TRUSTED® COMPUTING GROUP

SPECIFICATION

DICE Protection Environment Command Response Buffer

Version 1.0
November 7, 2025

Contact: admin@trustedcomputinggroup.org

PUBLISHED

© TCG 2025

DISCLAIMERS, NOTICES, AND LICENSE TERMS

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.

Without limitation, TCG disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification and to the implementation of this specification, and TCG disclaims all liability for cost of procurement of substitute goods or services, lost profits, loss of use, loss of data or any incidental, consequential, direct, indirect, or special damages, whether under contract, tort, warranty or otherwise, arising in any way out of use or reliance upon this specification or any information herein.

This document is copyrighted by Trusted Computing Group (TCG), and no license, express or implied, is granted herein other than as follows: You may not copy or reproduce the document or distribute it to others without written permission from TCG, except that you may freely do so for the purposes of (a) examining or implementing TCG specifications or (b) developing, testing, or promoting information technology standards and best practices, so long as you distribute the document with these disclaimers, notices, and license terms.

Contact the Trusted Computing Group at www.trustedcomputinggroup.org for information on specification licensing through membership agreements.

Any marks and brands contained herein are the property of their respective owners.

CONTENTS

DISCLAIMERS, NOTICES, AND LICENSE TERMS 1
CONTENTS 2
1 SCOPE 4
1.1 Key Words 4
1.2 Statement Type 4
2 REFERENCES 5
3 TERMS AND DEFINITIONS 6
3.1 Glossary 6
3.2 Acronyms 6
4 INTRODUCTION 7
4.1 Command Protocol 7
4.2 Discoverability 8
4.3 Implementation Consideration for Multiple Localities 9
4.4 Command Duration 9
4.5 Interface Timeouts 10
5 CRB Interface Identifier 11
5.1 Field: DID 12
5.2 Field: VID 12
5.3 Field: RID 12
5.4 Field: CRBSizeExp 12
5.5 Field: InterfaceControlDisable 12
5.6 Field: CapCancel 12
5.7 Field: CommandDurationTimeoutExp 13
5.8 Field: CapDataXferSizeSupport 13
5.9 Field: CapCRBIdleBypass 13
5.10 Field: InterfaceVersion 13
5.11 Field: NumLocalities 13
6 CRB Interface Requirements 15
6.1 CRB Addresses 15
6.2 Interface Control Area 19
6.3 DPE CRB Control Area Request Register 20
6.3.1 Field: cmdReady 21
6.3.2 Field: goIdle 21
6.4 Status Register (DPE_CRB_CTRL_STS_x) 22
6.4.1 Field: dpeIdle 23

 © TCG 2025

6.4.2 Field: dpeSts 23
6.5 Cancel Register (DPE_CRB_CTRL_CANCEL_x) 23
6.6 Start Register (DPE_CRB_CTRL_START_x) 24
6.7 Command Size (DPE_CRB_CTRL_CMD_SIZE_x) 25
6.8 Command Address (DPE_CRB_CTRL_CMD_LADDR_x and DPE_CRB_CTRL_CMD_HADDR_x) 25
6.9 Response Size (DPE_CRB_CTRL_RSP_SIZE_x) 25
6.10 Response Address (DPE_CRB_CTRL_RSP_ADDR_x) 26
6.11 DPE CRB Data Buffer 26
6.11.1 Access Restrictions 26
6.11.2 DPE_CRB_DATA_BUFFER_x Access Restrictions 26
6.12 Interface Controls 26
6.12.1 CRB Interface State Transitions 28
6.13 Reset Timing 31
Appendix A: DPE Interfaces and Localities 32

 © TCG 2025

1 SCOPE

This specification defines a hardware interface between a client and a DICE Protection Environment (DPE) [1] implementation that can be used to transport either direct or message-based DPE interface commands and responses. This hardware interface is the Command Response Buffer (CRB) interface. The CRB design can work with many hardware implementation options. Additionally, this specification includes a mechanism for differentiating DPE localities (refer to [1]) over the CRB interface. Designers, developers, and implementers of Trusted Computing technologies are the target audience for this specification.

1.1 Key Words

The key words “MUST,” “MUST NOT,” “REQUIRED,” “SHALL,” “SHALL NOT,” “SHOULD,” “SHOULD NOT,” “RECOMMENDED,” “MAY,” and “OPTIONAL” in this document normative statements are to be interpreted as described in RFC-2119, Key words for use in RFCs to Indicate Requirement Levels.

1.2 Statement Type

Please note a very important distinction between different sections of text throughout this document. There are two distinctive kinds of text: informative comment and normative statements. Because most of the text in this specification will be of the kind normative statements, the authors have informally defined it as the default and, as such, have specifically called out text of the kind informative comment. They have done this by flagging the beginning and end of each informative comment and highlighting its text in gray. This means that unless text is specifically marked as of the kind informative comment, it can be considered a kind of normative statement.

EXAMPLE:

Start of informative comment

This is the first paragraph of 1–n paragraphs containing text of the kind *informative comment* ...

This is the second paragraph of text of the kind *informative comment* ...

This is the nth paragraph of text of the kind *informative comment* ...

To understand the TCG specification the user must read the specification. (This use of MUST does not require any action).

End of informative comment

 © TCG 2025

2 REFERENCES

[1] Trusted Computing Group, "DICE Protection Environment Version 1.0 Revision 0.14," 25 October 2024. [Online]. Available: https://trustedcomputinggroup.org/resource/dice-protection-environment/.

[2] Trusted Computing Group, "Hardware Requirements for Device Identifier Composition Engine Version 1.0, Revision 0.91," 8 August 2024. [Online]. Available: https://trustedcomputinggroup.org/resource/hardware-requirements-for-a-device-identifier-composition-engine/.

[3] Trusted Computing Group, "DICE Layering Architecture Version 1.0 Revision 0.19," 23 July 2020. [Online]. Available: https://trustedcomputinggroup.org/resource/dice-layering-architecture/.

[4] Trusted Computing Group, "DICE Attestation Architecture Version 1.1 Revision 18," 6 January 2024. [Online]. Available: https://trustedcomputinggroup.org/resource/dice-attestation-architecture/.

[5] Trusted Computing Group, "DICE Certificate Profiles Version 1.0 Revision 0.01," 23 July 2020. [Online]. Available: https://trustedcomputinggroup.org/resource/dice-certificate-profiles/.

[6] Trusted Computing Group, "TCG Glossary Version 1.1 Revision 1.0," 11 May 2017. [Online]. Available: https://trustedcomputinggroup.org/resource/tcg-glossary/.

[7] Trusted Computing Group, "TCG ACPI Specification Version 1.5," [Online]. Available: https://trustedcomputinggroup.org/resource/tcg-acpi-specification/.

[8] Trusted Computing Group, "TCG PC Client Platform TPM Profile (PTP) Specification," 4 September 2020. [Online]. Available: https://trustedcomputinggroup.org/resource/pc-client-platform-tpm-profile-ptp-specification/.

 © TCG 2025

3 TERMS AND DEFINITIONS

For the purposes of this specification, the following terms and definitions apply. This specification assumes the reader is familiar with the TCG DICE specifications [1] [2] [3] [4] [5] and common Trusted Computing terminology, as defined in [6].

3.1 Glossary

GLOSSARY TERM	DESCRIPTION
CLEAR	Bit with a value of zero (0), or the action of causing a bit to have the value of zero (0).
Command	The values sent to the DPE to indicate the operation to be performed
Command Duration	The time between a DPE's receipt of a 1 in DPE_CRB_CTRL_START_x and a DPE completing the command as evidenced by the DPE clearing DPE_CRB_CTRL_START_x to 0.
CRB Data Buffer	Data structure used to transport DPE commands and responses to and from the DPE (see Section 6.11).
Response	Values returned by the DPE when it completes processing of a command.
SET	Bit with a value of one (1), or the action of causing a bit to have the value of one (1).

3.2 Acronyms

ABBREVIATIONS	DESCRIPTION
CRB	Command Response Buffer
DICE	Device Identifier Composition Engine [2]
DID	Device Identifier
DPE	DICE Protection Environment [1]
IP	Intellectual Property, e.g., IP Block
RID	Revision Identifier
SoC	System on Chip
TCG	Trusted Computing Group
VID	Vendor Identifier

 © TCG 2025

4 INTRODUCTION

Start of informative comment

A DICE Protection Environment (DPE) [1] protects DICE-related secrets and helps enforce DICE-related policies. The DPE specification does not define a standard interface of DPE to software clients running on the host CPU. Without a standard interface, software clients might have to implement different protocols for interacting with DPE implementations of different silicon vendors. This would duplicate effort for development, validation, and security analysis.

The basic concept for the Command Response Buffer (CRB) interface is that the DPE's CRB is defined as a linear range of memory. The client performs the actual device interface access, which, in the case of this specification, implies the manipulation of the CRB interface. The client inserts command data in the DPE's CRB Data Buffer by writing to consecutive addresses in the processor's address space. Similarly, the client reads response data from the DPE's CRB Data Buffer by reading from consecutive addresses in the processor's linear address space.

The client is restricted to writing the bytes of the CRB Data Buffer to consecutive addresses in order to enable the DPE to process the data in a consecutive order. The control flow for the CRB is a simple handshake through a memory-mapped Interface Control structure (refer to Section 6.2 Interface Control Area). The Interface Control structure is defined such that it can either be implemented in memory or hardware registers of the DPE. The Interface Control structure and associated memory areas used to transport commands and command responses comprise the entirety of the CRB. This specification places no constraints on how those memory areas are provided. The memory areas can all be in system RAM, or all be in device memory, or any combination.

End of informative comment

1) The Interface Control structure needs to occupy a unique address space, but the command and response buffers (known as the CRB Data Buffer) SHALL occupy the same address space.

4.1 Command Protocol

Start of informative comment

The CRB command protocol works as follows:

1. A client places a DPE command to be executed into the CRB Data Buffer for a specific locality and SETs Start in the Interface Control structure for that locality. Refer to [1] for more details on DPE localities. Also, see Section 4.3 for implementation considerations with multiple localities exposed via the CRB interface.
2. The DPE will detect that Start has been SET and process the command in the CRB Data Buffer. While Start is SET, the client should not access the CRB Data Buffer.
3. When the DPE has completed processing the command, it puts a response in the CRB Data Buffer before clearing Start. While Start is CLEARed, the DPE does not access the CRB Data Buffer.
4. After a client SETs Start, the client is required to poll Start to determine when the DPE has completed processing of the command.

