OBI 1

License Statement

Revision History

Revision	Date	Author	Comment
v1.0_draft00	12/2/2019	A.Bink	• Initial Revision
v1.0_draft01	1/31/2020	A.Bink	• Renamed to OBI 1 (Open Bus Interface, version 1). • Made all OBI signal names lower case and all parameter and property names upper case. • Removed ERROR_SIGNALING, RREADY_SIGNALING, and USER_SIGNALING properties. • Redefined DUSER to only be valid for read transactions and renamed to ruser. • Added wuser signal. • Added ADDR_WIDTH property. • Introduced COMB_GNT property. • Defined address and response phase more precisely. • Defined address and response phase signal validity more precisely. • Restricted DATA_WIDTH to 32 and 64. • Defined byte lane enables and address LSBs more precisely. • Introduced channel names (A, R). • Removed definitions related to misaligned load/store and address/response phase forwarding (as they are not related to the OBI protocol itself). • Added transaction IDs.
v1.0	3/5/2020	A.Bink	• Clarified address/response phase definition (Figure 4). • Added comments on unpredictable slave behavior in case of invalid *be*, *addr* signaling. • Added explanation on required consistency between *be* and *addr* values. • Added note that OBI is (only) defined for little-endian systems. • Redefined manner in which outstanding transactions are counted. • Updated proposed CV32E40P subsystem.

Table of contents

List of figures

List of tables

List of requirements

1 References

Reference ID	Reference Title	Comment/Location
[ARM-AMBA5-AHB]	ARM AMBA 5 AHB Protocol Specification (ARM IHI 0033B.b (ID102715))	https://developer.arm.com/docs/ihi0033/bb/arm-amba-5-ahb-protocol-specification
[ARM-AMBA5-AXI]	AMBA AXI and ACE Protocol Specification (ARM IHI 0022G (ID073019))	https://static.docs.arm.com/ihi0022/g/IHI0022G_amba_axi_protocol_spec.pdf
[PULP-RI5CY-UM]	PULP RI5CY: User Manual Revision 3.0	https://github.com/pulp-platform/riscv/blob/master/doc/user_manual.doc
[LOWRISC-IBEX-UM]	lowRISC Ibex User Manual	https://ibex-core.readthedocs.io/en/latest/

2 Introduction

This document is the Architecture Specification for the OBI (**O**pen **B**us **I**nterface) standard. To enable future extensions this specific document relates to OBI 1 (the version number will be increased for major updates).

The RISC-V Foundation (only) specifies the RISC-V ISA, not a specific implementation nor a related bus interface. OBI is the **O**pen **B**us **I**nterface protocol used for point-to-point bus interface connections of CV32E40* CPUs ([PULP-RI5CY-UM]) and related bus infrastructure components. The protocol is a request-grant based protocol similar to the AMBA AXI protocol ([ARM-AMBA5-AXI]).

The starting point for the OBI definition is the custom bus interface as used on the RI5CY ([PULP-RI5CY-UM]) and Ibex ([LOWRISC-IBEX-UM]) RISC-V cores. The protocol is clarified, restricted, extended and generalized to ease design of bus infrastructure components, to ease interfacing to standard AMBA protocols ([ARM-AMBA5-AHB], [ARM-AMBA5-AXI]), and to ease system level timing.

Bus protocol compliance of bus masters and slaves is an essential requirement of chip platforms. These protocols allow modules to interface with each other in a known standard way. Potential bugs that can result from breaking protocol range from low-order issues like incorrect write/read data all the way to full system deadlock.

2.1 General documentation notes

This document contains some color-coded text, e.g.:

- Highlighted text in yellow are specifications that still have ongoing investigation or may be of particular interest to reviewers before ratification
- Red text is used for notes to the reader

Additionally, the documentation contains a commentary format:

> *The commentary format is intended to provide relevant background regarding a design decision. It should not be used to communicate key architectural elements and specifications. The reader should be able to extract all architectural requirements even with ignoring commentary sections*

Some requirements only apply under certain conditions. Such conditions are included as *(condition)* at the start of a requirement.

3 Functional description

OBI is the bus interface protocol used for point-to-point bus interface connections on CV32E40* and related bus infrastructure components. All lines going in and out of the OBI crossbar (OBIXBAR) (and all lines inside the OBIXBAR) in Figure 1 are OBI links. Each OBI link consists of a so-called Address Channel (**A**) and a Response Channel (**R**).

