ARM® AMBA®3 AHB-Lite

Overview

ARM® AMBA® Open Specification

- Open standard (No License required)
- The de facto standard for on-chip communication
- Used as on-chip interconnect for connecting and managing functional blocks in a System-on-Chip
- Promotes design re-use by defining common interface standards for SoC modules
- AMBA Family: AMBA 5, AMBA 4, AMBA 3 & AMBA 2
- AMBA 5 CHI (Coherent Hub Interface) specification is the latest addition to the AMBA (mainly used for server and networking SoCs)
- More info: http://www.arm.com/products/system-ip/amba/amba-open-specifications.php

ARM

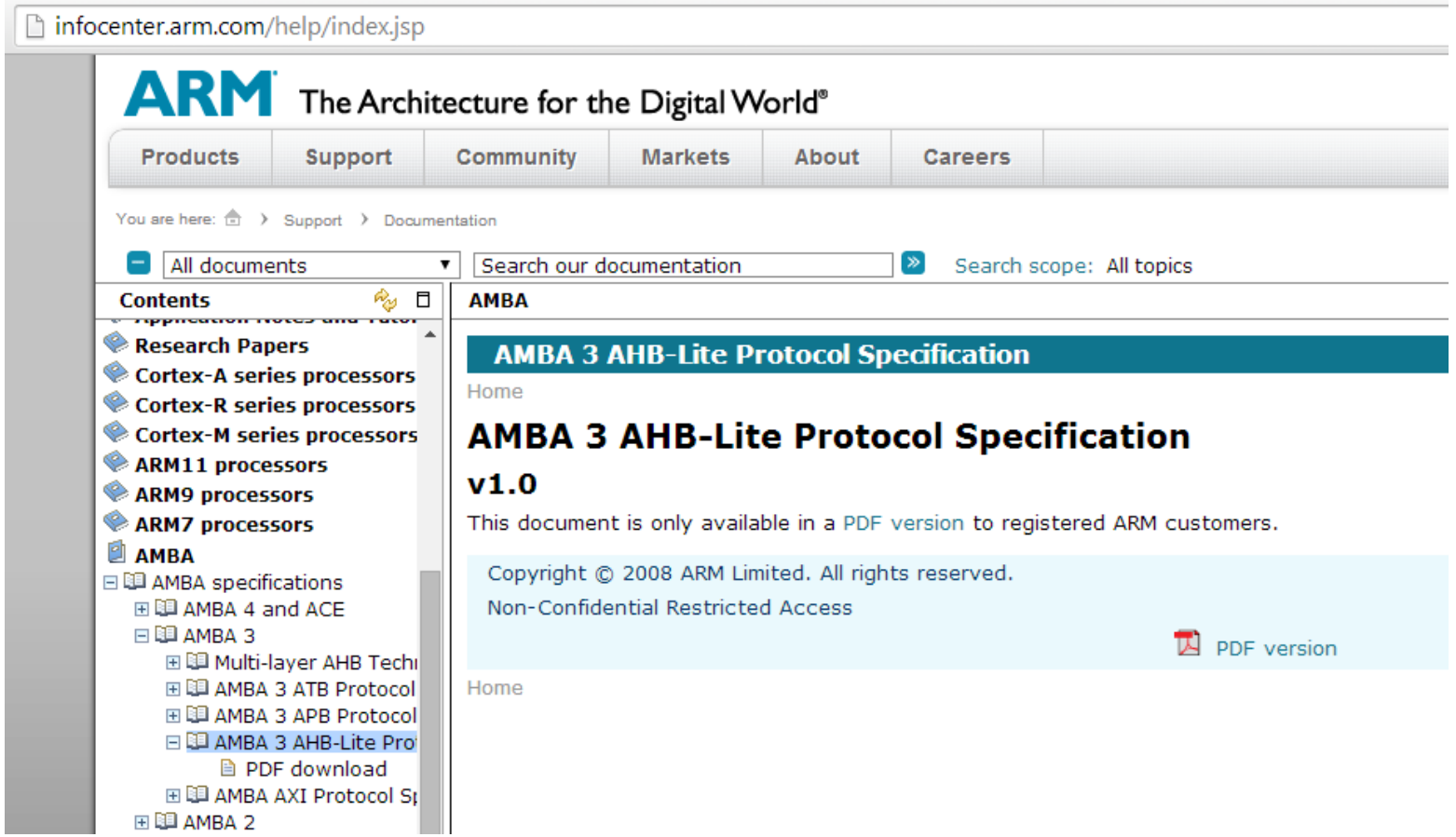
AMBA Acronyms

- Acronyms
  - AMBA® → Advanced Microcontroller Bus Architectures
  - AXI → Advanced eXtensible Interface
  - ACE → AXI Coherency Extensions
  - AHB → Advanced High-Performance Bus
  - APB → Advanced Peripheral Bus
  - ATB → Advanced Trace Bus
  - ASB → Advanced System Bus

Image Source: Google.com

How to access the full specification?

- Go to http://infocenter.arm.com/



ARM

AMBA 3 AHB-Lite

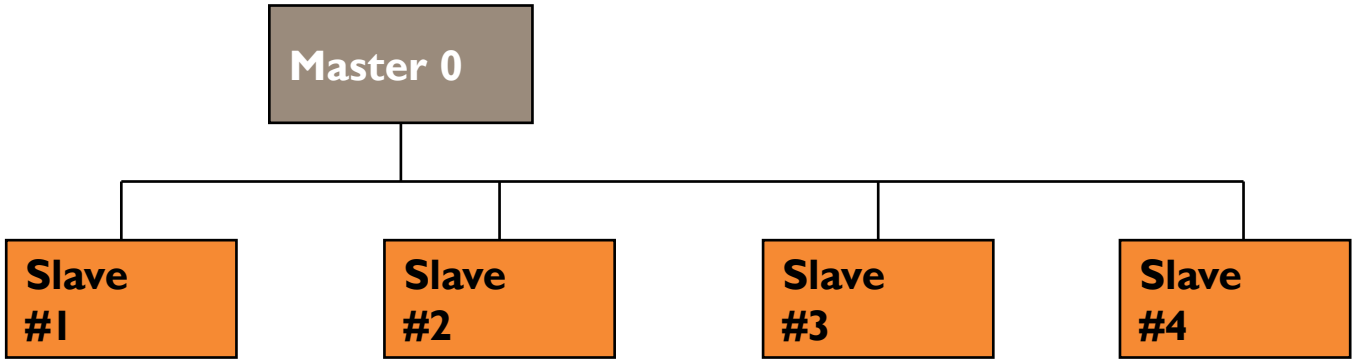
- Original AHB Specification was part of AMBA 2
- Subset of original AHB
- Reduced interconnect logic
- Simplifies slave design
- Master slave architecture
- Most of the designs have single master in the system
- Multiple masters still possible on multi-layer interconnect

AMBA 3 AHB-Lite

Image Source: Walt Disney

ARM

AMBA 3 AHB-Lite



- Single Master
- Simple slaves
- Easier module design/debug
- No arbitration issues

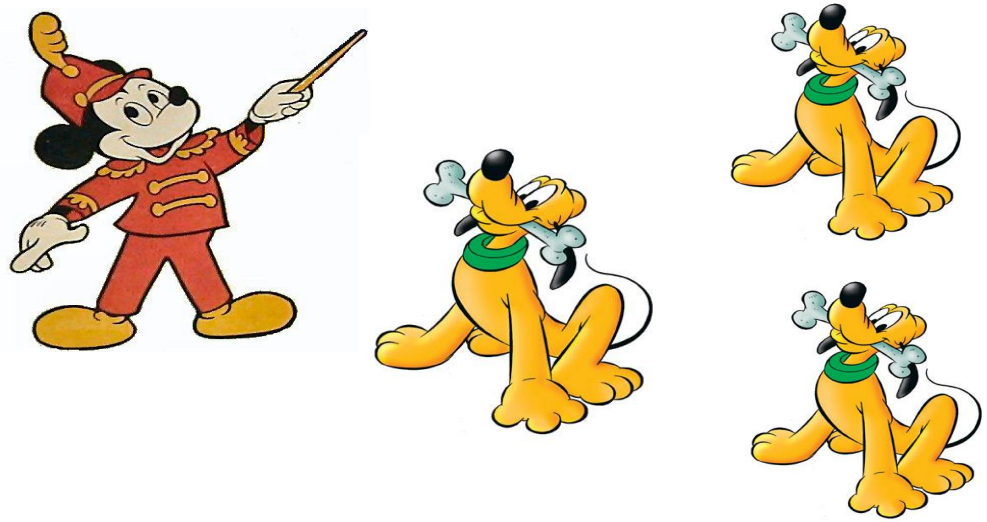


Image Source: Walt Disney

ARM

AHB-Lite transactions

- Master
  - Register Read
  - Register Write
  - Burst Read
  - Burst Write

- Slave/Peripheral
  - Can make Master wait
  - Can give error response

Image Source: Walt Disney

ARM

AHB-Lite Features

- Single Clock Edge operation
- Uni-directional busses
  - No tri-state signals
  - Good for synthesis
- Pipelined Operation

