PCISIG ReSpec Example

30 May 2014

This version:

Latest published version:

none

Latest editor"s draft:

http://sglaser.github.io/respec/examples/xxx.html

Editor:

Steve Glaser

Contact the PCI-SIG office to obtain the latest revision of this specification.

Questions regarding this specification or membership in PCI-SIG may be forwarded to:

Membership Services	Technical Support
administration@pcisig.com	techsupp@pcisig.com
+1-503-619-0569 (Phone)	
+1-503-644-6708 (Fax)	

DISCLAIMER

This Specification is provided “as is” with no warranties whatsoever, including any warranty of merchantability, noninfringement, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. PCI-SIG disclaims all liability for infringement of proprietary rights, relating to use of information in this specification. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

Abstract

This is just a very basic document. The following items automatically generate references in the References Appendix. [REX] Yeah for memes [MEMES]. [DOM4], [XML10], [[WHATEVER]]

Note that this example document turns on the `addDefinitionMap` flag in the `respecConfig`. This inserts the Definition Map Appendix.

There are a number of areas that still need work. Some of them are visible in this document and some of them are background issues. The main html that users would write is pretty solid. Work areas include:

1. ~~Copyright needs to be updated~~ **Done**
2. ~~Move styles into an external css file~~ **Done**
3. Validate external figure last modified date against the corresponding raw source file last modified date.
4. Better build process independent of w3c usage
5. ~~Banner at left needs to be populated automatically~~ **Done**
6. Populate cross reference content automatically (the "See Section 1.2.3.4" stuff)
7. MathML doesn't work on most browsers (exception: Firefox, Safari). Switch to http://www.mathjax.org/
8. Link field names and field in the automatically generated figure
9. Make Table of Contents, Table of Figures, Table of Tables into sidebars that can expand and collapse
10. Support breaking large documents into a set of html files for faster loading
11. Provide diff / changebar support. This would automatically create a derived document with <ins> and <del> tags inserted to reflect differences between two commits.
12. PDF output: ~~Perhaps u~~U~~sing Prince XML http://www.princexml.com **In Progress**

> ISSUE 1: PDF Output Status
>
> The princexml tool works on a spec snapshot. It can't handle the raw spec input because it doesn't implement the setTimeout function (defined for browsers, but not part of ECMAScript). The css file needs work; it donesn't have any @print rules.

Table of Contents §

Table of Figures

Table of Tables

1. Informative section §

This section is non-normative.

blah

1.1 Inner section §

blahblah

1.2 Other inner section §

blahblah

2. Terms and Acronyms §

Support has been added to automatically detect, format, and link terms and associated definitions.

Terms are defined using the `<dfn class="xxx">` tag. Term namespaces are independent for each class (e.g. it's OK to have `<dfn class="pin">foo</dfn>` and `<dfn class="signal">foo</dfn>`. Valid class values are:

- class="dfn" — *the default*
- class="pin"
- class="signal"
- class="term"
- class="field"
- class="register"
- class="state"
- class="value"
- class="parameter"
- class="argument"

Term references use the <a> tag with no href attribute. The class is optional and is only needed to resolve ambiguity (e.g. `<a class="pin">foo</a>` vs `<a class="signal">foo</a>`). When the contents of a term definition contains HTML tags (e.g. <sub>, <abbr>), this HTML is remembered and is copied to the references. This makes subscripts and

abbreviations more convenient since <a>D0active could expand to <a href="#state-D0-active" class="state">D0_{active} D0activeThe <a> references the text of the in <dfn> the <a> is the text after removing HTML tags. The contents of the <dfn> tag is copied into these references. The term contains subscripts, these are automatically copied into the reference.

Term1

: This is the definition for Term #1. It uses no markup.

PIN2#

PIN2A#

: This is the definition for PIN2# and PIN2a#. It uses two <dt> tags, containing <dfn class="pin">PIN2#</dfn> and <dfn class="pin">PIN2a#</dfn>.

TERM3

: This is the definition for Term #3. It uses <dfn><abbr title="Term number 3">TERM3<abbr></dfn>

Term number 4

Term4

: This is the definition for Term #4. It uses <dfn><abbr title="Term number 4">Term4</abbr></dfn>

D3cold

: This is the definition for D3cold. It <dfn class="state">D3_{cold}</dfn>.