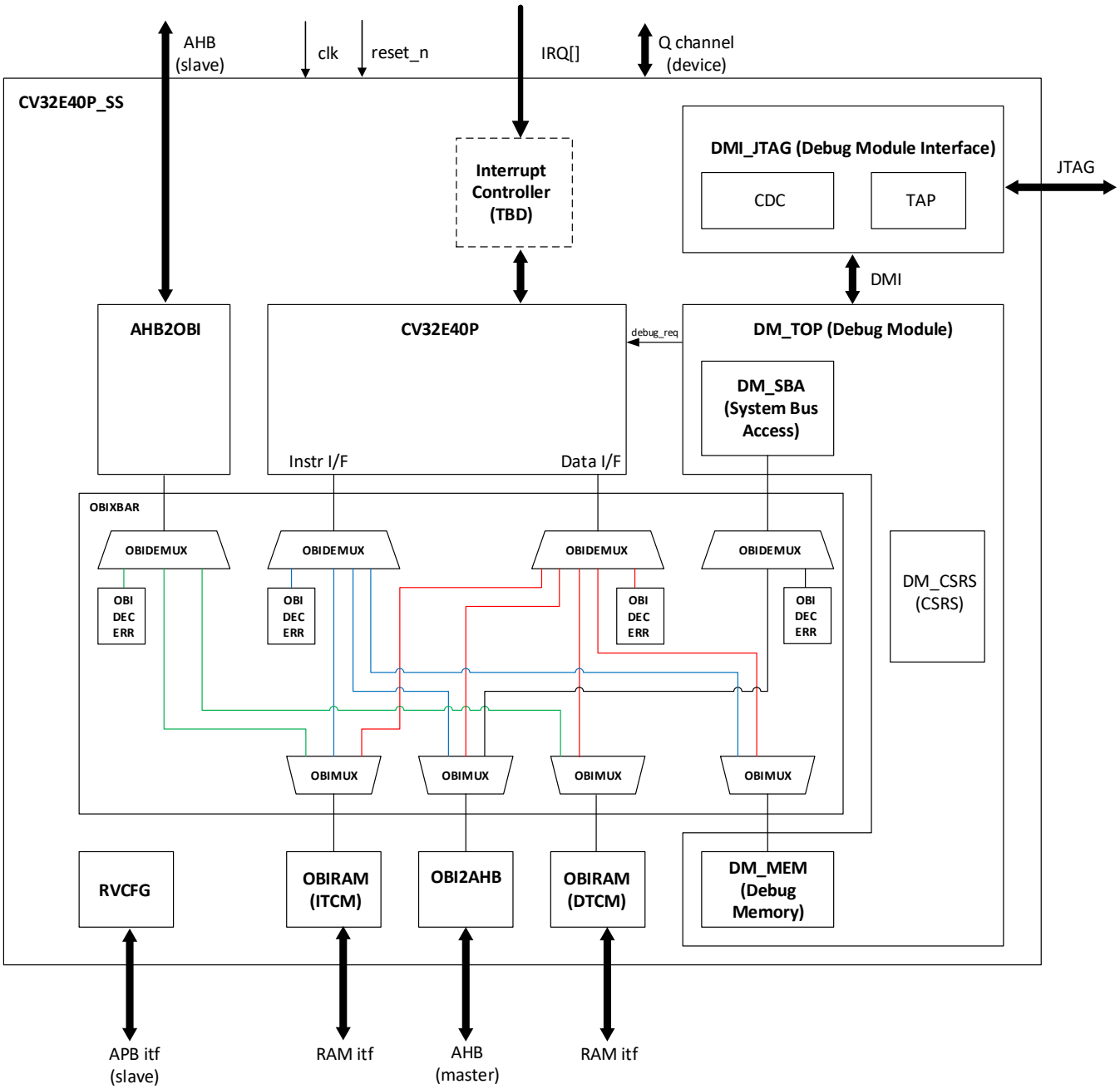

Figure 1 RISC-V subsystem (proposal for OpenHW BHAG)

3.1 Port list

Table 1 shows the pin names used in point-to-point OBI links. Note that this list differs slightly from the used OBI port names in RI5CY and CV32E40* (as in these CPUs the OBI port names are prefixed with *instr_* or *data_* (to differentiate the instruction and data side bus interfaces) and postfixed with *_i* or *_o* (to indicate the port direction)). Furthermore, RI5CY and CV32E40* do not support the *err*, *auser*, *wuser*, *ruser*, *aid*, *rid* extensions nor the *rready* handshake signal. Additions to the RI5CY bus interface are highlighted in green. Furthermore, non-used signals are omitted from these RISC-V cores (e.g. *we*, *be*, and *wdata* on the instruction interface).

