Writing Device Drivers Under Linux

SCSI Device Drivers

Introduction to SCSI

- Small Computer System Interface
- Efficient I/O bus for up to 8 devices
- Transfer rates up to 40 Mb/sec
- Industry standard for high-performance disk, tape, etc. interfaces

Copyright 1997 by Theodore Ts'o

SCSI bus

- Each device on the SCSI bus as a SCSI ID
- Typically the computer is SCSI ID #7
- Each device on the SCSI bus is connected together in a linear chain
- Each end of the SCSI chain must have a terminator

SCSI bus magic

- "SCSI is **not** magic. There are **fundamental technical reasons** why you have to sacrifice a young goat to your SCSI chain every now and then."
  - Bus length limitation (5 meters)
  - SCSI cable quality
  - Termination issues

Copyright 1997 by Theodore Ts'o

Why use SCSI?

- Faster than IDE/ATAPI interface
  - Higher transfer rates
  - Can support multiple simultaneous operations
- Industry standard
  - Unix workstation market
  - High-end PC market

SCSI Bus Phases

- BUS FREE
  - The SCSI bus is idle and unused
- ARBITRATION
  - How a SCSI device gain control of the SCSI bus
  - The SCSI bus must have been in the BUS FREE state

Copyright 1997 by Theodore Ts'o

SCSI Bus Phases II

- SELECTION
  - The initiator asserts the SCSI ID of the target
- RESELECTION
  - Like SELECTION, but used when the target connects to the initiator
  - Used after initiator disconnects from the target
  - Both initiator and target must support disconnection

SCSI Bus Phases III

- COMMAND
  - The initiator sends a command to the target
- DATA OUT/DATA IN
  - During these phases, data is transfered between initiator and target
  - Only entered if SCSI command required data transfer

Copyright 1997 by Theodore Ts'o

SCSI Bus Phases IV

- STATUS
  - This phase entered after all commands
  - Target sends a status byte to initiator
- MESSAGE OUT/MESSAGE IN
  - Additional information transfer
  - SCSI disconnection uses the messages SAVE DATA POINTERS, RESTORE POINTERS and DISCONNECT

SCSI Commands

- Issued by computer (the initiator)
- Directed at a SCSI device
- Example SCSI commands
  - READ, WRITE
  - TEST UNIT READY
  - REQUEST SENSE

Copyright 1997 by Theodore Ts'o

What you will need to write a SCSI driver

- A bootable Linux system
- Technical manual for your SCSI host adapter
- SCSI-II standard (optional)
  - ANSI X3.131-1994
- USENET comp.periphs.scsi FAQ (optional)
- Printouts of scsi.h and hosts.h

Simple or complex?

- SCSI protocol is very complex
- Fortunately, the Linux SCSI layer handles most of the complexity
- Functional requirements of low-level driver thus can be very simple

Copyright 1997 by Theodore Ts'o

A SCSI driver is Simple

- Operations required of the device driver
  - Device initialization
  - Send a command to the SCSI bus
  - Read status from completed commands
  - Perform SCSI abort and SCSI resets as requested
  - Get BIOS geometry

A SCSI driver is Complex!

- Many optional parameters for efficiency
  - Scatter-gather lists
  - Multiple outstanding commands
- Hardware interface difficulties
  - ISA, DMA, and 16Mb limits
  - Communicating with the host adapter

Copyright 1997 by Theodore Ts'o

Example SCSI driver

- See aha1740.c (page 112 in the Example Sources)
- For the Adaptec 1740
- Simple and relatively clearly written
- EISA bus, so no 16mb DMA limits

SCSI Host Template

- Interface between driver and SCSI layer
- See aha1740.c:548 and aha1740.h:177
- Definition of SCSI Host Template struct is on hosts.h:55
- Must add entry to drivers/scsi/hosts.c for non-module drivers

Copyright 1997 by Theodore Ts'o

aha1740_command()

- See aha1740.c:443
- Older interface which does not return until the SCSI command is finished.
- Front-end for aha1740_queuecommand()

aha1740_queuecommand()

- See aha1740.c:265
- Fills in an Adaptec-specific Enhanced Control Block (ECB) block
- Sends the ECB to the SCSI Controller
- When a SCSI command completes, the controller signals an interrupt

Copyright 1997 by Theodore Ts'o

aha1740_intr_handle()

- See aha1740.c:180
- Checks to see if results from a SCSI command are available
- Gets results, and calls the done() function

aha1740_abort()

- See aha1740.c:518
- Not implemented by this driver
- Called by the high-level SCSI if a SCSI command takes too long
- Adaptec card has its own timeouts

Copyright 1997 by Theodore Ts'o

aha1740_reset()

- See aha1740.c:529
- Not implemented by this driver
- It should cause a SCSI bus reset

aha1740_biosparam()

- See aha1740.c:535
- Returns the version of the geometry as reported by the Adaptec BIOS
- Required for fdisk
  - Logical vs. physical geometry

Copyright 1997 by Theodore Ts'o

aha17440_detect()

- See aha1740.c:463
- Tries to see if the card is present in the system
- If present, find its IRQ

SCSI controller types

- "Smart" controllers with ma
  - Examples: aha1740, aha154x
  - Simplest to progra
- "Dumb" controllers
  - Example: aha152x
  - Device Driver has to manage the SCSI bus

Copyright 1997 by Theodore Ts'o

SCSI controller types II

- Controllers w/ microcode
  - Examples: AIC-7xxx, NCR53C8XX
  - Like "smart" controllers, but you have to supply the firmware
  - More room for customization
  - Detailed technical specs required!

January 7, 1997 SCSI Device Drivers 25