

LINUX JOURNAL™

Since 1994: The Original Magazine of the Linux Community
FEBRUARY 2006

Embedded Linux

Get Started
Programming
for Qtopia

Python, home
automation,
security and you

THE NOKIA 770

This palm-sized
Internet tablet
is so flexible
and powerful
it may make
your notebook
obsolete



The **TI
MSP430
Processor
and Linux**

Embed the **db4o
Object-Oriented
Database**

**SINGLE SIGN-ON
AND THE CORPORATE
DIRECTORY, PART III**

USA \$5.00 CAN \$6.50
www.linuxjournal.com



THE UNSUNG HERO:

Network Administrator

- 9:42 am** Singapore branches go offline, trouble ticket created
- 9:44 am** Jeff uses diagnostics to isolate failure to core router – not responding
- 9:45 am** Out-of-band access to core router established via the **AlterPath™ ACS**
- 9:47 am** Router shows subnet mask set incorrectly during previous configuration
- 9:48 am** Jeff resets subnet mask properly, reboots router
- 9:49 am** Link to Singapore restored, Singapore comes back online
- 9:50 am** Jeff is planning his next vacation

Advanced Console Server



AlterPath™ ACS



Download a **FREE** White Paper on Console Management
www.cyclades.com/wpcm



www.cyclades.com/ljb
1.888.cyclades • sales@cyclades.com

The competition doesn't stand a chance.



If you base deployment decisions on performance and price, Coyote Point's for you. We've cornered that market.

To prove it we asked The Tolly Group to evaluate our E350si application traffic manager against the competition. The results speak for themselves.

Throughput? Almost 40% more than others in our space. Cost of transactions per second? Up to four times less. Connection rate? In some cases, one-sixth the cost. One-sixth! And we're told Coyote Point is the #1 choice for today's open source networks.

But don't just take our word for it. Get the facts. Call 1.877.367.2696 or write info@coyotepoint.com for your free copy of the full Tolly Report.



www.coyotepoint.com



COVER STORY

36 A FIRST LOOK AT THE NOKIA 770

The Nokia 770 seems to be on perpetual back order, and we can see why. This little Linux-powered hand-held does everything from Opera Web browsing to multimedia streaming. Better still, new features are just an upgrade away. Doc Searls has all the details and more, but be careful not to drool all over the article and smear the ink before you're finished reading it.

FEATURES

40 MSP430 DEVELOPMENT WITH LINUX

Blinking LEDs can be fun and instructive for using a nifty TI processor.

BRIAN C. LANE

48 SHELL SCRIPTING A CAMERA SERVER

A few contortions are needed but you can still shell script a surveillance system.

ERIK INGE BOLSO

52 GETTING STARTED WITH QTOPIA

Want to write an application for the increasingly popular PDA platform?

LORN POTTER

56 BUILDING A HOME AUTOMATION AND SECURITY SYSTEM WITH PYTHON

Want to use Python and cheap hardware to feel more secure about your home?

FRED STELTER

62 EMBEDDING THE DB40 OBJECT-ORIENTED DATABASE

How to get this single-library powerful database into your embedded system.

RICK GREHAN

INDEPTH

76 THIN CLIENTS PAY MORE

Thin is always in and cost effective too.

STEPHEN SEFTON

80 HETEROGENEOUS PROCESSING: A STRATEGY FOR AUGMENTING MOORE'S LAW

What do you do when you can no longer enforce Moore's law?

AMAR SHAN

86 EMBEDDING PYTHON IN YOUR C PROGRAMS

Ever wonder how to put Python on a C food diet?

WILLIAM NAGEL

92 THREE CASE STUDIES IN COMMUNITY-ORIENTED, OPEN-SOURCE SOFTWARE DEVELOPMENT

What do a PDA, video capture card and multimedia appliance have in common?

RANDALL P. EMBRY

TOOLBOX

16 AT THE FORGE

Assessing Ruby on Rails

REUVEN M. LERNER

20 COOKING WITH LINUX

Little Bitty Applications

MARCEL GAGNÉ

26 WORK THE SHELL

Conditional Statements and Flow Control

DAVE TAYLOR

28 PARANOID PENGUIN

Single Sign-On and the Corporate Directory, Part III

TI LEGGETT

COLUMNS

36 LINUX FOR SUITS

A First Look at the Nokia 770

DOC SEARLS

96 /ETC/RANT

Skim Cream not Scum

NICHOLAS PETRELEY

REVIEW

72 FAT MAN AND LITTLE BOY

JAMES TURNER

DEPARTMENTS

4 FROM THE EDITOR

6 LETTERS

12 UPFRONT

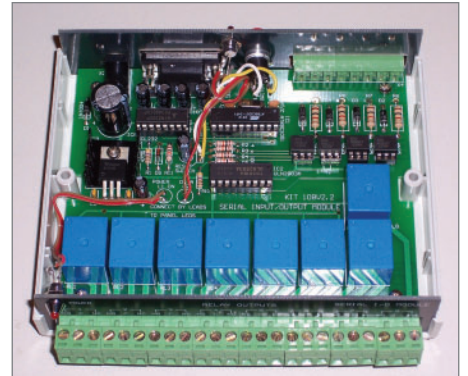
70 NEW PRODUCTS

81 ADVERTISERS INDEX

95 MARKETPLACE

LINUX JOURNAL

FEBRUARY 2006 ISSUE 142



Build your own home security system—see page 56 for details.

NEXT MONTH

SECURITY

Linux is by nature one of the most secure operating systems available. This doesn't mean a vulnerability isn't waiting around the corner, however. Find out several ways to configure your Linux system to keep out intruders, including Perl scripts to manage firewall rules, why the friendly Guarddog firewall tool has a bite better than its bark and how to lock down your Web server with the Apache mod_security module. But wait, there's more, including a stealth e-mail system for those of you who have always wanted to play James Bond at work.

We wrap up our single sign-on corporate directory series in the next issue with some impressive feats. Want to know how to set up printers for all, manage users and SSH keys, and even automate firewall rules via LDAP? We've got the scoop.

And that's not all. What do shell scripts and blackjack have in common? Why are Web services just a URL away? Sorry, you'll have to read the next issue to find out.

COVER IMAGE COURTESY OF NOKIA



Nick Gets His Wish

Introducing Nicholas Petreley as the new Editor in Chief of *Linux Journal*. **BY NICHOLAS PETRELEY**

I often bump into writer's block, but it's easy to start my first and probably only From the Editor column for *Linux Journal*. All I have to do is start by talking about how I often bump into writer's block, which allows me to segue into just about any topic at all.

When I first started the Webzine *LinuxWorld*, I remember *Linux Journal* publisher Phil Hughes gave me a dirty look at a convention in Atlanta, Georgia. Apparently, he thought we were out to crush *Linux Journal* with a well-funded magazine. Wrong on all accounts. *LinuxWorld* was not well funded, and it could never crush *Linux Journal*. When Phil showed some discomfort about *LinuxWorld*, I told him how much I preferred *Linux Journal*. He gave me a puzzled look and moved on.

Here it is, many years later, and I finally got my wish to join the marvelous *Linux Journal* team. Executive Editor Jill Franklin is a dream, and by far the best editor I've ever had the pleasure to work with. Garrick Antikajian is *Linux Journal's* extraordinary artist. Subscribe and download a PDF copy of *TUX* magazine (www.tuxmagazine.com) if you want to see how Garrick can turn anything into a work of art. VP of Sales and Marketing Carlie Fairchild is unspeakably kind and helpful. And I can't wait to work more closely with one of my heroes, Doc Searls. I could go on, but I'd run out of space.

I can't sum up where I want to take *Linux Journal* in 25 words or less. So suffice it to say I want to

keep all the good stuff, make some things more fun, add more articles of practical value and tune others to have more practical value. Stay tuned.

Oh, why is this probably the only From the Editor column you'll see from me? I prefer to do something I enjoy a lot more. Rant. Look for my monthly rant at the end of each magazine, including this one. Okay, I'm perfectly aware that some of you don't like reading rants. For \$20, I'll let you folks skip that page. (I can't do that? Sorry.) But I know some of you not only need to rant, yourselves, you also have some of the same pet peeves I have. Some of you have pet peeves totally opposite of mine. But I have a feeling I'll hit a nerve one way or another, and that's a good thing.

Last but definitely not least, send us e-mail at ljeditor@ssc.com. I'll fess up right now: I'm overwhelmed with the task of coming up to speed on *Linux Journal*. That, and I'm a full-time single dad of two young kids. The combo ain't easy, so I won't always get to your e-mail in a timely manner. But I want your input. Always. That's how to keep *Linux Journal* the magazine you want it to be. So give me some time to adjust, but drop us a line and let us know what you're thinking. We love it.■

Nicholas Petreley is Editor in Chief of *Linux Journal* and a former programmer, teacher, analyst and consultant who has been working with and writing about Linux for more than ten years.

EDITOR IN CHIEF Nicholas Petreley, ljeditor@ssc.com

EXECUTIVE EDITOR Jill Franklin, jill@ssc.com

SENIOR EDITOR Doc Searls, doc@ssc.com

WEB EDITOR Heather Mead, heather@ssc.com

ART DIRECTOR Garrick Antikajian, garrick@ssc.com

EDITOR EMERITUS Don Marti, dmarti@ssc.com

TECHNICAL EDITOR Michael Baxter, mab@cruzio.com

SENIOR COLUMNIST Reuven Lerner, reuven@lerner.co.il

CHEF FRANÇAIS Marcel Gagné, mggagne@salmar.com

SECURITY EDITOR Mick Bauer, mick@visi.com

PRODUCTS EDITOR James Turner, newproducts@ssc.com

CONTRIBUTING EDITORS

David A. Bandel • Greg Kroah-Hartman • Ibrahim Haddad • Robert Love • Zack Brown • Dave Phillips • Marco Fioretti • Ludovic Marcotte • Paul Barry • Paul McKenney

PROOFREADER Geri Gale

VP OF SALES AND MARKETING Carlie Fairchild, carlie@ssc.com

MARKETING MANAGER Rebecca Cassity, rebecca@ssc.com

INTERNATIONAL MARKET ANALYST James Gray, jgray@ssc.com

REGIONAL ADVERTISING SALES

NORTHERN USA: Joseph Krack, +1 866-423-7722 (toll-free)
EASTERN USA: Martin Seto, +1 416-907-6562
SOUTHERN USA: Laura Whiteman, +1 206-782-7733 x119
INTERNATIONAL: Annie Tiemann, +1 866-965-6646 (toll-free)

ADVERTISING INQUIRIES ads@ssc.com

PUBLISHER Phil Hughes, phil@ssc.com

ACCOUNTANT Candy Beauchamp, acct@ssc.com

LINUX JOURNAL IS PUBLISHED BY, AND IS A REGISTERED

TRADE NAME OF, SSC MEDIA CORPORATION

PO Box 55549, Seattle, WA 98155-0549 USA • linux@ssc.com

EDITORIAL ADVISORY BOARD

Daniel Frye, Director, IBM Linux Technology Center
Jon "maddog" Hall, President, Linux International
Lawrence Lessig, Professor of Law, Stanford University
Ransom Love, Director of Strategic Relationships, Family and Church History Department, Church of Jesus Christ of Latter-day Saints
Sam Ockman, CEO, Penguin Computing
Bruce Perens
Bdale Garbee, Linux CTO, HP
Danese Cooper, Open Source Diva, Intel Corporation

SUBSCRIPTIONS

E-MAIL: subs@ssc.com • URL: www.linuxjournal.com
PHONE: +1 206-297-7514 • FAX: +1 206-297-7515
TOLL-FREE: 1-888-66-LINUX • MAIL: PO Box 55549, Seattle, WA 98155-0549 USA • Please allow 4-6 weeks for processing address changes and orders • PRINTED IN USA

USPS *LINUX JOURNAL* (ISSN 1075-3583) is published monthly by SSC Media Corporation, 2825 NW Market Street #208, Seattle, WA 98107. Periodicals postage paid at Seattle, Washington and at additional mailing offices. Cover price is \$5 US. Subscription rate is \$25/year in the United States, \$32 in Canada and Mexico, \$62 elsewhere. POSTMASTER: Please send address changes to *Linux Journal*, PO Box 55549, Seattle, WA 98155-0549. Subscriptions start with the next issue. Back issues, if available, may be ordered from the Linux Journal Store: store.linuxjournal.com.

LINUX is a registered trademark of Linus Torvalds.

TYAN

Fast, Flexible, and Feature-Rich!

PCI Express and EM64T Servers Have Arrived

Dual Core Ready!!



Support up to two Dual-Core Intel® Xeon™ processors with 800MHz FSB and EM64T



8 DDR333/266 DIMM sockets for reg. ECC memory



One proprietary TARO SO-DIMM connector on PCI-X bus, supports U320 SCSI or SATA IDE RAID



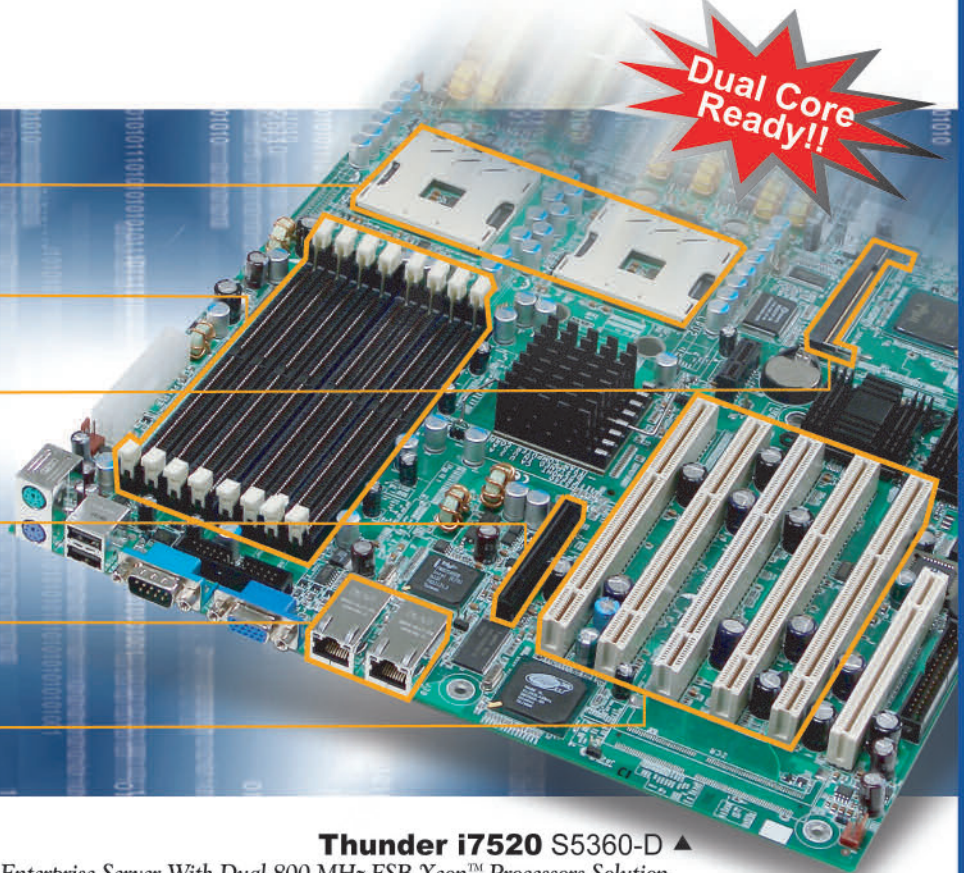
One PCI Express (x8) slot



Dual 10/100/1000 GbE LAN ports on 64-bit PCI-X bus



- Three PCI-X 133/100/66 MHz slots
- One PCI-X 100/66 MHz slot
- One PCI-X 66 MHz slot



Thunder i7520 S5360-D ▲

Enterprise Server With Dual 800 MHz FSB Xeon™ Processors Solution



Thunder i7520 S5360-D



- Supports two Dual-Core Intel® Xeon™ processors with 800 MHz FSB and EM64T
- 8 DIMMs for DDR266/333 memory
- Three PCI-X 133/100/66 MHz slots, one PCI-X 100/66 MHz slot, one PCI-X 66 MHz slot and one 33 MHz PCI slot
- One PCI Express™ x8 slot
- One proprietary SO-DIMM connector on PCI-X bus, supports U320 SCSI or SATA
- Dual GbE LAN

Tiger i7320 S5350-D



- Supports two Dual-Core Intel® Xeon™ processors with 800 MHz FSB and EM64T
- 8 DIMMs for DDR266/333 memory
- Two PCI-X 64/66 MHz slots; three 32/33 PCI 2.3 slots
- One proprietary SO-DIMM connector on PCI-X bus, supports U320 SCSI or SATA
- Dual PCI Express GbE LAN

TYAN COMPUTER CORP.

Tyan Computer USA

3288 Laurelview Court
Fremont, CA 94538 USA
Tel: +1-510-651-8868 Fax: +1-510-651-7688
Pre-Sales Tel: +1-510-651-8868 x5120
Email: marketing@tyan.com

For more information about this and other Tyan products, please contact Tyan Pre-Sales at (510) 651-8868 x5120, or contact your local Tyan system integrator/reseller.

www.tyan.com

Infocom Games

In the January 2006 Get Your Game On column, Dee-Ann LeBlanc wrote: “The closest I got this time was pulling out my *Classic Text Adventure Masterpieces* CD with old Infocom games.”

You and many of your readers probably didn't know that you can already play most of these under Linux very smoothly. It turns out that the Infocom games were written for a virtual machine (sort of like Java); they come in two pieces, the actual game and the virtual machine interpreter. Several replacement virtual machine interpreters have been written that run on Linux and are under the GPL. One is called Frotz, and the current version is 2.43. It's available from www.cs.csuak.edu/~dgriffi/proj/frotz; it's also packaged for Debian and probably other distributions. There are many others.

You have to experiment a bit to find out where the actual game file is on the CD, since they've been put in different places for each game. For *zork 1*, it's in `pc/zork1/data/zork1.dat`, so run `frotz pc/zork1/data/zork1.dat` in a console and you're off. Some of the game files have a .zip extension rather than a .dat extension (that's what Infocom called its virtual machine format, back in the days before PKZIP).

Frotz doesn't behave properly on the games with graphics, like *Zork Zero*; support for those is in a derivative GPL program that runs only on Windows, called WindowsFrotz2002, and will hopefully be ported back to the original someday. In the meantime, WindowsFrotz2002 runs under Wine, and can be downloaded from www.ifarchive.org/indexes/if-archive/Xinfocom/Xinterpreters/Xfrotz.html, but it requires pictures translated into the Blorb format, which you'll have to poke around the Net for.

--
Nathanael Nerode

Dee-Ann LeBlanc replies: *Thank you very much! I think I might have known this at some point along the way many moons ago and then forgot. When I get home from being on the road, I just might pull out that Infocom CD again and cover this.*

Another Name for Linux

Regarding the frivolous little bit, “Might Be Just Right” [*LJ*, December 2005, page 94], does the word Lagom apply to Linux? I think

it might, but maybe another word that applies to Linux is Gaia, in that Linux consists of many organisms working together as a single organism. Linus Torvalds did not create an operating system; he created a kernel that was made into a successful operating system by obtaining the cooperation of an array of GNU utilities, which in turn spawned many computer geeks around the world to use the fledgling and somewhat useful UNIX system to fix and enhance the system and its utilities, adapt other stuff to work with Linux and write new stuff from scratch and so on. You get the picture. There is no real master organism, but there are many organisms spontaneously working together toward the one common good.

Another definition: Linux is an example of a near-perfect anarchy for all the same reasons that make it a Gaia—all work together for the advancement of their common good without a government or owner directing their activities. Alexander Berkman would be proud.

Is that black helicopters I hear coming this way?
--
Richard

Clarification

I'd encourage Maestro Mr Taylor to make a point of reminding the readers of his Work the Shell column that his work environment is specific to the Bourne shell, and some users, for any number of legitimate reasons, may be confronted with a C-Shell environment. As a nicety to these readers, maybe a quick side-track into how to get their own Bourne shell environment? [See Dave Taylor's Work the Shell column, beginning in the December 2005 issue of *LJ*.]

Thank you all for your hard work!
--
Michael C. Tiernan

Dave Taylor replies: *I knew we couldn't go too long without someone bringing up the great religious war of the shell scripting world, which shell to use. I tried to highlight in my first column that I would be writing for the Bourne Again Shell (though almost everything will work with any modern Bourne Shell too, especially if it's a Posix-compliant distro), but just to clarify, I don't think it really matters what shell you opt to use; with just a few relatively minor syntactic changes, the basic concept of scripting and how you utilize Linux commands to accomplish extraordinarily difficult programming tasks in just 5–10 lines remains the same.*



Also, you can certainly have the C Shell or one of its variants as your command-line interpreter/login shell and still use Bash as your scripting environment of choice.

To find out what shell you're currently running, simply type in the following command: `ps -p $$`. It'll either say `sh`, `csh`, `bash`, `tcsh` or similar. The syntax used throughout the shell scripting column is for `sh` or `bash`.

Thanks for the Old School Article

Matthew Hoskins' article "UNIX: Old School" in the December 2005 issue was a gem. I found it intriguing that a vintage 1974 release of UNIX could be booted under a PDP-11 simulator and experienced firsthand. A terrific retrospective.

--
Troy Hanson

On Patents

Subject: Re: On Patents

Roger Wolff's response to Don Marti [see Letters in the December 2005 issue of *LJ*] displays a weak understanding of patents. He uses the descriptions of the patents to decide what they cover. That doesn't work. To understand what the patent covers you must read the claims section. Everything else is little more than window dressing. Unfortunately, the claims section is written in lawyer-ese. Reading the rest of the patent can be helpful for setting context for the claims. But that is all that it is.

For his example "method for coding an audio signal", the claims could be very narrow or could be broad. Much depends on how careful the patent examiner was. But you simply cannot tell from the description.

Disclaimer: I am not a lawyer. But I do have some patents.

--
Allen Brown

User Management

- support more than 3000 PPPoE or HotSpot clients
 - full RADIUS support for user parameters
 - tx/rx speed, address, filter rules
 - supports RADIUS real time modification of parameters while users are online
- Peer to Peer protocol control (P2P)
 - per client P2P tx/rx rules with burst support
 - P2P pool
 - complete blocking of P2P

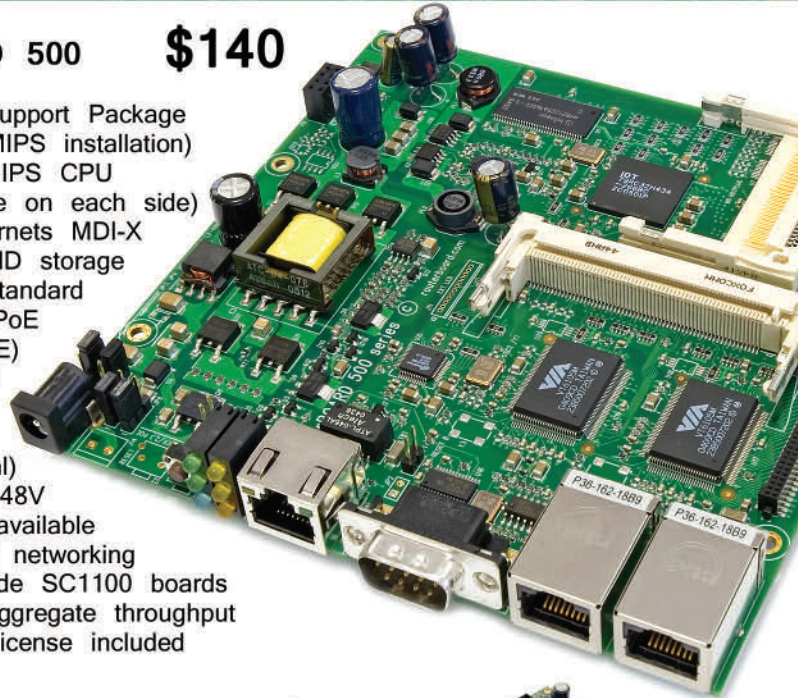
Wireless AP and Backbone

- Wireless monitoring
 - Frequency scanning with detailed report
 - Raw wireless packet sniffer
 - streaming option to Ethereal analyzer
 - option to save to a file format supported by Ethereal
 - Snooper packet inspection
 - analyzes all raw frames received for wireless parameters
 - monitors a single channel or all channels
- Nstreme wireless polling protocol
 - no decrease in speed over long distances (as seen with the 802.11 ACK packet bottleneck)
 - polling improves speed and eliminates contention for access to the wireless bandwidth
 - access point control over Nstreme clients tx data to optimize use of the wireless medium
 - RADIUS support for the access control list including bandwidth settings for wireless clients
- Full 802.11a/b/g support

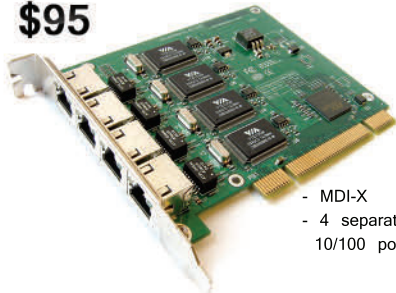
The above is a brief description of a few features, for more

RouterBOARD 500 \$140

- Linux Board Support Package (full Debian MIPS installation)
- 266-400MHz MIPS CPU
- 2 miniPCI (one on each side)
- 3 10/100 Ethernet MDI-X
- 64/128MB NAND storage
- PoE 802.3af standard and passive PoE (also 12V PoE)
- Compact Flash
- Low power
- 32MB DDR (64MB optional)
- 6-24V and 25-48V power mode available
- 2-3x faster for networking than the Geode SC1100 boards
- 200-300Mb/s aggregate throughput
- L3 RouterOS license included



\$95



- MDI-X
- 4 separate 10/100 ports

RouterBOARD 44

For the Router Builder !

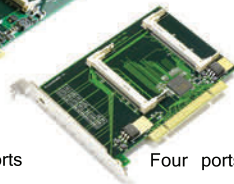
- rackmount servers and routers
- up to 24 Ethernet ports in a PC
- no more straight/cross cable problems
- server quality VIA VT6105 chips

\$120



Eight ports

\$65



Four ports

RouterBOARD 11/14/18

Multi radio tower !

MiniPCI to PCI adapters for multi radio system. Tested with sixteen radios in one Router/AP.

\$195



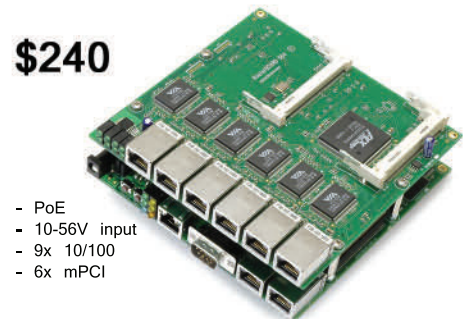
Linux Journal Special
Free 64MB SoDIMM

RouterBOARD 230

No feature left behind !

Integrated router with various interfaces. Use as an AP on a tower with up to 500ft PoE. Includes IDE/CF, miniPCI, USB, PCMCIA, UART, PCI, GPIO, LCD controller, Linux SDK, and more.

\$240



- PoE
- 10-56V input
- 9x 10/100
- 6x mPCI

RouterBOARD 500 & RouterBOARD 564

The Wireless Switchboard !

For a complete multi-radio tower system, the RouterBOARD 500 can carry a daughterboard (RouterBOARD 564) which adds six ethernet and four miniPCI.



Monarch Computer.com



Visit our website or Call 1-800-611-0875



nero 7 ULTRA EDITION

with an upgrade to AMD Athlon™ 64!



FREE! - Get NERO 7 Ultra, a Patriot SD FlashCard Kit, and a LIVESTRONG™ Wristband when you upgrade to Athlon 64/FX/X2!



Purchase any AMD Athlon™ 64/FX/X2 bundle from Monarch and get NERO 7 Ultra, a 128 MB Patriot SD Flash Card Kit w/USB Reader and a LIVESTRONG™ wristband for helping Monarch support the battle against cancer. Monarch Computer and you can make a difference.



FREE INSTALLATION SETUP & TESTING BY CERTIFIED TECHS

FREE TECH SUPPORT!

Monarch makes it quick and easy to upgrade with FREE setup and testing on Motherboard Combos and \$18.00 build fee on Barebones.

AMD Motherboard Combos

ASRock 939 Dual-SATA2 w/ AMD Opteron™ processor 148 (939 pin)

Only \$295

Tyan S2865ANRF nF4 w/AMD Opteron™ processor 175 (939 dual-core)

Only \$649



DFI NF4 Infinity nForce4 with AMD Athlon™ 64 processor 3700+ (939)

Only \$329

Abit AN8 SLI nForce4 w/AMD Athlon™ 64 X2 processor 4600+ (939 dual-core)

Only \$775

Tyan S2895A2NRF K8WE w/ AMD Opteron™ processor 246 (940 pin)

Only \$665

GigaByte GA-7A8DW w/AMD Opteron™ processor 275 (940 dual-core)

Only \$1239

Mainboard - Processors - Heatsink and Fan with Memory Options - FREE INSTALLATION AND TESTING
LATEST BIOS loaded for easy upgrades - AMD Sempron™ Athlon™ 64, Athlon™ 64 FX, and Opteron™ Combos Available

AMD Barebone Systems

Go to www.monarchcomputer.com, select Barebones from the menu. Choose AMD Sempron™, AMD Athlon™ 64 or AMD Opteron™. Then configure your barebones online or call 1-800-611-0875.



Enermax CS-10182-BA E-ATX Tower w/460W PS
Tyan S2875ANRF (Tiger K8W 8151) MB
AMD Opteron™ processor 242 (940)1.6 GHz

Starting @ \$609

SuperMicro (Barebone) 1020A-8 w/Hot-Swap SCSI & 420W PS
Abit KN8 nF4 Motherboard
AMD Opteron™ processor 244 (940)

Starting @ \$1065

Antec SLK1650B Black Mini Tower w/350W PS
MSI K8N NEO4-F Motherboard
AMD Opteron™ processor 148 (939)

Starting @ \$379

Lian-Li PC-V1200 Plus Aluminum Tower w/460W PS
Tyan S2895UA2NRF K8WE Motherboard
AMD Opteron™ processor 280 (940)

Starting @ \$2129

***AMD Athlon 64 and Athlon 64 FX are the first Windows®-compatible 64-bit PC processors

AMD Price Drop!

AMD Opteron™ OEM CPUs (939)
AMD Opteron™ 146 2.0GHz (939) \$121.00
AMD Opteron™ 148 2.2GHz (939) \$207.00
AMD Opteron™ 150 2.4GHz (939) \$311.00
AMD Opteron™ 152 2.6GHz (939) \$775.00

AMD Opteron™ OEM CPUs (940)
AMD Opteron™ 146 2.0GHz \$121.00
AMD Opteron™ 148 2.2GHz \$207.00
AMD Opteron™ 150 2.4GHz \$311.00
AMD Opteron™ 244 1.8GHz \$158.00
AMD Opteron™ 246 2.0GHz \$203.00
AMD Opteron™ 248 2.2GHz \$307.00
AMD Opteron™ 250 2.4GHz \$441.00
AMD Opteron™ 252 2.6GHz \$669.00
AMD Opteron™ 254 2.8GHz \$825.00
AMD Opteron™ 848 2.2GHz \$677.00
AMD Opteron™ 850 2.4GHz \$847.00
AMD Opteron™ 852 2.6GHz \$1130.00
AMD Opteron™ 854 2.8GHz \$1469.00

AMD Opteron™ Dual Core OEM CPUs
AMD Opteron™ 265 1.8GHz \$669.00
AMD Opteron™ 270 2.0GHz \$825.00
AMD Opteron™ 275 2.2GHz \$1019.00
AMD Opteron™ 280 2.4GHz \$1260.00
AMD Opteron™ 865 1.8GHz \$1130.00
AMD Opteron™ 870 2.0GHz \$1469.00
AMD Opteron™ 875 2.2GHz \$2085.00
AMD Opteron™ 880 2.4GHz \$2570.00

AMD Athlon™ 64 X2 Retail CPUs
AMD Athlon™ 64 X2 3800+ \$284.00
AMD Athlon™ 64 X2 4200+ \$363.00
AMD Athlon™ 64 X2 4400+ \$467.00
AMD Athlon™ 64 X2 4600+ \$556.00
AMD Athlon™ 64 X2 4800+ \$643.00

AMD Athlon™ 64 Retail Box CPUs
AMD Athlon™ 64 3000+ (939) \$109.00
AMD Athlon™ 64 3200+ (939) \$152.00
AMD Athlon™ 64 3500+ (939) \$201.00
AMD Athlon™ 64 3700+ (939) \$233.00
AMD Athlon™ 64 3800+ (939) \$282.00
AMD Athlon™ 64 4000+ (939) \$334.00
AMD Athlon™ 64 FX-57 (939) \$811.00
AMD Athlon™ 64 FX-60 (939) \$1011.00

AMD Opteron™ Dual Core OEM CPUs (939 pin)
AMD Opteron™ 165 1.8GHz \$270.00
AMD Opteron™ 170 2.0GHz \$356.00
AMD Opteron™ 175 2.2GHz \$461.00

The AMD Athlon™ 64 X2 dual-core processor provides the same level of system features customers have grown to expect with the AMD Athlon™ 64 product family: HyperTransport™ technology - Enhanced Virus Protection for Microsoft® Windows® XP, SP2 - Cool'n'Quiet™ technology

Components and Upgrades 1000s of In-Stock Components

100988 Antec SLK1650B Black Mini Tower w/350W PS \$63.00

150419 Seagate (ST3400832AS) 400 GB 7200 RPM 8MB Cache SATA \$235.00

100340 PC Power & Cooling 1000W SLI/EPS TC1KW Power Supply \$459.00

140442 1 GB DDR (400) PC-3200 Corsair (VS1GB400C3) \$83.00

190522 ATI (Connect3D) Radeon 9250 256MB DDR/AGP TV-Out/DVI (Retail Box) \$53.00

150034 Seagate (ST3300831AS) 300 GB 7200 RPM 8MB Cache SATA \$152.00

150142 3Ware 9550SX-8LP Kit, 8 port SATA low profile PCI 64/133 RAID Controller \$471.00

110748 Gigabyte GA-K8N51PMT-9 nF4 Video/Audio/GB-LAN/IEEE-1394 PCI-E/TV-Out/SATA3G DDR/ATX 64 939 \$87.00



Educational and Government POS Welcome.

We ship to the Continental U.S., Alaska, Hawaii, APOs, Puerto Rico, and Canada



Commercial leasing available for purchases as low as \$1000.



Prices subject to change without notice. Monarch Computer not responsible for typographical errors. AMD, the AMD Arrow logo, AMD Athlon, and combinations thereof, are trademarks of Advanced Micro Devices, Inc. All brands and product names are trademarks or registered trademarks of their respective companies. AMD model numbers indicate relative software performance among this AMD processor family. Enhanced Virus Protection (EVP) is only enabled by certain operating systems including the current versions of Microsoft® Windows®, Linux®, Solaris®, and BSD Unix. After properly installing the appropriate operating system release, users must enable the protection of their applications and associated files from buffer overflow attacks. Consult your OS documentation for information on enabling EVP. Contact your application software vendor for information regarding use of the application in conjunction with EVP. AMD and its partners strongly recommend that users continue to use a third party anti-virus software as part of their security strategy.

Monarch Has The **LOWEST PRICES**
 Custom 64-Bit Servers,
 Workstations & Desktops
 Available with AMD Dual-Core Technology!



Take the Monarch Challenge: Give us 5 minutes and Save \$100s if not \$1000s! Come see our jaw-dropping prices! In most cases, we're cheaper than the competition **BY AS MUCH AS 20%!**

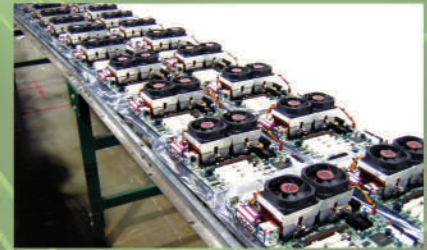
FIRST EVER 4-Way Dual-Core Opteron™ Tower!
Only Monarch can deliver this much for so little!

Monarch Empro™ 4-Way Tower Server



Part #: 80809
SELECTED COMPONENTS:
 Cooler Master RC-810 CM Stacker 810 (Silver or Black)
 PC Power & Cooling Turbo-Cool 850W ATX EPS/SSI SLI Power Supply
 Tyan S4882UG2NR 8131 Motherboard
 Video/GB-LAN/USB/SATA/SCSI/DDR-400 E-ATX Dual-Core Quad-Opteron
 Featuring 2 AMD Opteron™ Processors 870
 16 GB DDR (400) PC-3200 Patriot REG ECC Signature (PSD2G40036ERB)
 4 x Seagate Cheetah 146 GB 15KRPM
 Ultra 320 68pin SCSI w/RAID 5 w/Hot Spare
 Adaptec 2200S Ultra 320 SCSI RAID Controller
 Plextor PX-716AL 6x-DVD+R DL 16x8x16x-DVD±RW
 Mitsumi 1.44MB 3.5" (Black) Floppy Drive
 Choice of Sound Card
 Choice of PCI Video Card
 Choice of Linux or Microsoft OS
 Choice of Network Options
 Industry Standard Upgradable
 1 or 3 year warranties available
 24/7 on-site service available

Starts @ Only **\$2784!** This config **\$18299!**



1 Configuration review

Our senior technical staff review every configuration to eliminate hardware and software incompatibilities.



2 Production scheduling

We allocate all parts for your system, and chart your system's assembly path through our production facility.



3 Board test

We assemble your motherboard, processor and memory and test these core components extensively and load the latest BIOS.



4 Build stage

All components for your system are assembled into your chassis. All cables are tied off and tucked away to increase airflow and cooling.



5 Burn in diagnostic

We combine hands-on diagnostics with a battery of automated burn-in testing to ensure all your components are operating properly together.



6 Software load

We load your OS onto your hard drive along with all factory tested updates and the most recent hardware drivers.



7 Quality control

Our QC experts put your system through a rigorous 62 point inspection to verify the system is in working order before shipping.



8 Packing & shipping

We expertly pack your system for secure and safe shipping using customized packaging and double boxing.

Monarch Furia™ Custom Workstation



Part #: 80307
SELECTED COMPONENTS:
 Enermax CA-3030-BS Midtower Case w/400W PS ATX
 MSI RS482M4-ILD ATI RS482 Video/Audio LAN/IEEE-1394/USB/PCI-E/SATA/DDR
 AMD Opteron™ Processors 146 (2.0GHz 1MB 64/32-Bit 939-pin)
 1 GB (2 pcs 512) DDR (400) PC-3200 Patriot (PSD1G400K)
 Western Digital 80GB SATA 8MB Cache 7200 RPM (WD800JD)
 LITE-ON SOHR-5239V 52X32X52 CD-RW
 Industry Standard Upgradable
 1 or 3 year warranties available
 24/7 on-site service available

Operating System and Accessories available Separately

This config @ ONLY **\$599!**



And More on Patents

I found it interesting to read the “Archives, Patents” letter by Kari Laine, and the answer from *LJ* [November 2005 issue, page 10]. The answer totally avoids the issues involved, when you form a symbiosis with something that you’re trying to fight. Kari’s letter gives the impression that patents are evil, and yet Kari suggests a model where a part of the Open Source community relies on patents to fill the bank account. How can you wholeheartedly fight something that puts food on your table?

Now, I’m not saying that the Patent Commons Project is bad because it forms a symbiosis with the patent system—open-source purists will take on that discussion if needed.

Maybe *LJ* can clarify the implications of forming symbiosis with what you’re trying to fight, by having Doc Searls write about it, all the way from putting hooks to binary-only modules into the kernel (pwc/pwxc), over the emulation of “the deselected OS” (Wine), to making money on patents when you actually don’t like them?

--

Martin A. Bogelund

A Few Words on Usability

I have a few comments to share about the article “Bringing Usability to Open Source” in the January 2006 issue of *LJ*.

First off, I must thank Nate and the *Linux Journal* for printing the article. I have to stress that usability should be one of the number one goals of all software creators, open source or otherwise. There is no substitute for observing users, because as Nat has pointed out, they often have a different mindset from that of the developer. Furthermore, developers often consider a feature finished when it “works”, without realizing how normal users flow through the processes they use on a daily basis. More times than I can count, I have sat down at my software with users and realized immediately that I missed the usability boat entirely as I watched them go through several painstaking and frustrating steps I had never envisioned to accomplish their task. In most of these cases, it wasn’t a case of “bad user”, it was a case of “bad software”—which is hard for a guy like me to admit.

However, I think one has to be careful when making software more “usable” not to be equally blinded by the actions of a few users.

As an exercise, imagine you took a young child

out to your vehicle, set up a camera, and asked the child to “take me to the store”. While this is a very simple task, you might imagine your young friend might grasp the wheel, step on a couple of pedals, turn the key (with the clutch out) smash into the garage door (oops that wasn’t right), and then give up and admit that he or she doesn’t know how to do it.

In my mind, this does in no way imply that the car is “broken” from a usability perspective. A well-meaning engineer may take this feedback to indicate that the vehicle should be altered in some way to ensure that this novice would succeed next time. The problem is that any individual with driving experience would get in this “fixed” vehicle and become instantly annoyed at having to step through some new sequence of steps, most likely taking more time, and ultimately being much less usable.

In Nat’s specific example of the New button being wrong, I would have to disagree respectfully. First of all, the New button is quite consistently used in many software packages to begin a new action. New may be too terse, but conceptually it’s correct. In my opinion Send would be incorrect, because the option doesn’t send anything, it only creates a new message. If you want to send the message, you need to press Send. This would be my interpretation, of course, which doesn’t imply it is correct.

In this case, I might recommend that the button should say New Email (for English speakers). This is both more clear and yet remains true to the action it represents. Perhaps this user would not have stumbled with this minor change in place.

The point is not to disagree with the article or Nat’s intentions. The point is to ensure that software engineers and developers understand the need to take usability very seriously and look at a problem from several angles before making a decision. Because writing bad software is much easier than writing good software, and there are more wrong ways than right ways, selecting the best way is a huge challenge.

There are a lot of factors that go into usable software. I have always believed that some key elements are 1) consistency, both internally and among other applications. 2) Clarity—language should fit the action precisely and be backed up by verbose tooltips that users can rely on to clarify (tooltips should always be present as a matter of consistency). 3) Efficiency—displays should present all relevant information organized comprehensibly. Help should be available

and abundant. Help should never restate the question (that is, help like “The Sort button will sort your results” is not really very helpful). The number of keystrokes and mouse clicks needed to accomplish tasks should be minimized. Repetitive tasks should have shortcuts. Process flow should always be taken in to consideration. Things like allowing multi-select rather than single select can mean the difference of an hour of work versus five seconds of work. Always automate where appropriate, but give options. Never lock the user in to a process but ensure that the most basic processes are well designed and flow well and have a minimum of “speed bumps”. Allow the user to turn off or skip features designed to assist a novice. You can try to anticipate what a user wants, but don’t insist.

