The Linux BootPrompt-HowTo

by Paul Gortmaker.

v1.4, Mar 21, 2003

Questo è il BootPrompt-Howto, è una raccolta di tutti i possibili argomenti di avvio che possono essere passati al kernel Linux nel momento dell'avvio. Con una discussione di come il kernel ordina gli argomerti di avvio, inoltre è inclusa una descrizione di alcuni software popolari usati per avviare i kernel Linux. Traduzione a cura di Sandro Cardelli, revisione a cura di Giulio Daprelà (giulio at pluto dot it)

Contents

1 Introduzione

Il kernel ha la capacità di accettare informazioni all’avvio nella forma di ‘riga di comando’, simile ad una lista di argomenti che si darebbe ad un programma. In generale questo viene usato per fornire al kernel informazioni circa i parametri hardware che il kernel non sarebbe in grado di determinare di per se, o evitare/sovrascrivere i valori che il kernel altrimenti rileverebbe.

È il lavoro del boot loader (es. LILO, loadlin o Grub) di prendere queste informazioni dall’utente e metterle in un posto precedentemente concordato dove il kernel possa trovarle una volta che si avvia.

La presente revisione copre i kernel fino e incluso il v2.4.20. e v2.5.63

Il BootPrompt-Howto è di:

> Paul Gortmaker, `p_gortmaker @ yahoo.com`

This document is Copyright (c) 1995-2003 by Paul Gortmaker. Please see the Disclaimer and Copying information at the end of this document (12.1 (copyright)) for information about redistribution of this document and the usual ‘we are not responsible for what you manage to break...’ type legal stuff.

1.1 A chi si rivolge e applicabilità

La maggior parte degli utenti Linux non dovrebbero neanche guardare questo documento. Linux fa eccezionalmente un buon lavoro nella rilevazione della maggior parte dell'hardware, selezionando ragionevolmente le impostazioni di default per la maggior parte dei parameteri. L'informazione in questo documento mira agli utenti che possono desiderare modificare alcune impostazioni di default per ottimizzare il kernel per le proprie particolari macchine, o a un utente che ha 'personalizzato il proprio' kernel per supportare un hardware non comune per il quale i default automatici non sono l'ottimo.

Per il bene di questo documento è meglio dividere gli argomenti di avvio in due categorie generali; (a)una gestita dal kernel e (b)quella gestita da un driver di dispositivo. Gli esempi sarebbero `init=` che dice al kernel quale dovrebbe essere il primo programma da eseguire, contro `aha154x=` che dice ad un driver di dispositivo per una scheda SCSI quale risorsa hardware dovrebbe usare. Questo documento si concentra nel dare informazioni dettagliate su questi per i motivi descritti sotto.

NOTA IMPORTANTE: Gli argomenti di avvio relativi ai driver si applicano soltanto ai driver hardware che sono compilati direttamente nel kernel. Essi non hanno *nessun effetto* sui driver che sono caricati come moduli. La maggior parte delle distribuzioni Linux hanno un kernel di base 'nudo' , e i driver sono piccoli moduli che sono caricati dopo l'inizializzazione del kernel. Se non si è sicuri che si stiano usando moduli provare `lsmod`, guardare `man depmod` e `man modprobe` insieme con il contenuto del proprio `/etc/modules.conf`.

Alla luce di questo, gli argomenti di avvio dei driver di dispositivo sono realmente usati solo da poche persone che compilano i propri kernel, e perciò hanno i sorgenti del kernel alla mano. Queste persone usualmente controllano i sorgenti per le opzioni e le sintassi richieste da quel driver per ottenere le informazioni più aggiornate.

Per esempio, se si stanno cercando quali argomenti dovrebbero essere passati al driver AHA1542 SCSI, allora si dovrebbe andare nella directory `linux/drivers/scsi`, e cercare nel file `aha1542.c` `__setup(...   , ...)`. La prima cosa tra parentesi è l'argomento che si fornisce all'avvio, e la seconda cosa è il nome della funzione che processa il proprio argomento. Di solito vicino alla cima di questa funzione o all'inizio del file del sorgente si troverà una descrizione degli argomenti di avvio che il driver accetta.

1.2 Documentazione Relativa

Per adesso, i sorgenti del kernel hanno il file `linux/Documentation/kernel-parameters.txt`. Questo file contiene una breve lista di tutti gli argomenti di avvio che si possono fornire, insieme a veloci link al posto dove, tra i sorgenti, si possa trovare dove sono analizzati gli argomenti. L'idea è che questo file dia agli sviluppatori un posto veloce e facile per aggiungere una breve descrizione di ogni nuovo argomento che essi aggiungono lavorando ai sorgenti. Così che esso probabilmente sarà sempre più aggiornato di questo documento. Attualmente sto consideranto con discontinuità questo documento alla luce dell'esistenza del `kernel-parameters.txt`. (Opinioni?)

La directory `linux` si trova di solito in `/usr/src/` nella maggior parte delle distribuzioni. Tutti i riferimenti in questo documento a file che vengono forniti col kernel avranno il loro nome del path abbreviato che inizia per `linux` - si dovrà aggiungere `/usr/src/` o quant'altro appropriato per il proprio sistema. Alcune distribuzioni possono non installare i sorgenti completi del kernel di default, e inserire solo la directory `linux/include`. Se non si trova il file in questione allora installare i sorgenti del kernel e/o e fare uso dei comandi `find` e `locate` . Se non si trova il pacchetto dei sorgenti del kernel nella propria distribuzione allora i sorgenti del kernel sono disponibili su:

Kernel Source Home

La succesiva miglior cosa da leggere nei sorgenti C del kernel, sarà qualsiasi altro file di documentazione che viene distribuito con il kernel stesso. Ci sono ora alcuni di questi, e la maggior parte di loro può essere trovata nella directory `linux/Documentation` e nelle sue sottodirectory. A volte ci sarà il file `README.foo` che può essere trovato nella relativa directoty driver (es. `linux/drivers/???/`, dove ci dovrebbero essere esempi di `??? scsi`, `char`, o `net`). La tendenza generale è di spostare questi file dentro la directory Documentation, così se un file menzionato in questo documento non è più lì probabilmente è stato spostato.

Se si è capito quali argomenti di avvio si intende usare, e ora si vuole sapere come passare l'informazione al kernel, allora guardare la documentazione che è fornita con il software che si usa per avviare il kernel (es.LILO o loadlin). Una breve descrizione è fornita di seguito, ma non è il sostituto della documentazione fornita con il software di avvio.

1.3 Nuove Versioni di questo Documento

Nuove versioni di questo documento possono essere trovate via anonymous FTP nella maggior parte dei siti FTP Linux nella directory `/pub/Linux/docs/HOWTO/`. Saranno fatti aggiornamenti quando saranno disponibili nuove informazioni e/o driver. Se questa copia che si sta leggendo è più vecchia di sei mesi, allora si dovrebbe probabilmente controllare se ne esiste una più nuova. Suggerirei di vederlo via browser WWW o nel formato Postscript/dvi. Entrambi contengono cross-reference che sono rimaste in una semplice versione formato testo.

Se si vuole avere la copia ufficiale, ecco l'URL.

BootPrompt-HOWTO

2 Sommario degli argomenti di avvio

Questa sezione dà alcuni esempi di software che può essere usato per passare gli argomenti di avvio del kernel al kernel stesso. Dà anche un'idea di come gli argomenti vengono processati, quali limitazioni ci sono su questi, e come essi vengono passati a ogni dispositivo appropriato per il quale sono predisposti.

È *importante* notare che gli spazi *non* dovrebbero essere usati in un argomento di boot, ma solo tra argomenti separati. Una lista di valori che sono per un singolo argomento devono essere separati da una virgola tra i valori, e ancora senza nessuno spazio. Vedere i seguenti esempi di sotto.

```
        ether=9,0x300,0xd0000,0xd4000,eth0  root=/dev/hda1            *GIUSTO*
        ether = 9, 0x300, 0xd0000, 0xd4000, eth0  root = /dev/hda1    *ERRATO*
```

Una volta che il kernel Linux è su ed è in esecuzione, si possono vedere gli argomenti della riga di comando che sono stati piazzati all'avvio semplicemente digitando `cat /proc/cmdline` al prompt di una shell.

2.1 LILO (LInux LOader)

Il programma LILO (LInux LOader) scritto da Werner Almesberger è il più comunemente usato. Ha l'abilità di avviare vari kernel, e memorizza l'informazioni di configurazione in un file formato testo. La maggior parte delle distribuzioni hanno LILO come boot-loader di default. LILO può avviare DOS, OS/2, Linux, FreeBSD, etc. senza nessuna difficoltà, ed è molto flessibile.

Una tipica configurazione, dopo aver acceduto al proprio computer, farà fermare LILO e visualizzare brevemente `LILO:` Esso aspetterà poi qualche secondo per ogni input opzionale da parte dell'utente, e in caso contrario avvierà il sistema di predefinito. Le etichette tipiche dei sistemi che le persone usano nel file di configurazione di LILO sono `linux` e `backup` e `msdos`. Se si vuole digitare un argomento di avvio occorre digitarlo qui, dopo aver digitato l'etichetta del sistema che si vuole che LILO avvii, come mostrato nell'esempio di sotto.

```
        LILO: linux root=/dev/hda1
```

LILO è fornito di un'eccellente documentazione, e per gli scopi di avvio di argomenti discussi qui, il comando LILO `append=` è di particolare importanza quando si vuole aggiungere un argomento all'avvio come un aggiunta permanente al file di configurazione di LILO. Si aggiunga semplicemente qualcosa tipo `append = "foo=bar"` al file `/etc/lilo.conf`. Può anche essere aggiunto in cima al file di configurazione, facendolo applicare a tutte le sezioni, o in una singola sezione di sistema aggiungendolo dentro una sezione `image=` . Prego vedere la documentazione LILO per una più completa descrizione.

2.2 LoadLin

L'altro loader Linux usato comunemente è 'LoadLin' il quale è un programma DOS che ha la capacità di lanciare un kernel Linux da un prompt DOS (con argomenti d'avvio) presumendo che alcune risorse siano disponibili. Questo è buono per persone che usano il DOS e vogliono lanciare Linux dal DOS.