Table 1 OBI port list

Name	Source	Destination	Description
Global signals			
clk	Clock source	All	The bus clock times all bus transfers. All signal timings are related to the rising edge of *clk*.
reset_n	Reset controller	All	The bus reset signal is active LOW and resets the system and the bus. This is the only active LOW signal.
A Channel signals			
req	Master	Slave	Address transfer request. *req*=1 signals the availability of valid address phase signals.
gnt	Slave	Master	Grant. Ready to accept address transfer. Address transfer is accepted on rising *clk* with *req*=1 and *gnt*=1.
addr[] [1, 5]	Master	Slave	Address
we	Master	Slave	Write Enable, high for writes, low for reads.
be[] [2, 5]	Master	Slave	Byte Enable. Is set for the bytes to write/read.
wdata[] [2, 5]	Master	Slave	Write data. Only valid for write transactions. Undefined for read transactions.
auser[] [3,4]	Master	Slave	Address Phase User signals. Valid for both read and write transactions.
wuser[] [3,4]	Master	Slave	Additional Address Phase User signals. Only valid for write transactions. Undefined for read transactions.
aid[] [4]	Master	Slave	Address Phase transaction identifier.
R Channel signals			
rvalid	Slave	Master	Response transfer request. *rvalid*=1 signals the availability of valid response phase signals. Used for both reads and writes.
rready [4]	Master	Slave	Ready to accept response transfer. Response transfer is accepted on rising *clk* with *rvalid*=1 and *rready*=1.
rdata[] [2, 5]	Slave	Master	Read data. Only valid for read transactions. Undefined for write transactions.
err [4]	Slave	Master	Error.
ruser[] [3,4]	Slave	Master	Response phase User signals. Only valid for read transactions. Undefined for write transactions.
rid[] [4]	Slave	Master	Response Phase transaction identifier.

[1] Default address width is 32-bit. Address width is controlled by the ADDR_WIDTH property.
[2] Default data width is 32-bit (so 32-bit rdata and wdata, 4-bit be). Data width is controlled by the DATA_WIDTH property.
[3] Default auser, wuser and RUSER are not present (0 width). Their widths are controlled by the AUSER_WIDTH, WUSER_WIDTH and RUSER_WIDTH properties respectively. The semantics of these signals is implementation defined.
[4] New signals not present on RI5CY, Ibex, or CV32E40*. The tie-off values as specified in section 3.7 shall be assumed if signals are not present.
[5] The OBI protocol is defined only for little-endian systems.

3.2 Properties

Table 2 OBI properties

Property	Default Value	Comment
COMB_GNT	False	Defines whether *gnt* is allowed to combinatorially depend on OBI inputs.
AUSER_WIDTH	0	Width of the *auser* signal. RI5CY, Ibex, CV32E40* do not have the *auser* signal.
WUSER_WIDTH	0	Width of the *wuser* signal. RI5CY, Ibex, CV32E40* do not have the *wuser* signal.
RUSER_WIDTH	0	Width of the *ruser* signal. RI5CY, Ibex, CV32E40* do not have the *ruser* signal.
ADDR_WIDTH	32	Width of the *addr* signal.
DATA_WIDTH	32	Width of the *rdata* and *wdata* signals. *be* width is DATA_WIDTH / 8. Valid DATA_WIDTH settings are 32 and 64.
ID_WIDTH	0	Width of the *aid* and *rid* signals.

3.3 Protocol adherence

OBI is a point-to-point protocol (between master and slave), similar to a much simplified AXI protocol. The protocol works as follows (simplified, details omitted in this initial description).