The DPE processes one command per locality at a time. The order that the DPE processes commands across localities is implementation specific.

A client can signal to the DPE that the DPE can enter a lower power state by setting goIdle. When a client has a command for the DPE, the client should SET cmdReady and wait for Status to CLEAR. When Status is CLEARed, the client can once again load the command buffer with a new DPE command.

See the remaining sections of this document for details of the CRB protocol.

End of informative comment

1) A DPE implementation SHALL only access the CRB Data Buffer for a given locality when Start has been SET for that locality.

 © TCG 2025

2) If a DPE supports entering a lower power state and has powered down or entered a low-power state, then the DPE SHALL become functional again when cmdReady has been SET on any supported locality.
3) To signal to the client that the DPE is ready to receive commands again, the DPE SHALL CLEAR Status.

4.2 Discoverability

Start of informative comment

DPE support for the CRB interface depends on the DPE implementation.

Because the CRB interface has the same layout for all localities, implementations only need to publish the base address for locality 0 of the CRB interface registers.

This specification is only capable of supporting at most 16 localities over the CRB interface.

A DPE's CRB interface registers are duplicated for each supported locality. The system base address for locality 0 (BASE_ADDRESS_0) is decided by the silicon vendor and documented in the DPE's user manual.

In this specification, "localities" refer to the localities implemented by a DPE and accessed by the client over the CRB interface.

The DPE's supported localities comprise CRB localities and other DPE interface localities. See Appendix A: DPE Interfaces and Localities for more details on DPE interfaces and localities in relation to the CRB interface.

The DPE Profile specification [1] defines the mapping of DPE locality to CRB locality number (0 - 15).

End of informative comment

1) A DPE implementation MAY publish the base of the CRB interface registers at a fixed address.BASE_ADDRESS_0 MAY be included in the TCG ACPI Table for DPE 1.0 (refer to [7]) for software to discover.
2) The number of localities supported by the DPE MAY be obtained by reading locality 0's CRB interface identifier register.
3) A DPE MAY implement other interfaces in addition to the CRB and, accordingly, other localities. Address Layout

Start of informative comment

Each supported locality implements six areas of CRB interface registers. The layout of registers for each locality is identical. The six areas include:

- Reserved (48 bytes, offsets 0_0000h – 0_002Fh)
- CRB Interface Identifier (8 bytes, offsets 0_0030h – 0_0037h, refer to Section 5)
- Reserved (8 bytes, offsets 0_0038h – 0_003Fh)
- Interface Control (48 bytes, offsets = 0_0040h – 0_006Fh, refer to Section 6.2)
- Reserved (16 bytes, offsets 0_0070h – 0_007Fh)
- Command Response Data Buffer ((LocalitySize-128) bytes, for offset, refer to Sections 6.8 and 6.10).

Each locality's CRB interface registers consume LocalitySize bytes in the system memory. The following sections of this specification define these areas of registers, respectively.

The address layout is intended to align with the TPM CRB interface layout (refer to [8]). The Reserved address areas are not required by functionality defined in this specification. The TPM CRB uses the first Reserved area (48 bytes, from 0_000h to 0_002Fh) for Locality Control when all localities are overlayed over the same memory location. This specification instead defines each locality's CRB in its own address space, so Locality Control is not needed. The TPM CRB uses the second Reserved area (8 bytes, from 0_0038h to 0_003Fh) for a CRB Control Area Extension to aid with transferring command/response data in chunks. This specification does not support this

 © TCG 2025

CRB Control Area Extension and chunking mechanism. The TPM CRB also reserves the third Reserved area (16 bytes, from 0_0070h to 0_007Fh).

The base address of locality 0 is BASE_ADDRESS_0 which can be identified either in system vendor documentation or discovered via the TCG ACPI Table for DPE 1.0 (refer to [7]). The base address for all other localities can be calculated from this base address using the following formula:

BASE_ADDRESS_x = BASE_ADDRESS_0 + x * LocalitySize, where x is the target locality.

For example, locality 1's base address can be calculated as follows:

BASE_ADDRESS_1 = BASE_ADDRESS_0 + 1 * LocalitySize.

The value of LocalitySize determines the size of the CRB per locality supported. This specification supports LocalitySize in the range of 1K to 512K bytes, inclusive. Refer to the description of CRBSizeExp in Table 2 for details on how to calculate LocalitySize based on the value reported in the CRBSizeExp field of the CRB Interface Identifier register.

A DPE reports its supported locality count over the CRB interface as DPE_CRB_INTF_ID_0.NumLocalities field + 1.

For example, a DPE that supports 1 locality over the CRB interface would reports a value of 0 in DPE_CRB_INTF_ID_0.NumLocalities; a DPE that supports 2 localities reports a value of 1, and so on.

The DPE_CRB_INTF_ID_0.NumLocalities field is 4-bits wide (i.e., a DPE compliant with this specification can support at most 16 localities over the CRB interface).

End of informative comment

1) The client SHOULD read DPE_CRB_INTF_ID_0.NumLocalities to make sure a locality beyond locality 0 exists, before accessing the CRB interface for that locality.

4.3 Implementation Consideration for Multiple Localities

Start of informative comment

For a DPE that supports multiple localities (see [1]), this specification does not require coordination between multiple clients concurrently accessing different localities of the DPE CRB interface.

See Section 6.1 for more details on locality address layout when multiple CRB localities are supported.

End of informative comment

1) A DPE that supports the CRB interface SHALL support at least one locality over the CRB interface.
2) All supported localities SHALL be accessible by clients and function independently without relying on shared resources that are used by other localities.
3) The CRB Data Buffer for a given locality SHALL NOT be shared with other localities.
4) A DPE implementation that supports multiple localities over the CRB interface SHALL expose the CRB localities sequentially.
5) That is, a DPE that supports two CRB localities SHALL support localities 0 and 1 over the CRB interface and a DPE that supports five CRB localities SHALL support localities 0, 1, 2, 3, and 4 over the CRB interface.

4.4 Command Duration

Start of informative comment

It is important to distinguish between the terms: Command Duration and Command Duration Timeout. Command Duration is the amount of time a DPE takes to execute a command once the DPE has received the command's complete set of bytes and the client SETS DPE_CRB_CTRL_START_x to start the operation.

Command Duration has no relationship to the timing of the interface protocol. Command Duration Timeout, referenced below, is not related to interface timeouts, as defined in Section 4.5 Interface Timeouts. Command Duration Timeout allows DPE clients to know when to cancel a command in an attempt to recover a DPE. If a DPE

 © TCG 2025

fails to cancel the command within TIMEOUT_A (for commands that do not cause RSA key generation) or TIMEOUT_B (for commands that cause RSA key generation) or complete the command within the Command Duration Timeout, the client may safely assume a DPE has encountered a problem and may not be recoverable.

End of informative comment

1) A DPE SHALL report its Command Duration Timeout in the CommandDurationTimeoutExp field defined in Section 5.7 Field: CommandDurationTimeoutExp.
2) A normal operating DPE SHALL NOT exceed the Command Duration Timeout value reported in the CommandDurationTimeoutExp field.

4.5 Interface Timeouts

Start of informative comment

The term timeout, in this section, applies to timings between various states or transitions within the interface protocol. Interface timeout values are not related to duration (see Section 4.4 Command Duration).

DPE clients should be able to determine when a DPE has failed and is no longer expected to respond. The timeout values in Table 1 were chosen to be sufficiently large, allowing a wide variety of DPE implementations.

Note: TIMEOUT_B applies to the time it takes to cancel a command. As canceling a command affects more than the interface, this is the only timeout that applies to higher levels of a DPE client stack.

End of informative comment

1) There are four timeout values designated: TIMEOUT_A, TIMEOUT_B, TIMEOUT_C, and TIMEOUT_D.
2) The values for the timeouts SHALL be as shown in Table 1.

Table 1 – Definition of Interface Timeouts

TIMEOUT Label	Timeouts
TIMEOUT_A	750 milliseconds
TIMEOUT_B	2,000 milliseconds
TIMEOUT_C	200 milliseconds
TIMEOUT_D	30 milliseconds

 © TCG 2025

5 CRB Interface Identifier

Start of informative comment

A client can query the CRB Interface Identifier register to determine the vendor ID, device ID, revision ID, interface version, number of localities supported by the DPE, size of CRB for each locality, and whether the CRB Interface Control registers are currently enabled.

The InterfaceControlDisable field is unique per CRB locality to allow the DPE interface to be disabled on a per locality basis.

See Section 5.5 for more details on the InterfaceControlDisable field.

End of informative comment

1) A DPE SHALL support a CRB Interface Identifier register as defined in Table 2 for each CRB locality supported by the DPE.
2) All fields of the CRB Interface Identifier register except the InterfaceControlDisable field SHALL be aliased across all supported localities.

Table 2 – CRB Interface Identifier Register DPE_CRB_INTF_ID_x (Where x is the Locality)

Abbreviation:			DPE_CRB_INTF_ID_x
General Description:			Interface Identifier Register
Bits	**Read / Write**	**Name**	**Description**
63:48	Read Only	DID	Device ID – vendor specific
47:32	Read Only	VID	Vendor ID – assigned by TCG. This is represented within the register in big-endian format. For example, a vendor ID of 0x1234 would be represented as: Bits 7:0 = 34 (0011 0100); Bits 15:8 = 12 (0001 0010).
31:24	Read Only	RID	Revision ID – specifies the revision of the DPE device. Vendor specific.
23:20	Read Only	CRBSizeExp	A 4-bit integer in big-endian format. This value is used to calculate the size (in bytes) of the CRB interface for one locality. $LocalitySize = 2^{CRBSizeExp}$ K bytes For this version of the specification, CRBSizeExp = 0-9 are valid values, which indicates LocalitySize = 1K to 512K bytes. Values of 10-15 are reserved.
19	Read / Write	InterfaceControlDisable	0 – Interface Control for this locality is enabled 1 – Interface Control for this locality is disabled. The default value of this field on power on or initialization is 0.
18	Read Only	CapCancel	0 – The DPE does not support Cancel. 1 – The DPE supports Cancel.
17	Read Only	Reserved	This field is 0.
16:13	Read Only	TimeoutExp	A 4-bit integer in big-endian format. This value is used to calculate the timeout (in millisecond) of the DPE. $Timeout = 2^{TimeoutExp}$ milliseconds
12:11	Read Only	CapDataXferSizeSupport	00 – The DPE supports a 4-byte transfer size only. 01 – The DPE supports both 8-byte and 4-byte transfers. 10 – The DPE supports 32-byte, 8-byte, and 4-byte transfers. 11 – The DPE supports a 64-byte, 32-byte, 8-byte, and 4-byte transfers.
10	Read Only	Reserved	This field is 0.

