AXI Bus

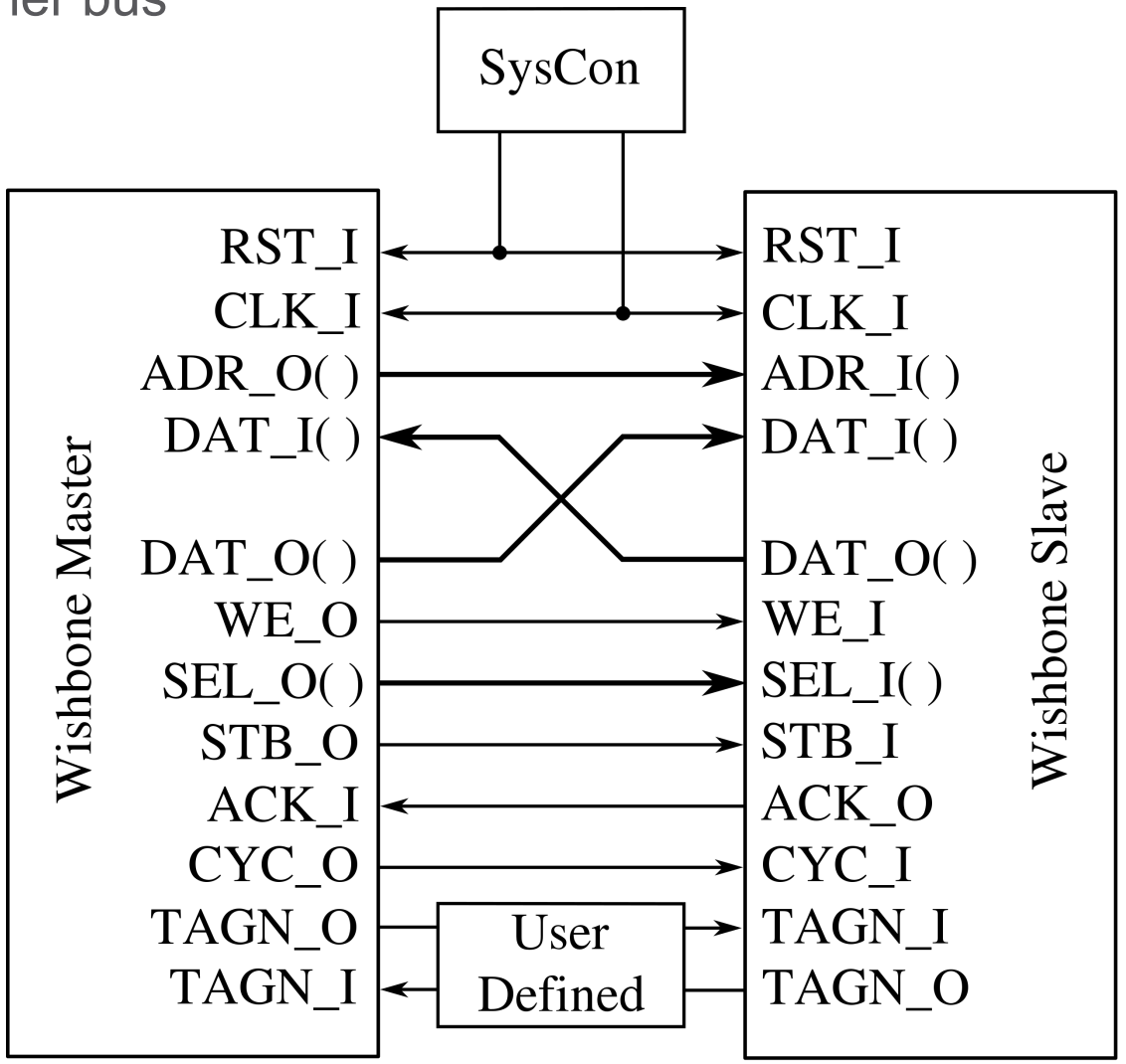
ECEN 427

BYU Electrical & Computer Engineering
IRA A. FULTON COLLEGE OF ENGINEERING

What is a bus?

Wishbone Bus

- First let's look at another bus



A1: Introduction

BYU Electrical & Computer Engineering
IRA A. FULTON COLLEGE OF ENGINEERING

The AXI protocol is burst-based and defines the following independent transaction channels:

- read address
- read data
- write address
- write data
- write response.

An address channel carries control information that describes the nature of the data to be transferred. The data is transferred between master and slave using either:

- A write data channel to transfer data from the master to the slave. In a write transaction, the slave uses the write response channel to signal the completion of the transfer to the master.
- A read data channel to transfer data from the slave to the master.