È anche molto utile se si ha un certo hardware che quale dipende dai driver forniti in DOS per avere l'hardware in uno stato conosciuto. Un esempio comune è la scheda audio 'SoundBlaster Compatibile' che richiede il driver DOS per impostare alcuni registri proprietari per rendere la scheda in modalità compatible SB. Avviando il DOS con i driver forniti, e caricando poi Linux da un prompt DOS con `LOADLIN.EXE` evita la reimpostazione della scheda, cosa che accadrebbe se invece avvenisse un riavvio. Perciò la scheda è lasciata in modalità compatibile SB e pertanto è usabile sotto Linux.

Ci sono anche altri programmi che possono essere usati per avviare Linux. Per una lista completa, guardare nei programmi disponibili sul proprio mirror locale ftp Linux, sotto `system/Linux-boot/`.

2.3 L'utilità "rdev"

Ci sono alcuni dei parametri di avvio del kernel che hanno i loro valori di default memorizzati in vari byte nell'immagine del kernel stessa. C'è un'utilità chiamata `rdev` che è installata sulla maggior parte dei sistemi, che sa dove sono questi valori e come modificarli. Può anche modificare cose che non hanno un argomento equivalente di avvio del kernel, come la modalità video usata di default.

L'utilità rdev ha anche degli alias come swapdev, ramsize, vidmode e rootflags. Queste sono le cinque cose che rdev può modificare, quello che è il dispositivo di root, il dispositivo di swap, i parametri della RAM disk, la modalità video di default, e il settaggio sola-lettura/lettura-scrittura del dispositivo di root.

Maggior informazioni su `rdev` possono essere trovate digitando `rdev -h` o leggendo la pagina di manuale inclusa (`man rdev`).

2.4 Come il Kernel ordina gli argomenti

La maggior parte degli argomenti di avvio ha la forma di:

```
        nome[=valore_1][,valore_2]...[,valore_11]
```

dove 'nome' è una chiave unica che è usata per identificare a quale parte del kernel deve essere dato il valore associato (se c'è). Argomenti di avvio multipli sono elencati separati solo da spazio come il formato di sopra. Notare che il limite a 11 è reale, perché il codice attuale gestisce solo 11 parametri separati da virgole per chiave. (Comunque, si può ri-usare la stessa chiave con altri 11 parametri aggiuntivi in situazioni inusualmente complicate, assumendo che la funzione di setup le supporti.) Notare anche che il kernel divide la lista in un massimo di dieci argomenti interi, e una seguente stringa, così non si possono realmente fornire 11 interi a meno che non si converta l'undicesimo argomento da una stringa a un intero nello stesso driver.

La maggior parte dell'ordinamento viene effettuato in `linux/init/main.c`. Prima il kernel controlla per vedere se l'argomento è uno degli argomenti speciali 'root=', 'ro', 'rw', o 'debug'. Il significato di questi argomenti speciali è descritto in aggiunta a questo documento.

Dopo scorre una lista di funzioni di setup (contenute nella tabella `bootsetups`) per vedere se la stringa specificata dell'argomento (come 'foo') è stata associata con una funzione di setup (`foo_setup()`) per un particolare dispositivo o parte del kernel. Se si, passa al kernel la riga `foo=3,4,5,6,bar` dopo il kernel dovrebbe cercare nella tabella `bootsetups` per vedere se 'foo' è registrato. Se c'è, esso chiamerebbe la funzione di setup associata a 'foo' (`foo_setup()`) per gestire gli argomenti interi 3, 4, 5 e 6 come sono passati alla riga di comando del kernel, e gestire anche l'argomento stringa `bar`.

2.5 Impostazione delle variabili d'ambiente.

Qualsiasi cosa nella forma 'foo=bar' che non è accettata come una funzione di setup come descritto sopra viene allora interpretata come una variabile d'ambiente che deve essere impostata. Un esempio sarebbe usare `TERM=vt100` o `BOOT_IMAGE=vmlinuz.bak` come argomento di avvio. Queste variabili d'ambiente sono tipicamente testate negli script di inizializzazione per abilitare o disabilitare una vasta gamma di cose.

2.6 Il passaggio di argomenti al programma 'init'

Ogni argomento rimanente che non è stato preso dal kernel e non è stato interpretato come variabile d'ambiente viene passato al processo uno, il quale è usualmente il programma `init` . Il più comune argomento passato al processo `init` è la parola *single* la quale istruisce `init` di avviare il computer in modalità singolo utente, non lanciando tutti i demoni usuali. Controllare la pagina di manuale per la versione di `init` installata sul proprio sistema per vedere quali argomenti accetta.

3 Argomenti di avvio Generali non specifici di dispositivo

Questi sono argomenti di avvio che non sono relativi a nessuno specifico dispositivo o periferica. Essi sono invece relativi a certi parametri interni del kernel, come la gestione della memoria, gestione del ramdisk, gestione del filesystem di root e altri.

3.1 Opzioni del Filesystem di root

Tutte le seguenti opzioni si riferiscono a come il kernel seleziona e gestisce il filesystem di root.

3.1.1 L'argomento 'root='

Questo argomento dice al kernel quale dispositivo deve essere usato come filesystem di root durante l'avvio. L'impostazione di default è il valore del dispositivo di root del sistema sul quale è stato compilato il kernel. Per esempio, se il kernel in questione è stato compilato su un sistema che usava '/dev/hda1' come parizione di root, allora il default del dispositivo di root dovrebbe essere '/dev/hda1'. Per sovrascrivere questo valore di default, e selezionare il secondo floppy drive come dispositivo di root, si dovrebbe usare 'root=/dev/fd1'.

Dispositivi di root validi sono ciascuno dei seguenti dispositivi:

(1) da /dev/hdaN a /dev/hddN, che sarebbe la partizione N su disco compatibile ST-506 da 'a a d'.

(2) da /dev/sdaN a /dev/sdeN, che sarebbe la partizione N su disco compatibile SCSI da 'a a e'.

(3) da /dev/xdaN a /dev/xdbN, che sarebbe la partizione N su disco compatibile XT da 'a a b'.

(4) /dev/fdN, che sarebbe il disco floppy drive numero N. Avendo N=0 dovrebbe essere il DOS drive 'A:', e N=1 dovrebbe essere 'B:'.

(5) /dev/nfs, che non è realmente un dispositivo, ma piuttosto un flag per dire al kernel di prendere il fs di root via rete.

(6) /dev/ram, che è la RAM disk.

È anche accettata la più inopportuna e meno portabile specificazione numerica dei possibili dispositivi di disco di sopra nel formato major/minor. (es. /dev/sda3 è major 8, minor 3, così si potrebbe usare `root=0x803` come alternativa.)

Questo è uno dei pochi argomenti di avvio del kernel che ha i propri default memorizzati nell'immagine del kernel, e che può perciò essere alterato con l'utilità `rdev` .

3.1.2 L'argomento 'rootflags='

Questo argomento permette di dare opzioni che si riferiscono al montaggio del filesystem di root proprio come si farebbe col programma `mount` . Un esempio potrebbe essere dare l'opzione `noatime` a un fs ext2.

3.1.3 L'argomento 'rootfstype='

Questa opzione permette di dare una lista di tipi di fs separati da virgola che saranno provati per il corretto accoppiamento quando si prova a montare il filesystem di root. Questa lista sarà usata al posto di quella interna di default che usualmente inizia con ext2, minix ecc.

3.1.4 L'argomento 'ro'

Quando il kernel si avvia, ha bisogno di un filesystem di root per leggerci le cose basilari. Questo è il filesystem di root che viene montato all'avvio. Comunque se il filesystem di root fosse montato con accesso di scrittura, non si potrebbe controllare in modo affidabile l'integrità del filesystem con alcuni file mezzi-scritti in corso. L'opzione 'ro' dice al kernel di montare il filesystem di root come 'sola lettura' così che ogni programma di controllo della consistenza del filesystem (fsck) possa con sicurezza assumere che non ci sono file mezzi-scritti in corso durante l'esecuzione del controllo. Nessun programma o processo può scrivere su file sul filesystem in questione fino a che sia 'ri-montato' con capacità di lettura/scrittura.

Questo è uno dei pochi argomenti di avvio del kernel che ha i suoi default memorizzati nell'immagine del kernel, e che possono perciò essere modificati con l'utilità `rdev` .

3.1.5 L'argomento 'rw'

Questo è l'esatto opposto di quello di sopra, in quanto dice al kernel di montare il filesystem di root in lettura/scrittura. Il default è di montare il filesystem di root in sola lettura. Non eseguire nessun programma tipo 'fsck' su un filesystem che è montato in lettura/scrittura.

Lo stesso valore memorizzato nel file immagine menzionato sopra è usato anche per questo parametro, accessibile via `rdev`.

3.1.6 L'argomento 'nfsroot='

Questo argomento dice al kernel quale macchina, quale directory e quale opzioni NFS usare per il filesystem di root. Notare anche che è richiesto l'argomento `root=/dev/nfs` . Informazioni detagliate sull'uso di un fs NFS di root sono nel file `linux/Documentation/nfsroot.txt`.

3.1.7 L'argomento 'ip=' o 'nfsaddrs='

Se si sta usando NFS come filesystem di root, allora non ci sono programi tipo `ifconfig` e `route` presenti fino a che il fs di root non viene montato, e così il kernel deve configurare l'interfaccia di rete direttamente. Questo argomento di avvio imposta i vari indirizzi dell'interfaccia di rete che sono richiesti per comunicare attraverso la rete. Se questo argomento non viene fornito allora il kernel prova a usare RARP e/o BOOTP per reperire questi parametri.

3.2 Opzioni relative alla gestione del RAM Disk

Le seguenti opzioni sono tute relative a come il kernel gestisce il dispositivo disco di RAM, il quale è normalmente usato per l'avvio di macchine durante la fase d'installazione, o per macchine con driver modulari che necessitano di essere installati per accedere al filesystem di root.

3.2.1 L'argomento 'ramdisk_start='

Per permettere a un'immagine del kernel di risiedere su un floppy disk insieme con un'immagine compressa di ramdisk, è stato aggiunto il comando 'ramdisk_start=<offset>'. Il kernel non può essere incluso nell'immagine compressa del filesystem di ramdisk, perché perché dovrebbe essere memorizzato ad partire dal blocco zero così che il BIOS possa caricare il bootsector e poi il kernel possa avviarsi e funzionare.