An OBI link consists of two channels (plus a global *clk* and *reset_n*):

- The address channel, called the **A** channel.
- The response channel, called the **R** channel.

An OBI transaction consists of two transfers:

- An address phase transfer over the A channel.
- A response phase transfer over the R channel.

The address phase transfer is as follows:

- The master indicates the validity of its address phase signals (i.e. *addr*, *wdata*, *we*, *be*, *auser*, *aid*, *wuser*) by settings its request (*req*) high.
- The slave indicates its readiness to accept the address phase signals by setting grant (*gnt*) high.
- The address phase of a transaction starts in the cycle in which *req* goes high and completes on the rising *clk* edge when both *req* and *gnt* are high.

Note that the write data (wdata) is also part of the address phase signals in the A channel.

The response phase transfer is as follows:

- After a granted request (in the A channel), the slave indicates the validity of its response phase signals (i.e. *rdata*, *err*, *ruser*, *rid*) by setting *rvalid* high.
- The master indicates its readiness to accept the response phase signals by setting *rready* high.
- The response phase of a transaction starts in the cycle in which *rvalid* goes high and completes on the rising *clk* edge when both *rvalid* and *rready* are high.

Note that the response phase of a transaction per definition has no overlap with the address phase of that same transaction. However, the response phase does not necessarily immediately follow the address phase as wait states can be inserted in between these two phases by keeping rvalid low (and this is not considered part of the response phase).

Figure 2 show a basic (fastest possible) OBI transaction plus its translation to AHB5.

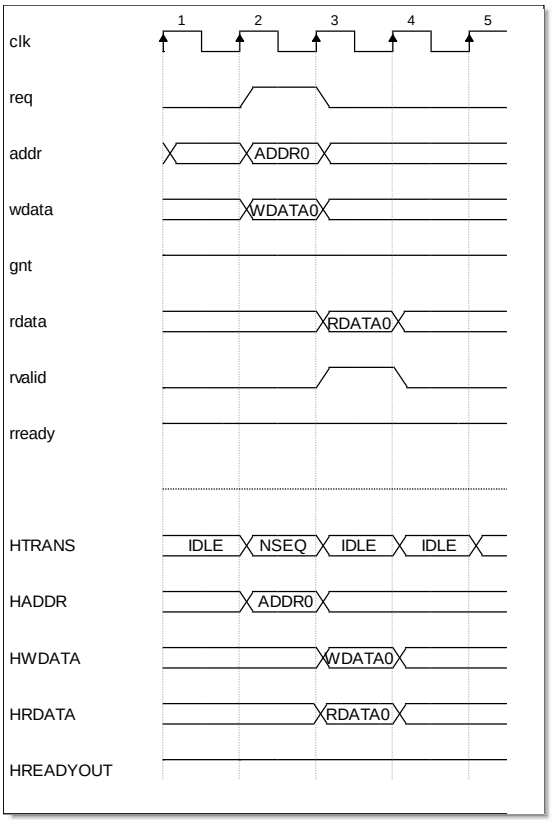

Figure 2 Basic OBI transfer (plus its mapping to AHB5)

The following definitions will be used in the remainder of this document:

- *Address phase signals*

  The signals from master to slave of which the validity is controlled by *req* (and *gnt*): *addr*, *wdata*, *we*, *be*, *auser*, *wuser*, *aid*.

  The value of the address phase signals is undefined outside the address phase.

In figures addr is often used as a representative of the address phase signals. Timing and validity of addr, wdata, we, be, auser, wuser, aid are equal.

- *Response phase signals*

  The signals from slave to master of which the validity is controlled by *rvalid* (and *rready*): *rdata*, *err*, *ruser*, *rid*.

  The value of the response phase signals is undefined outside the response phase. Note that this also implies that the value of *err* is undefined when *rvalid* is low.

In figures rdata is often used as a representative of the response phase signals. Timing and validity of rdata, err, ruser, rid are equal (although rdata and ruser are undefined for write transactions).

- *Address channel (**A**)*

The address channel (**A**) is the combination of *req*, *gnt*, and the address phase signals. An address channel is similar to AXI's write address channel, read address channel, and write data channel (although in OBI these channels are basically all mapped onto one channel).

