AMBA CHI Issue E.a Errata

Architecture & Technology Group

Document number:	ARM AES 0037
Release Number:	2.0
Confidentiality:	Non-Confidential
Date of Issue:	22-Mar-2021

Abstract

This document includes clarifications and corrections to CHI Issue E.a specification.

Contents

About this document

Release Information

The change history table lists the changes that have been made to this document.

Date	Version	Confidentiality	Change
10 December 2021	1.0	Confidential	First limited release
22 March 2021	2.0	Non-Confidential	First public release

References

This document refers to the following documents.

Ref	Document Number	Title
1	ARM IHI 0050E.a	**AMBA® 5 CHI Architecture Specification**

AMBA SPECIFICATION LICENCE

THIS END USER LICENCE AGREEMENT ("LICENCE") IS A LEGAL AGREEMENT BETWEEN YOU (EITHER A SINGLE INDIVIDUAL, OR SINGLE LEGAL ENTITY) AND ARM LIMITED ("ARM") FOR THE USE OF ARM'S INTELLECTUAL PROPERTY (INCLUDING, WITHOUT LIMITATION, ANY COPYRIGHT) IN THE RELEVANT AMBA SPECIFICATION ACCOMPANYING THIS LICENCE. ARM LICENSES THE RELEVANT AMBA SPECIFICATION TO YOU ON CONDITION THAT YOU ACCEPT ALL OF THE TERMS IN THIS LICENCE. BY CLICKING "I AGREE" OR OTHERWISE USING OR COPYING THE RELEVANT AMBA SPECIFICATION YOU INDICATE THAT YOU AGREE TO BE BOUND BY ALL THE TERMS OF THIS LICENCE.

"LICENSEE" means You and your Subsidiaries.

"Subsidiary" means, if You are a single entity, any company the majority of whose voting shares is now or hereafter owned or controlled, directly or indirectly, by You. A company shall be a Subsidiary only for the period during which such control exists.

1. Subject to the provisions of Clauses 2, 3 and 4, Arm hereby grants to LICENSEE a perpetual, non-exclusive, non-transferable, royalty free, worldwide licence to:

(i) use and copy the relevant AMBA Specification for the purpose of developing and having developed products that comply with the relevant AMBA Specification;

(ii) manufacture and have manufactured products which either: (a) have been created by or for LICENSEE under the licence granted in Clause 1(i); or (b) incorporate a product(s) which has been created by a third party(s) under a licence granted by Arm in Clause 1(i) of such third party's AMBA Specification Licence; and

(iii) offer to sell, sell, supply or otherwise distribute products which have either been (a) created by or for LICENSEE under the licence granted in Clause 1(i); or (b) manufactured by or for LICENSEE under the licence granted in Clause 1(ii).

2. LICENSEE hereby agrees that the licence granted in Clause 1 is subject to the following restrictions:

(i) where a product created under Clause 1(i) is an integrated circuit which includes a CPU then either: (a) such CPU shall only be manufactured under licence from Arm; or (b) such CPU is neither substantially compliant with nor marketed as being compliant with the Arm instruction sets licensed by Arm from time to time;

(ii) the licences granted in Clause 1(iii) shall not extend to any portion or function of a product that is not itself compliant with part of the relevant AMBA Specification; and

(iii) no right is granted to LICENSEE to sublicense the rights granted to LICENSEE under this Agreement.

3. Except as specifically licensed in accordance with Clause 1, LICENSEE acquires no right, title or interest in any Arm technology or any intellectual property embodied therein. In no event shall the licences granted in accordance with Clause 1 be construed as granting LICENSEE, expressly or by implication, estoppel or otherwise, a licence to use any Arm technology except the relevant AMBA Specification.

4. THE RELEVANT AMBA SPECIFICATION IS PROVIDED "AS IS" WITH NO REPRESENTATION OR WARRANTIES EXPRESS, IMPLIED OR STATUTORY, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF SATISFACTORY QUALITY, MERCHANTABILITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE, OR THAT ANY USE OR IMPLEMENTATION OF SUCH ARM TECHNOLOGY WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADE SECRETS OR OTHER INTELLECTUAL PROPERTY RIGHTS.

