Sven Stauber and Felix Friedrich

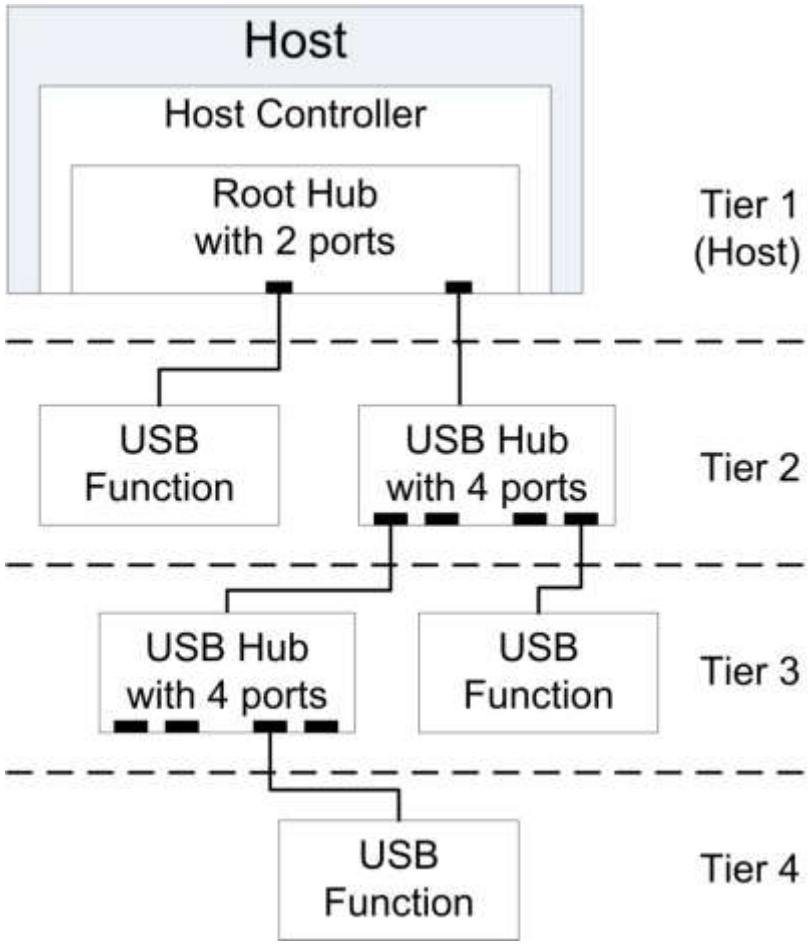
2.5 UNIVERSAL SERIAL BUS

Agenda

- Part I: USB
  - Architectural Overview
  - USB Device Framework
  - Communication Flow
  - Transfer Scheduling
- Part II: A2 USB Implementation
  - Architectural Overview
  - USB Driver Interface
  - USB Device Driver Example

Architectural Overview

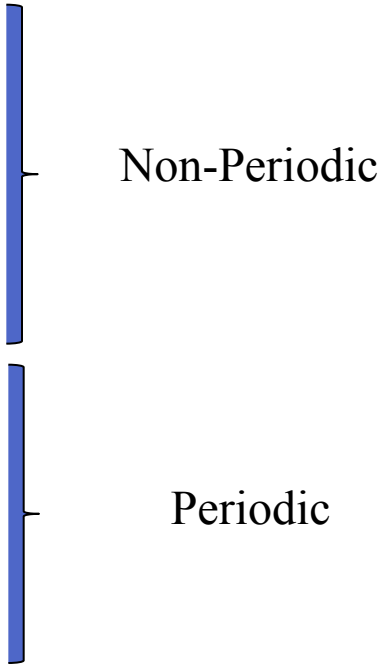
- One host controller per bus
- Max. 127 USB devices per bus
  - **Functions**
    Provide capabilities to the system, e.g. mouse, printer, etc.
  - **Hubs**
    Provide additional attachment points to the USB
- Physical Topology: Tiered star
  max. 7 tiers
- Logical Topology: Bus
- Devices share USB bandwidth through a host-scheduled, token-based protocol (USB is a *polled bus*)



Architectural Overview

Transfer Types:

