OpenDriver Specification 0.1

Mithrill Group

January 14, 2005

Contents

Chapter 1

Introduction

Chapter 2

Architecture

Chapter 3

C Run-Time Environment

3.1 Types

int8_t
int16_t
int32_t
int64_t
int128_t
int256_t

uint8_t
uint16_t
uint32_t
uint64_t
uint128_t
uint256_t

real32_t
real64_t
real128_t

size_t
pointer_t

bool_t
char_t
string_t

Chapter 4

C++ Run-Time Environment

Chapter 5

Java Run-Time Environment

Chapter 6

Driver

Chapter 7

Enumerator

Chapter 8

ISA Bus

8.1 Architecture

8.1.1 Overview

The ISA bus was the original bus or the IBM PC. It was an 8-bit bus without any support for Plug&Play. In the IBM AT it was expanded to a 16 bit bus, with support for more IRQ levels and DMA channels. Some of the controllers of the early PCs (which are now intergrated into the *southbridge* or modern chipsets) were also conected to the ISA bus, but because they were not limited by the connector they had access to additional IRQ levels and DMA channels. This is sometimes called the LPC (*Legacy PC*) bus. An example is the keyboard controller.

Later a Plug&Play option was added, but very few cards supported this. The device should already be configured when the driver is loaded.

The original bus and DMA controller ran at 4.77 MHz but in modern systems this is increased to 8.33 MHz (a quarter of the PCI clock).

8.1.2 I/O Ports

While the IA-32 processors are able to access 2^{16} I/O ports, the ISA bus allows only access to 2^{10} I/O ports. Because I/O port addresses above 0x3FF are not filtered out, every I/O port x *might* be aliased at $x + 0\text{x}400k$, with $k \in \{1, 2, ..., 63\}$.

8.1.3 Interrupts

The ISA bus uses two cascaded 8259A PIC controllers. These allow for a total of 16 IRQ levels, but only 11 are wired onto the bus. The ISA bus only supports *edge-triggered* interrupts, so officially interrupt sharing is not possible. If care is taken that two devices are not active at the same time it can be done, however. In some PCs the sound card and the second parallel port both use IRQ 7, for

example.

IRQ 2 is used for the cascade, the pin on the bus was rerouted to IRQ 9 on the AT and is sometimes refered to as IRQ 2/9 or on 8-bit cards as IRQ 2.

IRQ 13 was used to report FPU errors for the 80287 and 80387 but is not used on the 80487 and up in native mode.

Level	Bus	Usage
0	No	Timer
1	No	Keyboard
2	No	Cascade
3	Yes	Serial Port 2
4	Yes	Serial Port 1
5	Yes	Parallel Port 2
6	Yes	Floppy Disk
7	Yes	Parallel Port 1
8	No	Real-Time Clock
9	Yes	PCI
10	Yes (16)	PCI
11	Yes (16)	PCI
12	Yes (16)	Mouse
13	No	FPU
14	Yes (16)	Hard Disk 1
15	Yes (16)	Hard Disk 2

8.1.4 DMA

The ISA bus uses two cascaded 8237A DMA controllers providing 8 DMA channels. Channels 0 to 3 can do 8-bit transfers, while channels 5 to 7 can only do 16-bit transfers. Channel 4 is used for cascading and is not available for use by devices. Transfers cannot be done simultaniously.

A common misconception is that channel 0 is used for the refreshing of the RAM. This was only done in the original PC-XT. ATs and higher use dedicated hardware and channel 0 was made available, although only for 16-bit cards.

For 8-bit transfers the DMA controller can transfer upto 64 kB. It can only address 64 kB also. To solve this an extra 8-bit page register is used to extend the address space to the full 16 MB. This page register is not updated during a transfer however, meaning that the data read/written will wrap back on a 64 kB boundary. To avoid this you should align transfers on 64 kB boundaries. If you planned for this behaviour the operating system will support it, however.

For 16-bit transfers the DMA controller can transfer upto 128 kB. As far as I know the low bit op the page register is ignored allowing for 128 kB DMA pages. The Intel SIO also supported extra page register allowing 128 MB (27-bits) or 4 GB (32-bits) of physical memory to be addressed. This information is not on the Intel PIIX or Intel ICH datasheets. The Intel SIO could also program individual DMA channels into 8-bit, 16-bit or the new 16-bit count-by-byte modes.

Channel	Bus	Width	Usage
0	Yes (16)	8	
1	Yes	8	Parallel Port 1
2	Yes	8	Floppy Disk
3	Yes	8	Parallel Port 2
4	No	16	Cascade
5	Yes (16)	16	Hard Disk 2
6	Yes (16)	16	
7	Yes (16)	16	Hard Disk 1

8.1.5 Memory

The ISA bus supports mapped memory to support option ROMs, video memory and EMS frames. Only 24 address-bits are put onto the bus (20 for 8-bit cards) so the mapped memory must be located below 16 MB (1 MB for 8-bit cards).

8.2 Interface

enum isa_resourcetype_t {
ISA_RESOURCETYPE_IRQ,
ISA_RESOURCETYPE_DMA,
ISA_RESOURCETYPE_IOPORT,
ISA_RESOURCETYPE_MEMORY
};

struct isa_resource_t {
isa_resourcetype_t type;
isa_resourceaddress_t address;
};

8.2.1 Interface

void isa_initialize(isa_resource_t resourcelist[ ]);

void isa_finalize(void);

8.2.2 Methods

bool_t isa_resource_allocate(isa_resource_t);

bool_t isa_resource_free(isa_resource_t);

void isa_dma_read(isa_resourceaddress_t, pointer_t, size_t);

This operating will lock the DMA controller so no other device driver can use it. It will then configure it for a read operationg. If the DMA controller is already locked this call waits until it is unblocked.

resourceaddress: the DMA channel.
pointer: the buffer.
size: the amount of bytes/words (8-bit/16-bit transfers) to transfer.

void isa_dma_write(pointer_t, isa_resourceaddress_t, size_t);

bool_t isa_dma_polldone(isa_resourceaddress_t);

void isa_dma_waitdone(isa_resourceaddress_t);

bool_t isa_dma_pollrequest(isa_resourceaddress_t);

void isa_dma_waitrequest(isa_resourceaddress_t);

interrupt_t isa_irq_getinterrupt(isa_resourceaddress_t);

int8_t isa_io_read8(isa_resourceaddress_t);
int8_t isa_io_read16(isa_resourceaddress_t);
int8_t isa_io_read32(isa_resourceaddress_t);
void isa_io_write8(int8_t, isa_resourceaddress_t);
void isa_io_write16(int16_t, isa_resourceaddress_t);
void isa_io_write32(int32_t, isa_resourceaddress_t);

8.2.3 Examples

```
isa_dma_read( FLOPPY_DMA, buffer, 512 );
isa_io_write8( FLOPPY_COMMAND, FLOPPY_READ );
interrupt_wait( isa_irq_getinterrupt( FLOPPY_IRQ ) );
isa_dma_polldone( FLOPPY_DMA );
```

8.3 Interface

Chapter 9

Audio Devices

Chapter 10

Graphics Devices

Chapter 11

Video Devices

Chapter 12

OpenGL