Advanced eXtensible Interface

- 作为 AMBA 3.0 的一部分
- 发展历史

Change history

Date	Issue	Confidentiality	Change
16 June 2003	A	Non-Confidential	First release
19 March 2004	B	Non-Confidential	First release of V1.0
03 March 2010	C	Non-Confidential	First release of V2.0

Key Feature

- 分开的地址/控制和数据阶段
- 使用 byte stroes 支持unaligned 数据传输
- 基于 burst 的传输，只提供传输开始地址
- 可以发出多个未决的地址
- 乱序完成
- 可以很容易添加寄存器缓存帮助收敛时序

信号定义

- 所有的信号根据完成的功能分成几个channel
- 每个 channel 之间通过 VALID/READY 来握手，每个channel 都可以独立传输

Channel 介绍

- Read and write address channels(RA/WA)
  读/写分别有自己的 address channel 主要传递的有：请求的地址，传输的类型，长度，字长等
- Read data channel (R)
  传送从 slave 到 master 读取的数据及状态
- Write data channel (W)
  传递从 Master 到 Slave 写入的数据

Channel 介绍

- Write response channel (B)
  对于读传输，读状态通过 read data channel 来传送，对于写传输，通过这个 channel 来报告完成状态

Read transfer using channels

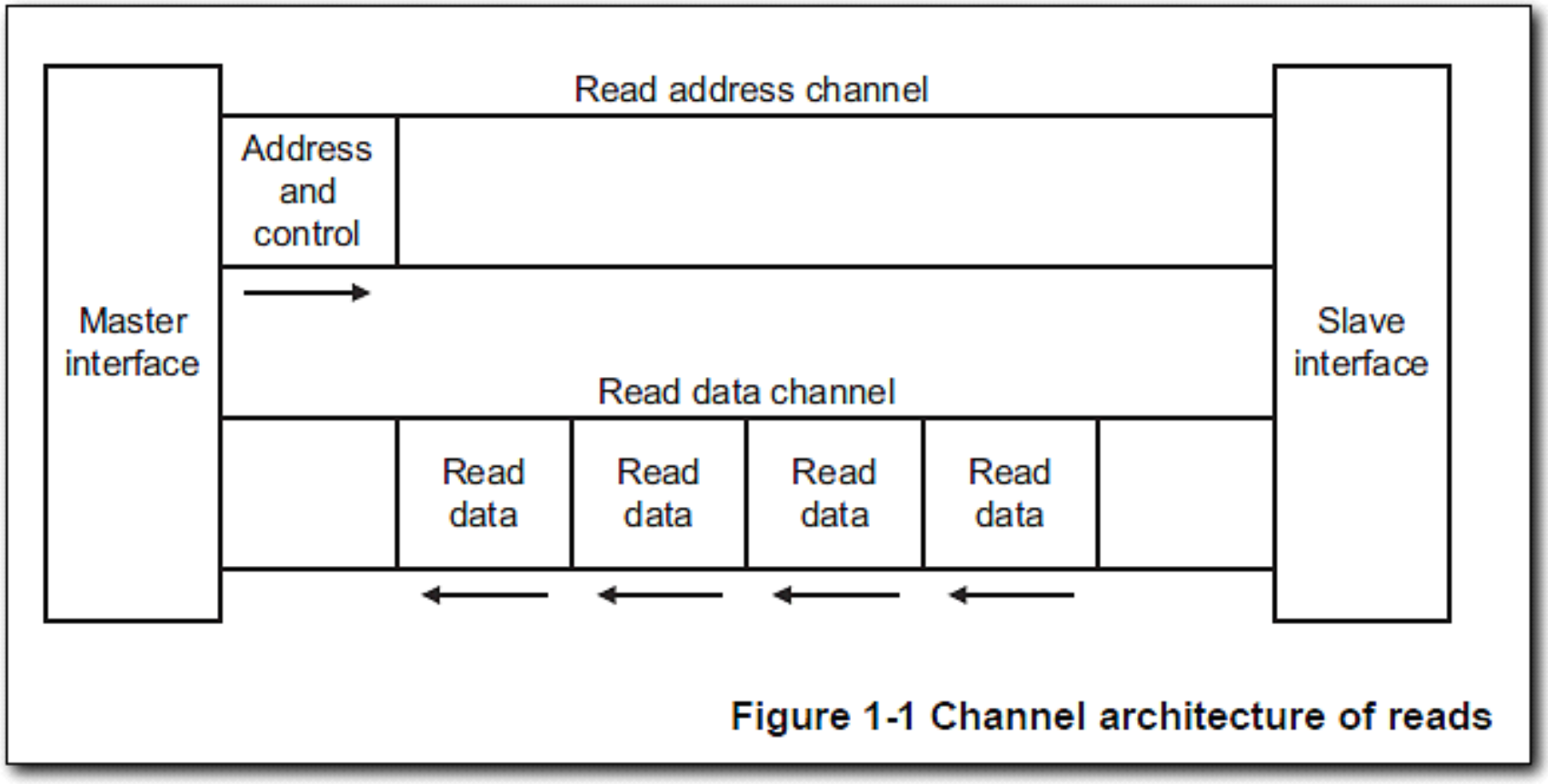
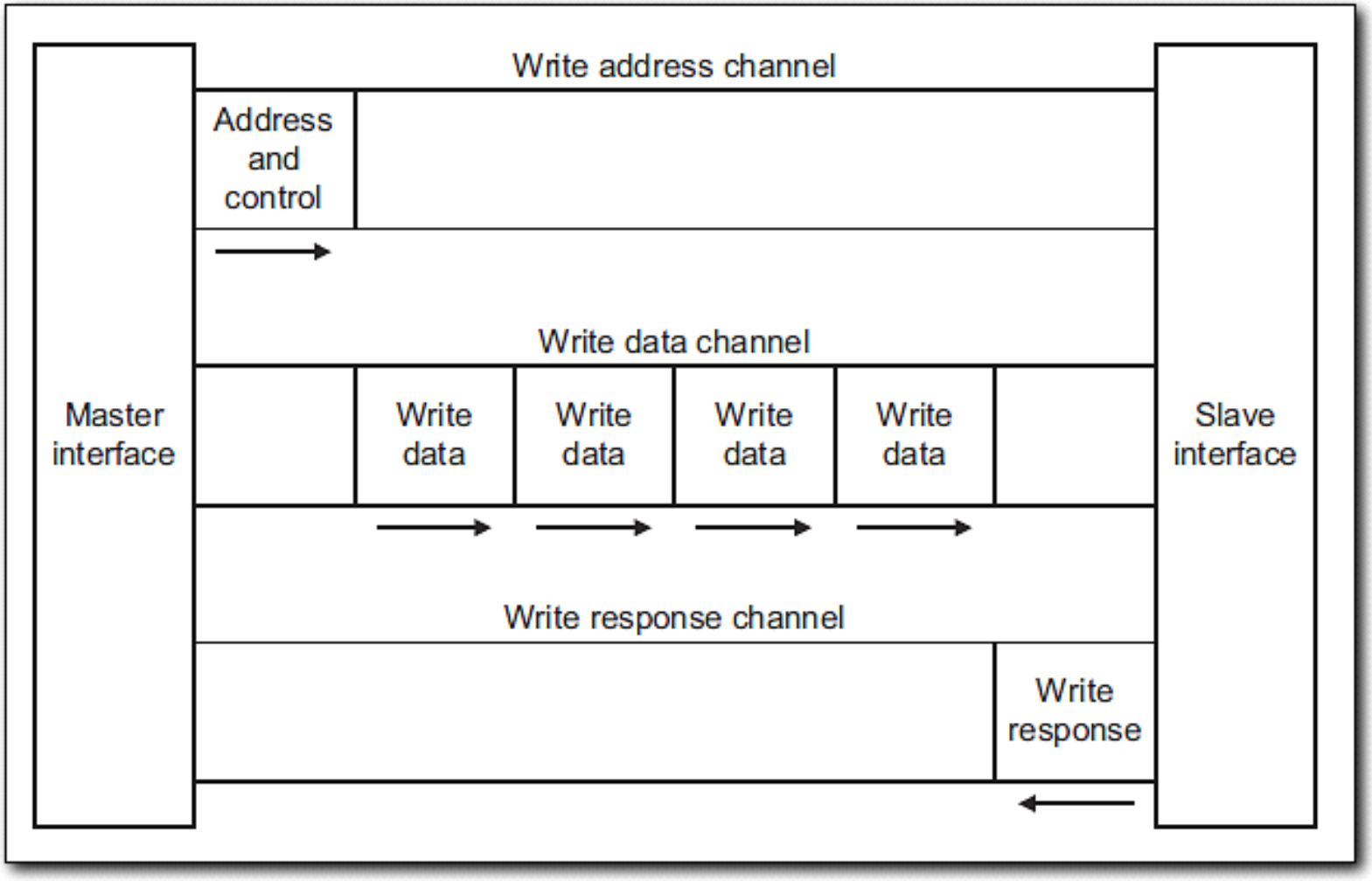


