UNIVERSITÀ DI BRESCIA
FACOLTÀ DI INGEGNERIA
Dipartimento di Elettronica per l'Automazione

Laboratorio di Robotica Avanzata
Advanced Robotics Laboratory

Corso di Robotica
(Prof. Riccardo Cassinis)

Configurazione robusta del S.O. Linux con ramdisk ed attivazione delle interfacce di rete

Elaborato di esame di: **Roberto Franchina, Vittorio Gregorelli**

Consegnato il: **15 Gennaio 2002**

Sommario

Il nostro lavoro è stato suddiviso in due parti. La prima di esse permette al calcolatore di caricare, all'avvio, il sistema operativo (una versione "leggera" di Linux che verrà meglio descritta nel seguito della relazione) ed il filesystem (tutte le directory ed i file necessari al funzionamento della macchina) in un ramdisk (col termine ramdisk si definisce una porzione di memoria RAM che viene vista come un dispositivo di memorizzazione di massa). A quel punto il sistema lavora correttamente come se risiedesse fisicamente sul disco fisso. L'hard disk a stato solido installato sulla piastra madre (ci riferiremo a questo dispositivo utilizzando sempre il termine "flashdisk") viene visto dal sistema come un supporto di memorizzazione esterno e montato (ogni dispositivo di memorizzazione, in Linux, per essere utilizzato deve prima venire montato in una directory), sempre in fase di avvio, in un'apposita directory.

La seconda parte del nostro lavoro consisteva nell'abilitare la comunicazione tra il nostro calcolatore (MARMOT) e gli altri calcolatori del laboratorio. Tutto questo è stato possibile configurando un'apposita scheda PCMCIA che permette la comunicazione attraverso una wavelan (una rete LAN ad onde radio).

1. Introduzione

Questo elaborato è stato svolto nel Laboratorio di Robotica Avanzata (LRA) della facoltà di Ingegneria dell'Università degli Studi di Brescia nel periodo che va dalla metà di Settembre 2001 alla metà di Dicembre 2001.

Il nostro lavoro rientra nell'ambito della costruzione del robot M.A.R.M.O.T. (acronimo di Mobile Advanced Robot for Multiple Office Tasks). Come noi, infatti, altri gruppi di studenti sviluppano elaborati che puntano l'attenzioni sui diversi aspetti che la costruzione di un robot richiede (movimentazione, comunicazione, gestione dei sensori etc…).

2. Il problema affrontato

MARMOT, come ogni dispositivo privo di cavo d'alimentazione, viene alimentato tramite una batteria (12 volt). Può succedere che essa si scarichi, causando un improvviso calo di tensione che porta allo spegnimento del calcolatore montato a bordo del robot. Quando un sistema Linux si spegne improvvisamente, senza cioè

aver eseguito l'opportuna procedura di shutdown, il filesystem si può corrompere e i file possono diventare illeggibili. Questo succede perché Linux, durante il suo funzionamento, non scrive continuamente su disco, ma utilizza, ad intervalli, la cache del disco fisso. Un comportamento del genere migliora moltissimo le prestazioni, ma sta anche a significare che in caso d'improvviso spegnimento la cache può contenere molti dati e quello che si trova sul disco può non essere un filesystem che funziona perfettamente (perché solo alcune cose vi sono state scritte).

Il nostro compito è appunto quello di configurare un sistema Linux affinché sia robusto agli improvvisi spegnimenti.

In secondo luogo si vuole che MARMOT possa comunicare con gli altri calcolatori del laboratorio senza dover ricorrere all'utilizzo di cavi di collegamento che sarebbero sicuramente d'intralcio. Per questo motivo a bordo del robot viene installata una scheda PCMCIA che permetta l'utilizzo di una Wavelan (o Wireless Lan) nel laboratorio. Nostro compito è appunto quello di installare tutto il software necessario per il funzionamento della suddetta scheda di comunicazione.

3. La soluzione adottata

Di seguito vengono elencati i passi seguiti per la soluzione del problema. Nel seguito della relazione questi argomenti verranno meglio descritti ed approfonditi.

- Formattazione della flash ed installazione della distribuzione White Dwarf.
- Installazione sull'hard disk di una Mandrake 8 completa.
- Compilazione di una versione del kernel 2.2.20
- Installazione dei driver della PCMCIA sull'hard disk (Mandrake)
- Copia dei file creati al passo precedente nella flash
- Configurazione e verifica del funzionamento della PCMCIA
- Creazione di un'immagine compressa del filesystem della flash
- Configurazione del ramdisk

4. Modalità operative

Descriveremo ora nel dettaglio quello che abbiamo fatto per arrivare alla soluzione del problema.