5. NOTWITHSTANDING ANYTHING TO THE CONTRARY CONTAINED IN THIS AGREEMENT, TO THE FULLEST EXTENT PETMITTED BY LAW, THE MAXIMUM LIABILITY OF ARM IN AGGREGATE FOR ALL CLAIMS MADE AGAINST ARM, IN CONTRACT, TORT OR OTHERWISE, IN CONNECTION WITH THE SUBJECT MATTER OF THIS AGREEMENT (INCLUDING WITHOUT LIMITATION (I) LICENSEE'S USE OF THE ARM TECHNOLOGY; AND (II) THE IMPLEMENTATION OF THE ARM TECHNOLOGY IN ANY PRODUCT CREATED BY LICENSEE UNDER THIS AGREEMENT) SHALL NOT EXCEED THE FEES PAID (IF ANY) BY LICENSEE TO ARM UNDER THIS AGREEMENT. THE EXISTENCE OF MORE THAN ONE CLAIM OR SUIT WILL NOT ENLARGE OR EXTEND THE LIMIT. LICENSEE RELEASES ARM FROM ALL OBLIGATIONS, LIABILITY, CLAIMS OR DEMANDS IN EXCESS OF THIS LIMITATION.

6. No licence, express, implied or otherwise, is granted to LICENSEE, under the provisions of Clause 1, to use the Arm tradename, or AMBA trademark in connection with the relevant AMBA Specification or any products based thereon. Nothing in Clause 1 shall be construed as authority for LICENSEE to make any representations on behalf of Arm in respect of the relevant AMBA Specification.

7. This Licence shall remain in force until terminated by you or by Arm. Without prejudice to any of its other rights if LICENSEE is in breach of any of the terms and conditions of this Licence then Arm may terminate this Licence immediately upon giving written notice to You. You may terminate this Licence at any time. Upon expiry or termination of this Licence by You or by Arm LICENSEE shall stop using the relevant AMBA Specification and destroy all copies of the relevant AMBA Specification in your possession together with all documentation and related materials. Upon expiry or termination of this Licence, the provisions of clauses 6 and 7 shall survive.

8. The validity, construction and performance of this Agreement shall be governed by English Law.

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Product Status

The information in this document is final, that is for a developed product.

Web Address

http://www.arm.com

1 Introduction

This document lists errata on AMBA CHI Issue E.a.

Each errata description is organized as a brief reason for the change, along with the precise change.

1.1 Classification of the change.

Each listed item has a classification ID, of the form XYYY, where:

X is the errata classification type as follows, C, R, E or D:

C: Clarification

Informative change only

R: Relaxation

Backward-compatible normative change, modifying existing functionality

E: Enhancement

Backward-compatible normative change, adding new functionality.

D: Defect

Non-backward compatible normative change

YYY is an Arm internal tracking number.

2 Errata list

2.1 D577: SnpAttr in DVMOp for TLBI is applicable

The SnpAttr bit in a request, except in DVMOp and PrefetchTgt, is used to determine if the request is Snoopable or not. The concept of Snoopability is not applicable in DVMOp transactions. The same bit in the request with the same name is used to differentiate inner from outer domain for DVMOp requests.

The DVMOp entry in the Table 2-14 “Snoop attributes for the different transaction types” on page 2-124 is correctly marked as ‘n/a’. But the table footnote for the entry erroneously requires the SnpAttr bit to be set to zero.

The precise change

On page 2-124,

in the Table 2-14 “Snoop attributes for the different transaction types” replace footnote

a. Not applicable, must be set to zero.

With

a. Not applicable, not used to determine Snoopability. See section 8.3.7 “DVM domain” on page 8-321.

2.2 R580: SLCRepHint value in retried request

This specification requires certain fields in retried requests to be the same value as in the original request, while permitting certain others to be different. There are several reasons for permitting different values. AllowRetry in the retried request is changed to zero where-as the value in the original request must be one. PCrdType in the retried request is copied from the value in the RetryAck response. Others such as TxnID, QoS among others whose value is derived from internal heuristics are permitted to be different. SLCRepHint is also a field whose value is dynamic and derived from internal heuristics thus this update relaxes SLCRepHint to be added to the list of fields that can have a different value in the retried request.

The precise change

On page 2-147, at the end of bulleted list under " The transaction that is resent must have the same field values as the original request, except when the field is inapplicable or is one of the following:" replace

- TraceTag.

With

- TraceTag.
- SLCRepHint.