Notare: Se si sta usando un'immagine di ramdisk non-compressa, allora il kernel può essere una parte dell'immagine del filesystem che è stato caricato nel ramdisk, e il floppy può essere avviato con LILO, o i due possono essere separati come viene fatto per le immagini compresse.

Se si sta usando un'impostazione a due-dischi boot/root (kernel sul disco 1, immagine di ramdisk sul disco 2) allora il ramdisk dovrebbe iniziare dal blocco zero, e dovrebbe essere usato un offset zero. Dal momento che questo è il valore di default, in realtà non si dovrebbe affatto usare tale comando.

3.2.2 L'argomento 'load_ramdisk='

Questo parametro dice al kernel se provare a caricare un'immagine di ramdisk o no. Specificando 'load_ramdisk=1' si dirà al kernel di caricare un floppy in ramdisk. Il valore di default è zero, significa che il kernel non dovrebbe provare a caricare un ramdisk.

Vedere il file `linux/Documentation/ramdisk.txt` per una descrizione completa dei nuovi argomenti di avvio, e come usarli. Viene anche descritto come questi parametri possono essere impostati e memorizzati nell'immagine del kernel via 'rdev'.

3.2.3 L'argomento 'prompt_ramdisk='

Questo parametro dice al kernel se dare o no un prompt per chiedere di inserire il floppy contenente l'immagine del ramdisk. In una configurazione con singolo floppy l'immagine del ramdisk è sullo stesso floppy del kernel che ha appena terminato il caricamento/avvio, così non occorre un prompt. In questo caso si può usare 'prompt_ramdisk=0'. In una configurazione a due floppy, occorrerà dare la possibilità di cambiare disco, perciò si può usare 'prompt_ramdisk=1'. Dal momento che questo è il valore di default non occorre in realtà che sia specificato. (Nota storica: Le persone scaltre usavano l'opzione di LILO 'vga=ask' per mettere in pausa temporaneamente il processo di avvio e avere la possibilità di cambiare il floppy di boot con quello di root.)

Vedere il file `linux/Documentation/ramdisk.txt` per una descrizione completa dei nuovi argomenti di avvio, e come usarli. Viene anche descritto come questi parametri possono essere impostati e memorizzati nell'immagine del kernel attraverso 'rdev'.

3.2.4 L'argomento 'ramdisk_size='

Mentre è vero che il ramdisk cresce dinamicamente come richiesto, c'è un limite superiore per la propria dimensione così che non possa consumare tutta la RAM disponibile e rimanere in difficoltà. Il default è 4096 (cioè 4MB), che dovrebbe essere sufficentemente grande per qualsiasi bisogno. È possibile sovrascrivere il default con una dimensione più grande o più piccola con questo argomento di avvio.

Vedere il file `linux/Documentation/ramdisk.txt` per una descrizione completa dei nuovi argomenti di avvio, e di come usarli. Viene anche descritto come questi parametri possono essere impostati e memorizzati nell'immagine del kernel attraverso 'rdev'.

3.2.5 L'argomento 'ramdisk_blocksize='

Questo può essere regolato per una migliore gestione del comportamento della memoria. Citazione dal driver di ramdisk `rd.c`:

Dovrebbe essere auspicabile avere un soft-blocksize (che nel caso di driver di ramdisk è anche l'hard-blocksize ;) di PAGE_SIZE perché facendo questo si raggiungerà una migliore impronta di memoria. Usando un rd_blocksize di BLOCK_SIZE nel peggiore dei casi renderemo le pagine buffer PAGE_SIZE/BLOCK_SIZE page non liberabili. Invece con un rd_blocksize di PAGE_SIZE si è sicuri che solo una pagina sarà protetta. Dipende dalla dimensione del ramdisk che si vuole per modificare il blocksize di ramdisk per raggiungere il migliore o il peggiore comportamento MM. Il default è ancora BLOCK_SIZE (che occorre a rd_load_image che suppone che il filesystem nell'immagine usi un blocksize BLOCK_SIZE)

3.2.6 L'argomento 'ramdisk=' (obsoleto)

(NOTARE: questo argomento è obsoleto, e non dovrebbe essere usato ad eccezione dei kernel v1.3.47 e più vecchi. I comandi che dovrebbero essere usati per il dispositivo di ramdisk sono documentati sopra. Kernel più nuovi possono accettare questo come un alias per `ramdisk_size`.)

Questo specifica la dimensione in kB del dispositivo di RAM disk. Per esempio, se si vuole avere un filesystem di root su un floppy da 1.44MB che sia caricato su un dispositivo di RAM disk, si dovrebbe usare:

```
        ramdisk=1440
```

Questo è uno dei pochi argomenti di avvio del kernel che ha i suoi default memorizzati nell'immagine del kernel, e che possono perciò essere modificati con l'utilità `rdev` .

3.2.7 L'argomento 'noinitrd' (RAM disk iniziale)

I kernel v2.x e successivi hanno una caratteristica dove il filesystem di root può essere inizialmente un RAM disk, e il kernel esegue `/linuxrc` su quell'imagine RAM. Questa caratteristica è usata tipicamente per permettere il caricamento di moduli che occorrono per montare il reale filesystem di root (es. carica i moduli di driver SCSI memorizzati nell'imagine di RAM disk, e dopo monta il reale filesystem di root su un disco SCSI.)

L'effettvo argomento 'noinitrd' determina che cosa accada ai dati initrd dopo che il kernel è stato avviato. Quando specificato, invece di convertirlo in un RAM disk, esso è accessibile via `/dev/initrd`, il quale può essere letto una volta prima che la RAM sia rilasciata al sistema. Per completi dettagli su come usare il RAM disk iniziale, prego consultare `linux/Documentation/initrd.txt`. In aggiunta, le più recenti versioni di `LILO` e `LOADLIN` dovrebbero avere informazioni utili aggiuntive.

3.3 Argomenti di avvio relativi alla gestione della memoria

I seguenti argomenti alterano come Linux rileva o gestisce la memoria fisica e virtuale del proprio sistema.

3.3.1 L'argomento 'cachesize='

Sovrascrive il rilevamento della dimensione della cache di CPU di secondo livello (in kB). A volte bug hardware di CPU fanno valutare la dimensione della cache incorrettamente. Il kernel farà dei tentativi per correggere i problemi conosciuti, ma per alcune CPU non è possibile determinare quale dovrebbe essere la dimensione corretta. Questa opzione fornisce una sovrascrittura per queste situazioni.

3.3.2 L'argomento 'mem='

Questo argomento ha diversi scopi: lo scopo origninale era di specificare l'ammontare della memoria installata (o un valore minore di quello se si voleva limitare l'ammontare della memoria disponibile a linux).

Il successivo scopo (e difficilmente usato) è di specificare `mem=nopentium` il quale dice al kernel Linux di non usare la caratteristica di prestazione della page table di di 4MB. Se si vuole usarlo per entrambi gli scopi, usare un `mem=` separato per ciascuno.

La chiamata originale del BIOS definita nella specificazione del PC che riporta l'ammontare della memoria installata era progettata solo per essere in grado di riportare fino a 64MB. (sì, un'altra mancata previsione, proprio come il cilindro 1024 del disco... sigh.) Linux usa questa chiamata del BIOS all'avvio per determinare quanta memoria è installata. Una specifica più nuova (e820) permette al BIOS di rilevare questo dato corretamente sulla maggior parte delle macchine di oggi. Se si ha più di 64MB di RAM installata su una vecchia macchina, si può usare questo argomento di avvio per dire a Linux quanta memoria si ha. Ecco una citazione da Linus sull'uso del parametro `mem=` .

"Il kernel accetterà ogni parametro 'mem=xx' gli si fornisca, e se risulta che gli si è mentito, esso terminerà orribilmente prima o poi. Il parametro indica il più alto indirizzo di RAM indirizzabile, così 'mem=0x1000000' significa che si hanno 16MB di memoria, per esempio. Per una macchina di 96MB questo dovrebbe essere 'mem=0x6000000'. Se si dice a Linux che ha più memoria di quella che ha realmente, accadranno brutte cose: forse non la prima volta, ma prima o poi sicuramente."

Notare che l'argomento non deve essere in esadecimale, e può essere usato il suffisso 'k' e 'M' (case insensitive) per specificare kilobyte e Megabyte, rispettivamente. (Un 'k' causerà uno spostamento di 10 bit al proprio valore, e una 'M' causerà uno spostamento di 20 bit.) Un tipico esempio per una maccina da 128MB dovrebbe essere "`mem=128m`".

In alcuni casi, anche la memoria riportata attraverso e820 può essere sbagliata, e così è stata aggiunto `mem=exactmap` . Si usa questo insieme alla specifica di un'esatta mappa di memoria, come:

```
mem=exactmap mem=640K@0 mem=1023M@1M
```

per una macchina da 1GB con l'usuale 384k di spazio I/O mappato per la memoria ISA escluso dall'utilizzo.

3.3.3 L'argomento 'memfrac='

La memoria è suddivisa in zone; su gli i386 queste zone corrispondono al 'DMA' (per i dispositivi ISA legacy che possono indirizzare solo fino a 16MB via DMA); 'Normal' per memorie da 16MB fino a 1GB, e 'HighMem' per memorie oltre 1GB (assumendo che il proprio kernel sia compilato con il supporto highmem abilitato). I due (o tre) interi forniti qui determinano quanta memoria dovrebbe essere tenuta libera in ogni zona - con la dimensione della zona divisa per il numero fornito usato come minimo (così più piccoli sono i numeri significa tenerne più libera nella zona). I default sono attualmente `memfrac=32,128,128`.

3.3.4 L'argomento 'swap='

Questo permette all'utente di aggiustare alcuni dei parametri della memoria virtuale (VM) che sono relativi allo swapping su disco. Esso accetta i seguenti otto parametri:

```
MAX_PAGE_AGE
PAGE_ADVANCE
PAGE_DECLINE
PAGE_INITIAL_AGE
AGE_CLUSTER_FRACT
AGE_CLUSTER_MIN
PAGEOUT_WEIGHT
BUFFEROUT_WEIGHT
```

Agli hacker interessati si consiglia di leggere `linux/mm/swap.c` e anche di notare le buone cose in `/proc/sys/vm`. I Kernel sono forniti con della documentazione utile su questo nella directory `linux/Documentation/vm/` directory.

