AXI

- ## What is AXI?

AXI (Advanced eXtensible Interface) is a part of the ARM Advanced Microcontroller Bus Architecture (**AMBA**) family, which is widely adopted in the industry. It is a protocol specification that defines the standard for the interface between various components in a system on a chip (SoC).

- ## Why AXI?

AXI (Advanced eXtensible Interface) is preferred in modern digital and embedded system designs for several reasons:

Burst Transfers

AXI supports burst data transfers, allowing multiple data transactions to be executed with a single address phase. This reduces the overhead of address transactions and improves data transfer efficiency.

Separate Address and Data Channels

AXI separates address/control signals and data signals into distinct channels. This separation allows for concurrent processing of address and data phases, leading to more efficient bus utilization and higher performance.

Multiple Outstanding Transactions

AXI can handle multiple transactions simultaneously. This capability, known as "outstanding transactions," allows the interface to maintain multiple active transactions, improving overall bus utilization and system performance.

Out-of-Order Transaction Completion

AXI allows transactions to be completed out of order. This flexibility is particularly useful in systems where different transactions may have varying priorities or where reordering can optimize performance.

Flexibility and Scalability

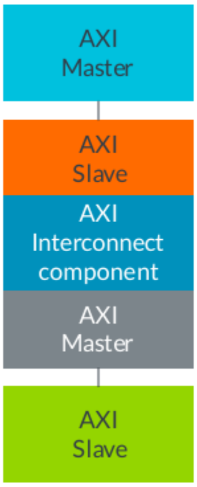
AXI is highly flexible and scalable, supporting a wide range of data widths, burst lengths, and transaction types. This flexibility makes it suitable for a variety of applications, from simple low-power devices to complex high-performance systems.

Support for Multiple Bus Masters and Slaves

AXI supports multiple bus masters and slaves, enabling complex system architectures where multiple processors, memory controllers, and peripherals can communicate efficiently.

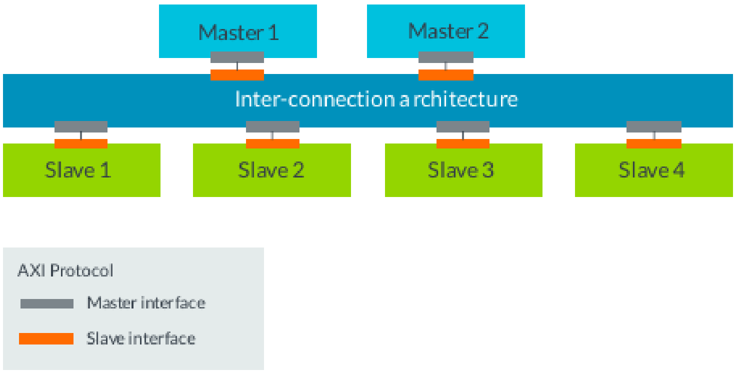
- ## AXI protocol overview

AXI is an interface specification that defines the interface of IP blocks, rather than the interconnect itself. The following diagram shows how AXI is used to interface an interconnect component:



In AX3 and AXI4, there are only two AXI interface types, master and slave. These interface types are symmetrical. All AXI connections are between master interfaces and slave interfaces.

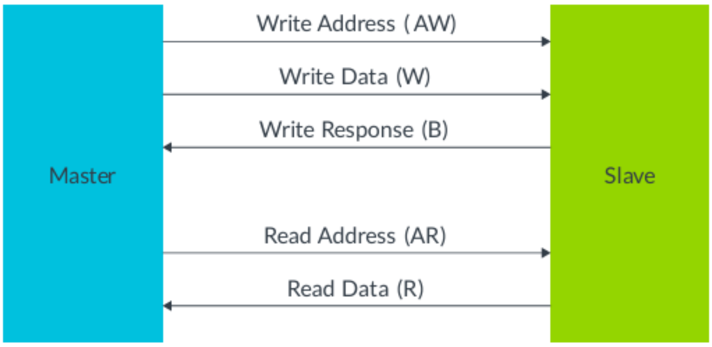
- ### AXI in a multi-master system



The AXI protocol defines the signals and timing of the point-to-point connections between masters and slaves. Where multiple masters and slaves are involved, an interconnect fabric is required. This interconnect fabric also implements slave and master interfaces, where the AXI protocol is implemented.

- ### AXI channels

The following diagram shows the five main channels that each AXI interface uses for communication:



Write operations use the following channels:

• The master sends an address on the Write Address (AW) channel and transfers data on the Write Data (W) channel to the slave.

• The slave writes the received data to the specified address. Once the slave has completed the write operation, it responds with a message to the master on the Write Response (B) channel.

Read operations use the following channels:

• The master sends the address it wants to read on the Read Address (AR) channel.

• The slave sends the data from the requested address to the master on the Read Data (R) channel. The slave can also return an error message on the Read Data (R) channel. An error occurs if, for example, the address is not valid, or the data is corrupted, or the access does not have the right security permission.

Note: Each channel is unidirectional, so a separate Write Response channel is needed to pass responses back to the master. However, there is no need for a Read Response channel, because a read response is passed as part of the Read Data channel.