4.1. Installazione delle distribuzioni White Dwarf v1.11 e Mandrake 8

Dovendo lavorare con una flash di soli 30 Mb siamo stati costretti ad installare una distribuzione Linux molto compatta. E' stata scelta la versione White Dwarf v.1.11 (si consiglia di consultare il sito www.whitedwarflinux.org per maggiori dettagli). White Dwarf è una distribuzione consigliata per la scheda Jumptec PC-104 della quale è dotato il nostro robot (www.jumptec.com). Questa piccola distribuzione di Linux può essere installata sia da CD sia da Internet via FTP. L' installazione da CD richiede un floppy di boot, mentre l'installazione da rete necessita, oltre al boot-disk, anche di un disco di root. Quando abbiamo iniziato il nostro lavoro, sul flashdisk era già installata la distribuzione White Dwarf v.1.11, ma è stato necessario reinstallarla più volte a causa di nostri errori e distrazioni (una volta abbiamo, ad esempio, cancellato la directory /etc, un'altra volta abbiamo eliminato involontariamente l'intero contenuto della flashdisk…). La procedura di installazione da CD da noi seguita è la seguente: si avvia il PC con il floppy di boot inserito nell'apposito drive (si consiglia di fare molta attenzione nel maneggiare questo dispositivo perché se si dovesse rompere il cavetto che lo collega alla piastra madre i tempi di sostituzione sarebbero davvero lunghi, in quanto sembra che ce siano solo 2 esemplari in tutta la provincia di BS…) e quando viene richiesto si seleziona la voce "clean" che inizia la formattazione della flash. Terminata questa fase si inizia ad installare la distribuzione selezionando gli opportuni programmi ed escludendo quelli che non servono (in seguito verrà presentato un elenco di quello che effettivamente bisogna installare).

La dimensione di questa distribuzione può variare da 5 ai 50 Mb a seconda dei pacchetti che si vogliono installare. Nel nostro caso occupava circa 20 Mb. Di seguito presentiamo un elenco di tutti i pacchetti disponibili con questa distribuzione ed una brevissima descrizione del loro contenuto. I pacchetti preceduti da un asterisco sono quelli noi che abbiamo installato.

* **aaa_base** This package installs the basic file structure for wd linux
* **bash** Standard full featured shell
* **devs** Installs all of the necessary device files
* **e2fsbn** Tools to create and repair filesystems
* **elflibs** ELF libraries. libc5 based
* **etc** Files to setup the etc directory
* **fileutls** Utilities to handle file management
* **kernel** 2.2.14 kernel and modules
* **ldso** Dynamic linker and loader
* **lilo** Linux boot loader
* **modutils** Tools to allow the use of kernel module
* **procps** Tools to allow the use of the /proc filesystem

*** shadow** Tools to handle users and groups
*** sysvinit** System V style init and related items
*** zoneinfo** Timezone data
bin Binary utilities including: DOS utilities, crontab, time, which, crond, apmd.
*** elvis** vi text editor
*** find** GNU find utilities, includes find, locate, and updatedb
*** gzip** GNU zip tools
*** sh_utils** date,echo,false,pwd,stty,true,env,nohup,sleep,chroot
*** tar** GNU tar archiving tool
txtutils Text tools: cat, cut, head, cksum, comm, csplit, expand, fmt, fold, join, md5sum, nl, od, paste, pr, sort, split, sum, tac, tail, tr, unexpand, uniq, wc.
*** util** More utilities: arch, hostname, setterm, dmesg, getopt, more, umount, mount, agetty, hwclock, rdev, clock, kbdrate, mkswap, swapon, fdisk, losetup, mkfs, setserial, update
apache A stripped down version of apache(v 1.3.11)
*** ash_sh** Very small shell to replace the larger standard shells like csh, sh, tcsh or bash
bzip Compression utility
dev gcc development system
diff File compare utilities
getty Alternate getty tools for serial/console handling
gpm mouse driver
grep Tool to allow regular expressions from the command line
kbd Keyboard configuration utilities
less Less is more only better
lynx Text based www browser. Setup to go to http://www.emjembedded.com/linux/ by default
minicom Full-featured terminal emulation package
perl The perl scripting language
pkgtool Slackware's package tools
ppp ppp drivers and scripts
sysklogd syslog and klog system message loggers
*** tcpip** Basic TCP/ip tools: ftp, telnet, ping, hostname, ftpd, telnetd, ifconfig, route, whois, traceroute, tcpd, inetd
vim Great replacement for vi
wget Tool for automating ftp or http file transfer
kernel_source
ncurses
perl_libs

La distribuzione Mandrake 8 è stata installata sull'hard disk montandolo su un nostro calcolatore. Non è infatti possibile collegare contemporaneamente un lettore cd-rom ed un hard disk al calcolatore che MARMOT avrà a bordo in quanto si ha a disposizione un solo controller IDE (se si toglie la flash dal primario il calcolatore non si avvia).