2.3 D549: Permitted to use ReqOrder ordering for WriteNoSnpZero

A WriteNoSnpZero transaction is permitted to use Request Order or Endpoint Order, as described in the Ordering Requirements section on p121. However, the summary on p173 has omitted Request Order. The specification is updated to include this ordering requirement in the summary.

The precise changes

On page 4-173 in bulleted items for WriteNoSnpZero replace,

- From HN-I to SN-I: No Order and Endpoint Order only.

With

- From HN-I to SN-I: No Order, Request Order and Endpoint Order only.

2.4 D550: Permitted ReadNoSnpSep.Order field values.

A ReadNoSnpSep transaction is permitted to use Order field set to 0b00 or 0b01, in the same manner that ReadNoSnp is permitted to. However, the summary on p165 has omitted the use of 0b00. The specification is updated to include this ordering requirement in the summary.

The precise changes

On page 4-165 in bulleted items for ReadNoSnpSep replace,

- The Order field of the request must be set to b01.

with

- Permitted Order field values are b00 and b01.

Add the following note to the update.

This Update applies retrospectively to CHI Issue C and Issue D.

2.5 D541: Change of Physical Instruction Cache Invalidate DVM message required behavior

AMBA specifications support the use of DVM (Distributed Virtual Memory) messages to allow the broadcast maintenance of certain caching structures within a system that are not covered by the standard data coherency protocol. DVM messages support TLB Invalidation, Branch Predictor Invalidation, and Instruction Cache Invalidation.

For the purposes of Instruction Cache Invalidation there are two variants of message that can be used, either a Physical Instruction Cache Invalidation or a Virtual Instruction Cache Invalidation. A component that receives DVM messages must support both forms of message, independent of the style of instruction cache implemented. It might be necessary to over-invalidate in the case where a message is received in a format that is not native to the cache type.

For the IC IALLUIS instruction, under certain conditions, is required to act on all instruction cache lines, which includes both Non-secure and Secure entries. With the current AMBA specifications there is a Virtual Instruction Cache Invalidate message that is required to operate on both Non-secure and Secure entries. However, the Physical Instruction Cache Invalidate messages are specified to act on either Non-secure entries or Secure entries, but not both.

It appears that a large number of implementations have used an approach where the Secure Physical Instruction Cache Invalidate messages causes an invalidation of both Non-secure and Secure entries.

To match this de-facto approach, an erratum is raised on the specification to modify the required Secure Physical Instruction Cache Invalidate behavior so that it requires the invalidation of both Non-secure and Secure entries.

This modification does mean that the encoding of the Security bits is non-standard for the Physical Instruction Cache Invalidate message.

This erratum is applied retrospectively to all earlier versions of the CHI specification.

The precise changes

1. On page 8-305 in Table 8-7 “DVMOp fields and encodings” replace,

Security	2	Indicate that the transaction applies to:	
		0b00	Secure and Non-secure
		0b01	Non-secure address from a Secure context
		0b10	Secure
		0b11	Non-secure

With

Security	2	Indicate that the transaction applies to:	
		0b00	Secure and Non-secure
		0b01	Non-secure address from a Secure context
		0b10	Secure and Non-secure, if DVMOp type is PICI and the invalidation operation is cache Invalidate all Secure, Otherwise
		0b11	Non-secure

2. On page 8-317 in Table 8-25 “Physical Instruction Cache Invalidate operations” replace,

   Secure Physical Address Cache Invalidate all

With

   Secure and Non-secure Physical Address Cache Invalidate all

2.6 C556: Corrupt data must be marked as in error.

The specification is updated to clarify that read data or write data that is corrupt must be marked with Poison, DERR or NDERR.

The precise changes

1. On page 9-327 in section 9.4.1 “Read transactions” add the following,

   Read data which is known to be corrupt must have appropriate Error indication where the error can be Poison, DERR, or NDERR.

2. On page 9-330 in section 9.4.3 “Write transactions” add the following,

   Write data which is known to be corrupt must have appropriate Error indication where the error can be Poison or DERR.

3. On page 9-332 in section 9.4.4 “Atomic transactions” add the following.

   Read data or Write data which is known to be corrupt must have appropriate Error indication. Where error on Read data can be Poison, DERR, or NDERR and on Write data can be Poison or DERR.

4. On page 9-335 in section 9.4.7 “Snoop transactions” add the following,

   Data which is known to be corrupt must have appropriate Error indication where the error can be Poison or DERR.