3.3.5 L'argomento 'buff='

Simile all'argomento 'swap=', questo permette all'utente di aggiustare alcuni dei parametri relativi alla gestione del buffer di memoria. Esso accetta i seguenti sei parametri:

```
MAX_BUFF_AGE
BUFF_ADVANCE
BUFF_DECLINE
BUFF_INITIAL_AGE
BUFFEROUT_WEIGHT
BUFFERMEM_GRACE
```

Agli hacker interessati si consiglia di leggere `linux/mm/swap.c` e anche di notare le buone cose in `/proc/sys/vm`.I Kernel sono forniti con della documentazione utile su questo nella directory `linux/Documentation/vm/`.

3.4 Altri vari argomenti di avvio del kernel

Questi vari argomenti di avvio permettono all'utente di aggiustare certi parametri interni del kernel.

3.4.1 L'argomento 'acpi='

Attualmente questo accetta solo 'off' per disabilitare il sottosistema ACPI.

3.4.2 L'argomento 'console='

Di solito la console è il primo terminale virtuale, e così i messaggi d'avvio appaiono sul proprio video VGA. A volte è interessante avere la possibilità di usare un'altro dispositivo tipo una porta seriale (o perfino una stampante!) per fargli fare da console quando non è presente alcun dispositivo video. È anche utile catturare i messaggi d'avvio se un problema ferma il loro progredire prima che essi possano essere loggati su disco. Un esempio sarebbe di usare `console=ttyS1,9600` per selezionare la seconda porta seriale a 9600 baud per fare da console. Maggiori informazioni possono essere trovate in `linux/Documentation/serial-console.txt`.

3.4.3 L'argomento 'debug'

Il kernel comunica messaggi importanti (e non così importanti) all'operatore attraverso la funzione `printk()` . Se il messaggio è considerato importante, la funzione `printk()` metterà una copia sulla console attuale così come la passerà al servizio `klogd()` così che esso rimanga loggato sul disco. La ragione per stampare i messaggi importanti alla console così come loggarli sul disco è perché sotto sfortunate circostanze (es. un guasto di un disco) il messaggio non verrà riportato su disco e sarà perso.

Il limite per cosa è e cosa non è considerato importante è impostato dalla variabile `console_loglevel` . Il default è di loggare ogni cosa più importante di `DEBUG` (livello 7) alla console. (Questi livelli sono definiti nel file include `kernel.h`) Specificando `debug` come argomento di avvio imposterà il livello di log della console a 10, così che *tutti* i messaggi del kernel appariranno sulla console.

Il livello di log della console può anche essere impostato usualmente all'avvio attraverso un'opzione del programma `klogd()` . Controllare la pagina di manuale per la versione installata sul proprio sistema per vedere come fare ciò.

3.4.4 L'argomento 'decnet='

Se si sta usando DECnet, si possono fornire qui due interi separati da virgola per dare rispettivamente la propria area e nodo.

3.4.5 L'argomento 'devfs='

Se si sta usando devfs, invece dei dispositivi standard statici in `/dev/` allora con questo argomento si possono fornire le parole `only` o `mount`. Ci sono anche argomenti aggiuntivi di debug che sono elencati nel sorgente.

3.4.6 L'argomento 'gpt'

Se si sta usando la gestione della tabella delle partizioni EFI GUID, si può usare questo per sovrascrivere i problemi associati ad un invalido PMBR.

3.4.7 L'argomento 'idle='

Impostando questo a 'poll' causa al loop idle nel kernel di verificare l'occorrenza del flag di reschedule invece di aspettare il verificarsi dell'interrupt. Questo può determinare un incremento di performance su sistemi SMP (anche se al costo di un incremento del consumo di energia).

3.4.8 L'argomento 'init='

Il kernel per default esegue il programma 'init' all'avvio, il quale ha cura di impostare il computer per gli utenti attraverso l'esecuzione di programmi getty, eseguendo gli script 'rc' e similari. Il kernel prima cerca `/sbin/init`, poi `/etc/init` (deprecato), e per ultimo, proverà a usare `/bin/sh` (possibilmente su `/etc/rc`). Se per esempio, il proprio programma init è corrotto e perciò blocca la possibilità dell'avvio, si può semplicemente usare il prompt di avvio `init=/bin/sh` il quale all'avvio si calerà direttamente in una shell, permettendo la sostituzione del programma corrotto.

3.4.9 L'argomento 'isapnp='

Questo prende la forma di: `isapnp=read_port,reset,skip_pci_scan,verbose`

3.4.10 L'argomento 'isapnp_reserve_dma='

Questo prende la forma di: `isapnp_reserve_dma=n1,n2,n3,...nN` dove n1 ... nN sono i numeri di canale DMA da non usare per il PnP.

3.4.11 L'argomento 'isapnp_reserve_io='

Questo prende la forma di: `isapnp_reserve_irq=io1,size1,io2,size2,...ioN,sizeN` dove ioX,sizeX sono le coppie d'inizio e lunghezza delle regioni I/O nello spazio I/O che non devono essere usate dal PnP.

3.4.12 L'argomento 'isapnp_reserve_irq='

Questo prende la forma di: `isapnp_reserve_irq=n1,n2,n3,...nN` dove n1 ... nN sono i numeri degli interrupt da non usare per il PnP.

3.4.13 L'argomento 'isapnp_reserve_mem='

Questo prende la forma di: `isapnp_reserve_mem=mem1,size1,mem2,size2,...memN,sizeN` dove ioX,sizeX sono le coppie di inizio e lunghezza delle regioni I/O nello spazio di memoria che non devono essere usate dal PnP.

3.4.14 L'argomento 'kbd-reset'

Normalmente su macchine basate su i386, il kernel Linux non resetta il controller della tastiera all'avvio, dal momento che si suppone lo faccia il BIOS. Ma come di solito, non tutte le macchine fanno quello che dovrebbero. Fornendo questa opzione può aiutare se si hanno problemi con il comportamento della propria tastiera. Esso semplicemente forza un reset al momento dell'inizializzazione. (Qualcuno sostiene che questo dovrebbe essere il comportamento di default in ogni caso).

3.4.15 L'argomento 'lockd.udpport=' e 'lockd.tcpport'

Questi dicono al kernel di usare i numeri di porta forniti per l'operazione lockd di NFS (per operazioni o UDP o TCP).

3.4.16 L'argomento 'maxcpus='

Il numero fornito con questo argomento limita il numero massimo di CPU attivate in modalità SMP. Usando un valore 0 è equivalente all'opzione `nosmp` .

3.4.17 L'argomento 'mca-pentium'

Le macchine IBM modello 95 Microchannel sembra si blocchino sul test che Linux usualmente fa per rilevare il tipo di accoppiamento del chip matematico. Dal momento che tutti i chip Pentium hanno incluso il processore matematico questo test (e il problema del blocco) può essere evitato usando questa opzione d'avvio.

3.4.18 L'argomento 'md='

Se il proprio filesystem di root è su un dispositivo multiplo allora si può usare questo (assumendo che si sia compilato nel supporto d'avvio) per dire al kernel il layout del dispositivo multiplo. Il formato (dal file `linux/Documentation/md.txt`) è:

```
md=md_device_num,raid_level,chunk_size_factor,fault_level,dev0,dev1,...,devN
```

Dove `md_device_num` è il numero del dispositivo md, es. 0 significa md0, 1 significa md1, ecc. Per `raid_level`, usare -1 per la modalità lineare e 0 per la modalità striped. Altre modalità non sono attualmente supportate. Il `chunk_size_factor` è solo per il raid-0 e raid-1 e imposta il chunk size come PAGE_SIZE spostando a sinistra il valore specificato. Il `fault_level` è solo per il raid-1 e imposta il numero massimo di fault al numero specificato. (Attualmente non supportato a causa della mancanza di supporto all'avvio per il raid1).

I `dev0-devN` sono una lista di dispositivi separati da virgola che compongono il dispositivo md individuale: es. `/dev/hda1,/dev/hdc1,/dev/sda1`

Vedere anche `raid=`.

3.4.19 L'argomento 'nmi_watchdog='

Fornendo un intero non-zero si abiliterà l'interrupt watchdog non mascherabile (assumendo che il supporto IO APIC sia incluso). Questo verifica se il contatore di interrupt si incrementa (indicando la normale attività del sistema) e in caso contrario assume che un processore sia inceppato e forza un dump d'errore di informazioni diagnostiche.

3.4.20 L'argomento 'no387'

Alcuni chip coprocessori i387 hanno bug si evidenziano quando usati in modalità protetta a 32 bit. Per esempio alcuni dei primi chip ULSI-387 causerebbero forti blocchi durante l'esecuzione di calcoli a virgola mobile, apparentemente dovuti a un bug nelle istruzioni FRSAV/FRRESTOR. L'uso dell'argomento di avvio 'no387' fa sì che Linux ignori il coprocessore matematico in caso se ne abbia uno. Certamente si deve poi aver compilato il proprio kernel con il supporto all'emulazione matematica! Questo può anche essere utile se si ha una di queste macchine 386 *veramente* vecchie che possono usare un 80287 FPU, dal momento che Linux non può usare un 80287.

3.4.21 L'argomento 'no-hlt'

La famiglia delle CPU i386 (e da ciò i suoi successori) ha un'istruzione 'hlt' che dice alla CPU che non deve accadere niente fino a che un dispositivo esterno (tastiera, modem, disco, etc.) invita la CPU a eseguire un task. Questo permette alla CPU di entrare in una modalità a 'basso consumo' dove essa si ferma come uno zombie fino a che un dispositivo esterno la sveglia (usualmente attraverso un interrupt). Alcuni dei primi chip i486DX-100 ebbero un problema con l'istruzione 'hlt', in quanto essi non potevano ritornare attendibilmente alla modalità operativa dopo che era stata usata questa istruzione. Usando l'istruzione 'no-hlt' si dice a Linux di eseguire un loop infinito quando non c'è niente altro da fare, e di *non* fermare la propria CPU quando non c'è attività. Questo permette alle persone con questi chip guasti di usare Linux, sebbene essi dovrebbero essere ben avvisati di cercare una sostituzione in garanzia dove possibile.

3.4.22 L'argomento 'no-scroll'

Usando questo argomento all'avvio disabilita le caratteristiche di scroll che rendono difficile l'uso di terminali Braille.

