AMBA PROTOCOL

AXI4-LITE

Aarti Kumari

DCRUST - ECED

WHAT IS AMBA?

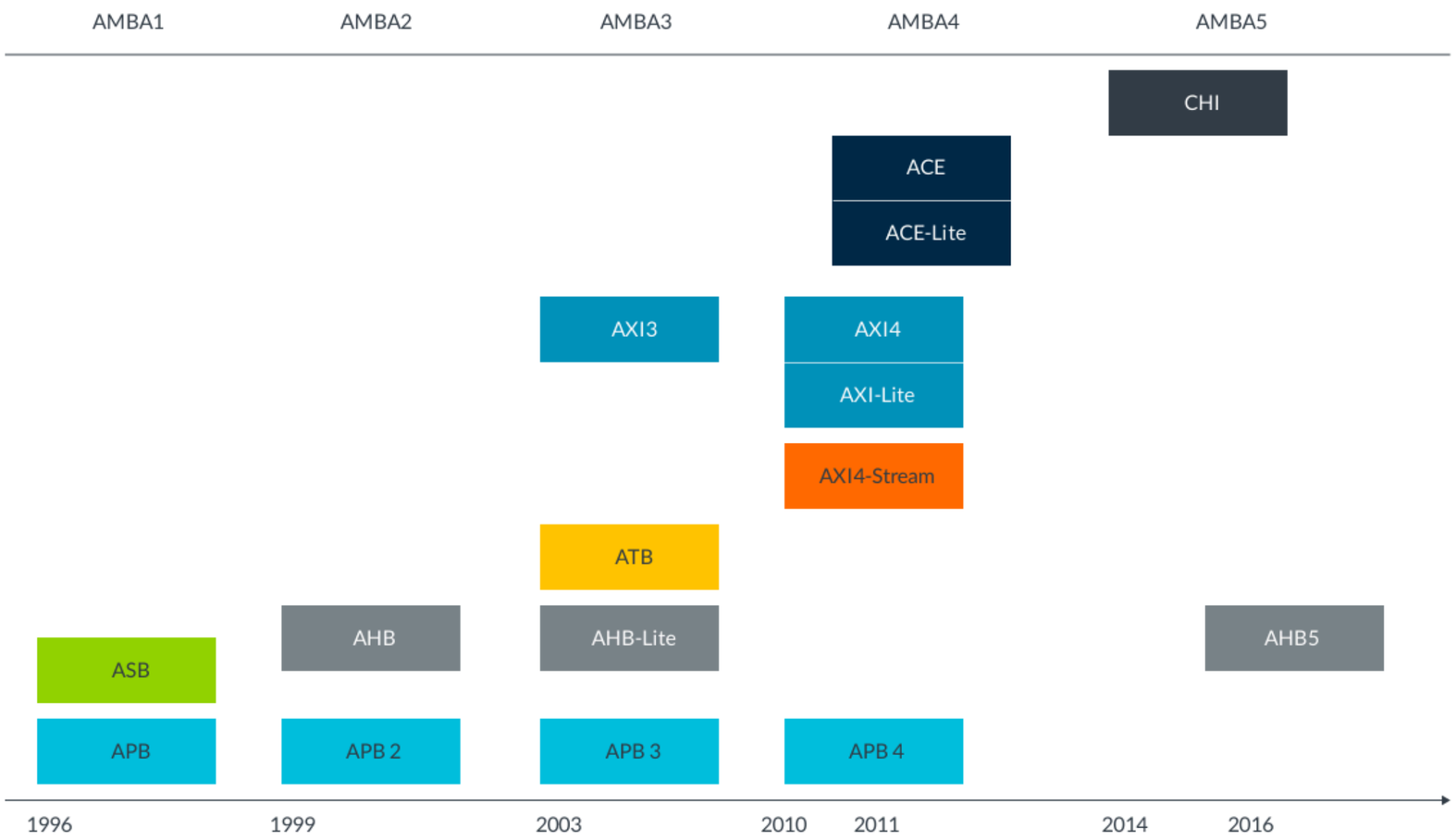
The Arm Advanced Microcontroller Bus Architecture, or AMBA, is an open-standard, on-chip interconnect specification for the connection and management of functional blocks in system-on-a chip (SoC) designs. Essentially, AMBA protocols define how functional blocks communicate with each other.

Where is AMBA used?

AMBA simplifies the development of designs with multiple processors and large numbers of controllers and peripherals. However, the scope of AMBA has increased over time, going far beyond just microcontroller devices.

Today, AMBA is widely used in a range of ASIC and SoC parts. These parts include applications processors that are used in devices like IoT subsystems, smartphones, and networking SoCs.

HOW HAS AMBA EVOLVED?



AMBA 1 (late 1990s):

- Introduced ASB (now obsolete) and APB (still used for low-bandwidth peripheral control).

AMBA 2 (1999):

- Added AHB, a pipelined, high-performance single-master bus; APB remained for simple, non-pipelined use cases.

AMBA 3 (2003):

- Introduced AHB-Lite (single-master simplified AHB) and AXI (Advanced eXtensible Interface), designed for high-frequency, high-performance interconnects with independent address/control and data channels.