2.7 C543: Poison and MTE – Poison on MTE is not supported.

The specification is updated to provide additional implementation information regarding the use of Poison on MTE transactions that include Tags. This provides more clarity to the statement “Poison on tags is not supported”.

The precise changes

1. On page 12-376 replace

   Poison on tags is not supported.

With

   Poison on tags is not supported. See Poison on page 9-337.

2. On page 9-337 add the following to the end of section 9.5 "Poison".

---------- Note ----------

Although Poison on tags is not supported, implementations might choose to do one of the following. Other implementations are possible.

- Poison associated with the data results in the tag being poisoned. Depending on the granularity of the poison associated with the tag, it may not be possible to clear the poison using the same techniques that would be used to clear poison associated with data.
- Poison associated with the data does not result in the tag being poisoned. This means that a corrupted tag might subsequently be used in an MTE Match operation, which could incorrectly fail. The rate at which this occurs should be significantly lower than the rate at which data corruption occurs.
- A mixture of approaches can be used, depending on the caching or storage structures that are used.

2.8 D551: Write transaction Response packets legal RespErr field values

The table of legal RespErr fields is updated for two reasons. Firstly, to indicate that the DBIDResp column covers both DBIDResp and DBIDRespOrd responses. Secondly, to add the permitted RespErr fields for a CompDBIDResp response when used to complete a WriteNoSnpZero or WriteUniqueZero transaction.

The precise change

On page 9-330 in Table 9-6 “Write transaction Response packets legal RespErr field values”

1. Replace ‘DBIDResp’ with ‘DBIDResp*’.
2. Replace, ‘- - - -‘ with ‘Y N Y Y’, in cells corresponding to Write*Zero and CompDBIDResp.

2.9 D570: Transaction error cannot be indicated in DBIDResp response

The list of Atomic transaction responses that are permitted to include Data Error or Non-data Error is updated to fix a typographical error.

Precise change

On page 9-332 section titled “Atomic transactions” in the bulleted list for “A Data Error or Non-data Error can be signaled at the following points within a transaction:” delete the following item,

- With the DBIDResp response.

2.10 R548: Relaxation of relationship between request NS field and MPAMNS field.

The CHI-D specification introduced MPAM (Memory System Performance Resource Partitioning and Monitoring) signaling and includes a restriction that the MPAMNS field must not be asserted when the transaction request NS field is de-asserted. This combination corresponds to Non-secure partition context accessing a Secure location.

While this combination of fields is not expected, there are instances where preventing the use of this combination of fields causes unnecessary complexity in the implementation. One example of such a situation is where the eviction of a Secure line from a system level cache is caused by access to a Non-secure partition and the eviction transaction uses the MPAM values derived from the access that caused the eviction. The MPAM architecture does permit this behavior downstream of the last level cache.

The specification is relaxed to permit this combination of fields, thus removing the previous restriction. This relaxation is permitted at all points within a system to align with similar relaxations anticipated in future MPAM architecture releases.

This erratum is applied retrospectively to CHI-D and CHI-E specifications.

The precise change

On page 11-356 replace

- Text "Table 11-5 shows the permitted combinations of MPAMNS and NS." and
- Table 11-5 Permitted value combinations of MPAMNS and NS

With

This specification permits all four combinations of values of MPAMNS and NS.

------ Note -------

This change is applied retrospectively to CHI-D specification.

2.11 C544: WriteDataCancel MTE field values

The following information is added to explicitly state the requirements for the MTE field values associated with a WriteDataCancel message. Previously this information had to be derived from statements in the previous Permitted TagOp values section.

MTE fields in WriteDataCancel are inapplicable and must be set to zeroes. That is,

- TagOp = Invalid (00)
- TU = 0000
- Tags = all zeroes

The precise change

On page 12-369 in section 12.5.2 “TagOp, TU and tags relationship” add the following at the end,

> In WriteDataCancel write data response, irrespective of the TagOp value in the Write request, the MTE fields are inapplicable and must be set to zero.

2.12 D546: DoNotGoToSD field value in SnpQuery

The requirements for the DoNotGoToSD field in a SnpQuery are updated to fix a typographical error.

The precise change

On page 13-420 in section 13.10.35 “DoNotGoToSD”,

1. Delete SnpQuery from list of “Applicable and must be set to 1 in items”.
2. Replace

   The field is inapplicable and must be set to 0 in SnpDVMOp.

   With

   Inapplicable and must be set to 0 in:
   - SnpQuery
   - SnpDVMOp.