3.4.23 L'aromento 'noapic'

L'uso di questa opzione dice a un kernel SMP di non usare alcune delle caratteristiche avanzate del controller di interrupt su macchine multi processore. L'uso di questa opzione può essere richiesto quando un dispositivo (come quelli che usano i driver ne2k-pci o 3c59xi) ferma la generazione di interrupt (es. `cat`

`/proc/interrupts` mostra lo stesso contatore di interrupt). Vedere `linux/Documentation/IO-APIC.txt` per maggiori informazioni.

3.4.24 L'argomento 'noht'

Questo disabiliterà l'hyper-threading su processori intel che hanno questa caratteristica.

3.4.25 L'argomento 'noisapnp'

Se ISA PnP è compilato nel kernel, questo lo disabiliterà.

3.4.26 L'argomento 'nomce'

Alcuni processori più recenti hanno la capacità di auto-monitorare e rilevare le inconsistenze che non dovrebbero normalmente accadere. Se viene rilevata un'inconsistenza, avverrà un "Machine Check Exception" e il sistema sarà fermato (piuttosto che la perdita prosegua e i propri dati vengano corrotti). Si può usare questo argomento per disabilitare questa caratteristica, ma prima assicurarsi di controllare che la propria CPU non sia in surriscaldamento o altrimenti difettosa.

3.4.27 L'argomento 'nosmp'

L'uso di questa opzione dirà a un kernel SMP su una macchina SMP di operare in singolo processore. Tipicamente usata solo per il debug e la determinazione se un particolare problema sia relativo al SMP.

3.4.28 L'argomento 'noresume'

Se è abilitata la sospensione del software, ed è stata specificata una sospensione su file su disco, usando questo argomento si avrà un avvio normale e la sospensione dei dati sarà ignorata.

3.4.29 L'argomento 'notsc'

L'uso di questa opzione dirà al kernel di non usare per niente il Time Stamp Counter, anche se la CPU ne ha uno.

3.4.30 L'argomento 'nofxsr"

L'uso di questa opzione dirà al kernel di non usare alcun trucco di accellerazione incluso l'unità a virgola mobile, anche se il processore li supporta.

3.4.31 L'argomento 'panic='

Nello sfortunato evento di un kernel panic (es. un errore interno che viene rilevato dal kernel, e che il kernel decide sia serio a sufficenza da urlarlo e poi fermare il tutto), il comportamento di default è solo di fermarsi fino a che qualcuno si faccia avanti, si accorga del "panic message" sullo schermo e riavvii la macchina. Comunque se una macchina è in esecuzione non sorvegliata in un posto isolato può essere desiderabile farla resettare automaticamente così che la macchina ritorni in linea. Per esempio, usando all'avvio `panic=30` il kernel dovrebbe provare a riavviarsi da solo 30 secondi dopo l'avvenuta del kernel panic. Un valore di zero dà il comportamento di default, che è di aspettare indefinitamente.

Notare che questo valore di timeout può anche essere letto e impostato attraverso l'interfaccia sysctl sul `/proc/sys/kernel/panic`.

3.4.32 L'argomento 'pirq='

Usando questa opzione si dà a un kernel SMP l'informazione sul valore di IRQ dello slot PCI per le schede madri SMP che non sono conosciute (o conosciute essere in blacklist). Vedere `linux/Documentation/IO-APIC.txt` per maggiori informazioni.

3.4.33 L'argomento 'profile='

Gli sviluppatori del Kernel possono descrivere come e dove il kernel spende i suoi cicli di CPU nello sforzo di massimizzare l'efficenza e la performance. Questa opzione lascia impostare il contatore dello spostamento di profilo all'avvio. Tipicamente è impostato a due. Occorre un tool come `readprofile.c` che possa fare uso dell'output di `/proc/profile`

3.4.34 L'argomento 'quiet'

Questo è molto grazioso e opposto all'argomento 'debug'. Quando questo viene fornito, sono visualizzati alla console solo i messaggi del kernel critici di sistema e quelli importanti. I messaggi normali circa il rilevamento dell'hardware all'avvio sono soppressi.

3.4.35 L'argomento 'raid='

Al momento accetta `noautodetect` . Vedere anche `md=`.

3.4.36 L'argomento 'reboot='

Questa opzione controlla il tipo di reboot che Linux farà quando si resetta il computer (tipicamente attraverso `/sbin/init` gestendo un Control-Alt-Delete). Il default dei kernel v2.0 è di fare un riavvio a 'freddo' (es. un completo reset, il BIOS fa il controllo della memoria, ecc.) invece di un riavvio a 'caldo' (es. reset parziale, nessun controllo della memoria). È stato modificato per essere freddo di default dal momento che si tende a lavorare su hardware economico/guasto che fallisce il riavvio quando si richiede un riavvio a caldo. Per avere il vecchio comportamento (es. riavvio a caldo) usare `reboot=w` o in pratica funzionerà qualsiasi parola che inizia per `w`.

Le altre opzioni accettate sono ‘c’, ‘b’, ‘h’, e ‘s’, che stanno per cold/freddo, bios, hard, e SMP rispettivamente. L’opzione ‘s’ prende una cifra opzionale per specificare quale CPU dovrebbe gestire il riavvio. Le opzioni possono essere combinate, dove ciò è sensato, es. `reboot=b,s2`

3.4.37 L’argomento ‘reserve=’

Questo viene usato per *proteggere* regioni di porte I/O dalla rilevazione. La forma del comando è:

```
reserve=iobase,extent[,iobase,extent]...
```

In alcune macchine può essere necessario prevenire che i driver dei dispositivi controllino i dispositivi (autorilevamento) in una regione specifica. Questo può essere a causa di hardware mal progettato che causa *blocchi* all’avvio (così come alcune schede ethernet), hardware che viene identificato erroneamente, hardware il cui stato viene modificato da un primo rilevamento, o solo hardware che non si vuole che il kernel inizializzi.

L’argomento di avvio `reserve` risolve questo problema con la specifica di una regione di porta I/O che non dovrebbe essere rilevata. Quella regione è riservata nella tabella di registrazione delle porte del kernel come se un dispositivo fosse già stato trovato in quella regione (con il nome `reserved`). Notare che questo meccanismo non dovrebbe essere necessario sulla maggior parte delle macchine. Dovrebbe essere necessario il suo uso solo quando c’è un problema o un caso speciale.

Le porte I/O nella regione specificata sono protette contro i rilevamenti di dispositivi i quali eseguono un `check_region()` prima della cieca rilevazione dentro una regione di spazio I/O. Questo è stato inserito per essere usato quando alcuni driver erano sospesi su una NE2000, o mal-identificavano alcuni altri dispositivi come loro propri. Un corretto driver di dispositivo non dovrebbe scandagliare una regione riservata, a meno che un’altro argomento di avvio specifichi esplicitamente di farlo. Questo implica che `reserve` sarà molto spesso usato con altri argomenti di avvio. Quindi se si specifica una regione `riservata` per proteggere uno specifico dispositivo si deve generalmente specificare poi una rilevazione esplicita per quel dispositivo. La maggior parte dei driver ignorano la tabella di registrazione delle porte se gli viene fornito un indirizzo esplicito.

Per esempio, la riga di avvio

```
        reserve=0x300,32  blah=0x300
```

riserva dalla rilevazione tutti i driver di dispositivo eccetto il driver per ‘blah’ `0x300-0x31f`.

Come solito con le specifiche dell’avvio c’è un limite di 11 parametri, perciò si possono specificare solo 5 regioni riservate per chiave `reserve`. Specifiche multiple di `reserve` funzioneranno se si ha una insolita e complicata richiesta.

3.4.38 L’argomenyo ‘resume=’

Se si sta usando il software di sospensione, allora questo permetterà di specificare il nome del file dei dati di sospensione su disco che si vuole dal quale la macchina esegua il ripristino.

3.4.39 L’argomento ‘vga=’

Notare che questo non è realmente un argomento di avvio. È un’opzione che viene interpretata da LILO e non dal kernel come lo sono tutti gli altri argomenti di avvio. Comunque il suo uso è divenuto così comune che qui si merita una menzione. Esso può essere impostato attraverso l’uso di `rdev -v` o equivalentemente `vidmode` nel file vmlinuz. Questo permette di impostare il codice da usare per il BIOS video per modificare la modalità di display di default prima del reale avvio del kernel Linux. Le modalità tipiche sono 80x50, 132x44 e così via. Il miglior modo per usare questa opzione è di iniziare con `vga=ask` la quale proporrà una lista di varie modalità che possono essere usate con il proprio adattatore video prima di avviare il kernel. Una volta scelto un numero che si vuole usare dalla lista di sopra, si può inserirlo al posto di ‘ask’. Per maggiori informazioni, vedere il file `linux/Documentation/svga.txt` che viene fornito con tutte le recenti versioni del kernel.

Notare che i nuovi kernel (dal v2.1 in poi) hanno il codice di setup che modifica la modalità video come un’opzione, viene elencato come `supporto alla selezione della modalità video` così che occorre abilitare questa opzione se si desidera usare questa caratteristica\.

4 Argomenti di avvio per controllare il comportamento del bus PCI (‘pci=’)

L’argomento ‘pci=’ (non disponibile nei kernel v2.0) può essere usato per modificare il comportamento della rilevazione dei dispositivi a bus PCI e il comportamento del dispositivo stesso. In primo luogo il file `linux/drivers/pci/pci.c` controlla le opzioni `pci=` indipendentemente dall’architettura. Gli argomenti rimanenti permessi sono gestiti in `linux/arch/???/kernel/bios32.c` e sono elencati di seguito per ???=i386.

4.1 L’argomento ‘pci=assign-busses’

Questo dice al kernel di assegnare sempre tutti i numeri di bus PCI, sovrascrivendo qualsiasi cosa il firmware può aver fatto.

4.2 Gli argomenti ‘pci=bios’ e ‘pci=nobios’

Questi sono usati per impostare o azzerare il flag che indica che la rilevazione PCI sta avvenendo attraverso il BIOS PCI. Il default è di usare il BIOS.

4.3 Gli argomenti ‘pci=conf1’ e ‘pci=conf2’

Se è abilitata la modalità diretta PCI, l’uso di questi abilita o il tipo di configurazione 1 o il tipo 2. Questi implicitamente azzerano anche il flag di rilevazione BIOS PCI (es. ‘pci=nobios’).

