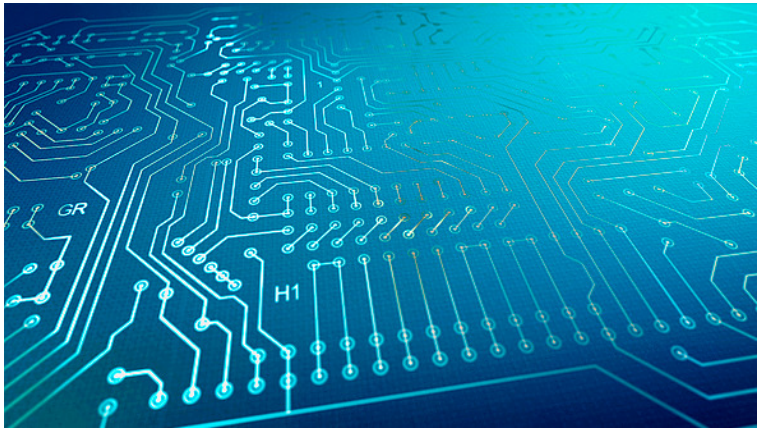


Buses de Interconexión para SoC

Alejandro Furfaro

6 de octubre de 2020

Temario

Temario

Principios de Diseño de un Bus

Cuando se emprende el diseño de un SOC, no solo es importante definir que bloques se van a incluir en el sistema, sino como se los va a interconectar.

Principios de Diseño de un Bus

No es tan trivial como conectar los terminales de address data y control de cada componente y diseñar un decodificador de direcciones. Hace tiempo que es mas sofisticado que eso.

Principios de Diseño de un Bus

Forman parte de la Arquitectura e involucran interfaces de hardware diseminadas en los diferentes componentes, lo cual les confiere independencia de los dispositivos, y controladores permitiendo interconectar de manera transparente una variedad muy heterogénea de microcontrollers, CPUs de propósito general y hasta GPU's en un mismo SOC con diversidad de dispositivos periféricos, y memorias, de diferentes velocidades de acceso y tecnologías.

Bus AMBA

Bus AMBA

- **AMBA** por **A**dvanced **M**icrocontroller **B**us **A**rchitecture.

Bus AMBA

- **AMBA** por **A**dvanced **M**icrocontroller **B**us **A**rchitecture.
- Estándar abierto presentado en 1996 por ARM.

Bus AMBA

- **AMBA** por **A**dvanced **M**icrocontroller **B**us **A**rchitecture.
- Estándar abierto presentado en 1996 por ARM.
- Objeto: Conectar y controlar los periféricos y bloques que se conectan a una CPU ***en el interior de un SoC***.

Bus AMBA

- **AMBA** por **A**dvanced **M**icrocontroller **B**us **A**rchitecture.
- Estándar abierto presentado en 1996 por ARM.
- Objeto: Conectar y controlar los periféricos y bloques que se conectan a una CPU ***en el interior de un SoC***.
- Escaló desde microcontroladores hasta procesadores de alta performance, incluso con soporte para SMP y AMP.

Bus AMBA

- **AMBA** por **A**dvanced **M**icrocontroller **B**us **A**rchitecture.
- Estándar abierto presentado en 1996 por ARM.
- Objeto: Conectar y controlar los periféricos y bloques que se conectan a una CPU ***en el interior de un SoC***.
- Escaló desde microcontroladores hasta procesadores de alta performance, incluso con soporte para SMP y AMP.
- La primer versión de la especificación cubrió dos versiones:

Bus AMBA

- **AMBA** por **A**dvanced **M**icrocontroller **B**us **A**rchitecture.
- Estándar abierto presentado en 1996 por ARM.
- Objeto: Conectar y controlar los periféricos y bloques que se conectan a una CPU ***en el interior de un SoC***.
- Escaló desde microcontroladores hasta procesadores de alta performance, incluso con soporte para SMP y AMP.
- La primer versión de la especificación cubrió dos versiones:
- ✓ "**A**dvanced **S**ystem **B**us" (**ASB**),

Bus AMBA

- **AMBA** por **A**dvanced **M**icrocontroller **B**us **A**rchitecture.
- Estándar abierto presentado en 1996 por ARM.
- Objeto: Conectar y controlar los periféricos y bloques que se conectan a una CPU ***en el interior de un SoC***.
- Escaló desde microcontroladores hasta procesadores de alta performance, incluso con soporte para SMP y AMP.
- La primer versión de la especificación cubrió dos versiones:

✓ "**A**dvanced **S**ystem **B**us" (**ASB**),

✓ "**A**dvanced **P**eriferal **B**us " (**APB**).

Bus AMBA

- **AMBA** por **A**dvanced **M**icrocontroller **B**us **A**rchitecture.
- Estándar abierto presentado en 1996 por ARM.
- Objeto: Conectar y controlar los periféricos y bloques que se conectan a una CPU ***en el interior de un SoC***.
- Escaló desde microcontroladores hasta procesadores de alta performance, incluso con soporte para SMP y AMP.
- La primer versión de la especificación cubrió dos versiones:

✓ "**A**dvanced **S**ystem **B**us" (**ASB**),

✓ "**A**dvanced **P**eriferal **B**us " (**APB**).

- La segunda versión, de 1999, es hoy un estándar de facto en Microcontrollers: **AHB** (**A**dvanced **H**igh-Performance **B**us), que trabaja en único flanco de clock.

Bus AMBA

- **AMBA** por **A**dvanced **M**icrocontroller **B**us **A**rchitecture.
- Estándar abierto presentado en 1996 por ARM.
- Objeto: Conectar y controlar los periféricos y bloques que se conectan a una CPU ***en el interior de un SoC***.
- Escaló desde microcontroladores hasta procesadores de alta performance, incluso con soporte para SMP y AMP.
- La primer versión de la especificación cubrió dos versiones:

✓ "**A**dvanced **S**ystem **B**us" (**ASB**),

✓ "**A**dvanced **P**eriferal **B**us " (**APB**).

- La segunda versión, de 1999, es hoy un estándar de facto en Microcontrollers: **AHB** (**A**dvanced **H**igh-Performance **B**us), que trabaja en único flanco de clock.
- No solo ARM utiliza estos buses. Leon II y III utilizan **AHB**, y **APB**.

Bus AMBA

- La versión 3 de **AMBA** introduce **AXI**: **A**dvanced e**X**tensible **I**nterface, con el propósito de lograr mayor performance en la interconexión, y **ATB**: **A**dvanced **T**race **B**us como parte de la interfaz CoreSight on chip para debug y Trace.

Bus AMBA

- La versión 3 de **AMBA** introduce **AXI**: **A**dvanced e**X**tensible **I**nterface, con el propósito de lograr mayor performance en la interconexión, y **ATB**: **A**dvanced **T**race **B**us como parte de la interfaz CoreSight on chip para debug y Trace.
- En 2010 se libera AMBA v4, inicialmente con AXI4, completada en 2011 con **AXI ACE** (por **AXI A**dvanced **C**oherency **E**xtensions) que proporciona coherencia para todo el resto del sistema AMBA 4.

Bus AMBA

- La versión 3 de **AMBA** introduce **AXI**: **A**dvanced e**X**tensible **I**nterface, con el propósito de lograr mayor performance en la interconexión, y **ATB**: **A**dvanced **T**race **B**us como parte de la interfaz CoreSight on chip para debug y Trace.
- En 2010 se libera AMBA v4, inicialmente con AXI4, completada en 2011 con **AXI ACE** (por **AXI A**dvanced **C**oherency **E**xtensions) que proporciona coherencia para todo el resto del sistema AMBA 4.
- En 2013 se presenta AMBA 5 **C**oherent **H**ub **I**nterface (**CHI**) con un fuerte rediseño de alta velocidad en la capa de transporte y eliminación de congestiones.

Temario

Introducción a AMBA v2

La especificación AMBA v2, contiene tres protocolos

Introducción a AMBA v2

La especificación AMBA v2, contiene tres protocolos

- **Advanced High-Performance Bus (AHB)**: Protocolo para interconexión de módulos de alta frecuencia y alta performance introducido en AMBAv2. Interconecta CPU's, módulos de memoria on-chip y off-chip, con interfaces de periféricos de macrocelda de baja potencia. Introduce nuevas técnicas de síntesis y testing que permiten agilizar y simplificar los diseños.

Introducción a AMBA v2

La especificación AMBA v2, contiene tres protocolos

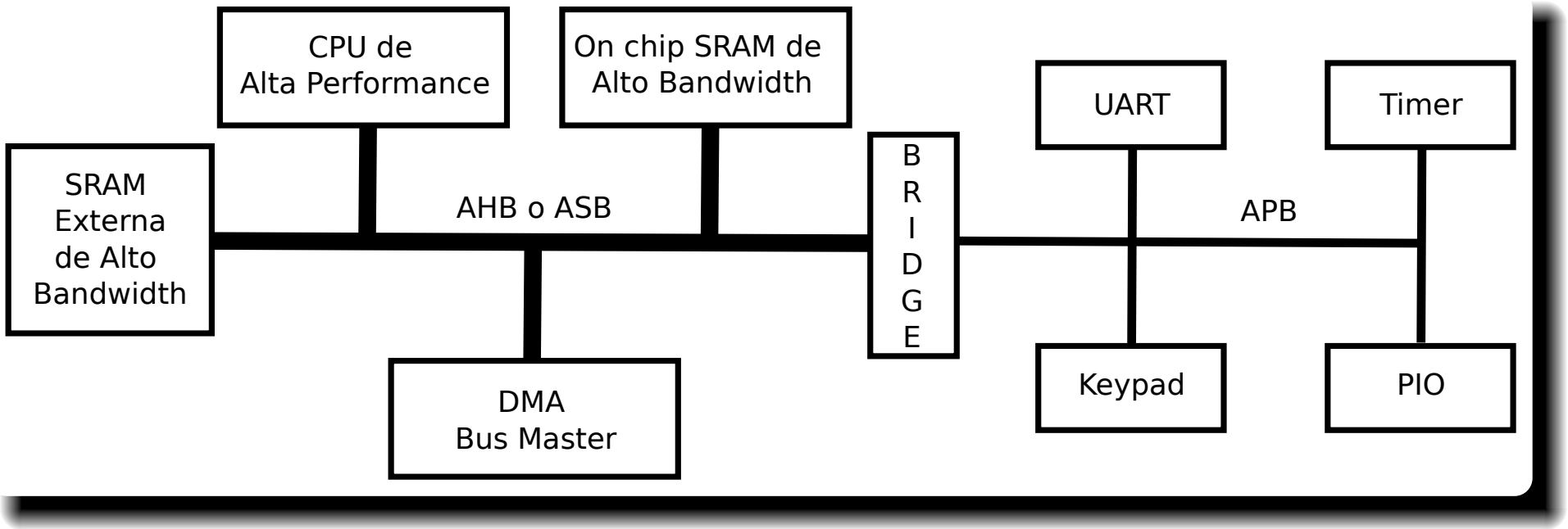
- **Advanced High-Performance Bus (AHB)**: Protocolo para interconexión de módulos de alta frecuencia y alta performance introducido en AMBAv2. Interconecta CPU's, módulos de memoria on-chip y off-chip, con interfaces de periféricos de macrocelda de baja potencia. Introduce nuevas técnicas de síntesis y testing que permiten agilizar y simplificar los diseños.
- **Advanced System Bus (ASB)**: Protocolo original para módulos de alta performance. Interconecta CPU's, módulos de memoria on-chip y off-chip, con interfaces de periféricos de macrocelda de baja potencia. Puede aplicarse cuando no son necesarias las nuevas funciones de **AHB**.

Introducción a AMBA v2

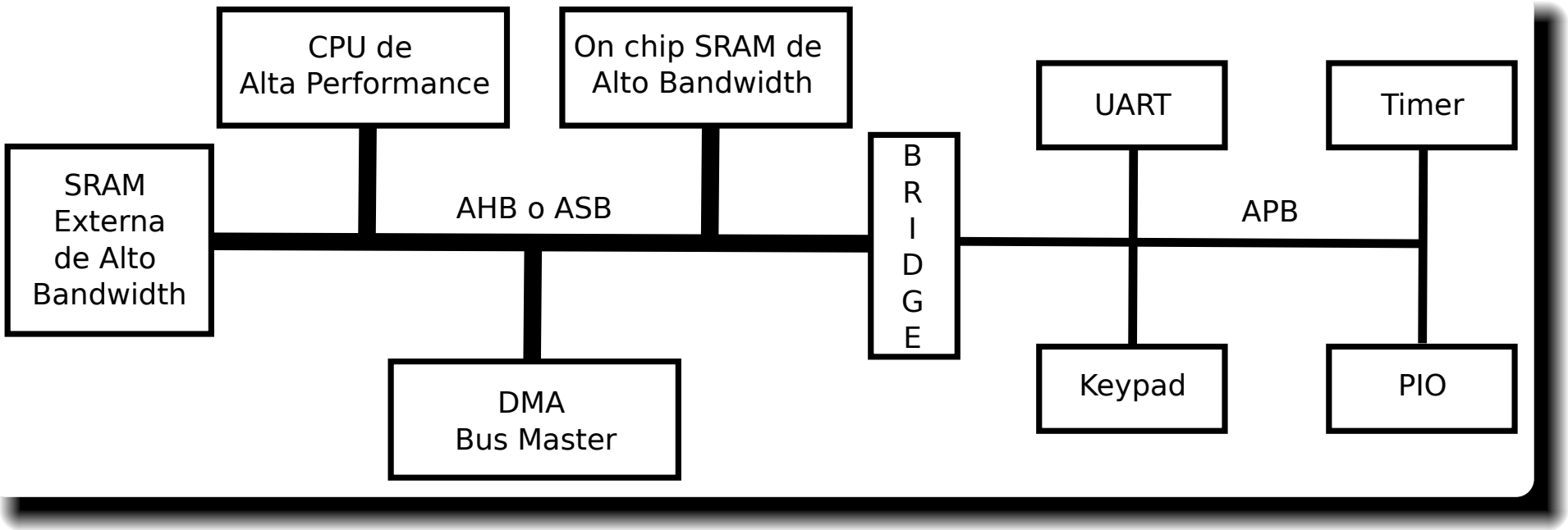
La especificación AMBA v2, contiene tres protocolos

- **Advanced High-Performance Bus (AHB)**: Protocolo para interconexión de módulos de alta frecuencia y alta performance introducido en AMBAv2. Interconecta CPU's, módulos de memoria on-chip y off-chip, con interfaces de periféricos de macrocelda de baja potencia. Introduce nuevas técnicas de síntesis y testing que permiten agilizar y simplificar los diseños.
- **Advanced System Bus (ASB)**: Protocolo original para módulos de alta performance. Interconecta CPU's, módulos de memoria on-chip y off-chip, con interfaces de periféricos de macrocelda de baja potencia. Puede aplicarse cuando no son necesarias las nuevas funciones de **AHB**.
- **Advanced Peripheral Bus (APB)**: Protocolo pensado para conectar dispositivos periféricos de baja potencia y complejidad. Puede utilizarse en conjunto con cualquier otra versión del Bus del Sistema.

Introducción a AMBA v2



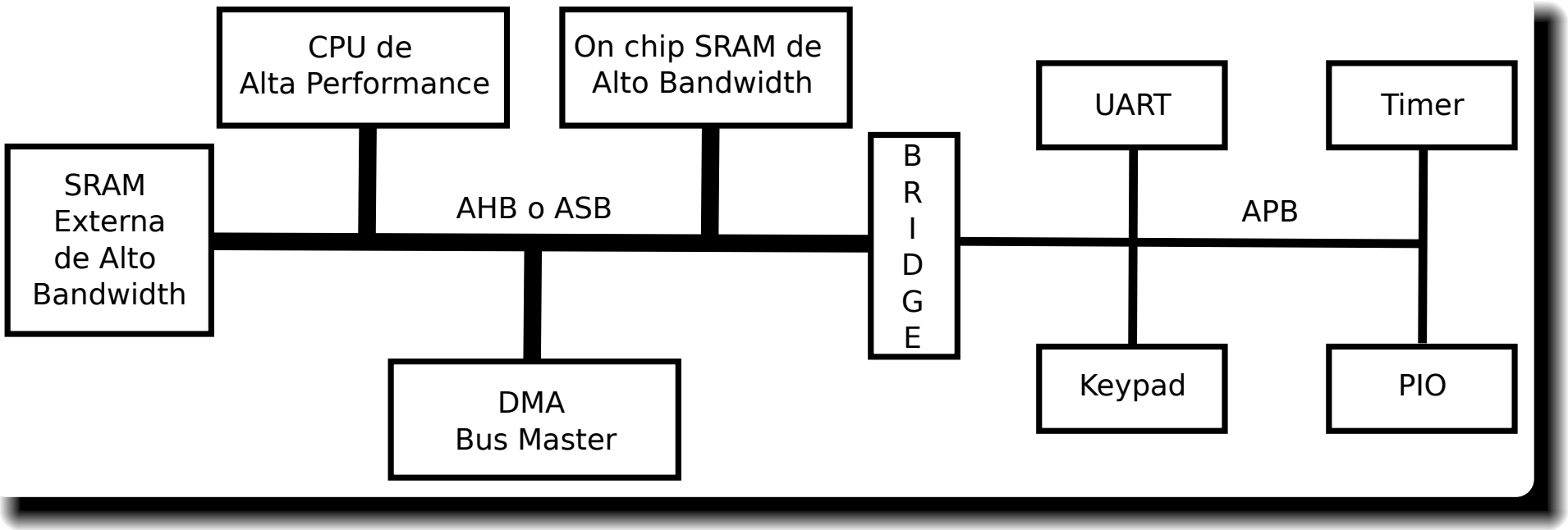
Introducción a AMBA v2



AHB

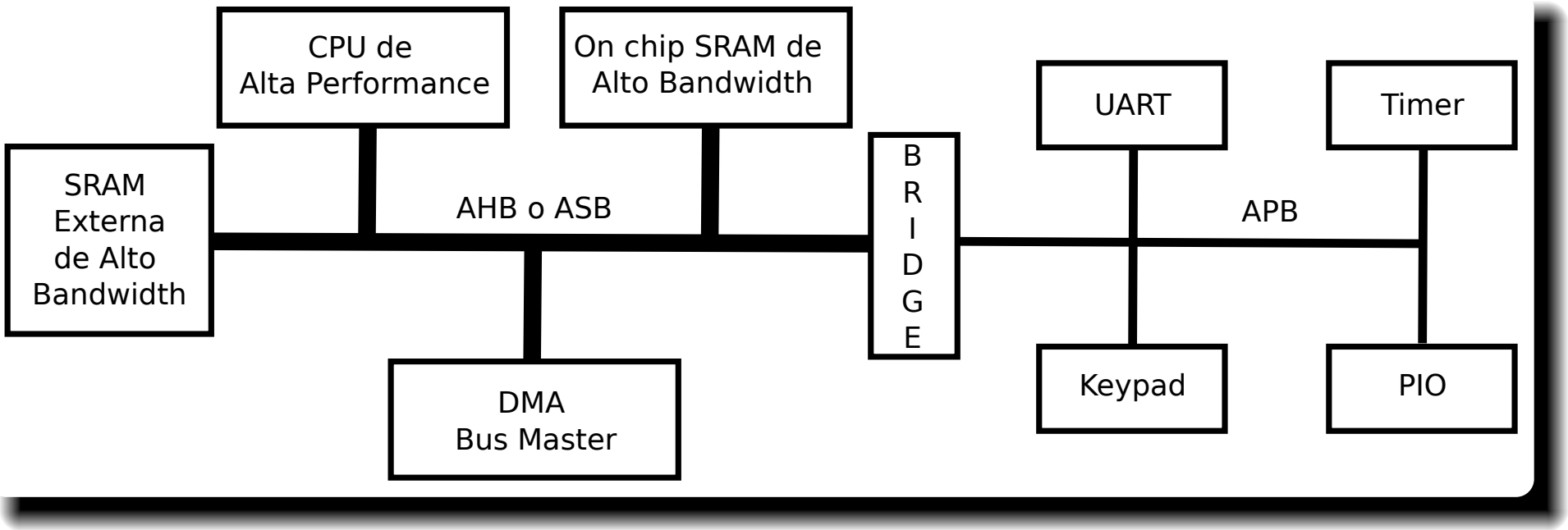
- Alto rendimiento
- Operación en Pipeline
- Acepta Múltiples bus masters
- Transferencias Burst
- Transacciones divisibles

Introducción a AMBA v2



- Alto rendimiento (ASB)
- Operación en Pipeline
- Acepta Múltiples bus masters
- Transferencias Burst
- Transacciones divisibles

Introducción a AMBA v2



- Alto rendimiento
- Operación en Pipeline
- Acepta Múltiples bus masters
- Transferencias Burst
- Transacciones divisibles

APB

- Low power
- Address y Control Latcheados
- Interfaz Simple
- Adecuado para gran variedad de periféricos

Terminología

Terminología

- ***Bus cycle***: Es la duración de un período de clock. En **AHB APB** y es el tiempo entre dos flancos ascendentes consecutivos. En **ASB** entre dos flancos ascendentes consecutivos.

Terminología

- ***Bus cycle***: Es la duración de un período de clock. En **AHB APB** y es el tiempo entre dos flancos ascendentes consecutivos. En **ASB** entre dos flancos ascendentes consecutivos.

- ***Bus Transfer***: Es la lectura o escritura de un objeto de datos. Toma varios ***Bus cycles***. Termina cuando el dispositivo Slave, responde en forma completa la transacción. Soporta 8, 16, 32, 64, y 128 bits. (***ASB*** solo soporta 8 16 y 32 bits). En ***APB*** son lecturas y escrituras de datos e insumen dos ciclos de Bus.

Terminología

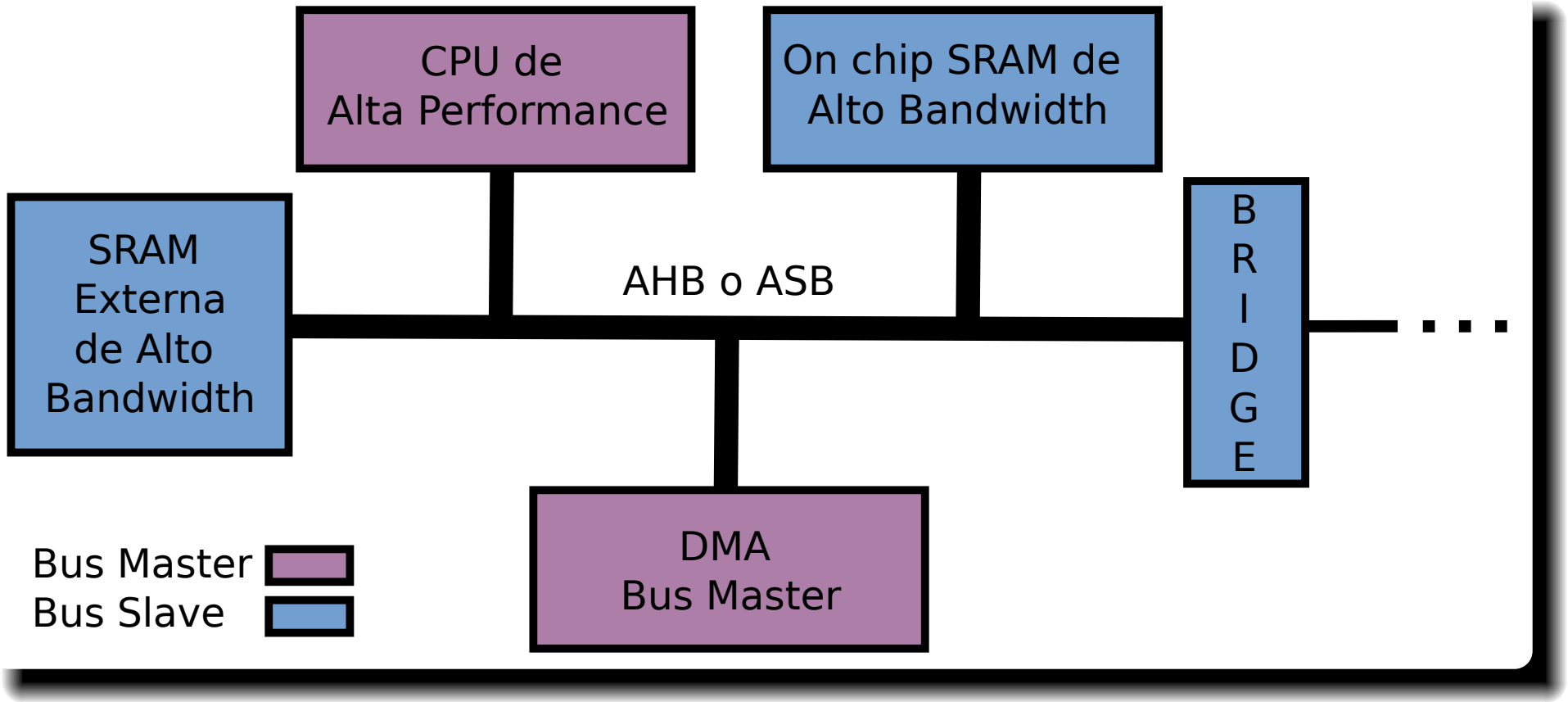
- ***Bus cycle***: Es la duración de un período de clock. En **AHB APB** y es el tiempo entre dos flancos ascendentes consecutivos. En **ASB** entre dos flancos ascendentes consecutivos.

- ***Bus Transfer***: Es la lectura o escritura de un objeto de datos. Toma varios ***Bus cycles***. Termina cuando el dispositivo Slave, responde en forma completa la transacción. Soporta 8, 16, 32, 64, y 128 bits. (***ASB*** solo soporta 8 16 y 32 bits). En ***APB*** son lecturas y escrituras de datos e insumen dos ciclos de Bus.

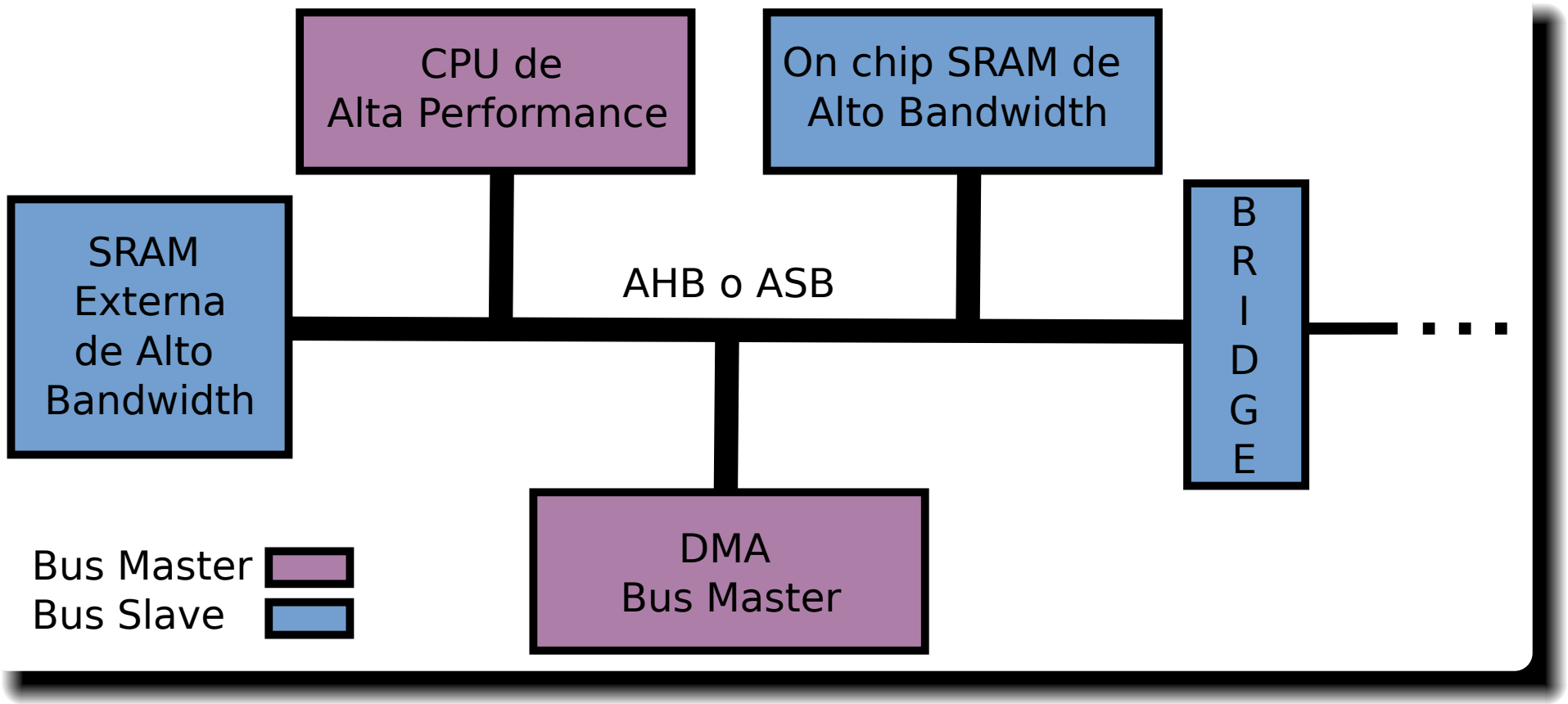
- ***Bus operation***: Soportada solo por ***AHB***. Un Master inicia transacciones de uno a mas datos hacia o desde una zona contigua de memoria, con autoincremento de puntero en pasos determinados por el tamaño de datos (byte, halfword o word)

Temario

Bus **AHB**

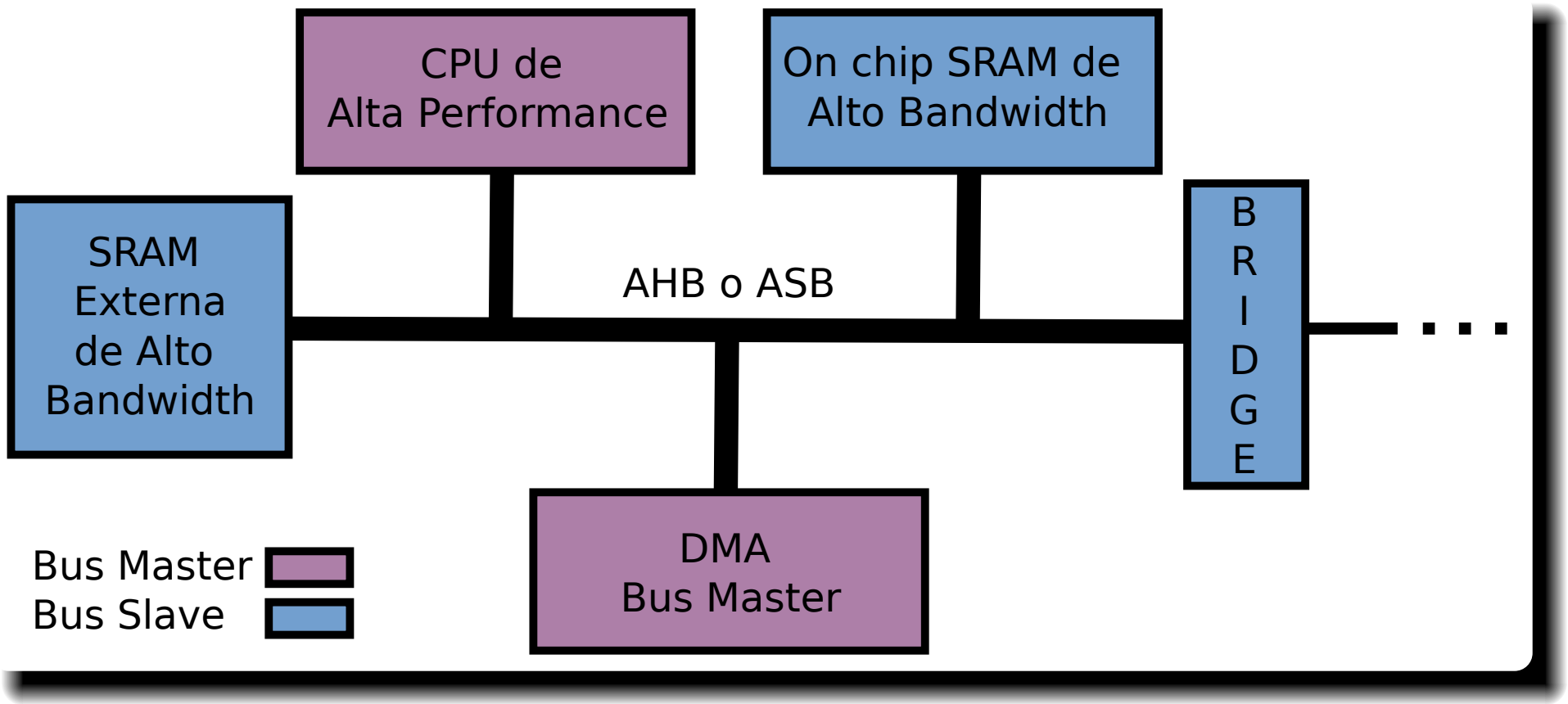

CPU de
Alta Performance
On chip SRAM de
Alto Bandwidth
SRAM
Externa
de Alto
Bandwidth
AHB o ASB
B
R
I
D
G
E
DMA
Bus Master
Bus Master
Bus Slave

Bus **AHB**



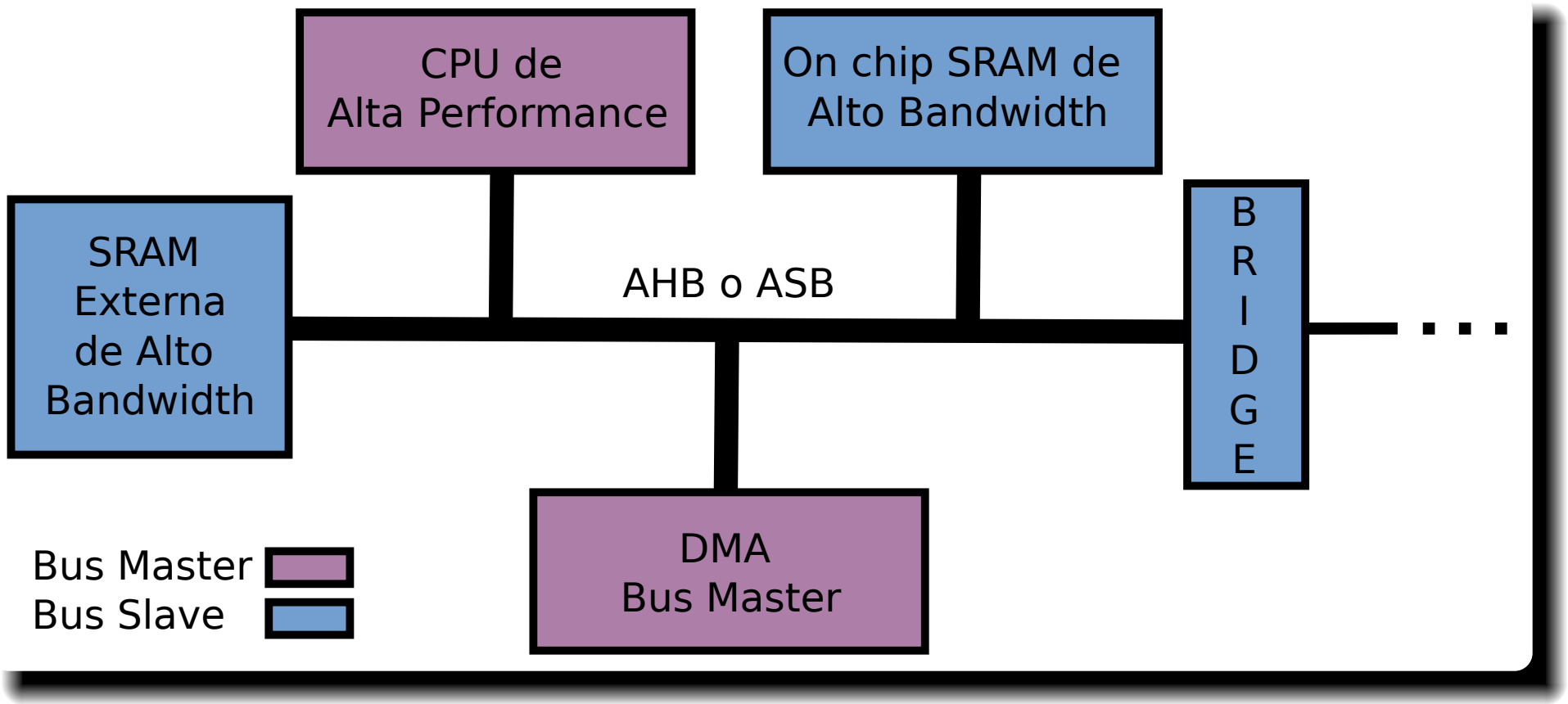
AHB Master es cualquier dispositivo capaz de iniciar una transacción de lectura o de escritura, proporcionando la información de dirección y de control necesaria, en las líneas de bus. Soporta conexión multi master, pero solo uno puede estar activo a la vez.

Bus **AHB**



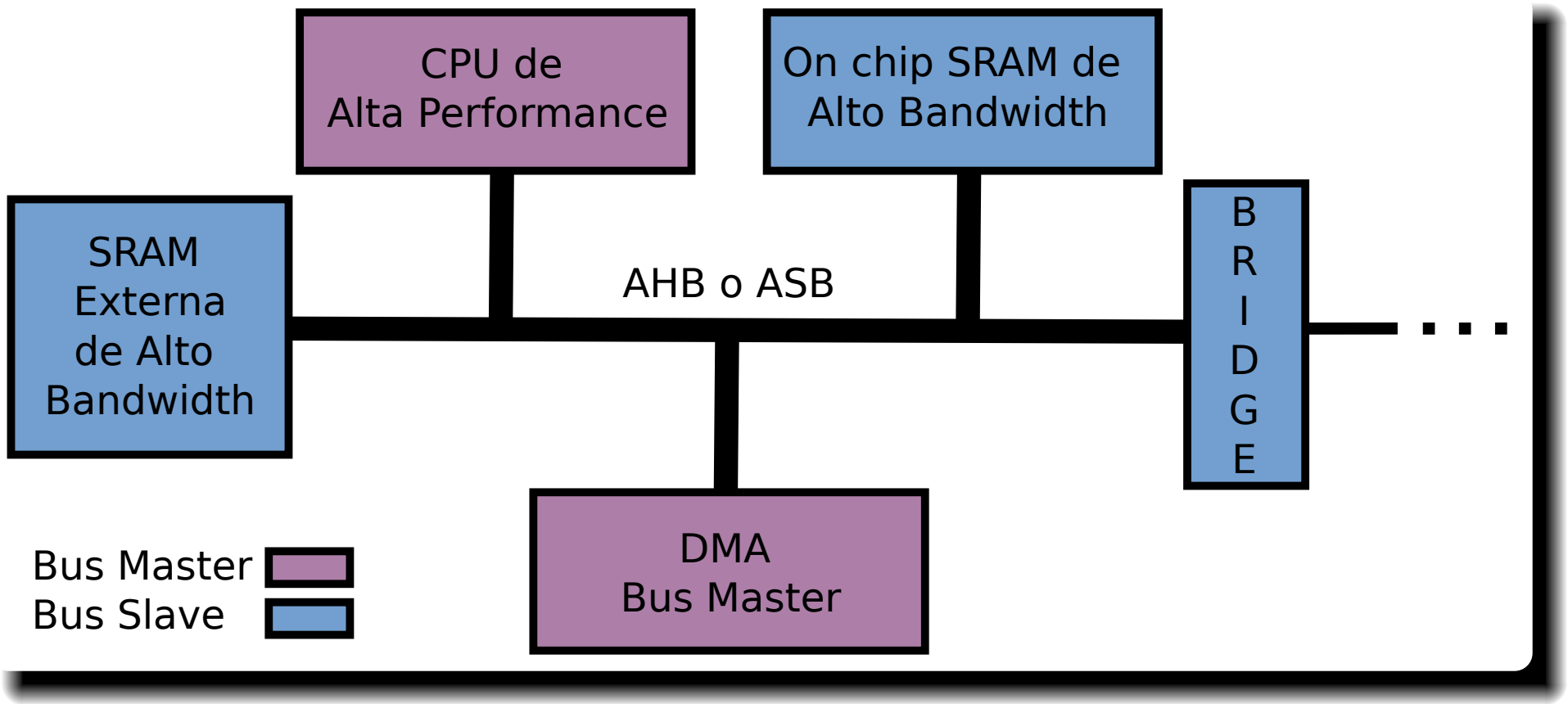
AHB Slave es un dispositivo que responde a una transacción de lectura o escritura para el rango de direcciones dado, al que pertenece la dirección puesta en el bus por un Master al cual señaliza indicando la realización, falla, o espera de la transacción.

Bus **AHB**



AHB Arbiter es un dispositivo que arbitra el acceso al bus por parte de los diferentes ***AHB Master***. Si bien el protocolo es fijo pueden emplearse diferentes algoritmos en base a los requerimientos particulares de cada sistema: *fair access*, *highest priority*. Debe existir aun cuando se tenga un solo Master.

Bus **AHB**



AHB Decoder es el encargado de decodificar la dirección de cada transferencia y generar la señal de Chip Select para el ***AHB Slave*** involucrado. Debe haber un único ***AHB Decoder*** centralizado en cualquier implementación AHB.

Temario

Bus **APB**

Bus **APB**

- Está encapsulado como un Slave del ***AHB***.

Bus **APB**

- Está encapsulado como un Slave del ***AHB***.
- El Bridge APB maneja el handshake con ***AHB***/***ASB***, adaptando el timing y convirtiendo señales en caso de ser necesario hacia el lado ***APB***.

Bus **APB**

- Está encapsulado como un Slave del ***AHB***.
- El Bridge APB maneja el handshake con ***AHB***/***ASB***, adaptando el timing y convirtiendo señales en caso de ser necesario hacia el lado ***APB***.
- Al ***APB*** se conectan aquellos periféricos con bajos requerimientos de ancho de banda, performance, e interfaz de bus pipeline.

Bus **APB**

- Está encapsulado como un Slave del ***AHB***.
- El Bridge APB maneja el handshake con ***AHB***/***ASB***, adaptando el timing y convirtiendo señales en caso de ser necesario hacia el lado ***APB***.
- Al ***APB*** se conectan aquellos periféricos con bajos requerimientos de ancho de banda, performance, e interfaz de bus pipeline.
- Todas sus transiciones se llevan a cabo en el flanco ascendente de la señal de clock.

Bus **APB**

- El bridge convierte transacciones ***AHB***/***ASB*** en ***APB***, latcheando todas las señales de dirección, datos, y control, y proveer un nivel de decodificación de estas mismas señales en términos del ***APB***.

Bus **APB**

- El bridge convierte transacciones ***AHB***/***ASB*** en ***APB***, latcheando todas las señales de dirección, datos, y control, y proveer un nivel de decodificación de estas mismas señales en términos del ***APB***.

- Los dispositivos conectados al ***APB*** son Slaves del Bridge. El acceso es asincrónico controlado por handshake desde el Bridge, siendo validas durante todo el acceso las señales de Direcciones y control, y los datos solo en accesos de escritura.

Bus **APB**

- El bridge convierte transacciones ***AHB***/***ASB*** en ***APB***, latcheando todas las señales de dirección, datos, y control, y proveer un nivel de decodificación de estas mismas señales en términos del ***APB***.
- Los dispositivos conectados al ***APB*** son Slaves del Bridge. El acceso es asincrónico controlado por handshake desde el Bridge, siendo validas durante todo el acceso las señales de Direcciones y control, y los datos solo en accesos de escritura.
- El bus ***APB*** está estático (zero-powered) si no hay actividad

Temario

Bus de sistema y Bus de periféricos

Bus de sistema y Bus de periféricos

- El diseño mas simple de un SoC es poner todos los periféricos del lado ***AHB***. Pero:

Bus de sistema y Bus de periféricos

- El diseño mas simple de un SoC es poner todos los periféricos del lado ***AHB***. Pero:
- Se incrementa la cantidad de cargas en el Bus incrementando la disipación de potencia.

Bus de sistema y Bus de periféricos

- El diseño mas simple de un SoC es poner todos los periféricos del lado ***AHB***. Pero:
- Se incrementa la cantidad de cargas en el Bus incrementando la disipación de potencia.
- Los dispositivos mas lentos van a limitar la performance de todo el bus.

Bus de sistema y Bus de periféricos

- El diseño mas simple de un SoC es poner todos los periféricos del lado ***AHB***. Pero:
- Se incrementa la cantidad de cargas en el Bus incrementando la disipación de potencia.
- Los dispositivos mas lentos van a limitar la performance de todo el bus.
- Los dispositivos mas lentos requieren latchear señales de address y control, y el ***AHB*** está preparado en oposición para realizar transacciones pipeline.

Bus de sistema y Bus de periféricos

- El diseño mas simple de un SoC es poner todos los periféricos del lado ***AHB***. Pero:
- Se incrementa la cantidad de cargas en el Bus incrementando la disipación de potencia.
- Los dispositivos mas lentos van a limitar la performance de todo el bus.
- Los dispositivos mas lentos requieren latchear señales de address y control, y el ***AHB*** está preparado en oposición para realizar transacciones pipeline.
- El bus ***AHB*** es sincrónico mientras que muchos de los periféricos lentos son asincrónicos.

Bus de sistema y Bus de periféricos

- El diseño mas simple de un SoC es poner todos los periféricos del lado ***AHB***. Pero:
- Se incrementa la cantidad de cargas en el Bus incrementando la disipación de potencia.
- Los dispositivos mas lentos van a limitar la performance de todo el bus.
- Los dispositivos mas lentos requieren latchear señales de address y control, y el ***AHB*** está preparado en oposición para realizar transacciones pipeline.
- El bus ***AHB*** es sincrónico mientras que muchos de los periféricos lentos son asincrónicos.
- Generalmente los periféricos sencillos se activan con una señal de strobe y luego con la de lectura / escritura, sin requerimientos de clock, ni alta frecuencia.