4.2. La compilazione del kernel

Il Kernel è il cuore del sistema operativo, cioè la sua parte fondamentale, che ne governa e controlla il sistema. Esso, in pratica, fornisce al resto del sistema operativo, e alle applicazioni degli utenti, tutti i servizi relativi alla gestione dell'hardware. Il kernel viene caricato in memoria al momento dell'avvio di Linux, e vi rimane per tutto il tempo in cui Linux è attivo: è memorizzato fisicamente sull'hard disk sotto forma di file compresso, che si autodecomprime al momento del suo caricamento in

memoria. LILO non fa altro che accedere direttamente a tale file (il cui nome è inserito nel file di configurazione lilo.conf) ed iniziare la sua decompressione.
Il kernel di Linux provvede anche alla gestione dei vari file system, del networking, dell'audio, dei controller dei dischi, del processore, ma anche dei set di caratteri e della tastiera. Inoltre il kernel gestisce il caricamento dei moduli. Un modulo è una porzione del kernel che si occupa di gestire uno solo degli aspetti del sistema operativo, e viene caricato in memoria solo quando necessario, cioè al momento del suo utilizzo. E' possibile scegliere, almeno in parte, quali parti del kernel utilizzare, rendendolo più versatile e sicuramente ottimizzato al meglio. La gestione dei moduli (cioè il loro caricamento e il loro scarico) è affidata al kernel stesso.

Compilare il kernel significa essenzialmente prendere i file sorgenti di esso (compresi in tutte le distribuzioni e aggiornabili poi tramite Internet), impostare il file che contiene la configurazione desiderata, e avviare il compilatore gcc (che quindi deve essere installato); il risultato è un nuovo file contenente il kernel (diverso da quello utilizzato fino a quel momento) utilizzabile per l'avvio (cioè un nuovo kernel, di fatto). E' bene notare che è possibile conservare sul disco quanti file (e quindi quante versioni del kernel) si vuole, e ognuno di essi è utilizzabile liberamente in qualsiasi momento (per esempio tramite Lilo, che può associare un nome di avvio ad ogni diversa versione).

La ricompilazione del kernel prevede 4 passi fondamentali a cui è necessario attenersi, dopo essersi connessi al sistema come utente root.

Prima di avviare tale procedura, però, è necessario che i file sorgenti del kernel siano installati correttamente sul disco. Noi abbiamo scaricato i file sorgenti del kernel versione 2.2.20 dal sito **www.kernel.org** nel formato **tar.gz** e decompressi nella directory **/usr/src**; la decompressione creerà la sottodirectory chiamata **/linux-2.2.20**. In **/usr/src/** bisogna creare un link simbolico a tale directory chiamato **linux** usando il comando :

ln –s /usr/src/linux-2.2.20 linux

Il primo passo da compiere per la compilazione del kernel consiste nell'aggiornamento (o nella creazione, se è la prima volta che lo si fa) del file di configurazione del kernel. Per eseguire tale operazione digitare la seguente istruzione:

make menuconfig
lancia il programma che permette all'utente di personalizzare il kernel. Nel seguito spiegheremo meglio questo passo.

I successivi tre comandi possono essere digitati insieme o separatamente:

make dep

Il comando "make dep" controlla che tutti i file necessari per la compilazione siano presenti, creando poi delle dipendenze tra questi file

make clean

il comando "make clean" elimina i file temporanei che potrebbero distrurbare il processo di compilazione

make bzImage

avvia la compilazione e crea l'eseguibile vero e proprio.

