Parallel Communications AHB

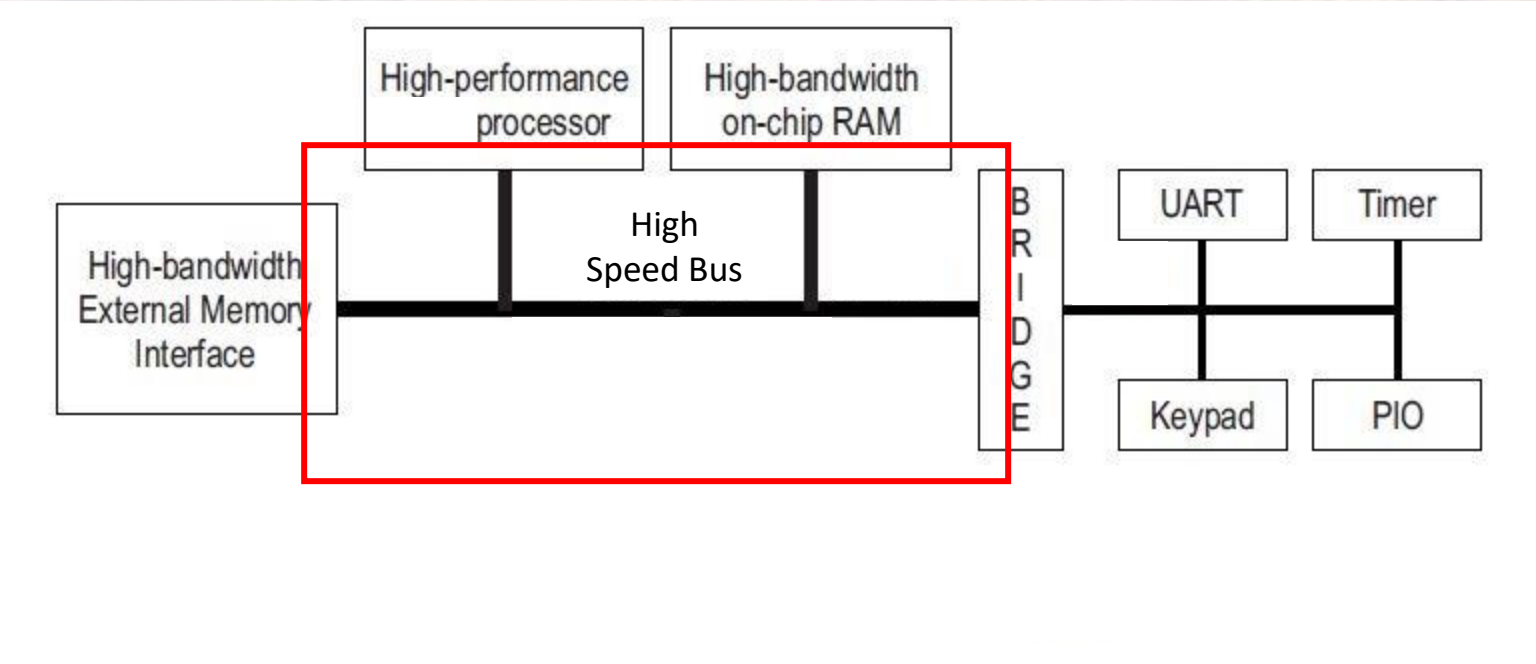
Last updated 2/15/24

Parallel Communications - AHB

- ARM
  - AMBA – Advanced Microprocessor Bus Architecture
    - CHI - Coherent Hub Interface - The highest performance, used in networks and servers
    - ACE - AXI Coherency Extensions - Used in big.LITTLE™ systems for smartphones, tablets, etc.
    - AXI - Advanced eXtensible Interface - The most widespread AMBA interface. Connectivity up to 100's of Masters and Slaves in complex SoC's
    - AHB - Advanced High-Performance Bus - The main system bus in microcontroller usage
    - APB - Advanced Peripheral Bus - Minimal gate count for peripherals
    - ATB - Advanced Trace Bus - For moving trace data around the chip

