```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE book PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"
        "http://www.oasis-open.org/docbook/xml/4.1.2/docbookx.dtd" []>

<book id="scsimid">
  <bookinfo>
    <title>SCSI Interfaces Guide</title>

    <authorgroup>
      <author>
        <firstname>James</firstname>
        <surname>Bottomley</surname>
        <affiliation>
          <address>
            <email>James.Bottomley@hansenpartnership.com</email>
          </address>
        </affiliation>
      </author>

      <author>
        <firstname>Rob</firstname>
        <surname>Landley</surname>
        <affiliation>
          <address>
            <email>rob@landley.net</email>
          </address>
        </affiliation>
      </author>

    </authorgroup>

    <copyright>
      <year>2007</year>
      <holder>Linux Foundation</holder>
    </copyright>

    <legalnotice>
      <para>
        This documentation is free software; you can redistribute
        it and/or modify it under the terms of the GNU General Public
        License version 2.
      </para>

      <para>
        This program is distributed in the hope that it will be
        useful, but WITHOUT ANY WARRANTY; without even the implied
        warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
        For more details see the file COPYING in the source
        distribution of Linux.
      </para>
    </legalnotice>
  </bookinfo>

  <toc></toc>

  <chapter id="intro">
```

```
  <title>Introduction</title>
  <sect1 id="protocol_vs_bus">
    <title>Protocol vs bus</title>
    <para>
      Once upon a time, the Small Computer Systems Interface defined both
      a parallel I/O bus and a data protocol to connect a wide variety of
      peripherals (disk drives, tape drives, modems, printers, scanners,
      optical drives, test equipment, and medical devices) to a host
      computer.
    </para>
    <para>
      Although the old parallel (fast/wide/ultra) SCSI bus has largely
      fallen out of use, the SCSI command set is more widely used than ever
      to communicate with devices over a number of different busses.
    </para>
    <para>
      The <ulink url='http://www.t10.org/scsi-3.htm'>SCSI protocol</ulink>
      is a big-endian peer-to-peer packet based protocol.  SCSI commands
      are 6, 10, 12, or 16 bytes long, often followed by an associated data
      payload.
    </para>
    <para>
      SCSI commands can be transported over just about any kind of bus, and
      are the default protocol for storage devices attached to USB, SATA,
      SAS, Fibre Channel, FireWire, and ATAPI devices.  SCSI packets are
      also commonly exchanged over Infiniband,
      <ulink url='http://i2o.shadowconnect.com/faq.php'>I20</ulink>, TCP/IP
      (<ulink url='http://en.wikipedia.org/wiki/ISCSI'>iSCSI</ulink>), even
      <ulink url='http://cyberelk.net/tim/parport/parscsi.html'>Parallel
      ports</ulink>.
    </para>
  </sect1>
  <sect1 id="subsystem_design">
    <title>Design of the Linux SCSI subsystem</title>
    <para>
      The SCSI subsystem uses a three layer design, with upper, mid, and low
      layers.  Every operation involving the SCSI subsystem (such as reading
      a sector from a disk) uses one driver at each of the 3 levels: one
      upper layer driver, one lower layer driver, and the SCSI midlayer.
    </para>
    <para>
      The SCSI upper layer provides the interface between userspace and the
      kernel, in the form of block and char device nodes for I/O and
      ioctl().  The SCSI lower layer contains drivers for specific hardware
      devices.
    </para>
    <para>
      In between is the SCSI mid-layer, analogous to a network routing
      layer such as the IPv4 stack.  The SCSI mid-layer routes a packet
      based data protocol between the upper layer's /dev nodes and the
      corresponding devices in the lower layer.  It manages command queues,
      provides error handling and power management functions, and responds
      to ioctl() requests.
    </para>
  </sect1>
</chapter>
```

```
  <chapter id="upper_layer">
    <title>SCSI upper layer</title>
    <para>
      The upper layer supports the user-kernel interface by providing
      device nodes.
    </para>
    <sect1 id="sd">
      <title>sd (SCSI Disk)</title>
      <para>sd (sd_mod.o)</para>
<!-- !Idrivers/scsi/sd.c -->
    </sect1>
    <sect1 id="sr">
      <title>sr (SCSI CD-ROM)</title>
      <para>sr (sr_mod.o)</para>
    </sect1>
    <sect1 id="st">
      <title>st (SCSI Tape)</title>
      <para>st (st.o)</para>
    </sect1>
    <sect1 id="sg">
      <title>sg (SCSI Generic)</title>
      <para>sg (sg.o)</para>
    </sect1>
    <sect1 id="ch">
      <title>ch (SCSI Media Changer)</title>
      <para>ch (ch.c)</para>
    </sect1>
  </chapter>

  <chapter id="mid_layer">
    <title>SCSI mid layer</title>

    <sect1 id="midlayer_implementation">
      <title>SCSI midlayer implementation</title>
      <sect2 id="scsi_device.h">
        <title>include/scsi/scsi_device.h</title>
        <para>
        </para>
!Iinclude/scsi/scsi_device.h
      </sect2>

      <sect2 id="scsi.c">
        <title>drivers/scsi/scsi.c</title>
        <para>Main file for the SCSI midlayer.</para>
!Edrivers/scsi/scsi.c
      </sect2>
      <sect2 id="scsicam.c">
        <title>drivers/scsi/scsicam.c</title>
        <para>
          <ulink url='http://www.t10.org/ftp/t10/drafts/cam/cam-r12b.pdf'>SCSI
          Common Access Method</ulink> support functions, for use with
          HDIO_GETGEO, etc.