Il processo di compilazione sul PC di MARMOT (Pentium 166Mhz) richiede un tempo di circa 40 minuti. Al termine di tale operazione il nuovo kernel si troverà nella directory **/usr/src/linux/arch/i386/boot** con il nome **bzImage**. Tale file va copiato nella root directory ricordandosi di rinominarlo per evitare di sovrascrivere un kernel già esistente. Successivamente si deve procedere all'aggiornamento del LILO editando il file di configurazione **lilo.conf** presente nella directory **/etc** ed aggiungendo ad esso una nuova configurazione (per maggiori informazioni scaricare "LILO mini-HOWTO" dal sito www.linuxdoc.org). Per rendere effettive queste modifiche bisogna digitare il comando **lilo** che scrive nel master boot record la nuova configurazione del LILO.
Ecco una parte del file di configurazione generato dal menuconfig. Per motivi di spazio riportiamo solo la parte che abbiamo effettivamente impostato nella configurazione, tralasciando tutto il resto. Nella documentazione allegata alla relazione è comunque possibile trovare il file completo questomipiace (il nome del file non è molto ortodosso, ma è stato dettato dall'euforia scaturita quando ci siamo accorti che la configurazione del Kernel era appropriata) generato dal menuconfig durante la compilazione del kernel 2.2.20. La compilazione di un Kernel, è un'operazione che può rivelarsi (come nel nostro caso) lunga, ripetitiva e molto frustrante, in quanto ad ogni modifica apportata nel menuconfig segue la sopracitata procedura di compilazione. Più volte, nel testare il Kernel appena generato, la macchina si bloccava durante la fase di avvio, a causa di errate o mancanti scelte dei moduli. In questi casi, si rendeva necessario riavviare la macchina digitando, al prompt del LILO, la voce "mandrake" che richiamava una versione del Kernel funzionante (nello specifico, il Kernel 2.2.18).
La lettera "y" sta a significare che l'apposito supporto viene incluso direttamente nel kernel, al contrario la lettera "m" permette di caricare tale supporto nel kernel come modulo separato. Si noti che gli unici supporti settati con la "m" sono **CONFIG_BLK_DEV_IDECD e CONFIG_BLK_DEV_IDEFLOPPY.**

```
# Code maturity level options is not set

# Processor type and features
CONFIG_M586TSC=y
CONFIG_X86_WP_WORKS_OK=y
CONFIG_X86_INVLPG=y
CONFIG_X86_BSWAP=y
CONFIG_X86_POPAD_OK=y
CONFIG_X86_TSC=y
CONFIG_1GB=y

# Loadable module support
CONFIG_MODULES=y
CONFIG_MODVERSIONS=y
CONFIG_KMOD=y

# General setup
CONFIG_NET=y
```

```
CONFIG_PCI=y
CONFIG_PCI_GOANY=y
CONFIG_PCI_BIOS=y
CONFIG_PCI_DIRECT=y
CONFIG_PCI_OLD_PROC=y
CONFIG_SYSVIPC=y
CONFIG_BINFMT_AOUT=y
CONFIG_BINFMT_ELF=y
CONFIG_BINFMT_MISC=y
CONFIG_PARPORT=y
CONFIG_PARPORT_PC=y

# Plug and Play support is not set

# Block devices
CONFIG_BLK_DEV_FD=y
CONFIG_BLK_DEV_IDE=y
CONFIG_BLK_DEV_IDEDISK=y
CONFIG_BLK_DEV_IDECD=m
CONFIG_BLK_DEV_IDEFLOPPY=m
CONFIG_BLK_DEV_IDEPCI=y
CONFIG_BLK_DEV_IDEDMA=y
CONFIG_IDEDMA_AUTO=y
CONFIG_BLK_DEV_LOOP=y
CONFIG_BLK_DEV_NBD=y
CONFIG_BLK_DEV_RAM=y
CONFIG_BLK_DEV_RAM_SIZE=16384
CONFIG_BLK_DEV_INITRD=y
CONFIG_PARIDE_PARPORT=y

# Networking options
CONFIG_PACKET=y
CONFIG_NETLINK=y
CONFIG_RTNETLINK=y
CONFIG_UNIX=y
CONFIG_INET=y
CONFIG_IP_MULTICAST=y
CONFIG_IP_ADVANCED_ROUTER=y
CONFIG_RTNETLINK=y
CONFIG_NETLINK=y
CONFIG_SKB_LARGE=y

# Telephony Support is not set

# SCSI support is not set

# CONFIG_SCSI is not set

# I2O device support is not set

# Network device support
CONFIG_NETDEVICES=y

# ARCnet devices
CONFIG_DUMMY=y

# Ethernet (10 or 100Mbit)
CONFIG_NET_ETHERNET=y
CONFIG_NET_EISA=y
```

```
CONFIG_EEXPRESS_PRO100=y

# Ethernet (1000 Mbit) is not set

# Token ring devices is not set

# Wan interfaces is not set

# Amateur Radio support is not set

# IrDA (infrared) support is not set

# ISDN subsystem is not set

# Old CD-ROM drivers (not SCSI, not IDE) is not set

# Character devices
CONFIG_VT=y
CONFIG_VT_CONSOLE=y
CONFIG_SERIAL=y

# Joysticks is not set

# Video For Linux is not set

# Ftape, the floppy tape device driver is not set

# USB support
CONFIG_USB=y

# Filesystems
CONFIG_AUTOFS_FS=y
CONFIG_ISO9660_FS=y
CONFIG_PROC_FS=y
CONFIG_EXT2_FS=y
CONFIG_UFS_FS=y

# Network File Systems
CONFIG_NFS_FS=y
CONFIG_SUNRPC=y
CONFIG_LOCKD=y
CONFIG_SMB_FS=y

# Partition Types
CONFIG_NLS=y

# Native Language Support
CONFIG_NLS_DEFAULT="cp437"
CONFIG_NLS_CODEPAGE_437=y
CONFIG_NLS_CODEPAGE_850=y
CONFIG_NLS_ISO8859_1=y

# Console drivers
CONFIG_VGA_CONSOLE=y

# Sound is not set

# Kernel hacking
CONFIG_MAGIC_SYSRQ=y
```