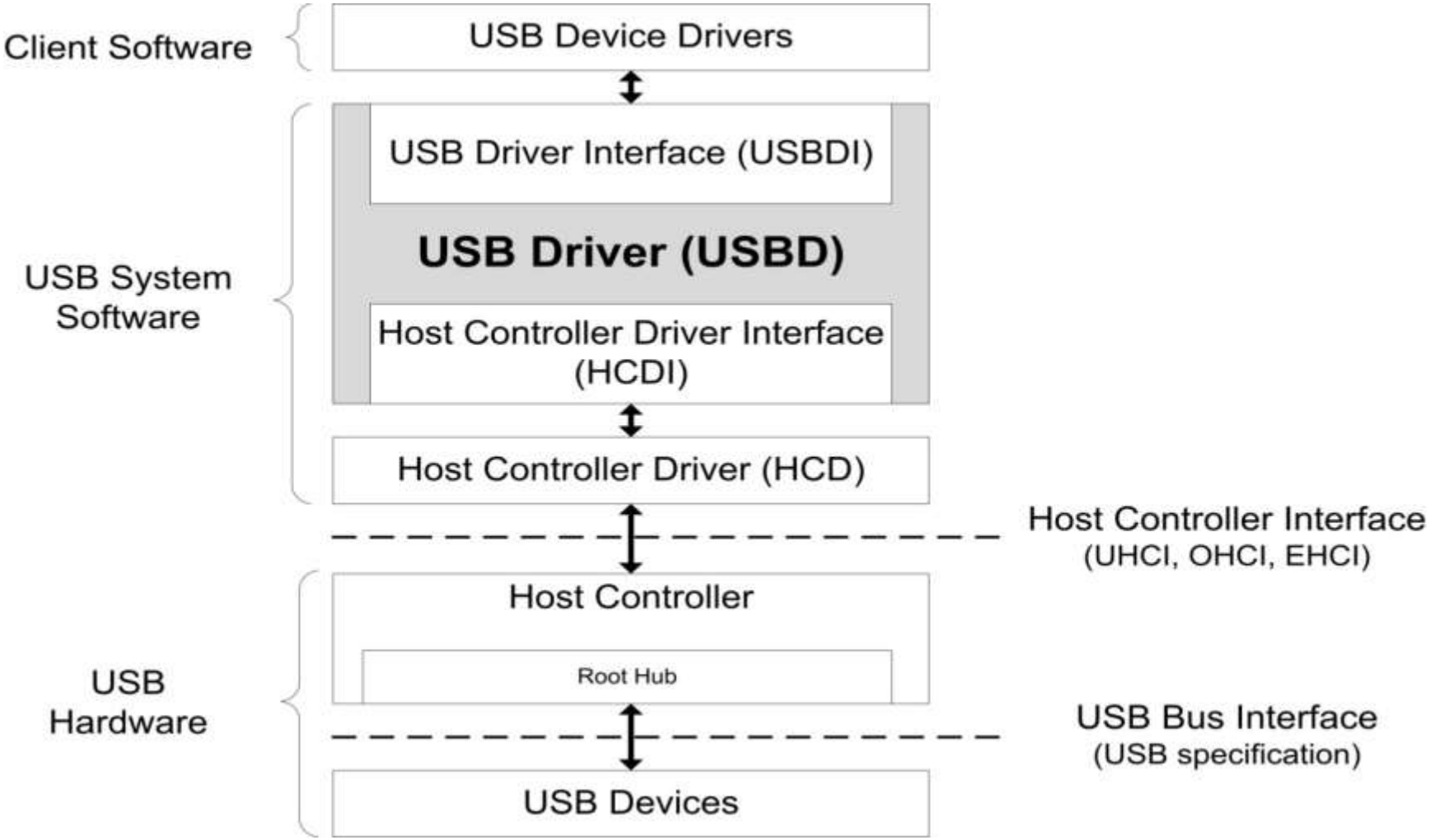
- **Control transfers**
  - Request/Data/Status paradigm
- **Bulk transfers**
  - Large amounts of data with relaxed latency requirements
- **Interrupt transfers**
  - Small amounts of data, guaranteed latency
- **Isochronous transfers**
  - Guaranteed bandwidth & latency, not reliable



Transfer Speeds:

- Low-speed (1.5 Mb/s), Full-speed (12 Mb/s), High-speed (480 Mb/s), SuperSpeed (5.0 Gbps)

USB Host: Hardware and Software

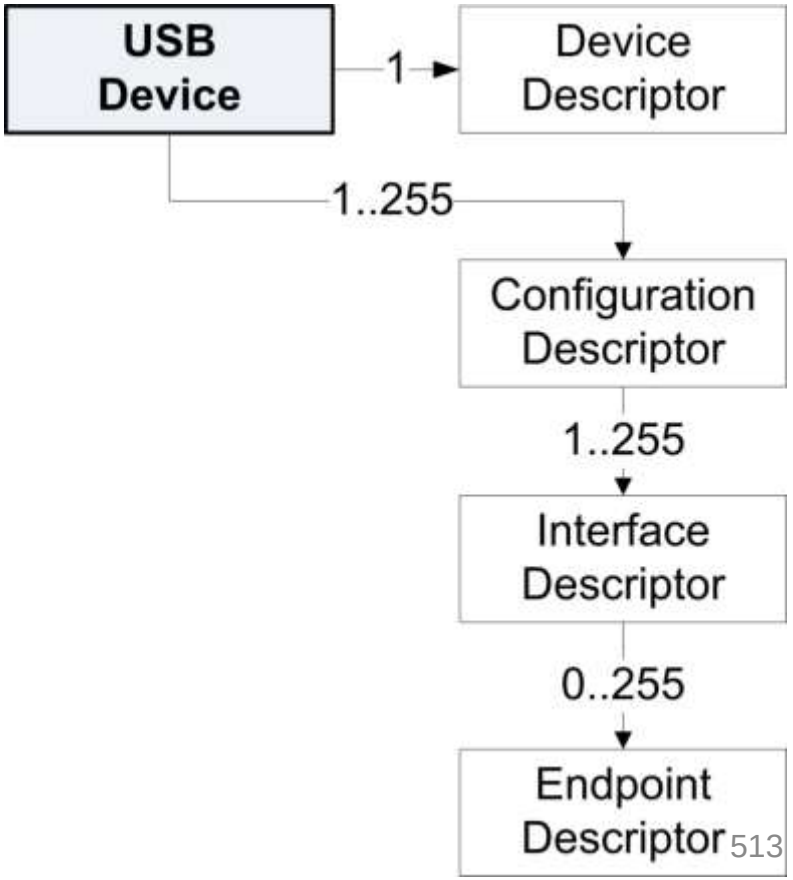


USB Device Framework

The USB specification defines some attributes and operations common to all USB devices.