AMBA 4 (2010):

Extended AXI with AXI4 family:

- AXI4: For high-bandwidth memory-mapped interconnect.
- AXI4-Lite: A lightweight subset of AXI4, optimized for simple register-mapped interfaces (low bandwidth, low complexity, single transactions without bursts). Commonly used for connecting control/status registers in IP cores.
- AXI4-Stream: For high-speed streaming data transfers (no addressing phase).
- ACE / ACE-Lite: Extensions for cache coherency across multiple processors.

AMBA 5 (2014 onward):

- Introduced CHI (Coherent Hub Interface) for scalable multi-core SoCs, AHB5 with TrustZone security, and ATP for traffic modeling.

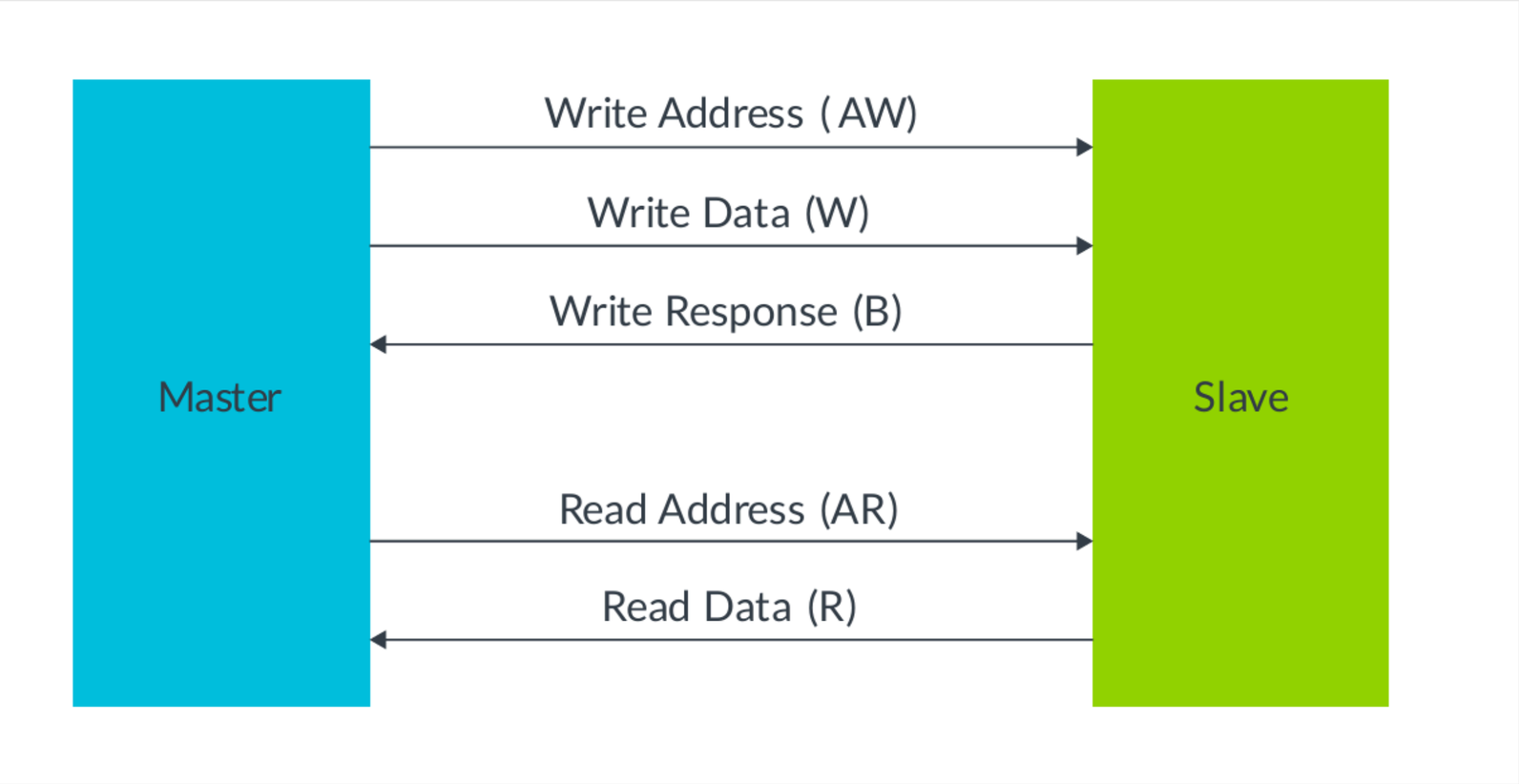
INTRODUCTION TO AXI4-LITE

The AXI4-Lite protocol is a simplified subset of the AXI4 (Advanced eXtensible Interface) protocol, part of the AMBA (Advanced Microcontroller Bus Architecture) specification by ARM.
AXI4-Lite is designed for low-complexity, low-bandwidth memory-mapped communications, and low-latency interconnects making it ideal for control and configuration registers in hardware IP blocks.

It consists of 5 channels, 2 for Reading and 3 for Writing namely:

- READING:
  - Read Data Channel (R)
  - Read Address Channel (AR)
- WRITING:
  - Write Data Channel (W)
  - Write Address Channel (AW)
  - Write Response Channel (B)

CHANNELS IN AXI4-LITE



To note: Each channel is unideirectional.

