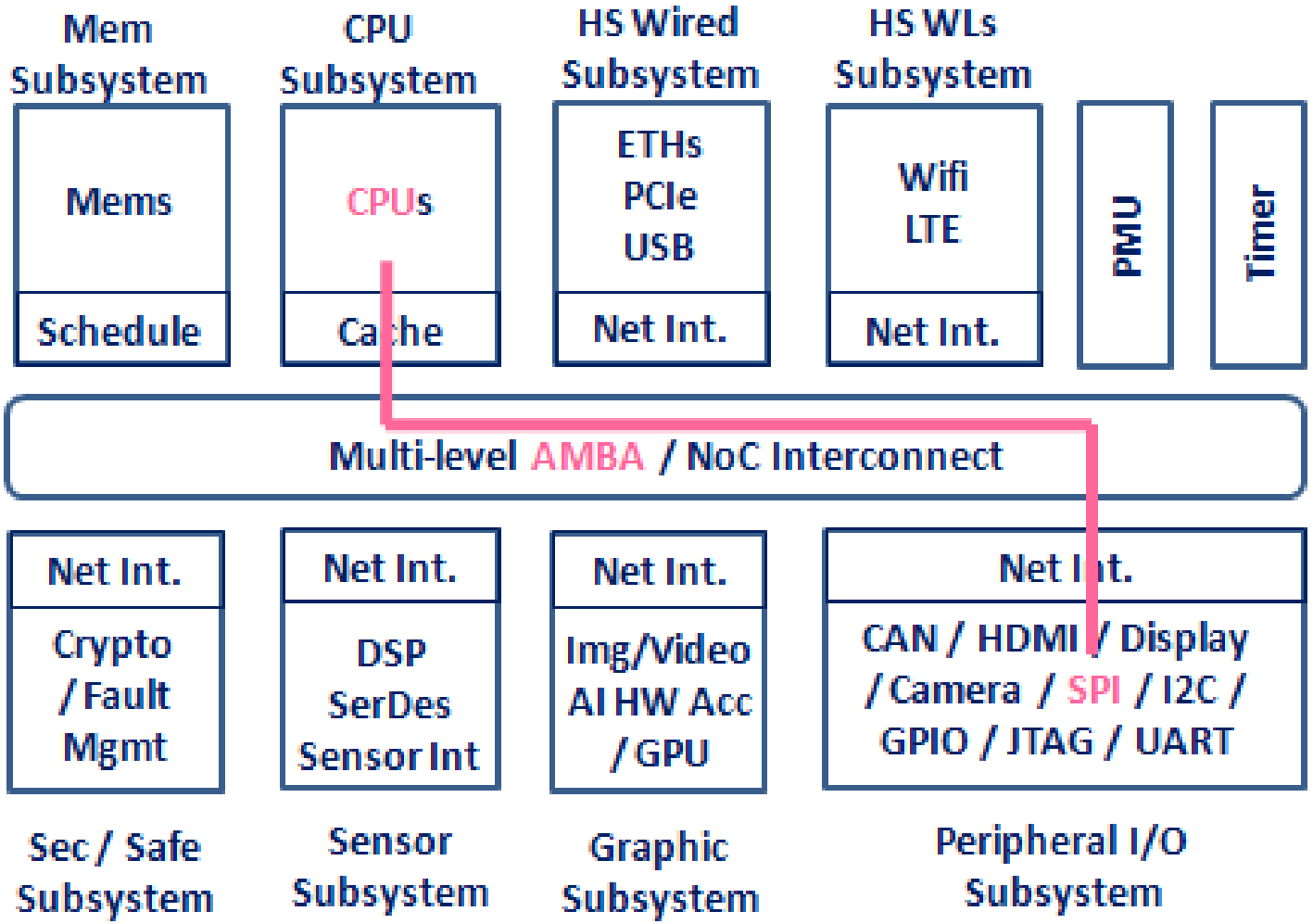
SPI Bus (Serial Peripheral Interface Bus)

