Linux on the Routerboard 532

Presented by:

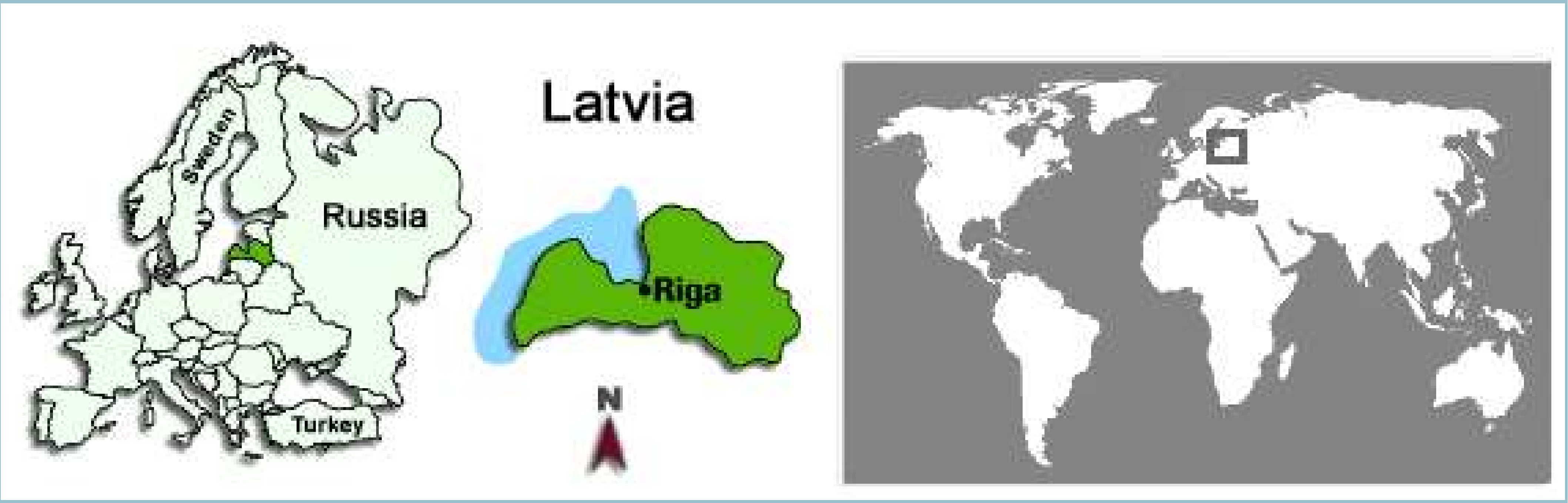
Andy Stewart
Worcester Linux Users' Group
Worcester, MA USA

February 16, 2006

Introduction

- My project goals
  - Use GPL software to make a home firewall
  - Desire to replace existing P233 Firewall
  - No moving parts, no noise, small size
  - Use SuSE Firewall scripts due to familiarity
- Routerboard 532
  - Single board computer
  - Many ads in Linux Journal magazine
  - Made by MikroTik® in Riga, Latvia
  - MikroTik RouterOS™ software in Flash
  - Linux reference images available for download