Reads

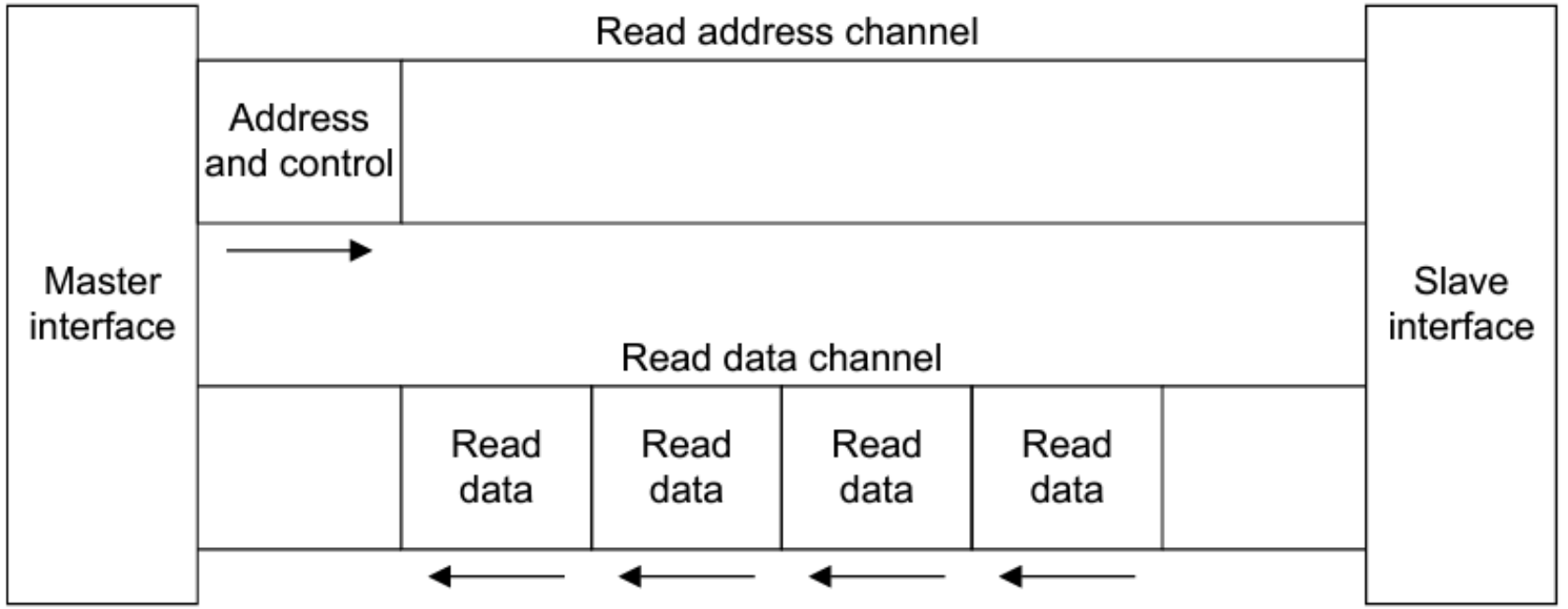


Figure A1-1 Channel architecture of reads

Writes

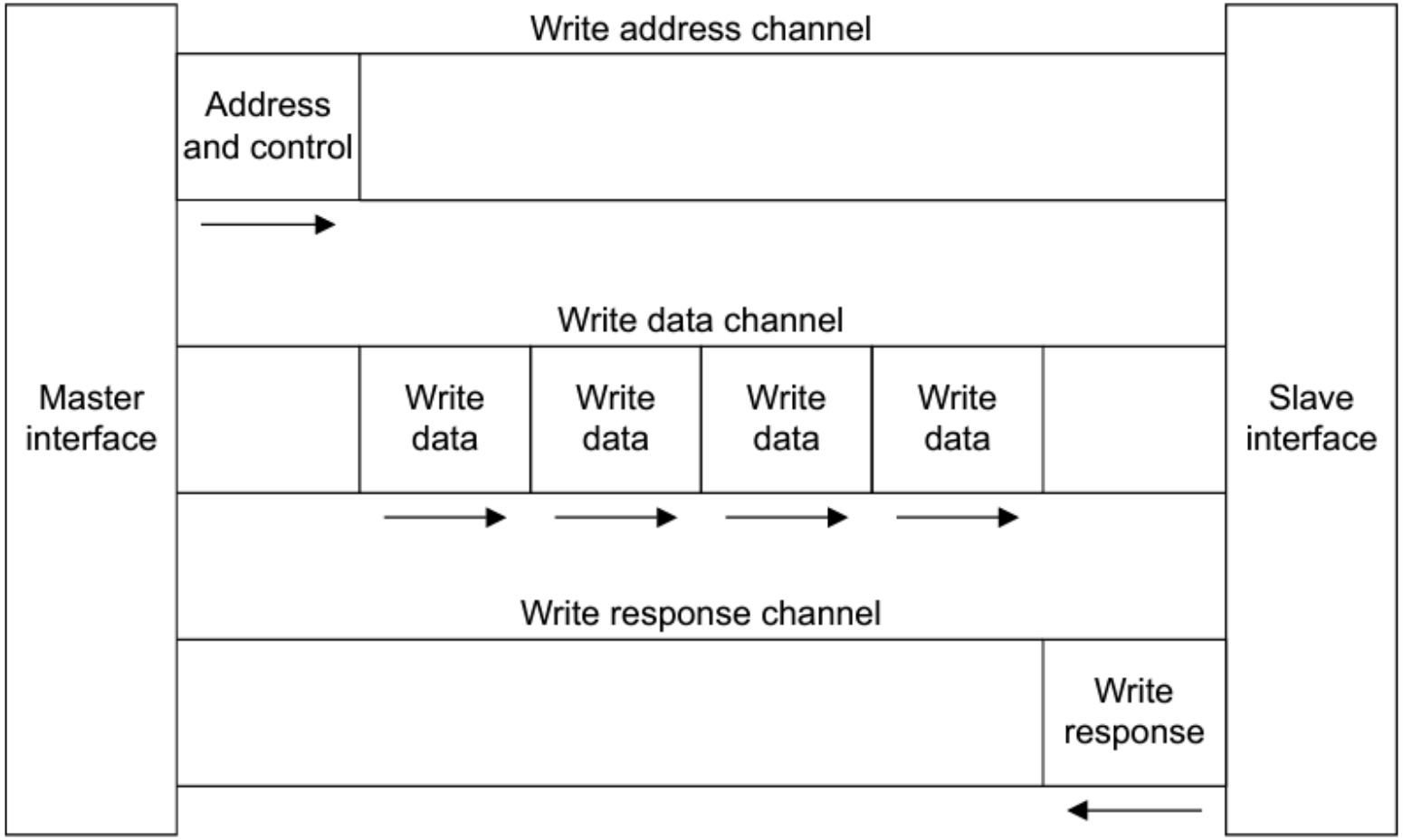


Figure A1-2 Channel architecture of writes

The AMBA AXI protocol supports high-performance, high-frequency system designs.

The AXI protocol:

- is suitable for high-bandwidth and low-latency designs
- provides high-frequency operation without using complex bridges
- meets the interface requirements of a wide range of components
- is suitable for memory controllers with high initial access latency
- provides flexibility in the implementation of interconnect architectures
- is backward-compatible with existing AHB and APB interfaces.

The key features of the AXI protocol are:

- separate address/control and data phases
- support for unaligned data transfers, using byte strobes
- uses burst-based transactions with only the start address issued
- separate read and write data channels, that can provide low-cost *Direct Memory Access* (DMA)
- support for issuing multiple outstanding addresses
- support for out-of-order transaction completion
- permits easy addition of register stages to provide timing closure.

Protocol

Each of the independent channels consists of a set of information signals and **VALID** and **READY** signals that provide a two-way handshake mechanism. See *Basic read and write transactions* on page A3-37.

The information source uses the **VALID** signal to show when valid address, data or control information is available on the channel. The destination uses the **READY** signal to show when it can accept the information. Both the read data channel and the write data channel also include a **LAST** signal to indicate the transfer of the final data item in a transaction.

Read data channel

The read data channel carries both the read data and the read response information from the slave to the master, and includes:

- the data bus, that can be 8, 16, 32, 64, 128, 256, 512, or 1024 bits wide
- a read response signal indicating the completion status of the read transaction.