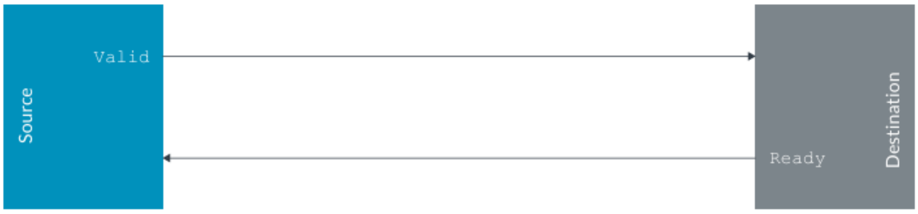
Each of these **five channels** contains several signals, and all these signals in each channel have the prefix as follows:

- **AW** for signals on the Write Address channel
- **AR** for signals on the Read Address channel
- **W** for signals on the Write Data channel
- **R** for signals on the Read Data channel
- **B** for signals on the Write Response channel

- ## Channel transfers and transactions

  - ### Channel handshake

    All of these channels share the same handshake mechanism that is based on the VALID and READY signals, as shown in the following diagram:



    - The VALID signal goes from the source to the destination, and READY goes from the destination to the source.
    - Whether the source or destination is a master or slave depends on which channel is being used. For example, the master is a source for the Read Address channel, but a destination for the Read Data channel.
    - The source uses the VALID signal to indicate when valid information is available. The VALID signal must remain asserted, meaning set to high, until the destination accepts the information. Signals that remain asserted in this way are called sticky signals.
    - The destination indicates when it can accept information using the READY signal. The READY signal goes from the channel destination to the channel source.
    - This mechanism is not an asynchronous handshake, and requires the rising edge of the clock for the handshake to complete.

  - ### Differences between transfers and transactions
    - A transfer is a single exchange of information, with one VALID and READY handshake.
    - A transaction is an entire burst of transfers, containing an address transfer, one or more data transfers, and, for write sequences, a response transfer.

  - ### Write transaction:

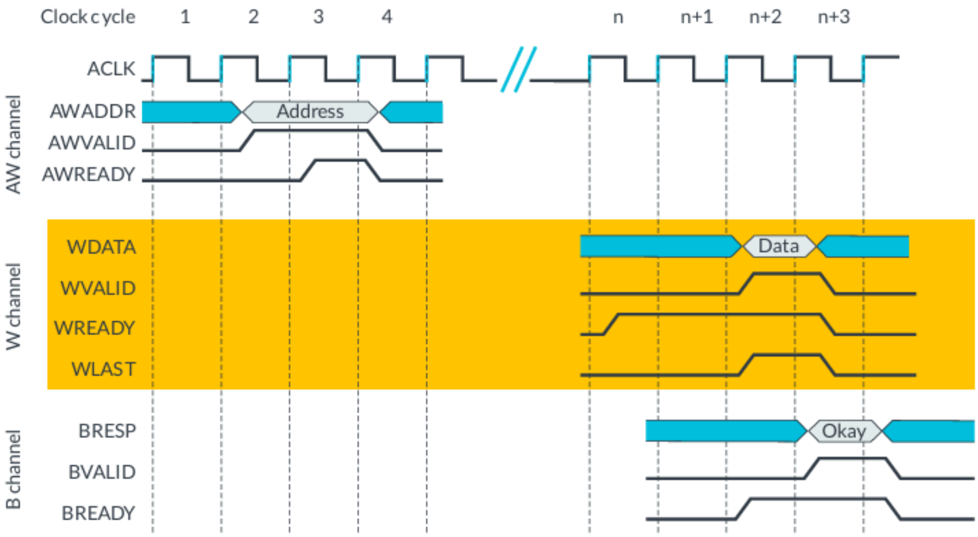
    #### Sequence of Events

    i. Address Phase:
       - Address and control information are presented on the AWADDR, AWVALID is asserted.
       - AWREADY is asserted when the address is accepted.

    ii. Data Phase:
       - Write data is presented on WDATA with WVALID asserted.
       - WREADY is asserted when data is accepted.
       - WLAST is asserted to indicate the last data transfer.

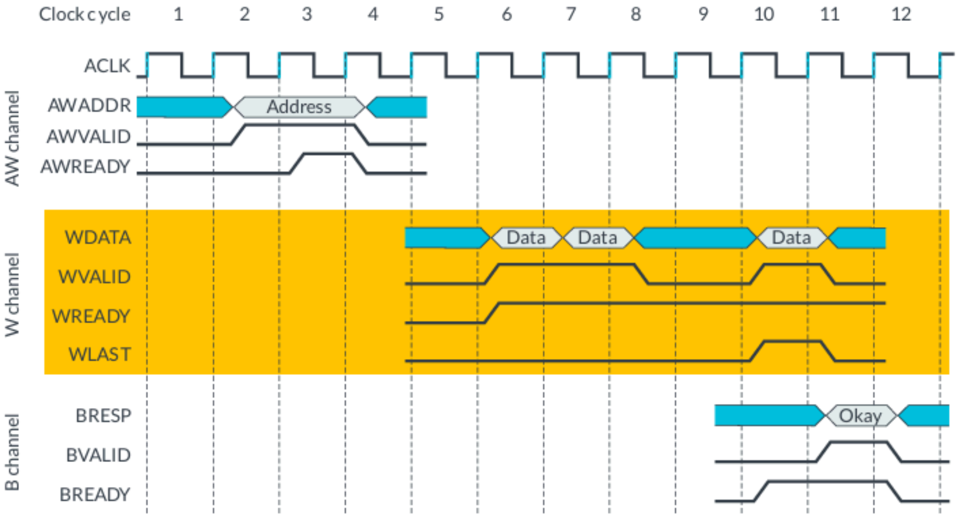
    iii. Response Phase:
       - BRESP indicates the result of the write transaction.
       - BVALID is asserted to indicate the response is valid.
       - BREADY is asserted when the response is accepted.

Single data Transfer



Multiple data transfer

While using burst transactions



Once all WDATA transfers are received, the slave gives a single BRESP value on the B channel. One single BRESP covers the entire burst. If the slave decides that any of the transfers contain an error, it must wait until the entire burst has completed before it informs the master that an error occurred.