4.3. Il file di configurazione lilo.conf

LILO è il programma memorizzato nel master boot record dell'hard disk primario che si occupa di caricare all'accensione del calcolatore il sistema operativo desiderato (LILO significa appunto LInux LOader).

Per la sua configurazione bisogna editare un apposito file chiamato **lilo.conf** presente nella directory **/etc**.

Di seguito mostriamo il lilo.conf da noi scritto ed usato durante lo svolgimento di questo lavoro. Le linee precedute dal carattere '#' sono semplicemente righe di commento.

```
# Franchina Roberto, Gregorelli Vittorio       Dicembre 2001
# Questo e' il lilo.conf utilizzato durante lo sviluppo del ramdisk e l'installazione
# della PCMCIA

# generated by 'liloconfig'
#
# Start LILO global section
boot = /dev/hda
#compact        # faster, but won't work on all systems.
delay = 5
prompt
vga = normal    # force sane state
# End LILO global section
# Linux bootable partition config begins
image = /vmlinuz
  root = /dev/hda1
  label = linux
  read-only # Non-UMSDOS filesystems should be mounted read-only for checking
image = /boot/bzIm-2.2.20
  root = /dev/hdb6
  label = kernel-2.2.20
  read-only
image = /boot/vmlinuz-2.4.3-20mdk
  root = /dev/hdb6
  label = kernel-2.4.3
  read-only
image = /boot/bzIm-2.2.20
  initrd = /boot/filesystem_image.gz
  root = /dev/ram0
  label = ramdisk
  read-only
image = /boot/bzIm-2.2.20
  root = /dev/hda1
  label = 2.2.20-flash
  read-only
image = /vmlinuz
  root = /dev/hdb6
  label = mandrake
  read-only
# Linux bootable partition config ends
```

Quando il calcolatore viene avviato l'utente deve scegliere quale versione del sistema operativo caricare dal seguente elenco:

- **linux**
- **kernel-2.2.20**
- **kernel-2.4.3**
- **ramdisk**
- **2.2.20-flash**
- **mandrake**

Digitando **linux**, come si può vedere dal file di configurazione, viene caricato il kernel originale della flash (quello della White Dwarf) con il filesystem della flash (e quindi ancora quello della distribuzione White Dwarf 1.11).

Scegliendo **kernel-2.2.20** viene invece lanciato il kernel 2.2.20 che noi abbiamo ricompilato con il filesystem della distribuzione Mandrake 8 presente sull'hard disk (hdb6).

Se si digita **2.2.20-flash** viene caricato ancora il kernel 2.2.20 da noi ricompilato, ma con il filesystem della flash.

Se si sceglie invece **kernel-2.4.3** viene caricato il kernel ed il filesystem della Mandrake 8 presente su hard disk.

Scegliendo **ramdisk** il programma initrd carica il kernel 2.2.20 e scompatta in un ramdisk da 16384 Kb il filesystem compresso presente nel file **/boot/zippettonepaolo.gz**. Da quel momento in poi questo ramdisk viene considerato come il dispositivo di root e la flash viene montata nella directory **/ram** e vista come un qualsiasi supporto di memorizzazione.

Digitando **2.2.20-flash** si avvia ancora il kernel 2.2.20 ma con il filesystem della flash.

Infine se si sceglie **mandrake** viene caricato il kernel della White Dwarf con il filesystem della Mandrake 8. Durante lo sviluppo del nostro lavoro abbiamo utilizzato un disco rigido che ora non è più disponibile, e senza il quale le voci **kernel-2.2.20**, **Kernel-2.4.3** e **mandrake** non possono essere richiamate.

4.4. Installazione del software per la scheda PCMCIA

Di seguito mostriamo i passi che abbiamo seguito per installare i driver ed il supporto della scheda PCMCIA. Per una maggiore documentazione sull'argomento si rimanda

alla lettura del file **PCMCIA-HOWTO** contenuto assieme ai driver della scheda PCMCIA.