Write data channel

The write data channel carries the write data from the master to the slave and includes:

- the data bus, that can be 8, 16, 32, 64, 128, 256, 512, or 1024 bits wide
- a byte lane strobe signal for every eight data bits, indicating which bytes of the data are valid.

Write data channel information is always treated as buffered, so that the master can perform write transactions without slave acknowledgement of previous write transactions.

Interconnect

A typical system consists of a number of master and slave devices connected together through some form of interconnect, as Figure A1-3 shows.

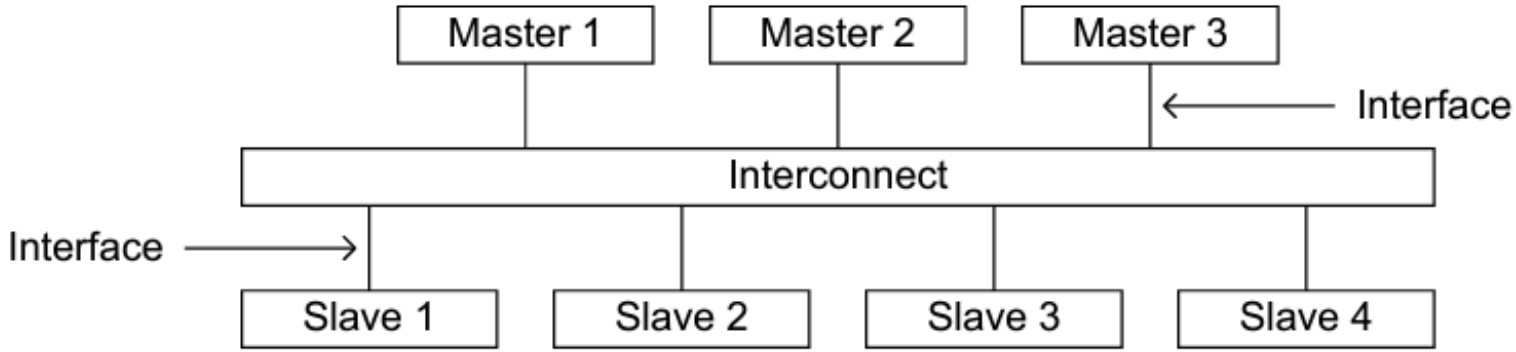


Figure A1-3 Interface and interconnect

The AXI protocol provides a single interface definition, for the interfaces:

- between a master and the interconnect
- between a slave and the interconnect
- between a master and a slave.

Registers

Register slices

Each AXI channel transfers information in only one direction, and the architecture does not require any fixed relationship between the channels. This means a register slice can be inserted at almost any point in any channel, at the cost of an additional cycle of latency.

Note

This makes possible:

- a trade-off between cycles of latency and maximum frequency of operation
- a direct, fast connection between a processor and high performance memory, but to use simple register slices to isolate a longer path to less performance critical peripherals.

A2: Signals

BYU Electrical & Computer Engineering
IRA A. FULTON COLLEGE OF ENGINEERING

Global Signals

Signal	Source	Description
ACLK	Clock source	Global clock signal. See *Clock* on page A3-36.
ARESETn	Reset source	Global reset signal, active LOW. See *Reset* on page A3-36.

All signals are sampled on the rising edge of the global clock.

Write Address

Signal	Source	Description
AWID	Master	Write address ID. This signal is the identification tag for the write address group of signals. See *Transaction ID* on page A5-77.
AWADDR	Master	Write address. The write address gives the address of the first transfer in a write burst transaction. See *Address structure* on page A3-44.
AWLEN	Master	Burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address. This changes between AXI3 and AXI4. See *Burst length* on page A3-44.
AWSIZE	Master	Burst size. This signal indicates the size of each transfer in the burst. See *Burst size* on page A3-45.
AWBURST	Master	Burst type. The burst type and the size information, determine how the address for each transfer within the burst is calculated. See *Burst type* on page A3-45.
AWLOCK	Master	Lock type. Provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4. See *Locked accesses* on page A7-95.
AWCACHE	Master	Memory type. This signal indicates how transactions are required to progress through a system. See *Memory types* on page A4-65.
AWPROT	Master	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access. See *Access permissions* on page A4-71.
AWQOS	Master	*Quality of Service*, QoS. The QoS identifier sent for each write transaction. Implemented only in AXI4. See *QoS signaling* on page A8-98.
AWREGION	Master	Region identifier. Permits a single physical interface on a slave to be used for multiple logical interfaces. Implemented only in AXI4. See *Multiple region signaling* on page A8-99.
AWUSER	Master	User signal. Optional User-defined signal in the write address channel. Supported only in AXI4. See *User-defined signaling* on page A8-100.
AWVALID	Master	Write address valid. This signal indicates that the channel is signaling valid write address and control information. See *Channel handshake signals* on page A3-38.
AWREADY	Slave	Write address ready. This signal indicates that the slave is ready to accept an address and associated control signals. See *Channel handshake signals* on page A3-38.

Write Data

Signal	Source	Description
WID	Master	Write ID tag. This signal is the ID tag of the write data transfer. Supported only in AXI3. See *Transaction ID* on page A5-77.
WDATA	Master	Write data.
WSTRB	Master	Write strobes. This signal indicates which byte lanes hold valid data. There is one write strobe bit for each eight bits of the write data bus. See *Write strobes* on page A3-49.
WLAST	Master	Write last. This signal indicates the last transfer in a write burst. See *Write data channel* on page A3-39.
WUSER	Master	User signal. Optional User-defined signal in the write data channel. Supported only in AXI4. See *User-defined signaling* on page A8-100.
WVALID	Master	Write valid. This signal indicates that valid write data and strobes are available. See *Channel handshake signals* on page A3-38.
WREADY	Slave	Write ready. This signal indicates that the slave can accept the write data. See *Channel handshake signals* on page A3-38.

Write Response

Signal	Source	Description
BID	Slave	Response ID tag. This signal is the ID tag of the write response. See *Transaction ID* on page A5-77.
BRESP	Slave	Write response. This signal indicates the status of the write transaction. See *Read and write response structure* on page A3-54.
BUSER	Slave	User signal. Optional User-defined signal in the write response channel. Supported only in AXI4. See *User-defined signaling* on page A8-100.
BVALID	Slave	Write response valid. This signal indicates that the channel is signaling a valid write response. See *Channel handshake signals* on page A3-38.
BREADY	Master	Response ready. This signal indicates that the master can accept a write response. See *Channel handshake signals* on page A3-38.