Figure 1-1 Channel architecture of reads

Write transfer using channels



Interface and interconnect

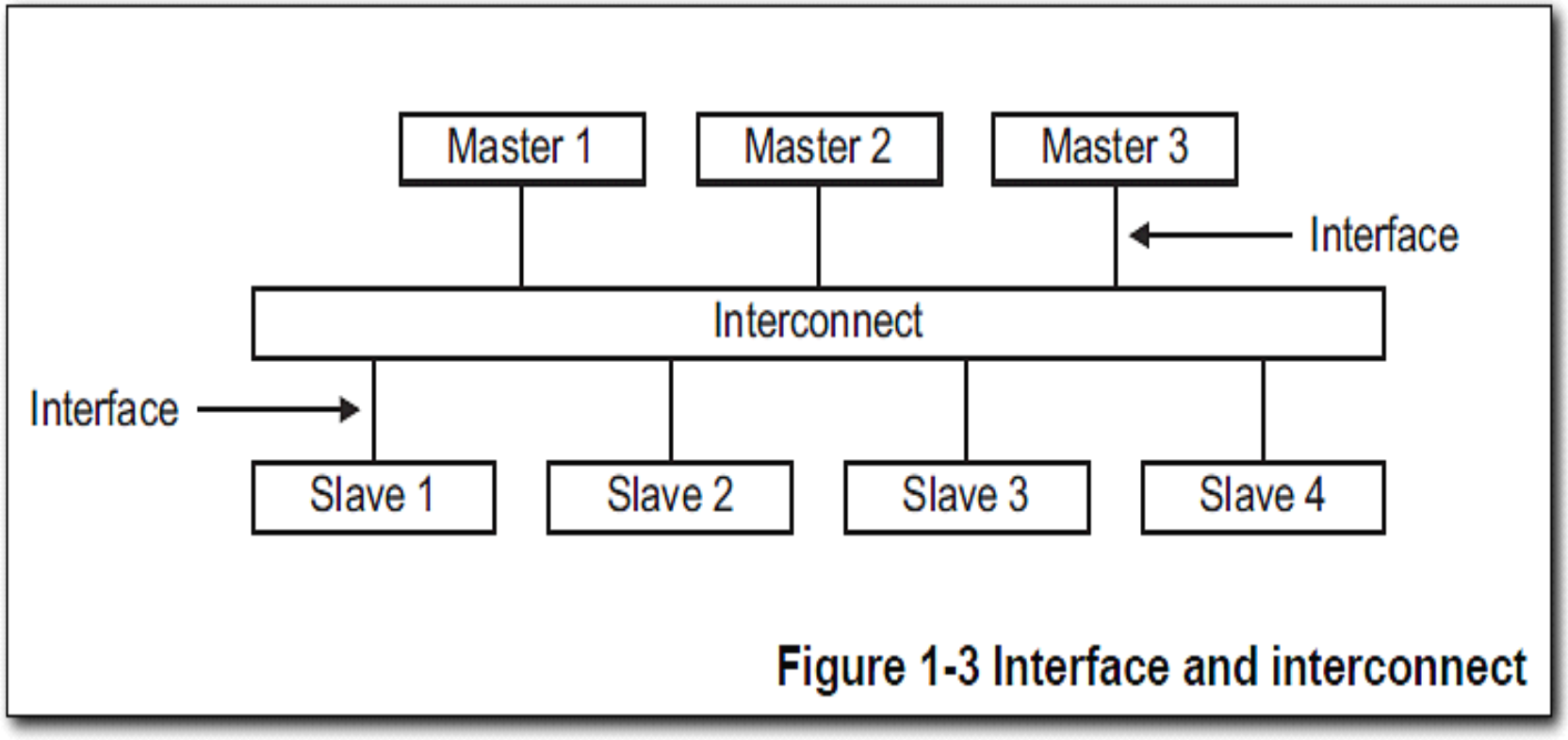
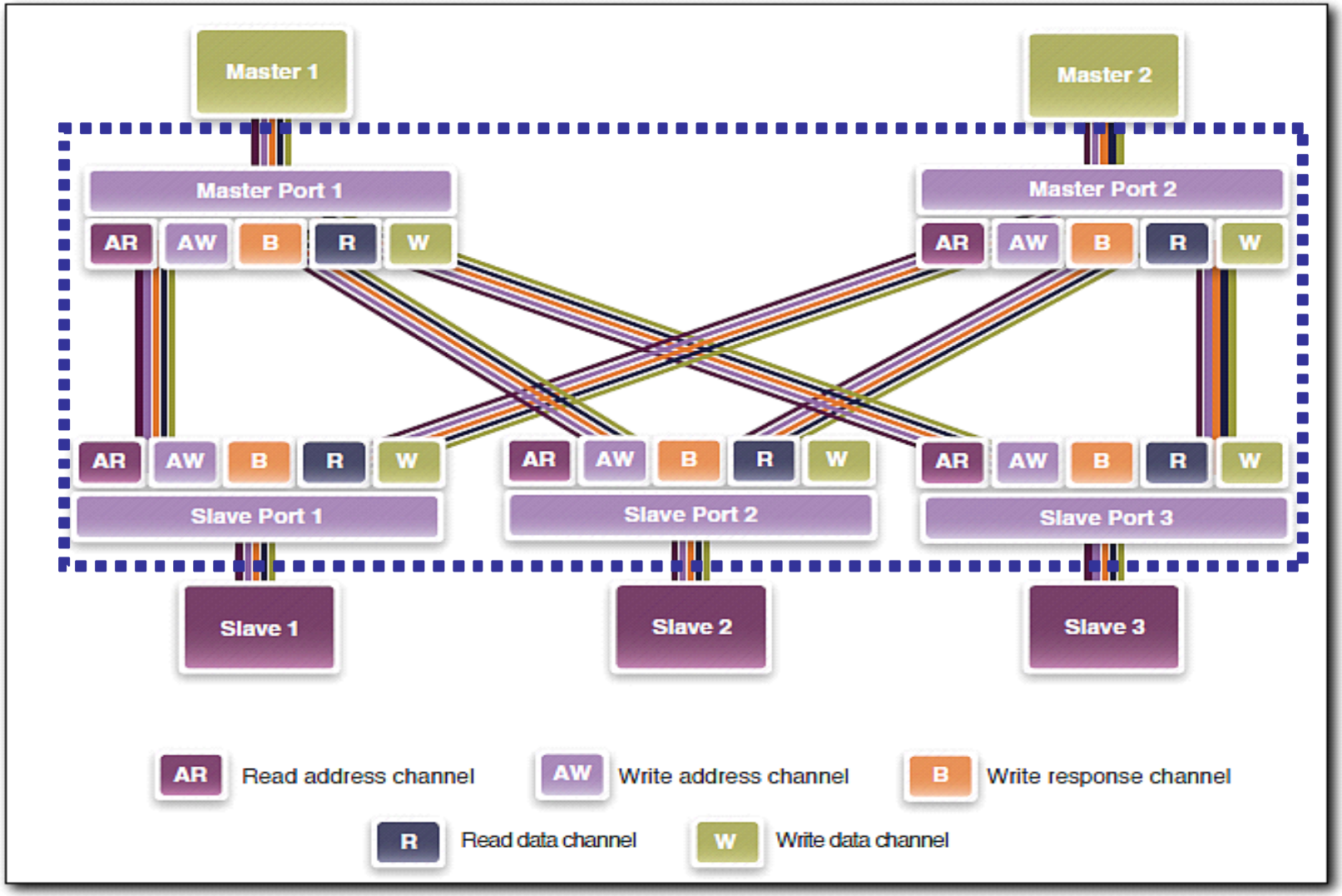
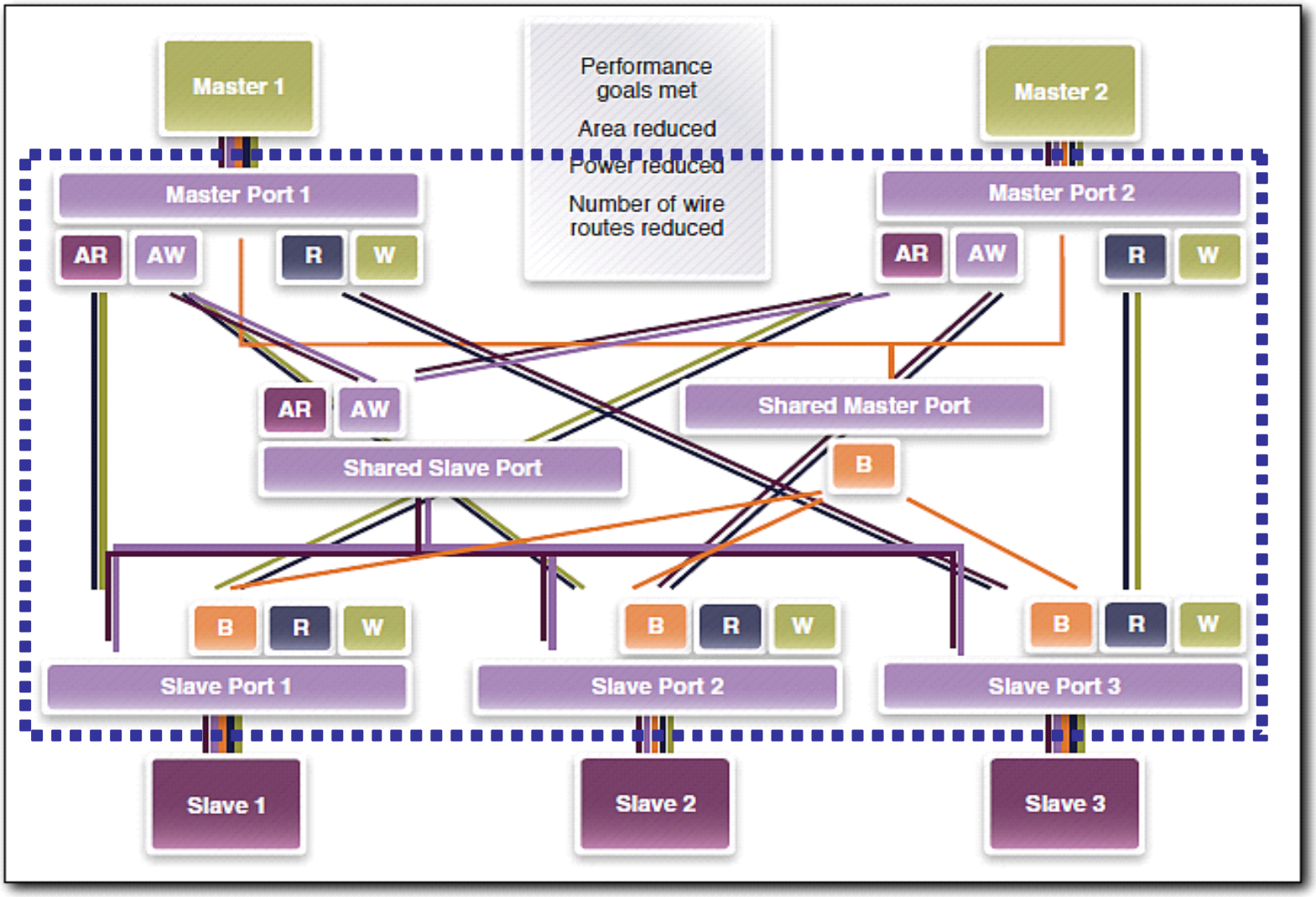