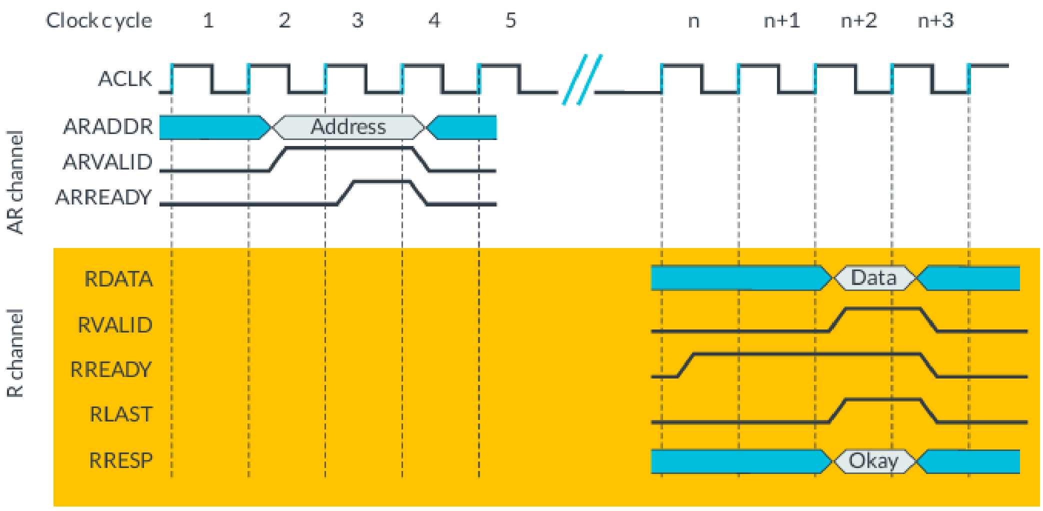
- ## Read transaction:

  ### Sequence of Events

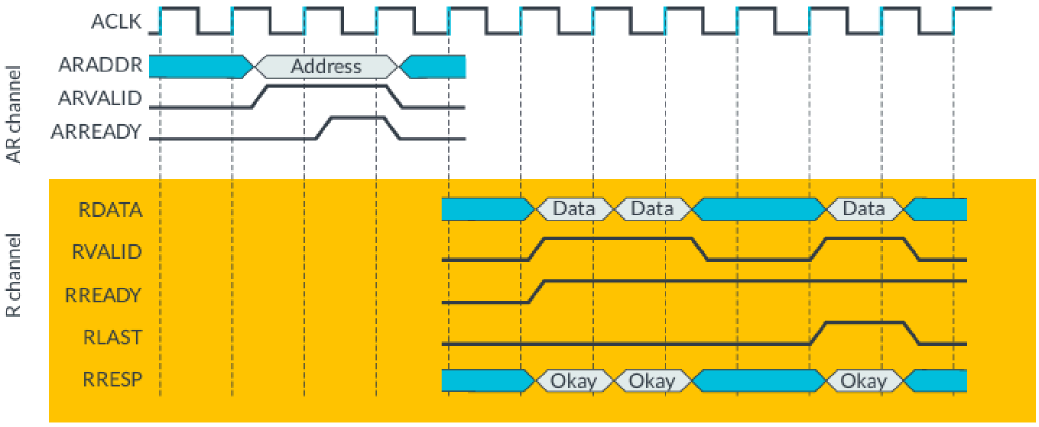
  1. Address Phase:
     - Address and control information are presented on the ARADDR, ARVALID is asserted.
     - ARREADY is asserted when the address is accepted.
  2. Data Phase:
     - RREADY is asserted to indicate that it is waiting for data.
     - Read data is presented on RDATA with RVALID asserted.
     - RLAST is asserted to indicate the last data transfer.

- RRESP indicates the result of the read transaction.

- **single data Transfer**



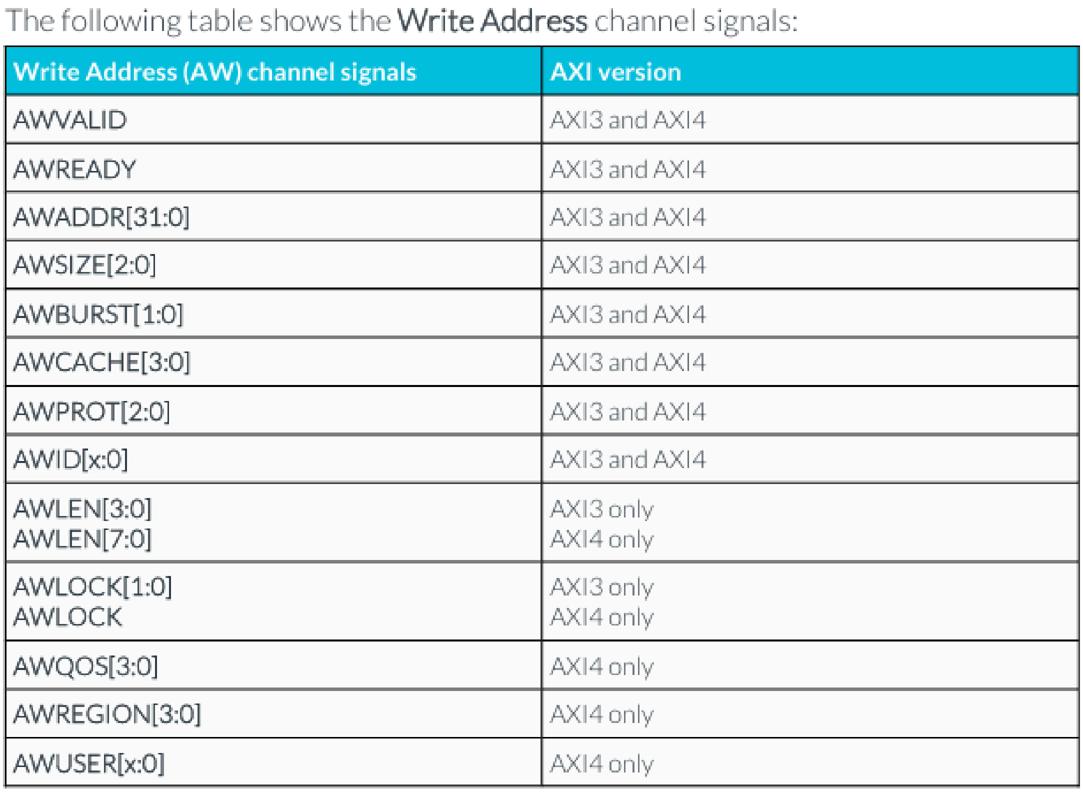
- **multiple data Transfer**



One difference between a read transaction and a write transaction is that for a read transaction there is an RRESP response for every transfer in the transaction. This is because, in the write transaction, the slave has to send the response as a separate transfer on the B channel. In the read transaction, the slave uses the same channel to send the data back to the master and to indicate the status of the read operation.