Read Address

Signal	Source	Description
ARID	Master	Read address ID. This signal is the identification tag for the read address group of signals. See *Transaction ID* on page A5-77.
ARADDR	Master	Read address. The read address gives the address of the first transfer in a read burst transaction. See *Address structure* on page A3-44.
ARLEN	Master	Burst length. This signal indicates the exact number of transfers in a burst. This changes between AXI3 and AXI4. See *Burst length* on page A3-44.
ARSIZE	Master	Burst size. This signal indicates the size of each transfer in the burst. See *Burst size* on page A3-45.
ARBURST	Master	Burst type. The burst type and the size information determine how the address for each transfer within the burst is calculated. See *Burst type* on page A3-45.
ARLOCK	Master	Lock type. This signal provides additional information about the atomic characteristics of the transfer. This changes between AXI3 and AXI4. See *Locked accesses* on page A7-95.
ARCACHE	Master	Memory type. This signal indicates how transactions are required to progress through a system. See *Memory types* on page A4-65.
ARPROT	Master	Protection type. This signal indicates the privilege and security level of the transaction, and whether the transaction is a data access or an instruction access. See *Access permissions* on page A4-71.
ARQOS	Master	*Quality of Service*, QoS. QoS identifier sent for each read transaction. Implemented only in AXI4. See *QoS signaling* on page A8-98.
ARREGION	Master	Region identifier. Permits a single physical interface on a slave to be used for multiple logical interfaces. Implemented only in AXI4. See *Multiple region signaling* on page A8-99.
ARUSER	Master	User signal. Optional User-defined signal in the read address channel. Supported only in AXI4. See *User-defined signaling* on page A8-100.
ARVALID	Master	Read address valid. This signal indicates that the channel is signaling valid read address and control information. See *Channel handshake signals* on page A3-38.
ARREADY	Slave	Read address ready. This signal indicates that the slave is ready to accept an address and associated control signals. See *Channel handshake signals* on page A3-38.

Read Data

Signal	Source	Description
RID	Slave	Read ID tag. This signal is the identification tag for the read data group of signals generated by the slave. See *Transaction ID* on page A5-77.
RDATA	Slave	Read data.
RRESP	Slave	Read response. This signal indicates the status of the read transfer. See *Read and write response structure* on page A3-54.
RLAST	Slave	Read last. This signal indicates the last transfer in a read burst. See *Read data channel* on page A3-39.
RUSER	Slave	User signal. Optional User-defined signal in the read data channel. Supported only in AXI4. See *User-defined signaling* on page A8-100.
RVALID	Slave	Read valid. This signal indicates that the channel is signaling the required read data. See *Channel handshake signals* on page A3-38.
RREADY	Master	Read ready. This signal indicates that the master can accept the read data and response information. See *Channel handshake signals* on page A3-38.

A3: Single Interface Requirements

BYU Electrical & Computer Engineering
IRA A. FULTON COLLEGE OF ENGINEERING

Clock

Each AXI component uses a single clock signal, **ACLK**. All input signals are sampled on the rising edge of **ACLK**. All output signal changes must occur after the rising edge of **ACLK**.

On master and slave interfaces there must be no combinatorial paths between input and output signals.

Reset

The AXI protocol uses a single active LOW reset signal, **ARESETn**. The reset signal can be asserted asynchronously, but deassertion must be synchronous with a rising edge of **ACLK**.

During reset the following interface requirements apply:

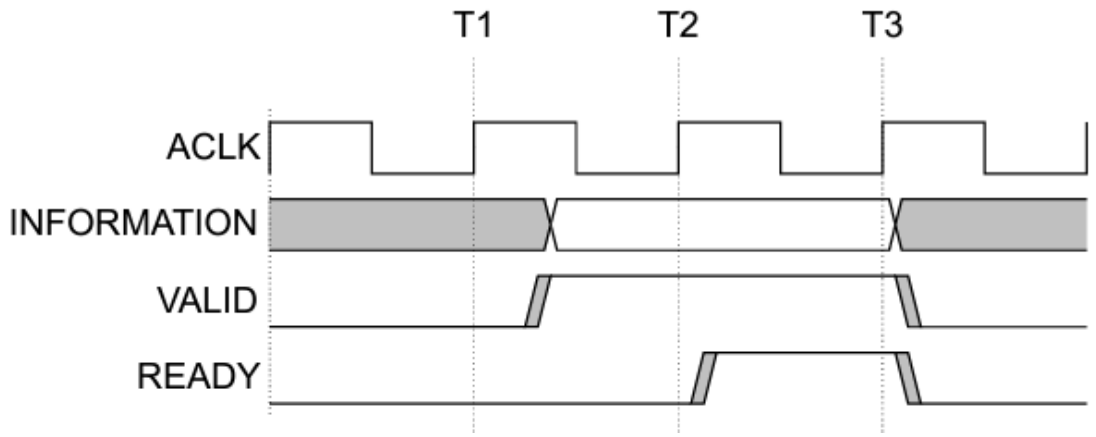
- a master interface must drive **ARVALID**, **AWVALID**, and **WVALID** LOW
- a slave interface must drive **RVALID** and **BVALID** LOW
- all other signals can be driven to any value.

Handshake Process

All five transaction channels use the same **VALID/READY** handshake process to transfer address, data, and control information. This two-way flow control mechanism means both the master and slave can control the rate at which the information moves between master and slave. The *source* generates the **VALID** signal to indicate when the address, data or control information is available. The *destination* generates the **READY** signal to indicate that it can accept the information. Transfer occurs only when *both* the **VALID** and **READY** signals are HIGH.

On master and slave interfaces there must be no combinatorial paths between input and output signals.

In Figure A3-2, the source presents the address, data or control information after T1 and asserts the **VALID** signal. The destination asserts the **READY** signal after T2, and the source must keep its information stable until the transfer occurs at T3, when this assertion is recognized.



A source is not permitted to wait until **READY** is asserted before asserting **VALID**.

Once **VALID** is asserted it must remain asserted until the handshake occurs, at a rising clock edge at which **VALID** and **READY** are both asserted.

Ready before Valid

In Figure A3-3, the destination asserts **READY**, after T1, before the address, data or control information is valid, indicating that it can accept the information. The source presents the information, and asserts **VALID**, after T2, and the transfer occurs at T3, when this assertion is recognized. In this case, transfer occurs in a single cycle.