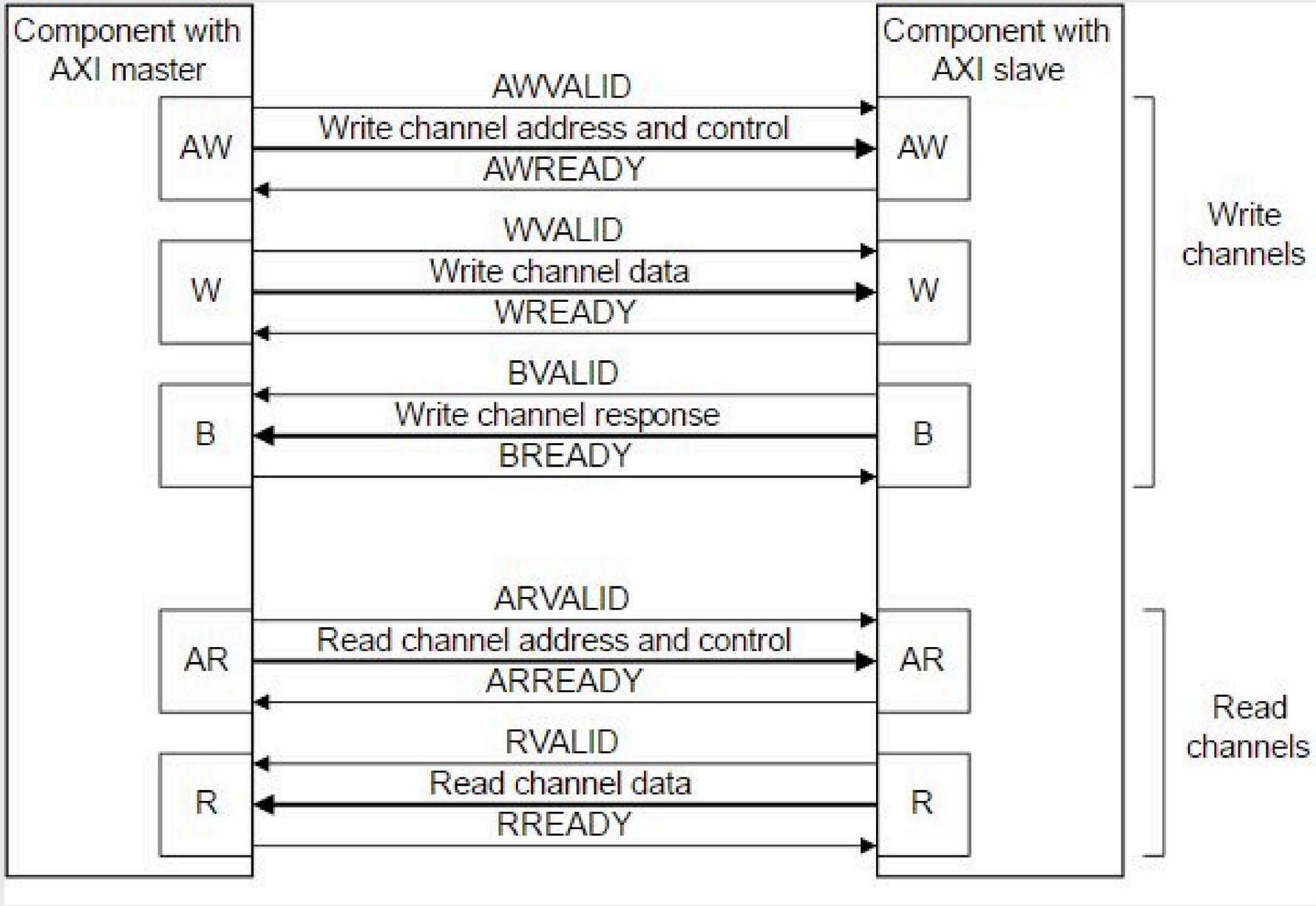
HANDSHAKE SIGNALS

The AXI4 protocol defines five different channels, as described in AXI channels. All of these channels share the same handshake mechanism that is based on the **VALID** and **READY** signals.

The VALID signal goes from the source to the destination, and READY goes from the destination to the source.

- The source (master or slave) asserts VALID when it has information ready (address, data, or response). VALID must stay high until the information is accepted – this makes it a sticky signal.
- The destination asserts READY when it is able to accept the information.
- The actual transfer only happens when VALID = 1 and READY = 1 on the rising clock edge.

Analogy:
VALID (from sender): "I have valid data/address/response ready."
READY (from receiver): "I am ready to accept it."