Image Source: Walt Disney

A system based on AHB-Lite

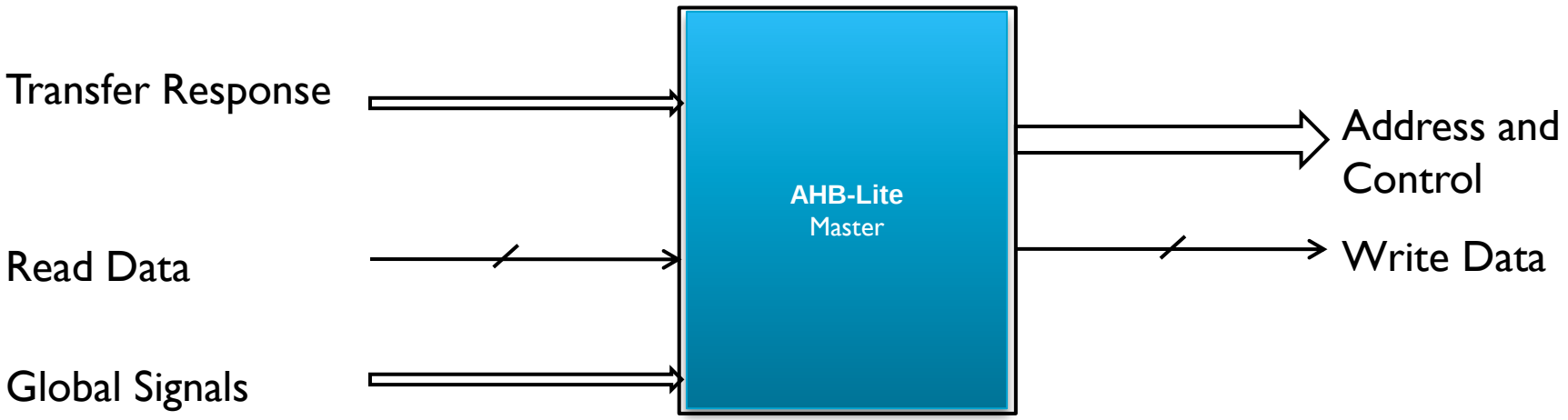
ARM

Components of AHB-Lite System

- Master
- Slaves/Peripherals
- Address Decoder
- Multiplexor

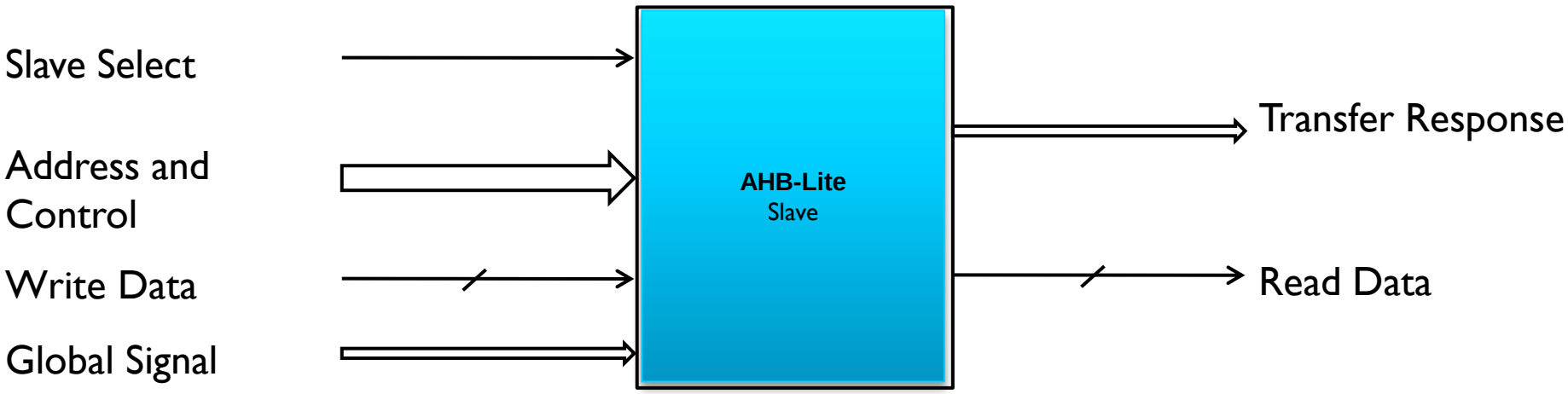
} **AHB-Lite Interconnect**

AHB-Lite Master



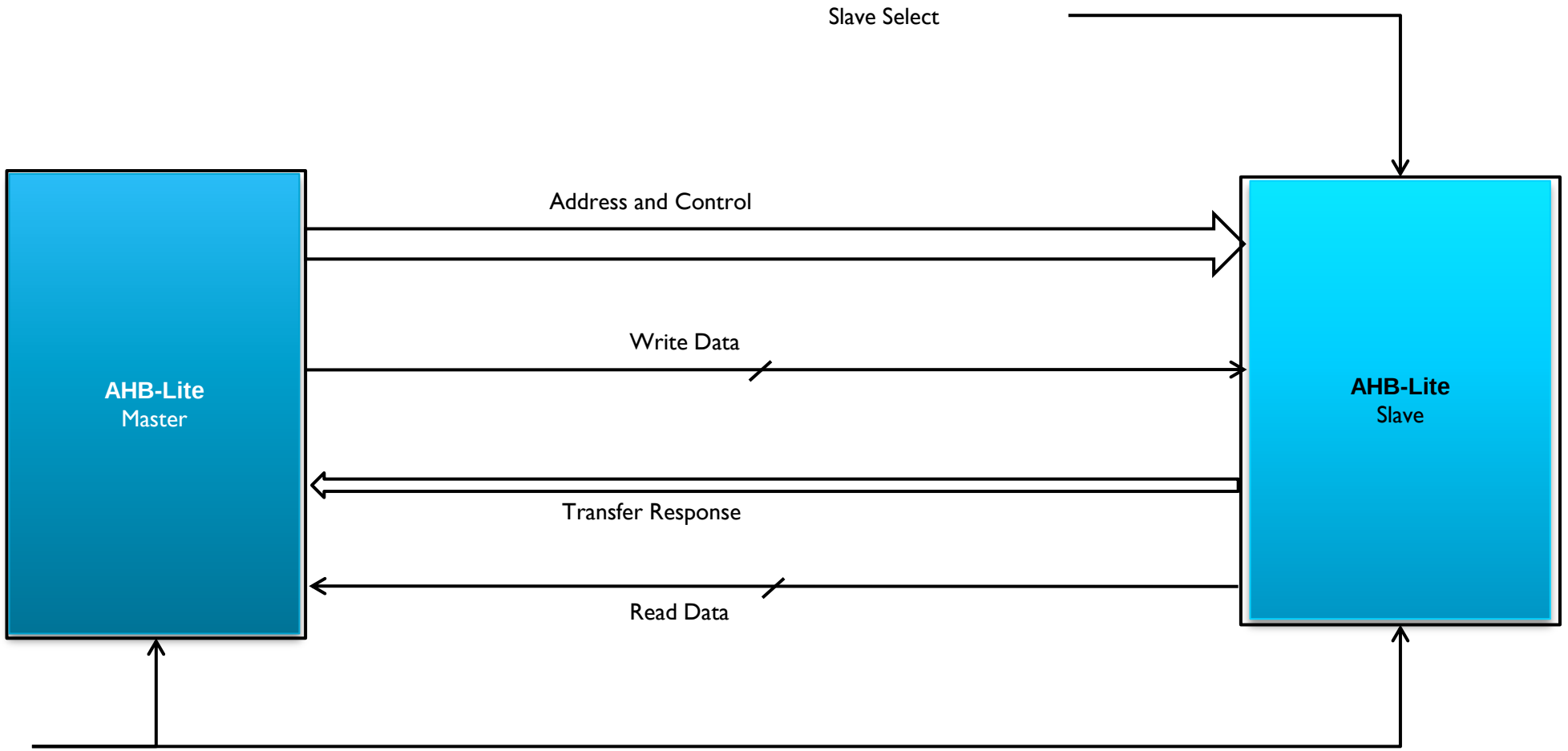
ARM

AHB-Lite Slave



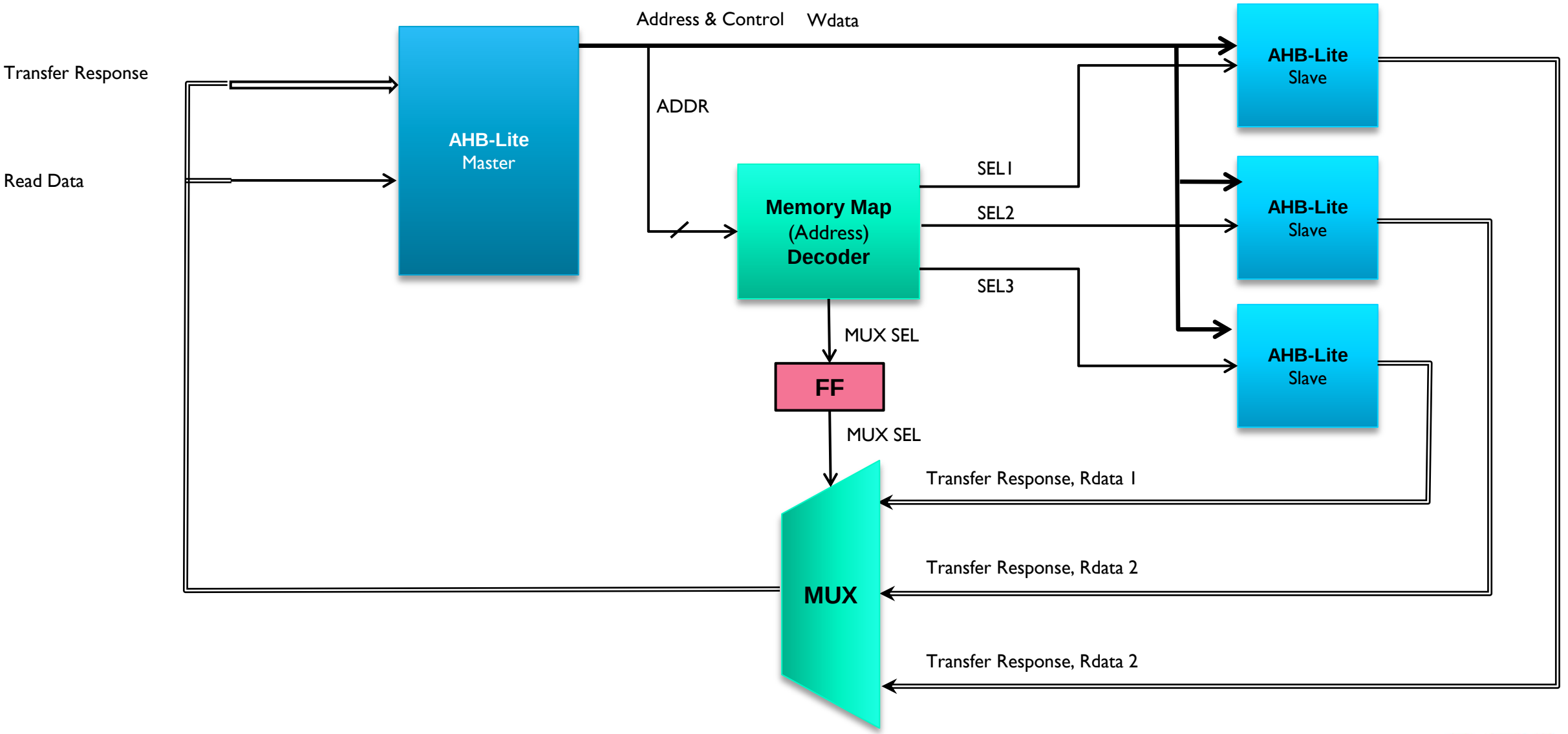
ARM

AHB-Lite Master & Slave



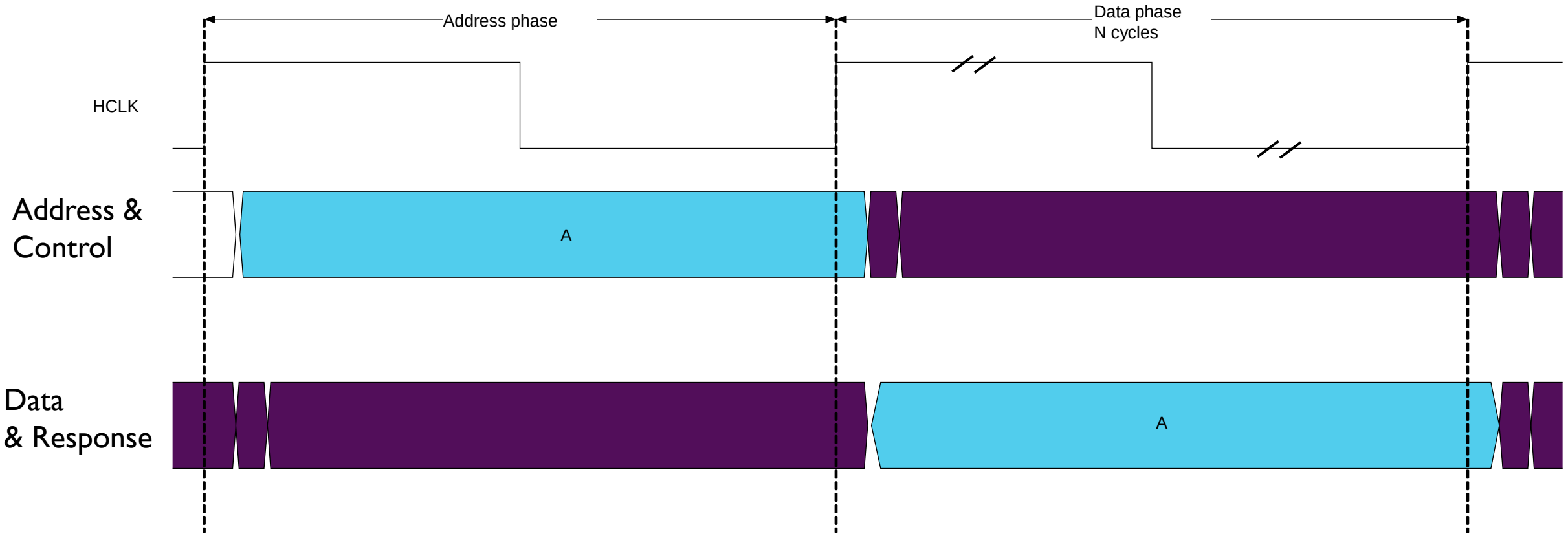
ARM

Decoder & MUX



ARM

Pipelined Transactions (Conceptual Level)

