ARM AXI INTERFACE

Slaven Vidaković

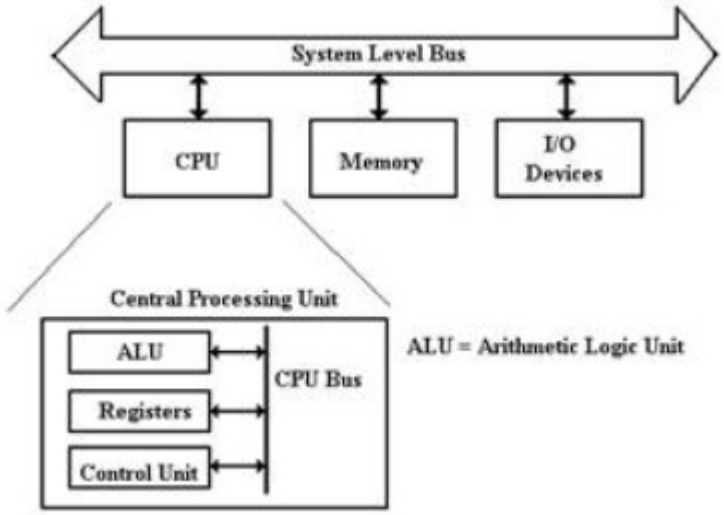
INTRODUCTION AND FEATURES

BUSSES

-What is a bus?

-Why do we need to use busses?

-Where does AXI come in?



A typical bus.[1]

ARM AXI INTERFACE

What is it?

-Advanced extensible interface

-A high performance communications protocol that is suitable for use with bus systems.[2]

ARM AXI INTERFACE

-Suitable for high frequency designs.

Recommended frequencies are up to 200 MHz.[3]

-Based on managers and subordinates (masters and slaves)

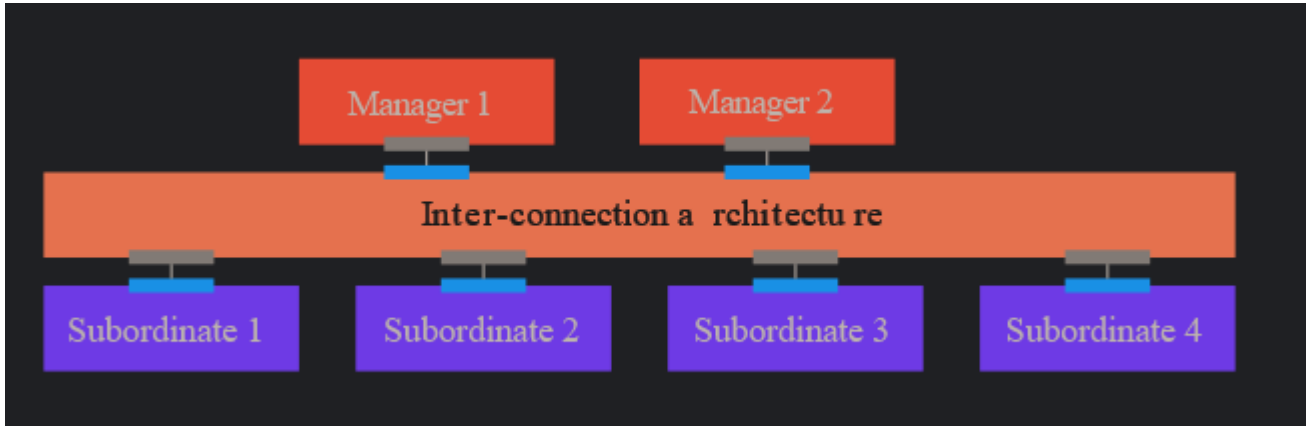
Managers initiate all communications[2]

ARM AXI INTERFACE

-What do the managers communicate with?

Either with a memory device which will have many addresses, or a peripheral device which only has one address, also called an AXI stream.[4]

AXI – MULTI MANAGER SYSTEMS



An example of a bus with multiple AXI managers[5]

AXI - BURSTS

-Communications are based on bursts.

