The Linux Plug-and-Play HOWTO

David S.Lawyer *dave@lafn.org* v0.11, maggio 2000

Un aiuto per districarsi e gestire i dispositivi Plug-and-Play. Come far sì che il proprio sistema Linux supporti il Plug-and-Play. Traduzione di Giovanni Bortolozzo (*borto at pluto.linux.it*).

Indice

1 Introduzione

1.1 Copyright, marchi registrati, liberatoria e ringraziamenti

1.1.1 Copyright (in inglese)

1.1.2 Liberatoria

Anche se non ho intenzionalmente provato a mettervi fuori strada, probabilmente ci sono parecchi errori in questo documento. Vi invito a farmeli notare. Poiché questa è documentazione libera, dovrebbe essere ovvio che né io né gli autori precendenti possiamo essere ritenuti legalmente responsabili per qualsiasi errore.

N.d.T.: ovviamente nemmeno il traduttore può essere ritenuto legalmente (e nemmeno moralmente) responsabile di eventuali errori.

1.1.3 Marchi registrati

Qualsiasi nomr di prodotto (che inizia con un lettera maiuscola) dovrebbe essere considerato un marchio registrato. Tali marchi appartengono ai rispettivi proprietari.per qualsiasi errore.

1.1.4 Crediti

Daniel Scott ha controllato questo documento nel marzo del 2000 e ha trovato molti errori di battitura, ecc.

1.2 Piani futuri; come aiutare

Invito a segnalarmi qualsiasi errore in fatti, opinioni, logica, battitura, grammatica, chiarezza, link, ecc. Ma per prima cosa, se la data del documento è più vecchia di un mese, si controlli se si è in possesso dell'ultima versione. Invito inoltre ad inviarmi qualsiasi informazione che si pensa sia pertinente a questo documento.

Non ho studiato in dettaglio né gli isapnptools né le patch per il kernel (ma ho in programma di farlo). Non ho ancora ben capito come il BIOS configura il PnP (dipende dal tipo di BIOS), e neppure come Windows9x aggiorna la ESCD. Quindi questo HOWTO è incompleto e potrebbe essere inaccurato (mi si faccia sapere cosa c'è di errato). In questo HOWTO qualche volta ho usato ?? per indicare che in realtà non so la risposta.

1.3 Nuove versioni di questo HOWTO

Una nuova versione del Plug-and-Play-HOWTO dovrebbe essere disponibile circa ogni mese per la consultazione e il download nei siti che fanno da mirror a LDP. Per una lista di tali siti si veda: `http://linuxdoc.org//mirrors.html` . Sono disponibili diversi formati. Se si vuole semplicemente controllare la data dell'ultima versione potrebbe essere meglio non usare un mirror, quindi si dia un occhio a: `http://linuxdoc.org/HOWTO/Plug-and-Play-HOWTO.html` . La versione che si sta leggendo è: v.0.11, May 2000. Le novità di questa versione sono l'utilità scanport, diverse correzioni ortografiche e la difficoltà d'uso di setpci.

N.d.T.: l'ultima (e più recente) traduzione in italiano di questo documento è reperibile nei mirror di *ILDP* `http://www.pluto.linux.it/ildp/` e scaricabile in diversi formati dai relativi siti ftp.

2 Cosa dovrebbe fare il PnP: allocare «risorse-bus»

2.1 Cos'è il Plug-and-Play (PnP)?

Semplificando al massimo, il Plug-and-Play dice automaticamente al software (device driver o gestore (driver) di dispositivo) dove trovare i diversi pezzi hardware (i dispositivi) quali modem, schede di rete, schede audio, ecc. Il compito del Plug-and-Play è di far «accoppiare» dispositivi fisici con il software (gestori di dispositivo) che funziona con essi e di stabilire dei canali di comunicazione tra ogni dispositivo e il suo gestore. Più precisamente, il PnP alloca le seguenti «risorse-bus» sia al driver e che all'hardware: indirizzi I/O, IRQ, canali DMA (solo per il bus ISA) e regioni di memoria. Se non si capisce cosa siano queste 4 risorse-buse si leggano le sotto sezioni che seguono. Un articolo della Linux Gazette su 3 di queste risorse-bus è *Introduction to IRQs, DMAs and Base Addresses* . Una volta assegnate queste risorse-bus (e se è installato il gestore corretto) i file di dispositivo (device file) nella directory /dev/ sono pronti per l'uso.

Questo assegnamento PnP di certe risorse del bus è talvolta detto «configurazione», ma è solamente un tipo di configurazione a basso livello. Anche utilizzando pienamente il PnP, buona parte della configurazione dei dispositivi è fatta da qualche altra cosa. Ad esempio, per la configurazione dei modem, è inviata una «stringa di inizializzazione» al modem attraverso l'indirizzo di I/O del «canale». Questa «stringa di inizializzazione» non ha niente a che vedere con il PnP, sebbene il «canale» usato per inviarla al modem è stato allocato dal PnP. L'impostazione della velocità (e di molti altri parametri) di una porta seriale è fatta inviando messaggi al gestore del dispositivo («device driver») da programmi eseguiti dall'utente (spesso automaticamente all'avvio del sistema). Anche questa configurazione non ha niente a che spartire con il PnP. Quindi quando si parla di PnP con «configurazione» si intende solo un certo tipo di configurazione. Sebbene altra documentazione (come quella per MS Windows) chiama le risorse-bus semplicemente «risorse», io ho coniato il termine «risorse-bus» in modo da distringuerle dalla moltitudine di altri tipi di risorse.

2.2 Come fa un computer per trovare i dispositivi (e viceversa)

Un computer consta di una CPU/processore per effettuare la computazione e di memoria per immagazzinare programmi e dati. Inoltre, ci sono diversi dispositivi come vari tipi di dischi, una scheda video, una tastiera, scheda di rete, schede modem, schede audio, porte seriali e parallele, ecc. C'è anche un alimentatore per fornire energia elettrica, diversi bus sulla scheda madre per connettere assieme i dispositivi e la CPU, e un case per metterci tutto dentro.

In tempi antichi quasi tutti i dispositivi avevano la proprio scheda di interfaccia (scheda a circuito stampato). Oggi, oltre a tali schede, molti «dispositivi» hanno un piccolo chip montato permanentemente sulla «scheda madre». Le schede che vengono inserite nella scheda madre possono contenere più di un dispositivo. I chip di memoria sono qualche volta considerati come dispositivi ma non sono Plug-and-Play nel senso usato in questo HOWTO.

Affinché un computer funzioni correttamente, ogni dispositivo dev'essere sotto il controllo del suo gestore o «device driver». È un software che è parte del sistema operativo (talvolta caricato come modulo) e che viene eseguito nella CPU. I gestori di dispositivo sono associati con «file speciali» nella directory /dev (notare che non sono propriamente dei file). Hanno nomi come hda1 (la prima partizione sul disco fisso a), ttyS0 (la prima porta seriale), eth1 (la seconda scheda ethernet), ecc. Per rendere le cose più complicate, il particolare device driver selezionato, diciamo per eth1, dipenderà dal tipo di scheda ethernet che si ha. Quindi eth1 non può semplicemente essere assegnato ad una qualsiasi gestore di scheda ethernet. Deve essere assegnato ad un particolare gestore che funzionerà con il tipo di scheda ethernet installata. Per controllare un dispositivo, la CPU (sotto il controllo del device driver) invia comandi (e dati) e legge informazioni dai vari dispositivi. Per poterlo fare ogni driver di dispositivo deve conoscere l'indirizzo del dispositivo che controlla. La conoscenza di tale indirizzo è equivalente all'impostazione di un canale di comunicazione, anche se il «canale» fisico è in realtà il bus dati dentro al PC che è condiviso praticamente da qualsiasi altra cosa presente nel computer.

Il canale di comunicazione è in realtà un po' più complesso di come appena descritto. Un «indirizzo» è in realtà un intervallo di indirizzi ed esiste la controparte del canale (nota come interrupt) che permette ai dispositivi di inviare richieste urgenti di «aiuto» ai loro gestori.

2.3 Indirizzi I/O, ecc.

I PC hanno 3 spazi di indirizzi: I/O, memoria principale e configurazione (solo nel bus PCI). Tutti questi 3 tipi di spazi di indirizzi condividono lo stesso bus indirizzi (address bus) all'interno del PC. Ma il potenziale di alcune linee dedicate nel bus del PC segnala in quale «spazio» è un indirizzo: I/O, memoria principale o configurazione. Si veda la sezione 8.1 (Indirizzi) per maggiori dettagli. I dispositivi sono originariamente posizionati nello spazio di indirizzi I/O, sebbene in alcuni casi questi allochino pure dello spazio in memoria principale. Un indirizzo I/O qualche volta è detto semplicemente «I/O», «IO», «i/o» o «io». È pure usato il termine «porta di I/O». Ci sono due passi principali per allocare gli indirizzi I/O (e altre risorse-bus, come gli interrupt):

1. Impostare l'indirizzo I/O, ecc. sulla scheda (in uno dei suoi registri)
2. Fare in modo che il gestore del dispositivo sappia qual è questo indirizzo I/O, ecc.

Il precedente processo in due passi è qualcosa di simile al problema in due parti di trovare il numero di casa di qualcuno in una via. Si deve conoscere il numero di casa e qualcuno deve attaccare un numero nella facciata della casa in modo che possa essere trovata. Nei computer, il device driver deve conoscere l'indirizzo e il dispositivo hardware deve impostare lo stesso indirizzo in uno dei suoi registri. Entrambe queste cose devono essere fatte, ma alcuni fanno l'errore di farne solo una e poi si sorprendono quando il computer non riesce a trovare il dispositivo. Per esempio, provano ad usare «setserial» per assegnare un indirizzo ad una porta seriale senza realizzare che ciò comunica un indirizzo solamente al driver. Non imposta l'indirizzo nell'hardware della porta seriale stessa. Se la porta seriale ha in realtà un diverso indirizzo (o adirittura nessuno) e si dice qualcosa di errato a setserial, allora sì che si è nei casini.

Un altro vincolo ovvio è che un indirizzo deve essere per prima cosa impostato nella scheda prima che il gestore del dispositivo possa usarlo. Poiché i driver dei dispositivi spesso partono non appena è partito il computer, talvolta tentato di accedere ad una scheda (per vedere se è là, ecc.) prima che l'indirizzo sia stato impostato nella scheda dal programma PnP di configurazione. In tal caso si vede un messaggio d'errore che dice che non è possibile trovare la scheda sebbene questa ci sia (ma non ha ancora un indirizzo).