Detect.Quiet

: This is <dfn class="state">Detect.Quiet</dfn>

Term6

: This is the definition for Term6. It <dfn>Term₆</dfn>.

Term7

: This is the definition for Term7. It <dfn><abbr>Term₇</abbr></dfn>.

Term number 8

Term8

: This is the definition for Term8. It <dfn><abbr title="Term number 8">Term₈</abbr></dfn>.

TERM9

: This is the definition for Term #9. It uses <dfn class="pin">.

TERM9

: This is the definition for Term #9. It uses <dfn class="signal">.

Term9

: This is the definition for Term #9. It uses <dfn class="term">.

Term9 ***Duplicate definition of 'term-term9'***

: This is the definition for Term #9. It uses <dfn class="term">.

Term 9

: This is the definition for Term #9. It uses <dfn class="term">.

2.1 References: §

- Term1 Term1

- tErM1 Term1
- PIN2# PIN2#
- PIN2a# PIN2A#
- TERM3 TERM3
- term number 3 TERM3
- title="term number 3" This is term number 3
- TERM3 xxx
- Term number 4 Term4
- D3cold D3cold
- d3cold D3cold
- D3cold D3cold
- xyzzy title="d3cold" xyzzy
- Term6 Term6
- Term7 Term7
- term number 8 Term number 8
- Term9 class="pin" TERM9
- Term9 class="signal" TERM9
- Term9 class="term" Term9 **Duplicate definition of 'term-term9'**
- Term9 no class (ambiguous) Term9 **Ambiguous reference to 'pin-signal-term-term9', resolved as 'pin-term9'**
- Term9 class="externalDFN" Term9
- Term 9 (with space before the '9') Term 9
- Detect.Quiet Detect.Quiet
- first first
- Z Z
- bit13 bit13

3. MathML Test §

This is a simple MathML equation (pi times r squared). $\pi \times r^2$

4. JSON Example Register Figure §

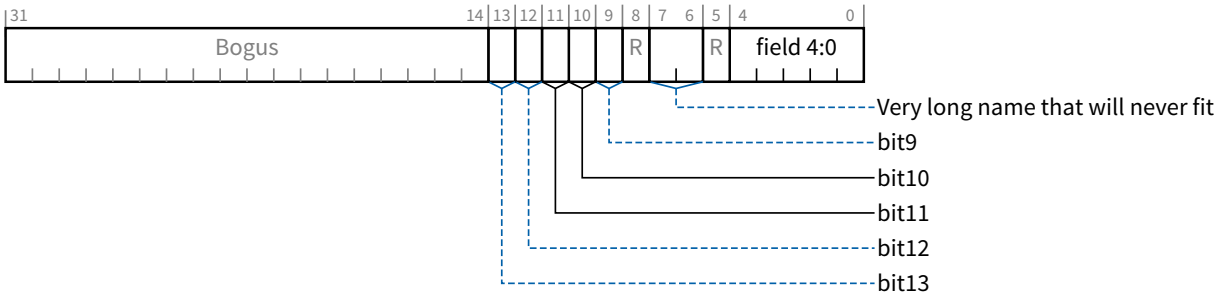

Figure 4-1: Graph for JSON Defined Register Example

These are references to fields that were defined as terms in this JSON example.

- field 4:0 field 4:0
- Very long name that will never fit Very long name that will never fit
- bit9 bit9
- bit10 bit10
- bit11 bit11
- bit12 bit12
- bit13 bit13
- bit13 <a>bit13
- Bit 13 Bit 13