- *Response channel (**R**)*

  The response channel (**R**) is the combination of *rvalid*, *rready*, and the response phase signals. A response channel is similar to AXI's write response channel and read data channel (although in OBI these channels are basically all mapped onto one channel).

- *OBI link*

  An OBI link is the combination of an address channel (**A**) with its associated response channel (**R**). An OBI link relies on the presence of a global *clk* and *reset_n* as well.

- *Address phase*

  The first cycle of the address phase is the cycle in which *req*=1; the last cycle of the address phase is the cycle in which both *req*=1 and *gnt*=1.

- *Address phase transfer*

  The transfer of address phase signals over the A channel from master to slave that takes place during the address phase.

- *Response phase*

  The first cycle of the response phase is the cycle in which *rvalid*=1; the last cycle of the response phase is the cycle in which both *rvalid*=1 and *rready*=1.

- *Response phase transfer*

  The transfer of response phase signals over the R channel from slave to master that takes place during the response phase.

- *In order*

  OBI links are in order, i.e. the response phase transfers are sent in in the same order as their corresponding address phase transfers were issued.

- *Transaction*

  An OBI (read or write) transaction consists of an address phase transfer which is (not necessarily immediately) followed by an associated response phase transfer. A transaction starts when its address phase transfer starts. A transaction ends when its response phase transfer ends.

- *Outstanding (accepted) transaction*

  A transaction that has been accepted, but has not ended yet is called an outstanding transaction (or equivalently, an outstanding accepted transaction). Multiple outstanding transactions can be present at the same time as new address phase transfers can be issued by the master before the slave has issued all corresponding response transfers.

Figure 3 gives an example of how the number of outstanding (granted) transactions is counted (assuming 0 transaction requests in the not-shown preceding cycles):

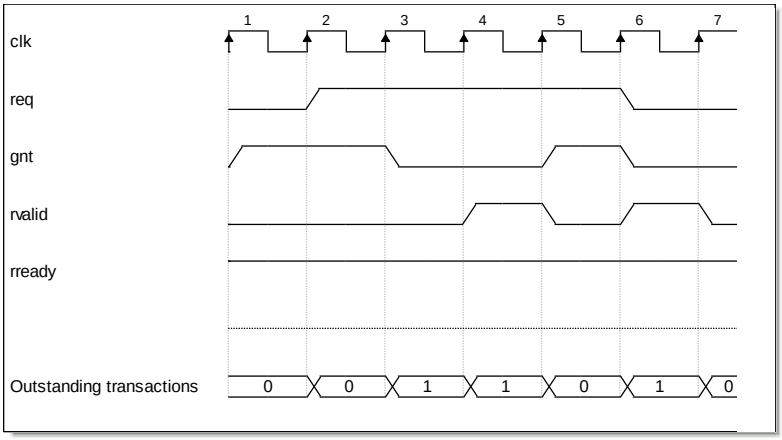

Figure 3 Outstanding (accepted) transactions

- clk cycle 1: A (default) *gnt* is present, but *req* is still 0 and no transactions have been accepted yet, so the number of outstanding (accepted) transactions is 0.
- clk cycle 2: *req* becomes 1 and this request will be accepted on the next rising *clk* edge. During this cycle the number of outstanding (accepted) transactions remains 0.
- clk cycle 3: The address phase of the first transaction has now been accepted and the corresponding response transfer has not started nor finished yet. The number of outstanding transactions therefore increases to 1.
- clk cycle 4: Nothing changes. *rvalid*=1 and *rready*=1, but this will only affect the count in the next cycle.
- clk cycle 5: The response phase of the first transaction finished and therefore the number of outstanding transactions decreases to 0. *req*=1 and *gnt*=1, but this only affects the count in the next cycle.
- clk cycle 6: The address phase of the second transaction has now finished, and therefore the outstanding (granted) transaction count increases to 1. *rvalid*=1 and *rready*=1, but this will only affect the count in the next cycle.
- clk cycle 7: The response phase of the second transaction finished and therefore the outstanding transaction count decreases to 0.

3.3.1 Clock and reset

R-1: OBI links shall rely on a single interface clock (*clk*) and reset (*reset_n*) that is common between the master and slave.

R-1.1: Masters and slaves shall only ever sample OBI signals on the positive edge of *clk*.

Any other sampled address or response channel signals shall be considered unstable and ignored, including data sampled on faster clock edges or negative clock edges.