- ## Channel signals

  - ### Write Address (AW) Channel Signals



The following table shows the **Write Address** channel signals:

Write Address (AW) channel signals	AXI version
AWVALID	AXI3 and AXI4
AWREADY	AXI3 and AXI4
AWADDR[31:0]	AXI3 and AXI4
AWSIZE[2:0]	AXI3 and AXI4
AWBURST[1:0]	AXI3 and AXI4
AWCACHE[3:0]	AXI3 and AXI4
AWPROT[2:0]	AXI3 and AXI4
AWID[x:0]	AXI3 and AXI4
AWLEN[3:0] AWLEN[7:0]	AXI3 only AXI4 only
AWLOCK[1:0] AWLOCK	AXI3 only AXI4 only
AWQOS[3:0]	AXI4 only
AWREGION[3:0]	AXI4 only
AWUSER[x:0]	AXI4 only

AWVALID: Indicates that the write address and control information are valid and available on the address channel.

AWREADY: Indicates that the slave is ready to accept an address and associated control signals.

AWADDR[31:0]: Address of the first transfer in a write burst transaction.

AWSIZE[2:0]: Indicates the size of the data transfer in the burst (number of bytes per beat).

AWBURST[1:0]: Indicates the burst type, defining how the address for each transfer within the burst is calculated.

AWCACHE[3:0]: Indicates the bufferable, cacheable, write-through, and allocate attributes of the transaction.

AWPROT[2:0]: Protection type signals indicating the privilege and security level of the transaction.

AWID[x:0]: Write address ID tag, used to identify the write transaction.

AWLEN[7:0]: Indicates the number of data transfers in the burst.

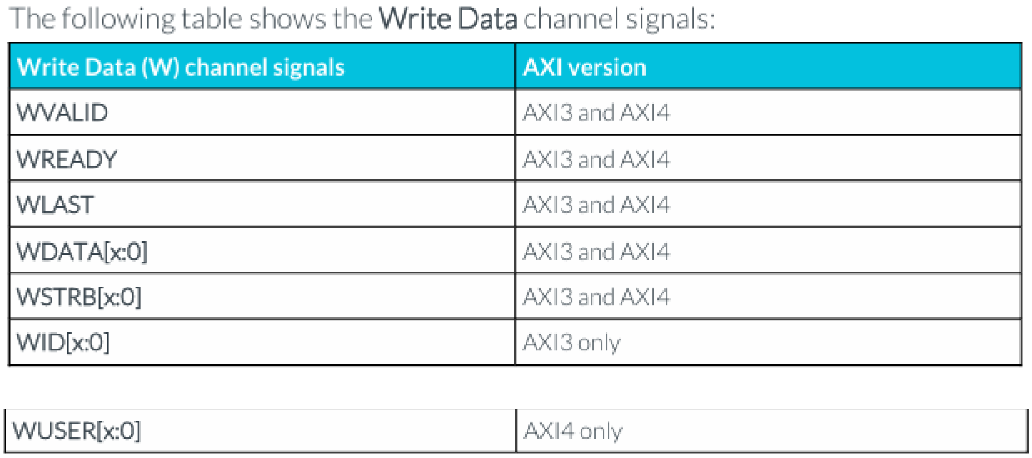
AWLOCK[1:0]: Indicates the lock type to provide exclusive access to a resource.

AWQOS[3:0]: Indicates the quality of service level for the write transaction.

AWREGION[3:0]: Indicates the region identifier, used for routing transactions within a system.

AWUSER[x:0]: User-defined sideband signals for the write address channel.

- ### Write Data (W) Channel Signals



The following table shows the Write Data channel signals:

Write Data (W) channel signals	AXI version
WVALID	AXI3 and AXI4
WREADY	AXI3 and AXI4
WLAST	AXI3 and AXI4
WDATA[x:0]	AXI3 and AXI4
WSTRB[x:0]	AXI3 and AXI4
WID[x:0]	AXI3 only
WUSER[x:0]	AXI4 only

WVALID: Indicates that valid write data and control information are available on the WDATA, WSTRB, and WLAST signals.

WREADY: Indicates that the slave can accept the write data.

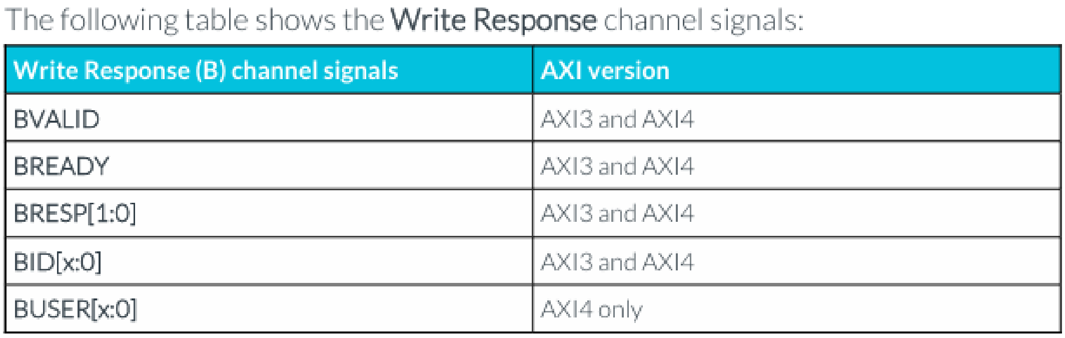
WLAST: Indicates the last transfer in a write burst.