Subsequent data transfers which use the first memory address to compute all other addresses.[6]

Each of these transfers is called a ‘beat’.

CHANNELS

CHANNELS WITHIN AN AXI INTERFACE

-The system is based on five independent channels to maintain its high performance

AR– Read address channel

R - Read channel

AW- Write address channel

W – Write channel

B - Write response channel[2]

READING AND WRITING

-AW and AR channels can be collectively referred to as AX and they carry addresses and additional control signals

W and R channels carry actual data

B channel carries write responses from subordinate to manager[2]

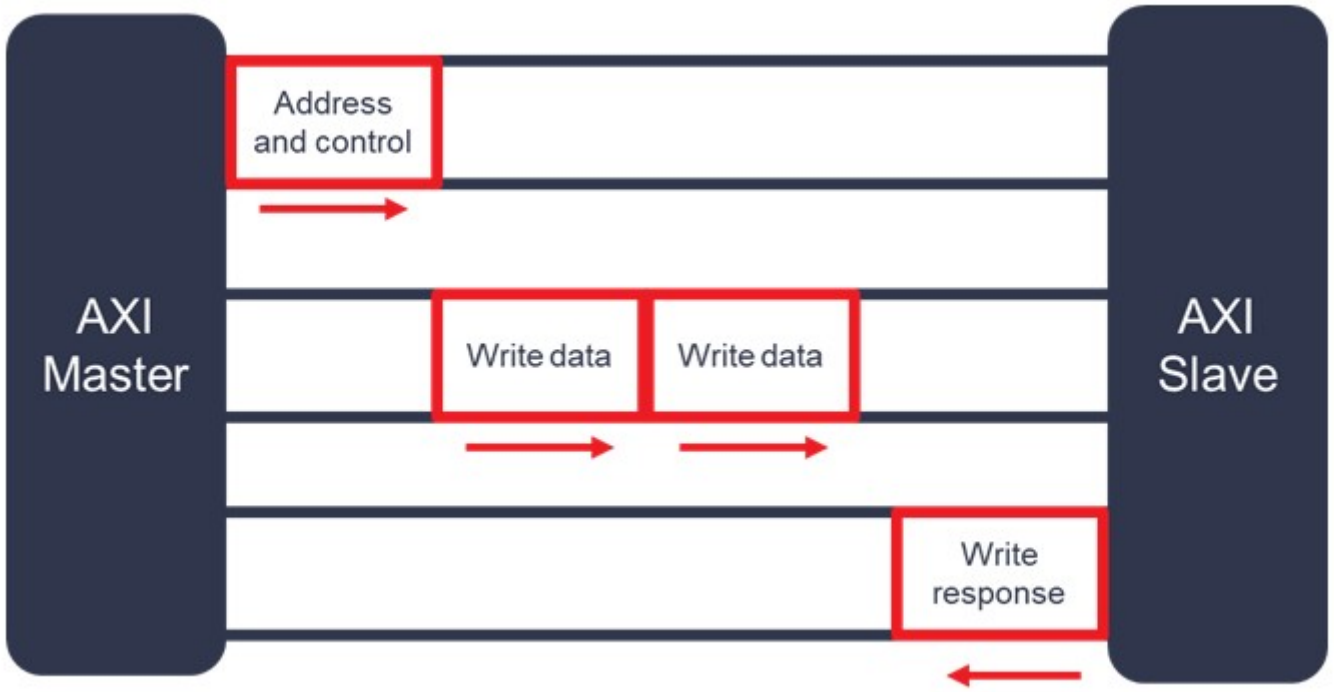
READING AND WRITING

-What is reading? What is writing?