Component with AXI master
Component with AXI slave
AWVALID
Write channel address and control
AWREADY
AW
AW
WVALID
Write channel data
WREADY
W
W
BVALID
Write channel response
BREADY
B
B
Write channels
ARVALID
Read channel address and control
ARREADY
AR
AR
RVALID
Read channel data
RREADY
R
R
Read channels

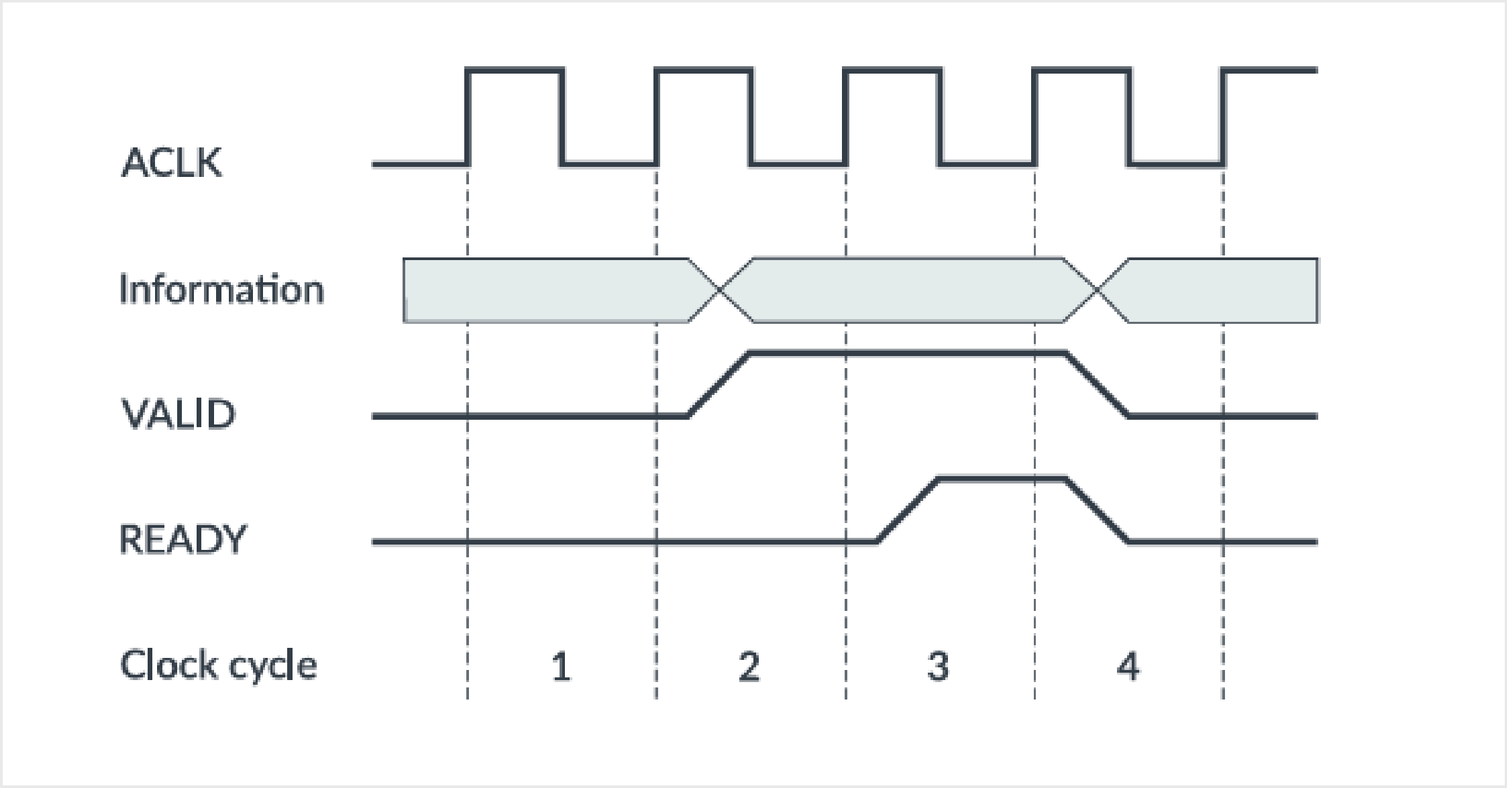
CHANNEL TRANSFER EXAMPLE

This example has the following sequence of events:

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.



WRITE TRANSACTION

A write transaction in AXI4-Lite is made up of three phases, each using the VALID/READY handshake.

1. Write Address Phase (AW channel)

- The master places the write address on the bus and asserts AWVALID.
- The slave asserts AWREADY when it can accept the address.
- When both are high on a clock edge → the address is accepted.