 © TCG 2025

9	Read Only	CapCRBIdleBypass	0 – The DPE supports fast transition from the Idle to the Command Ready state 1 – The DPE supports a transition from the Command Completion state directly to the Ready state and supports a transition from the Command Completion state directly to the Command Reception state.
8	Read Only	Reserved	This field is 0.
7:4	Read Only	InterfaceVersion	The CRB interface version in big-endian format. For this version of the specification, InterfaceVersion is 0001.
3:0	Read Only	NumLocalities	The number of localities supported by the DPE minus 1, in big-endian format. For example, a value of 3 means the DPE supports 4 localities (locality 0, locality 1, locality 2, and locality 3).

5.1 Field: DID

1) A DPE SHALL report a 16-bit vendor-specific device ID in the DID field.

5.2 Field: VID

1) A DPE SHALL report the vendor's 16-bit TCG-assigned vendor ID in the VID field.
2) A DPE profile MAY define commands that also return a vendor ID.
3) The vendor ID in this field and the vendor ID returned in the DPE profile command response MAY NOT be identical but SHALL represent the same vendor.

5.3 Field: RID

1) A DPE SHALL report the 8-bit revision number of the DPE device in the RID field.

5.4 Field: CRBSizeExp

Start of informative comment

The CRBSizeExp field provides the DPE with flexibility of supporting CRB Data Buffers of different sizes.

End of informative comment

1) For this version of this specification, the size of CRB for any locality SHALL be configured to be in the range 1K to 512K bytes.

5.5 Field: InterfaceControlDisable

Start of informative comment

The InterfaceControlDisable field allows a client to disable the CRB Interface Control Area for a specific locality until next DPE reset, thereby disabling the ability for a client to communicate with the DPE on that locality. If InterfaceControlDisable is SET to 1, a client can read the DPE_CRB_INTF_ID_x registers, but the DPE will ignore writes to the Interface Control Area (see Section 6.2) for this locality.

End of informative comment

1) If a client SETs InterfaceControlDisable to 1 for a specific locality, then the DPE SHALL ignore writes to the Interface Control Area (see Section 6.2) for that locality until the DPE is reset and the DPE SHALL ignore further writes to the InterfaceControlDisable field until the DPE is reset.
2) When InterfaceControlDisable has a value of 1, a DPE SHALL return the current value of each register read from this locality, including InterfaceControlDisable.

5.6 Field: CapCancel

Start of informative comment

 © TCG 2025

The CapCancel field indicates whether a DPE supports “cancel”, i.e., allowing the client to abort an outstanding operation in process. Refer to Section 6.5 for more information.

End of informative comment

5.7 Field: CommandDurationTimeoutExp

Start of informative comment

The CommandDurationTimeoutExp field is used to calculate the Command Duration Timeout value for a DPE in milliseconds, with the value reported being a power of 2. A DPE that reports a CommandDurationTimeoutExp value of 0 indicates the DPE supports a Command Duration Timeout of 2^0 = 1 millisecond. A DPE that reports a TimeoutExp value of 15 indicates the DPE supports a timeout of 2^{15}= 32,768 milliseconds. If the DPE does not provide a response (by setting *Start* to 0) within this timeout after the client sends a command (by setting *Start* to 1), this is an indication that the DPE is not responding and the DPE needs to be reset to make it functional again.

End of informative comment

5.8 Field: CapDataXferSizeSupport

Start of informative comment

The CapDataXferSizeSupport field indicates the size of transfers supported by a DPE on its client hardware interface.

End of informative comment

5.9 Field: CapCRBIdleBypass

Start of informative comment

The CapCRBIdleBypass field indicates the behavior of a DPE in transitioning states from Command Completion to Ready, and from Command Completion to Command Reception. If CapCRBIdleBypass is CLEAR, a DPE will transition from Command Completion to Idle and on receiving a write of 1 to cmdReady, will transition to Ready. A DPE with CapCRBIdleBypass SET to 1 will accept a write of 1 to cmdReady from Command Completion to bypass the Idle state. A DPE with CapCRBIdleBypass SET to 1 will also accept data written to the CRB Data Buffer, bypassing the Idle and Ready states, and going directly to Command Reception. There is no differentiating indicator in the interface between Command Completion and Ready, so the client will need to keep track of the state in which it leaves the DPE. It is expected that the client will only exercise this bypass mechanism when multiple commands are queued up for the DPE, as the DPE might perform necessary background processing while in Idle.

A client should take care when using this capability as DPEs require the Idle state to perform clean-up operations, e.g., freeing up NVRAM or entering low-power mode. DPEs that provide this capability need to account for the possibility they might not enter Idle for extended periods of time.

End of informative comment

1) If a DPE supports transitions from the Command Completion state to the Ready state and from the Command Completion state to the Command Reception state, CapCRBIdleBypass SHALL be SET to 1.
2) If the DPE supports fast transition from the Idle state to the Ready state, CapCRBIdleBypass SHALL be CLEAR.

5.10 Field: InterfaceVersion

1) A DPE SHALL report the CRB interface version in the InterfaceVersion field.
2) A DPE that implements this version of this specification SHALL return a value of 1 in the InterfaceVersion field.

5.11 Field: NumLocalities

Start of informative comment

The NumLocalities field indicates the number of localities over the CRB interface supported by a DPE, minus 1. A value of 0 implies that the DPE supports only one locality (locality 0) over the CRB interface.

 © TCG 2025

See Section 4.3 for more details on supporting multiple DPE localities over the CRB interface.

End of informative comment

1) A DPE compliant with this specification SHALL support at least one locality over the CRB interface.
2) A DPE SHALL NOT support more than 16 localities (localities 0 – 15) over the CRB interface.

 © TCG 2025

6 CRB Interface Requirements

6.1 CRB Addresses

Start of informative comment

A DPE compliant with this specification will implement the CRB Data Buffer at the specified location for DPE_CRB_DATA_BUFFER_x in either Table 3 or Table 4.

The CRB registers DPE_CRB_CTRL_CMD_LADDR_x, DPE_CRB_CTRL_CMD_HADDR_x, and DPE_CRB_CTRL_RSP_ADDR_x return the address of the DPE_CRB_DATA_BUFFER_x on read requests. For a DPE compliant with the CRB Data Buffer co-located with the Interface Control Area, the CRB locality address layout is depicted in Table 3. For a DPE CRB-compliant DPE with the CRB Data Buffer disjointed from the Interface Control registers, the CRB locality address layout is depicted in Table 4.

The fields within the CRB Interface are aligned so that 8-bit fields are located at 8-bit addressable locations and 64-bit fields are at 64-bit addressable locations. For this reason, the address for the CRB Data Buffer is separated into two 32-bit fields (DPE_CRB_CTRL_CMD_LADDR_x and DPE_CRB_CTRL_CMD_HADDR_x), because the start of the DPE_CRB_CTRL_CMD_LADDR_x is not on a 64-bit addressable boundary. An 8-bit addressable location located on a 64-bit boundary may not be accessible using a 64-bit transaction.

Some instantiating hardware may have size and alignment restrictions when accessing any of the fields in the CRB interface. Some hardware may require access to any of the data within a field or buffer be performed by reads or writes that are aligned to an address that is an integer multiple of the field's size in bits. For this reason, for best compatibility with most hardware implementations, a client should access the fields and buffers defined in this interface using instructions that do not cause an access to cross an alignment boundary and should not use string move instructions.

For example, accessing the Response Address using a 64-bit access will work; accessing the Command Address using a 64-bit access may not work.

End of informative comment

Table 3 Allocation of Register Space with CRB Data Buffer co-located with Interface Control Area

Offset from BASE_ADDRESS_0[1]	Register Name	Description
Locality 0		
002Fh – 0000h	Undefined	Reserved
0037h – 0030h	DPE_CRB_INTF_ID_0	Used to identify the Interface types supported by the DPE as well as the Vendor ID, Device ID, and Revision ID
003Fh – 0038h	Reserved	Reserved
0043h – 0040h	DPE_CRB_CTRL_REQ_0	A register used to initiate transactions for the CRB interface.
0047h – 0044h	DPE_CRB_CTRL_STS_0	A register used by the DPE to provide the status of the CRB interface.
004Bh – 0048h	DPE_CRB_CTRL_CANCEL_0	A register used by the client to cancel command processing.
004Fh – 004Ch	DPE_CRB_CTRL_START_0	A register used to indicate the presence of command or response data in the CRB buffer.
0057h – 0050h	Reserved	Reserved
005Bh – 0058h	DPE_CRB_CTRL_CMD_SIZE_0	The size of the CRB Data Buffer. This register SHALL be aliased across localities.
005Fh – 005Ch	DPE_CRB_CTRL_CMD_LADDR_0	The lower 32 bits of the CRB Data Buffer start address for Locality 0.