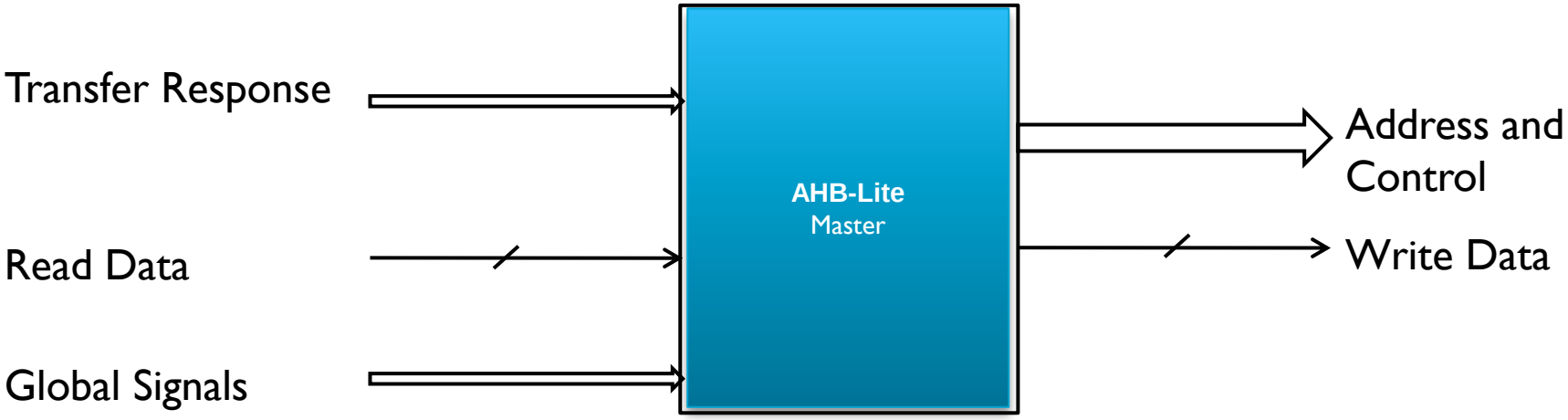


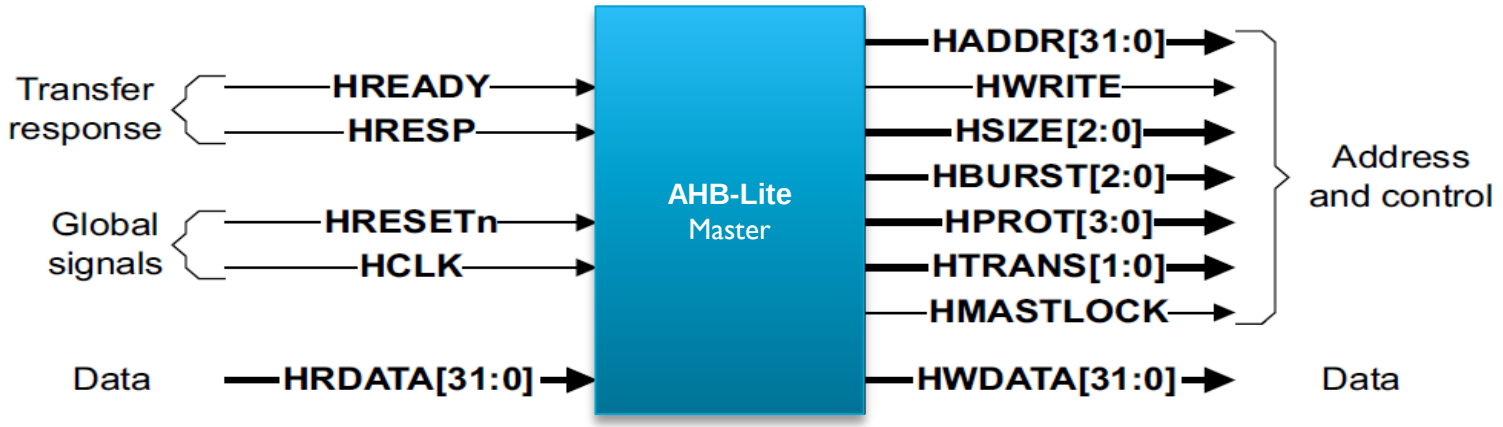
ARM

AHB-Lite Signals

ARM

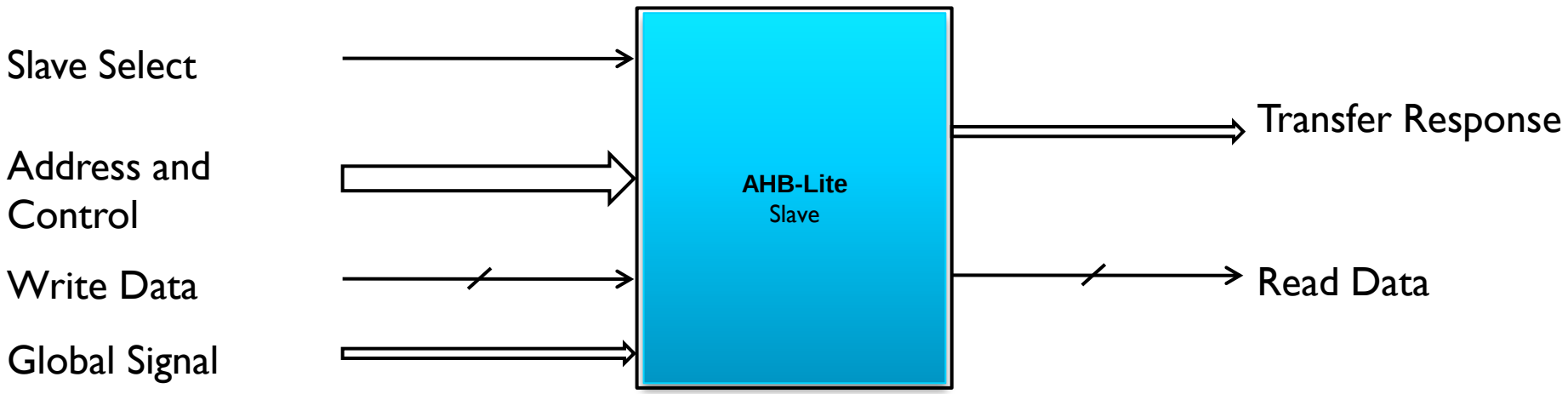
AHB-Lite Master Signals

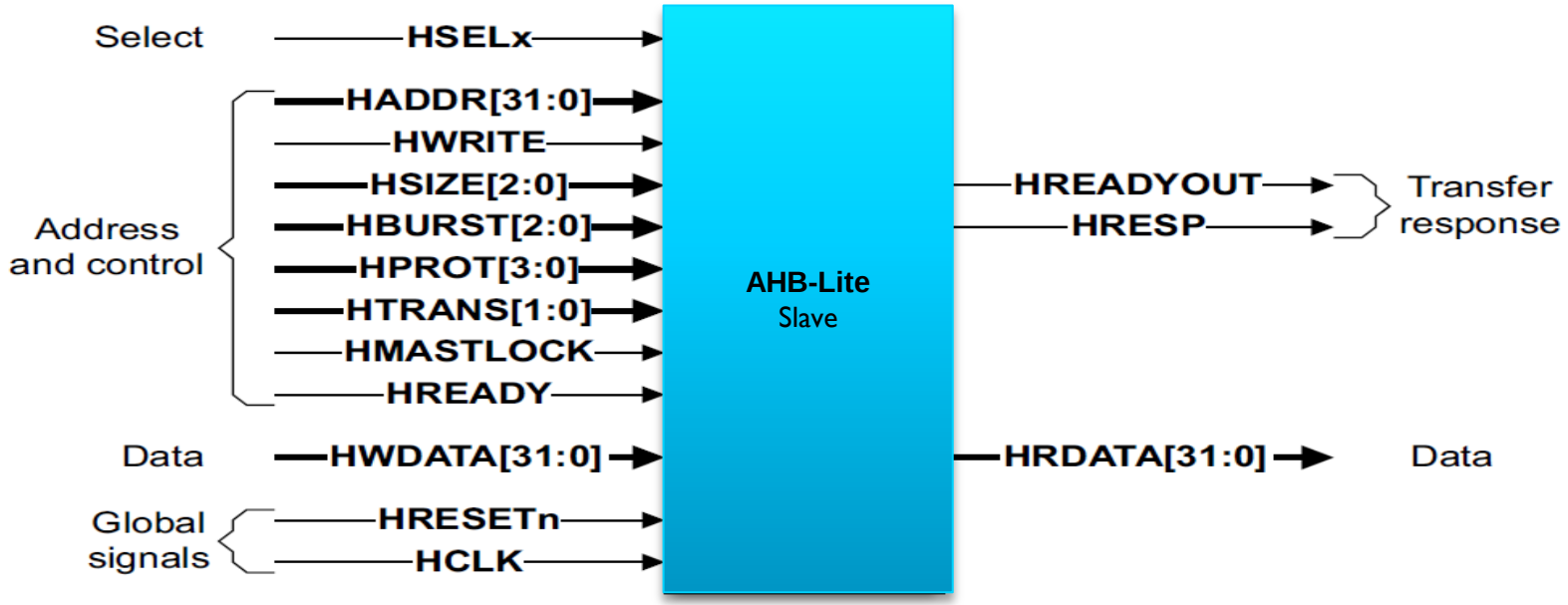




ARM

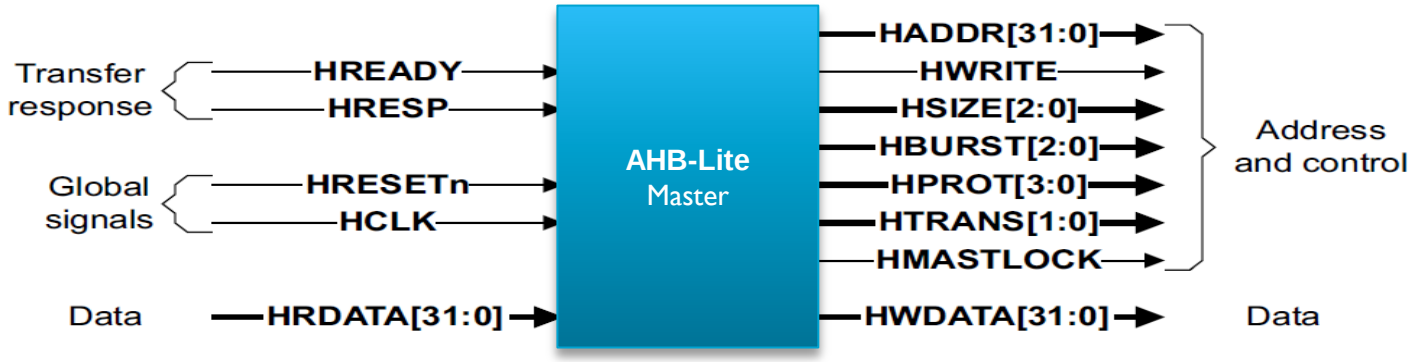
AHB-Lite Slave Signals

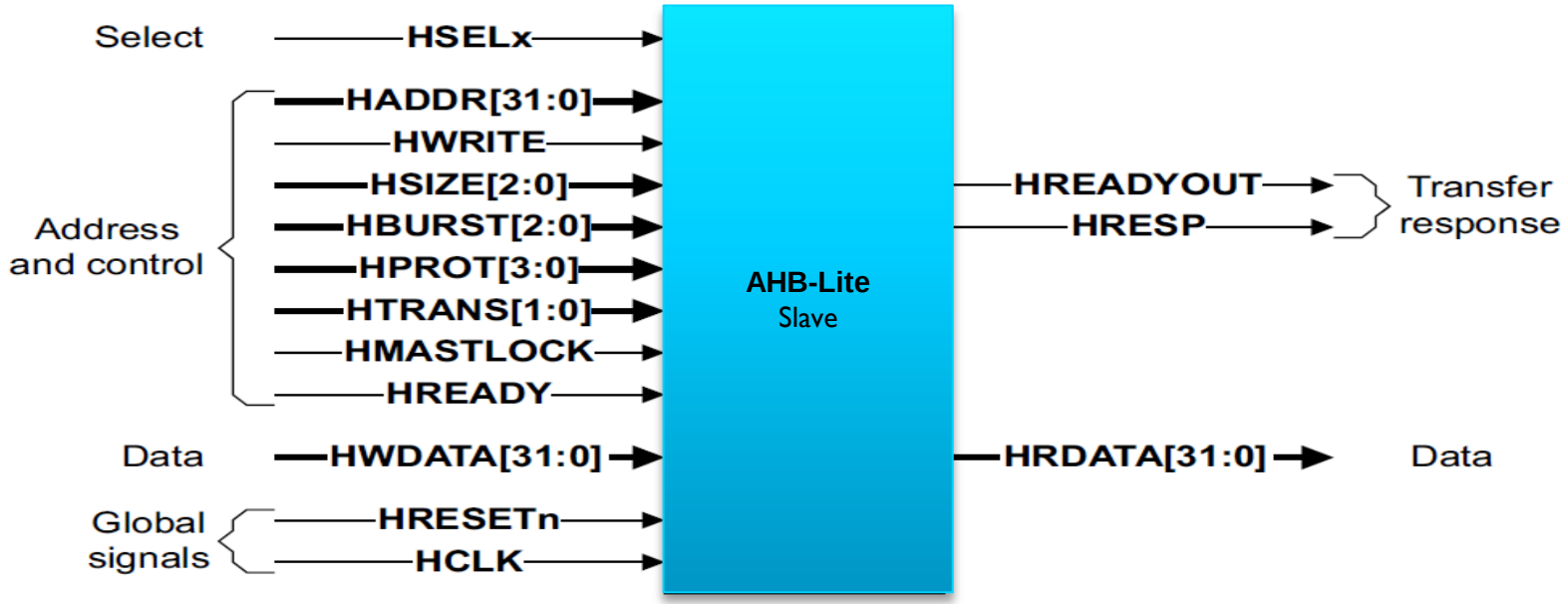




ARM

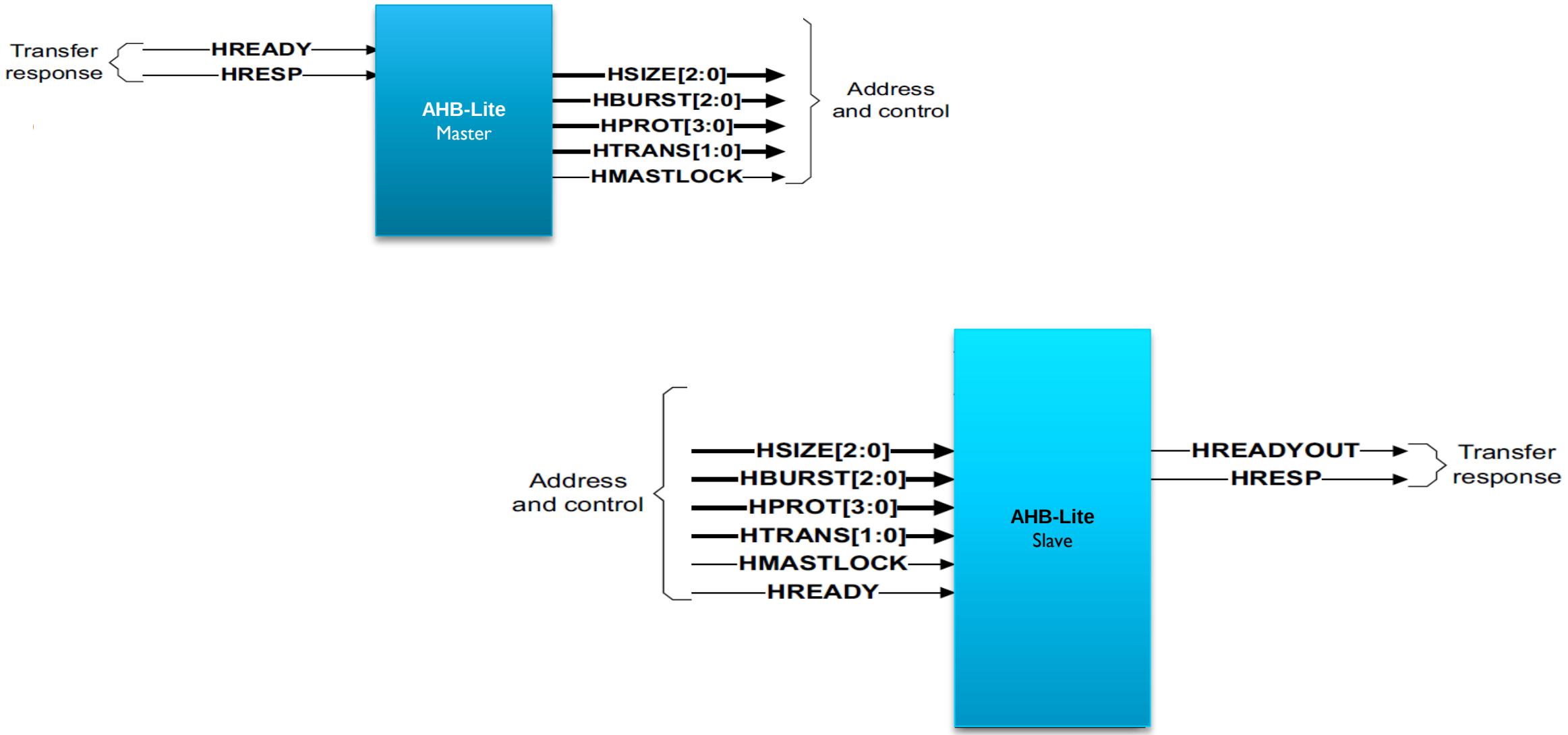
AHB-Lite Master & Slave





ARM

AHB-Lite Master & Slave

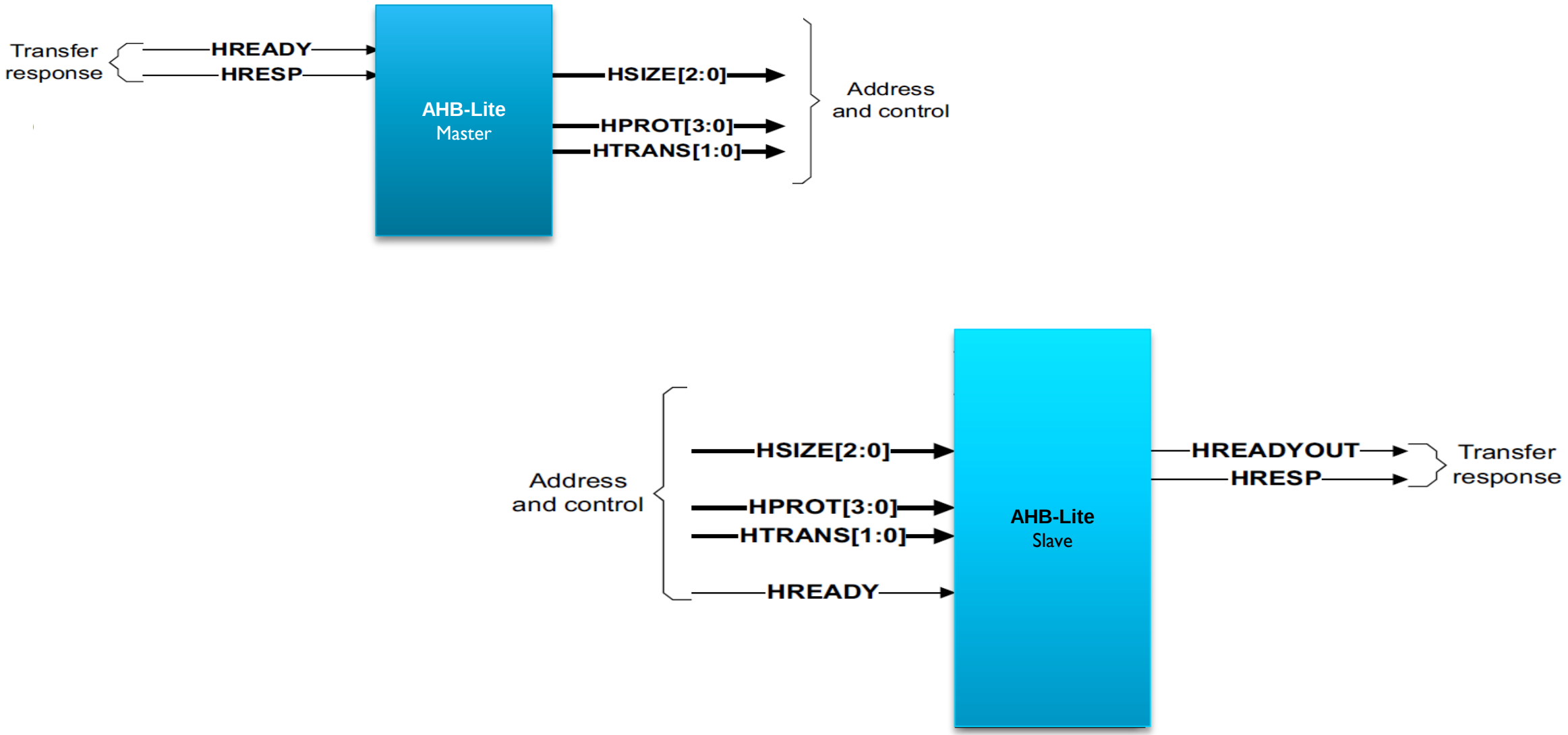


ARM

Cortex M0 doesn't speak the entire language !

- Cortex M0 does not support BURST transaction
  - HBURST[2:0] is always 3'b000
- Cortex M0 does not support locked transactions
  - HMASTLOCK is always 1'b0
- Cortex M0 issues only non-sequential transfers
  - HTRANS[1:0] is either 2'b00 (IDLE) or 2'b10 (Non Sequential)

AHB-Lite Master & Slave



ARM

HTRANS[1:0]

HTRANS	Type	Description
00	IDLE	Master does not wish to perform a transfer
01	BUSY	Bus Master is in the middle of a burst but cannot immediately continue with the next transfer