Reading: data goes from subordinate to manager

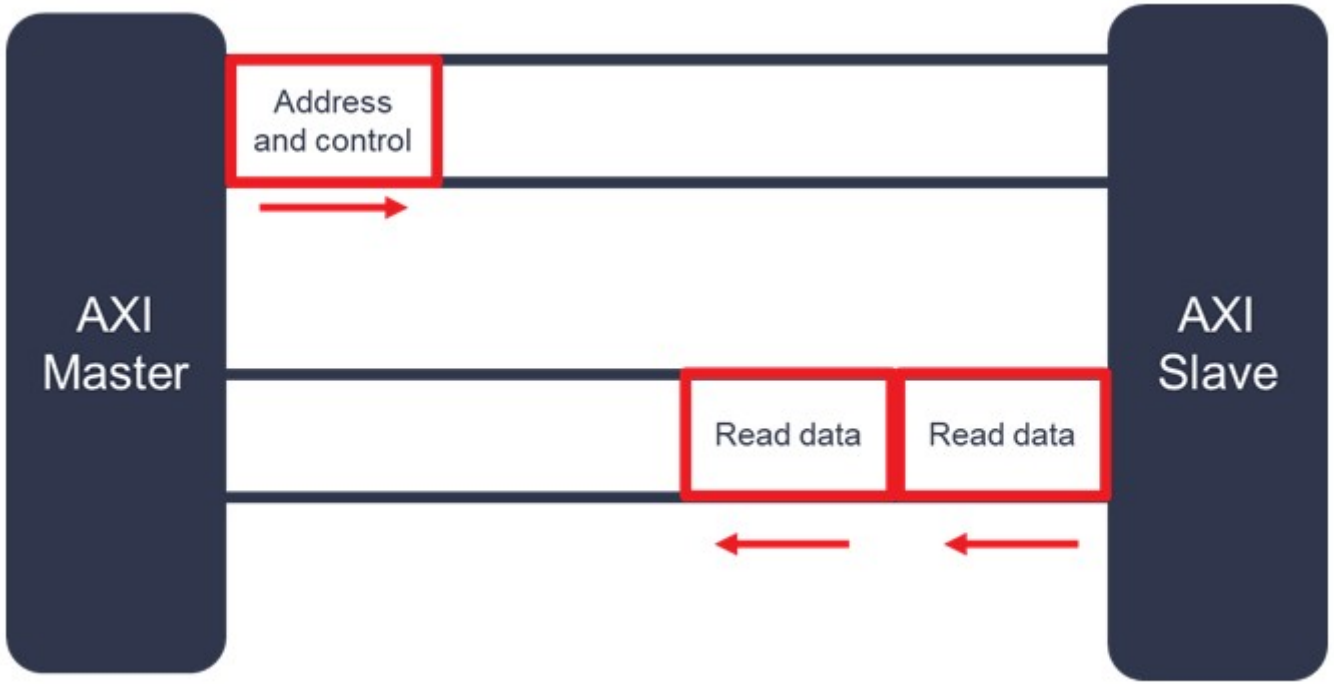
Writing: data goes from manager to subordinate[7]

READING AND WRITING



An example of data write[8]

READING AND WRITING



An example of data read[8]

TRANSACTION COMPLETION

-When are transactions considered completed?

Two different completion criteria for writing and for reading transactions.

Based on write and read responses.