 © TCG 2025

Offset from BASE_ADDRESS_0[1]	Register Name	Description
0063h – 0060h	DPE_CRB_CTRL_CMD_HADDR_0	The upper 32 bits of the CRB Data Buffer start address for Locality 0.
0067h – 0064h	DPE_CRB_CTRL_RSP_SIZE_0	The size of the CRB Data Buffer. The value of this field SHALL be identical to the value in the DPE_CRB_CTRL_CMD_SIZE_0 buffer. This register SHALL be aliased across localities.
006Fh – 0068h	DPE_CRB_CTRL_RSP_ADDR_0	The address of the start of the CRB Data Buffer. This field SHALL contain the same address as that contained in DPE_CRB_CTRL_CMD_HADDR_0 and DPE_CRB_CMD_LADDR_0.
007Fh – 0070h	Reserved	Reserved
(LocalitySize-1) – 0080h[2]	DPE_CRB_DATA_BUFFER_0	The buffer allocated for Command and Response data to be transferred to/from a DPE via the DPE CRB. This buffer SHALL be a single buffer used for both command and response transfers. Command/Response Data might be defined as large as 130944 bytes. This is implementation specific. This field accepts data transfers from 1B up to the size indicated by DPE_CRB_INTF_ID_0.CapDataXferSizeSupport (see Section 5.8).
Locality 1		
(LocalitySize+2Fh) – (LocalitySize)	Reserved	Reserved
(LocalitySize+37h) – (LocalitySize+30h)	DPE_CRB_INTF_ID_1	Same as DPE_CRB_INTF_ID_0
(LocalitySize+3Fh) – (LocalitySize+38h)	Reserved	Reserved
(LocalitySize+43h – (LocalitySize+40h)	DPE_CRB_CTRL_REQ_1	Same as DPE_CRB_CTRL_REQ_0
(LocalitySize+47h) – (LocalitySize+44h)	DPE_CRB_CTRL_STS_1	Same as DPE CRB_CTRL_STS_0
(LocalitySize+4Bh) – (LocalitySize+48h)	DPE_CRB_CTRL_CANCEL_1	Same as DPE CRB_CTRL_CANCEL_0
(LocalitySize+4Fh) – (LocalitySize+4Ch)	DPE_CRB_CTRL_START_1	Same as DPE_CRB_CTRL_START_0
(LocalitySize+57h) – (LocalitySize+50h)	Reserved	Reserved
(LocalitySize+5Bh) – (LocalitySize+58h)	DPE_CRB_CTRL_CMD_SIZE_1	Same as DPE_CRB_CTRL_CMD_SIZE_0
(LocalitySize+5Fh) – (LocalitySize+5Ch	DPE_CRB_CTRL_CMD_LADDR_1	Same as DPE_CRB_CTRL_CMD_LADDR_0
(LocalitySize+63h) – (LocalitySize+60h)	DPE_CRB_CTRL_CMD_HADDR_1	Same as DPE_CRB_CTRL_CMD_HADDR_0
(LocalitySize+67h) – (LocalitySize+64h)	DPE_CRB_CTRL_RSP_SIZE_1	Same as DPE_CRB_CTRL_RSP_SIZE_0
(LocalitySize+6fh) – (LocalitySize+68h)	DPE_CRB_CTRL_RSP_ADDR_1	Same as DPE_CRB_RSP_ADDR_0
(LocalitySize+7Fh) – (LocalitySize+70h)	Reserved	Reserved
(LocalitySize+LocalitySize-1) – (LocalitySize+80h)	DPE_CRB_DATA_BUFFER_1	Same as DPE_CRB_DATA_BUFFER_0
Locality x		

 © TCG 2025

Offset from BASE_ADDRESS_0[1]	Register Name	Description
LOCALITY_OFFSET+2Fh : LOCALITY_OFFSET	Reserved	Reserved
LOCALITY_OFFSET+37h : LOCALITY_OFFSET+30h	DPE_CRB_INTF_ID_x	Same as DPE_CRB_INTF_ID_0
LOCALITY_OFFSET+3Fh : LOCALITY_OFFSET+38h	Reserved	Reserved
LOCALITY_OFFSET+43h : LOCALITY_OFFSET+40h	DPE_CRB_CTRL_REQ_x	Same as DPE_CRB_CTRL_REQ_0
LOCALITY_OFFSET+47h : LOCALITY_OFFSET+44h	DPE_CRB_CTRL_STS_x	Same as DPE CRB_CTRL_STS_0
LOCALITY_OFFSET+4Bh : LOCALITY_OFFSET+48h	DPE_CRB_CTRL_CANCEL_x	Same as DPE CRB_CTRL_CANCEL_0
LOCALITY_OFFSET+4Fh : LOCALITY_OFFSET+4Ch	DPE_CRB_CTRL_START_x	Same as DPE_CRB_CTRL_START_0
LOCALITY_OFFSET+53h : LOCALITY_OFFSET+50h	Reserved	Reserved
LOCALITY_OFFSET+5Bh : LOCALITY_OFFSET+58h	DPE_CRB_CTRL_CMD_SIZE_x	Same as DPE_CRB_CTRL_CMD_SIZE_0
LOCALITY_OFFSET+5Fh : LOCALITY_OFFSET+5Ch	DPE_CRB_CTRL_CMD_LADDR_x	Same as DPE_CRB_CTRL_CMD_LADDR_0
LOCALITY_OFFSET+63h : LOCALITY_OFFSET+60h	DPE_CRB_CTRL_CMD_HADDR_x	Same as DPE_CRB_CTRL_CMD_HADDR_0
LOCALITY_OFFSET+67h : LOCALITY_OFFSET+64h	DPE_CRB_CTRL_RSP_SIZE_x	Same as DPE_CRB_CTRL_RSP_SIZE_0
LOCALITY_OFFSET+6Fh : LOCALITY_OFFSET+68h	DPE_CRB_CTRL_RSP_ADDR_x	Same as DPE_CRB_RSP_ADDR_0
LOCALITY_OFFSET+7Fh : LOCALITY_OFFSET+70h	Reserved	Reserved
LOCALITY_OFFSET+LocalitySize-1 : LOCALITY_OFFSET+80h	DPE_CRB_DATA_BUFFER_x	Same as DPE_CRB_DATA_BUFFER_0
Note 1: These offsets are offsets from the BASE_ADDRESS_0 (see Section 4.2). Note 2: This table uses the following definition: LOCALITY_OFFSET = Locality * LocalitySize		

Table 4 Allocation of Register Space with disjoint CRB Data Buffer

Offset from BASE_ADDRESS_0[1]	Register Name	Description
Locality 0		
002Fh : 0000h	Undefined	Reserved
0037h : 0030h	DPE_CRB_INTF_ID_0	Used to identify the Interface types supported by the DPE as well as the Vendor ID, Device ID, and Revision ID
003Fh : 0038h	Reserved	Reserved
0043h : 0040h	DPE_CRB_CTRL_REQ_0	A register used to initiate transactions for the CRB interface.
0047h : 0044h	DPE_CRB_CTRL_STS_0	A register used by the DPE to provide the status of the CRB interface.

 © TCG 2025

Offset from BASE_ADDRESS_0[1]	Register Name	Description
004Bh : 0048h	DPE_CRB_CTRL_CANCEL_0	A register used by the client to cancel command processing.
004Fh : 004Ch	DPE_CRB_CTRL_START_0	A register used to indicate the presence of command or response data in the CRB buffer.
0057h : 0050h	Reserved	Reserved
005Bh : 0058h	DPE_CRB_CTRL_CMD_SIZE_0	The size of the CRB Data Buffer. This register SHALL be aliased across localities.
005Fh : 005Ch	DPE_CRB_CTRL_CMD_LADDR_0	The lower 32 bits of the CRB Data Buffer start address for Locality 0.
0063h : 0060h	DPE_CRB_CTRL_CMD_HADDR_0	The upper 32 bits of the CRB Data Buffer start address for Locality 0.
0067h : 0064h	DPE_CRB_CTRL_RSP_SIZE_0	The size of the CRB Data Buffer. The value of this field SHALL be identical to the value in the DPE_CRB_CTRL_CMD_SIZE_x buffer. This register SHALL be aliased across localities.
006Fh : 0068h	DPE_CRB_CTRL_RSP_ADDR_0	The address of the start of the CRB Data Buffer. This field SHALL contain the same address contained in DPE_CRB_CTRL_CMD_HADDR_x and DPE_CRB_CMD_LADDR_x.
007Fh : 0070h	Reserved	Reserved
CRB_DATA_0_START[2] : CRB_DATA_0_END[2]	DPE_CRB_DATA_BUFFER_0	The buffer allocated for Command and Response data to be transferred to/from a DPE via the DPE CRB. This buffer SHALL be a single buffer used for both command and response transfers. The CRB Data Buffer for disjoint CRB Data Buffer locations might be defined as large as 4,294,967,296 bytes. This is implementation specific. This field accepts data transfers from 1B up to the size indicated by DPE_CRB_INTF_ID_x.CapDataXferSizeSupport (see Section 5.8).
Locality 1		
00AFh : 0080h	Reserved	Reserved
00B7h : 00B0h	DPE_CRB_INTF_ID_1	Same as DPE_CRB_INTF_ID_0
00BFh : 00B8h	Reserved	Reserved
00C3h : 00C0h	DPE_CRB_CTRL_REQ_1	Same as DPE_CRB_CTRL_REQ_0
00C7h : 00C4h	DPE_CRB_CTRL_STS_1	Same as DPE CRB_CTRL_STS_0
00CBh : 00C8h	DPE_CRB_CTRL_CANCEL_1	Same as DPE CRB_CTRL_CANCEL_0
00CFh : 00CCh	DPE_CRB_CTRL_START_1	Same as DPE_CRB_CTRL_START_0
00D7h : 00D0h	Reserved	Reserved
00DBh : D8h	DPE_CRB_CTRL_CMD_SIZE_1	Same as DPE_CRB_CTRL_CMD_SIZE_0
00DFh : 00DCh	DPE_CRB_CTRL_CMD_LADDR_1	Same as DPE_CRB_CTRL_CMD_LADDR_0
00E3h : 00E0h	DPE_CRB_CTRL_CMD_HADDR_1	Same as DPE_CRB_CTRL_CMD_HADDR_0
00E7h : 00E4h	DPE_CRB_CTRL_RSP_SIZE_1	Same as DPE_CRB_CTRL_RSP_SIZE_0
00EFh : 00E8h	DPE_CRB_CTRL_RSP_ADDR_1	Same as DPE_CRB_RSP_ADDR_0
00FFh : 00F0h	Reserved	Reserved
CRB_DATA_1_START[3] : CRB_DATA_1_END[3]	DPE_CRB_DATA_BUFFER_1	Same as DPE_CRB_DATA_BUFFER_0
Locality x		
LOCALITY_OFFSET[4]+2Fh : LOCALITY_OFFSET	Reserved	Reserved

