This book is intended for individuals writing interface and device drivers for HP-UX Workstations and servers.

This chapter describes how the HP-UX I/O subsystem is structured, and how the driver fits into this environment.
How the I/O Subsystem is Structured

The I/O subsystem provides a uniform interface for user processes when reading information from and writing information to devices, providing the System Administrator with information about the system’s I/O hardware and software. It also provides an environment within which drivers can control I/O devices. This environment provides drivers with tools used by developers to do their work and contexts in which driver code can run.

Much of this environment is provided by kernel code; but configuring the system (associating drivers with devices) and setting the values of system parameters (tuning the system) also play their parts.

The HP-UX I/O system has two sections: a **General I/O System** (GIO), and one or more **Context-Dependent I/O** (CDIO) modules.

The GIO, which is always present, provides all the functionality that is global to the I/O system, and provides services the CDIOs can use.

CDIOs contain all bus-specific and device-specific functionality. A system administrator configures CDIOs into a system only as necessary.

**General I/O System (GIO)**

The GIO manages the system’s I/O resources and data structures, drives the system-configuration process, providing an interface to the system-administration utilities. GIO functionality includes:

- **Management of data structures used for I/O configuration.**
  - Data structures that can be manipulated by system-administration utilities or that are global to the system must be maintained by the GIO. These include:
    - I/O tree node
    - Block and character switch tables
    - **Kernel Device Table** (KDT)
  - The algorithms driving system configuration.
    - System configuration is driven by the GIO, although all interaction with interface cards and devices is handled by CDIOs.

- The system administration interface
  - System administration utilities must see a consistent view of the system that are independent of individual drivers' views.
Context-Dependent I/O Modules (CDIOs)

CDIOs contain functionality specific to a particular bus or driver environment (the context is defined by the bus or driver environment). Individual, independent CDIOs provide a way to hide any driver-environment specifics from other environments. This, in turn, allows individual component CDIOs to be configured in or out of a system as needed.

There are two categories of CDIOs:

1. Bus-nexus CDIOs (CDIOs that communicate directly with a bus), which provide bus-dependent services to other CDIOs. They may have bus-nexus drivers to control bus adapters or bus converters. A kernel can contain the following bus-nexus CDIOs:
   - CORE CDIO (Core Context Dependent I/O Module)
   - PA CDIO (Precision Architecture Context Dependent I/O Module)
   - EISA CDIO — optional (EISA Context Dependent I/O Module)
   - PCI CDIO — optional (PCI Context Dependent I/O Module)

2. Driver Environment CDIOs, which provide drivers with a defined environment. Drivers within a CDIO’s environment share a common set of services and entry points. A kernel can contain the Workstation I/O (WSIO) Context-Dependent I/O module CDIO.

Basic Components of a CDIO

- **GIO Interface**
  
  The GIO interface contains entry points invoked by the GIO to access the CDIOs. Generic configuration requests are converted by the CDIO into the appropriate context-dependent functions.

- **Inter-CDIO Communication Interface**
  
  Inter-CDIO communication is provided by services that allow one CDIO to claim hardware modules found by another CDIO, or to gain access to hardware resources maintained by another CDIO.

- **Driver Services**
  
  Driver services that define a driver environment are part of the CDIO. There may be cases where a service in one CDIO is called by a driver in another CDIO. This happens, for instance, with some EISA card drivers that are part of the WSIO CDIO, but that call bus-dependent functions from the EISA CDIO. However, bus-dependent functions are mostly hidden by services in the WSIO CDIO driver environment to reduce dependencies. Therefore, drivers that control CORE functions and EISA cards do not always require the EISA CDIO (if there is no EISA hardware in the system).

  For example, some drivers can control cards in either a CORE CDIO or EISA CDIO environment. The WSIO CDIO driver environment helps to hide the bus-specific services, so that both EISA and CORE do not need to be pulled in every time the driver is configured.

- **Drivers**
  
  In most cases, a CDIO contains drivers. In a bus-nexus CDIO like EISA, the driver is the EISA bus-nexus manager that configures the EISA adaptor and that provides services specific to EISA. A driver-environment CDIO like WSIO can support many drivers. In most cases, the drivers you write will interact with the WSIO CDIO.

- **Management of I/O resources**
  
  A CDIO controls resources specific to a device. For example, the PA CDIO controls resources specific to Precision Architecture, such as interrupt bits. The WSIO CDIO manages common structures like the Interface Select Code (ISC) table.
How the Driver Environment Works

The WSIO CDIO was originally designed for the workstation single-processor environment. With HP-UX Release 10.20, its functionality was expanded to encompass the server multiprocessor environment. Because the WSIO CDIO is a driver-environment CDIO, it provides a consistent environment regardless of the configuration with bus-nexus CDIOs. Drivers residing within the WSIO CDIO continue to operate smoothly without knowing the underlying configuration.

For example, a driver in the WSIO CDIO (such as a SCSI disk driver) can make the same service calls whether it is configured to work with a CORE CDIO, an EISA CDIO, or with the system’s I/O bus (see Figure 1-1, “Same Driver Can Operate in Many Configurations.”) To the driver, all configurations look the same. It is the task of the WSIO CDIO to interpret the service calls and to take the appropriate actions for the given configuration.

Figure 1-1 Same Driver Can Operate in Many Configurations

Another way to view the role of the WSIO CDIO is to see it as a buffer zone that protects its drivers from the peculiarities of the bus-nexus CDIO that it is configured with, see Figure 1-2, “WSIO CDIO as a Buffer Zone.” The WSIO CDIO masks and hides all interface differences from the driver, handles configuration issues, and monitors resources.
Refer to the *HP-UX 11i v1 Driver Development Reference Guide* for the reference pages for WSIO CDIO routines, services, and data structures.
Overview of the Driver Environment

How the I/O Subsystem is Structured