4.4 L’argomento ‘pci=irqmask=’

Questo permette all’utente di fornire un valore di maschera di IRQ, il quale è convertito usando strtol(). Esso imposterà un bit di maschera di valori IRQ che possono essere assegnati automaticamente ai dispositivi PCI. In questo modo si può far escludere al kernel gli IRQ delle proprie schede ISA.

4.5 L’argomento ‘pci=lastbus=’

Questo permette all’utente di specificare un valore di ultimo bus, il quale è convertito usando strtol(). Esso scansionerà tutti i bus fino al bus N. Può essere utile se il kernel non è in grado di trovare i propri bus secondari e si vuole dirgli esplicitamente quali sono.

4.6 L’argomento ‘pci=noacpi’

Questo disabilita l’uso dell’informazione di routing di ACPI durante la fase di configurazione dei PCI.

4.7 L’argomento ‘pci=nopeer’

Questo disabilita il default di sistemazione dei peer bridge, il quale stando alle informazioni fa quanto segue:

“In caso ci siano host bridge peer fa la scansione del bus di ognuno. Anche se alcune fonti sostengono che i bridge host dovrebbero avere il l’header tipo 1 e avere assegnato un numero di bus come per i bridge PCI2PCI la realtà non supera questa prova e il numero di bus è usualmente impostato dal BIOS al primo valore libero.”

4.8 L’argomento ‘pci=nosort’

Usando questo argomento si istruisce il kernel di non ordinare i dispositivi PCI durante la fase di rilevamento.

4.9 L’argomento ‘pci=off’

Usando questa opzione si disabilita il rilevamento di tutti i bus PCI. Ogni driver di dispositivo che fa uso di funzioni PCI per trovare e inizializzare l’hardware molto probabilmente non funzionerà.

4.10 L’argomento ‘pci=usepirqmask’

Questo imposta il flag USE_PIRQ_MASK durante l’init PCI. Il kernel onorerà la possibile maschera IRQ memorizzata nella tabella PIR BIOS. Questo è necessario su alcuni sistemi con BIOS fallati, specialmente alcuni notebook HP Pavilion N5400 e Omnibook XE3. Questo non avrà effetto se è abilitato il routing ACPI IRQ.

4.11 L'argomento 'pci=rom'

Questo imposta il flag ASSIGN_ROM durante la fase di rilevamento. Il kernel assegnerà un'indirizzo di spazio per l'espansione ROM. Usare con cautela, poiché certi dispositivi condividono l'indirizzo decodificato tra ROM e altre risorse.

5 Argomenti di avvio per driver video frame buffer

L'argomento 'video=' (non disponibile nei kernel v2.0) è usato quando il livello di astrazione del dispositivo di frame buffer è compilato dentro il kernel. Se questo suona complicato, bene non è realmente troppo difficile. Di base significa che invece di avere un diverso programma video (il server X11R6) per ogni marca di scheda video (es. XF86_S3, XF86_SVGA, ...), il kernel dovrebbe avere un driver incluso disponibile per ogni scheda video ed esportare una singola interfaccia per il programma video così da richiedere solo un server X11R6 (XF86_FBDev). Questo è simile a come funziona la rete adesso - il kernel ha driver disponibili per ogni marca di scheda di rete ed esporta una singola interfaccia di rete così che esiste solo una versione del programma di rete (tipo Netscape) che funzionerà per tutti i sistemi, senza guardare alla marca della scheda di rete.

Il formato tipico di questo argomento è `video=name:option1,option2,...` dove `name` è il nome di una generica opzione o di un driver frame buffer. L'opzione `video=` è passata da `linux/init/main.c` in `linux/drivers/video/fbmem.c` per ulteriori processi. Qui è controllato per alcune opzioni generiche prima di provare a accoppiarlo con un nome di driver conosciuto. Una volta che l'accoppiamento con un nome di driver è fatto, la lista di opzioni separate da virgola viene poi passata a quel particolare driver per l'esecuzione finale. La lista di nomi di driver validi può essere trovata leggendo la tabella `fb_drivers` nel file `fbmem.c` monzionato sopra.

Informazioni sulle opzioni che ogni driver supporta saranno eventualmente trovate in `linux/Documentation/fb/` ma attualmente (v2.2) solo poche ci sono descritte. Sfortunatamente il numero di driver video e il numero delle opzioni per ciascuno è di per se abbastanza per un'altro documento e quindi troppo per essere elencato qui.

Se non c'è file di Documentazione per la propria scheda, si dovrà prendere l'informazioni sulle opzioni direttamente dal driver. Andare in `linux/drivers/video/` e guardare il file appropriato `???fb.c` (i ??? si baseranno sul nome della scheda). Là cercare una funzione con `_setup` nel nome e si dovrebbero vedere quali opzioni il driver prova ad accoppiare, come `font` o `mode` o...

5.1 L'argomento 'video=map:...'

Questa opzione è usata per impostare o sovrascrivere la console alla mappatura del dispositivo di frame buffer. Una lista di numeri separati da virgola imposta la mappatura, con il valore dell'opzione N preso a essere il numero di dispositivo frame buffer per la console N.

5.2 L'argomento 'video=scrollback:...'

Un numero dopo i due punti imposterà la dimensione di memoria allocata per il buffer di scrollback. (Usare i tasti Shift e Pagina Su o Pagina Giù per fare lo scroll). Un suffisso di 'k' o 'K' dopo il numero indicherà che il numero deve essere interpretrato come kilobyte invece di byte.

5.3 L'argomento 'video=vc:...'

Un numero o una gamma di numeri (es. `video=vc:2-5`) specificherà la prima, o la prima e l'ultima console virtuale frame buffer. L'uso di questa opzione ha anche l'effetto di impostare la console frame buffer perché *non* sia la console di default.

6 Argomenti di avvio per perfiferiche SCSI.

Questa sezione contiene le descrizioni degli argomenti di avvio che sono usati per passare informazioni circa gli adattatori host SCSI installati, e i dispositivi SCSI.

6.1 Argomenti per i driver Superiori e Medio-livello

I driver di livello superiore gestiscono tutti gli oggetti SCSI, senza badare se essi siano dischi, unità a nastri, o CD-ROM. I driver di medio livello gestiscono oggetti come dischi, CD-ROM e unità a nastri senza entrare nelle specifiche dei driver di dispositivi degli adattatori host di basso livello.

6.1.1 Massime LUN rilevate ('max_scsi_luns=')

Ogni dispositivo SCSI può avere un numero di 'sotto-dispositivi' contenuti in se stesso. L'esempio più comune è ogni CDROM SCSI che gestisce più di un disco per volta. Ogni CD è indirizzato come un 'Numero di Unità Logica' (LUN) di quel particolare dispositivo. Ma la maggior parte dei dispositivi, come hard disk, unità a nastro e simili sono solo un dispositivo, e saranno assegnati alla LUN zero.

Il problema sorge con dispositivi a singolo LUN con firmware difettoso. Alcuni dispositivi SCSI mal progettati (vecchi e sfortunatamente anche nuovi) non possono gestire la rilevazione per LUN non uguale a zero. Essi risponderanno con un blocco, e la possibilità di far cadere con loro l'intero bus SCSI.

Il kernel ha un'opzione di configurazione che permette di impostare il numero massimo di LUN rilevati. Il default è di rilevare solo LUN zero, per evitare il problema descritto sopra.

Per specificare il numero di LUN rilevati all'avvio, si immette 'max_scsi_luns=n' come argomento di avvio, dove n è un numero tra uno e otto. Per evitare i problemi descritti sopra, si dovrebbe usare n=1 per evitare imprevisti con dispositivi guasti.

6.1.2 Registrazione avvenimenti SCSI (‘scsi_logging=’)

Fornendo un valore non-zero a questo argomento di avvio abilita la registrazione di tutti gli eventi SCSI (errori, scansioni, mlqueue, mlcomplete, llqueue, llcomplete, hlqueue, hlcomplete). Notare che un miglior controllo di questi eventi che sono registrati può essere ottenuto attraverso l’interfaccia `/proc/scsi/scsi` se non si è interessati agli eventi che accadono all’avvio prima che il filesystem `/proc/` sia accessibile.

6.1.3 Parametri per unità a nastro SCSI (‘st=’)

Alcune configurazioni di avvio di unità a nastri SCSI può essere realizzata con l’uso del seguente:

```
        st=buf_size[,write_threshold[,max_bufs]]
```

I primi due numeri sono specificati in unità di kB. Il default `buf_size` è 32kB, e la massima dimensione che può essere specificata è un ridicolo 16384kB. Il `write_threshold` è il valore al quale il buffer viene connesso all’unità a nastro, con un valore di default di 30kB. Il massimo numero di buffer varia con il numero di drive rilevati, e ha un default di due. Un esempio d’uso dovrebbe essere:

```
        st=32,30,2
```

Dettagli completi possono essere trovati nel file `README.st` che è nella directory `scsi` dell’albero dei sorgenti del kernel.

6.2 Argomenti per driver di adattatori host SCSI

Questi sono argomenti per driver di dispositivi host SCSI di basso livello, e come tali sono tipicamente usati solo da chi compila il proprio kernel con i driver SCSI inclusi. Queste persone sono avvisate di controllare i sorgenti per la lista aggiornata delle opzioni che possono essere fornite ai loro driver.

`aha152x=` Adaptec aha151x, aha152x, aic6260, aic6360, SB16-SCSI

`aha1542=` Adaptec aha1540, aha1542

`aic7xxx=` Adaptec aha274x, aha284x, aic7xxx

`advansys=` AdvanSys SCSI Host Adaptors

`in2000=` Always IN2000 Host Adaptor

`AM53C974=` AMD AM53C974 based hardware

`BusLogic=` ISA/PCI/EISA BusLogic SCSI Hosts

`eata=` EATA SCSI Cards

`tmc8xx=` Future Domain TMC-8xx, TMC-950

`fdomain=` Future Domain TMC-16xx, TMC-3260, AHA-2920

`ppa=` IOMEGA Parallel Port / ZIP drive

`ncr5380=` NCR5380 based controllers

`ncr53c400=` NCR53c400 based controllers

`ncr53c406a=` NCR53c406a based controllers

`pas16=` Pro Audio Spectrum

`st0x=` Seagate ST-0x

`t128=` Trantor T128

`u14-34f=` Ultrastor SCSI cards

`wd7000=` Western Digital WD7000 cards

7 Hard Disk

Questa sezione elenca tutti gli argomenti di avvio associati con MFM/RLL, ST-506, XT standard, e dispositivi disco drive IDE. Notare che entrambi i driver IDE e generici ST-506 HD accettano l'opzione 'hd='.