Que conectar en cada Bus

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:

✓ Bus Masters

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:

✓ Bus Masters

✓ Memorias on-chip

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:

✓ Bus Masters

✓ Memorias on-chip

✓ Interfaces con Memoria externa.

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:

✓ Bus Masters

✓ Memorias on-chip

✓ Interfaces con Memoria externa.

✓ Periféricos que requieren gran ancho de banda y poseen interfaces FIFO interfaces

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:

✓ Bus Masters

✓ Memorias on-chip

✓ Interfaces con Memoria externa.

✓ Periféricos que requieren gran ancho de banda y poseen interfaces FIFO interfaces

✓ Periféricos Slave con DMA.

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:

✓ Bus Masters

✓ Memorias on-chip

✓ Interfaces con Memoria externa.

✓ Periféricos que requieren gran ancho de banda y poseen interfaces FIFO interfaces

✓ Periféricos Slave con DMA.

- Que se recomienda conectar en el ***APB***:

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:

✓ Bus Masters

✓ Memorias on-chip

✓ Interfaces con Memoria externa.

✓ Periféricos que requieren gran ancho de banda y poseen interfaces FIFO interfaces

✓ Periféricos Slave con DMA.

- Que se recomienda conectar en el ***APB***:

✓ Dispositivos Slave simples register mapped

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:

✓ Bus Masters

✓ Memorias on-chip

✓ Interfaces con Memoria externa.

✓ Periféricos que requieren gran ancho de banda y poseen interfaces FIFO interfaces

✓ Periféricos Slave con DMA.

- Que se recomienda conectar en el ***APB***:

✓ Dispositivos Slave simples register mapped

✓ Interfaces de muy baja potencia que requieren que no se rutee en forma global el clock.

Que conectar en cada Bus

- Que se recomienda conectar en el ***AHB***:
  - ✓ Bus Masters
  - ✓ Memorias on-chip
  - ✓ Interfaces con Memoria externa.
  - ✓ Periféricos que requieren gran ancho de banda y poseen interfaces FIFO interfaces
  - ✓ Periféricos Slave con DMA.
- Que se recomienda conectar en el ***APB***:
  - ✓ Dispositivos Slave simples register mapped
  - ✓ Interfaces de muy baja potencia que requieren que no se rutee en forma global el clock.
  - ✓ Agrupamientos de periféricos narrow-bus que eviten cargar al bus del sistema.

Temario

Terminología AMBA

Terminología AMBA

- Los diferentes buses manejan un grupo de señales del mismo tipo y que por lo tanto poseen el mismo nombre genérico

Terminología AMBA

- Los diferentes buses manejan un grupo de señales del mismo tipo y que por lo tanto poseen el mismo nombre genérico
- La diferenciación se hace mediante un simple sistema de prefijos

Terminología AMBA

- Los diferentes buses manejan un grupo de señales del mismo tipo y que por lo tanto poseen el mismo nombre genérico
- La diferenciación se hace mediante un simple sistema de prefijos
  - H Para las señales del ***AHB***

Terminología AMBA

- Los diferentes buses manejan un grupo de señales del mismo tipo y que por lo tanto poseen el mismo nombre genérico
- La diferenciación se hace mediante un simple sistema de prefijos
  - H Para las señales del ***AHB***
  - S Para las señales del ***ASB***

Terminología AMBA

- Los diferentes buses manejan un grupo de señales del mismo tipo y que por lo tanto poseen el mismo nombre genérico
- La diferenciación se hace mediante un simple sistema de prefijos
  - H Para las señales del ***AHB***
  - S Para las señales del ***ASB***
  - P Para las señales del ***APB***

Terminología AMBA

- Los diferentes buses manejan un grupo de señales del mismo tipo y que por lo tanto poseen el mismo nombre genérico
- La diferenciación se hace mediante un simple sistema de prefijos
  - H Para las señales del ***AHB***
  - S Para las señales del ***ASB***
  - P Para las señales del ***APB***
  - A Para las señales del ***AXI***

Terminología AMBA

- Los diferentes buses manejan un grupo de señales del mismo tipo y que por lo tanto poseen el mismo nombre genérico
- La diferenciación se hace mediante un simple sistema de prefijos
  - H Para las señales del ***AHB***
  - S Para las señales del ***ASB***
  - P Para las señales del ***APB***
  - A Para las señales del ***AXI***
- Por ejemplo ***HADDR[31:0]*** y ***PADDR[31:0]*** son las líneas de Address del ***AHB*** y ***APB*** respectivamente.

Terminología AMBA

- Los diferentes buses manejan un grupo de señales del mismo tipo y que por lo tanto poseen el mismo nombre genérico
- La diferenciación se hace mediante un simple sistema de prefijos
  - H Para las señales del ***AHB***
  - S Para las señales del ***ASB***
  - P Para las señales del ***APB***
  - A Para las señales del ***AXI***
- Por ejemplo ***HADDR[31:0]*** y ***PADDR[31:0]*** son las líneas de Address del ***AHB*** y ***APB*** respectivamente.
- En la implementaciones AXI el Bus de Address se separa para lectura y escritura, no obstante respeta las terminologías, siendo respectivamente ***ARADDR[31-0]*** y ***AWADDR[31-0]***

Lista de Señales AHB

Nombre	Fuente	Descripción

Lista de Señales AHB

Nombre	Fuente	Descripción
HCLK Bus clock	Clock source	Temporiza todas las transferencias. Todas las señales temporizan en el flanco ascendente de **HCLK**

Lista de Señales AHB

Nombre	Fuente	Descripción
HCLK Bus clock	Clock source	Temporiza todas las transferencias. Todas las señales temporizan en el flanco ascendente de **HCLK**
HRESETn Reset	Controlador de Reset	Señal activa Baja utilizada para resetear al sistema y al Bus. Es la única señal activa Baja.

Lista de Señales AHB

Nombre	Fuente	Descripción
HCLK Bus clock	Clock source	Temporiza todas las transferencias. Todas las señales temporizan en el flanco ascendente de **HCLK**
HRESETn Reset	Controlador de Reset	Señal activa Baja utilizada para resetear al sistema y al Bus. Es la única señal activa Baja.
HADDR[31:0] Address bus	Master	Address bus de 32 bits del Sistema.

Lista de Señales AHB

Nombre	Fuente	Descripción
HCLK Bus clock	Clock source	Temporiza todas las transferencias. Todas las señales temporizan en el flanco ascendente de **HCLK**
HRESETn Reset	Controlador de Reset	Señal activa Baja utilizada para resetear al sistema y al Bus. Es la única señal activa Baja.
HADDR[31:0] Address bus	Master	Address bus de 32 bits del Sistema.
HTRANS[1:0] Transfer type	Master	Indica el tipo de transferencia en curso, que puede ser **NONSEQUENTIAL**, **SEQUENTIAL**, **IDLE** o **BUSY**.

Lista de Señales AHB

Nombre	Fuente	Descripción
HCLK Bus clock	Clock source	Temporiza todas las transferencias. Todas las señales temporizan en el flanco ascendente de **HCLK**
HRESETn Reset	Controlador de Reset	Señal activa Baja utilizada para resetear al sistema y al Bus. Es la única señal activa Baja.
HADDR[31:0] Address bus	Master	Address bus de 32 bits del Sistema.
HTRANS[1:0] Transfer type	Master	Indica el tipo de transferencia en curso, que puede ser **NONSEQUENTIAL**, **SEQUENTIAL**, **IDLE** o **BUSY**.
HWRITE Transfer dir	Master	Cuando es Alta esta señal indica una transferencia de escritura y cuando es Baja una transferencia de lectura.

Lista de Señales AHB

Nombre	Fuente	Descripción
HCLK Bus clock	Clock source	Temporiza todas las transferencias. Todas las señales temporizan en el flanco ascendente de **HCLK**
HRESETn Reset	Controlador de Reset	Señal activa Baja utilizada para resetear al sistema y al Bus. Es la única señal activa Baja.
HADDR[31:0] Address bus	Master	Address bus de 32 bits del Sistema.
HTRANS[1:0] Transfer type	Master	Indica el tipo de transferencia en curso, que puede ser **NONSEQUENTIAL**, **SEQUENTIAL**, **IDLE** o **BUSY**.
HWRITE Transfer dir	Master	Cuando es Alta esta señal indica una transferencia de escritura y cuando es Baja una transferencia de lectura.
HSIZE[2:0] Transfer size	Master	Indica el tamaño de la transferencia: byte (8-bit), half-word (16-bit) o word (32-bit). El protocolo permite un máximo de 1024 bits, para transferencias largas.

Lista de Señales AHB

Nombre	Fuente	Descripción
HCLK Bus clock	Clock source	Temporiza todas las transferencias. Todas las señales temporizan en el flanco ascendente de **HCLK**
HRESETn Reset	Controlador de Reset	Señal activa Baja utilizada para resetear al sistema y al Bus. Es la única señal activa Baja.
HADDR[31:0] Address bus	Master	Address bus de 32 bits del Sistema.
HTRANS[1:0] Transfer type	Master	Indica el tipo de transferencia en curso, que puede ser **NONSEQUENTIAL**, **SEQUENTIAL**, **IDLE** o **BUSY**.
HWRITE Transfer dir	Master	Cuando es Alta esta señal indica una transferencia de escritura y cuando es Baja una transferencia de lectura.
HSIZE[2:0] Transfer size	Master	Indica el tamaño de la transferencia: byte (8-bit), half-word (16-bit) o word (32-bit). El protocolo permite un máximo de 1024 bits, para transferencias largas.
HBURST[2:0] Burst type	Master	Indica si la transferencia forma parte de una ráfaga (**burst**). Tamaños de burst soportados (**beat bursts**): 4, 8 y 16. Y puede ser incremental o wrapping.

Lista de Señales AHB

Nombre	Fuente	Descripción
HPROT[3:0] Protection control	Master	Proveen información adicional del acceso al bus, estando en principio previstas para ser utilizadas por cualquier módulo que necesite implementar algún nivel de protección. Indican si la transferencia es un **opcode fetch** o un acceso a datos, como también si la transferencia se realiza en modo privilegiado o en modo usuario. Para Masters con MMU estas señales también indican si la dirección actual es cacheable o bufferable.

Lista de Señales AHB

Nombre	Fuente	Descripción
HPROT[3:0] Protection control	Master	Proveen información adicional del acceso al bus, estando en principio previstas para ser utilizadas por cualquier módulo que necesite implementar algún nivel de protección. Indican si la transferencia es un **opcode fetch** o un acceso a datos, como también si la transferencia se realiza en modo privilegiado o en modo usuario. Para Masters con MMU estas señales también indican si la dirección actual es cacheable o bufferable.
HWDATA[31:0] Write data bus	Master	Se emplea para transferir datos desde el Master a los Slaves durante operaciones de escritura. Se recomiendo un ancho de bus mínimo de 32 bits. Este ancho puede ampliarse fácilmente para extender la operación a mayor ancho de banda.

Lista de Señales AHB

Nombre	Fuente	Descripción
HPROT[3:0] Protection control	Master	Proveen información adicional del acceso al bus, estando en principio previstas para ser utilizadas por cualquier módulo que necesite implementar algún nivel de protección. Indican si la transferencia es un **opcode fetch** o un acceso a datos, como también si la transferencia se realiza en modo privilegiado o en modo usuario. Para Masters con MMU estas señales también indican si la dirección actual es cacheable o bufferable.
HWDATA[31:0] Write data bus	Master	Se emplea para transferir datos desde el Master a los Slaves durante operaciones de escritura. Se recomiendo un ancho de bus mínimo de 32 bits. Este ancho puede ampliarse fácilmente para extender la operación a mayor ancho de banda.
HSELx Slave select	Decoder	Cada Slave AHB tiene una señal de Chip Select que le indica que es el target de la transacción en curso. Esta señal se obtiene decodificando mediante lógica combinatoria las líneas de Address.

Lista de Señales AHB

Nombre	Fuente	Descripción
`HRDATA[31:0]` Read data bus	Slave	Es el camino de datos para transferencias desde un Slave hacia los Masters durante operaciones de lectura. Se recomiendo como mínimo 32 bits de ancho. Este ancho puede ampliarse fácilmente para extender la operación a mayor ancho de banda.

Lista de Señales AHB

Nombre	Fuente	Descripción
HRDATA[31:0] Read data bus	Slave	Es el camino de datos para transferencias desde un Slave hacia los Masters durante operaciones de lectura. Se recomiendo como mínimo 32 bits de ancho. Este ancho puede ampliarse fácilmente para extender la operación a mayor ancho de banda.
HREADY Transfer done	Slave	Cuando **HREADY** va al estado Alto indica que la transferencia en el Bus terminó. Llevándola en estado BAJO se puede extender la transferencia. Nota: Los Slaves conectados al bus toman **HREADY** como señal de entrada y de salida.

Lista de Señales AHB

Nombre	Fuente	Descripción
HRDATA[31:0] Read data bus	Slave	Es el camino de datos para transferencias desde un Slave hacia los Masters durante operaciones de lectura. Se recomiendo como mínimo 32 bits de ancho. Este ancho puede ampliarse fácilmente para extender la operación a mayor ancho de banda.
HREADY Transfer done	Slave	Cuando **HREADY** va al estado Alto indica que la transferencia en el Bus terminó. Llevándola en estado BAJO se puede extender la transferencia. Nota: Los Slaves conectados al bus toman **HREADY** como señal de entrada y de salida.
HRESP[1:0] Transfer response	Slave	Proporciona información adicional del estado de la transferencia. Hay cuatro respuestas posibles: **OKAY**, **ERROR**, **RETRY** y **SPLIT**.

Lista de Señales AHB para Multimaster

Nombre	Fuente	Descripción
HBUSREQx Bus Request	Master	Señal de un Master x al Árbitro de Bus para solicitarle el Bus. El árbitro tiene una señal como ésta por cada Master. Máximo 16.

Lista de Señales AHB para Multimaster

Nombre	Fuente	Descripción
HBUSREQx Bus Request	Master	Señal de un Master x al Árbitro de Bus para solicitarle el Bus. El árbitro tiene una señal como ésta por cada Master. Máximo 16.
HLOCKx Locked Transfers	Master	Cuando va al estado Alto indica el Master requiere bloquear el acceso al bus a cualquier otro Master. Cualquier otro Master que solicite el bus tiene que esperar que ésta línea tome estado BAJO.

Lista de Señales AHB para Multimaster

Nombre	Fuente	Descripción
HBUSREQx Bus Request	Master	Señal de un Master x al Árbitro de Bus para solicitarle el Bus. El árbitro tiene una señal como ésta por cada Master. Máximo 16.
HLOCKx Locked Transfers	Master	Cuando va al estado Alto indica el Master requiere bloquear el acceso al bus a cualquier otro Master. Cualquier otro Master que solicite el bus tiene que esperar que ésta línea tome estado BAJO.
HGRANTx Bus Grant	Árbitro	Indica que el Master x del Bus es de todos el que mayor prioridad tienen a partir de este momento. La posesión de las líneas de Address y Control finaliza cuando se activa (Alta) **HREADY**. De modo que un Master consigue acceso al Bus cuando **HGRANTx** y **HREADY** están Altas.

Lista de Señales AHB para Multimaster

Nombre	Fuente	Descripción
HMASTER[3:0] Master Number	Árbitro	Indica que Master de Bus está cursando la transferencia en el Bus. Es utilizada por los Slaves que soportan tipos de transferencias **SPLIT**para determinar que Master está intentando el acceso. El Timing de esta señal está alineado con el de las señales de Address y Control.

Lista de Señales AHB para Multimaster

Nombre	Fuente	Descripción
HMASTER[3:0] Master Number	Árbitro	Indica que Master de Bus está cursando la transferencia en el Bus. Es utilizada por los Slaves que soportan tipos de transferencias **SPLIT**para determinar que Master está intentando el acceso. El Timing de esta señal está alineado con el de las señales de Address y Control.
HMASTLOCK Locked sequence	Árbitro	Indica que el Master actualmente en poder del Bus está cursando una transferencia con el bus lockeado. Tiene el mismo Timing que **HMASTLOCK**

Lista de Señales AHB para Multimaster

Nombre	Fuente	Descripción
HMASTER[3:0] Master Number	Árbitro	Indica que Master de Bus está cursando la transferencia en el Bus. Es utilizada por los Slaves que soportan tipos de transferencias **SPLIT**para determinar que Master está intentando el acceso. El Timing de esta señal está alineado con el de las señales de Address y Control.
HMASTLOCK Locked sequence	Árbitro	Indica que el Master actualmente en poder del Bus está cursando una transferencia con el bus lockeado. Tiene el mismo Timing que **HMASTLOCK**
HSPLITx[15:0 Split completion request	Slave (con SPLIT)	Este Bus de 16 bits sirve para que un Slave le indique al Árbitro cual de los Masters se debe rehabilitar para reintentar una transferencia **SPLIT**. *Cada uno de estos bits identifica a uno de los Masters*.

Lista de Señales APB

Nombre	Descripción
PCLK Bus clock	El flanco ascendente de **PCLK** se usa para temporizar todas las transferencias sobre APB.

Lista de Señales APB

Nombre	Descripción
PCLK Bus clock	El flanco ascendente de **PCLK** se usa para temporizar todas las transferencias sobre APB.
PRESETn APB reset	La señal de reset del APB es activa Baja y normalmente se conecta directamente a la señal de Reset del bus.

Lista de Señales APB

Nombre	Descripción
PCLK Bus clock	El flanco ascendente de **PCLK** se usa para temporizar todas las transferencias sobre APB.
PRESETn APB reset	La señal de reset del APB es activa Baja y normalmente se conecta directamente a la señal de Reset del bus.
PADDR[31:0] APB address bus	Es el Bus de Direcciones del APB, que puede tener hasta 32 bits de ancho y es manejado por la Unidad de Bridge con el AHB.

Lista de Señales APB

Nombre	Descripción
PCLK Bus clock	El flanco ascendente de **PCLK** se usa para temporizar todas las transferencias sobre APB.
PRESETn APB reset	La señal de reset del APB es activa Baja y normalmente se conecta directamente a la señal de Reset del bus.
PADDR[31:0] APB address bus	Es el Bus de Direcciones del APB, que puede tener hasta 32 bits de ancho y es manejado por la Unidad de Bridge con el AHB.
PSELx APB select	Es una señal desde cada decodificador secundario dentro de la Unidad de Bridge del Bus de Periféricos, hasta cada periférico Slave x del Bus. Esta señal indica al dispositivo Slave que ha sido seleccionado y que se requiere una transferencia de Datos. Debe haber una señal **PSELx** por cada dispositivo Slave.

Lista de Señales APB

Nombre	Descripción
PENABLE APB strobe	Esta señal de strobe se emplea para temporizar todos los accesos sobre los periféricos del Bus. La señal de habilitación se utiliza para indicar el inicio del segundo ciclo de una transferencia APB. El ciclo ascendente de **PENABLE** ocurre en el medio de una transferencia APB.

Lista de Señales APB

Nombre	Descripción
PENABLE APB strobe	Esta señal de strobe se emplea para temporizar todos los accesos sobre los periféricos del Bus. La señal de habilitación se utiliza para indicar el inicio del segundo ciclo de una transferencia APB. El ciclo ascendente de **PENABLE** ocurre en el medio de una transferencia APB.
PWRITE APB transfer dir	Cuando es Alta, esta señal indica un acceso de escritura en el APB, y cuando es Baja un acceso de lectura.

Lista de Señales APB

Nombre	Descripción
PENABLE APB strobe	Esta señal de strobe se emplea para temporizar todos los accesos sobre los periféricos del Bus. La señal de habilitación se utiliza para indicar el inicio del segundo ciclo de una transferencia APB. El ciclo ascendente de **PENABLE** ocurre en el medio de una transferencia APB.
PWRITE APB transfer dir	Cuando es Alta, esta señal indica un acceso de escritura en el APB, y cuando es Baja un acceso de lectura.
PRDATA APB read data bus	El bus de datos de lectura es controlado por el Slave durante los ciclos de lectura (**PWRITE** Baja). El Bus de datos de lectura puede tener hasta 32 bits de ancho.

Lista de Señales APB

Nombre	Descripción
PENABLE APB strobe	Esta señal de strobe se emplea para temporizar todos los accesos sobre los periféricos del Bus. La señal de habilitación se utiliza para indicar el inicio del segundo ciclo de una transferencia APB. El ciclo ascendente de **PENABLE** ocurre en el medio de una transferencia APB.
PWRITE APB transfer dir	Cuando es Alta, esta señal indica un acceso de escritura en el APB, y cuando es Baja un acceso de lectura.
PRDATA APB read data bus	El bus de datos de lectura es controlado por el Slave durante los ciclos de lectura (**PWRITE** Baja). El Bus de datos de lectura puede tener hasta 32 bits de ancho.
PWDATA APB write data bus	El bus de datos de escritura es manejado por el Bridge del Bus de Periféricos durante ciclos de escritura (PWRITE Alta). Este bus puede tener hasta 32 bits de ancho.

Temario

Sistema de Interconexión

Sistema de Interconexión

- El protocolo AMBA AHB está pensado para funcionar en base a un esquema de interconexión basado en un Multiplexor centralizado.

Sistema de Interconexión

- El protocolo AMBA AHB está pensado para funcionar en base a un esquema de interconexión basado en un Multiplexor centralizado.
- De tal modo que todos los Masters ponen sus señales de Address y Control indicando el tipo de transferencia que desean cursar.

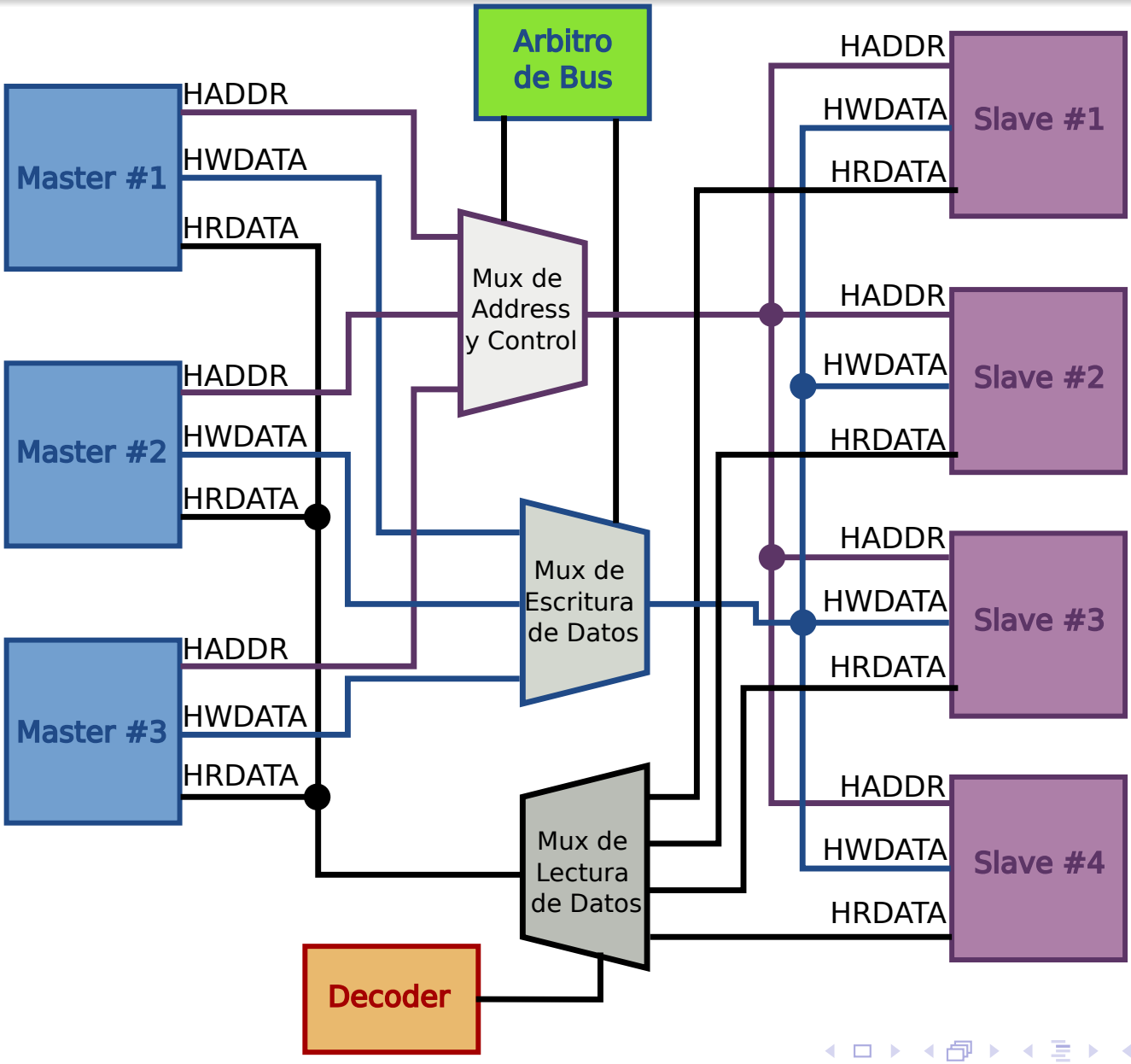
Sistema de Interconexión

- El protocolo AMBA AHB está pensado para funcionar en base a un esquema de interconexión basado en un Multiplexor centralizado.
- De tal modo que todos los Masters ponen sus señales de Address y Control indicando el tipo de transferencia que desean cursar.
- Un Árbitro de bus selecciona el Bus de Address y Control de uno de los Master para rutearlo hacia los slaves.

Sistema de Interconexión

- El protocolo AMBA AHB está pensado para funcionar en base a un esquema de interconexión basado en un Multiplexor centralizado.
- De tal modo que todos los Masters ponen sus señales de Address y Control indicando el tipo de transferencia que desean cursar.
- Un Árbitro de bus selecciona el Bus de Address y Control de uno de los Master para rutearlo hacia los slaves.
- Se requiere además de un decodificador centralizado que controle los datos leídos y las señales de respuesta del Multiplexor, que seleccionará las señales del Slave involucrado en la transferencia.

Arquitectura detallada de AHB



Operación del Bus AHB

Operación del Bus AHB

- Un Master inicia una transferencia AHB, una vez que se le haya concedido el bus (“granted”).

Operación del Bus AHB

- Un Master inicia una transferencia AHB, una vez que se le haya concedido el bus (“granted”).
- Cuando un Master necesita iniciar una transferencia, señaliza al Árbitro de Bus: ***Request***

Operación del Bus AHB

- Un Master inicia una transferencia AHB, una vez que se le haya concedido el bus (“granted”).
- Cuando un Master necesita iniciar una transferencia, señaliza al Árbitro de Bus: ***Request***
- El Árbitro le indica al Master cuando puede utilizar el bus: ***Grant***. Condición indispensable para inicio de una transferencia.

Operación del Bus AHB

- Un Master inicia una transferencia AHB, una vez que se le haya concedido el bus (“granted”).
- Cuando un Master necesita iniciar una transferencia, señaliza al Árbitro de Bus: ***Request***
- El Árbitro le indica al Master cuando puede utilizar el bus: ***Grant***. Condición indispensable para inicio de una transferencia.
- El Master inicia la transferencia colocando en el Bus la información necesaria: dirección a la que necesita acceder, el sentido y el ancho de la transferencia, y si ésta forma parte de un Burst.

Operación del Bus AHB

- Un Master inicia una transferencia AHB, una vez que se le haya concedido el bus (“granted”).
- Cuando un Master necesita iniciar una transferencia, señaliza al Árbitro de Bus: ***Request***
- El Árbitro le indica al Master cuando puede utilizar el bus: ***Grant***. Condición indispensable para inicio de una transferencia.
- El Master inicia la transferencia colocando en el Bus la información necesaria: dirección a la que necesita acceder, el sentido y el ancho de la transferencia, y si ésta forma parte de un Burst.
- Una transferencia Burst puede ser incremental o wrapping dependiendo de como opera al alcanzar la dirección límite.

Operación del Bus AHB

- Un Master inicia una transferencia AHB, una vez que se le haya concedido el bus (“granted”).
- Cuando un Master necesita iniciar una transferencia, señaliza al Árbitro de Bus: ***Request***
- El Árbitro le indica al Master cuando puede utilizar el bus: ***Grant***. Condición indispensable para inicio de una transferencia.
- El Master inicia la transferencia colocando en el Bus la información necesaria: dirección a la que necesita acceder, el sentido y el ancho de la transferencia, y si ésta forma parte de un Burst.
- Una transferencia Burst puede ser incremental o wrapping dependiendo de como opera al alcanzar la dirección límite.
- Cada transferencia consiste de un ciclo de Address y Control, seguido de uno o mas ciclos de datos.

Operación del Bus AHB

- Un Master inicia una transferencia AHB, una vez que se le haya concedido el bus (“granted”).
- Cuando un Master necesita iniciar una transferencia, señaliza al Árbitro de Bus: ***Request***
- El Árbitro le indica al Master cuando puede utilizar el bus: ***Grant***. Condición indispensable para inicio de una transferencia.
- El Master inicia la transferencia colocando en el Bus la información necesaria: dirección a la que necesita acceder, el sentido y el ancho de la transferencia, y si ésta forma parte de un Burst.
- Una transferencia Burst puede ser incremental o wrapping dependiendo de como opera al alcanzar la dirección límite.
- Cada transferencia consiste de un ciclo de Address y Control, seguido de uno o mas ciclos de datos.
- El ciclo de dirección no es extensible. Los slaves deben latchear la información de Address y Control.

Operación del Bus AHB

- El ciclo de datos está sujeto a la señal **HREADY**. Mientras es Baja se insertan ***wait states*** .

Operación del Bus AHB

- El ciclo de datos está sujeto a la señal **`HREADY`**. Mientras es Baja se insertan ***wait states*** .
- El Slave utiliza el par de líneas de respuesta **`HRESP[1:0]`**, para indicar el estado o resultado de la transferencia:

Operación del Bus AHB

- El ciclo de datos está sujeto a la señal **HREADY**. Mientras es Baja se insertan ***wait states*** .
- El Slave utiliza el par de líneas de respuesta **HRESP[1:0]**, para indicar el estado o resultado de la transferencia:

  OKAY La transferencia se cursa con normalidad. Cuando **HREADY** tome estado alto finalizará exitosamente.

Operación del Bus AHB

- El ciclo de datos está sujeto a la señal **HREADY**. Mientras es Baja se insertan ***wait states*** .
- El Slave utiliza el par de líneas de respuesta **HRESP[1:0]**, para indicar el estado o resultado de la transferencia:

  OKAY La transferencia se cursa con normalidad. Cuando **HREADY** tome estado alto finalizará exitosamente.

  ERROR Indica error en la transferencia.

Operación del Bus AHB

- El ciclo de datos está sujeto a la señal **HREADY**. Mientras es Baja se insertan ***wait states*** .
- El Slave utiliza el par de líneas de respuesta **HRESP[1:0]**, para indicar el estado o resultado de la transferencia:

  OKAY La transferencia se cursa con normalidad. Cuando **HREADY** tome estado alto finalizará exitosamente.

  ERROR Indica error en la transferencia.

  RETRY La transferencia no puede completarse de manera inmediata, pero el Master debe reintentarla.

Operación del Bus AHB

- El ciclo de datos está sujeto a la señal **HREADY**. Mientras es Baja se insertan ***wait states*** .
- El Slave utiliza el par de líneas de respuesta **HRESP[1:0]**, para indicar el estado o resultado de la transferencia:

OKAY La transferencia se cursa con normalidad. Cuando **HREADY** tome estado alto finalizará exitosamente.

ERROR Indica error en la transferencia.

RETRY La transferencia no puede completarse de manera inmediata, pero el Master debe reintentarla.

SPLIT Es similar a **RETRY**.

Operación del Bus AHB

- El ciclo de datos está sujeto a la señal **HREADY**. Mientras es Baja se insertan ***wait states*** .
- El Slave utiliza el par de líneas de respuesta **HRESP[1:0]**, para indicar el estado o resultado de la transferencia:

  OKAY La transferencia se cursa con normalidad. Cuando **HREADY** tome estado alto finalizará exitosamente.

  ERROR Indica error en la transferencia.

  RETRY La transferencia no puede completarse de manera inmediata, pero el Master debe reintentarla.

  SPLIT Es similar a **RETRY**.

- Lo normal es que una transferencia Burst se complete antes de que el Árbitro de Bus habilite a otro Master.

Operación del Bus AHB

- El ciclo de datos está sujeto a la señal **HREADY**. Mientras es Baja se insertan ***wait states*** .
- El Slave utiliza el par de líneas de respuesta **HRESP[1:0]**, para indicar el estado o resultado de la transferencia:

  OKAY La transferencia se cursa con normalidad. Cuando **HREADY** tome estado alto finalizará exitosamente.

  ERROR Indica error en la transferencia.

  RETRY La transferencia no puede completarse de manera inmediata, pero el Master debe reintentarla.

  SPLIT Es similar a **RETRY**.
- Lo normal es que una transferencia Burst se complete antes de que el Árbitro de Bus habilite a otro Master.
- En ocasiones para evitar demoras el Árbitro puede ordenar a un Master que interrumpa una transferencia Burst.

Operación del Bus AHB

- El ciclo de datos está sujeto a la señal **HREADY**. Mientras es Baja se insertan ***wait states*** .
- El Slave utiliza el par de líneas de respuesta **HRESP[1:0]**, para indicar el estado o resultado de la transferencia:

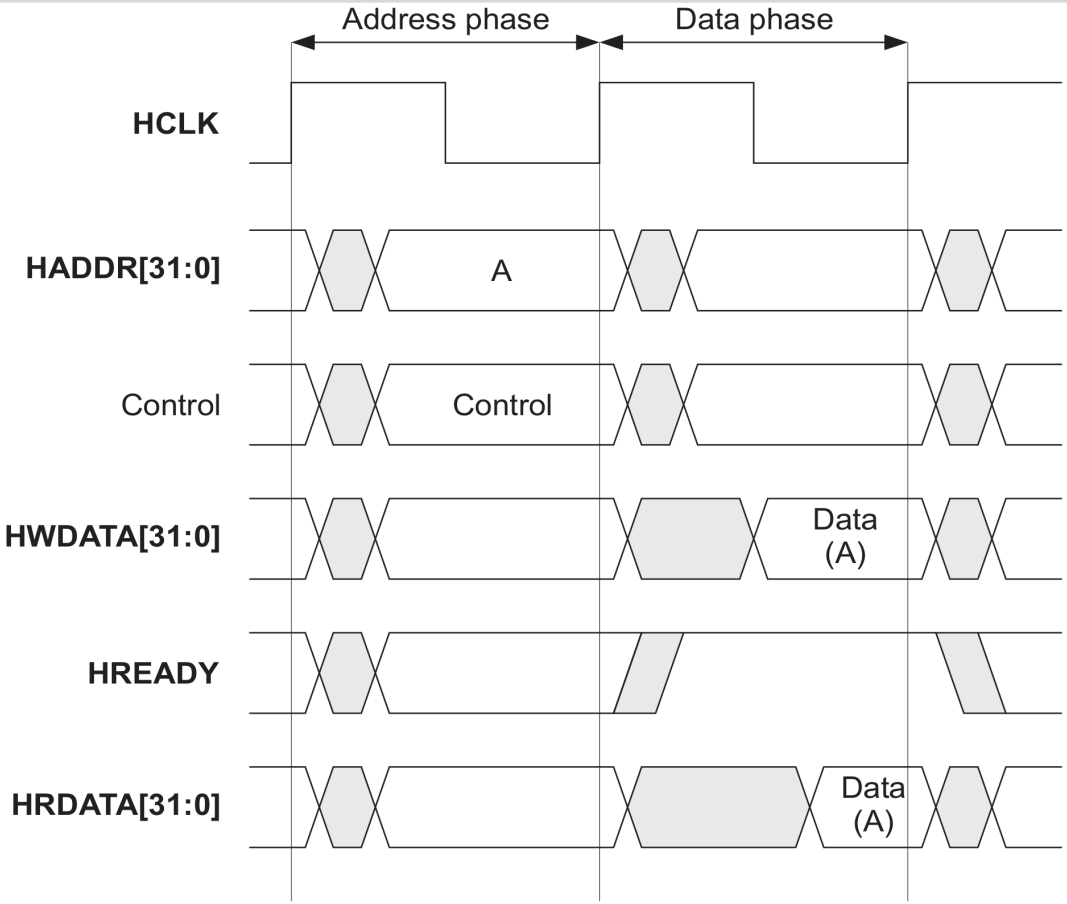
  OKAY La transferencia se cursa con normalidad. Cuando **HREADY** tome estado alto finalizará exitosamente.

  ERROR Indica error en la transferencia.

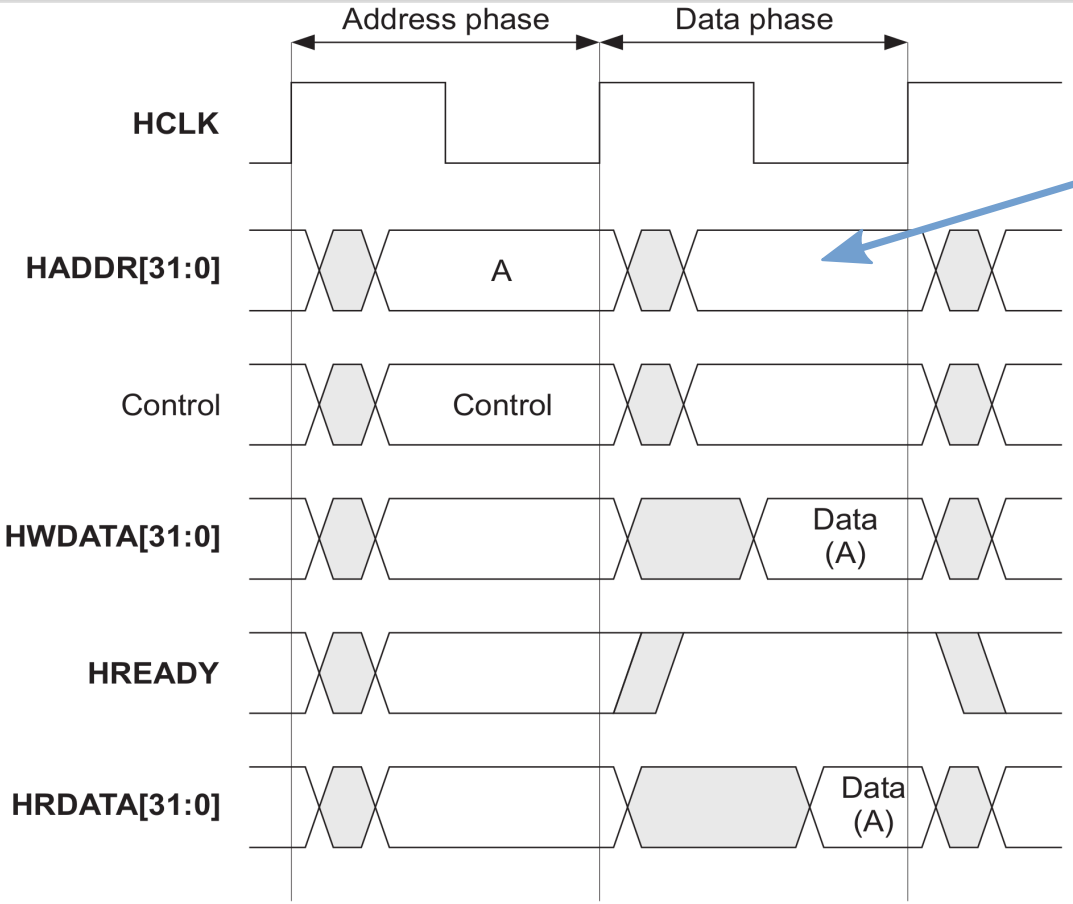
  RETRY La transferencia no puede completarse de manera inmediata, pero el Master debe reintentarla.

  SPLIT Es similar a **RETRY**.
- Lo normal es que una transferencia Burst se complete antes de que el Árbitro de Bus habilite a otro Master.
- En ocasiones para evitar demoras el Árbitro puede ordenar a un Master que interrumpa una transferencia Burst.
- El Master deberá rearbitrar la transferencia a posteriori.

Transferencia AHB Básica

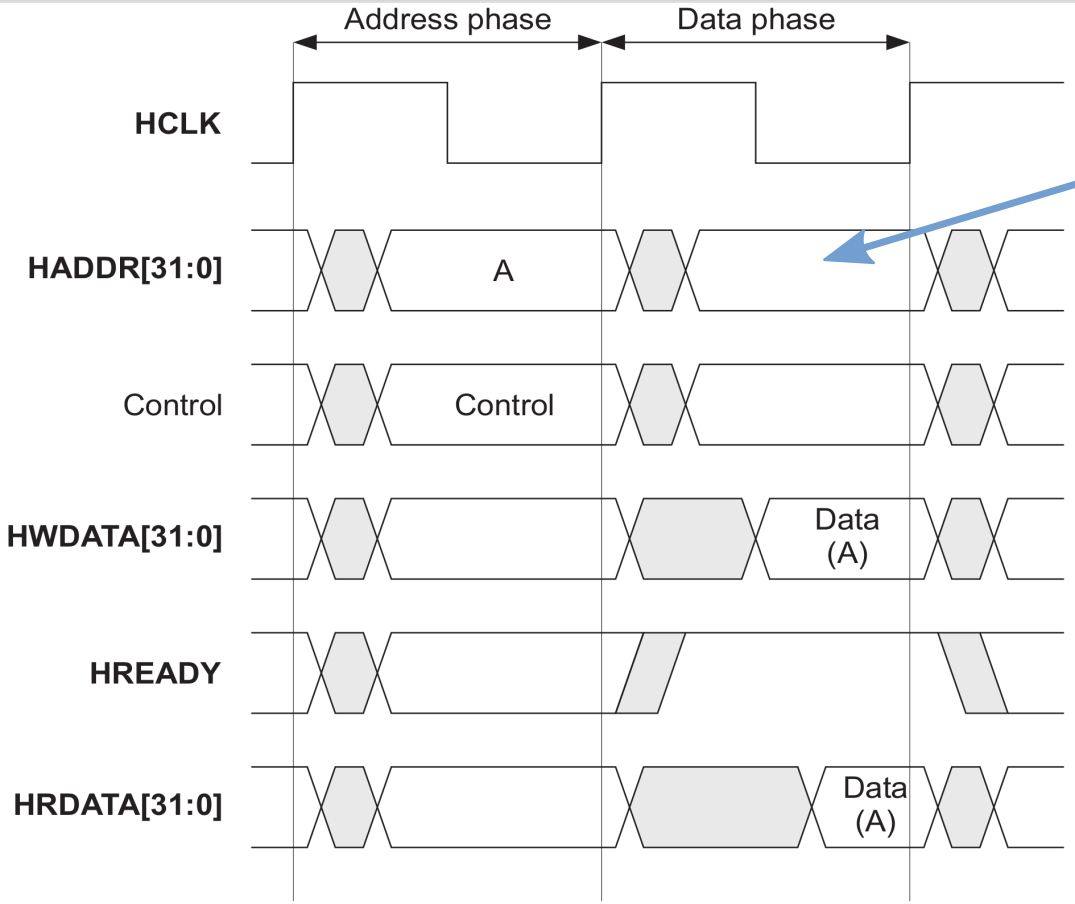


Transferencia AHB Básica



Durante la fase de datos de la transferencia en curso, se coloca la información de Address y Control de la próxima transferencia.

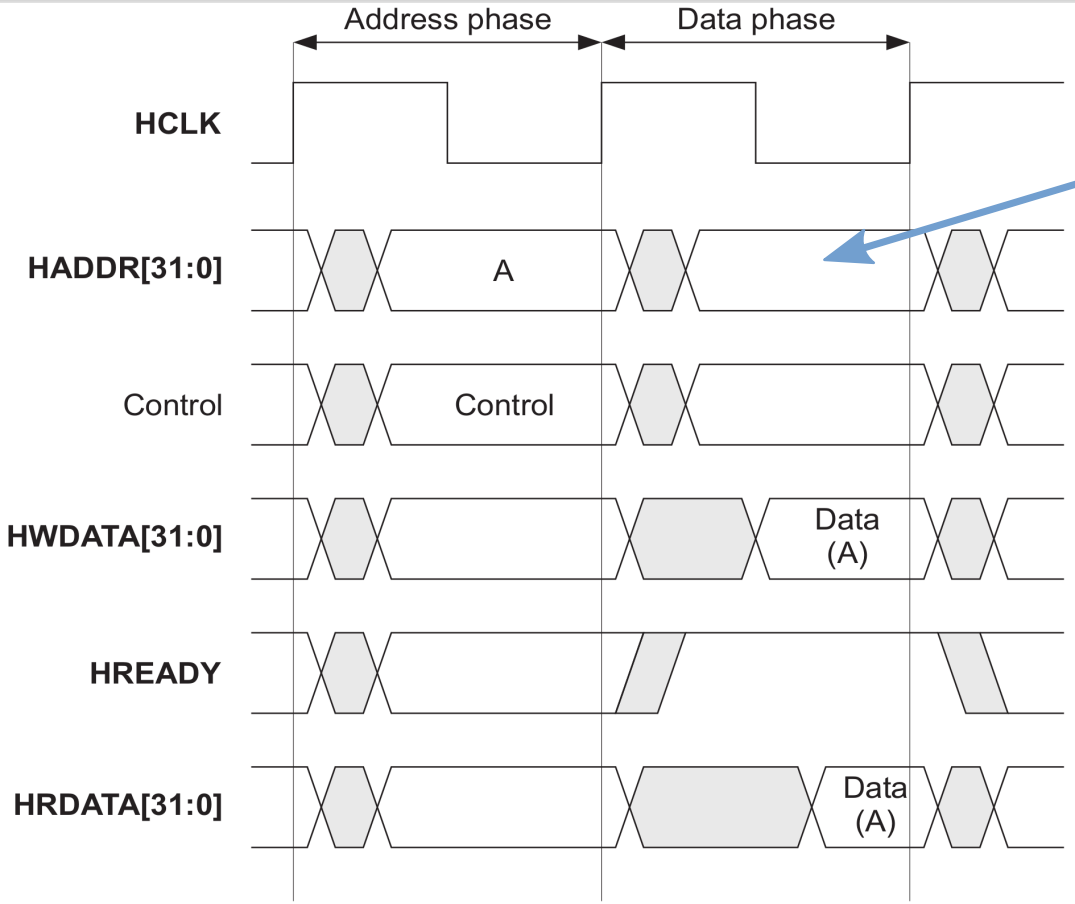
Transferencia AHB Básica



Durante la fase de datos de la transferencia en curso, se coloca la información de Address y Control de la próxima transferencia.