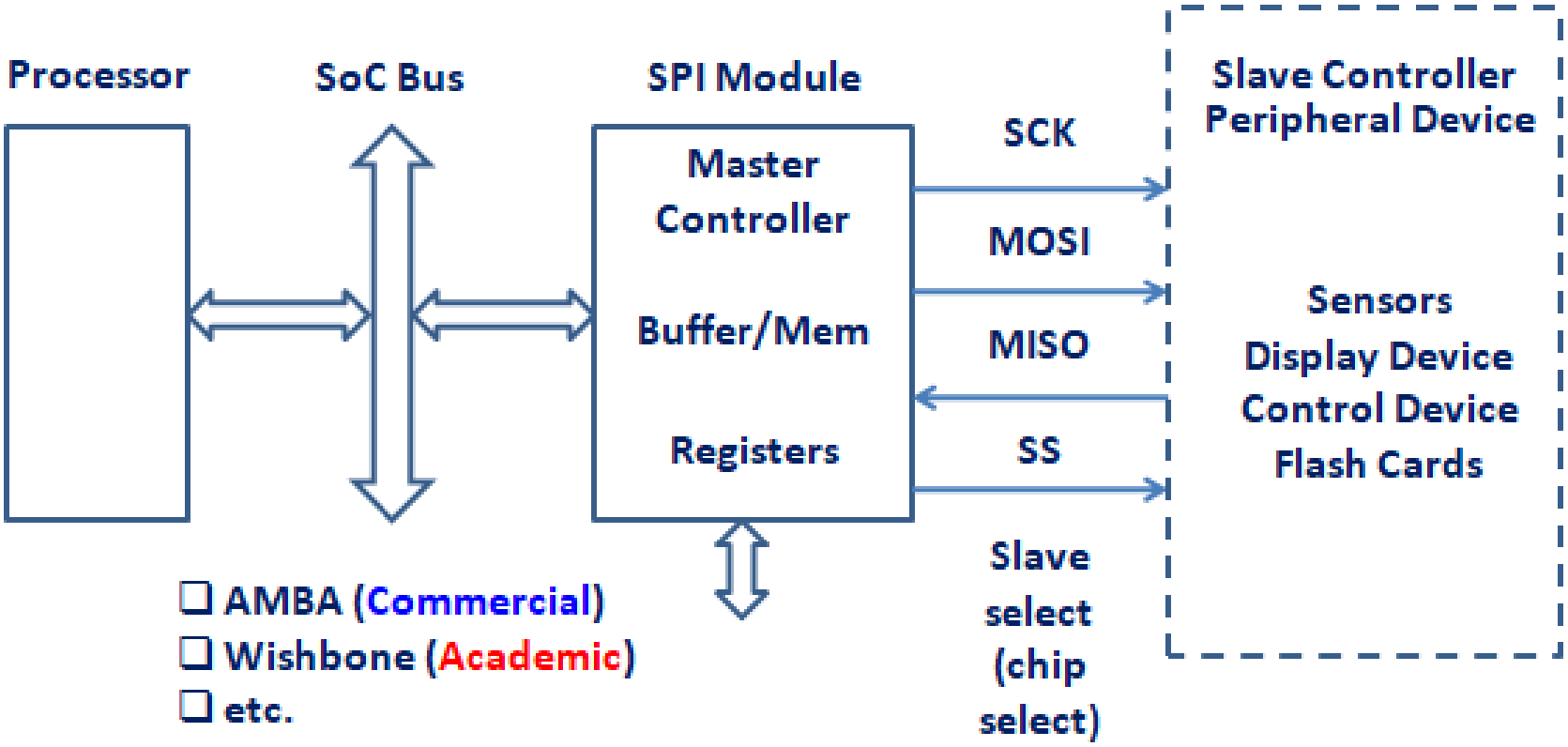
SoC - SPI

Tuan Nguyen-viet

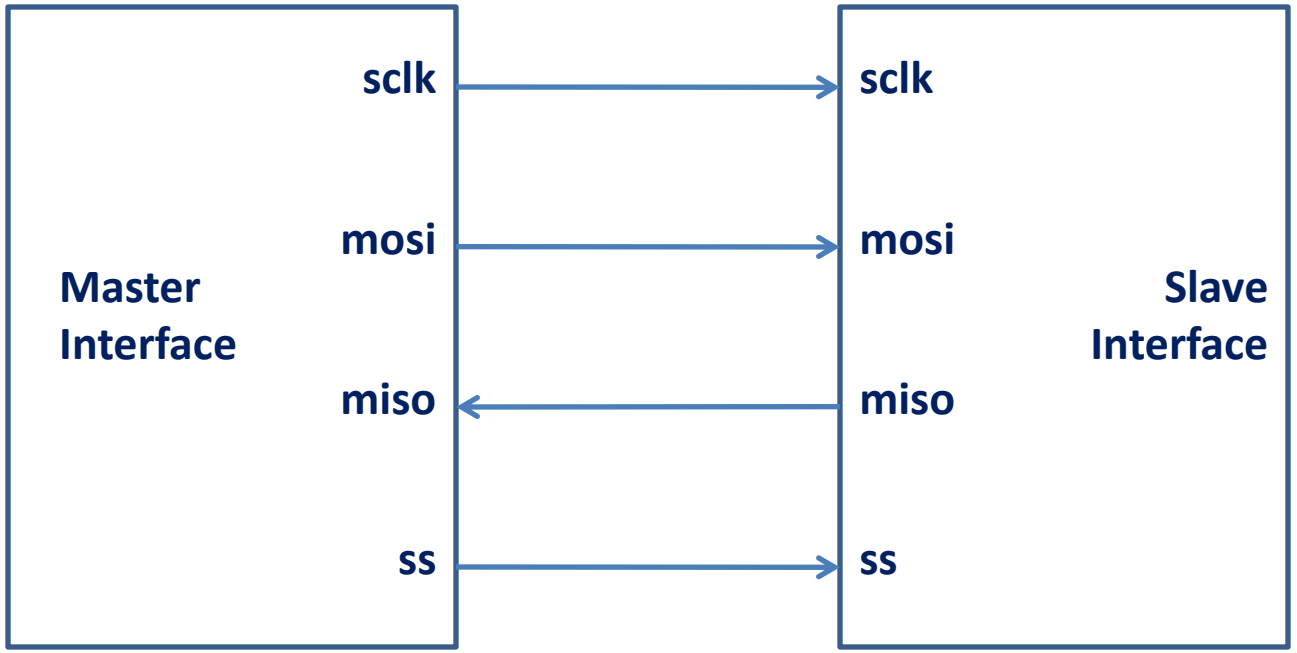
Processor-SPI Interaction



Processor-SPI Interaction

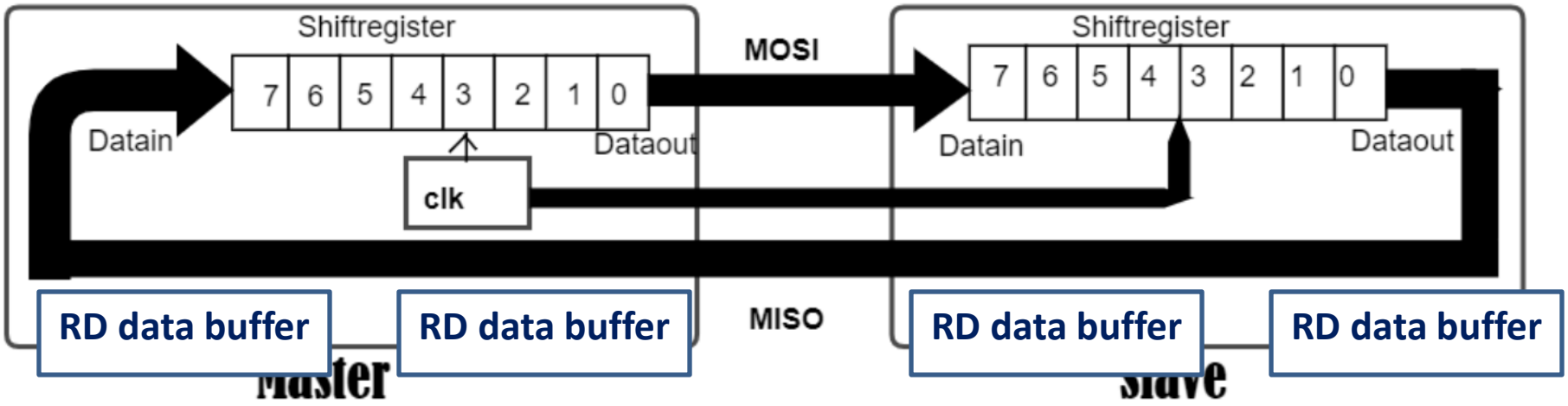


SPI Master / Slave

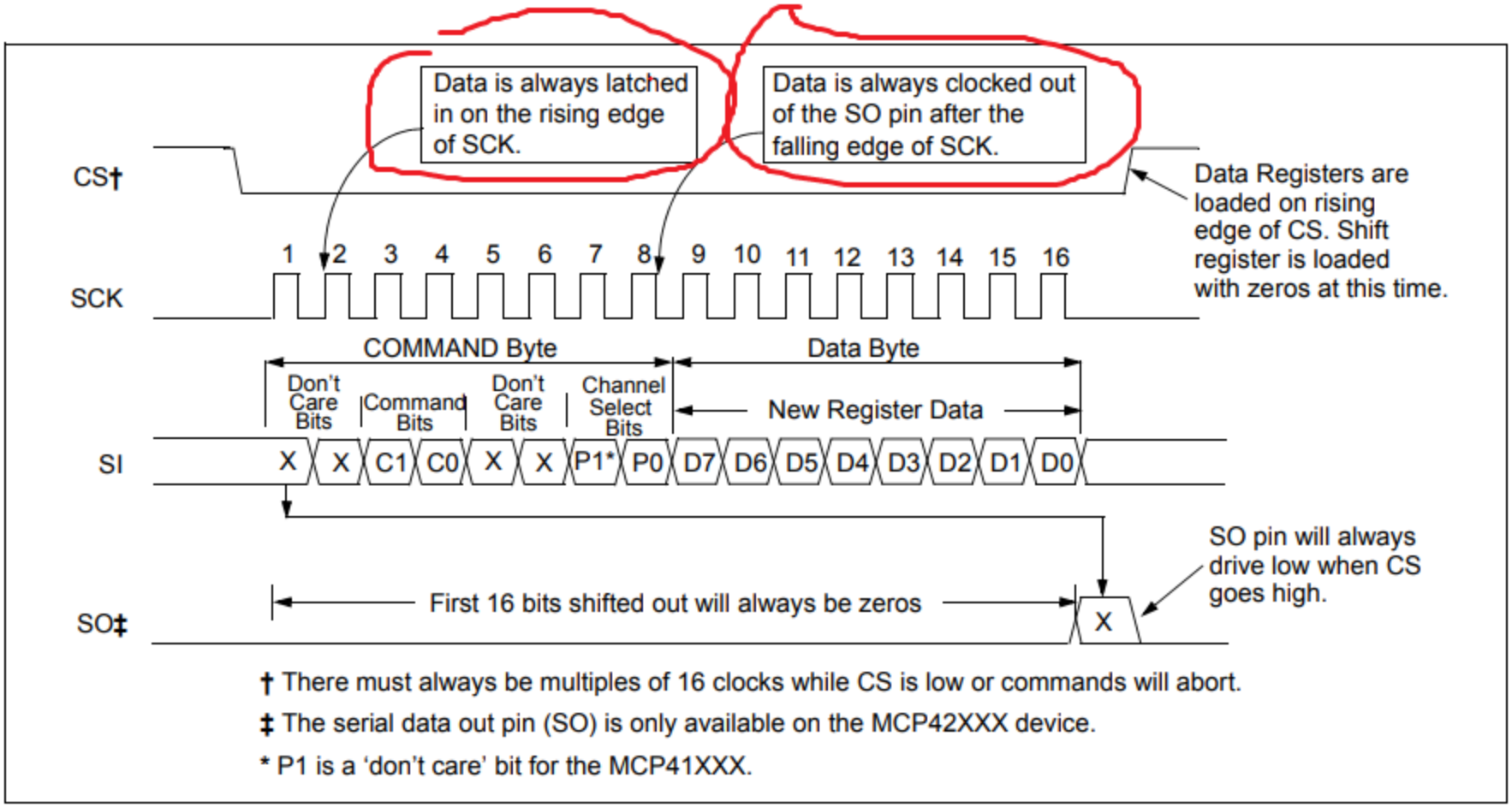


8-bit transfer in the SPI

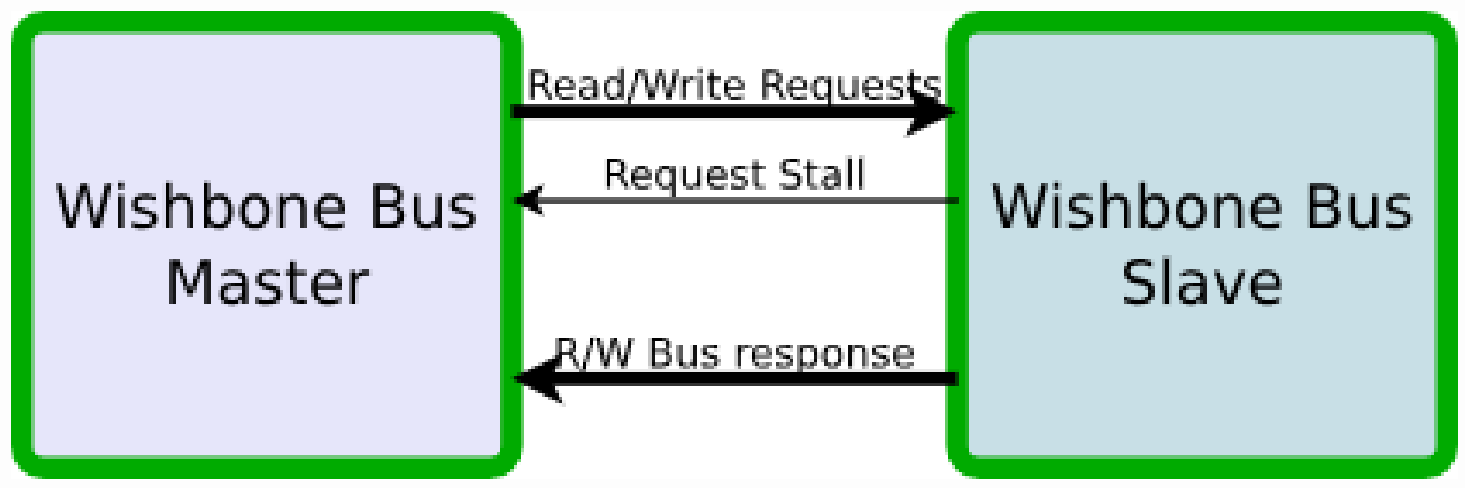
- **8-bit shift register** and **RD data buffer** is the key blocks of SPI interface.
- It is single buffered in the transmit direction
  - to ensure a new character is transferred
    - only after the transfer of the previous character is complete
  - and is double-buffered in the receive direction.