        </para>
!Edrivers/scsi/scsicam.c
      </sect2>
```

```
      <sect2 id="scsi_error.c">
        <title>drivers/scsi/scsi_error.c</title>
        <para>Common SCSI error/timeout handling routines.</para>
!Edrivers/scsi/scsi_error.c
      </sect2>
      <sect2 id="scsi_devinfo.c">
        <title>drivers/scsi/scsi_devinfo.c</title>
        <para>
          Manage scsi_dev_info_list, which tracks blacklisted and whitelisted
          devices.
        </para>
!Idrivers/scsi/scsi_devinfo.c
      </sect2>
      <sect2 id="scsi_ioctl.c">
        <title>drivers/scsi/scsi_ioctl.c</title>
        <para>
          Handle ioctl() calls for SCSI devices.
        </para>
!Edrivers/scsi/scsi_ioctl.c
      </sect2>
      <sect2 id="scsi_lib.c">
        <title>drivers/scsi/scsi_lib.c</title>
        <para>
          SCSI queuing library.
        </para>
!Edrivers/scsi/scsi_lib.c
      </sect2>
      <sect2 id="scsi_lib_dma.c">
        <title>drivers/scsi/scsi_lib_dma.c</title>
        <para>
          SCSI library functions depending on DMA
          (map and unmap scatter-gather lists).
        </para>
!Edrivers/scsi/scsi_lib_dma.c
      </sect2>
      <sect2 id="scsi_module.c">
        <title>drivers/scsi/scsi_module.c</title>
        <para>
          The file drivers/scsi/scsi_module.c contains legacy support for
          old-style host templates.  It should never be used by any new driver.
        </para>
      </sect2>
      <sect2 id="scsi_proc.c">
        <title>drivers/scsi/scsi_proc.c</title>
        <para>
          The functions in this file provide an interface between
          the PROC file system and the SCSI device drivers
          It is mainly used for debugging, statistics and to pass
          information directly to the lowlevel driver.

          I.E. plumbing to manage /proc/scsi/*
        </para>
!Idrivers/scsi/scsi_proc.c
      </sect2>
      <sect2 id="scsi_netlink.c">
        <title>drivers/scsi/scsi_netlink.c</title>
```

```
        <para>
          Infrastructure to provide async events from transports to userspace
          via netlink, using a single NETLINK_SCSITRANSPORT protocol for all
          transports.

          See <ulink
url='http://marc.info/?l=linux-scsi&amp;m=115507374832500&amp;w=2'>the
          original patch submission</ulink> for more details.
        </para>
!Idrivers/scsi/scsi_netlink.c
      </sect2>
      <sect2 id="scsi_scan.c">
        <title>drivers/scsi/scsi_scan.c</title>
        <para>
          Scan a host to determine which (if any) devices are attached.

          The general scanning/probing algorithm is as follows, exceptions are
          made to it depending on device specific flags, compilation options,
          and global variable (boot or module load time) settings.

          A specific LUN is scanned via an INQUIRY command; if the LUN has a
          device attached, a scsi_device is allocated and setup for it.

          For every id of every channel on the given host, start by scanning
          LUN 0.  Skip hosts that don't respond at all to a scan of LUN 0.
          Otherwise, if LUN 0 has a device attached, allocate and setup a
          scsi_device for it.  If target is SCSI-3 or up, issue a REPORT LUN,
          and scan all of the LUNs returned by the REPORT LUN; else,
          sequentially scan LUNs up until some maximum is reached, or a LUN is
          seen that cannot have a device attached to it.
        </para>
!Idrivers/scsi/scsi_scan.c
      </sect2>
      <sect2 id="scsi_sysctl.c">
        <title>drivers/scsi/scsi_sysctl.c</title>
        <para>
          Set up the sysctl entry: "/dev/scsi/logging_level"
          (DEV_SCSI_LOGGING_LEVEL) which sets/returns scsi_logging_level.
        </para>
      </sect2>
      <sect2 id="scsi_sysfs.c">
        <title>drivers/scsi/scsi_sysfs.c</title>
        <para>
          SCSI sysfs interface routines.