Este solapamiento de fases constituye la naturaleza pipeline del bus lo cual aumenta notablemente su rendimiento, sin requerir mas velocidad del Slave

Transferencia AHB Básica

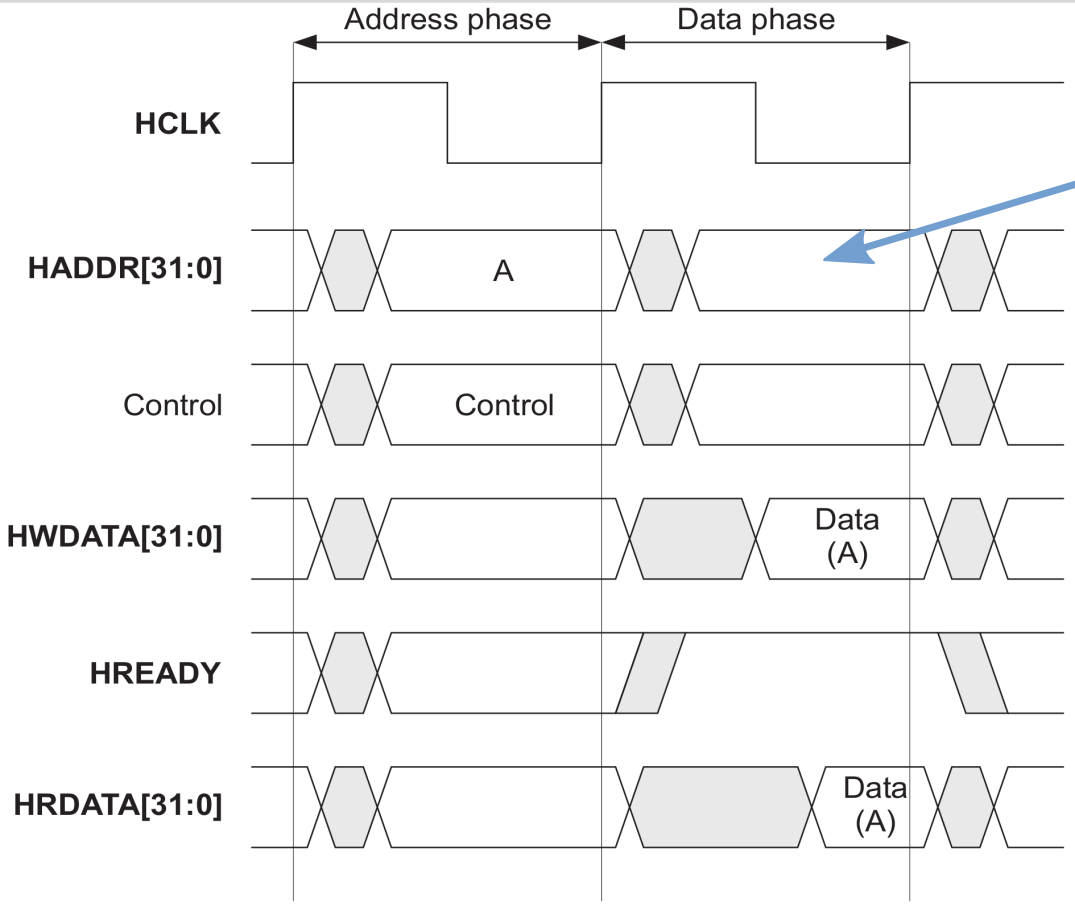


Durante la fase de datos de la transferencia en curso, se coloca la información de Address y Control de la próxima transferencia.

Este solapamiento de fases constituye la naturaleza pipeline del bus lo cual aumenta notablemente su rendimiento, sin requerir mas velocidad del Slave

El ciclo de Address / Control siempre dura un ciclo de **HCLCK**.

Transferencia AHB Básica

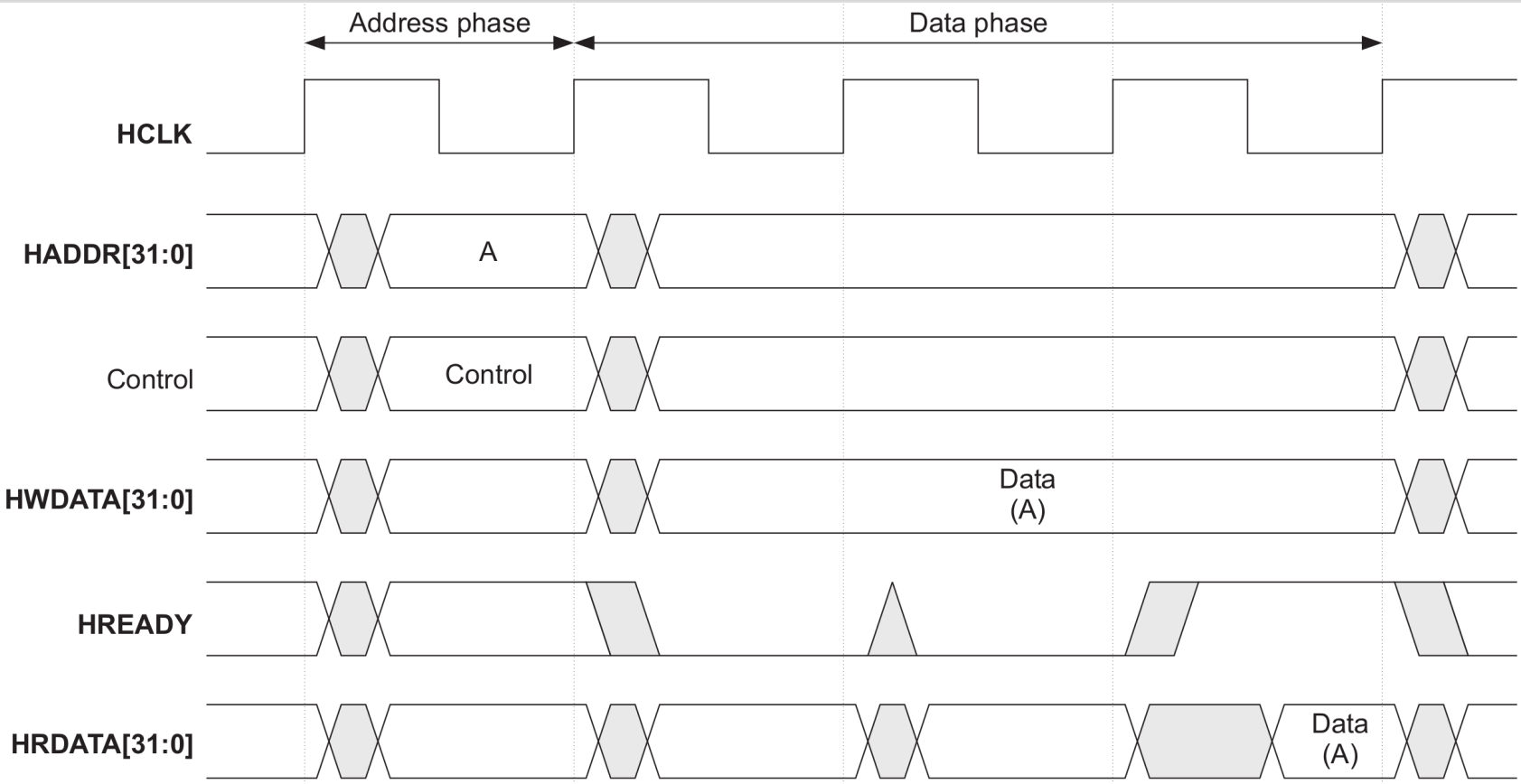


Durante la fase de datos de la transferencia en curso, se coloca la información de Address y Control de la próxima transferencia.

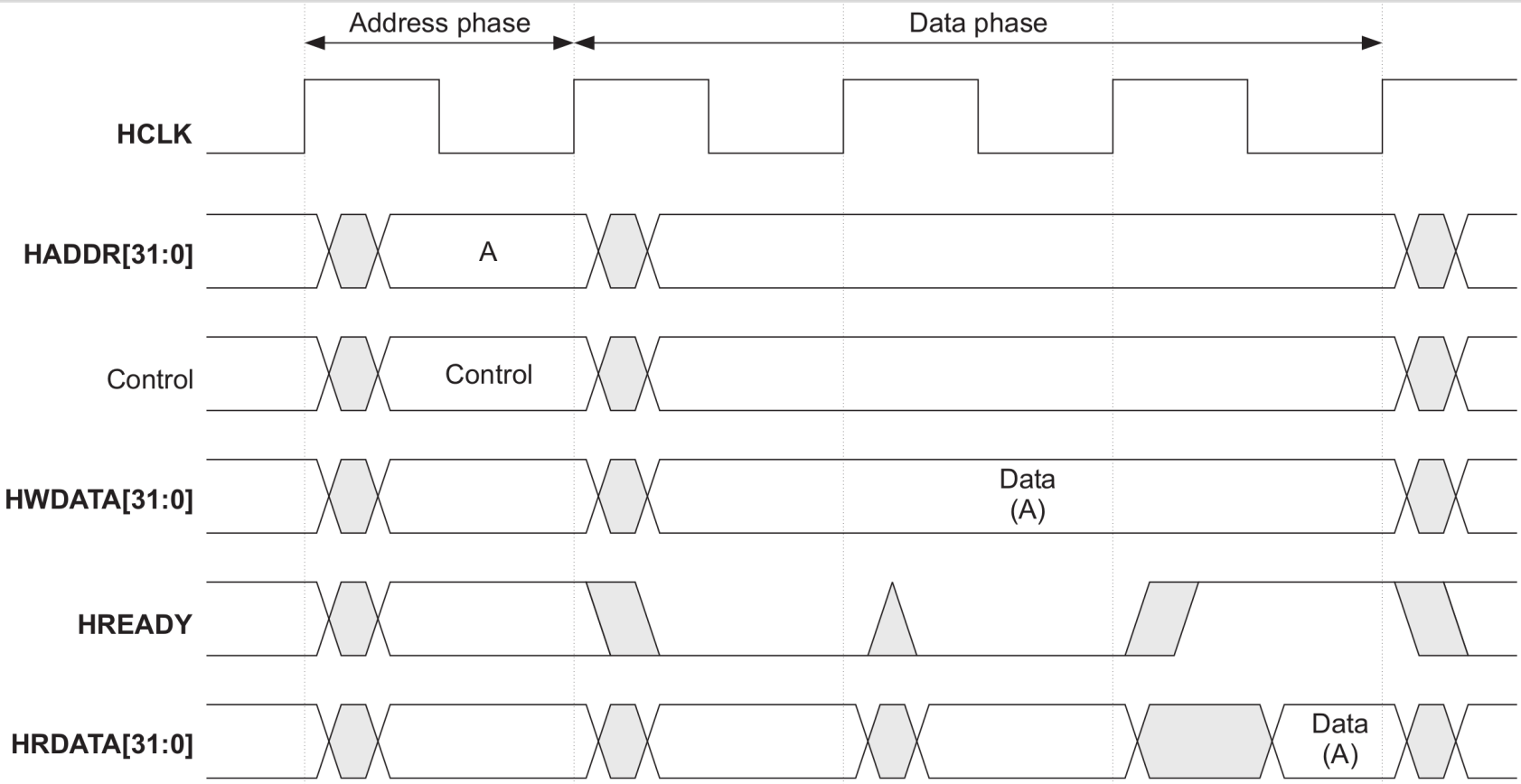
Este solapamiento de fases constituye la naturaleza pipeline del bus lo cual aumenta notablemente su rendimiento, sin requerir mas velocidad del Slave

El ciclo de Address / Control siempre dura un ciclo de **HCLCK**.
El Slave samplea siempre en el flanco ascendente de **HCLCK**.

Transferencia AHB Básica con wait states

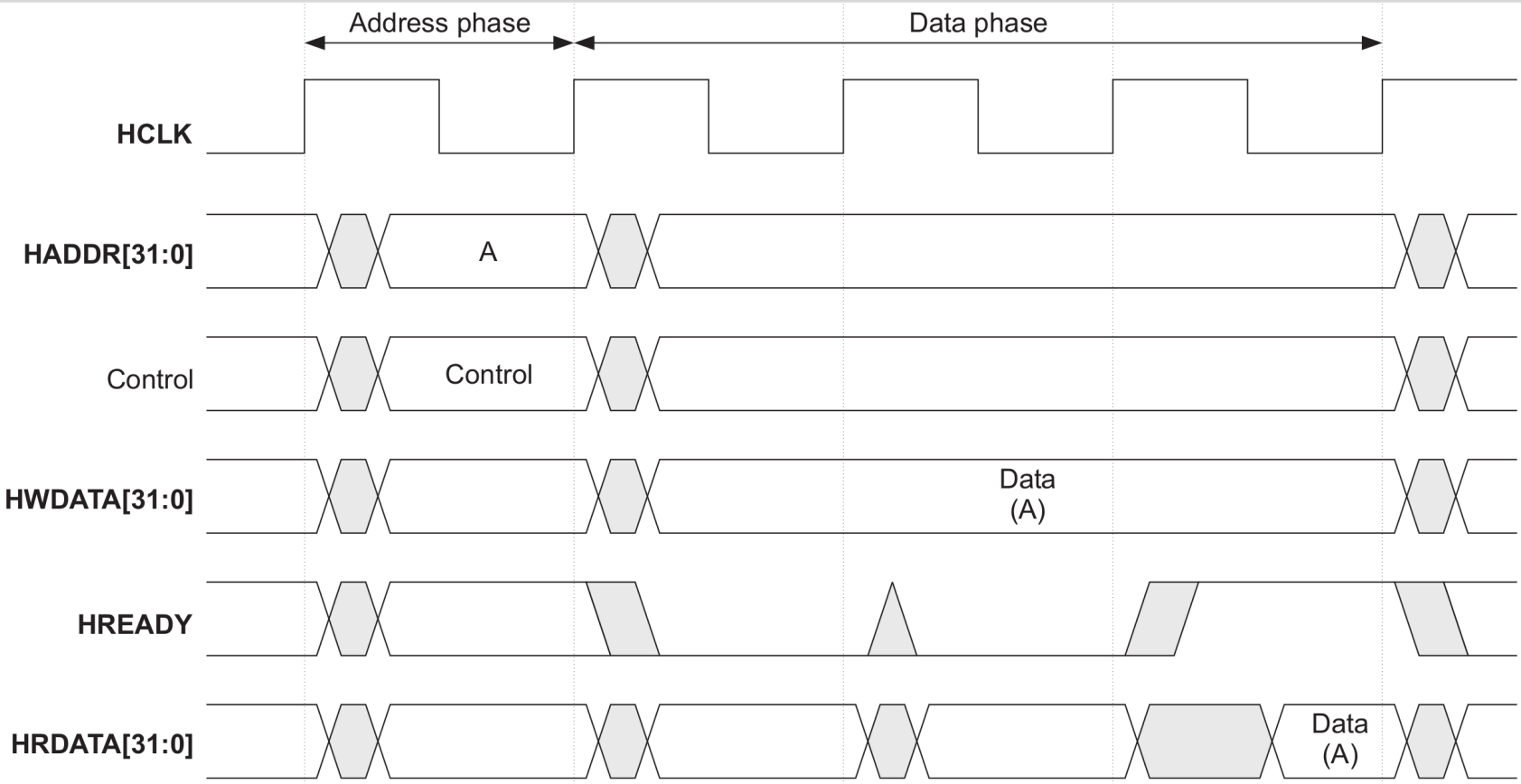


Transferencia AHB Básica con wait states



✓Si el Slave requiere mas tiempo para responder, tiene la señal **HREADY**

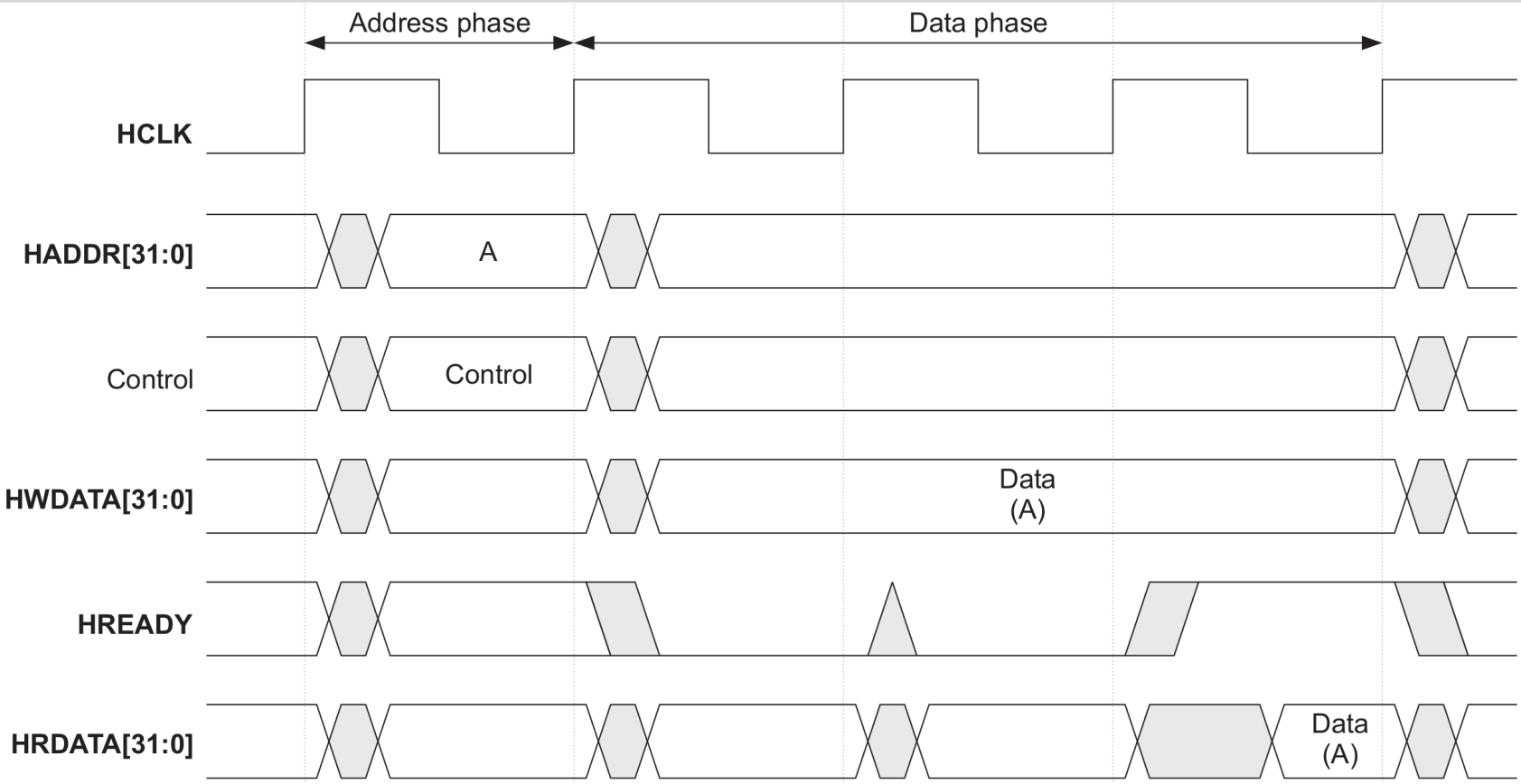
Transferencia AHB Básica con wait states



✓Si el Slave requiere mas tiempo para responder, tiene la señal **HREADY**

✓Si la transferencia es de escritura el Master debe sostener los datos válidos en el bus durante los ciclos extendidos.

Transferencia AHB Básica con wait states

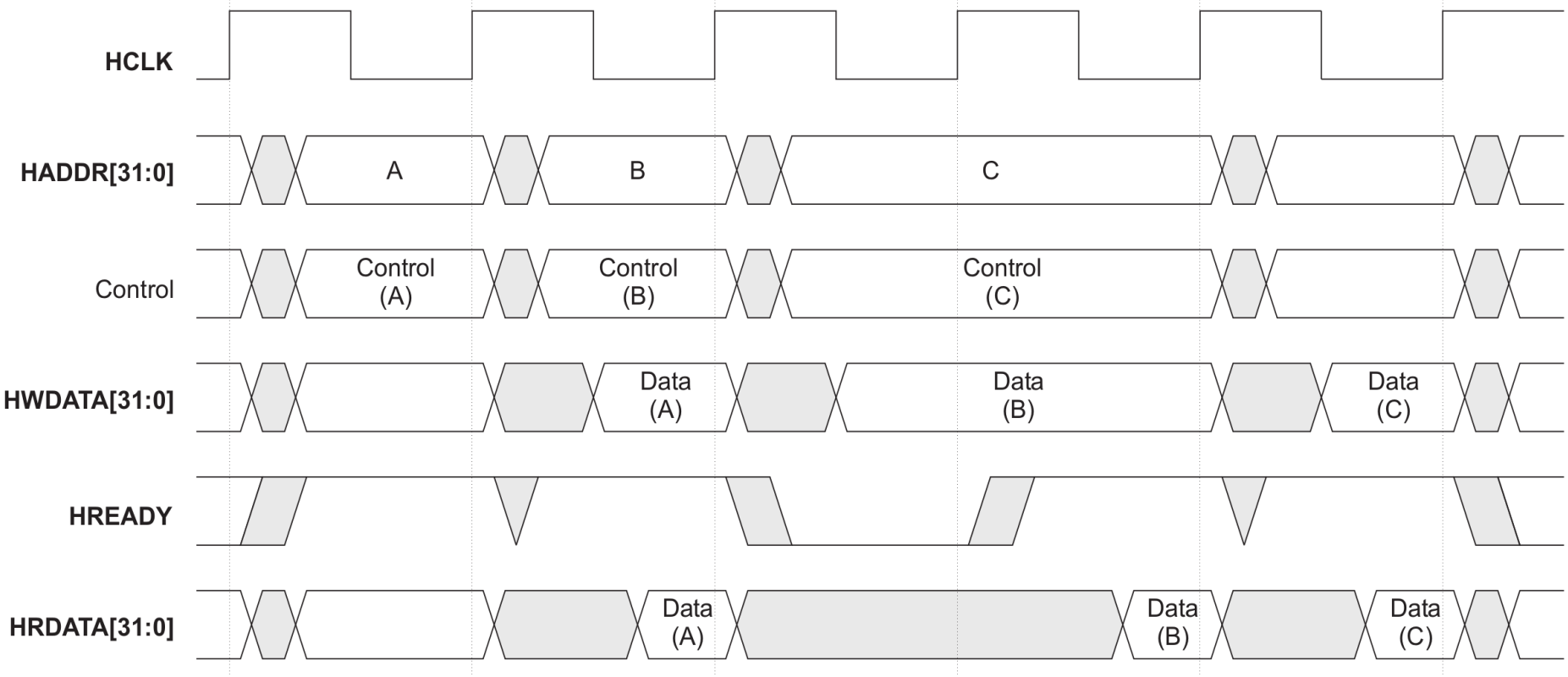


✓Si el Slave requiere mas tiempo para responder, tiene la señal **HREADY**

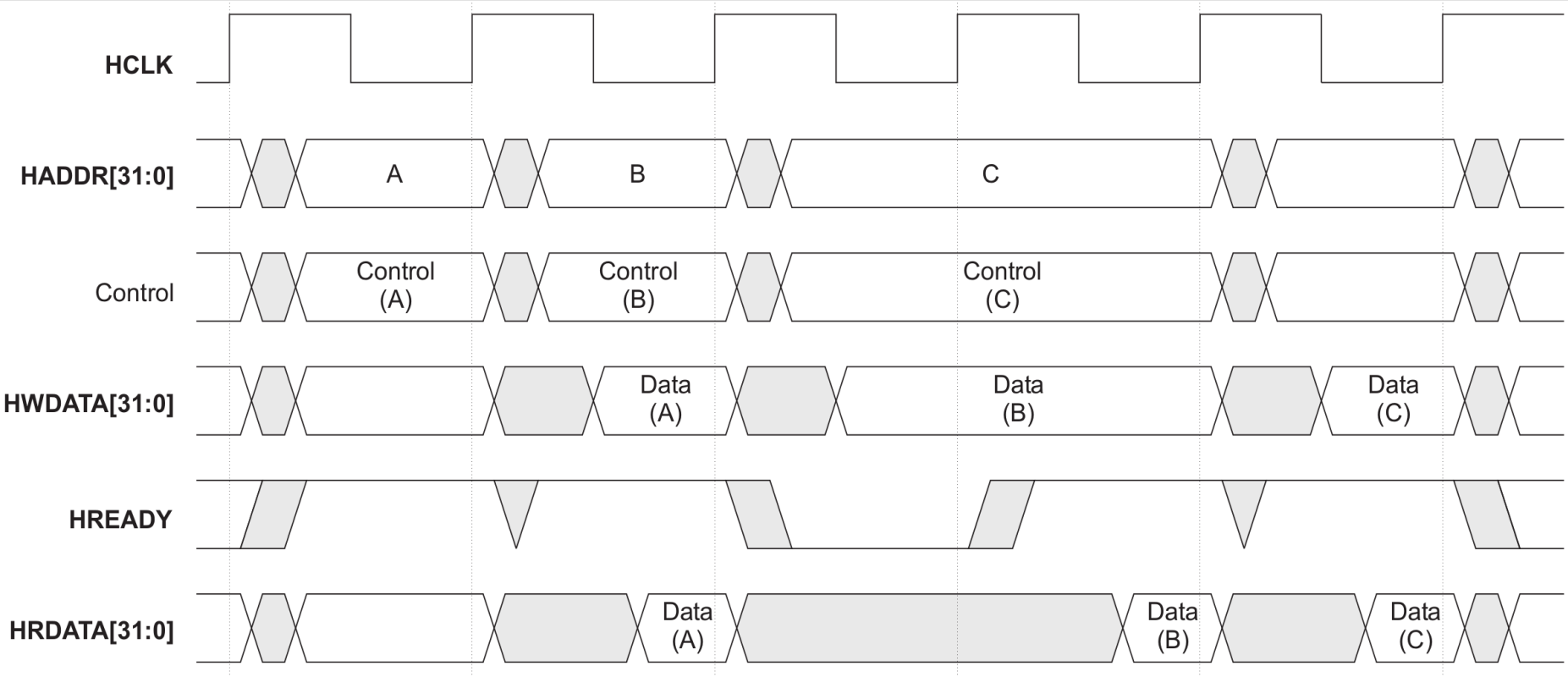
✓Si la transferencia es de escritura el Master debe sostener los datos válidos en el bus durante los ciclos extendidos.

✓Si es de Lectura, el Slave no proveerá datos válidos hasta que se active **HREADY**.

Transferencia AHB Múltiple

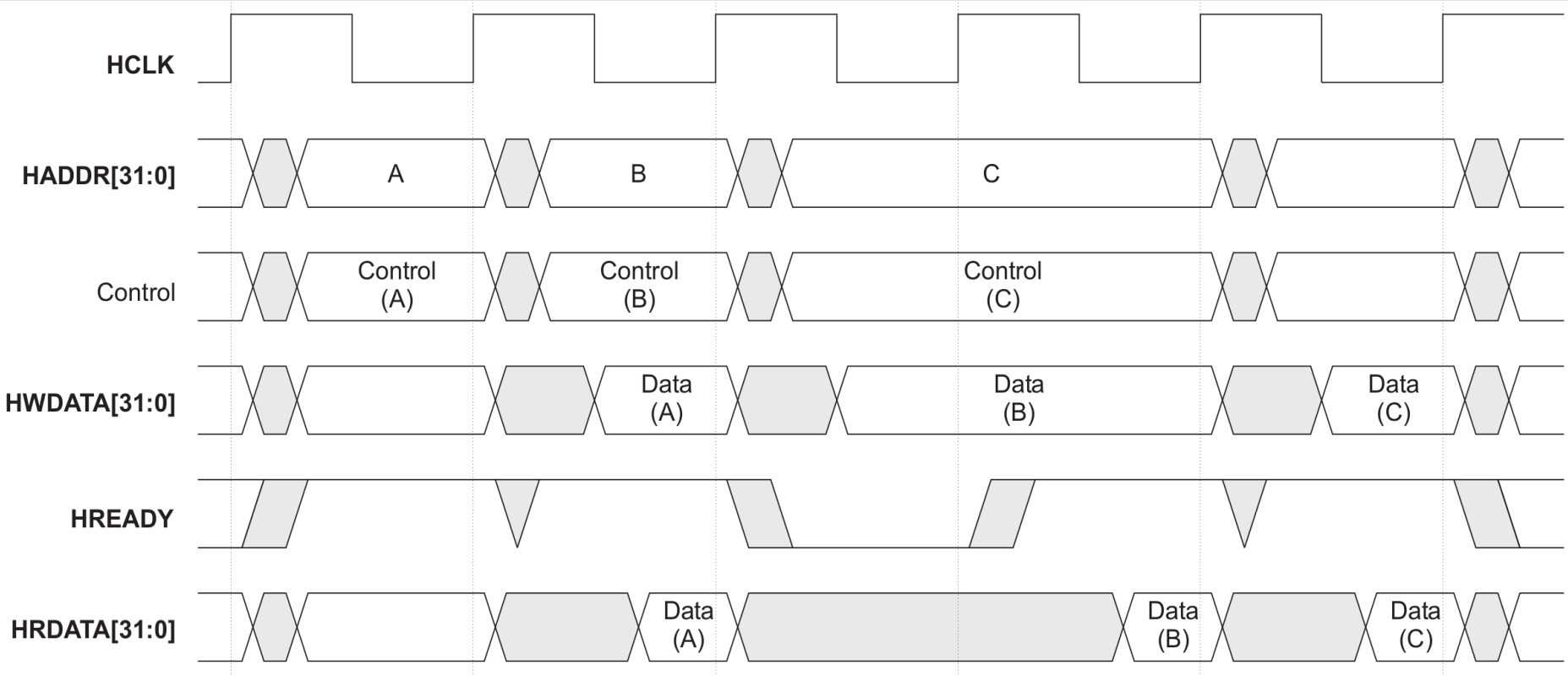


Transferencia AHB Múltiple



✓Las transferencias a direcciones A y C son 0 ***wait states*** .

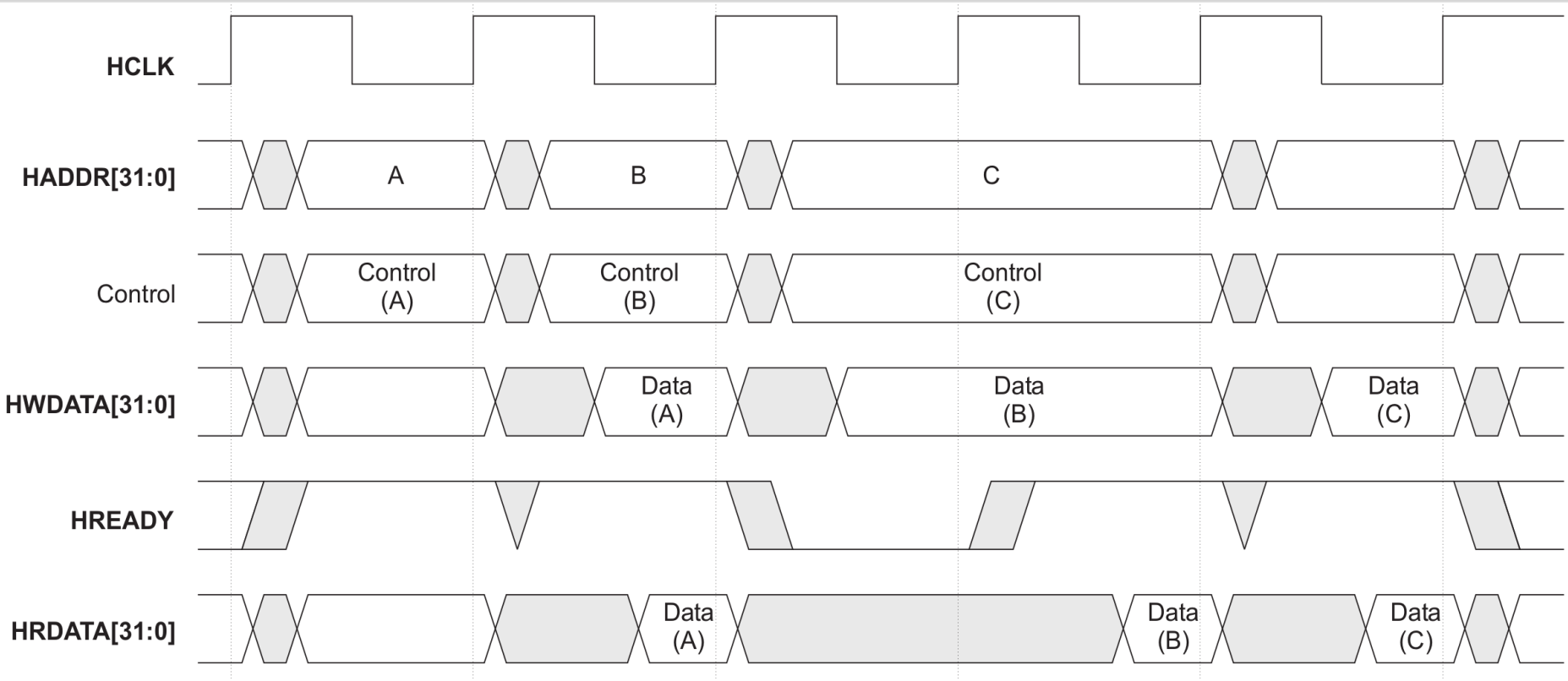
Transferencia AHB Múltiple



✓Las transferencias a direcciones A y C son 0 ***wait states*** .

✓La transferencia a la dirección B inserta un ***wait states*** .

Transferencia AHB Múltiple



- ✓Las transferencias a direcciones A y C son 0 ***wait states*** .
- ✓La transferencia a la dirección B inserta un ***wait states*** .
- ✓Extender la fase de datos de la transferencia a la dirección B, genera demora en la fase de Address de la transferencia a la dirección C.

Operación Burst

Operación Burst

- Ráfagas de 4, 8 o 16 **`beats`**, y user-defined de hasta 1024 bytes.

Operación Burst

- Ráfagas de 4, 8 o 16 **beats**, y user-defined de hasta 1024 bytes.
- En modo Incremental se accede a un área contigua de direccionamiento en donde cada dirección se obtiene incrementando la anterior.

Operación Burst

- Ráfagas de 4, 8 o 16 **`beats`**, y user-defined de hasta 1024 bytes.
- En modo Incremental se accede a un área contigua de direccionamiento en donde cada dirección se obtiene incrementando la anterior.
- En modo Wrapping si los datos no comienzan en direcciones alineadas, la transferencia es por la cantidad de ***beats***.

Operación Burst

- Ráfagas de 4, 8 o 16 **`beats`**, y user-defined de hasta 1024 bytes.
- En modo Incremental se accede a un área contigua de direccionamiento en donde cada dirección se obtiene incrementando la anterior.
- En modo Wrapping si los datos no comienzan en direcciones alineadas, la transferencia es por la cantidad de ***beats***.
- Por lo tanto el wrap se produce al alcanzarse la dirección tope en cantidad de bytes del burst.

Operación Burst

- Ráfagas de 4, 8 o 16 **`beats`**, y user-defined de hasta 1024 bytes.
- En modo Incremental se accede a un área contigua de direccionamiento en donde cada dirección se obtiene incrementando la anterior.
- En modo Wrapping si los datos no comienzan en direcciones alineadas, la transferencia es por la cantidad de ***beats***.
- Por lo tanto el wrap se produce al alcanzarse la dirección tope en cantidad de bytes del burst.
- En una transferencia burst wrapping de 4 ***beats*** de acceso word, constará de 16 bytes en total. El wrapping se produce en la dirección múltiplo de 16 bytes.

Operación Burst

- Ráfagas de 4, 8 o 16 **`beats`**, y user-defined de hasta 1024 bytes.
- En modo Incremental se accede a un área contigua de direccionamiento en donde cada dirección se obtiene incrementando la anterior.
- En modo Wrapping si los datos no comienzan en direcciones alineadas, la transferencia es por la cantidad de ***beats***.
- Por lo tanto el wrap se produce al alcanzarse la dirección tope en cantidad de bytes del burst.
- En una transferencia burst wrapping de 4 ***beats*** de acceso word, constará de 16 bytes en total. El wrapping se produce en la dirección múltiplo de 16 bytes.
- Ejemplo: Si ésta transferencia comienza en la dirección 0x38, entonces son cuatro transferencias en las siguientes direcciones: 0x38, 0x3C, 0x30, 0x34.

Codificación de tipos de Burst

HBURST[2:0]	Tipo	Descripción
000	SINGLE	Single transfer
001	INCR	Incrementing burst of unspecified length
010	WRAP4	4-beat wrapping burst
011	INCR4	4-beat incrementing burst
100	WRAP8	8-beat wrapping burst
101	INCR8	8-beat incrementing burst
110	WRAP16	16-beat wrapping burst
111	INCR16	16-beat incrementing burst

El límite máximo de una transferencia Burst es 1 Kbyte. Si en un burst incremental se trabaja con el tipo no especificado debe cuidarse de no superar este límite.
Las transferencias dentro de un burst deben estar alineadas acorde al tamaño del dato básico que se transmite.

Interrupción de un Burst

Interrupción de un Burst

✓Bajo ciertas condiciones, una transferencia burst puede ser suspendida.

Interrupción de un Burst

- ✓Bajo ciertas condiciones, una transferencia burst puede ser suspendida.
- ✓El Slave involucrado debe tener notificación inmediata.

Interrupción de un Burst

✓Bajo ciertas condiciones, una transferencia burst puede ser suspendida.
✓El Slave involucrado debe tener notificación inmediata.
✓La información que un Slave debe monitorear es el estado de las Líneas **`HTRANS[1:0]`**. Si desde el inicio de la transacción el estado es **`BUSY`** o **`SEQUENTIAL`** la transacción burst prosigue normalmente.

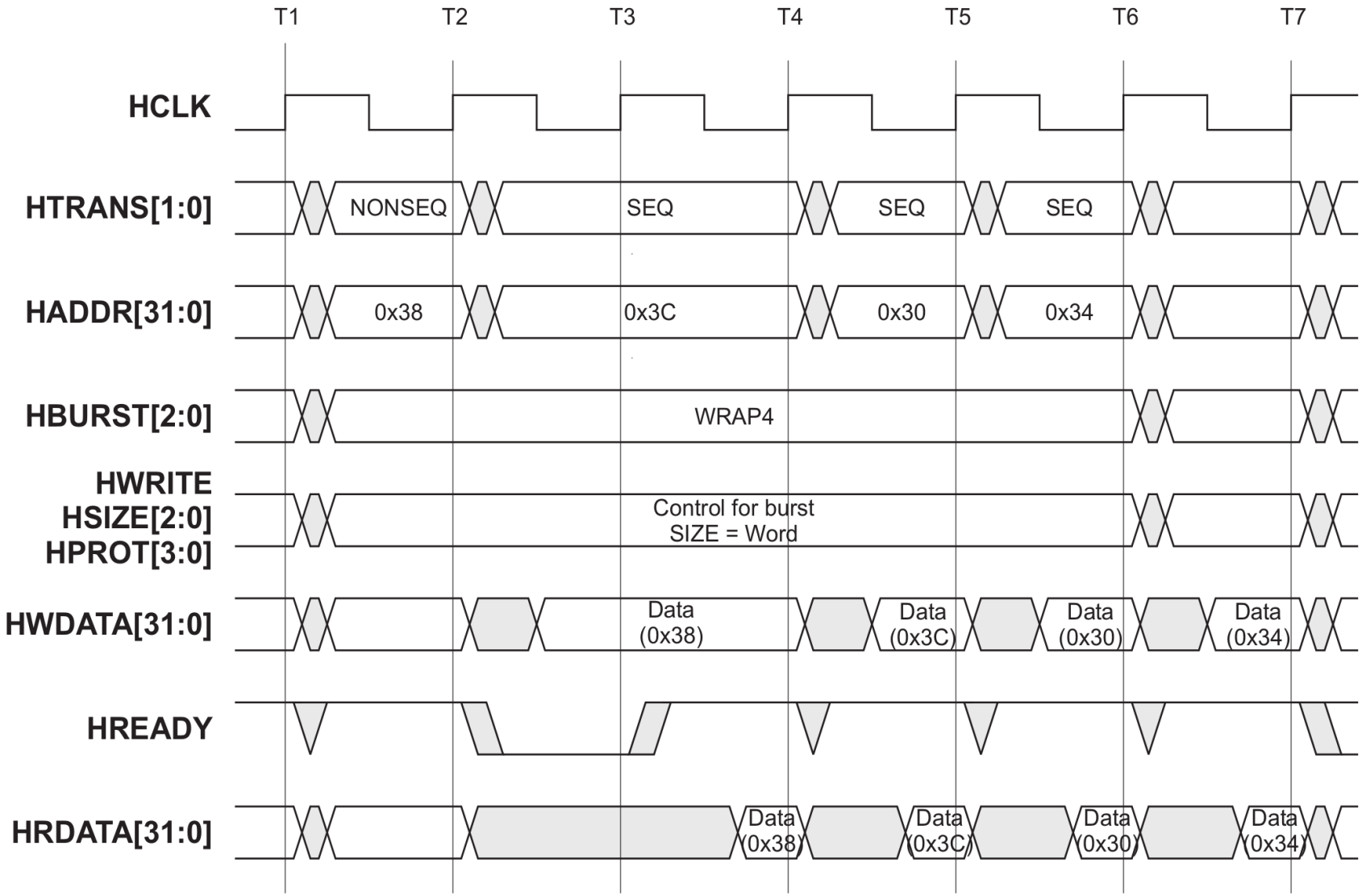
Interrupción de un Burst

✓Bajo ciertas condiciones, una transferencia burst puede ser suspendida.
✓El Slave involucrado debe tener notificación inmediata.
✓La información que un Slave debe monitorear es el estado de las Líneas **`HTRANS[1:0]`**. Si desde el inicio de la transacción el estado es **`BUSY`** o **`SEQUENTIAL`** la transacción burst prosigue normalmente.
✓Si detecta el estado **`NONSEQUENTIAL`** o **`IDLE`**, significa que se ha iniciado una nueva transferencia burst, y que la anterior ha sido terminada.

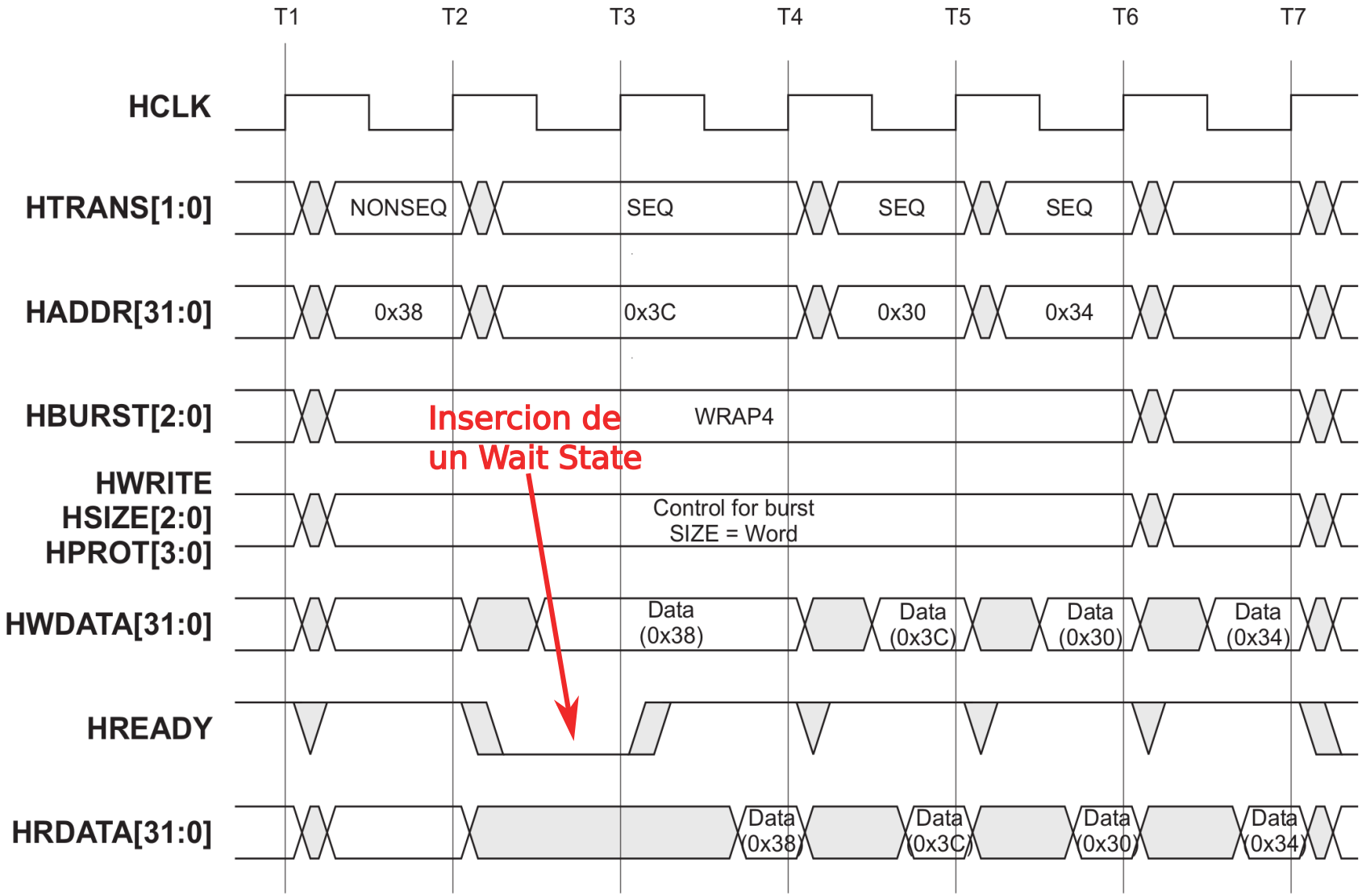
Interrupción de un Burst

✓Bajo ciertas condiciones, una transferencia burst puede ser suspendida.
✓El Slave involucrado debe tener notificación inmediata.
✓La información que un Slave debe monitorear es el estado de las Líneas **`HTRANS[1:0]`**. Si desde el inicio de la transacción el estado es **`BUSY`** o **`SEQUENTIAL`** la transacción burst prosigue normalmente.
✓Si detecta el estado **`NONSEQUENTIAL`** o **`IDLE`**, significa que se ha iniciado una nueva transferencia burst, y que la anterior ha sido terminada.
✓El Master que no logró completar la transferencia burst por haber perdido el control del Bus, deberá completarla cuando logre recuperar el bus. Para ello normalmente se inicia una transferencia de tamaño Undefined que curse los restantes ***beats*** que quedaron pendientes en la transferencia suspendida.