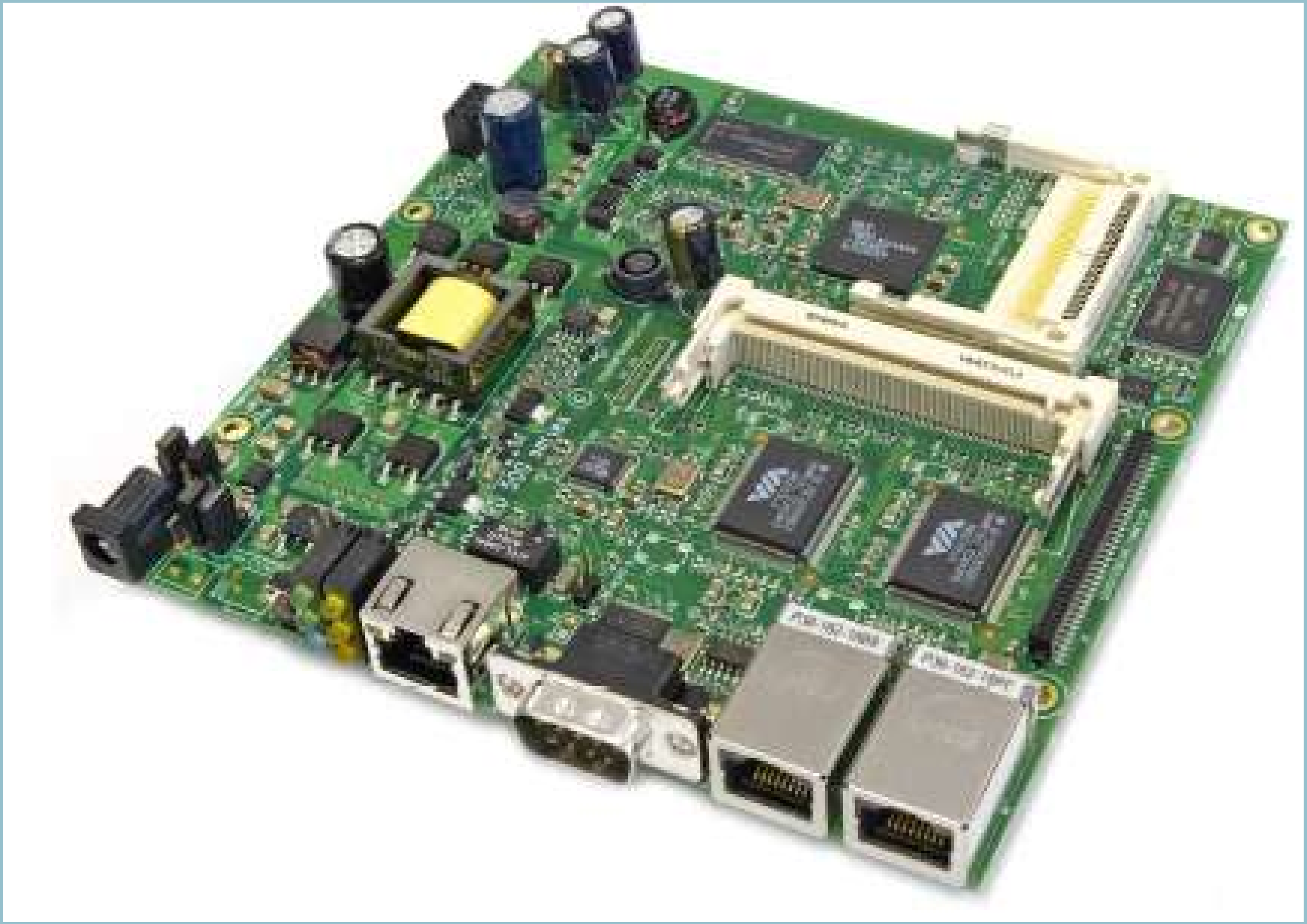
Where is Latvia?



