Uniform Driver Interface

UDI PCI Bus Binding Specification Version 1.01

http://www.project-UDI.org/specs.html

UDI PCI Bus Binding Specification

Abstract

The UDI PCI Bus Binding Specification defines the requirements for use of the UDI Physical I/O Specification on the PCI Bus. This is an optional extension to the UDI Physical I/O Specification, which is defined in a separate book. The intended audience for this book includes driver writers, environment implementors, and metalanguage implementors.

UDI drivers that require use of physical I/O on a PCI bus must be written to conform to this specification, and can assume that all services described herein are available. Environments that don't need such drivers may choose not to support the PCI extensions, but any environment that supports UDI PCI drivers must conform to this specification, as well as to the UDI Physical I/O Specification and the UDI Core Specification.

See the Document Organization chapter in the UDI Core Specification for a description of other books in the UDI Specification collection, as well as references to additional tutorial materials.

Status of This Document

This document has been reviewed by Project UDI Members and other interested parties and has been endorsed as a Final Specification. It is a stable document and may be used as reference material or cited as a normative reference from another document. This version of the specification is intended to be ready for use in product design and implementation. Every attempt has been made to ensure a consistent and implementable specification. Implementations should ensure compliance with this version.

Copyright Notice

Acknowledgements

The authors would like to thank everyone who reviewed working drafts of the specification and submitted suggestions and corrections.

The authors would especially like to thank their significant others for putting up with the many hours of overtime put into the development of this specification over long periods.

Thanks to the following folks who contributed significant amounts of time, ideas, or authoring in support of the development of this specification or in working on the prototype implementations which helped us validate the specification:

Richard Arndt (IBM)
Bob Barned (Lockheed Martin)
Mark Bradley (Adaptec)
Darren Busing (Adaptec)
Steve Bytnar (STG)
Thomas Clark (Sun)
Deven Corzine
Jack Craig (SCO)
Betty Dall (HP)
Tim Damron (IBM)
Burkhard Daniel (STG)
Don Dugger (Intel)
Mark Evenson (HP)
Barry Feild (SCO)
Scott Feldman (Intel)
Mike Firman (STG)
Kurt Gollhardt (SCO)
Bob Goudreau (Data General)
James Hall (SCO/Sun)
Jim Heidbrink (Lockheed Martin)
Chris Herzog (STG)
Chris Ilnicki (HP)
Bret Indrelee (SBS Technologies)
David Kahn (Sun)
Matt Kaufmann (SCO)
Andrew Knutsen (SCO)
Ahuva Kroizer (Intel)
Man Fai Lau (SCO)
John Lee (Sun)
Robert Lipe (SCO)
Mike Lyons (IBM)
Alex Malone (DEC)
Lynne McCue (IBM)
Bill Nicholls
Guru Pangal (Starcom)
Mark Parenti (DEC)
James Partridge (IBM)
Scott Popp (SCO)
Hiremane Radhakrishna (Intel)
John Ronciak (Intel)
Kevin Quick (Interphase)
Larry Robinson (Adaptec)
Andrew Schweig (STG)
Sam Shteingart (HP)
Ajmer Singh (SCO)
James Smart (Compaq)
Pete Smoot (HP)
David Stoft (HP)
Rob Tarte (Pacific Codeworks)
Wolfgang Thaler (Sun)
Ramaswamy Tummala (Starcom)
Linda Wang (Sun)
Kevin Van Maren (Unisys)
Mike Wenzel (HP)

Countless people have helped in one way or another and any omissions or errors on our part in the list above are just that: omissions or errors on our part.

Thanks to Kevin Quick and the folks at Interphase for hosting the Interoperability events which have provided a great venue for validating both prototype and production UDI products.

Finally, thanks to David Roberts (Certek Software Designs) for designing the Project UDI logo.

Table of Contents

Introduction to the PCI Bus Binding 1

1.1 Overview

The UDI PCI Bus Binding specifies the usage details for the UDI Physical I/O Specification that are specific to the PCI Bus. This chapter defines general requirements for use of the UDI PCI Bus Binding Specification. The next chapter defines the specifics of the UDI PCI Bus Binding.