I can easily give examples of “bad” usability that has been coded into software. Just for fun, consider some features that I have experienced while running Windows. Every time I put in a new CD the OS tries to “guess” what I want to do with it, wasting my time when I could have already been doing it. And I really love it when Windows tries to “help” me clean up the icons on my desktop. Yeah, I really want to delete those links, thanks for asking me (*every day*). This does not improve my experience, and it’s frustrating how difficult it is to disable these behaviours, even for a relative expert without a lot of time to go hunting down the method.

And yes, I could give examples in Linux as well.

I’m sure I could write on the subject for days. Though I don’t consider myself an expert at interface design, I am fairly passionate about the need for usability in software, and I’ve made my share of mistakes. I’ve also been subjected to my share of bad software as I go about my own daily tasks, as I’m sure most readers have.

It is my hope that the article and the work of the Novell Usability Labs will help spur more work on the subject. Success of the Linux desktop has more to do with mind-share than anything else at this point. The platform is ready. We are close folks! Let’s keep going!

--

JC Simonsen

We welcome your letters. Please submit “Letters to the Editor” to ljeditor@ssc.com or SSC/Editorial, PO Box 55549, Seattle, WA 98155-0549 USA.

MORE SPACE. LESS MONEY.



UNLIMITED AFFORDABLE NETWORK STORAGE

Everybody needs more space. And they need to spend less money. What if you can both have more space and spend less money?

What if you could put 7½ terabytes in only 3 rack units? What if that 7½ terabytes cost less than \$10,000? Including the SATA disk drives. Imagine if you could glue it all together with a RAID appliance into one system. What if you could add as much storage as you wanted, one shelf at a time, and never have to 'fork-lift' anything?

Coraid's new SATA EtherDrive Storage allows you to do just that. Using industry standard SATA disk drives, EtherDrive Storage connects disks directly to your Ethernet network. Each disk appears as a local drive to any Linux, FreeBSD or Solaris system using our open ATA-over-Ethernet (AoE) protocol. Since the disks just appear as local drives you already know how to use them.

The EtherDrive® SATA Storage Shelf is a 3U rack-mount network appliance that contains 15 SATA drive slots. Its triple redundant power supply protects you from your most likely failure. Its dual Gb Ethernet interfaces allow your data to go fast; 200MB per second. And at a very affordable price. List price for the EtherDrive Storage Shelf, without disks, is only \$3,995.

Our companion product, the RAIDBlade RAID controller, allows a virtually unlimited number of Storage Shelves to be combined into a set of logical AoE storage devices.

Now you can have unlimited storage at a very affordable price. For complete information, visit our website at www.coraid.com, or call, toll-free, 1-877-548-7200. And we'll show how we've made network storage so affordable, you can have all the space you want.

www.coraid.com
info@coraid.com
1.706.548.7200



ON THE WEB

Are you thinking that it might be time to move your music writing and production projects to Linux? Or, have you heard some talk about the much-improved ALSA Project, but aren't exactly sure what tools are available and what they can do? Whether you are starting fresh on creating your musical oeuvre or want to move your work to open source, Dave Phillips' long-running LJ.com series, *At the Sounding Edge*, offers all sorts of overviews, tips and how-tos on ALSA, Ardour, Planet CCRMA, Hydrogen, MIDIs, soundfile editors and more.

Dave's current topic is music notation software for Linux—programs that do both music scoring and music typesetting. So far, he's covered the Lisp-based Common Music Notation (www.linuxjournal.com/article/8670) and the abc music notation specification language (www.linuxjournal.com/article/8629). In addition, Dave also wrote a two-part article on LilyPond for us. LilyPond is the current favorite among many Linux musicians when it comes to music notation, because, as Dave writes, "LilyPond automatically formats most music for excellent printed output, at the same time permitting highly detailed customizations to accommodate virtually any music scoring requirement, including unusual and idiosyncratic notations." For details on how to get LilyPond, what it can do and what its GUIs offer, check out Dave's articles at www.linuxjournal.com/article/7657 and www.linuxjournal.com/article/7719.

For a complete list of all Dave's *At the Sounding Edge* articles, plus other Linux audio coverage, take a look under the Audio/Video Category (www.linuxjournal.com/taxonomy/term/14/9) on LJ.com. And if you recently completed a musical masterpiece with the help of Linux audio software or wrote your own piece of audio software, drop us a note at webeditor@ssc.com.

diff -u

What's New in Kernel Development

The **ksymoops** tool is no longer needed for decoding OOPSes under the 2.6 kernel. The kernel does it all for you if you build your kernel with **CONFIG_KALLSYMS** enabled. After this, the OOPS output may be transcribed directly from the screen to a bug report and sent to the linux-kernel mailing list. Although it's hard to find anything "convenient" about a kernel OOPS, this at least represents one big step saved for anyone reporting crashes to the kernel developers. The 2.4 tree will continue to rely on ksymoops for all OOPS decoding, and this is unlikely to change due to the late hour of 2.4 development. **Marcelo Tosatti** has finally begun to stave off additional features successfully, and we can expect him to grow ever more strict over time.

Pantelis Antoniou has written code to allow **AMD's Au1x00** embedded processor to support communication over the serial port, via the standard 8250 serial driver. The code was not actually so complex—the main oddities being that the Au1x00 has registers at different offsets than the 8250 serial driver expects, and this requires a mapping function to do the conversion. Modem status-change interrupts must be disabled for this hardware, because not all members of the Au1x00 family support it. These oddities have been #ifdefed out of the more standard parts of the 8250 driver, but the ultimate form of the code probably will be determined by the main kernel folks. Although #ifdefs are generally frowned upon, they also are still quite common in the kernel.

Jaya Kumar has coded up support for the **AMD Geode CS5535** audio device, and he has listed himself as the official CS5535 audio ALSA driver maintainer. This looks like an easy sell, with no one opposed, and only minor technical objections to overcome. **Andrew Morton** examined the driver himself and found no serious flaws. **SPDIF support** is planned, but Jaya as yet says he has no way to test the feature once it's implemented. The CS5535 was designed as a client device for the GX 533@1.1W processor, and other Northbridge components, to produce embedded appliances. Linux may use it this way or may find some twisted yet brilliant alternative.

The primary **kernel.org** server has moved to the **Oregon State University Open Source Lab**, where it has better

bandwidth, better backups and a staff to tend it. **Javier Henderson** flew the machine there himself, on a private plane, for minimal downtime. See osuosl.org/photos/kernel/view for photos of the touching event and to find out more about OSL. Several glitches had to be worked out after the move, including problems mirroring to the other kernel.org nodes, and one situation where different nodes ended up with different versions of git repositories, causing developers to see errors when they tried to sync with **Linus Torvalds'** tree. But these were ironed out quickly, and the new kernel.org setup seems to be working out very well.

The **Linux boot code**, like the boot code of most if not all operating systems, is complex and messy. And judging by a recent failed attempt to clean it up, this may be the state of affairs for a long time to come. **Etienne Lorrain**, who's been hacking the boot code since 1998, finally decided to rewrite it from twisted Assembly to sparkling clean C. Unfortunately, the only way he could see to do this was to abandon support for the LILO and GRUB bootloaders. True, Etienne added a number of nice features, such as eliminating any restriction on kernel size and providing boot-time access to the BIOS in a fully functioning state. But as **Pavel Machek** pointed out to him, "We have bad assembly bootup code. Adding good C bootup code, that is incompatible with LILO/GRUB, does nothing to clean the mess up."

Stephen Hemminger has set up a new **wiki for Linux networking documentation**, at linux-net.osdl.org. Originally started as a repository for his own work, Stephen opened the wiki up to all networking-related contributions. This has drawn some criticism from folks like **Greg Kroah-Hartman**, who point out that the wiki at wiki.kernelnewbies.org has existed for quite a while and would have welcomed contributions by Stephen. Stephen's reply to this is that there were Linux wikis before, and there would be more later. In fact, both wikis seem quite good, and there is always the possibility that they will merge later on.

—ZACK BROWN

The Power of Convergence...

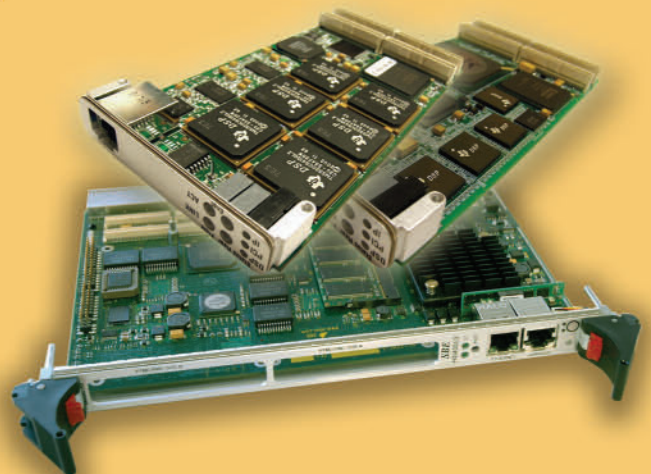


Reliability. Scalability. Mobility...on a gateway blade.

Open architecture drives costs down and performance up.

Modular VoIP "blades" based around standard, interoperable modules like PMC and AdvancedMC reduce costs by limiting the number of unique blades that telecom OEMs and carriers have to purchase and stock. A softswitch or media gateway controller can be deployed in a minimal configuration and scaled up later (to OC-3 and beyond) without replacing the whole blade and without taking it offline. SBE provides high-performance DSP resource modules that deliver premium carrier class voice processing with world-class features using Texas Instruments' DSPs with Telogy Software. In addition, these modules support transcoding and transrating to enable the integration of voice, video, data, and wireless.

SBE products are scalable from daughterboard modules to complex gateway blades, and provide telecom carriers/service providers with a choice of programmable voice platforms featuring SBE's line of network interface cards, ranging from T1 and T3 to Gigabit Ethernet and IPsec/SSL/WLAN acceleration. Full Linux support is available on every board.



Linux On Demand

flexibility on demand | 925-355-2000 | info@sbei.com | www.sbei.com

LJ INDEX FEBRUARY 2006

- | | | |
|--|--|---|
| 1. Number of Linux videos found by Google Video's public beta on November 8, 2005: 13 | 11. Millions of \$100 laptops planned for production late this year or early next: 10 | 19. Number of results in a WalMart.com search for "Linux": 314 |
| 2. Number of Linux videos found by Yahoo on November 8, 2005: 2,441 | 12. Millions of US dollars donated to the OLPC program by Red Hat, AMD, Google, News Corp. and Brightstar Corp., apiece: 2 | 20. Lowest-priced Linux in US dollars (Linspire) PC sold at WalMart.com: 219.84 |
| 3. Percent annual Linux growth rate: 25.9 | 13. Low estimate millions of kids expected to receive a \$100 laptop: 100 | |
| 4. Billions of US dollars in Linux business revenues in 2005: 20 | 14. High estimate millions of kids expected to receive a \$100 laptop: 150 | |
| 5. Projected billions of US dollars in Linux business revenues in 2008: 40 | 15. Price in US dollars reportedly offered by Steve Jobs for OS X, for the laptops: 0 | |
| 6. Number of members in OSDL: 80 | 16. Percentage of the world to which AMD plans to bring Internet and computing access by 2015: 50 | |
| 7. Number of members in the new Open Invention Network: 5 | 17. Total millions of PCs expected to ship in 2005: 200 | |
| 8. Number of members in the new LiPS (Linux Phone Standard): 10 | 18. Minimum laptop percentage of total 2005 PC shipments: 50 | |
| 9. Number of memers of CELF (Consumer Electronics Linux Forum): 53 | | |
| 10. The actual dollar price that children will be charged for the \$100 laptop from the One Laptop Per Child Program (OLPC): 0 | | |

Sources:

- 1: video.google.com
- 2: Yahoo.com
- 3-5: International Data Corp., via Open Invention Network
- 6: OSDL
- 7: Open Invention Network
- 8: LinuxDevices
- 9: CELF
- 10-16: *Wall Street Journal*
- 17, 18: theInquirer.net
- 19, 20: WalMart.com

—DOC SEARLS

They Said It

The best way to generate startup ideas is to do what hackers do for fun: cook up amusing hacks with your friends.

—PAUL GRAHAM, www.paulgraham.com/ideas.html

Whether or not you wish to argue about Freedom to code versus the cost of the software, what a lot of us are worried about is something even more intrinsic to the problem. We're worried about keeping the time that was spent at the front of the creation equation. We should be able to donate that time and not have it used against us later. We should be able to be a part of a community, and that simply means giving some things away. The GPL guarantees that our gifts will have the longevity they deserve.

—PAUL FERRIS, laxer.com/module/newswire/view/47217/index.html

My goal is to do all of the work it takes to be explaining to the Supreme Court in 2025 why broadcasting is unconstitutional.

—EBEN MOGLEN, www.forbes.com/business/2005/10/18/open-source-software-FCC_cz_df_1018opensource.html

Imagine what's going to happen if a law gets passed saying, in effect, "You put something on my computer without my knowledge, and that's breaking and entering"?

—PAUL FERRIS, laxer.com/module/newswire/view/47021/index.html

Some expect the Microsoft Windows market share, today well in excess of 90% worldwide, to erode in the coming years. Market share for Mac OS X is expected to remain flat, and demand for every other non-Microsoft desktop operating system is expected to dwindle. So Linux, already the fastest growing desktop operating system, is poised to continue making desktop inroads.

—MARK STONE, INTEL,

www.intel.com/cd/ids/developer/asmo-na/eng/240846.htm
(via Tom Adelstein)

EmperorLinux

...where Linux & laptops converge



You choose your laptop from a wide selection of top tier laptops manufactured by Dell, IBM, Lenovo, Sharp, and Sony. You choose your distribution from among the most popular Linux distributions. We'll install your distribution; add our custom kernel; then configure your laptop for full hardware support, including: X at the native resolution, wireless, power management, 3-D graphics, optical drives...



The Toucan: 5 lb Linux

- ThinkPad T series by Lenovo
- 14.1" SXGA+/15.0" UXGA
- X@1400x1050/1600x1200
- ATI FireGL graphics
- 1.6-2.26 GHz Pentium-M 7xx
- 40-100 GB hard drive
- 512-2048 MB RAM
- CDRW/DVD or DVD-RW
- ACPI suspend/hibernate
- Ask about our Sharp laptops



The Rhino: 7 lb Linux

- Dell Latitude D810/M70
- 15.4" WUXGA screen
- X@1920x1200
- NVidia Quadro or ATI Radeon
- 1.73-2.26 GHz Pentium-M 7xx
- 40-100 GB HD (7200 rpm)
- 512-2048 MB RAM
- CDRW/DVD or DVD±RW
- 802.11a/b/g wireless, GigE
- Ask about our Sony laptops



To: support@emperorlinux.com
From: customer@homebase.net
Subject: Configuration of 802.11a wireless

Just bought a new wireless card to learn how to configure it on my PC and my laptop so I can use it anywhere in my house.

And I still want it to work in my office, too.

Let EmperorLinux do the rest. Since 1999, EmperorLinux has provided pre-installed Linux laptops to universities, corporations, and Linux enthusiasts. We specialize in the configuration of Linux on the finest laptop and notebooks. We offer a range of the latest Linux distributions and boot options. All systems come with one year of Linux technical support and manufacturers' warranty.

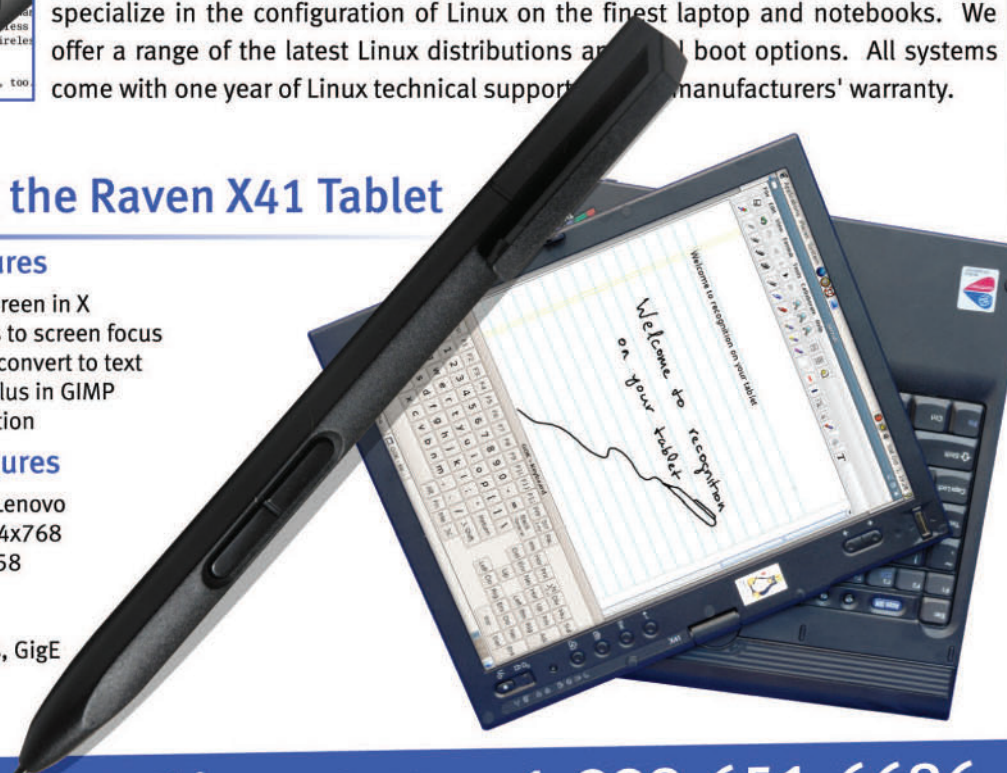
Introducing the Raven X41 Tablet

Linux Tablet Features

- Pen/stylus input to screen in X
- Handwrite commands to screen focus
- Handwrite notes and convert to text
- Pressure sensitive stylus in GIMP
- Rotate screen orientation

Linux Laptop Features

- ThinkPad X series by Lenovo
- 12.1" XGA w/ X@1024x768
- 1.5 GHz Pentium-M 758
- 60 GB hard drive
- 512-1536 MB RAM
- 802.11a/b/g wireless, GigE
- ACPI hibernate



www.EmperorLinux.com

1-888-651-6686

Model prices, specifications, and availability may vary. All trademarks are the property of their respective owners.

Assessing Ruby on Rails

All the dope on Zope versus Ruby on Rails

BY REUVEN M. LERNER

Several years ago, at the height of the dot-com boom, the phone was ringing off the hook with consulting work. My employees and I scrambled to fulfill all of the projects people were throwing at us. In the midst of this boom, it became obvious that nearly every project had similar characteristics, and that we were spending time (and clients' money) re-inventing the wheel with each new project. We began to look for ways in which we could reuse code, or at least techniques, across different projects. This, we assumed, would make us not only a more competitive business, but it also would make our day-to-day work more interesting. It is, after all, more interesting to work on the new and different elements of each project, rather than creating yet another user-group permission system.

We soon abandoned our plans for a common code system, in part because other developers had not only solved many of these problems, but also had released their solutions under an open-source license. And so over the years, we did a variety of different projects using Web development frameworks, many of which I have described in earlier editions of *At the Forge*.

But as anyone who has worked with such frameworks has learned, there is no free lunch. Nearly every framework tries to shoehorn you into doing things in a particular way, making its own set of trade-offs that might (or might not) fit the way you want to develop solutions. I have used a number of these frameworks over the years, and although I enjoyed various parts of them, I didn't feel like any of them allowed me to express myself the way I wanted.

I'm, thus, one of many developers who has become increasingly excited about a relative newcomer to the arena, known as Ruby on Rails. As we have seen during the last few months, Rails is a framework that provides a number of different functions, including an object-relational mapper, an MVC (model-view-controller) approach to design, an integrated templating system and built-in support for testing.

Rails has become extremely popular in the year or so since it was first released, and though it is still rough around some edges, the momentum is undeniable. Moreover, Rails has now become so popular that other frameworks are springing up, claiming to be Rails-like or with many features that are "just like Rails" or "better than Rails".

Why are so many people excited about Rails? More important, should you consider using it for your next Web/database

project? Finally, what trade-offs does it force developers to make, and how might these trade-offs affect your decisions?

Before Rails

I have been developing Web applications since the days when the phrase Web application described CGI programs that sent e-mail, rather than a billion-dollar industry. Every framework I have used has brought something to the table, and has made it easier for me to develop applications in one or more ways. At the same time, each frustrated me with the trade-offs I was expected to make in order to work with the system.

For example, Mason was one of the first Web development frameworks that I worked with, and it spoiled me with its flexibility and ease of use. Mason is written in Perl, and it is designed to work most easily with mod_perl and Apache. Installation and configuration have become trivially simple over the years, assuming you already have a working copy of Apache and mod_perl on your server. Also, Mason integrates beautifully with the many Perl modules available on CPAN, and with the mature and robust development tools the Perl community has created over the years. When I have to create an on-line system with Perl, Mason is definitely the first tool I turn to.

But what has always frustrated me with Mason is the small number of components that comes with the system. Sure, I could create a system for handling user accounts, and even for permissions and groups. But did I really want to write such code from scratch for every project I worked on? Moreover, although Mason's templates are highly expressive for developers, they include a great deal of Perl code and unusual constructs that can scare or surprise nontechnical developers.

I was thus drawn to OpenACS, an Open Source community system that has a significantly smaller following than Mason. However, the OpenACS templating system separated each viewed page into two components, one written in Tcl and the other in a modified form of HTML, with a specified "contract" between the two. In addition, OpenACS came with a standard data model designed to be used by all of the different applications in the system. You didn't need to worry about creating a registration module, because one came standard with the system. You also didn't need to create forums, Weblogs or calendars, because those also came in the standard system.

The centralized, standard data model and set of administrative applications were certainly appealing; however, OpenACS also had its problems. Perhaps the biggest one was the weird way in which OpenACS implemented its data model, using a relational database to keep track of hierarchies and objects. This system had a great deal of intellectual appeal; relational databases are fast, stable and cheap, and object-oriented programming has made it easier to model many types of data. But the marriage of the two meant that creating even a simple OpenACS application could be quite complicated. Moreover, as the OpenACS community grew, the data models became increasingly difficult to keep small, because everyone's needs were slightly different.

I also looked into Zope, a Web development framework written largely in Python. Zope has a large, strong community, and it continues to be developed and enhanced by Zope Corporation. Zope has many attractive features, including an extremely robust development environment, compartmental-

ized “products” that can be added and upgraded individually, and a sophisticated system of users, roles and permissions. Zope also pioneered the idea of object publishing, in which a URL describes the method that should be called on a particular object. Thus the URL /Foo/bar means that we’re invoking Foo.bar, passing inputs via the HTTP request and receiving any output via the HTTP response.

The most commonly heard complaint about Zope is that it is complicated to learn. This is somewhat true; it took some time before I found myself understanding the “Zope zen”, as it is known. In addition, many things I would expect to be straightforward require some coding acrobatics in order to work correctly—which might be a reflection on my coding style, but it also seems to be an artifact of how some Zope design decisions were made and the pervasive way in which objects are used within Zope.

Early on, Zope’s designers decided to avoid the problems associated with relational databases by building their own object-oriented database. On the one hand, this gave Zope a number of big advantages over its rivals, including the ability to undo changes to the system, built-in permissions and a storage system that mapped perfectly onto data types in Zope. But given the speed and pervasive nature of relational databases, SQL was also necessary. Zope thus provides the ability to connect to and work with relational databases, using a version of its DTML-templating language.

But this means that many Zope products—and certainly all of the products I have worked on—must coordinate the relational and object databases. This is generally a not-too-terrible way to handle things, but I always have ended up wondering why my life needs to be so complicated. And for all of its sophistication, I have often found myself creating the same types of create-update-delete methods and templates time after time.

Where Rails Fits In

So, it should come as no surprise that Ruby on Rails fills many of the gaps I have long perceived for Web developers. That said, my above descriptions also should make it clear where I think Rails should go if it is to continue to be successful.

One of the biggest draws for Rails is the speed and ease with which developers can create code that talks to a relational database. And although I am far from convinced by demos of what you can do in 15 minutes, my experience confirms that the demos are quite realistic. This is because Rails assumes you will create your database tables following its conventions, such as plural table names, ID fields named id and time/date fields ending with _at.


If you follow these conventions, you will discover you have to write a ridiculously small amount of code to handle many standard situations. Indeed, you probably will find yourself writing a handful of lines of code for many of the model objects you create, because the Active Record mapper within Rails will have done almost all of the work for you.

This means that much of the work needed for a Rails application is on the controllers (that is, objects whose methods are exposed via URLs) and views (that is, Ruby-HTML hybrid templates). Each controller method can produce its own output in plain text, HTML or via a template of the same name in the views directory. There is even a built-in Rails function for

sending a file to the user, allowing you to set up downloads of binary files without having to worry about the syntax for specifying MIME types and filenames.

Zope advocates undoubtedly will say that these latter features are available in Zope, and have been around for several years. This is true—but figuring out how to use them, and where they go, can be maddeningly difficult for newcomers. By providing reasonable defaults for a great number of activities, and then allowing developers to change those defaults, Rails manages to make the simple cases trivially easy, and the difficult cases only moderately hard. Moreover, the scaffolding generators included with Rails provide just enough of a basic, initial set of controllers and templates to get people going without having to spend hours creating and modifying various code files.

Because of these intelligent defaults, there are a limited number of objects and methods that a new Rails developer must master before starting to create an application. This stands in stark contrast with all of the other frameworks I described, which require understanding a fairly large number of objects and methods, as well as how they fit together, in order to work productively. True, Rails is growing in size and sophistication, and it runs the risk of gaining some of the bloat we see with more-established frameworks. So far, Rails has managed to avoid many of those problems and complications,



LAYER 42™

- Redundant UPS and generator
- Nationwide network
- Free tech support

<p>2U</p> <p>256kbps ~80GB</p> <p>\$60/mo.</p>	<p>4U or Mid-tower</p> <p>256kbps ~80GB</p> <p>\$80/mo.</p>
<p>1/4 Rack</p> <p>512kbps (14U) ~165GB</p> <p>\$200/mo.</p>	<p>1/2 Rack</p> <p>1mbps (28U) ~330GB</p> <p>\$350/mo.</p>

www.layer42.net

All prices include 100Mbps port, Firewall,
24x7 Monitoring and DNS hosting

408-450-5740
2312 Walsh Ave., Santa Clara, CA 95051

and the developers seem committed to keeping things as simple as possible.

Considering Rails

As I wrote above, each Web development framework has made its own set of design trade-offs. What is missing from Rails that would make it even better? What should you keep in mind when considering whether to use Rails for an application you are developing?

First, working with Rails requires an acceptance of the Ruby language. I had been looking at Ruby for some time before working with Rails, and I am increasingly enjoying it. However, undoubtedly many programmers will resent elements of Ruby, from the syntax to the object model. Ruby is also less mature than Perl and PHP when it comes to third-party add-on libraries, which means that you might need to write some specialized routines yourself, rather than rely on the community for support. And finally, Ruby lacks true Unicode support, which means that many multilingual Web sites will be unable to use it for the time being.

But if you are willing to consider Ruby as a language for Web development, should you use Rails? I would argue that the smaller the development team and the more ambitious the project, the more likely you are to benefit from Rails. Very small projects don't need the overhead that Rails requires, and they are probably best served with CGI programs and PHP. But the moment you need a relational database with more than one table, you're likely to benefit from working with Rails.

However, both Ruby and Rails are designed for small teams of programmers, and even programmers who are working alone. If more than one person is going to work on a Rails project successfully, it will require great discipline on the part of the programmers to ensure that no one modifies files that are in someone else's purview. The fact that each Rails application resides in a single directory might increase the likelihood for such confusion.

Large projects, thus, might benefit from some of the larger frameworks, such as Zope—or even one of the many Java-based projects that has been released in recent years. My biggest hangup with Java is that it is relatively cumbersome and slow, especially when compared with languages such as Perl, Python and Ruby. But when you are working on a large project with many other programmers, it might be an advantage to have more compile-time checking, explicit declarations and safeguards that are missing from languages such as Ruby.

You also might want to consider the sophistication of your Web designers when thinking about Rails as a platform. Some relish the idea of working with code within templates, and others are scared of it and might even erase or change code. I still think that ZPT and OpenACS templates are a better system of templates, so I was encouraged by the recent announcement of Liquid, a templating system for Rails in the style of PHP's Smarty. I have been impressed by Smarty in the past, and I think this might help speed the introduction of Rails into large, established Web shops.

The fact that each Rails application uses a single directory of plain-text files has both advantages and disadvantages. One major advantage is that everything can be stored easily in CVS or a similar version-control system; installing the application in a new location can be as simple as checking out the code.

However, this approach means that it's a bit harder to have multiple instances of the same application running on the same server, as with OpenACS packages and Zope products. We always can create multiple copies of the Rails application's directory tree, but it doesn't seem possible to have multiple instances of the same package.

As I mentioned above, I was originally quite attracted to OpenACS because of its single, standardized data model. I now understand that such a heavy, centralized data model is almost always going to be inadequate, but I still have to wonder why Rails doesn't come with any generic set of built-in permissions or registration. The answer, I suppose, is the growing number of Rails plugins, among which are several registration systems that can be integrated into existing Rails applications. I still would prefer to see more standardization on this front, but that is probably a lost cause at this point.

Finally, one problem Rails shares with every other environment is that of legacy code. Rails is so new, and so different, does it mean that its adoption will force us to abandon what we have already done? Possibly, but not necessarily. Rather than rewrite a mature Perl library in Ruby, I simply wrote a wrapper using XML-RPC. Ruby has an easy-to-use XML-RPC client, which I used within my Rails application to contact the Perl code. This has worked smoothly and easily, and it means I can benefit from Rails and CPAN at the same time. The fact that Rails lets developers override its database naming conventions also means it can be used with existing databases, rather than force users to create new database schemas that conform with Rails conventions.

Conclusion

Some people are hailing the arrival of Rails as the beginning of a new era in Web development. And indeed, I think Rails has set a new standard for what we can expect in a Web development framework. No longer will developers believe that it should take more than a few lines of code to create a "hello, world" program, or even to handle basic database actions.

Also, Rails is starting to convince developers that common conventions can be conducive to rapid, bug-free development. It took many years for developers to agree that garbage-collected languages were an improvement over malloc(), and it is taking a similarly long time for us to agree that conventions are better than configuration files. But the popularity of Rails probably means that we are increasingly ready for such a change.

Although no Web development framework is perfect, I believe that Rails has hit the sweet spot for many of the applications I have found myself writing for more than a decade. Both Ruby (the language) and Rails (the framework) are still maturing—but if this is how they are as relatively immature tools, I can't wait to see what they're like when they are finally ready.

Resources for this article: www.linuxjournal.com/article/8693.

Reuven M. Lerner, a longtime Web/database consultant, is currently a PhD student in Learning Sciences at Northwestern University in Evanston, Illinois. He and his wife recently celebrated the birth of their third child, a boy.



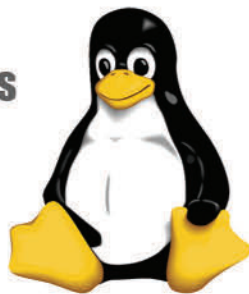
Thinkmate server and workstation products offer more configurations and more customizable options than any other system builder.



Thinkmate 1U Server, 2x Serial-ATA or SCSI drives with Intel® Xeon™ Processors

Dozens of customizable systems online, unlimited possibilities by phone.

- ▶ Rackmount Servers
- ▶ High-Performance Servers
- ▶ Storage Servers
- ▶ Pedestal Servers
- ▶ Silent Workstations
- ▶ Blade Servers
- ▶ Clustering
- ▶ Notebooks



Operating System Options:

Thinkmate systems are available with either No OS, or pre-loaded with Linux or Microsoft Windows operating systems. Thinkmate also offers dual-boot and virtualization options.

100% True Hardware Customization:

Thinkmate is an innovative provider of an extensive variety of computer solutions. We completely customize all of our machines to match your individual needs. Our online quoting and ordering system has more customizable options than any other system builder on the web. If you can't find exactly what you need on our site, then give us a call and we would be more than happy to help you find it!

Service:

Thinkmate takes customer service to a new level. All of our systems have a minimum of a 3-year advanced replacement warranty and offer up to a 3-year next business day onsite warranty through IBM Global Services. We understand mission critical situations and provide superior services to keep all of our customers satisfied.

GSA Scheduling:

We offer rapid GSA scheduling for custom configurations. If you have a specific hardware requirement, we can have your configuration posted on the GSA schedule within 2-4 weeks.



THINKMATE

(800) 371-1212

WWW.THINKMATE.COM

Little Bitty Applications

Even a panel of experts would agree these panel tips are priceless. **BY MARCEL GAGNÉ**

You certainly have a lot of tools at your fingertips, François, but don't you think you might be overdoing things just a little bit? Of course, I'm proud that you have taken to this subject so quickly, but having only a portion of your screen left on which to run applications, such as a Web browser or word processor, kind of limits its usefulness, don't you think? The whole point of applets is that they are small, but you have so many running, there is little room left for anything else.

No matter, François, we'll discuss this later. In fact, embedded applications such as these fit in very nicely with tonight's menu. But now we need to get ready as our guests will arrive at any moment. Too late, François. They are already here. Welcome, everyone, to *Chez Marcel*, home of exquisite wines and the finest in Linux fare. Please, sit and make yourselves comfortable. François, hurry down to the wine cellar and bring back the 1999 Brunello di Montalcino from Tuscany. Check the southeast wing of the cellar, right past the Chianti.

While my faithful waiter fetches the wine, let me tell you about today's selection. Earlier on, I was demonstrating the power of those small applications you find embedded in the panel at the bottom of your graphical desktop. These little programs are called applets, and many of them are engineered to fit nicely into your taskbar while still providing useful functionality. Many people tend to think of this as the taskbar, but in reality, the taskbar is just one of many applications embedded in that panel. Other embedded applications include the task switcher, the menu, the clock and that system tray itself.

Before we continue, I should mention that KDE users generally have one, all-encompassing panel at the bottom of their screens, while GNOME users often have two panels running: one at the top and one at the bottom of their screens. I cover both desktops today, starting with KDE. These helpful and common applets (the taskbar, panel and so on) aren't the only ones that came with your system. In fact, there are several just waiting for you to try.

Adding an applet to your KDE panel is easy. Right-click on the taskbar, and move your mouse over the Add to Panel menu, then to the Applet submenu (Figure 1).

A rather large collection of applets appears from which you can make your selection. Click on the applet of choice, and it appears in your KDE panel. I will now confess my love for these little applets in general, but none more so than the dictio-

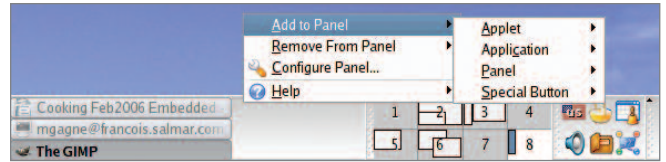


Figure 1. Additional applets can be added from the KDE panel by right-clicking on the panel.



Figure 2. I consider the dictionary and weather applets to be absolutely essential.

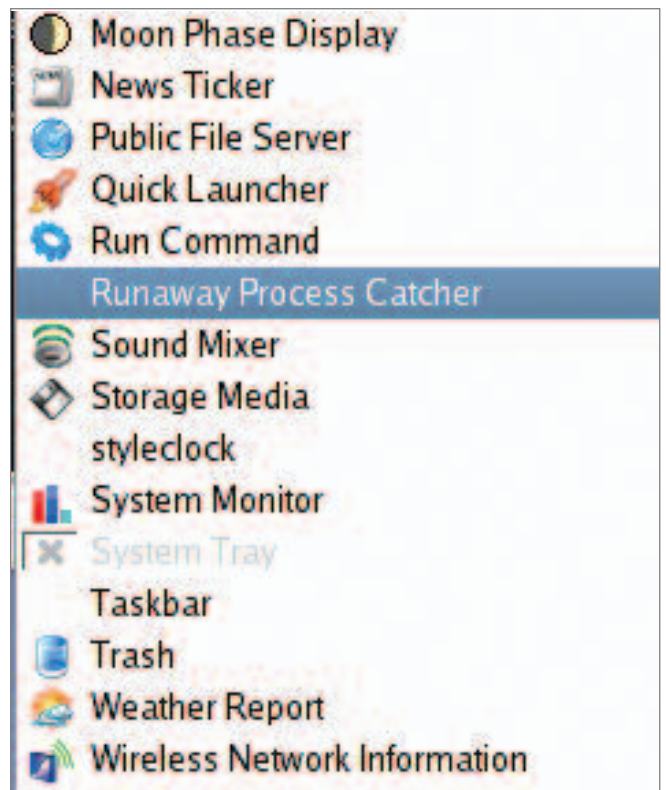


Figure 3. This is just a fraction of the many applets included with KDE.

nary applet (Figure 2). When I configure a new system for myself (or anyone else for that matter), I invariably add the dictionary applet. I can't imagine the panel without this incredibly useful little tool. Simply type in a word, press Enter and the program searches various on-line dictionaries for a definition.

Although this may be my favorite applet, I find many others are very useful to have. Like most people around here, I tend to obsess about the weather, and so I run the KDE weather applet (cloudy and 5°C, if you're curious). Right-click the applet, select Configure KWeather and select your location from the city list. The color picker is particularly useful if you are doing a lot of graphics work and you want to capture any color on the screen, even from Web pages. Then there's the moon phase applet—nothing like knowing that it's only six days till the next full moon. If you are feeling particularly geeky, add a binary clock. You'll find a system monitor as well. Take time to try each of them out. There are certainly plenty to choose from (Figure 3).

Find out why some of the world's largest internet sites choose Open Source Storage for their enterprise data center needs.

Servers

Stability

Storage

Support

Savings

Speed



The VS1200D server includes the following items below:
2x(2P/4C) Dual-core AMD Opteron™ processors Model 265
Server Motherboard with:
 On-board Video
 Dual Gigabit LAN
 Supports SATA RAID 0/1
2x 1GB PC3200 ECC REG Memory (2GB total)
1x 80GB SATA Hard Drive
2x Hot-Swap Drive Bays
1U Black rackmount case
3 Year Parts and Labor Warranty
Free On-Site Support within Silicon Valley, CA
30 Day Money Back Guarantee
\$2699*



The AMD64 Platform

Specializing in high performance
computing and complete
data center implementation



©2003 Advanced Micro Devices, Inc.

1-866-664-STOR
1900 Wyatt Dr. Bldg.16&17
Santa Clara, CA 95054
<http://www.OSShpc.com>
*(Subject to availability).


**Open
Source Storage**

©2002-2005 Open Source Storage, Inc. All Rights Reserved. AMD, AMD Opteron, combinations thereof, are trademarks of Advanced Micro Devices, Inc.

PGI[®] Compilers are building the 64-bit applications infrastructure.

PGI Fortran, C and C++ compilers deliver world-class performance on a wide spectrum of 64-bit scientific and engineering applications. With PGI you get an easy-to-use integrated suite of dual-core and MPI-capable compilers, debugger, and profiler to simplify porting and tuning of 64-bit applications for AMD64 and EM64T processor-based workstations, servers and clusters. With comprehensive cross-platform support for Linux and 64-bit Windows operating systems on both Intel and AMD processors, PGI delivers a uniform development environment across your key target systems. The leading independent software vendors in structural analysis, computational chemistry, computational fluid dynamics, and automotive crash testing have chosen PGI compilers and tools to build and optimize their 64-bit applications.

Visit www.pgroup.com to learn what PGI Compilers and Tools can do for you.



The Portland Group[™]
www.pgroup.com ++ 01 (503) 682-2806

64-bit STAR-CD Compiled With PGI

Eventually, you may find that your panel is getting a bit crowded with little room for tasks or even smaller system tray applets. What are you to do when you fill up your panel? One option is to remove applets you aren't using. To do so, right-click on the panel, go to the Remove From Panel menu and select Applet from the submenu.

If, however, you believe that too much of a good thing is just about right, you may not want to remove your applets. Happily, you don't have to. Simply add a child panel. Incidentally, this is what François and I were discussing when you arrived—he had one too many panels and a very cluttered desktop. Because I know you will all show restraint, I'll tell you how it's done. Right-click on the panel and move your mouse to the Add to Panel menu and then to the Panel submenu where you'll find several options, including another Panel. Click here and a plain, gray Panel appears on your screen directly above your current panel. You now have several options, one of which is to start populating this child panel with even more applets. There are other things you can do though, and I would like to tell you about a few.

First and foremost, you may not like the position of the new panel. To move it, simply click and drag the panel to another place on the screen. Your options are along the sides, at the top of the screen or below the current panel. Another possibility is to right-click and select Configure panel. This approach also lets you decide on the size of the panel.

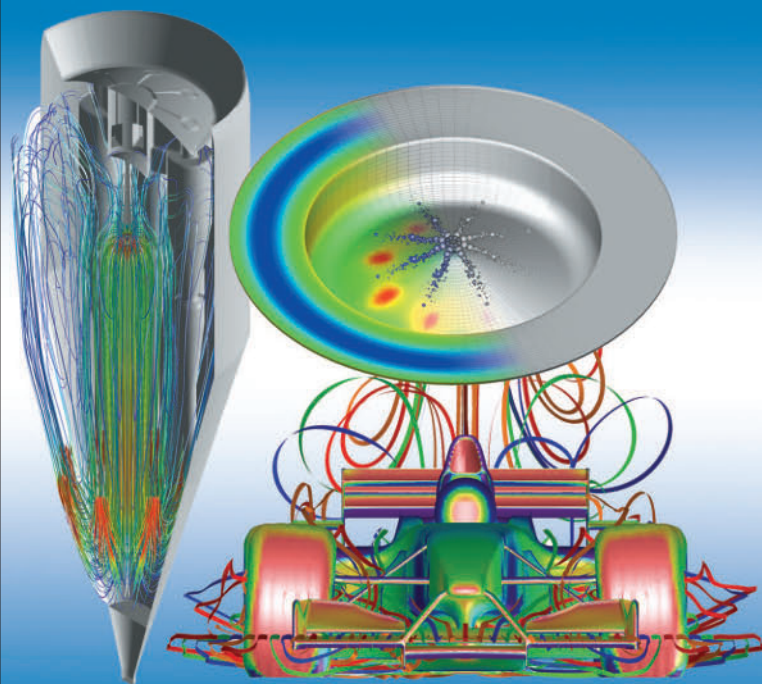
GNOME users certainly aren't left out of the applet madness. As I mentioned earlier, the concept of a panel with embedded applications exists here as well. Although we could work with either (or both) of the GNOME panels, I concentrate on the bottom one for now.