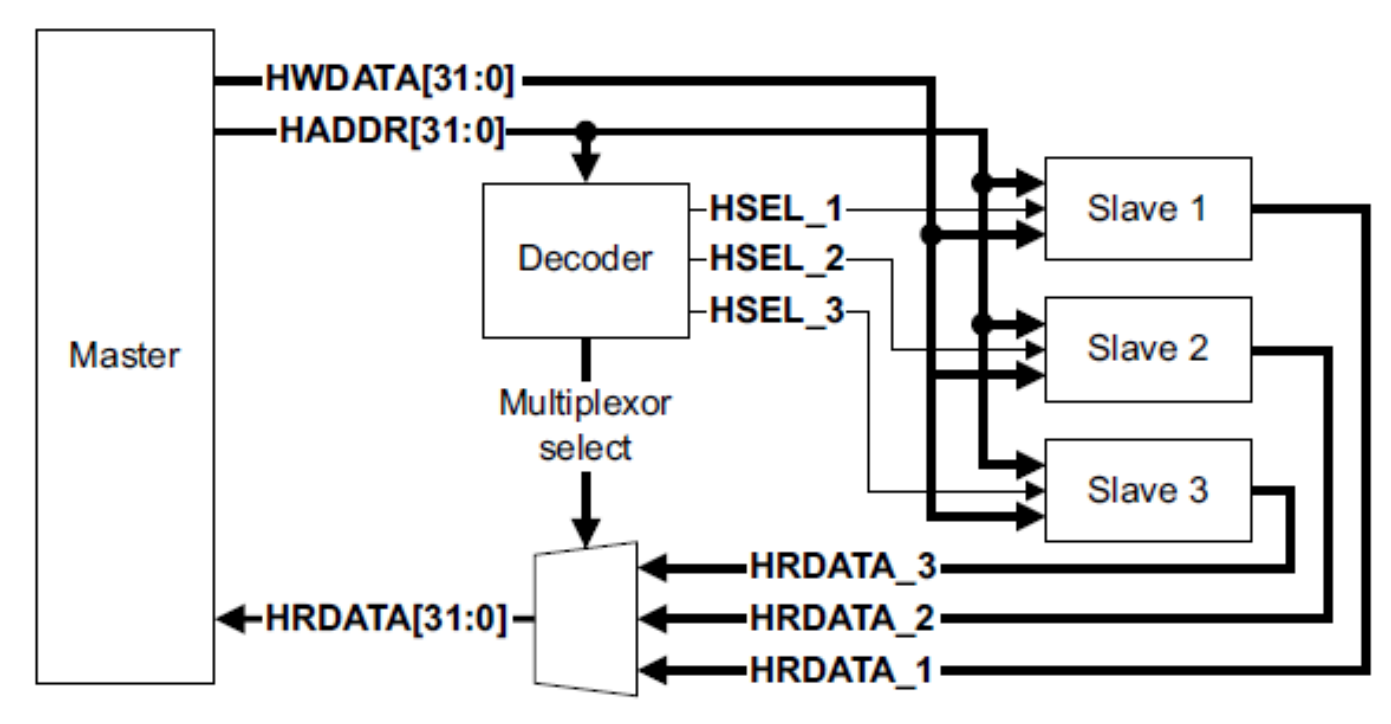
Parallel Communications - AHB

- AMBA – AHB-Lite
  - Lite → Single Master System Bus



