

Linux-MVME Targets Using Motorola Board Support

*Epics Collaboration Meeting – June 2006
IOC Operation Systems*

Joe Sullivan – EPICS System Developer (AES/BCDA)

14 June 2006





Overview

- Future of Embedded Linux
- Linux for vxWorks Developers
- Motorola MVME Linux Support
- Embedded Linux Development Kit
- Embedded Linux Installation and Implementation
- Linux Runtime
- EPICS Target Development
- EPICS Results
- Remaining Work

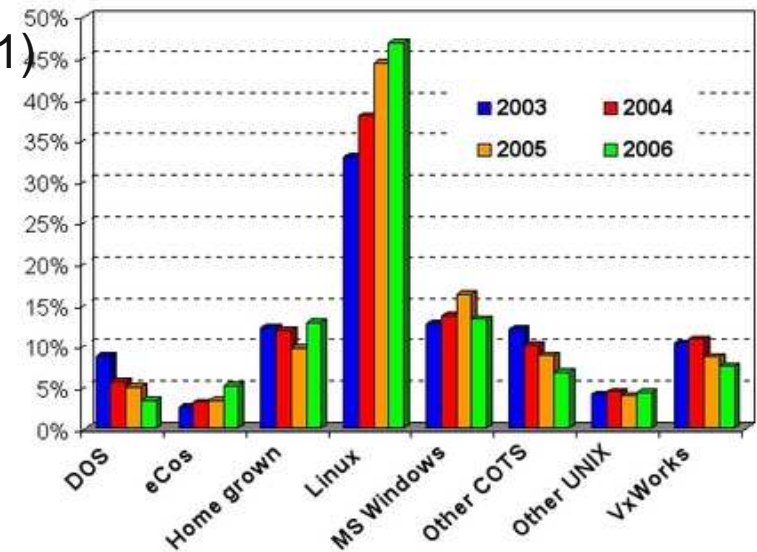


Future of Embedded Linux

Is Linux taking over the world? The 'truthiness' is....

- Google “embedded linux” – 6,830,000
 - Sponsored Links (2 pages)
- Single Board Computer Vendors claim Linux support
 - Motorola, SBS, Xycom, ...
- Web Magazine (www.linuxdevices.com)
 - Survey of Embedded Linux Distributions ('01)
 - 19 Commercial
 - 13 Open Source
- Wind River Workbench
 - Builds Linux targets.
- Popularity of EPICS Soft IOC's

Embedded OS sourcing trends



Future of Embedded Linux





Embedded Linux for VxWorks Developers

- OS Kernel
 - Soft Realtime
 - Configurable
 - Small
- Board Support Package
 - Hardware device support (counter/timers, I/O Ports, mezzanine slots)
 - VME Bus Support (memory windows , interrupt service, DMA)
 - Customizable (source code)
- Build Tools
 - Support Libraries
 - Cross Compiler/Linker
- Network Booting (diskless)



Motorola MVME Linux Support

- Current MVME Products
 - MVME3100 512MB, 844MHz, \$2100
 - MVME5500 512MB, 1GHz, \$3200
 - MVME6100 1GMB, 1.2GHz, \$4200
- Active in-house development to support Linux
 - Current Kernel Supported (v2.6.14)
 - *5 months behind the most recent: v2.6.16*
(<http://www.kernel.org/pub/linux/kernel>)
 - Improving VME bus memory and interrupt drivers based on customer feedback. (*Commercial RT Linux vendors perhaps?*).
- Linux Kernel support for other SBC (CompactPCI, PMC, etc)





Motorola MVME Linux Support

- Distribution is currently...informal
 - Contact: Ajit Prem Ajit.Prem@motorola.com
- Distribution Contents
 - Generic linux kernel patch file
 - *patch-2.6.14-ecc.ir01.03282006.gz*
 - Documentation
 - *README (linux_2614_eec_readme.txt)*
 - *vme_driver.pdf*
 - Example code – VME drivers
 - *vme_utils.tar.gz*
- Recommended Embedded Linux Development Kit
 - www.denx.de -> *ELDK*





Embedded Linux Development Kit

- What is it?
 - Linux distribution targeting embedded system development
- Pre-built cross-platform software development tools for embedded processors (PowerPC, ARM,).
 - Make, cross-compiler/linker(gcc), libraries.
- Prebuilt root filesystem.
 - /bin, /dev, /etc, /lib, /proc, /sys, /var, /usr,
- Nothing that could not be produced in-house but ..it's a pain!