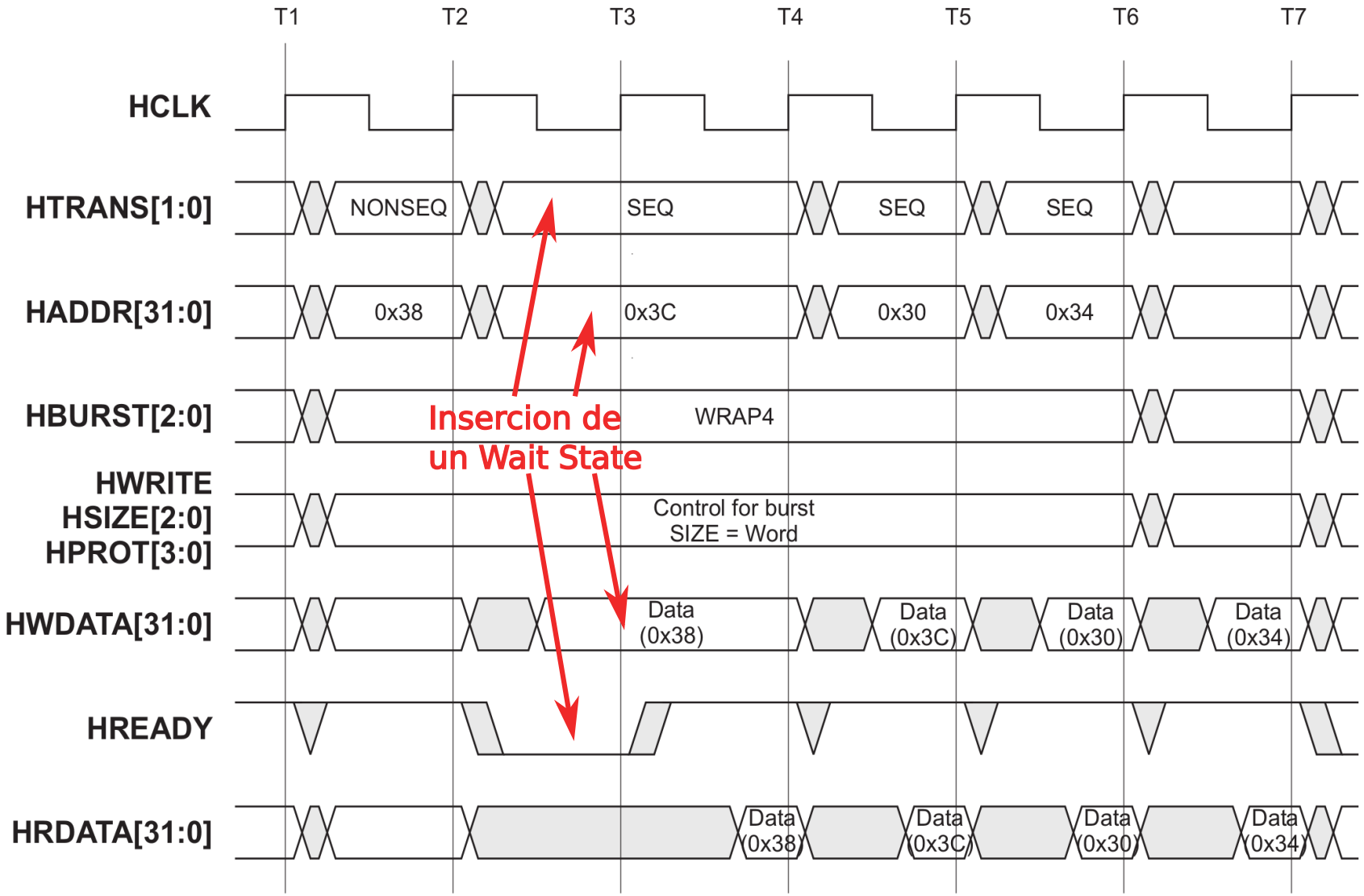
Burst Wrapping de 4 beats



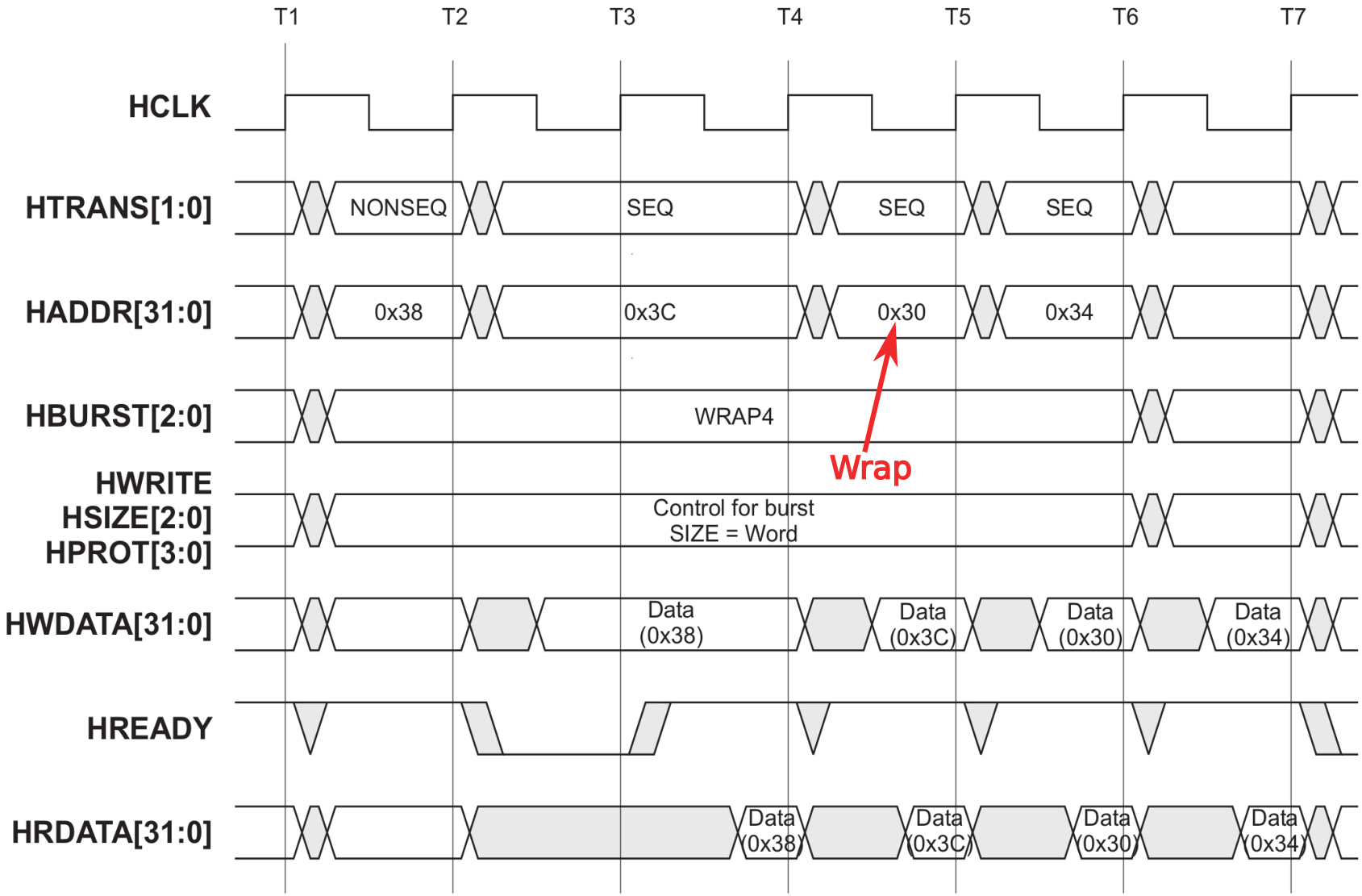
Burst Wrapping de 4 beats



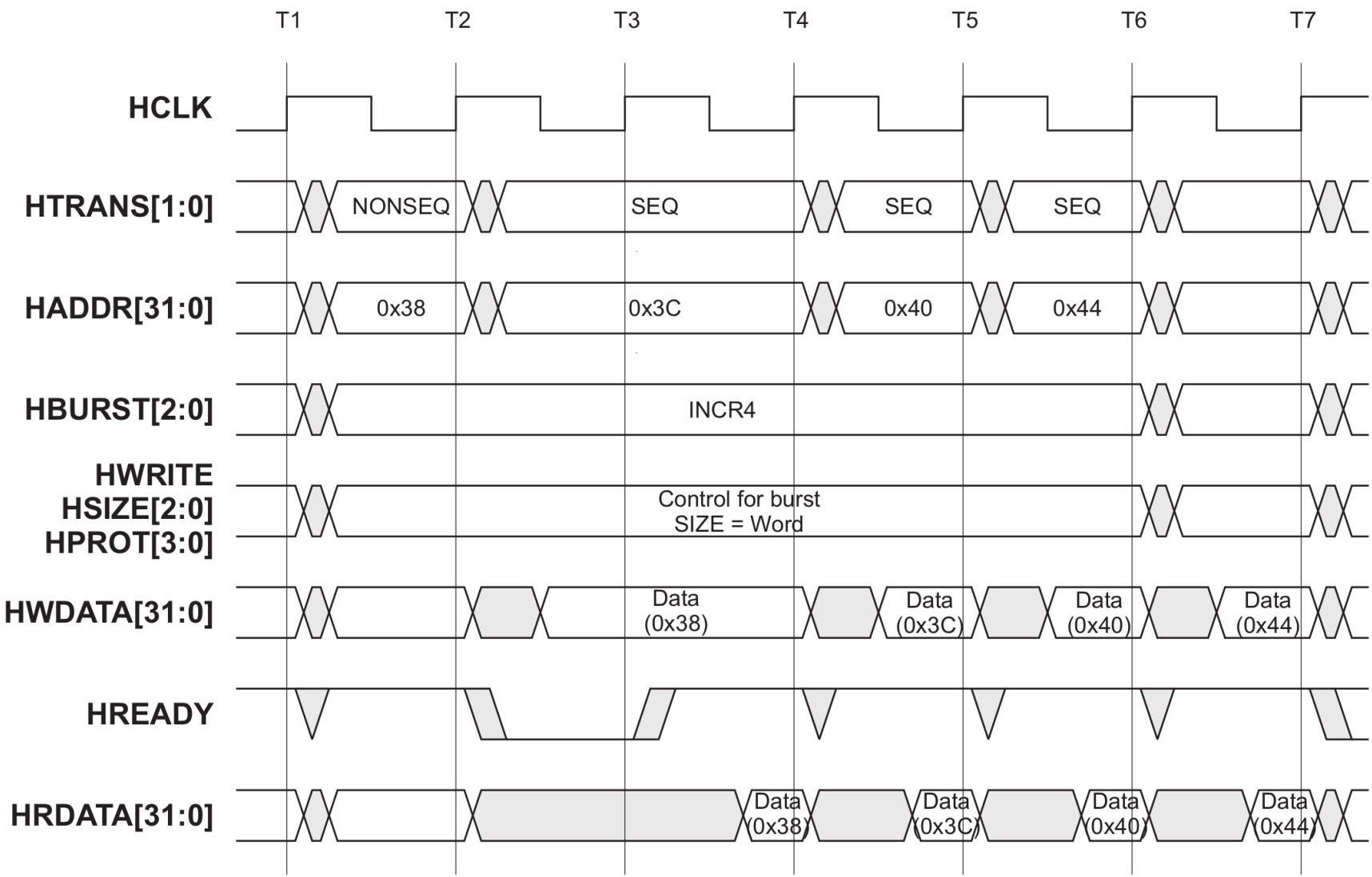
Burst Wrapping de 4 beats



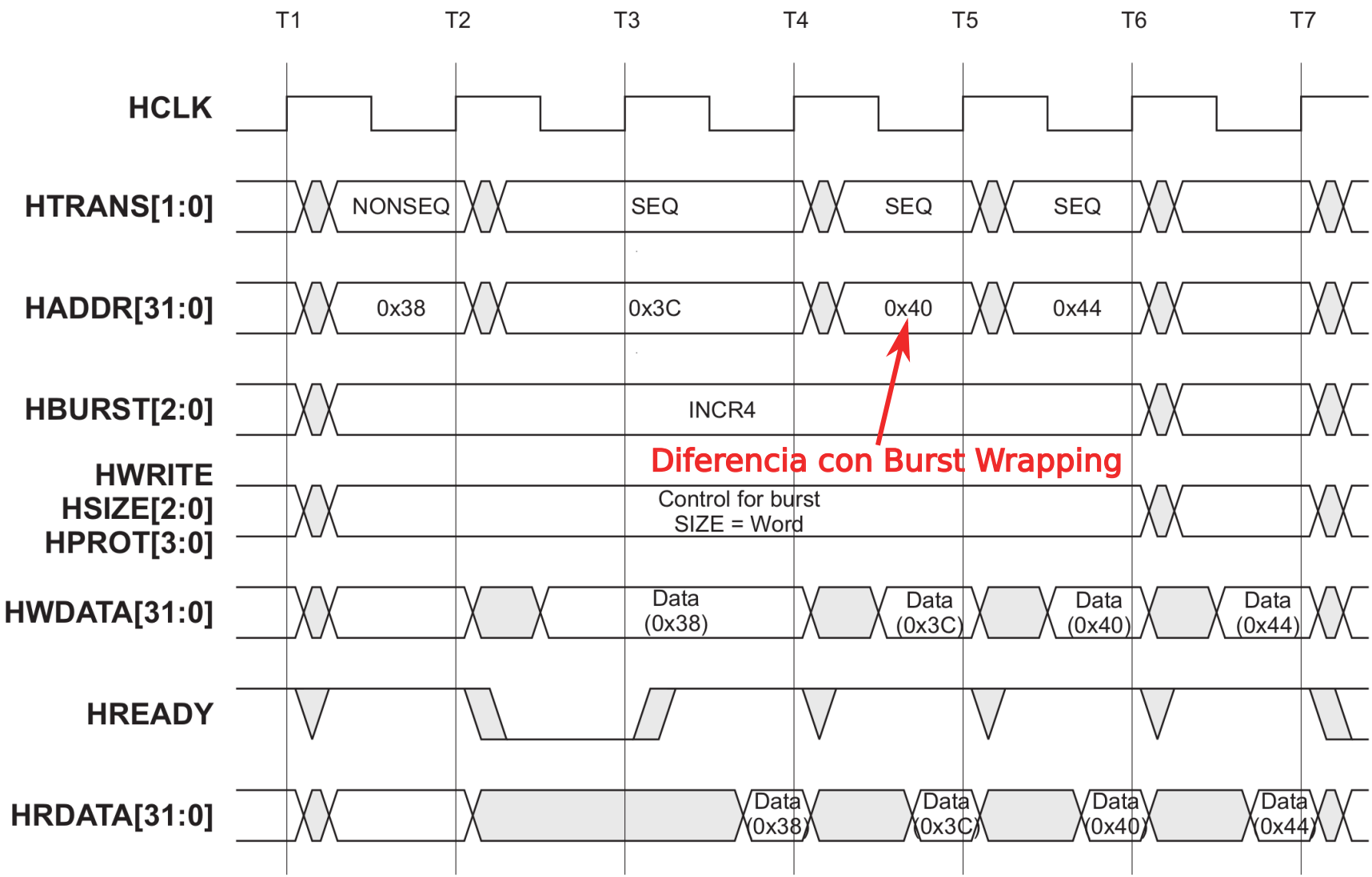
Burst Wrapping de 4 beats



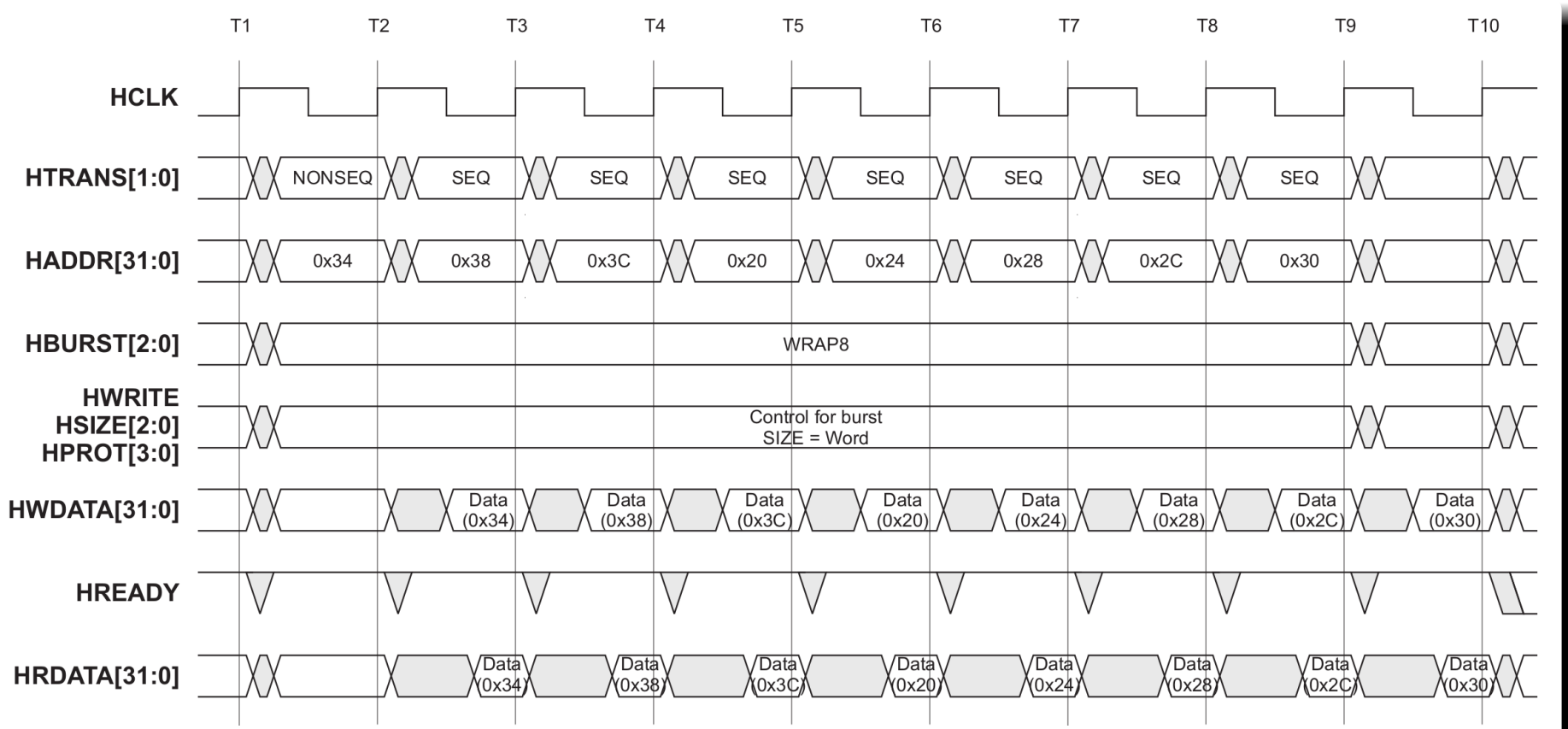
Burst Incremental de 4 beats



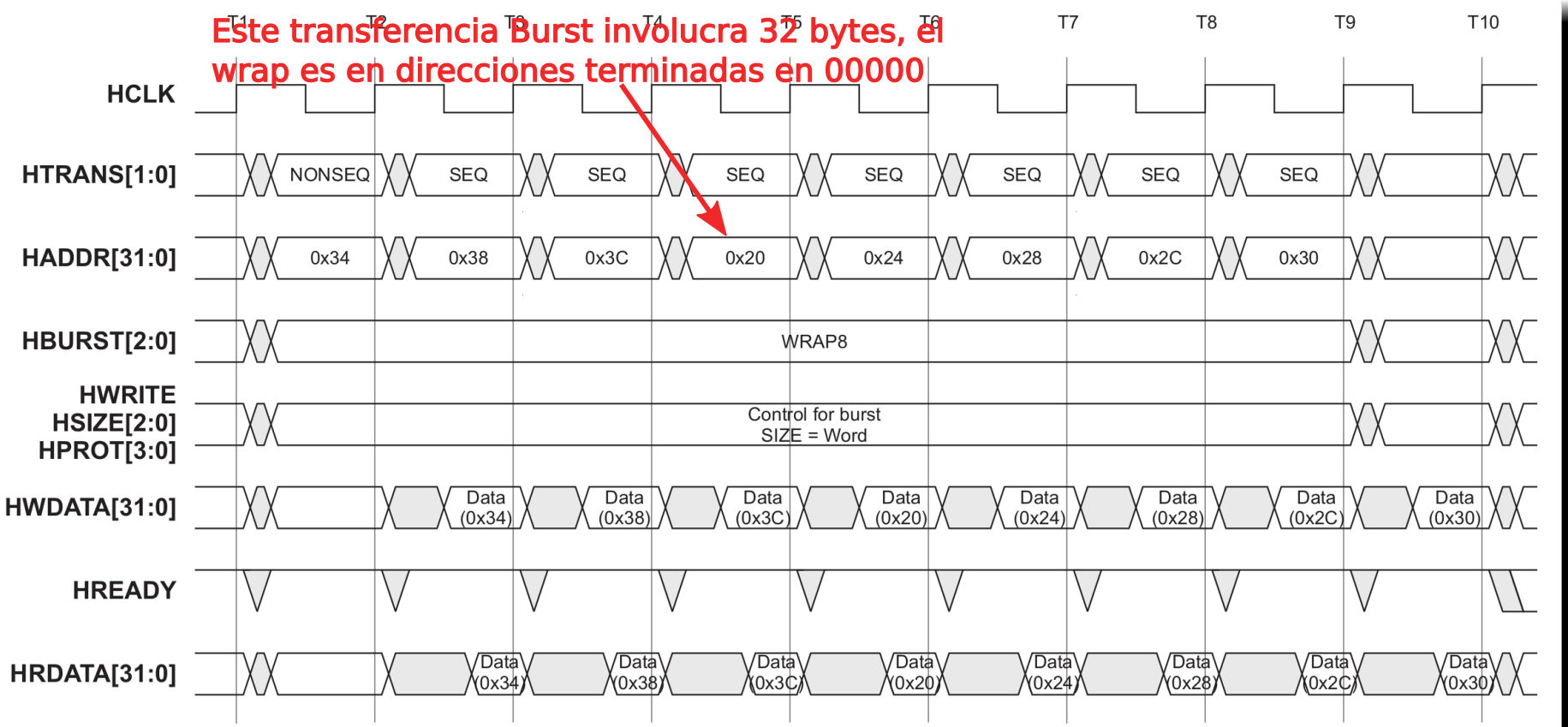
Burst Incremental de 4 beats



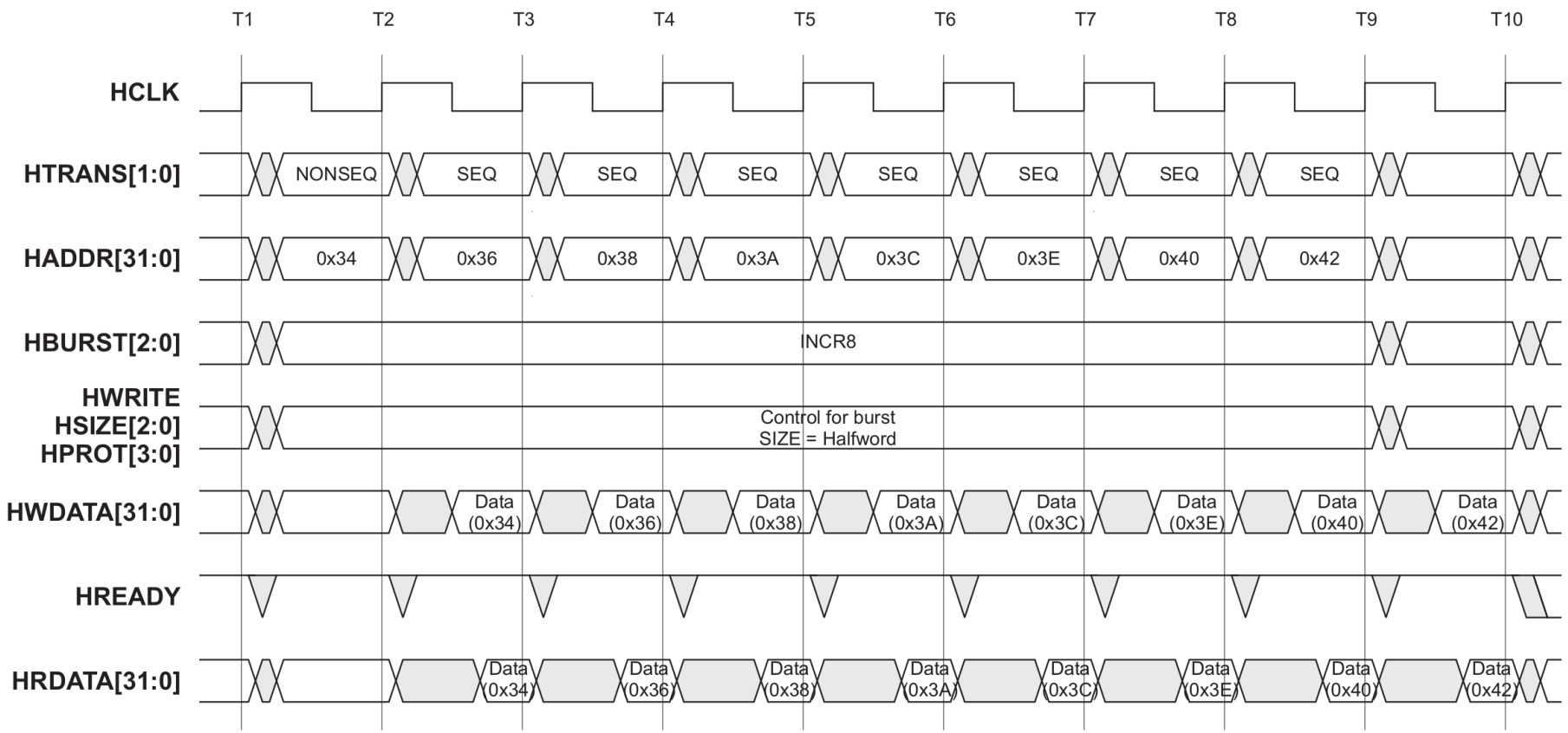
Burst Wrapping de 8 beats



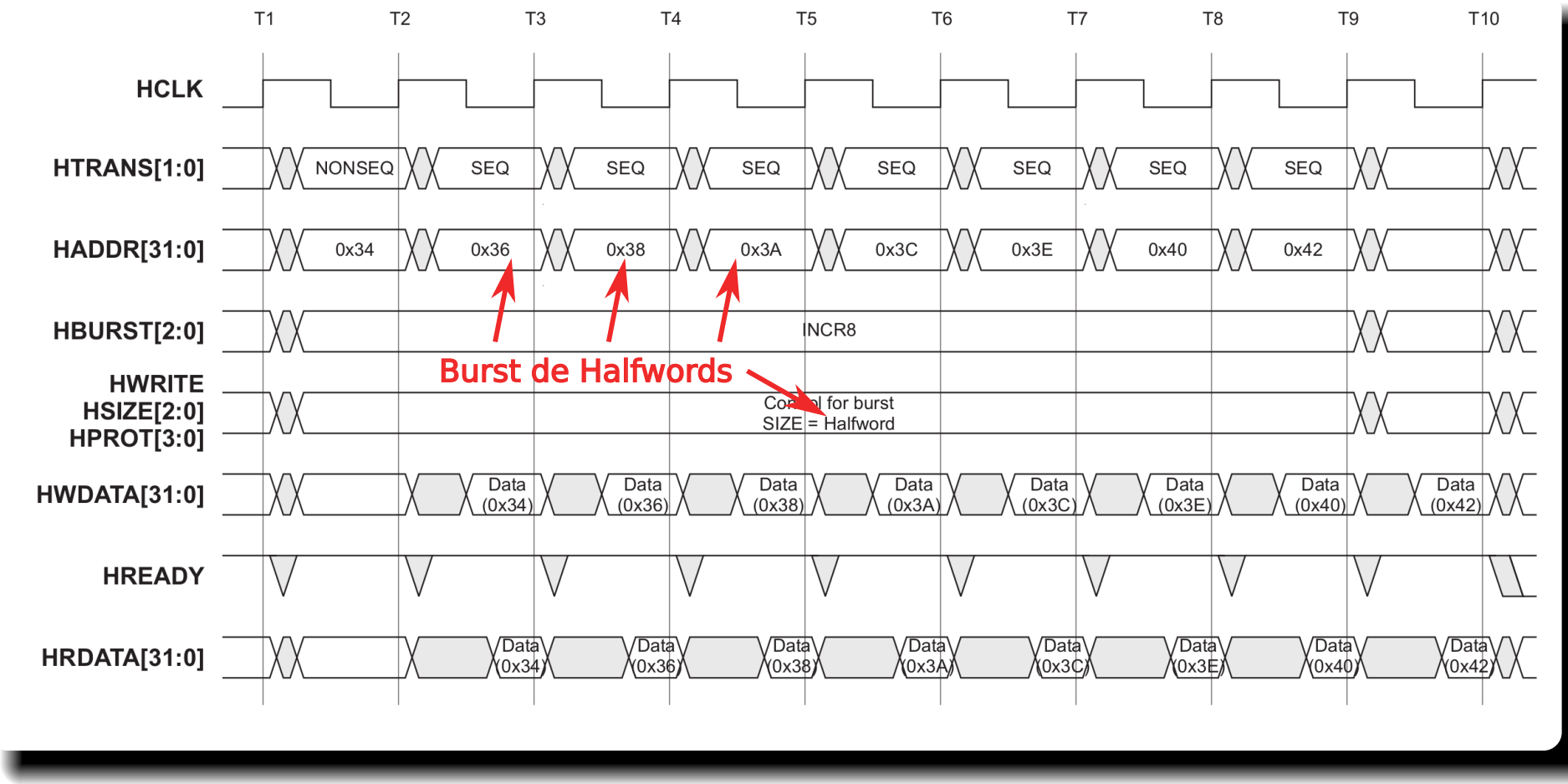
Burst Wrapping de 8 beats



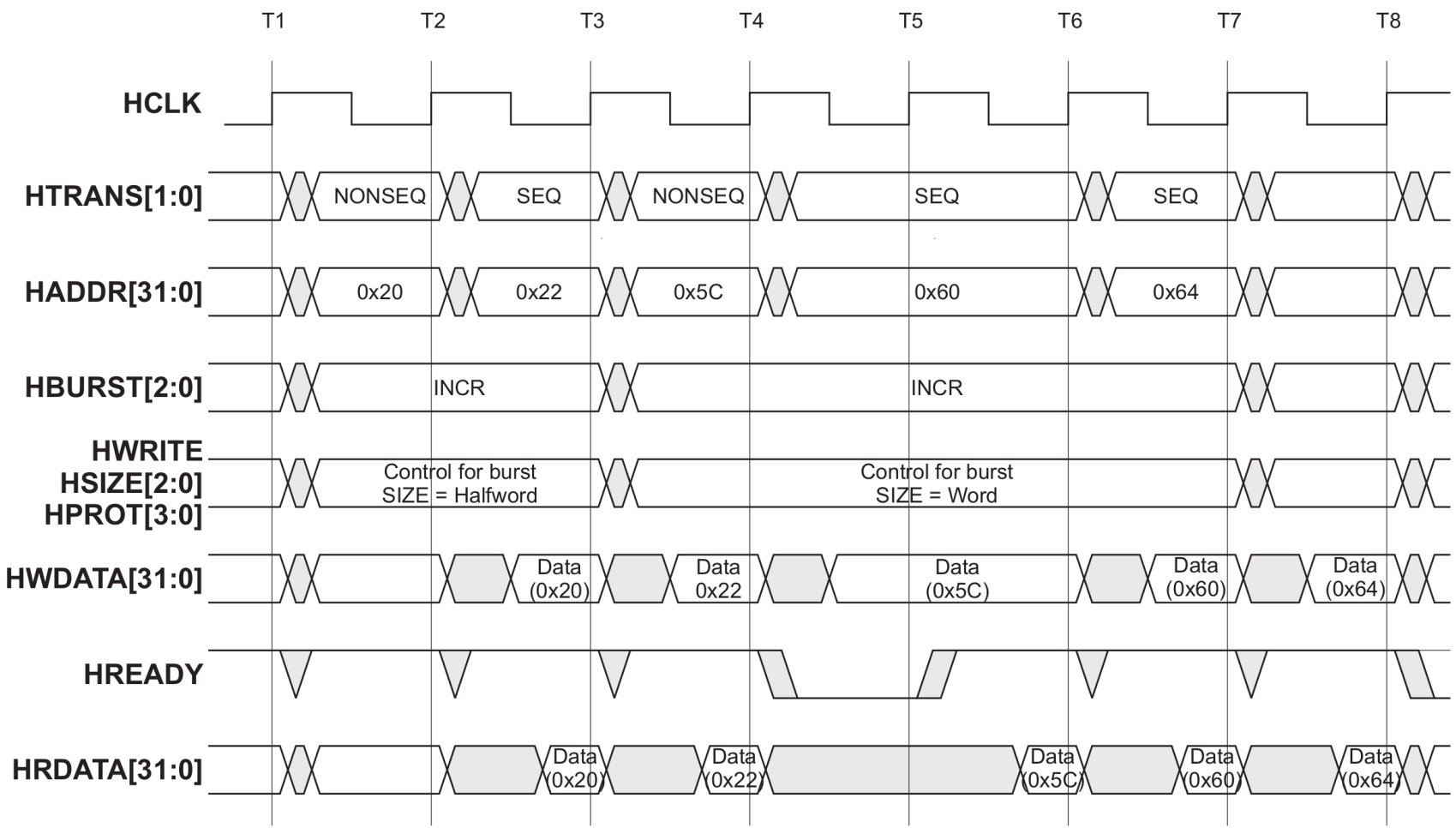
Burst Incremental de 8 beats



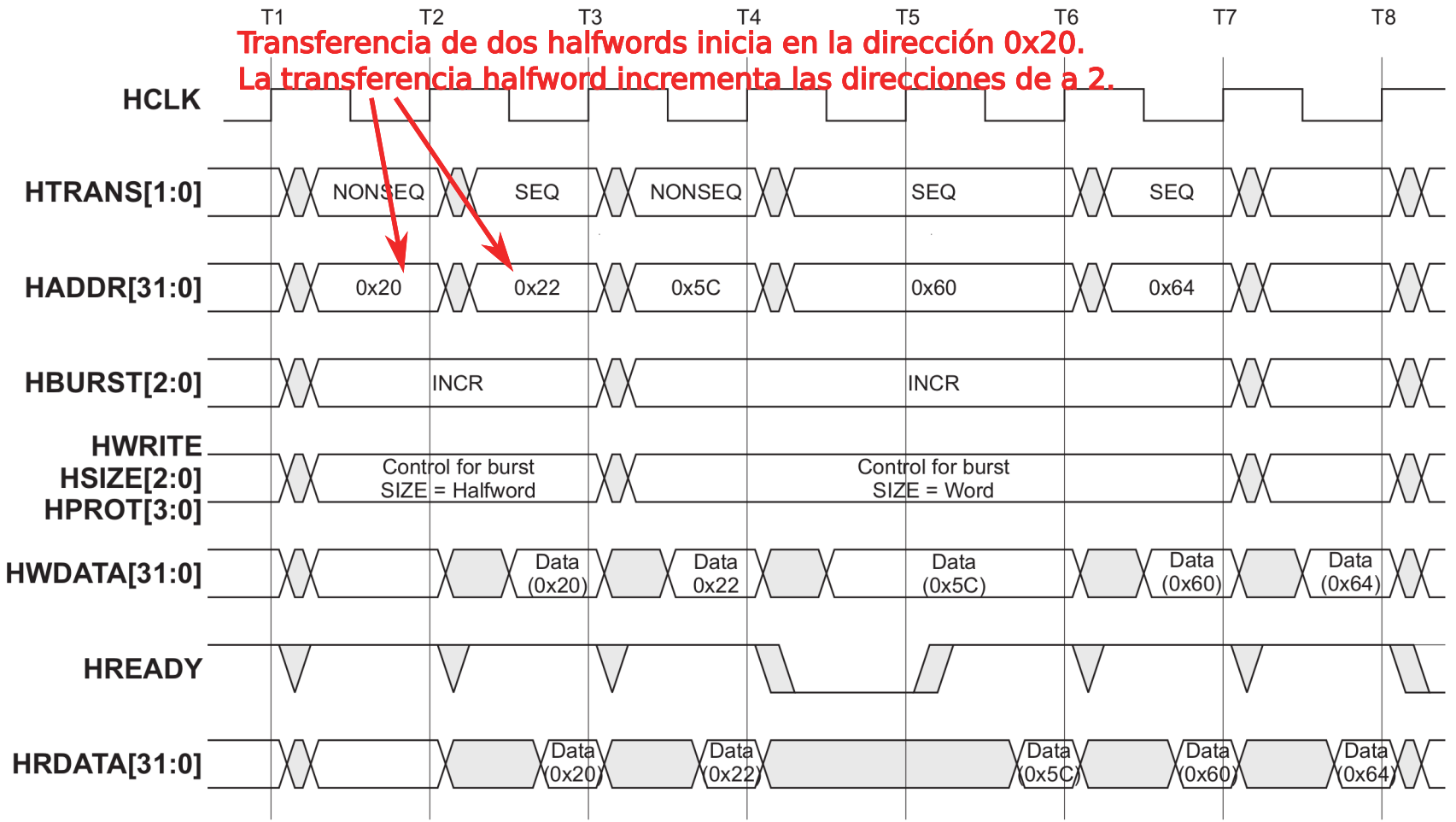
Burst Incremental de 8 beats



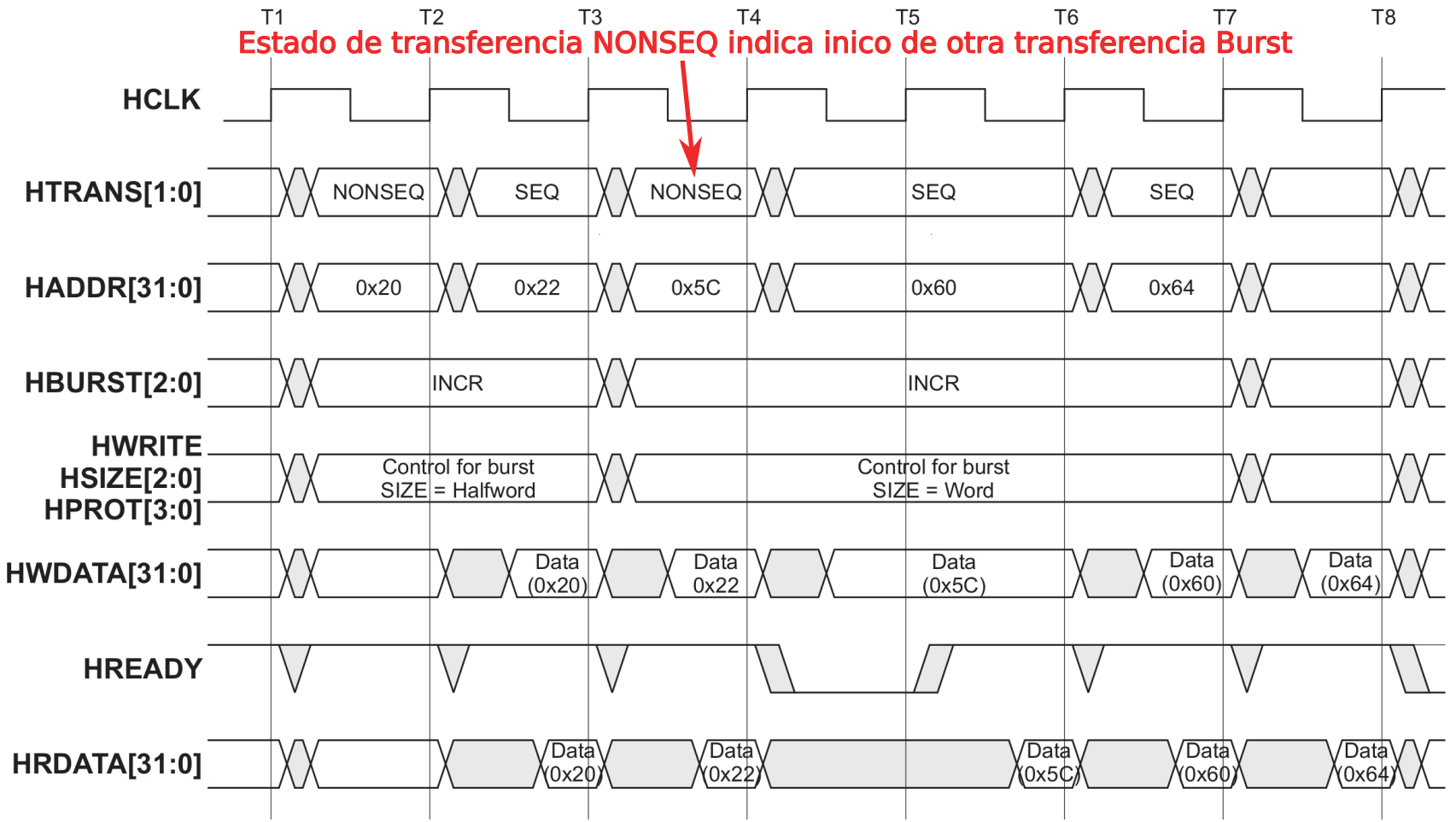
Burst de longitud desconocida



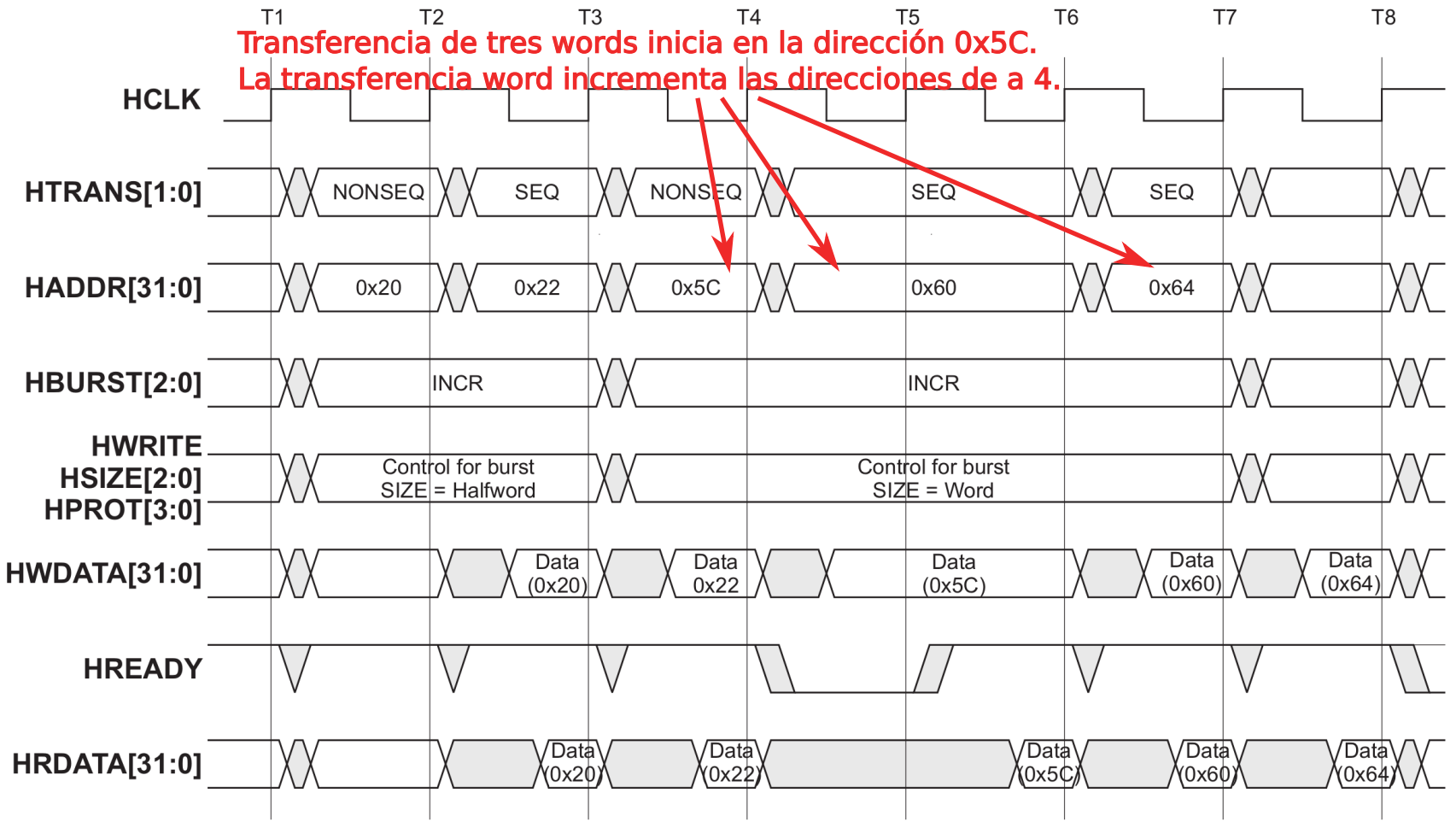
Burst de longitud desconocida



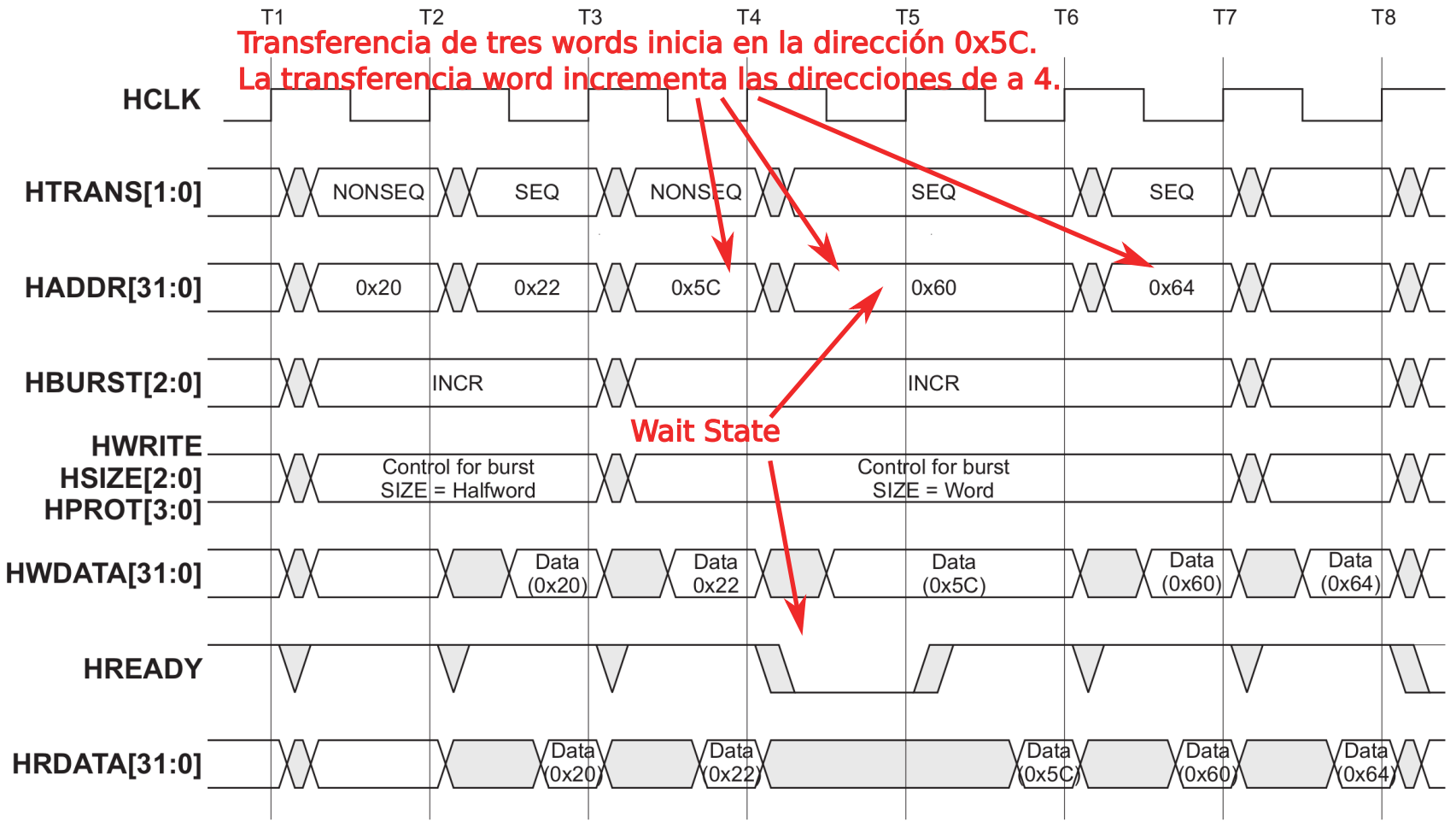
Burst de longitud desconocida



Burst de longitud desconocida



Burst de longitud desconocida



Señales de control

Señales de control

- **HWRITE**: Si es Alta indica una transferencia de escritura. El Master debe colocar en **HWDATA[31:0]** la word a enviar al Slave seleccionado. Si es Baja la transferencia es de Lectura y el Slave deberá generar los datos en **HRDATA[31:0]**

Señales de control

- **`HWRITE`**: Si es Alta indica una transferencia de escritura. El Master debe colocar en **`HWDATA[31:0]`** la word a enviar al Slave seleccionado. Si es Baja la transferencia es de Lectura y el Slave deberá generar los datos en **`HRDATA[31:0]`**
- **`HSIZE[2:0]`**: Tamaño de la transferencia

Señales de control

- **HWRITE**: Si es Alta indica una transferencia de escritura. El Master debe colocar en **HWDATA[31:0]** la word a enviar al Slave seleccionado. Si es Baja la transferencia es de Lectura y el Slave deberá generar los datos en **HRDATA[31:0]**
- **HSIZE[2:0]**: Tamaño de la transferencia

HSIZE[2]	HSIZE[1]	HSIZE[0]	Tamaño	Descripción
0	0	0	8 bits	Byte
0	0	1	16 bits	Half Word
0	1	0	32 bits	Word
0	1	1	64 bits	-
1	0	0	128 bits	4-word line
1	0	1	256 bits	8-word line
1	1	0	512 bits	-
1	1	1	1024 bits	-

Señales de control

- **`HSIZE[2:0]`** junto con **`HBURST[2:0]`** determinan la dirección límite para transferencias Burst Wrapping.