Quanto detto negli ultimi due paragrafi riguardo gli indirizzi I/O si applica tale e quale alle altre risorse-bus: 2.4 (IRQ – Panoramica), 2.5 (Canali DMA), e 2.6 (Regioni di Memoria). Cosa siano queste cose verrà spiegato nelle tre sezioni che seguono.

2.4 IRQ – Panoramica

Dopo aver letto questa sezione si può leggere 8.2 (Interrupt – Dettagli) per ulteriori dettagli. Quanto segue è intenzionalmente super semplificato. Oltre agli indirizzi, si deve gestire anche un numero di interruzione («interrupt») (come l'IRQ 5), detto numero di IRQ (Interrupt ReQuest = Richiesta di Interruzione). Si è già detto prima che il gestore del dispositivo deve conoscere l'indirizzo di una scheda per poter essere in grado di comunicare con quest'ultima. Ma cosa dire della comunicazione in senso opposto? Si supponga che il dispositivo debba comunicare qualcosa al suo gestore con una certa urgenza. Per esempio, il dispositivo potrebbe aver appena ricevuto un sacco di byte destinati alla memoria principale e quindi gli serve chiamare il suo gestore per trasferire questi dati dal suo buffer praticamente pieno alla memoria principale.

Il dispositivo come può chiedere aiuto? Non può usare il bus dati principali perché sicuramente è già in uso. Invece impone un particolare potenziale elettrico in una linea di interrupt dedicata (parte del bus), spesso riservata per quel particolare dispositivo. Questo segnale è detto interrupt («interruzione»). Ci sono l'equivalente di 16 linee di questo tipo in un PC e ognuna di queste comanda (indirettamente) da particolare gestore di dispositivo. Ogni linea ha un numero di IRQ (Interrupt ReQuest) univoco. Il dispositivo deve mettere il suo interrupt nella linea corretta e il gestore del dispositivo deve restare in attesa dell'interruzione sulla linea corretta. Su quale linea deve mettersi è specificato dal numero di IRQ salvato nel dispositivo. Lo stesso numero di IRQ dev'essere noto al gestore del dispositivo in modo che questo sappia su quale linea di IRQ restare in ascolto.

Una volta che il driver del dispositivo riceve l'interrupt (una chiamata di aiuto) deve capire perché è stato generato ed intraprendere l'azione appropriata per servire l'interrupt. Sul bus ISA ogni dispositivo ha bisogno di un proprio numero di IRQ univoco. Nel bus PCI, è permessa la condivisione degli IRQ.

2.5 Canali DMA

I canali DMA esistono solo per il bus ISA. DMA sta per «Direct Memory Access» (Accesso Diretto alla Memoria). È il posto dove un dispositivo ha la possibilità di prendere (rubare) alla CPU il controllo del bus principale del computer e trasferire i dati direttamente nella memoria principale. Normalmente la CPU vorrebbe fare questo trasferimento in due passi:

1. leggendo i dati dallo spazio di I/O del dispositivo e mettendoli dentro la CPU stessa
2. scrivendo questi dati dalla CPU alla memoria principale.

Con il DMA si fa solitamente un solo passo inviando i dati direttamente dal dispositivo alla memoria. Il dispositivo deve avere tale funzionalità nel suo hardware e quindi non tutti i dispositivi possono fare un DMA. Mentre è in esecuzione il DMA la CPU non può fare molto in quanto il bus principale è usato per il trasferimento DMA.

Il bus PCI non ha in realtà nessun DMA. Ha piuttosto qualcosa di ancora migliore: il bus mastering. Funziona in maniera simile al DMA e qualche volta è chiamato DMA (per esempio, i dischi fissi chiamati «UltraDMA»). Permette ad un dispositivo di diventare temporaneamente il padrone del bus (bus master) e di trasferire i dati in maniera analoga a quanto fa la CPU quando ne ha lei il potere. Non usa nessun numero di canale in quanto l'organizzazione del bus PCI è tale che l'hardware PCI sa quale dispositivo ha attualmente il controllo del bus e quali dispositivo lo sta chiedendo di diventare il prossimo bus master. Per il bus PCI non c'è quindi allocazione dei canali DMA.

Quando un dispositivo nel bus ISA vuole fare un DMA invia una richiesta DMA usando una linea dedicata per la richiesta di DMA in modo analogo ad una richiesta di interrupt. In realtà il DMA potrebbe essere gestito usando interrupt ma ciò introdurrebbe alcuni ritardi, cosicché è più veloce da fare utilizzando un tipo particolare di interrupt detto DMA-request. Come le interruzioni, le richieste DMA sono numerate in modo da identificare quale dispositivo sta facendo la richiesta. Questo numero è chiamato DMA-channel (canale DMA). Poiché tutti i trasferimenti DMA usano il bus principale (e solo uno può essere attivo in un determinato istante) in realtà usano tutti lo stesso canale, ma il numero del «canale DMA» serve per identificare chi sta usando il «canale». Nella scheda madre esistono dei registri hardware che contengono lo stato attuale di ogni «canale». Quindi per poter effettuare una richiesta DMA, il dispositivo deve conoscere il suo numero di canale DMA che deve essere salvato in un registro nel dispositivo fisico.

2.6 Intervalli di memoria

Ad alcuni dispositivi è assegnato dello spazio di indirizzi nella memoria principale. È spesso «memoria condivisa» (shared memory) o «I/O mappato in memoria» (memory mapped I/O). Altre volte è la memoria ROM stessa del dispositivo. Quando si parla delle risorse-bus si usa spesso semplicemente il termine «memoria». Quindi un dispositivo può anche usare uno spazio di indirizzi di I/O.

Quando di installa una scheda di questo tipo, in effetti si sta inserendo un modulo di memoria per la memoria principale. Questa memoria può essere sia ROM (Read Only Memory, Memoria a Sola Lettura) che memoria condivisa. Questa memoria può servire come mezzo di «trasferimento» diretto di dati tra il dispositivo e la memoria principale. Non è veramente un trasferimento in quanto il dispositivo mette i dati nella propria memoria che «casualmente» è la memoria principale. Sia la scheda che il gestore del dispositivo devono sapere dov'è questa memoria. L'indirizzo di memoria è solitamente molto alto in modo da non entrare in conflitto con gli indirizzi più bassi dei chip di memoria del computer.

La ROM è diversa. È come un programma (forse un gestore di dispositivo) che sarà usato con il dispositivo. Idealmente, dovrebbe funzionare con Linux e non solo con Windows ?? Può essere necessario che venga «oscurato» (shadowed), intendendo con ciò che viene copiato nei propri chip di memoria principale in modo da funzionare più velocemente. Un volta «oscurato» non è più a sola lettura.

2.7 «Risorse» sia al dispositivo che al gestore

I gestori di dispositivo devono quindi essere in qualche modo «collegati» all'hardware che controllano. Questo è fatto fornendo le risorse-bus (I/O, Memoria, IRQ e DMA) sia al dispositivo fisico che al software di gestione. Per esempio, una porta seriale usa solo 2 (delle 4 possibili) risorse: un IRQ e un indirizzo I/O. Entrambi questi valori deve essere comunicati sia al gestore del dispositivo che al dispositivo fisico stesso. Al gestore (e al suo dispositivo) è dato anche un nome (come ttyS1) nella directory /dev. L'indirizzo e il numero di IRQ sono salvati dal dispositivo in registri della scheda (o in un chip nella scheda madre). Nel caso di schede con i ponticelli (jumper), queste informazioni sono sempre salvate nell'hardware del dispositivo (sulla scheda, ecc.). Ma nel caso del PnP, i dati dei registri spesso vengono persi quanto il PC viene spento, cosicché le informazioni sulle risorse devono essere forniti di nuovo ad ogni dispositivo ogni volta che il PC viene riacceso.

2.8 Il problema

L'architettura di un PC fornisce solo in numero limitato di IRQ, canali DMA, indirizzi I/O e regioni di memoria. Se ci fossero solo alcuni dispositivi e tutti questi avessero risorse-bus standardizzate (come indirizzi I/O e numeri IRQ unici) non ci sarebbero problemi a collegare i gestori di dispositivo ai dispositivi stessi. Ogni dispositivo avrebbe delle risorse fisse e non andrebbe in conflitto con qualsiasi altro dispositivo nel computer. Non ci sarebbero due dispositivi che vorrebbero avere gli stessi indirizzi, non ci sarebbero conflitti di IRQ, ecc. Ogni driver potrebbe quindi essere programmato con indirizzi, IRQ, ecc. univochi già codificati all'interno del programma. La vita sarebbe più semplice.

Ma così non è. Non solo oggi ci sono così tanti dispositivi che i conflitti sono frequenti, e qualche volta uno ha necessità di avere anche più di uno dello stesso tipo. Per esempio qualcuno potrebbe avere più di un disco fisso, un po' di porte seriali, ecc. Per queste ragioni i dispositivi devono possedere un minimo di flessibilità in modo che possano essere impostati a un qualsivoglia indirizzi, IRQ, ecc. siano necessari per evitare i conflitti. Ma alcuni IRQ e indirizzi, come quello del clock e della tastiera, sono piuttosto standard. Questi non hanno bisogno di tale flessibilità.

A parte il problema nel conflitto nell'allocazione delle risorse-bus, c'è anche un problema nel caso si commetta un errore nel dire al device driver quali sono le risorse-bus. Per esempio, si supponga di aver messo IRQ 4 in un file di configurazione quando in realtà il dispositivo è impostato all'IRQ 5. Questo è un altro tipo di errore nell'allocazione delle risorse-bus.

L'allocazione delle risorse-bus, se fatta correttamente, stabilisce i canali di comunicazione tra l'hardware fisico e i relativi device driver. Per esempio, se un determinato intervallo di indirizzi I/O (risorsa) è allocato sia al gestore di dispositivo che ad un pezzo di hardware, allora in questo modo si è stabilito un canale di comunicazione unidirezionale tra loro. Il driver può inviare comandi ed informazioni al dispositivo. In realtà è un po' più che unidirezionale, in quanto il driver può ottenere informazioni dal dispositivo leggendone i suoi registri. Ma il dispositivo in questo modo non può inizializzare una qualsiasi comunicazione. Per inizializzarla il dispositivo ha bisogno di un IRQ in modo da creare un canale bidirezionali nel quale sia il driver che il dispositivo possono iniziare una comunicazione.

2.9 Il PnP trova i dispositivi connessi nelle porte seriali

Anche i dispositivi esterni connessi alla porta seriale attraverso un cavo (come i modem esterni) possono essere chiamati Plug-and-Play. Poiché solo la porta seriale stessa ha bisogno di risorse-bus (un IRQ e un indirizzo I/O) non è necessario allora risorse-bus per questo tipo di dispositivi. Quindi per essi il PnP non è in realtà necessario. Comunque, c'è una specifica PnP anche per questi dispositivi seriali esterni.

