, Simulators, Programmers,

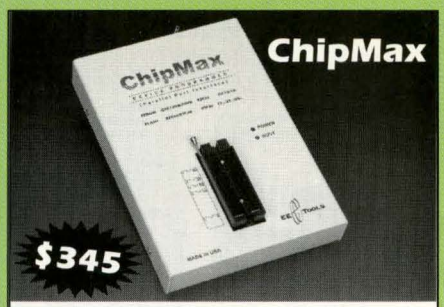
In-Circuit Emulators

Non-intrusive, with trace feature, hardware unconditional & complex breakpoints, triggers, programmable clock, memory mapping & banking, full project & source level support for C compilers...

www.phyton.com

(718) 259-3191 **Phyton**

ChipMax



$345

- Programs over 1200 devices (E(E)PROM, Flash, Serial, PALCE, GAL, 875x/895x, 93Cxx, 17xx, PIC16xx)
- Fast parallel link to any PC, even laptops
- 40-pin universal pin driver and current limit
- On-board processor and built-in power supply
- Unbeatable programming speed
- Checks for incorrect device insertion
- Automatic EPROM ID search
- Supports WIN95/98/NT
- NO fans & modules are required in circuit
- Made in USA, Lifetime free software
- Visit web site for demo software

EE Tools

Sunnyvale, California, USA
Tel: 408•734•8184
Fax: 408•734•8185
www.eetools.com

"The best emulator I ever used!"

8051



- more than 500 derivatives supported
- small emulation probes
- real-time access to internal bus
- can trigger on internal bus events
- cascading triggers
- trace with timestamps
- dual-ported emulation memory
- external trace and triggers
- excellent HLL support
- code coverage
- performance analysis

hitex DEVELOPMENT TOOLS

1-800-45-hitex **www.hitex.com**

INTROL

is a full featured cross-platform, multi-target microcontroller development environment.

68HC05
68HC08
68HC11
68HC12
68HC16
68332

- Simulators
- C Compilers
- Macro assemblers
- Source Level Debugger

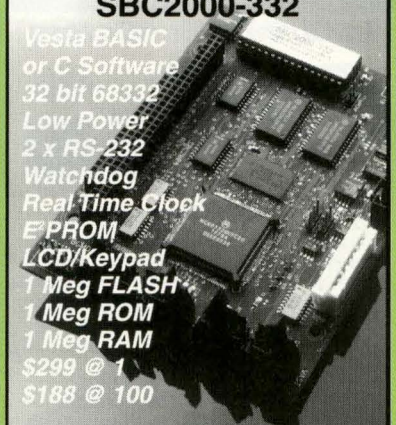
Integrated Development Environment

CODE comes with a full year of technical support and free upgrades.

Visit us on the web @ www.introl.com
email: sales@introl.com
U.S. and Canada: 800.327.7171
Other: 414.273.6100
FAX: 414.273.6106

Vesta Technology, Inc.
Single Board Computers for the 21st Century

SBC2000-332



Vesta BASIC or C Software
32 bit 68332
Low Power
2 x RS-232
Watchdog
Real Time Clock
EPROM
LCD/Keypad
1 Meg FLASH
1 Meg ROM
1 Meg RAM
$299 @ 1
$188 @ 100

(303) 422-8088

www.sbc2000.com

IN-CIRCUIT EMULATORS

The Right Emulator at the Right Price!

8051 • 68HC05
68HC11 • COP8 • CR16
DS80C320/520/530 nX65K
78K4 • SM6000/SM8500

Analog Devices • Atmel
Dallas • Intel • Motorola
National • NEC • OKI
Philips • Sharp • Siemens
SST • Temic • Winbond



- Windows-Based Interface
- Source-level debug
- Real-time/Non-intrusive
- Rentals available

MetaLink Corporation

Headquarters, Arizona, USA
Tel: 480.926.0797
Fax: 480.926.1198
sales@metaice.com
www.metaice.com

MetaLink - Europe GmbH
Tel: 49 (8091) 56960
Fax: 49 (8091) 2386
islinger@metalink.de
www.metalink.de

Debugger+Programmer



Complete System **$375**
F R E E
Demo/Prototype board

All tools **qualified** by Scenix Semiconductor

SX-ISD

- Supports SX18/20/28/48/52 • In-circuit run-time debugging • Real-time code execution • Source level debugging • Built-in programmer • Real-time breakpoint • Conditional animation break • External break input • Frequency synthesizer • Selectable internal frequency • External oscillator support • Software trace • Unlimited watch variables • Parallel Port Interface • Runs under Win 95/98/NT • Comes with SASM Assembler

Single, Gang Programmers and SMT adapters are also available

Advanced TransdAtA CORPORATION

Dallas, Texas
Tel 972.980.2960 **www.adv-transdata.com**

PIC Real-time 'Emulator'

$159



PIC-ICD

PIC-ICD
= Debugger + Programmer + DemoBoard

- Designed for 16F87X • Can also support most 16C6X/7X • In-circuit run-time debugging • Real-time code execution • 2.5-6V operating voltage • Source level debugging • Built-in programmer • Real-time breakpoint • Conditional animation break • 2 External break inputs • Selectable internal frequency • Software trace • Unlimited watch variables • Parallel Port Interface • Runs under PICICD IDE (Win95/98/NT) or MPLAB (Win95/98)

Advanced TransdAtA CORPORATION

Dallas, Texas
Tel 972.980.2960 **www.adv-transdata.com**

PIC16/17 Emulator

New Low Prices

Complete system starts at **$599**



RICE17A

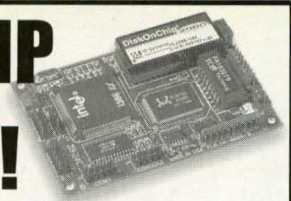
- For PIC12/16/17 • 3-5volt emulation • 64K program memory • 32K real-time trace • 12-clip external probe • Source level debugging • External break input • Trigger and break output • Real-time breakpoint • Unlimited software break and trigger points • Selectable internal frequency • Unlimited watch variables • Parallel Port Interface • Runs under Win95/98/NT

Probes for **16F87x** and **16C77x** by Jan 2000
With data break support

Advanced TransdAtA CORPORATION

Dallas, Texas
Tel 972.980.2960 **www.adv-transdata.com**

JUMP on the NET!



The µFlashTCP gives you 10BASE-T Ethernet connectivity, a full-function TCP/IP stack and 2 serial ports in a package 30% smaller than PC/104 solutions.

Field-proven TCP/IP stack, DOS and PC-compatible BIOS make development quick and easy.

Prices start at $169 qty 100.
Development kits are available.

Call 530-297-6073 Fax 530-297-6074
See us at the Emb Sys Conf San Jose #5028
www.jkmicro.com/uflash

JK microsystems

From the Author of WATTCP

eRTOS

DOS Realtime Kernel with TCP/IP Support

- *Preemptive & Cooperative threads*
- *Critical Section Protection*
- *Interthread Messaging*
- *Complete Re-entrant TCP/IP*
- *Web, CGI, FTP, Email, Telnet*
- *Web Graphics*
- *Interrupt-driven Serial Support*

www.ertos.com

Call 530-297-6073 Fax 530-297-6074

JK microsystems

NOW! Optimised ANSI C on the Microchip PIC.

the world's first

pic

ANSI C COMPILER

CALL NOW
1-800-735-5715
Fax 1-407-722-2902
www.htsoft.com/pic sales@htsoft.com

HI-TECH SOFTWARE

Embedded Systems Development Tools

TechTools™

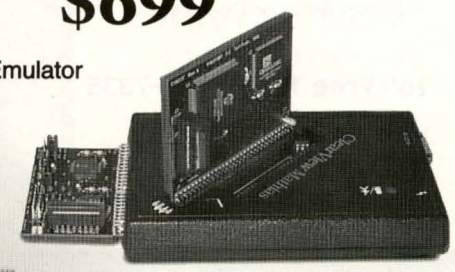


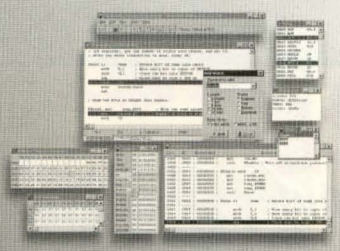
Complete PICmicro® Development System

Get the TOTAL Package for only

$699

PICmicro® In-Circuit Emulator
PICmicro® Assembler
PICmicro® Debugger
Windows® IDE





ClearView™ Mathias is a full-featured In-Circuit Emulator with a highly productive Development and Debugging environment for the PICmicro.

Fully emulates the selected PICmicro, including program memory, register memory, EEPROM, I/O activity, SLEEP mode and all peripherals.

Intuitive, Easy to Learn full-featured environment with integrated ClearView Debugger.

TDE provides integrated source-level debugging for ALL popular PIC Compilers and Assemblers.