Señales de control

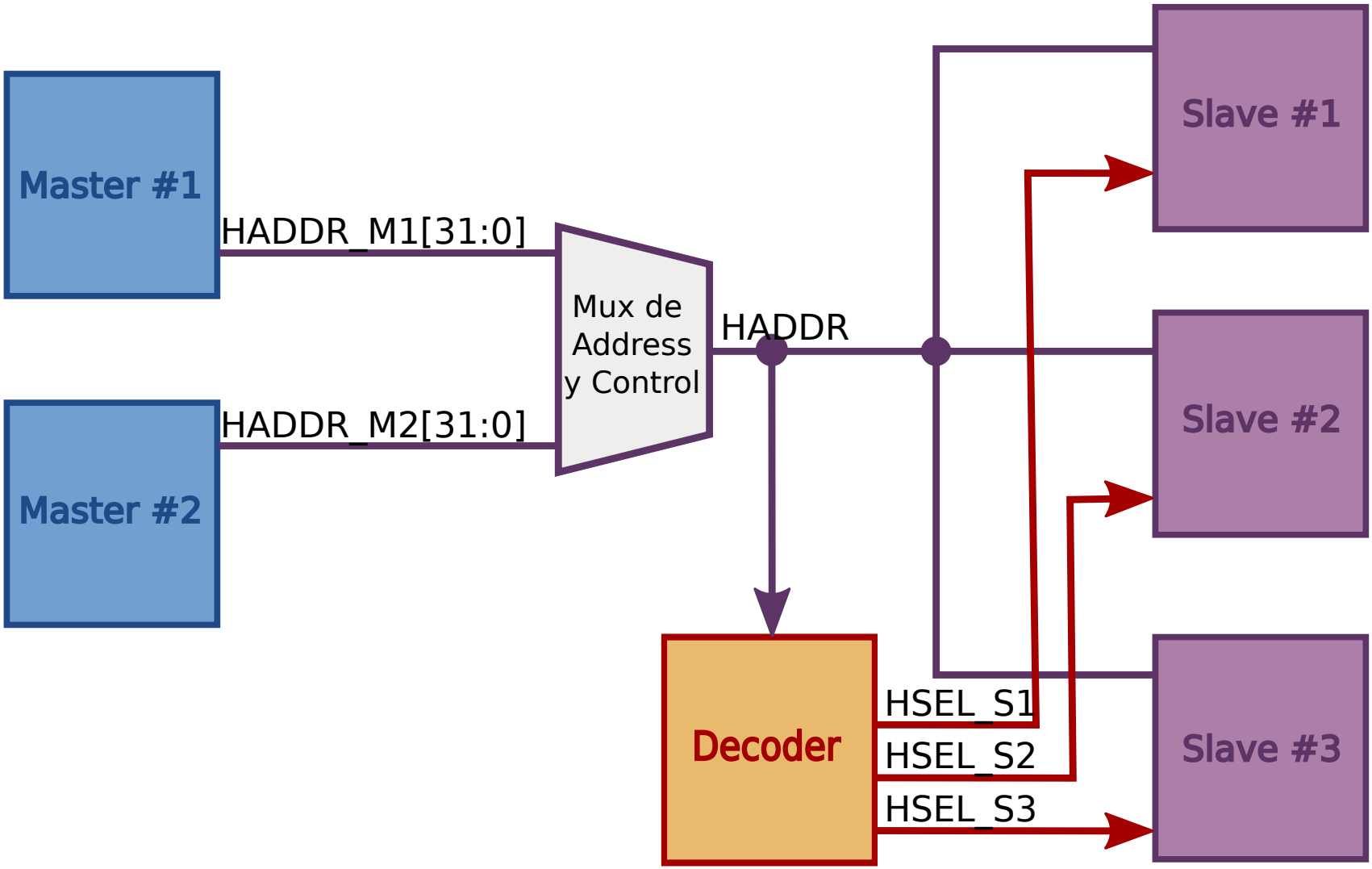
- **`HSIZE[2:0]`** junto con **`HBURST[2:0]`** determinan la dirección límite para transferencias Burst Wrapping.
- **`HPROT[3:0]`**: Provee información acerca del destino del acceso al bus. Para procesadores con MMU provee información importante para implementar cierto nivel de protección. No se recomienda el uso en Slaves.

Señales de control

- **HSIZE[2:0]** junto con **HBURST[2:0]** determinan la dirección límite para transferencias Burst Wrapping.
- **HPROT[3:0]**: Provee información acerca del destino del acceso al bus. Para procesadores con MMU provee información importante para implementar cierto nivel de protección. No se recomienda el uso en Slaves.

HPROT[3] Cacheable	HPROT[2] Buffereable	HPROT[1] Priviedged	HPROT[0] Data/Opcode	Descripción
-	-	-	0	Opcode Fetch
-	-	-	1	Data Access
-	-	0	-	User Access
-	-	1	-	Priviledged Access
-	0	-	-	Not Buffereable
-	1	-	-	Buffereable
0	-	-	-	Not Cacheable
1	-	-	-	Cacheable

Decodificación de Direcciones



Decodificación de Direcciones

Decodificación de Direcciones

- El espacio mínimo que debe ocupar un Slave es 1 KByte.

Decodificación de Direcciones

- El espacio mínimo que debe ocupar un Slave es 1 KByte.
- Los Bus Masters deben diseñarse de modo de no realizar transferencias incrementales de mas de 1 KByte y así nunca traspasar el límite de direccionamiento de un Slave.

Decodificación de Direcciones

- El espacio mínimo que debe ocupar un Slave es 1 KByte.
- Los Bus Masters deben diseñarse de modo de no realizar transferencias incrementales de mas de 1 KByte y así nunca traspasar el límite de direccionamiento de un Slave.
- Si un Slave no llena el Kbyte de direccionamiento se puede implementar un Slave default que responda adecuadamente para aquellas direcciones no implementadas en el original, a saber:

Decodificación de Direcciones

- El espacio mínimo que debe ocupar un Slave es 1 KByte.
- Los Bus Masters deben diseñarse de modo de no realizar transferencias incrementales de mas de 1 KByte y así nunca traspasar el límite de direccionamiento de un Slave.
- Si un Slave no llena el Kbyte de direccionamiento se puede implementar un Slave default que responda adecuadamente para aquellas direcciones no implementadas en el original, a saber:
- Si se intentan transferencias **`NONSEQUENTIAL`** o **`SEQUENTIAL`** a una dirección inexistente el Slave default debe proveer respuesta **`ERROR`**.

Decodificación de Direcciones

- El espacio mínimo que debe ocupar un Slave es 1 KByte.
- Los Bus Masters deben diseñarse de modo de no realizar transferencias incrementales de mas de 1 KByte y así nunca traspasar el límite de direccionamiento de un Slave.
- Si un Slave no llena el Kbyte de direccionamiento se puede implementar un Slave default que responda adecuadamente para aquellas direcciones no implementadas en el original, a saber:
- Si se intentan transferencias **`NONSEQUENTIAL`** o **`SEQUENTIAL`** a una dirección inexistente el Slave default debe proveer respuesta **`ERROR`**.
- Por su parte, transferencias **`IDLE`** o **`BUSY`** a direcciones inexistentes deben generar una respuesta **`OKAY`** con 0 ***wait states*** .

Decodificación de Direcciones

- El espacio mínimo que debe ocupar un Slave es 1 KByte.
- Los Bus Masters deben diseñarse de modo de no realizar transferencias incrementales de mas de 1 KByte y así nunca traspasar el límite de direccionamiento de un Slave.
- Si un Slave no llena el Kbyte de direccionamiento se puede implementar un Slave default que responda adecuadamente para aquellas direcciones no implementadas en el original, a saber:
- Si se intentan transferencias **`NONSEQUENTIAL`** o **`SEQUENTIAL`** a una dirección inexistente el Slave default debe proveer respuesta **`ERROR`**.
- Por su parte, transferencias **`IDLE`** o **`BUSY`** a direcciones inexistentes deben generar una respuesta **`OKAY`** con 0 ***wait states*** .
- El Slave Default debe implementarse como parte del Decodificador Central de Direcciones

Respuesta desde el Slave

Respuesta desde el Slave

- Una vez que el Master inicia una transferencia, el control de la misma corresponde al Slave.

Respuesta desde el Slave

- Una vez que el Master inicia una transferencia, el control de la misma corresponde al Slave.
- No está previsto que el Master intervenga para cancelar una transferencia.

Respuesta desde el Slave

- Una vez que el Master inicia una transferencia, el control de la misma corresponde al Slave.
- No está previsto que el Master intervenga para cancelar una transferencia.
- El Slave tiene la señal **`HREADY`** para extender la transferencia el tiempo que necesite, en combinación con **`HRESP[1:0]`**, por donde informa el estado de la transferencia.

Respuesta desde el Slave

- Una vez que el Master inicia una transferencia, el control de la misma corresponde al Slave.
- No está previsto que el Master intervenga para cancelar una transferencia.
- El Slave tiene la señal **`HREADY`** para extender la transferencia el tiempo que necesite, en combinación con **`HRESP[1:0]`**, por donde informa el estado de la transferencia.
- El Slave puede terminar una transferencia de diversas formas.

Respuesta desde el Slave

- Una vez que el Master inicia una transferencia, el control de la misma corresponde al Slave.
- No está previsto que el Master intervenga para cancelar una transferencia.
- El Slave tiene la señal **`HREADY`** para extender la transferencia el tiempo que necesite, en combinación con **`HRESP[1:0]`**, por donde informa el estado de la transferencia.
- El Slave puede terminar una transferencia de diversas formas.
  - Completando la transferencia de manera inmediata.

Respuesta desde el Slave

- Una vez que el Master inicia una transferencia, el control de la misma corresponde al Slave.
- No está previsto que el Master intervenga para cancelar una transferencia.
- El Slave tiene la señal **`HREADY`** para extender la transferencia el tiempo que necesite, en combinación con **`HRESP[1:0]`**, por donde informa el estado de la transferencia.
- El Slave puede terminar una transferencia de diversas formas.
  - Completando la transferencia de manera inmediata.
  - Insertando uno o varios ***wait states*** , para darse tiempo de completar la transferencia.

Respuesta desde el Slave

- Una vez que el Master inicia una transferencia, el control de la misma corresponde al Slave.
- No está previsto que el Master intervenga para cancelar una transferencia.
- El Slave tiene la señal **`HREADY`** para extender la transferencia el tiempo que necesite, en combinación con **`HRESP[1:0]`**, por donde informa el estado de la transferencia.
- El Slave puede terminar una transferencia de diversas formas.
  - Completando la transferencia de manera inmediata.
  - Insertando uno o varios ***wait states*** , para darse tiempo de completar la transferencia.
  - Señalar un error indicando el fin de la transferencia.

Respuesta desde el Slave

- Una vez que el Master inicia una transferencia, el control de la misma corresponde al Slave.
- No está previsto que el Master intervenga para cancelar una transferencia.
- El Slave tiene la señal **`HREADY`** para extender la transferencia el tiempo que necesite, en combinación con **`HRESP[1:0]`**, por donde informa el estado de la transferencia.
- El Slave puede terminar una transferencia de diversas formas.
  - Completando la transferencia de manera inmediata.
  - Insertando uno o varios ***wait states*** , para darse tiempo de completar la transferencia.
  - Señalar un error indicando el fin de la transferencia.
  - Demorar el completado de la transferencia, permitiendo al Master y a si mismo liberar el bus, de modo que sea utilizado en otras transferencias.

Respuesta desde el Slave

- Cada Slave debe tener un número máximo predeterminado de ***wait states*** a insertar alcanzado el cual vuelve a ceder el bus, de modo de permitir estimaciones de demoras de acceso al bus por parte de un Master

Respuesta desde el Slave

- Cada Slave debe tener un número máximo predeterminado de ***wait states*** a insertar alcanzado el cual vuelve a ceder el bus, de modo de permitir estimaciones de demoras de acceso al bus por parte de un Master
- Se recomienda un máximo de 16 ***wait states*** . No es obligatorio.

Respuesta desde el Slave

- Cada Slave debe tener un número máximo predeterminado de ***wait states*** a insertar alcanzado el cual vuelve a ceder el bus, de modo de permitir estimaciones de demoras de acceso al bus por parte de un Master
- Se recomienda un máximo de 16 ***wait states*** . No es obligatorio.
- Las respuestas se codifican e informan por las líneas **`HRESP[1:0]`**.

Respuesta desde el Slave

- Cada Slave debe tener un número máximo predeterminado de ***wait states*** a insertar alcanzado el cual vuelve a ceder el bus, de modo de permitir estimaciones de demoras de acceso al bus por parte de un Master
- Se recomienda un máximo de 16 ***wait states*** . No es obligatorio.
- Las respuestas se codifican e informan por las líneas **`HRESP[1:0]`**.
- Cuando un Slave necesita insertar una cantidad de ***wait states*** debe colocar **`OKAY`**.

Respuesta desde el Slave

- Cada Slave debe tener un número máximo predeterminado de ***wait states*** a insertar alcanzado el cual vuelve a ceder el bus, de modo de permitir estimaciones de demoras de acceso al bus por parte de un Master
- Se recomienda un máximo de 16 ***wait states*** . No es obligatorio.
- Las respuestas se codifican e informan por las líneas **`HRESP[1:0]`**.
- Cuando un Slave necesita insertar una cantidad de ***wait states*** debe colocar **`OKAY`**.
- **`ERROR`**,**`RETRY`**, y **`SPLIT`** necesitan dos ciclos ya que en el primero se fuerza **`HREADY`** a estado Bajo, y en el segundo ciclo se la regresa a estado Alto, para colocar el código correspondiente en **`HRESP[1:0]`**.

Respuesta desde el Slave

- Cada Slave debe tener un número máximo predeterminado de ***wait states*** a insertar alcanzado el cual vuelve a ceder el bus, de modo de permitir estimaciones de demoras de acceso al bus por parte de un Master
- Se recomienda un máximo de 16 ***wait states*** . No es obligatorio.
- Las respuestas se codifican e informan por las líneas **`HRESP[1:0]`**.
- Cuando un Slave necesita insertar una cantidad de ***wait states*** debe colocar **`OKAY`**.
- **`ERROR`**,**`RETRY`**, y **`SPLIT`** necesitan dos ciclos ya que en el primero se fuerza **`HREADY`** a estado Bajo, y en el segundo ciclo se la regresa a estado Alto, para colocar el código correspondiente en **`HRESP[1:0]`**.
- De este modo y por el funcionamiento pipeline del bus el Master tiene tiempo de cancelar la transferencia en curso y poner **`HTRANS[1:`** en **`IDLE`**, para iniciar una transferencia nueva.

Respuesta desde el Slave

HRESP[1:0]	Respuesta	Descripción

Respuesta desde el Slave

HRESP[1:0]	Respuesta	Descripción
0 0	OKAY	Cuando **HREADY** es Alta éste código indica que la transferencia se completó con éxito. Si combina con **HREADY** Baja, indica que se insertan ciclos adicionales

Respuesta desde el Slave

HRESP[1:0]	Respuesta	Descripción
0 0	OKAY	Cuando **HREADY** es Alta éste código indica que la transferencia se completó con éxito. Si combina con **HREADY** Baja, indica que se insertan ciclos adicionales
0 1	ERROR	Indica la ocurrencia de un error, para que el Master lo tome y pueda proceder. La respuesta requiere dos ciclos de bus.

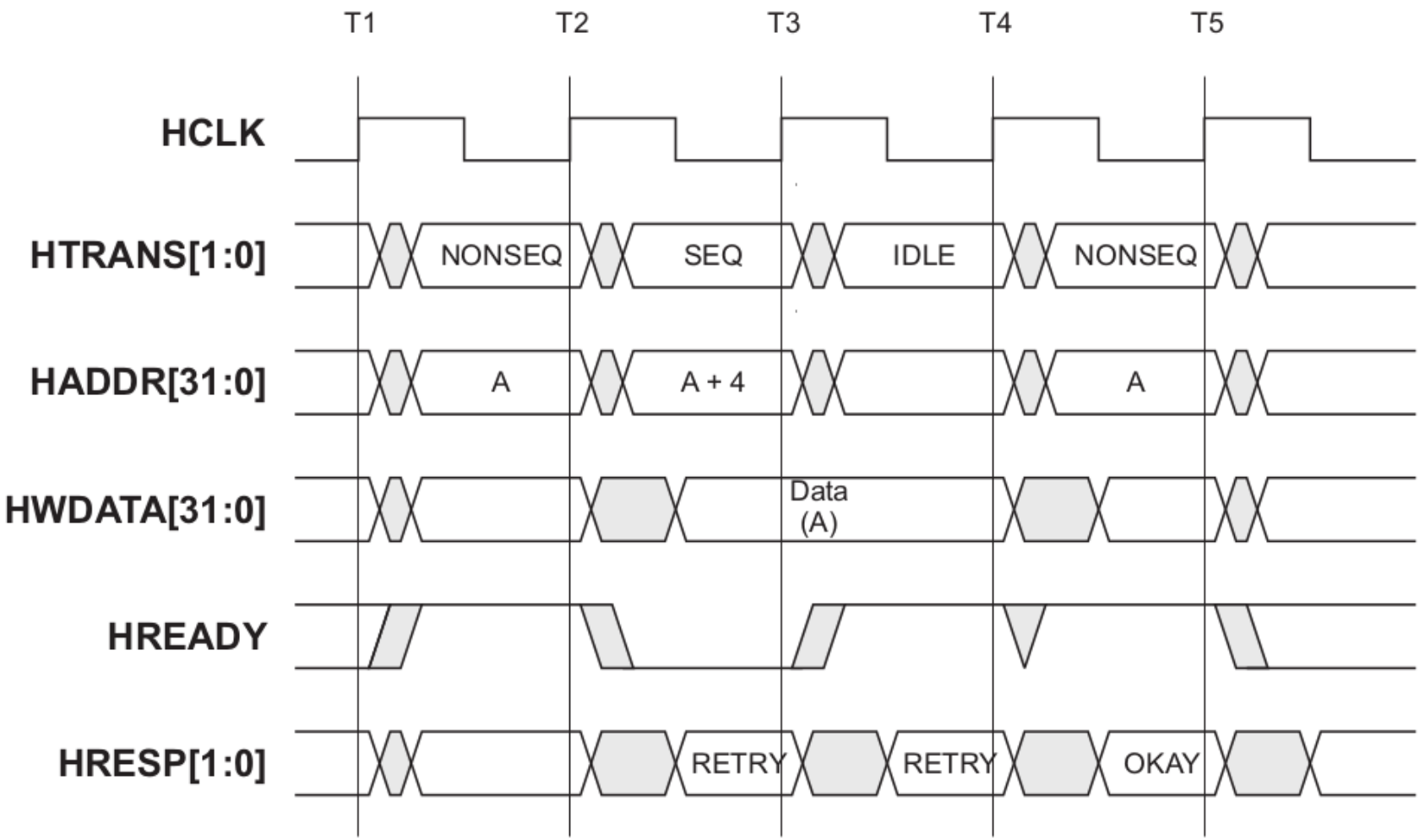
Respuesta desde el Slave

HRESP[1:0]	Respuesta	Descripción
0 0	OKAY	Cuando **HREADY** es Alta éste código indica que la transferencia se completó con éxito. Si combina con **HREADY** Baja, indica que se insertan ciclos adicionales
0 1	ERROR	Indica la ocurrencia de un error, para que el Master lo tome y pueda proceder. La respuesta requiere dos ciclos de bus.
1 0	RETRY	La transferencia no ha sido completada, y el Master debe continuar reintentándola hasta completarla. También insume dos ciclos de bus.

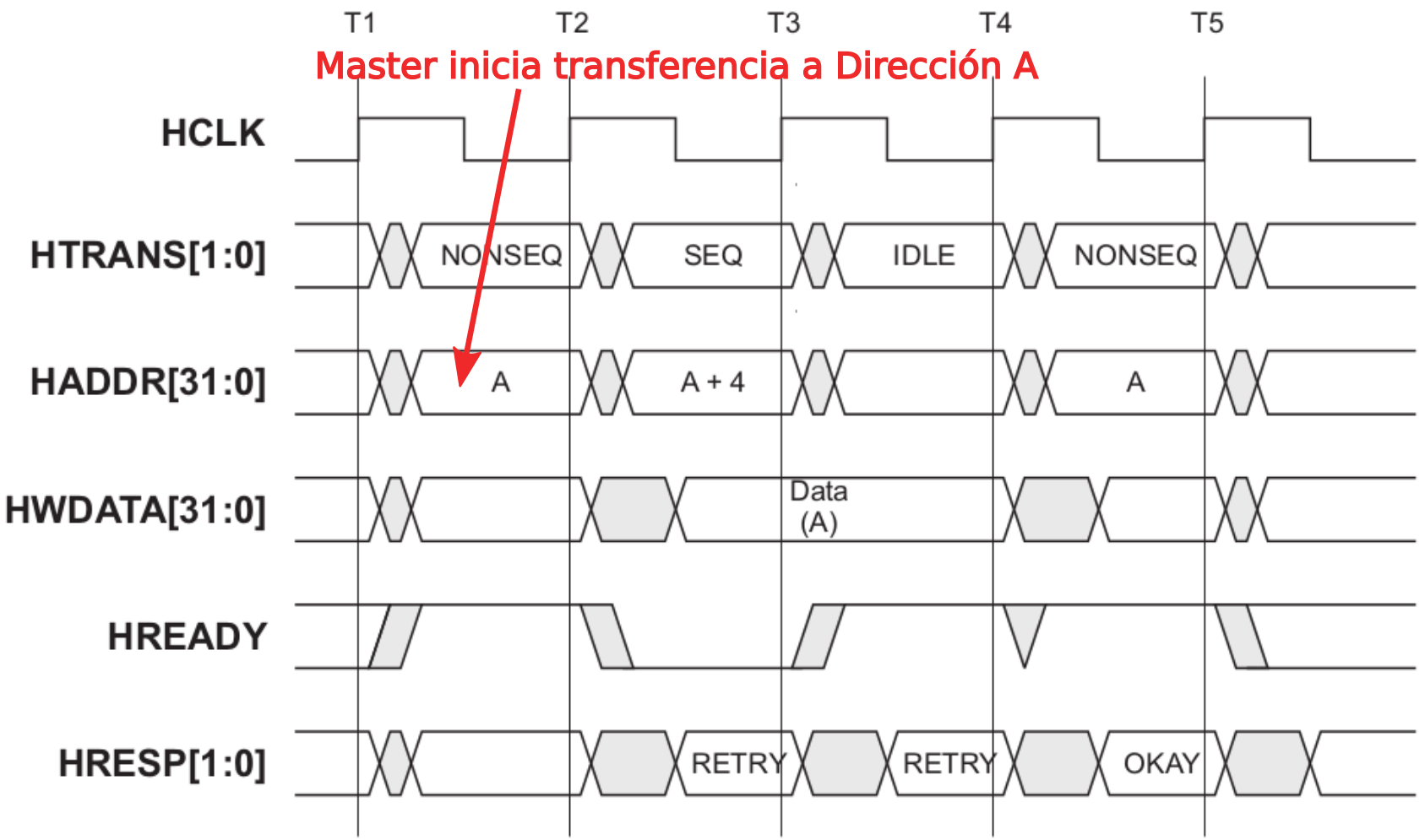
Respuesta desde el Slave

HRESP[1:0]	Respuesta	Descripción
0 0	OKAY	Cuando **HREADY** es Alta éste código indica que la transferencia se completó con éxito. Si combina con **HREADY** Baja, indica que se insertan ciclos adicionales
0 1	ERROR	Indica la ocurrencia de un error, para que el Master lo tome y pueda proceder. La respuesta requiere dos ciclos de bus.
1 0	RETRY	La transferencia no ha sido completada, y el Master debe continuar reintentándola hasta completarla. También insume dos ciclos de bus.
1 1	SPLIT	La transferencia no se ha completado. El Master deberá reintentar la transferencia cuando le sea concedido el bus. También insume dos ciclos de bus