WDATA[x:0]: Write data bus carrying the data to be written.

WSTRB[x:0]:Write strobes indicating which byte lanes are valid in the write data bus.

WUSER[x:0]: User-defined sideband signals for the write data channel.

- ### Write Response (B) Channel Signals



The following table shows the Write Response channel signals:

Write Response (B) channel signals	AXI version
BVALID	AXI3 and AXI4
BREADY	AXI3 and AXI4
BRESP[1:0]	AXI3 and AXI4
BID[x:0]	AXI3 and AXI4
BUSER[x:0]	AXI4 only

BVALID: Indicates that a valid write response is available on the BRESP, BID, and BUSER signals.

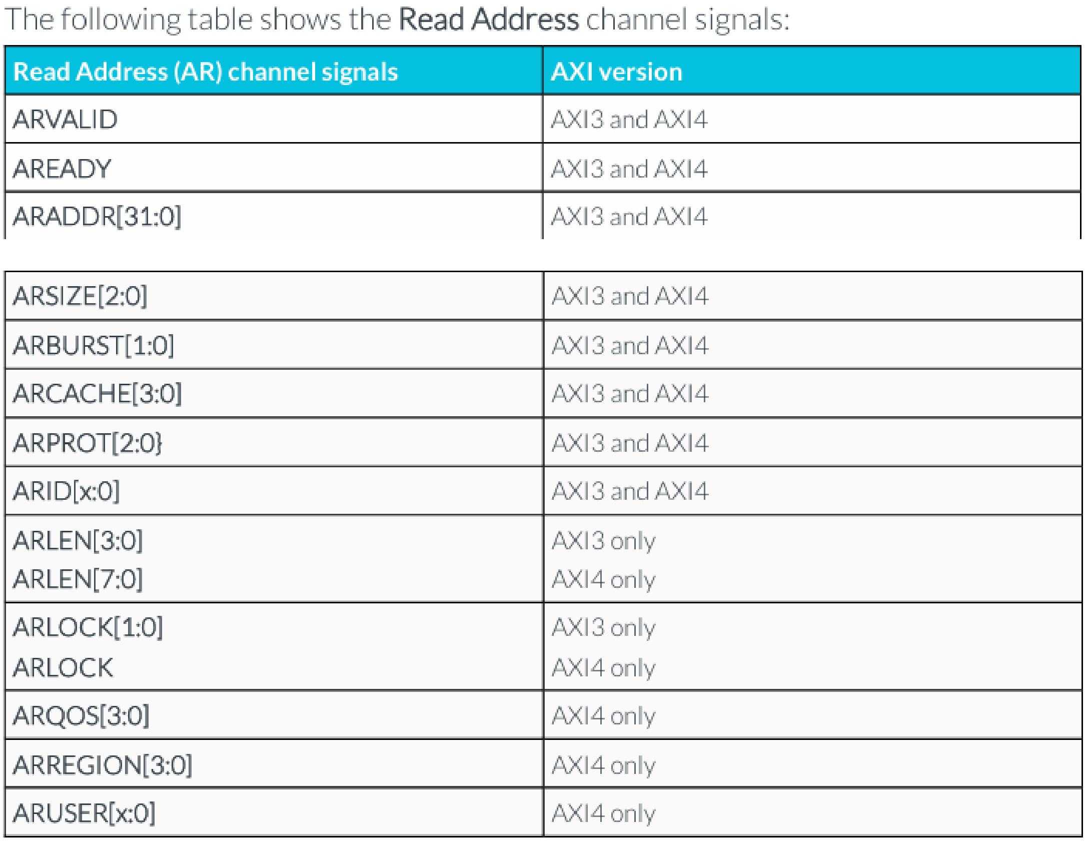
BREADY: Indicates that the master can accept the write response.

BRESP[1:0]: Write response indicating the status of the write transaction.

BID[x:0]: Write response ID tag, used to identify the write response.

BUSER[x:0]: User-defined sideband signals for the write response channel.

- ### Read Address (AR) Channel Signals



The following table shows the Read Address channel signals:

Read Address (AR) channel signals	AXI version
ARVALID	AXI3 and AXI4
AREADY	AXI3 and AXI4
ARADDR[31:0]	AXI3 and AXI4
ARSIZE[2:0]	AXI3 and AXI4
ARBURST[1:0]	AXI3 and AXI4
ARCACHE[3:0]	AXI3 and AXI4
ARPROT[2:0}	AXI3 and AXI4
ARID[x:0]	AXI3 and AXI4
ARLEN[3:0] ARLEN[7:0]	AXI3 only AXI4 only
ARLOCK[1:0] ARLOCK	AXI3 only AXI4 only
ARQOS[3:0]	AXI4 only
ARREGION[3:0]	AXI4 only
ARUSER[x:0]	AXI4 only

ARVALID: Indicates that the read address and control information are valid and available on the address channel.

AREADY: Indicates that the slave is ready to accept an address and associated control signals.

ARADDR[31:0]: Address of the first transfer in a read burst transaction.

ARSIZE[2:0]: Indicates the size of the data transfer in the burst (number of bytes per beat).

ARBURST[1:0]: Indicates the burst type, defining how the address for each transfer within the burst is calculated.

ARCACHE[3:0]: Indicates the bufferable, cacheable, write-through, and allocate attributes of the transaction.

ARPROT[2:0]: Protection type signals indicating the privilege and security level of the transaction.

ARID[x:0]: Read address ID tag, used to identify the read transaction.

ARLEN[7:0]: Indicates the number of data transfers in the burst.