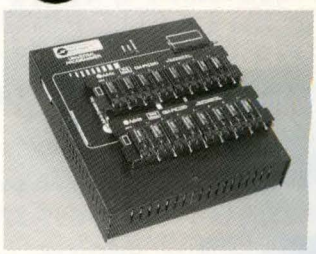
See why Engineers choose TechTools
Visit our website and request your FREE CDROM!

www.tech-tools.com

CONTACTS: (972) 272-9392 Fax: (972) 494-5814 Email: sales@tech-tools.com

Copyright © 1999, TechTools, P.O. Box 462101, Garland, Texas 75046-2101 • ClearView, CVASM16, PICwriter, PICstation, the "Wizard" symbol and TechTools are trademarks of TechTools, P.O. Box 462101 Garland, Texas 75046. • PICmicro is a registered trademark of Microchip Technology, Inc. • All other trademarks are trademarks or registered trademarks of their respective company.

Advin



Production and Engineering Programmers
Extensive support for
Microchip PICs and others,
Wide variety of package types supported.
Thousands of happy customers worldwide are living proof of our product quality.
Please contact us and find out why Advin is your best choice.
– Accept trade-ins of out-dated Data I/O models –
www.advin.com
1-888-GO-ADVIN 408-243-7000

SB-56K Multi-DSP Emulator



Support for the Motorola DSPs:
DSP560xx, DSP563xx, DSP566xx, DSP568xx

SB-56K supports any combination and any count (up to 255) of the devices from the above families. With its accurate counter allowing to measure code execution (benchmarking), small size (1"x2.5"x4"), high speed RS-232 interface, the SB-56K can provide independent support for multiple devices with option to access each device on the target board from different workstations connected through LAN, WAN or Internet.

DOMAIN TECHNOLOGIES, INC.
1700 Alma Dr.,#495,Plano, TX 75075
Tel.:(972) 578-1121 / Fax: (972) 578-1086
info@domaintec.com
www.domaintec.com

GALEP-III
Pocket Multiprogrammer

This size fits all!



◆ Programs 8-bit and 16-bit EPROMs, EEPROMs, Zero Power RAMs, Flash, serial EEPROMs ◆ GAL, PALCE, ATF ◆ 87xxx, 89xxx, PIC12/16/17Cxx ◆ All DIL devices **without adaptor** ◆ **Lightning fast** parallel data transfer (e.g. 27C512 read/compare 2 sec!) ◆ Power supply independent due to **rechargeable battery** ◆ Uses PC printer port ◆ Hex, JEDEC, and binary file formats ◆ Hex and fusemap buffer editor ◆ Split & shuffle for 8-bit, 16-bit and 32-bit targets ◆ Runs under Win3.1, 95, 98, **NT** ◆ 'Remote control' by DDE scripts ◆ Designed for the future due to flexible pin driver technology - new devices will be added every month ◆ Device list, demo software and **lifetime free updates** from our website www.conitec.com!

GALEP-III Set with cable, battery, recharger... **$333.00**
PLCC Adaptor for 8-bit EPROMs / 16-bit EPROMs / GALs .... **each $149.00**

CONITEC 1951 4th Ave, Ste 301 • San Diego, CA 92101 Tel: (619) 702-4420 • http://www.conitec.com

The Debugging Company

Real-Time Debugging

CPU32 ColdFire 68HC12
68302 68020 68306

call toll free 888-88-DEBUG
info@noral.com
www.noral.com

WE SUPPORT YOU FROM DEVELOPMENT...

LV48 Series



- Supports over 5500 devices
- Low voltage to 1.8V
- Windows 95/98/NT software
- EPMaster S495
- Micromaster S995

...TO PRODUCTION



Matrix

- Reliable, high speed production programmers
- 28F800 9 sec program
- Link up to 48 sites with no loss of speed

ICE Technology Inc.
430 Peninsula Ave, Suite 4
San Mateo, CA 94401
iceusa@icetech.com
www.icetech.com

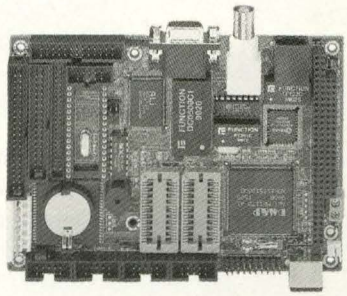
1 888 ICE 2305
1 888 423 2305
Fax: 1 650 375 8666
outside US/Canada call:
+1 650 375 0427
credit cards accepted

A Real x86 PC Based Embedded Solution

It's a System on a Chip

embedded@nucleus1.com

- Wide ranges of 386 and 5x86 available
- Low power consumption and extended temperature ranges
- Guaranteed low cost and long-term supply
- ISA bus, PC/104, 3.5" or custom sizes available
- Custom design or De-population available



Toll Free 1(800) 683-7335

www.nucleus1.com

NUCLEUS Electronic Corp.
Tel: (909) 468-5700 Fax: (909) 468-5704

w w w . n c i - u s a . c o m

The fastest, easiest, most affordable way to test digital designs.

- 72 Channels @ 250MHz
- 36 Channels @ 500MHz
- Up to 2 Meg/Channel
- Easy-to-use Windows Interface



NEW! Portable Logic Analyzer



GoLogic™
72 Channels $3,995
36 Channels $2,995

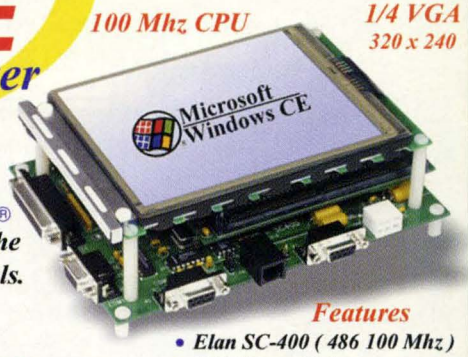
NCI Logic Analyzers

6438 University Drive • Huntsville, AL 35806
phone: 256.837.6667 • fax: 256.837.5221

Windows® CE
Windows & Microsoft Are Registered Trade Mark Of Microsoft Corp.

Embedded Computer

Touch Screen 6"
100 Mhz CPU
1/4 VGA 320 x 240



FREE! *Visual Basic*
Visual C++
Embedded Tools Microsoft®

100% *Ready to program using the latest Microsoft 32-Bit Embedded tools. Built-in Remote Debug and Touch Screen makes this complete package easy to use.* *For More Information and Pricing Visit Our Web Site.*

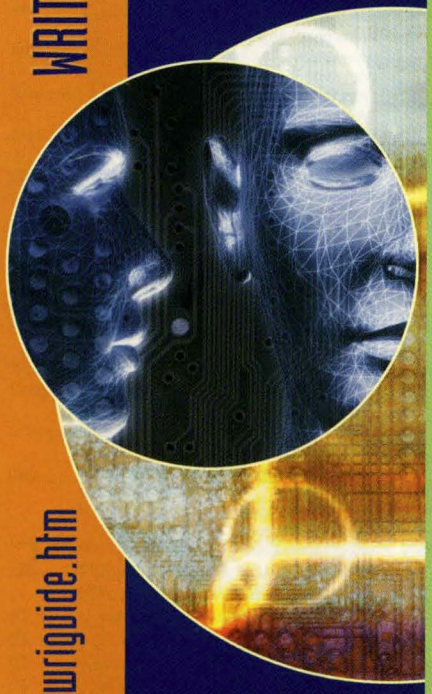
Features

- *Elan SC-400 (486 100 Mhz)*
- *Solid State Disk*
- *Serial Ports (3)*
- *Printer Port*
- *Digital I/O (16)*
- *Analog Inputs (6), 12 Bit*
- *PCMCIA Card Interface*

RLC *www.RLC.com*
Enterprises, Inc.
Toll Free 1-888-RLC-TECH

WRITE FOR EMBEDDED SYSTEMS PROGRAMMING

gain fortune and fame



www.embedded.com/wriguide.htm

contact us
mbarr@cmp.com

SIMULATOR DEBUGGERS

CHIPVIEW®

8051 and derivatives
Intel 80196 / 80296

This remarkable tool utilizes ActiveX technology to provide fast, cycle-accurate simulation of CPU, interrupts, and on-chip peripherals.

Add your own ActiveX controls to simulate off-chip peripherals, ASICs, and even entire systems!

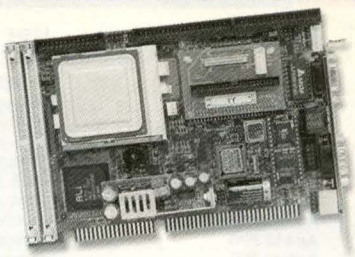
Attach to HTML ActiveX controls to mock-up and test your interface!

Interface to your HDL simulator for hardware/software co-verification!