The default look and size of the panel varies somewhat depending on the distribution or release. My GNOME 2.12 panel had a default size of 24 pixels. Depending on the nature of the program, applets embedded in a panel of this size tend to be a little hard to look at. Should you find yourself in the same situation, right-click in a blank area of the panel and select Properties from the pop-up menu. Once you do this, the Panel Properties dialog appears (Figure 4).

In my experience, and on my monitor, 48 pixels is pretty much ideal. However, how large you choose to make the panel is entirely up to you and may vary with the amount of wine you drink. Just a little joke, *mes amis*, but an excellent idea, *non?* François, kindly refill our guests' glasses.

Now that you've prepped your panel, it's time to find and add some of those applets. Right-click on the panel and a small menu appears. Click Add to Panel and a list of all the available applets appears (Figure 5). This list may be one single list with a short description of the applet in question, or it may be organized into categories. Scroll down and find something you like, then click the Add button. For my first applet, I chose the one labeled Fish, an animated swimming fish that pops up random bits of cleverness using the `fortune` command. And, yes, the fish's name is Wanda. *Non, mes amis*, I did not make this up.

In a few seconds, your applet will appear in the panel. When an applet starts, it rarely starts in exactly the position I want. Luckily, moving it is an easy process. Simply right-click on the applet and a small menu appears (Figure 6). One of the options is Move. Click here and a small hand icon appears that allows you to drag the applet to wherever you want it to live. You even



STAR-CD delivers leading CFD technology with a choice of STAR-Design or CAD-embedded options, enabling engineers to build models easily with their own choice of plug-ins. These models can be imported into the pro-STAR GUI for advanced CFD analysis using the full capabilities of STAR solvers. STAR-CD's multi-level, full spectrum approach enables enterprise-wide CFD strategies and encourages collaboration between design, application and R&D engineers. Visit www.cd-adapco.com to learn more about STAR-CD simulation solutions for 64-bit systems.

STAR-CD software is built for AMD Opteron and Intel EM64T processor-based systems using *PGI Compilers and Tools*.



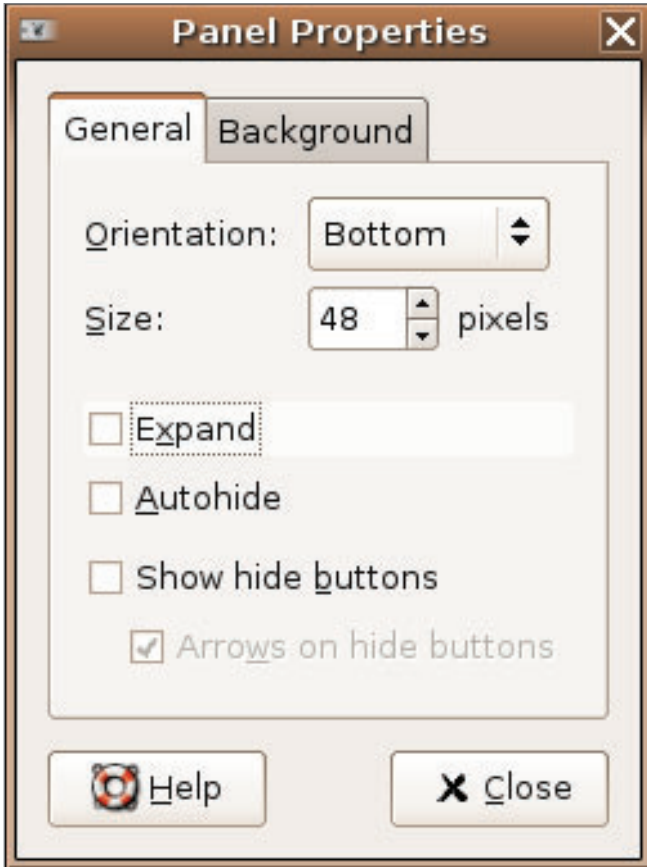


Figure 4. To make some of these applets useful, you may want to increase the default panel size.

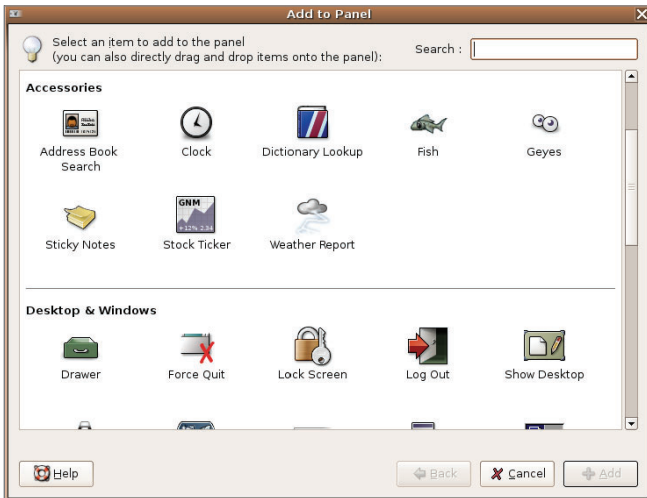


Figure 5. Adding an Applet to the GNOME Panel

can move it to another panel (more on that in a moment).

Have another look at that menu, and you'll see that there are some other useful options here. For instance, each application may have its own set of configurations. The fish applet I had you install comes with a small handful of additional graphics—you may not like fish. Simply click on the Preferences menu to make your changes. Finally, should you decide that the program you chose is just taking up space, there's a Remove option.

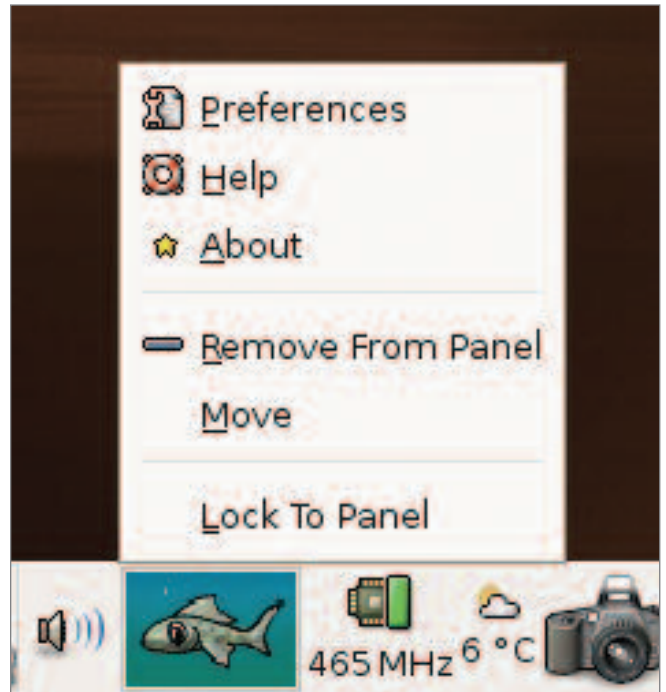


Figure 6. Each applet can be configured, moved or removed.

When applet mania has taken hold, you will find yourself out of space. Yes, it is time to add another panel. Right-click somewhere in your panel's empty space. From the pop-up menu, select New Panel. A blank, gray panel appears that you can drag to any location you see fit. As with the original panel, selecting the Properties dialog lets you change the size to fit the applets you have in mind.

On the GNOME applet side, I have some of the same favorites, although they are not entirely the same in terms of functionality. The dictionary applet is a must, as is the weather reporting program. The fortune fish (above) is fun, as are the eyes that follow your mouse pointer around the screen. The Take Screenshot applet is particularly useful when you are doing documentation. Because I work on a variety of systems, I'm also rather fond of the Terminal Server Client and the Connect to Server applets.

Once again, *mes amis*, the clock on the wall would seem to imply that we have once again filled up our allotment of time. Surely, there must be some way to add another few hours to our day as easily as adding another panel. Until then, we have my faithful waiter, François, ready to refill your glasses one final time before before we say, "Au revoir". Please raise your glasses, *mes amis*, and let us all drink to one another's health. *A votre santé! Bon appétit!*

Marcel Gagné is an award-winning writer living in Mississauga, Ontario. He is the author of the all new *Moving to Linux: Kiss The Blue Screen of Death Goodbye!* 2nd edition (ISBN 0-321-35640-3), his fourth book from Addison-Wesley. He also makes regular television appearances as Call for Help's Linux guy. Marcel is also a pilot, a past Top-40 disc jockey, writes science fiction and fantasy, and folds a mean Origami T-Rex. He can be reached via e-mail at mggagne@salmar.com. You can discover lots of other things (including great Wine links) from his Web site at www.marcelgagne.com.



MBX is the Industry Leader for Server Appliances

From design to delivery we are dedicated to building a partnership with you.



MBX RP-1013 Platform

- Intel® Celeron 336 Processor at 2.8 GHz
- 1U Rackmount Chassis 16.5" Deep
- 512MB PC4200 DDR2 Memory
- Maxtor 80GB Serial ATA Hard Drive
- Eight Gigabit NIC's, 4 ports with by-pass
- Optional 16x2 LCD with Keypad
- On-board Compact Flash Socket
- Brandable With Your Color and Logo
- Custom OS and Software Install
- No Minimum Quantity Required
- 3 Year Warranty

\$1,499 other configurations available
please call for pricing



MBX RP-1110 Platform

- Intel® Celeron 336 Processor at 2.8 GHz
- 1U Rackmount Chassis
- 512MB PC3200 DDR Memory
- Maxtor 80GB Serial ATA Hard Drive
- Dual On-board Gigabit NIC's
- Custom OS and Software Install
- No Minimum Quantity Required
- 3 Year Warranty

\$899 other configurations available
please call for pricing

MBX is the leader in custom appliances. Many premier application developers have chosen MBX as their manufacturing partner because of our experience, flexibility and accessibility. Visit our website or better yet, give us a call. Our phones are personally answered by experts ready to serve you.



www.mbx.com
1-800-681-0016

Conditional Statements and Flow Control

IF you need conditions in your scripts, THEN this is the column for you. **BY DAVE TAYLOR**

The last of the basic building blocks of shell scripting are conditional statements, allowing you to decide programmatically whether to execute a block of statements based on a logical test, and flow control statements, the great innovation from the earliest days of programming when you could have a block of code execute more than once. We explore both of these in this column and, finally, are done with the proverbial Lego blocks of scripting, allowing us to start exploring how to solve complex scripting problems with novel and unique combinations of simple statements.

The most obvious conditional statement is if-then-else, which in shell scripting looks like:

```
if condition ; then ; statements ; else ; statements2 ; fi
```

Of course, you'd usually see this on multiple lines, so it's more likely to look like this:

```
if condition; then
    statements
else
    statements2
fi
```

There are some variations on this, including safely omitting any sort of else clause, but more interestingly, you can "chain" conditionals together with an else if structure:

```
if condition; then
    statements
elif condition2 ; then
    statements2
fi
```

That's perfectly valid and, worth noting, functionally different from the structure:

```
if condition
then
    statements
    if condition2; then
        statements2
    fi
fi
```

The difference will be obvious to anyone who has programmed before. In the first example, statements2 would execute if condition were false and condition2 were true. In the latter example, however, statements2 would be executed only if condition were true and condition2 were true. Subtle, but very important!

Specific logical conditions can take on a wide variety of appearances, because the only requirement for a conditional expression is that it returns zero if the evaluated condition is false and nonzero if it should be considered true. Indeed, there are commands in Linux called false and true, so you can use statements like "if true; then...." Most conditions, however, are built around the invaluable test command, with its many different flags and options.

Want to compare two string (text) values? You could use:

```
if test $myvar = "exit" ; then
```

or its shortcut alternative of:

```
if [ $myvar = "exit" ] ; then
```

Compare two numeric values with:

```
if test $numval -lt 10 ; then
```

There's also a world of file and variable tests available in the test command too, including -r to test if a file is readable, -e to see if it exists at all, -s to see if the file exists and has a nonzero size, -d to test for a directory and -f to test for a regular file.

So if you want to differentiate whether \$filename is a file, directory or other file type, you could use a statement sequence like:

```
if test -f $filename ; then
    echo "$filename is a regular file"
elif test -d $filename ; then
    echo "$filename is a directory"
else
    echo "$filename is neither a file nor a directory."
fi
```

Check out the test man page (use `man test`) to read about all the many different conditionals you can use in a shell script.

Flow Control

There are a number of different looping and flow control structures above and beyond simply the if-then-else conditional,

luckily, and here are the big three:

- `for x in y; do; statements; done`
- `while x; do; statements; done`
- `case x in ; condition1) statements ;; condition2) statements ;; esac`

There are more conditional statements, but you'll find that in the vast majority of cases, having for loops, while loops, case statements and if-then-else statements will serve as the building blocks of even the most complex script.

The for loop is particularly useful in its variations. Want to step through the parameters given to the shell script itself? Use something like this:

```
for value ; do ; statements ; done
```

Want to step through a set of matching filenames for a given pattern? Here's how to do that in a script:

```
for filename in *.c ; do
  statements
done
```

Let's look at how a couple of these can be combined in useful ways, rather than just duplicate the man page, however. Here's a simple script that examines each entry in the current directory, indicating whether it's a file or directory:

```
for name in *
do
  if [ -f "$name" ] ; then
    echo "$name is a file"
  elif [ -d "$name" ] ; then
    echo "$name is a directory"
  else
    echo "$name is neither a file nor directory"
  fi
done
```

For illustrative purposes, let's try another version of this script, one that recognizes *.c as C source files, *.h as included header files and *.o as intermediate object files, but this time we'll use the case statement:

```
for name in *
do
  case "$name" in
    *.c ) echo "$name is a C source file" ;;
    *.h ) echo "$name is a header file" ;;
    *.o ) echo "$name is an object file" ;;
  esac
done
```

From a readability perspective, the case statement is hard to beat!

Wrapping Up

There are, of course, many different ways to create more-advanced and sophisticated scripts, notably including shell script functions, but we'll delve into those as we proceed. I'm a big fan of just doing rather than talking around the topic forever.

I hope that's enough on the basics of flow control and conditional expression evaluation in this column. If you've some questions, don't forget that `man sh` produces more information on the power and capabilities of the Bourne Shell.

I don't know about you, but I'm eager to get moving on to some more complex and interesting scripting tasks, and I invite you to let me know via e-mail if there are specific types of scripts that you're interested in seeing featured here.▀

Dave Taylor is a 25-year veteran of UNIX, creator of The Elm Mail System and most recently author of both the best-selling *Wicked Cool Shell Scripts* and *Teach Yourself Unix in 24 Hours*, among his 16 technical books. His main Web site is at www.intuitive.com.



Know IT All™

System and Storage Asset Inventory and Configuration Tool

v6.0 Now Available

What: Hardware, Software, Storage, Licenses, OS, Network, and more!

OS: Linux, MacOS, Windows, AIX, FreeBSD, HP-UX, IRIX, Solaris

Storage: NetApp, EMC Symmetrix, EMC CLARiiON, Veritas, and more!

Full CLI, GUI, Perl/C/Shell API

Free Demo Download!
www.MagniComp.com/lj1

MagniComp™

Single Sign-On and the Corporate Directory, Part III

Combine Samba with OpenLDAP for a mail and SSH single sign-on system. **BY TI LEGGETT**

Welcome to the third installment of how to implement a single sign-on and corporate directory system. In this article, we tackle integrating Microsoft Windows clients. There's a lot involved to make it all happen, so put on your work gloves and let's get to it.

When you want to integrate Windows clients into a heterogeneous environment, you have some choices to make. Although you can run an Active Directory (AD) server and have your Linux and Apple clients bind to it for authentication and identity management, the costs involved are not minimal. It also wouldn't make for an interesting article on an open-source single sign-on and directory implementation.

When you're binding Windows clients to an open-source solution, you have two more choices to make. Do you bind them to the Kerberos realm for authentication or do you bind them to LDAP for identity management? This is an either/or choice because although Windows clients know how to speak both Kerberos and LDAP, they know how to speak them at the same time only when talking to an AD server. In other words, Windows clients can talk to a non-AD Kerberos server only when the user's identities are kept locally. Likewise, a Windows client can get identities from LDAP via Samba, but only when the passwords are also served via Samba, and Samba can't, at the moment, authenticate via Kerberos.

Having Windows authenticate against our Kerberos KDC is easier to set up, but it could be harder to maintain because every user who uses the Windows client needs to have a local account. This is fine if all you have is one Windows client to maintain, but if you have any more than that, you'll need to add every user to every client. I won't explore this option; however, if you're interested you should pick up Jason Garman's *Kerberos: The Definitive Guide*.

Configuring Samba

Because we're dealing with a corporate directory, I'm assum-

ing you probably have more than one Windows machine on your network. In order to make using them and incorporating them as painless as possible, we use Samba tied to our LDAP directory as a back end. Even though we'll be configuring Samba a little differently, you should first read Craig Swanson and Matt Lung's "OpenLDAP Everywhere Revisited" (see the on-line Resources), as it will give you a good foundation on which to build. I created an organizational unit branch in the directory named samba for Samba-specific entries such as machines and ID maps. Listing 1 shows the hierarchy of these special branches, and Listing 2 shows the LDIF for them.

Listing 1. Additional Organizational Units

```
+ o=ci,dc=example,dc=com
|- ou=samba
   |- ou=hosts
   |- ou=idmap
```

Listing 2. LDIF for Additional Organizational Units

```
dn: ou=samba,o=ci,dc=example,dc=com
objectClass: organizationalUnit
ou: samba

dn: ou=hosts,ou=samba,o=ci,dc=example,dc=com
objectClass: organizationalUnit
ou: hosts

dn: ou=idmap,ou=samba,o=ci,dc=example,dc=com
objectClass: organizationalUnit
objectClass: sambaUnixIdPool
uidNumber: 15000
gidNumber: 15000
```

I don't use the smbldap scripts from IDEALX for creating necessary entries, because I'm using LDAP for more than just Samba authentication. One main reason for not using the smbldap tool is because it assumes that it and Samba will be the only point for actions such as adding users and groups. In my environment, all users don't have the ability to log in to Windows machines. Some users may start off as Linux-only users, but then need to be given access to Windows machines later. The smbldap tools don't handle this case very well. However, the smbldap tools do handle other things nicely, so like all things, investigate all the tools available and choose the best one(s) suited to your needs.

We need several users in LDAP that will do various tasks. First we need a user who has write access to certain pieces of the directory. If you notice in `/etc/samba/smb.conf`, there is an option, `ldap admin dn`, that defines the DN of this user. This user, named `samba_server`, should be stored in the LDAP directory itself, and it will be the only user in the directory with a password associated with it. Because this user isn't of the `posixAccount` objectClass, the account is not recognized under Linux. To create this user, first run `slappasswd` to gener-

Do you take

"the computer doesn't do that"

as a personal challenge?

So do we.

LINUX
JOURNAL™

Since 1994: The Original Monthly Magazine of the Linux Community

Subscribe today at www.linuxjournal.com

Listing 3. LDIF for the Samba User

```
dn: uid=samba_server,ou=people,o=ci,dc=example,dc=com
objectClass: top
objectClass: person
objectClass: uidObject
sn: samba_server
cn: samba_server
userPassword: {SSHA}xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
uid: samba_server
```

Listing 4. LDIF for the root Account

```
dn: uid=root,ou=people,o=ci,dc=example,dc=com
objectClass: account
objectClass: sambaSamAccount
cn: root
uid: root
displayName: root
sambaSID:
➔S-1-5-21-2162541494-3670296480-3949091320-1000
sambaPrimaryGroupSID:
➔S-1-5-21-2162541494-3670296480-3949091320-512
sambaNTPassword: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
sambaAcctFlags: [U          ]
sambaLogonTime: 0
sambaLogoffTime: 2147483647
sambaKickoffTime: 2147483647
```

Listing 5. LDIF for the Regular Samba User

```
dn: uid=leggett,ou=People,o=ci,dc=uchicago,dc=edu
objectClass: top
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: sambaSamAccount
cn: Ti Leggett
givenName: Ti
sn: Leggett
mail: leggett@mcs.anl.gov
uid: leggett
uidNumber: 1001
homeDirectory: /home/leggett
loginShell: /bin/bash
gidNumber: 1000
sambaSID:
➔S-1-5-21-2162541494-3670296480-3949091320-3002
sambaPrimaryGroupSID:
➔S-1-5-21-2162541494-3670296480-3949091320-512
sambaNTPassword: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
sambaAcctFlags: [U          ]
sambaLogonTime: 0
sambaLogoffTime: 2147483647
sambaKickoffTime: 2147483647
```

ate the hashed password. Then, take the hash and create an ldif file that's similar to Listing 3.

Next, we need to tell Samba how to access the LDAP directory as samba_server user by using the smbpasswd command:

```
# /usr/bin/smbpasswd -w <password>
Setting stored password for
➔"uid=samba_server,ou=people,o=ci,dc=example,dc=com"
➔in secrets.tdb
```

For added security, you should turn off your shell's history logging as the password is given on the command line. The smbpasswd command takes the password given and stores it in /var/lib/samba/private/secrets.tdb keyed to the Samba domain and the admin dn, so if either of those values change, you need to rerun smbpasswd.

Because Samba uses this user to query and modify values in the directory, we need to allow the Samba admin write access to certain attributes in the directory, so make sure to add the appropriate ACLs to /etc/openldap/slapd.conf.

At this point, we can get the SID for our domain. To obtain the domain's SID, you need to be root on the primary domain controller (PDC) for the domain, and run:

```
# net getlocalsid
SID for domain CI-PDC is:
➔S-1-5-21-2162541494-3670296480-3949091320
```

If you won't be using the smbldap tools to create all of the Samba LDAP entries, you need to use this SID when creating those. I've included a sample LDIF containing all the entries you need to create in the on-line Resources.

Samba also needs a user with uid 0 in the LDAP directory temporarily to perform certain actions. The entry need not be a full posixAccount user, but it should look like Listing 4.

Notice that this user entry does have an NT password, but this password need not be the same as the actual root password, and it's only temporary to get rights assigned to normal users. I've included a simple Perl script in the on-line Resources that you can use to generate the NT password hashes that are needed. You need the Crypt::SmbHash and Term::ReadKey Perl modules to use it.

The last user to modify is your own, so that it's recognized as a Samba user and is a Domain Admin. Listing 5 shows the LDIF.

A Note about SIDs and RIDs

The SID for your domain is a unique key and is prepended to domain entities. The RID is a unique identifier for identities in the domain and is appended to the SID to make the sambaSID. The algorithm I follow to make sure RIDs are unique is to take the uid for a user, multiply it times 2 and add 1000. For groups, take the gid, multiply it times 2 and add 1001. This ensures that users all have even RIDs and groups have odd RIDs. The only exception to this rule are the Domain Admins, Domain Users and Domain Guests. The RIDs for these groups are the same as their gid for consistency with how Windows assigns them.

Reclaim lost time



Levanta Intrepid™

The world's first Linux management appliance

Plug the Levanta Intrepid™ into your network and perform the most important Linux management tasks in a fraction of the time you spend now. And gain power and flexibility that you've never had before:

- **Fast & Portable:** Provision servers or workstations practically anywhere, anytime – in minutes. Swap them around, mix it up.
- **Flexible:** Supports commodity hardware, blades, virtual machines, and even mainframes.
- **Out of the Box:** Includes pre-defined templates for servers, workstations, & software stacks. Or create your own.
- **Total Control:** Track any file changes, by any means, at any time. And undo them at will.
- **Disaster Recovery:** Bring dead machines quickly back to life, even if they're unbootable.

Based upon technology that's already been proven in Fortune 500 enterprise data centers. Now available in a box, priced for smaller environments. **Just plug it in and go.**

30-Day
Money-Back Guarantee
Order online by 2/28/06
Get \$500 Off

Enter PROMO CODE: LJ0206


LEVANTA
www.levanta.com
1.877.LEVANTA



You'll notice that this account also has an NT password in the LDAP directory. Unfortunately, as of this writing, Samba has no stable support for using Kerberos authentication unless Samba is authenticating against an AD server. There are ways, however, to store Kerberos principal data in an LDAP directory if you're using the Heimdal Kerberos implementation. This potentially could make Samba authentication a little bit cleaner, though it won't make your Samba domain an AD server. Because we're not using Heimdal and this isn't officially supported, we must store Samba passwords in the directory. I've provided some links in the on-line Resources on the Kerberos/LDAP solution if you're interested.

We're now ready to start Samba, but make sure you also start the winbind service as well. Under Gentoo, modify /etc/conf.d/samba.

Samba Privileges

With Samba v3.0.11, the notion of privileges was introduced. Prior to this version, a network-accessible uid 0 user was required for all user, group, machine and printer management. As of v3.0.11, a user with the proper privileges can initiate these types of requests. A uid 0 user is still necessary eventually for some of these, but it no longer need be network-accessible. So, you might be wondering why we added a uid 0 account to the directory. Well, there's a bit of a chicken-and-egg problem when initially setting up Samba. In order to perform these special operations, you need the proper privileges, but you can't grant yourself those privileges without having those privileges. So, to grant a normal user those privileges, you need the uid 0 user briefly, and then you can remove it from the directory. To find out which privileges your version supports, you can use the net command:

```
# net rpc rights list -U root
Password:
SeMachineAccountPrivilege  Add machines to domain
SePrintOperatorPrivilege   Manage printers
SeAddUsersPrivilege        Add users and groups to the domain
SeRemoteShutdownPrivilege Force shutdown from a remote system
SeDiskOperatorPrivilege    Manage disk shares
```

Newer versions have added more privileges so make sure you know all that your version supports before proceeding. Now we need to assign privileges to groups and/or users. The obvious first step is to grant all privileges to the Domain Admins group:

```
# net rpc rights grant "CI\Domain Admins" \
SeMachineAccountPrivilege SePrintOperatorPrivilege \
SeAddUsersPrivilege SeRemoteShutdownPrivilege \
SeDiskOperatorPrivilege -U root
Password:
Successfully granted rights.
```

At this stage, we should be able to remove the root user from the directory, because any member of the Domain Admins group should be able to issue administrative Samba commands.

Joining a Machine to the Domain

So we have a Samba user but really nowhere for this user to log in to. In the Windows world, machines must join the domain for user domain accounts to be valid. When a machine joins the domain, it needs to create a domain account for itself. This account looks exactly like a regular user account except that it ends with a dollar sign. Because I don't use the smbldap tools, I wrote a small Perl script that reads the admin dn's password from the secrets.tdb and adds the machine account to the LDAP directory. The script is available from the on-line Resources and depends on the Crypt::SmbHash, Net::LDAP, File::Temp and TDB_File Perl modules. Once you have this script in place, you can add the machine to the domain by right-clicking on My Computer, choosing Properties, choosing the Computer Name tab and then clicking on the Change... button. Enter an appropriate computer name if one isn't already provided, then choose the Domain: option in the Member of field and enter your Samba domain name (Figure 1). Once you click OK, it will ask you for a user name and password. Enter the user name and password for the user that is a member of the Domain Admins group—leggett, in my case. After a few moments, you should receive a message that welcomes you to the domain. Once you reboot, you'll have the chance to log in as a domain user.

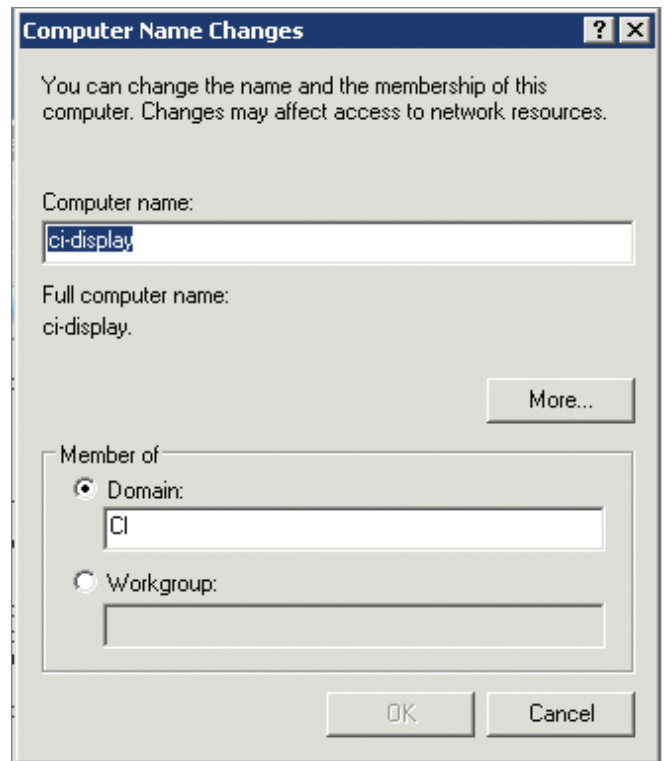


Figure 1. Joining the Domain

Single Sign-On and Windows

Although it's fine that you now have Windows machines plugged in to your infrastructure, this article is also about single sign-on. You might ask yourself "But authentication isn't being served by Kerberos, so how will single sign-on work?" MIT has a Kerberos for Windows package that allows

Flexibility to power the enterprise.



From mail servers to databases, ZT X9000 series servers powered by the 64-bit Intel® Xeon™ Processor can run the full range of 32-bit applications and offer extended flexibility for your 64-bit needs. So you can create powerful, all-purpose IT infrastructure that enhances business agility – and the bottom line.



ZT Optimum 1U Server X9535

- Intel® Xeon™ Processor 3.20 GHz (2MB L2 Cache, 3.20 GHz, 64bit, 800MHz FSB)**
- Intel® E7320 Chipset Server Board
 - 1GB ECC Registered DDR 333 SDRAM (Up to 16GB)
 - 4 x Seagate® 250GB SATA Hard Drive (Total 1TB Storage)
 - 4 x 1" Hot-swap SATA Drive Bays
 - 4 Channel High Performance SATA Controller (RAID 0,1,5,10,JBOD Support)
 - Slim CD-ROM & Slim Floppy Drive
 - Integrated 2-port SATA Controller (RAID 0, 1 Support)
 - Integrated Marvell 4-port SATA Controller (RAID 0, 1, JBOD Support)
 - 2 x Intel® 10/100/1000 Gigabit Network Controller
 - 1U Rackmount Chassis w/ 500W Cold-swap Power Supply
 - Super Doctor III Server Management Software
 - 3-Year Limited Warranty

\$2,599

ZT Optimum 2U SAS Server X9543

- Dual Intel® Xeon™ Processors 3 GHz (2MB L2 Cache, 3 GHz, 64bit, 800MHz FSB)**
- Intel® E7520 Chipset Server Board
 - 1GB ECC Registered DDR 333MHz SDRAM (Up to 16GB)
 - 1 x Seagate® 36GB 15,000rpm SAS Hard Drive (0/S)
 - 3 x Seagate® 73GB 15,000rpm SAS Hard Drive (Raid 5)
 - 6 x 1" Hot-swap SAS Drive Bays
 - 16x DVD-RW & Floppy Drive
 - 64bit High Performance SAS RAID Controller
 - 2 x Intel® 82541GI Single Port Gigabit Controller
 - 2U Rackmount Chassis w/ 550W Power Supply
 - Super Doctor III Server Management Software
 - 3-Year Limited Warranty

\$3,999

ZT Optimum 4U SAS Server X9545

- Intel® Xeon™ Processor 3 GHz (2MB L2 Cache, 3 GHz, 64bit, 800MHz FSB)**
- Intel® E7520 Chipset Server Board
 - 1GB ECC Registered DDR2 400MHz SDRAM (Up to 16GB)
 - 2 x Seagate® 73GB 15,000rpm SAS Hard Drive (Raid 1 for 0/S)
 - 3 x Seagate® 250GB 7,200rpm SATA Hard Drive (Raid 5 for Data)
 - 8 x 1" Hot-swap SAS / ESATA Drive Bays
 - 64bit High Performance SAS / ESATA RAID Controller
 - 16x DVD-RW & Floppy Drive
 - 2 x Intel® 82541PI Single Port Gigabit Controller
 - 4U Rackmount Chassis w/ 650W Redundant-Cooling Power Supply
 - Super Doctor III Server Management Software
 - 3-Year Limited Warranty

\$3,499

Free 3-5 DAY SHIPPING !

With purchased of select business servers. For a limited time only. \$19 handling charge applied. Call or go online to eligible servers

Find out how ZT can help maximize your Business Solution

Go to
Call

ztgroup.com/go/linuxjournal

866- ZTGROUP (866-984-7687)

promote code : LJ0206



Purchaser is responsible for all freight costs on all returns of merchandise. Full credit will not be given for incomplete or damaged returns. Absolutely no refunds for merchandise returned after 30 days. All prices and configurations are subject to change without notice and obligation. Opened software is non-refundable. All returns have to be accompanied with an RMA number and must be in re-sellable condition including all original packaging. System's picture may include some equipments and/or accessories, which are not standard features. Not responsible for errors in typography and/or photography. All rights reserved. All brands and product names, trademarks or registered trademarks are property of their respective companies. Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centinno, Intel Centinno logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, Pentium, and Pentium III Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

you to obtain and manage tickets similar to Apple's Kerberos.app (Figure 2).

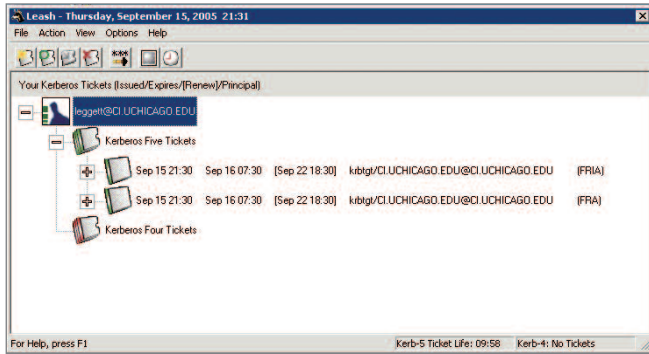


Figure 2. MIT Kerberos for Windows Leash

The two main needs for single sign-on are SSH access and mail access. Certified Security Solutions has patched the PuTTY SSH client for Windows to allow GSSAPI authentication. In order to use the MIT Kerberos for Windows under Windows 2000 and XP systems, copy the file plugin_mitgss.dll to plugingss.dll in the PuTTY install directory. Once you fire up PuTTY, go to the Auth menu in the Connection/SSH category and check Attempt GSSAPI/Kerberos 5 authentication (Figure 3). Make sure you have valid Kerberos credentials, and away you go.

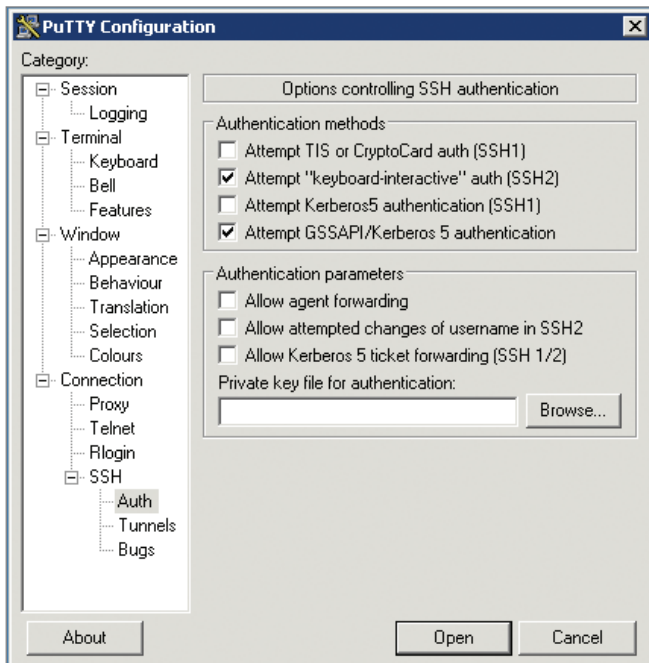


Figure 3. PuTTY GSSAPI Configuration

The last major piece to get working is mail access. Microsoft does not use GSSAPI as its authentication scheme. Instead it uses what is called SPNEGO. Because of this, Outlook and Outlook Express will not work with our single sign-on environment. But there's good news. Qualcomm's Eudora e-mail package supports GSSAPI, and it has a free version to boot.

Configuring Eudora v6.2

Start the account creation process, and choose Skip directly to advanced account setup. Enter the required information for the SMTP and IMAP settings. Make sure to choose If Available, STARTTLS for the Secure Sockets settings, and under the Incoming Mail tab, make sure to select Kerberos as the authen-

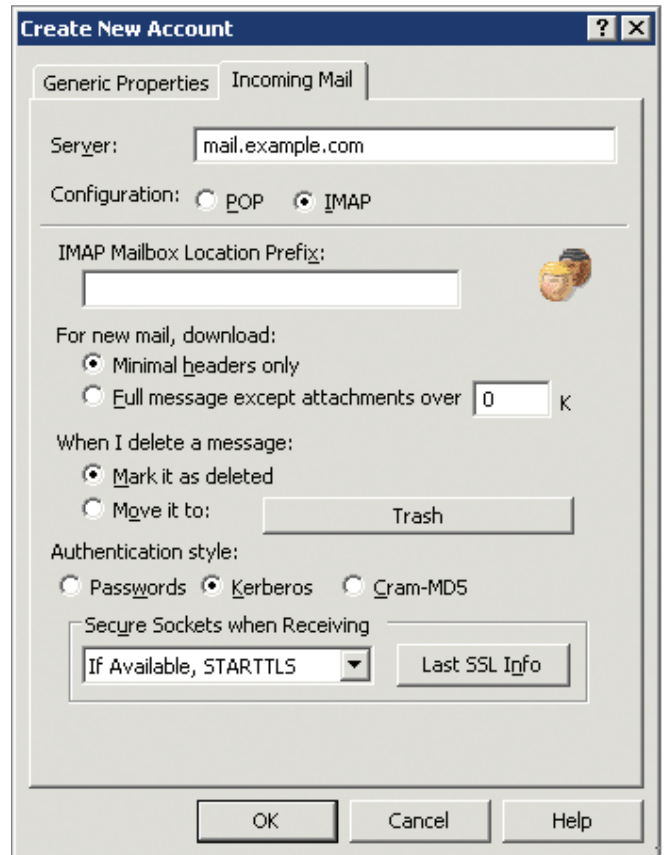


Figure 4. Eudora Account Creation

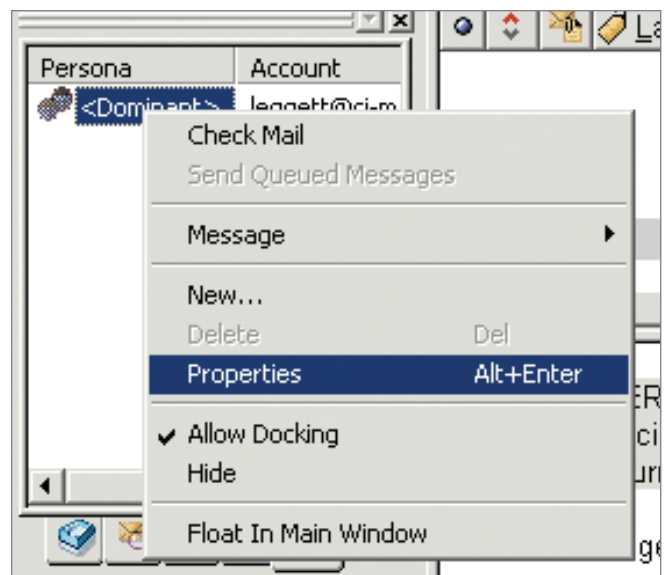


Figure 5. Eudora Personality Properties

Zervex

Hosting Control Panel



ServerCP is the perfect solution for hosting companies and organizations who need a webserver, mailserver, dns server, or a fully featured web hosting environment.

Features

1. RPM based update deployment
2. Automated Software Installer
3. Apache, PHP, MySQL
4. Multi-platform Integration

Supported Operating Systems

RedHat Linux 7.3 and 9
Fedora Core 1, 2, and 3
RedHat Enterprise Linux 3 and 4
CentOs 3, 3.1, 3.3, 3.4, and 4



Free to Try

or buy for

\$9.95/month

www.zervex.com

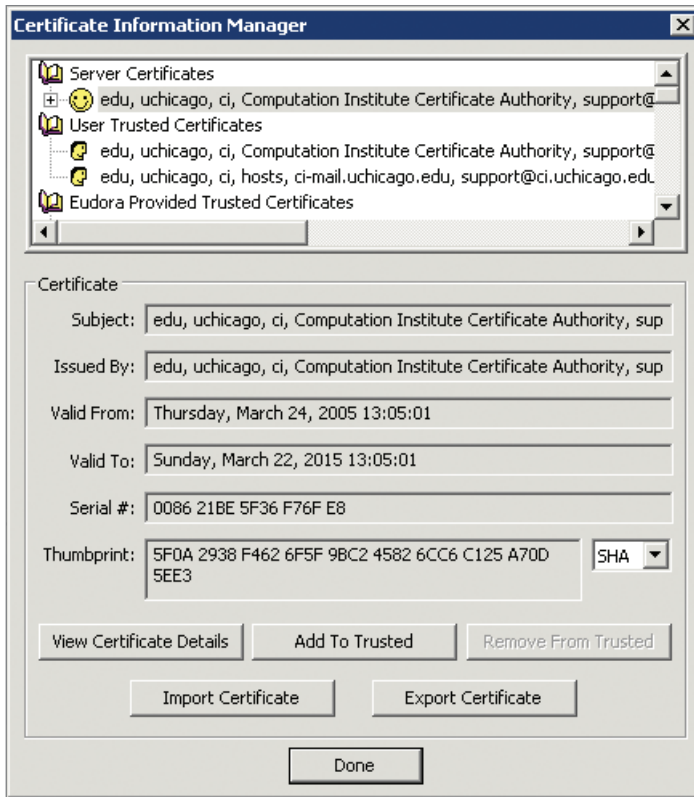


Figure 6. Eudora Certificate Information Manager

tication style (Figure 4).