1.2 General Requirements

Certain basic rules apply to all UDI PCI drivers (for both bus bridges and adapters that use the PCI bus). In order to be UDI-compliant, such a driver must follow all of these rules. UDI PCI drivers must also follow the rules specified in the UDI Physical I/O Specification and the UDI Core Specification. Rules specific to PCI drivers are listed here.

Before including any UDI header files, the driver must define the preprocessor symbol, UDI_PCI_VERSION, to indicate the version of the UDI PCI Bus Binding Specification to which it conforms. For this version of the specification, UDI_PCI_VERSION must be set to 0x101:

```
#define UDI_PCI_VERSION 0x101
```

Each device driver source file must include the file "`udi_pci.h`" after it includes "`udi.h`" and "`udi_physio.h`", as follows:

```
#include <udi.h>
#include <udi_physio.h>
#include <udi_pci.h>
```

These header file contains environment-specific definitions of standard UDI structures and types, as well as all function prototypes and other definitions needed to use the core, physical I/O, and PCI bus UDI interfaces and services. Additional include files may be needed for other non-core services and metalanguages as defined in other UDI Specifications.

To maintain portability across UDI supportive platforms, device driver writers shall not assume any knowledge of the contents of these header files with respect to implementation-dependent aspects of the UDI interfaces (such as the definition of handles or abstract types). Similarly, drivers shall not access any functions or objects external to the driver except those defined in the UDI Specifications to which they conform.

1.3 Normative References

The UDI PCI Bus Binding Specification references the non-UDI standards listed below. These standards contain provisions that, through reference in this document, constitute provisions of the UDI PCI Bus Binding Specification.

Normative References

1. PCI Local Bus Specification.

PCI Bus Bindings 2

Some of the UDI services interfaces defined in the UDI Physical I/O Specification require bus binding information to appropriately use the interface and set parameter values. This chapter specifies the bus bindings for the PCI bus.

2.1 PIO Bindings

2.1.1 udi_pio_map

The following ***regset_idx*** values are defined for PCI:

```
#define UDI_PCI_CONFIG_SPACE            255
#define UDI_PCI_BAR_0                   0
#define UDI_PCI_BAR_1                   1
#define UDI_PCI_BAR_2                   2
#define UDI_PCI_BAR_3                   3
#define UDI_PCI_BAR_4                   4
#define UDI_PCI_BAR_5                   5
```

Any 64-bit BARs must be accessed by the lowest of the two BAR numbers used to hold the 64-bit value.

Any other values passed to `udi_pio_map` in the ***regset_idx*** argument are illegal.

2.2 Interrupt Bindings

2.2.1 Interrupt Index Values

Since PCI allows only one interrupt pin per PCI function, only one ***interrupt_idx*** value, zero, is used for PCI devices.

2.2.2 Event Info

There is no event info for PCI bus interrupts. Event info size must always be zero.

2.3 Instance Attribute Bindings

2.3.1 Enumeration Attributes

The following enumeration attributes are defined for PCI devices. All numeric attributes are stored as UDI_ATTR_UBIT32 type attributes, which are automatically converted to/from each driver's endianness. Since PCI configuration space values are little-endian, the bus bridge driver must combine individual bytes into a numeric value (e.g. "attr_value = lo_byte + (hi_byte << 8)"), or store the little-endian values directly in ***attr_value*** of the udi_instance_attr_list_t rather than using UDI_ATTR32_SET().