Download a 30-Day Trial Today!
www.chiptools.com

CHIPTOOLS®
905-274-6244 FAX:905-891-2715

Pentium Work Horse



- Ethernet - 10/100 mps
- Up to 350 MHz
- LCD/VGA support
- Compact - 7" x 4.8"
- 386-Pentium II SBCs
- 3-14 Slot Chassis
- Solid State Disks

mcsi "The Embedded PC Specialists"

888-860-9575 ISO 9001

www.mcsi1.com • email: mcsi@mcsi1.com

Embedded Systems Programming Marketplace

East Coast Sales
Damon Graff
781.839.1285

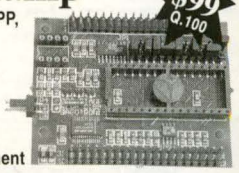
California Sales
Andres Diaz
415.278.5274

West Coast Sales
Sam Louis
415.278.5223

EMBEDDED INTERNET MADE EASY! with the
DOS Stamp™

From $99 Q.100

- Free TCP/IP, PPP, Mini-server
- Easy to Use
- Low Cost
- Low Power
- Tiny Size
- High Speed
- DOS Environment



Easy Software Development: Use your C/C++ or Basic compiler to produce DOS EXE. Download EXE to flash disk via serial port. EXE runs on power up.

Standard Features: BIOS & DOS-ROM, 128K flash disk, 512K SRAM, 40 MHz AM188ES CPU, 16 digital I/O (opto rack interface), 2 RS-232, 2 timer/counters, simple bus interface, real-time clock with timed power-up

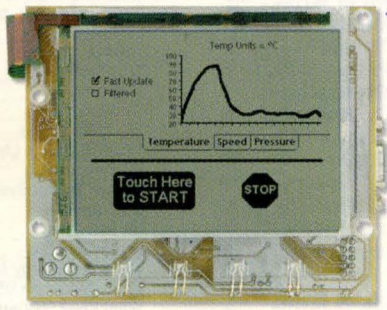
Options: 8-ch 12-bit ADC, flash disks up to 288 MB, 1 RS-485

Tiny Size, Low Power: 2"x2.6", 5V @ 200 mA at full speed.

Visit http://www.bagotronix.com for info, prices, and FAQs

BAGOTRONIX
2900-1 Crescent Drive
Tallahassee, FL 32308
850-942-7905 phone & fax

Now... GUI and LCD Control in a Single Package!



Visit our website at:
AmuletTechnologies.com
or phone us today at (408) 244-0363

- ***1/4 VGA, 3.8", Monochrome Display*** – with ultra-bright backlight and fully-integrated analog touch panel
- ***Dedicated GUI Controller*** – manages the GUI, interacts with the user, and controls the LCD
- ***Processor Independent*** – easily interfaces to most microprocessors (8/16/32-bit and even DSPs)
- ***HTML-Based GUI*** – converts from HTML, JPEG, and GIF into small, quickly-executable Amulet µHTML™ pages
- ***Replaces Traditional GUI Library*** – No library porting, complex GUI programming, or RTOS required
- ***Standard RS232 Interface*** – Up to 115.2 Kbps, cable included
- ***64K-Bytes of Onboard Flash Memory*** – For storing hundreds of Amulet µHTML pages that you create

Amulet Technologies
GUI Engines For Embedded Systems

See Us at Embedded Systems Conference 2000, Booth 8053

©2000 µHTML is a Trademark of Amulet Technologies. U.S. Patents Pending.

ADVERTISER INDEX

Advertiser	URL	Page
3DO	www.3do.com	205
Abraxas Software	www.abxsoft.com	196
Accelent Systems	www.accelent.com	75
Accelerated Technology	www.atinucleus.com	173
Acrosser Technology Co.	www.acrosser.com	211
Advanced Transdata Corp.	www.adv-transdata.com	212
Advantech	www.advantec.com/epc	180
Advin Systems	www.advin.com	213
Agilent PPG	www.agilent.com	16
Agilent PPG	www.agilent.com	85
Agilent PPG	www.agilent.com	157
Alcatel	www.alcatel.com	210
Allant Software Corp.	www.allant.com	123
American Arium	www.arium.com	C3
American Raisonance	www.americanraisonance.com	82
American Raisonance	www.americanraisonance.com	86
Amulet Technologies	www.amulettechnologies.com	215
Analog Devices	www.analog.com	131
Applied Data Systems	www.applieddata.net	184
Applied Signal Technology	www.appsig.com	208
ARC Cores Ltd.	www.arccores.com	53
Archimedes Software	www.archimedessoftware.com	190
Arcom Control Systems	www.arcomcontrols.com	70
Artisan Software Tools	www.artisansw.com	121
Ashling	www.ashling.com/inftools	44
Avocet Systems Inc.	www.avocetsystems.com	44
Avocet Systems Inc.	www.avocetsystems.com	C4
Axiom Technology	www.axiomtek.com	39
Bagotronix Inc.	www.bagotronix.com	215
Blackhawk	www.blackhawk-dsp.com	137
Blunk Microsystems	www.blunkmicro.com	211
Boeing	www.boeing.com/employment	206
Bsquare Corporation	www.bsquare.com	108
Byte Craft Limited	www.bytecraft.com	150
Cadence	www.cadence.com/systems	73
Cadence	www.cadence.com/systems	10
CAD-UL	www.cadul.com	140
Ceibo	www.ceibo.com	93
ChipTools	www.chiptools.com	215
CMX Systems, Inc.	cmx.com	192
Cogent Computer Systems	www.cogcomp.com	170
Conitec	www.conitec.com	214
Cosmic Software	www.cosmic-software.com	118
Dinkumware Ltd.	www.dinkumware.com	190
Domain Technologies	www.domaintec.com	213
dSPACE	www.dspaceinc.com	133
EBS, Inc.	www.ertfs.com	201
EBSnet	www.ebsnetinc.com	105
eDevice	www.edevice.com	25
EDS	www.eds.com/careers	208
Electronic Engineering Tools	www.eetools.com	212
EMAC, Inc.	www.emacinc.com	221
Embedded Power	www.embeddedpower.com	185
Embedded Power	www.embeddedpower.com	63
Embedded Systems Academy	www.esacademy.com	42
Embeddedtechnology.com	www.embeddedtechnology.com	165
Embedded University	www.embeddedu.com	43
Emware	www.emware.com	111
Enea OSE Systems, Inc.	www.enea.com	91
esmertec ag	www.esmertec.com	68
Espial Group	www.espial.com	71
Express Logic	www.ghs.com/armsolutions	178
Express Logic	www.expresslogic.com	171
General Software	www.gensw.com	182
Grammar Engine Inc.	www.gei.com	170
Grammar Engine Inc.	www.gei.com	211
Green Hills Software Inc.	www.ghs.com	14, 15
Green Hills Software Inc.	www.ghs.com/armsolutions	178
Green Hills Software Inc.	www.ghs.com	6
Green Hills Software, Inc.	www.ghs.com	42
Hitachi Semiconductor	www.superh.com	87
HI-TECH Software	www.htsoft.com/pic	213
HI-TECH Software	www.htsoft.com	195
Hitex Development Tools	www.hitex.com	212
Hitex Development Tools	www.hitex.com	151
Hitex Development Tools	www.hitex.com	43
Hiware	www.hiware.com	104
Hughes Network Systems	www.hns.com	204
Hyperstone Electronics	www.hyperstone.com	13
IAR Systems	www.iar.com	183
Ice Technology, Inc.	www.icetech.com	214
I-Logix	www.ilogix.com	21
I-Logix	www.ilogix.com	127
Infineon Space Program	www.spacetools.com	43
Infineon Technologies	www.infineon.com	26, 27
Intel Ceg	www.intel.com	29
InterNiche Technologies	www.iniche.com	107
Int'l Microsystems, Inc.	www.cardduper.com	192
Introl	www.introl.com	212
iSYSTEM USA, Llc	www.isystem.com	136