5. PCISIG style Registers §

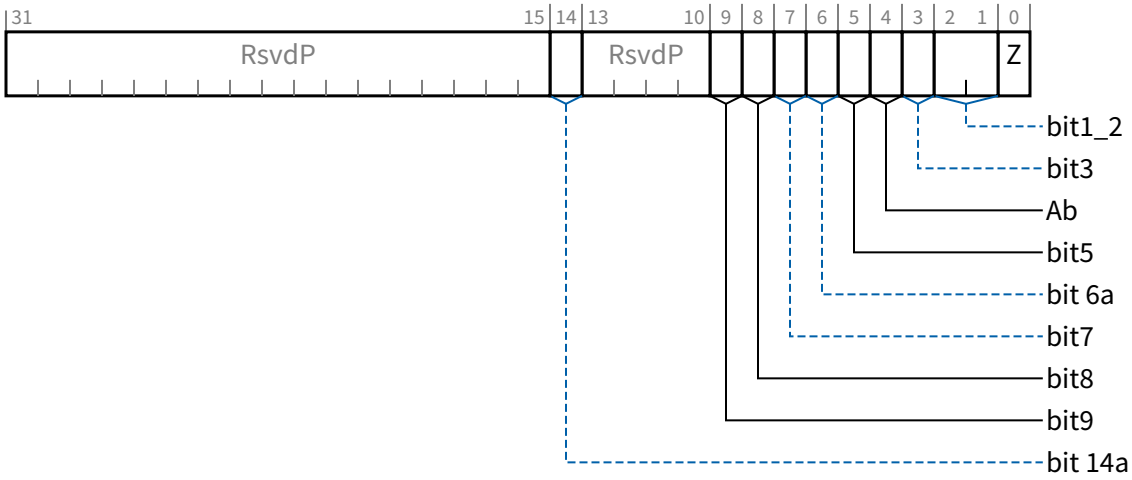

Figure 5-1: PCISIG-Style Register #1

Table 5-1: PCISIG-Style Register #1

Bit Location	Register Description	Attributes
0	**Z** – This field is a very special field. This is the middle paragraph. This is the last paragraph.	RO
2:1	**bit1_2** – Bits 1 & 2	RW
3	**bit3** – three	RW
4	**Ab** – four	RW

Bit Location	Register Description	Attributes
5	bit5– five	RW
6	**bit 6a** – six a **bit 6b** – six b bit 6c six c	RW
7	**bit7**	RW
8	**bit8** - eight	RW
9	**bit9**	RW
14	**bit 14a** Col 1 | Col 2 | Col 3 **first** | **second** | **third** second row **fourth**	RW

