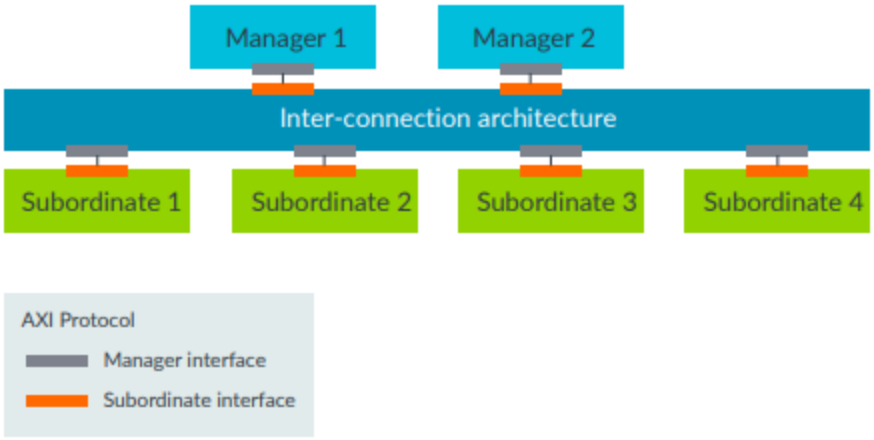
Figure 3-2: Multi master high-level



An Arm processor is an example of a manager, and a simple example of a subordinate is a memory controller.

The AXI protocol defines the signals and timing of the point-to-point connections between manager and subordinates.

Note

The AXI protocol is a point-to-point specification, not a bus specification. Therefore, it describes only the signals and timing between interfaces.

The previous diagram shows that each AXI manager interface is connected to a single AXI subordinate interface. Where multiple managers and subordinates are involved, an interconnect fabric is required. This interconnect fabric also implements subordinate and manager interfaces, where the AXI protocol is implemented.

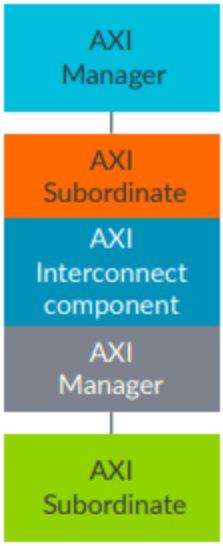
The following diagram shows that the interconnect is a complex element that requires its own AXI manager and subordinate interfaces to communicate with external function blocks:

3. 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:

Figure 3-1: AXI interface



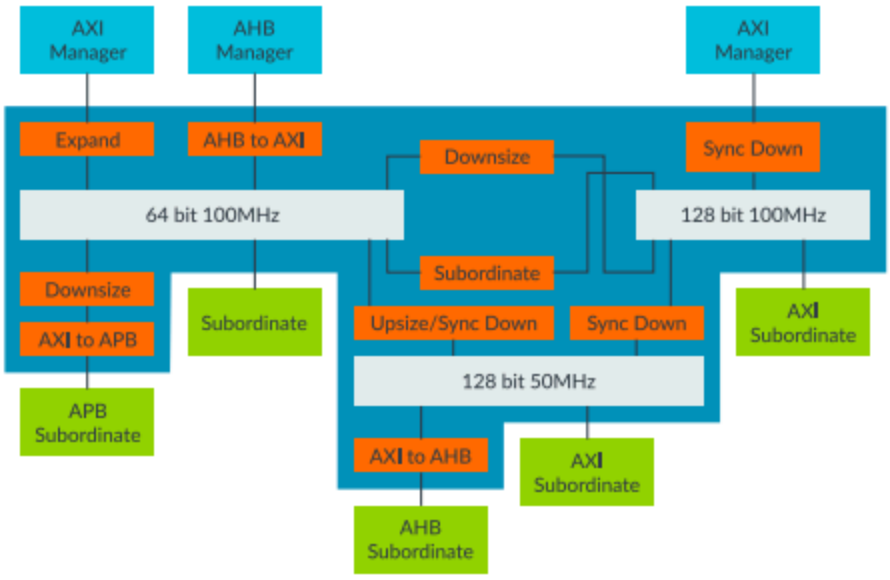
There are only two AXI interface types, manager and subordinate. These interface types are symmetrical. All AXI connections are between manager interfaces and subordinate interfaces.

AXI interconnect interfaces contain the same signals, which makes integration of different IP relatively simple. The previous diagram shows how AXI connections join manager and subordinate interfaces. The direct connection gives maximum bandwidth between the manager and subordinate components with no extra logic. And with AXI, there is only a single protocol to validate.