- Standard Descriptors
- Device States
- Standard Device Requests
- Default Control Endpoint (endpoint 0)

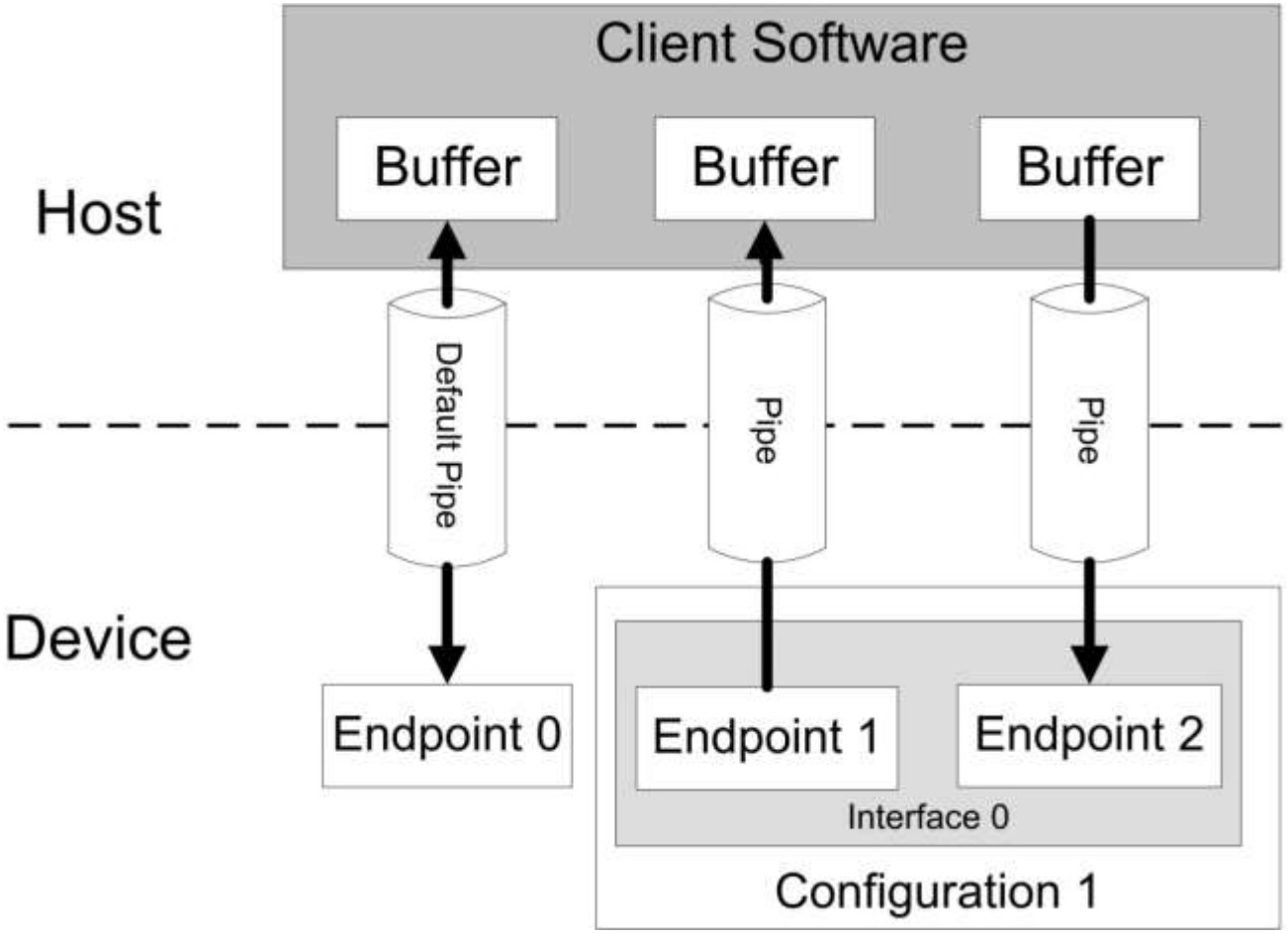
Logical USB device