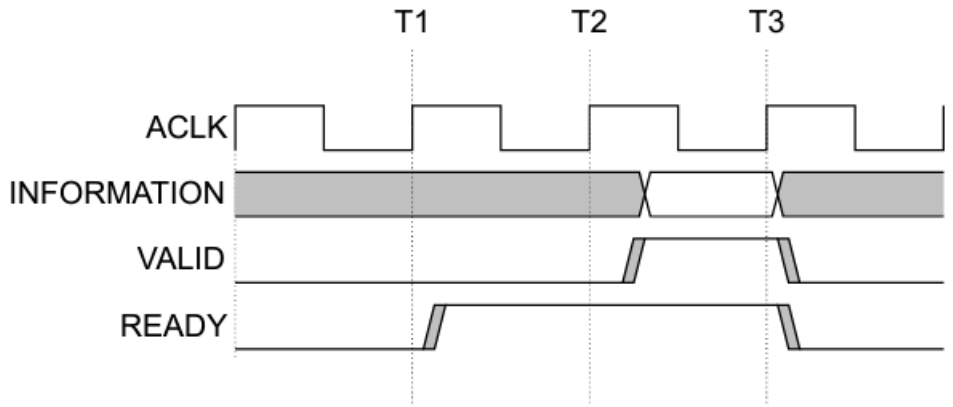
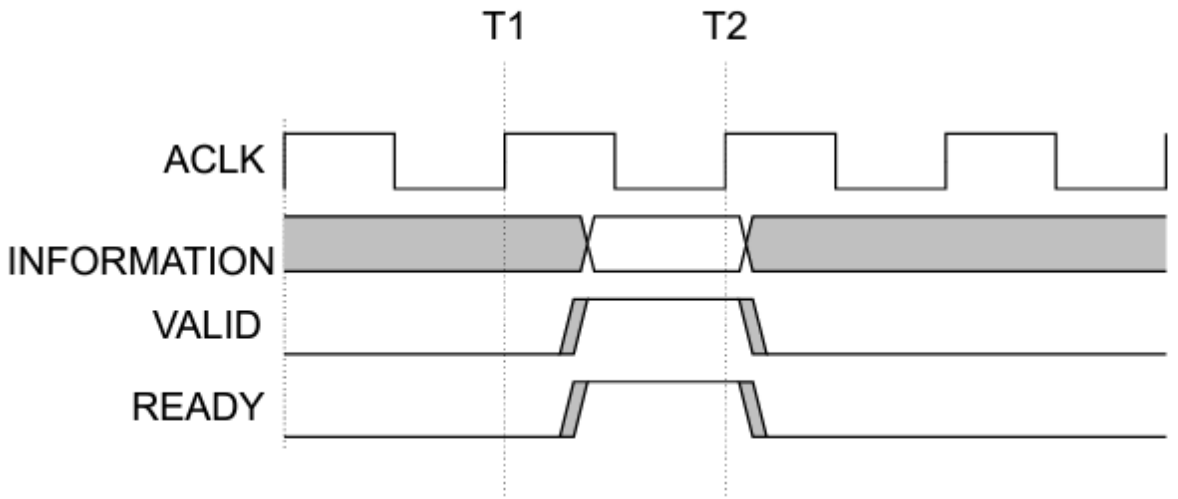


Figure A3-3 READY before VALID handshake

A destination is permitted to wait for **VALID** to be asserted before asserting the corresponding **READY**.

If **READY** is asserted, it is permitted to deassert **READY** before **VALID** is asserted.

In Figure A3-4, both the source and destination happen to indicate, after T1, that they can transfer the address, data or control information. In this case the transfer occurs at the rising clock edge when the assertion of both **VALID** and **READY** can be recognized. This means the transfer occurs at T2.



Valid/Ready per Channel

Each channel has its own **VALID/READY** handshake signal pair. Table A3-1 shows the signals for each channel.

Table A3-1 Transaction channel handshake pairs

Transaction channel	Handshake pair
Write address channel	**AWVALID, AWREADY**
Write data channel	**WVALID, WREADY**
Write response channel	**BVALID, BREADY**
Read address channel	**ARVALID, ARREADY**
Read data channel	**RVALID, RREADY**

Relationships between the channels

The AXI protocol requires the following relationships to be maintained:

- a write response must always follow the last write transfer in the write transaction of which it is a part
- read data must always follow the address to which the data relates
- channel handshakes must conform to the dependencies defined in *Dependencies between channel handshake signals*.

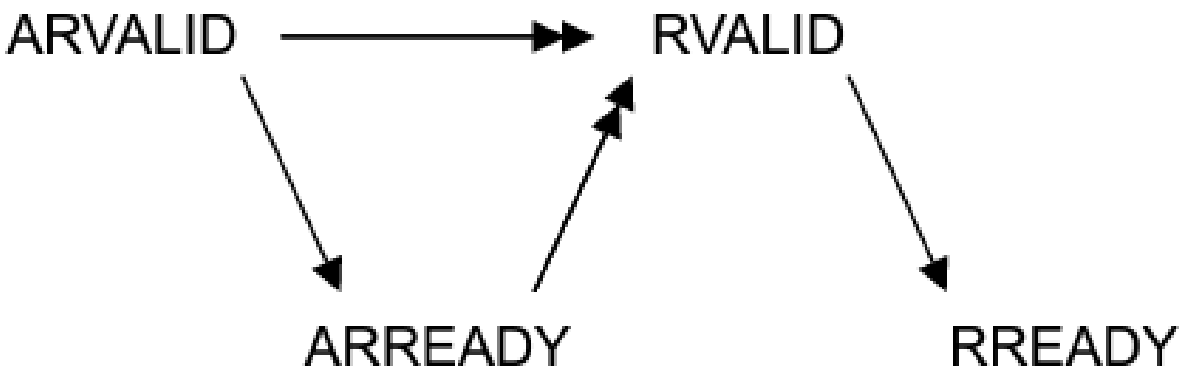
Otherwise, the protocol does not define any relationship between the channels.

This means, for example, that the write data can appear at an interface before the write address for the transaction. This can occur if the write address channel contains more register stages than the write data channel. Similarly, the write data might appear in the same cycle as the address.