Once you've gotten your account configured, you may get an error the first time you try to connect saying that either the connection has broken or GSSAPI failed. These errors aren't very descriptive of the actual problem, which is that Eudora doesn't trust your self-signed SSL certificates. To fix this, edit the properties for your newly created personality (Figure 5). Click on the Incoming Mail tab, then the Last SSL Info button, and then the Certificate Information Manager button (Figure 6). If you click the Add To Trusted button, your self-signed certificate will be trusted by Eudora as valid. You need to do this for your SMTP server as well the first time you try to send mail.

Wrapping Up

You now have integrated one more major architecture into your single sign-on and corporate directory infrastructure. There are still some pieces that could be added or enhanced, such as a way to keep passwords in sync between Kerberos and Samba, LDAP searches in Eudora and more-robust Samba user management scripts. However, you can

see how Kerberos and LDAP can make administration and use of your system much easier and more unified. In my last article in this series, I'll explore some ways to think about using your new infrastructure for administrative functions. Until then, keep expanding and using your corporate directory!

Acknowledgements

This work was supported by the Mathematical, Information, and Computational Sciences Division sub-program of the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract W-31-109-ENG-38.4:08.

Resources for this article:

www.linuxjournal.com/article/8701.

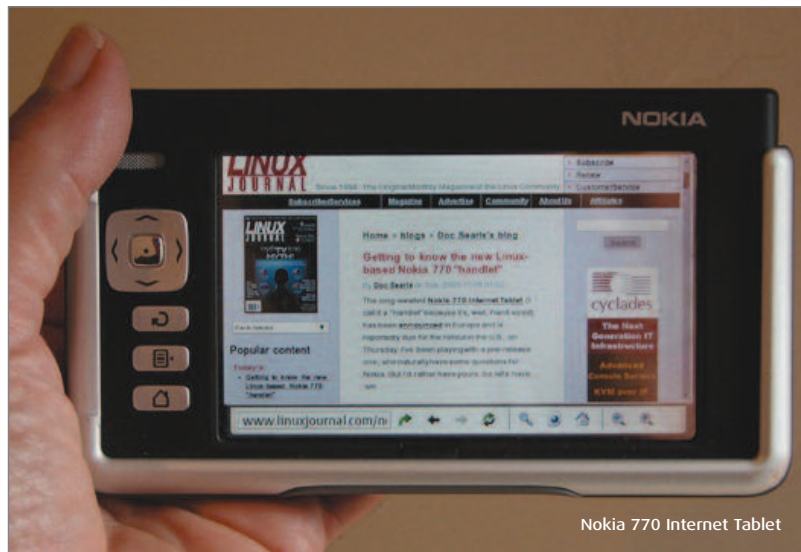
Ti Leggett (leggett@mcs.anl.gov) is a systems administrator for the Futures Laboratory of the Mathematics and Computer Science Division at Argonne National Laboratory. He also has a joint appointment with the Computation Institute at the University of Chicago.





A First Look at the Nokia 770

Our Senior Editor gives a high five to Nokia's hot new palm-sized Linux-based Internet Tablet. **BY DOC SEARLS**



Nokia 770 Internet Tablet

In November 2005, the long-awaited Nokia 770 Internet Tablet hit the stores, first in Europe and then in the US, where it promptly sold out at a retail price of \$350 US. When we posted a mini-review of a pre-release unit on the *Linux Journal* Web site (see the on-line Resources), we quickly got dozens of comments, some of which were very helpful in writing this first-look review.

The 770 is a radical departure for Nokia, mostly because it isn't a phone. Nor is it based on Symbian, the mobile device OS co-created by Nokia and running in zillions of Nokia devices. Instead, the 770 is a palm-sized tablet PC running on Linux.

It's small (5.5" x 3.1" x 0.7"), light and features a crisp 4.3" (800 x 480) 16-bit color screen. That means it crams a lot of detail in an area that requires easy zooming. For that the 770 provides both soft and hard buttons. Another hard button zooms the browser window to full screen. You can enter text (typically, URLs) with a stylus on the virtual keyboard or in handwritten letters that the 770 recognizes easily. You also can connect self-powered devices via Bluetooth or USB. It has no hard drive, though you can attach one by USB.

Although I haven't tested it in a formal way, I am impressed with the 770's battery life as well. It sleeps quickly, which helps. But it also wakes just as quickly, which is a must for a portable device like this.

The easiest way to connect to the Net is through Wi-Fi. The 770 tells you

the names, signal strengths and open/locked (WEP'd) statuses of each signal and makes it easy to override whatever choices the 770 makes on its own. It also supports WPA encryption (thanks, we are told, to Samuel Ortiz, a Linux kernel engineer who works for Nokia). Getting it up and going on the Net is so easy that my nine-year-old son (who is not a computer whiz—at least not yet) figured it out in a matter of seconds. He also figured out many of the unit's other fun features.

My own favorite is the 770's Internet radio. Many Web-based radio stations (such as, RadioParadise.com, KCRW.com, SmoothJazz.com, WEMU.org, WNYC.org and WUNC.org) publish their streams' IP addresses as page links. Clicking on the .mp3 stream links brings up a radio/audio player that pumps out excellent audio through a speaker or a standard 3.5mm stereo plug port. I've used it to drive headphones and home audio systems for several weeks now. It has essentially become our first-choice Internet radio.

Memorizing streams is less than obvious. I've figured most of it out, though I still don't know if I can get any of my own favorites to join or replace the three Virgin Radio streams defaulted as the only choices in the application.

The video performance (using a Real viewer) is also remarkably good. I've taken to watching the BBC's "News in Three Minutes" at news.bbc.co.uk.

The radio and video app are just

two of the first few apps that come with the unit. Other base apps include a browser (Opera 8—yes, with pop-up blocking), v6 Flash player, an e-mail client, a news reader, audio and video players, an image viewer, PDF viewer and file manager. The OS is called Internet Tablet 2005. It's upgradable. At the time of this writing, the 2006 edition is expected (and may be out by the time you read this) and will support additional services, including VoIP and instant messaging.

The 770's CPU is a ~220MHz OMAP 1710 powered by an ARM9 core. Memory is 64Mb DDR RAM, expandable through an RS-MMC (Reduced Size MultiMediaCard), or through full-size SD cards.

There's a development site (see Resources) with an active and highly useful wiki that should fill your need for deep data about the product, while equipping you to develop for it too.

I had many questions that were answered by e-mail exchanges with Nokia folks. I'll condense them into this Q&A:

Doc: What kernel version do you use?

Nokia: We take the kernel directly from kernel.org, but we use Debian package management to create our own "internal distro".

Doc: And what desktop (GNOME, presumably)?



Nokia: GNOME, yes. We've created our own widget set to provide the 770 look and feel. We call that the Hildon widget set. It is based on the GTK+ toolkit, which is an integral part of GNOME, and it is all open source [see Resources].

Doc: What were the design considerations for those?

Nokia: For the kernel/distro, we want to stay current with the Linux community. We want to benefit fully from the open-source development happening there. For that reason, we want to follow the latest kernel releases directly, and we actively submit our contributions back to open source. We are very active in the kernel's OMAP tree, for example. That gives us the most efficient way of working: use the latest, work with the community directly and submit your changes back ASAP, which results in speed, quality and cost benefits.

A commercial distro vendor between us and the Linux community would slow us down, would be more expensive and would prevent us from benefitting immediately from the work we do with the communities. Also, such a vendor possibly would control the tools, versions and partners we want to use. In addition, we have many issues that an external distro vendor could not solve, such as adaptation of Nokia-specific hardware and Linux.

Working with the communities directly is the way to go for us!

For the UI, we wanted to create a consumer device utilizing genuine open source (GNOME) and Nokia's expertise on UI design. GNOME is genuinely open source, it doesn't include any one single company's control, and it allows developers and companies to work with free, open-source components and tools. GNOME is the leading desktop environment—thus an obvious choice for us. Also, the GNOME community has been extremely supportive and has

www.faircom.com/go/?securelinux

Citibank demands
**unrivaled.
security**
for banking applications
at over 4,000 branches.

FairCom database
technology makes
it possible.



FairCom®

Other company and product names are registered trademarks or trademarks of their respective owners. © 2005 FairCom Corporation

helped us significantly to get things done. They've been really a part of our 770 team.

As for the UI, we wanted to create a consumer device that is easy and intuitive to use. We follow a so-called task-oriented design principle. Thus, instead of harassing users with a huge number of options and possibilities, we wanted to present the most-used features (browsing, e-mail) in a very accessible way. Instead of a Swiss-Army knife, we created a focused product. Our UI reflects that. The 770 is not your general-purpose PC or PDA. It is a consumer device—a tablet—that lets users get on-line (surf and do e-mail), regardless of the time and space. The UI supports that fully and makes that extremely easy for the user.

Doc: How closely did Nokia work with Linux developers on the product?

Nokia: As close as you can. We've been an integral part of the GNOME community, we've got many Debian developers on our payroll, we submit our code back a lot and so forth. So this is truly an open-source effort. Examples of the collaboration include, GNOME component work (gconf, D-BUS, gnome-vfs, GTK+, Gazpacho and so on) with Mikael Hallendal, Anders Carlson, Richard Hult, Michael Natterer Matchbox and X.org; GTK+ with Matthew Allum, Ross Burton and Richard Purdie; GStreamer with Christian Schaller, Wim Taymans and others; and GPE palmtop with Nils Faerber and Florian Boor and others, and so on.

Doc: As an "embedded" use of Linux, what parts (modules) of the kernel, if any, were left out?

Nokia: We have not left anything significant out. This is a "desktop product" much more than an "embedded product", really.

Doc: Is there an easy way to get to terminal mode? How? This is very important for my readership, which is mostly Linux experts who will want to work in the command line.

Nokia: We decided *not* to put a terminal into the device because it is really a consumer device—we do not expect soccer moms will want to deal with the command-line interface. However, anybody can go to the Application Catalog and download and easily install an xterm into the 770. By the way, our development site (see Resources) offers tools for a developer to work on the 770. We provide tools, documentation, example apps, a wiki, discussion mailing lists, support, source code and even a developer root filesystem so that you can really hack your way through the 770 if you want.

Doc: What is the story for external keyboards?

Nokia: Once again, for most of the target audience, an external keyboard is a bit—awkward. So we do not have a support for it built in—but as with the xterm, if you need one, you can have one through the maemo.org developer site. The kb plugin was developed independently and is not

a Nokia feature as such. However, if people really like it, we may integrate it with the device software in the future—and that goes for other apps and plugins too.

Doc: What is the audio player? Real? Helix?

Nokia: As a multimedia framework, we use GStreamer. As far as formats, we support audio (MP3, MPEG4-AAC, WAV, AMR and MP2) and video (MPEG-1, MPEG-4, RealVideo, H.263, AVI and 3GP). So we have a Real player too. And Flash, of course.

Doc: In Date/Time, is there a way to get the device to set through an Internet time server or site?

Nokia: Not in this software release. By the way, this leads to an important topic—unlike phones, the 770 is software upgradable. So, there will be new software versions coming, and customers can upgrade their software. The new versions *may* have some of the features you proposed, but it will include VoIP and instant messaging for sure.

If that dialog doesn't encourage you, perhaps Greg Kroah-Hartman will. Greg is the driver subsystems maintainer for the Linux kernel and one of the most prolific contributors to the kernel itself. In November 2005, he wrote this in his blog (see Resources):


My first reaction was like everyone else's, "Damn, that's a nice screen." After playing around with it this week more, I'm really hooked. It handles streaming Internet music just fine, replacing my laptop for this task. And putting a ripped DVD on the memory card makes the kids happy to watch the *Jack-Jack-Attack* short film over and over.

But it goes deeper than that. It's actually a useful Web browser. I can successfully read different news sites just fine, all from a tiny little device with a very good battery life (at least compared to my laptop).

Combine that with a very active development community already (Nokia was smart in seeding it with devices, very wise move), and I think this will be one platform that will be worth watching for some time. The number of applications will only grow and get better. It's already fun to use an xterm on the thing.

Now to wait for the kernel source tree for it to be released, so I can get to tweaking on it, and figure out why they are using my pl2303 driver [see Resources] when I don't see a serial output anywhere....

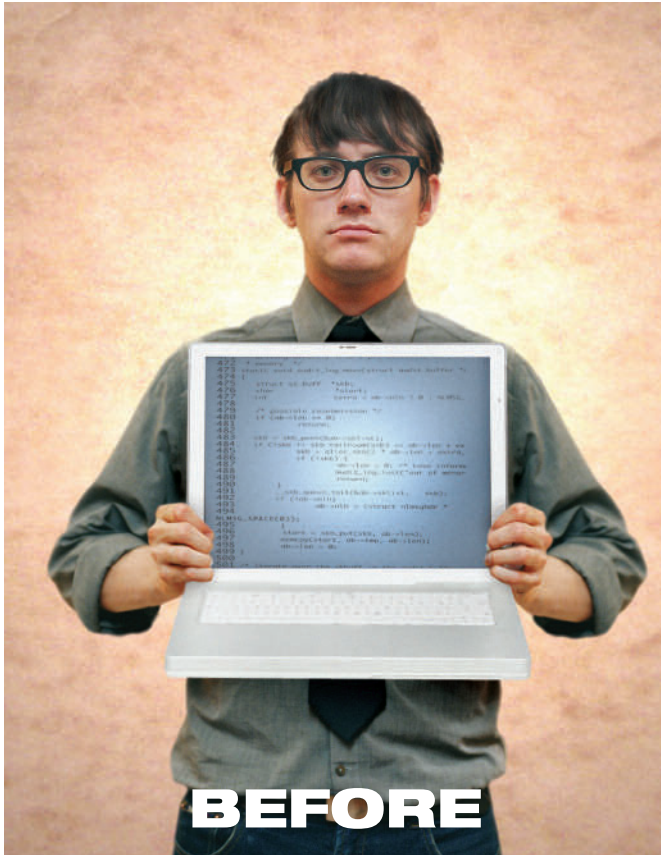
I expect the 770 to be not only a hot product, but to open a hot new device category as well.

Resources for this article: www.linuxjournal.com/article/8700.

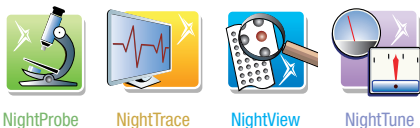
Doc Searls is Senior Editor of *Linux Journal*.

WARNING:

TO THOSE CONSIDERING PERFORMANCE ENHANCEMENT



Natural ability and ordinary debuggers can take you just so far. That's why you need NightStar LX™. An integrated suite of tools that gives you full visibility into your Linux® application. You can debug, monitor, analyze and tune at application speed, so you see real execution behavior. Plus, you'll reduce test time and lower costs. NightStar LX. Experience real power for a change.



The NightStar LX suite is extremely habit-forming, and may cause feelings of euphoria.



See if NightStar LX is right for you. Download a **Free Trial** at

www.ccur.com/nightstar/LX

800-666-4544 • 954-974-1700

Email us at nightstar@ccur.com



MSP430 Development with Linux

Using Linux and the TI MSP430 processor to create blinking LEDs is a learning exercise, not just a way to make cheap sci-fi movie panels.

BY BRIAN C. LANE

Linux embedded development doesn't only mean embedding Linux in a product, it also means using Linux as a development platform for embedded microprocessors. The majority of computer processors is not on the desktop or in Beowulf clusters, they are embedded in the millions of devices that you use every day: your alarm clock, microwave, thermostat, car, cell phone and so on. Linux can be used to develop software for embedded projects using microprocessors like the Microchip PIC, Atmel AVR, Philips LPC ARM and TI MSP430 devices.

You don't need to be a professional developer to have fun with these tiny computer devices. The development tools are inexpensive, and the hardware required is minimal. Soldering skills and some experience writing in C are really all you need to get started creating a controller for your next Sumo Bot, remote controlled helicopter or digital lock.

The Texas Instruments (TI) MSP430 family of microprocessors have a wide range of features:

- Power consumption as low as 0.1uA in off mode.
- Multiple 16-bit timers with capture/compare.
- PWM outputs, which are very useful for robotics.
- A/D converters for monitoring sensors.
- SPI and Asynchronous UARTs for communications.
- Integrated LCD display controllers.
- Onboard oscillators.
- Multiple clock sources for low-power sleep modes.

The '430 family includes more than 80 devices in four major groups. Flash, where the program is stored, ranges from a paltry 1KB to an extravagant 60KB. The available RAM is as large as 5KB or as small as 128 bytes.

Of course, 128 bytes isn't much RAM, but it is enough to get the job done for small projects. Many of my embedded projects are written in assembly code to conserve space, but with the MSP430, I have found myself using C as the primary language. Instead of using a commercial compiler or IDE, I have

chosen to use the GNU GCC toolchain, which has had MSP430 support added to it. The GCC compiler does a pretty good job of generating code, and C is certainly the better choice if the code is going to be maintained over a number of years. There's nothing worse than returning to a heavily optimized assembly project five years later and trying to make adjustments without the whole system crumbling to its knees.

With the Microchip PIC processors that I wrote about in 1998, the development process was a bit tedious. I would write some code, compile it, flash the PIC, wonder why it didn't work and then repeat. I used a couple of I/O lines and LEDs as debugging tools, but there's only so much information that you can grok from two flashing LEDs. I really had no way to know exactly what was going on inside the processor when things went wrong. One solution could have been to buy an In Circuit Emulator (ICE). An In Circuit Emulator is a device you plug in to the socket where the processor normally goes. It emulates the CPU and lets you examine every instruction, memory locations and much more. But the \$1,200+ US price for an ICE was out of my reach.

Today, things are much easier for both the the hobbyist and professional. Many of the newer processors include built in real-time debugging support using the IEEE Std 1149.1 JTAG specification. This six-wire interface allows real-time debugging of the software running on the target device. You can step through your code, watch registers and memory locations change, insert breakpoints and modify RAM on the fly. This is a dramatic improvement over the old 1- or 2-bit diagnostic line.

Instead of dropping a dozen c-notes on expensive debugging tools, you can get a parallel port JTAG adapter from Olimex for the amazing price of \$15 US. This allows people to debug their code with interactive debuggers like gdb instead of relying on blinking LEDs and serial ports.

Setting Up MSP430 Software Development

MSP430 support was added to GCC by a group headed by Dmitry Diky and Chris Lichti. Their project includes the gcc compiler, linker, libraries, gdb debugger and a closed-source interface to the parallel JTAG adapter. The main emphasis of the mspgcc project has been on Windows. Getting it working on Linux is a bit of a struggle, involving compiling the alternate toolchain, libraries and so on. Maybe I'm getting old, lazy or just used to .deb and .rpm packages, but these days, I prefer not to fight with the software I'm installing. Running `rpm -Uhv <package>` saves my energy for debugging



The holy GRAIL

Easy, high-performance clustering. For years, many searched, but none could find it. Some said it didn't exist. But not the Penguin.

Penguin Computing® made easy, high-performance clustering a quest. Now you can find Linux hardware and software solutions, configured to order, driven by Scyld's commercially supported, industry-leading Linux clustering software.

For the turnkey clusters you need to run even your most important applications, come to Penguin Computing. Penguin Computing's dedicated experts, who are 100% focused on Linux, are waiting to serve.

Powerful, easy clustering. It's the once and future thing. **Love what you do.** ©

www.penguincomputing.com



my MSP430 code.

Thanks to Stephan Linz and his Cross Development Kit for MSP430, I don't need to spend more than five minutes installing development tools. He has done all the hard work of getting mspgcc compiled and packaged as RPMs. Stephan also has created Cross Development Kits for the AVR processor and the Altera soft core NIOS, if you are interested in those target processors. If I ever need to write code for the AVR, I know where to go.

The cdk4msp project is available from SourceForge, and here is a minimal list of the packages that need to be installed from the cdk4msp SourceForge download page:

- cdk-msp-base-0.2-20031111.i386.rpm
- cdk-msp-binutils-2.14-20031106.i386.rpm
- cdk-msp-examples-libc-20031101cvs-20031102.noarch.rpm
- cdk-msp-examples-mspgcc-20031101cvs-20031102.noarch.rpm
- cdk-msp-gcc-3.3.2-20031106.i386.rpm
- cdk-msp-gdb-5.1.1-20031106.i386.rpm
- cdk-msp-gdb-proxy-5.1.1-20031106.i386.rpm
- cdk-msp-jtag-lib-20031101cvs-20031102.i386.rpm
- cdk-msp-libc-20031101cvs-20031102.noarch.rpm

Additional document packages can be downloaded, depending on your preference for man pages, info files, PDF or HTML pages. I have successfully used these packages on Fedora Core releases 1 through 4, and although I haven't tried any other RPM-based distributions, I expect them to work just fine. These RPM packages function as a self-contained unit, and don't depend on any outside packages.

Install the packages in this order with the following commands:

```
rpm -Uhv cdk-msp-base-0.2-20031111.i386.rpm
rpm -Uhv cdk-msp-binutils-2.14-20031106.i386.rpm
rpm -Uhv cdk-msp-libc-20031101cvs-20031102.noarch.rpm
rpm -Uhv cdk-msp-gcc-3.3.2-20031106.i386.rpm
rpm -Uhv cdk-msp-gdb-5.1.1-20031106.i386.rpm
rpm -Uhv cdk-msp-jtag-lib-20031101cvs-20031102.i386.rpm
rpm -Uhv cdk-msp-gdb-proxy-5.1.1-20031106.i386.rpm
rpm -Uhv cdk-msp-examples-libc-20031101cvs-20031102.noarch.rpm
rpm -Uhv cdk-msp-examples-mspgcc-20031101cvs-20031102.noarch.rpm
```

The install places everything in the directory tree below /opt/cdk4msp. Take a look at the examples in /opt/cdk4msp/examples/mspgcc and the documents in the /opt/cdk4msp/doc, info and man directories, depending on which style of documentation you installed.

A Simple Blinking LED

Blinking an LED is the embedded equivalent of "Hello World". We will modify one of the examples to make it a little easier to understand. Copy the leds example from /opt/cdk4msp/examples/mspgcc/leds/ to a working directory,

and replace main.c with the simplified version below. Edit the Makefile and set the CPU variable to msp430x149 so that it compiles for the correct target. If you are using a different version of the MSP430, you can get a list of the supported types by running msp430-gcc --target-help | less and then set CPU to the appropriate type.

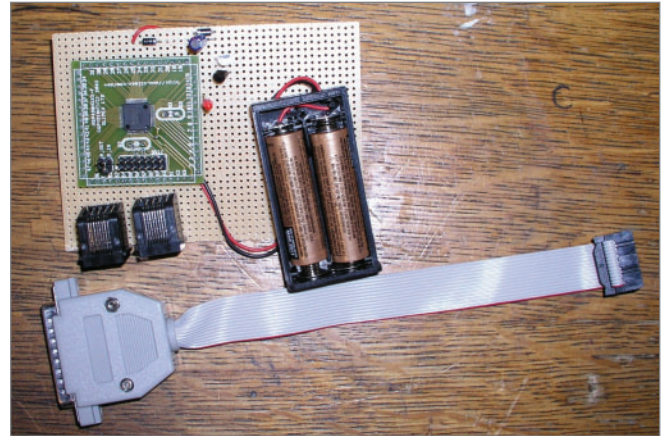


Figure 1. Picture of My LED Blinker Development Board and JTAG Adapter

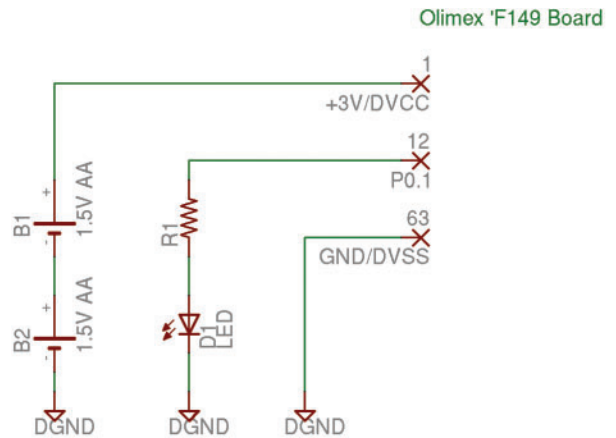


Figure 2. Simple Schematic of the LED Blinker Setup

Replacement for main.c:

```
/* Simple LED Blinker program for MSP430 */
#include <msp430x14x.h>

/* Brute Force delay loop */
void delay(unsigned int d)
{
    for (; d>0; d--) {
        nop();
        nop();
    }
}

int main( void )
{
    /* Init watchdog timer to off */
    WDTCTL = WDTPW|WDTHOLD;
```

```

/* Init Output ports to GND */
P1OUT = 0x00;
P2OUT = 0x00;

/* I/O not module control */
P1SEL = 0x00;
P2SEL = 0x00;

/* Set up the data direction registers
   P1.0 output, input on the rest
*/
P1DIR = 0x01;
P2DIR = 0x00;

/* No Interrupts on Port Pins */
P1IES = 0x00;
P2IES = 0x00;
P1IE = 0x00;
P2IE = 0x00;

/* Loop until the universe breaks down */
while (1) {
    /* Toggle P1.0 output pin */
    P1OUT ^= 0x01;

    /* Delay for a while before blinking */
    delay(0x4fff);
} /* while */
}

```

Run make to compile it. You should see no warnings or errors:

```

msp430-gcc -mmcu=msp430x149 -O2 -Wall -g -c -o main.o main.c
msp430-gcc -mmcu=msp430x149 -o leds.elf main.o
msp430-objcopy -O ihex leds.elf leds.a43
msp430-objdump -dSt leds.elf >leds.lst

```

At this point, we have the software development set up, but no hardware to run it on or LEDs to blink.

MSP430 Simple Hardware Setup

Very little hardware or money is required to get started with the MSP430. You need a PC board with the processor on it and a JTAG adapter to connect the board to the parallel port. Olimex makes a number of inexpensive evaluation boards and JTAG adapters for the MSP430, ARM, AVR and PIC. In the US, their products are carried by a neat place called Spark Fun Electronics, which carries the Olimex boards as well as its own unique collection of adapter boards and projects. Table 1 shows what's needed to build the circuit on the schematic in Figure 1.

One problem with low-powered devices like the MSP430 is that when you try to turn them off, they don't discharge the supply capacitors all the way to ground. This can result in a brownout condition when the processor won't reboot until reset properly. For the sake of simplicity, we are going to run the processor directly off of a pair of AA batteries. It will also run off the power from the JTAG adapter itself if you don't have any batteries handy.

In a production design, I would add a power supply with reset manager to prevent any brownout problems, but for our

ASA COMPUTERS

Want your business to be more productive?

The ASA Servers powered by the Intel® Xeon™ Processor provides the quality and dependability to keep up with your growing business.

Hardware Systems For The Open Source Community—Since 1989 (Linux, FreeBSD, NetBSD, OpenBSD, Solaris, MS, etc.)



6TB + in 5U—\$7,699

Intel 7501, Dual Intel® Xeon™ 2.4GHz
512 MB DDR ECC RAM *Max: 8GB*
6TB + IDE Storage
Dual Gigabit LAN, CD+FD, VGA
Options: SATA Drives, Firewire,
DVD+RW, CD+RW, 64 Bit
OS Configurations, etc.

1U Dual Itanium IDE—\$3,701

Dual Intel® Itanium® 2 1.4 Ghz
2 GB ECC DDR
1 of 4 x 40 GB HDD
Dual Gigabit LAN
Based on Supermicro 6113M-i



14" Deep Appliance Server—\$865

Intel® Xeon™ 2.4 Ghz Processor
40 GB Hard Drive, One GigE
Options: CD, FD, 2nd HD, Your Logo
on Bezel
Call for Low Cost Options.

1U Dual Xeon™ EM64T Superserver—\$1,675

SuperMicro 6014H-82 Barebones
1 of 2 Intel® Xeon™ 2.8 GHz 800 FSB
1 GB DDR II-400 RAM *Max: 16GB*
36 GB 10K RPM SCSI *Max: 4 HS HDD*
CD+FD, Dual GigE, VGA, RAILS
Options: RAID, etc.



Your Custom Appliance Solution

Let us know your needs, we will get you a solution



ASA Collocation

\$75 per month for 1U Rack - 325 GB/month

ASA Collocation Special

First month of collocation free.*

Storage Solutions

IDE, SCSI, Fiber RAID solutions
TB storage options
3Ware, Promise, Adaptec,
JMR, Kingston/Storcase solutions

Clusters

Rackmount and Desktop nodes
HP, Intel, 3Com, Cisco switches
KVM or Cyclades Terminal Server
APC or Generic racks

All systems installed and tested with user's choice of Linux distribution (free). ASA Collocation—\$50 per month



2354 Calle Del Mundo,
Santa Clara, CA 95054

www.asacomputers.com

Email: sales@asacomputers.com

P: 1-800-REAL-PCS | FAX: 408-654-2910



Intel®, Intel® Xeon™, Intel Inside®, Intel® Itanium® and the Intel Inside® logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Prices and availability subject to change without notice. Not responsible for typographical errors.

Table 1. Materials and Cost for Circuit		
Description	Part #	Price (US)
Olimex MSP430F149 header board	MSP-H149	\$22
Olimex MSP430 JTAG Adapter	MSP-JTAG	\$15
AA battery pack from DigiKey	2463K-ND	\$0.78
Red LED from DigiKey	160-1499-ND	\$3.60/10
330-ohm resistor	220QBK-ND	\$0.56/10
	Total:	\$41.96

simple circuit, pulling the batteries loose for a few seconds is sufficient to reset it.

The next thing we need to do is set up the gdb proxy and JTAG software. This requires using the PC's parallel port, and if your Linux installation is set up to use a local printer, you need to remove the printer and disable the printer daemon before you can use the port for JTAG. You also may need to add your development user to the group that has access to the port. On Fedora Core, this is the lp group, and you can add users to it by editing the /etc/group file as root.

Now that we have the required hardware set up, it's time to compile and flash our LED blinker program. In the directory where we ran make earlier, now run make download-jtag, and the program will be flashed into the target processor:

```
msp430-jtag -e leds.elf
MSP430 parallel JTAG programmer Version: 1.3
SHF_MASKPROC = 0xf0000000
Mass Erase...
Program ...
188 bytes programmed.
```

Next, we need to start the gdb proxy that creates a local port for gdb to connect to and handles the communication with the target hardware. Run this in a second window, because it outputs debugging info to stdout while it is running:

```
msp430-gdbproxy --debug --port=2000 msp430

Remote proxy for GDB, v0.7.1, Copyright (C) 1999 Quality Quorum Inc.
MSP430 adaption Copyright (C) 2002 Chris Liechti and Steve Underwood

GDBproxy comes with ABSOLUTELY NO WARRANTY; for details
use --warranty' option. This is Open Source software. You are
welcome to redistribute it under certain conditions. Use the
'--copying' option for details.

debug: msp430: msp430_open()
info: msp430: Target device is a 'MSP430F149' (type 7)
notice: msp430-gdbproxy: waiting on TCP port 2000
```

GDB needs to know how to connect to the MSP430 JTAG

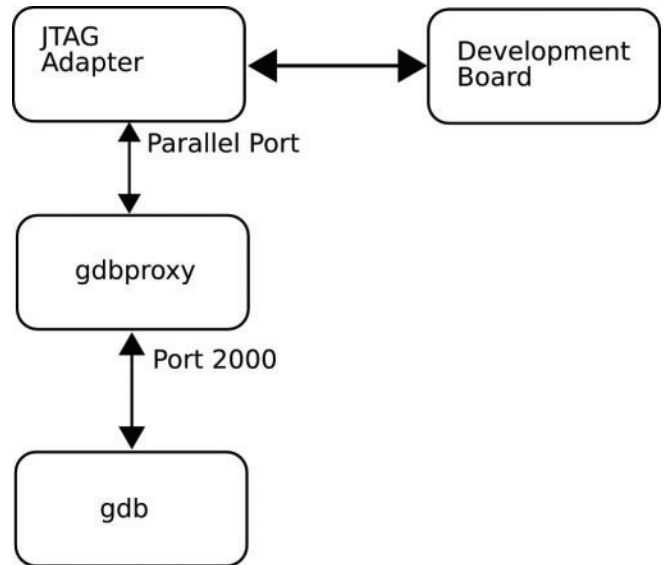


Figure 3. Simple Diagram of GDB to Target Board Connections

proxy port. Create a file named .gdbinit in your working directory and put the following three lines into it:

```
set remoteaddresssize 64
set remotetimeout 999999
target remote localhost:2000
```

Now we are ready to debug our LED blinker program. When you ran make download-jtag, the LED should have begun to blink, and when the gdbproxy was started, it should have stopped, because the processor is being held in reset by the JTAG.

Start debugging by using the msp430-gdb program:

```
msp430-gdb leds.elf
```

Run until main() by entering:

```
break main
c
```

Debug as you normally would. Here is an example session:

```
(gdb) break main
Breakpoint 1 at 0x1152: file main.c, line 16.
(gdb) c
Continuing.

Breakpoint 1, main () at main.c:16
16      WDTCTL = WDTPW|WDTHOLD;
(gdb) n
19      P1OUT = 0x00;
(gdb) n
20      P2OUT = 0x00;
(gdb) break delay
Breakpoint 2 at 0x1140: file main.c, line 7.
(gdb) c
Continuing.
```

```

Breakpoint 2, delay (d=20479) at main.c:7
7         for (; d>0; d--) {
(gdb) n
8             nop();
(gdb) n
9             nop();
(gdb) n
7         for (; d>0; d--) {
(gdb) print d
$1 = 20479
(gdb)

```

What's happening here is that the proxy is communicating with the JTAG adapter and opening up port 2000 to accept connections from the gdb debugger. The debugger needs to know where to connect to, hence the creation of the .gdbinit file with the port number and timeout in it. When you run msp430-gdb, it is making a TCP/IP connection to the JTAG proxy program, which is in turn communicating with the target hardware if all is going well.

Schematics and Custom PC Boards

Flashing LEDs get boring pretty quickly. Once you get the bugs worked out of your blinking LED, you probably are going to want to design a custom PC board with interfaces to the outside world. One of my projects is a one-wire lock controller with an RS-232 interface that uses Dallas Semiconductor

iButtons for access control.

For this project, I chose one of the smallest of the family, the MSP430F1101 with 1KB of Flash and 128 bytes of RAM. It uses a pair of transistors to switch a DC motor on and off and a MAX3221 for serial communications with a PC. The C code to control the lock just barely fits into the 1K Flash space of the '1101. A low dropout voltage regulator is used to power the board and provide a clean reset to the processor. I drew the schematic and designed the board using Eagle CAD Lite under Linux. Eagle has several versions of its schematic and PCB auto-router, including a free version for noncommercial use:

- Free for noncommercial use.
- Board size limited to 3.2" x 4" and two layers.
- One schematic sheet.
- Lite version for \$49 US with same limitations as the free version.
- Standard version for \$600 US.
- Pro versions for \$1,200 US.
- Linux, Windows and Mac versions are available.

Hurricane Electric Internet Services...**Speed and Reliability** That Sets You Apart from the Competition!

Flat Rate Gigabit Ethernet

1,000 Mbps of IP

\$13,000/month*

Full 100 Mbps Port

Full Duplex

\$2,000/month

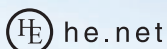
Colocation Full Cabinet

Holds up to 42 1U
servers

\$400/month

Order Today!

email sales@he.net or call 510.580.4190



* Available at PAIX in Palo Alto, CA; Equinix in Ashburn, VA; Equinix in Chicago, IL; Equinix in Dallas, TX; Equinix in Los Angeles, CA; Equinix in San Jose, CA; Telehouse in New York, NY; Telehouse in London, UK; NIKHEF in Amsterdam, NL; Hurricane in Fremont, CA and Hurricane in San Jose, CA

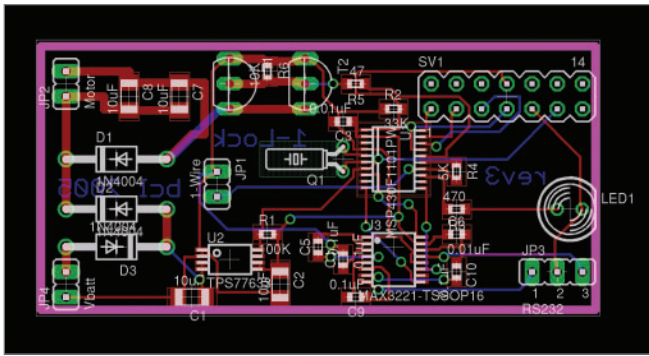


Figure 4. Lock Project PC Board Layout from Eagle

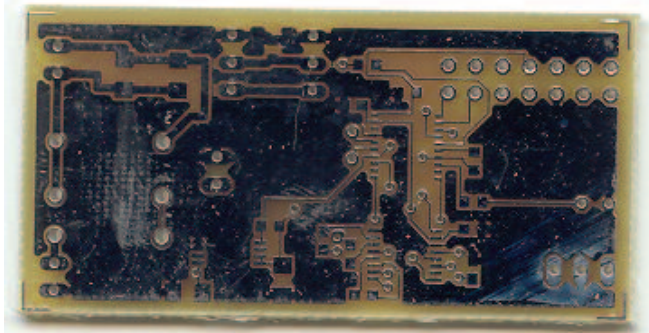


Figure 5. Picture of the Prototype PC Board

Eagle CAD is easy to get started with, low in cost and very powerful. A user scripting language allows you to add features and customize the program to fit your needs. User support for Eagle is very strong, and the Web site has an extensive collection of user-created libraries.

The Eagle auto-router supports advanced features like back-annotation, keep-out areas, design rules checks and one of my favorites—flood fill with thermal relief. If you have ever tried to solder a pin surrounded by a large ground plane, you will appreciate the advantage of thermal relief on power pins. Without it the ground plane acts like a large heat-sink and solder won't stick. All levels of the PCB and Schematic editor support the concept of back (and forward) annotation. You can make changes on the PCB, and they will be reflected on the schematic and updates made to the schematic are reflected on the PCB.

After designing a PC board, you actually need to make one. You can etch your own, but it is difficult to match the quality of even the least-expensive board manufacturers. Some manufacturers will accept Eagle PCB files directly, which saves you the step of converting the design to the Gerber format. The Gerber format is a lot like an old pen plotter; it tells PCB etching equipment where to draw the trace and how large a line to draw. Most PCB manufacturers still require Gerber files, so Eagle includes a script to output the necessary Gerber files.

One difficulty in dealing with Gerber files is that although Eagle can export the PCB in the correct format, it has no way to view the output to verify it was converted correctly. Linux has needed a good Gerber viewer for years, but it has been available only with recent releases of the gerbv program. It

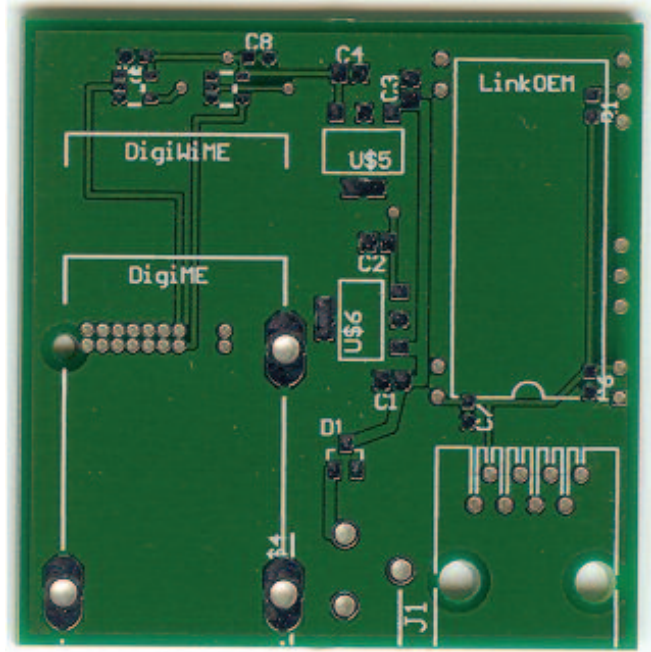


Figure 6. Picture of LinkWiFi PC Board

isn't as intuitive as I would like, but it does function well enough to display the Gerber files so that you can check the final output before sending it off to have 1,000 of your latest widget design created.

I have used four different PCB manufacturers myself. Their prices and features vary, but customer service and quality from all four have been excellent. Olimex and PCB Pool both accept Eagle CAD files directly, with no need for conversion to Gerber. Olimex is in Bulgaria, and turn-around time can be up to three weeks, but prices are excellent. PCB Pool is in Ireland and has quick turn-around or longer turn-around times, depending on price (as do most). I used PCB Pool for the one-wire Wi-Fi boards (Figure 6).

AP Circuits is in Canada and has very good prices and very fast turn-around. The bare one-wire lock boards were ordered on a Saturday, and I received them on Wednesday. I ordered them with no silkscreen or solder mask in order to keep the price low. For production, I used E-Teknet for my DT-1A temperature sensor boards with excellent results.

The MSP430 is a fun and easy-to-use processor; its wide range of features and access to free development tools and low-cost JTAG hardware make this processor a good choice for both the hobbyist and the professional developer. Using the GNU gcc toolchain reduces the learning curve and allows you to use the same tools for developing code on the MSP430 that you would use for Linux projects. My set of development tools includes make, gcc, gdb and joe.

Resources for this article: www.linuxjournal.com/article/8697.

Brian C. Lane lives in Port Orchard, Washington, with his wife and son, who is a huge *Tux Racer* fan. He serves as Webmaster For Life for the Kitsap Peninsula Linux User Group and writes Linux apps in his spare time.