10	NON-SEQ	Indicates the first transfer of a burst or a single transfer
11	SEQ	The remaining transfers in the burst are sequential address steps from the previous transfer. Step size is that of data width of transfer (which is shown by HSIZE)

Cortex M0 Always generates NON-SEQ Transactions

ARM

HSIZE[1:0]

Address-phase:		Data-phase:			
HSIZE [1:0]	HADDR [1:0]	HxDATA [31:24]	HxDATA [23:16]	HxDATA [15:8]	HxDATA [7:0]
00	00	-	-	-	Rd[7:0]
00	01	-	-	Rd[7:0]	-
00	10	-	Rd[7:0]	-	-
00	11	Rd[7:0]	-	-	-
01	00	-	-	Rd[15:8]	Rd[7:0]
01	10	Rd[15:8]	Rd[7:0]	-	-
10	00	Rd[31:24]	Rd[23:16]	Rd[15:8]	Rd[7:0]

ARM

HPROT[3:0] Protection Signal Encoding

HPROT[3] Cacheable	HPROT[2] Bufferable	HPROT[1] Privileged	HPROT[0] Data/Opcode	Description
-	-	-	0	Opcode fetch
-	-	-	1	Data access
-	-	0	-	User access
-	-	1	-	Privileged access
-	0	-	-	Non-bufferable
-	1	-	-	Bufferable
0	-	-	-	Non-cacheable
1	-	-	-	Cacheable

Transactions

Transaction	Access
HTRANS[1:0] = 2'b00	IDLE
HTRANS[1:0] = 2'b10 HPROT[0] = 1'b0 HSIZE[1:0] = 2'b10 HWRITE = 1'b0	FETCH