Dependency Diagrams

In the dependency diagrams:

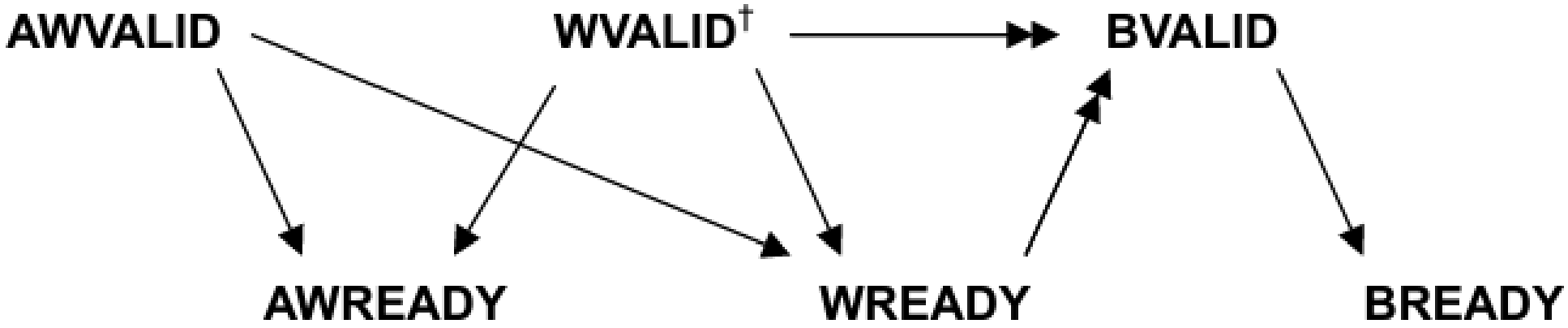
- **single-headed arrows** point to signals that ***can*** be asserted ***before or after*** the signal at the start of the arrow
- **double-headed arrows** point to signals that ***must be asserted only after*** assertion of the signal at the start of the arrow.



Dependency Diagrams

In the dependency diagrams:

- **single-headed arrows** point to signals that ***can*** be asserted ***before or after*** the signal at the start of the arrow
- **double-headed arrows** point to signals that ***must be asserted only after*** assertion of the signal at the start of the arrow.

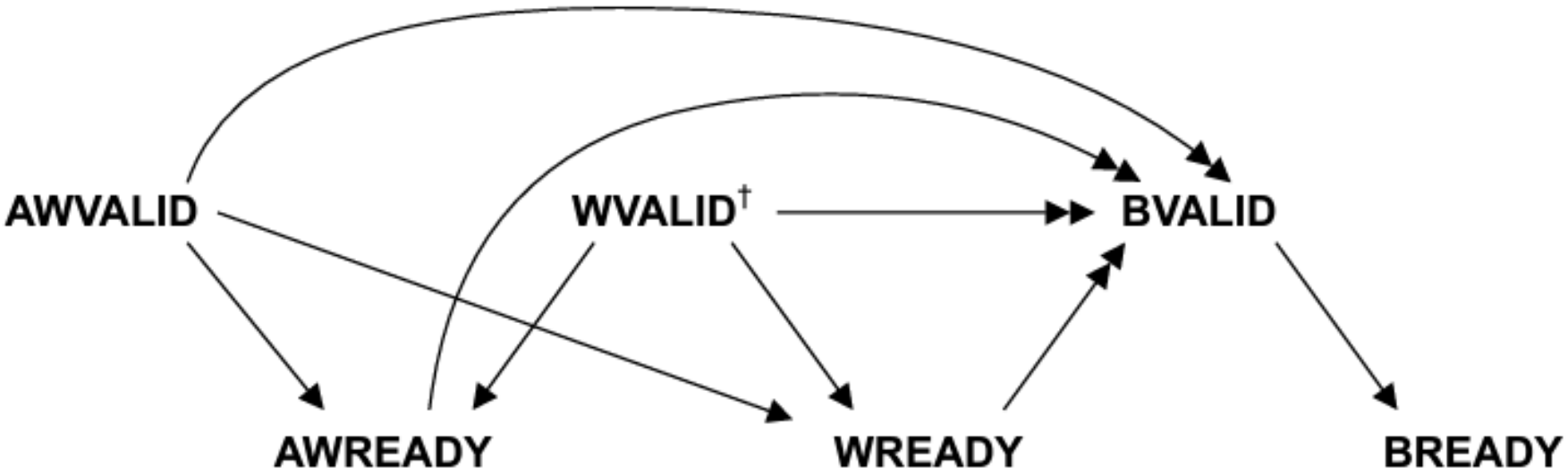


Caution

The dependency rules must be observed to prevent a deadlock condition. For example, a master must not wait for **AWREADY** to be asserted before driving **WVALID**. A deadlock condition can occur if the slave is waiting for **WVALID** before asserting **AWREADY**.

AXI4 (vs AXI3)

Write response isn't provided until data **AND** address are provided.



Burst Length

Burst length

The burst length is specified by:

- **ARLEN[7:0]**, for read transfers
- **AWLEN[7:0]**, for write transfers.

No component can terminate a burst early. However, to reduce the number of data transfers in a write burst, the master can disable further writing by deasserting all the write strobes. In this case, the master must complete the remaining transfers in the burst. In a read burst, the master can discard read data, but it must complete all transfers in the burst.

Burst Size

The maximum number of bytes to transfer in each data transfer, or beat, in a burst, is specified by:

- **ARSIZE[2:0]**, for read transfers
- **AWSIZE[2:0]**, for write transfers.

In this specification, **AxSIZE** indicates **ARSIZE** or **AWSIZE**.

Table A3-2 shows the **AxSIZE** encoding.

Table A3-2 Burst size encoding

AxSIZE[2:0]	Bytes in transfer
0b000	1
0b001	2
0b010	4
0b011	8
0b100	16
0b101	32
0b110	64
0b111	128

If the AXI bus is wider than the burst size, the AXI interface must determine from the transfer address which byte lanes of the data bus to use for each transfer. See *Data read and write structure* on page A3-49.

Burst Type

FIXED In a fixed burst, the address is the same for every transfer in the burst.

This burst type is used for repeated accesses to the same location such as when loading or emptying a FIFO.

INCR Incrementing. In an incrementing burst, the address for each transfer in the burst is an increment of the address for the previous transfer. The increment value depends on the size of the transfer. For example, the address for each transfer in a burst with a size of four bytes is the previous address plus four.

This burst type is used for accesses to normal sequential memory.

WRAP A wrapping burst is similar to an incrementing burst, except that the address wraps around to a lower address if an upper address limit is reached.

AxBURST[1:0]	Burst type
0b00	FIXED
0b01	INCR
0b10	WRAP
0b11	Reserved

Write Strobes

Write strobes

The **WSTRB[n:0]** signals when HIGH, specify the byte lanes of the data bus that contain valid information. There is one write strobe for each eight bits of the write data bus, therefore **WSTRB[n]** corresponds to **WDATA[(8n)+7: (8n)]**.

A master must ensure that the write strobes are HIGH only for byte lanes that contain valid data.

When **WVALID** is LOW, the write strobes can take any value, although this specification recommends that they are either driven LOW or held at their previous value.