Figure 1-3 Interface and interconnect

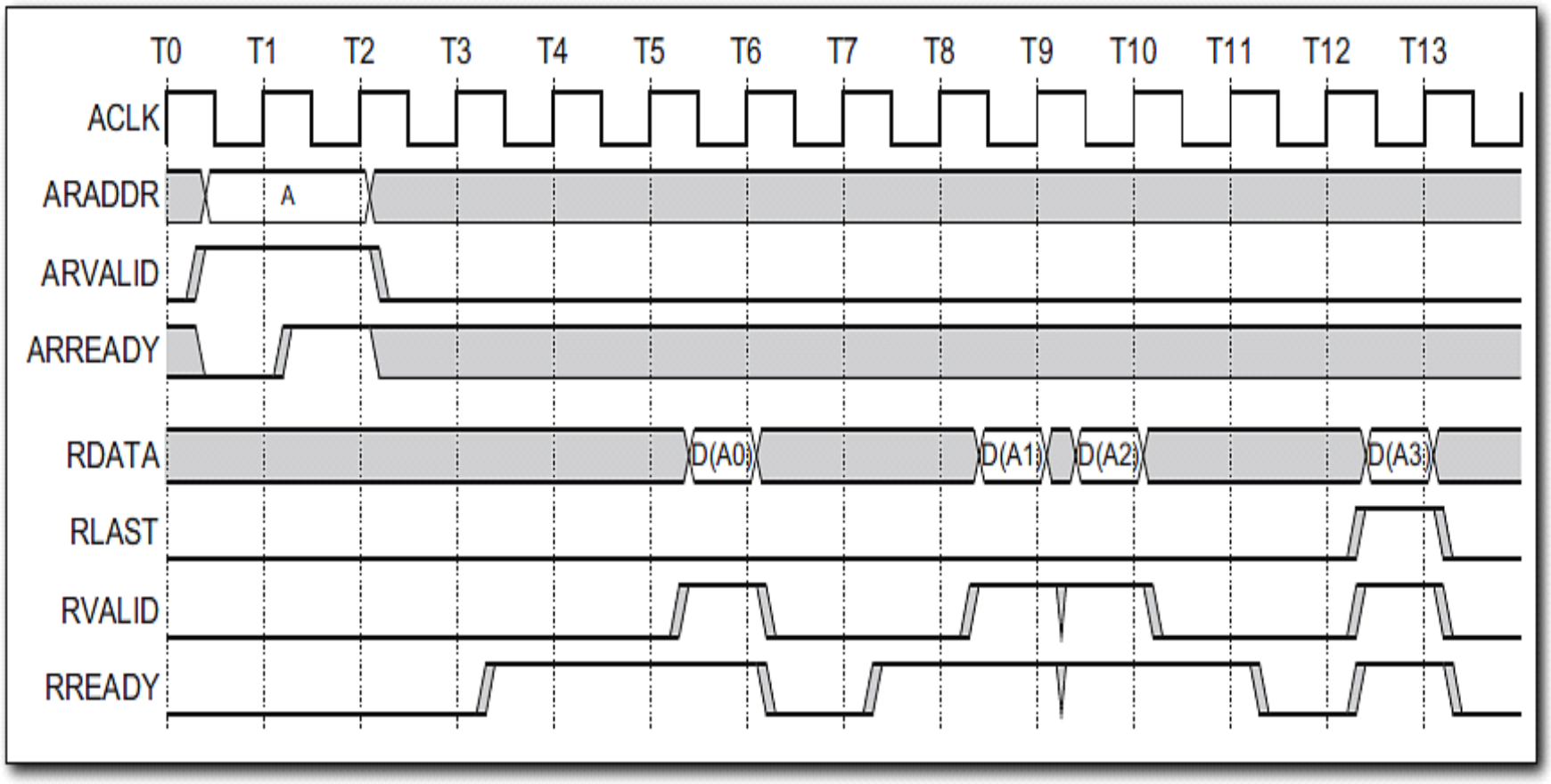
Traditional interconnect



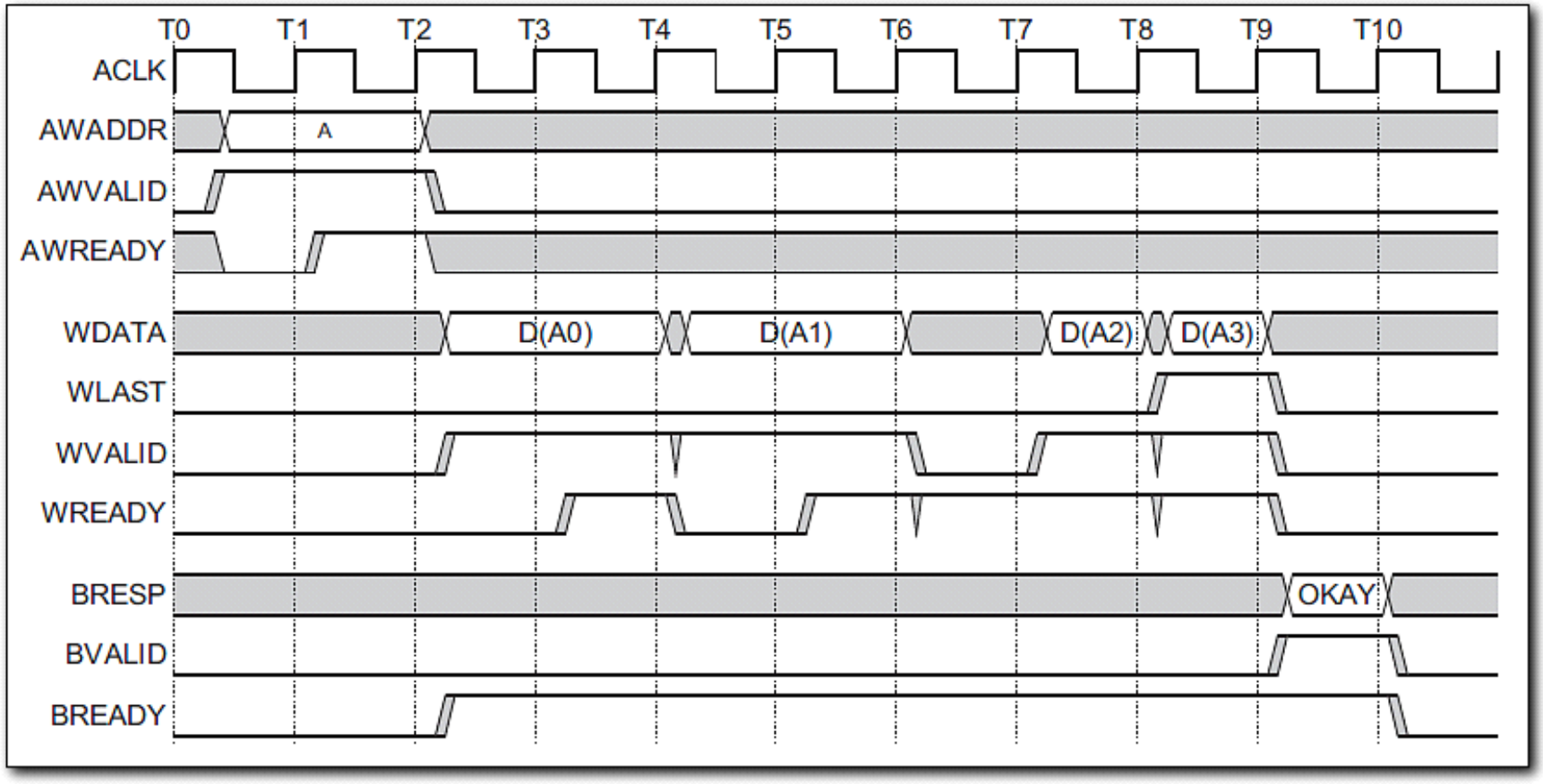
Another Implement (from synopsys)