7.1 Parametri di driver Dischi/CD-ROM IDE

Il driver IDE accetta un numero di parametri, che spazia dalle specifiche della geometria del disco, al supporto avanzato di controller di chip o controller chip guasti. I seguenti sono il sunto di alcuni dei più comuni argomenti di avvio. Per dettagli completi, si dovrebbe *veramente* consultare il file `ide.txt` nella directory `linux/Documentation`, dal quale è estratto questo riassunto.

```
"hdx="  è riconosciuto per tutte le "x" da "a" a "h", come "hdc".
"idex=" è riconosciuto per tutte le "x" da "0" a "3", come "ide1".

"hdx=noprobe"         : l'unità può essere presente, ma non rilevarla
"hdx=none"            : l'unità NON è presente, ignorare cmos e non rilevarla
"hdx=nowerr"          : ignorare il bit WRERR_STAT su questa unità
"hdx=cdrom"           : l'unità è presente, ed è un'unità cdrom
"hdx=cyl,head,sect"   : disk drive è presente, con la geometria specificata
"hdx=autotune"        : il driver proverà a sintonizzare la velocità dell'interfaccia
                              alla più veloce modalità PIO supportata,
                              se possibile solo per questa unità.
                              Non completamente supportata da tutti i tipi di chipset,
                              abbastanza probabile che causi problemi con
                              unità più vecchie o insolite.

"idex=noprobe"        : non provare ad accedere o usare questa interfaccia
"idex=base"           : rilevare un'interfaccia all'indirizzo specificato,
                              dove "base" è usualmente 0x1f0 o 0x170
```

```
                              e "ctl" è assunto essere "base"+0x206
"idex=base,ctl"        : specifica entrambi base e ctl
"idex=base,ctl,irq"    : specifica base, ctl, e numero di irq
"idex=autotune"        : il driver proverà a sintonizzare la velocità dell'interfaccia
                              alla più veloce modalità PIO supportata,
                              per tutte le unità su questa interfaccia.
                              Non pienamente supportata da tutti i tipi di chipset,
                              abbastanza probabile che causi problemi con
                              unità IDE più vecchie o insolite.
"idex=noautotune"      : il driver NON proverà a sintonizzare la velocità dell'interfaccia
                              Questo è il default per la maggior parte dei chipset,
                              eccetto il cmd640.
"idex=serialize"       : non sovrapporre operazioni su idex e ide(x^1)
```

I seguenti sono validi SOLO su ide0, e i default per la base ctl port non devono essere alterati.

```
"ide0=dtc2278"         : esamina/supporta l'interfaccia DTC2278
"ide0=ht6560b"         : esamina/supporta l'interfaccia HT6560B
"ide0=cmd640_vlb"      : *RICHIESTO* per schede VLB con il chip CMD640
                         (non per PCI -- rilevato automaticamente)
"ide0=qd6580"          : esamina/supporta l'interfaccia qd6580
"ide0=ali14xx"         : esamina/supporta chipset ali14xx (ALI M1439/M1445)
"ide0=umc8672"         : esamina/supporta chipset umc8672
```

Durante l'installazione di qualche sistema PCMCIA, si può essere in grado di rilevare il proprio CD-ROM usando:

```
"ide2=0x180,0x386"     : rileva la locazione dell'interfaccia tipica PCMCIA IDE
```

Qualsiasi altra cosa è rigettata con il messaggio ”BAD OPTION”. Notare anche che c'è un implicito `ide0=0x1f0 ide1=0x170` in assenza di ogni altro argomento ide di avvio.

7.2 Old MFM/RLL/Standard ST-506 Disk Driver Options (‘hd=’)

I driver di disco standard possono accettare argomenti di geometria per i dischi simili ai driver IDE. Notare comunque che ci si aspetta solo tre valori (C/H/S) – altri in più o in meno saranno ignorati silenziosamente. Inoltre, si accetta solo ‘hd=’ come argomento, es. ‘hda=’, ‘hdb=’ e così via non sono validi qui. Il formato è come il seguente:

```
        hd=cyls,heads,sects
```

Se ci sono due dischi installati, il suddetto viene ripetuto per i parametri di geometria del secondo disco.

7.3 Opzioni di driver disco XT (‘xd=’, ‘xd_geo=’)

Se si è sufficentemente sfortunati da usare uno di queste vecchie schede a 8 bit che trasferiscono dati al massimo a 125kB/s allora ecco lo scoop. Il programma di rilevamento per queste schede cerca un BIOS installato, e se non è presente, non rileverà la propria scheda. Oppure, se la stringa di firma del proprio BIOS non è riconosciuta anche in questo caso non verrà rilevato. In ogni caso, si dovrà usare un argomento di avvio della forma:

```
        xd=type,irq,iobase,dma_chan
```

Il valore `type` specifica la particolare marca della scheda, e i valori sono i seguenti: 0=generic; 1=DTC; 2,3,4=Western Digital, 5,6,7=Seagate; 8=OMTI. La sola differenza tra tipi multipli dello stesso costruttore è la stringa del BIOS usata per il rilevamento, la quale non è usata se è specificato type.

La funzione `xd_setup()` non controlla i valori, e assume che si immettano tutti i quattro valori. Non deluderla. Ecco un esempio di utilizzo per un controller WD1002 con il BIOS disabilitato o rimosso, usando i parametri di ‘default’ del controller XT:

```
        xd=2,5,0x320,3
```

Se la geometria del disco che il kernel stampa appare tutta sbagliata rispetto a quella che si conosce, è possibile sovrascriverla facilmente, con:

```
        xd_geo=cyl_xda,head_xda,sec_xda
```

Aggiungere un’altra virgola e altri tre valori CHS se si è sufficentemente sciocchi da avere due dischi del vecchio pezzo da buttare...

8 I driver sonori

Notare che sono stati riscritti molta parte del nucleo audio e relativi driver. Il materiale vecchio è generalmente chiamato ‘OSS’ e il nuovo è chiamato ‘ALSA’. L’intenzione è di abbandonare l’OSS. Per evitare conflitti di nome, il materiale ALSA generalmente ha ‘snd-’ come prefisso a tutti i parametri di avvio.

Notare che ogni driver ha il proprio argomento individuale di avvio condiviso (kernel molto vecchi usavano `sound=`). Inoltre, generalmente nessun default era impostato al momento della compilazione (es. *occorre* fornire un argomento di avvio per far rilevare vecchie schede ISA non-PNP.) La miglior fonte di informazioni per la propria scheda sono i file in `linux/Documentation/sound/`.

8.1 Argomenti Individuali di driver di dispositivi audio

8.1.1 ALSA ISA drivers

`snd-dummy=` Dummy soundcard

`snd-mpu401=` mpu401 UART

`snd-mtpav=` MOTU Midi Timepiece

`snd-serial=` Serial UART 16450/16550 MIDI

`snd-virmidi=` Dummy soundcard for virtual rawmidi devices

`snd-ad1816a=` ADI SoundPort AD1816A

`snd-ad1848=` Generic driver for AD1848/AD1847/CS4248

`snd-als100=` Avance Logic ALS100

`snd-azt2320=` Aztech Systems AZT2320 (and 2316)

`snd-cmi8330=` C-Media's CMI8330

`snd-cs4231=` Generic driver for CS4231 chips

`snd-cs4232=` Generic driver for CS4232 chips

`snd-cs4236=` Generic driver for CS4235/6/7/8/9 chips

`snd-dt019x=` Diamond Technologies DT-019x

`snd-es1688=` Generic ESS AudioDrive ESx688

`snd-es18xx=` Generic ESS AudioDrive ES18xx

`snd-gusclassic=` Gus classic

`snd-gusextreme=` Gus extreme

`snd-gusmax=` Gus Max

`snd-interwave=` Interwave

`snd-interwave-stb=` Interwave

`snd-opl3sa2=` Yamaha OPL3SA2

`snd-opti93x=` OPTi 82c93x based cards

`snd-opti92x-cs4231=` OPTi 82c92x/CS4231

`snd-opti92x-ad1848=` OPTi 82c92x/AD1848

`snd-es968=` ESS AudioDrive ES968

`snd-sb16=` SoundBlaster 16

`snd-sbawe=` SoundBlaster 16 AWE

`snd-sb8=` Old 8 bit SoundBlaster (1.0, 2.0, Pro)

`snd-sgalaxy=` Sound galaxy

`snd-wavefront=` Wavefront

8.1.2 driver OSS

`ad1848=` AD1848

`adlib=` Adlib

`mad16=` MAD16

`pas2=` ProAudioSpectrum PAS16

`sb=` SoundBlaster

`uart401=` UART 401 (on card chip)

`uart6850=` UART 6850 (on card chip)

`opl3=` Yamaha OPL2/OPL3/OPL4 FM Synthesizer (on card chip)

`opl3sa=` Yamaha OPL3-SA FM Synthesizer (on card chip)

`opl3sa2=` Yamaha OPL3-SA2/SA3 FM Synthesizer (on card chip)

8.1.3 driver ALSA PCI

`snd-ali5451=` ALi PCI audio M5451

`snd-als4000=` Avance Logic ALS4000

`snd-cmipci=` C-Media CMI8338 and 8738

`snd-cs4281=` Cirrus Logic CS4281

`snd-cs46xx=` Cirrus Logic Sound Fusion CS46XX

`snd-emu10k1=` EMU10K1 (SB Live!)