Installation and Implementation

ELDK Installation

- Distribution (eldk.tgz – 2Gigbytes)
 - Buy CD \$99 Euro
 - Download:
 - *<http://sunsite.utk.edu/ftp/pub/linux/eldk/4.0/ppc-linux-x86>*
Use gftp (recursive download ~ 4hrs)
- Unpack distribution
 - cd /local/eldk
 - tar xzf ~/eldk.tgz
- Documentation
 - <file:///local/linux/eldk/4.0/ppc-linux-x86/distribution/README.html>
- Check files for executable permission
 - tools/bin/rpm, tools/usr/lib/rpm/rpmd, install, ELDK_MAKEDEV, ELDK_FIXOWNER



Installation and Implementation

ELDK Installation (cont)

- Install ELDK
 - `./install -d /local/eldk/eldk`
- Setup cross-built environment variables
 - `setenv CROSS_COMPILE ppc_74xx-`
 - `set path=($path /local/eldk/usr/bin /local/eldk/bin)`





Installation and Implementation

Embedded Linux Kernel Build

- 1) Get Kernel patch file and README from Ajit Prem Ajit.Prem@motorola.com
Save: ~/patch-2.6.14-ecc.ir01.03282006.gz

- 2) Downloaded Linux 2.6.14 Kernel from <http://www.kernel.org/pub/linux/>
Save: kernel/v2.6/linux-2.6.14.tar.gz -> ~/linux-2.6.14.tar.gz

- 3) Unzip Linux
 - cd /local/linux
 - tar xzf ~/linux-2.6.14.tar.gz

- 4) Patch Linux Kernel
 - cd linux-2.6.14
 - patch -Np1 < ~/patch-2.6.14-ecc.ir01.03282006



Installation and Implementation

Embedded Linux Kernel Build (cont)

- 5) Configure build for target architecture
/local/linux/ linux-2.6.14/Makefile edit
ARCH ?= ppc
CROSS_COMPILE ?= ppc_74xx-

- 6) Build Kernel (using ELDK make)
 - ppc-linux-make mrproper
 - ppc-linux-make menuconfig (*text based screen menus*)
 - Load Default Configuration (Load an Alternate Configuration File)
./arch/ppc/configs/mvme5500_defconfig
 - Set kernel boot command string (Platform Options)
See: Motorola README file.
See: <linuxTop>/Documentation/nfsroot.txt
 - ppc-linux-make zImage.initrd *<compressed kernel image>*



Installation and Implementation

Embedded Linux Kernel Build (cont)

Example: CONFIG_CMDLINE="console=ttyS0,9600 root=/dev/nfs rw
nfsroot=/local/eldk/eldk/ppc_74xx
ip=164.54.8.189:164.54.8.137:164.54.8.1:255.255.252.0:iocxxx:eth0:off
vme=vme_slotnum=1"

Goal: One kernel built for all targets.

nfsroot can use RARP or BOOTP to fill the client IP and name from the server.

- 7) Copy kernel to tftp server: (required for MOTLoad's netboot util).
/local/linux/linux-2.6.14/arch/ppc/boot/images/zImage.initrd.pplus

Uncompress Kernel Size:

1676763 zImage.mv5500 (1.67Mb)



Installation and Implementation

ELDK: Root file system setup (*must be done as root*)

1) Make root:/dev directory

```
> cd /local/eldk/eldk/ppc_74xx/dev  
> sudo /local/eldk/distribution/ELDK_MAKEDEV
```

2) Change files to be owned by **root**

```
> cd /local/eldk  
> sudo /local/eldk/distribution/ELDK_FIXOWNER
```

1) Create sysfs (/sys) (MVME hardware access)

```
> mkdir (/local/elkd/elkd/ppc_74xx)/sys  
Add mount point to (/local/elkd/elkd/ppc_74xx)/etc/fstab  
none /sys sysfs defaults 0 0
```





Installation and Implementation

Export IOC's root file system on NFS Server

- Add Linux IOC to server's export list
 - File: /etc/exports
/local/eldk iocLinux(rw,no_root_squash,secure)
 - Command: *sudo exportfs -a*