- Scaricare l'ultima versione dei driver **pcmcia-cs** da www.sourceforge.net;
- Scompattarli in **/usr/src**;
- Entrare nella directory **./pcmcia** che viene creata dalla scompattazione precedente
- Eseguire in successione i tre comandi **make config, make all, make install.**

Dopo aver lanciato **make config** viene chiesto all'utente di specificare la directory contenente la sorgente del kernel, nel nostro caso è la directory alla quale punta il link **/usr/src/linux.**

Successivamente viene chiesto di inserire la directory in cui verranno installati i nuovi moduli del kernel, che nel nostro caso è **/lib/modules**.

L'esecuzione di **make all** seguita da **make install** costruisce ed installa i moduli del kernel ed i programmi di utilità. Tali moduli vengono installati dentro **/lib/modules/2.2.20/pcmcia/**. I programmi cardmgr e cardctl invece vengono messi in **/sbin**.

I file di configurazione vengono installati nella directory **/etc/pcmcia**.

Lavorando con la distribuzione installata sulla flash (Whitedwarf 1.11) non si ha a disposizione il programma **make**. Questo ci ha costretti ad installare il supporto per la scheda PCMCIA sull'hard disk dove avevamo installato una Mandrake 8 (eseguendo i passi sopra citati). Durante tale installazione abbiamo trascritto tutti i file che i comandi **make all** e **make install** andavano a creare o a "sporcare". Al termine dell'installazione, manualmente, abbiamo copiato tali file nelle appropriate directory della flash.

4.5. La creazione del ramdisk

Arrivati a questo punto la distribuzione Linux presente sulla flash dovrebbe permettere il corretto funzionamento della scheda PCMCIA. Si consiglia, prima di proseguire, di effettuare tutte le prove necessarie per assicurarsi che tale dispositivo sia configurato correttamente (provare cioè a far comunicare questa scheda con un'altra installata su un altro calcolatore). Se tutto funziona correttamente si può passare alla creazione del filesystem che verrà caricato nel ramdisk all'avvio del calcolatore. Questo passo è abbastanza semplice, basta infatti copiare il contenuto della flash dentro un'opportuna directory dell'hard disk (sulla flash non ci sarebbe spazio), nel nostro caso abbiamo chiamato tale directory **/filesystempaolo1**. Fare attenzione a non copiare la directory **/HD** nella quale viene montato l'hard disk.

Ora formattiamo un ramdisk da 16384 Kb con il comando:

mke2fs /dev/ram0

Montiamo questo dispositivo nella directory **/ram** con il comando:

mount /dev/ram0 /ram

e copiamo tutto il contenuto di **/filesystempaolo1** dentro il ramdisk (cioè dentro **/ram**) con il comando:

cp -af /filesystempaolo1/* /ram

Adesso bisogna prendere tutto quello che è stato copiato nel ramdisk e metterlo in un unico file che chiamiamo **zippettone** scrivendo:

dd if=/dev/ram0 bs=1k count=16384 of=/boot/filesystem_image

questo file viene poi compresso col comando (ovviamente bisogna posizionarsi prima nella directory **/boot** dell'hard disk):

gzip filesystem_image

che restituisce il file **filesystem_image.gz**

Questo file deve essere copiato nella directory **/boot** della flash (ricordo che la flash è montata nella directory **/flashdisk**) scrivendo:

cp filesystem_image.gz /flashdisk/boot

Ora sulla flash sono presenti tutti e due i file che servono per il funzionamento del ramdisk, ossia il kernel opportunamente compilato con il supporto per il ramdisk (**kernel-2.2.20**) e l'immagine "gzippata" del filesystem funzionante della flash (**zippettone.gz**).

Non resta che configurare il LILO (LInux LOader) per fare in modo che avvii il ramdisk quando viene digitata la voce **ramdisk**, aggiungendo al file **lilo.conf** che si trova in **/etc** le seguenti righe:

image=kernel-2.2.20

initrd=/boot/filesystem_image.gz

root=dev/ram0

label=ramdisk

read-only

Per maggiori dettagli riguardanti il funzionamento del programma **initrd** ed alla gestione di un ramdisk si rimanda alla lettura dei file **initrd.txt** e **ramdisk.txt** forniti come documentazione allegata a questa relazione.

All'avvio del ramdisk, la flash viene montata nella directory **/flash**

All'interno della directory **/flash/boot** è presente il file **init.sh**; questo file è uno script nel quale possono essere inseriti comandi che verranno eseguiti all'avvio.

Attualmente, all'interno di **init.sh** è presente il comando:

ifconfig eth1 up

che attiva i servizi della wavelan.