Advertiser	URL	Page
JK Microsystems, Inc.	www.jkmicro.com	213
JK Microsystems, Inc.	www.jkmicro.com	213
Kadak Products	www.kadak.com	177
Keil Software	www.keil.com	142
Keil Software	www.keil.com	42
Kidde Fenwal	www.kidde-fenwal.com	203
Lantronix	www.lantronix.com	115
Lauterbach Inc.	www.lauterbach.com	57
Lauterbach Inc.	www.lauterbach.com	42
Lineo	www.lineo.com	83
Lineo	www.lineo.com	196
LTX Corp.	www.ltx.com	207
LynuxWorks, Inc.	www.lynuxworks.com	198
Megatel	www.megatel.com	194
Mentor Graphics	www.mentor.com	163
Metalink Corp.	www.metaice.com	212
MetaWare, Inc.	www.metaware.com	191
Metrowerks	www.metrowerks.com	188
Micro Catalog	www.microcatalog.com	44
Micro Digital	www.smxinfo.com	175
Micro/sys	www.embeddedsys.com	139
Microchip	www.microchip.com	67
Microcomputer Specialists	www.mcsi1.com	215
Microsoft Windows CE	www.microsoft.com/windowsce/embedded	30, 31
Microsoft	www.microsoft.com	193
Microtek International	www.microtekintl.com	4
Microware Systems Corp.	www.microware.com	169
MontaVista Software, Inc.	www.mvista.com	8, 9
Morgan Kaufman	www.mkp.com	141
Motorola Semiconductor Hp	www.digitaldna.motorola.com	59
Motorola Semiconductor Hp	www.digitaldna.motorola.com	18, 19
National Semiconductor	www.national.com	153
Nat'l Engineering Search	www.nesnet.com	204
NCI	www.nci-usa.com	214
Needhams Electronics	www.needhams.com	211
Net Media, Inc.	www.netmedia.com	C2
Nohau Corp.	www.nohau.com	211
Nohau Corp.	www.nohau.com	34, 35
Nohau Corp.	www.nohau.com	44
Noral Micrologics Limited	www.noral.com	214
Nucleus Electronics Corp.	www.nucleus1.com	214
Pacific Softworks	www.pacsoft.com	103
Panasonic	www.panasonic-wireless.com	207
Paradigm	www.devtools.com	174
ParaSoft	www.parasoft.com	155
Phar Lap Software Inc.	www.pharlap.com	100
Phoenix Technologies Ltd.	www.pheonix.com	61
Phytec	www.phytec.com	44
Phyton, Inc.	www.phyton.com	212
PointBase Inc.	www.pointbase.com	77
Precise Software	www.psti.com	187
ProSyst USA, Inc.	www.prosyst.com	99
Rabbit Semiconductor	www.rabbitsemiconductor.com	74
Radisys Corp.	www.radisys.com/ss7	145
Rational Software	www.rational.com	33
Rational Software	www.rational.com	65
Reasoning	www.reasoning.com	119
RLC Enterprises	www.rlc.com	214
Scientific Placement	www.scientific.com	204
Signum Systems	www.signum.com	211
Sophia Systems	www.sophia.com	149
Starbase Corp.	www.starbase.com/embedded	159
Swell Software	www.swellsoftware.com	223
Synopsys	www.synopsys.com	55
Tasking	www.tasking.com	181
Tech Tools	www.tech-tools.com	213
Tektronix	www.tektronix.com/tla600	197
Tern Inc.	www.tern.com	211
The Math Works	www.mathworks.com	125
Tilcon Software Ltd.	www.tilcon.com	122
TQ Components	www.tq-group.com	44
Trillium Digital Systems	www.trillium.com	112
US Software	www.ussw.com	94
US Software	www.ussw.com	44
VenturCom Inc.	www.vci.com	37
Vesta Technology Inc.	www.sbc2000.com	212
White Mountain DSP	www.wmdsp.com	135
Wind River Systems	embeddedtools.windriver.com	1
Wind River Systems	www.wrs.com	22, 23
Working Engines Inc.	www.workingengines.com	204
Waferscale	www.waferscale.com	154
Waferscale	www.waferscale.com	43
Xilinx	www.xilinx.com	143
Yokogawa Digital Computer	www.ydcusa.com	211
ZF Linux Devices	www.zflinux.com	81
Zilog Inc.	www.zilog.com	49
Zucotto Systems, Inc.	www.zucotto.com	110
Z-World	www.zworld.com	211

FREE SUBSCRIPTION REQUEST FORM

Embedded Systems PROGRAMMING

P.O. BOX 3404 • NORTHBROOK, IL 60065-9468
www.embedded.com

PLEASE ANSWER ALL QUESTIONS
(FRONT AND BACK) THEN SIGN AND DATE THE CARD.
Incomplete cards cannot be processed or acknowledged.

1. Do you wish to receive/continue to receive EMBEDDED SYSTEMS PROGRAMMING?

☐ **YES** ☐ No

Signature (required) X_____

Name (please print) _____

Date _____

Job Title _____

Company Name _____

Address _____ ☐ Work ☐ Home

Mail Stop_____

City _____ State _____ Zip _____

Phone ( ) _____

Fax: ( ) _____

E-Mail:_____

If you do not wish to receive future communications from *CMP/Miller Freeman, Inc.* via e-mail please check here ☐

If you would prefer delivery to your home, please complete home address.
Company name and address are still required to qualify.

Address _____

City _____ State _____ Zip _____

2. Check all of the following that you are involved in doing or managing at your company or at those companies to whom you consult. (Please check all that apply)

01 ☐ Architecture Selection/ Specification
02 ☐ Writing Software for Embedded Systems
03 ☐ Writing/Embedding Real-Time Operating System/Kernel
04 ☐ Debugging Software
05 ☐ Debugging Hardware
06 ☐ Hardware/Software Integration
07 ☐ Hardware/Software Co-Design
08 ☐ Device Programming
09 ☐ Project Management
11 ☐ Software Design/Analysis
12 ☐ Prototype Testing
13 ☐ Designing Hardware for Embedded Systems
14 ☐ Board Layout/Design
17 ☐ Hardware/Software Co-Verification
18 ☐ Hardware/Software Partitioning
19 ☐ Software Testing
20 ☐ SOC (System-on-Chip) Design
21 ☐ Internet Appliance Design
15 ☐ Other (please specify) _____
16 ☐ I'm not involved in Embedded Development in any way.

3. What is your principal job function? (Please check only one)

Engineering/Computer Management
01 ☐ Executive Management (i.e., President, VP, Owner, Chairman, Partner)
02 ☐ Engineering Management (i.e., Technical Director, Chief Engineer, Department Manager, Group Manager)
03 ☐ Software Engineering/ Programming/Development Management (i.e., Sr Software Engineer, Principal Software Programmer)
04 ☐ Systems Engineering/ Development Management (i.e., Sr Design, Hardware, or Test Engineer)
05 ☐ Other Management (please specify)_____

Engineering/Programming Personnel
06 ☐ Software Engineering/ Programming/Development
07 ☐ Systems Engineering/ Development (i.e., Design, Hardware, or Test Engineer)
08 ☐ Engineering Support (Technician, Programming Staff)
09 ☐ Scientific/R&D/Education
10 ☐ Other staff (please specify)_____

4. Please check all products which you specify, recommend, authorize, or purchase. (Please check all that apply)

ICs and Semiconductors
01 ☐ Microcontrollers/ Microprocessors
02 ☐ 4/8-bit µC/µP
03 ☐ 16-bit µC/µP
04 ☐ 32-bit µC/µP
05 ☐ 64-bit µC/µP
06 ☐ X-86/Pentium
07 ☐ Digital Signal Processors
08 ☐ EPROM / EEPROM
09 ☐ Flash
10 ☐ DRAM/SRAM
11 ☐ Communication ICs
12 ☐ Media Processors
13 ☐ CPLDs/FPGAs
14 ☐ System-on-Chip (SOC)
15 ☐ MCU Peripheral Chips
16 ☐ Hardware IP/Cores

System Boards
20 ☐ Single Board Computers
21 ☐ VME Boards
22 ☐ Embedded PCs
23 ☐ PCI Boards
24 ☐ cPCI Boards
25 ☐ DSP Boards

Computer Systems
29 ☐ PCs
30 ☐ NT Workstations
31 ☐ Unix Workstations
32 ☐ Linux Workstations

Software
36 ☐ Real-Time Operating Systems/ Kernels
37 ☐ Compilers/Cross Compilers
38 ☐ Assemblers/ Cross Assemblers
39 ☐ Software Debuggers
40 ☐ Object-Oriented Design Tools
41 ☐ Simulators/Modeling Tools
42 ☐ Version/Change Control Software
43 ☐ Communications Software/ Protocols
44 ☐ ROMable DOS Tools
45 ☐ Device Driver Tools
46 ☐ Embedded Databases
47 ☐ Embedded Web/Internet Tools
48 ☐ GUI Development Tools
49 ☐ Open Source Tools
50 ☐ Java Tools
51 ☐ Software Testing Tools
52 ☐ Integrated Development Environments (IDEs)
53 ☐ Windows CE Tools

Design Tools/Test Equipment
56 ☐ In-Circuit Emulators
57 ☐ Logic Analyzers
58 ☐ Oscilloscopes
59 ☐ Data Acquisition Equipment
60 ☐ Device Programmers
61 ☐ Hardware/Software Co-Design Tools
62 ☐ Hardware/Software Co-Verification Tools
63 ☐ None of the above

5. What is the primary end product or service performed at your location? (Please check only one)

01 ☐ Computers/Peripherals/Office Automation
02 ☐ Communications/Telecommunications/Networking
03 ☐ Consumer Electronics/Entertainment/Multimedia
04 ☐ Automotive Transportation Systems and Equipment
05 ☐ Government/Military Electronics
06 ☐ Aerospace/Space Electronics
07 ☐ Industrial Controls
08 ☐ Electronic Instruments/ATE/Design & Test Equipment
09 ☐ Medical Electronic Equipment
10 ☐ Other: (please specify)_____

6. What is your engineering/development responsibility? (Please check only one)

01 ☐ I manage an engineering or software development department
02 ☐ I manage a project team
03 ☐ I manage a project
04 ☐ I am a member of a project team
05 ☐ Other (please specify)_____

MORE QUESTIONS ON BACK →

www.espmag.com

B009

CMP

7. Do you specify or buy through distributors?

01 ☐ Yes
02 ☐ No

8. How many employees are there at your company?

01 ☐ 1000 or more
02 ☐ 500-999
03 ☐ 250-499
04 ☐ 100-249
05 ☐ 50-99
06 ☐ 1-49

9. Please check the publications below that you receive personally addressed to you by mail. (Please check all that apply)

02 ☐ EDN
03 ☐ EE Times
04 ☐ Electronic Design
06 ☐ Penton's Embedded Systems Development
05 ☐ None of the above

10. How many other people read your copy of EMBEDDED SYSTEMS PROGRAMMING?

01 ☐ 1
02 ☐ 2
03 ☐ 3
04 ☐ 4
05 ☐ 5
06 ☐ 6
07 ☐ 7
08 ☐ None

11. Please list the names of others at your location who may be interested in a free subscription to EMBEDDED SYSTEMS PROGRAMMING

Name Title

Company name

Company address

City State Zip

PUBLISHER RESERVES THE RIGHT TO SERVE ONLY THOSE INDIVIDUALS WHO QUALIFY.

www.espmag.com

FOR FASTER SERVICE FAX BOTH SIDES TO: (847) 291-4816

FOLD FOR MAILING

NORTHBROOK IL 60065-9468
PO BOX 3404
Embedded Systems PROGRAMMING
CMP

POSTAGE WILL BE PAID BY ADDRESSEE

BUSINESS REPLY MAIL
FIRST-CLASS MAIL PERMIT NO. 2253 NORTHBROOK IL

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

PLEASE TAPE ALONG THIS EDGE. NO STAPLES.