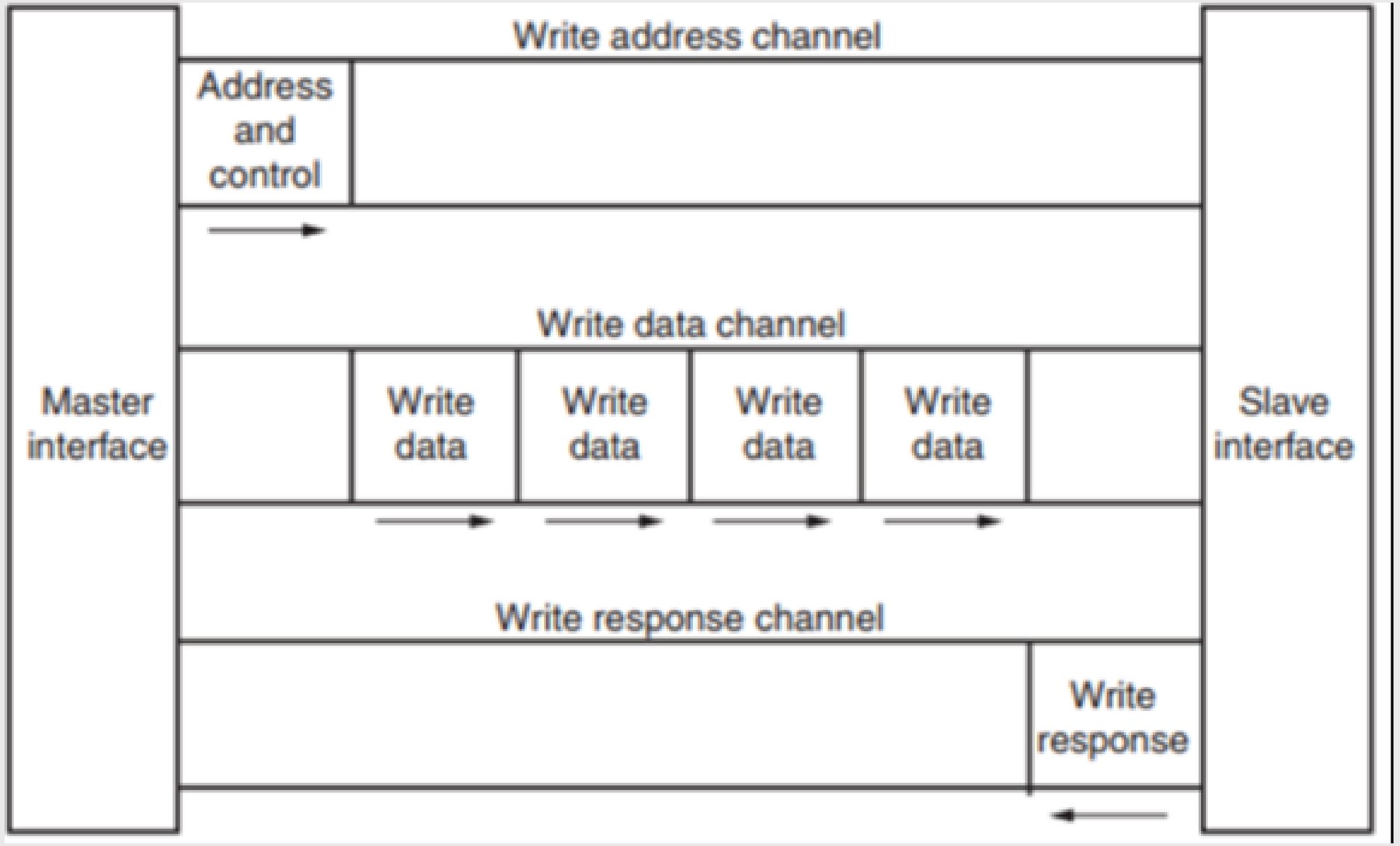
2. Write Data Phase (W channel)

- The master puts the write data (and byte strobes WSTRB) on the bus and asserts WVALID.
- The slave asserts WREADY when it can accept the data.
- When both are high on a clock edge → the data is accepted.

3. Write Response Phase (B channel)

- After receiving address + data, the slave generates a response (OKAY for success, or SLVERR for error) and asserts BVALID.
- The master asserts BREADY when it is ready to take the response.
- When both are high on a clock edge → response is accepted, and the transaction ends.