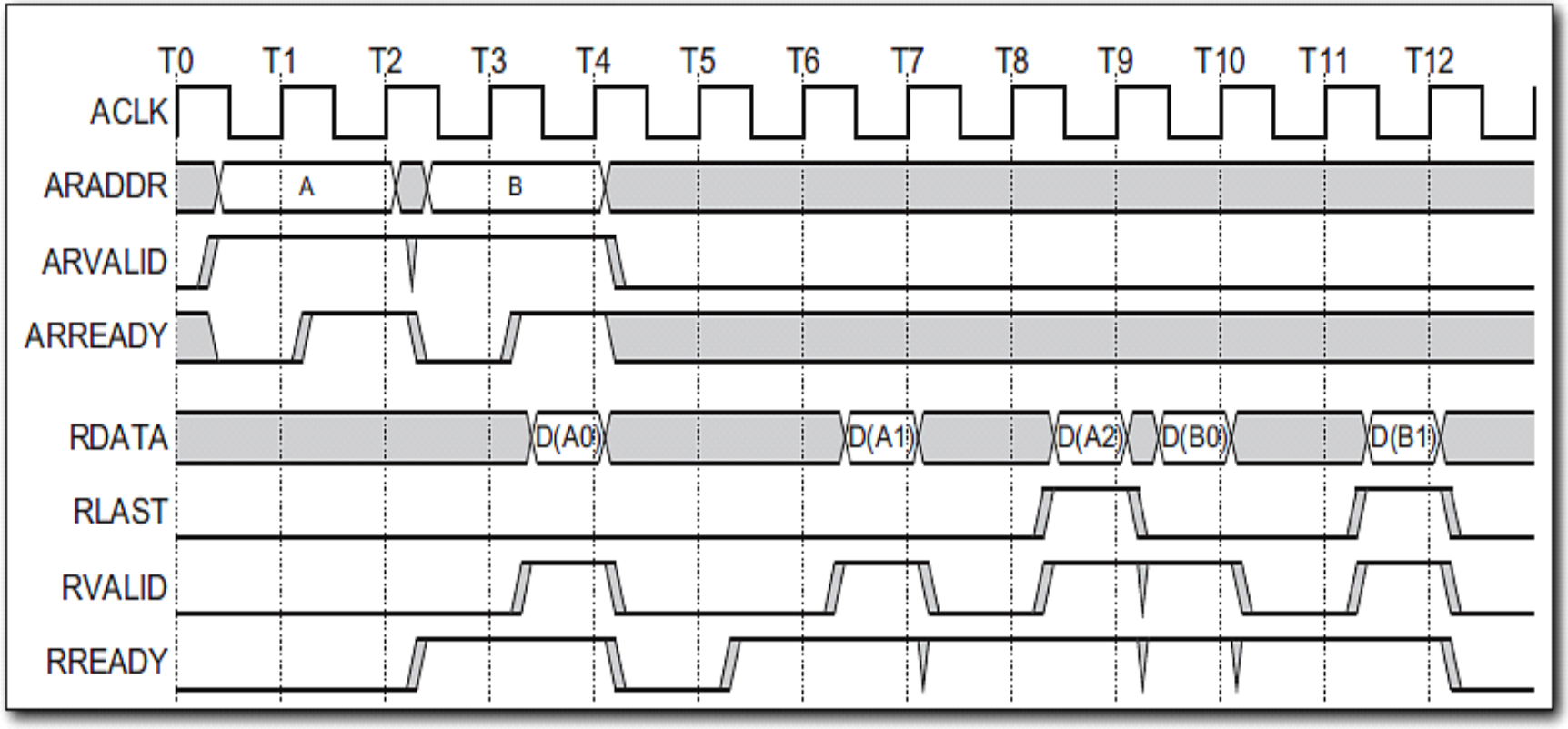
Timing diagram for burst read



Timing diagram for burst write



Overlapped read burst



READY/VALID 握手机制

- 每个channel 都有一对 VALID/READY 信号
- 发送方用 VALID 指示什么时候控制和数据是有效地，接收方使用READY 指示可以接受数据或控制信息，当VALID 和 READY 均为高时，握手完成，信息传输完毕

a. 发送方不能等接收方的ready后才去VALID，并且一旦VALID 有效，必须等待握手完成

b. 接受方可以等待发送方VALID 之后再去READY，并且如果VALID 无效，READY 可以随时撤销

Relations between channels

- 各个channel 都可以独立握手，相互之间的关系是灵活的。例如：write data channel 可以先于 write address channel 先到，也可以与后者同时到达。interconnect负责解析地址中要求的slave，需要对齐这些请求以保证WVALID 信号送到正确的slave
- 两条关系需要严格满足：

✓ read data follows the address to which the data relates

✓ write response follows the last write transfer in the write transaction to which the write response relates

READY/VALID 依赖关系

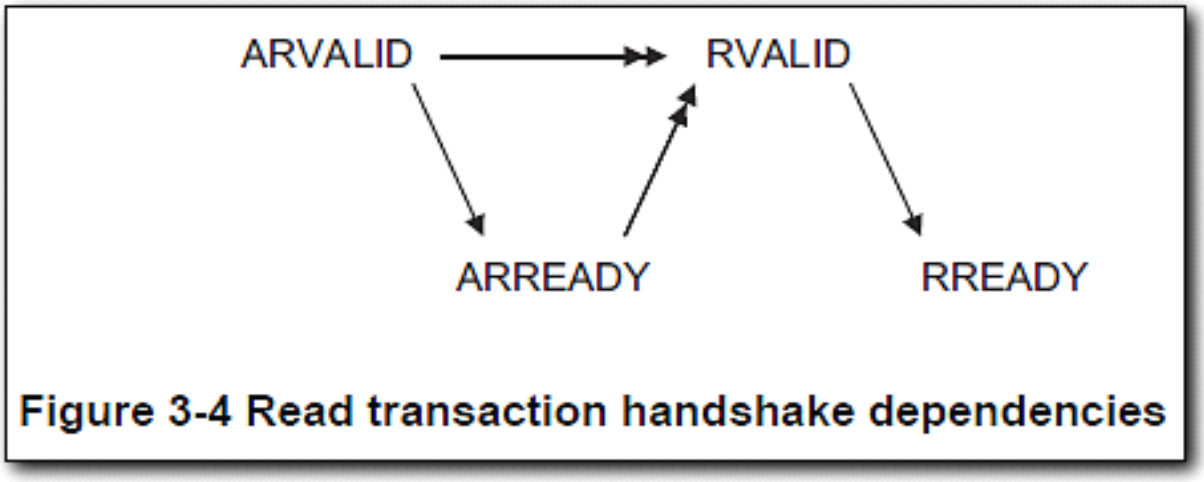


Figure 3-4 Read transaction handshake dependencies

A -> B: B可以等待 A， 但是 A 不能等待 B, 否则易死锁
A ->> B: B一定要等待 A 完成， A 不能等待 B

- the slave can wait for **ARVALID** to be asserted before it asserts **ARREADY**
- the slave must wait for both **ARVALID** and **ARREADY** to be asserted before it starts to return read data by asserting **RVALID**.