        </para>
!Edrivers/scsi/scsi_sysfs.c
      </sect2>
      <sect2 id="hosts.c">
        <title>drivers/scsi/hosts.c</title>
        <para>
          mid to lowlevel SCSI driver interface
        </para>
!Edrivers/scsi/hosts.c
      </sect2>
      <sect2 id="constants.c">
        <title>drivers/scsi/constants.c</title>
```

```
        <para>
          mid to lowlevel SCSI driver interface
        </para>
!Edrivers/scsi/constants.c
      </sect2>
    </sect1>

    <sect1 id="Transport_classes">
      <title>Transport classes</title>
      <para>
        Transport classes are service libraries for drivers in the SCSI
        lower layer, which expose transport attributes in sysfs.
      </para>
      <sect2 id="Fibre_Channel_transport">
        <title>Fibre Channel transport</title>
        <para>
          The file drivers/scsi/scsi_transport_fc.c defines transport attributes
          for Fibre Channel.
        </para>
!Edrivers/scsi/scsi_transport_fc.c
      </sect2>
      <sect2 id="iSCSI_transport">
        <title>iSCSI transport class</title>
        <para>
          The file drivers/scsi/scsi_transport_iscsi.c defines transport
          attributes for the iSCSI class, which sends SCSI packets over TCP/IP
          connections.
        </para>
!Edrivers/scsi/scsi_transport_iscsi.c
      </sect2>
      <sect2 id="SAS_transport">
        <title>Serial Attached SCSI (SAS) transport class</title>
        <para>
          The file drivers/scsi/scsi_transport_sas.c defines transport
          attributes for Serial Attached SCSI, a variant of SATA aimed at
          large high-end systems.
        </para>
        <para>
          The SAS transport class contains common code to deal with SAS HBAs,
          an aproximated representation of SAS topologies in the driver model,
          and various sysfs attributes to expose these topologies and management
          interfaces to userspace.
        </para>
        <para>
          In addition to the basic SCSI core objects this transport class
          introduces two additional intermediate objects:  The SAS PHY
          as represented by struct sas_phy defines an "outgoing" PHY on
          a SAS HBA or Expander, and the SAS remote PHY represented by
          struct sas_rphy defines an "incoming" PHY on a SAS Expander or
          end device.  Note that this is purely a software concept, the
          underlying hardware for a PHY and a remote PHY is the exactly
          the same.
        </para>
        <para>
          There is no concept of a SAS port in this code, users can see
          what PHYs form a wide port based on the port_identifier attribute,
```

```
          which is the same for all PHYs in a port.
        </para>
!Edrivers/scsi/scsi_transport_sas.c
      </sect2>
      <sect2 id="SATA_transport">
        <title>SATA transport class</title>
        <para>
          The SATA transport is handled by libata, which has its own book of
          documentation in this directory.
        </para>
      </sect2>
      <sect2 id="SPI_transport">
        <title>Parallel SCSI (SPI) transport class</title>
        <para>
          The file drivers/scsi/scsi_transport_spi.c defines transport
          attributes for traditional (fast/wide/ultra) SCSI busses.
        </para>
!Edrivers/scsi/scsi_transport_spi.c
      </sect2>
      <sect2 id="SRP_transport">
        <title>SCSI RDMA (SRP) transport class</title>
        <para>
          The file drivers/scsi/scsi_transport_srp.c defines transport
          attributes for SCSI over Remote Direct Memory Access.
        </para>
!Edrivers/scsi/scsi_transport_srp.c
      </sect2>
    </sect1>

  </chapter>

  <chapter id="lower_layer">
    <title>SCSI lower layer</title>
    <sect1 id="hba_drivers">
      <title>Host Bus Adapter transport types</title>
      <para>
        Many modern device controllers use the SCSI command set as a protocol to
        communicate with their devices through many different types of physical
        connections.
      </para>
      <para>
        In SCSI language a bus capable of carrying SCSI commands is
        called a "transport", and a controller connecting to such a bus is
        called a "host bus adapter" (HBA).
      </para>
      <sect2 id="scsi_debug.c">
        <title>Debug transport</title>
        <para>
          The file drivers/scsi/scsi_debug.c simulates a host adapter with a
          variable number of disks (or disk like devices) attached, sharing a
          common amount of RAM.  Does a lot of checking to make sure that we are
          not getting blocks mixed up, and panics the kernel if anything out of
          the ordinary is seen.
        </para>
        <para>
          To be more realistic, the simulated devices have the transport
```

```
          attributes of SAS disks.
        </para>
        <para>
          For documentation see
          <ulink
url='http://www.torque.net/sg/sdebug26.html'>http://www.torque.net/sg/sdebug26.h
tml</ulink>
        </para>
<!-- !Edrivers/scsi/scsi_debug.c -->
      </sect2>
      <sect2 id="todo">
        <title>todo</title>
        <para>Parallel (fast/wide/ultra) SCSI, USB, SATA,
        SAS, Fibre Channel, FireWire, ATAPI devices, Infiniband,
        I20, iSCSI, Parallel ports, netlink...
        </para>
      </sect2>
    </sect1>
  </chapter>
</book>
```