WRITE TRANSACTION

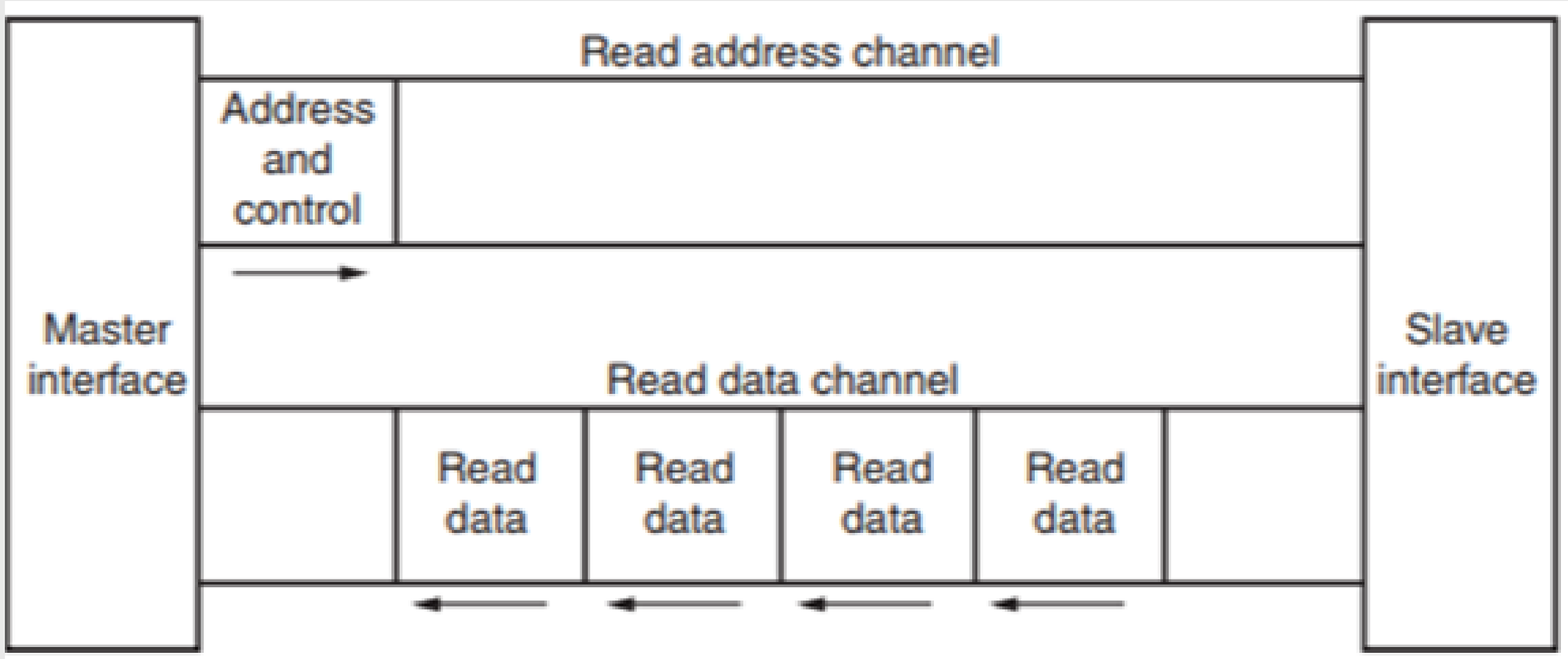


READ TRANSACTION

A read transaction in AXI4-Lite has two main phases, each using the VALID/READY handshake.

1. Read Address Phase (AR channel)
   - The master places the read address on the bus and asserts ARVALID.
   - The slave asserts ARREADY when it can accept the address.
   - When both are high on a clock edge → the address is accepted.
2. Read Data Phase (R channel)
   - The slave retrieves the requested data from memory or peripheral and places it on the bus, asserting RVALID.
   - The master asserts RREADY when it is ready to accept the data.
   - When both are high on a clock edge → the data is accepted, completing the transaction.
   - The slave also provides a response signal (RRESP) indicating success (OKAY) or error (SLVERR).

READ TRANSACTION



DESIGNING AND VERIFICATION OF AMBA AXI4-LITE

Designed and verification using verilog HDL language At Xilinx Vivado and Vivado Simulator.

SUMMARY OF CODING AXI4 LITE

The design implements a minimal AXI4-Lite master–slave system using simple FSMs and standard valid/ready handshakes for single-shot read and write transactions.

Master Module: Parameterized by address/data width, the master drives AXI channels via FSM states (IDLE, write, read phases), asserting addresses/data and waiting for handshakes to complete transactions.

Slave Module: Implements a 32-word register file, responds to AXI addresses/data, and manages READY/VALID signals and OKAY responses using FSMs for write and read operations.

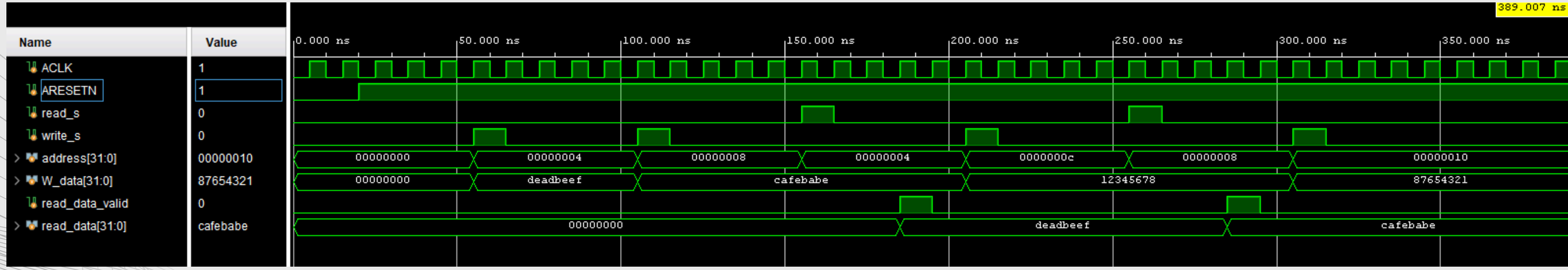
Top Module: Instantiates master and slave, directly wires AXI channels between them, and exposes control/data signals for integration and functional testing of single transactions.

WAVEFORM ANALYSIS

Tcl Console | Messages | Log

```
=== All Transactions Completed ===
Total simulation time: 395000

=== Enhanced Simulation Summary ===
Testbench completed successfully
Check waveform file: axi4_lite_test.vcd
All AXI4-Lite transactions executed as per sequence:
  1. IDLE
  2. WRITE (0xDEADBEEF to 0x04)
  3. WRITE (0xCAFEBABE to 0x08)
  4. READ (from 0x04) - with verification
  5. WRITE (0x12345678 to 0x0C)
  6. READ (from 0x08) - with verification
  7. WRITE (0x87654321 to 0x10)
```