Polywell PC Appliance Solutions

OEM Service for Small to Medium Size Projects

AMD Dual-Core technology Enables one platform to meet the needs of multi-tasking and multi-threaded environments; providing platform longevity

Allows users to run 32- and 64-bit applications as they desire – without sacrificing performance

1U 4-way, 64GB DDR, 2TB RAID

- 2 x AMD® Opteron™ Dual-Core Processors 280+
- with Hyper Transport Technology
- Upto 64GB 400MHz ECC DDR (16 Sockets)
- Upto 2TB 4 x 500G Swap Drive per 1U Rack
- 1 x PCI-X 133MHz, 1 x PCI-E Slot
- Dual Gigabit Ethernet, ATI Graphics, 4 SATA-RAID
- Slim CD-ROM Drive, Optional DVD-RW or CDRW
- Optional Slim Floppy Drive
- 1U 24" Depth Rack Chassis with upto 600W PS
- 4 x Swappable Drive Bays (SATA or SCSI)
- Supports Linux, FreeBSD or Windows
- Custom Configuration Available
- Please call for other Options



1114AIS-2050M, 280+,32GB, 2TB **\$12950**
 1114AIS-2050M, 265+,16GB, 1TB **\$6,299**
 1112ES-2200A, 244+, 2GB, 500G **\$1,999**



4024AIS 4U 12TB NAS/SAN **\$14,999**
 4024AIS 4U 6TB NAS Storage **\$7,999**

Linux Appliance PCs

- Custom Made Odd Size Chassis
- AC or DC Power Supply
- Low Power Voltage AMD Sempron™ Processor
- or High Performance AMD Athlon™ 64 Processor
- Diskless or Flash OS Boot Drive
- Swappable Hard Drive, CD-ROM, FDD
- Integrated Graphics, Ethernet, USB
- Optional LCD LED Control Module
- IS2 Audio, MPEG2/4 Hardware Video
- Upto 4 Ethernet Ports or 4 Serial Ports
- We have over 18 years OEM Experience in Set top Box, Digital Media Player, POS Kios, Thin Client, Networking Appliance,



SX2500SPLJ11B
 OEM Appliance starts at **\$299**

64GB DDR 4-way Workstation

- 2 x AMD® Opteron™ Dual-Core Processors 280+
- Upto 64GB 400MHz ECC DDR (16 Sockets)
- 2 x 133MHz PCI-X, 1 x PCI-E x16 Slots
- Dual Gigabit Ethernet, 4 x SATA-RAID Controller
- Quiet and Cool 12-Bay Tower + 600W 80Plus P/S
- 250G HD, DVD-RW, Floppy, Optional Card Reader
- 8-Layer Motherboard with Special Quiet Cooling
- On-board ATI Graphics, Sound Blaster 7.1 Sound
- Supports 64/32-bit Linux, FreeBSD or Windows
- Special for Large Memory Intensive Applications
- Built-to-Order or Configure-to-Order



64G RAM, 2x280+, QuadroFX4500 **\$22,999**
 32G RAM, 2x265+, QuadroFX3400 **\$9,999**
 16G RAM, 2x244+, QuadroFX1300 **\$4,699**

PolyStation 2050M
4-Way Two Dual-Core Processors
Up to 64GB memory, 16 Sockets

1U Power Saving ISP Server

- AMD Sempron™ or Opteron™ Processor
- 512M DDR 400MHz Memory
- 80GB Hard Drive
- 10/100Mbit Ethernet
- We provide Drive Image Service
- 1U 14" Short Rack, allow 2 x 1U per Rack
- Low Power Usage to save your Data Center Cost
- Perfect Entry Level ISP Server or Appliance System
- IDE Flash Drive is available
- Supports Linux, FreeBSD or Windows
- Custom Configuration is Available
- Please call us to discuss your specification



Order# VX2500SP1U-17LJ11A
 starts at **\$399**

Pick the chassis to fit your application



2U 8-way, 5U 16-way Servers

- 8 or 4 AMD® Opteron™ Dual-Core Processors 865+ with Hyper Transport Technology
- Upto 128GB DDR Memory for 16-way (32 sockets)
- Upto 64GB DDR Memory for 8-way (16 sockets)
- 4x Gigabit LAN, 8x SATA RAID-5 for 16-way
- 2x Gigabit LAN, 4x SATA, U320 SCSI for 8-way
- 4 x 133/100/66MHz PCI-X Slots for 16-way
- 2 x 133, 2 x 66MHz PCI-X, 1x PCI Slots for 8-way
- On-board ATI Graphics, USB 2.0
- 5U 26" Rack 1300W 3+1 Redundant P/S 16-way
- 2U 27" Rack 700W PFC P/S for 8-way
- Supports Linux, FreeBSD or Windows
- Custom Configuration Available
- Please call for other Options



5U 16-way 865+, 32GB, 2TB, 8800U5 **\$25,999**
 2U 8-way 865+, 8GB, 750GB, 8422C **\$9,500**

Mainstream Linux Systems

- AMD® Athlon™64 or X2 Dual-Core Processors with Hyper Transport Technology
- 512MB to 4GB Dual Channel DDR Memory
- 80GB to 500GB Hard Drive (SATA-RAID-0/1/5)
- 2 PCI, 2 PCIe, Gigabit LAN, 7.1 Audio, 1394a
- nForce 430 Chipset, GeForce 6100 Graphics
- Desktop, Tower, MiniBox or Rackmount Chassis
- Internet Server or Linux Appliance Configuration

939NX4LX starts at **\$399**

Polywell OEM Services, Your Virtual Manufacture
Prototype Development with Linux/FreeBSD Supports
Small Scale to Mass Production Manufacturing
Fulfillment, Shipping and RMA Repairing



AMD64 Investment Protection
 Migrate to 64-bit platforms seamlessly.
 Add 64-bit application as necessary.

888.765.9686

www.Polywell.com/us/LJ



AMD64 architecture reduces I/O bottlenecks, increases bandwidth, and reduces memory latency. Critical information gets to those who need it quickly and efficiently.

Shell Scripting a Camera Server

Adding functionality to an embedded device doesn't have to be complex. How about some shell scripting?

BY ERIK INGE BOLSØ

The Swedish company Axis Communications AB introduced a new concept several years ago, when it launched its line of camera servers. Perhaps recognizing that its line of network cameras (one of which is reviewed in *LJ*, September 2000) could not fill all the niches of old-style analog surveillance cameras, it also offered its control and digitizing electronics in a separate, rugged, fanless enclosure. Available in versions with one or four analog video inputs and one pass-through output, camera servers now have offered companies a way to modernize their surveillance systems incrementally for most of a decade.

The Camera

Naturally, the appliances run Linux these days, on rather specialized hardware. We came into contact with this server because a client had needs that could not be met by Axis' own cameras, but its Web-browser-based interface was well liked by both us and the client. So we were quoted some pro-level third-party hardware from a traditional surveillance camera supplier that was tested to work well with the camera server.

Like computer hardware, cameras and their optics are money sinks. Depending on desired sophistication, it is possible to spend any given sum you might have available.

For our purposes, we needed a remote-controllable pan-tilt-zoom (PTZ) camera for outdoor use, with a respectable amount of magnification. You can find PTZ cameras for a few hundred dollars for basic indoor versions, and a few thousand dollars for variants tolerant of outdoor climate, direct sunlight and minor vandalism.

We went with a pendant-mount enclosed clear dome system at the time—the type you might see at modern airports. For outdoor winter use in Norway, we needed a heated enclosure to avoid ice buildup. Here are the specifications:

- Pan movement: 360 degrees continuous.
- Vertical tilt: +2 to -92 degrees.
- Image sensor: 1/4 inch CCD (3.2 x 2.4mm).

- Zoom: 22X optical, focal length 4 to 88mm.
- Sensitivity: 0.07 lux at 1/1.5 s shutter speed.
- Shutter speed: 1/1.5 to 1/30,000 s.
- Minimum F-stop: f/1.6.
- Operating environment: -40 to +50 degrees Celsius, sustained.

The Problem

Now, part of the idea in using a PTZ camera for this project was periodically imaging several fixed points and uploading these images to a Webserver. Here we encountered a problem. The stock software could do periodic imaging and FTP just fine. However, we had no way to tell it to go to a PTZ position before snapping the image. The functionality was not essential for our first customer, so delivery went ahead while we researched the issue.

The software in this appliance is open to modification in a few ways. Source code is available for all open-source components of the firmware image—so the administration interface CGI is missing, but the rest is mostly available for inspection and modification. The source code for a specific firmware release is not downloadable, though; you must request it in writing, from the Axis IPR Department. They will send you the source code on a CD for a nominal fee.

We had more customers in the pipeline, so research went ahead. That work eventually yielded several APIs that could be mined for functionality.

Here are the available APIs:

- The normal admin interface (Web browser).
- HTTP API.
- Scripting.
- Shell scripting.

The PHP3 Lite dialect is the easiest option for adding custom dynamic pages to the browser-based interface, and it can also be used for general-purpose internal scripting.

- PHP3 Lite.
- GCC SDK for Linux/cris.

As with most embedded devices, there are some restrictions and inconveniences. First and foremost, the severely limited space. Less than 100kb-writable filesystem space is available for third-party modifications, out of the 4MB Flash storage. All standard software is on a read-only filesystem, not replaceable without creating a custom firmware image.

No SSH server or client was available at the time, so custom shell scripts had to be triggered by timer or run from a PHP script. There was a telnet server available for development use, however. And nowadays, Dropbear SSH has been ported to the architecture.

We wanted to make do without an additional server just for automation, if we could, so our effort went toward some internal shell scripting, triggered at a set interval by the task scheduler utask. Incidentally, this task scheduler has some extra capabilities compared to a vanilla cron—it can react to external events, like a digital input low-high transition, or loss of video signal on camera #2.

Ways of Exposing Functionality

Exposing functionality always is an issue in embedded development—how much effort should be put into easy customization? In this case, much effort has gone into it—no two surveillance installations are alike.

The browser-based interface is easy to use, featureful and meant for human consumption. It is often not especially well suited for machine automation.

The HTTP API is a simple request-response API, made for automation, offering most options available in the browser-based interface. It generally returns just a status code or the bare requested object, like a snapshot or

video stream. This lends itself well to remote control, and several third-party vendors sell software that uses this API for controlling tens to hundreds of cameras from one or a few central nodes.

The PHP3 Lite dialect is the easiest option for adding custom dynamic pages to the browser-based interface, and it can also be used for general-purpose internal scripting.

Shell scripting, via the provided sh-compatible shell and attendant utilities, is flexible and quick when some local intelligence is needed—for example,

reacting to the push of a doorbell by snapping a picture, then opening a gate via a relay.

And if some advanced local smarts are needed, the GCC SDK for the platform is available.

What Do We Have?

As previously mentioned, we chose shell scripting for the issue at hand. In the relevant firmware revision, we had quite a few programs worth using:

- BusyBox: including sash shell as /bin/sh.

Better SERVERS Better SERVICE!

3.0GHz CELERON D 64 Bit
160GB SATA2 Drive!
1GB DDR 400 RAM
1200GB Throughput (4Mbps) **\$59** /mo **LINUX**

\$199 /mo **DUAL XEON HT, 2.8GHz**
160GB SATA2 Drive!
2GB DDR 400 RAM
1200GB Throughput
64 Bit, Quad Processor!
Linux. \$219/mo for Windows Server 2003.

Visit www.Cari.net/LJ
or call 888.221.5902 today!

carinet



- mish: minix sh-compatible shell.
- utask: task scheduler, not cron-compatible.
- bufferd: image capture/buffering.
- sftpclient: simple FTP client.
- shttplib: simple HTTP client.
- smtpclient: simple mailer.

Of special note is the unobtrusive shttplib. This allows us to use the HTTP API from internal shell scripts, which we needed for PTZ control. It also could be used in many other ways, of course—signaling events to another Webservice or video server, uploading pictures via HTTP, and other things not needing more than basic authentication. It is a simple HTTP client, after all, not wget or cURL.

Problems we encountered when pushing this into production use mostly turned out to have nothing to do with the scripting. We had some intermittent failures to upload images—these turned out to be caused by a climbing vine colonizing one of the antennas for the wireless bridge.

All good things end. I'll leave you now, but first, the final script we cobbled together is shown in Listing 1. Not particularly elegant, granted, but small and not that hard to write, thanks to the consideration of the original embedded developers. That's a fair lesson to take away from this, isn't it?

Resources for this article: www.linuxjournal.com/article/8695.

Erik Inge Bolsø is a UNIX consultant and epee fencer who lives in Molde, Norway, and has been running Linux since 1996. Another of his hobbies can be found via a Google search for "balrog genealogy", and he can be reached at ljcomment@tvilsom.org.

Listing 1. Final Script

```
#!/bin/mish
PATH=/bin:/sbin:/usr/bin:/usr/sbin

# stop button pressed? skip the rest.
stopp=`ls /tmp/stopp` ;
if [ "x$stopp" = "x" ]; then
# clear working directory
bufferd -reset -buffername BILDE ;
# read configured positions, '~' signifies a unset
# position
grep -v '~' /etc/sysconfig/presetpos.conf \
>/tmp/presets ;
# pos 1-10 reserved for non-public views
num=11;
while [ $num -lt 21 ]; do
# another stop button check - break the loop
stopp=`ls /tmp/stopp` ;
if [ "x$stopp" = "x" ]; then
grep Pos$num /tmp/presets >/tmp/canptz ;
canptz=`cat /tmp/canptz` ;
if [ "x$canptz" != "x" ]; then
# go to position
shttplib "http://127.0.0.1/axis-cgi/\
com/ptz.cgi?camera=1&gotoserverpresetno=$num";
# wait for picture to settle
sleep 6 ;
# save picture to /tmp/BILDE/
bufferd -start -buffername BILDE -snapshot\
-pre 1 -format snapshot_pos$num.jpg -uri\
'ftp://127.0.0.1/jpg/1/704x576.jpg' ;
bufferd -stop -buffername BILDE ;
tmpwait=20 ;
while [ $tmpwait -gt 0 ]; do
sleep 2;
expr $tmpwait - 2 >/tmp/tmpwait ;
tmpwait=`cat /tmp/tmpwait` ;
if [ $tmpwait -eq 0 ]; then
# timeout creating jpeg, kill process and
# settle for potentially incomplete picture
logger "timeout waiting for bufferd -stop,\
killing image_buffer and continuing."
# ps is only available as a builtin
# command in /bin/sh
/bin/sh -c ps >/tmp/kverk ;
grep image_buffer /tmp/kverk >/tmp/kverk2 ;
imbuf_pid=`cut -b 0-5 /tmp/kverk2` ;
rm /tmp/kverk; rm /tmp/kverk2;
kill $imbuf_pid ;
fi;
if [ -f /tmp/BILDE/status ]; then
# status file appeared - picture complete
rm /tmp/BILDE/status ;
tmpwait=0;
fi;
done;
rm /tmp/tmpwait ;
fi;
fi;
expr $num + 1 >/tmp/A ;
num=`cat /tmp/A` ; rm /tmp/A;
done;
rm /tmp/presets ;
rm /tmp/canptz ;
# batch ftp all the pictures from this round
sftpclient -L -m 10.0.0.1 -k /tmp/BILDE -c \
/var/www/pictures/c2 -u web -w P2SsW1Rd -t \
/var/www/pictures/c2/temp_c2.jpg ;
fi;
```

EXPOSE YOURSELF TO LINUX & OPEN SOURCE:

Best Practices, Best Technology, Best Insights.

Pick the brains of the best in the Open Source industry. Hear the visionaries. See the future at the Open Source Pavilion.



Get an overview of Open Source licensing concepts and of the actual licenses that implement them, including GPL, LGPL, BSD, CDDL and others. Also gain insight into the license revision activities and references for deployment of and auditing the use of Open Source software.

It's all at FOSE!

March 7-9, 2006

Washington D.C. Convention Center

Register today at www.FOSE.com

Use Priority Code: S6LJ

Open Source approaches to modern IT security challenges



Linux & Open Source in Aerospace and Defense... a progress report



How Enterprise and Government IT is embracing Linux



Getting Started with Qtopia

From the horse's mouth (Trolltech) come instructions on how to get started writing applications for Qtopia. **BY LORN POTTER**

Trolltech's Qtopia is an embedded application framework designed to give vendors maximum customization and third-party developers the choice of developing commercial applications or free GPL applications. Qtopia is available under dual licenses. Developing for Qtopia is as easy as any Qt/KDE desktop development, except for a few extra steps and tools needed.

I started developing my program Gutenbrowser, which is a reader/downloader for the thousands of free etexts available from Project Gutenberg. It started out as a Linux-only application and soon made its way to Windows, Qtopia and then to the Open Palmtop Integrated Environment (Opie). Opie is based on Qtopia GPL and is community developed.

Qtopia is built on Qt Embedded, so an application that is created with Qt easily can be made to run in the Qtopia environment. It took about two weeks part-time/open-source-developer time to get Gutenbrowser running on the Sharp Zaurus, and that includes learning Qtopia API and cross-platform compilation as well!

If you want to port a Qt 4 application to Qtopia, it would be best to wait for Qtopia 4 to be released, as there are significant changes between Qt Embedded 2.3 and Qt Embedded 4. Qt 3, KDE and even Gtk+ applications have been ported to Qtopia versions 1 and 2 but require back-porting, class substitution and the use of mikrokde sources for the KDE programs.

Tools You Need

To get started developing for Qtopia, you need a few tools. Qtopia is Linux-only currently, so you need a Linux desktop on which to develop. You also obviously need an editor, such as emacs or vi. For this project, I chose KDevelop as it comes with a simple Qtopia application template.

If you are developing for a device, you need a cross-compiler. Our target device, the Archos PMA430 uses arm-linux-gcc version 2.95 for Qtopia. Although gcc 3 produces better optimized code, we want to run on and be compatible with software currently existing on hardware, so 2.95, as old as it is, will do. You can get ARM cross-toolchains from various sites on the Internet. In this case, Archos has a toolchain available at www.archos.com/products/overview/pma_400_sdk.html and links are also available from qtopia.net.

Source Code or SDK?

Of course, you also need Qtopia, but you have the choice of downloading the source code or using a ready-made Qtopia SDK. The SDK for the PMA430 is available in commercial and GPL versions, just like Qtopia itself. The commercial SDK can be purchased for a reasonable sum from www.trolltech.com/products/qtopia/pricing.html, and the free, GPL version can be downloaded from ftp.trolltech.com/qtopia/sdk. These install to /opt/Qtopia. Then from a command prompt do:

```
# ln -s /opt/Qtopia/sharp /opt/Qtopia/arm
```

if there is no /opt/Qtopia/arm directory.

KDevelop Project

Start KDevelop, and from the Project menu choose New Project. Open the C++ directory icon, under the directory

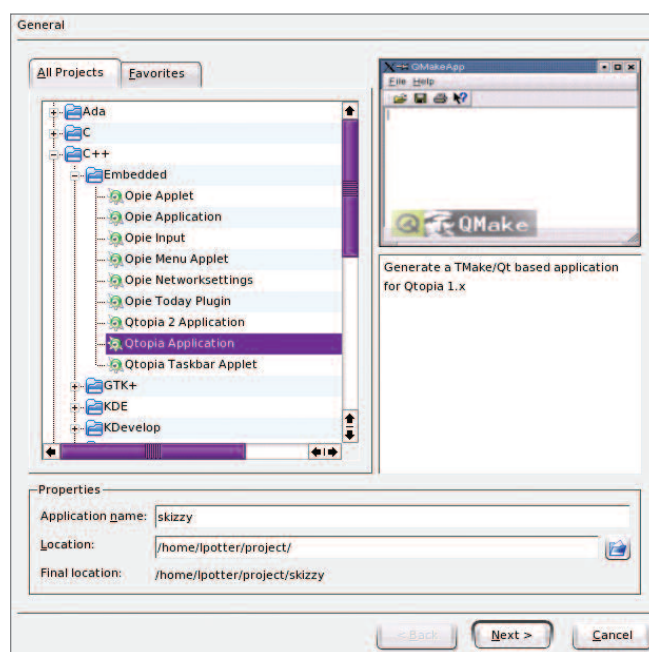


Figure 1. Creating the skizzy Project

Embedded. Click the file called Qtopia Application to start a new Qtopia project. I could name this anything, like hippopotamus, but instead I will name my project skizzy. See Figure 1 for an example dialog for creating this project.

Once you have a project, you can start editing it to suit your needs. You need to be sure to use Designer from Qt 2 when you edit .ui (user interface) files for Qtopia, as .ui files generated from later versions of Qt are not compatible. I do this by setting up a custom external tool and then opening my .ui file from within Designer 2.

Note: do *not* open by clicking on the .ui file, because Designer 3 will open up within KDevelop, and you can mangle your .ui file. Because of this, you have to run KDevelop from a command line, after exporting a few variables:

```
export PATH=/opt/Qtopia/bin:$PATH
export LD_LIBRARY_PATH=/opt/Qtopia/lib:$LD_LIBRARY_PATH
```

You also want to set up the Qt Virtual framebuffer tool, called QVfb, by pointing to /opt/Qtopia/bin/qvfb, in which the application will run on the desktop. Qtopia displays directly to the framebuffer, and therefore it does not need the overhead of the X-11 display server.

Setting Up the Development Environment

We need to set some environmental variables for our KDevelop project. Run KDevelop, and then click on Project→Project Options→Run Options. Add these variables:

```
Name: QTDIR Value: /opt/Qtopia
Name: QPEDIR Value: /opt/Qtopia
Name PATH Value: /opt/Qtopia/bin:$PATH
Name LD_LIBRARY_PATH Value:
/opt/Qtopia/lib:$LD_LIBRARY_PATH
```

Similarly, add to the Make Options, for desktop development:

```
Name: QTDIR Value: /opt/Qtopia
Name: QPEDIR Value: /opt/Qtopia
Name PATH Value: /opt/Qtopia/bin:/opt/Qtopia/tmake/bin:$PATH
Name LD_LIBRARY_PATH Value: /opt/Qtopia/lib:$LD_LIBRARY_PATH
Name TMAKEPATH Value:/opt/Qtopia/tmake/lib/qws/linux-generic-g++
```

Add -lqtopia to the LIBS line in skizzy.pro, as Qtopia 1.7 adds a new library.

At this point, you need to generate a Makefile manually, as KDevelop does not use tmake correctly:

```
# export TMAKEPATH=/opt/Qtopia/tmake/lib/qws/linux-generic-g++
# tmake -o Makefile skizzy.pro
```

Then you can build the project (F8) from within KDevelop. This little glitch will be resolved in newer versions of Qtopia that use qmake to generate Makefiles.

Getting Gritty

Let's add some functionality to skizzy.

Start the Designer 2 application and open skizzybase.ui, and delete the QLabel. Add a QTabWidget with a QComboBox on the first tab, a QListBox on the second tab and a

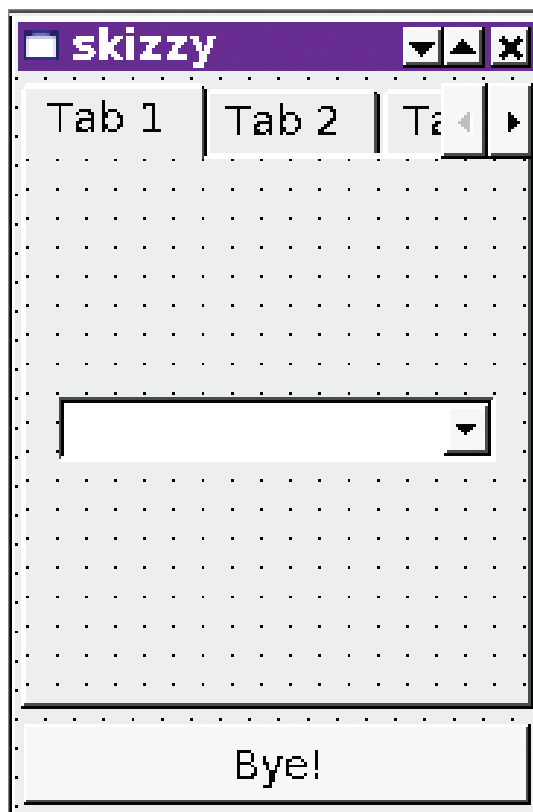


Figure 2. If you use QLayouts, it allows your application to resize depending on the display resolution or screen rotation.

QMultiLineEdit on the third tab, for example (Figure 2).

Save the .ui file.

Open the file skizzy.cpp with KDevelop. You will see our application is derived from skizzyBase:

```
skizzy::skizzy( QWidget* parent, const char* name, WFlags fl )
: skizzyBase( parent, name, fl )
```

I want to change main.cpp to a better method of constructing the application that was added since the time that the Qtopia templates for KDevelop were created.

We change the usual main() function:

```
int main( int argc, char ** argv )
{
    QPEApplication a( argc, argv );
    skizzy mw;
    a.showMainWidget( &mw );
    return a.exec();
}
```

to Qtopia's application macro:

```
QTOPIA_ADD_APPLICATION("skizzy",skizzy);
QTOPIA_MAIN
```

This allows us to create a quicklaunch application, which helps speed up startup time, by using the common application constructor that is already in memory.

My application skizzy doesn't do anything yet, so include:

```
#include <qpe/fontdatabase.h>
```

Add a private member:

```
FontDatabase fdb;
```

and some functions as private slots:

```
private slots:
void fillCombo();
void comboSelected(const QString &);
void showFont( QListBoxItem *);
```

We need to add a few lines in skizzy.cpp here for things we will use:

```
#include <qstringlist.h>
#include <qcombobox.h>
#include <qtabwidget.h>
#include <qlistbox.h>
#include <qmultilinedit.h>
#include <qfont.h>
#include <qfontinfo.h>
```

```
#include <qpe/fontdatabase.h>
```

and then add the implementations, to which we will connect our widgets' signals:

```
/*
This function uses Qtopia's FontDatabase to
fill the combobox with a list of font names.*/
void skizzy::fillCombo()
{
    QStringList families = fontdb.families();
    for ( QStringList::Iterator f = families.begin(); f != families.end(); ++f ) {
        QString family = *f;
        ComboBox1->insertItem( family);
    }
}

/*
This gets called when the combobox is selected, and
fills the listbox on the second tab with the name,
style and point size for the family of fonts
selected, and raises it. */
void skizzy::comboSelected(const QString &selectedFont)
{
    ListBox1->clear();

    QStringList styles = fdb.styles( selectedFont );
    for ( QStringList::Iterator s = styles.begin(); s != styles.end(); ++s ) {
        QString style = *s;
        QList<int> smoothies = fdb.smoothSizes( selectedFont, style );
        for ( QList<int>::Iterator points = smoothies.begin(); points != smoothies.end(); ++points ) {
            QString pointSize = selectedFont + " " + style + " " + QString::number( *points ) + " ";
            ListBox1 ->insertItem( pointSize);
        }
    }
}
```

```
TabWidget2->showPage(tab2);
}

/*
This shows example text of the selected font in
the QMultiLineWidget on the 3rd tab, and raises it.*/
void skizzy::showFont( QListBoxItem *item)
{
    QStringList fontItemString = QStringList::split(' ',item->text());
    QString family, style, point;

    family = fontItemString[0];
    style = fontItemString[1];
    point = fontItemString[2];
    bool ok;
    int i_size = point.toInt(&ok,10);

    if (!ok) {
        style += " "+fontItemString[2];
        point = fontItemString[3];
        i_size = point.toInt(&ok,10);
    }

    QFont selectedFont( family);
    selectedFont.setPointSize(i_size);

    if(style.find("Italic",0,TRUE) != -1) {
        selectedFont.setItalic(TRUE);
    }

    if(style.find("Bold",0,TRUE) != -1) {
        selectedFont.setWeight(QFont::Bold);
    }

    if(style.find("Light",0,TRUE) != -1) {
        selectedFont.setWeight(QFont::Light);
    }

    MultiLineEdit1->setFont( selectedFont);
    MultiLineEdit1->setText( tr( "The Quick Brown Fox Jumps Over The Lazy Dog" ) );
    MultiLineEdit1->setWordWrap( QMultiLineEdit::WidgetWidth);

    TabWidget2->showPage(tab3);
}
```

Qt and Qtopia use signals for sending messages between widgets. Every widget has some kind of signal it emits, and we can use the connect macros to link up functionality.

Connect ComboBox1's activated signal, and ListBox1's clicked signal to our slots, like this:

```
skizzy::skizzy( QWidget* parent, const char* name,
WFlags fl )
: skizzyBase( parent, name, fl )
{
    connect(bye, SIGNAL(clicked()), this,
SLOT(goodBye()));
    connect(ComboBox1, SIGNAL(activated(const QString
&)), this, SLOT(comboSelected(const QString &)));
    connect(ListBox1, SIGNAL( clicked ( QListBoxItem *
```

```

)), this, SLOT(showFont( QListBoxItem*)));
    fillCombo();
}

```

Notice how the slot function takes an argument of the exact same type as the signal.

In KDevelop, either press F8, or in the menu, select Build→Build Project.

It should now compile, using the native compiler. To run it, start up QVFB, and simply select Build→Execute Main Program. The skizky application should show up in QVFB.

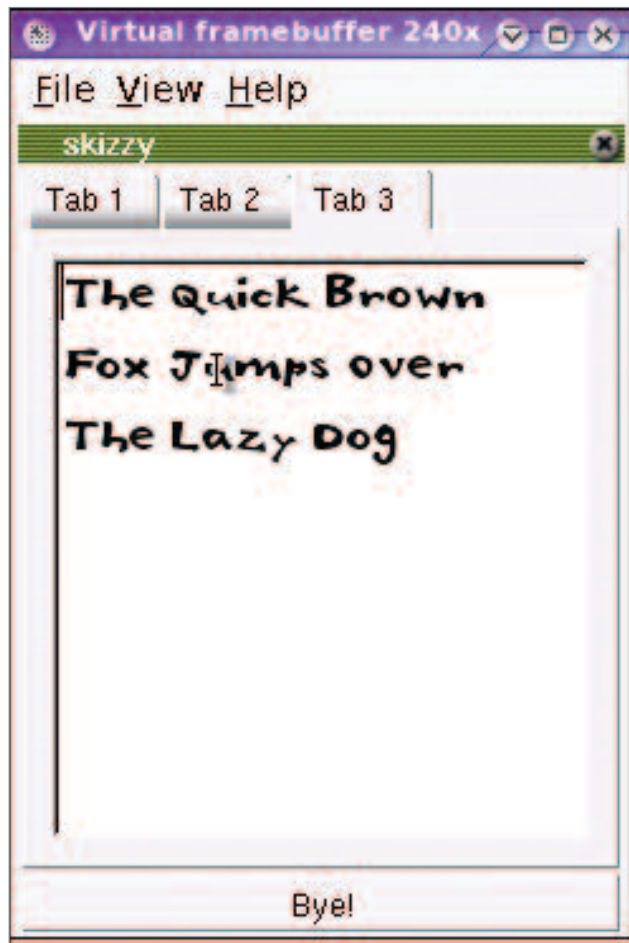


Figure 3. Skizky with real, if not useful features.

Cross-Compiling

So, now that we have a reasonably working program, we need to cross-compile this for the Archos device. We have to change the project settings to find the proper libraries.

We are now ready to cross-compile, so clean the project by selecting Build→Clean Project from the menu.

You need to change the Make Options, using the Project Options→Make Options dialog:

```

Name: QTDIR Value: /opt/Qtopia/arm
Name: QPEDIR Value: /opt/Qtopia/arm
Name PATH Value:

```

```

/usr/local/arm/bin:/opt/Qtopia/tmake/bin:$PATH
Name TMAKEPATH Value:/opt/Qtopia/tmake/lib/qws/linux-arm-g++

```

Delete the Makefile, and run the following command from the command line to create the Makefile for compiling using the arm-linux compiler:

```

# export TMAKEPATH=/opt/Qtopia/tmake/lib/qws/linux-arm-g++
# tmake -o Makefile skizky.pro

```

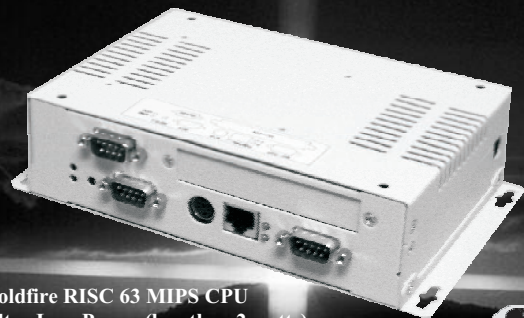
Press F8 to build the project. You can now take the resulting binary, transfer it to the Archos device using USB, and run it from there!

If you want to create an installable package, Qtopia uses Itsy Package Management (ipkg), available from handhelds.org, to install things using the Software Packages application. More information about ipkg and Qtopia development are available from Trolltech's Qtopia.net Web site.¶

Lorn Potter works for Trolltech as the Qtopia Community Manager. He is an American who lives in sunny Brisbane, Australia, with his Australian wife and son. He is a self-taught open-source programmer who is a core developer for the Opie (Open Palmtop Integrated Environment) Project. He also has worked as a musician, sound engineer and snow ski bum.



EMBEDDED WEB-SERVER



- Coldfire RISC 63 MIPS CPU
- Ultra Low Power (less than 2 watts)
- MMC/SD Flash disk up to 1 Gig
- 10/100 Base-T Ethernet
- Reliable (No Moving Parts)
- Two RS-232 & One RS232/422/485 Serial Ports
- General Purpose I/O Lines, A/D, & Optional D/A
- Optional Dial-Up Modem & CAN 2.0b Port



**Low Power (LP-SIB)
Server-In-a-Box
Starting at \$350.00
Quantity 1.**

- uClinux 2.6 with Minix Shell
- Das Uboot Bootloader with TFTP
- Eclipse Development Environment
- HTTP and FTP Servers
- PPP Dial In/Out Server & Client
- Telnet Server

Since 1985
OVER
20
YEARS OF
SINGLE BOARD
SOLUTIONS

EMAC, inc.
EQUIPMENT MONITOR AND CONTROL

Phone: (618) 529-4525 • Fax: (618) 457-0110 • www.emacinc.com

Building a Home Automation and Security System with Python

Grab some relatively inexpensive hardware and use Python to string it all together into a home security system.

BY FRED STELTER

In this article, I describe the process I used to create a home automation system using off-the-shelf products, Python and Linux. I start by describing the Serial I/O Kit, drive alert, water alarm, smoke detector and cameras that make up the hardware portion of the system. Next, I give a short description of the command-line programs and packages that are used by the Python program. I finish up by giving a brief rundown of the major parts of the Python program that tie all of the hardware and packages together.

The idea for this system was born when some of my neighbor's party guests parked in my drive for several minutes while they were trying to figure out if they were at the correct house. I was caught off-guard, because I didn't immediately notice that someone had pulled up. It became very clear to me that they could have entered my residence or shop unnoticed. I decided to create a system on my own, because I knew that a commercial security system that would monitor driveway traffic and capture images from the various cameras positioned around my property would be expensive.

Because I have coded several programs at work in Python, I have had firsthand experience with the rapid development provided by Python. It allowed me to concentrate on the problem at hand rather than complicated language syntax and semantics. I would recommend Python for any program that crunches text-based data, because it is easy enough for beginning programmers, but powerful and flexible enough to handle larger tasks.

The Python home automation system is centered around an Isolated Serial I/O Kit (Figure 1) from Quality Kits, which is available assembled for an extra \$10 US. The kit contains four inputs and eight relays. The inputs detect DC voltage from a source within the 6–24-volt range. The relays can be used to turn voltage to a source on or off, so they can be used to control the power to multiple 12V light bulbs or any other gadget that requires DC voltage within the relay's specified voltage range. The Serial I/O Kit uses simple read and write commands to a serial port connection for setting the relays and monitoring the inputs. The inputs and relay circuits are isolated, which means there is no direct connection between these circuits and the computer's serial connection. This prevents damage to the computer if something goes wrong with the Serial I/O box. Figure 2 shows how to connect the Mier Drive Alert and alarms to the Serial I/O box inputs.

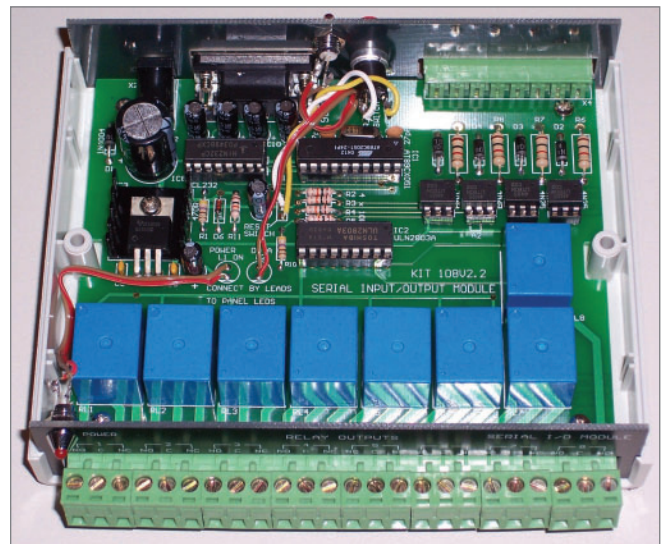


Figure 1. Isolated Serial I/O Kit

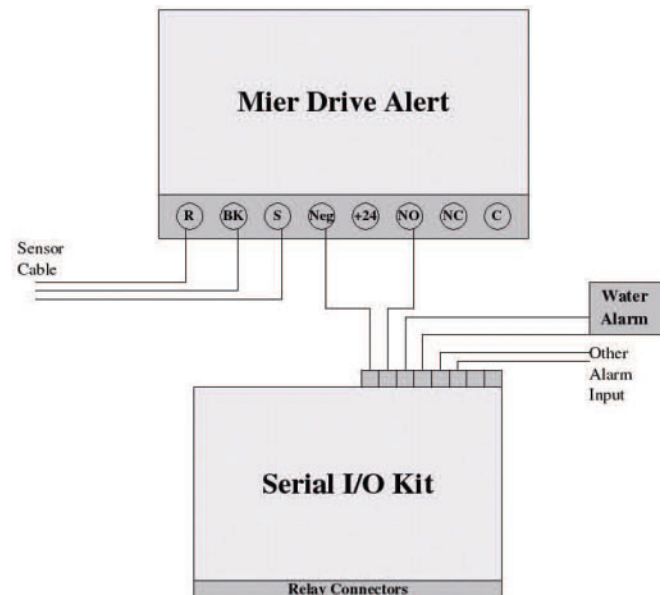


Figure 2. Connecting the Mier Drive Alert and Alarms to the Serial I/O Box Inputs

The second major part of the system is the Mier Drive Alert (Figure 3). This system is reliable, and it also provides an adjustable timed 24-volt output when activity is detected. If you decide to use the Mier Drive Alert, connect one of the Serial I/O inputs to the terminals marked Neg and NO on the Mier Drive Alert controller. Otherwise, any drive alert system that provides a voltage output in the 6–24-volt range when activity is detected should work. If you are not sure that the unit you chose provides this type of output, you probably need to contact the company.

The Mier controller box detects very small changes in voltage, which are produced by the sensor probe due to changes in the earth's magnetic field when a metal object passes by the sensor probe. This is why the sensor probe cannot be connected directly to the Serial I/O Kit's input, because the voltage produced by the probe is much smaller than the input's minimum voltage detection value of 5 volts.

Other drive alert systems are available that use a beam or a pressure-activated switch and a rubber hose. This type was once pretty common at drive-thrus and full-service gas stations. I didn't choose the beam type, because it would detect anything that moves, and I doubt that a rubber hose would have held up very well on my gravel drive. The Mier unit produces an occasional false alarm due to inclement weather and lightning strikes, but these are minimal and can be reduced if the controller sensitivity is adjusted.



Figure 3. Mier Drive Alert

As we all know, water and house interiors don't mix, which is why I added this cheap and simple water-detection circuit to the mix. The circuit performed well in a test environment using a pan of water, but lacks any real-life testing. The circuit shown in Figure 4 was created from a schematic that is listed in the on-line Resources. I removed the timer, buzzer and other unnecessary components, leaving a single switching transistor and a resistor. The circuit can be powered by a DC adapter in the 6–9-volt range. The probes in my system are simply a short piece of scrap copper tubing,



ASA COMPUTERS

www.asacomputers.com

1-800-REAL-PCS

Hardware Systems For The Open Source Community—Since 1989

(Linux, FreeBSD, NetBSD, OpenBSD, Solaris, MS, etc.)

The AMD Opteron™ processors deliver high-performance, scalable server solutions for the most advanced applications. Run both 32- and 64-bit applications simultaneously

AMD Opteron™ Value Server—\$795

- 1U 14.3" Deep
- AMD Opteron™ 140
- 512MB RAM *Max 8GB*
- 40GB IDE HDD
- 2x 10/100/1000 NIC
- Options: CD, FD or 2nd HD, RAID



Front I/O Dual AMD Opteron™ Cluster Node—\$1,375

- 1U Dual AMD Opteron™ Capable Front I/O
- Single 240 AMD Opteron™
- 1GB RAM *Max RAM 16GB*
- 80GB HDD
- Dual PCI Expansion Slot



8 Hot Swap Bays in 2U AMD Opteron™—\$1,900

- 1 of 2 AMD Opteron™ 240
- 512MB RAM *Max 16GB*
- 3x80GB IDE RAID # 5
- 2xGigE, CD+FD
- Options: SATA/SCSI, Redundant PS



No Frills AMD Opteron™ Storage Server—\$7,749

- 6TB+ IDE/SATA Storage in 5U
- Dual AMD Opteron™ 240
- 512MB RAM
- 6TB IDE Storage
- Dual GigE, CD
- Options: SATA HDD, DVD+RW etc.



Your Custom Appliance Solution

Let us know your needs, we will get you a solution



Custom Server, Storage, Cluster, etc. Solutions

Please contact us for all type of SCSI to SCSI, Fibre to SATA, SAN Storage Solutions and other hardware needs.



2354 Calle Del Mundo, Santa Clara, CA 95054

www.asacomputers.com

Email: sales@asacomputers.com

P: 1-800-REAL-PCS | FAX: 408-654-2910

Prices and availability subject to change without notice.

Not responsible for typographical errors. All brand names and logos are trademark of their respective companies.

and they just sit on the floor of my shop bathroom several inches apart. I have multiple probes connected to the same water alarm circuit, which provides multiple monitoring points per circuit at the loss of being able to identify the exact location of the leak. I leave it up to you to determine the correct probe separation for your application, because I don't have any exact distances. The water alarm (Figure 5) that I created contains two transistors and thus two alarm circuits on the same board.