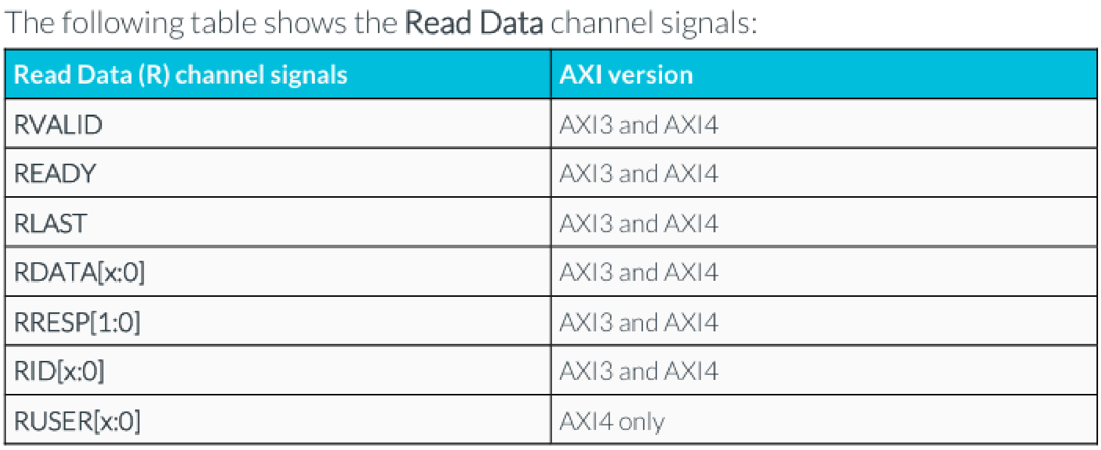
ARLOCK[1:0]: Indicates the lock type to provide exclusive access to a resource.

ARQOS[3:0]: Indicates the quality of service level for the read transaction.

ARREGION[3:0]: Indicates the region identifier, used for routing transactions within a system.

ARUSER[x:0]: User-defined sideband signals for the read address channel.

- ### Read Data (R) Channel Signals



The following table shows the Read Data channel signals:

Read Data (R) channel signals	AXI version
RVALID	AXI3 and AXI4
READY	AXI3 and AXI4
RLAST	AXI3 and AXI4
RDATA[x:0]	AXI3 and AXI4
RRESP[1:0]	AXI3 and AXI4
RID[x:0]	AXI3 and AXI4
RUSER[x:0]	AXI4 only

RVALID: Indicates that valid read data is available on the data bus.

RREADY: Indicates that the master can accept the read data.

RLAST: Indicates the last transfer in a read burst.

RDATA[x:0]: Read data bus carrying the data being transferred.

RRESP[1:0]: Read response indicating the status of the read transaction.

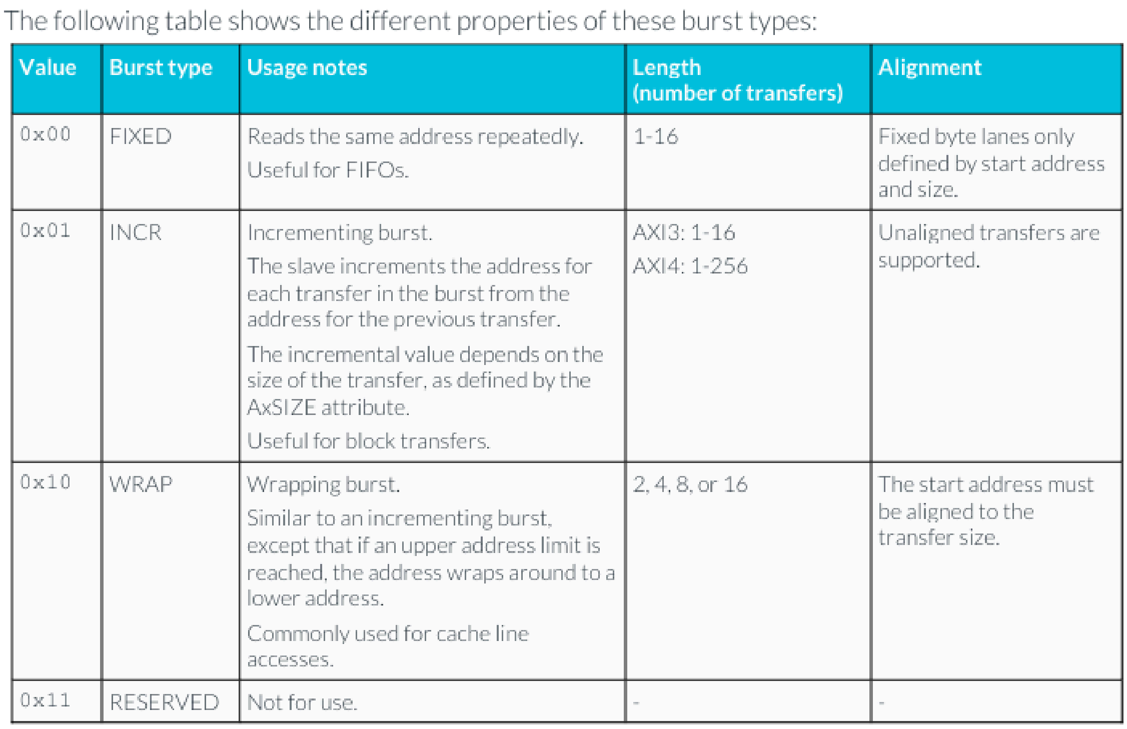
RID[x:0]: Read data ID tag, used to identify the read data transfers.

RUSER[x:0]: User-defined sideband signals for the read data channel.

- ## Detailed Usage of Some Signals

Data size, length, and burst type

- **AxLEN** describes the length of the transaction in the number of transfers. For AXI4, **AxLEN[7:0]** has 8 bits, which specifies a range of 1-256 data transfers in a transaction.
- **AxSize[2:0]** describes the maximum number of bytes to transfer in each data transfer. Three bits of encoding indicate 1, 2, 4, 8, 16, 32, 64, or 128 bytes per transfer.
- **AxBURST[1:0]** describes the burst type of the transaction: fixed, incrementing, or wrapping.