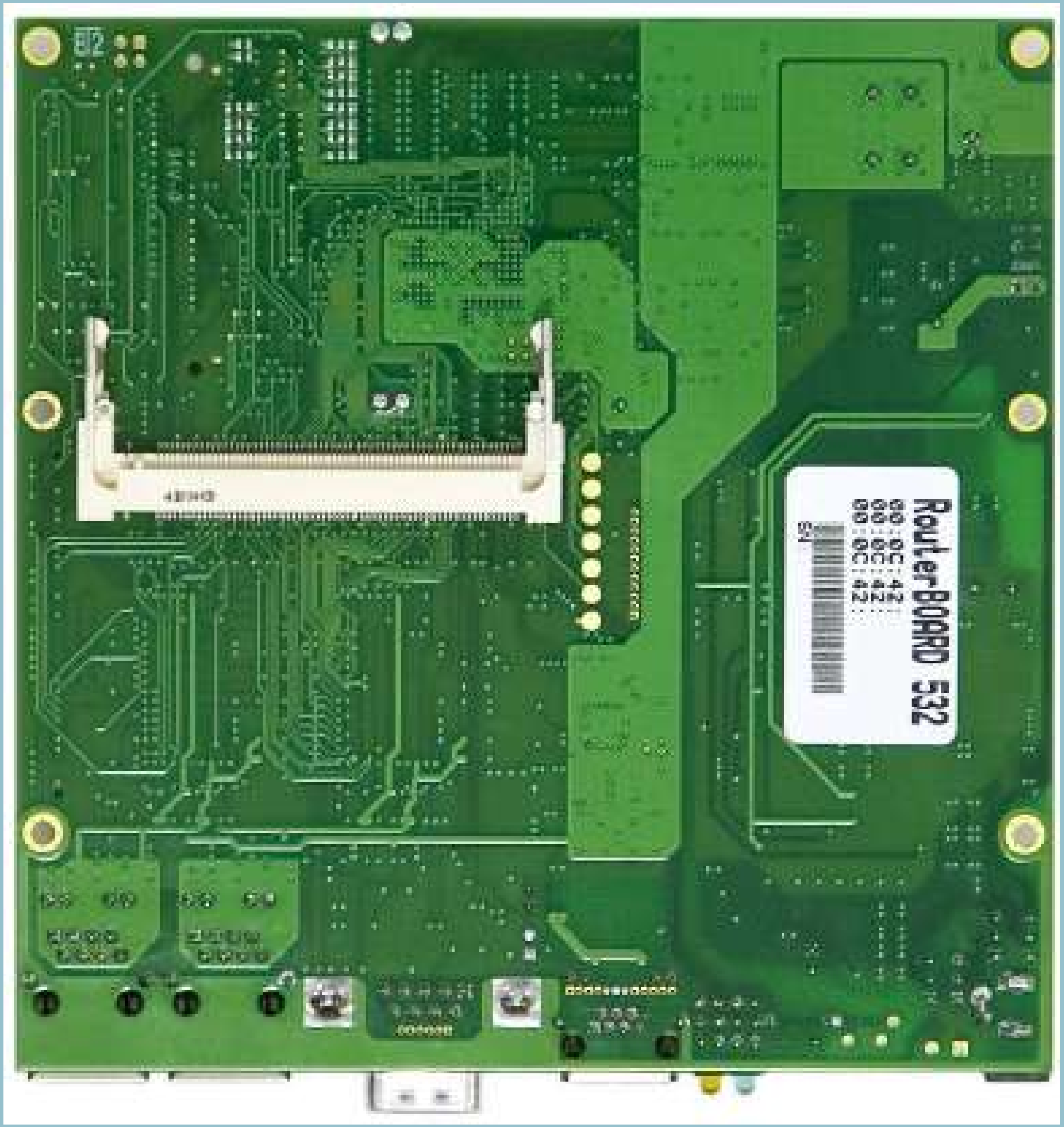
Details

- 266-400 MHz MIPS32 4Kc embedded μProc
  - IDT RC32434 (32 bit little endian, no FPU)
- 32 MB DDR memory, 64 MB NAND Flash
- Compact Flash slot (supports microdrives)
- 3 10/100 Ethernets (IDT Korina, VIA VT6105)
- 2 Mini PCI slots Type IIIA/IIIB
- 1 RS232C Serial Port (DB9)
- Input Voltage: 6-24VDC or 24-56VDC
- Power: 2-3 watts without extension cards
- Size: approx. 5.5” x 5.5”