6. Device Capabilities 2 Register (Offset 24h) §

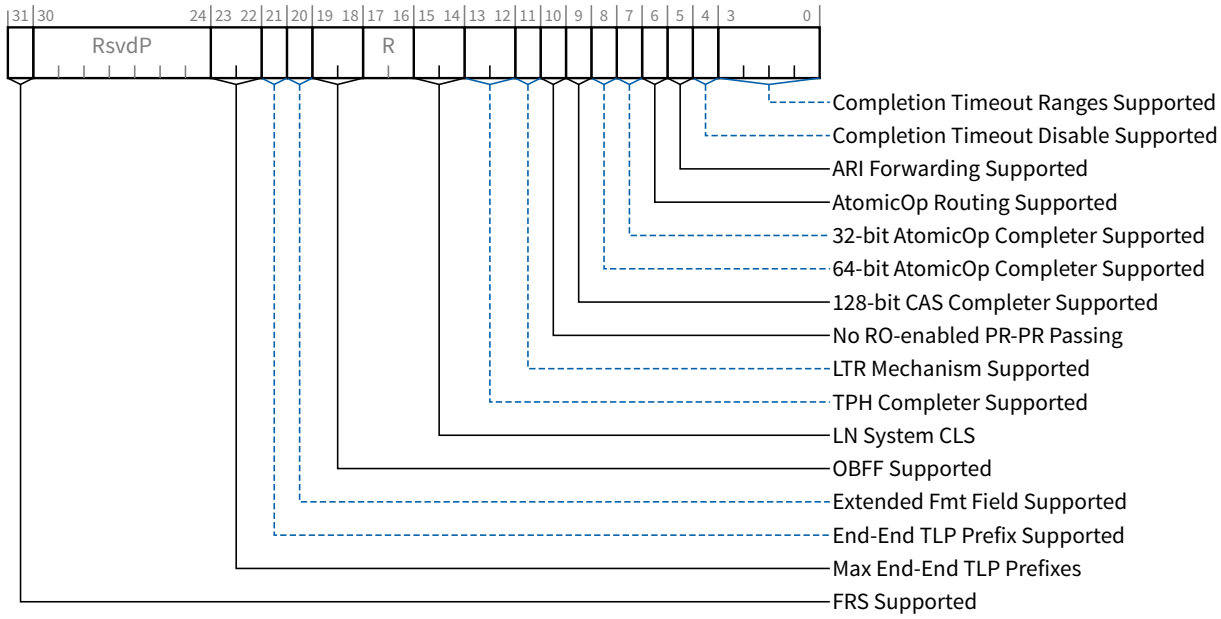

Figure 6-1: Device Capabilities 2 Register

Table 6-1: Device Capabilities 2 Register

Bit Location	Register Description	Attributes
3:0	**Completion Timeout Ranges Supported** – This field indicates device Function support for the optional Completion Timeout programmability mechanism. This mechanism allows system software to modify the Completion Timeout value. This field is applicable only to Root Ports, Endpoints that issue Requests on their own behalf, and PCI Express to PCI/PCI-X Bridges that take ownership of Requests issued on PCI Express. For all other Functions this field is Reserved and must be hardwired to 0000b. Four time value ranges are defined: **Range A** 50 µs to 10 ms	HwInit

Bit Location	Register Description	Attributes
	Range B 10 ms to 250 ms **Range C** 250 ms to 4 s **Range D** 4 s to 64 s Bits are set according to the table below to show timeout value ranges supported. **0000b** Completion Timeout programming not supported – the Function must implement a timeout value in the range 50 µs to 50 ms. **0001b** Range A **0010b** Range B **0011b** Ranges A and B **0110b** Ranges B and C **0111b** Ranges A, B, and C **1110b** Ranges B, C, and D **1111b** Ranges A, B, C, and D All other values are Reserved. It is strongly recommended that the Completion Timeout mechanism not expire in less than 10 ms.	
4	**Completion Timeout Disable Supported** – A value of 1b indicates support for the Completion Timeout Disable mechanism. The Completion Timeout Disable mechanism is required for Endpoints that issue Requests on their own behalf and PCI Express to PCI/PCI-X Bridges that take ownership of Requests issued on PCI Express. This mechanism is optional for Root Ports. For all other Functions this field is Reserved and must be hardwired to 0b.	RO
5	**ARI Forwarding Supported** – Applicable only to Switch Downstream Ports and Root Ports; must be 0b for other	RO

Bit Location	Register Description	Attributes
	Function types. This bit must be set to 1b if a Switch Downstream Port or Root Port supports this optional capability. See Section 6.13 for additional details.	
6	**AtomicOp Routing Supported** – Applicable only to Switch Upstream Ports, Switch Downstream Ports, and Root Ports; must be 0b for other Function types. This bit must be set to 1b if the Port supports this optional capability. See Section 6.15 for additional details.	RO
7	**32-bit AtomicOp Completer Supported** – Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. Includes FetchAdd, Swap, and CAS AtomicOps. This bit must be set to 1b if the Function supports this optional capability. See Section 6.15.3.1 for additional RC requirements.	RO
8	**64-bit AtomicOp Completer Supported** – Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. Includes FetchAdd, Swap, and CAS AtomicOps. This bit must be set to 1b if the Function supports this optional capability. See Section 6.15.3.1 for additional RC requirements.	RO
9	**128-bit CAS Completer Supported** – Applicable to Functions with Memory Space BARs as well as all Root Ports; must be 0b otherwise. This bit must be set to 1b if the Function supports this optional capability. See Section 6.15 for additional details.	RO
10	**No RO-enabled PR-PR Passing** – If this bit is Set, the routing element never carries out the passing permitted by Table 2-39 entry A2b that is associated with the Relaxed Ordering Attribute field being Set. This bit applies only for Switches and RCs that support peer-to-peer traffic between Root Ports. This bit applies only to Posted Requests being forwarded through the Switch or RC and does not apply to traffic originating or terminating within the Switch or RC itself. All Ports on a Switch or RC must report the same value for this bit. ISSUE 2: This is an issue	HwInit