Un sistema operativo PnP troverà questi dispositivi esterni e leggerà il loro numero di modello, ecc. Poi può essere in grado di trovarne il gestore di dispositivo più adatto, e quindi non si deve dire ad un programma applicativo che si ha un certo dispositivo diciamo su /dev/ttyS1. Poiché si è in grado di informare manualmente il proprio programma applicativo (attraverso un file di configurazione, ecc.) su quale porta seriale è presente il dispositivo (ed eventualmente dicendogli anche qual'è il suo numero di modello) in realtà non si ha veramente bisogno di questa caratteristica PnP per le porte seriali.

3 La soluzione Plug-and-Play (PnP)

3.1 Introduzione al PnP

Il termine Plug-and-Play (PnP) ha diversi significati. In senso lato è semplicemente l'autoconfigurazione che avviene quando uno installa un nuovo dispositivo e questo si configura da solo. Nel senso usato in questo HOWTO, la configurazione è solo la configurazione delle risorse-bus PnP e la successiva notifica al gestore del dispositivo. In senso stretto è pure l'impostazione delle risorse-bus in un dispositivo hardware. Si possono pure intendere le specifiche PnP le quali (assieme ad altre cose) specificano come vengono letti e scritti i dati delle risorse PnP nei dispositivi (spesso schede) sul bus ISA. Le specifiche PCI standard (e non quelle PnP) fanno la stessa cosa per il bus PCI.

Il PnP mette in corrispondenza i dispositivi con i relativi driver e specifica i loro canali di comunicazione. Sul bus ISA, prima del Plug-and-Play le risorse-bus erano impostate sui dispositivi hardware tramite ponticelli (jumper). Ai driver software erano invece assegnate tramite file di configurazione (o simili) o interrogando il dispositivo agli indirizzi dove ci si aspettava risiedesse. Il bus PCI ha supportato il qualcosa di simile al PnP fin dall'inizio ed è così stato banale implementarlo per questo bus. Poiché le specifiche del bus PCI non usano il termine PnP, non è chiaro se è o meno lecito chiamare PnP il bus PCI (ma supporta nell'hardware quello che a tutt'oggi è detto PnP).

3.2 Come funziona (semplificato)

Quanto segue è una panoramica oltremodo semplificata del funzionamento del Plug-and-Play. Il programma di configurazione del PnP (a volte un programma nel BIOS) trova tutti i dispositivi PnP e chiede ad ognuno di essi di quali risorse-bus hanno bisogno. Controlla poi quali risorse-bus (IRQ, ecc.) gli è possibile distribuire. Naturalmente ci sono risorse-bus riservate usate dai dispositivi non PnP (legacy) (se ne è nota la presenza) e non dà via nessuna di queste risorse riservate. Usa poi alcuni criteri (non specificati nelle specifiche PnP) per assegnare le risorse-bus in modo che non ci siano conflitti e che tutti i dispositivi abbiano (se possibile) quello di cui hanno bisogno. Dice poi ad ogni dispositivo fisico quali sono le risorse-bus a lui assegnate e i dispositivi stessi si auto-configurano per usare solo le risorse-bus assegnate. Poi, in qualche modo, i gestori (driver) di dispositivo scoprono quali risorse-bus usano i corrispondenti dispositivi e sono quindi in grado di comunicare attivamente con i dispositivi che controllano.

Per esempio, si supponga che una scheda necessiti di un interrupt (numero di IRQ) e di 1 MB di memoria condivisa. Il programma PnP legge queste richieste dalla scheda. Assegna poi a quest'ultima l'IRQ5 e 1 MB di memoria nello spazio indirizzi, a partire dall'indirizzo 0xe9000000. Però non è sempre così semplice, in quanto la scheda potrebbe specificare che può usare solo certi numeri di IRQ (solo nel caso ISA) o che il MB di memoria deve risiedere entro un ben determinato intervallo di indirizzi. I dettagli sono diversi nel caso di bus PCI o ISA, e sono maggiormente complessi nel secondo caso.

Esistono alcune scorciatoie che il software PnP potrebbe usare. Una di mantenere traccia di come sono state assegnate le risorse-bus nell'ultima configurazione (l'ultima volta che è stato usato il computer) e di riusare queste informazioni. Windows9x e i BIOS PnP usano questa mentre lo fa il Linux standard. Windows9x salva queste informazioni nel suo «Registry» sul disco fisso, mentre un BIOS PnP le salva nella memoria non volatile del PC (detta ESCD); si veda la sezione 5.3.1 (Il Database ESCD del BIOS)).

Sotto Linux ogni dispositivo è a sé e non esiste un registro centralizzato non volatile degli assegnamenti delle risorse. Alcuni driver di dispositivo salvano l'ultima configurazione che hanno usato e la usano la prossima volta che viene acceso il computer. Assumono implicitamente che il resto dell'hardware non abbia bisogno delle risorse-bus che usano loro.

Se i dispositivi ricordassero la propria configurazione precedente, allora non ci sarebbe nessun hardware da configurare al successivo riavvio, però questi sembrano dimenticarsene quando il computer viene spento. Alcuni contengono una configurazione predefinita (che non è necessariamente l'ultima usata). Quindi il programma di configurazione PnP deve essere lanciato ogni volta che viene acceso il PC. Inoltre, se viene aggiunto un nuovo dispositivo, allora dev'essere configurato. L'allocazione delle risorse-bus a questo nuovo dispositivo può comportare il portar via alcune risorse-bus ad un dispositivo preesistente e l'assegnamento a quest'ultimo di risorse-bus alternative che possa usare.

3.3 L'avvio del PC

Quando il PC viene acceso per la prima volta il chip del BIOS esegue il suo programma per far avviare il computer (il primo passo è la verifica dell'hardware). Se il sistema operativo è immagazzinato nel disco fisso (com'è ormai usuale) allora il BIOS deve essere a conoscenza della presenza del disco fisso. Se il disco fisso è PnP allora il BIOS potrebbe usare i metodi PnP per trovarlo. Inoltre, per permettere all'utente di configurare manualmente il CMOS del BIOS e di rispondere a messaggi d'errore nell'avvio del computer, sono pure necessari uno schermo (una scheda video) e una tastiera. Il BIOS deve quindi a sua volta configurare questi dispositivi.

Un volta che il BIOS ha identificato il disco fisso, la scheda video e la tastiera è pronto per fare il «boot» (caricare il sistema operativo dal disco fisso). Se si è detto al BIOS che si ha un sistema operativo PnP, allora dovrebbe avviare il PC e poi lasciare al sistema operativo la conclusione della configurazione PnP. Diversamente, (prima del boot) un BIOS PnP probabilmente proverà a fare da solo il resto della configurazione PnP dei dispositivi (ma non dei loro gestori).

3.4 Bus

ISA è il vecchio bus dei PC IBM, mentre PCI è il bus più nuovo e veloce di Intel. Il bus PCI è stato progettato per quello che oggi viene detto PnP. Rende semplice (se confrontato con il bus ISA) scoprire come devono essere assegnate le risorse-bus PnP ai dispositivi hardware. Per vedere che cosa è successo si usi il comando `lspci` e/o si veda `/proc/pci` o `/proc/bus/pci`. Sono utili anche i messaggi che appaiono sullo schermo all'avvio (si usi Shift+PageUp per tornare su). Si veda 7.1 (Messaggi all'avvio).

Nel caso del bus ISA c'è un problema reale con l'implementazione del PnP poiché nessuno aveva in mente il PnP quando è stato progettato il bus ISA e praticamente non ci sono indirizzi I/O disponibili per il PnP da usare per inviare informazioni di configurazione ai dispositivi fisici. Di conseguenza il modo di gestire il PnP nel bus ISA è complicato. Sull'argomento è stato scritto un libro intero (si veda la sezione 5.9 (Libri sul PnP)). Tra le altre cose, si richiede che il programma di PnP assegni un «handle» (identificativo) temporaneo a ciascun dispositivo PnP in modo che questi possano essere indirizzati per la successiva configurazione PnP. L'assegnazione di uno di questi «handle» è detta «isolation» (isolamento). Per i dettagli si veda la sezione 8.4 (Isolamento) in Appendice.

Probabilmente il bus ISA è destinato a scomparire. Quando lo farà, il PnP sarà più semplice in quanto sarà più facile scoprire come il BIOS ha configurato l'hardware. Ci sarà ancora la necessità di accopiare i gestori di dispositivo con i dispositivi, come pure la necessità di configurare i dispositivi aggiunti quanti il PC è già funzionante. Queste necessità saranno soddisfatte se Linux diventa un sistema operativo PnP.

3.5 Linux deve gestire meglio il PnP

Il PnP (per il bus ISA) è stato inventato da Compaq, Intel e Phoenix. La Microsoft ne è stata il principale promotore. Dal punto di vista di Linux sarebbe stato meglio se il PnP non fosse mai stato «inventato». Alla fine il bus ISA si sarebbe estinto e avrebbe prevalso il bus PCI PnP-like così avremmo avuto a tutti gli effetti un PnP facile da implementare. Ma che piaccia o no, a tutt'oggi la maggior parte del nuovo hardware ISA è PnP e Linux non ha scelta se non quella di gestire efficacemente il PnP. Ma il Linux standard (come quello dei primi del 1999) rende la gestione del PnP complicata (specialmente sul bus ISA) mentre lo scopo del PnP è quello di rendere le cose più semplici.

In un certo senso, Linux è già in qualche modo PnP per il bus PCI. Quando il PC si avvia si possono notare dai messaggi sullo schermo che alcuni driver di dispositivo di Linux spesso trovano i loro corrispondenti dispositivi hardware (e le risorse che il BIOS gli ha assegnato). Ma ci sono situazioni che un sistema operativo PnP potrebbe gestire meglio:

1. una deficienza di risorse-bus
2. più di un driver per un dispositivo fisico
3. un driver attivato che non riesce a trovare il suo dispositivo fisico
4. nstallazioni a caldo di un dispositivo .

Gli utenti di Linux non hanno bisogno di investigare i dettagli del PnP per configurare come si deve i dispositivi PnP ISA. Una soluzione potrebbe essere una versione standardizzata del kernel di Linux che supporti il Plug-and-Play su ISA, PCI e altri bus. È stata scritta una patch per il kernel sebbene la maggior parte dei driver non la supportino. Non è parte standard di Linux. Si veda la sezione 5.6 (Patch per il Kernel).

4 Configurazione del BIOS PnP

Quando un computer viene acceso, prima di caricare il sistema operativo viene eseguito il BIOS. I BIOS più recenti sono PnP e configureranno alcuni (o al limite tutti) i dispositivi PnP. In molti BIOS non c'è modo di disabilitare il PnP e quindi bisogna conviverci. Qui di seguito alcune delle scelte che potrebbero essere presenti nel menu CMOS del proprio BIOS:

- 4.1 (Si ha un sistema operativo PnP?)
- 4.2 (Come devono essere controllate le risorse-bus?)
- 4.3 (Reinizializzare la configurazione?)