R-1.2: Masters and slaves shall never generate OBI signals on the negative edge of *clk*.

R-1.3: OBI signals shall not be assumed to remain stable *between clk* edges (i.e. multi-cycle path exceptions shall never be used).

R-2: OBI protocol compliance requirements always apply except for during OBI reset assertion.

R-2.1: During reset assertion req shall be driven low.

R-2.2: During reset assertion *rvalid* shall be driven low.

OBI master or slave reconfiguration, software-based resets, clock disabling, etc. are no valid reasons to break OBI protocol compliance.

3.3.2 Handshake process

The OBI address channel uses a handshake process to transfer address phase info. The protocol is similar to AXI's VALID/READY handshake process (with OBI's *req* signal acting as AXI's VALID signal, and OBI's *gnt* signal acting as AXI's READY signal).

R-3: The address channel A shall use a two-way control handshake (*req*+*gnt*) between master and slave.

R-3.1: The master shall assert *req* to indicate the validity of the address phase signals.

R-3.1.1: The master shall keep its address phase signals stable during the address phase.

R-3.1.2: The master shall not de-assert (retract) *req* until after the last cycle of the address phase.

Request retraction is allowed during reset activation when both master and slave are being reset.

R-3.2: The slave shall indicate its readiness to accept the address phase transfer by asserting *gnt*.

R-3.2.1: The slave shall be allowed to assert gnt at any time (even before the corresponding req).

R-3.2.2: The slave shall be allowed to de-assert (retract) gnt at any time.

The OBI response channel uses a handshake process to transfer response phase info. The protocol is similar to AXI's VALID/READY handshake process (with OBI's *rvalid* signal acting as AXI's VALID signal, and OBI's *rready* signal acting as AXI's READY signal).

Certain masters (e.g. RI5CY) do not implement the rready signal. Such masters are always ready to accept the response transfer of any transaction they initiated. These masters can be thought of as having rready tied high.

R-4: The response channel R shall use a two-way control (*rvalid*+*rready*) between slave and master.

R-4.1: The slave shall assert *rvalid* to indicate the validity of the response phase signals.

R-4.1.1: The slave shall keep its response phase signals stable during the response phase.

R-4.1.2: The slave shall not de-assert (retract) rvalid until after the last cycle of the response phase.

R-4.1.3: Validity of *rdata* shall only be implied for read transactions.

R-4.2: The master shall indicate its readiness to accept the response phase transfer by asserting *rready*.

R-1.1.1: The master shall be allowed to assert *rready* at any time (even before the corresponding *rvalid*).

R-4.2.1: The master shall be allowed to de-assert (retract) *rready* at any time.

In addition to the above requirements for A and R channels separately, some requirements exist on the interaction between the A and R channels.

R-5: The response phase transfer for a transaction shall not start before the corresponding address phase transfer has finished (i.e. *req* and *gnt* need to have been sampled high before *rvalid*=1 is allowed).

So, for a single transaction the earliest that rvalid=1 can happen is in the clk cycle after both req and gnt have been sampled high. A slave can postpone its response by keeping rvalid=0 the initial cycles after both req and gnt have been sampled high.

Figure 4 shows an example of two OBI transactions with their address and response phases marked:

- The first transaction has a one cycle long address phase (marked AP0); the corresponding response phase (marked RP0) is delayed by the slave keeping *rvalid* = 0 during cycle 3. In this case the response phase only starts in cycle 4 and is one cycle long (as both *rvalid* and *rready* are 1 in that cycle).
- The second transaction has a three cycle long address phase (marked AP1); the corresponding response phase (marked RP1) is not delayed by the slave (as it starts the cycle after both *req* and *gnt* have been sampled high), but in this case the master stretches the response phase by keeping *rready* low during cycle 6 and 7.