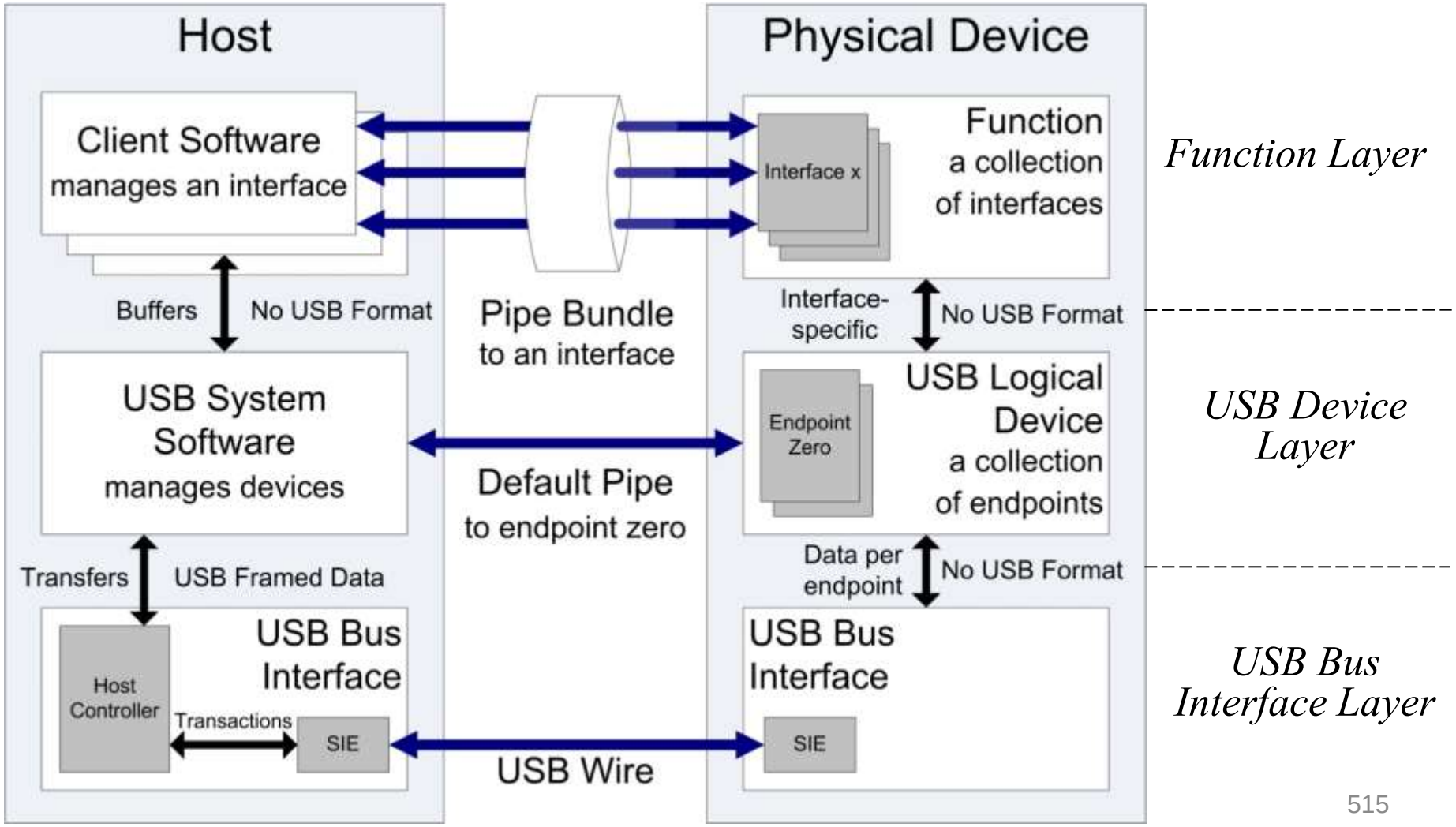
Communication Flow

USB Pipes

- Association between client software and device endpoint



Communication Flow



Transfer Scheduling (UHCI)