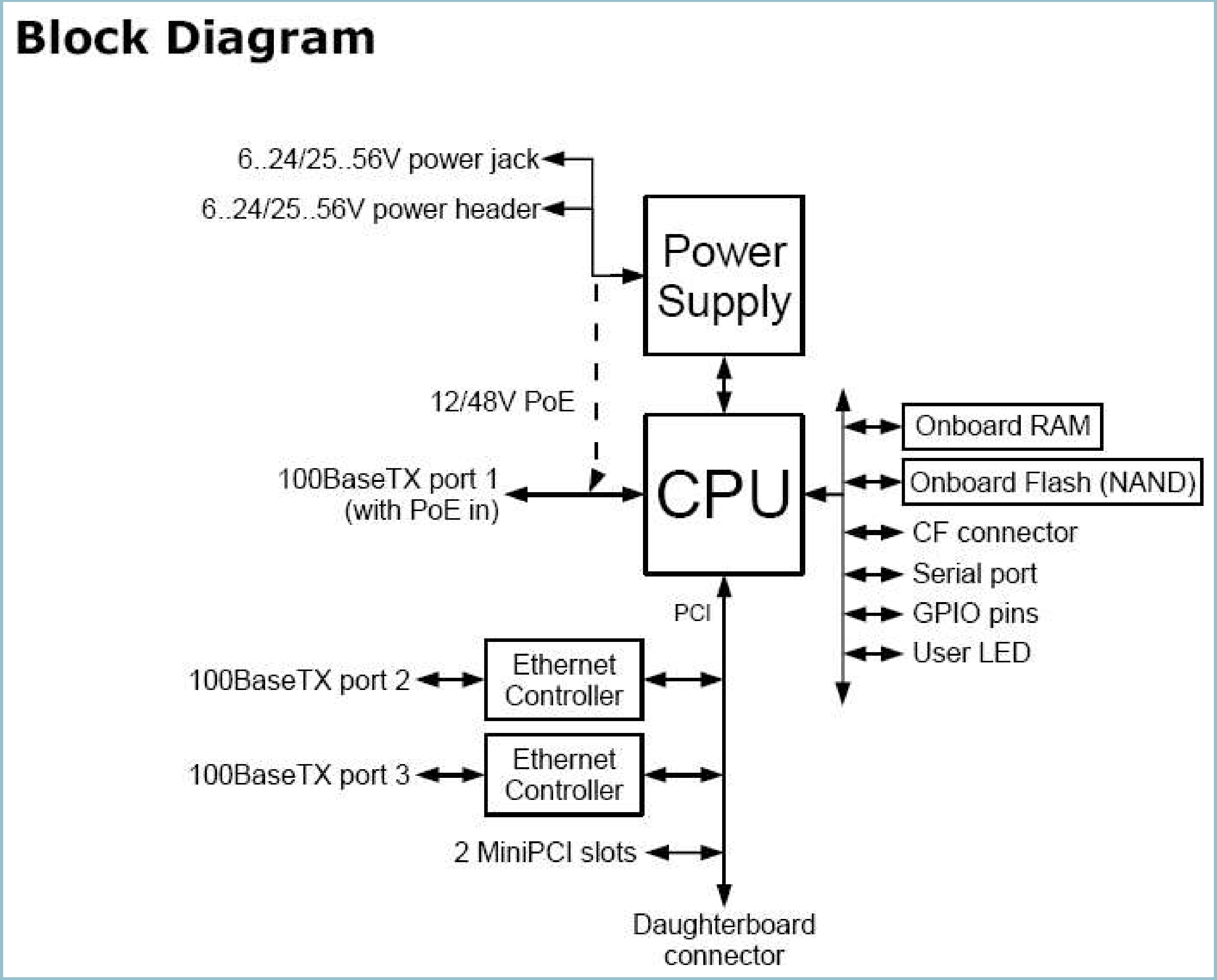
Front of the Routerboard 532



Back of the Routerboard 532



Block Diagram



Linux Reference Image

- For starters, copy the Linux reference images
  - http://www.routerboard.com/files/rb500-128.img.zip
  - http://www.routerboard.com/files/rb500-1024.img.zip
  - Based on Debian, Linux 2.4.30 kernel plus patches

- Copy this image onto the compact flash
  - Unzip the image file
  - fdisk -l rb500-128.img
  - Notice that there are 2 partitions
    - 1st: Type 27 (this is odd...)
    - 2nd: Type 83 (Linux...we can handle this!)

Copying Image to Compact Flash

- High Level Instructions
  - dd the first partition directly to the CF
  - Repartition the CF to modify size of 2nd partition
  - Loopback mount image file - 2nd partition
  - Copy those files to the CF with cp

- Details can be found on the cosam.org website (see References)

Booting the Reference Image

- Plug the CF into the Routerboard
- Connect serial cable to another machine
  - Use minicom to communicate with Routerboard
- Power on, set boot device to CF
- Boot Linux
- Log in and play

Creating Your Own From Scratch

- Starting from scratch – what to do?
- Considered Linux From Scratch
- Explored software for embedded Linux
- Discovered the following useful tools:
  - Buildroot – create cross compiler environment
    - Uses gcc and uClibc (uClibc == micro controller libc)
    - stripped down C library intended for embedded devices
  - Busybox – Swiss army knife of embedded Linux
    - Contains many programs in one executable!
    - Static or dynamic executable

Buildroot

- Download a buildroot snapshot
- Extract the files: tar xjvf .....
- cd buildroot
- make menuconfig
  - Hummm...this looks like the Linux kernel config...
  - target architecture is mipsel
  - I hacked in the 2.4.30 kernel headers
  - binutils 2.16.1, busybox 1.01, gcc 3.4.2, uClibc 0.9.28
  - Add other tools/programs as desired
    - dnsmasq, grep, iproute, iptables, nano, ntp, procps, strace – many others available!