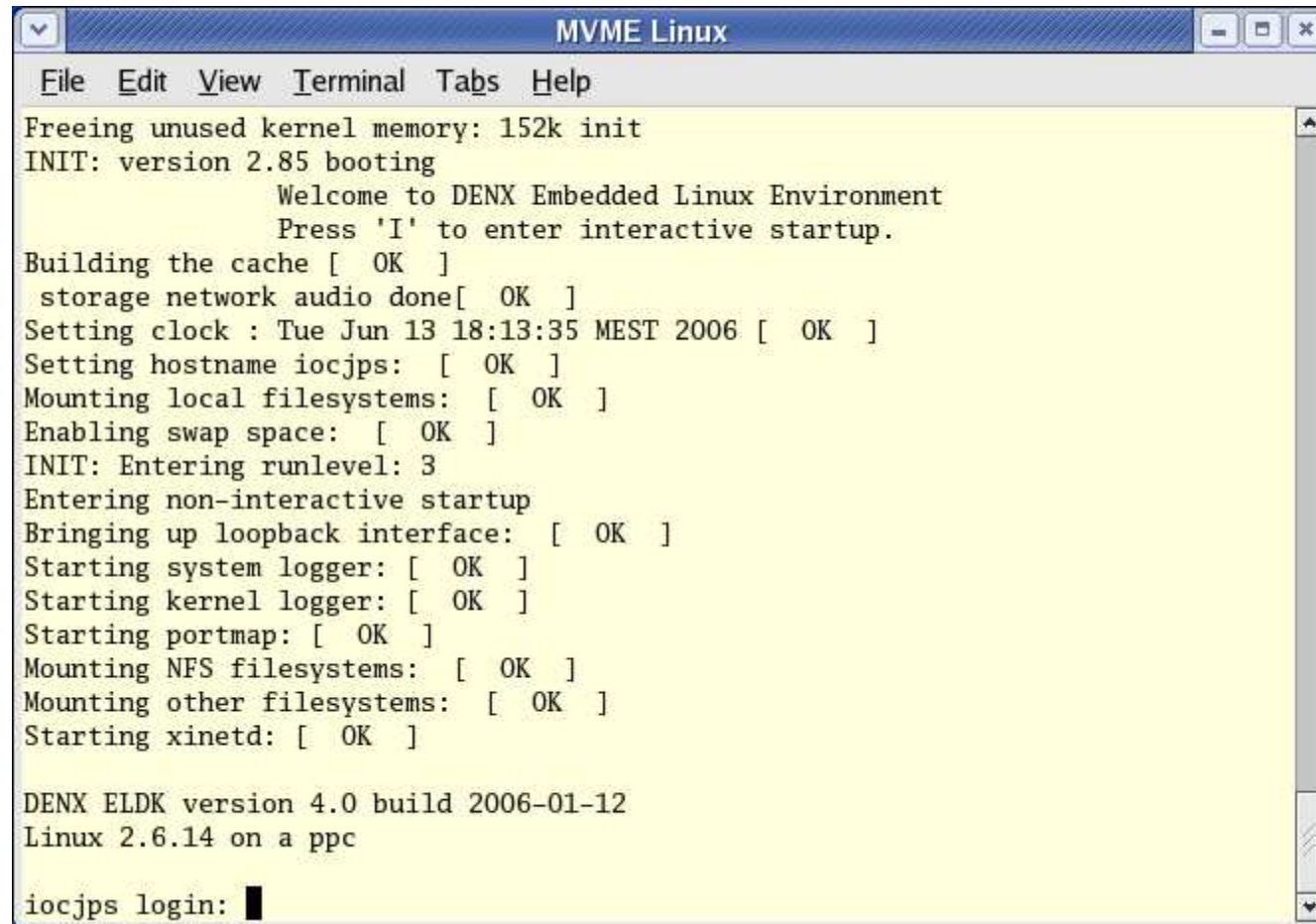
MVME network boot of linux kernel (w/autoboot)

- Connect to MVME thru serial port

```
MVME5500> gevInit
MVME5500> gevEdit mot-script-boot
netBoot -d/dev/enet1 -c164.54.8.189 -s164.54.8.103 -e400 -f
vxworks/MCGLINUX/zImage.mv5500
<cr>
```



Linux Runtime



```
File Edit View Terminal Tabs Help
Freeing unused kernel memory: 152k init
INIT: version 2.85 booting
      Welcome to DENX Embedded Linux Environment
      Press 'I' to enter interactive startup.
Building the cache [ OK ]
  storage network audio done[ OK ]
Setting clock : Tue Jun 13 18:13:35 MEST 2006 [ OK ]
Setting hostname iocjps: [ OK ]
Mounting local filesystems: [ OK ]
Enabling swap space: [ OK ]
INIT: Entering runlevel: 3
Entering non-interactive startup
Bringing up loopback interface: [ OK ]
Starting system logger: [ OK ]
Starting kernel logger: [ OK ]
Starting portmap: [ OK ]
Mounting NFS filesystems: [ OK ]
Mounting other filesystems: [ OK ]
Starting xinetd: [ OK ]

DENX ELDK version 4.0 build 2006-01-12
Linux 2.6.14 on a ppc

iocjps login: █
```