HANDSHAKE PROCESS

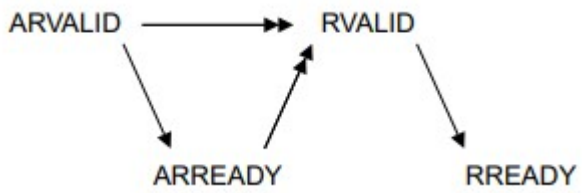
HANDSHAKE PROCESS

-Based on control signals VALID and READY

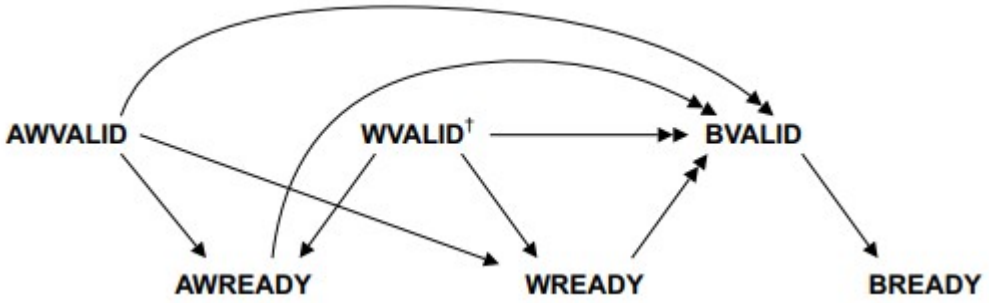
VALID – The data on a given channel is available

READY – The device is ready to receive data[2]

HANDSHAKE SIGNALING IN ORDER



Order of signaling for read operations[2]



Order of signaling for write operations[2]

HANDSHAKE PROCESS

-The transfer is defined as occurring on the next rising edge of the clock after both VALID and READY are HIGH.

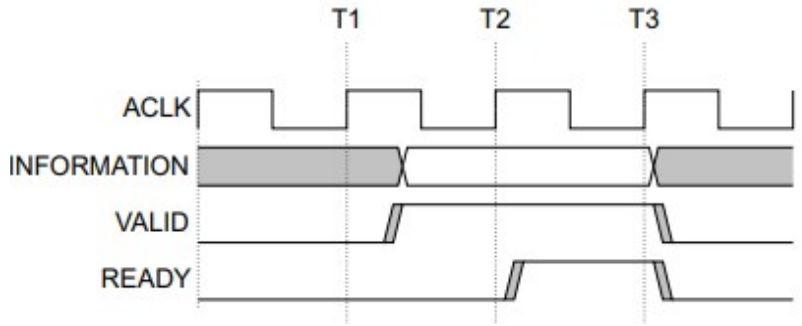
-They then go LOW once again.[2]

HANDSHAKE PROCESS

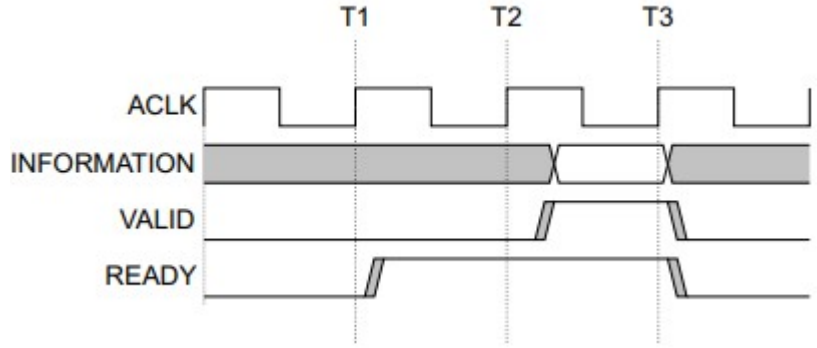
The source can not wait for READY to assert VALID, however, the recipient can wait for VALID in order to assert READY.

VALID, once asserted, can not de-assert before the transaction, unlike READY.[2]

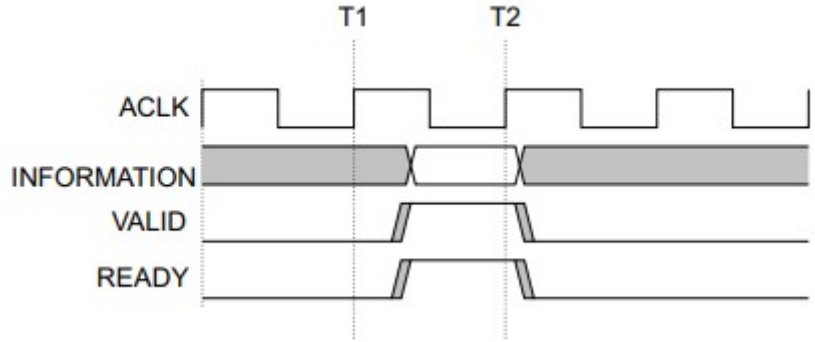
HANDSHAKE PROCESS - EXAMPLES



An example where VALID was asserted before READY[2]



An example where READY was asserted before VALID[2]



An example where VALID and READY were asserted at the same time[2]

OTHER SIGNALS

TRANSACTION IDS

-ID lines in each channel designate the ID of the transaction taking place.