`snd-ens1370=` Ensoniq ES1370 AudioPCI

`snd-ens1371=` Ensoniq ES1371 AudioPCI

`snd-es1938=` ESS Solo-1 (ES1938, ES1946, ES1969)

`snd-es1968=` ESS Maestro 1/2/2E

`snd-fm801=` ForteMedia FM801

`snd-intel8x0=` Intel ICH (i8x0) chipsets

`snd-maestro3=` ESS Maestro3/Allegro (ES1988)

`snd-korg1212=` Korg 1212 IO

`snd-rme32=` RME Digi32, Digi32/8 and Digi32 PRO

`snd-nm256=` NeoMagic 256AV and 256ZX

`snd-rme96=` RME Digi96, Digi96/8 and Digi96/8 PRO/PAD/PST

`snd-rme9652=` RME Digi9652 audio interface

`snd-hdsp=` RME Hammerfall DSP

`snd-sonicvibes=` S3 SonicVibes

`snd-trident=` Trident 4DWave DX/NX & SiS SI7018

`snd-via82xx=` VIA South Bridge VT82C686A/B/C, VT8233A/C, VT8235

`snd-ymfpci=` Yamaha DS1/DS1E

`snd-ice1712=` ICEnsemble ICE1712 (Envy24)

9 CD-ROM (Non-SCSI/ATAPI/IDE)

Questa sezione elenca tutti i possibili argomenti di avvio riguardanti questi vecchi dispositivi CD-ROM su schede di interfaccia proprietarie. Notare che questo non include CD-ROM SCSI o IDE/ATAPI. Per questi tipi di CD-ROM vedere l'appropriata sezione.

Notare che la maggior parte di questi CD-ROM hanno file di documentazione che *dovrebbero* essere letti, e sono tutti in un posto a portata di mano: `linux/Documentation/cdrom`.

9.1 Argomenti di driver per vecchi CD-ROM

`aztcd=` Aztech Interface

`cdu31a=` CDU-31A and CDU-33A Sony Interface (Also Old PAS)

`sonycd535=` CDU-535 Sony Interface

`gscd=` GoldStar Interface

`isp16=` ISP16 Interface

`mcd=` Mitsumi Standard Interface

`mcdx=` Mitsumi XA/MultiSession Interface

`optcd=` Optics Storage Interface

`cm206=` Phillips CM206 Interface

`sjcd=` Sanyo Interface

`sbpcd=` SoundBlaster Pro Interface

10 Driver Seriali e ISDN

10.1 I driver ISDN

Prego vedere `linux/Documentation/isdn/` per completi dettagli di tutte le opzioni che i seguenti driver ISDN accettano.

`icn=` ICN ISDN driver

`pcbit=` PCBIT ISDN driver

`teles=` Teles ISDN driver

10.2 I driver Seriali

Prego vedere `linux/Documentation/` e/o il file `README` in `linux/drivers/char` per completi dettagli di tutte le opzioni che i seguenti driver supportano.

`digi=` DigiBoard Driver

`riscom8=` RISCom/8 Multiport Serial Driver

`baycom=` Baycom Serial/Parallel Radio Modem

11 Altri dispositivi Hardware

Ogni altro dispositivo che non rientra in nessuna delle suddette categorie viene raggruppato qui.

11.1 Dispositivi Ethernet (‘ether=’, ‘netdev=’)

Diversi driver fanno uso di diversi parameteri, ma tutti almeno condividono il fatto di avere un IRQ, un valore di base di porta I/O, e un nome. Nella sua forma più generica, appare qualcosa di questo tipo:

```
        ether=irq,iobase[,param_1[,param_2,...param_8]]],name
```

Il primo argomento non numerico viene preso come nome. I valori `param_n` (se applicabili) hanno usualmente diversi significati per ogni scheda o driver diversi. I tipici valori `param_n` sono usati per specificare cose come l’indirizzo di memoria condivisa, la selezione dell’interfaccia, il canale DMA e similari.

L’uso più comune di questo parametro è di forzare la rilevazione di una seconda scheda ethernet, dal momento che il default è di rilevarne solo una (con kernel 2.4 o più vecchi). Questo può essere realizzato con un semplice:

```
        ether=0,0,eth1
```

Notare che i valori zero per l’IRQ e base I/O nell’esempio di sopra dicono al/ai driver di autorilevarli.

NOTE IMPORTANTI GLI UTENTI DI MODULI: Questo sopra *non* forzerà il rilevamento della seconda scheda se si usano i driver come moduli caricabili durante l’esecuzione (invece di averli compilati dentro il kernel). La maggior parte delle distribuzioni Linux usano un kernel minimale combinato con una grande selezione di driver modulari. Il parametro `ether=` si applica soltanto ai driver compilati direttamente dentro il kernel.

L’Ethernet-HowTo ha una documentazione completa e vasta sull’uso di schede multiple e sull’implementazione specifica dei valori `param_n` su schede e driver dove vengono utilizzati. I lettori interessati dovrebbero far riferimento alla sezione in quel documento delle loro particolari schede per maggiori e complete informazioni.

Ethernet-HowTo

11.2 Il Floppy Disk Driver (‘floppy=’)

Ci sono molte opzioni floppy driver, e sono tutte elencate in `floppy.txt` dentro `linux/Documentation`. Ci sono troppe opzioni in quel file per essere elencate qui. Invece, sono riportate qui solo quelle opzioni che possono essere richieste per fare un’installazione di Linux e per farlo funzionare con meno del normale hardware.

`floppy=0,daring` Dice al driver floppy che il proprio controller floppy dovrebbe essere usato con cautela (disabilita tutte le operazioni audaci).

`floppy=thinkpad` Dice al driver floppy che si ha un Thinkpad. I Thinkpad usano una convenzione invertita per il segnale di cambio disco.

`floppy=nodma` Dice al driver floppy di non usare DMA per il trasferimento dei dati. Questo serve sugli Omnibook HP, i quali non hanno un canale DMA praticabile per il driver floppy. Questa opzione è utile anche quando si riceve frequentemente il messaggio ‘Unable to allocate DMA memory’. L’uso di ‘nodma’ non è raccomandato se si ha un FDC senza un FIFO (8272A o 82072). 82072A e successivi vanno bene). Il modello FDC è riportato all’avvio. Inoltre occorre almeno un 486 per usare nodma.

`floppy=nofifo` Disabilita interamente il FIFO. Questo serve se si riceve il messaggio ‘Bus master arbitration error’ dalla propria scheda Ethernet (o da altri dispositivi) mentre si accede al floppy.

`floppy=broken_dcl` Non usare il segnale di cambio disco (DisckChangeLine), ma assumere che il disco sia stato cambiato ogni volta che il nodo del dispositivo viene riaperto. Serve su alcune box dove il segnale di cambio disco è guasto o non supportato. Questa dovrebbe essere considerata una misura di ripiego, effettivamente questo rende l’operazione col floppy meno efficente a causa dell’inutile pulitura della cache, e leggermente più inaffidabile. Verificare il collegamento del proprio cavo e le impostazioni dei jumper se si hanno problemi al DCL . Comunque, alcune vecchie unità, e anche alcuni Laptop sono conosciuti non avere il DCL.

`floppy=debug` Stampa messaggi (aggiuntivi) di debug.

`floppy=messages` Stampa messaggi informativi per alcune operazioni (notifiche di cambio disco, avvertenze circa sovra e sotto funzionamenti, e circa l’autorilevamento).

11.3 Il driver Bus Mouse (‘bmouse=’)

Il driver busmouse accetta solo un parametro, che rappresenta il valore di IRQ hardware che deve essere usato.

11.4 Il driver MS Bus Mouse (‘msmouse=’)

Il driver MS mouse acetta solo un parametro, che rappresenta il valore IRQ hardware che deve essere usato.

11.5 Il driver Printer (‘lp=’)

Con questo argomento di avvio si può dire al driver della stampante quale porte usare e quali porte *non* usare. Questo ultimo diventa pratico se non si vuole che il driver della stampante richieda tutte le porte parallele disponibili, così che altri driver (es. PLIP, PPA) possano usarle al suo posto.

Il formato dell’argomento è multipli i/o, coppie di IRQ. Per esempio, `lp=0x3bc,0,0x378,7` dovrebbe usare la porta a `0x3bc` in modalità (polling) senza IRQ , e usare IRQ 7 per la porta a `0x378`. La porta a `0x278` (se c’è) dovrebbe non essere rilevata, dal momento che l’autorilevamento avviene solo in assenza di un argomento `lp=`. Per disabilitare interamente il driver della stampante, si può usare `lp=0`.

11.6 Il driver della porta parallela IP (‘plip=’)

Usando `plip=timid` dirà al driver plip di evitare qualsiasi porta che appaia essere in uso da altri dispositivi a porta parallela, Diversamente si può usare `plip=parportN` dove `N` è un intero non-zero che indica la porta parallela da usare. (Usando `N`=0 disabiliterà il driver plip.)

12 Copie, Traduzioni, Chiusura, etc.

Hey, ce l’abbiamo fatta alla fine! (Phew...) Adesso solo le note legali.

12.1 Copyright and Disclaimer

This document is Copyright (c) 1995-1999 by Paul Gortmaker. Copying and redistribution is allowed under the conditions as outlined in the Linux Documentation Project Copyright, available from where you obtained this document, OR as outlined in the GNU General Public License, version 2 (see linux/COPYING).

This document is *not* gospel. However, it is probably the most up to date info that you will be able to find. Nobody is responsible for what happens to your hardware but yourself. If your stuff goes up in smoke, or anything else bad happens, we take no responsibility. ie. THE AUTHOR IS NOT RESPONSIBLE FOR ANY DAMAGES INCURRED DUE TO ACTIONS TAKEN BASED ON THE INFORMATION INCLUDED IN THIS DOCUMENT.

A hint to people considering doing a translation. First, translate the SGML source (available via FTP from the HowTo main site) so that you can then generate other output formats. Be sure to keep a copy of the original English SGML source that you translated from! When an updated HowTo is released, get the new SGML source for that version, and then a simple `diff -u old.sgml new.sgml` will show you exactly what has changed so that you can easily incorporate those changes into your translated SMGL source without having to re-read or re-translate everything.

If you are intending to incorporate this document into a published work, please make contact (via e-mail) so that you can be supplied with the most up to date information available. In the past, out of date versions of the Linux HowTo documents have been published, which caused the developers undue grief from being plagued with questions that were already answered in the up to date versions.

12.2 Chiusura

Se si fosse trovato qualsiasi errore di digitazione, o informazioni obsolete in questo documento, si prega di farmelo sapere. È facile omettere qualcosa, perché il kernel (e il numero dei driver) è immenso comparato a come era quando ho iniziato questo documento.

Grazie,

Paul Gortmaker, `p_gortmaker @ yahoo.com`