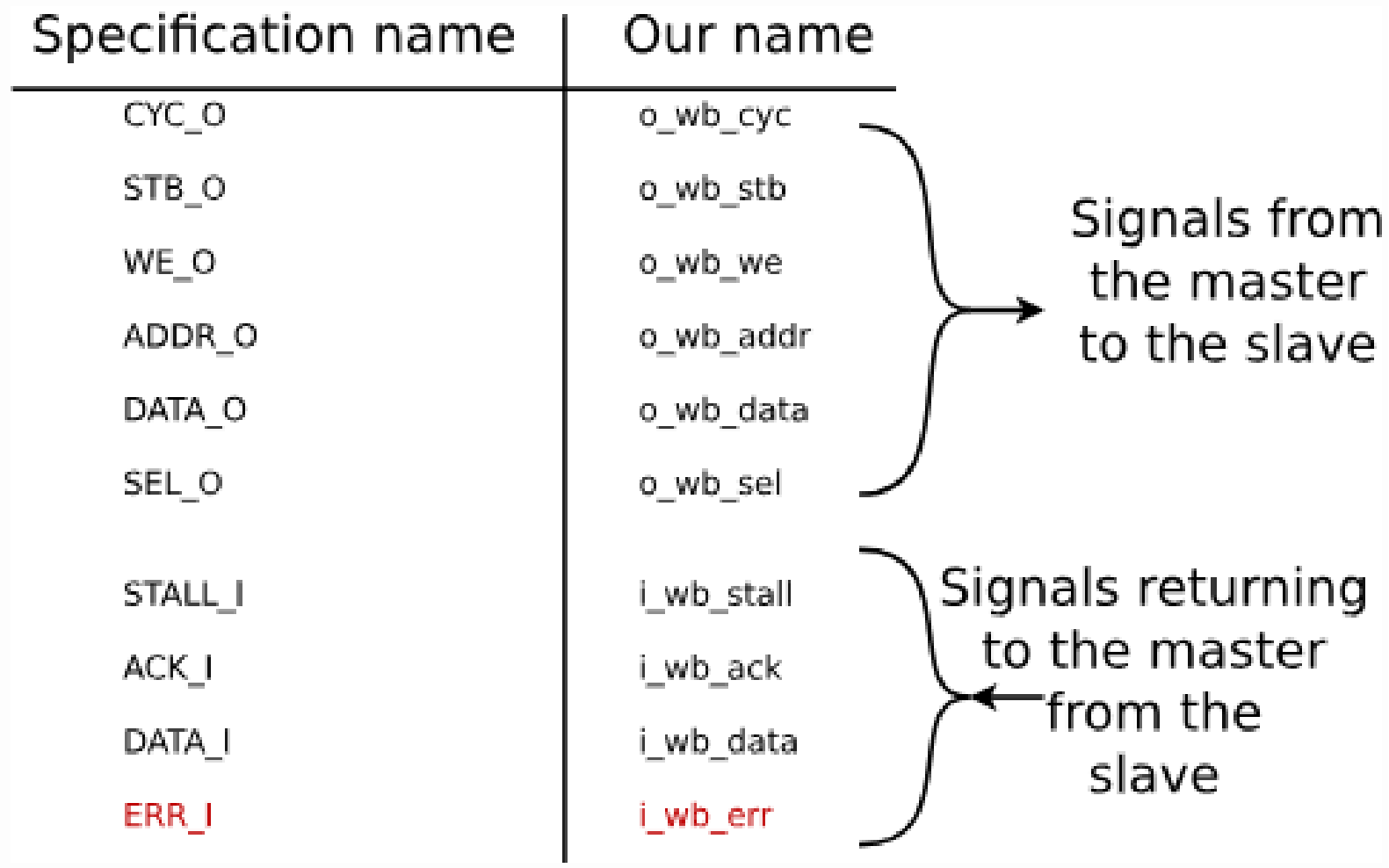
A Typical Transfer Format



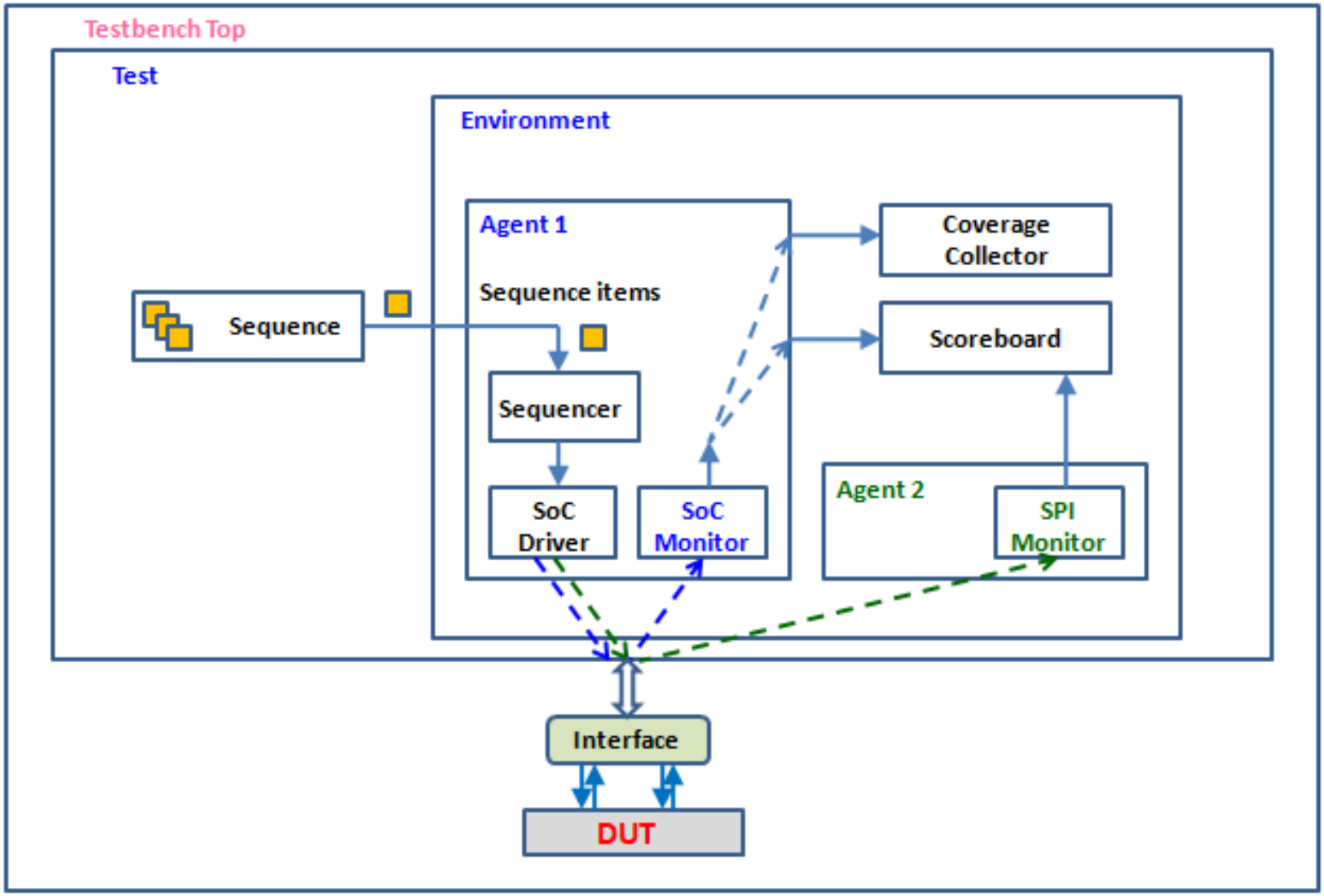
Academic Wishbone



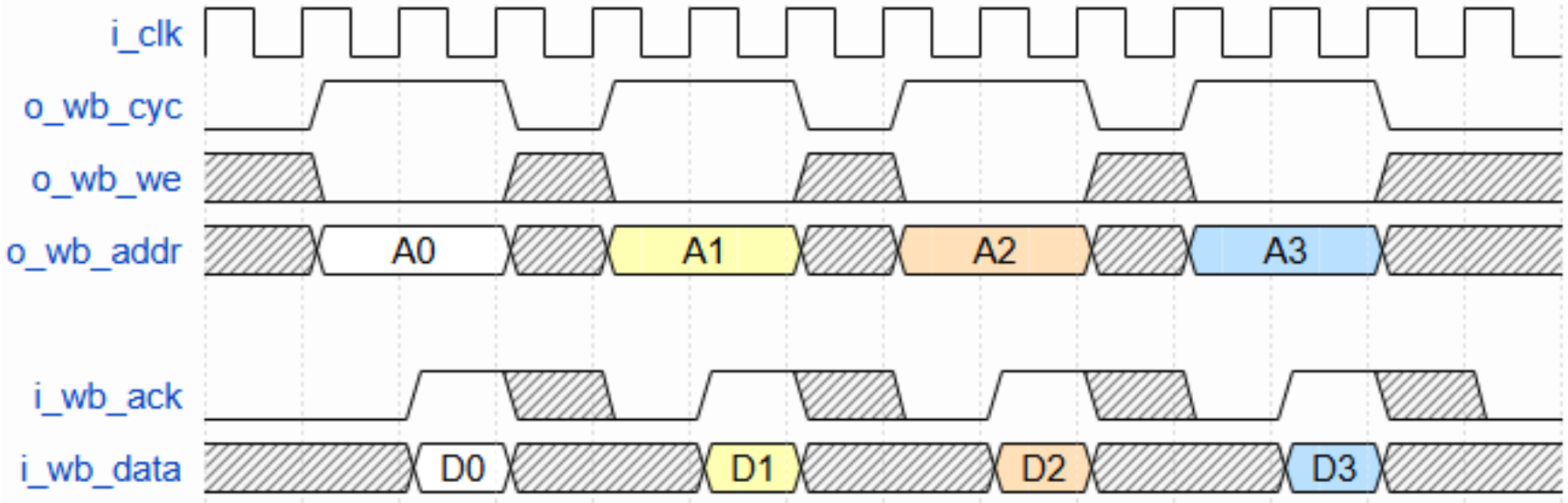
Academic Wishbone (2)