Set by the manager, they pair address and data lines as well as write responses to appropriate write transactions

AXI – ORDERING REQUIREMENT

-Transactions on the same master with the same ID will be carried out in order

-Transactions on the same master with different ID’s can be carried out out of order[2]

AXI – ORDERING REQUIREMENT

-This allows for increase in performance, since one physical manager can act as many logical ones, thus avoiding stalling by a slow peripheral, while also guaranteeing transaction ordering when needed.[2]

BURST LENGTH AND SIZE

-AxLEN defines the number of beats within the burst.

AxLEN[7:0] + 1.

-AxSIZE defines the size of each beat.

$2^{AxSIZE[2:0]}$.

-The last beat of each transaction is indicated by xLAST.[2]

BURST TYPES

-The AXI protocol defines 3 burst types, declared in the AxBURST line.[2]

- FIXED BURST (0)
- INCR BURST (1)
- WRAP BURST (2)

WRITE STROBES

-In cases when the byte count of the transaction is less than the byte count of the data lines, the manager will use strobe signaling.[2]

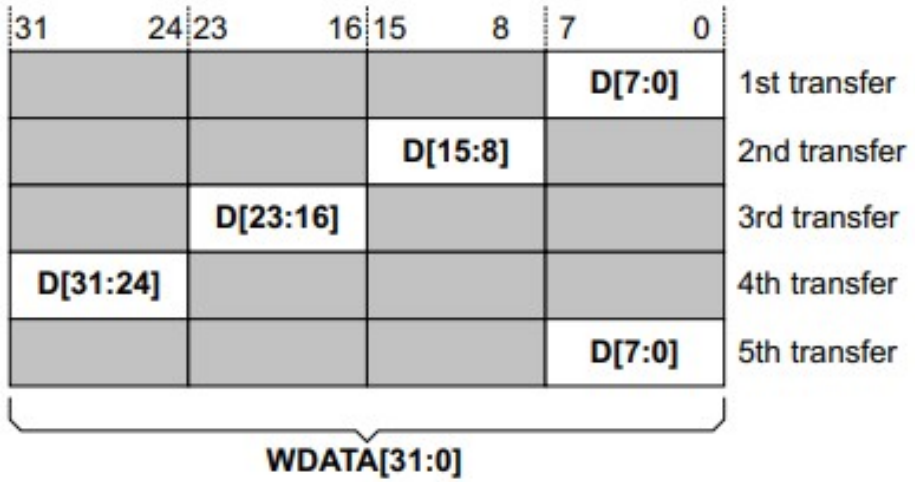
-What is strobe signaling?

WRITE STROBES

Strobe bits indicate in which byte lanes was the data written by the manager.

Each byte is associated with one bit in the WSTRB control signal.[2]

EXAMPLE OF WRITE STROBES



An example of five 1-byte transfers on a 4-byte data channel, white cells indicate active byte strobes[2]

WRITE AND READ RESPONSES

-Each transaction is finished with a response. This response can have the following values[2]

- OKAY (0)
- EXOKAY (1)
- SLVERR (2)
- DECERR (3)

MEMORY ATTRIBUTE SIGNALS

-A set of flags contained in the AxCACHE control signal.

Modifiable flag: Indicates if the transaction can be changed by other devices in the interconnect. This includes addresses and burst lengths, sizes and types. [2]

MEMORY ATTRIBUTE SIGNALS

Bufferable: Indicates if the write response or read data have to be given from the final write/read destination, or some intermediate point/a write transaction headed for the destination respectively.[2]

MEMORY ATTRIBUTE SIGNALS

Allocate: A set of two signals that indicate whether or not the data in the transaction must be looked up in a cache attached to the subordinate, since it could have been allocated previously, and whether or not it should be allocated.[2]

PROTECTION SIGNALS

-A set of flags contained in the AxPROT signal indicating the access permission a manager has.