ICSPAT

ICSPAT

International Conference on Signal Processing Applications & Technology

October 16-19, 2000 • Adam's Mark Hotel • Dallas, TX, USA

Meeting of the Minds



attention signal processing engineers and programmers!

Don't miss this unique opportunity to enhance your signal processing knowledge and design skills. Take advantage of the many opportunities available this fall to enhance and update your signal processing knowledge at **ICSPAT**, the premier international forum for emerging signal processing research and development.

At **ICSPAT**, you can review cutting-edge research from around the world. Hundreds of white papers will be presented in more than 30 application areas including: Audio, Biomedical, Communications, Industrial Applications, Image Processing, Multimedia, Robotics, Telephony and more!

What you'll get at icspat:

- Access to white papers in more than 30 application areas in the ICSPAT program
- A window on the future of signal processing and the applications in which it is used
- High quality technical knowledge and skill enhancement
- Expanded lectures offering more in-depth, enhanced coverage of topics
- Tutorial and plenary sessions as well as free product training representatives
- The official proceedings on CD-ROM

ESP

For a complete listing of ICSPAT lecture and poster abstracts, plus a comprehensive guide to plenary sessions, special events and industry activities, go to www.dspworld.com.

www.dspworld.com

4 easy ways to register

Web:
www.dspworld.com
Phone:
(800) 789-2223
Fax:
(415) 278-5390
Mail:
ICSPAT
C/o Corporate Registration
P.O. Box 612768
Dallas, TX 75261-2768

Media Co-Sponsors:

CMP EETimes

CMP Communication Systems Design

CMP EmbeddedSystems Programming

Gold Sponsors:

Motorola

StarCore

Silver Sponsors:

VxTel

Chameleon Systems, Inc.

Pentek

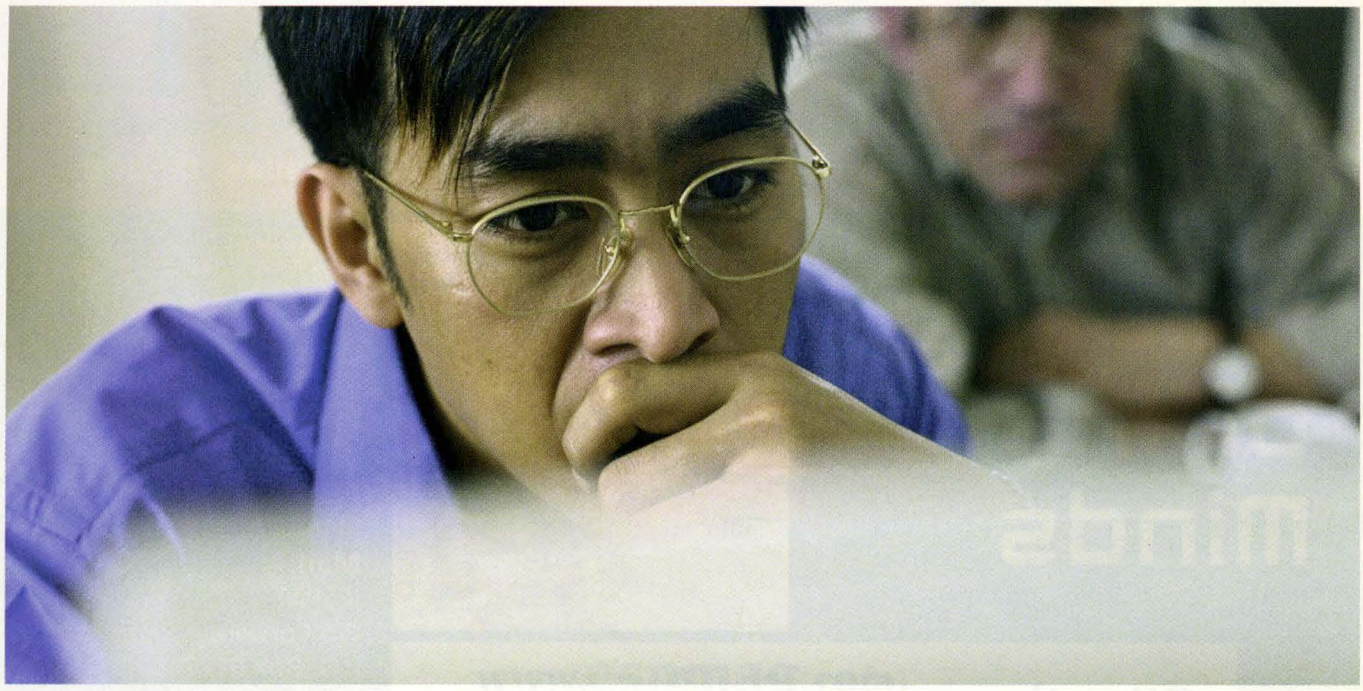
Analog Devices

BOPS

DSE

Precise Software Technologies Inc.

CMP



We interrupt this frustrating web search for EDA tools and design information to bring you the following announcement...

Now you have one place on the Internet where all the latest design automation news and product information is just a click or two away. It's www.eedesign.com. And it draws on the resources of the leading information providers, analysts and manufacturers in the business.

At EEdesign, you'll have instant access to the latest news coverage from *EE Times*' Richard Goering, Michael Santarini and Peter Clarke. You'll get the inside story on solving real-world design problems from the peer experts in *Integrated System Design*.

And speaking of engineering experts, you'll also have an exclusive direct connection to John Cooley's DeepChip.com site, home of the fiercely independent ESNUG email newsletter. That's something no other EDA portal site can offer!

Plus we're featuring Toolwire, creator of the Design Chain Management Network, as our partner for web-based electronics design.

You'll also have access to an industry-wide interactive design tools product directory. What's more, in coming months we'll add seminars, bulletin boards, polls and a lot more. All under the direction of Tets Maniwa, regarded by many as the leading EDA advocate in the world today.

So if you're looking for EDA tools and information and want it *right now*, log on to www.eedesign.com. It just might be the most productive design decision you've ever made!

EDTN NETWORK PARTNERS

EBN, EE Times, SBN, Embedded.com, ISD Magazine, Power Designers, Home Toys, PCN Alert, ChipCenter, Communications System Design, Design & Reuse, Circuits Assembly, HDI, PC Fab, Printed Circuit Design, Toolwire, DeepChip, EDA Connect, WebPRN (Web Product Realization Network)

www.edtn.com

EDTN network

It's all right here.

www.eedesign.com



Jack G. Ganssle

Repair or Replace?

An engineer's adventures in paradise: Jack shares what he's learned from watching engineers make do in the Bahamas.

Scott and I trudged up the hill, our pace slow, not so much due to the incline, but more because of the tropical sun. It beat down relentlessly, turning the quick walk of type-A people into an unhurried stroll.

Our goal was the 200-foot radio tower visible from every corner of Staniel Cay (pronounced "Key"), a small island in the Bahamas' Exuma chain. I'd been searching for an Internet connection for weeks. Desperation reached a peak in Warderick Wells (no phone service at all; only one family lives in the small group of islands) before I made friends with folks on a sailboat who transferred e-mail via single sideband radio. Though it was impossible to suck down my mail, I managed to send off a critical message that went through from my new acquaintances' computer, into their SSB radio, over the high frequency bands to a ham station ashore, that routed the packets into the 'Net. No drug addict has worked harder at getting a fix.

The guidebooks hinted that Staniel Cay was the center of Exuma civilization, offering all sorts of services. That suggested Internet to my too-long-unconnected brain, which, along with its attraction of having the caves used to film the James Bond movie *Thunderball*, brought us here from more remote places.

The tower provided phone service to all 60 inhabitants of the island by means of a microwave link to islands up the chain. Not a single wire connects this remote bit of paradise to the world; even power is generated internally, by the Cay's truck-sized diesel plant.