L' indirizzo IP della wavelan è:

192.168.103.2

mentre l'indirizzo Ethernet è:

192.167.22.72

4.6. Componenti necessari

Ecco un'immagine della nostra "postazione di lavoro" :

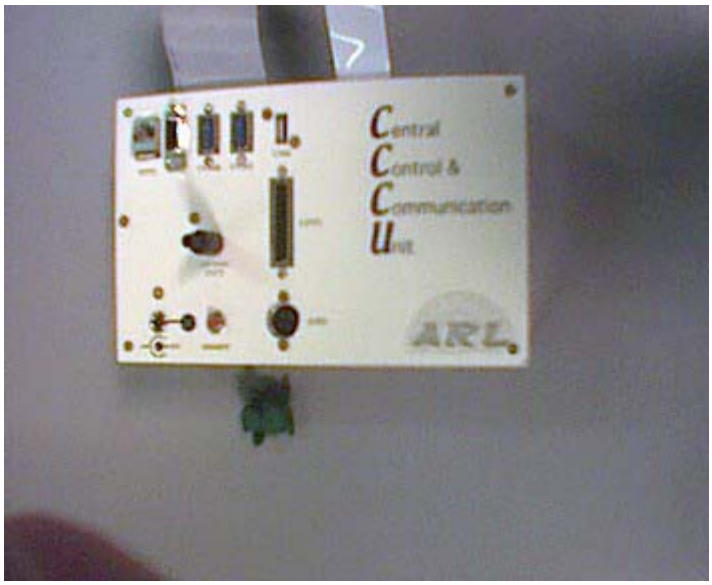
Il nostro lavoro è stato sviluppato sul calcolatore che poi verrà montato a bordo di MARMOT. Ovviamente per poter lavorare abbiamo dovuto collegare ad esso un monitor, una tastiera, un alimentatore, un lettore floppy, hard disk ed un lettore cd-rom.

L'hard disk ha una capacità di 4,3 Gb ed è stato usato per avere a disposizione una distribuzione completa di Linux, dotata cioè di tutti gli indispensabili programmi che servono sia per la compilazione del kernel che per la gestione dei file.

Il lettore floppy serve unicamente per poter leggere il dischetto di boot necessario per l'installazione della distribuzione White Dwarf sulla flash.

Il lettore cd-rom, usato anch'esso per installare la White Dwarf, viene montato quando serve al posto dell'hard disk, perché il calcolatore permette il collegamento di un solo dispositivo IDE slave.

Il calcolatore è totalmente contenuto nella scatola mostrata nell'immagine seguente e denominata Unità Centrale di Controllo e Comunicazione (C.C.C.U.).



All'interno della C.C.C.U. vi sono ovviamente il processore Pentium 166 MMX, la relativa scheda madre (dotata di 2 uscite seriali, una parallela ed una presa USB), la memoria flash di 30 Mb, 64 Mb di memoria RAM, una scheda di rete Ethernet ed una scheda ISA che costituisce lo slot dentro il quale verrà infilata la scheda PCMCIA. La scheda CPU è una Mops-lcd6 della Jumptec; di seguito riportiamo le caratteristiche ed i dati tecnici sia della CPU che della scheda PCMCIA.

Il PC/104 – MOPSlcd6

Il MOPSlcd6 (Minimized Open PC System) con il suo processore Intel Pentium MMX a 166 MHz offre alte prestazioni ed affidabilità. Visto il basso consumo di potenza (meno di 9 watt) non è necessaria una ventola di raffreddamento – un

semplice dissipatore è più che sufficiente. La MOPSlcd6 è una scheda ad alta integrazione, che include le interfacce standard 2xRS232C, USB, LPT, Floppy and EIDE, Ethernet 10/100 MBit. Il socket SO-DIMM permette di montare sulla scheda fino a 64 MByte di SDRAM. Il socket inferiore viene usato come interfaccia JIPA (JUMPtec Intelligent Panel Adaptation) per la connessione a pannelli ultrapiatti a cristalli liquidi. Il controller grafico PCI C&T Graphic-Controller supporta monoCD, TFT e display STN con resolutioni superiori all'SXGA (1280x1024). I 2MB di memoria video permettono l'utilizzo di applicazioni con più di 16 million colors. Un chipDISK JUMPtec (una flash disk IDE-compatibile) con oltre 96 MByte può essere collegato direttamente sull'interfaccia dell'hard disk da 2.5" .

Dati tecnici

CPU: Intel Pentium MMX - 166, 266 MHz

DRAM: up to 64MB SDRAM-SO-DIMM-module

bootable flash harddisk – chipDISK: IDE flash drive module up to 48 MByte

10/100 BaseT ethernet interface

Floppy interface

EIDE interface with Ultra33 DMA mode

Printer interface (SPP/EPP/ECP)

2 x RS232

1 x USB

Keyboard interface

Onboard PCI C&T Graphic-Controller

Resolutions:

640x480 up to 16 MIO colors

800x600 up to 16 MIO colors

1024x768 up to 64K colors

1280x1024 up to 256 colors

LCD and CRT simultaneous

JIPA interface for LCD's

2 MByte video RAM

Setup in EEPROM (i.e. bootable without a battery)

Watchdog timer

Power fail

Realtime clock

Dimension: 96x90 mm (3.8x3.6")

Power consumption only 5V 1,8 A typ.