2.13 D545: Link Deactivation and sending of protocol flits

The requirements for link activation and deactivation are defined section 14.5. The rules around sending protocol flits during the DEACTIVATE state are inconsistent between the fully descriptive table, Table 14-2, and the summary table, Table 14-3.

In addition, the text below the tables can be deleted as the race condition requirements are included in the tables.

The precise change

On page 14-437

1. In Table 14-3 “Summary of behavior for each Request and Acknowledge state” entry under DEACT, replace

   Must not send flits, except for credit return flits.

   with

   Expected to send credit return flits. Can send any flits.

2. Delete the section titled “Race conditions”.

2.14 C587: Rewrite of Broadcast signals, BI/BO, BCM and BP, description

The specification is updated to clarify description of optional broadcast pins behavior, especially description of BI, BO, BCM and BP signals.

The precise changes

On page 16-460 in section 16.2 "Optional interface broadcast signals" replace the text starting with **"BROADCASTINNER and BROADCASTOUTER** .." till the end of the page and Table 16-1 "CMO broadcast at the interface with unspecified BI and BO" on the next page with the following text.

BROADCASTINNER and **BROADCASTOUTER**

Used to control the issuing of Non-snoopable transactions from an interface. This specification requires that these two pins must be set to the same value. When the signals are present and deasserted, all transactions are converted to Non-snoopable equivalents before they are sent. The section below lists the transaction conversions.

BROADCASTCACHEMAINTENANCE

Used to control the issuing of cache maintenance operations when there are no software-managed caches downstream of the interface.

When BROADCASTCACHEMAINTENANCE, BROADCASTINNER and BROADCASTOUTER are all present and deasserted:

- CleanShared, CleanInvalid and MakeInvalid transactions are not issued
- All Combined Write transactions must be converted to stand alone Write transactions.

BROADCASTPERSIST

Used to control the issuing of CleanSharedPersist CMOs.

When present and deasserted:

- CleanSharedPersist transactions are converted to CleanShared.
- The conversion applies to both standalone CMO and Combined Write transactions.
- Note that the issuing of the CleanShared is controlled by the BROADCASTINNER, BROADCASTOUTER and BROADCASTCACHEMAINTENANCE signals.

Conversion of Snoopable transactions to Non-snoopable

When the BROADCASTINNER / BROADCASTOUTER signals indicate that Snoopable transactions are converted to Non-snoopable, the following conversions apply:

- All read transactions must be converted to ReadNoSnp.
- All Snoopable CMO transactions must be converted to Non-snoopable CMO.

- Dataless transactions that is, CleanUnique, MakeUnique, Evict, StashOnce, StashOnce*Unique and StashOnce*Shared must be dropped.
- All Combined Write must be converted to Combined WriteNoSnp.
- All Write transactions, except WriteEvictFull and WriteEvictOrEvict, must be converted to WriteNoSnp.
- WriteEvictFull and WriteEvictOrEvict transactions must be dropped.
- All Snoopable Atomic transactions must be converted to Non-snoopable.
- PrefetchTgt transaction conversion is not required.

2.15 D547: Updates to Table A-3 Request Message field mapping part 2

The following are corrections to Table A-3 “Request Message field mapping part 2”, to make the table entries consistent with the requirements described in corresponding text in different locations in the specification.

2.15.1 TagGroupID value

In Write and Combined Write transactions TagGroupID is applicable when TagOp value in the request is a Match.

The precise change

TagGroupID in WriteNoSnpPtl(CMO), WriteNoSnpFull(CMO), WriteUniquePtlStash, WriteUniqueFullStash, WriteUniquePtlStash, WriteUniquePtl(CMO) and WriteUniqueFull(CMO),

Replace ‘-‘ with ‘Y’.

2.15.2 PGroupID value

In Combined Write transactions PGroupID is applicable when the combined CMO is a Persistent CMO.

The precise change

PGroupID in WriteNoSnpPtl(CMO), WriteNoSnpFull(CMO), WriteUniquePtl(CMO), WriteUniqueFull(CMO), WriteCleanFull(CMO) and WriteBackFull(CMO),

Replace ‘-‘ with ‘Y’.

2.15.3 Deep value

In Combined Write transactions, Deep field is applicable when the CMO is a Persistent CMO.

The precise change

Deep in all Write*(CMO),

Replace ‘-‘ with ‘Y’.