Instruction Fetch

Data Access

Transaction	Access
HTRANS[1:0] = 2'b10 HPROT[0] = 1'b1 HSIZE[1:0] = 2'b00	BYTE
HTRANS[1:0] = 2'b10 HPROT[0] = 1'b1 HSIZE[1:0] = 2'b01	HALF-WORD
HTRANS[1:0] = 2'b10 HPROT[0] = 1'b1 HSIZE[1:0] = 2'b10	WORD

Control Signals Recap

HTRANS[1:0]

IDLE
BUSY
NONSEQ
SEQ

HBURST[2:0]

SINGLE
INCR
WRAP[4|8|16]
INCR[4|8|16]

HMASTLOCK

UNLOCKED
LOCKED

HSIZE[2:0]

Byte
Halfword
Word
Doubleword
...

HPROT[3:0]

Data/Opcode
Privileged/user
Bufferable
Cacheable

ARM

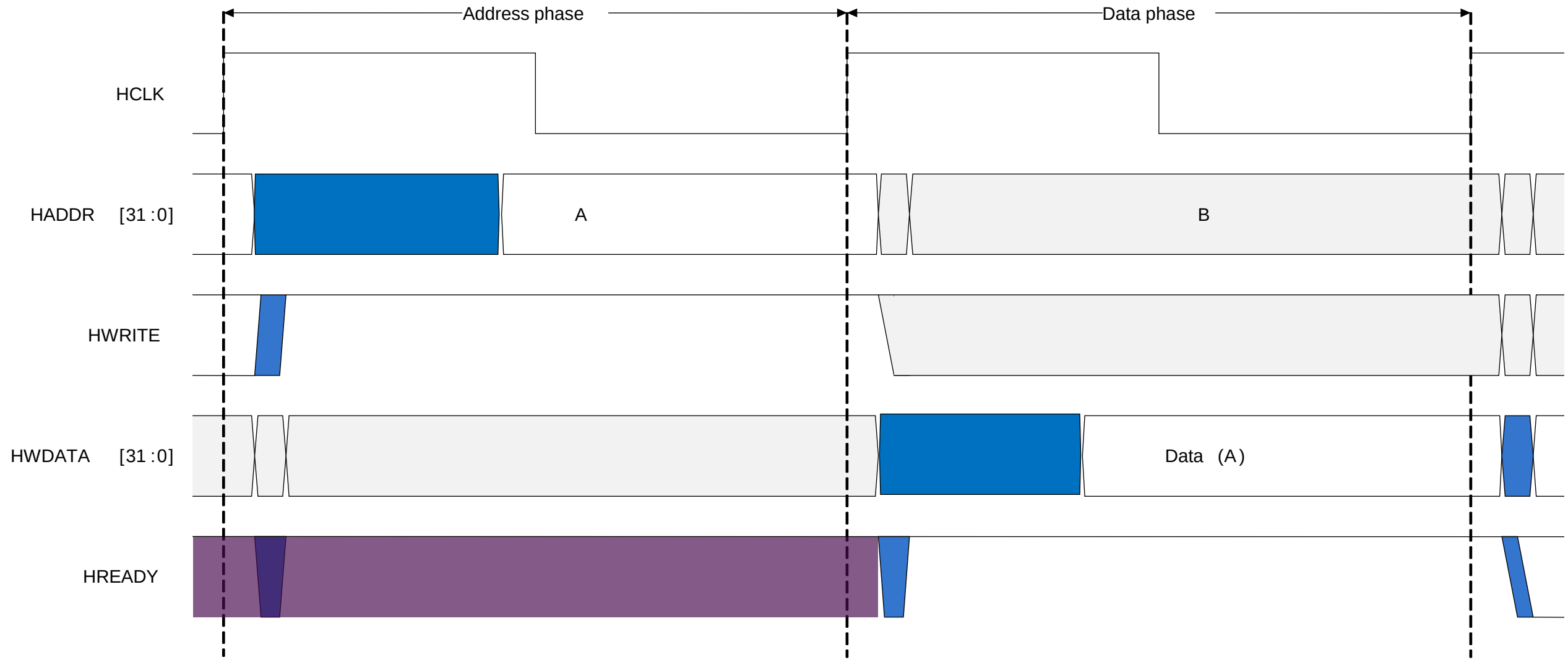
Transfer Response Signals

HREADYOUT	Multiplexor	When HIGH, the **HREADYOUT** signal indicates that a transfer has finished on the bus. This signal can be driven LOW to extend a transfer.
HRESP	Multiplexor	The transfer response, after passing through the multiplexor, provides the master with additional information on the status of a transfer. When LOW, the **HRESP** signal indicates that the transfer status is OKAY. When HIGH, the **HRESP** signal indicates that the transfer status is ERROR.