4.1 Si ha un sistema operativo PnP?

Se si risponde affermativamente, allora il BIOS PnP inizierà la configurazione PnP del disco fisso, della scheda video e della tastiera per rendere avviabile il sistema. Ma lascerà che il sistema operativo termini il lavoro di configurazione. Potrebbe fare un 8.4 (Isolamento) nel bus ISA lasciando i dispositivi disabilitati ma pronti per essere configurati dal sistema operativo. Nel caso di Linux si dovrebbe dire che non si ha un sistema operativo PnP. Se non lo si fa il BIOS potrebbe lasciare disabilitati i dispositivi ISA che non ha configurato ?? Inoltre potrebbero non venir configurati pure i dispositivi PCI ??

Se si dice al BIOS che non si ha un SO PnP, allora il BIOS farà tutta la configurazione da solo. A meno che non siano stati aggiunti nuovi dispositivi PnP, dovrebbe usare la configurazione che ha salvato nella memoria non volatile (ESCD). Si veda la sezione 5.3.1 (Il Database ESCD del BIOS). Se l'ultima sessione nel proprio computer è stata sotto Linux, allora non ci dovrebbero essere modifiche nella configurazione. Si veda la sezione 5.3 (Configurare il PnP dal BIOS). Ma se l'ultima sessione è avvenuta in Windows 9x (che è PnP) allora Windows potrebbe aver modificato l'ESCD. Si suppone lo faccia solo se si è «forzata» una configurazione o si è installato un dispositivo legacy. Si veda la sezione 5.3.2 (Usare Windows per Impostare l'ESCD). Se si sta usando isapnp o i programmi delle Utility PCI per la configurazione, questi saranno eseguiti dopo il BIOS e cambieranno le cose nel modo che gli si dice.

4.1.1 Interoperabilità con Windows9x

Se si sta usando sia Linux che Windows nello stesso PC, come rispondere alla domanda del BIOS: «Si ha un SO PnP?» Normalmente si dovrebbe dire no per Linux standard e sì per Windows9x. Ma è un sacco noioso dover reimpostare manualmente il menu CMOS ogni volta che si cambia SO. Una soluzione è di impostare il CMOS come non si avesse un SO PnP, anche se si avvia Windows. Uno si dovrebbe aspettare che Windows sia in grado di gestire questa situazione nella quale gli viene presentato l'hardware già completamente configurato dal BIOS. Ci si aspetta inoltre che anche se Windows non realizza che l'hardware è già configurato, lo configuri da se e poi funzioni correttamente. Ma le cose non sembrano andare così. Sembra che Windows possa dire ai suoi driver di dispositivo solo quello che è presente nel suo Registry. Ma la reale configurazione hardware (fatta dal BIOS) è quella salvata nell'ESCD e potrebbe non essere la stessa ==> problemi.

Un modo per renderle uguali è di installare (o reinstallare) Windows quando il BIOS è impostato per «nessun SO PnP». Ciò dovrebbe presentare a Windows l'hardware configurato dal BIOS. Se questa configurazione è senza conflitti, Windows teoricamente la manterrà e se la salverà nel suo Registry. A questo punto l'ESCD e in Registry sono sincronizzati. Se questa cosa funziona (e questa e l'ultima versione dell'HOWTO), fatemelo sapere perché ho ricevuto solamente una notifica di buon funzionamento.

Un altro metodo è di «rimuovere» i dispositivi che causa problemi in Windows, cliccando su rimuovi in «Gestione Periferiche». Poi si riavii con «Nessun SO PnP» (lo si imposti nel CMOS all'inizio del boot).

Windows poi reinstallerà i dispositivi, teoricamente usando le impostazioni sulle risorse-bus configurate dal BIOS. Attenzione che Windows probabilmente potrebbe chiedere di inserire il CD di installazione di Windows poiché talvolta non riesce a trovare i driver (e altri file simili) anche se ci sono già. Come prova ho «rimosso» una scheda NIC che aveva driver compatibili con Novell. Durante il riavvio, Windows l'ha reistallata come Microsoft networking invece di Novell. Ciò significa che ho dovuto reistallare il Client Novell. Fatemi sapere eventuali problemi con questo metodo (solo se questa è l'ultima versione di questo HOWTO).

4.2 Come devono essere controllate le risorse?

Questa scelta determina il metodo con cui si vuole siano allocati IRQ e DMA. Se impostata su «auto», il BIOS farà l'allocazione. Se impostata su «manual», si riserveranno manualmente alcuni IRQ per usarli con le schede «legacy» (non PnP). Ora il BIOS potrebbe o meno essere conscio delle schede legacy presenti. Il BIOS verrà a conoscenza delle schede legacy presenti se si eseguirà ICU (o simile) sotto Windows per dirglielo. Se ne è a conoscenza, allora si provi ad usare «auto». Se non le rileva allora si riservino manualmente gli IRQ necessari per le schede ISA legacy e si lasci quel che resta dell'allocazione al PnP del BIOS.

4.3 Reinizializzare la configurazione?

Questa scelta cancellerà la base di dati ESCD del BIOS che contiene sia il modo in cui configurare i dispositivi PnP che l'elenco delle configurazioni dei dispositivi legacy (non PnP). Mai fare questa cosa a meno che non si sia convinti che tale base di dati è errata e dev'essere ricostruita. Da qualche parte è affermato che lo si dovrebbe fare solo se non si riesce ad avviare il proprio computer. Se il BIOS perde i dati sui dispositivi legacy, allora si dovrà rieseguire ICA sotto DOS/Windows per ristabilire le informazioni.

5 Come gestire le schede PnP

5.1 Introduzione alla gestione delle schede PnP

Oggigiorno molto delle nuove schede interne sono Plug-and-Play (PnP). Sebbene esista del software anche sotto Linux per gestire il PnP, non è sempre facile da usare. Esistono i 6 diversi metodi sottoelencati per tener testa al PnP (ma alcuni potrebbero non essere adatti alla propria situazione). Quale, o quali, si dovrebbe usare dipende dai propri scopi. Quello che ora sembra il più conveniente, a lungo termine potrebbe rivelarsi essere né il più facile né il migliore. Un metodo apparentemente semplice è di non far niente e lasciare al BIOS PnP la configurazione, ma poi si dovranno fare un po' di ricerche per capire quello che il BIOS ha fatto. Un confronto tra tutti questi metodi dovrebbe essere scritto da qualcuno che li ha provati tutti. Potrebbe essere necessario usarne più di uno per portare a buon fine il lavoro.

- 5.2 (Disabilitare il PnP) tramite i ponticelli o software per DOS/Windows (ma molte schede non offrono questa possibilità);
- 5.3 (Configurare il PnP dal BIOS) (per il bus PCI si ha bisogno solamente di un BIOS PCI, altrimenti è necessario un BIOS PnP);
- 5.4 (Isapnp) è un programma che si può sempre usare per configurare i dispositivi PnP ma solo nel bus ISA
- 5.5 (Le Utility PCI) per la configurazione del bus PCI;
- 5.7 (Configurazione Tramite Windows) e successivo avvio di Linux da Windows/DOS. Da usare come ultima risorsa;

- 5.6 (Applicare una Patch al Kernel) per trasformare Linux in un sistema operativo PnP;
- 5.8 (Configurazione Fatta dal Device Driver), ma pochi lo fanno.

Ognuna delle suddette possibilità imposterà le risorse-bus nell'hardware. Ma solo le ultime due informeranno il gestore del dispositivo di quanto è stato fatto. Solo l'ultima dice tutto al driver (è il driver). Il modo nel quale viene informato il driver dipende dal driver stesso e può essere necessario far qualcosa per informarlo. Si veda 6 (Dire al Driver la Configurazione).

5.2 Disabilitare il PnP ?

Molti dispositivi sono solamente PnP senza la possibilità di disabilitarlo. Ma per alcuni si potrebbe essere in grado di disabilitare il PnP tramite ponticelli o eseguendo il programma per Windows che si è ricevuto assieme al dispositivo (configurazione senza ponticelli). Ciò eviterà l'onere, spesso complicato, della configurazione PnP. Non si dimentichi di dire al BIOS che queste risorse-bus sono riservate. Ci sono inoltre alcune ragioni per le quali si potrebbe non voler disabilitare il PnP:

1. Se si ha MS Windows nella stessa macchina, allora si vorrà permettere al PnP di configurare diversamente i dispositivi sotto Windows da come ha fatto sotto Linux.
2. L'intervallo di selezione dei numeri IRQ (o indirizzi di porta) ecc. potrebbe essere piuttosto limitato finché non si usa il PnP.
3. Si potrebbe avere un device driver di Linux che usa i metodi PnP per ricercare il dispositivo che controlla.
4. Se si ha, in futuro, la necessità di modificare la configurazione, può essere più semplice farlo se il dispositivo è PnP (nessuna modifica ai ponticelli o esecuzione di un programma per DOS/Windows).
5. Si hanno (o si avranno) altri dispositivi PnP che devono essere configurati e quindi si deve comunque fornire il PnP.

Una volta configurati come dispositivi non PnP, non possono essere configurati né da un qualsiasi software PnP né dal BIOS (finché non si spostino i ponticelli e/o si usi ancora il software di configurazione per DOS/Windows).

5.3 Introduzione all'uso del BIOS per la configurazione PnP

Se si ha un BIOS PnP, questo può configurare l'hardware. Ciò significa che il proprio BIOS legge le richieste di risorse da tutti i dispositivi e li configura (alloca a loro le risorse-bus). È quasi un rimpiazzo di un SO PnP tranne per il fatto che il BIOS non associa i dispositivi con i loro gestori né dice a quest'ultimi come ha effettuato la configurazione. Normalmente dovrebbe usare la configurazione che ha salvato nella sua memoria non volatile (ESCD). Se trova un nuovo dispositivo e se c'è un conflitto, il BIOS dovrebbe fare le modifiche necessarie alla configurazione e non userà esattamente quello che c'era nell'ESCD.

Il proprio BIOS deve supportare tale configurazione, ma talvolta non le fa correttamente o completamente. Un vantaggio nell'uso del BIOS è che è semplice in quanto nella maggior parte dei casi non c'è niente da impostare (se non dire nel menu CMOS del BIOS che non si ha un SO PnP). Mentre alcuni gestori di dispositivo possono essere in grado di rilevare automaticamente quello che ha fatto il BIOS, in alcuni casi sarà necessario determinarlo da sé (non sempre facile). Si veda la sezione 7 (Qual è la Mia Configurazione Corrente?). Un altro vantaggio è che il BIOS fa il suo lavoro prima dell'avvio di Linux cosicché tutte le risorse-bus sono pronte da usare (e trovate) dal driver del dispositivo che parte poco dopo.