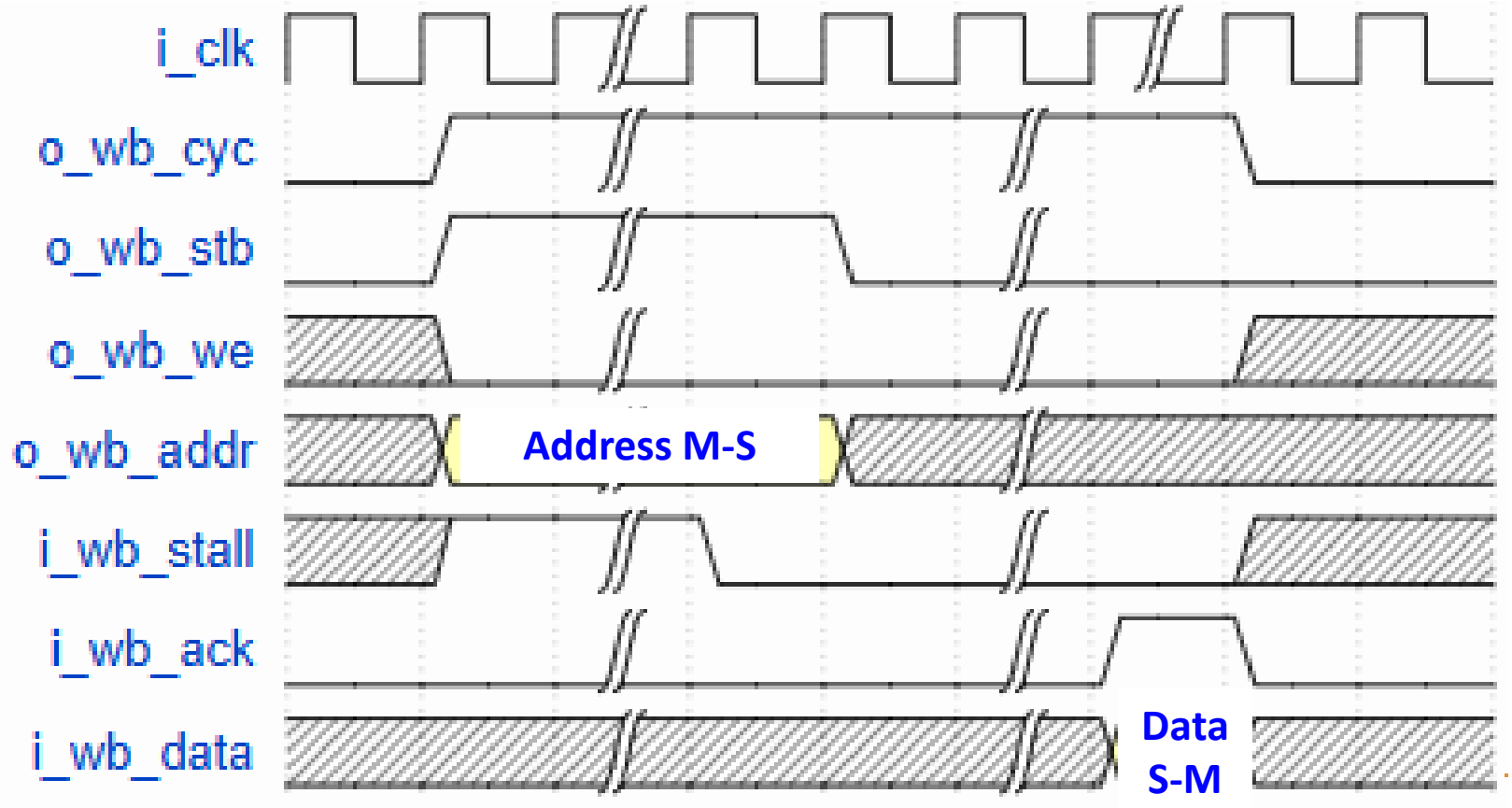
UVM Architecture



Classical Wishbone



A Read Request by Master



Address

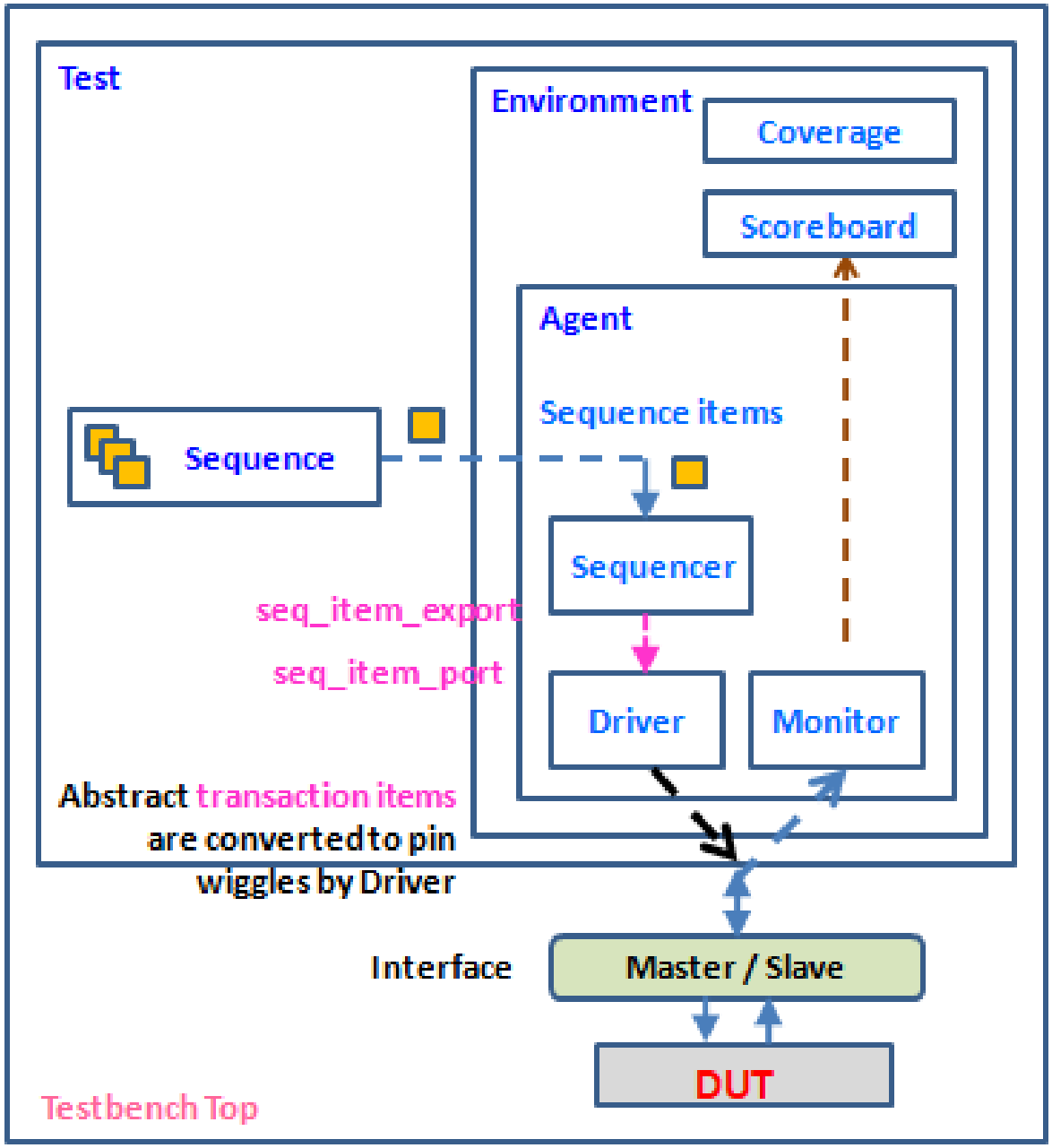
Description

- **Idle** is defined by **o_wb_cyc** and **o_wb_stb** both being low (logic 0). ACK line be low, too.

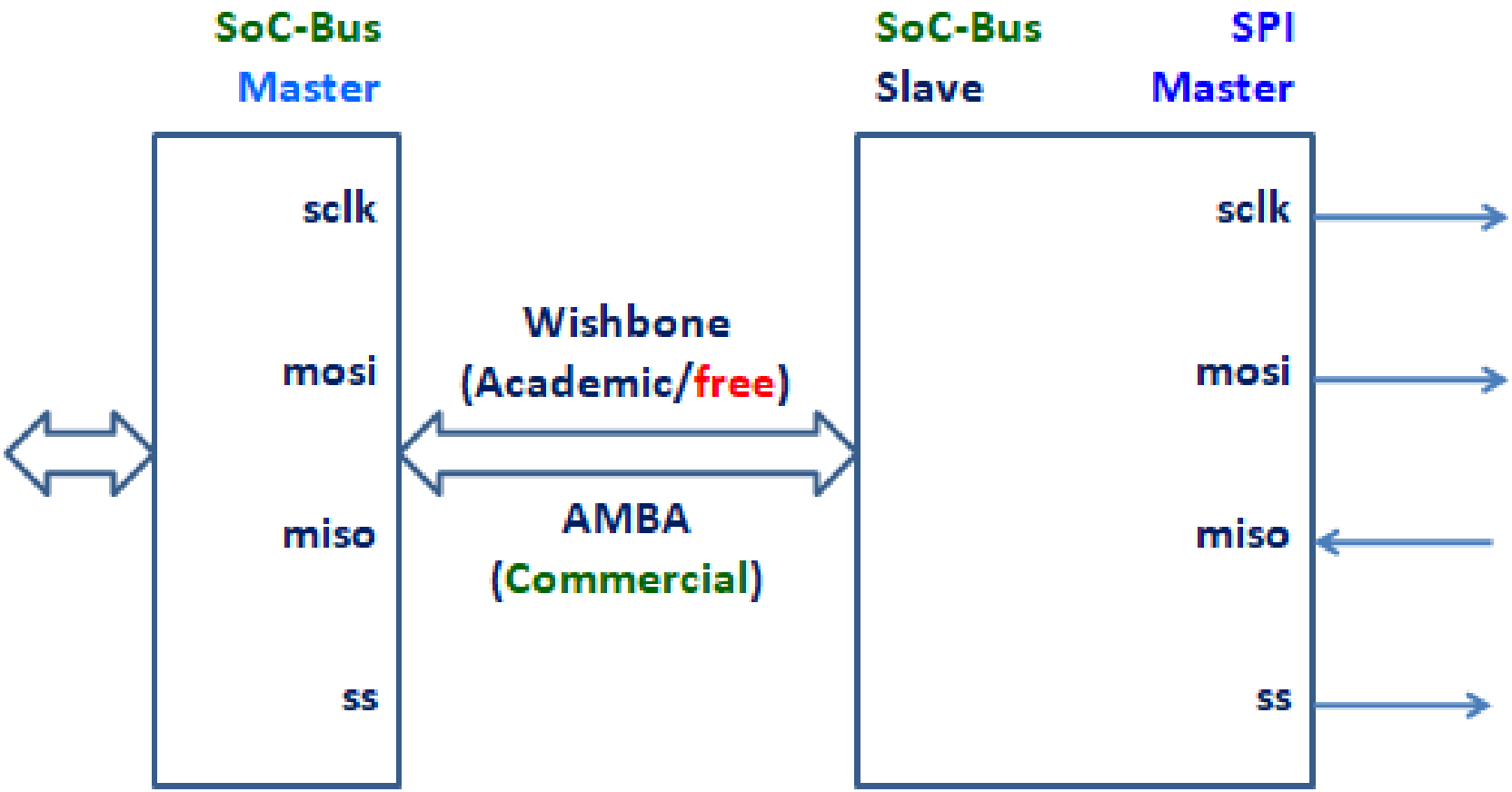
- When the **bus master** chooses to start a transaction,
  - it raises the **o_wb_cyc line** to high (logic 1)
- On the same clock, the **master** places a read request on the bus. This means that
  - the **o_wb_stb** is raised to high (logic 1)
  - and the address is placed on **o_wb_addr** (e.g., [4:0] adr).
- Since this is a read request, **o_wb_we** is held low (logic 0) at this time.
- The **slave** has the opportunity to tell the bus **master** that it is not (yet) ready to receive the request.
  - It does this by holding the **i_wb_stall** line high (logic 1).
  - As soon as **o_wb_stb** is true and **i_wb_stall** is false, the request has been accepted.

Write case

- If this had been a write request, **o_wb_we** would have been raised to logic 1 (high),
  - the data to be written would be placed on **o_wb_data**,
  - and **o_wb_sel** would be filled out with one bit per byte in **o_wb_data**
    - indicating which bytes are actually going to be written.