 © TCG 2025

Offset from BASE_ADDRESS_0[1]	Register Name	Description
LOCALITY_OFFSET+37h : LOCALITY_OFFSET+30h	DPE_CRB_INTF_ID_x	Same as DPE_CRB_INTF_ID_0
LOCALITY_OFFSET+3Fh : LOCALITY_OFFSET+38h	Reserved	Reserved
LOCALITY_OFFSET+43h : LOCALITY_OFFSET+40h	DPE_CRB_CTRL_REQ_x	Same as DPE_CRB_CTRL_REQ_0
LOCALITY_OFFSET+47h : LOCALITY_OFFSET+44h	DPE_CRB_CTRL_STS_x	Same as DPE CRB_CTRL_STS_0
LOCALITY_OFFSET+4Bh : LOCALITY_OFFSET+48h	DPE_CRB_CTRL_CANCEL_x	Same as DPE CRB_CTRL_CANCEL_0
LOCALITY_OFFSET+4Fh : LOCALITY_OFFSET+4Ch	DPE_CRB_CTRL_START_x	Same as DPE_CRB_CTRL_START_0
LOCALITY_OFFSET+53h : LOCALITY_OFFSET+50h	Reserved	Reserved
LOCALITY_OFFSET+5Bh : LOCALITY_OFFSET+58h	DPE_CRB_CTRL_CMD_SIZE_x	Same as DPE_CRB_CTRL_CMD_SIZE_0
LOCALITY_OFFSET+5Fh : LOCALITY_OFFSET+5Ch	DPE_CRB_CTRL_CMD_LADDR_x	Same as DPE_CRB_CTRL_CMD_LADDR_0
LOCALITY_OFFSET+63h : LOCALITY_OFFSET+60h	DPE_CRB_CTRL_CMD_HADDR_x	Same as DPE_CRB_CTRL_CMD_HADDR_0
LOCALITY_OFFSET+67h : LOCALITY_OFFSET+64h	DPE_CRB_CTRL_RSP_SIZE_x	Same as DPE_CRB_CTRL_RSP_SIZE_0
LOCALITY_OFFSET+6Fh : LOCALITY_OFFSET+68h	DPE_CRB_CTRL_RSP_ADDR_x	Same as DPE_CRB_RSP_ADDR_0
LOCALITY_OFFSET+7Fh : LOCALITY_OFFSET+70h	Reserved	Reserved
CRB_DATA_X_START[5] : CRB_DATA_X_END[5]	DPE_CRB_DATA_BUFFER_x	Same as DPE_CRB_DATA_BUFFER_0

Note 1: These offsets are offsets from the BASE_ADDRESS_0 (see Section 4.2), with the exception of the offsets for DPE_CRB_CMD_BUFFER_x, and DPE_CRB_RSP_BUFFER_x. The offsets specified for DPE_CRB_CMD_BUFFER_x and DPE_CRB_RSP_BUFFER_x are memory addresses calculated using the formulas defined below.

Note 2: Locality 0 buffer start and end addresses:
CRB_DATA_0_START = (DPE_CRB_CTRL_CMD_HADDR_0 << 32) + DPE_CRB_CTRL_CMD_LADDR_0
CRB_DATA_0_END = CRB_CMD_BUFFER_0_START + DPE_CRB_CTRL_CMD_SIZE_0

Note 3: Locality 1 buffer start and end addresses:
CRB_DATA_1_START = (DPE_CRB_CTRL_CMD_HADDR_1 << 32) + DPE_CRB_CTRL_CMD_LADDR_1
CRB_DATA_1_END = CRB_CMD_BUFFER_1_START + DPE_CRB_CTRL_CMD_SIZE_1

Note 4: This table uses the following calculations:
LOCALITY_OFFSET = Locality * LocalitySize

Note 5: Locality x buffer start and end addresses:
CRB_DATA_X_START = (DPE_CRB_CTRL_CMD_HADDR_x << 32) + DPE_CRB_CTRL_CMD_LADDR_x
CRB_DATA_X_END = CRB_CMD_BUFFER_X_START + DPE_CRB_CTRL_CMD_SIZE_x

6.2 Interface Control Area

Start of informative comment

The Interface Control Area contains status fields as well as physical address and size of the CRB Data Buffer.

This specification does not prescribe a specific access pattern to the fields of the Interface Control Area.

End of informative comment

 © TCG 2025

Table 5 – CRB Interface Control Area Layout

Field	Byte Length	Offset	Description
Request	4	00h	Used to control power state transition.
Status	4	04h	Used to indicate a status.
Cancel	4	08h	Used to abort command processing.
Start	4	0Ch	Used to indicate that a command is available for processing.
Reserved	8	10h	Reserved.
Command Size	4	18h	The size of the CRB Data Buffer. The value reported in this field is the same as the value reported in the Response Size field.
Command Address	8	1Ch	This field contains the physical address of the CRB Data Buffer. The value reported in this field is the same as the value reported in the Response Address field.
Response Size	4	24h	The size of the CRB Data Buffer. The value reported in this field is the same as value reported in the Command Size field.
Response Address	8	28h	This field contains the physical address of the CRB Data Buffer. The value reported in this field is the same as the value reported in the Command Address field.

1) A DPE compliant with this specification SHALL support client access (i.e., reads and writes) to only a portion of any field of the Interface Control Area.
2) A DPE SHALL support random access to the fields of the Interface Control Area.

6.3 DPE CRB Control Area Request Register

Start of informative comment

The Control Area Request register is used to manage DPE states as defined in Section 6.12 Interface Controls.

End of informative comment

Table 6 – DPE CRB Control Area Request

Abbreviation:				DPE_CRB_CTRL_REQ_x
General Description:				Control Area Request
Bits	**Read / Write**	**Name**	**Initial Value**	**Description**
31:2	Read / Write	Reserved	0	Reserved Reads return 0. Writes are ignored.
1	Read / Write	goIdle	0	Used by client to indicate transition of the DPE to and from the Idle state 1: Set by client to indicate response has been read from the CRB Data Buffer and DPE can transition to the Idle state 0: Cleared to 0 by DPE to acknowledge completion of the state change request to the Idle state with the resulting state reflected in the DPE_CRB_STS_x.dpeIdle field. DPE SHALL complete this transition within TIMEOUT_C. Writes of 0 are ignored.

 © TCG 2025

0	Read / Write	cmdReady	0	Used by client to request the DPE to transition to the Ready state. 1: Set to 1 by client to indicate the DPE needs to be ready to receive a command. 0: Cleared to 0 by DPE to acknowledge completion of the state change request to the Idle state with the resulting state reflected in the DPE_CRB_STS_x.dpeIdle field. DPE SHALL complete this transition within TIMEOUT_C

6.3.1 Field: cmdReady

Start of informative comment

The cmdReady field is written by a client to transition the DPE to the Ready state from the Idle state.

When in the Idle State, the DPE may perform background tasks, such as random number or key generation, or it may turn off some of its internal functions to conserve power. Because of the non-deterministic nature of what the DPE may be doing in the background, it is necessary to define a time within which the DPE must respond to a request to transition to the Ready state. This time, TIMEOUT_C (see Section 4.5 Interface Timeouts) is the maximum amount of time a client will need to wait before determining the DPE or its interface have encountered an error. DPE manufacturers should implement their DPEs to transition to the Ready state as quickly as possible.

There may be cases where a DPE client has a queue of commands to issue to the DPE. Some DPE implementations are limited in their ability to quickly transition in and out of the Idle state. These DPEs may support the ability to bypass the Idle and Ready states and transition directly to the Ready state or Command Reception state from the Command Completion state.

The ability to support this feature is reported in the CapCRBIdleBypass field (see Section 5.9). If this field is SET to 1, a DPE client may exercise this new behavior by writing this field from the Command Completion state and the DPE will transition directly to the Ready state. As there is no interface mechanism to distinguish between the Command Completion and Ready states, a DPE client may need to write this field twice to ensure the DPE is in the Ready state. DPE implementations that do not support this behavior are anticipated to perform the transition from Command Completion to Idle and from Idle to Ready in a shorter time frame than the sum of the timeouts for both state transitions. As DPE implementations use the Idle state to perform background processing that improves overall performance and reduces power consumption, DPE clients should not exercise this behavior unless there are multiple DPE commands in their queue.

The cmdReady field is valid and readable within the timing defined in Section 6.13 Reset Timing.

End of informative comment

1) If a DPE is in the Idle state:
    a. On a write of 1 to cmdReady, the DPE SHALL transition from the Idle state to the Ready state.
    b. The DPE SHALL acknowledge the write by clearing cmdReady to 0 within TIMEOUT_C, see Section 4.5 Interface Timeouts.
2) If a DPE is in the Command Completion state:
    a. On a write of 1:
        i. If DPE_CRB_INTF_ID_x.CapCRBIdleBypass is 1, the DPE SHALL transition to the Ready state.
        ii. If DPE_CRB_INTF_ID_x.CapCRBIdleBypass is 0, the DPE SHALL ignore the write.
    b. Writes of 0 SHALL be ignored.

6.3.2 Field: goIdle

Start of informative comment

When in the Command Completion state, a DPE client can use this field to communicate to the DPE that it has received all response data and the DPE may invalidate its buffer and transition to Idle.

 © TCG 2025

There are occasions when a DPE client may want to abort sending a command or reading a response, for example, a power transition. In these cases, it is desirable to have a mechanism to cause the DPE to transition back to the Idle state. This operation is not the same as a command abort (see Section 6.5) but provides approximation of that functionality.

When in the Command Execution phase, a transaction cannot be aborted using goIdle. The only option to interrupt the processing of the current command is to perform a Command Cancel (see Section 6.5).

The goIdle field is valid and readable within the timing defined in Section 6.13 Reset Timing.

End of informative comment

1) On a write of 1 to goIdle, the DPE SHALL acknowledge the write by clearing the field to 0 within TIMEOUT_C.
2) On a write of 1:
    a. The DPE SHALL invalidate the CRB Data Buffer of the target locality.
    b. If a DPE supports entering a lower power state, the DPE SHOULD power down or enter a low-power state. If the DPE powers down or enters a low-power state, it SHALL SET DPE_CRB_CTRL_STS_x.dpeIdle.
    c. To avoid possible denial of service across localities, the DPE SHALL NOT power down or enter a low-power state until goIdle has been SET on all supported localities.
3) Once a response is placed in the Command Response Buffer, the DPE SHALL maintain the response until client SETs goIdle to 1.
4) When a DPE is in the Ready, Command Reception, or Command Completion states, if DPE_CRB_CTRL_REQ_x.goIdle is SET to 1:
    a. The DPE SHALL invalidate the Command Response Buffer,
    b. The DPE SHALL SET the DPE_CRB_CTRL_STS_x.dpeIdle to 1, and then
    c. The DPE SHALL CLEAR the DPE_CRB_CTRL_REQ_x.goIdle field to 0.
5) When a DPE is in the Command Execution state, the DPE SHALL ignore writes to the DPE_CRB_CTRL_REQ_x field.
6) Writes of 0 SHALL be ignored.

6.4 Status Register (DPE_CRB_CTRL_STS_x)

Start of informative comment

This register is used to indicate the current state and status of the DPE. This field is aliased across all supported CRB localities.