Temperature: operation: 0?to 60°C storage: -10?to 85°C

Max. humidity: operation: 10% to 90% storage: 5% to 95%

La scheda PCMCIA

L'immagine seguente mostra la scheda ISA che costituisce lo slot dentro il quale si può vedere la scheda PCMCIA (Orinoco, Lucent Technology).Vengono riportati anche i dati tecnici.

Dati tecnici

Frequency range: 2400-2483.5 MHz

Number of frequency channels: 4/11/13/14 (depending on the country)

Modulation: Direct Sequence Spread Spectrum (CCK, DQPSK, DBPSK)

Protocol: CSMA/CA (Collision Avoidance) with ACK

Interface: PC card with ISA, PCI adapters

Size: 117.8x53.95x8.7 mm (PC Card)

Output power: 15 dBm

Power supply(+5V) : PC Card Doze mode 9 mA, Receiver mode 185 mA, Transmit mode 285 mA, ISA adapter - 25mA.

Program compatibility: Novell Client 3.x & 4.x, Windows 95/98/NT/2000, Apple, Windows CE

Support for IEEE 802.11b HR standard

4.7. Problemi riscontrati

Il primo grosso problema che abbiamo dovuto affrontare, è stata la compilazione di un Kernel funzionante. Inizialmente compilavamo Kernel che in fase di avvio si bloccavano durante l'inizializzazione del ramdisk, in quanto, nel menuconfig includevamo il supporto per il ramdisk come modulo, e non lo inserivamo direttamente nel kernel. Il ramdisk, infatti, viene attivato prima della fase di caricamento dei moduli, e quindi il suo supporto non veniva lanciato.

Un altro problema si verificava all'avvio causando il blocco del computer e mostrando il messaggio d'errore a video `: unable to mount root filesystem on 03:46`; questo in quanto nel menuconfig (durante la compilazione del Kernel) inserivamo come moduli i servizi relativi ai dispositivi IDE. Anche in questo caso vanno inclusi direttamente nel Kernel.

Ogniqualvolta si debba installare del software, o si voglia ricompilare un Kernel, è necessario verificare che la data e l'ora siano correttamente impostate all'interno del BIOS. Se non c'è coerenza temporale nella data ed ora di creazione di alcuni file, il sistema può dare dei problemi. Ricordiamo che a bordo della piastra madre non c'è una batteria di alimentazione, e quando viene tolta l'alimentazione, data e ora vengono persi.

Se non si riesce ad accedere alla rete tramite la scheda Ethernet è possibile che all'interno del BIOS sia disabilitato il campo relativo alla porta di comunicazione.

Consigliamo di prestare particolare attenzione quando si fanno delle prove sul ramdisk. Dopo aver avviato il PC in "modalità" **ramdisk**, infatti, abbiamo provato a cancellare tutto il filesystem (convinti di non fare danni, in quanto ad ogni riavvio della macchina l'intero filesystem viene rigenerato), ma ci siamo dimenticati di smontare dalla directory **/flashdisk** l'hard disk a stato solido. Così facendo abbiamo eliminato, oltre al filesystem della RAM, tutto il contenuto della flash. Immaginate cosa sarebbe successo se fosse stato montato anche l'hard disk: il lavoro di mesi sarebbe andato perso…

Attualmente i servizi di route non sono ottimizzati, in quanto il login via ftp e telnet su eth1 è troppo lento.

Probabilmente serve modificare il file **rc.inet1** contenuto nella directory **/etc/rc.d/**

A login effettuato, la connessione non dà problemi.

Bibliografia

La documentazione che abbiamo utilizzato e che elenchiamo di seguito è scaricabile dal sito http://www.linuxdoc.org/

[1] Tom Fawcett: "The Linux Bootdisk HOWTO";

[2] Miroslav Skoric: "LILO mini-HOWTO";

[3] Greg O'Keefe: "From Power-up to Bash Prompt HOWTO";

[4] Robert Nemkin, Al Dev, Markus Gutschke, Ken Yap, Gero Kuhlmann: "Diskless Nodes HOWTO";

[5] Werner Almesberger, Hans Lermen: "Using the initial RAM disk (initrd)" disponibile nel file /usr/src/linux/Documentation/initrd.txt

[6] Paul Gortmaker, "Using the RAM disk block device with Linux" disponibile nel file /usr/src/linux/Documentation/ramdisk.txt