WRITE TRANSACTION REPORT

Tcl Console | Messages | Log

```
# run 1000ns

=== AXI4-Lite Enhanced Testbench Started ===
Time: 0 - Applying Reset
Time: 30000 - Reset Released

--- IDLE Phase ---

--- First WRITE Transaction ---
Time: 50000 - Writing 0xDEADBEEF to address 0x00000004

--- Signal Monitor at Time: 75000 ---
MASTER STATE: 1 (   WRITE_CHANNEL)
SLAVE STATE:  0 (            IDLE)
  WRITE ADDR:   AWVALID=1, AWREADY=0, AWADDR=0x00000004
  WRITE DATA:   WVALID=1, WREADY=0, WDATA=0xdeadbeef, WSTRB=1111
  WRITE RESP:   BVALID=0, BREADY=1, BRESP=00

--- Signal Monitor at Time: 85000 ---
MASTER STATE: 1 (   WRITE_CHANNEL)
SLAVE STATE:  1 (   WRITE_CHANNEL)
  WRITE ADDR:   AWVALID=1, AWREADY=1, AWADDR=0x00000004
  WRITE DATA:   WVALID=1, WREADY=1, WDATA=0xdeadbeef, WSTRB=1111
  WRITE RESP:   BVALID=0, BREADY=1, BRESP=00
Time: 85000 - WRITE ADDRESS HANDSHAKE: Addr=0x00000004
Time: 85000 - WRITE DATA HANDSHAKE: Data=0xdeadbeef

--- Signal Monitor at Time: 95000 ---
MASTER STATE: 2 (   WRESP_CHANNEL)
SLAVE STATE:  2 (   WRESP_CHANNEL)
  WRITE RESP:   BVALID=1, BREADY=1, BRESP=00
Time: 95000 - WRITE RESPONSE HANDSHAKE: BRESP=00
Time: 95000 - Write completed: Addr=0x00000004, Data=0xdeadbeef
```

Tcl Console | Messages | Log

```
--- Second WRITE Transaction ---
Time: 95000 - Writing 0xCAFEBABE to address 0x00000008

--- Signal Monitor at Time: 125000 ---
MASTER STATE: 1 (   WRITE_CHANNEL)
SLAVE STATE:  0 (            IDLE)
  WRITE ADDR:   AWVALID=1, AWREADY=0, AWADDR=0x00000008
  WRITE DATA:   WVALID=1, WREADY=0, WDATA=0xcafebabe, WSTRB=1111
  WRITE RESP:   BVALID=0, BREADY=1, BRESP=00

--- Signal Monitor at Time: 135000 ---
MASTER STATE: 1 (   WRITE_CHANNEL)
SLAVE STATE:  1 (   WRITE_CHANNEL)
  WRITE ADDR:   AWVALID=1, AWREADY=1, AWADDR=0x00000008
  WRITE DATA:   WVALID=1, WREADY=1, WDATA=0xcafebabe, WSTRB=1111
  WRITE RESP:   BVALID=0, BREADY=1, BRESP=00
Time: 135000 - WRITE ADDRESS HANDSHAKE: Addr=0x00000008
Time: 135000 - WRITE DATA HANDSHAKE: Data=0xcafebabe

--- Signal Monitor at Time: 145000 ---
MASTER STATE: 2 (   WRESP_CHANNEL)
SLAVE STATE:  2 (   WRESP_CHANNEL)
  WRITE RESP:   BVALID=1, BREADY=1, BRESP=00
Time: 145000 - WRITE RESPONSE HANDSHAKE: BRESP=00
Time: 145000 - Write completed: Addr=0x00000008, Data=0xcafebabe
```

WRITE TRANSACTION REPORT

Tcl Console | Messages | Log

```
--- Third WRITE Transaction ---
Time: 195000 - Writing 0x12345678 to address 0x0000000C

--- Signal Monitor at Time: 225000 ---
MASTER STATE: 1 (   WRITE_CHANNEL)
SLAVE STATE:  0 (            IDLE)
  WRITE ADDR:   AWVALID=1, AWREADY=0, AWADDR=0x0000000c
  WRITE DATA:   WVALID=1, WREADY=0, WDATA=0x12345678, WSTRB=1111
  WRITE RESP:   BVALID=0, BREADY=1, BRESP=00

--- Signal Monitor at Time: 235000 ---
MASTER STATE: 1 (   WRITE_CHANNEL)
SLAVE STATE:  1 (   WRITE_CHANNEL)
  WRITE ADDR:   AWVALID=1, AWREADY=1, AWADDR=0x0000000c
  WRITE DATA:   WVALID=1, WREADY=1, WDATA=0x12345678, WSTRB=1111
  WRITE RESP:   BVALID=0, BREADY=1, BRESP=00
Time: 235000 - WRITE ADDRESS HANDSHAKE: Addr=0x0000000c
Time: 235000 - WRITE DATA HANDSHAKE: Data=0x12345678

--- Signal Monitor at Time: 245000 ---
MASTER STATE: 2 (   WRESP_CHANNEL)
SLAVE STATE:  2 (   WRESP_CHANNEL)
  WRITE RESP:   BVALID=1, BREADY=1, BRESP=00
Time: 245000 - WRITE RESPONSE HANDSHAKE: BRESP=00
Time: 245000 - Write completed: Addr=0x0000000c, Data=0x12345678
```