READY/VALID 依赖关系

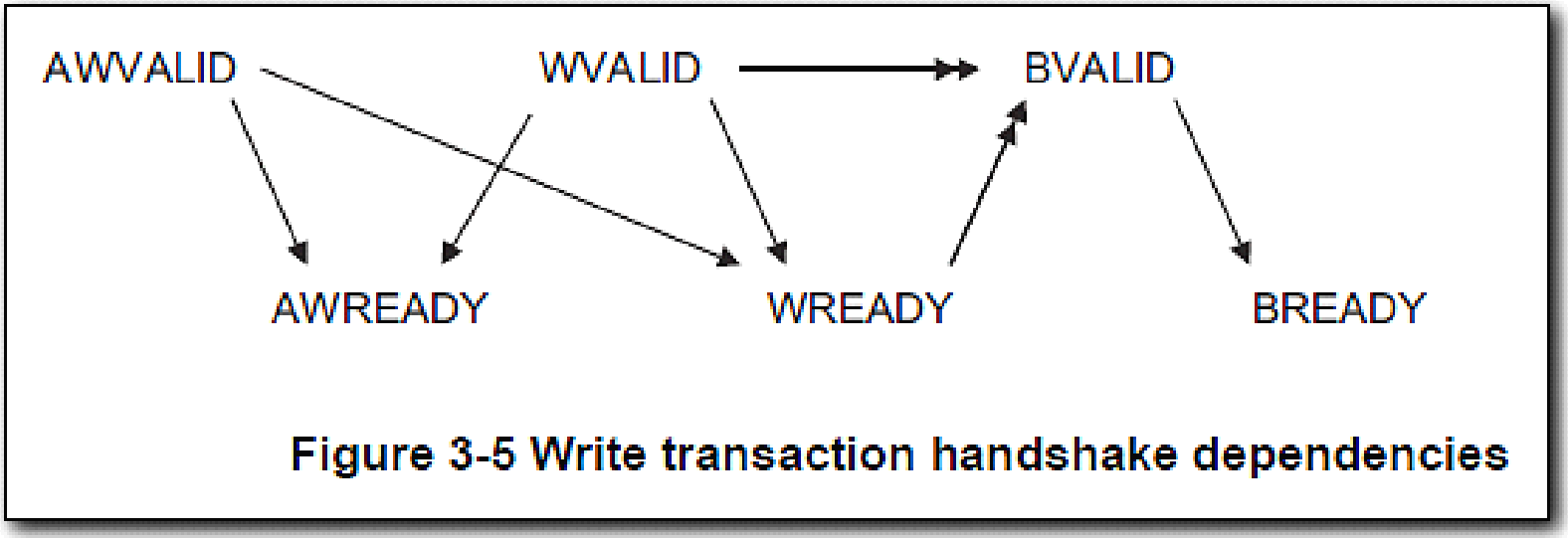


Figure 3-5 Write transaction handshake dependencies

the master must not wait for the slave to assert **AWREADY** or **WREADY** before asserting **AWVALID** or **WVALID**

the slave can wait for **AWVALID** or **WVALID**, or both, before asserting **AWREADY**

the slave can wait for **AWVALID** or **WVALID**, or both, before asserting **WREADY**

the slave must wait for both **WVALID** and **WREADY** to be asserted before asserting **BVALID**.

Address Option && Caculation

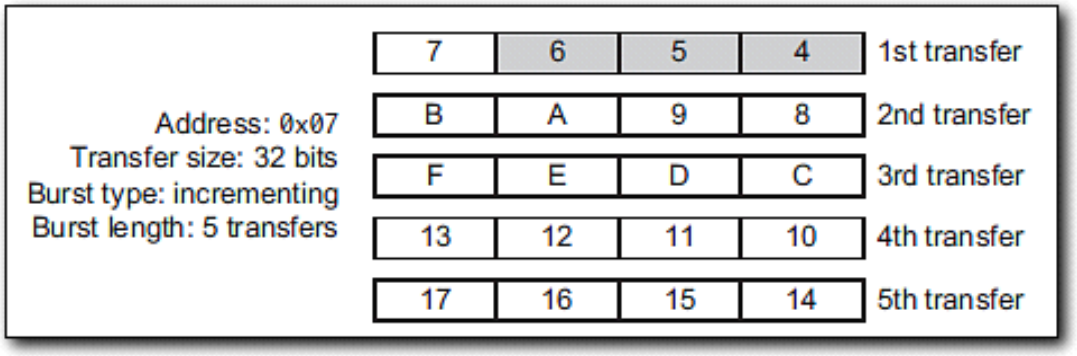
Table 4-1 Burst length encoding

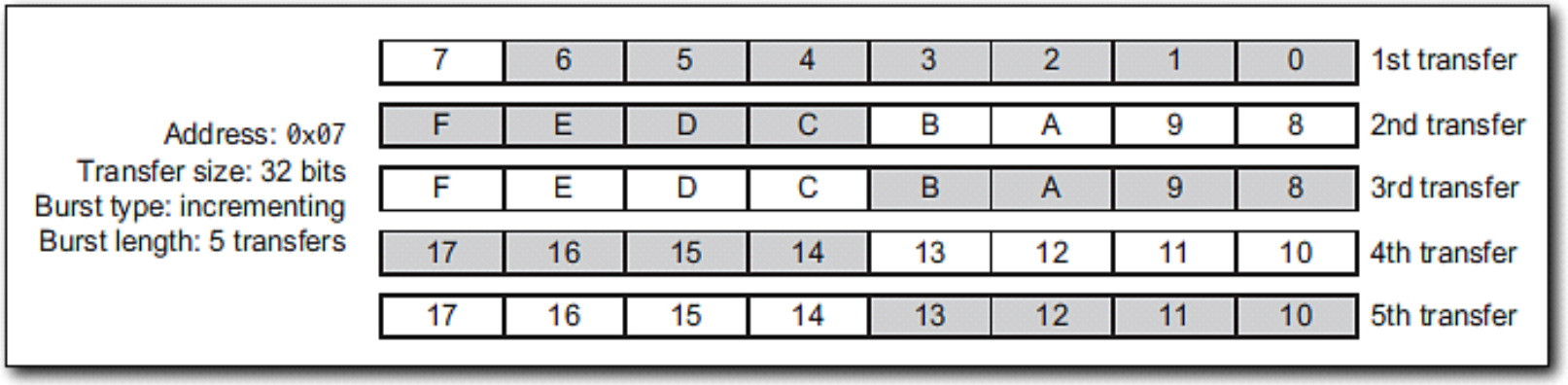
ARLEN[3:0] AWLEN[3:0]	Number of data transfers
b0000	1
b0001	2
b0010	3
. . .	
b1101	14
b1110	15
b1111	16

Table 4-2 Burst size encoding

ARSIZE[2:0] AWSIZE[2:0]	Bytes in transfer
b000	1
b001	2
b010	4
b011	8
b100	16
b101	32
b110	64
b111	128

Burst address && Unaligned access支持





具体实现

- SLAVE 不用关心，在写的时候按照

WSTRB[3:0]	Master	Write strobes. This signal indicates which byte lanes to update in memory. There is one write strobe for each eight bits of the write data bus. Therefore, WSTRB[n] corresponds to WDATA[(8 × n) + 7:(8 × n)].