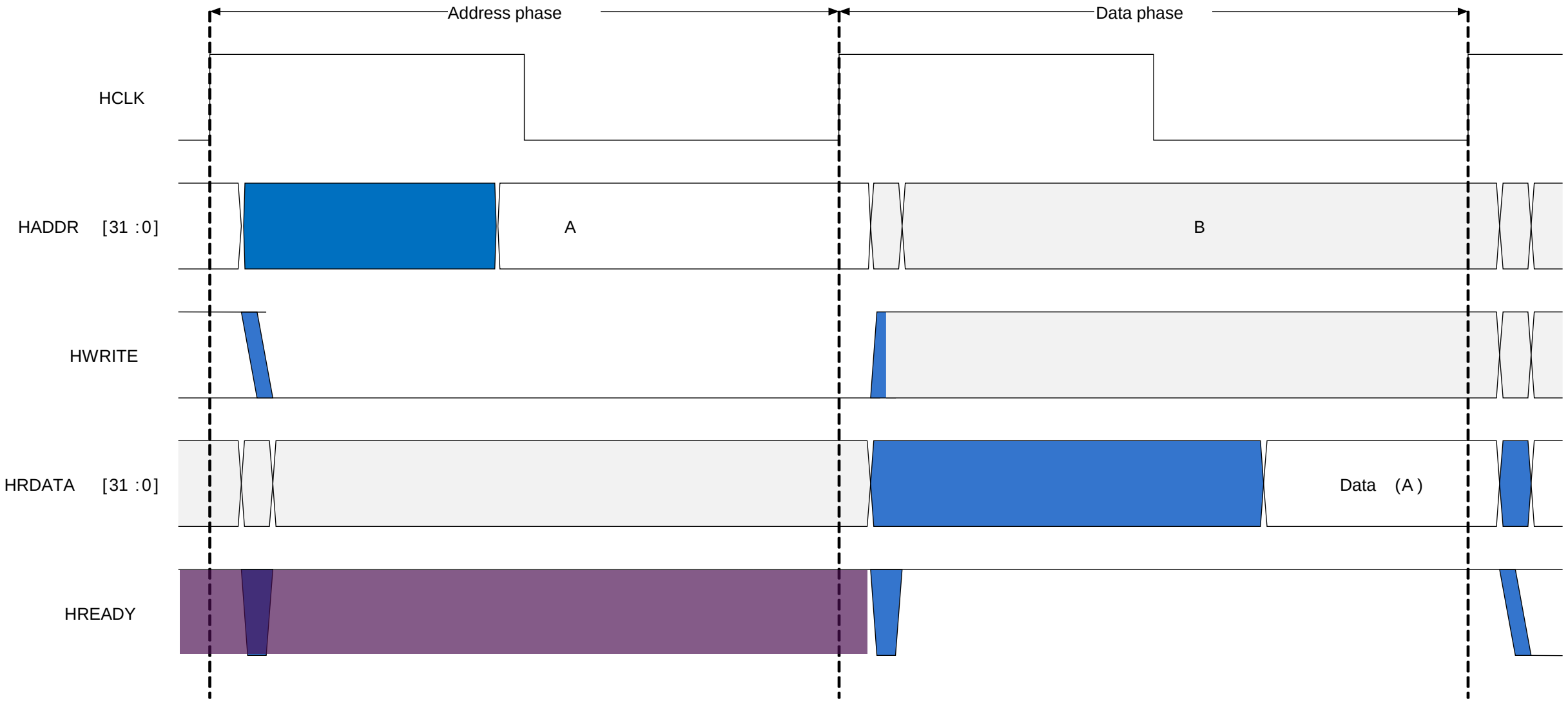
AHB-Lite Transactions

ARM

Basic transfer - Write

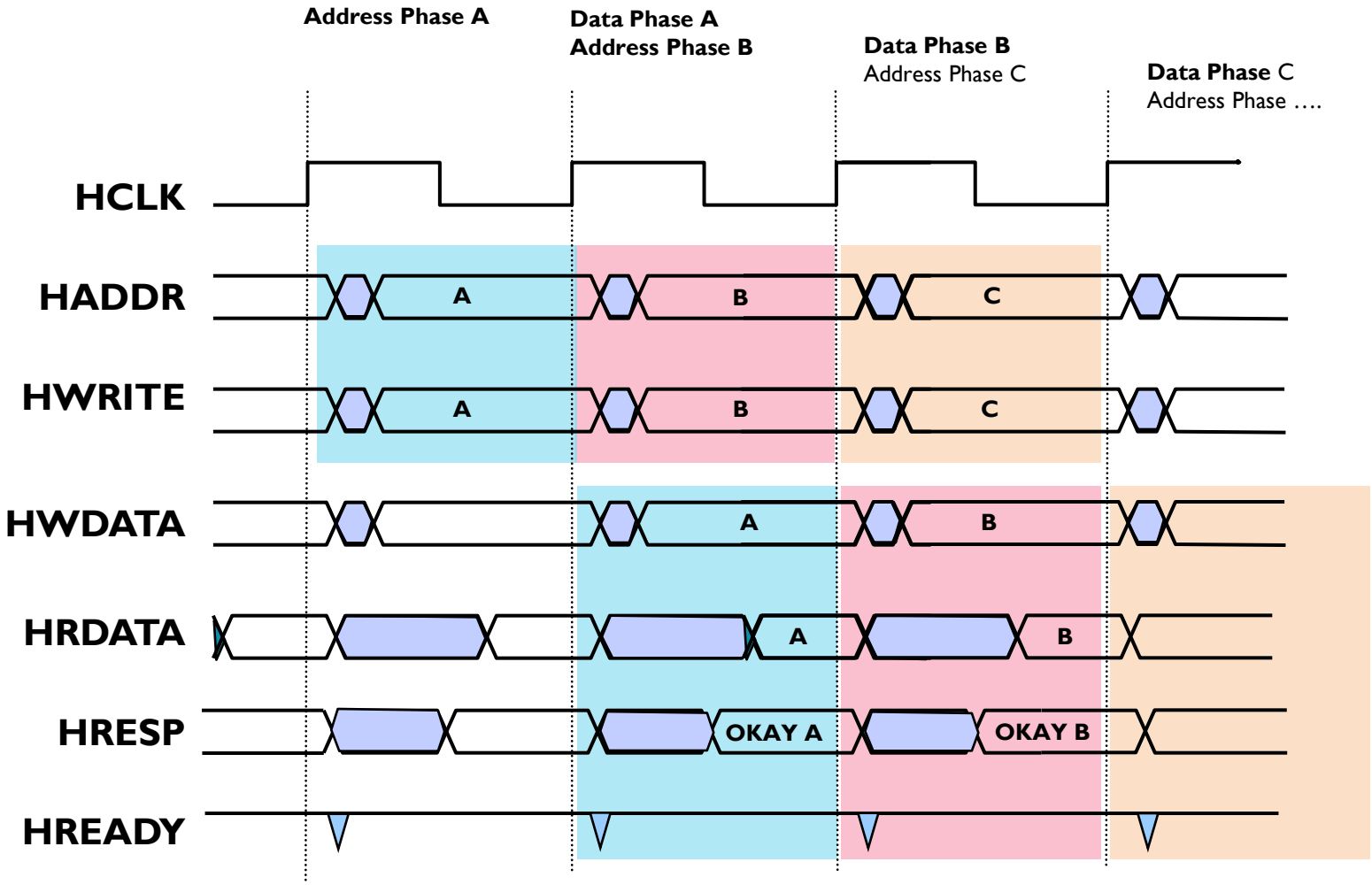


Basic transfer - Read



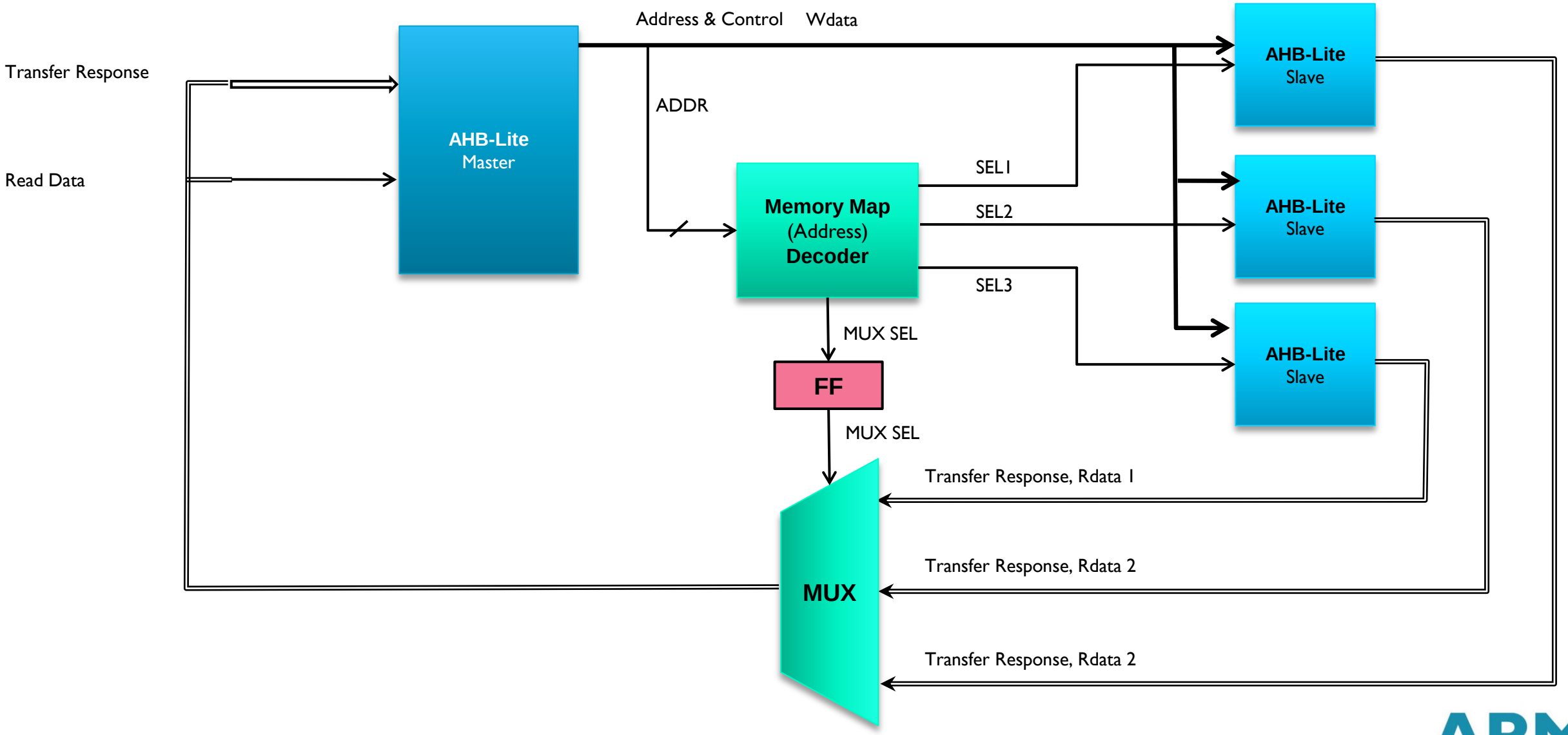
ARM

AHB Pipelined Transaction



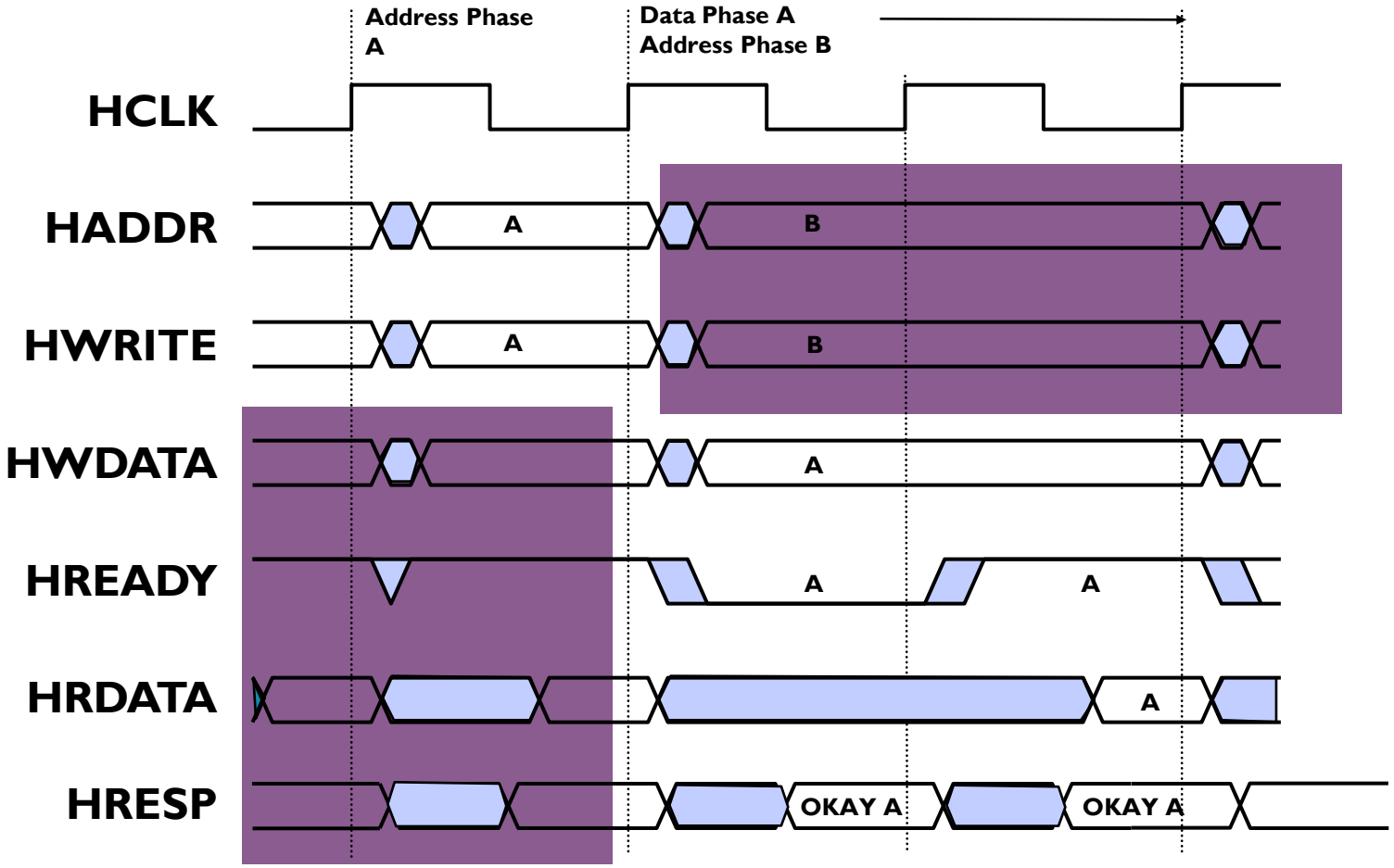
ARM

Pipelined Operation



ARM

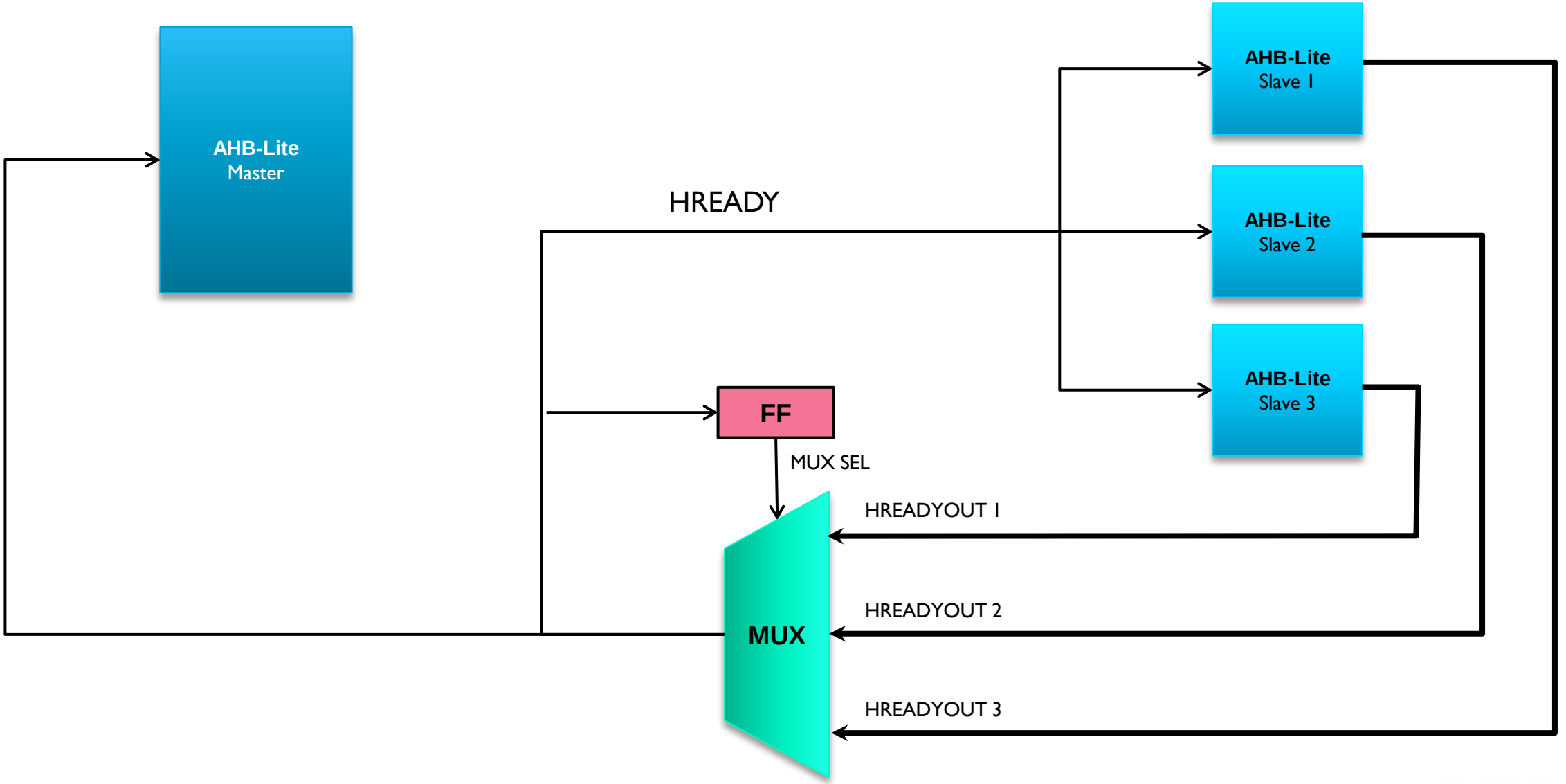
AHB basic signal timing – Adding wait states