Secondo la MS è solo opzionale (non richiesto) che il BIOS PnP sia in grado di effettuare la configurazione PnP dei dispositivi (senza l'aiuto di MS Windows). Ma sembra che la maggior parte dei BIOS fatti dopo il 1996 ?? possano farlo. Si dovrebbe inviare una nota di ringraziamento se lo fanno nel modo corretto. Configurano sia il bus ISA che il bus PCI, ma mi dicono che alcuni BIOS vecchi riescano a configurare solo il bus PCI. Per saperne di più sul proprio BIOS, si cerchi nel Web. Per piacere, non si chieda a me in quanto non ho dati in proposito. I dettagli del BIOS che si può voler sapere potrebbero essere difficili da trovare (o addirittura non disponibili). Alcuni BIOS hanno minime capacità PnP e provano girano la parte più difficile della configurazione alle utility per Windows. Se questo succede si deve trovare un altro metodo (come le isapnptools) o provare ad impostare il database ESCD se il BIOS ne ha uno. Si veda la sezione seguente.

5.3.1 Il database ESCD del BIOS

Il BIOS mantiene in un database non volatile una configurazione PnP che proverà ad usare. È detta ESCD (Extended System Configuration Data, ovvero Dati di Configurazione Estesa del Sistema). Ancora, la presenza dell'ESCD è opzionale, ma la maggior parte dei BIOS ce l'hanno. L'ESCD non solo immagazzina la configurazione delle risorse di tutti i dispositivi PnP ma contiene inoltre le informazioni di configurazione dei dispositivi non PnP (e se li segna come tali) in modo da evitare conflitti. I dati dell'ESCD sono solitamente salvati in un chip e rimangono integri quando viene tolta l'alimentazione, ma qualche volta sono salvati nel disco fisso??

L'ESCD è pensata per mantenere l'ultima configurazione usata, ma se si usa un programma come isapnp o le utility PCI di Linux (che non aggiornano l'ESCD) allora l'ESCD non saprà niente di quello che si è fatto e non salverà questa configurazione nell'ESCD. Un buon SO PnP aggiorna l'ESCD così la si può usare successivamente per tutti i SO non PNP (come il Linux standard). MS Windows può fare questa cosa solo in casi speciali. Si veda la sezione 5.3.2 (Usare Windows per Impostare l'ESCD).

Per usare quanto impostato nell'ESCD ci si assicuri di aver impostato «Not a PnP OS» («SO non PnP») o simile nel BIOS. Allora ogni volta che il BIOS parte (prima che sia caricato il SO Linux) dovrebbe configurare le cose in questo modo. Se il BIOS rileva una nuova scheda PnP che non è nell'ESCD, allora deve allocare le risorse-bus alla scheda e aggiornare l'ESCD. Potrebbe anche dover cambiare le risorse-bus assegnate a una scheda PnP già esistente e modificare l'ESCD di conseguenza.

Se ogni dispositivi ha salvato la sua ultima configurazione nel proprio hardware, non sarebbe necessaria la configurazione dell'hardware ogni volta che si accende il PC. Ma non funzionano in questo modo e quindi tutti i dati dell'ESCD devono essere mantenuti corretti se si usa il BIOS per il PnP. Ci sono alcuni BIOS che non hanno una ESCD ma hanno un po' di memoria non volatile per immagazzinare informazioni su quali risorse-bus siano riservate per l'uso con schede non PnP. Molti BIOS hanno entrambe le cose.

5.3.2 Usare Windows per impostare l'ESCD

Se il BIOS non imposta l'ESCD nel modo che si vuole (o nel modo in cui dovrebbe essere) allora sarebbe comodo avere un'utility sotto Linux per impostare l'ESCD. All'inizio del 1999 non ce n'è nessuna. Quindi uno potrebbe ripiegare nell'uso di Windows (se presente nello stesso PC) per farlo.

Ci sono tre modi per usare Windows per provare ad impostare/modificare l'ESCD. Un modo è di usare l'utility ICU progettata per DOS o Windows 3.x. Dovrebbe funzionare bene anche per Windows 9x/2k?? Un altro metodo è di impostare manualmente i dispositivi («forzarli») sotto Windows 9x/2x cosicché Windows metterà queste informazioni nell'ESCD quando viene normalmente chiuso il sistema. Il terzo è solamente per i dispositivi legagy che non sono plug-and-play. Se a Windows è nota la loro presenza e quali sono le risorse-bus che usano, allora Windows dovrebbe mettere queste informazioni nell'ESCD.

Se i dispositivi sono configurati automaticamente da Windows (senza che l'utente gli chieda di «modificare le impostazioni») allora tale configurazione probabilmente non finirà dentro l'ESCD. Naturalmente Windows potrebbe decidere da sé di configurare le cose nello stesso modo di come sono impostate nell'ESCD, cosicché alla fine sono identiche ma per una coincidenza.

I Windows 9x sono sistemi operativi PnP e configurano automaticamente i dispositivi. Mantengono il proprio database PnP ben nascosto nel Registry (salvato in un file binario di Windows). Nel Registry ci sono inoltre un sacco di altre cose relative alla configurazione oltre alle risorse-bus PnP. C'è sia una configurazione corrente delle risorse PnP in memoria che un'altra (forse la stessa) salvata sul disco fisso. Per vedere quella in memoria (?) sotto Windows98 o per forzare delle modifiche si usi «Gestione Periferiche».

In Windows98 ci sono due modi per avviare Gestione Periferiche: 1. Risorse del Computer –> Pannello di Controllo –> Proprietà del Sistema –> Gestione Periferiche. 2. (bottone destro del mouse) Risorse del Computer –> Proprietà –> Gestione Periferiche. Poi in Gestione Periferiche si selezioni il dispositivo (solitamente un processo in più passi se ci sono più dispositivi della stessa classe). Poi si clicchi su «Proprietà» e poi su «Risorse». Per modificare manualmente la configurazione si disabiliti «Usa impostazione automatica» e poi si clicchi su «Cambia Impostazioni». Ora si probi a modificare l'impostazione, ma Windows potrebbe non permettere la modifica. Se lo permette, si è «forzata» una modifica. Un messaggio dovrebbe infomarmare che si sta forzando. Se si vogliono mantenere le impostazioni esistenti mostrate da Windows ma renderle «forzate», allora si dovrà forzare una modifica in qualcosa e poi riforzarla al suo valore originale.

Per vedere cosa è stato «forzato» in Windows98 si veda l'elenco dell'«hardware forzato»: Avvio –> Programmi –> Accessori –> Sistema –> Informazioni sul Sistema –> Risorse Hardware –> Hardware Forzato. Quando si «forza» una modifica di risorse-bus in Windows, dovrebbe mettere tali modificge nell'ESCD (a patto che Windows sia chiuso correttamente). Dalla finestra «Informazioni del Sistema» si possono inoltre ispezionare quali IRQ e porte IO sono allocate sotto Windows.

Anche se Windows non mostra conflitti nelle risorse-bus, potrebbero essercene sotto Linux. Questo perché Windows potrebbe assegnare risorse-bus differenti da quello che fa l'ESCD. Nel raro caso in cui tutti i dispositivi sotto Windows siano legagy oppure siano stati «forzati», allora le configurazioni di Windows e dell'ESCD dovrebbero essere identiche.

5.3.3 Aggiungere un nuovo dispositivo (sotto Linux o Windows)

Se si aggiunge un nuovo dispositivo PnP e si ha il BIOS impostato a «nessun SO PnP», allora il BIOS dovrebbe configurarlo automaticamente e salvare la configurazione nell'ESCD. Se è un dispositivo legacy non PnP (o uno reso tale con i jumper, ecc.) allora ci sono un po' di opzioni per gestirlo.

Si può dire direttamente al BIOS (tramite il menu di configurazione CMOS) che certe risorse-bus (come gli IRQ) sono riservate e che non devono essere allocate dal PnP. Ciò non mette queste informazioni nell'ESCD. Ma ci può essere una scelta nel menu del BIOS che specifica quando, in caso di conflitto, queste scelte CMOS hanno o meno priorità su quanto è salvato nell'ESCD. Un altro metodo è di eseguire ICU sotto DOS/Windows. Un altro ancora è di installare manualmente il dispositivo sotto Windows 9x/2k e poi assicurarsi che la sua configurazione sia «forzata» (si veda la sezione precedente). Se è «forzata» Windows dovrebbe aggiornare l'ESCD quando si spegne il PC.

5.4 Isapnp (parte degli isapnptools)

Sfortunatamente, buona parte della documentazione per gli isapnptools è ancora difficile da comprendere a meno che non si conoscano le basi del PnP. Questo HOWTO dovrebbe aiutare, come d'altronde aiuta la FAQ distribuita con il programma. `isapnp` è solo per i dispositivi PnP sul bus ISA (non PCI). L'esecuzione del programma per Linux «isapnp» all'avvio del sistema configurerà tali dispositivi secondo i valori delle risorse specificate nel file /etc/isapnp.conf. È possibile creare automaticamente questo file di configurazione, ma poi lo si deve modificare manualmente per scegliere tra le varie opzioni. Con isapnp, un driver di dispositivo che sia parte del kernel può essere eseguito troppo presto, cioè prima che isapnp abbia impostato indirizzi, ecc. nell'hardware. Ciò risulta nel mancato rilevamento del dispositivo da parte del driver. Il driver prova negli indirizzi giusti, ma tali indirizzi non sono ancora stati impostati nell'hardware.

Se la propria distribuzione installa automaticamente gli isapnptools, allora probabilmente isapnp è già automaticamente eseguito all'avvio. In questo caso, tutto quel che si deve fare è modificare il file /etc/isapnp.conf come spiegato da «`man isapnp.conf`». Si noti che questa cosa è equivalente alla configurazione manuale del PnP poiché si fanno decisioni su come configurare il tutto modificando tale file di configurazione. Se si usa «isapnp» in questo modo e si ha un BIOS PnP, si dovrebbe dire al BIOS (quando lo si configura) che si ha un sistema operativo PnP ??