- Privileged access: The subordinate defines what a privileged access can do to the data. [2]

PROTECTION SIGNALS

- Secure access: The subordinate can have its memory space split into two parts. Non secure access can not read/write secure data

- Instruction or data: Used to differentiate accesses to instructions and data[2]

EXCLUSIVE ACCESS

What is it?

A mechanism for the manager to request exclusive access to a subordinate using the AxLOCK signal, indicated by a write request with this signal set to HIGH.[2]

EXCLUSIVE ACCESS

Any subsequent write with the same ID will return EXOKAY if the location wasn’t accessed by a different ID, or OKAY otherwise.

It ends by reading the same location, with the same ID and with the lock signal at LOW.[2]

QUALITY OF SERVICE

-A four bit control signal that does not have a preset use.

It is indicated by the AxQOS signal and the recommendation is to use it as a priority indication – higher value for higher priority (0 if the device makes no use of it). [2]

REGION SIGNALS

-The region information is signaled via the AxREGION signals.

It allows one physical subordinate interface to act as multiple logical interfaces (up to 16).[2]

USER SIGNALS

-User signaling makes use of the AxUSER signals.

No use case is indicated for these signals – they are meant to be defined by the devices in the network and used for any need.[2]

MINIMUM SIGNALING

-Read/Write only devices have no use for the opposite channels and as such they are not required.

-Apart from that, some other signals (such as handshake signals) are necessary.[2]

AXI 4 LITE

-A simpler version of the protocol without many of the optional signals and with restricted values of some control signals.[2]

EXAMPLE OF USING AXI

-The LED peripheral created in the SIP course is a write only AXI device.

It has an address defined by the Vivado software, the burst length and size are both 1, etc.

In the end it is a very simple device, but a good way to understand the AXI protocol.

EXAMPLE OF USING AXI

-Another example is the Intel Stratix memory controller.

It’s a high performance controller for high bandwidth memory developed by Intel.

It uses AXI4 to communicate with other systems.

Many of the control signals are either unused or preset to only one value or a subset of values, easing development.[9]

QUESTIONS?

References

[1] E. L. Bosworth Design and Architecture of Digital Computers: An Introduction. Columbus, GA, USA: Self published, 2011.

[2] ARM IHI. “AMBA AXI and ACE specifications”. https://documentation-service.arm.com/static/602a9df190ee6824a1e02b98?token= (accessed Nov. 07, 2023).

[3] AMD. “AMD AXI documentation”. https://docs.xilinx.com/r/en-US/pg067-axi-chip2chip/Clock-Frequencies (accessed Nov. 07, 2023).

[4] AMD. “How AXI4-Stream Works - 2023.2 English”. https://docs.xilinx.com/r/en-US/ug1399-vitis-hls/How-AXI4-Stream-Works (accessed Nov. 07, 2023).

[5] ARM Developer. “Learn the architecture - An introduction to AMBA AXI”.

https://developer.arm.com/documentation/102202/0300/AXI-protocol-overview (accessed Nov. 07, 2023).

[5] H. Gilbert. “Memory and Burst”. https://pclt.sites.yale.edu/memory-and-burst (accessed Nov. 07, 2023).

[6] P. R. Schaumont, A Practical Introduction to Hardware/Software codesign. Blacksburg VA, USA: Springer, 2012.

[7] AMD. “Introduction to AXI”. https://support.xilinx.com/s/article/1053914?language=en_US (accessed Nov. 07, 2023).

[8] Intel. “High Bandwidth Memory (HBM2) Interface Intel® FPGA IP User Guide”.

https://www.intel.com/content/www/us/en/docs/programmable/683189/21-3-19-6-1/about-the-high-bandwidth-memory-hbm2.html (accessed Nov. 07, 2023).