Tcl Console | Messages | Log

```
--- Fourth WRITE Transaction ---
Time: 295000 - Writing 0x87654321 to address 0x00000010

--- Signal Monitor at Time: 325000 ---
MASTER STATE: 1 (   WRITE_CHANNEL)
SLAVE STATE:  0 (            IDLE)
  WRITE ADDR:   AWVALID=1, AWREADY=0, AWADDR=0x00000010
  WRITE DATA:   WVALID=1, WREADY=0, WDATA=0x87654321, WSTRB=1111
  WRITE RESP:   BVALID=0, BREADY=1, BRESP=00

--- Signal Monitor at Time: 335000 ---
MASTER STATE: 1 (   WRITE_CHANNEL)
SLAVE STATE:  1 (   WRITE_CHANNEL)
  WRITE ADDR:   AWVALID=1, AWREADY=1, AWADDR=0x00000010
  WRITE DATA:   WVALID=1, WREADY=1, WDATA=0x87654321, WSTRB=1111
  WRITE RESP:   BVALID=0, BREADY=1, BRESP=00
Time: 335000 - WRITE ADDRESS HANDSHAKE: Addr=0x00000010
Time: 335000 - WRITE DATA HANDSHAKE: Data=0x87654321

--- Signal Monitor at Time: 345000 ---
MASTER STATE: 2 (   WRESP_CHANNEL)
SLAVE STATE:  2 (   WRESP_CHANNEL)
  WRITE RESP:   BVALID=1, BREADY=1, BRESP=00
Time: 345000 - WRITE RESPONSE HANDSHAKE: BRESP=00
Time: 345000 - Write completed: Addr=0x00000010, Data=0x87654321

--- Final IDLE Phase ---

=== All Transactions Completed ===
Total simulation time: 395000
```

READ TRANSACTION REPORT

Tcl Console | Messages | Log

```
--- First READ Transaction ---
Time: 145000 - Reading from address 0x00000004 (expecting 0xDEADBEEF)

--- Signal Monitor at Time: 175000 ---
MASTER STATE: 3 (   RADDR_CHANNEL)
SLAVE STATE:  0 (            IDLE)
  READ ADDR:    ARVALID=1, ARREADY=0, ARADDR=0x00000004
  READ DATA:    RVALID=0, RREADY=1, RDATA=0x00000000, RRESP=00

--- Signal Monitor at Time: 185000 ---
MASTER STATE: 3 (   RADDR_CHANNEL)
SLAVE STATE:  3 (   RADDR_CHANNEL)
  READ ADDR:    ARVALID=1, ARREADY=1, ARADDR=0x00000004
  READ DATA:    RVALID=0, RREADY=1, RDATA=0x00000000, RRESP=00
Time: 185000 - READ ADDRESS HANDSHAKE: Addr=0x00000004

--- Signal Monitor at Time: 195000 ---
MASTER STATE: 4 (   RDATA_CHANNEL)
SLAVE STATE:  4 (   RDATA_CHANNEL)
  READ DATA:    RVALID=1, RREADY=1, RDATA=0xdeadbeef, RRESP=00
  READ DATA MONITOR: RDATA=0xdeadbeef, RVALID=1, RREADY=1
Time: 195000 - READ DATA HANDSHAKE: Addr=0x00000000, Data=0xdeadbeef, RRESP=00
Time: 195000 - READ DATA AVAILABLE: RDATA=0xdeadbeef, RVALID=1, RREADY=1
Time: 195000 - SLAVE MEMORY READ: register_file[1] = 0x00000000
Time: 195000 - â€“ READ PASS: Addr=0x00000004, Expected=0xdeadbeef, Actual=0xdeadbeef
```

Tcl Console | Messages | Log

```
--- Second READ Transaction ---
Time: 245000 - Reading from address 0x00000008 (expecting 0xCAFEBABE)

--- Signal Monitor at Time: 275000 ---
MASTER STATE: 3 (   RADDR_CHANNEL)
SLAVE STATE:  0 (            IDLE)
  READ ADDR:    ARVALID=1, ARREADY=0, ARADDR=0x00000008
  READ DATA:    RVALID=0, RREADY=1, RDATA=0x00000000, RRESP=00

--- Signal Monitor at Time: 285000 ---
MASTER STATE: 3 (   RADDR_CHANNEL)
SLAVE STATE:  3 (   RADDR_CHANNEL)
  READ ADDR:    ARVALID=1, ARREADY=1, ARADDR=0x00000008
  READ DATA:    RVALID=0, RREADY=1, RDATA=0x00000000, RRESP=00
Time: 285000 - READ ADDRESS HANDSHAKE: Addr=0x00000008

--- Signal Monitor at Time: 295000 ---
MASTER STATE: 4 (   RDATA_CHANNEL)
SLAVE STATE:  4 (   RDATA_CHANNEL)
  READ DATA:    RVALID=1, RREADY=1, RDATA=0xcafebabe, RRESP=00
  READ DATA MONITOR: RDATA=0xcafebabe, RVALID=1, RREADY=1
Time: 295000 - READ DATA HANDSHAKE: Addr=0x00000000, Data=0xcafebabe, RRESP=00
Time: 295000 - READ DATA AVAILABLE: RDATA=0xcafebabe, RVALID=1, RREADY=1
Time: 295000 - SLAVE MEMORY READ: register_file[2] = 0x00000000
Time: 295000 - â€“ READ PASS: Addr=0x00000008, Expected=0xcafebabe, Actual=0xcafebabe
```

THANK YOU