Most islanders view Batelco (the Bahamas Telephone Company) with derision. In Nassau we learned it's not unusual to wait three to six months for a new phone, and that's in the very capital of the Bahamas. Far more pay phones are broken than operative. Yet I was struck by the difficulty of the company's mission in the out-islands. It's hard to believe that the minimal economy of some of the more remote Cays can support a telephone system, yet Batelco does indeed provide phones to virtually every resident. Here in Staniel the phone book shows 35 phone numbers, all sharing the same first five numbers of their seven digit dialing code.

At the base of the tower we found a hut. "Building" would be far too grand a word for the small concrete structure that houses the island's phone service. It turns out that long distance calls to the U.S.A. cost up to $6 per minute. Knowing that after three weeks I'd have 500 to 1,000 e-mail messages waiting, I decided to pass on my Internet quest until we returned to Nassau a few days hence. But the air conditioning felt so good after the tropical heat that we stayed and chatted up the engineer.

Both Scott and I are embedded folks, our careers in this field starting with the dawn of the microprocessor age. We were fascinated by the system's equipment and the story of how Batelco keeps everything running.

Phone service only came to most of these out islands within the last 20 years. Before that, before microprocessors made central offices small and cheap, electronic inter-island communication was non-existent or consisted of irregularly used radios. In the panic of getting products out the door, of deadlines and hassles, sometimes it's hard to see how our small part in the embedded world benefits people. Though I had nothing to do with the telecom industry, I felt no small pride in being in an industry that had clearly helped these communities. Staniel Cay doesn't even have a doctor, but when the inevitable medical emergency arises, a phone call to Nassau brings help by air in hours.

Most fascinating, though, were the problems associated with keeping the system running. Everything electronic breaks. In the U.S.A. this is an annoyance. We toss out old broken gear and shed only the smallest tear at the cost of the new CD player, cell phone, or whatever. As toy junkies, the tear is more theatrical than real as the failure of a product means we get to buy a newer, fancier, more feature-rich version. Dig out the American Express card, place a

EMBEDDED SYSTEMS CONFERENCE

free pass

to the Product Exhibits

Connect to tomorrow's complete embedded solutions

September 26 – 28, 2000 • San Jose Convention Center • San Jose, CA



The entire embedded industry under one roof!

The world's largest exhibition of embedded products features the ultimate in "one-stop shopping" for engineers, designers, and project managers. Get your hands on all the product options and talk face-to-face with the "factory people" who design your current favorites. Check out www.embedded.com/esc for frequently updated exhibitor lists.

Bring this FREE pass to see:

- Embedded Internet Tools
- Java for Embedded Systems
- Real-Time Operating Systems
- Development Tools
- System-on-a-Chip Technologies
- Emulators
- DSPs
- Simulators

Show Floor Hours

Tuesday, September 26
12:00 pm—8:00 pm

Wednesday, September 27
10:00 am—7:00 pm

Thursday, September 28
10:00 am—4:00 pm

Plus, your FREE registration includes admittance to the Special Events.

Keynote Address
Neil Gershenfeld
"When Things Start to Think"
Tuesday, September 26
6:00 pm – 7:00 pm

Special Guest Lecture
Bill Veghte
"Device Connectivity: A Vision"
Wednesday, September 27
6:00 pm – 7:00 pm

Panel Discussion
"The Open-Source Movement: Boon or Bane for Embedded Systems Developers?"
Monday, September 25
6:00 pm – 7:30 pm

Show Floor Reception
"Streets of San Francisco"
Tuesday, September 26
7:00 pm – 8:00 pm

Attendee Party
"Swinging Innovations"
Wednesday, September 27
7:00 pm – 9:30 pm
Tech Museum of Innovation

esp

www.embedded.com/esc • 800-789-2223

CMP

toll-free call, and the replacement widget arrives on our doorstep before 10:30 a.m. the next day. We're so accustomed to this sort of service that much of American business is planned around quick deliveries. Too often, we substitute FedEx for planning ahead. Need something on Thursday? Wait till Wednesday to start thinking about it.

Here in Staniel Cay there's no FedEx; no UPS; no postal system. The mail boat arrives once every two weeks, more or less, if the weather is good. Occasionally it sinks. Everything comes on the boat. Food, fuel, outboard motors, and mail. Need something before the boat arrives? You'll do without, learn to wait, and live with even more disappointment since seven times out of 10 the thing you order won't be on it anyway. "Sorry mon, it be here real soon now."

I met a fellow stuck in the Cay for two months while he waited on a part for his engine. ("Stuck" is perhaps the wrong word for being forced to spend time in paradise.) He finally went islander and cobbled up a fix using duct tape and bailing wire.

Batelco suffers along with the locals. When parts of the phone system go down, as they must in this difficult tropical environment, they either fix things here or wait, and wait, and wait, for spare parts. Though individual outages may be annoying, when the entire system goes down the island becomes telephonically isolated.

One corner of the the tower's equipment room is stacked with spare boards. Another has a much higher pile of defective boards. The engineer uses a rather comprehensive set of built-in diagnostics to isolate failures to a board, and then swaps in a replacement card. The microprocessors make diagnostics a critically necessary part of the system, and unlike too many self-test routines, which seem more aimed at filling a niche on the datasheet ("includes full diagnostics!"), these actually work.

He then calls Batelco headquarters in Nassau and orders a replacement board, ideally keeping at least one good spare on hand at all times. Unfortunately the system collapses in typical Bahamas fashion at this point. Headquarters accepts the order and promises a new part, but weeks go by before the mail boat comes, and all too often months pass while paperwork creeps from desk to desk.

The perversity of nature ensures the next failure will be on the same type of board, for which there's no longer a spare immediately at hand. The engineer earns his pay by cobbling up a solution.

The perversity of nature ensures the next failure will be on the same type of board, for which there's no longer a spare immediately at hand. The engineer earns his pay by cobbling up a solution.

As an American consumer, I'm always struck by the behavior of island economies. Here, as in so many other places, everything is imported. Trash is expensive. Where do old, used-up, things go? Small islands have little room for disposal, so tossing out defective gear is problematic. So cars have long lives. No one replaces alternators or starter motors; they're always rebuilt. Shops weld and repair parts that in America we'd just replace. Trash piles are plundered for bits of treasure, that small metallic thing that can fix the car, the generator, or whatever. When the decrepit conch-fishing boat sinks at anchor, salvagers raise the rotten wreck, rebuild the engine, and squeeze ever more life out of it. It might sink a couple of times a year, but



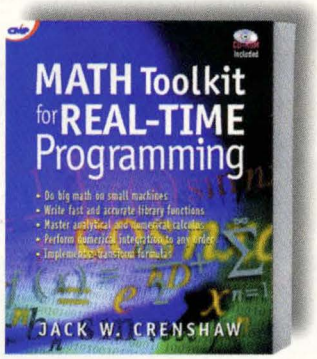
www.cmpbooks.com

CMP books!

Math Toolkit for Real-Time Programming

You get a solid course in applied math from Jack Crenshaw, the renowned columnist of *Embedded Systems Programming* magazine — and a versatile set of algorithms to use in your own projects!

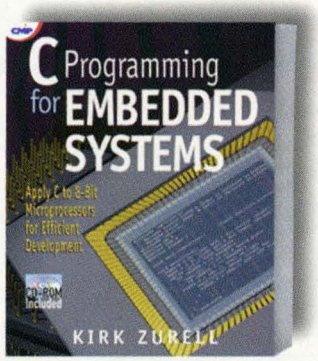
ISBN 1-929629-09-5
$49.95



C Programming for Embedded Systems

Get up and running fast with this clear presentation of 8-bit MCU development principles and demonstration project, complete with schematics, a parts list, and sample code. Learn the details of device-specific hardware development — from preliminary software design to implementation of a working consumer product.

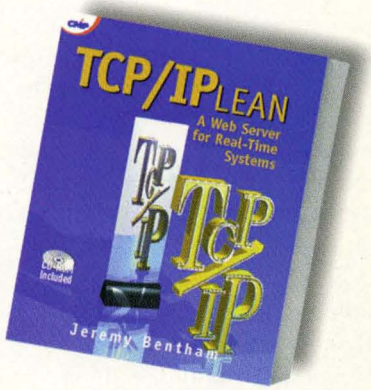
ISBN 1-929629-04-4
$29.95



TCP/IP Lean

TCP/IP demystified! You get a clear, hands-on explanation of TCP/IP programming and source code for a complete, compact TCP/IP stack that can be used to create a web server for your real-time systems.

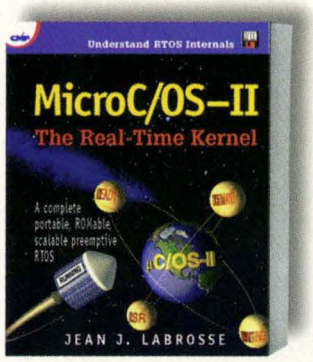
ISBN 1-929629-11-7
$49.95



MicroC/OS-II

Learn the inner workings of an RTOS! This latest release of MicroC/OS has been completely revised and includes many new features. It is a completely portable, ROMable, preemptive real-time kernel. Complete code is included for use in your own applications.

ISBN 0-87930-543-6
$69.95



Find CMP books in your local bookstore

call : 800-500-6875
e-mail : cmp@rushorder.com

ESP900

raising it is so much more effective than getting a replacement.