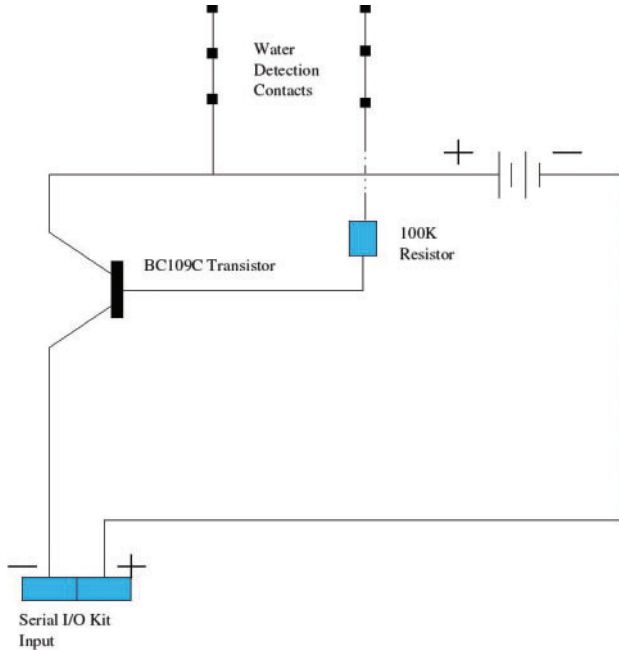


Figure 4. Circuit

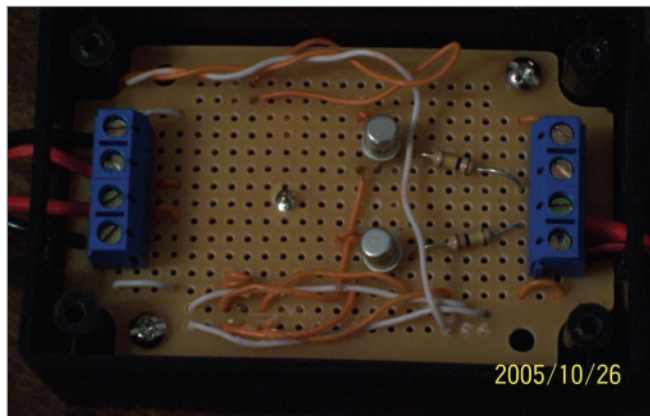


Figure 5. Water Alarm

The smoke alarm has never been tested in a real emergency and was added as an afterthought. *Do not* replace any existing smoke alarms with this modified version, because it must be tampered with. The following reasons are very good reasons not to rely on it in case a fire occurs, because if your computer, the Serial I/O Kit connection, and/or your Internet connection is the first to go, the alarm is useless. If you decide to duplicate this portion of the project, do so at your own risk. The smoke detector used in this portion of the project is just a

model with an exit light that should be available from your local home hardware center. It employs two 9-volt batteries; one to power the alarm, and the other to power the light. Now, you need to void the warranty by modifying the detector as follows. Begin by removing the cover and locating the exit light. Use a volt meter to determine the positive and negative connections to the light. Then just solder a couple of wires to the proper connection points and connect them to one of the Serial I/O Kit's inputs.

The system uses several relatively inexpensive Logitech Webcams along with a couple of network cameras. Any cam that's supported by video for Linux should work along with most network camera models. The best advice I can offer is to verify that the camera is supported by Linux and that a driver is available. Logitech Quickcam Pro models perform quite well, but not all of the Logitech models are supported by the same driver. The Quickcam Pro models use the pwc module available from saillard.org.

The system uses several command-line programs available for Linux that handle WAV-file playback, zip file creation and image file captures. The SOX package is a sound conversion and utility package for Linux. It provides the play command, which is used for playing the various alarm warning sounds. Several sound wrapper modules are available for Python, but I found that it was fairly easy to make a system call to the play command. The zip command is used to create zip files from the camera image captures. Python includes a zipfile module, but it was easier to use the command-line version, which easily can be replaced with another command, such as tar.

Image file captures are handled using Motion and Curl. The later is a fairly powerful program that will transfer data from a server. According to the man page, it handles http, https, ftp, telnet and some other formats. I make a system call to it to retrieve images from the D-Link network camera's HTTP server by using a command similar to this one, `curl http://192.168.0.98/IMAGE.JPG -uusername:password -m2 >outputfile.jpg`, where -m2 tells it to stop trying after two seconds. Motion creates MPEG motion capture and time-lapse movie files along with single JPEG image files at a preset rate. The simple HTTP server built into Motion provides an image stream, but it does not allow for single image retrieval using Curl. For more information on Motion, see the *Linux Journal* March 2005 article, "GNU Motion: Your Eye in the Sky for Computer Room Surveillance" by Phil Hollenback, because it does a good job explaining the details required to get it up and running.

Once you have a working Motion setup, you should change the `snapshot_interval` in Motion's config file to a one or a value acceptable to your application. Lower values are better, because they prevent duplicate image captures. Now set the `snapshot_filename` value to a temporary filename, such as, `tempmotionimagefile`, so that Motion creates only a single capture file. Leaving the default `motion.conf` `snapshot_filename` will work, but Motion will create a new file using the `snapshot_interval` value, which can result in a very large number of stored image files. These settings cause Motion to create a symbolic link named `lastsnap.jpg` that always points to the latest snapshot file, which in the case of the settings above, always is overwritten with the most recent image.

Now that all of the hardware and required programs above are configured, installed and functioning on their own, it is time to discuss the Python program that glues all of it together. I briefly touch on the most important parts of the program, along with any second-party modules that are required. The `pyserial` module contains a `posix` serial I/O implementation for Linux and is used to handle reads and writes to the Serial I/O Kit. A thread-enabled Python installation is required by the program, because it must perform several simultaneous operations in order to monitor input activity and so forth.

The `MonitorInputs` class is one of the most important, because it handles input monitoring by employing the `GetInputStatus` method (Listing 1). This method checks for input activity by writing the input name (I1 to I4) to the serial port. The write method is provided by the `Serial` class in the `pyserial` module. Notice that there is a Python list defined as `Expect` in the example code. This is a list of the expected output from the serial class read command executions that are required to clear and validate information that is returned from the write request. If an unexpected read result is obtained, the serial port connection is reset. This allows the program to recover from serial communication failures and short power losses to the Serial I/O Kit.

Listing 1. Employing the `GetInputStatus` Method

```
def GetInputStatus(self, Input = None):
    self.ser.write(Input + '\r\n')
    Expect = [Input[0], Input[1], '\r', '\n']
    cnt = 0
    while cnt <= 3:
        a = self.ser.read()
        if a == '' or a != Expect[cnt]:
            return -1
        cnt += 1
    val = self.ser.read()
    Expect = ['\r', '\n', '#']
    cnt = 0
    while cnt <= 2:
        a = self.ser.read()
        if a == '' or a != Expect[cnt]:
            return -1
        cnt += 1
    if val == '1':
        return 1
    else:
        return 0
```

When activity is detected, `MonitorInputs` checks to make sure that input activity has not occurred within the input activity timeout. The timeout is used to keep the alarm threads, which simply send a plain-text e-mail, from executing too many times during a single input activity voltage-on condition. The timeout is not the best solution, because the smoke and/or water alarm would still send a new e-mail every 60 seconds. This is acceptable to me, because if I receive a water alarm e-mail while at work,

I'm going to rush home. The unused Serial I/O Kit relays could be used to correct this shortcoming, because each input positive connection could be routed through a relay, which could be turned off to disable the alarm voltage.

Another solution is to signal the `GetInputStatus` method to ignore input activity on a specific input. Either method will work, but a remote trigger mechanism will be required in either case, because the serial port connection is maintained by the home automation program. A possible solution adds a server thread to the home automation program that would accept simple string commands from a client connection. This would allow a simple Python CGI script to send commands that could control input monitoring and/or the relay states. `Pyro` is a Python distributed object system that provides another more complex solution using an event server. This is very similar to the client/server approach, but `Pyro` is more robust and provides opportunities beyond the scope of this article. One of these solutions will probably find its way into a future upgrade to the home automation program.

Now that the program is monitoring for input activity, it needs to produce notifications, such as a warning sound or e-mail when activity is detected. Smoke and water alarm activities are handled by the generic threaded `Alarm` class, and drive alert activity is handled separately. The `Alarm` class

Linux Laptops

Starting at \$985

DON'T BE SQUARE!
GET CUBED!



R³ Technologies
Working Hard to Bring Technologies to Life

309.34.CUBED
shoprcubed.com

Listing 2. DriveAlert Class

```

class GetImage(threading.Thread):
    def __init__(self, cam = None, numImages = 1):
        self.cam = cam
        self.JobBegin = -1
        self.camCmd = CamCOMMANDS[cam]
        self.numImages = NumCamImages[cam]
        self.Zip = None
        threading.Thread.__init__(self)

    def run(self):
        for i in range(self.numImages):
            self.JobBegin =
int(time.strftime("%H%M%S",time.localtime(time.time())))
            if QUIET == 0:
                print 'Getting %s image' %self.cam
                filename = time.strftime("%H%M%S",
time.localtime(time.time())) + '.jpg'
                execcmd = self.camCmd %filename
                self.p = popen2.Popen3("exec " + execcmd, 1024)
                self.errReader = PipeReader(self.p.childerr);
self.errReader.start()
                self.outReader = PipeReader(self.p.fromchild);
self.outReader.start()
                try:
                    self.p.wait()
                except OSError, (errno, errnostr):
                    if QUIET == 0:
                        print 'ERROR: GetImage self.p.wait Errno %s: %s'
%(`errno`, `errnostr`)
                except:
                    if QUIET == 0:
                        print 'ERROR: self.p.wait Unknown error'
                time.sleep(IMAGE_DELAY)
            #Popen complete - create zipfiles
            self.Zip = ZipIt(self.cam)
            self.Zip.start()
            self.Zip.join() #Wait on zip file creation

```

plays a WAV file using the PlayWav class, and it also sends a notification e-mail using the MailAttachment class. The PlayWav class uses a popen call to the wavcmd value (sox play command) set in the configuration file. The PlayWav class is threaded to prevent a busy sound device from holding up the e-mail notifications. The end result of all of the threaded classes is that the input activity is monitored almost continuously with only slight delays.

The DriveAlert class handles detected input activity for the drive alert signal. This class employs the GetImage (Listing 2), PlayWav and

SSHRemote threaded classes. A new GetImage instance is created for each camera command (camcmd) set in the configuration file, so that images can be collected from each camera at about the same time. The GetImage class makes a popen call to the camera command and waits until it has completed. This is repeated until the number of images set in the configuration file have been collected and saved in the directories defined in the camdir section of the configuration file. Once all of the images have been collected, the GetImage class uses the ZipIt class to create a zip file via a

popen call to the zip command. When all of the image files are zipped up, the MailAttachment routine is used to e-mail the zip files. If you would like to stagger the images collected from the cameras, you can add a camera image delay section to the configuration file and modify the GetImage class by adding a call to the sleep function using the preset camera delay as input.

I briefly mention the SSHRemote class because the name is ambiguous. This class could be used to execute any command by replacing the ssh remote command in the configuration file with another one. I currently use it to play some tunes on my shop machine to make it appear that someone is home. The ssh call executes another simple Python script on the remote machine, which uses the play command to play all WAV files in a specified directory.

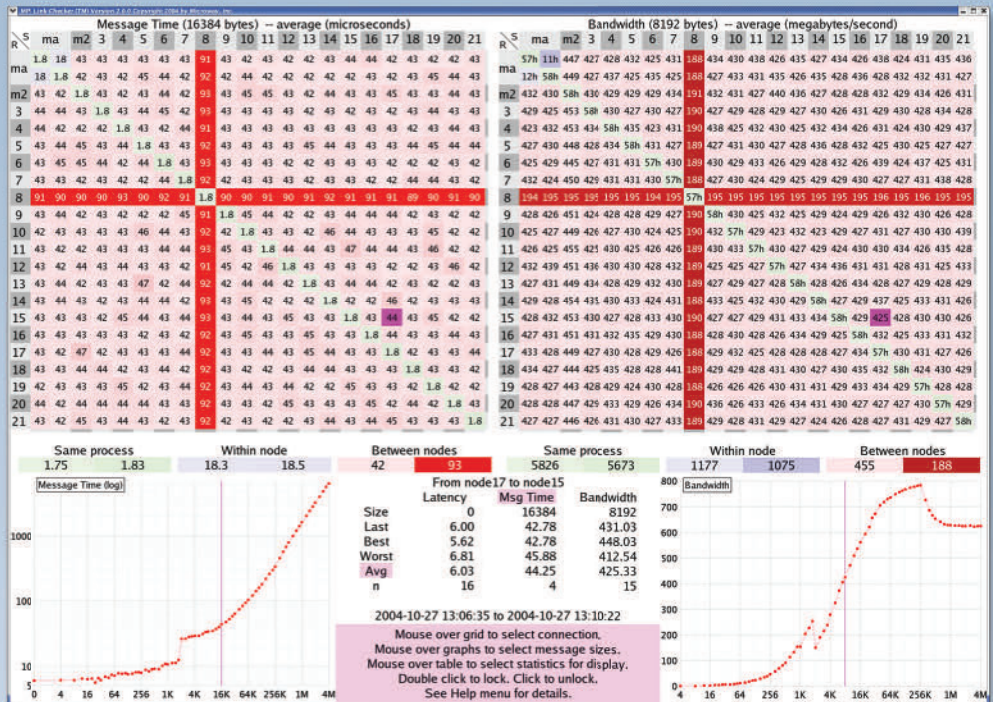
This article shows how Linux, Python and some cheap off-the-shelf hardware can be used to create a home automation system in a reasonable amount of time. The article focuses on the main parts of the system and cannot possibly describe the setup of all of the required components in detail. I must also stress that this system has not been tested in a production environment and therefore comes with no guarantees, express or implied, as to its suitability for any of the purposes listed above, so use it at your own risk. I am looking forward to making future enhancements, such as a voice modem that will dial a preset number and play a message. This will supplement the unreliable e-mail notifications, which are often delayed. I hope this article sparks your interest in simple monitoring systems and the flexibility of the Serial I/O Kit used in this project.

Resources for this article:

www.linuxjournal.com/article/8696

Fred Stelter has a BS in Computer Science from Baylor University in Waco. When he's not writing code for a local company, he likes to pop some tires at the local mountain bike trails, work on his hot rod or occasionally hit the water for some kneeboarding.

X Marks the Slow Node!



MPI Link-Checker™ to the Rescue!

A single slow node or intermittent link can cut the speed of MPI applications by half. Whether you use GigE, Myrinet, Quadrix, InfiniBand or InfiniPath HTX, there is only one choice for monitoring and debugging your cluster of SMP nodes:

Microway's MPI Link-Checker™

Our unique diagnostic tool uses an end-to-end stress test to find problems with cables, processors, BIOS's, PCI buses, NIC's, switches, and even MPI itself! The newest release provides ancillary data on inter-process and intra-CPU latency which can vary by a factor of 10 between MPI versions. MPI Link-Checker is also useful for porting applications to new hardware. It provides instant details on how latency and bandwidth vary with packet size. It is available now for a free 30 day evaluation!

Wondering what's wrong with your cluster, or need help designing your next one? Call our HPC staff at 508-746-7341. Visit microway.com to learn about new low latency interconnects including the PathScale InfiniPath HTX Adapter, which delivers unmatched MPI latency of under 1.5 microseconds.

Microway has been an innovator in HPC since 1982. We have thousands of happy customers. Isn't it time you became one?

Call us first at 508-746-7341 for quotes and benchmarking services. Find technical information, testimonials, and newsletter at www.microway.com.



Microway® Quad Opteron™ Cluster with 36 Opteron 852s, redundant power and 45 hard drives in CoolRak™ cabinet.



Embedding the db4o Object-Oriented Database

How to get started using this small-footprint object-oriented database in your embedded system programs. **BY RICK GREHAN**

db4o is an open-source, object-oriented database from db4objects. It is embeddable, in the sense that the entire db4o engine is supplied as a single library that links into your application. Versions of db4o exist for Java and .NET/Mono. Consequently, it's an ideal OODB engine for Linux applications written in either the Java or .NET frameworks. In this article, the examples provided have been written in C# and tested in Mono running on an Ubuntu Linux system. (We also ran the applications on a Monoppix system.)

Besides being an open-source library—so you can download it instantly and begin experimenting with its capabilities—db4o's other outstanding characteristic is its terse, easily grasped API. In most cases, you'll use methods drawn from a set of nine fundamental calls. In addition, the library's memory footprint is modest, making it ideal for resource-constrained applications (though by no means is db4o incapable of enterprise-level work).

Despite its small footprint and uncomplicated programming interface, db4o provides all the features you'd expect from a commercial database engine: it allows multiuser access, any access on the database is invisibly wrapped in a transaction and all operations adhere to ACID principles (atomicity, consistency, isolation and durability).

Unlike some object-oriented and object-relational database systems, db4o does not require you to pass your code through an instrumentation pre- or post-compilation step. Nor must classes whose objects are to be made persistent be derived from a special persistence-aware superclass. db4o is happy to work with ordinary objects, and you need not inform it of an object's structure before you store that object into a db4o database.

As we hope to show, this provides us with some unexpected capabilities.

A Dictionary Database

Suppose our application is a dictionary—a dictionary in the classic sense. That is, the application manipulates a database that stores words and their definitions. In such an application, we might define a class to model dictionary entries as follows:

```

/*
 * DictEntry
 */

using System;
using System.Collections;

namespace PersistentTrees
{
    /// <summary>
    /// DictEntry class
    /// A dictionary entry
    /// </summary>
    public class DictEntry
    {
        private string theWord;
        private string pronunciation;
        private ArrayList definitions;

        public DictEntry()
        {
        }

        // Create a new Dictionary Entry
        public DictEntry(string _theWord,
            string _pronunciation)
        {
            theWord = _theWord;
            pronunciation = _pronunciation;
            definitions = new ArrayList();
        }

        // Add a definition to this entry
        // Note that we do not check for duplicates
        public void add(Defn _definition)
        {
            definitions.Add(_definition);
        }

        // Retrieve the number of definitions
        public int numberOfDefs()
        {

```

```

        return definitions.Count;
    }

    // Clear the definitions array
    public void clearDefs()
    {
        definitions.Clear();
        definitions.TrimToSize();
    }

    // Properties
    public string TheWord
    {
        get { return theWord; }
        set { theWord = value; }
    }

    public string Pronunciation
    {
        get { return pronunciation; }
        set { pronunciation = value; }
    }

    // Get reference to the definitions
    public ArrayList getDefinitions()
    {
        return definitions;
    }
}
}

```

A DictEntry object consists of three elements: the word itself, its pronunciation and a list of definitions. Meanwhile, a class for describing definition objects might look like this:

```

/*
 * Defn
 *
 */
using System;
namespace PersistentTrees
{
    /// <summary>
    /// Description of Class1.
    /// </summary>
    public class Defn
    {
        public static int NOUN = 1;
        public static int PRONOUN = 2;
        public static int VERB = 3;
        public static int ADJECTIVE = 4;
        public static int ADVERB = 5;
        public static int CONJUNCTION = 6;
        public static int PARTICIPLE = 7;
        public static int GERUND = 8;

        private int pos;
        private string definition;

        public Defn(int _pos,

```

```

        string _definition)
    {
        pos = _pos;
        definition = _definition;
    }

    // Properties
    public int POS
    {
        get { return pos; }
        set { pos = value; }
    }

    public string Definition
    {
        get { return definition; }
        set { definition = value; }
    }
}
}

```

So, a Defn object includes an integer member indicating the part of speech and a string member that holds the text for the definition. This structure allows us to associate multiple definitions with a single entry in the dictionary. Storing such items into a db4o database is marvelously



Conférence PHP Québec

March 29-31 2006

Advanced PHP Techniques
Professional PHP Development
Database

Plaza Montreal Hotel
505 Sherbrooke East St.
Montreal (Quebec)

Zend
PHP Certification

http://conf.phpquebec.com

Sponsored by








simple. Assume that we want to add the word float to our dictionary and provide it with three definitions:

```
Defn _float1 = new Defn(VERB, "To stay on top of a liquid.");
Defn _float2 = new Defn(VERB, "To cause to float.");
Defn _float3 = new Defn(NOUN, "Anything that stays on top of water.");

DictEntry _float = new DictEntry("float", "flote");
_float.add(_float1);
_float.add(_float2);
_float.add(_float3);
```

At this point, we have a DictEntry object, _float, whose list of definitions includes three items.

First, we open the database itself. A db4o database is modeled by an ObjectContainer object, and we can open (or create, if it doesn't exist) an ObjectContainer with:

```
ObjectContainer db = Db4o.openFile("<filename>");
```

where <filename> is the path to the file that holds the persistent content of the ObjectContainer. You put an object into the ObjectContainer using the set() method. So, we can store our new definition with:

```
db.set(_float);
```

which, believe it or not, is just about all you need to know about the set() method. That one call stores not only the _float DictEntry object, but all of its contained Defn objects as well. When you call db4o's set() method, the db4o engine invisibly spiders through the object's references, persisting all the child objects automatically. Just pass set() the root object of a complicated object tree, and the whole shebang is stored in one shot. You don't have to tell db4o about your object's structure; it discovers it automatically.

To retrieve an object from an ObjectContainer, we locate it with the help of db4o's QBE (query by example) mechanism. A QBE-style query is guided by an example, or template, object. More specifically, you perform a query by creating a template object, populating its fields with the values you want matched, showing the template object to the query system and saying, "See this? Go get all the objects that look like this one."

So, assuming you want to retrieve our definitions for float, the process looks something like this:

```
// Create template
DictEntry DTemplate = new DictEntry("float", "");

// Execute QBE
ObjectSet results = db.get(DTemplate);

// Iterate through results set
while(results.hasNext())
{
    DictEntry _entry = (DictEntry)results.next();
    ... process the DictEntry object ...
}
```

First, we create the template object, filling the fields we're interested in with the values we want matched. Fields that shouldn't participate in the query are filled with zero, the empty string, or null—depending on the data type. (In the above example, we're simply looking for the word float in the dictionary. We put an empty string in the pronunciation field for the templater object constructor, because the pronunciation is irrelevant to the query.)

Then, we execute the query by calling the ObjectContainer's get() method, with the template object passed in as the single argument. The query returns an ObjectSet, through which we can iterate to retrieve the results of the match.

Adding Indexes

At this point, we can easily create a database, fill it with words and definitions, and retrieve them using db4o's QBE mechanism. But, what if we want to experiment with different indexing-driven retrieval mechanisms? Because the database preserves relationships among the persistent objects, we can create custom indexing and navigation structures, place them in the database as well and "wire" our data

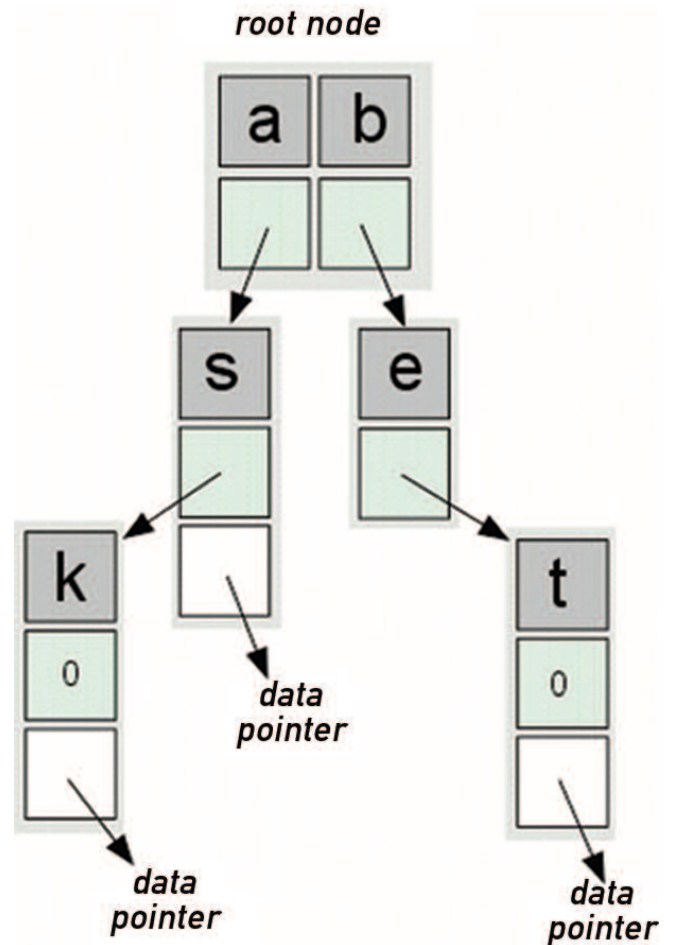


Figure 1. A trie. In a trie index, individual characters within a word are stored at different node levels. This particular trie holds three words: as, ask and bet. The data pointers are actually references to the DictEntry objects associated with the corresponding words.

Listing 1. Binary Tree

```
/*
 * Binary Tree
 */
using System;
using com.db4o;

namespace PersistentTrees
{
    /// <summary>
    /// Description of BinaryTree.
    /// </summary>
    public class BinaryTree
    {
        // The tree's root
        private TreeNode root;

        public BinaryTree()
        {
            root = null;
        }

        public static BinaryTree nullfactory()
        {
            return(new BinaryTree());
        }

        // insert
        // Add key to tree with associated object reference
        public void insert(string _key, Object _data)
        {
            // Use recursion for this
            root = insert(root, _key, _data);
        }

        // insert
        // This is worker method for inserting key and data
        // Insert _key into subtree t with _data associated
        private TreeNode insert(TreeNode t, string _key, Object _data)
        {
            // If this subtree is empty, build a new node
            if(t == null)
                t = new TreeNode(_key, _data);
            else
                if(_key.CompareTo(t.Key)<=0)
                    t.Left = insert(t.Left, _key, _data);
                else
                    t.Right = insert(t.Right, _key, _data);

            return(t);
        }

        // search
        // Search for a key in the tree.
        // Return the array from the TreeNode if found, null if
        // not
        // db is the ObjectContainer holding the tree.
        public Object[] search(string _key,
                               ObjectContainer db)
        {
            TreeNode t = search(root, _key, db);
            if(t==null) return(null); // Not found
            db.activate(t,4); // Activate to get data
            return(t.getData());
        }

        // search
        // This is the worker method for searching.
        private TreeNode search(TreeNode t,
                                string _key,
                                ObjectContainer db)
        {
            // Empty tree?
            if(t==null) return(null);
            if(_key.CompareTo(t.Key)==0) return(t);
            if(_key.CompareTo(t.Key)<0)
            {
                db.activate(t.Left,2);
                return(t = search(t.Left, _key, db));
            }
            db.activate(t.Right,2);
            return(t = search(t.Right, _key, db));
        }
    }
}
```

objects into those structures.

We illustrate how simple this is by creating two dissimilar indexing schemes.

First, we create a binary tree. Each node of the tree carries as its payload a key/data pair. The key will be a text word from the dictionary, and the associated data item will be a reference to the DictEntry object in the database. So, we can fetch the binary tree from the database, execute a search for a specific word in the dictionary and fetch the matching DictEntry object (if found).

The architecture and behavior of binary trees are well known, so we won't go into much detail about them here. (In fact, many frameworks now provide them as standard data structures. We've created an explicit one to show how easily it can be stored in the database.) Our implementation appears in Listing 1. It is rudimentary, supporting only insertion and searching. It doesn't guarantee a balanced tree, but it serves for the purposes of illustration. The TreeNode class, which defines the structure of nodes within the binary tree, appears in Listing 2. (Note, we'll explain the purpose of the calls to db.activate()

Listing 2. TreeNode Class

```

/*
 * TreeNode
 */
using System;

namespace PersistentTrees
{
    /// <summary>
    /// Description of TreeNode.
    /// </summary>
    public class TreeNode
    {
        public TreeNode()
        {
        }

        private TreeNode left; // Left child
        private TreeNode right; // Right child
        private string key; // Key for this node
        private Object[] data; // Data associated with key

        // Create a new TreeNode, loaded with
        // key and data.
        public TreeNode(string _key, Object _data)
        {
            left = null;
            right = null;
            key = _key;
            data = new Object[1];
            data[0] = _data;
        }

        // addData
        // Adds new data item to an existing node.
        // The array is extended.

        public void addData(Object _data)
        {
            Object[] newdata = new Object[data.Length+1];
            Array.Copy(data,0,newdata,0,data.Length);
            newdata[data.Length]=_data;
            data = newdata;
        }

        // Property access
        public TreeNode Left
        {
            get { return left; }
            set { left = value; }
        }

        public TreeNode Right
        {
            get { return right; }
            set { right = value; }
        }

        public string Key
        {
            get { return key; }
            set { key = value; }
        }

        public Object[] getData()
        {
            return data;
        }
    }
}

```

in Listing 1 shortly.)

Next, I create a trie, an indexing data structure specialized for searching text words. It is built as a series of nodes arranged in levels—each level holds a set of characters and associated pointers such that the characters on the topmost (or, root) level correspond to letters in a word’s first character position; characters in the second level correspond to letters in the second character position, and so on. References associated with each character serve to “string” characters like beads on a thread, so that following a thread from the root down into the tree spells out the word being searched for.

If this is difficult to visualize, the illustration in Figure 1 should help.

Inserting a new word into a trie is relatively simple. Starting with the first matching character, you examine the root node to see whether that character exists. If not, add it, and from that point on, the algorithm inserts new nodes

(each initialized with a subsequent letter) as it works through the target word. If the character does exist, the algorithm follows the associated pointer to the next level, and the examination process repeats. Ultimately, you’ve accounted for each character in the word, and the node you’re on is the node on which you attach the data reference.

Searching a trie is equally simple. Start at the root, and look for the first character. If the character is found, follow the associated reference to the next node; else, return a “not found” error. Otherwise, move to the next character, repeat, and if you get through the whole word, the data node associated with the terminal character points to the DictEntry object.

The code for the trie is shown in Listing 3.

As the code for inserting and searching both binary trees and tries illustrates, we can work with database objects almost as though they were purely in memory

Listing 3. Trie

```

/*
 * Trie
 */
using System;
using com.db4o;

namespace PersistentTrees
{
    /// <summary>
    /// Description of Trie.
    /// </summary>
    /// trie class
    public class Trie
    {
        private TriePnode root;      // Root of Trie

        // Constructor
        public Trie()
        {
            root = null;
        }

        // insert
        // Insert a key/data pair into the tree.
        // Allows duplicates
        public void insert(string key, // Key to insert
            Object data)              // Data assoc. with key
        {
            TriePnode t = root;
            TriePnode parent = null;
            int index=0;
            int slen = key.Length;
            for(int i=0; i< slen; i++)
            {
                char c = key[i];
                // If a node doesn't exist -- create it
                if(t == null) t = new TriePnode();
                // If this is the first node of the tree,
                // it is the
                // root. Otherwise, it is stored in the
                // pnodes array
                // of the parent
                if(i==0)
                    root = t;
                else
                    parent.setPnodePointer(index, t);

                // If the character is not on the node,
                // add it
                if((index=t.isCharOnNode(c))== -1)
                    index = t.addKeyChar(c);
                if(i == slen-1) break;
                parent = t;
                t = t.getPnodePointer(index);
            }

            // Finally, add the data item
            t.addData(index, data);
        }

        // search
        // Searches for a string in the trie.
        // If found, returns the Object[] data array associated.
        // Else, returns null
        // db is the ObjectContainer holding the trie
        public Object[] search(string _key,
            ObjectContainer db)
        {
            TriePnode t;
            char c;
            int index=0;

            // Empty trie?
            if((t=root)==null) return(null);

            int slen = _key.Length;
            for(int i=0; i<slen; i++)
            {
                c = _key[i];
                if((index=t.isCharOnNode(c))== -1) return(null);
                if(i==slen-1) break;
                db.activate(t,2);
                t = t.getPnodePointer(index);
            }

            // Get the data
            db.activate(t,3);
            return(t.getDnodePointers(index).getData());
        }
    }
}

```

objects. Specifically, we can attach an object to an index simply by storing its object reference in the data reference element.

In addition, because the database makes no distinction between index objects and data objects, we need not create a separate index and data files. This keeps everything in one

place, which is actually more of an advantage than one might first suppose.

Code for reading a text file with words and definitions, creating DictEntry objects and storing them in the database, and also building binary tree and trie indexes *and* attaching the DictEntry objects to them looks like this:


```

string theword;
string pronunciation;
int numdefs;
int partofspeech;
string definition;
DictEntry _dictEntry;

// Open a streamreader for the text file
FileInfo sourceFile = new FileInfo(textFilePath);
reader = sourceFile.OpenText();

// Open/create the database file
ObjectContainer db = Db4o.openFile(databaseFilePath);

// Create an empty Binary tree, and an empty trie
BinaryTree mybintree = new BinaryTree();
Trie mytrie = new Trie();

// Sit in an endless loop, reading text,
// building objects, and putting those objects
// in the database
while(true)
{
    // Read a word.
    // If we read a "#", then we're done.
    theword = ReadWord();
    if(theword.Equals("#")) break;

    // Read the pronunciation and put
    // it in the object
    pronunciation = ReadPronunciation();
    _dictEntry = new DictEntry(theword, pronunciation);

    // Read the number of definitions
    numdefs = ReadNumOfDefs();

    // Loop through definitions. For each,
    // read the part of speech and the
    // definition, add it to the definition
    // array.
    for(int i=0; i<numdefs; i++)
    {
        partofspeech = ReadPartOfSpeech();
        definition = ReadDef();
        Defn def = new Defn(partofspeech, definition);
        _dictEntry.add(def);
    }
    // We've read all of the definitions.
    // Put the DictEntry object into the
    // database
    db.set(_dictEntry);

    // Now insert _dictEntry into the binary tree
    // and the trie
    mybintree.insert(_dictEntry.TheWord, _dictEntry);
    mytrie.insert(_dictEntry.TheWord, _dictEntry);
}

// All done.
// Store the binary tree and the trie

```

```

db.set(mybintree);
db.set(mytrie);

// Commit everything
db.commit();

```

This, of course, presumes a number of helper methods for reading the source file, but the flow of logic is nonetheless apparent. Notice again that we were able to store each index—in entirety—simply by storing the root with a single call to `db.set()`.

Fetching something from the database is only somewhat trickier. As much as we'd like to treat persistent objects identically to transient objects, we cannot. Objects on disk must be read into memory, and this requires an explicit fetch. The initial fetch, of course, is a call to `db.get()` to locate the root of the index. So, code that allows us to search for a word using either the binary tree or the trie index would look like this:

```

public static void Main(string[] args)
{
    Object[] found;
    DictEntry _entry;

    // Verify proper number of arguments
    if(args.Length !=3)
    {
        Console.WriteLine("usage: SearchDictDatabase <database> B|T <word>");
        Console.WriteLine("<database> = path to db4o database");
        Console.WriteLine("B = use binary tree; T = use trie");
        Console.WriteLine("<word> = word to search for");
        return;
    }

    // Verify 2nd argument
    if("BT".IndexOf(args[1])!=-1)
    {
        Console.WriteLine("2nd argument must be B or T");
        return;
    }

    // Open the database file
    ObjectContainer db = Db4o.openFile(args[0]);
    if(db!=null) Console.WriteLine("Open OK");

    // Switch on the 2nd argument (B or T)
    if("BT".IndexOf(args[1])==0)
    { // Search binary tree
        // Create an empty binary tree object for the
        // search template
        BinaryTree btt = new BinaryTree();
        ObjectSet result = db.get(btt);
        BinaryTree bt = (BinaryTree) result.next();

        // Now search for the results
        found = bt.search(args[2].db);
    }
    else
    { // Search trie

```

```

// Create an empty trie object for the search
// template
Trie triet = new Trie();
ObjectSet result = db.get(triet);
Trie mytrie = (Trie) result.next();

// Now search for the results
found = mytrie.search(args[2],db);
}

// Close the database
db.close();

// Was it in the database?
if(found == null)
{
    Console.WriteLine("Not found");
    return;
}

// Fetch the DictEntry
_entry = (DictEntry)found[0];

... <Do stuff with _entry here> ...

```

And now we can explain the purpose of the calls to `db.activate()` in the search methods of both Listings 1 and 3.

When you call the `db.set()` method, as we explained, the `db4o` engine spiders through the object tree, persisting all reachable objects. (This is known as persistence by reachability.) In the reverse direction—that is, calling `db.get()` to fetch an object—`db4o` does not pull the entire object tree out of the database. If it did that, then fetching the root of, for example, the binary index, would cause `db4o` to pull the entire index, plus all the dictionary entries, plus all the definitions into memory at once—not very efficient if we want only one word.

Instead, `db4o` uses a concept called activation depth. Suppose I've fetched object A into memory from a `db4o` database using a `db.get()` call. If I then call `db.activate(A,6)`, that tells `db4o` also to fetch into memory all objects referenced by A, up to a depth of 6. So, the `db.activate()` calls that are sprinkled throughout the search routines of the binary tree and the trie classes ensure that the search operation always pulls in enough of the index so that the search can proceed. (And, at the end of a successful search, the dictionary objects are fetched.)

Custom-Made Indexes

OO databases provide the developer with flexibility not so easily gotten with an RDBMS. In particular, you can design complex, deep object structures, persist them to a database and not have to concern yourself with the translation between the object model and the relational model.

The OO database `db4o`'s simple-to-grasp API did not hinder our building indexing structures in the database side by side with the actual data. Though the binary tree and trie indexes we chose were uncomplicated, they demonstrate that the developer is free to augment a database with custom indexing and navigation structures of arbitrary complexity. So, we can tailor-make an organization scheme that fits the application's requirements of its data, and we can design it using plain-old objects—Java or Mono/.NET. Best of all, `db4o` is open source, so there's nothing stopping you from grabbing it for your next database application. For more information concerning `db4o`, see www.db4objects.com. ■

Rick Grehan's articles have appeared in *Byte*, *Embedded Systems Journal*, *JavaPro*, *InfoWorld*, *Microprocessor Report* and several other journals. He is coauthor of three books: one on remote procedure calls, another on embedded systems and a third on object-oriented Java databases. Currently, he is QA Lead at Compuware's NuMega Labs.

Freedom and Innovation
Innovación y Libertad

Software Libre
II Conferencia Internacional
II Open Source World Conference **Málaga 2006**

II Open Source World Conference
Malaga, Spain, February 15-17, 2006

II Conferencia Internacional de Software Libre
Málaga, 15-17 de febrero, 2006

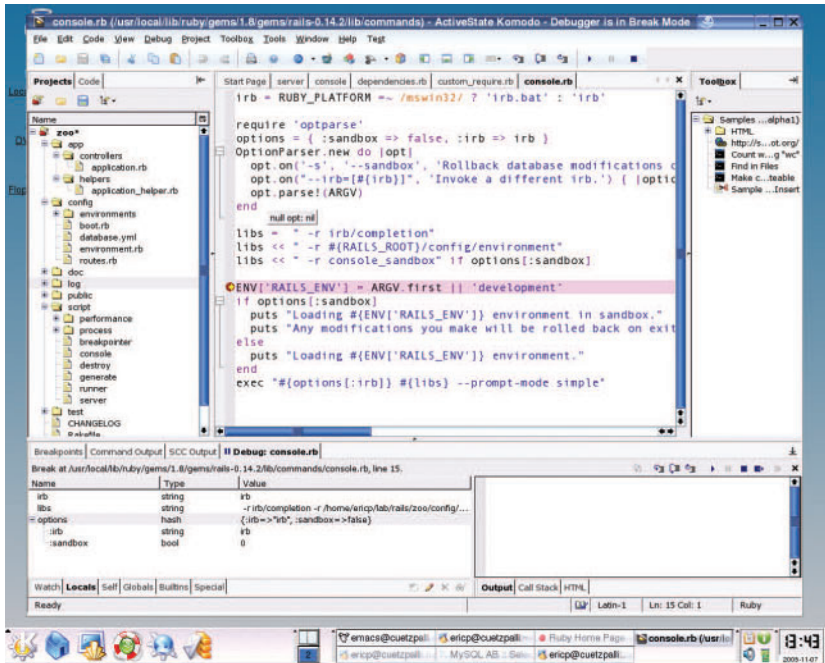
www.opensourceworldconference.com

See you in Malaga!

Organized by Organiza

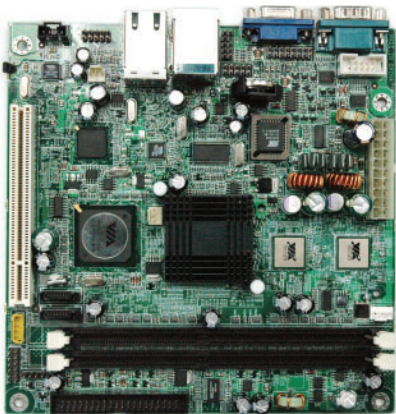
JUNTA DE ANDALUCIA JUNTA DE EXTREMADURA

ActiveState's Komodo 3.5



Ruby developers of the world, rejoice! ActiveState's latest release (3.5) of its Komodo integrated development environment has added support for the up-and-coming language. With the addition of Ruby, Komodo now supports five different languages, including Python, Perl, Tcl and PHP. Release 3.5 also includes support for Mac OS X and Ruby on Rails. Personal licenses are only \$29.95 US, and the professional version (which adds features such as CVS and Subversion integration) is \$295 US.

VIA's VT310-DP Mini-ITX



place and classroom.

It just wouldn't be New Products without something for the hardware geeks. VIA has released a reference design for 1U server clusters based on its VT310-DP Mini-ITX mainboard using dual Eden-N processors. Running at 1GHz and requiring no fan, you can pack 64 processors into a 16U chassis. With room for two 3.5 SATA drives on each board, you'll still be able to run the whole thing on less than a kilowatt. The intent is to move high-density server clusters out of the server farm and into the work-

Please send information about releases of Linux-related products to James Turner at newproducts@ssc.com or New Products c/o Linux Journal, PO Box 55549, Seattle, WA 98155-0549. Submissions are edited for length and content.

Centeris' Open Agent

Centeris is releasing an open-source technology that will allow Linux servers to be administered via the Microsoft Management Console. Likewise, Open Agent allows administrators who are more comfortable looking at system administration tasks through Microsoft-colored glasses to perform tasks such as setting up Samba shares and configuring Apache through the MMC. Centeris expects that as time progresses, more modules will be added by the community, allowing for more-specialized Linux tasks to be performed.

CrossOver Office 5.0

If you're still unable to cut the umbilical cord to some of your Windows applications, you may be in luck. CrossOver Office 5.0 is out, and it has added support for Office 2003. A personal version is available for \$39.95 US, and a professional edition is only \$30 more, with a free-trial option for the undecided.

VMware's VMware Player

Or, possibly you're looking for a more virtualized solution. In that case, VMware has just released the beta of a free VMware player, which will allow anyone to run a VMware virtual machine built using its products. Available for Linux and Windows, the player allows anyone to run a previously saved VMware environment without having to purchase a full VMware

OpenOffice.org 2.0

Of course, you could just do without those pesky Microsoft products altogether. OpenOffice.org 2.0 has finally hit the streets, with a new database module called Base (think Access). Version 2.0 also includes support for the OpenDocument format, improved PDF export support and better compatibility with proprietary formats. And, as always, it's free—something you definitely can't say about Microsoft Office. download.openoffice.org/2.0.0/index.html

O'REILLY®

ETech

EMERGING TECHNOLOGY CONFERENCE

Explore the applications and services, interface overhauls and algorithmic underpinnings, technological hacks and etiquette tweaks of what we call the Attention Economy.

At ETech you'll find these themes reflected in roll-up-your-sleeves tutorials, meaty and to-the-point plenary presentations, and real world focused breakout sessions.

Register today!