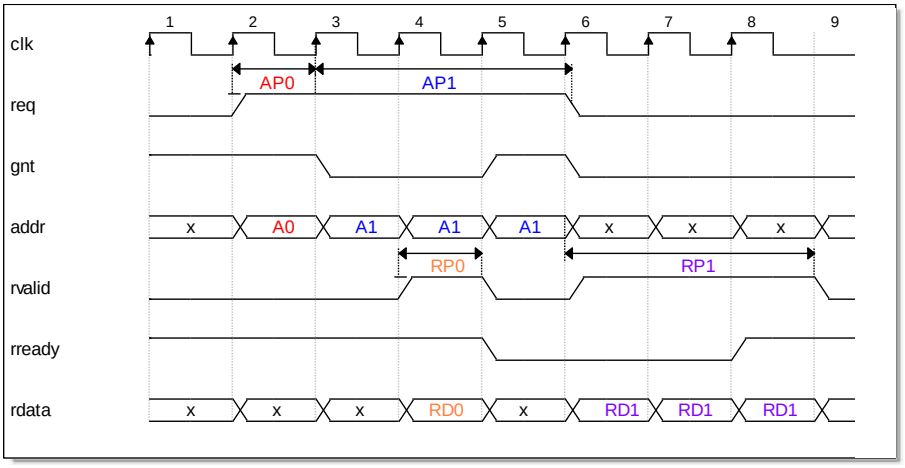

Figure 4 Address phase and response phase example

R-6: Response phase transfers shall be sent in the same order as their corresponding address phase transfers.

3.4 Signals

3.4.1 be

Not all *be* signal values are considered valid. The only valid *be* signals values are the ones needed to support transfers resulting from (possibly mis-aligned) byte, halfword, word, or double-word (for DATA_WIDTH=64) loads/stores.

A misaligned load/store is a load/store in which the address is not aligned to the related data, i.e. a load/store of a halfword with a non-halfword-aligned address or a load/store of a word with a non-word-aligned address. Masters that perform misaligned load/stores might have to issue multiple OBI transactions for each such load/store.

R-7: The *be* values during the address phase of a transaction shall be as follows:

- At least one of the *be* bits shall be set to 1.
- The 1's in *be* shall be contiguous.

So (for DATA_WIDTH=32) be = 4'b1000, be = 4'b0110, be = 4'b1110 are valid values whereas for example be = 4'b0000 and be = 4'b1010 are not valid values during the address phase of a transaction. Outside of the address phase of a transfer the be signal is not constrained and e.g. be = 4'b0000 is allowed.

The slave behavior in case of invalid *be* signaling is unpredictable. There is no requirement for slaves to signal an OBI error in case of invalid *be* signaling.

3.4.2 addr

R-8: The least significant *addr* bits (i.e. *addr[1:0]* in case DATA_WIDTH = 32, *addr[2:0]* in case DATA_WIDTH = 64) shall be consistent with the *be* value during the address phase of a transaction, i.e.:

- If *i* is the index of the least signification bit in *be* that is 1, then the least significant *addr* bits shall be <= *i*.

For example for DATA_WIDTH = 32: If be[] = 4'b0001, then as be[0] is the least significant set bit in be[], addr[1:0] shall be <= 0; if be[] = 4'b0100, then as be[2] is the least significant set bit in be[], addr[1:0] shall be <= 2; if be[] = 4'b0110, then as be[1] is the least significant set bit in be[], addr[1:0] shall be <= 1; reversely, addr[1:0] = 2'b00 is consistent with all be[] values that are allowed according to R-7: , and addr[1:0] = 2'b11 is consistent only with be[] = 4'b1000.

The slave behavior in case of invalid *addr* signaling is unpredictable. There is no requirement for slaves to signal an OBI error in case of invalid *addr-be* combinations.

3.4.3 aid, rid

The transaction identifiers (*aid*, *rid*) are intended to be used by interconnect infrastructure to route back the response transfers to the master from which the corresponding address phase transfer originated. For this purpose the interconnect will add bits to *aid* when routing a transfer towards a slave. A slave will mirror back the received *aid* via the *rid* of the same transaction, after which the interconnect will use (and strip) the added bits from *rid* to route back the response transfer to the correct master.

OBI links are always in-order (no matter *aid* or *rid*); the ordering model is not in any way related to the transaction identifiers.

R-9: For each OBI transaction an OBI slave shall 'mirror back' the value received on *aid* via *rid* (i.e. the *rid* for the response phase transfer shall be equal to the *aid* of the corresponding address phase transfer).

3.5 Dependencies

In order to ease system level integration and system level timing additional requirements are imposed on the OBI signals. These requirements are aimed at preventing deadlock, combinatorial loops, and unnecessarily long paths at the system level.