End of informative comment

Table 7 – Bit layout of CRB Status (DPE_CRB_CTRL_STS_x) field

Abbreviation:				DPE_CRB_CTRL_STS_x
General Description:				Interface Control Area Status
Bits	**Read / Write**	**Name**	**Initial Value**	**Description**
31:2	Read Only	Reserved	0	Reserved. Reads return 0.
1	Read Only	dpeIdle	1	Used by DPE to indicate it is in the Idle state. 1: Set by DPE when in the Idle state. 0: Cleared by the DPE on receipt of DPE_CRB_CTRL_REQ_x.cmdReady when DPE transitions to the Ready state. SHALL be CLEARed by TIMEOUT_C.

 © TCG 2025

0	Read Only	dpeSts	0	Used by DPE to indicate current status. 1: Set by DPE to indicate a FATAL Error. 0: Indicates DPE is operational.

6.4.1 Field: dpeIdle

Start of informative comment

The dpeIdle field is used by the DPE to indicate its current state. The DPE SETs this field to indicate that it is in the Idle state and CLEARs it to indicate it is in the Ready state.

The default state of this field indicates the DPE is in the Idle state.

The dpeIdle field is valid and readable within the timing defined in Section 6.13 Reset Timing.

End of informative comment

1) A DPE SHALL SET dpeIdle to 1 if going to the Idle state.
2) A DPE SHALL NOT SET dpeIdle to 1 if in any state other than Idle.
3) On a write to DPE_CRB_CTRL_REQ_x.cmdReady of 1, a DPE SHALL CLEAR dpeIdle and go to the Ready state within TIMEOUT_C.

6.4.2 Field: dpeSts

Start of informative comment

The dpeSts field is used by the DPE to indicate its current status. This field should only be used for fatal errors for which a DPE response code cannot be provided. A client may stop using a DPE when this field is SET. The initial state of this field should reflect the operational state of the DPE.

The dpeSts field is valid and readable within the timing defined in Section 6.13 Reset Timing.

End of informative comment

1) A DPE MAY SET dpeSts to 1 on a fatal error.
2) A DPE SHALL NOT SET dpeSts to 1 for any other reason other than a fatal error.

6.5 Cancel Register (DPE_CRB_CTRL_CANCEL_x)

Start of informative comment

Cancel may be used by a DPE client to request the DPE to terminate processing the current command. A client might request this, for example, because the system is going to a lower power state. The DPE will either complete the currently processing command and return the result or will cancel the command and return the Cancelled Command error. In general, for long running commands, the DPE may have checkpoints in its code to check the state of the Cancel field. If, at one of these checkpoints, the DPE sees a Command Cancel request, the DPE has the option of cancelling the command or completing the command. DPE's are not required to perform this check.

Cancel is an asynchronous input. DPE client writers should take care in use of this function. DPE implementations may have different behavior. Some DPE implementations CLEAR this field; some do not. If this field is not CLEARed by a DPE client, some DPE implementations continue to process Cancel requests until this field is CLEARed. If a DPE client unconditionally CLEARs this register, it will work for any DPE behavior.

Clients should take care to send Command Cancel requests only when the DPE is in Command Execution state and to only CLEAR the field when the DPE has exited the Command Execution state to avoid race conditions.

End of informative comment

 © TCG 2025

Table 8 – Bit layout of CRB Cancel (DPE_CRB_CTRL_CANCEL_x) field

Abbreviation:				DPE_CRB_CTRL_CANCEL_x
General Description:				Interface Control Area Cancel
Bits	**Read / Write**	**Name**	**Initial Value**	**Description**
31:0	Read / Write	Cancel	0	Reads return correct value. Writes (1): Cancel a command Writes (0): Clears field when command has been cancelled

1) If DPE_CRB_INTF_ID_x.CapCancel is SET to 1:
    a. When a DPE is in the Command Execution state and DPE_CRB_CTRL_CANCEL_x is SET to 1:
        i. The DPE MAY terminate command processing,
        ii. The DPE SHALL return a response and transition to the Command Completion state:
            1. If the command is successfully completed, the DPE SHALL return the command response.
            2. If the command is terminated, the DPE SHALL return Cancelled Command.
            3. For commands that do not cause RSA key generation, the DPE SHALL complete the command or cancel it within TIMEOUT_A, see Section 4.5 Interface Timeouts
            4. For commands that cause RSA key generation, the DPE SHALL complete the command or cancel it within TIMEOUT_B, see Section 4.5 Interface Timeouts.
            5. The DPE SHALL CLEAR the DPE_CRB_CTRL_START_x field to 0.
2) If DPE_CRB_INTF_ID_x.CapCancel is CLEAR (0), a DPE SHALL ignore writes to DPE_CRB_CTRL_CANCEL_x.
3) When a DPE is in any state other than Command Execution, the DPE SHOULD ignore writes to DPE_CRB_CTRL_CANCEL_x.

6.6 Start Register (DPE_CRB_CTRL_START_x)

Start of informative comment

Start is used by a client to signal the DPE to begin processing the contents of the CRB Data Buffer and transition the DPE into the Command Execution state. When the DPE has completed or cancelled a command, the DPE CLEARs this field and transitions to the Command Completion state.

DPE clients should poll on this field to detect the DPE transitioning from the Command Execution to the Command Completion states.

End of informative comment

Table 9 – Bit layout of CRB Start (DPE_CRB_CTRL_START_x) field

Abbreviation:				TPM_CRB_CTRL_START_x
General Description:				Interface Control Area Start
Bit	**Read / Write**	**Name**	**Initial Value**	**Description**
31:0	Read / Write	Start	0	When SET by client, indicates a command is ready for processing. When DPE CLEARs this field, the DPE has transitioned to Command Completion.

1) On a write of 1 to the Start field:
    a. If a DPE is in the Command Reception state, the DPE SHALL transition to the Command Execution state and begin processing the command in the buffer.
    b. If a DPE is in any other state, the DPE SHALL ignore the write.
2) A DPE SHALL ignore writes of 0 to this field.
3) When a DPE completes command processing, the DPE SHALL CLEAR this field to 0 and transition to the Command Completion state.

 © TCG 2025

6.7 Command Size (DPE_CRB_CTRL_CMD_SIZE_x)

Start of informative comment

The Command Size field is the size of the CRB Data Buffer in bytes.

The value of Command Size field should be chosen by a DPE designer such that the largest command or response that can be issued for the specific DPE implementation fits into the CRB Data Buffer.

Refer to Section 6.8 for more information.

End of informative comment

6.8 Command Address (DPE_CRB_CTRL_CMD_LADDR_x and DPE_CRB_CTRL_CMD_HADDR_x)

Start of informative comment

A client reads this field to determine the physical address of the CRB Data Buffer. A client will write a DPE command to this CRB Data Buffer. The DPE reads the DPE command from this CRB Data Buffer.

Writes of commands larger than a DPE's supported Command Size are effectively either writes to a different DPE CRB locality or writes to system address space outside the scope of the DPE CRB. Writes to a different DPE CRB locality are subject to the requirements of the location to which they are written. Behaviors for writes to system address space outside the scope of the DPE CRB are outside the scope of this specification.

Writes to the CRB Data Buffer address range when *Start* is SET have the potential to corrupt DPE command or response data while the DPE is processing this data, resulting in unexpected behavior. System hardware might prevent writing to the CRB Data Buffer address range if *Start* is SET.

When the Command Address and Response Address fields point to BASE_ADDRESS_x + 0x0080 (i.e., the CRB Data Buffer is co-located with the Interface Control Area), then the entire CRB Data Buffer falls within the memory range allocated for locality x and the size of the CRB Data Buffer is (LocalitySize – 0x0080) bytes. Alternatively, the DPE can allocate physical memory for the CRB Data Buffer outside of the CRB locality memory range, with a size potentially greater than (LocalitySize – 0x0080) bytes.

End of informative comment

1) The Command Address field SHALL contain the physical address of the CRB Data Buffer.
2) For locality x, the Command Address and Response Address fields SHOULD point to BASE_ADDRESS_x + 0x0080 with Command Size and Response Size of (LocalitySize - 0x0080) bytes.
3) The physical address of the CRB Data Buffer SHALL be unique per locality.
4) The physical address returned in the Command Address fields (DPE_CRB_CTRL_CMD_LADDR_x and DPE_CRB_CTRL_CMD_HADDR_x) and Response Address field for each locality SHALL be the same.

6.9 Response Size (DPE_CRB_CTRL_RSP_SIZE_x)

Start of informative comment

The Response Size field identifies the size of the response buffer in bytes.

Refer to Section 6.7 and Section 6.10 for more information.

End of informative comment

1) Because the DPE CRB uses a single buffer for transmitting both commands and responses to/from a DPE, the value reported in the Response Size field SHALL be the same as the value reported in the Command Size field.

 © TCG 2025

6.10 Response Address (DPE_CRB_CTRL_RSP_ADDR_x)

Start of informative comment

A client reads this field to determine the physical address of the CRB Data Buffer. The DPE will write the response to the last DPE command to the CRB Data Buffer. A client reads the DPE response from this CRB Data Buffer.

Reads of response data larger than “Response Size” results in reading more data than the response returned by the DPE. The behavior of the read operation and data returned in this case is outside the scope of this specification.

The CRB Data Buffer might contain corrupt DPE response data while *Start* is SET because a DPE has not completed processing a command and generating a response to the command until the DPE changes *Start* to CLEAR.

End of informative comment

1) The Response Address field SHALL contain the physical address of the CRB Data Buffer.

6.11 DPE CRB Data Buffer

6.11.1 Access Restrictions

Start of informative comment

The CRB Data Buffer is implemented as a single buffer used to transfer both commands and responses.

The CRB Data Buffer can only be written to initially at the base address. Subsequent writes are required to go to sequential addresses. A client can read any location of the CRB Data Buffer at any time but there is only guaranteed to be a valid response from the DPE after the DPE has CLEARed DPE_CRB_CTRL_START_x and until DPE_CRB_CTRL_REQ_x.goIdle or DPE_CRB_CTRL_REQ_x.cmdReady has been SET.

End of informative comment

6.11.2 DPE_CRB_DATA_BUFFER_x Access Restrictions

1) On a write to the buffer:
    a. If the initial transaction is to an address other than the base address for the buffer, a DPE MAY ignore the write.
    b. If subsequent transactions are to non-sequential addresses, a DPE MAY ignore the write.
    c. If the transaction is a size other than 1 byte or a power of 2 bytes, a DPE MAY ignore the write.
    d. A DPE MAY invalidate any existing data within the buffer upon a write to the base address of the buffer.
    e. A DPE SHALL replace invalidated data with data written upon a write to the base address of the buffer.
    f. Following the write to the buffer and prior to a write to DPE_CRB_CTRL_START_x, a DPE MAY ignore read requests.
2) On a read from the buffer:
    a. Upon completion of a command, the DPE SHALL maintain the response in the buffer until receipt of a write of 1 to DPE_CRB_CTRL_REQ_x.goIdle or DPE_CRB_CTRL_REQ_x.cmdReady.