Table 2-1 PCI Enumeration Attributes

ATTRIBUTE NAME	TYPE	SIZE	Description
bus_type	UDI_ATTR_STRING	4	"pci"
pci_vendor_id	UDI_ATTR_UBIT32	4	16-bit numeric PCI Vendor ID
pci_device_id	UDI_ATTR_UBIT32	4	16-bit numeric PCI Device ID
pci_revision_id	UDI_ATTR_UBIT32	4	8-bit numeric PCI Revision ID
pci_baseclass	UDI_ATTR_UBIT32	4	8-bit numeric PCI Base Class Code
pci_sub_class	UDI_ATTR_UBIT32	4	8-bit numeric PCI Sub-Class Code
pci_prog_if	UDI_ATTR_UBIT32	4	8-bit numeric PCI Programming Interface
pci_subsystem_vendor_id	UDI_ATTR_UBIT32	4	16-bit numeric PCI Subsystem Vendor ID
pci_subsystem_id	UDI_ATTR_UBIT32	4	16-bit numeric PCI Subsystem ID
pci_unit_address	UDI_ATTR_UBIT32	4	low-order 3 bits: PCI Function Number next 5 bits: PCI Device Number next 8 bits: PCI Bus Number
pci_slot	UDI_ATTR_UBIT32	4	8-bit physical slot number, if known

2.3.2 Filter Attributes

Of the above listed enumeration attributes, the following are valid filter attributes for enumeration filtering. For both of these, stride is interpreted linearly; that is, the stride value is simply added to the numeric value of these attributes.

```
pci_unit_address
pci_slot
```

2.3.3 Generic Enumeration Attributes

2.3.3.1 identifier attribute

For PCI devices, the "`identifier`" attribute encodes a combination of the `pci_vendor_id`, `pci_device_id`, `pci_revision_id`, `pci_subsystem_vendor_id`, and `pci_subsystem_id` attributes, as follows:

```
identifier = VVVVDDDDRRvvvvdddd
```

where `VVVV` is a four-digit upper-case hexidecimal-encoded ASCII representation of the PCI Vendor ID, `DDDD` is a four-digit upper-case hexidecimal-encoded ASCII representation of the PCI Device ID, `RR` is a two-digit upper-case hexidecimal-encoded ASCII representation of the PCI Revision ID, `vvvv` is a four-digit upper-case hexidecimal-encoded ASCII representation of the PCI Subsystem Vendor ID, and `dddd` is a four-digit upper-case hexidecimal-encoded ASCII representation of the PCI Subsystem ID.

2.3.3.2 address_locator attribute

For PCI devices, the "`address_locator`" attribute encodes the `pci_unit_address` attribute using the following syntax:

```
address_locator = BBDDF
```

where `BB` is a two-digit upper-case hexidecimal-encoded ASCII representation of the PCI Bus Number, `DD` is a two-digit upper-case hexidecimal-encoded ASCII representation of the PCI Device Number, and `F` is a one-digit upper-case hexidecimal-encoded ASCII representation of the PCI Function Number.

2.3.3.3 physical_locator attribute

For PCI devices, the "`physical_locator`" attribute encodes the `pci_slot` attribute using the following syntax:

```
physical_locator = SS
```

where `SS` is a two-digit upper-case hexidecimal-encoded ASCII representation of the physical slot number, if known. The `physical_locator` attribute is present if and only if `pci_slot` is provided for this device.

2.3.3.4 physical_label attribute

No "`physical_label`" attribute is defined generically for PCI. Platforms that have access to such information may set `physical_label` attributes.

2.3.4 Enumeration Attribute Ranking

To support the ranking of enumerated devices against available drivers for the `udi_mei_enumerate_rank_func_t`, the following combinations of enumeration attribute matches yield the corresponding ranking values. Attribute combinations not specified return a relative rank of 0 (the lowest possible rank). The combinations are unchanged by matches against non-rankable attributes.

Table 2-2 PCI Enumeration Attribute Ranking

	Rank Value														
Rankable Attributes[1]	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
identifier															Y
pci_subsystem_id													Y	Y	
pci_revision_id										Y		Y		Y	
pci_device_id									Y	Y	Y	Y			
pci_subsystem_vendor_id						Y	Y	Y			Y	Y	Y	Y	
pci_vendor_id			Y	Y	Y				Y	Y					
pci_sub_class		Y			Y			Y							
pci_baseclass	Y	Y		Y	Y		Y	Y							

1. Y indicates a valid match of the attribute. Only the attributes listed are rankable; all other enumeration attributes have no effect on the ranking value.

2.3.5 Parent-Visible Attributes

No parent-visible attributes are defined for PCI bus bridge drivers.