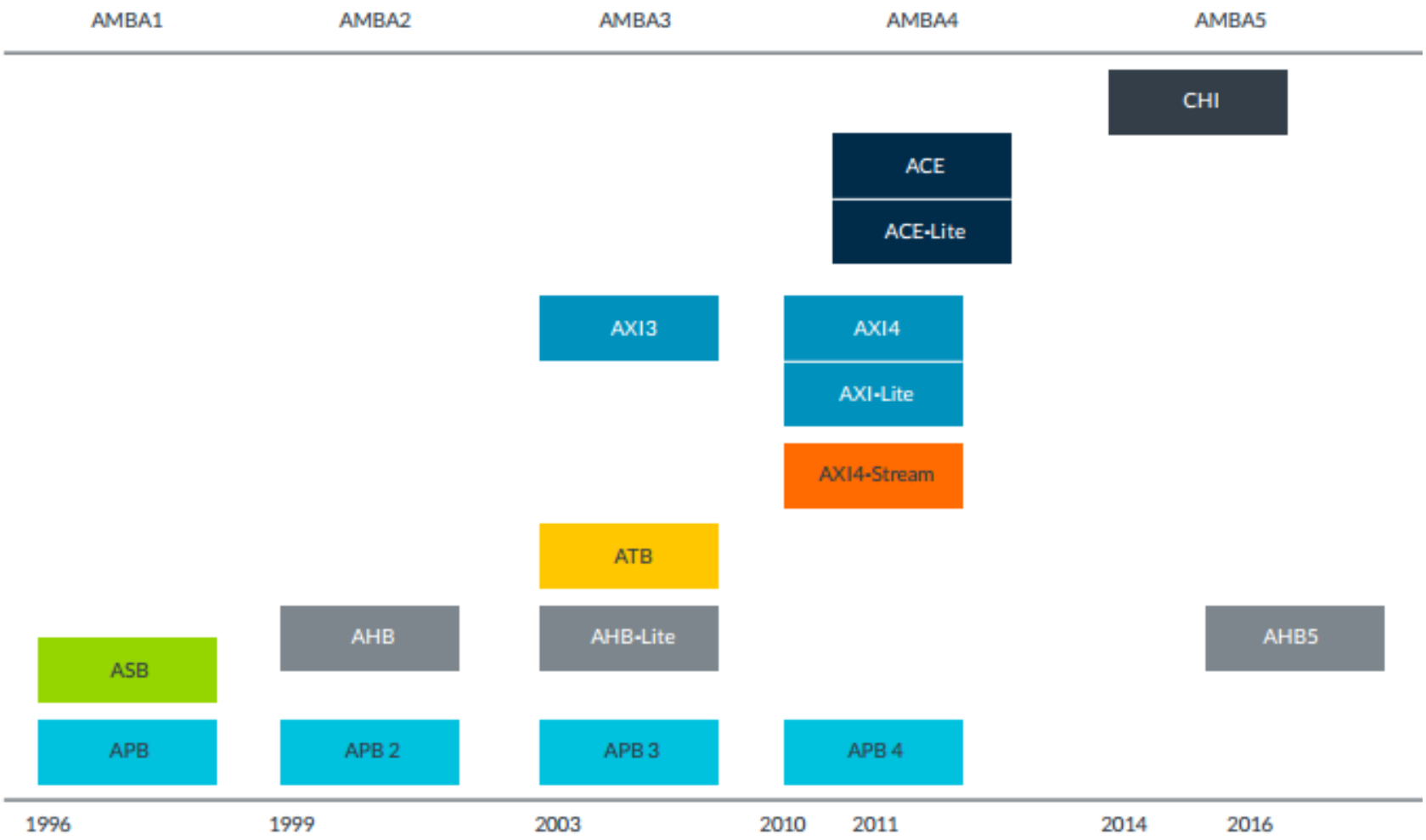

Test
Environment
Coverage
Scoreboard
Agent
Sequence items
Sequence
Sequencer
seq_item_export
seq_item_port
Driver
Monitor
Abstract transaction items
are converted to pin
wiggles by Driver
Interface
Master / Slave
DUT
Testbench Top

DUT – SoC SPI

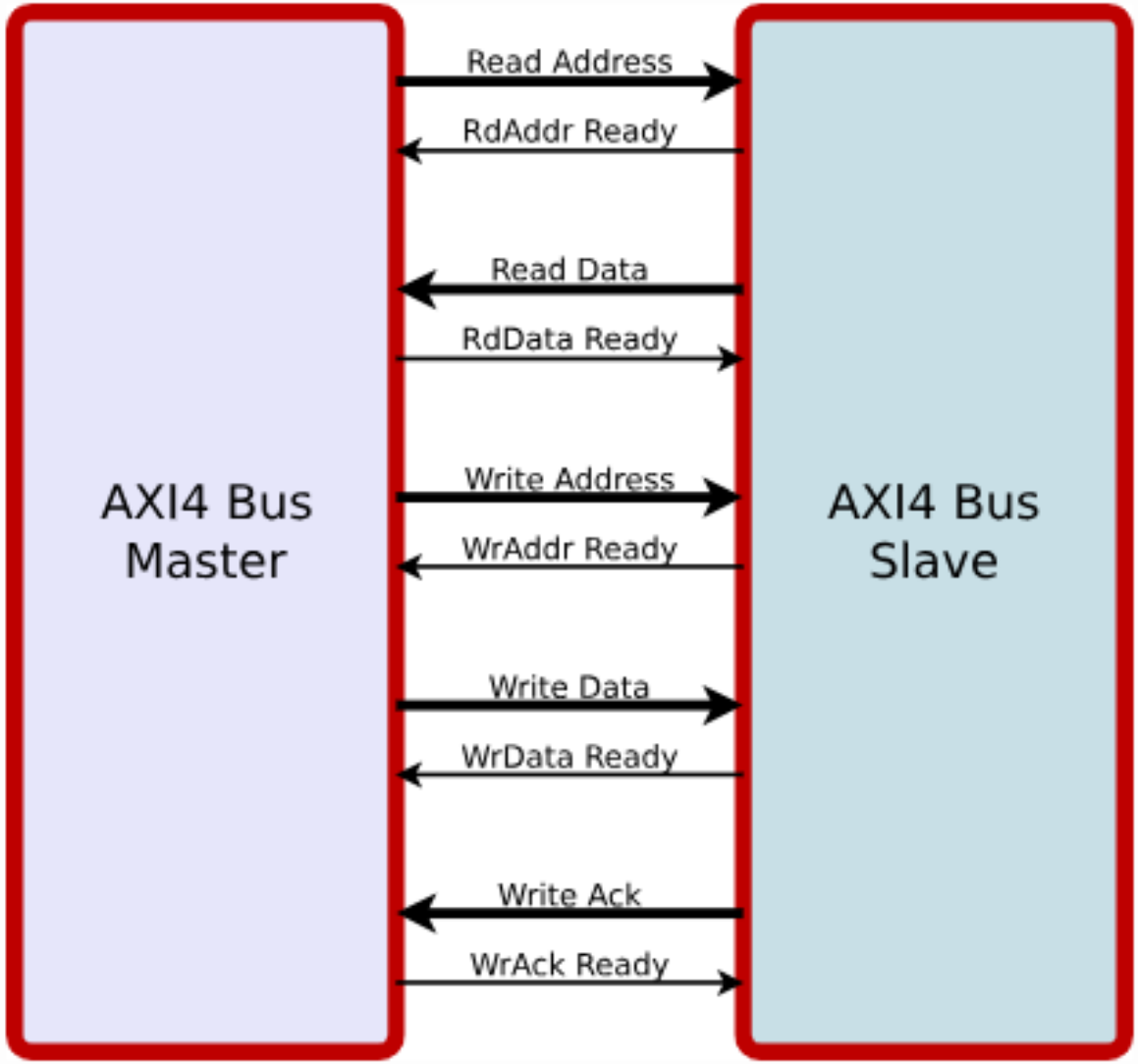

SoC-Bus
Master
SoC-Bus
Slave
SPI
Master
sclk
mosi
miso
ss
Wishbone
(Academic/free)
AMBA
(Commercial)
sclk
mosi
miso
ss

Commercial AMBA

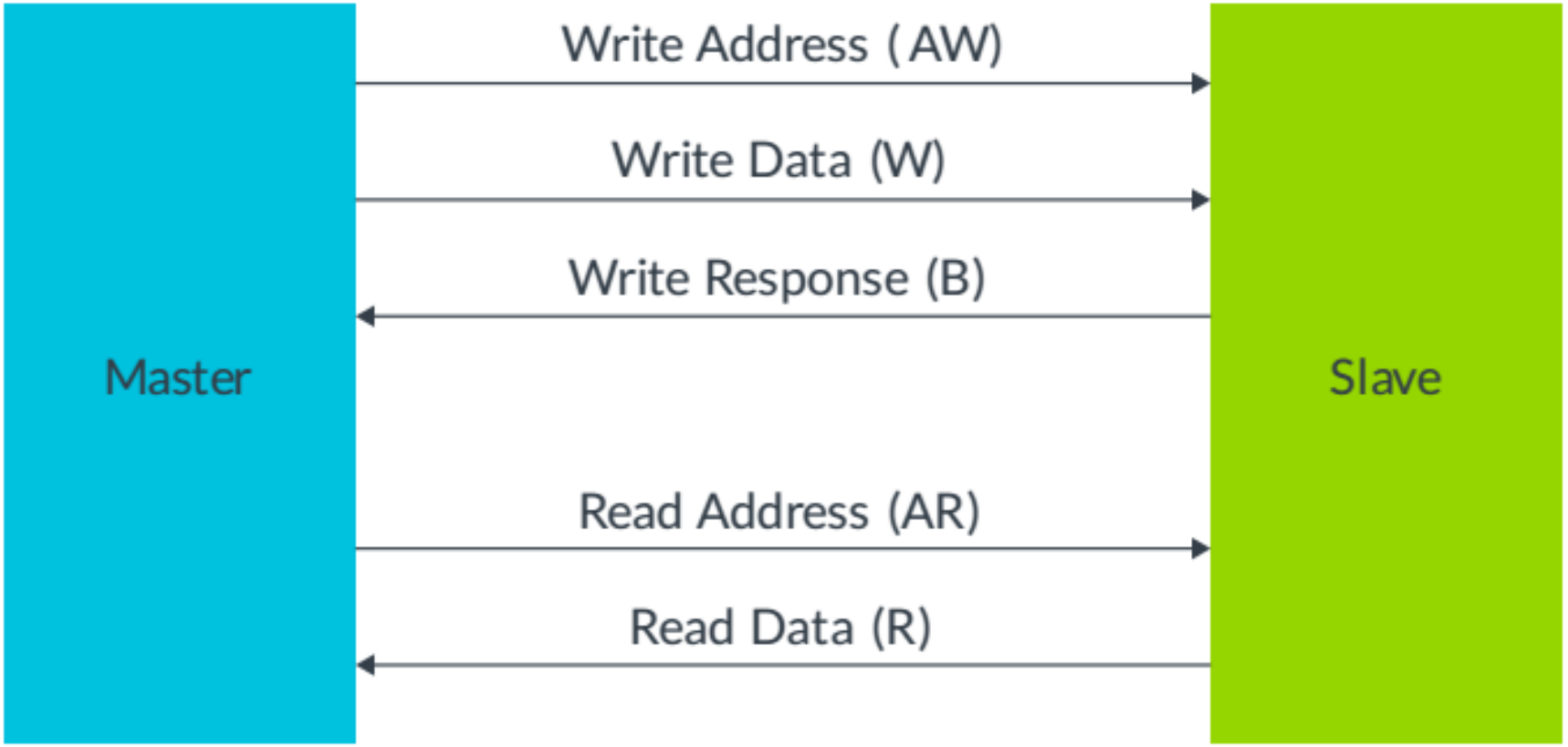
- The other bus in common usage today is the AMBA AXI4 bus.