AXI in a multi-manager system

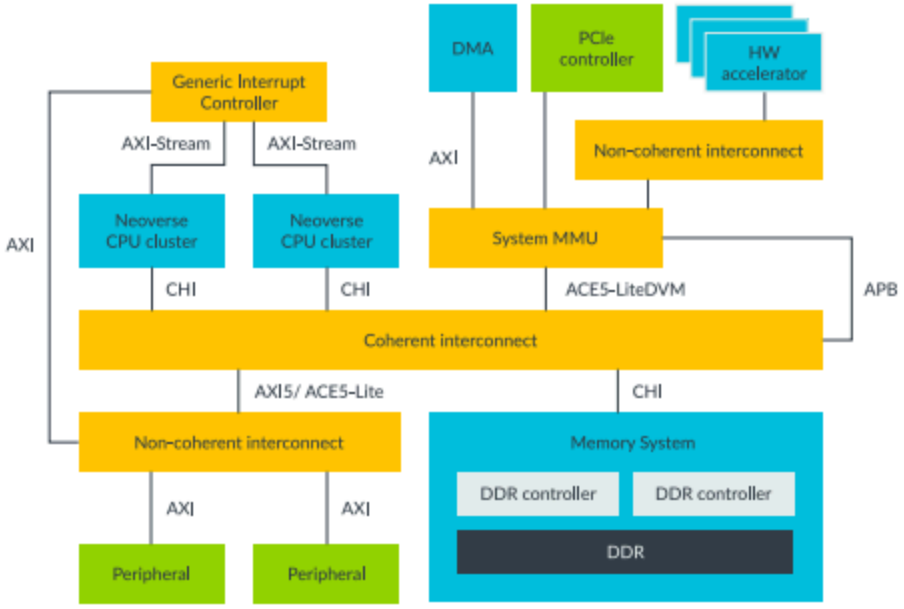
The following diagram shows a simplified example of an SoC system, which is composed of managers, subordinates, and the interconnect that links them all:

Figure 3-3: Multi master interconnect



The following diagram shows an example of an SoC with various processors and function blocks:

Figure 3-4: System diagram

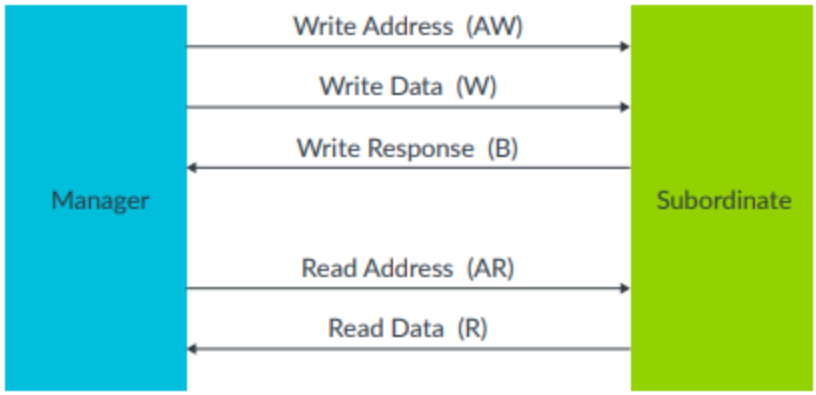


The previous diagram shows all the connections where AXI is used. You can see that AXI3 and AXI4 are used within the same SoC, which is common practice. In such cases, the interconnect performs the protocol conversion between the different AXI interfaces.

AXI channels

The AXI specification describes a point-to-point protocol between two interfaces: a manager and a subordinate. The following diagram shows the five main channels that each AXI interface uses for communication:

Figure 3-5: Axi channels



Write operations use the following channels:

- The manager sends an address on the Write Address (AW) channel and transfers data on the Write Data (W) channel to the subordinate.
- The subordinate writes the received data to the specified address. Once the subordinate has completed the write operation, it responds with a message to the manager on the Write Response (B) channel.

Read operations use the following channels:

- The manager sends the address it wants to read on the Read Address (AR) channel.
- The subordinate sends the data from the requested address to the manager on the Read Data (R) channel. The subordinate 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 manager. However, there is no need for a Read Response channel, because a read response is passed as part of the Read Data channel.

Using separate address and data channels for read and write transfers helps to maximize the bandwidth of the interface. There is no timing relationship between the groups of read and write channels. This means that a read sequence can happen at the same time as a write sequence.

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


Note

B stands for buffered, because the response from the subordinate happens after all writes have completed.

Main AXI features

The AXI protocol has several key features that are designed to improve bandwidth and latency of data transfers and transactions, as you can see here:

- Independent read and write channels: AXI supports two different sets of channels, one for write operations, and one for read operations. Having two independent sets of channel helps to improve the bandwidth performances of the interfaces. This is because read and write operations can happen at the same time.
- Multiple outstanding addresses: AXI allows for multiple outstanding addresses. This means that a manager can issue transactions without waiting for earlier transactions to complete. This can improve system performance because it enables parallel processing of transactions.
- No strict timing relationship between address and data operations: With AXI, there is no strict timing relationship between the address and data operations. This means that, for example, a manager could issue a write address on the Write Address channel, but there is no time requirement for when the manager has to provide the corresponding data to write on the Write Data channel.
- Support for unaligned data transfers: For any burst that is made up of data transfers wider than one byte, the first bytes accessed can be unaligned with the natural address boundary. For example, a 32-bit data packet that starts at a byte address of `0x1002` is not aligned to the natural 32-bit address boundary.
- Out-of-order transaction completion: Out-of-order transaction completion is possible with AXI. The AXI protocol includes transaction identifiers, and there is no restriction on the completion of transactions with different ID values. This means that a single physical port can support out-of-order transactions by acting as several logical ports, each of which handles its transactions in order.
- Burst transactions based on start address: AXI managers only issue the starting address for the first transfer. For any following transfers, the subordinate will calculate the next transfer address based on the burst type.