Se si esegue isapnp una volta per configurare i dispositivi ISA PnP, ma l'esecuzione di isapnp fallisce ogni volta che il computer viene avviato, allora se si ha MS Windows (95 o 98) sullo stesso PC la cosa potrebbe essere dovuta al seguente problema: quando si usa MS Windows (95 o 98), Windows potrebbe configurare diversamente le schede PnP in modo tale che non funzioneranno correttamente (se non del tutto) quando si torna in Linux. Si può usare il programma «pnpdump» per aiutarsi nella creazione del file di configurazione. Questo praticamente crea un file di configurazione anche se lo si deve modificare un po' prima di poterlo usare. Contiene alcuni commenti per aiutare nella modifica. Se si usa «isapnp» per configurare e si ha un BIOS PnP, probabilmente si dovrà dire al BIOS che non si ha un sistema operativo PnP poiché si vuole che il BIOS si occupi di configurare i dispositivi PCI. Sebbene il BIOS possa configurare i anche dispositivi ISA, isapnp comunque lo rifarà.

La terminologia usata nel file /etc/isapnp.conf all'inizio può sembrare un po' ostica. Per esempio per un indirizzo IO pari a 0x3e8 si potrebbe invece vedere «(IO 0 (BASE 0x3e8))». L'«IO 0» indica che questo è il primo (zeresimo) intervallo di indirizzi IO che questa dispositivo usa. Un altro modo per esprimere questa cosa sarebbe: «IO[0] = 0x3e8» ma isapnp non lo fa in questo modo. «IO 1» indica il secondo intervallo di indirizzi usato da questo dispositivo, ecc. «INT 0» ha un significato simile ma per gli IRQ (interrupt). Una singola scheda può contenere diversi dispositivi fisici e la spiegazione di prima riguarda solamente uno di questi.

5.5 Le utility PCI

Il nuovo pacchetto delle Utility PCI (= pciutils, incorrettamente detto «pcitools»), dovrebbe permettere di effettuare manualmente la configurazione PnP sul bus PCI. «lspci» elenca le risorse-bus, mentre «setpci» imposta l'allocazione delle risorse nei dispositivi hardware. Sembra che setpci sia pensato principalmente per l'uso negli script e al momento per usarlo si devono conoscere i dettagli dei registri di configurazione PCI. Questa informazioni non sono spiegate qui e nemmeno nella pagina man di setpci.

5.6 Applicare una patch al kernel per rendere Linux PnP

David Howells ha creato una patch per farlo chiamata «Linux Kernel Configuration/Resource Manager» (talvolta chiamata semplicemente Hardware Configuration Manager). Nel tardo 1999 la patch non era disponibile nel suo sito web. Ciò potrebbe indicare che non c'è una patch disponibile per le versioni recenti del kernel.

Con le patch precedenti il kernel risultante si diceva fosse stabile anche se erano stati segnalati dei bug. La patch include documentazione cone il serial.txt per mostrare come trattare la porta seriale. Fornisce «file» nell'albero /proc cosicché si può vedere cosa sta accadendo ed in uno di questi è possibile inserire comandi attraverso il comando echo per personalizzare la configurazione. Un problema è che la maggior parte dei gestori di dispositivo non sono consci delle funzionalità introdotte da questa patch e quindi per la configurazione si devono usare ancora i tradizionali file di configurazione, ecc. La pagina web è `http://www.astarte.free-online.co.uk` .

5.7 Configurazione tramite Windows

Si si ha Windows9x (o 2k) sullo stesso PC, allora semplicemente si avvii Windows e lo si lasci configurare il PnP. Poi si avvi Linux da Windows (o da DOS). È stato segnalato che Windows cancella tutti gli IRQ dai registri dei dispositivi PCI. Allora Linux segnala questa cosa dicendo che trova un IRQ zero. Quindi si potrebbe non essere in grado di usare questo metodo.

5.8 Configurazione fatta dal device driver

Alcuni gestori di dispositivo useranno i metodi PnP per impostare le risorse-bus nell'hardware ma solo per i dispositivo che controllano. Poiché la configurazione l'ha fatta il driver, ovviamente quest'ultimo conosce la configurazione e non è necessario fornirgli queste informazioni.

Il problema in questo caso è duplice. È difficile incorporare tutto questo nel driver, e il driver potrebbe prendere le risorse-bus di cui ha bisogno da altri dispositivi. Rende le cose facili per l'utente, ma un kernel Linux PnP sarebbe migliore. Si veda 3.5 (Linux Ha Bisogno Di Tener Testa Meglio al PnP).

5.9 Software e documentazione sul PnP

- *homepage degli Isapnptools* `http://www.roestock.demon.co.uk/isapnptools/`
- *Patch per rendere il kernel Linux PnP* `http://www.astarte.free-online.co.uk`
- *Progetto PnP driver* `http://www.io.com/~cdb/mirrors/lpsg/pnp-linux.html`
- *Specifiche sul PnP della Microsoft* `http://www.microsoft.com/hwdev/respec/pnpspecs.htm`
- Libro: PCI System Architecture, 3rd ed. di Tom Shanley +, MindShare 1995. Tratta le caratteristiche simili al PnP del bus PCI.
- Libro: Plug and Play System Architecture, di Tom Shanley, Mind Share 1995. Dettagli del PnP sul bus ISA. Solo una concisa panoramica del PnP sul bus PCI.
- Libro: Programming Plug and Play, di James Kelsey, Sams 1995. Dettagli sulla programmazione per comunicare con un BIOS PnP. Tratta i bus ISA, PCI e PCMCIA.

6 Dire la configurazione al driver

6.1 Introduzione

Come ciò è fatto dipende dal driver. Alcuni hanno più di un modo per scoprire come sono configurati i lori dispositivi fisici. Ad un estremo c'è il caso nel quale si devono codificare permanentemente le risorse-bus nel kernel e ricompilarlo. All'altro estremo, i driver fa tutto automaticamente e quindi non si deve fare niente. Potrebbe pure impostare le risorse-bus nell'hardware usando i metodi PnP.

In mezzo ci sono i casi nei quali si esegue un programma per fornire le informazioni sulle risorse al driver oppure si mettono tali informazioni in un file. In alcuni casi i driver potrebbe cercare i dispositivo agli indirizzi ai quali sospetta che questo risieda. Poi può provare a verificare diversi IRQ per vedere quale funziona. Può o meno fare queste cose automaticamente. In altri casi il driver può usare i metodi PnP per trovare il dispositivo e scoprire come sono state impostate le risorse-bus, ma non le imposterà lui stesso. Può pure guardare in alcuni file nella directory /proc.

Può essere necessario fornire le risorse-bus come parametro al kernel o a un modulo caricabile. Si veda /usr/lib/modules_help/descr.gz per un'elenco dei possibili parametri. Il modulo da caricare è elencato in

/etc/modules assieme con i propri parametri. In alcuni altri casi le risorse-bus possono essere date come parametri al kernel. Sono messe nel file `lilo.conf` come `append=...`. Poi dev'essere eseguito il programma lilo per salvare nel codice di avvio del kernel.

Sebbene ci sia una grande disuniformità su come i driver scoprono le informazioni sulle risorse-bus, il risultato finale è lo stesso. Ci sono così tanti dispositivi hardware diversi e driver per essi che è meglio guardare la documentazione per il proprio driver per capire come si informa sulle risorse-bus e cosa si deve fare per assicurasi che riceva le informazioni di cui ha bisogno. Nelle sezioni che seguono sono presentate brevi informazioni su alcuni gestori.

6.2 Il driver delle porte seriali: setserial

Per configurare il gestore della porta seriale standard (non per le schede multiporta) si usa setserial. Spesso è eseguito da uno dei file di inizializzazione. Nelle versioni più recenti c'è un file /etc/serial.conf che si può «editare» semplicemente usando in modo normale il comando setserial e quel che si imposta usando `setserial` è salvato nel file di configurazione `serial.conf`. Il file `serial.conf` dovrebbe essere consultato quando il comando `setserial` è eseguito da un file di inizializzazione. La popria distribuzione potrebbe o meno aver già impostato questa cosa.

Ci sono due modi diversi di usare `setserial` a seconda delle opzioni che gli si dà. Un modo è di usarlo per dire manualmente al driver la configurazione. L'altro è di indigare ad un indirizzo specificato e segnalare se là esiste una porta seriale. Può pure indigare a questo indirizzo e provare a rilevare quale IRQ è usato per questa porta. Il driver esegue qualcosa di simile a `setserial` all'avvio ma rileva gli IRQ, assegna semplicemente gli IRQ «predefiniti» che potrebbe pure essere sbagliati. Controlla invece l'esistenza di un aporta. Si veda il Serial-HOWTO per maggiori dettagli.

6.3 Gestori della scheda audio

6.3.1 OSS-Lite

Si deve fornire IO, IRQ e DMA come parametri al modulo oppure compilare il supporto all'interno del kernel fissando i valori. Alcune schede PCI saranno rilevate automaticamente (basta usare il comando lspci o equivalente). La RedHat fornisce un programma «sndconfig» che rileva le schede PnP ISA e configura automaticamente i moduli da caricare con le risorse-bus rilevate.

6.3.2 OSS (Open Sound System) e ALSA

Questi rileveranno la scheda tramite i metodi PnP e poi selezioneranno il driver appropriato e lo caricheranno. Imposteranno anche le risorse-bus su una scheda PnP ISA. Può essere necessario intervenire manualmente per evitare conflitti. Nel driver ALSA, il supporto per il PnP ISA è opzionale e se si vuole si possono usare gli isapnptools.

7 Qual è la mia configurazione corrente?

Qui «configurazione» indica l'assegnazione delle risorse-bus PnP (indirizzi, IRQ e DMA). Per ogni dispositivo questa domanda può essere vista da due parti. Da ogni parte dovrebbe avere la stessa risposta.

1. Qual è la configurazione del software di gestione del dispositivo? ovvero: Il driver come pensa sia configurato l'hardware?

2. Quale configurazione (se esiste) è impostata nel dispositivo hardware?

Naturalmente la configurazione del dispositivo hardware e del suo driver dovrebbe essere la stessa (e normalmente lo è). Ma se le cose non vanno per il verso giusto, ci possono essere delle differenze. Ciò significa che il driver ha informazioni incorrette sulla configurazione reale dell'hardware. E questo può essere un problema. Se il software che si usa non spiega adeguatamente cosa c'è che non va (oppure non configura tutto automaticamente) allora bisogna investigare per capire come sono configurati i dispositivi hardware e i relativi driver. Sebbene i device driver di Linux dovrebbero «dire tutto» in alcuni casi non è semplice determinare che cosa è stato impostato nell'hardware.

Un altro problema è che quando si vede un messaggio di configurazione sullo schermo, qualche volta non è chiaro se la configurazione riportata è quella del driver del dispositivo, del dispositivo stesso o di entrambi. Se al driver del dispositivo è assegnata una configurazione e poi questo verifica nell'hardware per vedere se è configurato nello stesso modo, allora la configurazione riportata dovrebbe essere quella sia dell'hardware che del driver.