Bit Location	Register Description	Attributes
	Clearly this applies when both requests are forwarded. What about when only one of the requests is forwarded? For all other functions, this bit must be 0b	
11	**LTR Mechanism Supported** – A value of 1b indicates support for the optional Latency Tolerance Reporting (LTR) mechanism. Root Ports, Switches and Endpoints are permitted to implement this capability. For a multi-Function device associated with an Upstream Port, each Function must report the same value for this bit. For Bridges and other Functions that do not implement this capability, this bit must be hardwired to 0b.	RO
13:12	**TPH Completer Supported** – Value indicates Completer support for TPH or Extended TPH. Applicable only to Root Ports and Endpoints. For all other Functions, this field is Reserved. Defined Encodings are: **00b** TPH and Extended TPH Completer not supported. **01b** TPH Completer supported; Extended TPH Completer not supported. **10b** Reserved. **11b** Both TPH and Extended TPH Completer supported. See Section 6.17 for details.	RO
15:14	**LN System CLS** – Applicable only to Root Ports and RCRBs; must be 00b for all other Function types. This field indicates if the Root Port or RCRB supports LN protocol as an LN Completer, and if so, what cacheline size is in effect. Encodings are: **00b** LN Completer either not supported or not in effect **01b** LN Completer with 64-byte cachelines in effect **10b** LN Completer with 128-byte cachelines in effect **11b** Reserved	HWInit

Bit Location	Register Description	Attributes
19:18	**OBFF Supported** – This field indicates if OBFF is supported and, if so, what signaling mechanism is used. **00b** OBFF Not Supported **01b** OBFF supported using Message signaling only **10b** OBFF supported using WAKE# signaling only **11b** OBFF supported using WAKE# and Message signaling The value reported in this field must indicate support for WAKE# signaling only if: • for a Downstream Port, driving the WAKE# signal for OBFF is supported and the connector or component connected Downstream is known to receive that same WAKE# signal • for an Upstream Port, receiving the WAKE# signal for OBFF is supported and, if the component is on an add-in-card, that the component is connected to the WAKE# signal on the connector. Root Ports, Switch Ports, and Endpoints are permitted to implement this capability. For a multi-Function device associated with an Upstream Port, each Function must report the same value for this field. For Bridges and Ports that do not implement this capability, this field must be hardwired to 00b.	Hwinit
20	**Extended Fmt Field Supported** – If Set, the Function supports the 3-bit definition of the Fmt field. If Clear, the Function supports a 2-bit definition of the Fmt field. See Section 2.2. Must be Set for Functions that support End-End TLP Prefixes. All Functions in an Upstream Port must have the same value for this bit. Each Downstream Port of a component may have a different value for this bit. It is strongly recommended that Functions support the 3-bit definition of the Fmt field.	RO
21	**End-End TLP Prefix Supported** – Indicates whether End-End TLP Prefix support is offered by a Function. Values are: **0b** No Support	HwInit

Bit Location	Register Description	Attributes
	1b Support is provided to receive TLPs containing End-End TLP Prefixes. All Ports of a Switch must have the same value for this bit.	
23:22	**Max End-End TLP Prefixes** – Indicates the maximum number of End-End TLP Prefixes supported by this Function. See Section 2.2.10.2 for important details. Values are: **01b** 1 End-End TLP Prefix **10b** 2 End-End TLP Prefixes **11b** 3 End-End TLP Prefixes **00b** 4 End-End TLP Prefixes If End-End TLP Prefix Supported is Clear, this field is RsvdP. Different Root Ports that have the End-End TLP Prefix Supported bit Set MAY are permitted to report different values for this field. For Switches where End-End TLP Prefix Supported is Set, this field MUST be 00b indicating support for up to four End-End TLP Prefixes.	HwInit
31	**FRS Supported** – When Set, indicates support for the optional Function Readiness Status (FRS) capability. Must be Set for all Functions that support generation or receipt capabilities of FRS Messages. Must not be Set by Switch Functions that do not generate FRS Messages on their own behalf.	HwInit

IMPLEMENTATION NOTE: Use of the No RO-enabled PR-PR Passing Bit

The No RO-enabled PR-PR Passing bit allows platforms to utilize PCI Express switching elements on the path between a requester and completer for requesters that could benefit from a slightly less 5 relaxed ordering model. An example is a device that cannot ensure that multiple overlapping posted writes to the same address are outstanding at the same time. The method by which such a device is enabled to utilize this mode is beyond the scope of this specification.