Technology that can't be repaired is a problem. Years ago aid agencies realized that large-scale projects in third world countries often resulted in large scale defects; from this sprang the concept of *appropriate technology*, that which is optimized for use in a particular culture. So a series of small, repairable water pumps makes more sense than a mega-scale water treatment plant in parts of Africa. Mechanical ignitions, which can be serviced with bailing wire, replace high-efficiency electronic spark systems.

There must be some critical size where the disposable society we've created "works." One where delivery and disposal are quick and easy. Where the economy is strong enough to support casual replacement. Though it pains me to be part of the disposable economy, it's the way America works. Technology changes so fast that yesterday's miracle is tomorrow's junk.

I once had a Tektronix 545 oscilloscope, a 100-pound vacuum tube beast that refused to die. The thing must have been 30 years old, worked like a champ, but wasn't fast enough for modern electronics. It had to go, but how could I toss such a huge, functional thing in the dumpster? Yet in the end it was obsolete, a relic with no value. I finally found a teenager who claimed to have a need for it, but figured that soon it, too, would be on the junk heap of outmoded technology.

Here in the Bahamas I suspect that my old scope would be a prize, no doubt to be repaired and maintained for decades to come. Though the shop that repairs computers might not find it useful, it would find it's way naturally through the junk chain to the shop that, perhaps, repairs radios where a 1MHz bandwidth is more than enough. Instead of just chucking the thing—because disposal itself is so expensive here—this Bahaman system I can't figure out seems to ensure that it'll find it's natural niche in the obsolete technology environment.

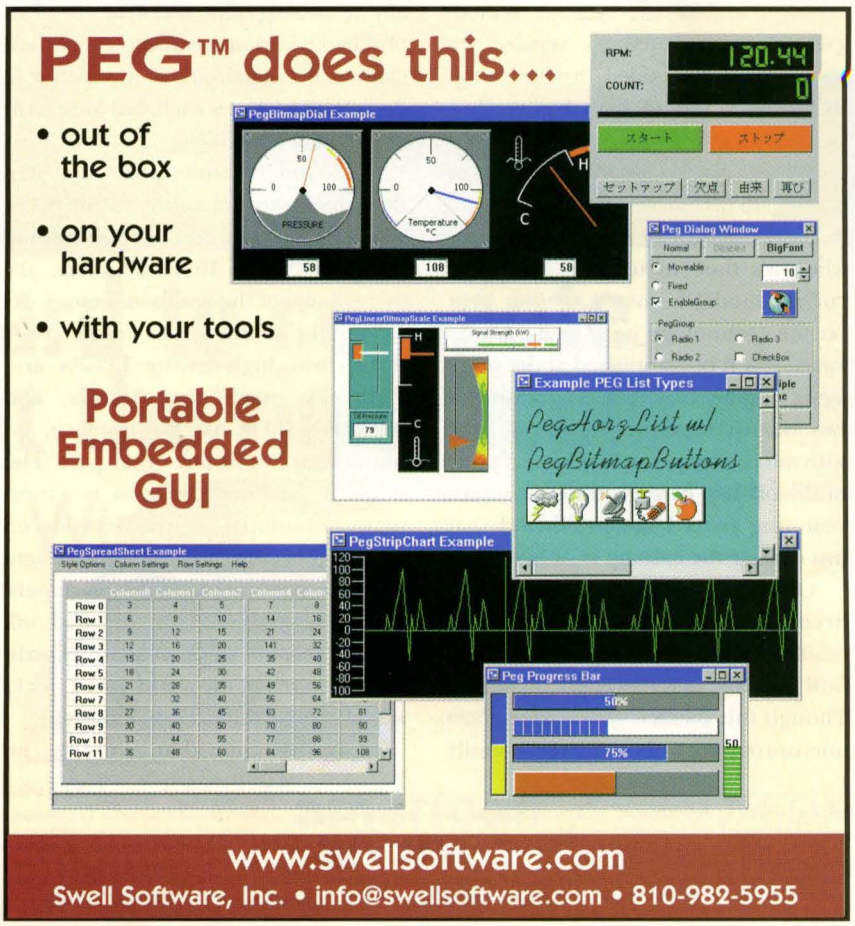
So the Batelco engineer repairs the broken board, or at least makes every attempt to fix the thing. All without much of a scope, logic analyzer, or development system (he has, in fact, never heard of one). His tools come from Sears. Wire cutters, needle nose pliers, clunky soldering gear. There's no surface mount station, not even an anti-static mat. Yet he's the master of a microprocessor-based telecom switching station.

I peered deeper, trying to understand better. The equipment is, by U.S. standards, old but robust. It uses 8085 microprocessors, a mid '70s-era CPU that's still designed into some products. The 8085 is a 40-pin DIP device, an 8-bitter that runs at a few megahertz.

Each circuit board is beautifully designed, with wider-than-usual tracks (perhaps .020), power and ground planes, and two circuit layers on the outside where they're accessible. Parts aren't crowded together like commuters in New York, but spread leisurely around. Each device is a DIP, with pins on 0.1 centers. Many are inserted in high-quality machined pin sockets, yielding a somewhat expensive but totally reliable connection.

If you're not a hardware person, or are one who came of age in the last decade, the previous paragraph says that parts are accessible, they're replaceable, and pins and connections are large enough that even the bluntest fingers can easily slip a probe on any desired node.

The sockets make part substitution hassle-free. More, they allow the engineer to bend pins up and disconnect them at will to run experiments, without soldering. It gives him options, allows him to exercise his skills at debugging without requiring a lab full of highly skilled production people.



It's easy to succumb to an arrogance of the office, where we lose touch with real customer needs, when we listen on the phone with but a single ear to a repair person in the field who is upset, while we make faces and wish only that the turkey would shut up.

The Batelco employees scattered on these remote islands remove parts from their stock of other failed boards, swapping them as necessary to make things work. That pile of bad boards that no one could repair is a resource full of potentially good components. They live with essentially no stock of known good parts, since the head office believes that one spare board is enough; in their wisdom, management figures new boards are just a mail boat away. Life in the field gives the lie to such easy dreams, so these engineers develop strategies, like stockpiling failed boards, to deal with reality.

This older technology is perfectly adapted for the low-tech area it's destined for. Can you imagine this engineer—or one of us—in some remote spot with few tools, trying to repair a typical modern embedded product? One that's chock full of high-density SMT parts? With everything integrated into a couple of proprietary ASICs, which, of course, are no longer available?

Next time your cell phone fails, or when it's time to upgrade and your current model becomes another item on the technology junk heap, tear it apart. You'll be astonished at the packaging engineering. You'll find one or two boards covered front and back with micro-components soldered practically on top of each other. Without removing parts I bet you'll find that just closing the case up is difficult.

One of the worst symptoms of our headlong plunge into ever more elaborate and sophisticated technology is how rapidly parts become obsolete. Though this Batelco system used 8085 microprocessors that are still easily and cheaply available, most of our systems use components which won't exist in a year or two.

I can't count the times people have called or e-mailed me in despair over parts issues. Their product is just about done. Six to 12 months of engineering has yielded just the thing the boss wants and the company needs. Yet as they go to market, one or more integrated circuits, be they microprocessors, PLDs, or memory components, no longer exists. The "last buy" message comes from the vendor even before these poor folks have shipped a single unit. The panic that ensues is for all of the wrong reasons. Developers go into a frenzy because they need to ship product, usually forgetting that obsolete or near-obsolete components mean the system cannot be repaired in a year or a decade. Shipping is swell, but long term maintenance is critical.

The kinder, gentler days when vendors had second sources for parts, when they produced even low volume components for 10 to 20 years, are gone. Some of the semiconductor folks tell me the solution is to design everything into high-density FPGAs and PLDs. I've made that mistake, and found that FPGAs, for instance, go obsolete as fast as any other part. The newer replacement version is almost never pin-compatible with its predecessor, necessitating a board redesign. Without pin-compatible replacement parts, the poor sod in the field will need a reliable stock of spare boards. In some parts of the world board stocks will always be thin to non-existent.

Seems to me that building an irreparable system is extremely poor engineering. I do recognize that today's reality sometimes means that repair is not an option, that a defective unit gets immediately tossed out. In this landfilled world that's scary, but it's a part of the way things work, no matter how annoying. But if your system is destined for a place where such casual disposal isn't possible, designing things that way is foolish.

As an engineer, you're making choices every day that will impact the life of your user. Recognize that some economies don't operate as frenetically as the U.S.'s, that sometimes repair is more important than features, size, and power consumption.