Ma alcuni driver non lo farlo in quanto possono accettare una configurazione senza verificarla. Per esempio, «setserial» accetta una configurazione senza verificarla (anche se gli si dice di rilevare le risorse-bus). Quindi «setserial» può solamente dire la configurazione del driver e non quella dell'hardware.

7.1 Messaggi all'avvio

Alcune informazioni sulla configurazione possono essere ottenute leggendo i messaggi del BIOS e di Linux che appaiono sullo schermo quando inizialmente si avvia il computer. Spesso questi messaggi scorrono troppo velocemente per poterli leggere, ma dopo che si sono fermati si prema alcune volte Shift+PageUp per vedere quelli passati. Digitando in qualsiasi momento «dmesg» al prompt della shell verranno mostrati solo i messaggi del kernel di Linux e si perderanno alcuni dei più importanti (tar cui quelli del BIOS). I messaggi di Linux qualche volta possono mostrare solamente quella che il driver di dispositivo pensa sia la configurazione, forse specificategli attraverso un file di configurazione sbagliato.

I messaggi del BIOS mostreranno la configurazione hardware in quel momento, ma un SO PnP, isapnp o le utility PCI possono successivamente modificarla. I messaggi del BIOS sono mostrati prima di quelli di Linux. Come alternativa all'uso alla fine di Shift+PageUp per leggerli, si provi a congelarli premento il tasto «Pause». Si prema un qualsiasi altro tasto per riprendere. Ma non appena iniziano ad apparire i messaggi di Linux è già troppo tardi per usare «Pause» in quanto non congelerà i messaggi di Linux.

7.2 Come sono configurati i miei driver di dispositivo?

Per determinarlo ci potrebbe essere un programma che si può eseguire da riga di comando (come «setserial» per le porte seriali). La directory /proc torna molto utile. /proc/ioports mostra gli indirizzi I/O che usano i driver (o provano ad usare se errati). Tali indirizzi possono non essere impostati nello stesso modo nell'hardware.

/proc/interrupts mostra solamente gli interrupt correntemente in uso. Molti di quelli che sono stati allocati ai driver non sono mostrati in quanto non attualmente in uso. Per esempio, anche se il proprio lettore di dischetti ha un dischetto a suo interno ed è pronto all'uso, il suo interrupt non sarà mostrato finché non lo si usa. Ancora, solo perché qui viene mostrato un interrupt non significa che esiste nell'hardware. Un traccia evidente che non esiste nell'hardware si avrà se sarà mostrato che per questo interrupt sono avvenute 0 interruzioni. Anche se mostra che alcuni interrupt hanno ricevuto alcune interruzioni, può significare che tale interrupt non esiste nel dispositivo mostrato a fianco, ma esiste piuttosto in qualche altro dispositivo non in uso, ma che in qualche modo ha ricevuto un paio di interruzioni. Nel kernel 2.2 l'albero /proc è stato cambiato.

7.3 Come sono configurati i miei dispositivi hardware?

Con il comando «lspci» è facile vedere quali risorse-bus sono state assegnate ad un dispositivo nel bus PCI. Per i kernel precedenti al 2.2: si veda `/proc/pci` o `/proc/bus/pci`. Si noti che gli IRQ in `/proc/pci` sono in esadecimale. Non si perda tempo a tentare di decifrare `/proc/bus/pci/devices` poiché «lspci» lo farà per voi.

Per il bus ISA si può provare ad eseguire `pnpdump -dumpregs` ma non è una cosa sicura. I risultati possono sembrare un po' criptici ma possono essere decifrati. Non si confornda l'indirizzo della read-port al quale `pnpdump` «prova» (e trova qualcosa) con l'indirizzo I/O del dispositivo trovato. Non sono la stessa cosa. Per provare a trovare l'hardware su bus ISA che manca (sia questo PnP o legacy) si provi il programma «scanport» ma si faccia attenzione perché potrebbe bloccare la macchina.

I messaggi del BIOS all'avvio del sistema dicono com'era allora la configurazione. Se ci si affida al BIOS per la configurazione, allora dovrebbe essere ancora la stessa. I messaggi di Linux possono essere generati dai driver che hanno controllato per vedere se l'hardware è effettivamente lì (e possibilmente verificato IRQ e DMA). Naturalmente, se il dispositivo funziona correttamente, allora probabilmente è configurato nello stesso modo del driver.

8 Appendice

8.1 Indirizzi

Esistono tre tipi di indirizzi: indirizzi nella memoria principale, indirizzi I/O e indirizzi di configurazione. Gli indirizzi di configurazione nel bus PCI costituiscono uno spazio di indirizzi separato proprio come fanno gli indirizzi I/O. Tranne per il caso complicato degli indirizzi di configurazione ISA, dove il fatto che un indirizzo nel bus sia o meno un indirizzo di memoria, un indirizzo di I/O o un indirizzo di configurazione dipende solo dal potenziale di tensione in altre linee (tracce) del bus.

8.1.1 Indirizzi di configurazione del bus ISA (Read-Port, ecc.)

Nel bus ISA, tecnicamente non c'è uno spazio di indirizzi di configurazione, ma c'è un modo speciale con il quale la CPU accede ai registri di configurazione PnP sulle schede PnP. Per questo scopo sono allocati 3 indirizzi di I/O. Non sono allocati 3 indirizzi per ognuna delle schede, ma 3 indirizzi condivisi da tutte le schede.

Questi 3 indirizzi sono chiamati read-port (porta di lettura), write-port (porta di scrittura) e address-port (posta di indirizzo). Ogni porta ha la dimensione di un solo byte. Ogni scheda PnP ha molti indirizzi di configurazione cosicché solamente questi 3 indirizzi non sono nemmeno sufficienti per i registri di una sola scheda. Per comunicare con una certa schede, è inviato a tutte le schede nella write-port un numero speciale assegnato alla scheda (handle). Dopo di che la sola scheda che rimane in ascolto è quella con quell'handle. Allora l'indirizzo del registro di configurazione (di quella scheda) è inviato nella address-port (di tutte le schede – ma solamente una è in ascolto). La comunicazione successiva inizia con un registro di configurazione su quella scheda leggendolo dalla read-port oppure scrivendolo nella write-port.

La write-port è sempre a A79 e la address-port è sempre a 279 (hex). La read-port invece non è fissata ma è impostata da software di configurazione ad un qualche indirizzo che si suppone non entri in conflitto con nessun'altra scheda ISA. Se c'è un conflitto, cambierà l'indirizzo. Tutte le schede PnP saranno poi «programmate» con questo indirizzo. Quindi se diciamo si usa isapnp per impostare o verificare i dati della configurazione si deve conoscere l'indirizzo di questa read-port.

8.1.2 Intervalli di indirizzi

Talvolta in questo documento il termine «indirizzi» è usato per intendere un intervallo contiguo di indirizzi. Poiché gli indirizzi sono dati in byte, un indirizzo singolo contiene solamente un byte, mentre l'I/O e la memoria principale ne hanno bisogno di molti di più. È quindi spesso usato un intervallo di diciamo 8 byte per gli indirizzi I/O mentre l'intervallo di indirizzi in memoria principale allocato ad un dispositivo è molto maggiore. Per una porta seriale (un dispositivo di I/O) è sufficiente specificare l'indirizzo di partenza dell'indirizzo I/O del dispositivo (ad esempio 3F8) in quanto è ben noto che l'intervallo di indirizzi per una porta seriale è di soli 8 byte. L'indirizzo di partenza è noto come «base address» (indirizzo base).

8.1.3 Spazio di indirizzi

In ISA, per accedere sia allo «spazio» di indirizzi I/O che a quello di memoria (principale) è usato lo stesso bus indirizzi (address bus) (le linee usate nel bus sono condivise). Come fa un dispositivo a sapere quando un indirizzo che appare sull'address bus è un indirizzo di memoria o un indirizzo I/O? Beh, ci sono 4 linee dedicate sul bus che trasportano questa informazione ed altro. Se una ben determinata di queste 4 linee è «attiva», dice che la CPU vuole leggere da un indirizzo I/O e quindi la memoria principale ignora l'indirizzo sul bus. Le altre 3 linee servono per scopi analoghi. In breve: esistono linee di lettura e di scrittura sia per gli indirizzi di memoria principale che per quelli di I/O (4 linee in tutto).

Nel bus PCI c'è la stessa idea di base, che usa sempre 4 linee, ma il tutto viene fatto in maniera un po' diversa. Invece di essere «attiva» solo una delle 4 linee, in queste viene posto un numero binario (16 diverse possibilità). Quattro di questi 16 numeri servono gli spazi di I/O e di memoria come visto prima. Inoltre c'è anche uno spazio di indirizzi di configurazione che usa altri 2 numeri. Gli altri 10 numeri extra sono lasciati per altri scopi.

8.1.4 Verifica dell'intervallo (Test ISA per i conflitti di indirizzo I/O)

Nel bus ISA, c'è un metodo costruito dentro ogni scheda PnP per verificare che non ci siano altre schede che usano gli stessi indirizzi. Se due o più schede usano gli stessi indirizzi I/O, probabilmente neanche una scheda funzionerà bene (se non tutte). Un buon software di PnP dovrebbe assegnare le risorse-bus in modo da evitare i conflitti, ma anche in questo caso una scheda legacy potrebbe nascondersi da qualche parte lo stesso indirizzo.

Il test inizia con una scheda che mette un numero di controllo nei proprio registri I/O. Successivamente il software PnP lo legge e verifica di aver letto lo stesso numero di controllo. Se non è vero, qualcosa è andato storto (ad esempio un'altra scheda con lo stesso numero di controllo). Ripete allora lo stesso test con un altro numero di controllo. Poiché in realtà verifica l'intervallo di indirizzi I/O assegnati alla scheda, è detto «range check» (verifica di intervallo). Un nome migliore potrebbe essere verifica dei conflitti di indirizzo. Se c'è un conflitto di indirizzo si ottiene un messaggio d'errore e lo si deve risolvere per proprio conto.

8.1.5 Comunicare direttamente attraverso la memoria

Tradizionalmente, la maggior parte dei dispositivi I/O usano solamente la memoria I/O per comunicare con la CPU. Per esempio, la porta seriale lo fa. Il device driver, in esecuzione nella CPU, vorrebbe leggere e scrivere dati sia dallo/nello spazio di indirizzi I/O che dalla/nella memoria principale. Il modo più veloce si avrebbe se il dispositivo stesso mettesse i dati direttamente nella memoria principale. Un modo per farlo è usare i 2.5 (Canali DMA) o il «bus mastering». Un altro modo è di allocare al dispositivo un po' di spazio in memoria principale. In questo modo il dispositivo legge e scrive direttamente nella memoria principale senza preoccuparsi di DMA e bus mastering. Tale dispositivo può usare anche indirizzi I/O.