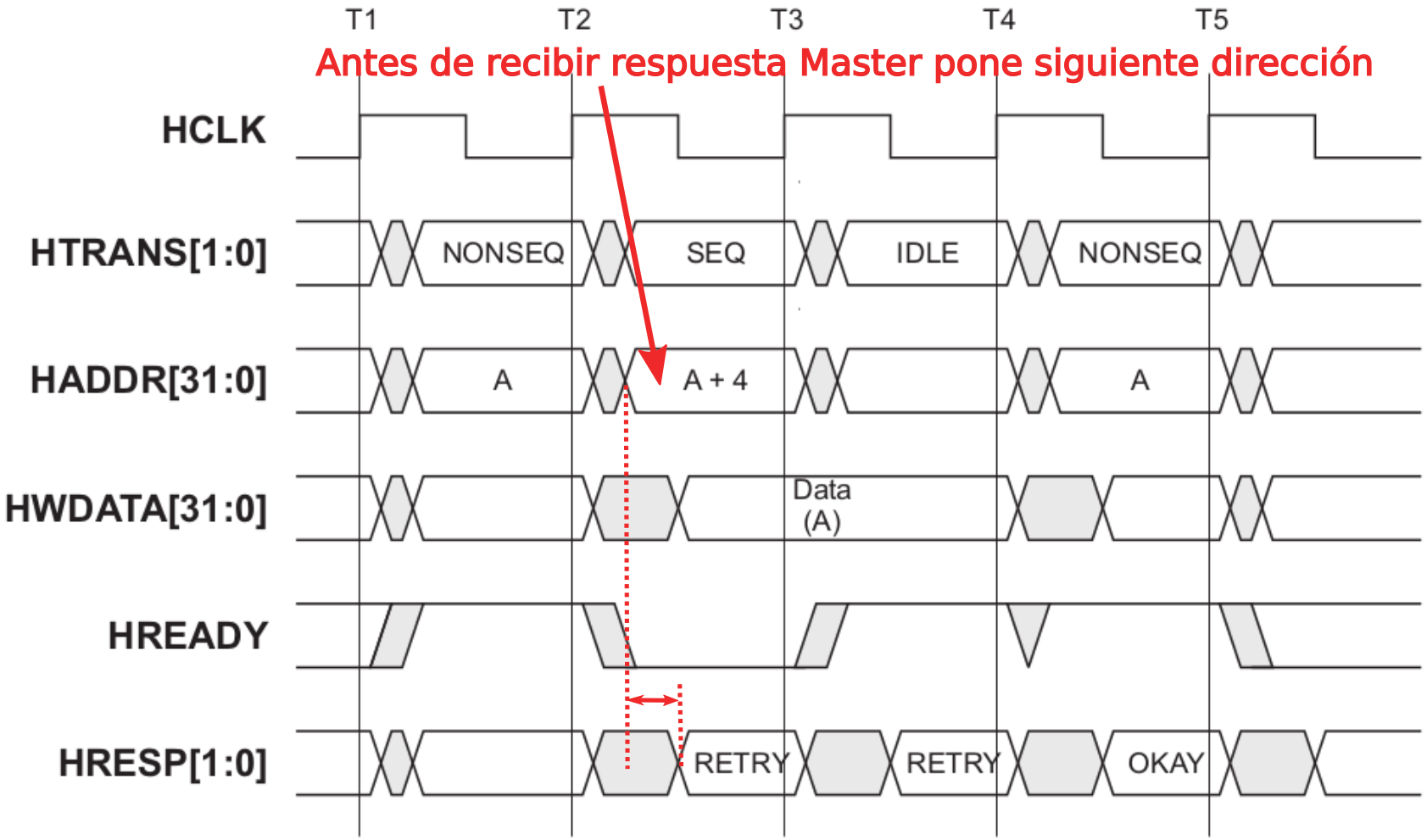
Respuesta RETRY



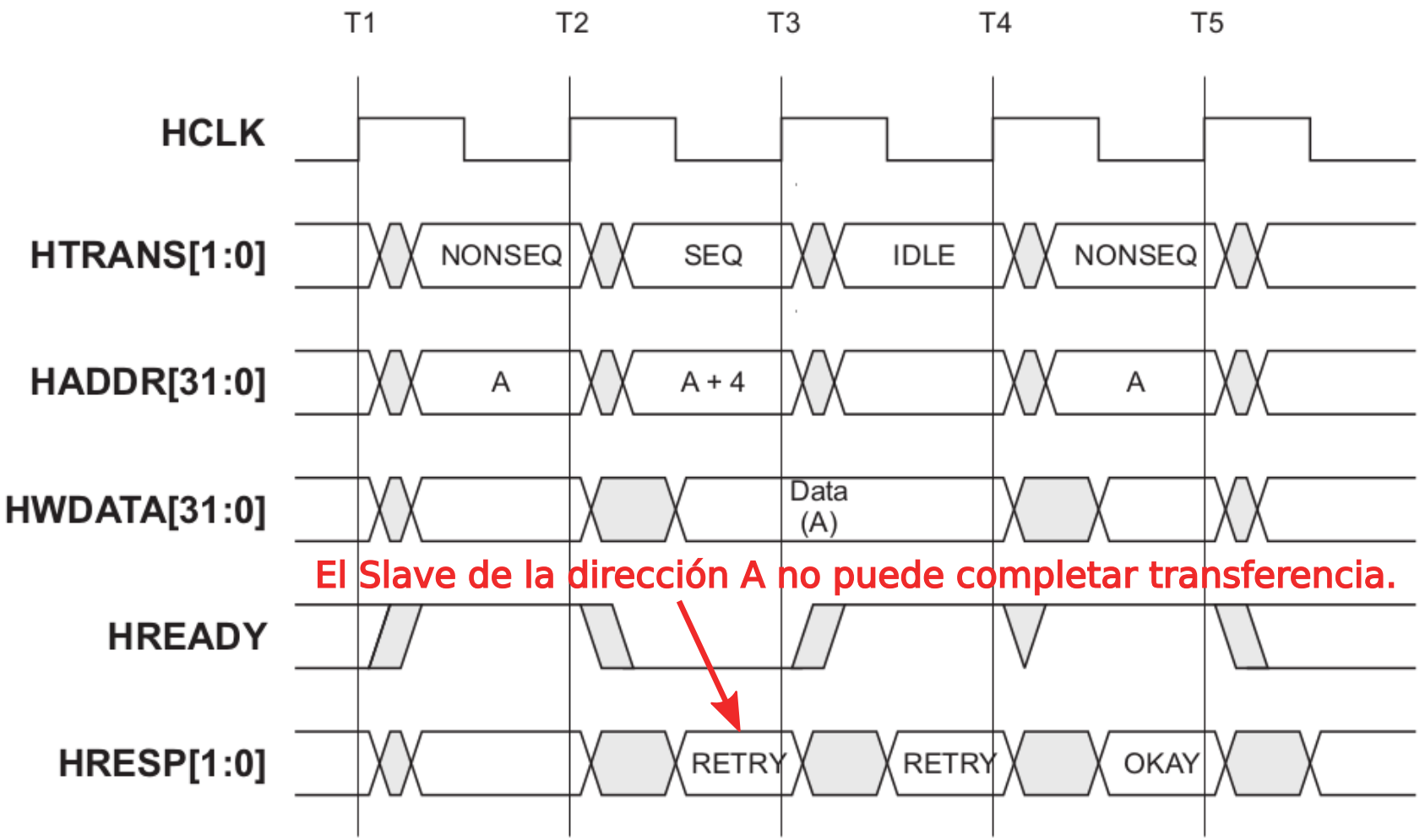
Respuesta RETRY



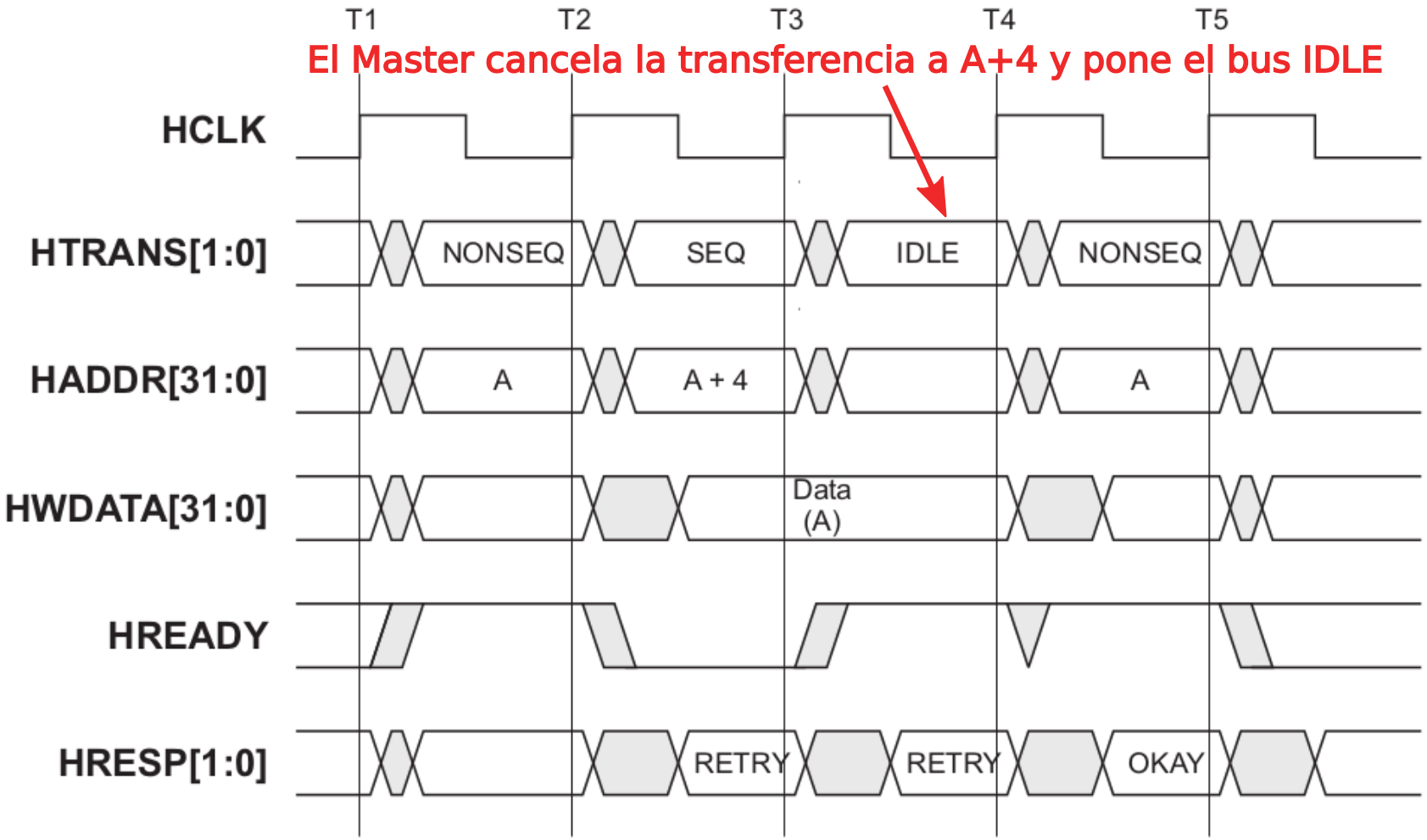
Respuesta RETRY



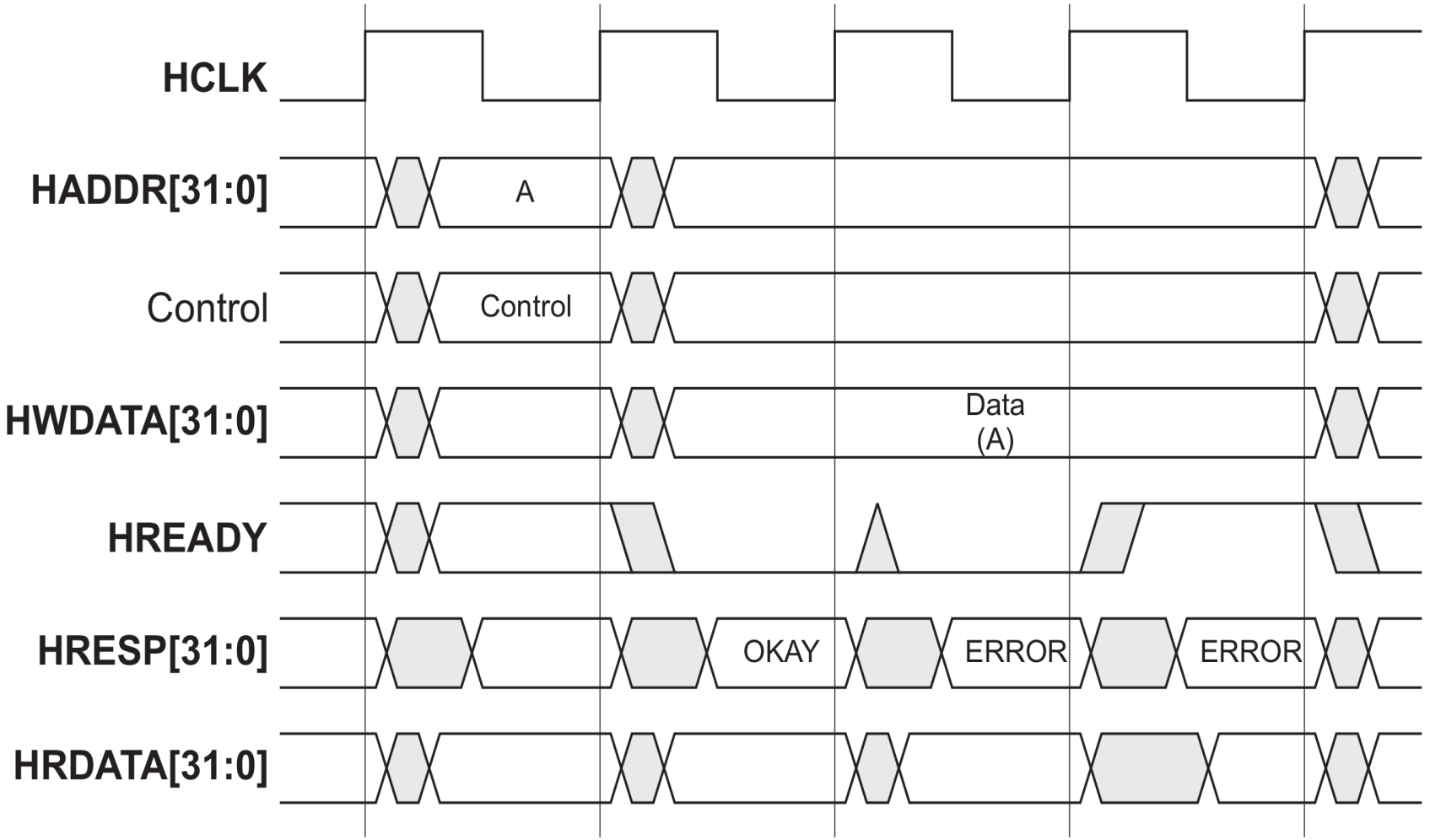
Respuesta RETRY



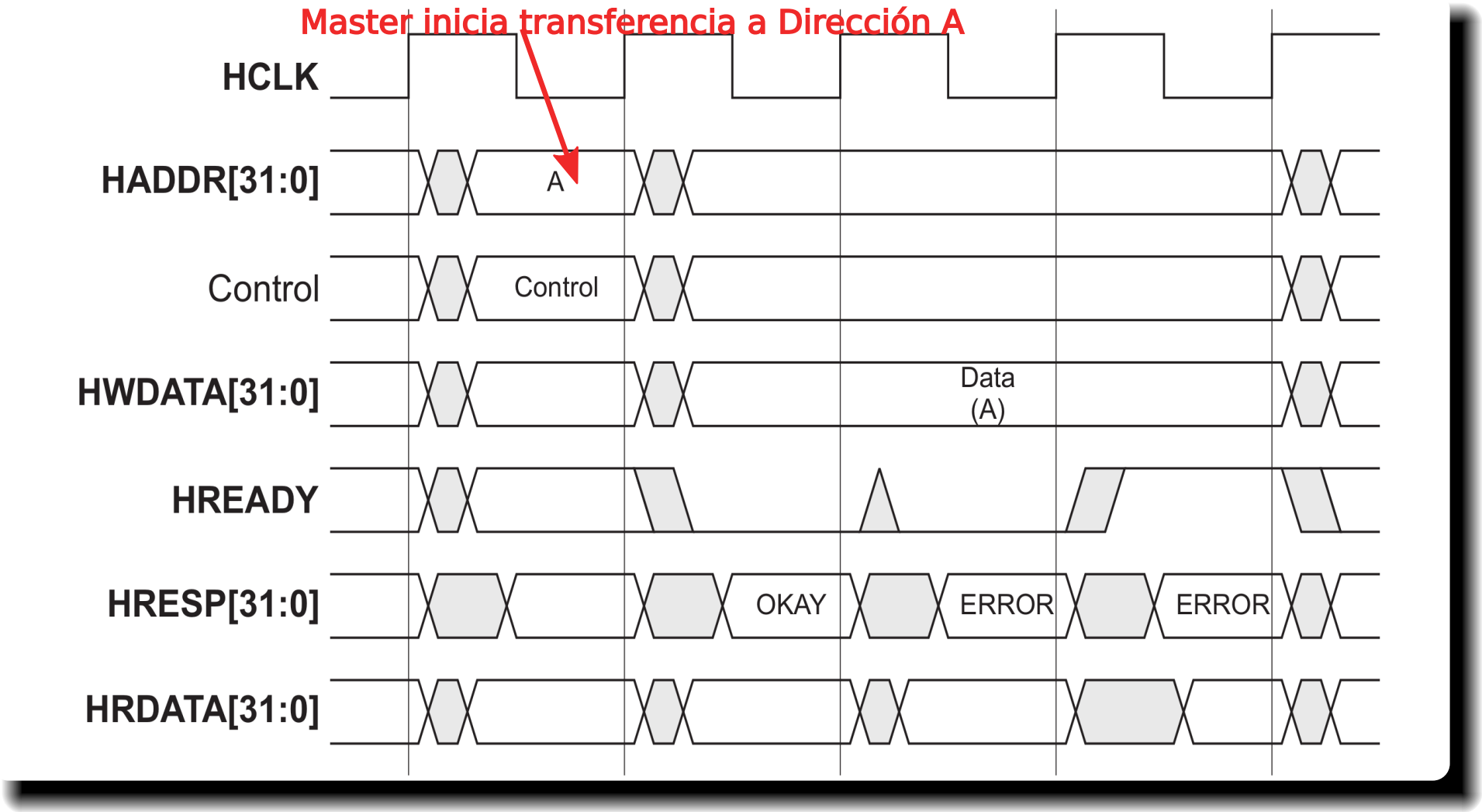
Respuesta RETRY



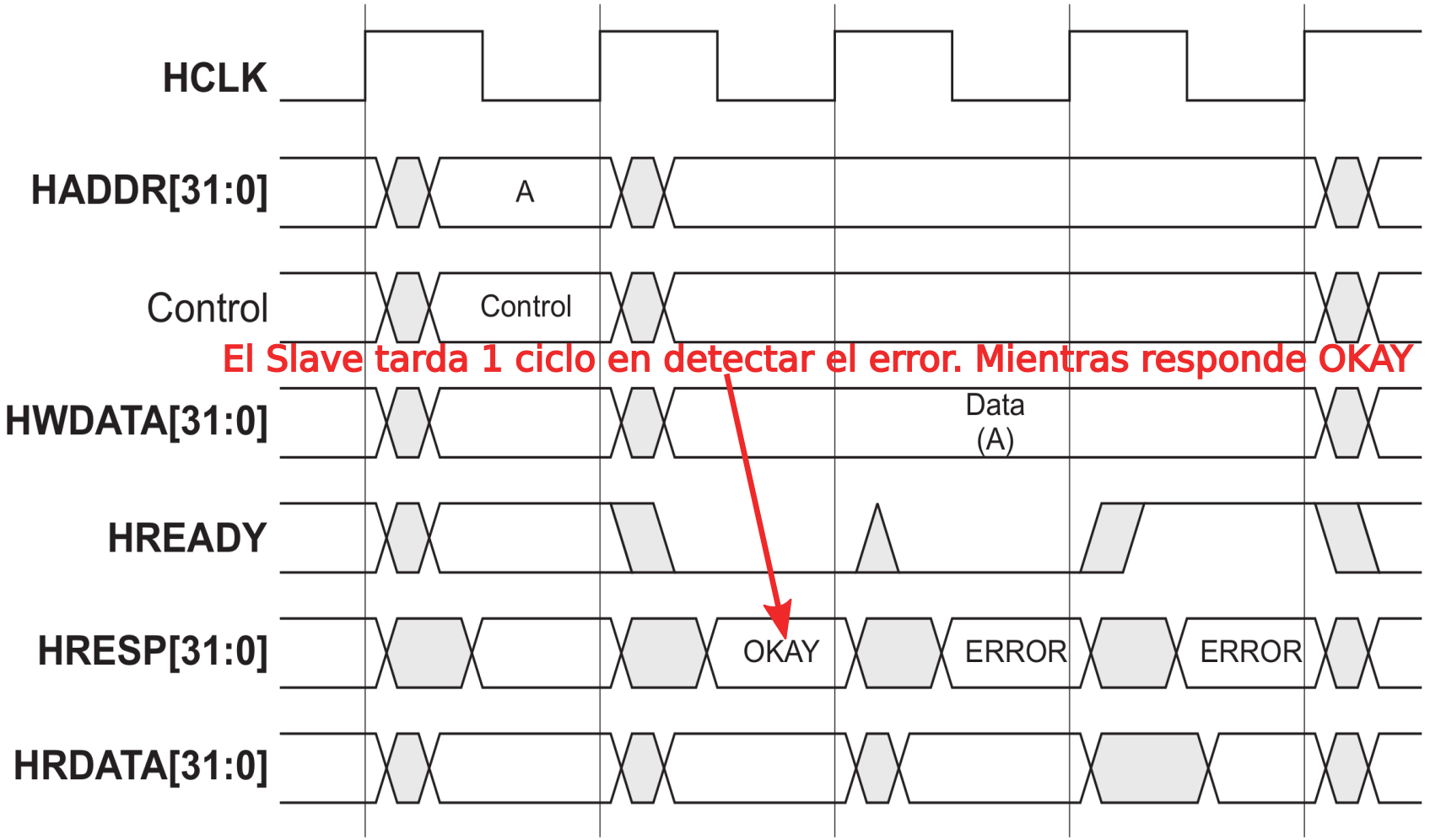
Respuesta ERROR



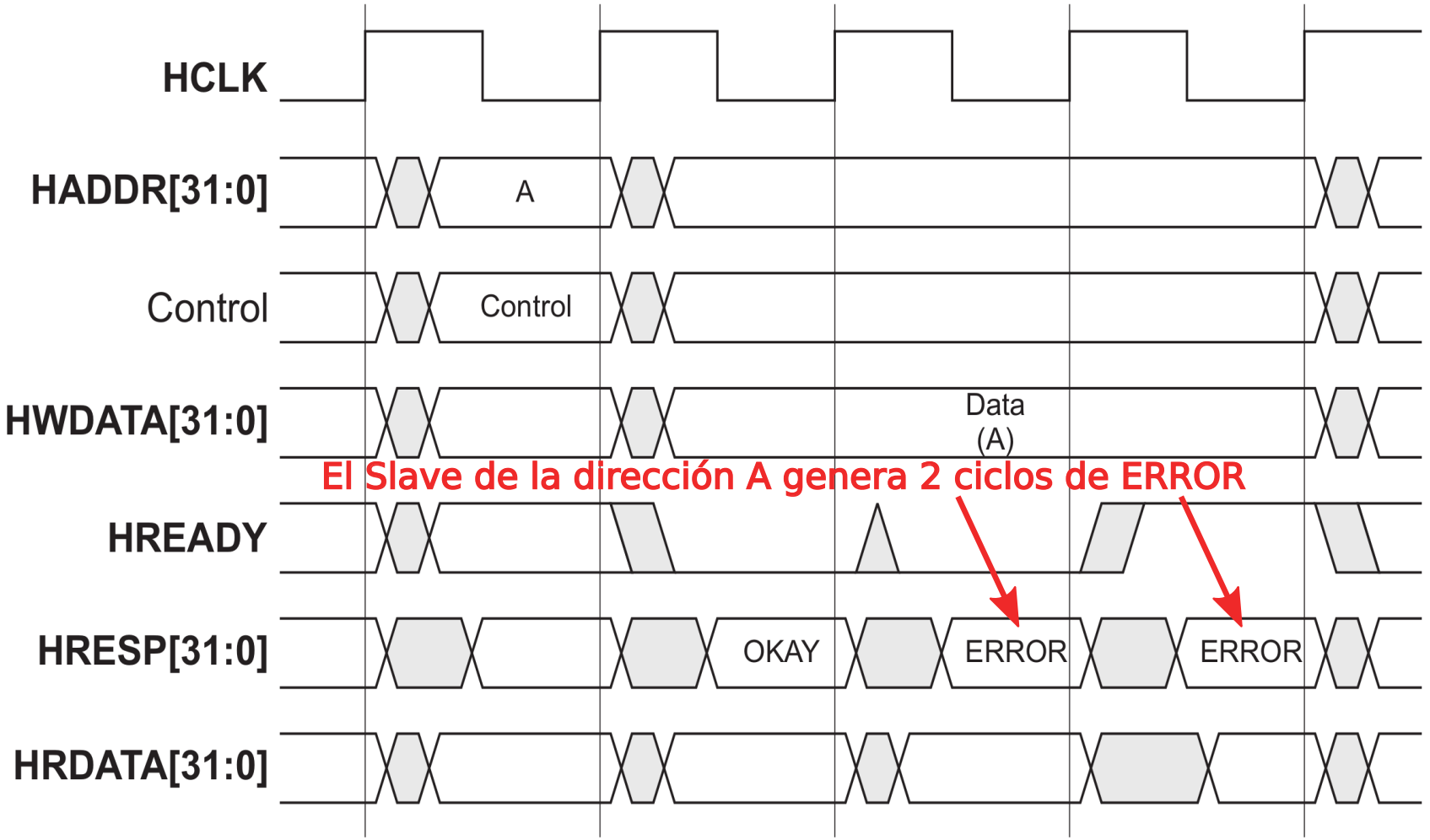
Respuesta ERROR



Respuesta ERROR



Respuesta ERROR



Respuesta ERROR

Cuando el Slave responde **ERROR**, es esperable que el Master suspenda el resto de una transferencia Burst, aunque no esté taxativamente indicado. No obstante a no estar taxativamente indicado, el Master tranquilamente puede continuar con el resto del burst.

Respuesta SPLIT y RETRY

Respuesta SPLIT y RETRY

- Liberan el bus permitiendo que un Master de mayor prioridad lo consiga.

Respuesta SPLIT y RETRY

- Liberan el bus permitiendo que un Master de mayor prioridad lo consiga.
- La diferencia entre ambas respuestas consiste en la forma en que el Árbitro del Bus asigna el bus luego de la respuesta.

Respuesta SPLIT y RETRY

- Liberan el bus permitiendo que un Master de mayor prioridad lo consiga.
- La diferencia entre ambas respuestas consiste en la forma en que el Árbitro del Bus asigna el bus luego de la respuesta.
- En el caso de **`RETRY`**, el Árbitro continúa con su esquema de prioridad habitual, y asignará el Bus a otro Master si éste lo está requiriendo y tiene mayor prioridad.

Respuesta SPLIT y RETRY

- Liberan el bus permitiendo que un Master de mayor prioridad lo consiga.
- La diferencia entre ambas respuestas consiste en la forma en que el Árbitro del Bus asigna el bus luego de la respuesta.
- En el caso de **RETRY**, el Árbitro continúa con su esquema de prioridad habitual, y asignará el Bus a otro Master si éste lo está requiriendo y tiene mayor prioridad.
- En el caso de **SPLIT** se ajustará el esquema de prioridades de modo que cualquier Master que lo requiera acceda, aun si tiene prioridad menor. Para completar la transferencia **SPLIT** el Master debe ser advertido cuando el Slave tenga el dato disponible.

Respuesta SPLIT y RETRY

- Liberan el bus permitiendo que un Master de mayor prioridad lo consiga.
- La diferencia entre ambas respuestas consiste en la forma en que el Árbitro del Bus asigna el bus luego de la respuesta.
- En el caso de **RETRY**, el Árbitro continúa con su esquema de prioridad habitual, y asignará el Bus a otro Master si éste lo está requiriendo y tiene mayor prioridad.
- En el caso de **SPLIT** se ajustará el esquema de prioridades de modo que cualquier Master que lo requiera acceda, aun si tiene prioridad menor. Para completar la transferencia **SPLIT** el Master debe ser advertido cuando el Slave tenga el dato disponible.
- Las transferencias **SPLIT** agregan complejidad tanto al Árbitro como al Slave pero optimizan el uso del bus cuando un Slave demora en completar la transferencia, permitiendo además que cualquier Master tenga acceso al bus aunque su prioridad sea menor.

Data buses

Data buses

- Para evitar el uso de drivers se trabaja con dos buses de datos separados de 32 bits.

Data buses

- Para evitar el uso de drivers se trabaja con dos buses de datos separados de 32 bits.
- **`HWDATA[31:0]`** para transferencias de escritura.

Data buses

- Para evitar el uso de drivers se trabaja con dos buses de datos separados de 32 bits.
- **`HWDATA[31:0]`** para transferencias de escritura.
- **`HRDATA[31:0]`** para transferencias de lectura.

Data buses

- Para evitar el uso de drivers se trabaja con dos buses de datos separados de 32 bits.
- **`HWDATA[31:0]`** para transferencias de escritura.
- **`HRDATA[31:0]`** para transferencias de lectura.
- Las transferencias deben estar alineadas de acuerdo con el tamaño de la transferencia. Para trasferencia de menor ancho que el bus, solo se utiliza el ***lane*** apropiado respetando el endianess del sistema.

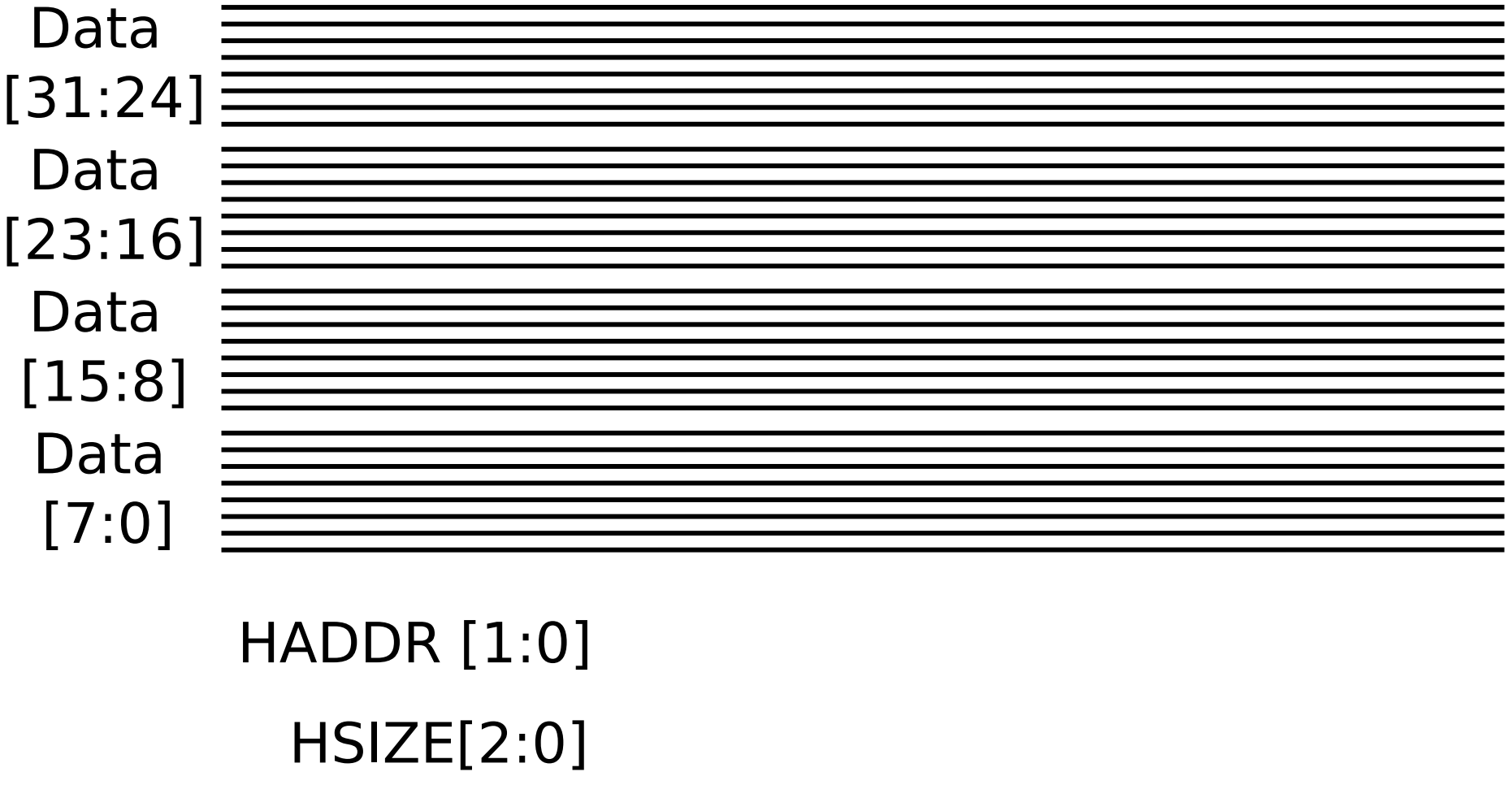
Data buses

- Para evitar el uso de drivers se trabaja con dos buses de datos separados de 32 bits.
- **`HWDATA[31:0]`** para transferencias de escritura.
- **`HRDATA[31:0]`** para transferencias de lectura.
- Las transferencias deben estar alineadas de acuerdo con el tamaño de la transferencia. Para trasferencia de menor ancho que el bus, solo se utiliza el ***lane*** apropiado respetando el endianess del sistema.
- Las escrituras Burt de menor ancho que el bus pueden tener activas simultáneamente diferentes ***lanes*** para cada ***beat***.

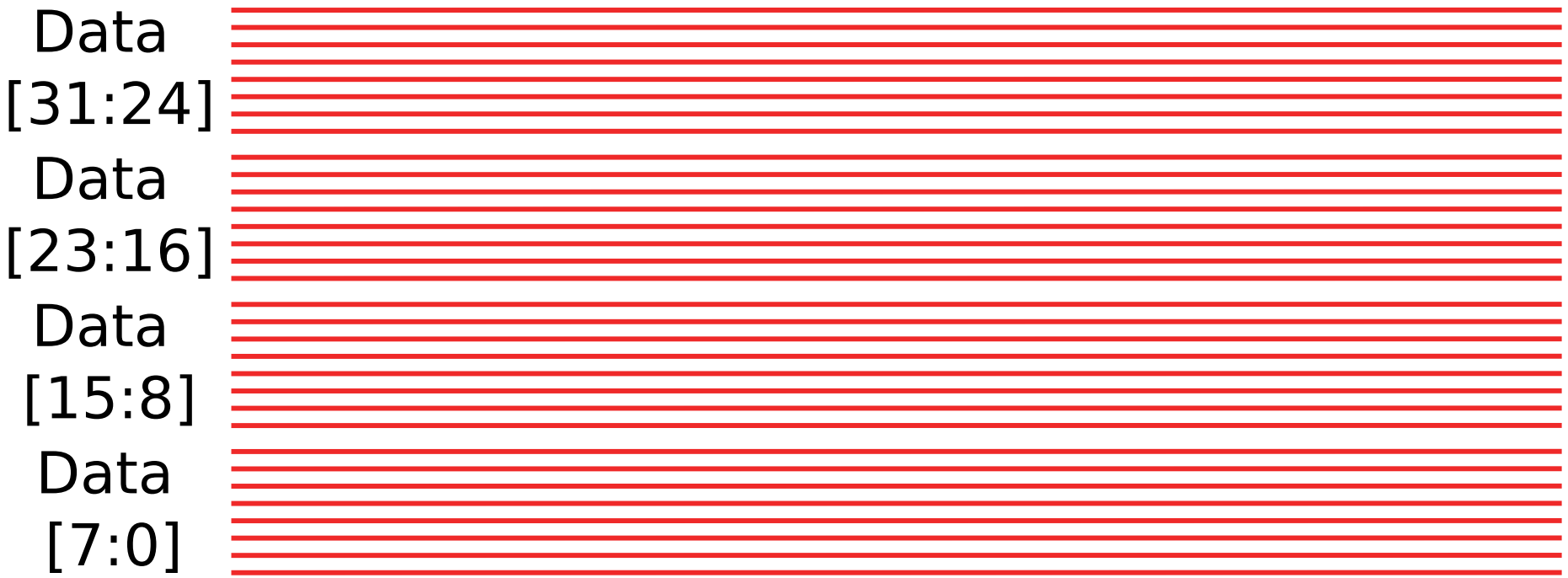
Data buses

- Para evitar el uso de drivers se trabaja con dos buses de datos separados de 32 bits.
- **`HWDATA[31:0]`** para transferencias de escritura.
- **`HRDATA[31:0]`** para transferencias de lectura.
- Las transferencias deben estar alineadas de acuerdo con el tamaño de la transferencia. Para trasferencia de menor ancho que el bus, solo se utiliza el ***lane*** apropiado respetando el endianess del sistema.
- Las escrituras Burt de menor ancho que el bus pueden tener activas simultáneamente diferentes ***lanes*** para cada ***beat***.
- El endianess se debe definir a nivel de diseño para todo el sistema. Dynamic Endianess es muy costosos en recursos y potencia disipada. Solo hay una recomendación a los diseñadores para realizar dispositivos de endianess configurable.

Endianess: Uso del bus de datos



Endianess: Uso del bus de datos

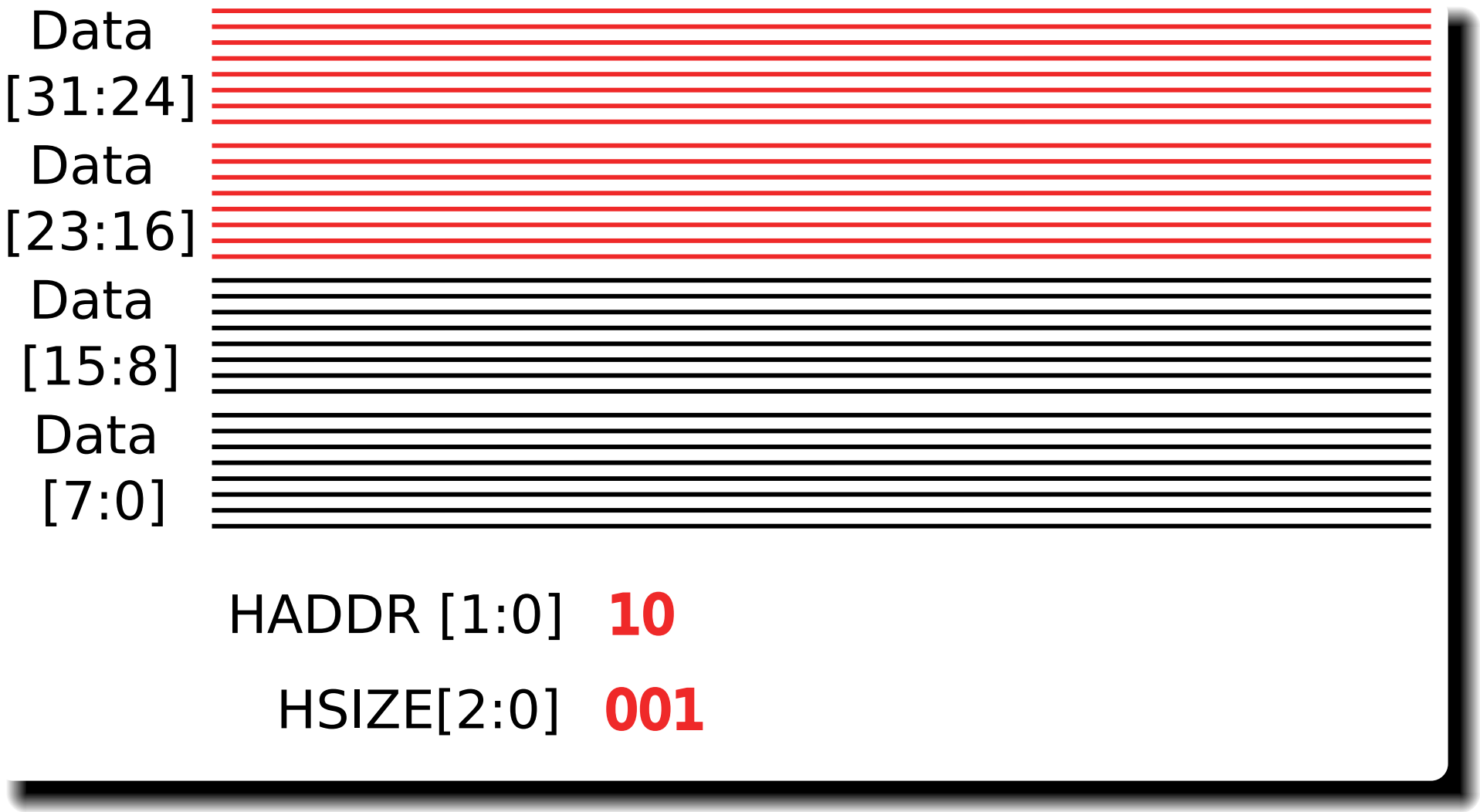


Tamaño Word: Little-Endian y Big-Endian

Endianess: Uso del bus de datos

Tamaño Half Word: Little-Endian

Endianess: Uso del bus de datos

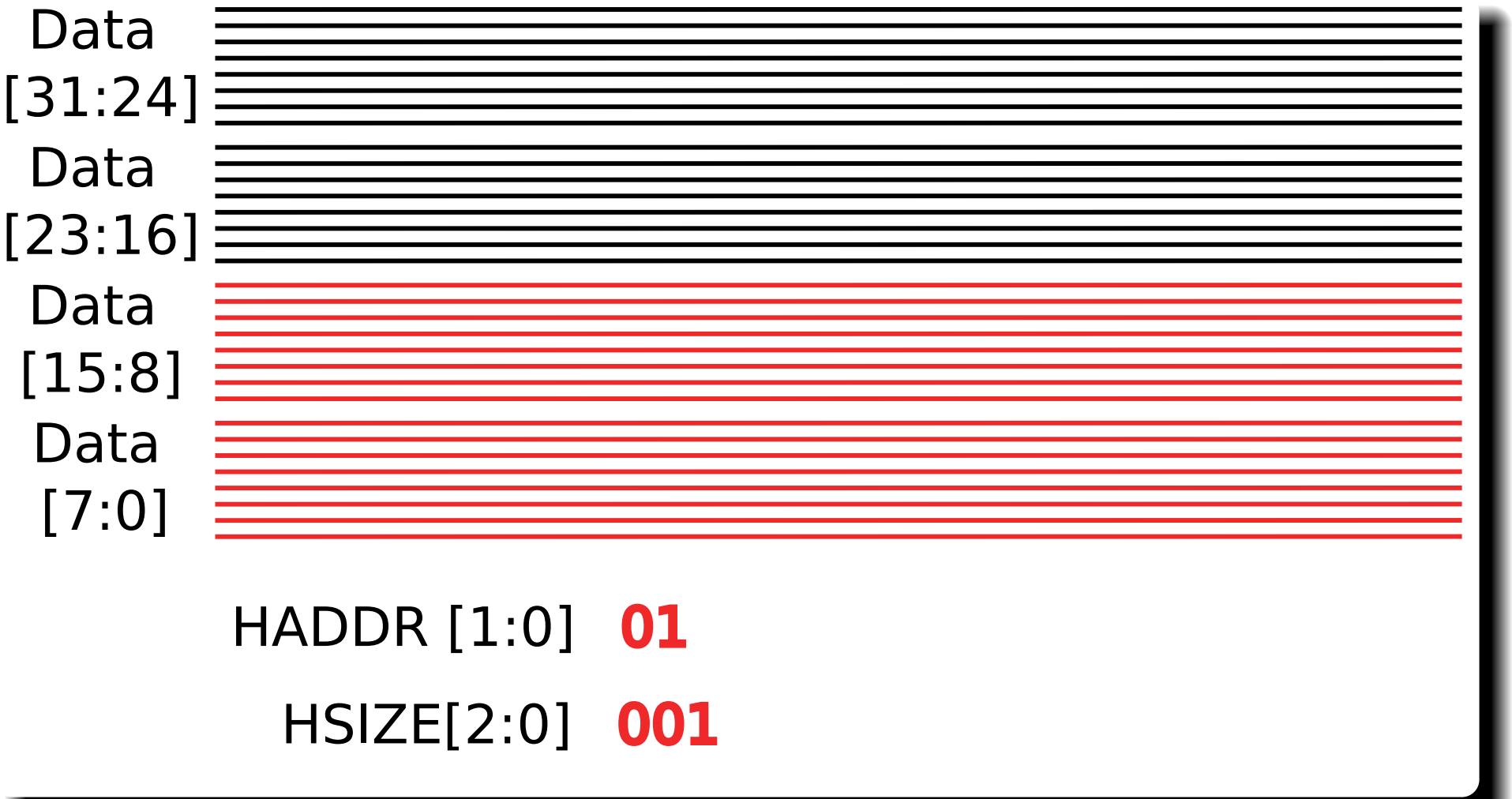


Tamaño Half Word: Little-Endian

Endianess: Uso del bus de datos

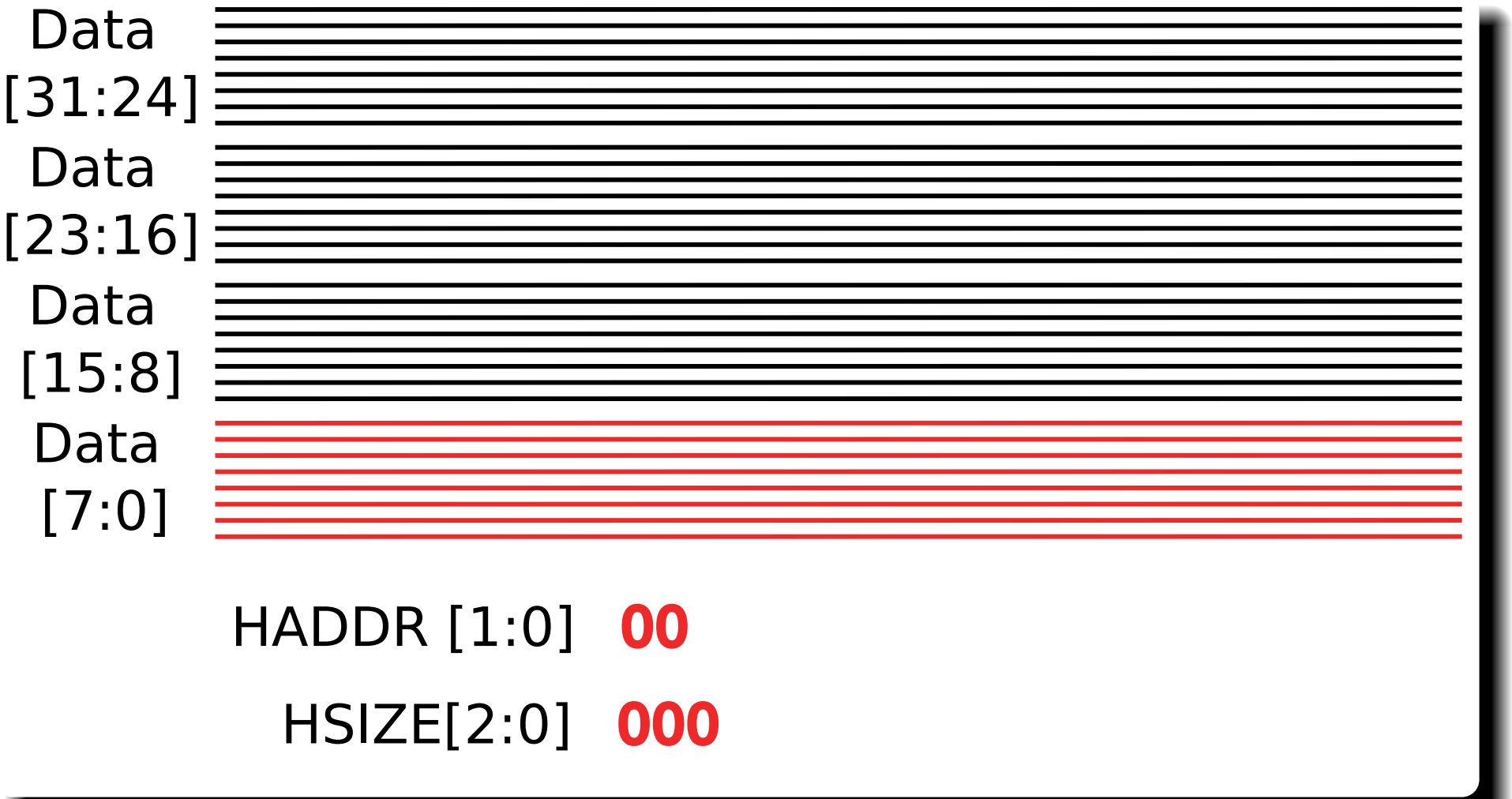
Tamaño Half Word: Big-Endian

Endianess: Uso del bus de datos



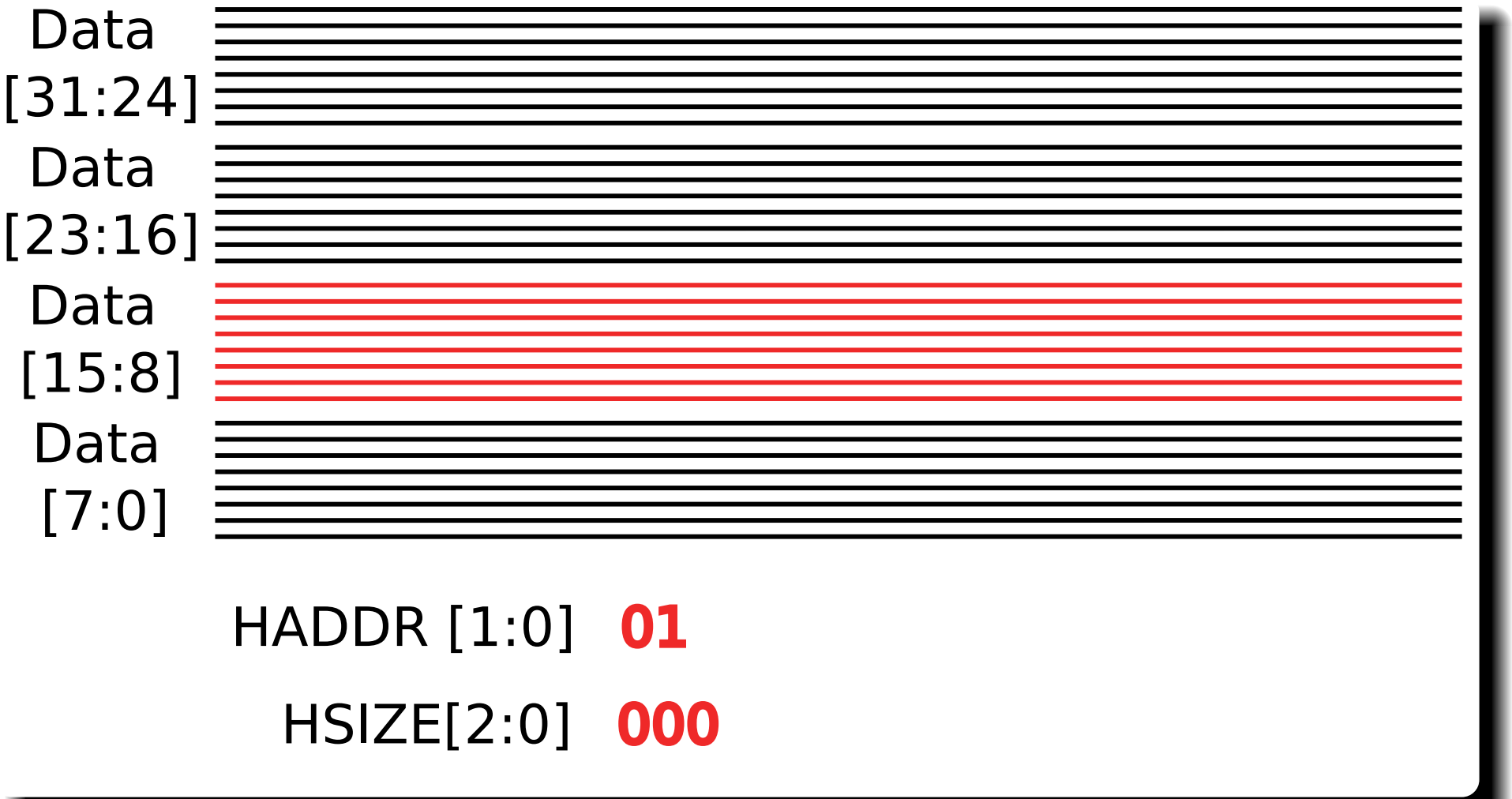
Tamaño Half Word: Big-Endian

Endianess: Uso del bus de datos



Tamaño Byte: Little-Endian

Endianess: Uso del bus de datos

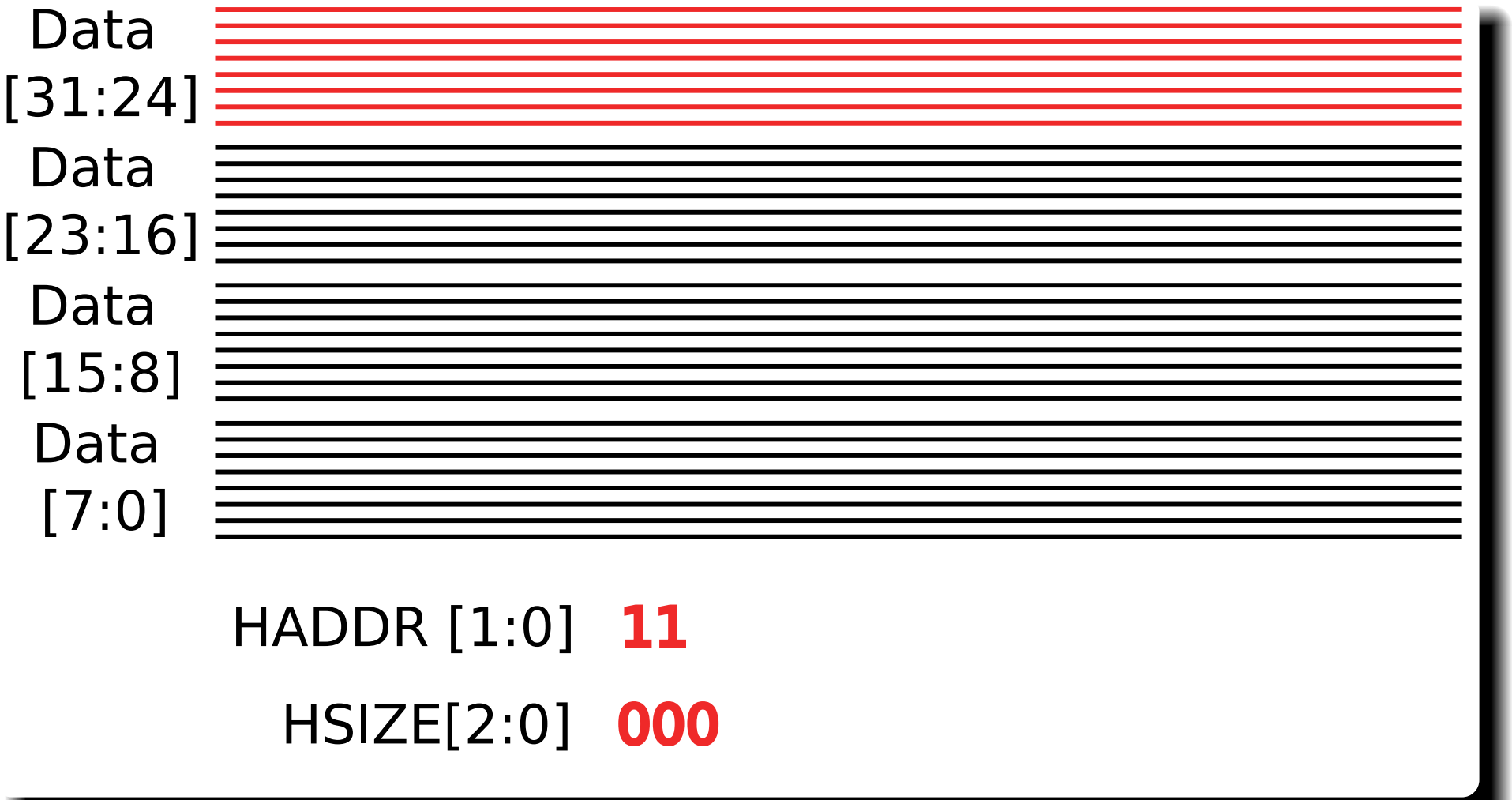


Tamaño Byte: Little-Endian

Endianess: Uso del bus de datos

Tamaño Byte: Little-Endian

Endianess: Uso del bus de datos



Tamaño Byte: Little-Endian

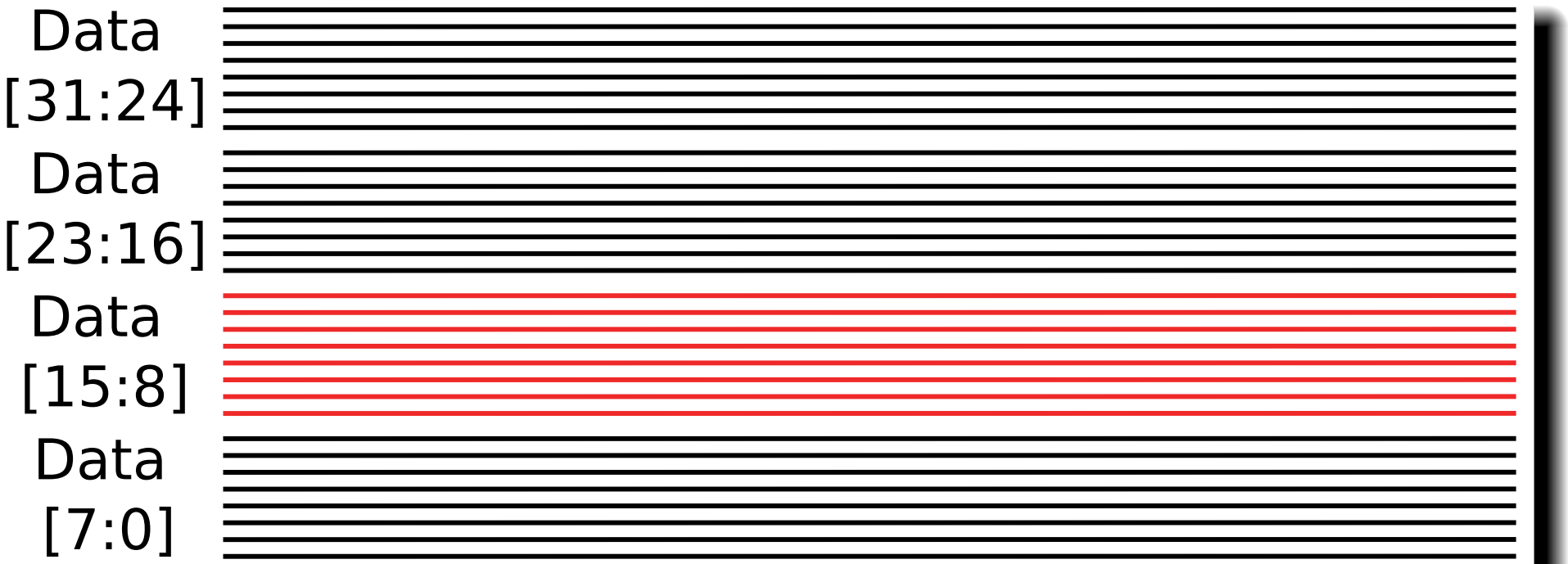
Endianess: Uso del bus de datos

Tamaño Byte: Big-Endian

Endianess: Uso del bus de datos

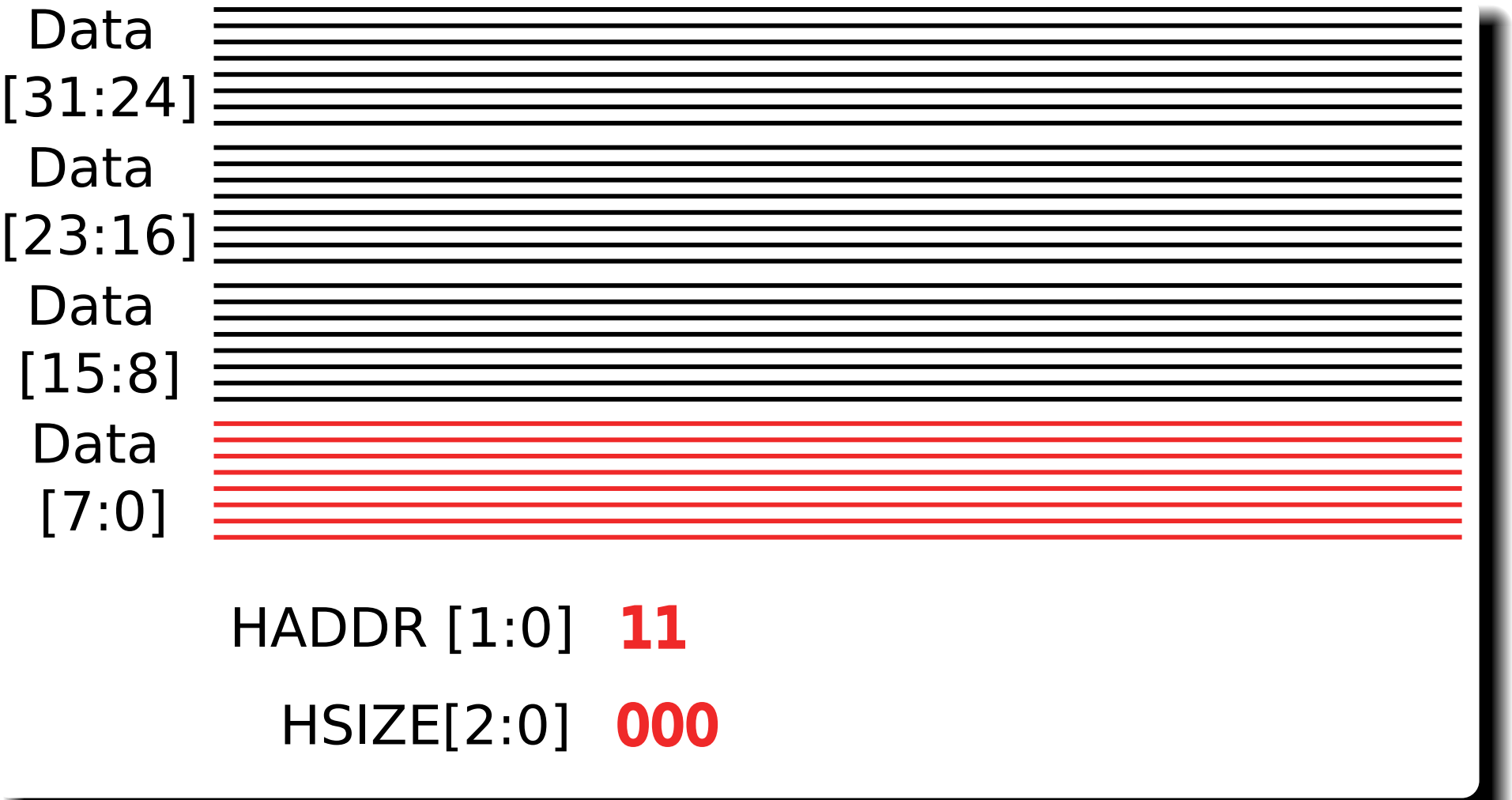
Tamaño Byte: Big-Endian

Endianess: Uso del bus de datos



Tamaño Byte: Big-Endian

Endianess: Uso del bus de datos



Tamaño Byte: Big-Endian

Arbitraje del AHB

Arbitraje del AHB

- Mecanismo de hardware diseñado para ***asegurar*** que en un momento determinado solo un Master acceda al bus.