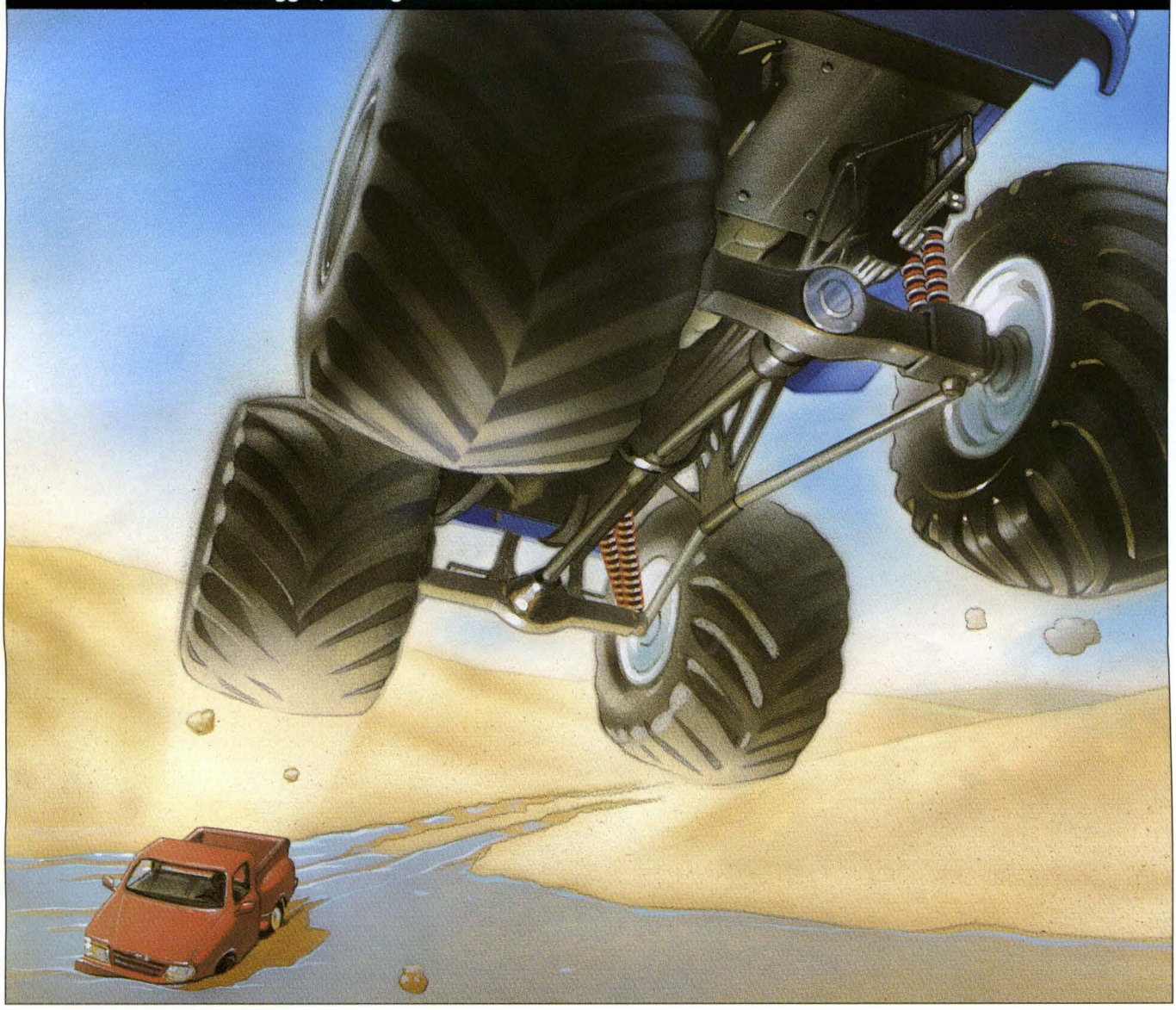
It's easy to succumb to an arrogance of the office, where we lose touch with real customer needs, when we listen on the phone with but a single ear to a repair person in the field who is upset, while we make faces and wish only that the turkey would shut up. As an engineering manager I learned to send the developers into the field, to let them do some service calls, interact with the customer, and feel first hand just how much pain these folks experience with the products. Let each engineer get beaten up in person by unhappy clients. It's a sure way to make them consumer advocates when they return, to give them a sense of what is really important.

So if your product is destined for more remote areas of the world, design appropriately. Help your customer succeed despite the challenges of the local environment. **esp**

Jack G. Ganssle is a lecturer and consultant on embedded development issues. He conducts seminars on embedded systems and helps companies with their embedded challenges. He founded two companies specializing in embedded systems. Contact him at jack@ganssle.com.

EMBEDDED SYSTEMS PROGRAMMING (ISSN 1040-3272) is published monthly, with an additional issue published in September, by CMP Media, 525 Market St., Ste. 500, San Francisco, CA 94105, (415) 905-2200. Please direct advertising and editorial inquiries to this address. SUBSCRIPTION RATE for the United States is $55 for 13 issues. Canadian/Mexican orders must be accompanied by payment in U.S. funds with additional postage of $6 per year. All other foreign subscriptions must be prepaid in U.S. funds with additional postage of $15 per year for surface mail and $40 per year for airmail. POSTMASTER: All subscription orders, inquiries, and address changes should be sent to EMBEDDED SYSTEMS PROGRAMMING, P.O. Box 1278, Skokie, IL 60076-8278. For customer service, telephone toll-free (888) 847-6177. Please allow four to six weeks for change of address to take effect. Periodicals postage is paid at San Francisco, CA and additional mailing offices. EMBEDDED SYSTEMS PROGRAMMING is a registered trademark owned by the parent company, Miller Freeman Inc. All material published in EMBEDDED SYSTEMS PROGRAMMING is copyright © 1999 by Miller Freeman Inc. All rights reserved. Reproduction of material appearing in EMBEDDED SYSTEMS PROGRAMMING is forbidden without permission. EMBEDDED SYSTEMS PROGRAMMING is available on microfilm/fiche from University Microfilms International, 300 N. Zeeb Rd., Ann Arbor, MI 48106, (313) 761-4700.

Debugger, In-Target Probes & In-Circuit Emulators for Intel Processors



It's hard to compete without the right tools.

Let's suppose you've just been tasked to develop an important new Intel processor based system and that management wants to take it to market *yesterday*.

Now what do you do? You certainly don't have time to get bogged down using those dusty old, general purpose tools you've got sitting around.

You need a proven development system and the tech support to make the most of it. You need



American Arium and the planet's best debugger. We offer...

- GNU support
- Unmatched breakpoint options
- Fast setup/ease of use
- Free upgrades and tech support for 1 year

We've got the tools you're going to need so why wait—call us today.

american arium

14811 Myford Rd., Tustin, CA 92780
Ph: 877-508-3970 Fax: 714-731-6344
E-mail: info@arium.com

For more information on any of our in-circuit emulators, visit...

www.arium.com

Visit us at Embedded Systems Conference 2000, Booth# 1621

8051

68HC11

Z80

Z180

68HC12

Mcore

683xx

PowerPC

68K

x86

68HC05

ColdFire

And Many More

From Concept...

With over 20 years of development expertise behind our hardware and software, we can make your trip from concept to customer a whole lot easier. One phone call to Avocet and you'll find the customer service you want, the tools you need, and the technical support you expect. We're in the business of making great ideas happen.

HMI-200 In-Circuit Emulator



The HMI-200 ProTool is one of the most popular in-circuit emulator lines ever made: dual trace buffers, overlay memory, complex break and trigger point capability, and built-in hardware-based software performance analysis are just a few of its powerful features. Support for 8051, 68HC16, 68K, Z80, 68302, CPU32, Z180/64180, 68306, CPU32+, 8085, 68307,68020, 68HC11, 8096, 68030, 6809, 68040

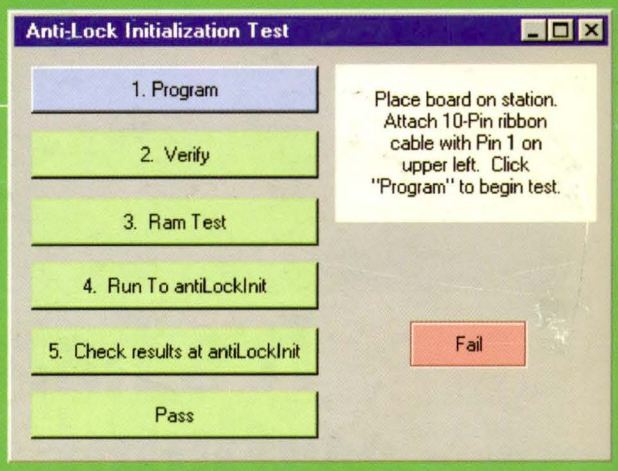


Background Mode Debugger

Harness the incredible speed and versatility of the USB interface with the USB-BMD from Avocet. Single-step code at the source or assembly level, manipulate CPU and peripheral registers, and program target Flash devices. Includes the full-featured SourceGate source level debugger.

SwifTest Production Tool Interface

Create your own custom production environment with the HMI-BMD. Enjoy the power and low-price of a BDM solution AND the ease-of-use and flexibility of an enhanced scripting language for easy GUI creation. Now you can have the debugging tool you need AND create the interface you want. Software bundle includes code samples, Tcl scripting language / interpreter, and the Tk GUI toolkit.



Sample custom interface created in less than an hour

...To Customer

AVOCET SYSTEMS, INC

(800) 448-8500

Visit us at Booth 116 at the Embedded Systems Conference

www.avocetsystems.com
sales@avocetsystems.com