AMBA AXI-4 Bus



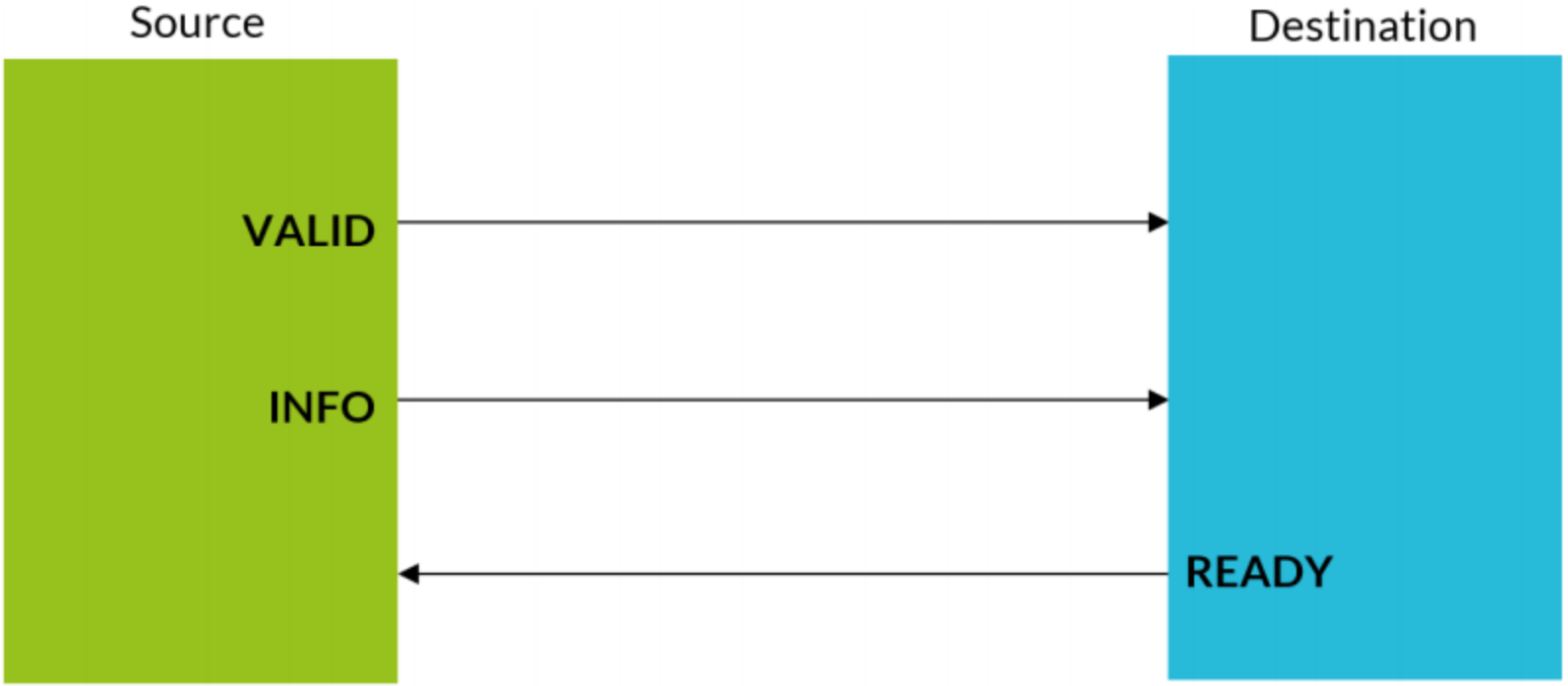
Abstraction



Abstraction (2)

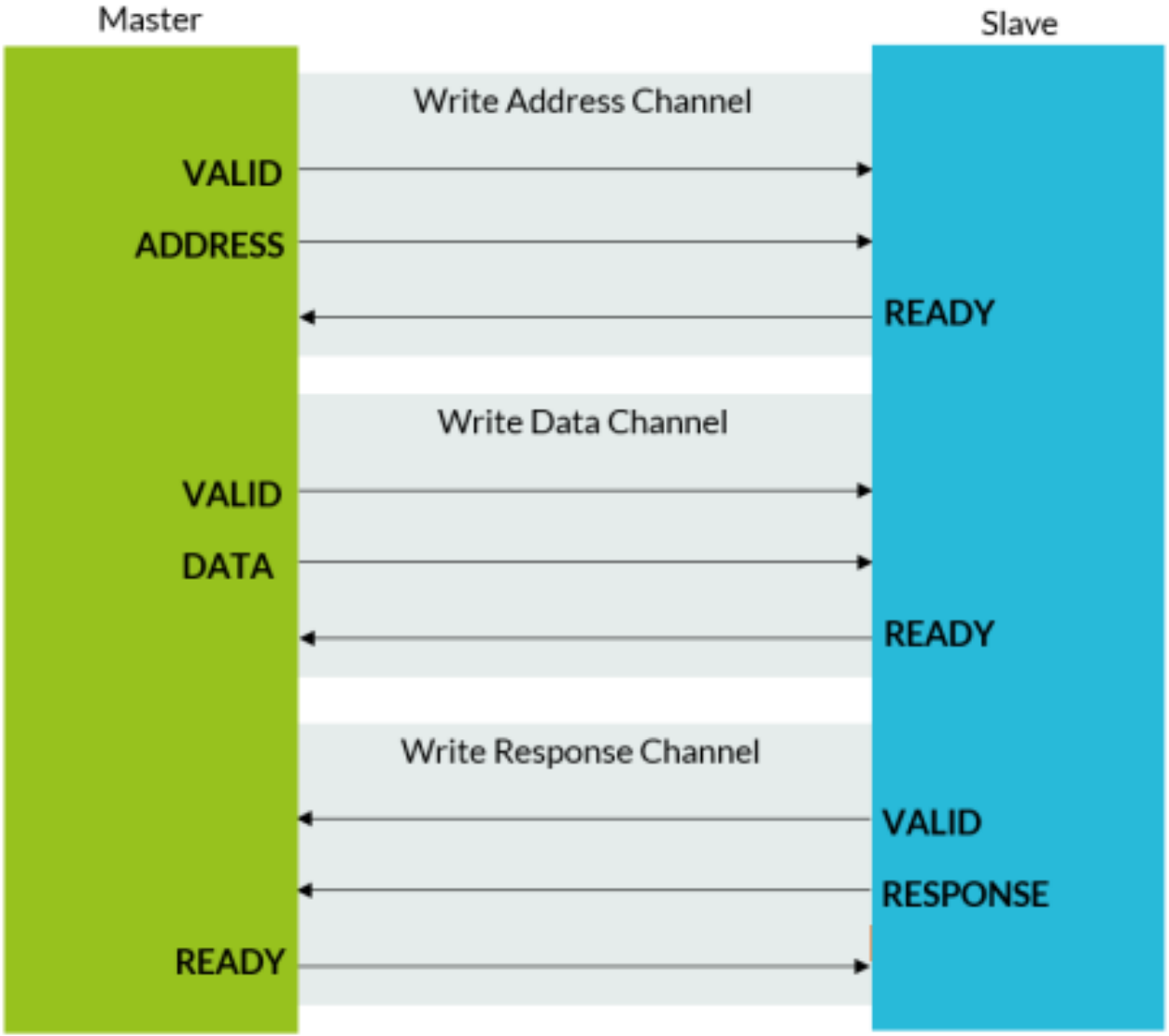
Transfers and Transactions

- A transfer is a single exchange of information, with one VALID and READY handshake.



Transfers and Transactions (2)

- A transaction is an entire burst of transfers, containing an address transfer, one or more data transfers, and, for write sequences, a response transfer.

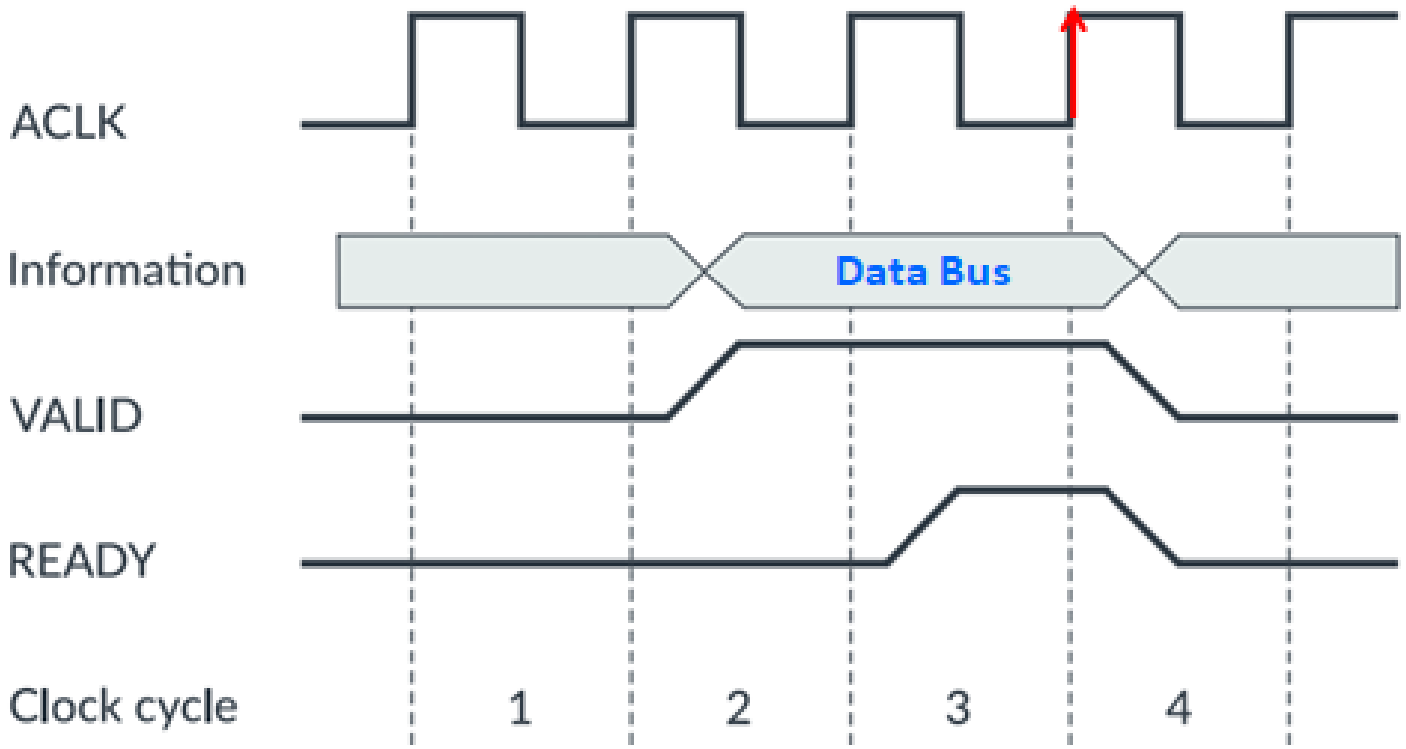


Channel Transfer Examples

Some examples of possible handshakes between source and destination.