Linux Runtime

```
bash-3.00$ free
              total        used        free      shared    buffers     cached
Mem:          514844        54868       459976          0          0        13216
-/+ buffers/cache:  41652       473192
Swap:           0           0           0

bash-3.00$ cd /
bash-3.00$ ls
bin  etc  images  local  opt  root  sys  usr
dev  home  lib  mnt  proc  sbin  tmp  var

bash-3.00$ cd dev
bash-3.00$ ls vme*
vne_ctl  vne_lm0  vne_n2  vne_n5  vne_regs  vne_s1  vne_s4  vne_s7
vne_dna0  vne_n0  vne_n3  vne_n6  vne_rm0  vne_s2  vne_s5
vne_dna1  vne_n1  vne_n4  vne_n7  vne_s0  vne_s3  vne_s6

bash-3.00$
```

Linux Runtime

```
MVME Linux
File Edit View Terminal Tabs Help
top - 18:45:29 up 32 min, 1 user, load average: 0.03, 0.01, 0.00
Tasks: 22 total, 1 running, 21 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0% us, 0.0% sy, 0.0% ni, 100.0% id, 0.0% wa, 0.0% hi, 0.0% si
Mem: 514844k total, 55488k used, 459356k free, 0k buffers
Swap: 0k total, 0k used, 0k free, 13436k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	16	0	1632	592	520	S	0.0	0.1	0:11.29	init
2	root	34	19	0	0	0	S	0.0	0.0	0:00.00	ksoftirqd/0
3	root	RT	0	0	0	0	S	0.0	0.0	0:00.04	watchdog/0
4	root	10	-5	0	0	0	S	0.0	0.0	0:00.08	events/0
5	root	13	-5	0	0	0	S	0.0	0.0	0:00.01	khelper
6	root	10	-5	0	0	0	S	0.0	0.0	0:00.00	kthread
15	root	16	-5	0	0	0	S	0.0	0.0	0:00.00	kblockd/0
59	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pdflush
60	root	15	0	0	0	0	S	0.0	0.0	0:00.00	pdflush
62	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	aio/0
61	root	24	0	0	0	0	S	0.0	0.0	0:00.00	kswapd0
702	root	18	-5	0	0	0	S	0.0	0.0	0:00.00	ata/0
706	root	25	0	0	0	0	S	0.0	0.0	0:00.00	mtdblockd
757	root	10	-5	0	0	0	S	0.0	0.0	0:00.01	rpciod/0
930	root	16	0	2240	768	628	S	0.0	0.1	0:00.02	syslogd
932	root	17	0	1636	404	328	S	0.0	0.1	0:00.00	klogd
954	bin	18	0	2088	704	580	S	0.0	0.1	0:00.00	portmap



EPICS Target

linuxELDK-ppc_74xx

- Create new EPICS target architecture (**use vxWorks as a model**)
 - See <epics base>/documentation/README.1st
- EPICS Configuration Files
 - <top>/configuration/CONFIG_SITE
 - *CROSS_COMPILER_TARGET_ARCHS = linuxELDK-ppc_74xx*
 - configure/os/
 - *CONFIG.Common.linuxELDK-ppc_74xx*
 - *CONFIG.linux-x86.linuxELDK-ppc_74xx*
 - *CONFIG_SITE.Common.linuxELDK-ppc_74xx*
- Use ELDK make (*with proper environment variables*)
 - ppc-linux-make





EPICS Results

- EPICS *linuxELDK-ppc_74xx* cross-compiler target
- EPICS Base - Built and Running (no problems)
 - EPICS 3.14.8.2
 - *exampleApp*
 - synApps 5.2
 - *Excluding (gensub, dxp, ccd, xxx)*
- iocxxx running synApps
softIOC configuration





Remaining Work

- Figure out the Motorola VME interface
 - Memory windows and interrupt handling
- Benchmark MVME Linux performance
- Attempt a port of a vxWorks driver to Linux (OMS VME58).
- Investigate uCLinux (www.uclinux.org) kernel patches to get around the MMU (get direct access to VME memory and interrupts).



Future of Embedded Linux