6.12 Interface Controls

A DPE SHALL be in one of the following defined states for each CRB locality it supports:

1) *Command Reception* occurs following a *Ready* state between the write of the first byte of a command to the command buffer and the receipt of a write of 1 to DPE_CRB_CTRL_START_x.
2) *Command Execution* occurs after receipt of a 1 to DPE_CRB_CTRL_START_x and the DPE clearing DPE_CRB_CTRL_START_x to 0.
3) *Command Completion* occurs after completion of a command (indicated by a DPE clearing DPE_CRB_CTRL_START_x to 0) and before a write of a 1 by a client to DPE_CRB_CTRL_REQ_x.goIdle.

 © TCG 2025

4) *Idle* is any time DPE_CRB_CTRL_STS_x.dpeIdle is 1. This will occur when the DPE clients across all DPE localities have SET TPM_CRB_CTRL_REQ_x.goIdle to 1. *Idle* is signaled by a DPE setting DPE_CRB_CTRL_STS_x.dpeIdle to 1. *Idle* is the initial state of the DPE upon initialization of the DPE. A DPE MAY skip this state on receipt of DPE_CRB_CTRL_REQ_x.cmdReady while in *Command Completion*.
5) *Ready* is any time a DPE is ready to receive a command. A DPE transitions from the *Idle* state to the *Ready* state by clearing DPE_CRB_CTRL_STS_x to 0 after receiving a write of 1 by a DPE client to DPE_CRB_CTRL_STS_x.cmdReady, as indicated by the Status field being CLEARed to 0. If DPE_CRB_INTF_ID.CapCRBIdleBypass is SET to 1, a DPE transitions from *Command Completion* state to the *Ready* state after clearing DPE_CRB_CTRL_REQ_x.cmdReady to 0 after receiving a write of 1 to DPE_CRB_CTRL_REQ_x.cmdReady.

Start of Informative Comment

The diagram in Figure 1 is derived from the above normative statements. It is informative and only for illustrating diagrammatically the above DPE states and their transitions. The numbers in parentheses reference the states represented by row numbers in Table 10.

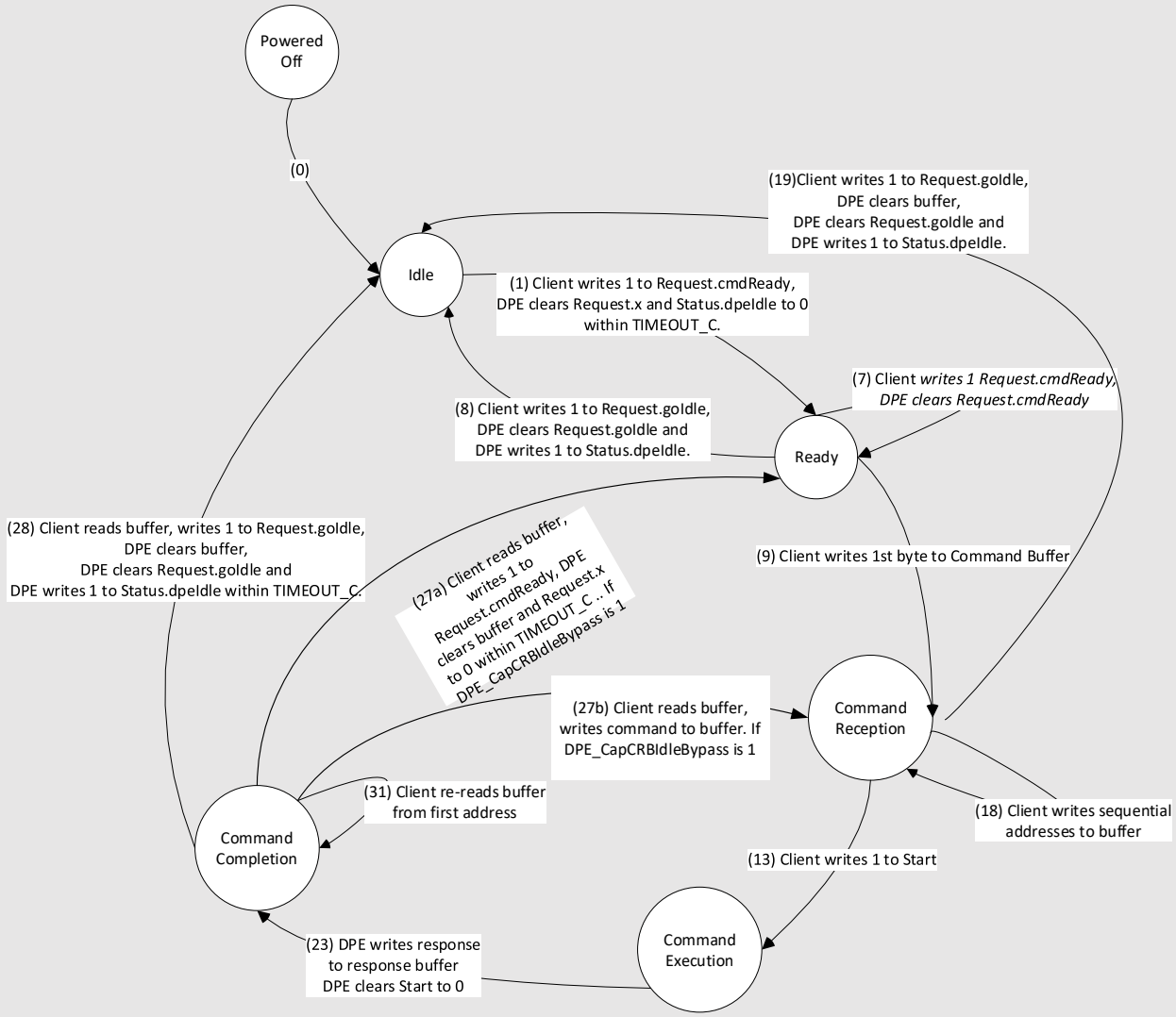


Figure 1 DPE State Diagram for CRB Interface

 © TCG 2025

End of Informative Comment

6.12.1 CRB Interface State Transitions

Start of Informative Comment

Table 10 shows the changes in CRB Interface fields based on the command or action done to a DPE. Notice this is not a state transition table covering states defined in Section 6.12, rather a table describing how the status fields change based on initial condition and action taken. The following rules apply to Table 10.

There could be intermediate states when a command has finished but DPE_CRB_CTRL_START_x field has not yet CLEARed. Clients are expected to poll until the appropriate status field is SET.

Table 10 applies to the CRB Interface field states for each locality independently. A DPE might reflect a state of *Idle* in one locality while also reflecting a state of *Ready* in another locality. A DPE is responsible for managing background tasks and low-power states appropriately to avoid negatively impacting one locality when another locality reflects an *Idle* state.

The statements in the column labeled “Action Taken” are informative when shaded and are derived from normative statements contained within the definitions of the CRB Interface in Section 6 CRB Interface Requirements. If there is an inconsistency between the Action Taken column and those normative statements, the normative statements take precedence.

Normal transitions are highlighted in yellow and are indexed to the state transitions illustrated in Figure 1.

The following abbreviations are used in Table 10:

Label	Bit Definition
C/R	DPE_CRB_CTRL_REQ.cmdReady
gI	DPE_CRB_CTRL_REQ.goIdle
I	DPE_CRB_CTRL_STS.dpeIdle
St	DPE_CRB_CTRL_STS.dpeSts
S	DPE_CRB_CTRL_START_x
CC	DPE_CRB_CTRL_CANCEL_x
—	No DPE access for the corresponding data element
X	Either 0 or 1. A DPE is allowed to maintain this field as either value for this state. This specification recommends clients not be designed to depend on a particular value for this state.

End of Informative Comment

 © TCG 2025

Table 10 - CRB Interface State Transitions

#	Present State of Control Area							Fields/Data Written					Next State & Result							Action Taken
	State	Request		Status		S	CC	Request			S	CC	State (Row)	Request		Status		S	CC	Description
		C/R	gI	I	St			C/R	gI	CRB Data Buffer				C/R	gI	I	St			
0	Powered Off	-	-	-	-	-	-	X	X	-	X	X	Idle (1)	0	0	1	0	0	0	DPE Transitions from Powered Off to Idle within TIMEOUT_B
1	Idle	0	0	1	0	0	0	1	0	-	X	X	Ready	0	0	0	0	0	0	DPE Transitions to Ready within TIMEOUT_C
2	Idle	0	0	1	0	0	0	0	1	-	X	X	Idle	0	0	1	0	0	0	DPE remains in Idle. DPE MAY ignore request.
3	Idle	0	0	1	0	0	0	X	X	-	1	0	Idle	0	0	1	0	0	0	DPE remains in Idle. DPE ignores request
4	Idle	0	0	1	0	0	0	X	X	-	0	1	Idle	0	0	1	0	0	0	No command to Cancel. DPE remains in Idle. DPE SHOULD ignore Cancel field.
5	Idle	0	0	1	0	0	0	X	X	Write access to buffer	0	0	Idle	0	0	1	0	0	0	DPE SHALL ignore any access to the CRB Data Buffer
6	Idle	0	0	1	0	0	0	X	X	Read access to buffer	0	0	Idle	0	0	1	0	0	0	DPE SHALL ignore any access to the CRB Data Buffer
7	Ready	0	0	0	0	0	0	1	0	-	0	0	Ready	0	0	0	0	0	0	DPE remains in Ready. DPE SHALL ignore request
8	Ready	0	0	0	0	0	0	0	1	-	0	0	Idle	0	0	1	0	0	0	Interface Reset. DPE goes to Idle state.
9	Ready	0	0	0	0	0	0	X	X	Write access to the buffer	0	0	Reception	0	0	0	0	0	0	DPE receives command
10	Ready	0	0	0	0	0	0	0	0	-	1	0	Ready	0	0	0	0	0	0	No data in the buffer, DPE remains in Ready. DPE SHALL ignore request
11	Ready	0	0	0	0	0	0	0	0	-	0	1	Ready	0	0	0	0	0	0	No command to cancel. DPE remains in Ready. DPE SHALL Ignore request.
12	Ready	0	0	0	0	0	0	X	X	Read access to buffer	0	0	Ready	0	0	0	0	0	0	No response to read. DPE remains in Ready. DPE SHOULD ignore request.
13	Reception	0	0	0	0	0	0	0	0	-	1	0	Execution	0	0	0	0	1	0	DPE transitions to Execution state. DPE begins processing command.
14	Reception	0	0	0	0	0	0	0	0	-	0	1	Reception	0	0	0	0	0	0	DPE remains in Reception. DPE SHALL ignore request.
15	Reception	0	0	0	0	0	0	1	0	-	0	0	Ready	0	0	0	0	0	0	DPE transitions to Ready.
16	Reception	0	0	0	0	0	0	0	0	-	0	0	Reception	0	0	0	0	0	0	DPE remains in Reception.
17	Reception	0	0	0	0	0	0	0	0	Read access to buffer	0	0	Reception	0	0	0	0	0	0	DPE remains in Reception. DPE SHALL ignore request.
18	Reception	0	0	0	0	0	0	0	0	Write access to the buffer	0	0	Reception	0	0	0	0	0	0	DPE in Reception. DPE continues to receive data.
19	Reception	0	0	0	0	0	0	0	1	-	0	0	Idle	0	0	1	0	0	0	Abort Command Reception and return to Idle.
20	Execution	0	0	0	0	1	0	0	0	-	1	0	Execution	0	0	0	0	1	0	DPE remains in Execution state. DPE MAY ignore writes to Start field.
21	Execution	0	0	0	0	1	0	1	0	-	X	0	Execution	0	0	0	0	1	0	DPE remains in Execution state. DPE SHALL ignore request.
22	Execution	0	0	0	0	1	0	0	1	-	X	0	Execution	0	0	0	0	1	0	DPE remains in Execution state. DPE SHALL ignore request.