Arbitraje del AHB

- Mecanismo de hardware diseñado para ***asegurar*** que en un momento determinado solo un Master acceda al bus.
- Utiliza un protocolo Request-Grant introducido mucho antes con la técnica de optimización de transferencias de datos conocida como **A**cceso **D**irecto **M**emoria (***DMA***).

Arbitraje del AHB

- Mecanismo de hardware diseñado para ***asegurar*** que en un momento determinado solo un Master acceda al bus.
- Utiliza un protocolo Request-Grant introducido mucho antes con la técnica de optimización de transferencias de datos conocida como **A**cceso **D**irecto **M**emoria (***DMA***).
- Se asume que una gama de dispositivos heterogéneos que pueden coexistir en un sistema controlando el flujo de transacciones en el bus: Microprocesadores, Microcontroladores, y Controladores de DMA entre otros.

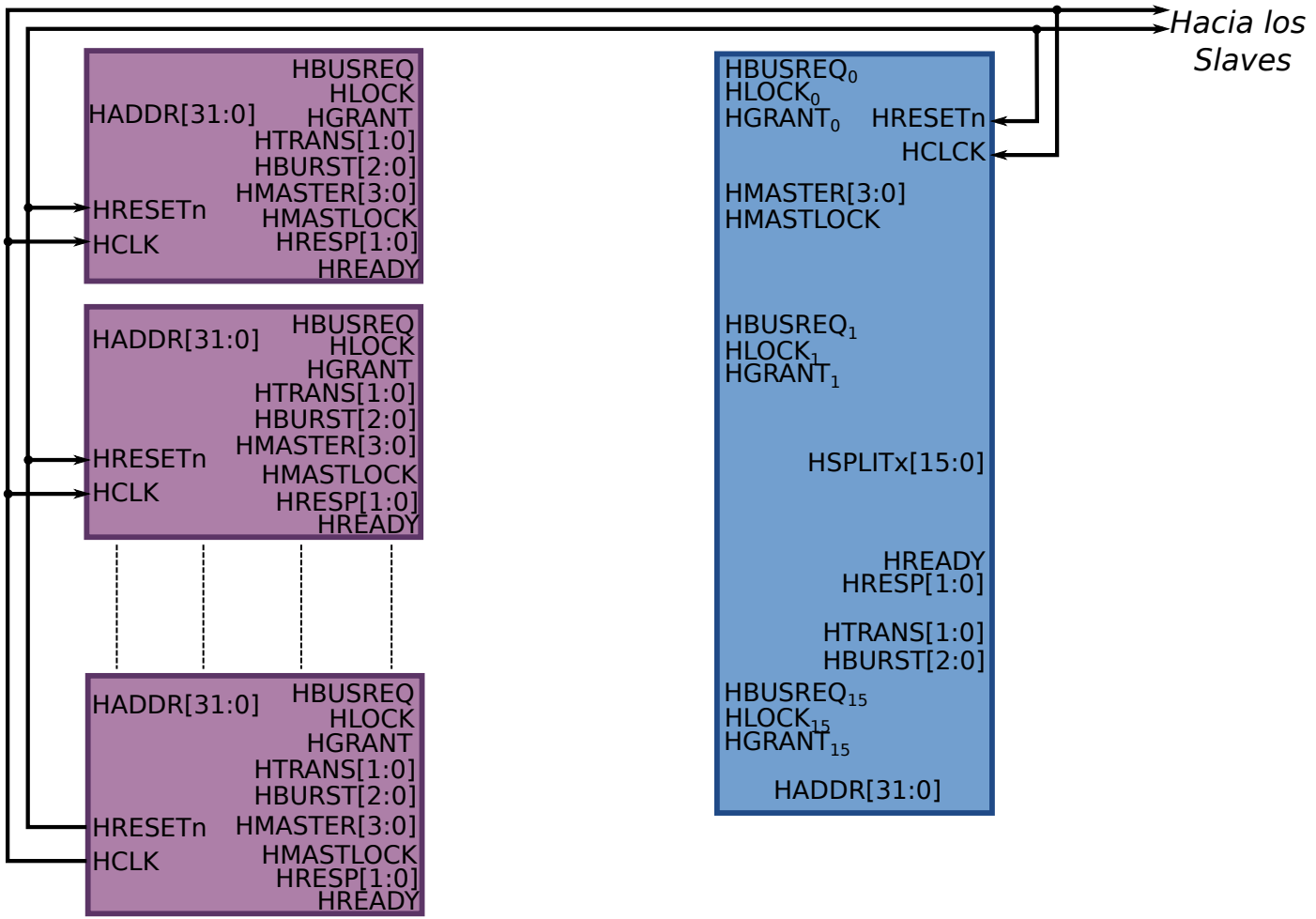
Arbitraje del AHB

- Mecanismo de hardware diseñado para ***asegurar*** que en un momento determinado solo un Master acceda al bus.
- Utiliza un protocolo Request-Grant introducido mucho antes con la técnica de optimización de transferencias de datos conocida como **A**cceso **D**irecto **M**emoria (***DMA***).
- Se asume que una gama de dispositivos heterogéneos que pueden coexistir en un sistema controlando el flujo de transacciones en el bus: Microprocesadores, Microcontroladores, y Controladores de DMA entre otros.
- El Árbitro de Bus además prioriza los requerimientos asignando el Bus a un Master por sobre el resto. El esquema de prioridades es variable de diseño de cada SoC.

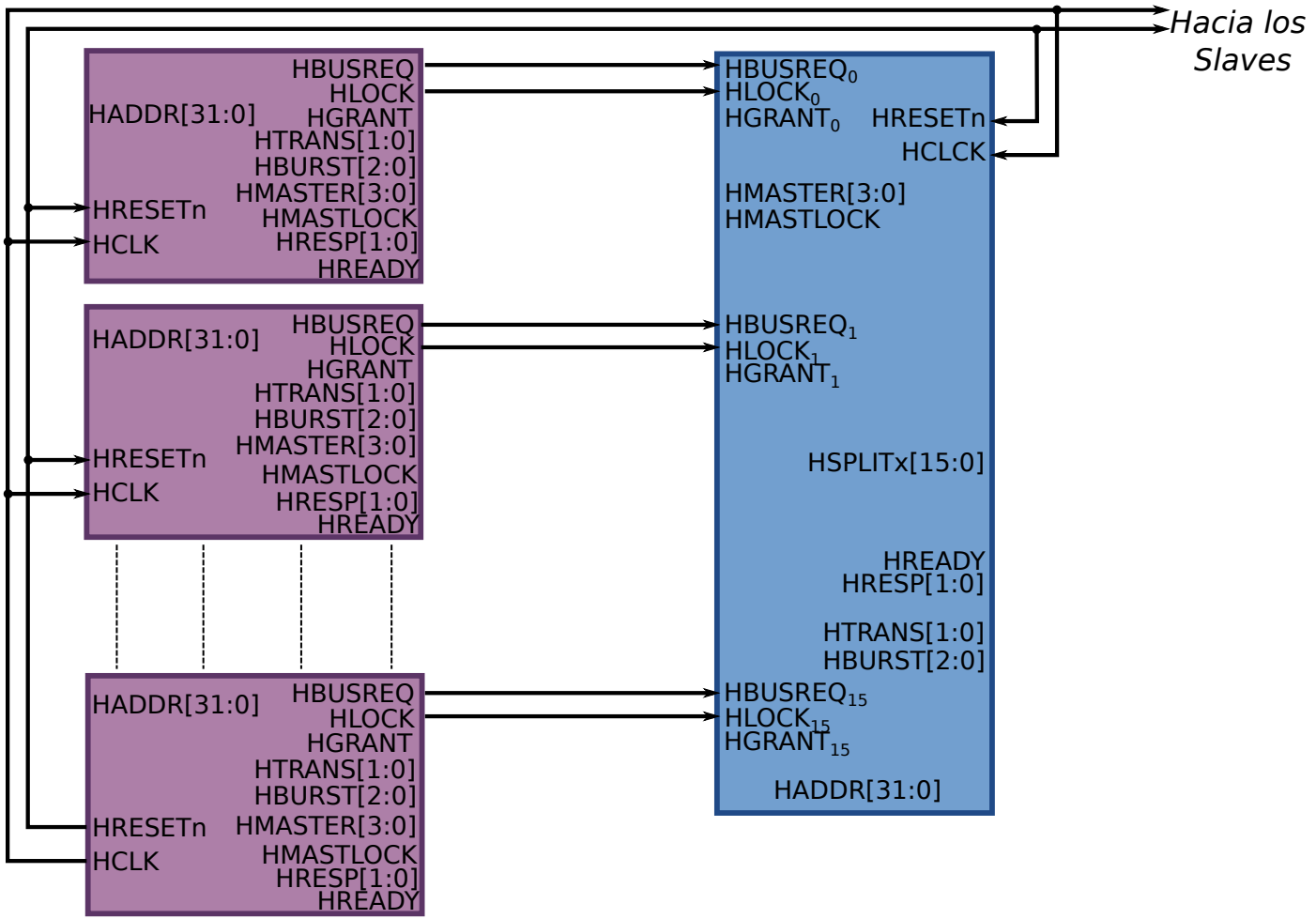
Arbitraje del AHB

- Mecanismo de hardware diseñado para ***asegurar*** que en un momento determinado solo un Master acceda al bus.
- Utiliza un protocolo Request-Grant introducido mucho antes con la técnica de optimización de transferencias de datos conocida como **A**cceso **D**irecto **M**emoria (***DMA***).
- Se asume que una gama de dispositivos heterogéneos que pueden coexistir en un sistema controlando el flujo de transacciones en el bus: Microprocesadores, Microcontroladores, y Controladores de DMA entre otros.
- El Árbitro de Bus además prioriza los requerimientos asignando el Bus a un Master por sobre el resto. El esquema de prioridades es variable de diseño de cada SoC.
- Por otra parte recibe señales de los Slaves que requieren completar transferencias **`SPLIT`**.

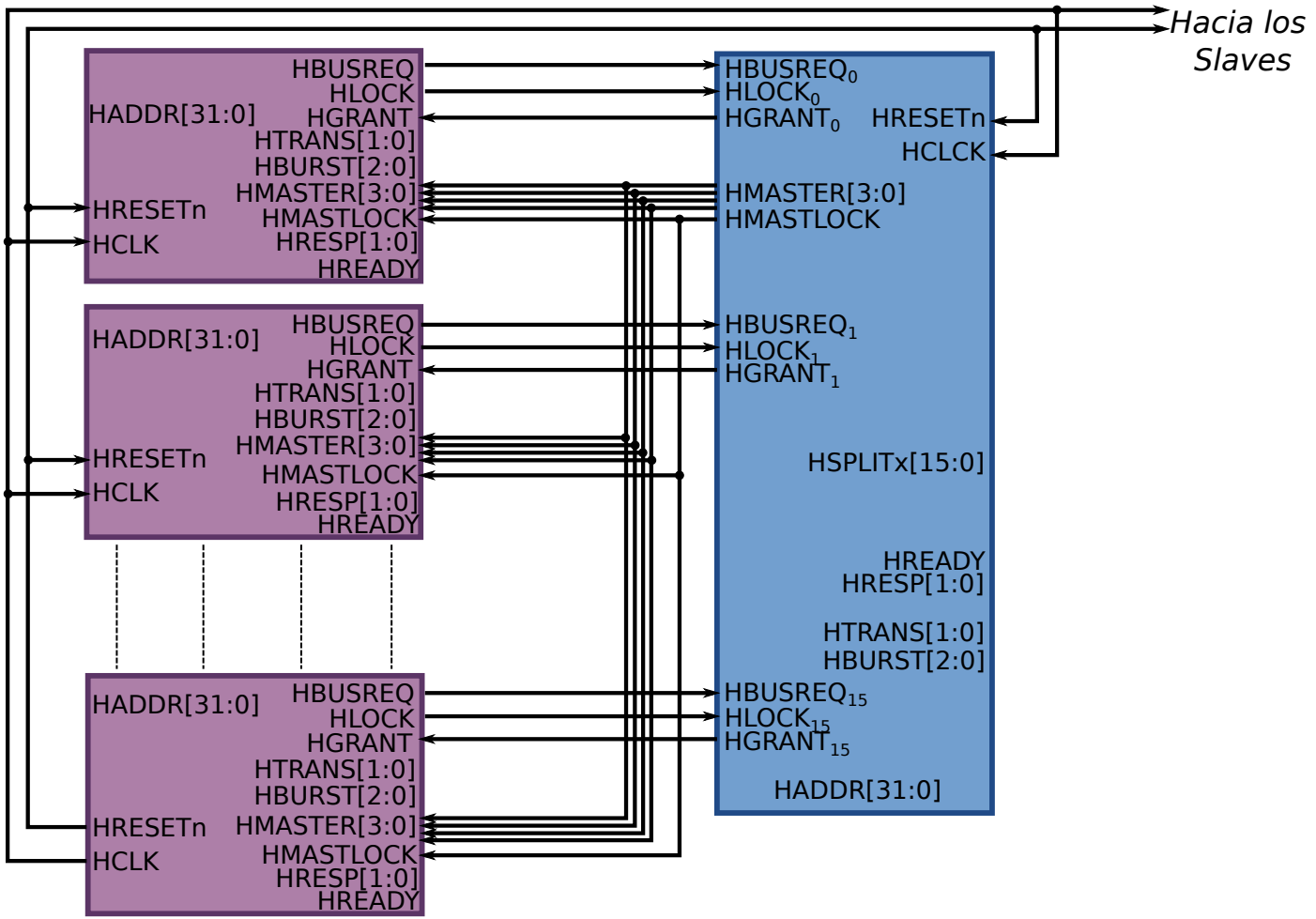
Arbitraje del AHB



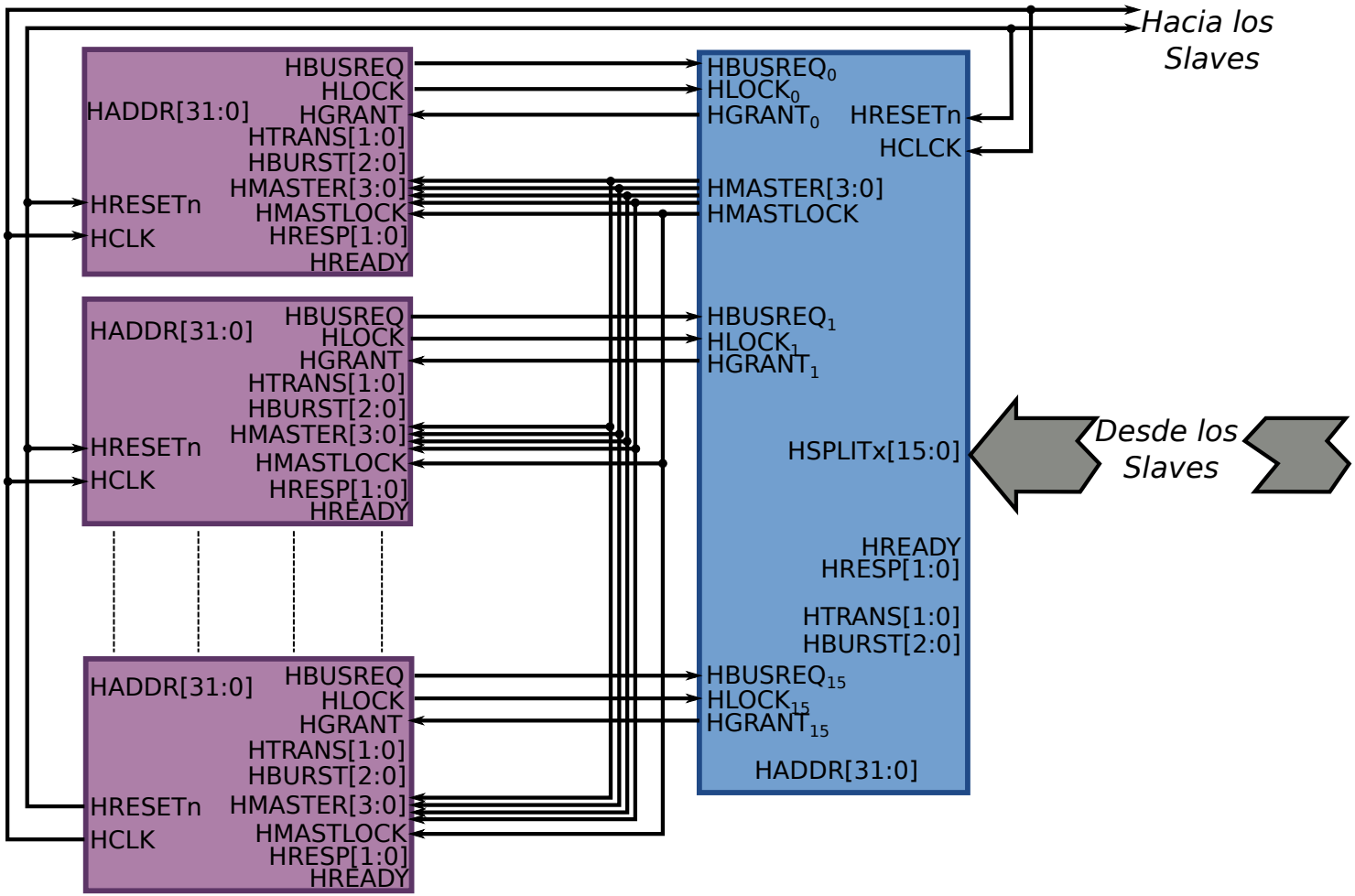
Arbitraje del AHB



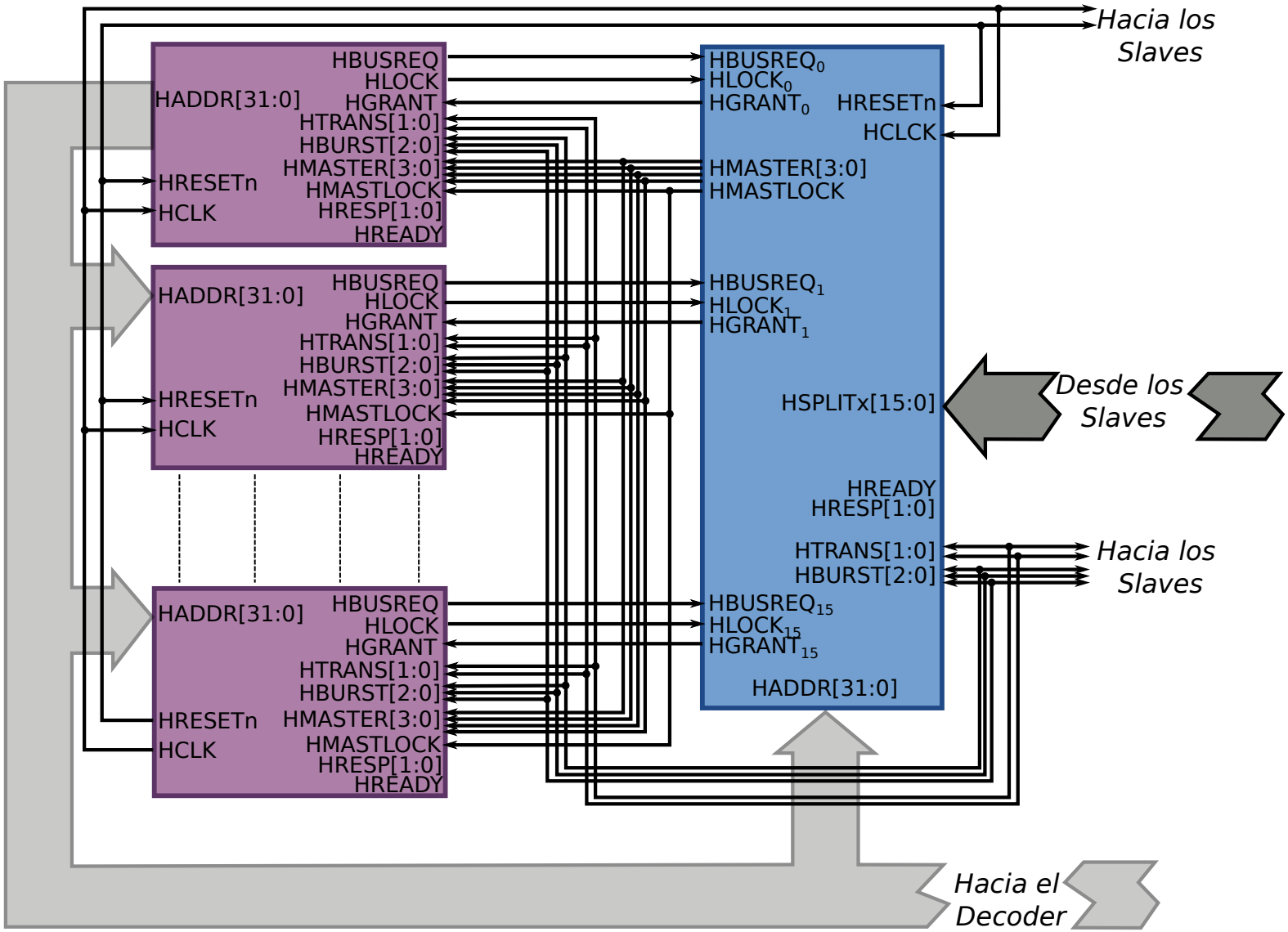
Arbitraje del AHB



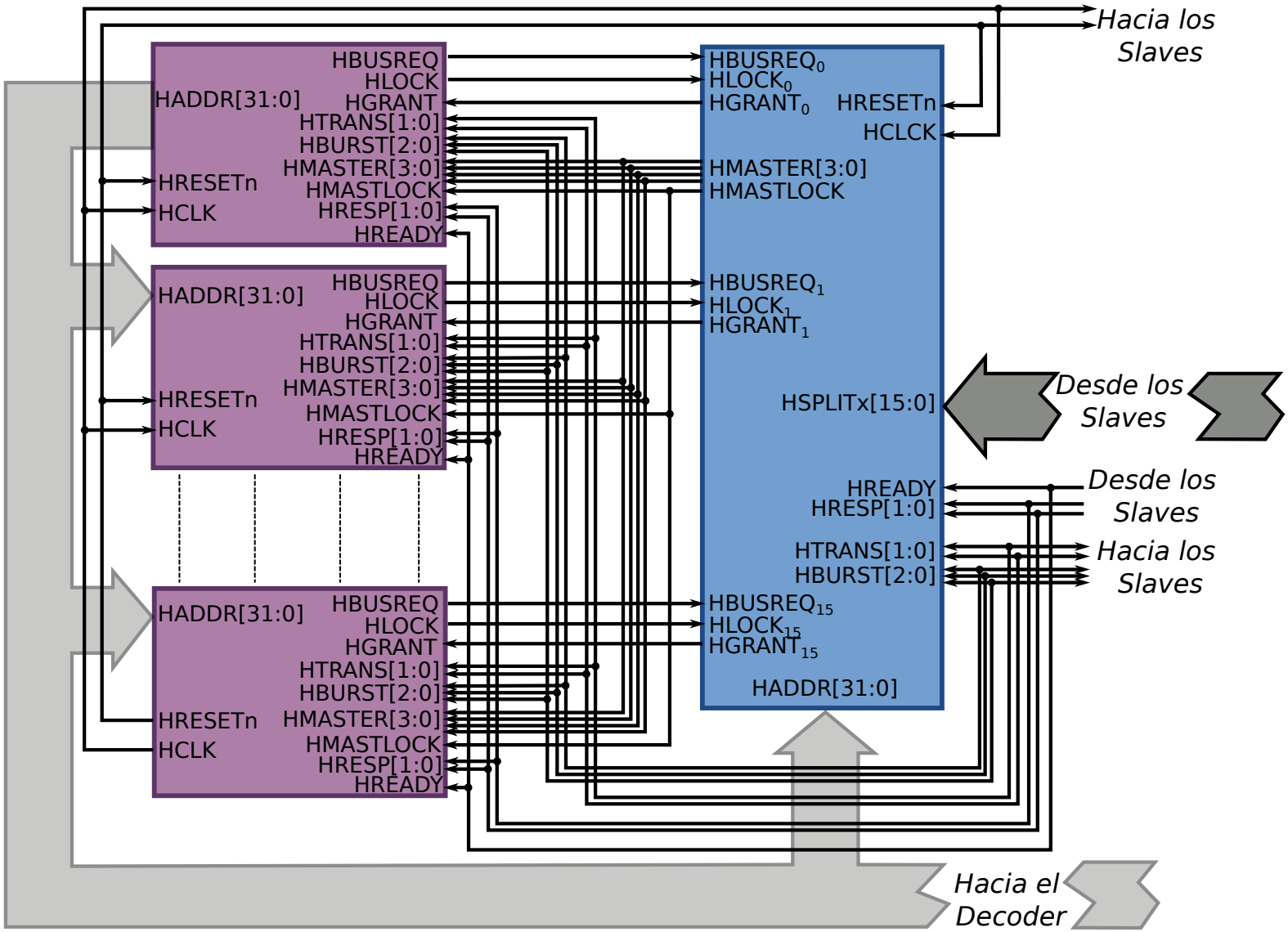
Arbitraje del AHB



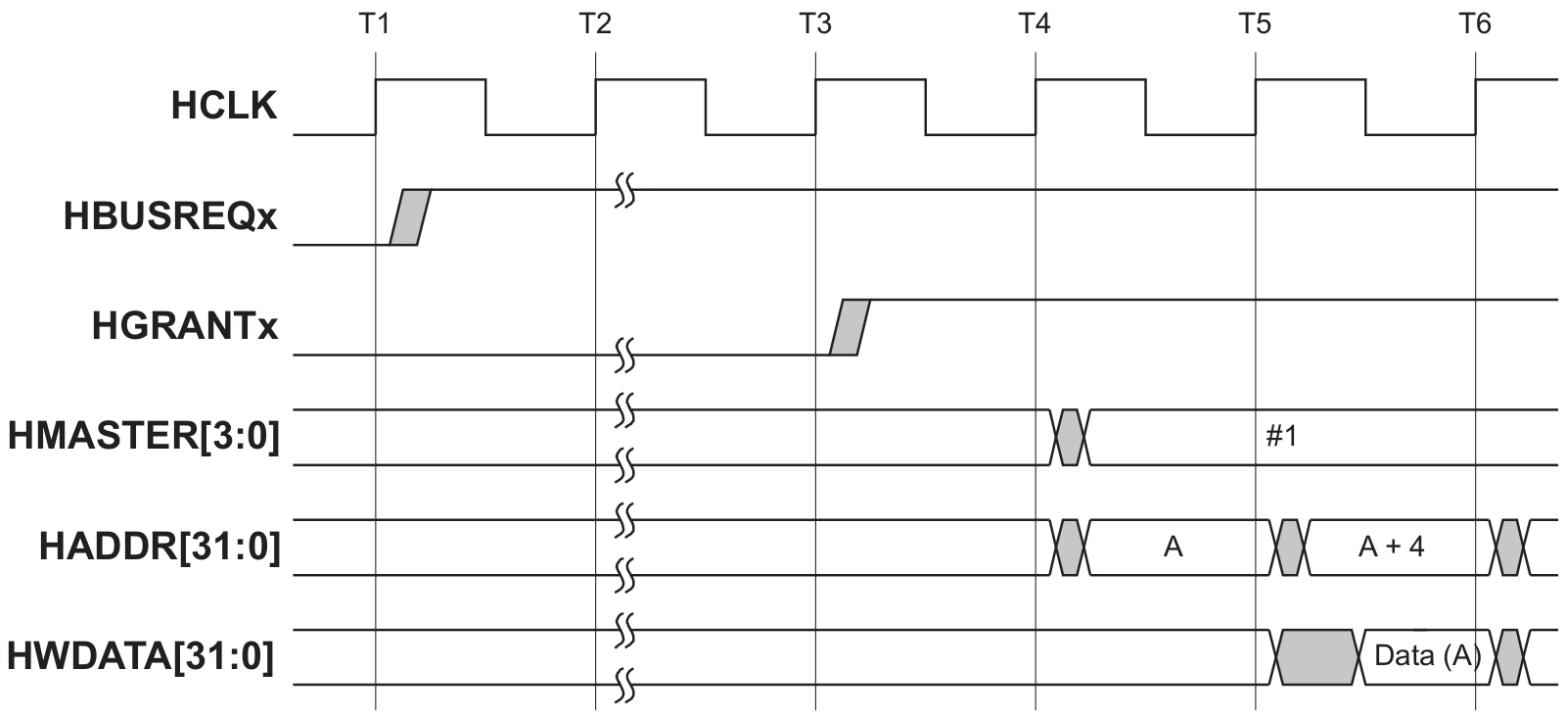
Arbitraje del AHB



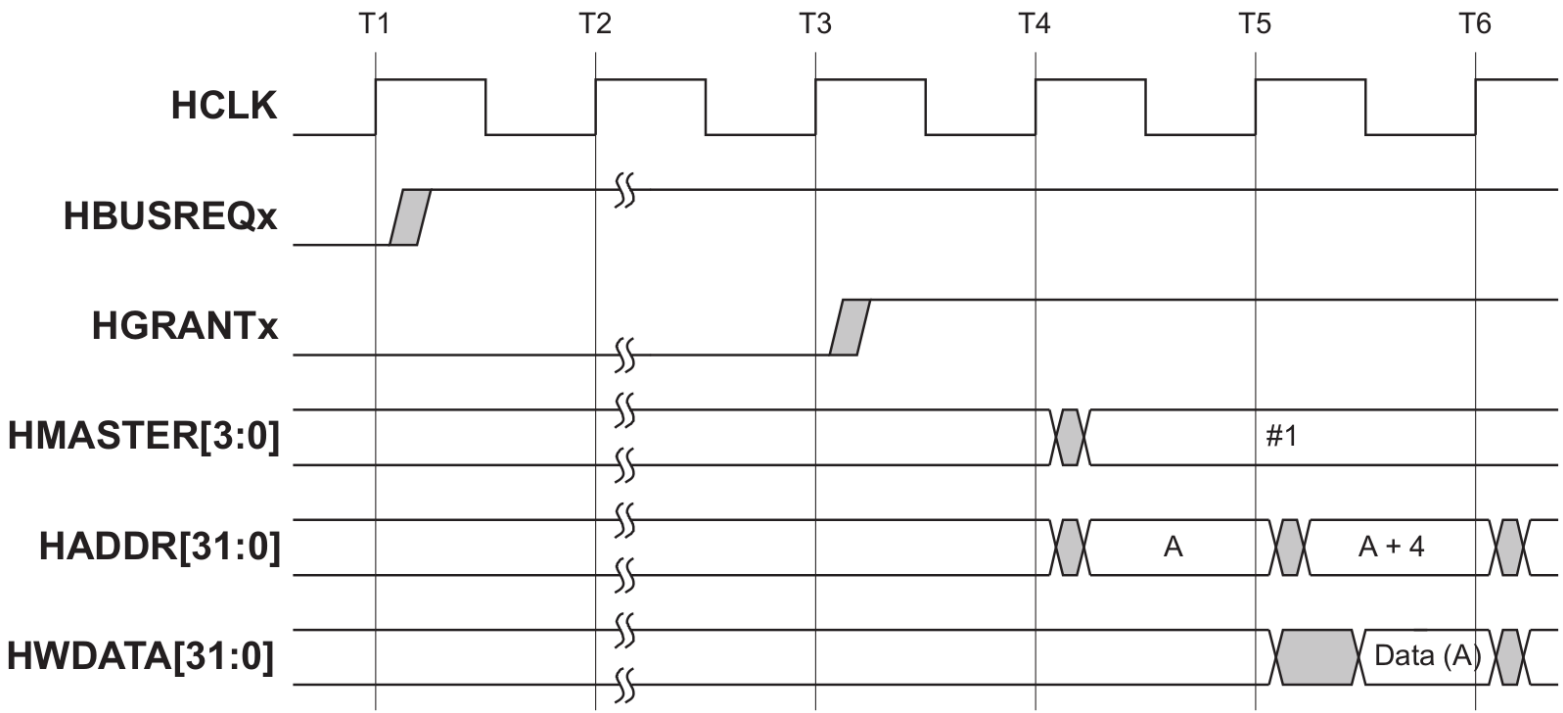
Arbitraje del AHB



Arbitraje del AHB: Cesión del Bus (Grant)

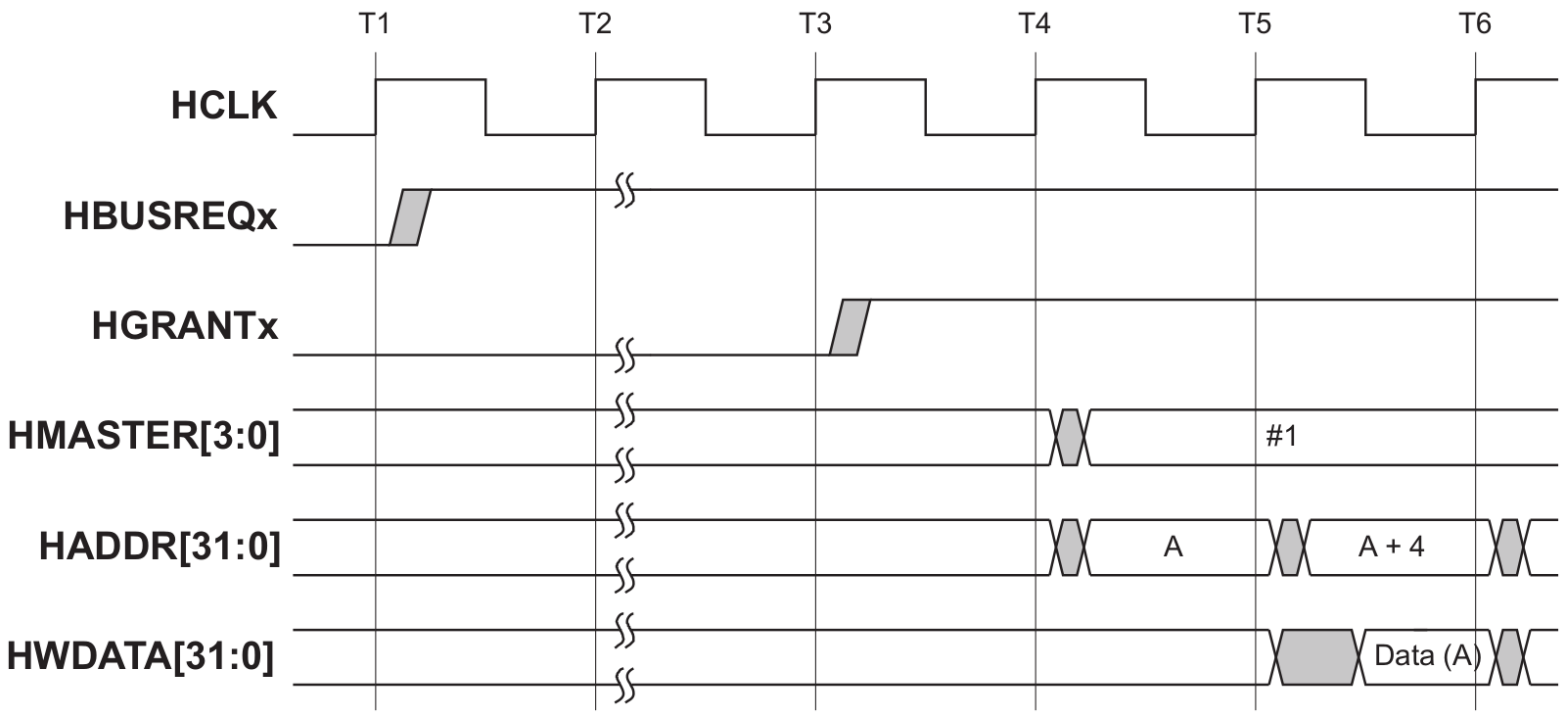


Arbitraje del AHB: Cesión del Bus (Grant)



✓Si el Master necesita Lockear el bus debe activar la señal **HLOCKn** correspondiente del Árbitro.

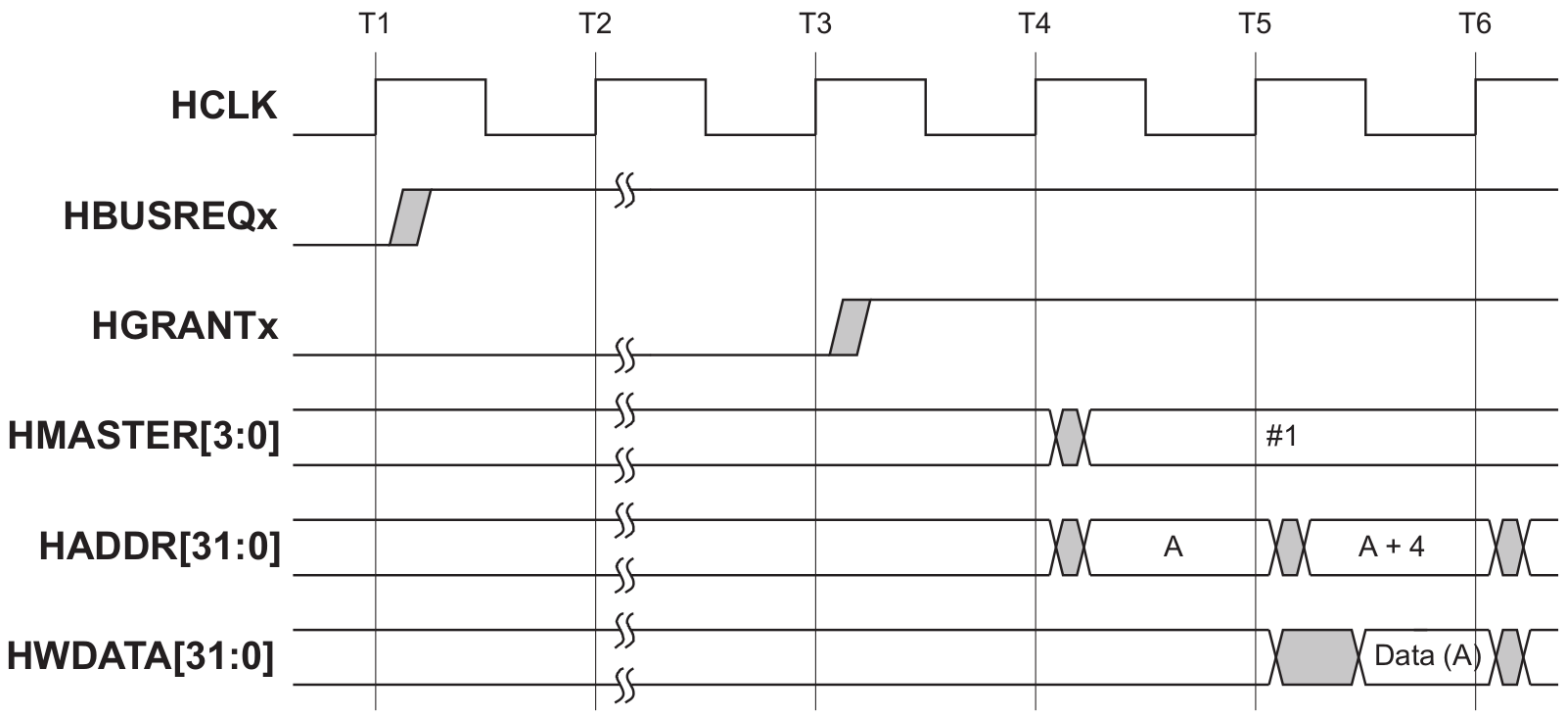
Arbitraje del AHB: Cesión del Bus (Grant)



✓Si el Master necesita Lockear el bus debe activar la señal **HLOCKn** correspondiente del Árbitro.

✓En una transferencia Burst, el Master puede levantar **HREQUESTn**.El Árbitro monitorea el burst con las líneas **HBURST[2:0]**.

Arbitraje del AHB: Cesión del Bus (Grant)

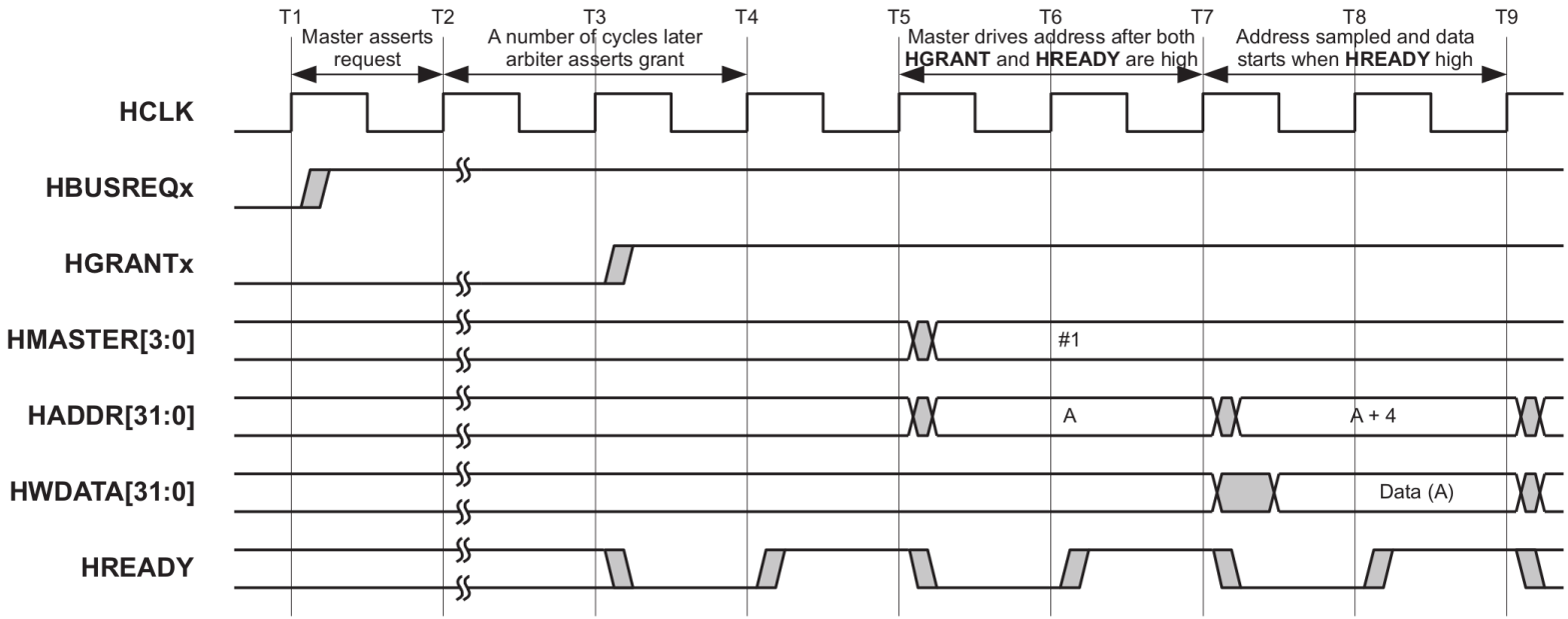


✓Si el Master necesita Lockear el bus debe activar la señal **HLOCKn** correspondiente del Árbitro.