- the burst has five transfers
- the starting address is 0
- each transfer is eight bits
- the transfers are on a 32-bit bus
- the burst type is INCR.

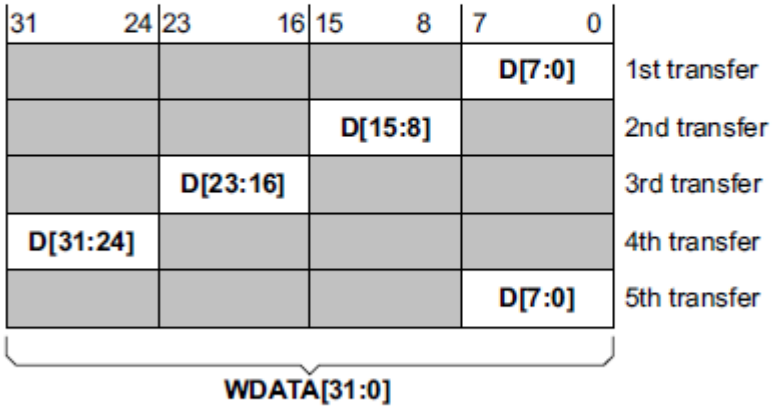


Figure A3-8 Narrow transfer example with 8-bit transfers

- the burst has three transfers
- the starting address is 4
- each transfer is 32 bits
- the transfers are on a 64-bit bus.

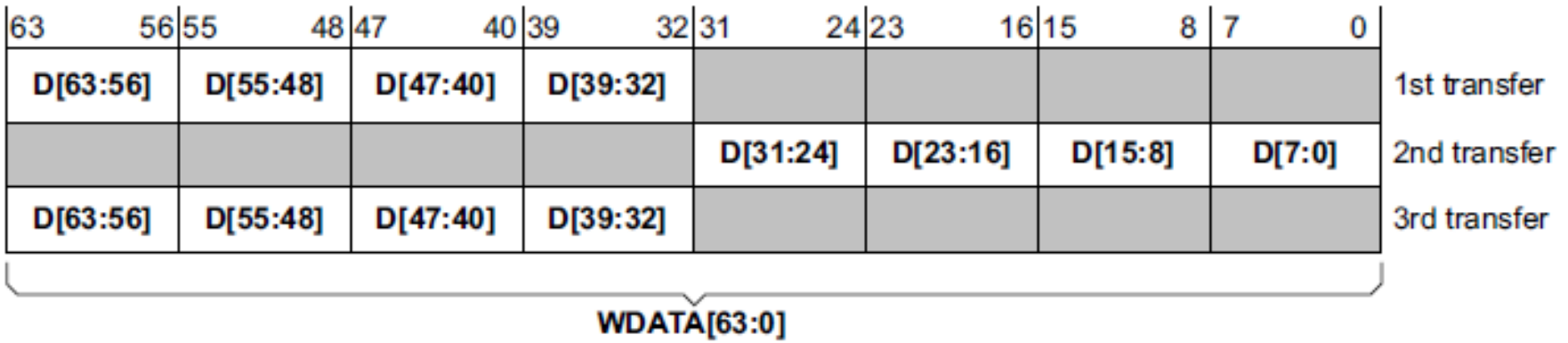


Figure A3-9 Narrow transfer example with 32-bit transfers

Read and write response structure

The AXI protocol provides response signaling for both read and write transactions:

- for read transactions the response information from the slave is signaled on the read data channel
- for write transactions the response information is signaled on the write response channel.

The responses are signaled by:

- **RRESP[1:0]**, for read transfers
- **BRESP[1:0]**, for write transfers.

OKAY Normal access success. Indicates that a normal access has been successful. Can also indicate an exclusive access has failed. See *OKAY, normal access success*.

EXOKAY Exclusive access okay. Indicates that either the read or write portion of an exclusive access has been successful. See *EXOKAY, exclusive access success* on page A3-55.

SLVERR Slave error. Used when the access has reached the slave successfully, but the slave wishes to return an error condition to the originating master. See *SLVERR, slave error* on page A3-55.

DECERR Decode error. Generated, typically by an interconnect component, to indicate that there is no slave at the transaction address. See *DECERR, decode error* on page A3-55.

RRESP[1:0] BRESP[1:0]	Response
0b00	OKAY
0b01	EXOKAY
0b10	SLVERR
0b11	DECERR

For a write transaction, a single response is signaled for the entire burst, and not for each data transfer within the burst.

In a read transaction, the slave can signal different responses for different transfers in a burst. For example, in a burst of 16 read transfers the slave might return an OKAY response for 15 of the transfers and a SLVERR response for one of the transfers.

A5: Multiple Transactions

BYU Electrical & Computer Engineering
IRA A. FULTON COLLEGE OF ENGINEERING

A5.1 AXI transaction identifiers

The AXI protocol includes AXI ID transaction identifiers. A master can use these to identify separate transactions that must be returned in order.

All transactions with a given AXI ID value must remain ordered, but there is no restriction on the ordering of transactions with different ID values.This means a single physical port can support out-of-order transactions by acting as a number of logical ports, each of which handles its transactions in order.

By using AXI IDs, a master can issue transactions without waiting for earlier transactions to complete. This can improve system performance, because it enables parallel processing of transactions.

Transaction channel	Transaction ID
Write address channel	**AWID**
Write data channel, AXI3 only	**WID**[a]
Write response channel	**BID**
Read address channel	**ARID**
Read data channel	**RID**

Transaction Ordering

A master can use the **AWID** and **ARID** transaction IDs to indicate its ordering requirements. The rules for the ordering of transactions are as follows:

- Transactions from different masters have no ordering restrictions. They can complete in any order.
- Transactions from the same master, but with different ID values, have no ordering restrictions. They can complete in any order.
- The data transfers for a sequence of read transactions with the same **ARID** value must be returned in the order in which the master issued the addresses, see *Read ordering*.
- The data transfers for a sequence of write transactions with the same **AWID** value must complete in the order in which the master issued the addresses, see *Normal write ordering* and *AXI3 write data interleaving* on page A5-79.
- There are no ordering restrictions between read and write transactions using a common value for **AWID** and **ARID**, see *Read and write interaction* on page A5-80.

Read Ordering

At a master interface, read data from transactions with the same **ARID** value must arrive in the order in which the master issued the addresses. Data from read transactions with different **ARID** values can arrive in any order. Read data of transactions with different **ARID** values can be interleaved.

A slave must return read data for a sequence of transactions with the same **ARID** value in the order in which it received the addresses. In a sequence of read transactions with different **ARID** values, the slave can return the read data in any order, regardless of the order in which the transactions arrived.