Master will extend Address Phase B

ARM

HREADY (Inform all)



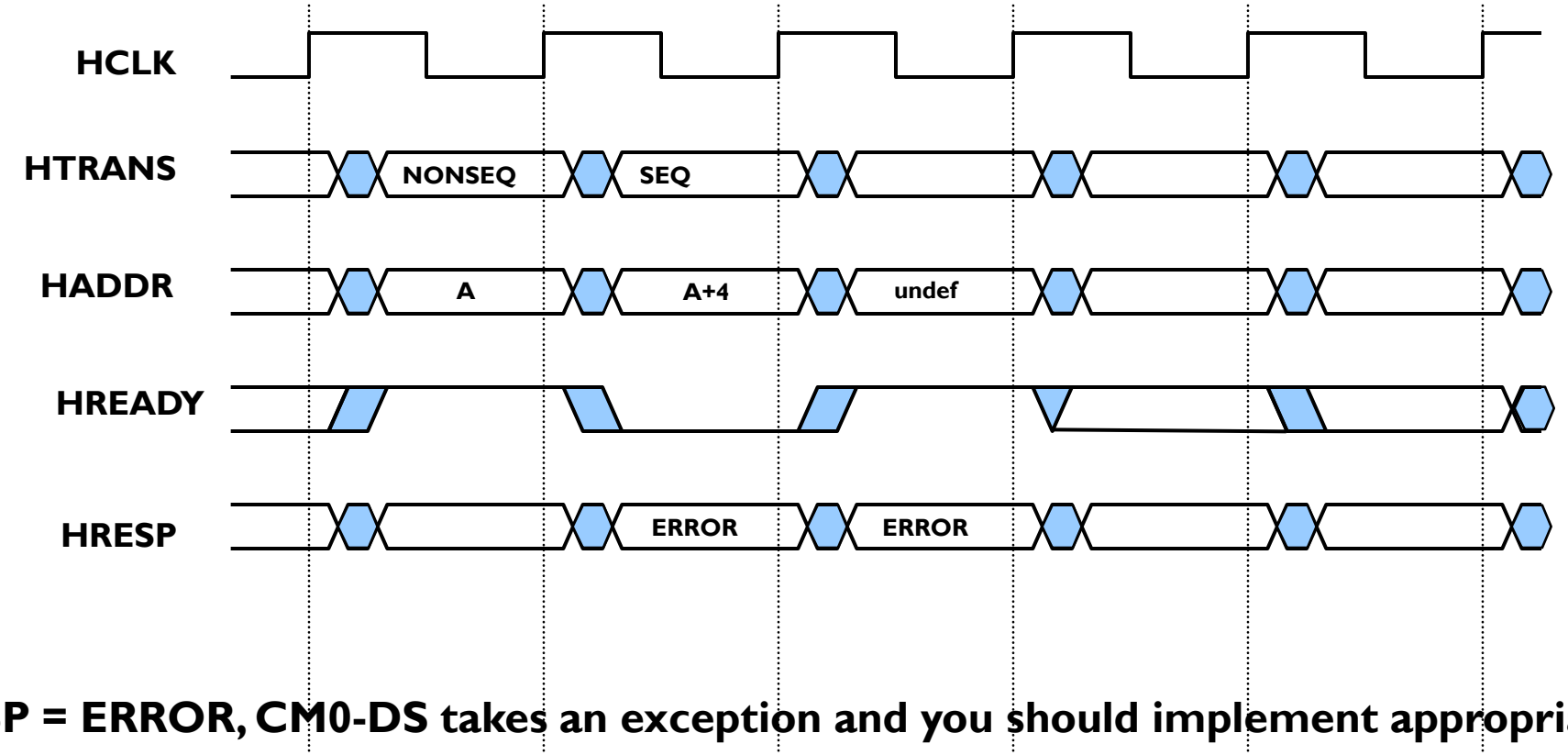
ARM

HRESP – Slave Response

HRESP	Event	Bus Master operation
OKAY	Access completed normally	
ERROR	Slave aborts access, (2 cycle response)	Master has option of continuing or terminating a burst containing an ERROR

It is permissible to continuously drive HRESP Low in a system which does not wish to generate any errors.