Nice features of Buildroot

- Buildroot downloads selected software
- Configures software
- Builds it using cross compiler tools
- Creates necessary directories, symlinks
- Populates directories
- Sets file permissions
- Creates tmpfs in RAM for /var, /tmp, etc.
- Creates default version of password files
- This is a nice tool!

Busybox

- Busybox also has Linux kernel style config
  - cd build_mipsel/busybox-1.01
  - make menuconfig
  - Many choices here – pick just what you need
- Built in versions of most common utilities
  - Many are not full featured (save space and memory)
- Fine grained control over utils and features
- Provides init, sh, ls, cat, cp, mv, dd, etc, etc.
  - These are symlinked to busybox executable
- One stop shopping – this tool has it all!

Copy files to Compact Flash

- Compile busybox and utilities
  - One make command does it all!
- Buildroot makes a file system for you
  - build_mipsel/root
- Copy this to the compact flash - 2nd partition
  - Use rsync for this
  - You'll do this more than once in development
- Put compact flash in the Routerboard
- Boot it – log in!
- Change the default password in build_mipsel/root/etc/shadow before rsync

Linux Kernel

- Replaced kernel in MikroTik reference images
  - Missing needed features for packet filtering
- Stock 2.4.30 kernel, added MikroTik patches
  - http://www.routerboard.com/files/linux-2.4.30-yaffs2.patch.gz
- Configured as desired – make menuconfig
- Use dd to copy vmlinux to CF 1st partition
  - 1st partition is weird...be careful
  - See document on cosam.org for details (see References)
- Routerboard boot loader looks for 1st partition - type 27
  - boots image - first ELF header it finds - vmlinux

Init Scripts

- No run levels
- Need to write your own init scripts
- Busybox init wants to run /etc/init.d/rcS
- Chose to mimic startup scripts like SuSE
  - S[0-9][0-9]<name>
  - My rcS just calls these scripts in order
- My scripts start these services
  - Network, ssh, random number generator, ntpdate
  - SuSE Firewall, dnsmasq (DNS and DHCP)

SuSE Firewall Scripts

- Desire to port SuSE firewall scripts to this box
- These scripts work well on existing firewall
- Supports internal, external, DMZ networks
- Hacked out references to runlevels
- These scripts create iptables commands
- Configuration is in one well commented file
- Error messages from iptables not helpful
  - Difficult to know cause of error
  - Needed to add features to kernel to correct errors
  - Many iterations until this was correct

Testing

- Plug “internal” interface into home network
- Plug laptop into “external” interface
- Run nessus – it thinks port is dead – good!
- Enable ssh, test it, it works, nessus is happy
- Now, try replacing the existing firewall
  - So far, so good!

Development Environment

- 2xOpteron 244 (x86_64)
- 5 GB disk space
  - Downloads and reference images
  - Development software – cross compiler
  - Busybox and other utilities
  - Files compiled for Routerboard
- Compact Flash “box” for reading/writing CF
- Serial cable / minicom
- Routerboard was using NFS-root in early development (flashed just the kernel image)

Things not yet Completed

- Have not played with NAND Flash and yaffs
- Considering mini PCI for wireless
- Software items to complete
  - syslog to network server
  - ntpd
    - No battery backup for onboard clock
    - clock drifts while powered on
- Tried openvpn
  - got a simple p-t-p connetion established
  - next step seemed quite involved (public key infrastructure, etc)
  - it seems to be overkill for my needs

Comparison to Commercial Alternatives

- Commercial solutions
  - easier
  - cheaper
  - probably contain proprietary software
  - perhaps not exactly what you want
  - not as much fun

Conclusions

- Expenses
  - Routerboard $150, Project case $20
  - Compact Flash – $20
  - Laptop power supply – free!
  - Time – maybe a dozen hours
- How many viruses are there for MIPS Linux ?
- Learned a lot about embedded Linux software
- Pondering other embedded Linux projects
- Fun project !

References

- Routerboard related websites
  - http://www.routerboard.com
  - http://www.routerboard.com/archive.html
  - http://www.cosam.org/computers/hardware/rb500.html
  - http://www.mikrotik.com/index.html
- Buildroot
  - http://buildroot.uclibc.org/
  - http://www.uclibc.org
- Busybox
  - http://busybox.net
- Miscellaneous
  - http://www.linuxdevices.com/news/NS9341546150.html
  - http://openvpn.net