R-10: OBI link outputs (excluding *gnt*) shall not combinatorially depend on OBI link inputs, specifically (but not limited to):

R-10.1: For a master, req shall not combinatorially depend on gnt or rvalid.

R-10.2: For a master, rready shall not combinatorially depend on gnt or rvalid.

R-10.3: For a slave, rvalid shall not combinatorially depend on *req* or *rready*.

R-11: (COMB_GNT == False) *gnt* shall not combinatorially depend on OBI link inputs (default COMB_GNT = false).

R-12: (COMB_GNT == True) *gnt* is allowed to combinatorially depend on OBI link inputs.

Requirement R-10: would have been quite restrictive if it would apply to gnt as well. For OBI implementations with relaxed frequency requirements or for OBI links which are internal to a module, the requirements on gnt generation can sometimes be relaxed. An OBI link in which the gnt output is allowed to combinatorially depend on OBI link inputs is called a combinatorial gnt OBI link. Such an OBI link (i.e. with COMB_GNT = True) might limit achievable performance and is therefore discouraged; functional behavior is however not compromised.

Above constraints apply to single OBI links. If a module contains multiple OBI links additional requirements exist between these links.

R-13: OBI link outputs of any master interface shall not combinatorially depend on OBI link inputs of any other master interface.

For example, a module's master port 0 req output shall not depend on that module's master port 1 gnt input.

R-14: OBI link outputs of any slave interface shall not combinatorially depend on OBI link inputs of any other slave interface.

For example, a module's slave port 0 rvalid output shall not depend on that module's slave port 1 rready input.

Above constraints are all timing related. The following requirements relate to deadlock prevention.

R-15: A transaction's req shall not depend on the *gnt* for that transaction.

Note that the above requirement is wider than only disallowing combinatorial paths. The reverse dependency is allowed (i.e. gnt is allowed do depend on req (as long as it does not introduce a combinatorial path)).

R-16: A transaction's *rvalid* shall not depend on the *rready* for that transaction.

Note that the above requirement is wider than only disallowing combinatorial paths. The reverse dependency is allowed (i.e. rready is allowed do depend on rvalid (as long as it does not introduce a combinatorial path)).

3.6 Timing

OBI signals will have specific timing budgets allocated for master, slave, and interconnect modules. These budgets are considered vendor specific (and are not part of this public specification).

3.7 Tie offs

Some modules do not use all the OBI signals shown in Table 1 (e.g. the instruction OBI interface of RI5CY). In case such incomplete OBI interfaces need to be converted to a complete OBI interface, the following applies.

R-17: Incompletely connected OBI interfaces shall be tied off as shown in Table 3 unless specified otherwise.

Table 3 OBI default tie offs

Name	Default tie off	Description
rready	1'b1	RI5CY, Ibex, and CV32E40 do not have an *rready* signal. They are always ready to accept the response transfer for any granted transaction they issued.
we	1'b0	Read-only
be[3:0] [1]	4'b1111	All byte lanes enabled

rdata[31:0] [1]	32'b0	All bits 0 (this tie off is for write-only OBI ports)
wdata[31:0] [1]	32'b0	Write data ignored
auser[]	'b0	All bits 0
aid[]	'b0	All bits 0
wuser[]	'b0	All bits 0
err	1'b0	No error
ruser[]	'b0	All bits 0
[1] Default data width is 32-bit. Tie offs show the value for the default width only.		

3.8 Comparison with the RI5CY bus interface

The major differences between OBI and the bus interface as used in RI5CY are as follows:

- RI5CY does not use a *rready* signal (it can be thought of as having this signal tied to 1).
- RI5CY does not use an *err* signal (it can be thought of as ignoring this signal).
- RI5CY does not use a *aid* signal (it can be thought of as having this signal tied to 0).
- RI5CY does not necessarily keep its address phase signals stable during the address phase (see https://github.com/pulp-platform/riscv/issues/128).
- RI5CY has combinatorial paths from rvalid to req (see https://github.com/pulp-platform/riscv/issues/126)
- RI5CY does not allow for *gnt* to be asserted before *req* (https://github.com/pulp-platform/riscv/issues/127) and assumes that slaves will combinatorially generate a *gnt* based on *req*.