THE ATTENTION ECONOMY

MARCH 6-9, 2006

MANCHESTER GRAND HYATT
SAN DIEGO, CALIFORNIA

REGISTER TODAY AND SAVE 10% Use code: ET06LXJ

conferences.oreilly.com/etech

THE FUTURE IS EMERGING



Fat Man and Little Boy

A tale of two pretties, a powerhouse dual-Opteron system and a powerful notebook too.

BY JAMES TURNER

This month, we're pitting David vs. Goliath. In one corner, we have a massive 70 lb workstation with a pair of dual-core Opteron processors. In the other corner, sits a lithe little notebook with a single Pentium M. It may seem like apples versus oranges, but as the test results show, sometimes looks can be deceiving.

Monarch Computer's Empro Custom Workstation is certainly an impressive piece of hardware, if size is all that counts. Housed in a full-size tower with lots of bay space for drives, it has high-end workstation written all over it. Monarch's product rep insisted that I open it up once I got it, and one look inside is all you'll need to see why. The cable management inside the chassis is truly impressive, making the interior look nearly empty in spite of a huge PNY graphics board and four SATA drives.

The pre-installed copy of Fedora came up without a hitch, and easily configured itself for my network. Performance on the *Linux Journal* benchmark suite was fair for the single-CPU tests (postgres and ffmpeg x 1), but really shone on the multiprocessor ffmpeg x 2 and ffmpeg x 4 tests. In spite of the heavy I/O loading and memory requirements that the benchmarks place on the system, the performance was nearly flat, whether I was running one or four jobs. Having two dual-core CPUs available didn't hurt, but clearly the motherboard is doing a good job of keeping competing processes out of each other's hair as well.

This isn't to say that there aren't some negatives to deal with, however. First, this is not a miserly machine when it comes to the power socket. With all four processors churning at full steam, the power meter measured a fuse-blowing 390 Watts. And, of course, where there's power coming in, there's going to be heat coming out.

In addition, this isn't what you'd call a quiet system, certainly not one to stick downstairs as a Media PC. The noise readings at one foot ranged from 51dB at the front and sides to 60dB at the rear, and it was quite loud in an office setting. Things got worse when the front

door was open, bumping the level up to 56dB. There is a fan-speed control in the front, but it didn't seem to make a huge impact on overall noise levels.

Finally, there's the matter of the quirky case design. The USB and audio jacks, which are normally mounted on the upper front section of a workstation, have instead been placed on the bottom front of the right-side panel. Not only is this an inconvenient location to reach, but it's also begging to be kicked out by the first passerby.

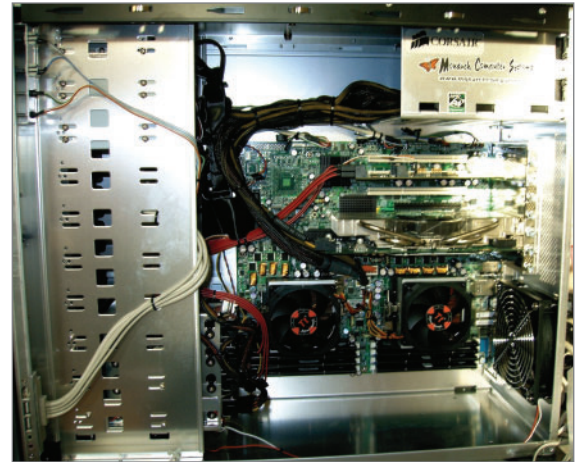


Figure 1. The obsessively neat interior of the Empro leaves enough empty space to park a Volkswagen.

Monarch Empro Custom Workstation

Processors: 2 x Opteron 280 Dual Core

Memory: 4GB PC-3200

Storage: 4 x Western Digital 10K 74GB SATA Raptor (RAID 10)

Graphics: PNY Quadro 4500 PCI-E 512MB Dual DVI

Motherboard: Tyan S2895A2NRF K8WE Dual Opteron, Dual PCI Express

Includes: Plextor Dual-Layer DVD Burner, ASUS CDRW Burner

List Price: \$8,964 US

Test Results:

- postgres: 636 secs
- ffmpeg x 1: 209 secs
- ffmpeg x 2: 214 secs
- ffmpeg x 4: 221 secs
- Power draw: 390 Watts
- Noise: 51dB front (56dB with door open) / 51dB sides / 60dB back

Summary:

Not a speed demon running a single process, but impressive under multiprocess load. The case is built like a tank, but it's noisy, power-thirsty and has odd placement for some external connectors.



Figure 2. This isn't exactly where I'd put the Monarch's USB ports if I were running the company.

I didn't get a chance to play with the impressive-looking PNY graphics board, complete with 512MB of RAM and PCI Express interface. I'm sure it would have chewed up *Quake* and spit it out the other end. However, you have to wonder about

the wisdom of putting a \$1,200 graphics board in a system whose real forte is multiprocess performance. If I were purchasing this system, it would probably be in a 4U rackmount configuration for use as a server, and I'd stick in the cheapest graphics card that would do the job.

On the other extreme, the Polywell PolyNote MXM915AS is positioning itself as a turnkey Linux notebook solution, offering a dual-boot Pentium M with SUSE on the Linux side.

Unfortunately, my out-of-the-box experience was not a happy one. My first criterion in looking at Linux notebooks is how well the hardware functions as shipped, and the PolyNote failed that test. From all appearances, someone had just stuck a stock copy of SUSE 9.3 on the laptop, and shipped it out. The wireless networking didn't function at all, and the CD drive was misconfigured so that

PolyNote MXM915AS

Processor: Centrino Pentium M 2.0GHz

Memory: 512MB DDR2 533MHz

Storage: 60G 5,400RPM SATA

Graphics: NVIDIA GeForce Go 6600 128M

Display: 1400x1050, 15.1"

Networking: Intel PRO/Wireless 2200BG, Intel Gigabit Ethernet

Includes: DVD/RW burner

List Price: \$1,350 US

Test Results:

- postgres: 357 secs
- ffmpeg x 1: 245 secs
- ffmpeg x 2: 483 secs
- ffmpeg x 4: 968 secs
- Battery life: 1:42
- Noise: 40dB (front and sides) / 45dB (rear)

Summary:

Surprisingly fast for such an inexpensive system, nice case design and good feel, but the pre-installed Linux needs work.

Low Price, Low Power, High Reliability for Embedded Systems



as low as

\$149 qty 1

\$119 qty 100



200 MHz CPU

Power as low as 1/2 Watt

- 3 boards, over 2000 configurations
- PC/104 expansion bus
- Fanless, no heat sink
- up to 256M onboard Flash
- up to 64M SDRAM
- 10/100 Ethernet
- up to 30 DIO ports
- 2 USB ports
- up to 5 COM ports
- Linux, NetBSD
- Real Time extension



Design your solution with one of our engineers

- Over 20 years in business
- Never discontinued a product
- Engineers on Tech Support
- Custom configurations and designs w/ excellent pricing and turn-around time
- Most products stocked and available for next day shipping



See our website for options, peripherals and x86 SBCs



We use our stuff.

visit our TS-7200 powered website at
www.embeddedARM.com

(480) 837-5200



Figure 3. The PolyNote: a generic notebook in Penguin clothing?

doing a mount /media/cdrom failed.

The system itself was well designed, with a nice solid feeling and no suggestion of flimsy plastic. It performed very well in benchmarks, actually outscoring the Monarch workstation in the postgres

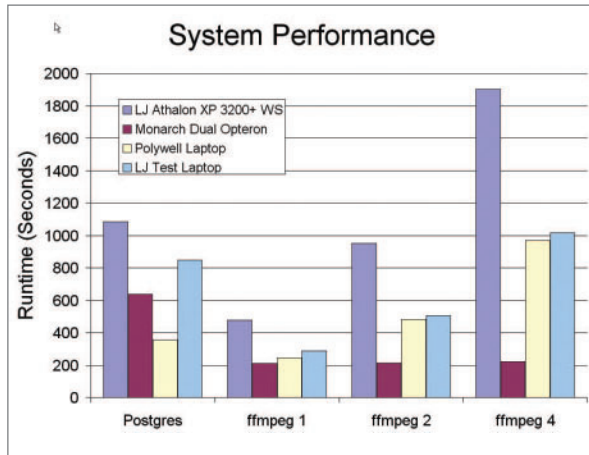


Figure 4. The Monarch Empro cruised through the multiprocessor tests, while the PolyNote did surprisingly well on the database test.

database test, and scoring close to it in the single-process ffmpeg transcode. Because the Pentium M is a single-core processor, no one should expect the kind of straight-line multiprocess performance that multiprocessor systems offer. However, the

PolyNote didn't bog down either, running two and four simultaneous processes in essentially two and four times the runtime, respectively.

The PolyNote just barely edged out the LJ test laptop (A Toshiba A75-S206 with a 2.8GHz P4) in overall performance, a system currently selling for around \$1,200 US on the street.

Power-wise, the PolyNote manages to make it through one hour and three-quarters on its battery alone


when running the benchmark suite in a constant loop. This compares with one hour on the LJ test laptop. To their credit, the PolyNote did handle suspend and restarting flawlessly—something that's frequently problematic on Linux laptops. It's not the quietest laptop you'll ever encounter, with dB measurements of around 40 from the front and sides, and a noisy 45 near the rear fan port.

The PolyNote weights in at about 5.5 pounds, but still manages to sport a 15.1" screen. The contrast and side visibility was generally good. The NVIDIA graphics processor ran *Tux Racer* speedily enough, with no tweaking of the X configuration required. The ALSA sound support also worked right from the start.

The Monarch workstation doesn't have the kind of "blow your sox off" single-process performance that would justify the price tag for most users, although in engineering development settings, the multiprocessor capabilities might be useful. It's also a bit too loud to want to use in most offices. The PolyNote is a nice enough laptop, but it's nothing special compared to other notebooks in the same price range. If Polywell wants to compete against vendors, such as EmperorLinux, who specialize in Linux laptops, they need to work on improving their installs so that everything works the first time. ■


James Turner is Product Review Editor for *Linux Journal*. He has written two books on Open Source Java development and is a Senior Software Engineer with Axis Technology, LLC.





Linux Certified

Linux Laptops: The New LC2000 Series

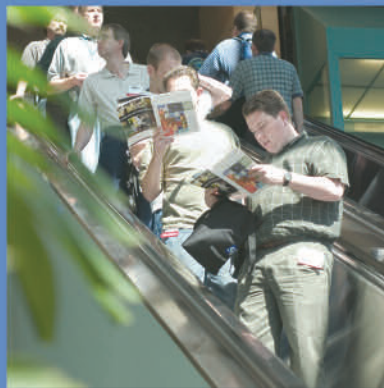


- High Performance
- Amazing ROI
- Robust
- Fully Compatible
- Cost Effective

Open Source Training, Services and Products 1-877-800-6873 www.linuxcertified.com



THE KNOWLEDGE YOU NEED



**EARLY
BIRD**
Register by
February 17th
and save up
to **\$400**

Over 200 classes and tutorials in:

Business of Software – *new track*

C++ – *including workshops from
Herb Sutter & Bjarne Stroustrup*

Java

.NET

Security

Web Services

XML

Modeling & Design

People, Process & Methods

Requirements & Analysis

Testing & Quality

Plus:

**Keynotes, Expo Floor,
BoFs, Parties &
Networking Events**

SD
WEST 2006
March 13 - 17
Santa Clara, CA

visit www.sdexpo.com for more information

Thin Clients Pay More

Thin clients make cost-effective diskless clients for education and the workplace. **BY STEPHEN SEFTON**

At the Superemos community education programme in Esteli, northern Nicaragua, we use secondhand computers as thin (diskless) clients in networks controlled by a Linux server. Many organizations already benefit from the Linux tools available for setting up such networks. They save money, simplify system administration, enhance security and increase autonomy. They are ideal for our low-budget education programmes in Nicaragua. Plenty of information and know-how on setting up these networks is published on the Web. Although that knowledge can be intimidating to try and make work, the effort certainly pays off.

This article explains how we have been using old hard drives and Flash drives to boot diskless clients over a network. It should be helpful to anybody on a tight budget who wants to offer a large group of people low-cost access to computing facilities. At our project, we are especially anxious that educators realise they can recycle older machines to deliver the latest software. But the principles apply equally to better-resourced outfits of all kinds including commercial businesses and government offices.

Before moving on to explain some of the basics for people with no experience of diskless client networks, I should detail and acknowledge our project's sources of tools and information. We have been using Novell's SUSE Linux 10.0 and Ubuntu's Breezy Badger distribution with a variety of new and old machines and parts supplied to us by the Rotary Club of Toronto-Leaside and by SSC Media Corporation, the publishers of *Linux Journal*. It's worth pointing out that our machines are all PCs. Optimally, it's worth trying to standardise as much as possible. That's always awkward to do when one depends on donated or inherited equipment.

Among the indispensable tools that make our project work smoothly are the superb diskless client software developed by the Linux Terminal Server Project and the comprehensive library of boot-ROM images (see the on-line Resources). The original tool we used for getting old hard disks to substitute for boot-ROM came from Andy Rabagliati (see Resources).

Diskless Client Network Basics

As in any network, diskless client systems consist of a server connected to clients, in our case by Ethernet cable. Once the server is powered up, the diskless clients receive their operating system from it. As each client is switched on, it learns from its BIOS that no system is available on hard disk. It then tries to boot from the local area network (LAN) by sending a

request via its network card for a server to give it an operating system. The server receives the request and looks to see whether it has the appropriate operating system to send out. If it does, the client boots up as normal using that operating system. For its users, the client machine works just as if it had its own operating system. In fact, it is receiving its operating system from the remote server.

It took us a while to understand the fundamental components of this concept and how they interact. The first thing to find out is whether a potential client has options in its BIOS allowing the machine to "Boot from LAN" (LAN stands for local area network) via boot-ROM. On some machines this is obvious, and on others, the settings are squirreled away in suboptions of the main BIOS. On still others it just does not exist. If it is available, it enables the machine to boot through a boot-ROM chip, usually with Pre-boot Execution Environment (PXE) capability located in the machine's network card.

If you find the Boot from LAN option and configure the BIOS to boot from LAN, you may well be lucky and everything will just work. But don't be dismayed if it does not. One of our machines with a VIA chipset swore solemnly that it would boot from LAN using PXE and persistently refused to do so before finally deciding one day that it would. Such frustrations are a trivial part of setting up a diskless client network and well worth overcoming in order to get a first-class network facility using whatever machines may be available.

Some machines have the network card integrated into their motherboard. If the network card is not integrated into the motherboard, it usually will be plugged in to the PCI slot. (It is possible to work with machines using older ISA cards, but they require special configuration so we have avoided using them.) If the network card is not integrated into the motherboard, it is unlikely to have pre-installed boot-ROM.

We found two main obstacles to using machines as diskless clients. One was that the potential client machine did not offer a Boot from LAN option. The second was that even if the machine offered to boot from LAN, the network card generally had no boot-ROM. We found we could readily overcome those obstacles by putting the necessary files to imitate a boot-ROM on an old hard drive or on a USB Flash drive. The core of this article is devoted to explaining how simple it is to do so. Doing this completely avoids using floppy disks, which, in Nicaragua, have simply become too unreliable.

The clients will work with just 32MB of RAM but seem happier with 64MB. Older machines with processor speeds of just 266MHz work okay, but processors with faster speeds obviously work better. Older mice, monitors and non-English keyboard layouts can be configured on the server if necessary. We found no configuration necessary for the majority of our hardware, thanks to the comprehensive LTSP software.

It is well worth investing resources in the server. We now use 1GB of RAM with a 2.4GHz processor and that provides really fast service for more than a dozen clients using Internet, office and game applications. It should be possible to run dozens of clients off one server if the server has adequate specifications. In this article, there's no space to say much about setting up the server for a thin-client network. A couple of excellent articles explaining how *Linux Journal* helped us do this have already been written by Kevin Brown (see Resources).

Assembling the Necessary Files

For our project, we use the LILO (Linux Loader) bootloader because all we are interested in is booting Linux. The version of LILO to use is important. We discovered that the latest version of LILO by default insists on using lba32 to manage disk geometry and that gave us problems configuring our Flash drive.

Fortunately, we found that older versions of LILO don't impose that option. We copied the LILO we used for our Flash drives from Andy Rabagliati's wizzy package. For convenience, we also copied the boot.b file from Andy's package. To configure old hard drives, we relied on Ubuntu and SUSE's prebundled LILO packages. (See the Configuring Old Hard Drives section for a brief mention on using the GRUB bootloader.)

We also need the relevant boot-ROM images for the different network cards that our boot-ROM disk may end up working with. Our network cards are either 3Com 905, Realtek 8139 or Via-Rhine. We got these image files from ROM-o-matic. It took a certain amount of trial and error to get the images that worked. ROM-o-matic updates its releases regularly. All the releases have similar options, and ROM-o-matic helps keep the trial-and-error process to a minimum by offering a button that gives a list of the cards that particular images will work for.

Once you have selected the card image that is probably the right one, you need to select the image type. Because we are using LILO, we chose the current zlilo image type as well as the older type called lzlilo. We used lzlilo on our Flash drive because we found the newer zlilo images seemed to work only on the hard drives. We have yet to figure out why. It's worth emphasizing that experimentation yields invaluable information. Here I summarise only the results of our own experimentation. Other people will certainly get different and very likely better results as they try out their own equipment.

ROM-o-matic offers a Get Rom button to download the image files it produces. That option leads to a prompt allowing you to save the image files in your local directory system. We downloaded the various .lzlilo and .zlilo images we thought we would need for our three types of network card. With those files and the LILO files, we had all we needed to make our boot-ROM disks using either a Flash drive or an old hard disk. So we copied them all to one place in a directory we decided to call /flashlilo. Then we needed to put them onto our boot-ROM disk.

Working with Flash Drives

For newer machines that don't boot from LAN but that do have a BIOS option allowing boot from USB hard drive, a Flash drive that imitates boot-ROM is a handier alternative than an old hard drive. Once one client is booted, the Flash drive can be removed and used to boot another client. We found that we could treat the USB Flash drive as if it were a SCSI hard drive. Plugging the Flash drive in to a convenient USB socket we powered up, although thanks to hotplug, we could have just plugged the USB drive in while the machine was already running. This machine ran SUSE, so the YaST configuration tool recognised the new device and asked whether to configure it. We said no.

To ensure the drive we wanted to use for our boot-ROM was clear of any formatting or partitioning problems, we deleted the existing partitions, as the root superuser, using fdisk,

and installed a new bootable partition. (If there is anything on those disks that you need, make copies. Partitioning afresh destroys everything on the disk!) We then used fdisk to partition the Flash drive by typing:

```
# fdisk /dev/sda
```

Something very important to note for later use when configuring the bootloader is the number of heads, sectors and cylinders fdisk discovers on the drive. Make a note of these. After using fdisk to set up a bootable partition free of any potentially bothersome history, we were ready to put a filesystem on it. We took the easy option and typed:

```
# mke2fs /dev/sda1
```

to put an ext2 filesystem on the Flash drive. In order to tell Linux what device to mount and where, we used the traditional directory /mnt as our mountpoint, checking first that it was empty. For the Flash drive, we typed:

```
# mount /dev/sda1 /mnt
```

Then we copied all the files in /flashlilo to /mnt:

```
# cp /flashlilo/* /mnt
```

At this point, we needed to write a configuration file for LILO. Being frightened of vi and emacs, we used pico:

```
boot = /dev/sda
disk = /dev/sda
    bios = 0x80
    sectors = 62
    heads = 4
    cylinders = 1015
install = /mnt/boot.b
map = /mnt/map
root = /dev/sda1
vga = normal
read-only
delay = 30
pROMpt
image = /mnt/viarhine6102.lzlilo
    label=viarhine2
    read-only
image = /mnt/3c905b.lzlilo
    label=3Com905b
    read-only
image = /mnt/rt8139.lzlilo
    label=RTL8139
    read-only
```

We saved that file to /mnt with the standard name lilo.conf.

The main part of that configuration file applies to the overall boot process before the user selects an option from the final menu. In this part, the first line tells the client machine to boot from the Flash drive. The second line and its sub-lines tell it about the disk geometry so it knows where to put things. (This is where you need that disk information you noted from fdisk!)

The install line tells it to use boot.b to install the boot stages.

The map line tells the thin client where to find stuff in a file created by the LILO program. The root line tells it where to find the filesystem. The vga line explains how to output information to the monitor. The read-only option keeps everything from getting interfered with. The delay line tells the machine how long to wait before displaying the prompt.

The image sections of the configuration file offer the client user different boot options. So when the client starts up, LILO offers these image options to the user via a menu with three choices of network card from which to boot: Via-RhineII, Realtek 8139 or 3Com 905.

Why are boot.b, map and LILO images referred to as residing in the /mnt directory? The reason is now that we have written the configuration file for LILO, we have to tell LILO to use that configuration file. We can do that only in the directory where we are working, which is where the drive we are working on is mounted. In our case, this is /mnt. The command to tell the LILO version we want to work with to update the lilo.conf file is therefore:

```
# /mnt/lilo -C lilo.conf
```

This is fine for the moment because we are in /mnt. But what happens when we unmount the Flash drive, remove it and insert in the machine we hope it will boot? Won't we need to change references in the lilo.conf file? And won't we then get into a manic circle of eternally having to run LILO to reconfigure lilo.conf? No, as it turns out. When we tried booting this in our client machines, it all ran smoothly. So, now that we have our boot-ROM drive, we move out of the /mnt directory and type:

```
# umount /dev/sda1
```

before removing the drive from the machine.

Configuring Old Hard Drives

When we first started configuring old hard disks to imitate boot-ROMs, we followed a procedure similar to the one we used for the Flash drive. We had to learn this might mean changing the jumper on the hard disk before connecting it to the machine. (The term jumper refers to a tiny connector located in a set of pins on the side of the hard drive.) It is very easy to change the jumper. Most hard drives have little diagrams on their casing explaining how to set the jumper for the different options. Usually there are three: Master, Slave and Chain Select. Chain Select allows the machine's BIOS to decide what status to assign the hard drive.

Most commonly used PC machines have just one IDE hard drive, which is usually the master drive. Linux identifies that drive as /dev/hda. If the old machine has enough memory and a fast enough CPU, it makes sense to connect a CD-ROM drive with its jumper set to Chain Select or Slave and configure the machine's hard drive directly. However, to save time on installation, we ended up deciding to use a much faster new machine with a newish CD-ROM to configure our old hard drives. That meant simply disconnecting the existing hard disk on the faster machine and connecting our old hard drive with the jumpers set to Master. So once we finish the configuration and we connect it to the motherboard of the diskless client, it will be

recognised straightaway as the primary master drive.

Whatever method we use to prepare the old hard drive for configuration, we install either Ubuntu's Breezy Badger directly from the CD or SUSE 10 over our local network. In either case, we do a minimum text-based installation and choose LILO as our bootloader. Once the installation is done, we can edit the /etc/lilo.conf configuration file to add the images exactly as they appear in the lilo.conf file described above for the Flash drive. Finally, we run the line:

```
# lilo -C /etc/lilo.conf
```

to update LILO with our new image options.

In relation to the GRUB bootloader, we had an old slow machine already running Linux that we wanted to turn into a client. We found we could get it to boot off a boot-ROM image simply by adding this entry to GRUB's menu.lst file (substituting the name of the appropriate boot-ROM image file):

```
title Via-Rhine Boot-ROM
    root (hd0,0)
    kernel /boot/via-rhine.zlilo
```

Before we rebooted, we set the hard drive as the first boot option. In this case, the machine on which we did the configuration has the same kind of network card as a couple of the older machines we want to use as diskless clients, so it was a handy control to make sure the hard drive is doing its job imitating boot-ROM. Once satisfied that everything works, we can transfer the drive we just configured to the machine it will be booting. Alternatively, if we configure an old machine by connecting a CD-ROM drive to it, when done we just disconnect the CD-ROM drive and reboot. We have found this simple configuration procedure also works well on old laptops, so long as they have a built-in CD-ROM and network card.

Configuring the Server for Its Clients

Our system in Esteli now serves nine clients, some that have boot-ROM PXE and the rest with various drives imitating boot-ROM. In the case of the Flash drive we configured to imitate boot-ROM, we just put it in a USB drive on the client machine before starting up. The BIOS needs to be set correctly for each client so that the first boot device corresponds to the relevant disk on that client. To access the BIOS, search the monitor screen when you start the computer.

There should be a line that says something like "Press DEL to enter Set-up". Whatever the key or combination of keys indicated, this will enable access to the machine's BIOS menus. The configuration of the order of boot devices is usually in the second menu, which should be called something like "Advanced BIOS options". In that menu, you should find the option to set the first boot device. In the case of the Flash drive, we found that setting the first boot device to USBHDD works fine.

The clients need to get a kernel image from the server. A kernel image is another way of describing the files that make up the operating system software that makes it possible to work with different applications. To get their kernel image, the clients have to provide the network server with their identities and establish a network address. Each client identifies itself by offering the unique code on its network card called the MAC

address. In response, the server provides each client with a network address called the IP (Internet Protocol) address.

The clients get this information from the server via a networking standard called the Dynamic Host Configuration Protocol (DHCP). Clients booting with PXE get assigned a dynamic address automatically from a general section of the DHCP configuration file `dhcpd.conf`. Clients without PXE get a fixed IP address through a specific configuration in `dhcpd.conf` to ensure they are recognised and can get an appropriate Linux kernel image. We have found older machines like to work with kernel images called `vmlinuz` rather than `bzImage`.

In our configuration for each non-PXE client, we told the server's `/etc/dhcpd.conf` file their Mac address, a fixed IP address and the kernel image they would like. It may sometimes be necessary to configure the Linux Terminal Server Project (LTSP) package that provides the filesystem architecture for our network. The LTSP `lts.conf` configuration file can be adjusted when a client is using a mouse or monitor or keyboard layout not recognised automatically by LTSP.

Here is some of our server's `dhcpd.conf` file:

```
ddns-update-style ad-hoc;
allow booting;
allow bootp;
subnet 198.186.207.0 netmask 255.255.255.0 {
range dynamic-bootp 198.186.207.205 198.186.207.220;
default-lease-time 21600;
max-lease-time 43200;
}
next-server 198.186.207.124;
filename "pxelinux.0";
option root-path "198.186.207.124:/opt/lts/i386";

host ws001 {
hardware ethernet 00:11:5B:86:46:B5;
fixed-address 198.186.207.201;
filename "/lts/vmlinuz-2.6.9-ltsp-3";
}

host ws002 {
hardware ethernet 00:60:08:C6:2B:43;
fixed-address 198.186.207.202;
filename "/lts/vmlinuz-2.6.9-ltsp-3";
}
```

And here is the main part of our server's LTSP `lts.conf` file:

```
[Default]
SERVER = 198.186.207.124
XSERVER = auto
X_MOUSE_PROTOCOL = "PS/2"
X_MOUSE_DEVICE = "/dev/psaux"
X_MOUSE_RESOLUTION = 400
X_MOUSE_BUTTONS = 3
XkbLayout = es
USE_XFS = N
SCREEN_01 = startx
```

LTSP's package permits a wide range of sophisticated configuration options for multimedia and other applications run off

individual client workstations that can be configured in `lts.conf` similarly to individual workstations in `dhcpd.conf`. We keep our system simple because we are working mainly with Internet and office applications. In our case, LTSP's default configuration recognised all our various clients' hardware except the keyboard layout, so we added a line configuring a Spanish keyboard.

This technology is incredibly accessible and flexible. For us in Nicaragua, it enables us to offer computer skills training to a large number of students on low incomes using old equipment to deliver the latest software. The widely available tools and information mean that even relative beginners in Linux can readily implement such diskless client systems for a broad range of educational, commercial and administrative purposes. Rarely can a bottom line show such a large beneficial return for so modest an investment of resources.

Resources for this article: www.linuxjournal.com/article/8699.

Stephen Sefton is an Irish citizen who has worked for nearly 20 years as a community development worker in Central America on a broad range of community concerns, including housing, human rights, health care, education and sustainable agriculture. On visits home to Wexford, he watches the progressive embayment of Rosslare Strand with his uncle Christy and wonders how long before the family home slides down the cliff into the sea. He remembers happy walks around the southern shore of Lake Nicaragua and the beach at Curraghcloe.

Advertise on LinuxJournal.com

Where professionals go to find out what's hot in Linux



For a decade, LinuxJournal.com has enabled Linux enthusiasts to make smart purchasing decisions with its award-winning editorial.

Showcase your company as a market leader to these influential professionals by placing a banner advertisement on the site.

Over **1,250,00** page views every month

LINUX
JOURNAL

www.linuxjournal.com/advertising

For further information:

Phone 206-782-7733 ext.2 or Email ads@linuxjournal.com

Heterogeneous Processing: a Strategy for Augmenting Moore's Law

One way to break the high-performance computing barrier imposed by the limitations of Moore's Law.

BY AMAR SHAN

Better application performance: everyone wants it, and in the high-performance computing (HPC) community, we've come to expect it. Maybe we've even gotten a little spoiled. After all, we've enjoyed basically continuous performance improvement for four decades, thanks to Moore's Law.

Now in its 40th year, that principle (which predicts a doubling of transistor density every 18 months) is still going strong. But unfortunately, ever-increasing transistor density no longer delivers comparable improvements in application performance. The reasons for this are well known. Adding transistors also adds wire delays and speed-to-memory issues. More aggressive single-core designs also inevitably lead to greater complexity and heat. Finally, scalar processors themselves have a fundamental limitation: a design based on serial execution, which makes it extremely difficult to extract more instruction-level parallelism (ILP) from application codes.

These issues are no longer the sole concern of a small, high-end user base, if they ever were. It is becoming more apparent that major performance improvements could have a profound effect on virtually every scientific field. The President's Information Technology Advisory Committee, which challenged HPC researchers to achieve a sustained petaflop on real applications by 2010, noted that trans-petaflop systems will be crucial for better weather and climate forecasting, manufacturing, pharmaceutical development and other strategic applications. Industry experts at conferences such as Petaflops II are demanding improvements for a laundry list of applications, including crash testing, advanced aircraft and spacecraft design, economic modeling, and combating pandemics and bio-terrorism.

The HPC community is responding by developing new strategies to augment Moore's Law and exploring innovative HPC architectures that can work around the limitations of conventional systems. These strategies include:

- Multicore systems that use two or more cores on a die to continue providing steady performance gains.
- Specialized processors that deliver enhanced performance in areas where conventional commodity processors fare poorly.
- Heterogeneous computing architectures, in which conventional and specialized processors work cooperatively.

Each of these strategies can potentially deliver substantial performance improvements. At Cray, we are exploring all three. But in the long term, we believe heterogeneous computing holds tremendous potential for accelerating applications beyond what one would expect from Moore's Law, while overcoming many of the barriers that can limit conventional architectures. As a participant in the DARPA High Productivity Computing Systems Program, we expect heterogeneous processing to become crucially important over the next several years.

An Immediate Solution: Multicore Processing

Placing multiple cores on a die is the fastest way to deliver continuous performance gains in line with Moore's Law. A well-known example of a multiple-core processor is the dual-core AMD Opteron.

Cray and other HPC manufacturers have already embraced this model. Today, Cray is delivering dual-core systems, with expectations to leverage more cores in the future. This strategy offers immediate doubling of computing density, while reducing per-processor power consumption and heat.

For many applications (especially those requiring heavy floating-point operations), multicore processing will provide performance gains for the foreseeable future, and the model will likely serve as the primary vehicle through which Moore's Law is upheld. However, for some applications (notably, those that depend on heavy bit manipulation, sorting and signal processing, such as database searching, audio/video/image processing and encryption/decryption), Moore's Law may not be enough. Major advances in these applications can be realized only with processing speeds orders of magnitude beyond what is available today (or likely to be available anytime soon) through conventional processors. So HPC researchers are exploring alternative models.

Innovative Processing Elements

In recent years, architectures based on clusters of commodity processors have overtaken high-end, specialized systems in the HPC community, due to their low cost and solid performance for many applications. But, as some users begin to bump up against the inherent limitations of scalar processing, we are beginning to see a reversal in that trend. Examples of this resurgence include:

The Heterogeneous Model

Heterogeneous computing is the strategy of deploying multiple types of processing elements within a single workflow, and allowing each to perform the tasks to which it is best suited. This model can employ the specialized processors described above (and others) to accelerate some operations up to 100 times faster than what scalar processors can achieve, while expanding the applicability of conventional microprocessor architectures. Because many HPC applications include both code that could benefit from acceleration and code that is better suited for conventional processing, no one type of processor is best for all computations. Heterogeneous processing allows for the right processor type for each operation within a given application.

Traditionally, there have been two primary barriers to widespread adoption of heterogeneous architectures: the programming complexity required to distribute workloads across multiple processors and the additional effort required if those processors are of different types. These issues can be substantial, and any potential advantages of a heterogeneous approach must be weighed against the cost and resources required to overcome them. But today, the rise of multicore systems is already creating a technology discontinuity that will affect the way programmers view HPC software, and open the door to new programming strategies and environments. As software designers become more comfortable programming for multiple processors, they are likely to be more willing to consider other types of architectures, including heterogeneous systems. And several new heterogeneous systems are now emerging.

The Cray X1E supercomputer, for example, incorporates both vector processing and scalar processing, and a specialized compiler that automatically distributes the workload between processors. In the new Cell processor architecture (designed by IBM, Sony and Toshiba to accelerate gaming applications on the new PlayStation 3), a conventional processor off-loads computationally intensive tasks to synergistic processing elements with direct access to memory. But one of the most exciting areas of heterogeneous computing emerging today employs field programmable gate arrays, or FPGAs.

The FPGA Coprocessor Model

FPGAs are hardware-reconfigurable devices that can be redesigned repeatedly by programmers to solve specific types of problems more efficiently. FPGAs have been used as programmable logic devices for more than a decade, but are now attracting stronger interest as reconfigurable coprocessors. Several pioneering conferences on FPGAs have been held recently in the United States and abroad, and the Ohio Supercomputer Center recently formed the OpenFPGA (www.openfpga.org) initiative to accelerate adoption of FPGAs in HPC and enterprise environments.

There's a reason for this enthusiasm: FPGAs can deliver orders of magnitude improvements over conventional processors on some types of applications. FPGAs allow designers to create a custom instruction set for a given application, and apply hundreds or even thousands of processing elements to an operation simultaneously. For applications that require heavy bit manipulation, adding, multiplication, comparison, convolution or transformation, FPGAs can execute these instructions on thousands of pieces of data at once, with low control overhead

and lower power consumption than conventional processors.

FPGAs have had their own historic barriers to widespread adoption. First, they traditionally have been integrated into conventional systems via the PCI bus, which limits their effectiveness like the specialized processors described above. More critically, adapting software to interoperate with FPGAs has been extremely difficult, because FPGAs must be programmed using a Hardware Design Language (HDL). Although these languages are commonplace for electronics designers, they are completely foreign to most HPC system designers, software programmers and users. Today, the tools that will allow software designers to program in familiar ways for FPGAs are just beginning to emerge. Users are also awaiting tools to port existing scalar codes to heterogeneous FPGA coprocessor systems. However, Cray and others are working to eliminate these issues.

The Cray XD1, for example (one of the first commercial HPC systems to use FPGAs as user-programmable accelerators), eliminates many performance limitations by incorporating the FPGA directly into the interconnect and tightly integrating FPGAs into the system's HPC Optimized Linux operating system. New tools also allow users to program for FPGA coprocessor systems with higher-level C-type languages. These include the Celoxica DK Design Suite (a C-to-FPGA compiler that is being integrated with the Cray XD1), Impulse C, Mitrion C and Simulink-to-FPGA from Matlab, which offers a model-based design approach.

Ultimately, as heterogeneous systems incorporating FPGAs become more widely used, we believe they will allow users to solve certain types of problems much faster than anything that will be provided in the near future through Moore's Law, and even support some applications that would not have been possible before. (For an example of the potential of FPGA coprocessor systems, see the sidebar on the Smith-Waterman bioinformatics application.)

Looking Ahead

Although many exciting avenues of exploration are underway today in the field of heterogeneous computing, we are not yet at the point where this model will take over as the dominant HPC system architecture. The barriers that remain (primarily, difficulties programming for and porting existing code to heterogeneous systems) are significant. However, Cray and others in the HPC community are already making strides in these areas.

As with any new technology, widespread adoption of heterogeneous systems will depend on an analysis of the gains that can be achieved versus the effort required to realize them. In the long term, we believe that the performance advantages offered by heterogeneous architectures for some applications will be too compelling to ignore. ■

Amar Shan is a senior product manager at global supercomputer leader Cray Inc. Shan joined Cray in 2004 when Cray acquired OctigaBay Systems Corporation, and he is responsible for setting product direction for Cray next-generation products and the Cray XD1 high-performance computing (HPC) system—the only Linux/Opteron system designed specifically for HPC applications. Shan holds a Master of Applied Science in Artificial Intelligence from the University of Waterloo and Bachelor of Applied Science in Electrical Engineering and Computer Science from the University of British Columbia.

Smith-Waterman: an Example of the FPGA Coprocessor Approach to Heterogeneous Computing

New techniques in genomics can provide millions of pieces of DNA from a few tests, but transforming the mountains of raw data into meaningful results can be a long, grueling process. Genes are usually represented as ordered sequences of nucleotides. (Similarly, protein sequences are strings of amino acids.) Investigators can infer a great deal about genes and proteins from their sequence alone, and answer questions such as the similarity of genes in different species by comparing sample sequences to ones already classified. However, to do this, accurate methods to determine the similarity between two sequences are critical.

Smith-Waterman is the most powerful algorithm available for accomplishing this (Temple F. Smith and Michael S. Waterman, "Identification of Common Molecular Subsequences", *J. Mol. Biol.*, 147:195–197, 1981). But the mathematical operations involved are difficult for commodity processors, and conventional systems deliver extremely poor performance. By attacking the problem with the Cray XD1—a heterogeneous system combining scalar processing with FPGA coprocessors—investigators can accelerate Smith-Waterman and get results up to 40 times faster than with conventional systems.

Characteristics of Smith-Waterman

The Smith-Waterman algorithm compares sample DNA or proteins against existing databases. Because both sample and database may have errors in the form of missing or added characters—and because a variation of a few characters can signify major biological differences—a highly accurate matching process is required.

Gene sequences contain four letters (G, C, A and T) for the four nucleotides, and protein sequences contain 20 amino acid characters. Because sequences are ordered strings, accurate comparisons must determine whether two strings align, as well as the letters they share. (For instance, in plain English, STOP and POTS share the same letters but cannot satisfactorily be aligned, while POTS and POINTS can, if a gap is created between the O and T in POTS.) Smith-Waterman uses "dynamic programming" to find the optimal alignment. This requires massive amounts of simple parallel computation, as well as heavy bit manipulation, and commodity scalar processors are extremely inefficient at these operations.

A conventional processor running Smith-Waterman requires thousands of unique steps to compare each piece of data. The number of instructions devoted to performing actual comparisons is a fraction of those devoted to determining the next comparison point and the surrounding logic. In fact, a scalar processor may devote only one instruction in 100 to comparisons—an efficiency rate of only 1%.

An HPC system using an FPGA coprocessor can provide several advantages that accelerate this algorithm. First, unlike general-

purpose processors designed to support many different types of codes, FPGAs allow for a custom instruction set that closely mirrors the application. FPGAs also offer huge amounts of inherent parallelism, and they can be programmed to build thousands of compare units side by side and perform thousands of comparisons every clock tick. In addition, hardware computation is inherently more efficient than software at bit manipulation.

The Cray XD1 Approach

To understand fully how the Cray XD1 accelerates Smith-Waterman, it is necessary to understand the system's unique FPGA coprocessor architecture (Figure 1), as well as how the application itself functions.

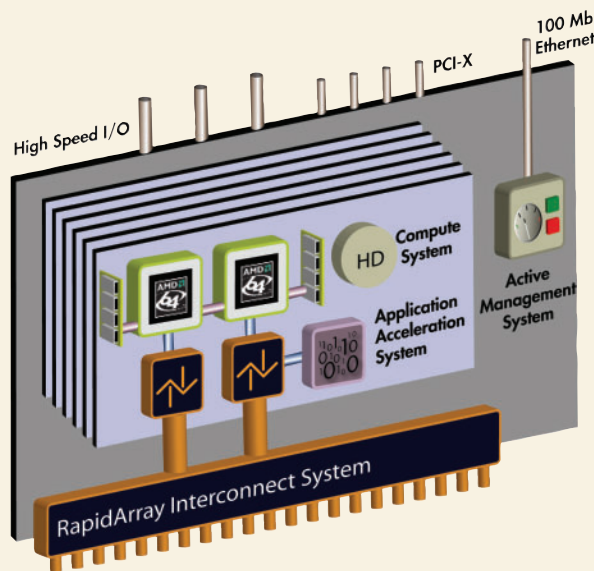


Figure 1. Cray XD1 System Architecture

Scoring Matrix

	0	A	C	G	T	A	T	G	C
0	0	0	0	0	0	0	0	0	0
A	0	2	0	0	0	2	0	0	0
C	0	0	4	2	1	0	1	0	2
G	0	0	2	6	4	3	2	3	1
A	0	2	1	4	5	6	4	3	2
A	0	2	1	3	3	7	5	4	3
C	0	2	4	2	2	5	6	4	6
C	0	0	2	3	1	4	4	5	6
C	0	0	2	1	2	3	3	3	7
T	0	0	0	1	3	2	5	3	5
T	0	0	0	0	3	2	4	4	4
G	0	0	0	2	1	2	2	6	4
C	0	0	2	0	1	0	1	4	8

Figure 2. Scoring Matrix

Smith-Waterman formulates matches by first creating a scoring matrix and calculating each cell according to the value of cells above and to the left. Once this matrix is created, the algorithm calculates a maximum score, traces back along the path that led to the score and delivers a final alignment (Figures 2 and 3).

To accelerate this operation, the Cray XD1 partitions the algorithm between the system's FPGA and Opteron processors. The system uses the FPGA for filling the scoring matrix (which involves parallel computation) and sends back traceback information to the Opterons to regenerate the matrix (a serial opera-

Final Alignment

A	C	G	A	A	C	C	C	T	T	G	C
A	C	G	T	A	-	-	-	-	T	G	C

Figure 3. Smith-Waterman Formulation

tion). Effectively, the system's HPC Optimized Linux operating system calls the FPGA solely for the kernel of the Smith-Waterman application. But the massive amount of parallelism available with the FPGA coprocessor delivers results 25 to 40 times faster than conventional HPC architectures.