The following table shows the different properties of these burst types:

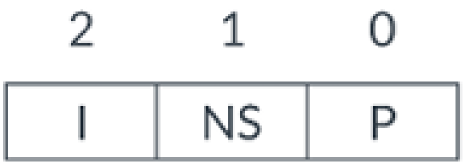
Value	Burst type	Usage notes	Length (number of transfers)	Alignment
0x00	FIXED	Reads the same address repeatedly. Useful for FIFOs.	1-16	Fixed byte lanes only defined by start address and size.
0x01	INCR	Incrementing burst. The slave increments the address for each transfer in the burst from the address for the previous transfer. The incremental value depends on the size of the transfer, as defined by the AxSIZE attribute. Useful for block transfers.	AXI3: 1-16 AXI4: 1-256	Unaligned transfers are supported.
0x10	WRAP	Wrapping burst. Similar to an incrementing burst, except that if an upper address limit is reached, the address wraps around to a lower address. Commonly used for cache line accesses.	2, 4, 8, or 16	The start address must be aligned to the transfer size.
0x11	RESERVED	Not for use.	-	-

Protection level support

The AXI (Advanced eXtensible Interface) protocol includes signals, AWPROT and ARPROT, which help manage access permissions for transactions, enhancing security within the system. These signals are particularly useful in environments with multiple levels of security, such as Arm Trust Zone, which features Secure and Non-secure states.

AxPROT Bit Allocation

AxPROT signals (AWPROT for write transactions and ARPROT for read transactions) include three bits that define access protection attributes:



i. AxPROT[0] (P) - Privilege Level
   - 1: Privileged Access
   - 0: Unprivileged Access

- Although some processors support more complex privilege levels, AXI distinguishes only between privileged and unprivileged access.

ii. AxPROT[1] (NS) - Security State

- 1: Non-secure Transaction
- 0: Secure Transaction

iii. AxPROT[2] (I) - Instruction/Data Access

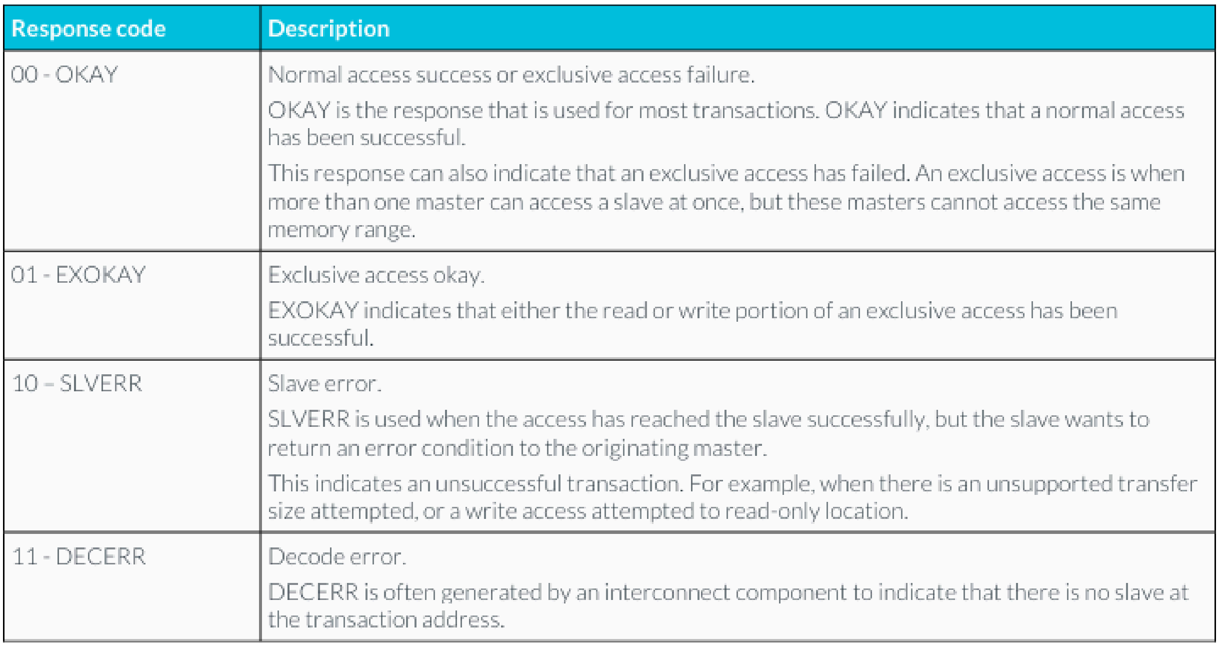
- 1: Instruction Access
- 0: Data Access
- This bit is considered a hint and may not always be accurate, especially in mixed transactions. It is recommended that masters set this bit to zero (data access) unless the access is definitively known to be instruction access.

Response signaling

For read transactions, the response information from the slave is signaled on the read data channel using **RRESP**.

For write transactions, the response information is signaled on the write response channel using **BRESP.**

RRESP and **BRESP** are both composed of two bits, and the encoding of these signals can transfer four responses, as shown in the following table:



Response code	Description
00 - OKAY	Normal access success or exclusive access failure. OKAY is the response that is used for most transactions. OKAY indicates that a normal access has been successful. This response can also indicate that an exclusive access has failed. An exclusive access is when more than one master can access a slave at once, but these masters cannot access the same memory range.
01 - EXOKAY	Exclusive access okay. EXOKAY indicates that either the read or write portion of an exclusive access has been successful.
10 – SLVERR	Slave error. SLVERR is used when the access has reached the slave successfully, but the slave wants to return an error condition to the originating master. This indicates an unsuccessful transaction. For example, when there is an unsupported transfer size attempted, or a write access attempted to read-only location.
11 - DECERR	Decode error. DECERR is often generated by an interconnect component to indicate that there is no slave at the transaction address.