ERROR Response



- **If HRESP = ERROR, CM0-DS takes an exception and you should implement appropriate exception handler to catch the error**

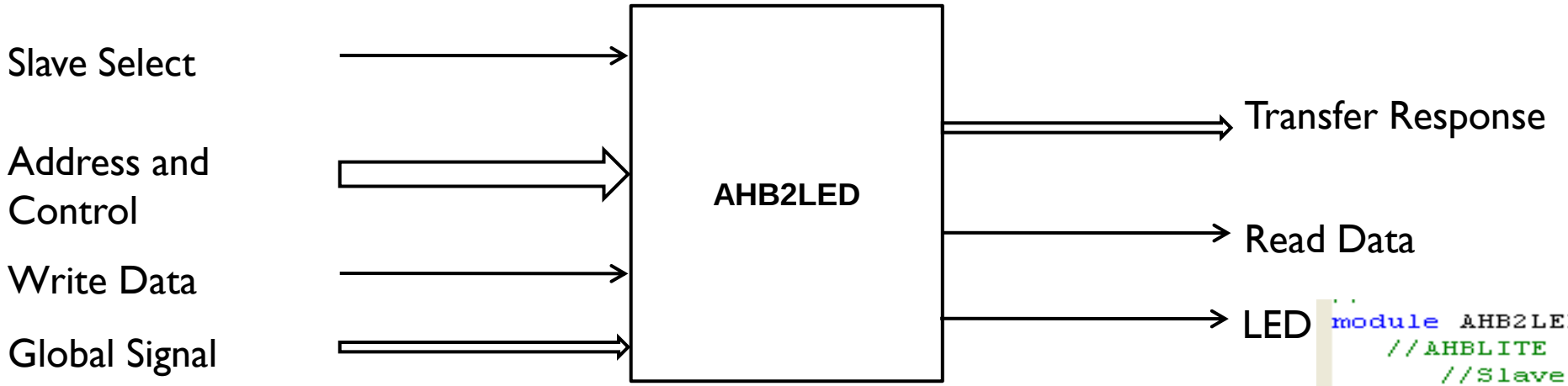
ARM

A simple AHB-Lite Slave

AHB2LED.v

ARM

AHB2LED TOP LEVEL



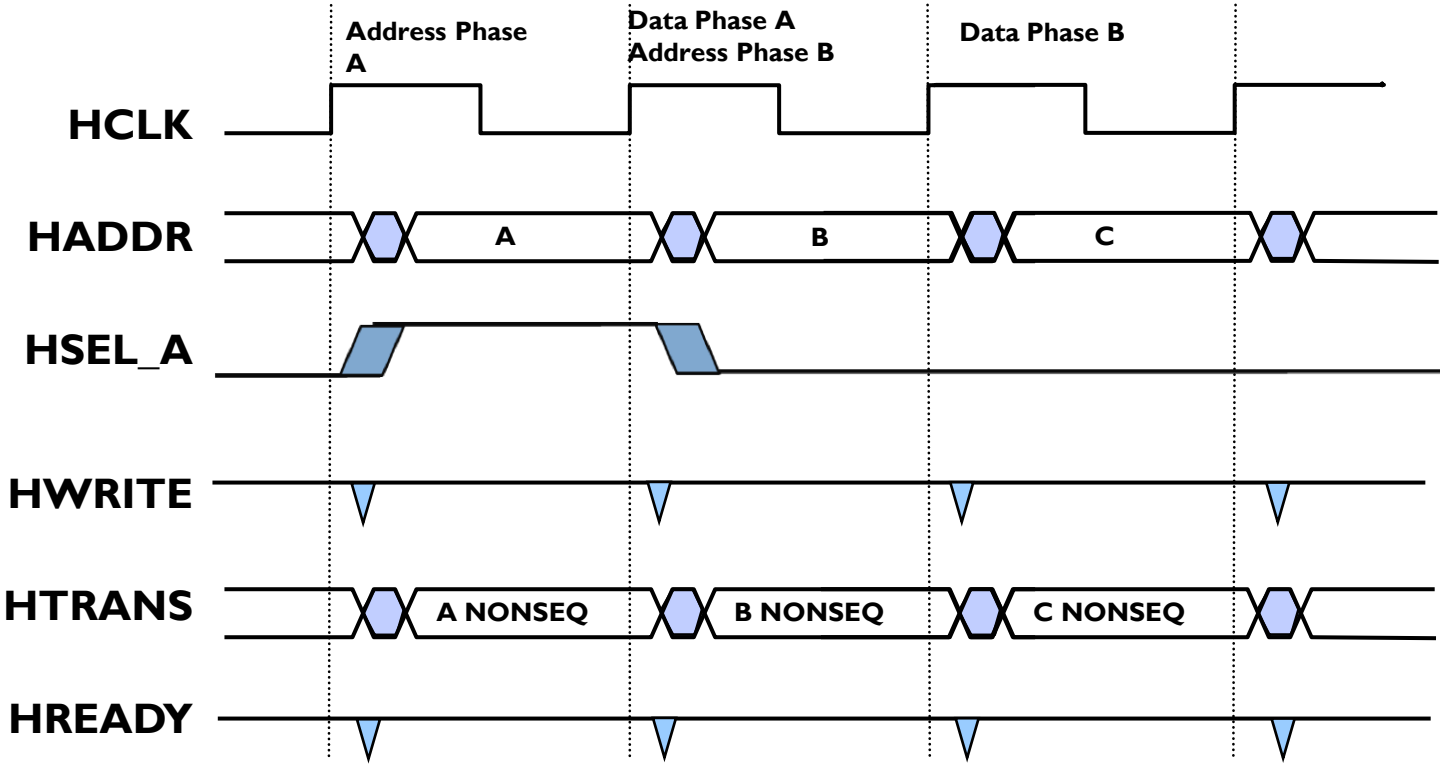

```
module AHB2LED(
    //AHBLITE INTERFACE
        //Slave Select Signals
            input wire HSEL,
        //Global Signal
            input wire HCLK,
            input wire HRESETn,
        //Address, Control & Write Data
            input wire HREADY,
            input wire [31:0] HADDR,
            input wire [1:0] HTRANS,
            input wire HWRITE,
            input wire [2:0] HSIZE,

            input wire [31:0] HWDATA,
        // Transfer Response & Read Data
            output wire HREADYOUT,
            output wire [31:0] HRDATA,

    //LED Output
            output reg [7:0] LED
);
```


ARM

Sampling Address & Control

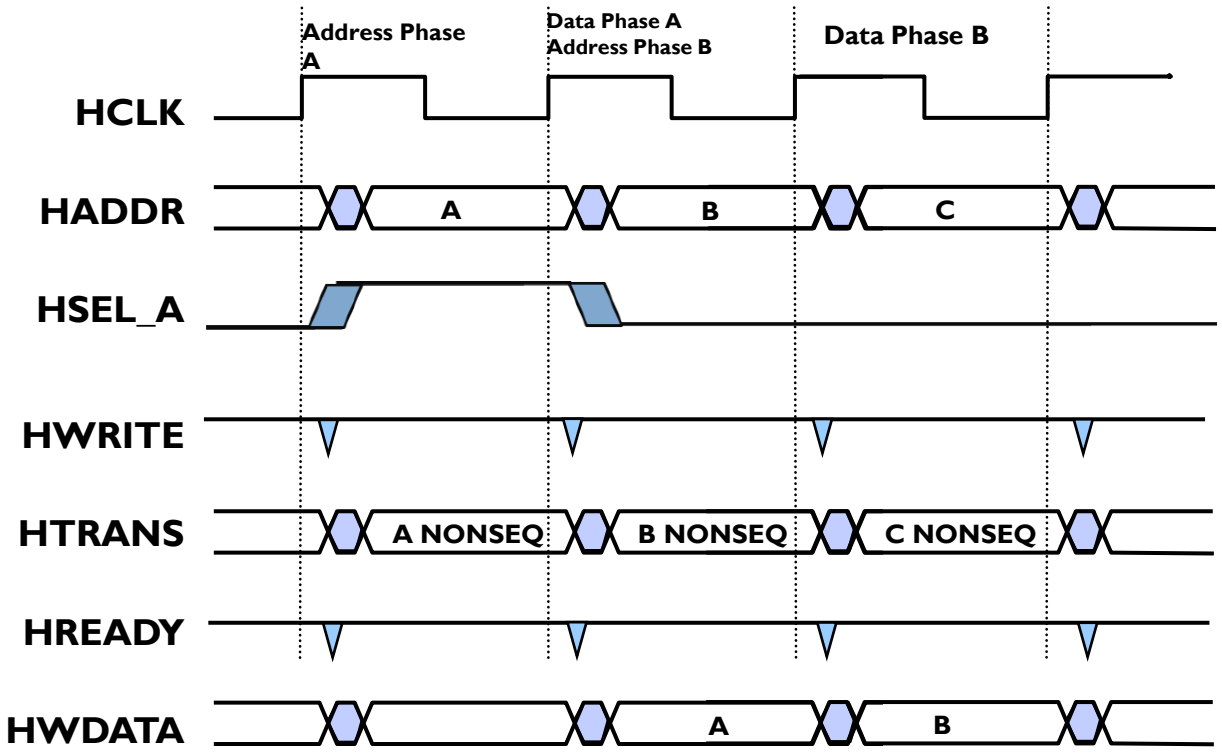



```
//Address Phase Sampling Registers
  reg rHSEL;
  reg [31:0] rHADDR;
  reg [1:0] rHTRANS;
  reg rHWRITE;
  reg [2:0] rHSIZE;


//Address Phase Sampling
  always @(posedge HCLK or negedge HRESETn)
  begin
    if(!HRESETn)
    begin
      rHSEL     <= 1'b0;
      rHADDR    <= 32'h0;
      rHTRANS   <= 2'b00;
      rHWRITE   <= 1'b0;
      rHSIZE    <= 3'b000;
    end
    else if(HREADY)
    begin
      rHSEL     <= HSEL;
      rHADDR    <= HADDR;
      rHTRANS   <= HTRANS;
      rHWRITE   <= HWRITE;
      rHSIZE    <= HSIZE;
    end
  end
```


ARM

Sampling Address & Control




```
//Address Phase Sampling Registers
  reg rHSEL;
  reg [31:0] rHADDR;
  reg [1:0] rHTRANS;
  reg rHWRITE;
  reg [2:0] rHSIZE;


//Address Phase Sampling
  always @(posedge HCLK or negedge HRESETn)
  begin
    if(!HRESETn)
    begin
      rHSEL     <= 1'b0;
      rHADDR    <= 32'h0;
      rHTRANS   <= 2'b00;
      rHWRITE   <= 1'b0;
      rHSIZE    <= 3'b000;
    end
    else if(HREADY)
    begin
      rHSEL     <= HSEL;
      rHADDR    <= HADDR;
      rHTRANS   <= HTRANS;
      rHWRITE   <= HWRITE;
      rHSIZE    <= HSIZE;
    end
  end
```

```
67  //Data Phase data transfer
68    always @(posedge HCLK or negedge HRESETn)
69    begin
70      if(!HRESETn)
71        LED <= 8'b0000_0000;
72      else if(rHSEL & rHWRITE & rHTRANS[1])
73        LED <= HWDATA[7:0];
74    end
75
76  //Transfer Response
77    assign HREADYOUT = 1'b1; //Single cycle Write & Read. Zero Wait state operations
78
79  //Read Data
80    assign HRDATA = {24'h0000_00,LED};
```


ARM

Lab

- Analyse the AHB2LED.v file provided
- In the next lab, we will look into system integration, simulation and implementation of a complete AHB-Lite System using Cortex M0 Design Start core