Below is an example of the system interacting with the FPGAs:

```
/* Tilt the arrays by copying them to the FPGA. */
static void tilt (int fp_id, u_64 *trans_matrix, int row_len)
{
    int i = 0;
    u_64 status = 0;

    /* Initialize the FPGA to accept a new stream of arrays. */
    fpga_wrt_appif_val (fp_id, TILT_START, TILT_APP_CFG, TYPE_VAL, &e);

    /* Copy the matrix to the FPGA. */
    memcpy((char *) fpga_ptr, (char *) trans_matrix,
    row_len*sizeof(u_64));

    /* Poll to see if the FPGA has completed tilting the arrays. */
    while (1) {
        fpga_rd_appif_val (fp_id, &status, TILT_APP_STAT, &e);
        if (status & TILT_DONE) break;
    }

    /* When the FPGA has finished, all the transposed data will have */
    /* been written by the FPGA to the transfer region of DRAM.      */
    /* Copy the data from the transfer region back to the array.      */
    // for(i=0;i<row_len;i++) {
    //   trans_matrix[i] = dram_ptr[i];
    // }

    return;
}
```

Advantages of the Cray XD1 Heterogeneous Architecture

With the application acceleration afforded by the Cray XD1, users can achieve much more timely results using the best algorithm available—instead of settling for tools that deliver less-accurate solutions more quickly. And, because the system uses the FPGA coprocessor solely for the kernel of Smith-Waterman, it can be updated easily as the rest of the application evolves. In addition, unlike dedicated hardware solutions that have been available to investigators, the Cray XD1 is a

true, general-purpose HPC system. It is not limited to running a single code, and it can be applied to other bioinformatics applications just as easily as Smith-Waterman. In short, the Cray XD1 provides an effective, affordable and investment-protected solution for delivering unprecedented performance on critical life science applications.

—AMAR SHAN

Why is LPI the Global Standard in Linux Certification?

Trusted.

All Linux Professional Institute certification programs are created using extensive community input, combined with rigorous psychometric scrutiny and professional delivery. We test the whole continuum of important Linux skills - we don't just focus on small, subjective tasks. LPI exams are not simply an afterthought used to help sell something else. LPI is a non-profit group that does not sell software, training or books. Our programs and policies are designed to meet educational requirements, not marketing.

Accessible.

LPI exams are available in seven languages, at more than 7,000 locations, in more than 100 countries. You take LPI exams when you want, where you want. In addition, special exam lab events around the world make our program even more affordable. And because we don't make exclusive partnerships, LPI is supported by a broad range of testing centers, book publishers and innovative suppliers of preparation materials.

Independent.

You switched to Linux to get away from single-vendor dependence. So why trade one form of vendor lock-in for another? LPI's program follows the LSB specification, so people who pass our tests can work on all major distributions. Because of its strong grass-roots base and corporate support both inside and outside the world of open source, LPI goes beyond "vendor-neutral" to truly address community needs.

LPI is IT certification done RIGHT!

For more information, please contact us at Info@lpi.org or visit us at www.lpi.org.



Embedding Python in Your C Programs

C, meet Python. Python, this is C. With surprisingly little effort, the Python interpreter can be integrated into your program to add features quickly that could take months if written entirely in C. **BY WILLIAM NAGEL**

The language of choice for large, high-performance applications in Linux is almost always C, or somewhat less often C++. Both are powerful languages that allow you to create high-performance natively compiled programs. However, they are not languages that lend themselves to runtime flexibility. Once a C/C++ application is compiled, its code is pretty much static. At times, that can be a real hindrance. For example, if you want to allow users of a program to create plugins easily that extend the application's functionality, you have to deal with complex dynamic linking issues that can cause no end of headaches. Additionally, your users will have to know C/C++ in order to extend the application, which severely limits the number of people capable of writing extensions.

A much better solution is to provide your users with a scripting language they can use to extend your application. With a scripting language, you will tend to have much more runtime flexibility, as well as shorter development times and a lower learning curve that will extend the base of users capable of creating extensions.

Unfortunately, creating a scripting language is very much a nontrivial task that easily could become a major portion of your program. Fortunately, you don't need to create a scripting language. With Python, you can embed the interpreter directly into your application and expose the full power and flexibility of Python without adding very much code at all to your application.

Including Python in an Application

Including the Python interpreter in your program is extremely simple. Python provides a single header file for including all of the definitions you need when embedding the interpreter into your application, aptly named `Python.h`. This contains a lot of stuff, including several of the standard headers. For compiling efficiency, it might be nice if you could include only those parts of the interface that you actually intend to use, but unfortunately Python doesn't really give you that option. If you take a look at the `Python.h` file, you'll see that it defines several important macros and includes a number of common headers that are required by the individual components included later in the file.

To link your application to the Python interpreter at compile time, you should run the `python-config` program to get a list of the linking options that should be passed to the compiler. On my system, those are:

```
-lpython2.3 -lm -L/usr/lib/python2.3/config
```

A Very Simple Embedded App

So, how much code does it take to run the Python interpreter from a C app? As it turns out, very little. In fact, if you look at Listing 1, you'll see that it can be done in as little as three lines

of code, which initialize the interpreter, send it a string of Python code to execute and then shut the interpreter back down.

Listing 1. Embedding Python in Three Lines

```
void exec_pycode(const char* code)
{
    Py_Initialize();
    PyRun_SimpleString(code);
    Py_Finalize();
}
```

Or, you could embed an interactive Python terminal in your program by calling `Py_Main()` instead, as in Listing 2. This brings up the interpreter just as if you'd run Python directly from the command line. Control is returned to your application after the user exits from the interpreter shell.

Listing 2. Embedding an Interactive Python

```
void exec_interactive_interpreter(int argc, char** argv)
{
    Py_Initialize();
    Py_Main(argc, argv);
    Py_Finalize();
}
```

The Python Environment

Embedding the interpreter in three lines of code is easy enough, but let's face it, just executing arbitrary strings of Python code inside a program is neither interesting nor all that useful. Fortunately, it's also far from the extent of what Python allows. Before I get too deep into what it can do though, let's take a look at initializing the environment that Python executes within.

When you run the Python interpreter, the main environment context is stored in the `__main__` module's namespace dictionary. All functions, classes and variables that are defined globally can be found in this dictionary. When running Python interactively or on a script file, you rarely need to care about this global namespace. However, when running the embedded interpreter, you'll often need to access this dictionary to get references to functions or classes in order to call or construct them. You also may find that you occasionally want to copy the global dictionary so that different bits of code can be run in distinct environments. For instance, you might want to create a new environment for each plugin that you load.

To get at the `__main__` module's dictionary, you first need to get a reference to the module. You can do this by calling the `PyImport_AddModule()` function, which looks up the module name you supply and returns a `PyObject` pointer to that object. Why a `PyObject`? All Python data types derive from `PyObject`, which makes it a handy lowest-common denominator. Therefore, almost all of the functions that you'll deal with when interacting with the Python interpreter will take or return pointers to `PyObject`s rather than another more specific Python data type.

Once you have the `__main__` module referenced by a `PyObject`, you can use the `PyModule_GetDict()` function to get a reference to the main module's dictionary, which again is returned as a `PyObject` pointer. You can then pass the dictionary reference when you execute other Python commands. For example, Listing 3 shows how you could duplicate the global environment and execute two different Python files in separate environments.

I'll get into the details of how `PyRun_File()` works in a little bit, but if you look carefully at Listing 3, you should notice something interesting. When I call `PyRun_File()` to execute the files, the dictionary gets passed in twice. The reason for this is that Python code actually has two environmental contexts when it is executed. The first is the global context, which I've already talked about. The second context is the local context, which contains any locally defined variables or functions. In this case, those are the same, because the code being executed is top-level code. On the other hand, if you were to execute a function dynamically using multiple C-level calls, you might want to create a local context and use that instead of the global dictionary. For the most part though, it's generally safe to pass the global environment for both the global and local parameters.

Manipulating Python Data Structures in C/C++

At this point, I'm sure you've noticed the `Py_DECREF()` calls that popped up in the Listing 3 example. Those fun little guys are there for memory management purposes. Inside the interpreter, Python handles memory management automatically by keeping track of all references to memory transparent to the programmer. As soon as it determines that all references to a given chunk of memory have been released, it deallocates the no-longer needed chunk. This can be a problem when you start working on the C side though. Because C is not a memory-managed language, as soon as a Python data structure ends up referenced from C, all ability to track the references automatically is lost to Python. The C application can make as many copies of the reference that it wants, and hold on to it indefinitely without Python knowing anything about it.

The solution is to have C code that gets a reference to a Python object handle all of the reference counting manu-

Listing 3. Making a Copy of the Environment

```
// Get a reference to the main module.
PyObject* main_module =
    PyImport_AddModule("__main__");

// Get the main module's dictionary
// and make a copy of it.
PyObject* main_dict =
    PyModule_GetDict(main_module);
PyObject* main_dict_copy =
    PyDict_Copy(main_dict);

// Execute two different files of
// Python code in separate environments
FILE* file_1 = fopen("file1.py", "r");
PyRun_File(file_1, "file1.py",
    Py_file_input,
    main_dict, main_dict);

FILE* file_2 = fopen("file2.py", "r");
PyRun_File(file_2, "file2.py",
    Py_file_input,
    main_dict_copy, main_dict_copy);
```

Ultra Dense, Powerful, Reliable... Datacenter Management Simplified!

15" Deep, 2-Xeon/Opteron or P4 (w/RAID) options



Customized Solutions for... Linux, BSD, W2K

High Performance Networking Solutions

- Data Center Management
- Application Clustering
- Network and Storage Engines

Rackmount Server Products

- **1U Starting at \$499:** C3-1GHz, LAN, 256MB, 20GB IDE
- 2U with 16 Blades, Fast Deployment & more...



iron
SYSTEMS™

Iron Systems, Inc.

540 Dado Street, San Jose, CA 95131

www.ironsystems.com

CALL: 1-800-921-IRON

ally. Generally, when a Python call hands an object out to a C program, it increments the reference count by one. The C code can then do what it likes with the object without worrying that it will be deleted out from under it. Then when the C program is done with the object, it is responsible for releasing its reference by making a call to `Py_DECREF()`.

It's important, though, to remember when you copy a pointer within your C program that may outlast the pointer from which you're copying, you need to increment the reference count manually, by calling `Py_INCREF()`. For example, if you make a copy of a `PyObject` pointer to store inside an array, you'll probably want to call `Py_INCREF()` to ensure that the pointed-to object won't get garbage-collected after the original `PyObject` reference is decremented.

Executing Code from a File

Now let's take a look at a slightly more useful example to see how Python can be embedded into a real program. If you take a look at Listing 4, you'll see a small program that allows the user to specify short expressions on the command line. The program then calculates the results of those expressions and displays them in the output. To add a little spice to the mix, the program also lets users specify a file of Python code that will be loaded before the expressions are executed. This way, the user can define functions that will be available to the command-line expressions.

Two basic Python API functions are used in this program, `PyRun_SimpleString()` and `PyRun_AnyFile()`. You've seen `PyRun_SimpleString()` before. All it does is execute the given Python expression in the global environment. `PyRun_SimpleFile()` is similar to the `PyRun_File()` function that I discussed earlier, but it runs things in the global environment by default. Because everything is run in the global environment, the results of each executed expression or group of expressions will be available to those that are executed later.

Getting a Callable Function Object

Now, let's say that instead of having our expression calculator execute a list of expressions, you'd rather have it load a function `f()` from the Python file and execute it a variable number of times to calculate an aggregate total, based on a number provided on the command line. You could execute the function simply by running `PyRun_SimpleString("f()")`, but that's really not very efficient, as it requires the interpreter to parse and evaluate the string every time it's called. It would be much better if we could reference the function directly to call it.

If you recall, Python stores all globally defined functions in the global dictionary. Therefore, if you can get a reference to the global dictionary, you can extract a reference to any of the defined functions. Fortunately, the Python API provides functions for doing just that. You can see it in use by taking a look at Listing 5.

To obtain the function reference, the program first gets a reference to the main module by "importing" it using the `PyImport_AddModule("__main__")` function. Once it has this reference to the main module, the program uses the `PyModule_GetDict()` function to extract its dictionary. From there, it's simply a matter of calling `PyDict_GetItemString(global_dict, "f")` to extract the function from the dictionary.

Now that the program has a reference to the function, it can call it using the `PyObject_CallObject()` function. As you can

Listing 4. A Simple Expression Calculator

```
#include <python2.3/Python.h>

void process_expression(char* filename,
                       int num,
                       char** exp)
{
    FILE* exp_file;

    // Initialize a global variable for
    // display of expression results
    PyRun_SimpleString("x = 0");

    // Open and execute the file of
    // functions to be made available
    // to user expressions
    exp_file = fopen(filename, "r");
    PyRun_SimpleFile(exp_file, exp);

    // Iterate through the expressions
    // and execute them
    while(num-- > 0) {
        PyRun_SimpleString(*exp++);
        PyRun_SimpleString("print x");
    }
}

int main(int argc, char** argv)
{
    Py_Initialize();

    if(argc != 3) {
        printf("Usage: %s FILENAME EXPRESSION+\n");
        return 1;
    }
    process_expression(argv[1], argc - 1, argv + 2);
    return 0;
}
```

see, this takes a pointer to the function object to call. Because the function itself already exists in the Python environment, it is already compiled. That means when you perform the call, there is no parsing and little or no compilation overhead, which means the function can be executed quite quickly.

Passing Data in Function Calls

At this point, I'm sure you're starting to think, "Gee whiz, this is great but it would be a whole lot better if I could actually pass some data to these functions I'm calling." Well, you need wonder no longer. As it turns out, you can do exactly that. One way is through the use of that mysterious `NULL` value that you saw being passed to `PyObject_CallObject` in Listing 5. I'll talk about how that works in a bit, but first there is a much easier way to call functions with arguments that are in the form of C/C++ data types, `PyObject_CallFunction()`. Instead of requiring you to perform C-to-Python conversions, this handy function takes a format string and a variable number of arguments,

Conferences: April 3 – 6, 2006
Expo: April 4 – 6, 2006

Boston Convention
& Exposition Center

2 Conferences
10 Tracks
100+ Sessions
Over 200 Exhibitors

Introducing: **OpenSolutions**
WORLD

OPEN Source.
OPEN Solutions.



OPEN. For Business.

LinuxWorld Conference tracks

- *Mobile & Embedded Linux*
- *Network Management*
- *Kernel and Driver Development*
- *Linux on the Desktop*

OpenSolutions World Conference tracks

- *Business Case*
- *Security*
- *Open Source Applications*
- *Enterprise Application Development*
- *Dynamic IT: Cluster/Grid Virtualization*
- *Emerging Trends*

Open source means business.

Find out what it can mean for your business at LinuxWorld Conference & Expo—in ten tracks and more than 100 in-depth sessions, tutorials, and workshops led by the open source movement's top minds.

Go beyond Linux.

Expand your vision at OpenSolutions World, a new conference at LinuxWorld covering the full spectrum of open source solutions and strategies for the enterprise.

Get the full picture.

Explore a comprehensive exhibition of open source products, technologies, and services from more than 200 key Linux and open source vendors—all under one roof.

Put it to work.

Come to LinuxWorld Conference & Expo April 3 – 6, 2006, and harness the power of the entire open source community for your business. The future of enterprise technology is wide open.

Register by 3/3 and save

Enter priority code D0102 and save up to \$500
www.linuxworldexpo.com/boston



Listing 5. Using Callable Function References

```
#include <python2.3/Python.h>

void process_expression(int num, char* func_name)
{
    FILE*      exp_file;
    PyObject*  main_module, * global_dict, * expression;

    // Initialize a global variable for
    // display of expression results
    PyRun_SimpleString("x = 0");

    // Open and execute the Python file
    exp_file = fopen(exp, "r");
    PyRun_SimpleFile(exp_file, exp);

    // Get a reference to the main module
    // and global dictionary
    main_module = PyImport_AddModule("__main__");
    global_dict = PyModule_GetDict(main_module);

    // Extract a reference to the function "func_name"
    // from the global dictionary
    expression =
        PyDict_GetItemString(global_dict, func_name);

    while(num--) {
        // Make a call to the function referenced
        // by "expression"
        PyObject_CallObject(expression, NULL);
    }
    PyRun_SimpleString("print x");
}
```

much like the `printf()` family of functions.

Looking back at our calculator program, let's say you want to evaluate an expression over a range of noncontiguous values. If the expression to evaluate is defined in a function provided by the loaded Python file, you can get a reference as normal and then iterate over the range. For each value, simply call `PyObject_CallFunction(expression, "i", num)`. The "i" string tells Python that you will be passing an integer as the only argument. If the function you were calling took two integers and a string instead, you could make the function call as `PyObject_CallFunction(expression, "iis", num1, num2, string)`. If the function has a return value, it will be passed to you in the return value of `PyObject_CallFunction()`, as a `PyObject` pointer.

That's the easiest way to pass arguments to a Python function, but it's not actually the most flexible. Think about it for a second. What happens if you are dynamically choosing the function to call? The odds are that you're going to want the flexibility to call a variety of functions that accept different numbers and types of arguments. However, with `PyObject_CallFunction()`, you have to choose the number and type of the arguments at compile time, which hardly fits with the spirit of flexibility inherent in embedding a scripting language.

The solution is to use `PyObject_CallObject()` instead. This function allows you to pass a single tuple of Python objects instead of the variable-length list of native C data items. The

downside here is that you will need to convert native C values to Python objects first, but what you lose in execution speed is made up for in flexibility. Of course, before you can pass values to your function as a Python tuple, you'll need to know how to create the tuple, which brings me to the next section.

Converting Between Python and C Data Types

Python data structures are returned from and passed to the Python interpreter in the form of `PyObject`s. To get to a specific type, you need to perform a cast to the correct type. For instance, you can get to a `PyIntObject` pointer by casting a `PyObject` pointer. If you don't know for sure what the variable's type is, though, blindly performing a cast could have disastrous results. In such a case, you can call one of the many `Check()` functions to see if an object is indeed of an appropriate type, such as the `PyFloat_Check()` function that returns true if the object could indeed be cast to a float. In other words, it returns true if the object is a float or a subtype of a float. If you'd rather know whether the object is exactly a float, not a subclass, you can use `PyFloat_CheckExact()`.

The opaque `PyObject` structure isn't actually useful to a C program though. In order to access Python data in your program, you'll need to use a variety of conversion functions that will return a native C type. For example, if you want to convert a `PyObject` to a long int, you can run `PyInt_AsLong()`. `PyInt_AsLong` is a safe function, and will perform a checked casting to `PyIntObject` before extracting the long int value. If you know for sure that the value you're converting is indeed an int, it may be wasteful to perform the extra checking—especially if it's inside of a tight loop.

Often, Python functions ask for or return Python sequence objects, such as tuples or lists. These objects don't have directly corresponding types in C, but Python provides functions that allow you to build them from C data types. As an example, let's take a look at building a tuple since you'll need to be able to do that to call a function using `PyObject_CallObject()`.

The first step to creating a new tuple is to construct an empty tuple with `PyTuple_New()`, which takes the length of the tuple and returns a `PyObject` pointer to a new tuple. You can then use `PyTuple_SetItem` to set the values of the tuple items, passing each value as a `PyObject` pointer.

Conclusion

You should now have enough to get started with embedding Python scripts inside your own applications. For more information, take a look at the Python documentation. "Extending and Embedding the Python Interpreter" goes into more detail on going the other direction and embedding C functions inside Python. The "Python/C API Reference Manual" also has detailed reference documentation on all of the functions available for embedding Python in your program. The *Linux Journal* archives also contain an excellent article from Ivan Pulley that discusses issues for multithreaded programs that embed Python.

Resources for this article: www.linuxjournal.com/article/8714.

William Nagel is the Chief Software Engineer for Stage Logic, LLC, a small software development company, where he develops real-time systems based on Linux. He is also the author of "Subversion Version Control: Using the Subversion Version Control System in Development Projects".





FREE SOFTWARE
FOUNDATION



be **FREE**



Join us at member.fsf.org



Three Case Studies in Community-Oriented, Open-Source Software Development

Community passion behind video devices and even a PDA makes these products viable to the Linux user.

BY RANDALL P. EMBRY

What do a Japanese PDA, a video capture card and a multimedia appliance that connects your television to the network have in common? They are all examples of specialized Linux-based devices with thriving subcultures surrounding them—communities of enthusiasts toiling nights and weekends obsessively coercing their toys into working exactly the way they want them to work. Of course, 15 years ago, Linux itself had a similar beginning.

“Hello everybody out there using minix—I’m doing a (free) operating system (just a hobby, won’t be big and professional like gnu) for 386(486) AT clones.” Not satisfied with the manufacturer’s choices, Linus Torvalds wanted to use his Intel-based computer on his terms. He started hacking, and then he used these simple words to reach out to the world, announcing his achievement and attracting helpers. Does this spirit live on today?

July 4, 2005. Eerily, once again, the IVTV developers have made a release right as I picked up the project. Maybe I should wait it out a bit until things settle down. Their progress fluctuates—first the audio isn’t right, then the video looks bad. The install script sure is easier this time—no more mysterious sequence of modprobes to get things working. The main guy working on this doesn’t even have the same model as I do—how does he do it?

So continues one of my typical holiday or weekend ventures into a randomly discovered Linux project. In this article, in addition to profiling a few specific Linux-friendly products and project accomplishments, I also provide a brief survey of some Open Source communities encompassing them, with a focus on one prominent person in each community.

Hauppauge WinTv PVR-150MCE

First, I start with a Microsoft Windows product—the prefix and suffix in its model number both describe its target market. The WinTv PVR-150MCE by Hauppauge is a TV tuner and a video capture card. Directly storing broadcast-quality video is impractical with current technology. Even if your hard drive’s

throughput is sufficient to receive the data, your available storage space will be quickly consumed. MPEG-2 compression, the same as used by the DVD standard, eases the requirements considerably, using less storage space and decreasing the streaming rate to a more manageable 6Mbps. Moderate compromises in quality reduces the storage requirements to 2G or less per hour of video. Compressing video is computationally intensive and taxes the fastest processors, but the PVR-150MCE gets around this by using its MPEG-2 encoder hardware, relieving your system from the time-consuming chore of compressing data. So, you can record compressed video in real time using modest hardware. It costs less than \$100 US, and if you don’t already know why I am describing a Windows Media Center Edition (MCE) product in a magazine about Linux, you will after reading the next paragraph.

IVTV is a project that allows Conexant-based video capture devices to run under Linux. The PVR-150MCE is but one of several in a family of cards based on the Conexant chipset. Hauppauge makes many of these cards, but several other manufacturers produce cards based on Conexant. Around 12 different cards are expected to work now or in a future version of the driver. By the way, the name IVTV is pronounced “Ivy TV”. It is not an acronym, but was inspired by an early Conexant model number, iTVC15. Other manufacturers who make compatible cards include Yuan, Avermedia and Adaptec. Please see the IVTV Web site (see the on-line Resources) for details. Although they have the same core chipset, these cards are by no means identical, and supporting all the subtle permutations is easier said than done. In extreme cases, you might even have to travel across the Atlantic to experience the differences.

In addition to several different models of these cards, both the European PAL and the US NTSC video standards are supported. Testing both of these signals is especially tricky, because a given developer has easy access only to one of these sources of video. This group has coordinated internationally to provide good support for both. Is this just the matter of flipping a bit somewhere in the source? Or are they exploiting the fact that light travels around the globe in microseconds and that a developer in Europe is going to bed around the time his or her

US counterpart is awakening? A Netherlands-based developer's request for an NTSC modulator leads me to believe the latter.

The lead developer for this project, Chris, welcomes donations:

I have been maintaining the IVTV driver since around December 2003, keeping all patches merged and also have reverse engineered the DMA, and redesigned the internal parts of the driver to allow growth and stability. I have devoted many hours to IVTV, often more than 40 hours a week in the past, and hope to continue work like this into the future. I have recently divorced, have a need for income since most is taken now, and would appreciate any donations to help and hopefully my work on the driver helps you too. I have a lot of fun working on the driver and am glad to be able to provide this service to people in getting the MPEG-2 encoding chip working the best it possibly can on these WinTV/etc. pci cards.

Hauppauge MediaMVP

Next, the MediaMVP is a solid-state device with an Ethernet jack, S-Video/Composite outputs and stereo RCA audio. For around \$100 US, you can telnet in to a box running Linux that plugs in to your TV. Well, almost. Before you can telnet in, you probably will need to install `mvpmc`.

Setup for this project is well documented, but it isn't trivial. It involves setting up a DHCP server and a TFTP server for delivering the alternative bootloader to the device. You proba-

shirt pocket) product only in Japan, but a lively community has developed around it, many of whom import the \$800 US device directly from Japan. It is an awesome machine—a 640x480 VGA display, 4G microdrive storage, in a clamshell case with a QWERTY keyboard, USB, CF and SD. A testament to its Japanese roots, it omits Bluetooth and Wi-Fi, as those wireless protocols are less ubiquitous in that market. The community offers advice ranging from tips on importing to practical experience with particular hardware compatibility.

The Zaurus has tremendous potential, but is not necessarily easy. There seems to be a trade-off between functionality and stability. Seeking the latest software can be rewarding. So whether you perceive it as half full or half empty, and whether you buy one and where you buy it are personal decisions. For warranty repairs, the unit would need to be returned to Japan. Who pays the shipping and handling costs? It depends on whether you purchase one from a value-added reseller or from a no-frills exporter. The community can help you with this choice. By the way, if you import it yourself, you might find the Japanese version of “tap the screen to get started” a bit intimidating, and you will probably need to take advantage of an English translation. Again, the Zaurus community has exploited the speed of light to overcome many international barriers—step-by-step guides to navigating the foreign instructions are readily available.

Zaurus dominates discussions on the Open Embedded

Compressing video is computationally intensive and taxes the fastest processors, but the PVR-150MCE gets around this by using its MPEG-2 encoder hardware, relieving your system from the time-consuming chore of compressing data.

bly also want to export one or more NFS shares to provide content to the device. Building the source involves setting up a cross-compiler, unless your `gcc` happens to target the `powerpc-405-linux-uclibc` architecture. Nightly automated builds are also available as an alternative to cross-compiling the source.

The reward for this hard work is that you can browse your filesystem for multimedia files or stream video from MythTV using a device that has no moving parts and no concerns about overheating or fan noise.

For nearly two years, Jon, a professional firmware engineer living in the United States, has tirelessly checked in improvements and responded to questions on the project's mailing lists. In response to the question, “Jon, did you ever consider accepting donations for this project via SourceForge?” he sums up a common but noble attitude: “No. I'd prefer it if people would send me code instead of money. Or hardware, if you want something supported that I wouldn't otherwise buy.”

Sharp Zaurus SL-C3100

Finally, if your definition of a personal information manager includes the latest version of Emacs that you compiled yourself, then the Sharp SL-C3100 could be a dream come true. Sharp sells this versatile (think modest laptop that fits in your

Software Foundation (OESF) forums. There you will probably meet “Meanie”, a prolific advocate for Sharp/Linux from Australia. He has captured the essence of Zaurus ownership in a series of Web pages that detail his customizations and enhancements. Meanie's hard work let me vicariously assess the product before committing to its purchase. In a recent posting, he alerts us that he fried his \$800 Zaurus SL-C3000 when using a generic USB cable. The community recognizes his valuable presence and offers to help, but he turns down their repeated offers to chip in to buy him a replacement. Instead, he chronicles his purchase of a newer model Zaurus SL-C3100 plus suggests that well-wishers donate to his favorite Zaurus developer.

But sometimes a little financial support goes a long way. Consider this request from `pdaXrom`, a popular Zaurus subproject: “Did you know? That our main programmer needs only a monthly total of \$200 US to keep all his bills paid and thereby keeping focus on developing....”

Conclusions

So, why these particular products? They happen to be three of my own personal latest technology purchases. I did my research before buying them, and I am certainly biased toward

I can use my Zaurus PDA's Web browser to access MythTV and select a program to record using my PVR-150MCE, and I can also use it to telnet in to my MediaMVP for debugging, exploration and upgrades.

Linux products, but otherwise, these communities were chosen fairly arbitrarily. In case you are wondering, yes, they work together smoothly. I can use my Zaurus PDA's Web browser to access MythTV and select a program to record using my PVR-150MCE, and I can also use it to telnet in to my MediaMVP for debugging, exploration and upgrades. (The actual video playback is initiated by the remote control.) More important, after some grueling transcoding, I can bypass the MediaMVP and watch *SpongeBob SquarePants* on the 3.7" screen.

Hauppauge has been commendably supportive by providing solid documentation and allowing redistribution of its copyrighted firmware, simplifying installation. This demonstrates the company's progressive values as well as the diligent communication from the IVTV developers. So don't let the questionably redundant product name dissuade you from considering this device. Who knows, maybe someday Hauppauge will re-brand

this card with a name like "LinTv PVR-150 MythTV Edition".

Sharp has complemented its hardware with GPL software and has, of course, followed through by supplying source for its derivative work. On top of that, it has encouraged developers with documentation and occasional promotions. Let's hope future versions of the Zaurus will again be directly promoted outside of the Japanese market.

Two of these projects benefit directly from the GPL's stringent requirements. I am not being a zealot—I have personally found occasions where other licenses were effective and convenient—but it is hard to imagine these manufacturers releasing source code if they were not obligated.

All of these projects are successful because they have passionate individuals banded together in a community. These individuals are otherwise ordinary developers who have become involved in these projects and are now key members of thriving project communities.

The people running these projects are not just sitting at home ranting on Slashdot—they are spending their time creatively, showing the discipline to do the next right thing in order to develop great products systematically for the world to enjoy freely. They are skilled technicians who have become community leaders, patiently and quickly answering questions and offering advice.

These communities are focused and not burdened by formality. It is a practical affair—typically, you come to the community with a specific problem and it supplies answers. I personally tend to lurk—scouring the forums until I find the answer to my problem, then slinking away. Perhaps, instead of slinking, it would be better for all if I posted a quick "thank you" message—something to the effect of, "Thanks for this valuable resource; I was able to find the answer to my question of XYZ by reading through the past threads. By the way, the answer is ABC, in case someone stumbles onto this message while in search of it."

How can you support these and similar projects? If the answer is not already posted in a FAQ, start by asking! There is no single answer—as noted here, some projects actively seek financial support while others shun it. Be sure to pay attention to any advertisements on the various sites and visit their sponsors (when appropriate). Study the community for a bit, and if it seems right, try jumping in.

Resources for this article: www.linuxjournal.com/article/8718.

Randall P. Embry (randall@embry.com) is an employee of Indiana University paid to develop open-source software for higher education. He has worked on the Sakai course management system and the OSP electronic portfolio. He is now a development manager for the Kuali financial information system.

YOUR AD HERE.

Contact Linux Journal to find out how your company can reach **hundreds of thousands** of Linux professionals every month.



Request a free media kit

206-782-7733 ext. 2 or ads@linuxjournal.com

www.linuxjournal.com/advertising



www.linuxjournal.com

FREE NEWSLETTER!

Wish you could get the latest from *LJ* more than once a month? You can—sign up today for *LJ's* weekly e-mail newsletter.

Each week the *LJ* newsletter features great tech tips, links to web-only articles, and news on the latest events in the Linux market.

Sign up for the *LJ* e-mail newsletter now:

<http://www.linuxjournal.com/>



Who says penguins can't fly?

<http://store.linuxjournal.com/>



Secure Remote Control & Support for Linux

Award-winning NetOp Remote Control for Linux provides secure, cross-platform, remote control, access and support. NetOp lets you view and control a remote PC's current desktop session, transfer and synchronize files, launch applications or chat with the remote user - just as if you were seated at that computer.

- > Cross-Platform support for Linux, Solaris, Mac OS X, & all Windows platforms
- > Advanced security including encryption, multiple passwords, even centralized authentication & authorization with the optional NetOp Security Server module



Try it Free - www.CrossTecCorp.com

NetOp and the red kite are registered trademarks of Danware Data AS. Other brand and product names are trademarks of their respective holders. 2005 Copyright Danware Data AS. Allrights reserved.

Make Your Resolution A Fitrix Software Solution!

Fitrix
Exact Fit Accounting & Business Software
Source Code Accounting & Distribution Software
For Linux

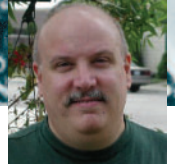
Low Total Cost Of Ownership!

Free Trial!

www.fitrix.com
800.374.6157
770.432.7623

SUBSCRIBE

www.linuxjournal.com/subscribe



Skim Cream not Scum

Pack up the Microsoft sycophants and shoot them off to Mars.

BY NICHOLAS PETRELEY

When did Microsoft lose its status among open-source developers as the evil, or better still, incompetent empire? When did open-source developers stop trying to make software better than Microsoft's and start imitating everything Microsoft does? Why do we have to have an open-source Outlook, or an open-source dotNet? Sure, there are examples of how we do things better in Linux than Windows. But I'm getting really tired of the monkey see Microsoft, monkey do Microsoft mentality that has infected open source. And the operative word here is "monkey"—hint, hint.

Aside from being open and free, isn't superiority what got Linux where it is today? Once upon a time, Microsoft was under intense pressure to catch up to Linux stability (in my unhumble opinion, Microsoft still has a long way to go). Now we have several projects that exist for no other reason than to catch up to and duplicate Microsoft software. Worse, we're duplicating architectural nightmares like the registry, and with no other apparent purpose than to be more like Windows.

I have nothing against cream-skimming the best features of Windows for use in Linux. But creating a registry for Linux is not cream-skimming. It's pond-scum-skimming. What happened to the days when people were appalled at the idea that you'd have to edit a registry in order to make this or that feature work the way you wanted? I don't care if the registry is binary or XML. It's a maintenance nightmare.

Next time you visit Redmond, take a look at big hole with teeth marks in the Microsoft butt. That's a "came back and bit it" bite mark left by the registry. While Microsoft is trying to get around its mistakes, we're busy duplicating them.

Here's another example. Emulating what OLE 2.0 brought to Microsoft Office is not cream-skimming. It's biohazardous-medical-waste-skimming.

I remember the original Microsoft demos of OLE 2.0. You paste spreadsheet cells into a Word document. You click on the cells and the word processor magically transforms into a spreadsheet program. That makes good demo, but did anyone ask what real value it offers? Aside from looking cool, that is? This feature is bad not because of what it does, but because of what it fails to do. It fails to make it easy to create a live link between the original spreadsheet data and what you paste into the document.

Fortunately, not everyone has imbibed from the punch bowl of Microsoft cool-aid. EIOffice, although it looks and feels

more like Microsoft Office than OpenOffice.org or KOffice, actually came up with a fresh idea. Imagine that. Innovation. But it took a commercial company, not an Open Source community, to do it. The folks at Evermore Software (the makers of EIOffice) must have at least one non-Microsoft drone on board to enlighten the developers as to what really matters. EIOffice gives you a menu selection to paste a bit of spreadsheet into a document where the cells are live-linked to the original spreadsheet data.

And this next bit of information should send open-source fanatics into a tizzy. EIOffice is based on that evil, despicable language called Java. How dare they? Mono C#, Python, Ruby, maybe even Perl. But Java? Won't that encourage Sun to become dictator of the world if EIOffice gets popular? It's perfectly fine to copy Win32 DLLs in order to make Linux do Windows tasks, but heaven forbid Linux should be infected with a Java runtime. How Sun replaced Microsoft as the evil empire is beyond me. But don't get me started on that.

Back to OLE 2.0 and its successors. Of course, the OpenOffice.org and KOffice folks have faithfully duplicated this monstrosity. Hey, it's how Microsoft Office works. It must be the way to go, right?

That's what they want you to think. Who is they? I don't know, but I can't help but wonder if one or more people within some of these open-source projects are Microsoft moles.

"Here's the plan. Infiltrate the Open Source community and neutralize it. Convince them that the only way to compete with Microsoft is to create open-source versions of Microsoft's great software and development tools. Without our patents, you'll always be several steps behind everything we do. And the fact that you're imitating us makes us look like the clear innovation leaders. End result? Free advertising and marginalization of the value of having Linux."

It's time to boot the Microsoft suck-ups out of the Open Source community. Give them a free copy of Windows XP, Visual Studio dotNet and two months to use these to create a navigation system for the rocket we'll use to send them to Mars.

Yeah, but what do I really think?■

Nicholas Petreley is Editor in Chief of *Linux Journal* and a former programmer, teacher, analyst and consultant who has been working with and writing about Linux for more than ten years.



Rackspace — Managed Hosting backed by Fanatical Support.™

Servers, data centers and bandwidth are not the key to hosting enterprise class Web sites and Web applications. At Rackspace, we believe hosting is a service, not just technology.

Fanatical Support is our philosophy, our credo. It reflects our desire to bring responsiveness and value to everything we do for our customers. You will experience Fanatical Support from the moment we answer the phone and you begin to interact with our employees.

Fanatical Support has made Rackspace the fastest-growing hosting company in the world. Call today to experience the difference with Fanatical Support at Rackspace.



Thanks for honoring us with the 2004 Linux Journal Readers' Choice Award for "Favorite Web-Hosting Service"



1.888.571.8976 or visit us at www.rackspace.com

The World's Fastest InfiniBand Switch

From a Company You've Trusted for 23 Years

Microway's **FasTree™** DDR InfiniBand switches run at 5GHz, twice as fast as the competition's SDR models.

FasTree's non-blocking, flow-through architecture makes it possible to create 24 to 72 port modular fabrics which

have lower latency than monolithic switches. They

aggregate data modulo 24 instead of 12, improving nearest neighbor latency in fine grain problems and doubling the size of the largest three hop fat tree

that can be built, from 288 to 576 ports. Larger fabrics can be created linking 576 port domains together.

▲ 72 Port FasTree™ Configuration

Working with PathScale's InfiniPath HTX Adapters, the number of hops required to move MPI messages between nodes is reduced, improving latency. The modular design makes them useful for SDR, DDR and future QDR InfiniBand fabrics, greatly extending their useful life. Please send email to fastree@microway.com to request our white paper entitled *Low Latency Modular Switches for InfiniBand*.



Microway's **QuadPuter®** includes four AMD single or dual core Opteron™ processors, 1350 Watt redundant power supply, and up to 5 redundant, hot swap hard drives—all in 4U. One of the most powerful processing platforms in the HPC industry, QuadPuter can serve as a cluster node or a standalone supercomputer.

Constructed with stainless steel, its RuggedRack™ architecture is designed to keep the processors and memory running cool and efficiently. The power supply exhaust does not mix with air in the motherboard chamber. Hard drives are cooled with external air and are front-mounted along with the power supply for easy access and removal. The RuggedRack™ is available with an 8-way motherboard, dual-core Opterons and up to 128 GB of memory for power- and memory-hungry SMP applications.

Call us first at 508-746-7341 for quotes and benchmarking services. Find technical information, testimonials, and newsletter at microway.com.



◀ QuadPuter® Navion™ with Hot Swap, Redundant Power & Hard Drives

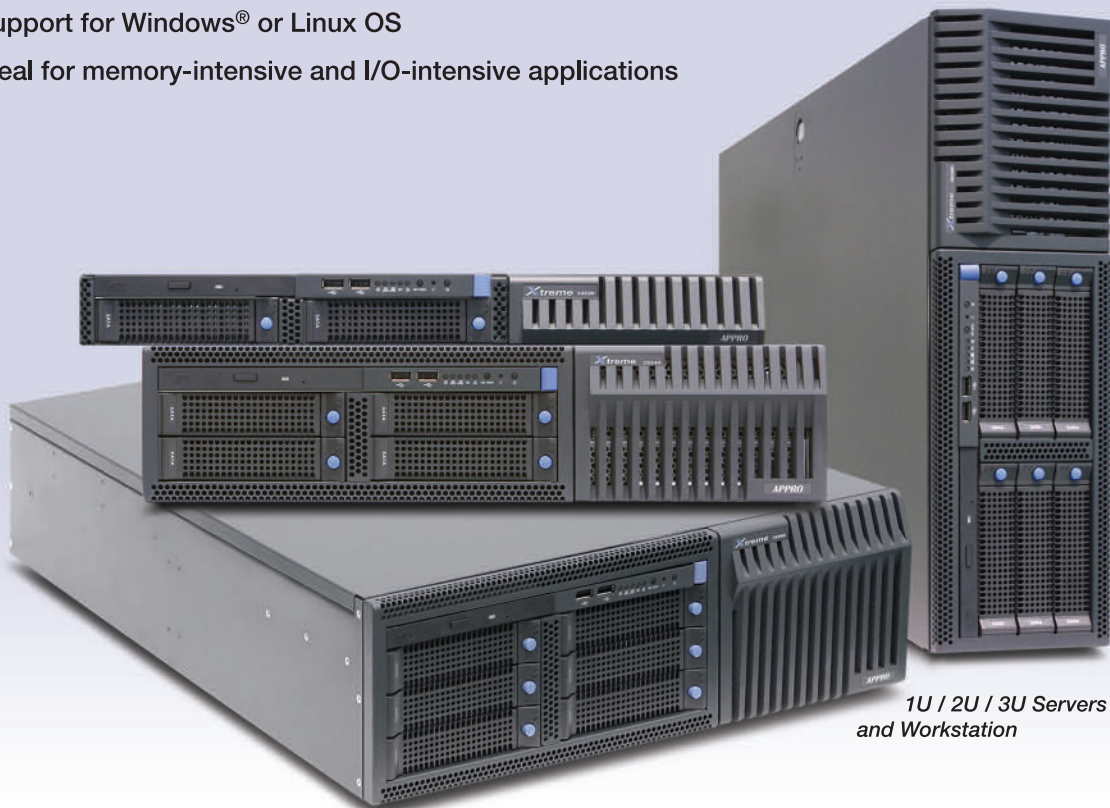


508.746.7341 microway.com

Let's be Xtreme

Introducing Appro Xtreme Servers & Workstations with 8 DIMM Sockets per CPU

- 2-way or 4-way, Single or Dual-Core AMD Opteron™ processors
- **Largest memory capacity - 8 DIMM Sockets per CPU, up to 128GB**
- PCI-Express technology to increase I/O bandwidth and reduce system latency
- Outstanding Remote Management – IPMI 2.0 compliant
- Cable-free design, ready to run, simple to install, service and maintain
- Support for Windows® or Linux OS
- Ideal for memory-intensive and I/O-intensive applications



1U / 2U / 3U Servers
and Workstation

AMD Opteron™ Processors - AMD64 dual-core technology reduces memory latency and increases data throughput
- Dual-core processors with Direct Connect Architecture deliver the best performance per watt in the industry with little or no increase in power consumption or heat dissipation.



Appro delivers high-performance computing solutions to help you maximize productivity for a solid ROI. On-site maintenance and installation services are also available.

For more information, please visit www.appro.com
or call Appro Sales at 800.927.5464, 408.941.8100.