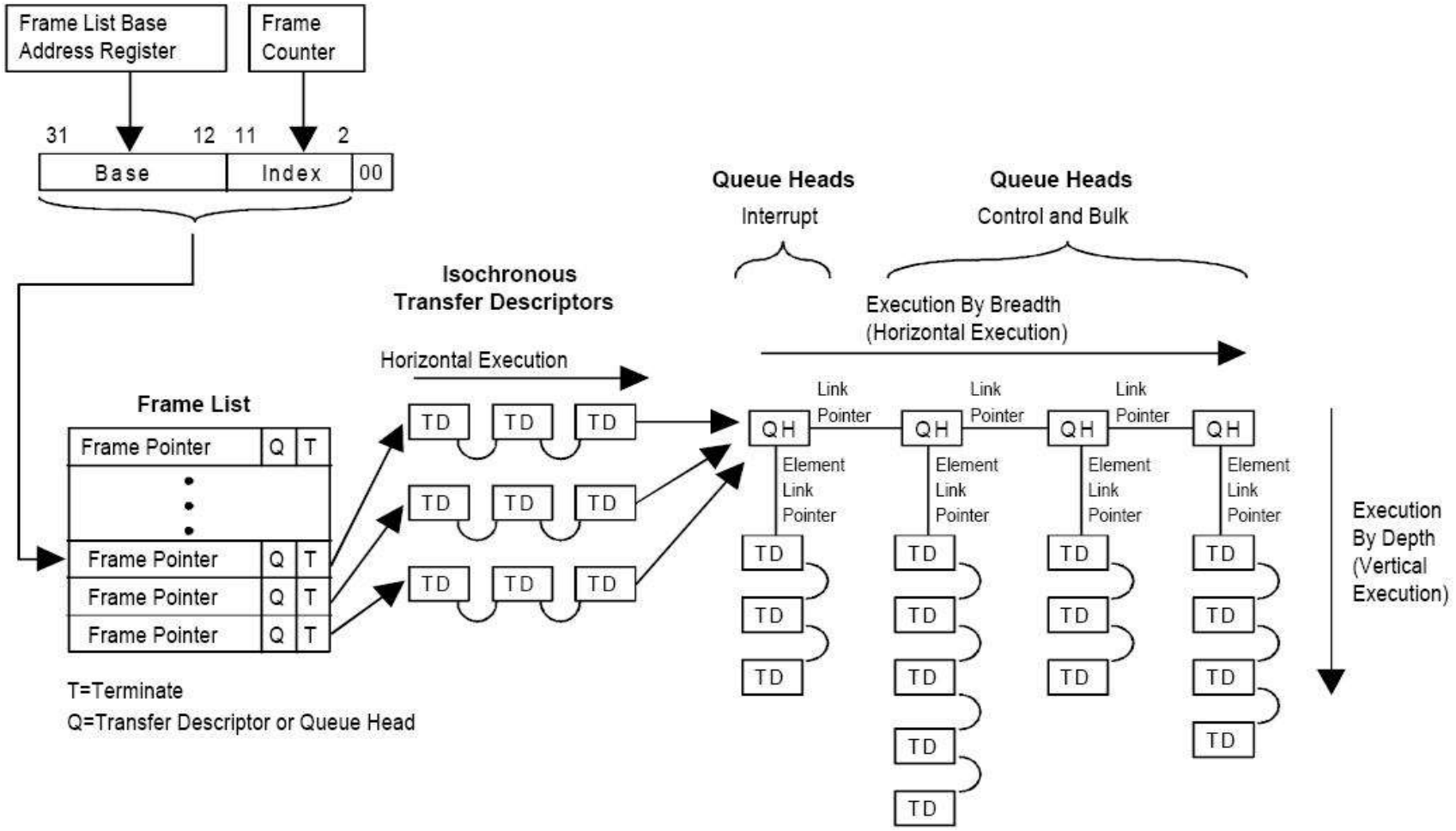


Figure: Example Schedule ([UHCI], p.6)

Data Structures (UHCI)

- Frame List Pointer

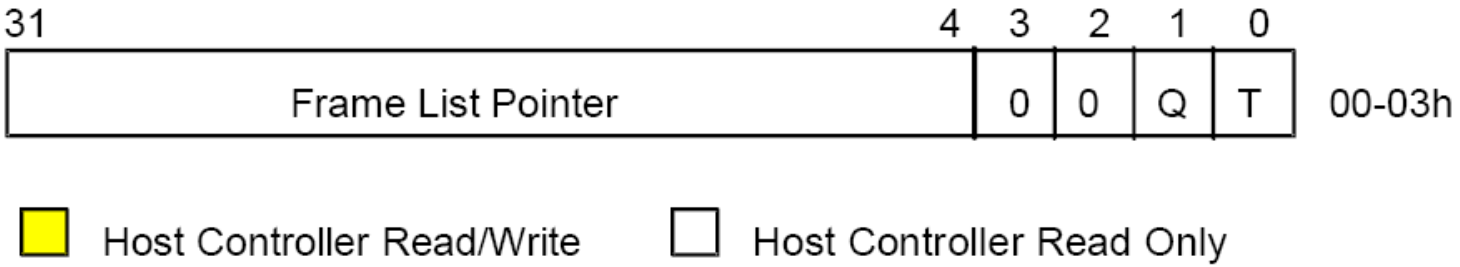


Figure: Frame List Pointer ([UHCI], p. 20)

- Queue Head

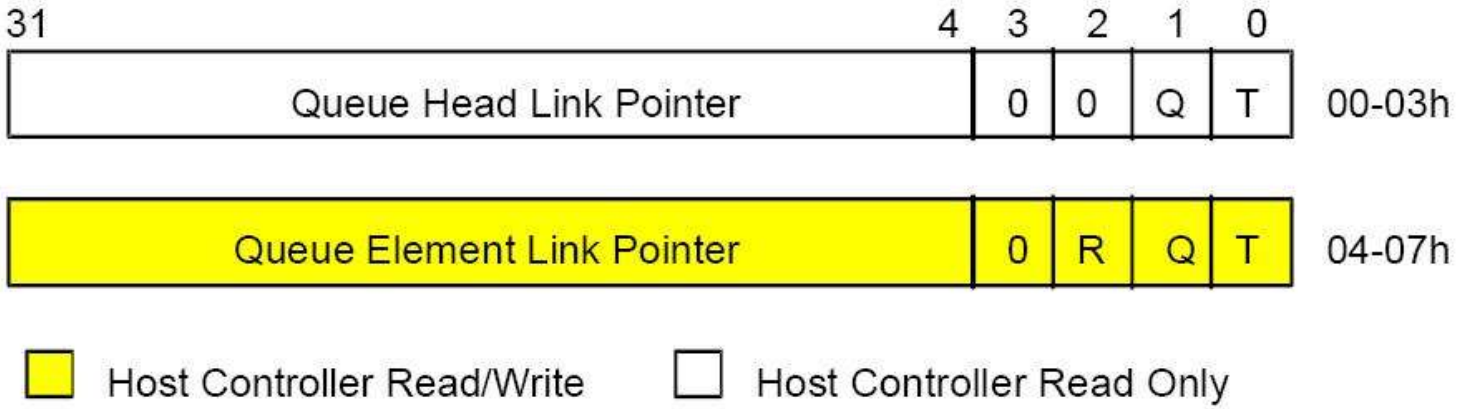


Figure: Generic Form of Queue Head ([UHCI], p. 25)