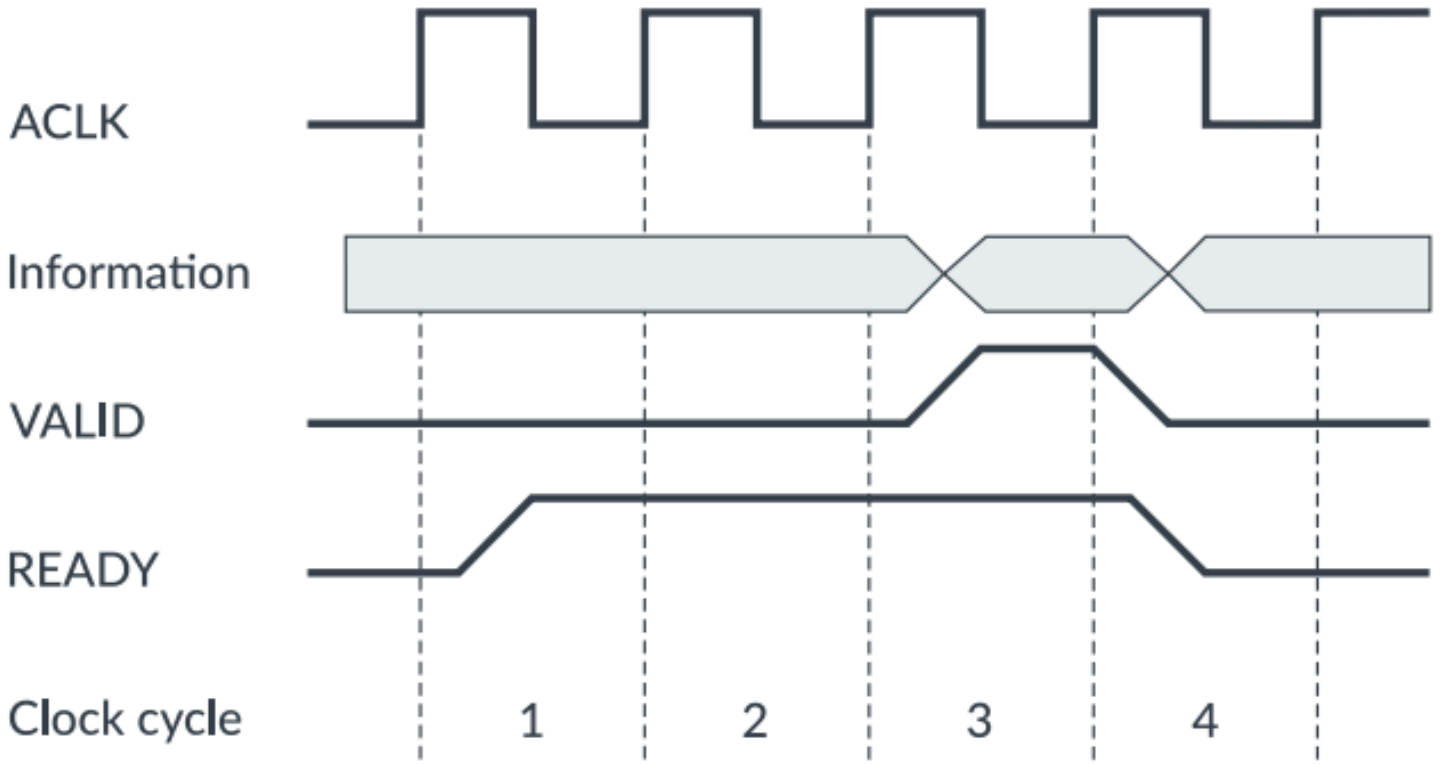
(1)

1. In clock cycle 2, the VALID signal is **asserted**, indicating that the **data** on the information channel is **valid**.
2. In clock cycle 3, the following clock cycle, the READY signal is asserted.
3. The handshake completes on the **rising edge** of clock cycle 4, because both READY and VALID signals are asserted.



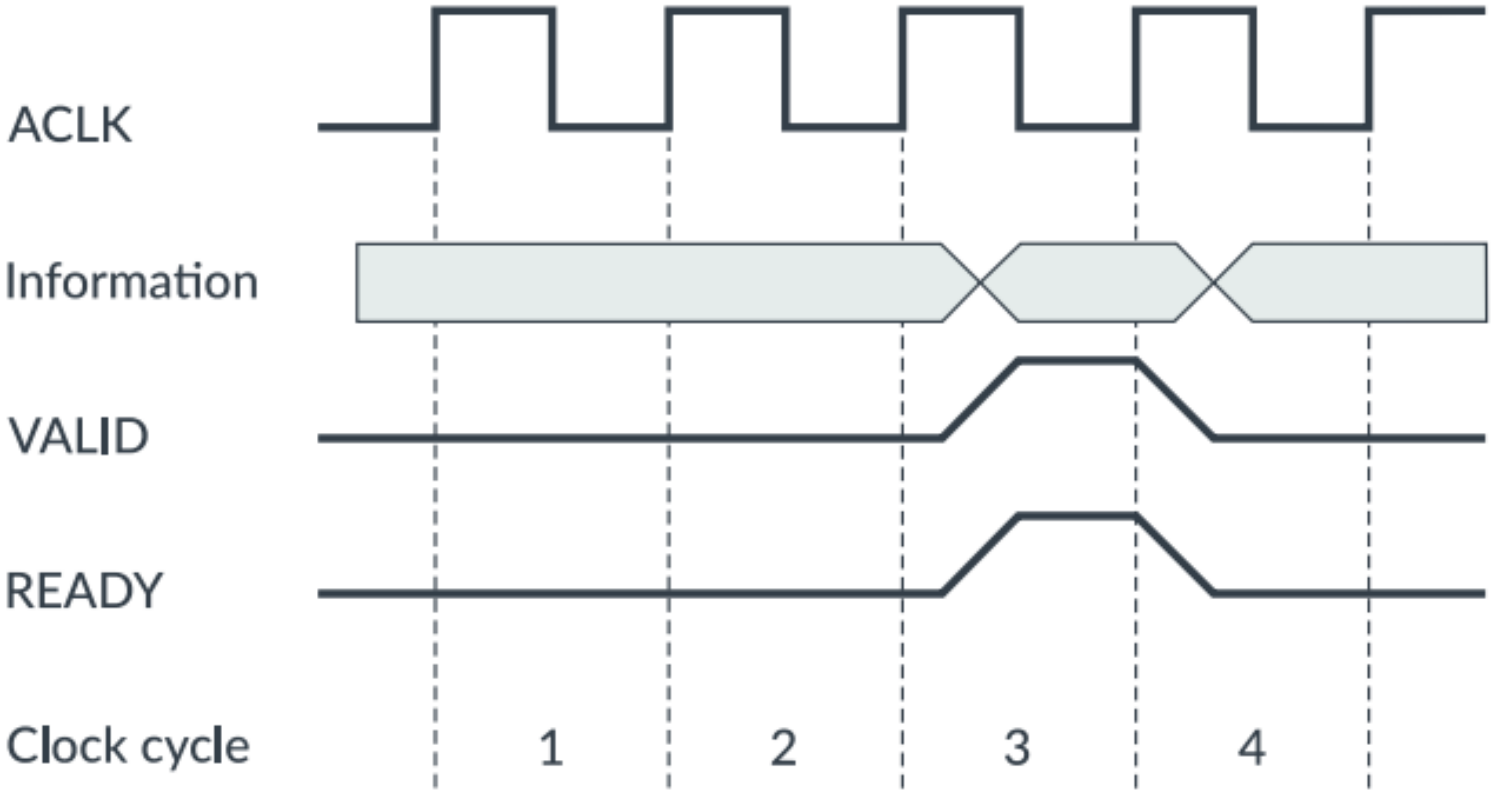
(2)

1. In clock cycle 1, the READY signal is asserted.
2. The VALID signal is not asserted until clock cycle 3.
3. The handshake completes on the rising edge of clock cycle 4, when both VALID and READY are asserted.



(3)

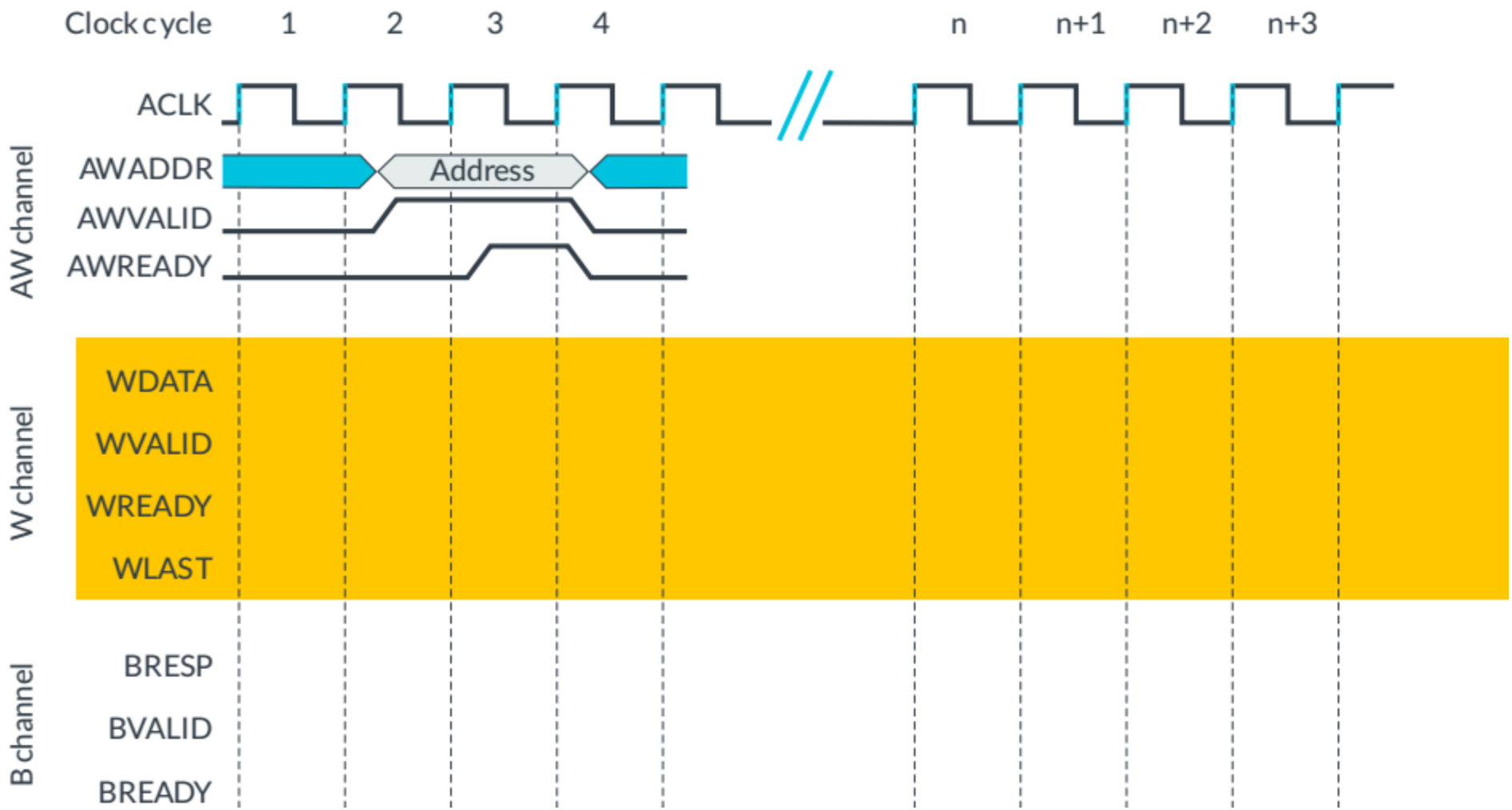
- Both VALID and READY signals being asserted during the clock cycle 3.
- The handshake completes on the rising edge of clock cycle 4, when both VALID and READY are asserted.