Write Data Strobes

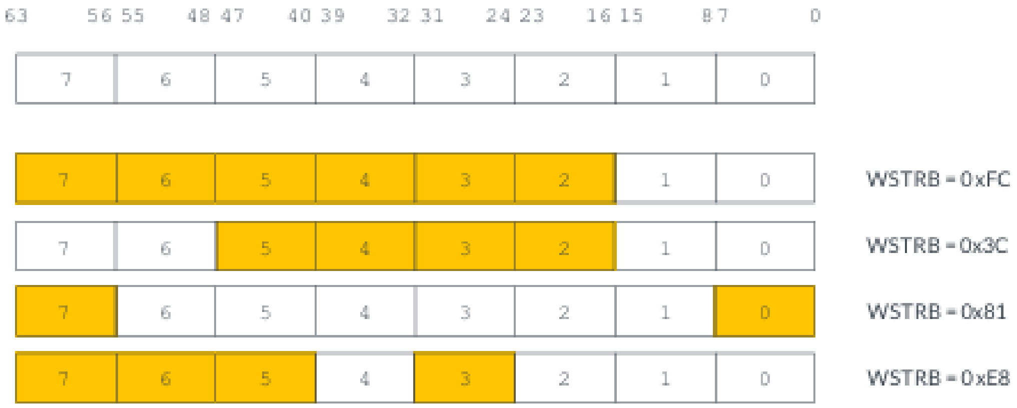
Overview

- Function: Indicates which bytes of the data bus are valid for writing.
- Usefulness: Optimizes data transfers, especially for cache accesses and sparse data arrays.

WSTRB Signal

- Structure: One strobe bit per byte on the data bus.
- Purpose: Ensures strobes are set to '1' for valid data bytes only.

Examples for 64-bit Data Bus (WDATA)



- WSTRB = 0xFC
    - Valid bytes: 7, 6, 5, 4, 3, 2
- WSTRB = 0x3C
    - Valid bytes: 5, 4, 3, 2
- WSTRB = 0x81
    - Valid bytes: 7, 0
- WSTRB = 0xE8
    - Valid bytes: 7, 6, 5, 3

Key Points

- Efficiency: Enhances handling of sparse data and unaligned addresses.
- Master's Role: Sets WSTRB correctly for valid data bytes.
- No Read Equivalent: Masters handle unwanted bytes in read transactions.

Atomic access with AxLOCK Signal

Exclusive Access

- Purpose: Ensures that a master can perform atomic operations without interference, maintaining data integrity.
- Efficiency: More efficient than older locked methods, as multiple masters can access a slave simultaneously without blocking each other.
- Usage: Ideal for operations that require exclusive access to shared resources, such as semaphores.

AxLOCK Signal

- Role: Indicates when a transaction is atomic.
- AXI4 Specifics:
    - Exclusive Access: Only exclusive access is supported, improving efficiency.
    - Signal Values:
        - 0b0: Normal access
        - 0b1: Exclusive access

Quality of Service (QoS)

The AXI4 protocol introduces extra signals to support the quality of service (QoS). Quality of service allows you to prioritize transactions allowing you to improve system performance, by ensuring that more important transactions are dealt with higher priority.

There are two quality of service signals:

- AWQOS is sent on the Write Address channel for each write transaction.
- ARQOS is sent on the Read Address channel for each read transaction.

Both signals are 4 bits wide, where the value 0x0 indicates the lowest priority, and the value 0xF indicates the highest priority.

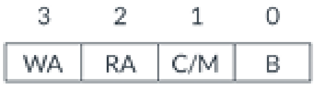
The default system-level implementation of quality of service is that any component with a choice of more than one transaction processes the transaction with the higher QoS value first.

- ### Region signaling
  - Region signaling is a new optional feature in AXI4.
  - When you use region identifiers, it means that a single physical interface on a slave can provide multiple logical interfaces. Each logical interface can have a different location in the system address map.
  - When the region identifier is used, the slave does not have to support the address decode between the different logical interfaces.
  - Region signaling uses two 4-bit region identifiers, AWREGION and ARREGION. These region identifiers can uniquely identify up to 16 different regions.

- ### Cache Support

  Overview

  AXI supports different cache policies for modern SoC systems with multiple cache levels (e.g., L2, L3 caches). The AxCACHE signal helps manage these policies efficiently**.**



3	2	1	0
WA	RA	C/M	B

AxCACHE Bit Allocations

- Bits 3 to 0:
  - WA (Write Allocate):
    - 0: No write allocation.
    - 1: Allocate cache line on write.
  - RA (Read Allocate):
    - 0: No read allocation.
    - 1: Allocate cache line on read.
  - C/M (Cacheable/Modifiable):
    - 0: Non-cacheable (or non-modifiable in AXI4).
    - 1: Cacheable or modifiable.
  - B (Bufferable):
    - 0: Not bufferable.
    - 1: Bufferable.

Attributes and Usage

- Bufferable Bit (B):

    - Function: Allows the interconnect or component to delay transactions for better performance.
- Cacheable Bit (C/M):
    - Function: Indicates if a transaction's attributes at the destination can differ from the original.
    - AXI3: Cacheable.
    - AXI4: Modifiable.
- Read Allocate Bit (RA):
    - Function: Indicates if the allocation of a read transaction is recommended.
- Write Allocate Bit (WA):
    - Function: Indicates if the allocation of a write transaction is recommended.

Key Points

- Write Allocate (WA) and Read Allocate (RA):
    - Recommended for optimizing cache performance.
    - Prevent immediate passing of transactions without cache lookup.
- Cacheable/Modifiable Bit (C/M):
    - Allows transaction attributes to vary at the destination.
    - Important for merging writes and optimizing read operations.
- Restrictions:
    - If the Cacheable/Modifiable bit is not set, Read Allocate and Write Allocate bits cannot be set.

- **User signals**
    - The AXI4 interface signal set has the option to include a set of user-defined signals, called the User signals.
    - User signals can be used on each channel to transfer extra custom control information between master and slave components.
    - These signals are optional and do not have to be supported on all channels. If they are used, then the width of the User signals is defined by the implementation and can be different on each channel.