Data Structures (UHCI)

- Transfer Descriptor

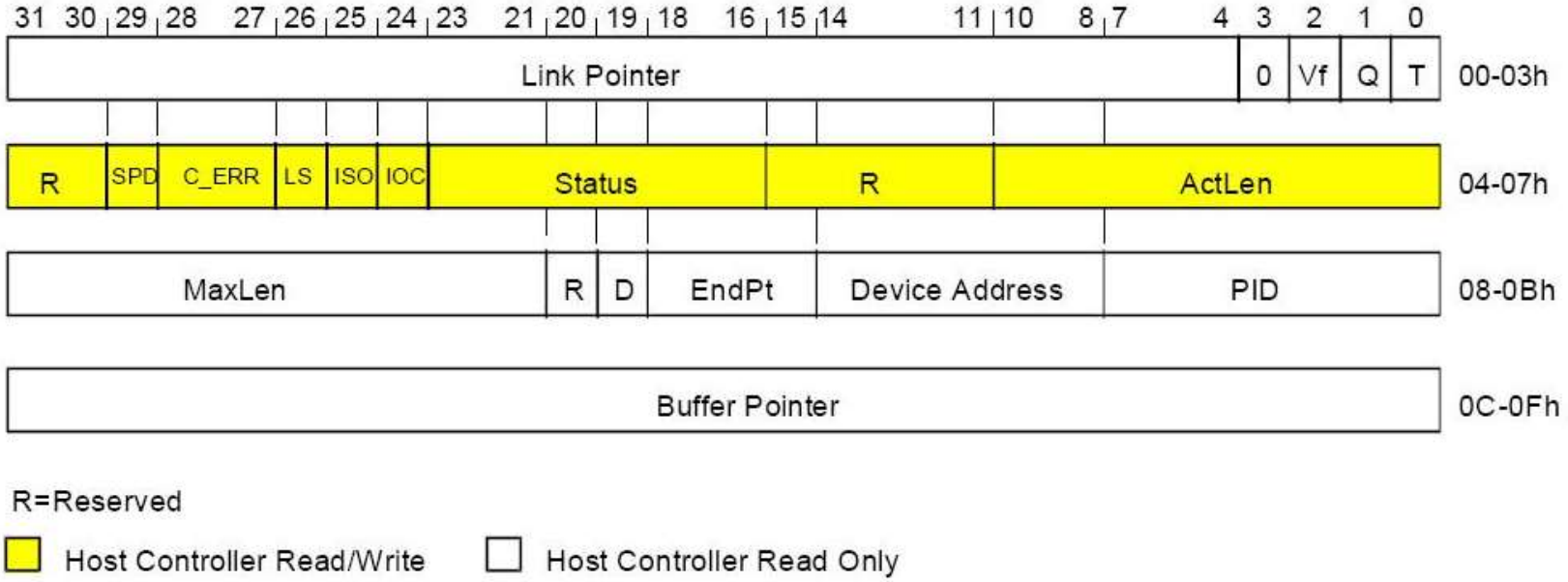
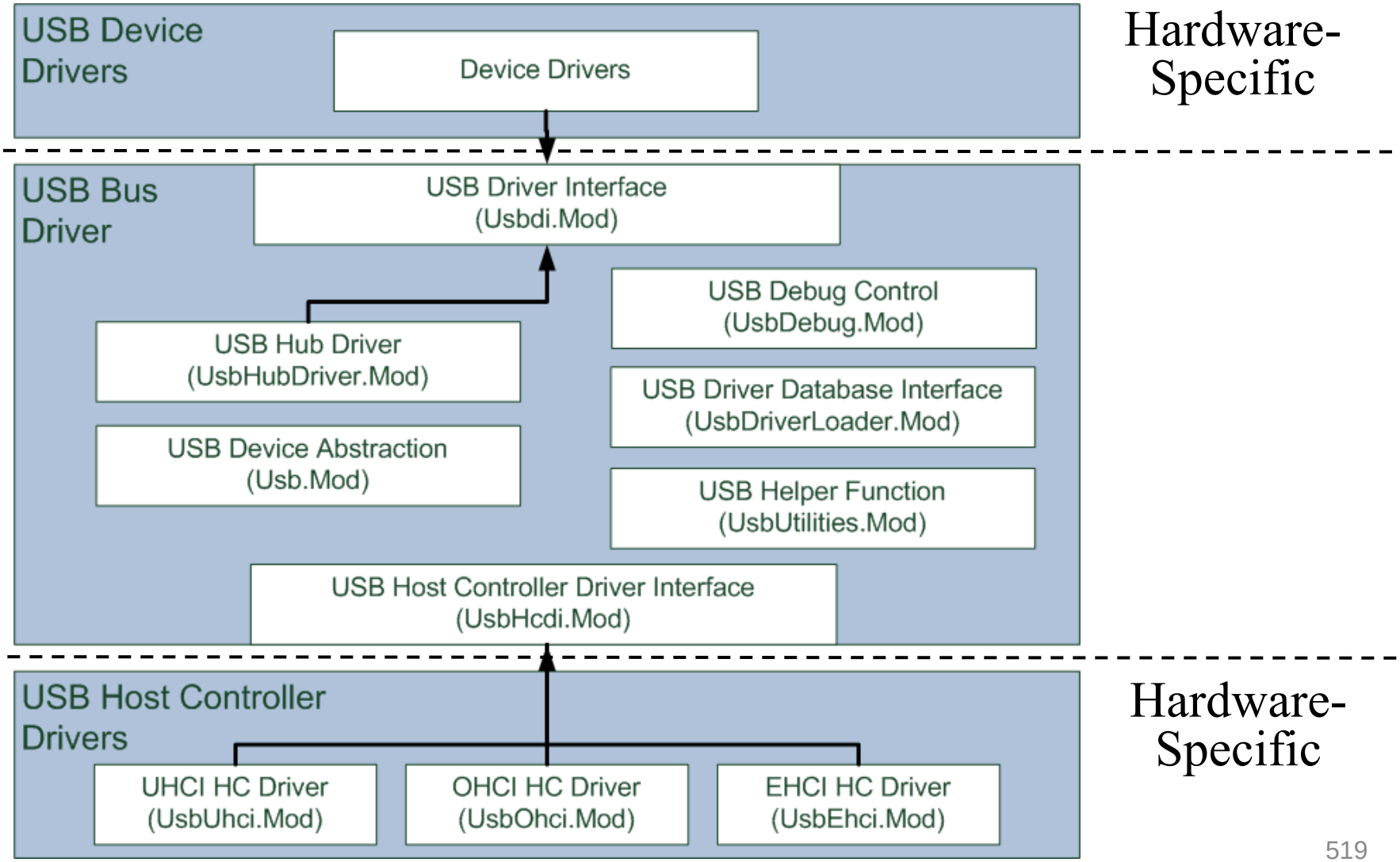