 © TCG 2025

23	Execution	0	0	0	0	1	0	0	0	-	X	0	Completion	0	0	0	0	0	0	DPE CLEARs Start field and transitions to Completion.
24	Execution	0	0	0	0	1	1	0	0		X	1	Completion	0	0	0	0	0	0	A DPE SHALL transition to Completion. DPE MAY Cancel command. DPE SHALL respond with Return code within TIMEOUT_B.
25	Execution	0	0	0	0	1	0	0	0	Write access to buffer	X	0	Execution	0	0	0	0	1	0	DPE remains in Execution state. DPE MAY ignore write requests
26	Execution	0	0	0	0	1	0	0	0	Read access to buffer	X	0	Execution	0	0	0	0	1	0	DPE remains in Execution state. DPE MAY ignore read requests
27	Completion	0	0	0	0	0	0	1	0	-	0	0	Completion	0	0	0	0	0	0	DPE remains in Completion. DPE SHALL ignore request.
27a	Completion	0	0	0	0	0	0	1	0	-	0	0	Ready	0	0	0	0	0	0	DPE SHALL transition to Ready if DPE_CAPCRB_INTF.CapCRBIdleBypass is SET to 1.
27b	Completion	0	0	0	0	0	0	0	0	Write access to buffer	0	0	Reception	0	0	0	0	0	0	DPE receives command if DPE_CAP_CRB_INTF.CapCRBIdleBypass is SET to 1.
28	Completion	0	0	0	0	0	0	0	1	-	0	0	Idle	0	0	1	0	0	0	DPE transitions to Idle, invalidates buffer.
29	Completion	0	0	0	0	0	0	0	0	-	1	0	Completion	0	0	0	0	0	0	DPE remains in Completion. DPE SHALL ignore request
30	Completion	0	0	0	0	0	0	0	0	-	0	1	Completion	0	0	0	0	0	0	DPE remains in Completion. DPE SHALL ignore request
31	Completion	0	0	0	0	0	0	0	0	Read access to buffer, sequential accesses	0	0	Completion	0	0	0	0	0	0	DPE returns data, remains in Completion and maintains data in buffer.
32	Completion	0	0	0	0	0	0	0	0	Read access to buffer, non-sequential accesses	0	0	Completion	0	0	0	0	0	0	Invalid access. DPE MAY ignore the transaction.
33	Completion	0	0	0	0	0	0	0	0	Read access, re-read from Base address	0	0	Completion	0	0	0	0	0	0	Retry. DPE resets internal byte counter and returns response from Base address.

 © TCG 2025

6.13 Reset Timing

Start of informative comment

The operation of the platform's CRTM likely occurs during a very time-sensitive period. Because of this, strict requirements are necessary for a DPE's reset timing. During this time, the platform's CRTM may need to make decisions based on the presence or absence of a DPE's response that affect the rest of the platform's boot cycle.

A DPE may need to perform maintenance or recovery of NV in cases such as an unexpected power loss during a field upgrade, following an unexpected loss of power, or in the case of recovering from NV corruption. It may take longer for a DPE to be ready to receive a command.

This section contains the timing requirements for a DPE's registers. The normative requirements describing the relationship between the individual fields within a register are contained in Section 5 CRB Interface , and Section 6 CRB Interface Requirements.

End of informative comment

1) All fields within a DPE_CRB_INTF_ID_x SHALL be valid and a DPE SHALL be ready to receive a command:
    a. For DPEs that do not comply with FIPS 140-3 Self-Test requirements, the DPE SHALL be ready within TIMEOUT_D, from DPE power-on.
    b. For DPEs that do comply with FIPS 140-3 Self-Test requirements, the DPE SHALL return the register contents within 200ms from DPE power-on.

 © TCG 2025

Appendix A: DPE Interfaces and Localities

Start of informative comment

A DPE might support multiple interfaces and serve various clients over different interfaces. This specification defines the CRB interface and localities accessed using the CRB interface. The CRB interface does not expose information of other interfaces or localities for other interfaces.

Figure 2 illustrates an example of a DPE that supports three interfaces. In this example, the DPE is implemented as the DICE Root of Trust (see [2]), a sub-component of an SoC's Root of Trust (RoT), that provides DPE services to software running on the host CPU and other elements/components in the SoC. The three interfaces and the entities served by the interfaces are:

The CRB for software clients executing on the SoC's primary CPU. This interface is defined in this specification.

A Mailbox for the other SoC elements.

A private bus with a proprietary interface for other components of the RoT, such as a Root of Trust for Measurement (RTM), Root of Trust for Reporting (RTR), and other RoT functionalities.

The DPE interfaces are intended to function independently of each other. The mechanisms for the DPE to manage access to different localities by different clients might be different over different interfaces. For example, access to different localities through the CRB is separated by system memory offsets, as described in this specification, access to different localities through the mailbox could be separated by different sets of mailbox send/receive registers.

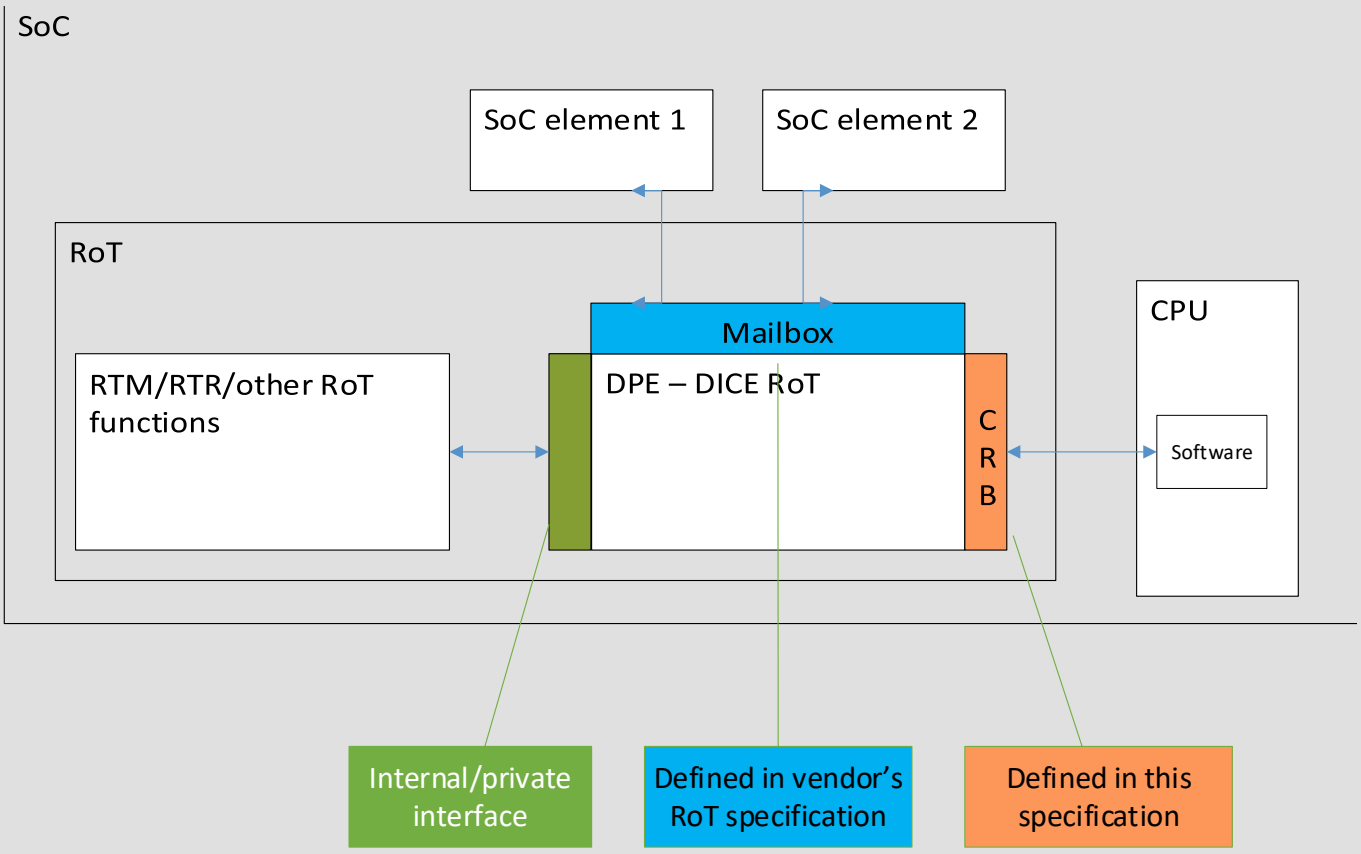


Figure 2 – Example of Multiple Interfaces of a DPE

End of informative comment

 © TCG 2025