Write Transaction: Single Data item

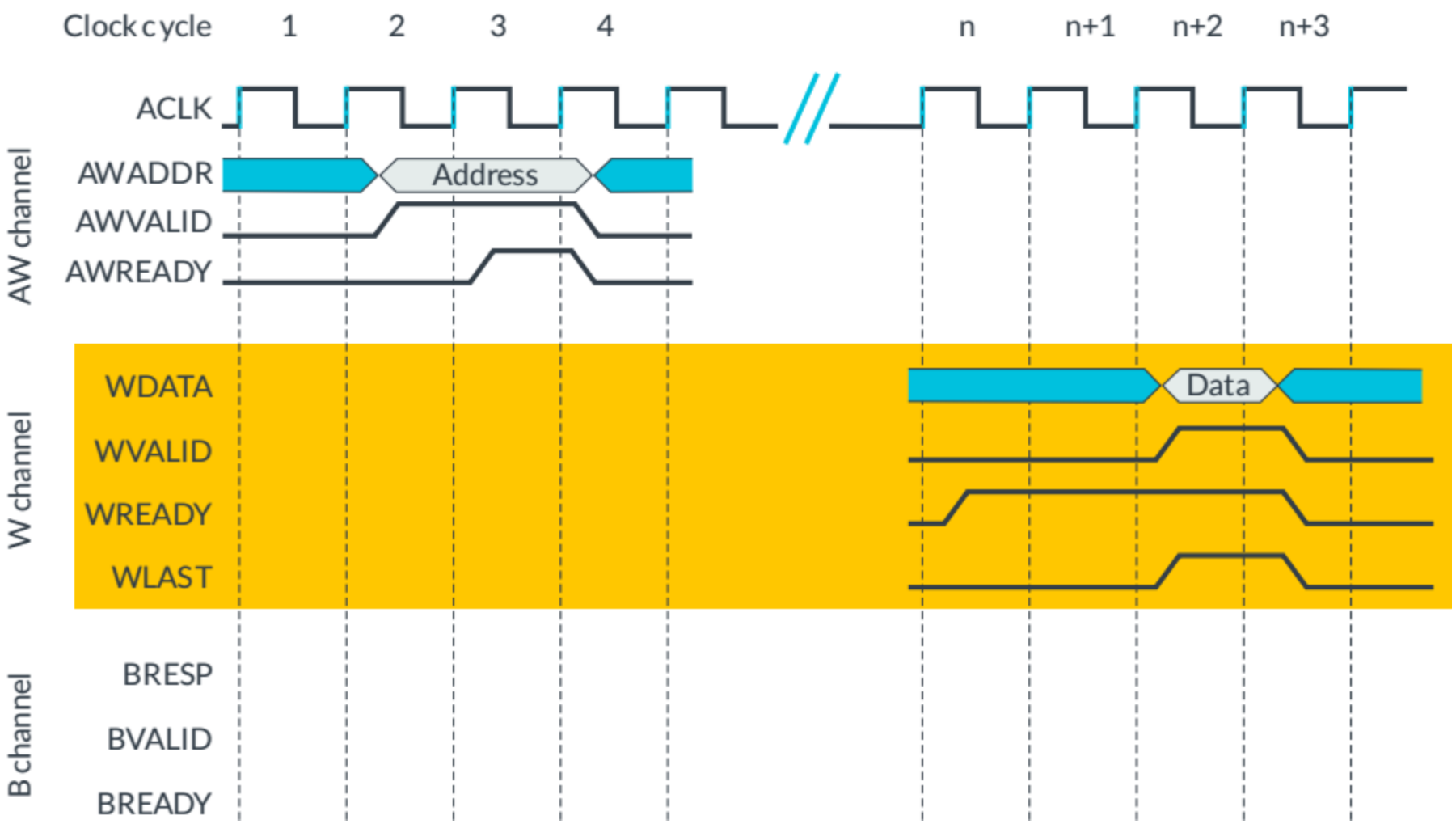
- Process of a write transaction for a single data item, and the different channels that are used to complete the transaction.
- This write transaction involves the following channels:
  - Write Address (AW)
  - Write (W)
  - Write Response (B)

  First, there is a handshake on the Write Address (AW) channel


Clock cycle
1
2
3
4
n
n+1
n+2
n+3
ACLK
AW channel
AWADDR
Address
AWVALID
AWREADY
W channel
WDATA
WVALID
WREADY
WLAST
B channel
BRESP
BVALID
BREADY

Description

- This handshake is where the master communicates the address of the write to the slave. The
  handshake has the following sequence of events:
  1. The master puts the address on AWADDR and asserts AWVALID in clock cycle 2.
  2. The slave asserts AWREADY in clock cycle 3 to indicate its ability to receive the address value.
  3. The handshake completes on the rising edge of clock cycle 4.

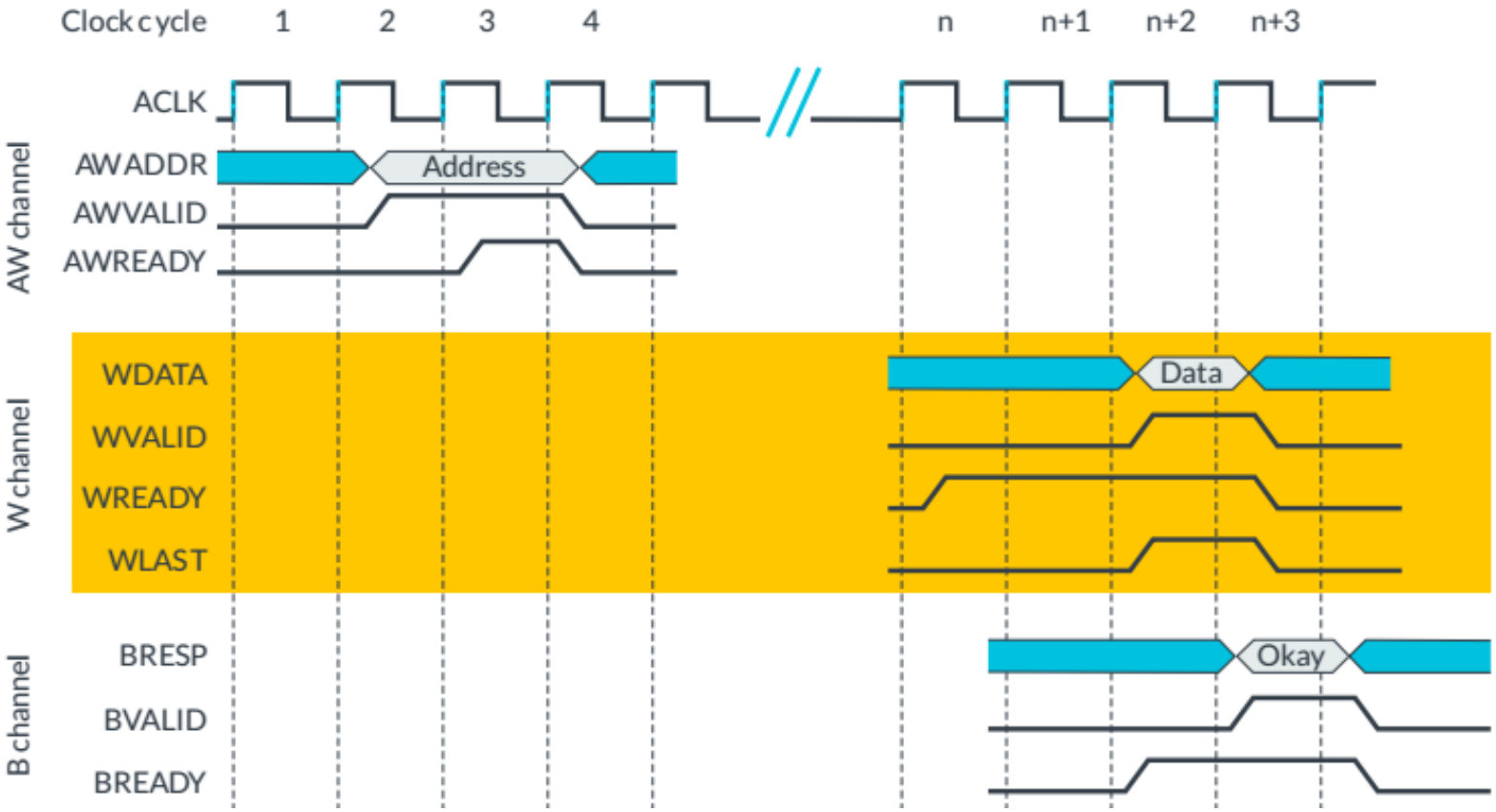

Clock cycle
1
2
3
4
n
n+1
n+2
n+3
ACLK
AW channel
AWADDR
Address
AWVALID
AWREADY
W channel
WDATA
Data
WVALID
WREADY
WLAST
B channel
BRESP
BVALID
BREADY

Description (2)

- The data transfer has the following sequence of events:
  1. The slave is waiting for data with WREADY set to high in clock cycle n.
  2. The master puts the data on the WDATA bus and asserts WVALID in clock cycle n+2.
  3. The handshake completes on the rising edge of clock cycle n+3

Description (3)

- Finally, the slave uses the Write Response (B) channel, to confirm that the write transaction has completed once all WDATA has been received.



Description (4)

- The write response has the following sequence of events:
  1. The master asserts BREADY.
  2. The slave drives BRESP to indicate success or failure of the write transaction, and asserts
  BVALID.
  The handshake completes on the rising edge of clock cycle n+3.

Thank You