Figure: Generic Form of Transfer Descriptor ([UHCI], p. 21)

A2 USB System Software

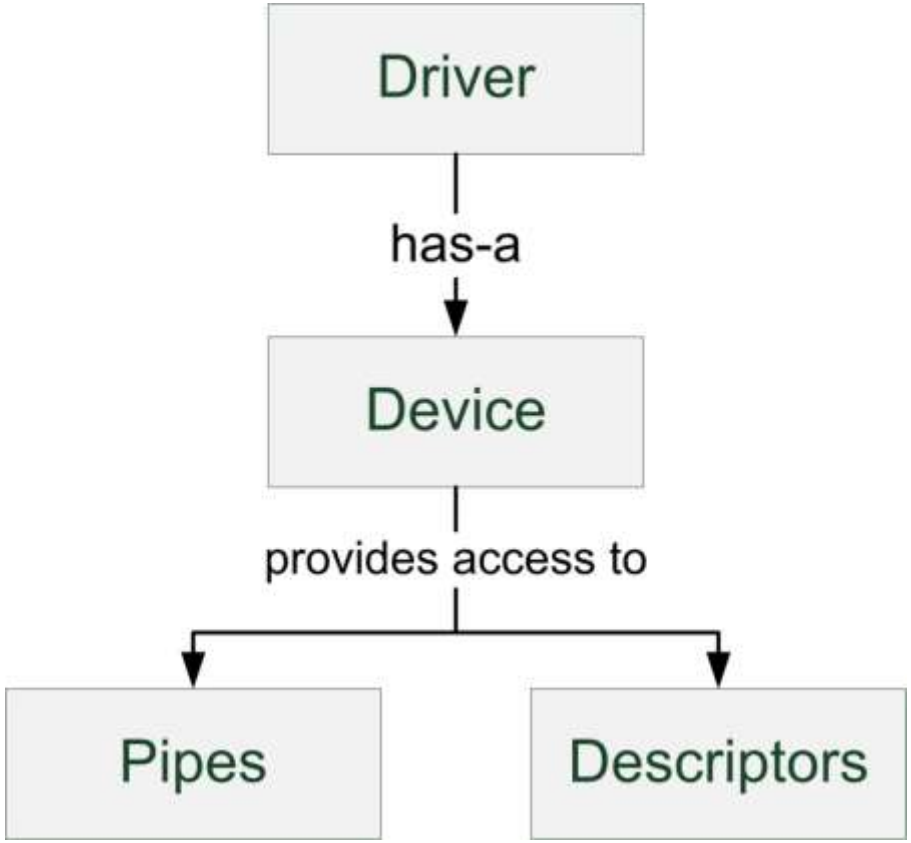
Architectural Overview



USB Driver Interface

Usbdi.Mod

- Defines abstract classes implemented by Usb.Mod and UsbHcdi.Mod
- Hide implementation details from clients
- **Basic idea:**
  Each device driver is a subclass of Usbdi.Driver, which provides access to the USB logical device



USB Driver Interface

Usbdi.Driver

- Base class of all USB device drivers

```
Driver* = OBJECT
VAR
   device* : UsbDevice;
   interface* : InterfaceDescriptor;

   PROCEDURE Connect*() : BOOLEAN;
   PROCEDURE Disconnect*;
END Driver;
```