The slave must ensure that the **RID** value of any returned data matches the **ARID** value of the address to which it is responding.

Write Ordering

Unless a master knows that a slave supports write data interleaving, it must issue the data of write transactions in the same order in which it issues the transaction addresses. See *AXI3 write data interleaving* on page A5-79.

Read/Write Interaction

AXI has no ordering restrictions between read and write transactions. They can complete in any order, even if the **ARID** value of a read transaction is the same as the **AWID** value of a write transaction.

If a master requires a given relationship between a read transaction and a write transaction then it must ensure that the earlier transaction is complete before it issues the later transaction. A master can only consider the earlier transaction is complete when:

- for a read transaction, it receives the last of the read data
- for a write transaction, it receives the write response.

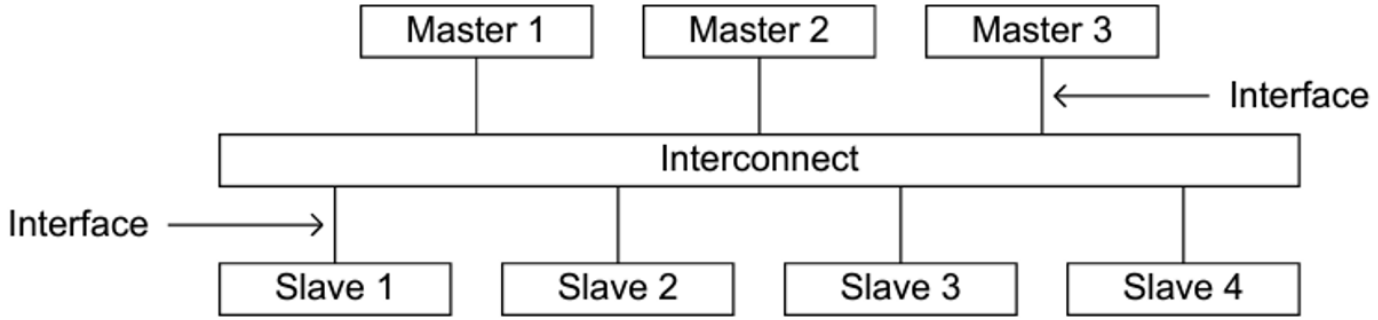
Interconnect + Transaction IDs

When a master is connected to an interconnect, the interconnect appends additional bits to the **ARID**, **AWID** and **WID** identifiers that are unique to that master port. This has two effects:

- masters do not have to know what ID values are used by other masters, because the interconnect makes the ID values used by each master unique, by appending the master number to the original identifier
- the ID identifier at a slave interface is wider than the ID identifier at a master interface.

For read data, the interconnect uses the additional bits of the **RID** identifier to determine which master port the read data is destined for. The interconnect removes these bits of the **RID** identifier before passing the **RID** value to the correct master port.

For write response, the interconnect uses the additional bits of the **BID** identifier to determine which master port the write response is destined for. The interconnect removes these bits of the **BID** identifier before passing the **BID** value to the correct master port.



A6: Ordering Model

BYU Electrical & Computer Engineering
IRA A. FULTON COLLEGE OF ENGINEERING

Definition of the ordering model

The AXI4 protocol supports an ordering model based on the use of the AXI ID transaction identifier.

The principles are that for transactions with the same ID:

- transactions to any single peripheral device, must arrive at the peripheral in the order in which they are issued, regardless of the addresses of the transactions
- memory transactions that use the same, or overlapping, addresses must arrive at the memory in the order in which they are issued.

Note:

Master Ordering

A master that issues multiple transactions in the same direction, read or write, with the same ID has the following guarantees about the ordering of these transactions:

- The order of response at the master to all transactions must be the same as the order of issue.
- For transactions to Device memory, the order of arrival at the slave must be the same as the order of issue.

Interconnect ordering

To meet the requirements of the ordering model, the interconnect must ensure that:

- The order of transactions in the same direction with the same ID to Device memory is preserved.
- The order of transactions in the same direction with the same ID to the same or overlapping addresses is preserved. See *Master ordering* on page A6-85 for the definition of overlapping addresses.
- The order of write responses with the same ID is preserved.
- The order of read responses with the same ID is preserved.
- Any manipulation of the AXI ID values associated with a transaction must ensure that the ordering requirements of the original ID values are maintained.
- Any component that gives a response to a transaction before the transaction reaches its final destination must ensure that the ordering requirements given in this section are maintained until the transaction reaches its final destination. See *Response before final destination* on page A6-88.

Slave Ordering

To meet the requirements of the ordering model, a slave must ensure that:

- Any write transaction for which it has issued a response must be observed by any subsequent write or read transaction, regardless of the transaction IDs.
- Any write transaction to Device memory must be observed by any subsequent write to Device memory with the same ID, even if a response has not yet been issued.
- Any write transaction to Normal memory must be observed by any subsequent write to the same or an overlapping address with the same ID, even if a response has not yet been given. This applies, also, to transactions to cacheable memory. That is, it applies to all valid write transactions for which **AWCACHE[3:1]** is not `0b000`.
- Responses to multiple write transactions with the same ID must be issued in the order in which the transactions arrived.
- Responses to multiple write transactions with different IDs can be issued in any order.
- Any read transaction for which it has issued a response must be observed by any subsequent write or read transaction, regardless of the transaction IDs.
- Any read transaction to Device memory must be observed by any subsequent read to Device memory with the same ID, even if a response has not yet been issued.
- Responses to multiple read transactions with the same ID must be issued in the order in which the transactions arrive.
- Responses to multiple read transactions with different IDs can be issued in any order.

B1:
AMBA AXI4-Lite

BYU Electrical & Computer Engineering
IRA A. FULTON COLLEGE OF ENGINEERING

Definition of AXI4-Lite

This section defines the functionality and signal requirements of AXI4-Lite components.

The key functionality of AXI4-Lite operation is:

- all transactions are of burst length 1
- all data accesses use the full width of the data bus
  - AXI4-Lite supports a data bus width of 32-bit or 64-bit.
- all accesses are Non-modifiable, Non-bufferable
- Exclusive accesses are not supported.

Table B1-1 AXI4-Lite interface signals

Global	Write address channel	Write data channel	Write response channel	Read address channel	Read data channel
ACLK	AWVALID	WVALID	BVALID	ARVALID	RVALID
ARESETn	AWREADY	WREADY	BREADY	ARREADY	RREADY
–	AWADDR	WDATA	BRESP	ARADDR	RDATA
–	AWPROT	WSTRB	–	ARPROT	RRESP