- 在读的时候，SLAVE 照样按照给出的align地址读，至于那些数据有效，需要Master 根据给定的地址去取，上图就是约定的次序

Ordering Model

- 写传输

step1	step2	step3
Master	Master	Slave
发出地址	发出数据	给出响应
AW	W	B
AWID	WID	BID

- AWID = WID = BID 保证了是同一次传输，而且 relation之间的关系保证了每次传输过程中 step1, step2, step3 是要先后进行的。
- 例如： Master 发出 AW 1, AW2, AW3 ,接下来可以是 W3, W2, W1, 然后slave 的响应可能是 B2, B1, B3。 但是 AW1, W1, B1 的关系是确定了的。
- 对于读传输，涉及到 RW, R 两个通道，也有 ARID , RID 来保证是同一个传输

Ordering Model

- reorder 是相对于传输而言，对于传输中的每个阶段不会 reorder，例如：burst 的write 过程，可能有很多个 beat 操作，各个beat的顺序是在AW中控制和地址信息就固定了的，不会再去 reorder了。
- master 发出的ID是4位的，interconnect 再根据不同master加上一个前缀，以保证不同 master发出的ID 是不一样的。这样 slave收到的ID 的位长要大一些，当 slave返回RID或者BID的时候，interconnect 会去掉前缀返回给 master。
- 对master的要求：
  可以不等前面的请求完成就又发出新的请求
- 对 slave 的要求：
  可以同时接受几个 pending的请求
- 这个特征不是强求的，设备可以不支持，那么默认所有的请求按序完成即可

Ordering Rules

- 不同master的transfer没有顺序要求，可以乱序完成
- 同一个master，但是给出不同的ID，那么可以乱序完成
- 对于几个ReadTransfer，如果 ARID 相同，必须顺序完成。
  - ✓ 当这些相同的ARID 是发送到同一个slave的话，那么slave必须保证这些请求要按序完成
  - ✓ 当这些相同的ARID 是发送到不同的slave的话，那么interconnect必须保证这些请求按序完成
- ARID 和 AWID 相同的传输没有顺序要求，可以乱序完成。如果master想要有某种顺序，master必须保证前面的请求已经完成，才能发出下一个请求。

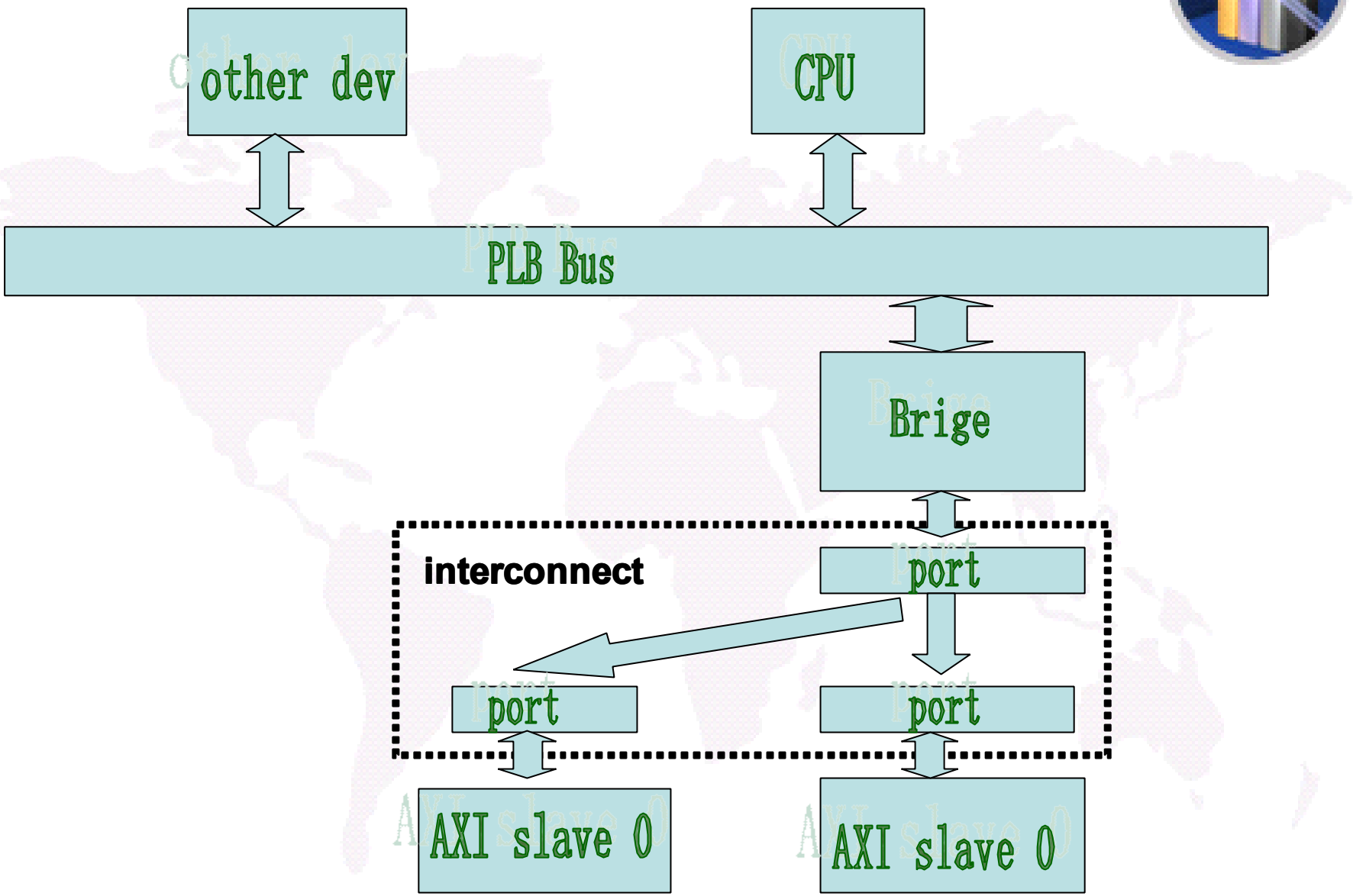
Ordering Rules

- read data reordering depth is the number of address pending in the slave that can be reordered. By default,the value is 1.

- write data interleaving depth indicates if the slave can accept interleaved write data from source with different AWID values.By default, the value is 1.

Ordering Rules

- 如果两个写传输，有不同的AWID，但是他们写的地址相同或者有重叠的部分，那么处理的顺序是未定义的，上层的协议应该考虑这种情况
- 对于同一个master的读传输和写传输，没有顺序限制，master应该自己保证顺序


other dev
CPU
PLB Bus
Brige
interconnect
port
port
port
AXI slave 0
AXI slave 0

update for AXI4

- support for burst lengths up to 256 beats
- Quality of Service (QoS) signaling
- support for multiple region interfaces
- updated write response requirements
- updated AWCACHE and ARCACHE signaling details
- additional information on Ordering requirements
- details of optional User signaling
- removal of locked transactions
- removal of write interleaving