8.2 Interrupt – Dettagli

Gli interrupt trasportano un sacco di informazioni, ma solo indirettamente. Il segnale di interrupt (una tensione in una linea) dice semplicemente ad un chip chiamato interrupt controller («controllore delle interruzioni») che un certo dispositivo chiede un po' di attenzione. L'interrupt controller allora lo segnala alla CPU. La CPU trova il driver per questo dispositivo e esegue una parte di esso nota come «interrupt service routine» (routine di servizio dell'interrupt) o «interrupt handler». Questa «routine» prova a capire cos'è successo e poi si occupa del problema, come il trasferimento di dati da (o nel) dispositivo. Questo programma (routine) può facilmente scoprire cos'è successo in quanto il dispositivo ha dei registri ad un indirizzo noto al software di gestione (premesso che le informazioni sul numero IRQ e l'indirizzo di I/O siano impostate correttamente). Questi registri contengono informazioni sullo stato del dispositivo. Il software legge il contenuto di questi registri e ispezionandone il contenuto, scopre cos'è successo e intraprende l'azione appropriata.

Quindi ogni device driver ha bisogno di sapere su quale numero di interrupt (IRQ) restare in ascolto. Nel bus PCI (e per le porte seriali sul bus ISA a partire dal kernel 2.2) è permesso che due (o più) dispositivi condividano lo stesso numero IRQ. Quando avviene tale interrupt, la CPU esegue la routine di servizio dell'interruzione per tutti questi dispositivi che usano quell'interrupt. La prima cosa che fa la prima delle routine è di controllare per vedere se l'interrupt è effettivamente avvenuto per il suo dispositivo. Se non c'erano interrupt (falso allarme) la routine termina e parte la successiva routine di servizio, ecc.

8.3 Interrupt PCI

Gli interrupt PCI sono diversi, ma poiché solitamente sono mappati su IRQ si comportano quasi nello stesso modo. Una differenza sostanziale è che gli interrupt PCI possono essere condivisi. Per esempio l'IRQ5 può essere condiviso da due dispositivi PCI. Questa funzionalità di condivisione è automatica: non è necessario hardware o software speciale. Si sono avute notizie di situazioni nelle quali questa condivisione non funzionava, ma solitamente era dovuto a difetti nel software di gestione del dispositivo. Si suppone che tutti i device driver per i dispositivi PCI forniscano la condivisione degli interrupt. Si noti che non è possibile condividere lo stesso interrupt tra il bus ISA e il bus PCI. Comunque, le condivisioni illegali funzioneranno a patto che i dispositivi in conflitto non siano in uso nello stesso istante. «In uso» qui significa che un è in esecuzione un programma che «apre» il device nel suo codice C di programmazione.

È necessario conoscere alcuni dettagli del sistema di interrupt PCI per poter impostare il CMOS del BIOS o i ponticelli nelle vecchie schede PCI. Ogni scheda PCI ha 4 possibili interrupt: INTA#, INTB#, INTC# e INTD#. Quindi per un sistema a 7 slot ci possono essere 7 x 4 = 28 diverse linee di interrupt. Ma le specifiche permettono un numero minore di linee di interrupt. Questo non è troppo restrittivo in quanto gli interrupt possono essere condivisi. Molti bis PCI sembrano essere fatti con solo 4 linee di interrupt. Chiamiamole linee (connessioni o tracce) W, X, Y e Z. Supponiamo di progettare l'interrupt B dello slot 3 come interrupt 3B. Allora la linea W potrebbe essere usato per condividere gli interrupt 1A, 2B, 3C, 4D, 5A, 6B e 7C. Ciò è fatto connettendo fisicamente la linea W alla linee 1A, 2B, ecc. Nello stesso modo la linea X potrebbe essere connessa alle linee 1B, 2C, 3D, 4A, 5B, 6C e 7D. Poi, all'avvio, il BIOS mappa le linee X, W, Y e Z su IRQ. Dopodiché scrive l'IRQ sul quale è mappato ogni dispositivo dentro un registro hardware presente su ogni dispositivo. Ora chiunque interroghi il dispositivo può trovare quale IRQ usa.

Le summenzionate linee X, W, Y e Z sono etichettate nelle specifiche PCI come INTA#, INTB#, INTC# e INTD#. Questa notazione PCI ufficiale può confondere in quanto ora INTA# ha due possibili significati a seconda si stia parlando di uno slot o del bus PCI. Per esempio, se 3C è mappato su X allora diciamo che l'INTC# dello slot 3 è connesso a INTA# (X) del bus PCI. Notazione piuttosto confusa.

Inoltre c'è un altro vincolo. Uno slot PCI deve usare per primo l'interrupt con la lettera più bassa. Quindi se una slot usa solo un interrupt, deve essere INTA#. Se ne usa due devono essere INTA# e INTB#, ecc. Una scheda in uno slot può avere su fino ad 8 dispositivi, ma per questi ci sono solo 4 interrupt PCI. Ciò va bene lo stesso in quanto gli interrupt possono essere condivisi in modo tale che ognuno degli 8 dispositivi (se esistono) può avere un interrupt. La lettera dell'interrupt PCI di un dispositivo spesso è fissa e scritta nell'hardware del dispositivo.

Il BIOS assegna IRQ (interrupt) in modo da evitare i conflitti con gli IRQ che sa sono già assegnati sul bus ISA. Talvolta nel menu CMOS del BIOS è possibile assegnare gli IRQ alle schede PCI (ma non è così semplice come spiegato sopra). Ci sono situazioni nelle quali Windows azzera tutti i numeri IRQ delle schede PCI dopo l'impostazione della mappatura degli IRQ. Se poi qualcuno avvia Linux da Windows ottiene come risultato che Linux può trovare solo IRQ nulli e quindi sbagliati.

Si potrebbe pensare che poiché il PCI sta usando gli IRQ (il bus ISA) potrebbe essere lento, ecc. Non in realtà. I chip ISA di controllo degli interrupt posseggono linee dirette di interrupt che vanno alla CPU in modo da ottenere attenzione immediata. Mentre i segnali nei bus ISA di dati e indirizzi devono passare per il bus PCI per ottenere la CPU, i segnali di IRQ ci vanno direttamente.

8.4 Isolamento

Questa cosa è pertinente solo con il bus ISA. L'isolamento («isolation») è un complesso metodo per assegnare un handle temporaneo (numero identificativo o Card Select Number = CSN) ad ogni dispositivo PnP presente nel bus ISA. Sebbene esistano modi molto più efficienti (ma più complessi) per fare questa cosa, alcuni dicono che questo è un metodo semplice. È usato solo un indirizzo di scrittura per tutte le scritture PnP a tutti i dispositivi PnP, cosicché quello che viene scritto in questo indirizzo va a tutti i dispositivi PnP che sono in ascolto. Questo indirizzo di scrittura è usato per inviare (assegnare) un handle univoco ad ognuno dei dispositivi PnP. Questa assegnazione richiede che solo un dispositivo sia in attesa quando è inviato (scritto) l'handle in questo indirizzo comune. Tutti i dispositivi PnP hanno un numero di serie univoco che usano per il processo di isolamento. Fare l'isolamento è qualcosa di simile ad un gioco. È fatto usando l'equivalente di una sola linea del bus che collega tutti i dispositivi PnP e il programma di isolamento.

Nel primo giro di questo «gioco» tutti i dispositivi PnP sono ascolto su questa linea e inviano simultaneamente una sequenza di bit nella linea. I bit permessi sono un «1» (tensione positiva) oppure uno «0 aperto» di nessuna tensione (circuito aperto e tri-state). Ogni dispositivo PnP inizia semplicemente ad inviare sequenzialmente su questa linea il suo numero di serie, bit a bit, iniziando con il bit più significativo. Se un qualsiasi dispositivo, invia un 1, un 1 sarà sentito sulla linea da tutti gli altri dispositivi. Se tutti i dispositivi inviano uno «0 aperto» nella linea non si sentirà niente. L'obbiettivo è di eliminare (alla fine di questa prima tornata) tutti i dispositivi tranne quello con il numero di serie più elevato. «Eliminato» significa che cessa restare in ascolto all'indirizzo di scrittura al quale restano in ascolto tutti i dispositivi ancora in gioco. Ciò è detto anche «dropping out». (Si noti che tutti i numeri di serie sono della stessa lunghezza.)

Per prima cosa si consideri solo il bit più significativo del numero di serie che viene posto per primo sulla linea da tutti i dispositivi che non hanno ancora un handle. Se un qualsiasi dispositivo PnP invia uno 0 (0 aperto) ma sente un 1, questo significa che qualche altro dispositivo PnP ha un numero di serie più alto, e quindi temporaneamente si toglie dal gioco e non ascolta più finché il giro non è terminato (ovvero quando viene assegnato un handle al vincitore: il dispositivo con il numero di serie più alto). Ora i dispositivi ancora in gioco hanno tutti il medesimo bit più significativo (un 1), e quindi, per la partecipazione futura a questo giro, possiamo trascurare questa cifra e considerare solo la parte restante del numero di serie. Adesso si torni all'inizio di questo paragrafo e lo si ripeta finché per tutti i dispositivi non siano stati esaminati completamente i numeri di serie (si veda sotto per il caso di tutti 0).

Quindi è chiaro che il numero di serie più alto non verrà eliminato dal gioco. Ma cosa succede se le cifre più significative (anche nel caso dei numeri di serie ridotti) sono tutte 0? In questo caso è stato inviato uno «0 aperto» nella linea e tutti i partecipanti rimangono in gioco. Se tutti hanno uno 0 come cifra più significativa allora gli 0 sono trascurati come è stato fatto nel paragrafo precedente con gli 1. Il gioco continua inviando la cifra successiva del numero di serie.

Alla fine di questa tornata (dopo che ogni partecipante rimasto ha inviato il bit meno significato del numero di serie) rimane solo il dispositivo PnP con il numero di serie più elevato. A questo viene assegnato un handle ed esce permanentemente dal gioco. Poi tutti quelli scartati precedentemente rientrano in gioco e inizia un nuovo giro con un partecipante in meno. Alla fine, viene assegnato un handle a ciascun dispositivo PnP. È facile vedere che questo algoritmo funziona.

Una volta assegnati, gli handle sono usati per indirizzare ogni dispositivo PnP, sia per potergli inviare una configurazione sia per leggere le informazioni di configurazione dal dispositivo stesso. Si noti che questi handle sono usati solo per la configurazione PnP e non sono usati per le normali comunicazioni con il dispositivo. Quando il computer viene avviato, tutti gli handle sono stati persi e quindi il BIOS PnP solitamente opera il processo di isolamento ogni qualvolta si riavvia il proprio PC.

FINE DEL Plug-and-Play-HOWTO