USB Driver Interface

Usbdi.UsbDevice

- Represents USB logical device

```
UsbDevice* = OBJECT
VAR
    descriptor* : DeviceDescriptor;
    configurations* : Configurations;
    actConfiguration* : ConfigurationDescriptor;

    PROCEDURE GetPipe*(endpointAddress: LONGINT) : Pipe;
    PROCEDURE FreePipe*(pipe: Pipe);
END UsbDevice;
```

USB Driver Interface

Usbdi.Pipe

- Transfer data between software buffer and device endpoint

```
CompletionHandler* =
   PROCEDURE {DELEGATE} (status: Status; actLen: LONGINT);

Pipe* = OBJECT
   PROCEDURE Transfer*(len, ofs: LONGINT; buffer: Buffer): Status;
   PROCEDURE Request*(
        bmRequestType: SET;
        bRequest, wValue, wIndex, wLength: LONGINT): Status;
   PROCEDURE SetTimeout*(milliseconds: LONGINT);
   PROCEDURE SetCompletionHandler*(handler: CompletionHandler);
END Pipe;
```

USB Driver Interface

Driver Registration

- Device driver provides procedure that uses device descriptors to check whether a given device can be handled

```
ProbeProc* = PROCEDURE {DELEGATE} (
    device: UsbDevice; interface: InterfaceDescriptor): Driver;
```

Called for each interface of the device

- Device drivers register themselves at the USB driver

```
Usbdi.drivers.Add*(probe: ProbeProc;
   name: Plugins.Name; desc: Plugins.Description;
   priority : LONGINT);

Usbdi.drivers.Remove*(name : Plugins.Name);
```

Minimal USB Mouse Driver

Registering

```
MouseDriver = OBJECT(Usbdi.Driver)
    (* driver code *)
    PROCEDURE Connect() : BOOLEAN;
    PROCEDURE Disconnect;
END MouseDriver;

PROCEDURE Probe(dev: Usbdi.UsbDevice; id: Usbdi.InterfaceDescriptor
  ) : Usbdi.Driver;
VAR driver : MouseDriver;
BEGIN
  IF id.bInterfaceClass # 3 THEN RETURN NIL; END; (* HID class *)
  IF id.bInterfaceSubClass # 1 THEN RETURN NIL; END; (* Boot protocol subclass *)
  IF id.bInterfaceProtocol # 2 THEN RETURN NIL; END; (* Mouse *)
  NEW(driver); RETURN driver;
END Probe;

PROCEDURE Install;
BEGIN
  Usbdi.drivers.Add(Probe, “UsbMouse”, “Minimal USB Mouse Driver”, 10);
END Install;
```

Minimal USB Mouse Driver

Getting Control

```
MouseDriver = OBJECT
VAR
   buffer : Usbdi.BufferPtr;
   pipe : Usbdi.Pipe;

   PROCEDURE Connect() : BOOLEAN;
   VAR status : Usbdi.Status;
   BEGIN
      pipe := device.GetPipe(81H); (* Assume endpoint 1 is interrupt IN pipe *)
      IF (pipe # NIL) THEN
         NEW(buffer, 3);
         pipe.SetTimeout(0); (* Non-Blocking *)
         pipe.SetCompletionHandler(CompletionHandler);
         status := pipe.Transfer(3, 0, buffer^); (* ignore res *)
      END;
      RETURN pipe # NIL;
   END Connect;

END MouseDriver;
```

Minimal USB Mouse Driver

Receive, Parse and Forward data

```
MouseDriver = OBJECT
VAR
   buffer : Usbdi.BufferPtr;
   pipe : Usbdi.Pipe;

   PROCEDURE CompletionHandler(status : Usbdi.Status; actLen : LONGINT);
   VAR mm : Inputs.MouseMsg;
   BEGIN
      IF (status = Usbdi.Ok) THEN
         mm.dx := SYSTEM.VAL(SHORTINT(buffer[1]));
         mm.dy := SYSTEM.VAL(SHORTINT(buffer[2]));
         IF 0 IN SYSTEM.VAL(SET, buffer[0]) THEN mm.keys := mm.keys + {0}; END;
         IF 1 IN SYSTEM.VAL(SET, buffer[0]) THEN mm.keys := mm.keys + {2}; END;
         IF 2 IN SYSTEM.VAL(SET, buffer[0]) THEN mm.keys := mm.keys + {1}; END;
         Inputs.mouse.Handle(mm);
         status := pipe.Transfer(3, 0, buffer^);
      END;
   END CompletionHandler;

END MouseDriver;
```

Specifications

- [USB2.0], “Universal Serial Bus Specification”, Revision 2.0, April 27 2000, USB Implementers Forum Inc., www.usb.org
- [EHCI], “Enhanced Host Controller Interface Specification for Universal Serial Bus”, Revision 1.0, March 2002, Intel, www.intel.com
- [UHCI], “Universal Host Controller Interface (UHCI) Design Guide”, Revision 1.1, March 1996, Intel, www.intel.com
- [OHCI], “OpenHCI Open Host Controller Interface Specification for USB”, Release 1.0a, September 1996, Microsoft, Compaq & National Semiconductors, ftp://ftp.compaq.com/pub/supportinformation/papers/hcir1_0a.pdf