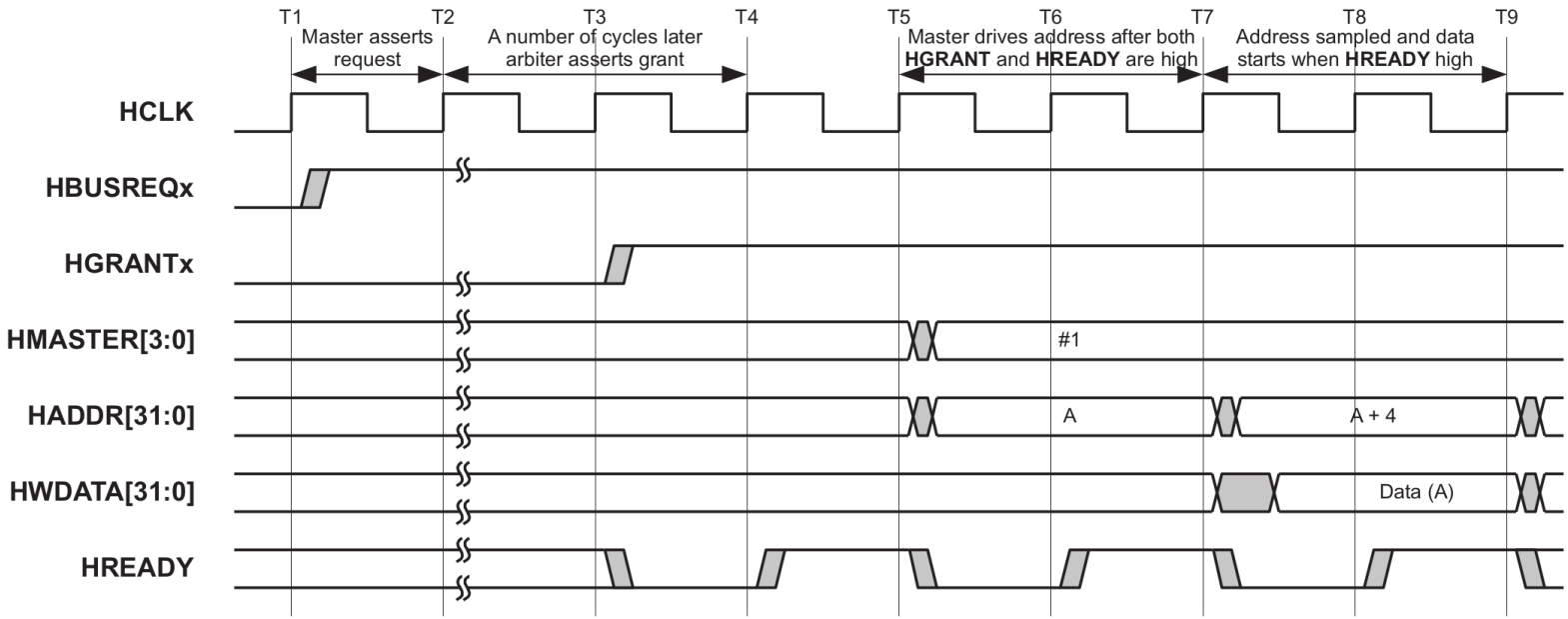
✓En una transferencia Burst, el Master puede levantar **HREQUESTn**.El Árbitro monitorea el burst con las líneas **HBURST[2:0]**.

✓Por cuestiones de prioridad el Árbitro puede interrumpir una transferencia burst, para asignar el bus a un Master de mayor prioridad.

Arbitraje del AHB: Cesión del Bus con WS

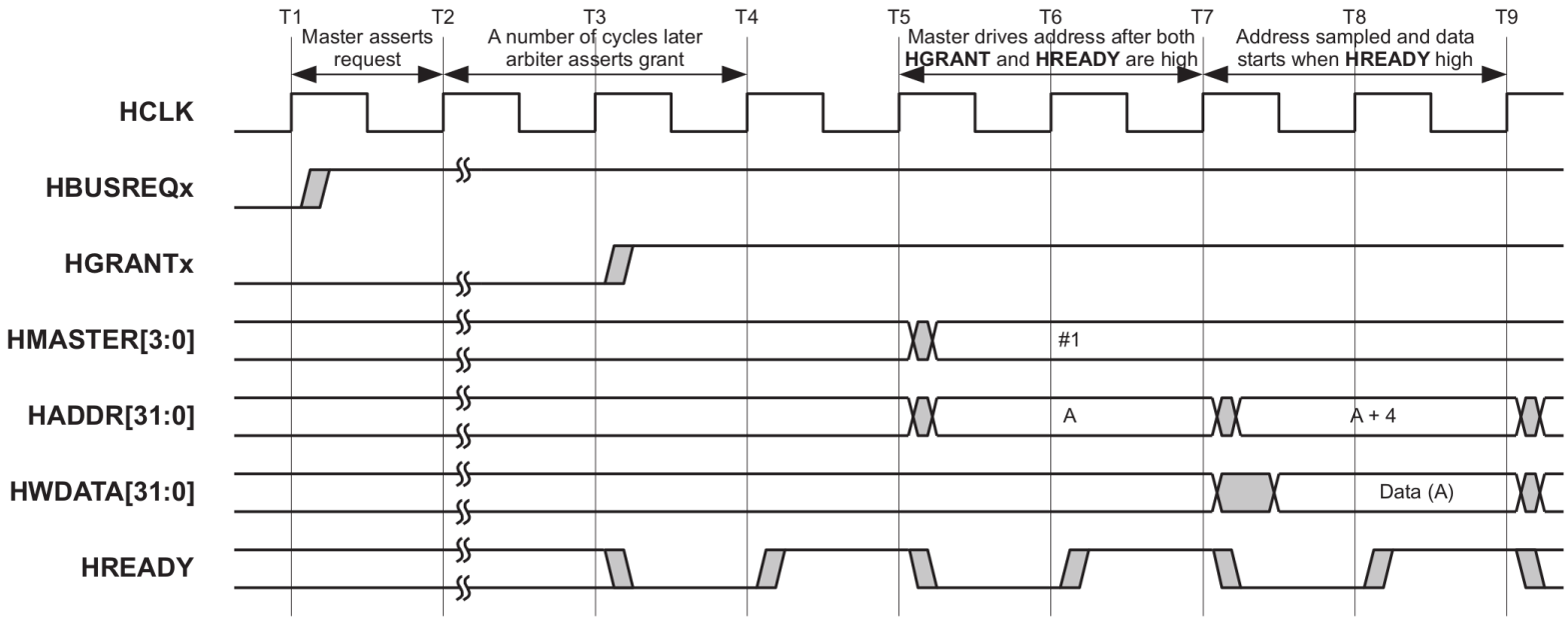


Arbitraje del AHB: Cesión del Bus con WS



✓El Árbitro no va a asignar el bus a un Master hasta que no se complete la transferencia en curso.

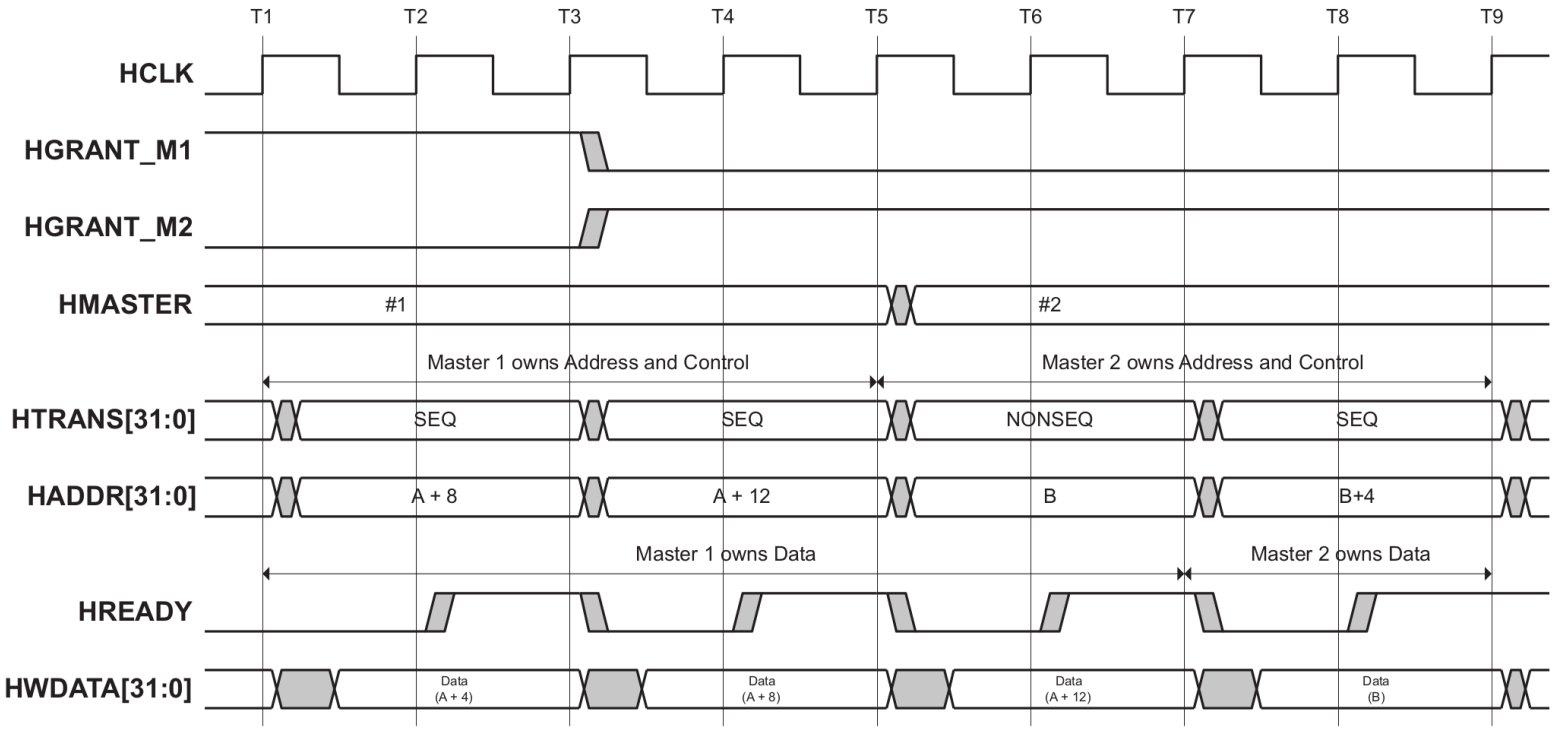
Arbitraje del AHB: Cesión del Bus con WS



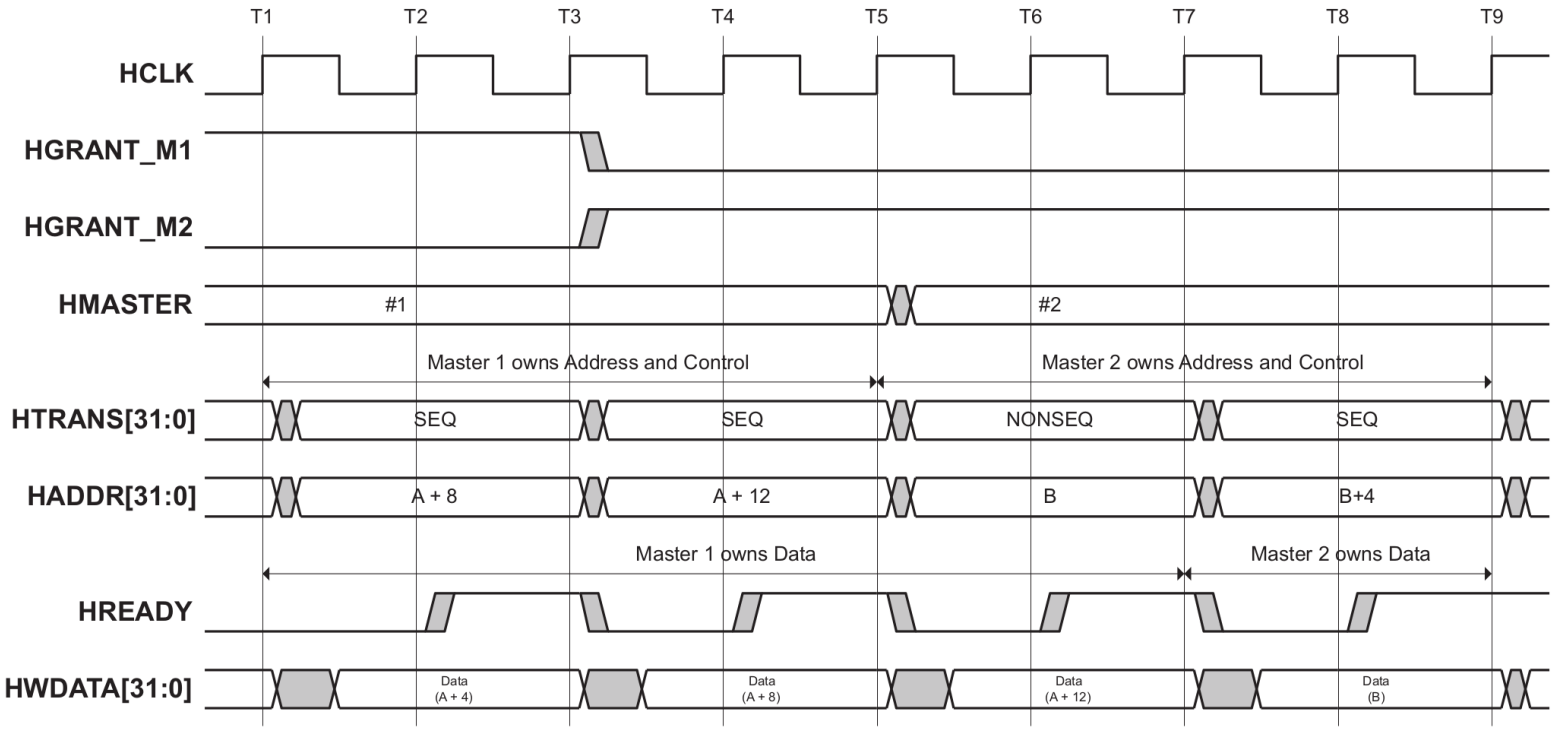
✓El Árbitro no va a asignar el bus a un Master hasta que no se complete la transferencia en curso.

✓Por lo tanto la condición además de **`HGRANTn`** activa y **`HREADY`** activa.

Arbitraje del AHB: Hand Over

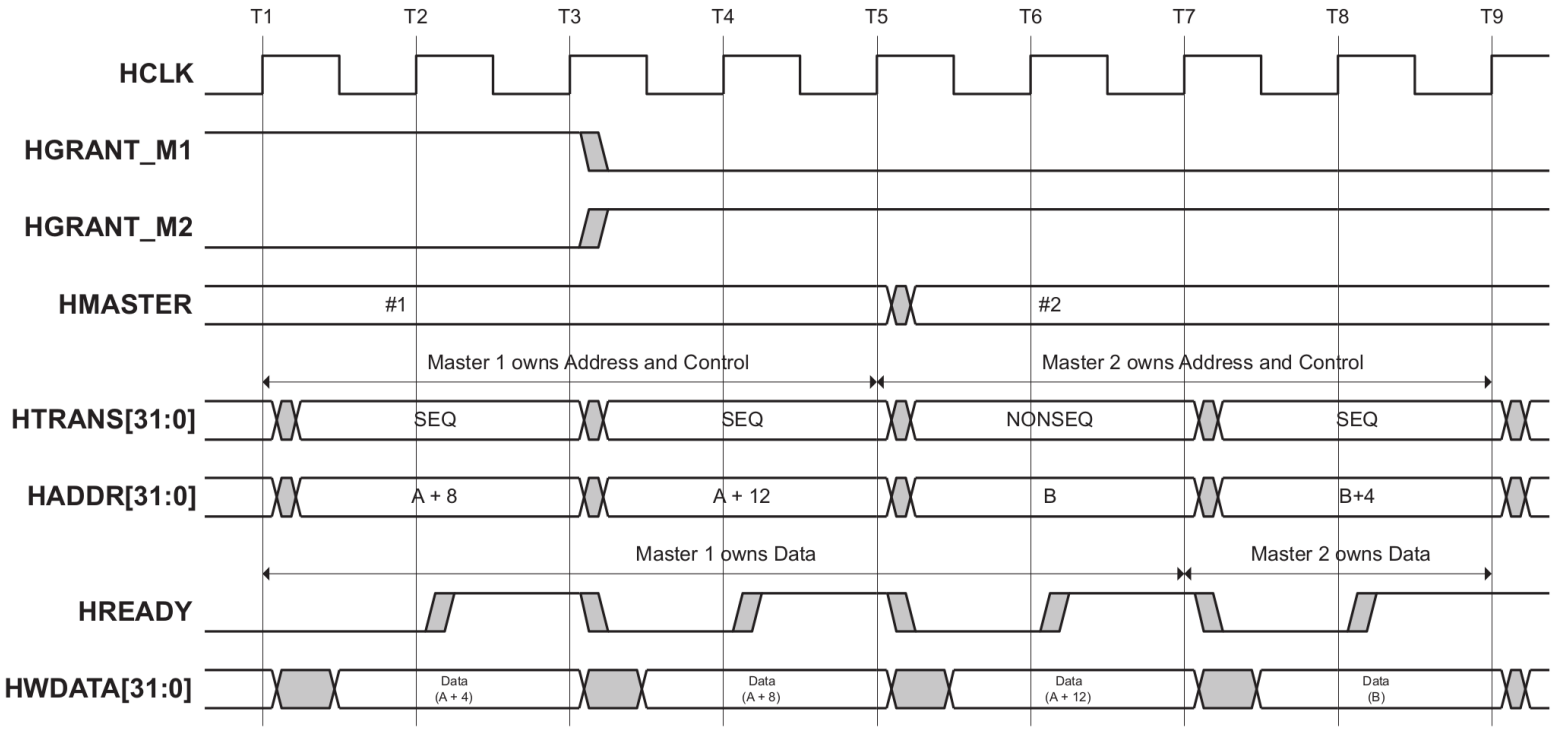


Arbitraje del AHB: Hand Over



✓Un Master con **HGRANTx** activa (alta), dispone del bus.

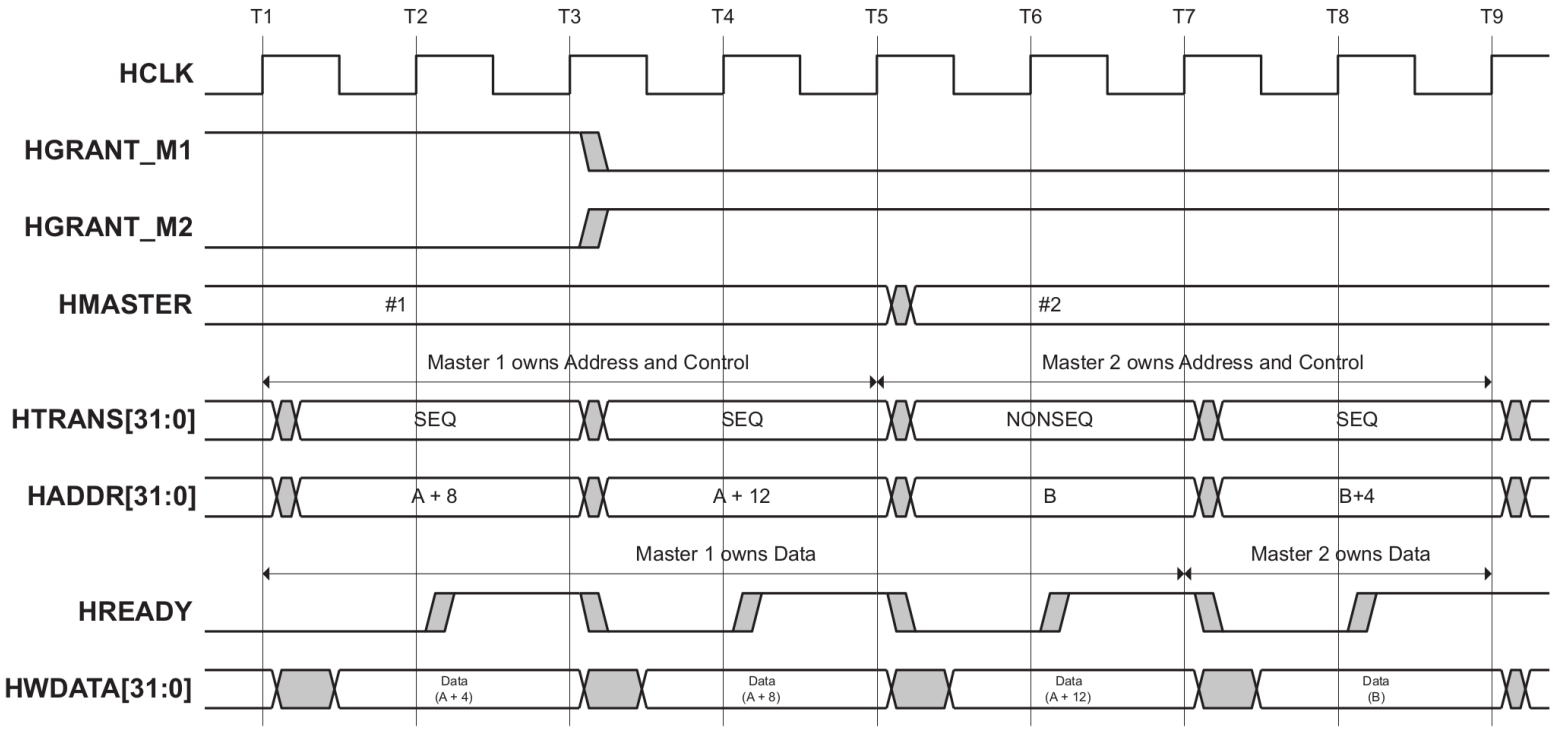
Arbitraje del AHB: Hand Over



✓Un Master con **`HGRANTx`** activa (alta), dispone del bus.

✓El Árbitro reasigna el bus alternando las líneas **`HGRANTn`** de dos Masters.

Arbitraje del AHB: Hand Over

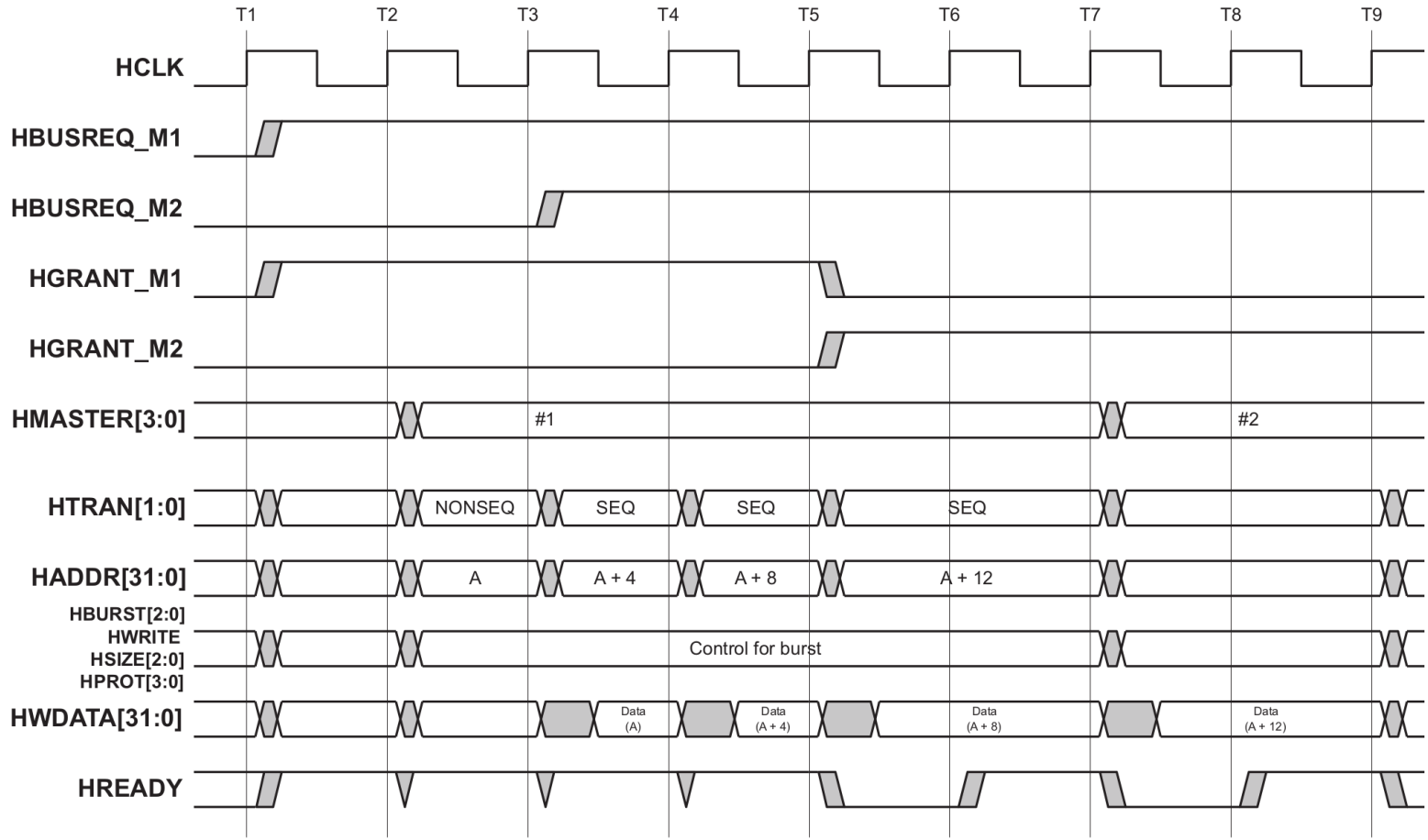


✓Un Master con **HGRANTx** activa (alta), dispone del bus.

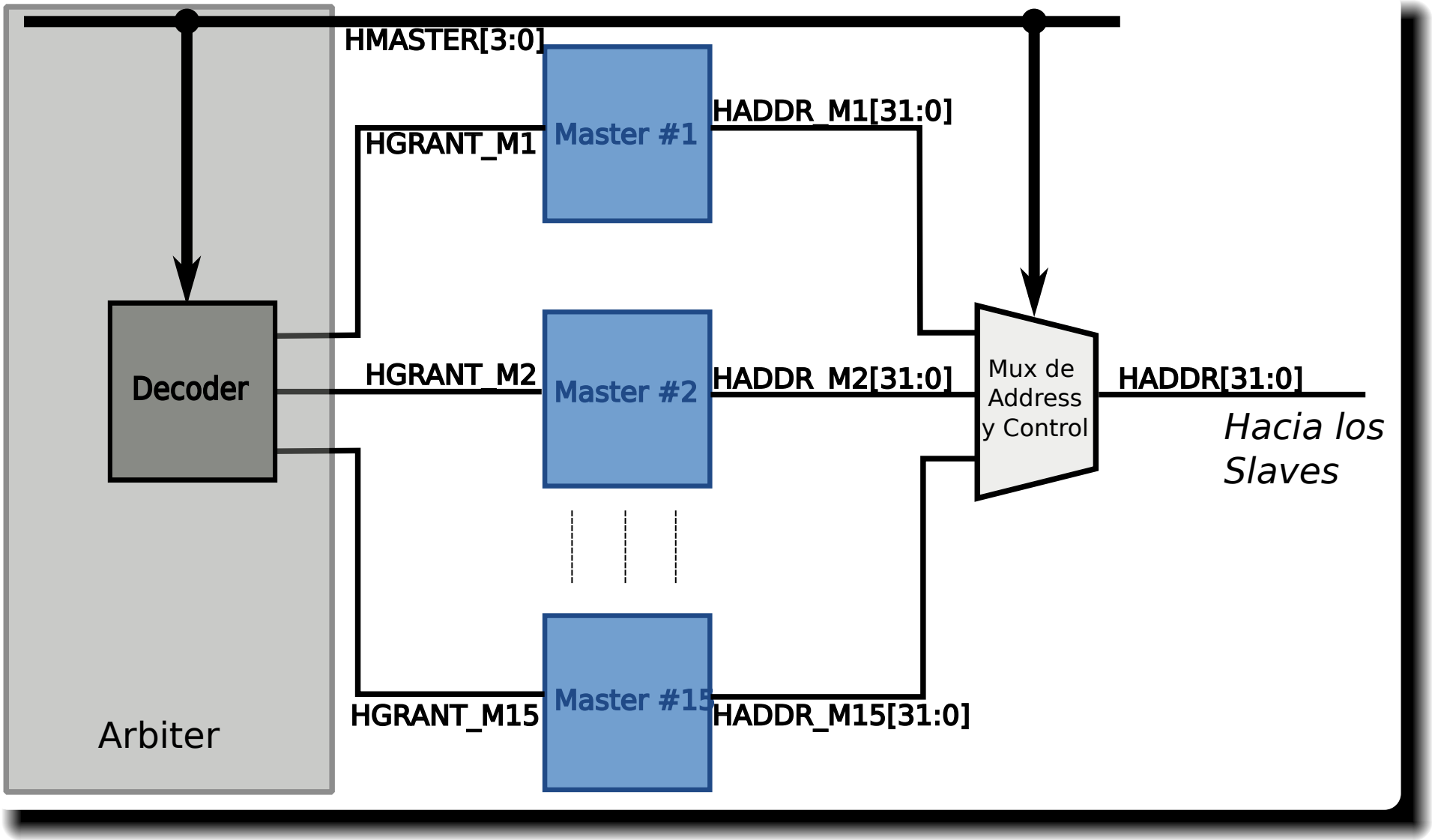
✓El Árbitro reasigna el bus alternando las líneas **HGRANTn** de dos Masters.

✓El Master con **HGRANTn** activa no inicia su transferencia sino hasta que la línea **HREADY** manejada por el Slave que está completando la transferencia en curo con el otro Master, esté activa

Arbitraje del AHB: Hand Over durante un burst



Arbitraje del AHB: Manejo de HGRANTn



Transferencias SPLIT

Transferencias SPLIT

- El Master inicia una transferencia como ya se ha visto.

Transferencias SPLIT

- El Master inicia una transferencia como ya se ha visto.
- Si el Slave puede proveer sin Wait States el dato, lo hace.

Transferencias SPLIT

- El Master inicia una transferencia como ya se ha visto.
- Si el Slave puede proveer sin Wait States el dato, lo hace.
- Si el Slave estima que demorará una cantidad de ciclos de bus excesivos devuelve **`SPLIT`** por **`HRESP[1:0]`**.

Transferencias SPLIT

- El Master inicia una transferencia como ya se ha visto.
- Si el Slave puede proveer sin Wait States el dato, lo hace.
- Si el Slave estima que demorará una cantidad de ciclos de bus excesivos devuelve **`SPLIT`** por **`HRESP[1:0]`**.
- El Arbitro por cada transferencia hace boadcast por **`HMASTER[3:0]`** del Master en poder del Bus.

Transferencias SPLIT

- El Master inicia una transferencia como ya se ha visto.
- Si el Slave puede proveer sin Wait States el dato, lo hace.
- Si el Slave estima que demorará una cantidad de ciclos de bus excesivos devuelve **`SPLIT`** por **`HRESP[1:0]`**.
- El Arbitro por cada transferencia hace boadcast por **`HMASTER[3:0]`** del Master en poder del Bus.
- El Árbitro busca un Master con Request pendiente y que no esté en **`SPLIT`**. Si no hay ninguno le asigna el Bus al Master Deafult.

Transferencias SPLIT

- El Master inicia una transferencia como ya se ha visto.
- Si el Slave puede proveer sin Wait States el dato, lo hace.
- Si el Slave estima que demorará una cantidad de ciclos de bus excesivos devuelve **`SPLIT`** por **`HRESP[1:0]`**.
- El Arbitro por cada transferencia hace boadcast por **`HMASTER[3:0]`** del Master en poder del Bus.
- El Árbitro busca un Master con Request pendiente y que no esté en **`SPLIT`**. Si no hay ninguno le asigna el Bus al Master Deafult.
- Cuando un Slave que suspendió transferencia con **`SPLIT`**, tiene el dato listo, escribe en **`HSPLIT[15:0]`** el código del Master al que debe reasignarse el Bus para completar la transferencia.

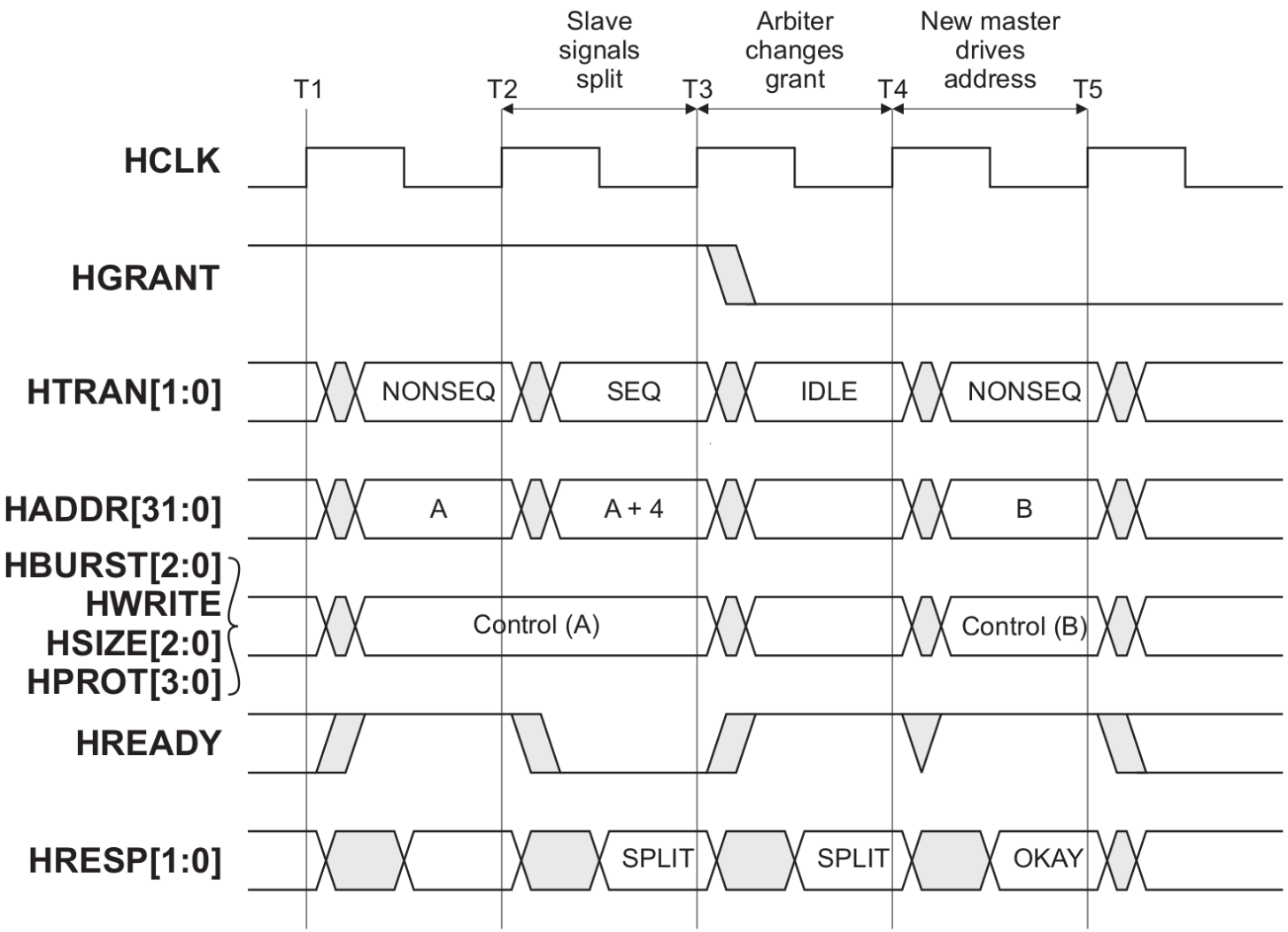
Transferencias SPLIT

- El Master inicia una transferencia como ya se ha visto.
- Si el Slave puede proveer sin Wait States el dato, lo hace.
- Si el Slave estima que demorará una cantidad de ciclos de bus excesivos devuelve **`SPLIT`** por **`HRESP[1:0]`**.
- El Arbitro por cada transferencia hace boadcast por **`HMASTER[3:0]`** del Master en poder del Bus.
- El Árbitro busca un Master con Request pendiente y que no esté en **`SPLIT`**. Si no hay ninguno le asigna el Bus al Master Deafult.
- Cuando un Slave que suspendió transferencia con **`SPLIT`**, tiene el dato listo, escribe en **`HSPLIT[15:0]`** el código del Master al que debe reasignarse el Bus para completar la transferencia.
- El Árbitro puede optar por mantener en el bus a un Master de mayor prioridad (no se garantiza hand over inmediato)

Transferencias SPLIT

- El Master inicia una transferencia como ya se ha visto.
- Si el Slave puede proveer sin Wait States el dato, lo hace.
- Si el Slave estima que demorará una cantidad de ciclos de bus excesivos devuelve **SPLIT** por **HRESP[1:0]**.
- El Arbitro por cada transferencia hace boadcast por **HMASTER[3:0]** del Master en poder del Bus.
- El Árbitro busca un Master con Request pendiente y que no esté en **SPLIT**. Si no hay ninguno le asigna el Bus al Master Deafult.
- Cuando un Slave que suspendió transferencia con **SPLIT**, tiene el dato listo, escribe en **HSPLIT[15:0]** el código del Master al que debe reasignarse el Bus para completar la transferencia.
- El Árbitro puede optar por mantener en el bus a un Master de mayor prioridad (no se garantiza hand over inmediato)
- Cuando el Master es reasignado se completa la transferencia. La transferecia finalizada se completa con **OKAY** por **HRESP[1:0]**.

Arbitraje del AHB: Manejo de HGRANTn



✓El Arbitro puede interrumpir un burst que responde a un **SPLIT** a expensas de un Master de mayor prioridad.

Temario

Protocolo AXI

Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.

Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.
- Apto para diseños que requieren al mismo tiempo gran Ancho de Banda y muy bajo retardo (latency)

Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.
- Apto para diseños que requieren al mismo tiempo gran Ancho de Banda y muy bajo retardo (latency)
- Alcanza altas frecuencias de trabajo sin agregar complejidad a los bridges

Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.
- Apto para diseños que requieren al mismo tiempo gran Ancho de Banda y muy bajo retardo (latency)
- Alcanza altas frecuencias de trabajo sin agregar complejidad a los bridges
- Adecuado para controladores de memoria de alta demora inicial.

Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.
- Apto para diseños que requieren al mismo tiempo gran Ancho de Banda y muy bajo retardo (latency)
- Alcanza altas frecuencias de trabajo sin agregar complejidad a los bridges
- Adecuado para controladores de memoria de alta demora inicial.
- Compatibilidad con ***AHB*** y ***APB***.

Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.
- Apto para diseños que requieren al mismo tiempo gran Ancho de Banda y muy bajo retardo (latency)
- Alcanza altas frecuencias de trabajo sin agregar complejidad a los bridges
- Adecuado para controladores de memoria de alta demora inicial.
- Compatibilidad con ***AHB*** y ***APB***.
- Manejo separado de las fases de Dirección, Control, y Datos.

Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.
- Apto para diseños que requieren al mismo tiempo gran Ancho de Banda y muy bajo retardo (latency)
- Alcanza altas frecuencias de trabajo sin agregar complejidad a los bridges
- Adecuado para controladores de memoria de alta demora inicial.
- Compatibilidad con ***AHB*** y ***APB***.
- Manejo separado de las fases de Dirección, Control, y Datos.
- Soporte para transferencias de datos no alienadas.

Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.
- Apto para diseños que requieren al mismo tiempo gran Ancho de Banda y muy bajo retardo (latency)
- Alcanza altas frecuencias de trabajo sin agregar complejidad a los bridges
- Adecuado para controladores de memoria de alta demora inicial.
- Compatibilidad con ***AHB*** y ***APB***.
- Manejo separado de las fases de Dirección, Control, y Datos.
- Soporte para transferencias de datos no alienadas.
- Transferencias Burst con solo enviar la dirección de inicio.

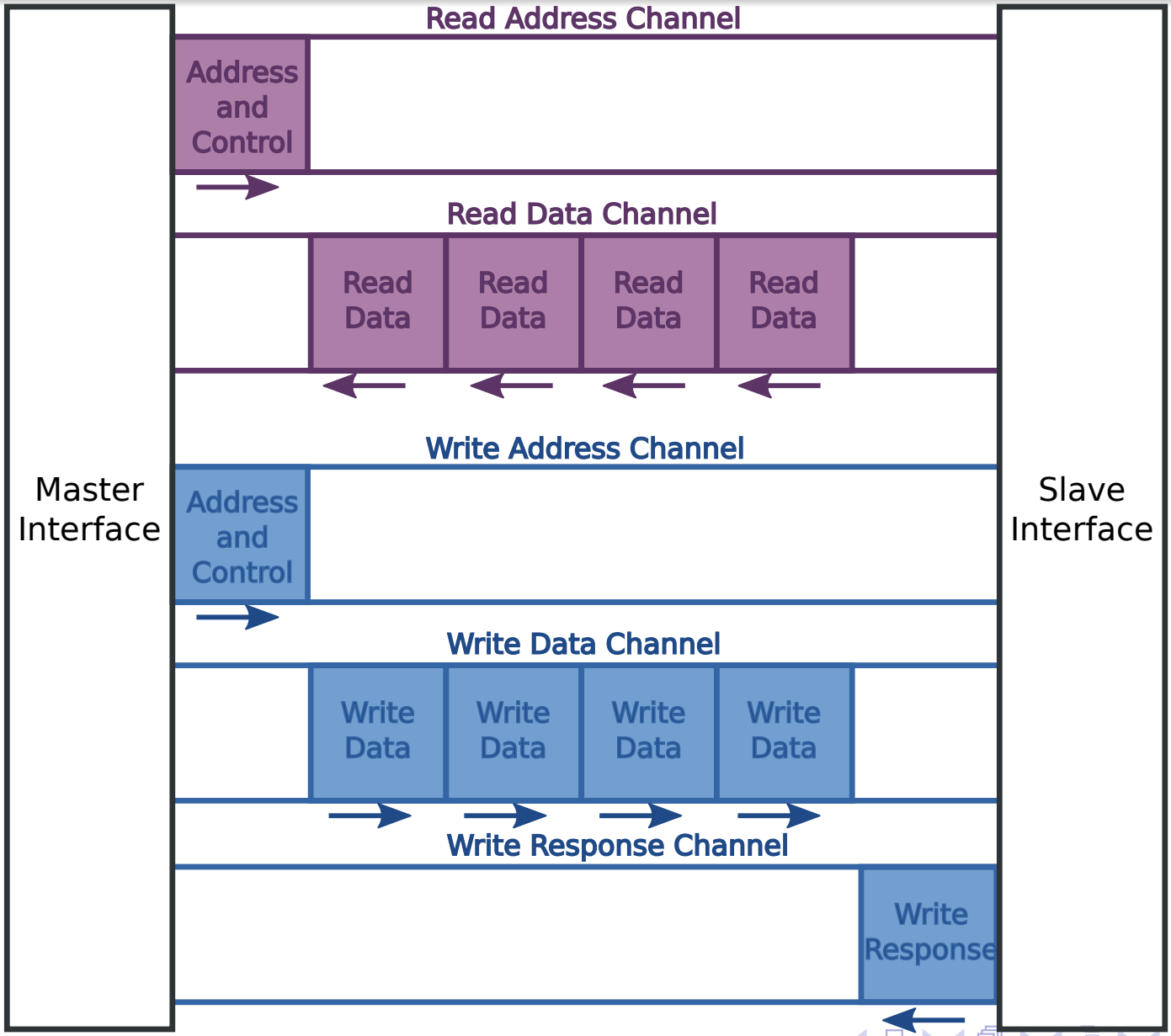
Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.
- Apto para diseños que requieren al mismo tiempo gran Ancho de Banda y muy bajo retardo (latency)
- Alcanza altas frecuencias de trabajo sin agregar complejidad a los bridges
- Adecuado para controladores de memoria de alta demora inicial.
- Compatibilidad con ***AHB*** y ***APB***.
- Manejo separado de las fases de Dirección, Control, y Datos.
- Soporte para transferencias de datos no alienadas.
- Transferencias Burst con solo enviar la dirección de inicio.
- Buses de Datos separados para lectura y escritura para minimizar el costo de transferencias DMA.

Protocolo AXI

- Introducido en la versión 3 de AMBA, pensada para la línea CORTEX-A.
- Apto para diseños que requieren al mismo tiempo gran Ancho de Banda y muy bajo retardo (latency)
- Alcanza altas frecuencias de trabajo sin agregar complejidad a los bridges
- Adecuado para controladores de memoria de alta demora inicial.
- Compatibilidad con ***AHB*** y ***APB***.
- Manejo separado de las fases de Dirección, Control, y Datos.
- Soporte para transferencias de datos no alienadas.
- Transferencias Burst con solo enviar la dirección de inicio.
- Buses de Datos separados para lectura y escritura para minimizar el costo de transferencias DMA.
- Capacidad de completar transacciones fuera de orden.

Arquitectura AXI



Arquitectura AXI

AXI está basado en transferencias Burst.