7. Name clashes after and deep stuff §

Something MUST happen.

ISSUE 3: A real problem

This is a problem

FEATURE AT RISK 4: A REAL problem

This is a problem

NOTE: About this

Just a note

IMPLEMENTATION NOTE: Imp Note Text

Body of the implementation note

A. Definition Map §

Kind	Name	ID	HTML
dfn	term number 3	dfn-term-number-3	TERM3
dfn	term number 4	dfn-term-number-4	Term number 4
dfn	term number 8	dfn-term-number-8	Term number 8
dfn	term1	dfn-term1	Term1
dfn	term4	dfn-term4	Term4
dfn	term6	dfn-term6	Term6
dfn	term7	dfn-term7	Term7
dfn	term8	dfn-term8	Term8
field	128-bit cas completer supported	field-x128-bit-cas-completer-supported	128-bit CAS Completer Supported
field	32-bit atomicop completer supported	field-x32-bit-atomicop-completer-supported	32-bit AtomicOp Completer Supported

Kind	Name	ID	HTML
field	64-bit atomicop completer supported	field-x64-bit-atomicop-completer-supported	64-bit AtomicOp Completer Supported
field	ab	field-ab	Ab
field	ari forwarding supported	field-ari-forwarding-supported	ARI Forwarding Supported
field	atomicop routing supported	field-atomicop-routing-supported	AtomicOp Routing Supported
field	bit 14a	field-bit-14a	bit 14a
field	bit 6a	field-bit-6a	bit 6a
field	bit 6b	field-bit-6b	bit 6b
field	bit10	bit10	bit10
field	bit11	field-json-register-bit11	bit11
field	bit12	bit12	bit12
field	bit13	field-json-register-bit13	bit13
field	bit1_2	field-bit1_2	bit1_2
field	bit3	field-bit3	bit3
field	bit5	field-fig-pcisig-style-register-1-bit5	bit5
field	bit7	field-bit7	bit7
field	bit8	field-bit8	bit8
field	bit9	field-bit9	bit9
field	bogus	field-json-register-Bogus	Bogus
field	completion timeout disable supported	field-completion-timeout-disable-supported	Completion Timeout Disable Supported
field	completion timeout ranges supported	field-completion-timeout-ranges-supported	Completion Timeout Ranges Supported
field	end-end tlp prefix supported	field-end-end-tlp-prefix-supported	End-End TLP Prefix Supported

Kind	Name	ID	HTML
field	extended fmt field supported	field-extended-fmt-field-supported	Extended Fmt Field Supported
field	field 4:0	field-field4_0	field 4:0
field	first	field-first	first
field	fourth	field-fourth	fourth
field	frs supported	field-frs-supported	FRS Supported
field	ln system cls	field-ln-system-cls	LN System CLS
field	ltr mechanism supported	field-ltr-mechanism-supported	LTR Mechanism Supported
field	max end-end tlp prefixes	field-max-end-end-tlp-prefixes	Max End-End TLP Prefixes
field	no ro-enabled pr-pr passing	field-no-ro-enabled-pr-pr-passing	No RO-enabled PR-PR Passing
field	obff supported	field-obff-supported	OBFF Supported
field	r	field-json-register-R	R
field	rsvdp	field-fig-pcisig-style-register-1-RsvdP	RsvdP
field	second	field-second	second
field	third	field-third	third
field	tph completer supported	field-tph-completer-supported	TPH Completer Supported
field	very long name that will never fit	field-json-register-Very long name that will never fit	Very long name that will never fit
field	z	field-z	Z
pin	pin2#	pin-pin2	PIN2#
pin	pin2a#	pin-pin2a	PIN2a#
pin	term9	pin-term9	Term9
signal	term9	signal-term9	Term9
state	d3cold	state-d3cold	$D3_{cold}$

Kind	Name	ID	HTML
state	detect.quiet	state-detect.quiet	Detect.Quiet
term	term 9	term-term-9	Term 9
term	term9	term-term9-1	Term9 `Duplicate definition of 'term-term9'`