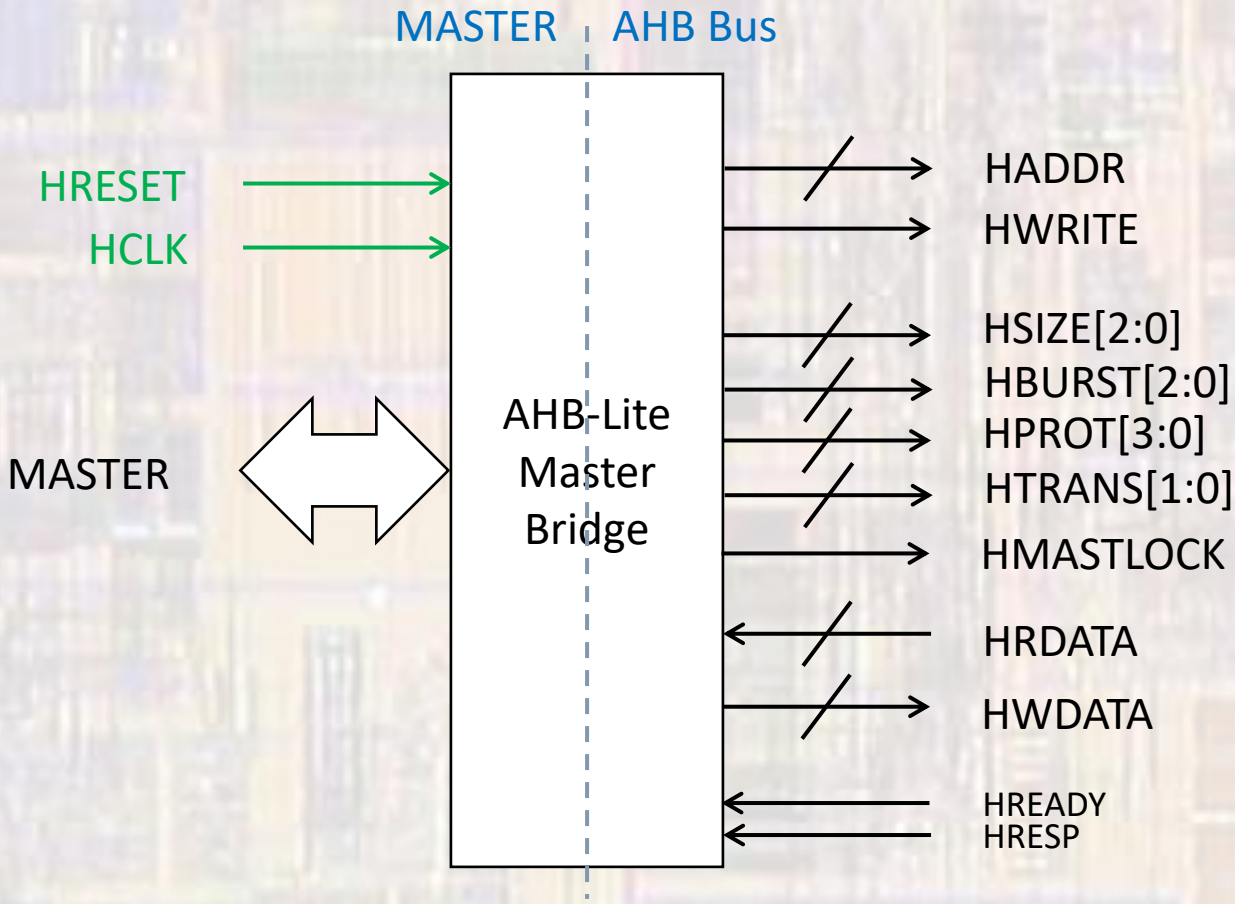
Parallel Communications - AHB

- AHB-Lite - Overview
  - Single Master System Bus



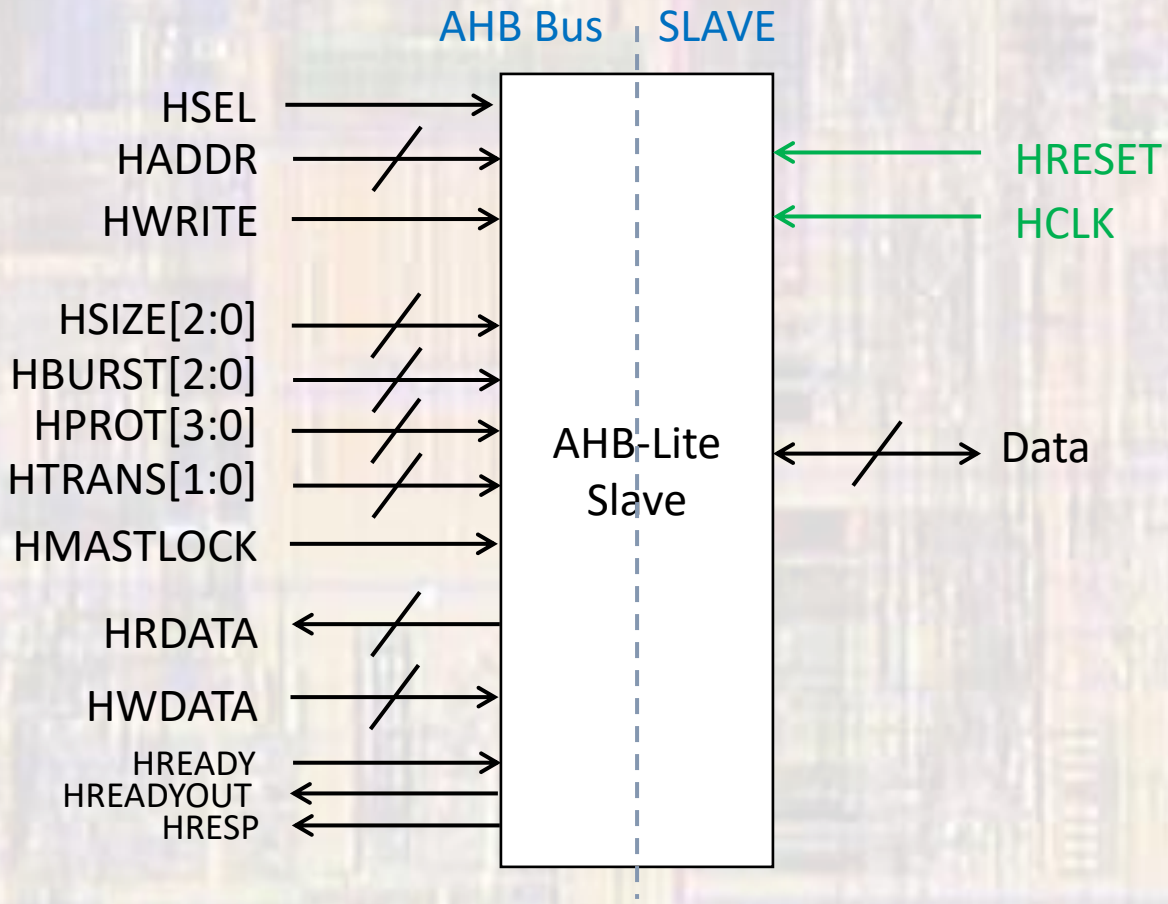
Parallel Communications - AHB

- AHB-Lite – Master Bridge



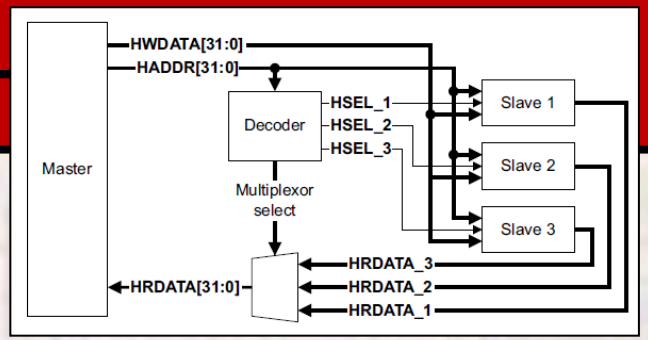
Parallel Communications - AHB

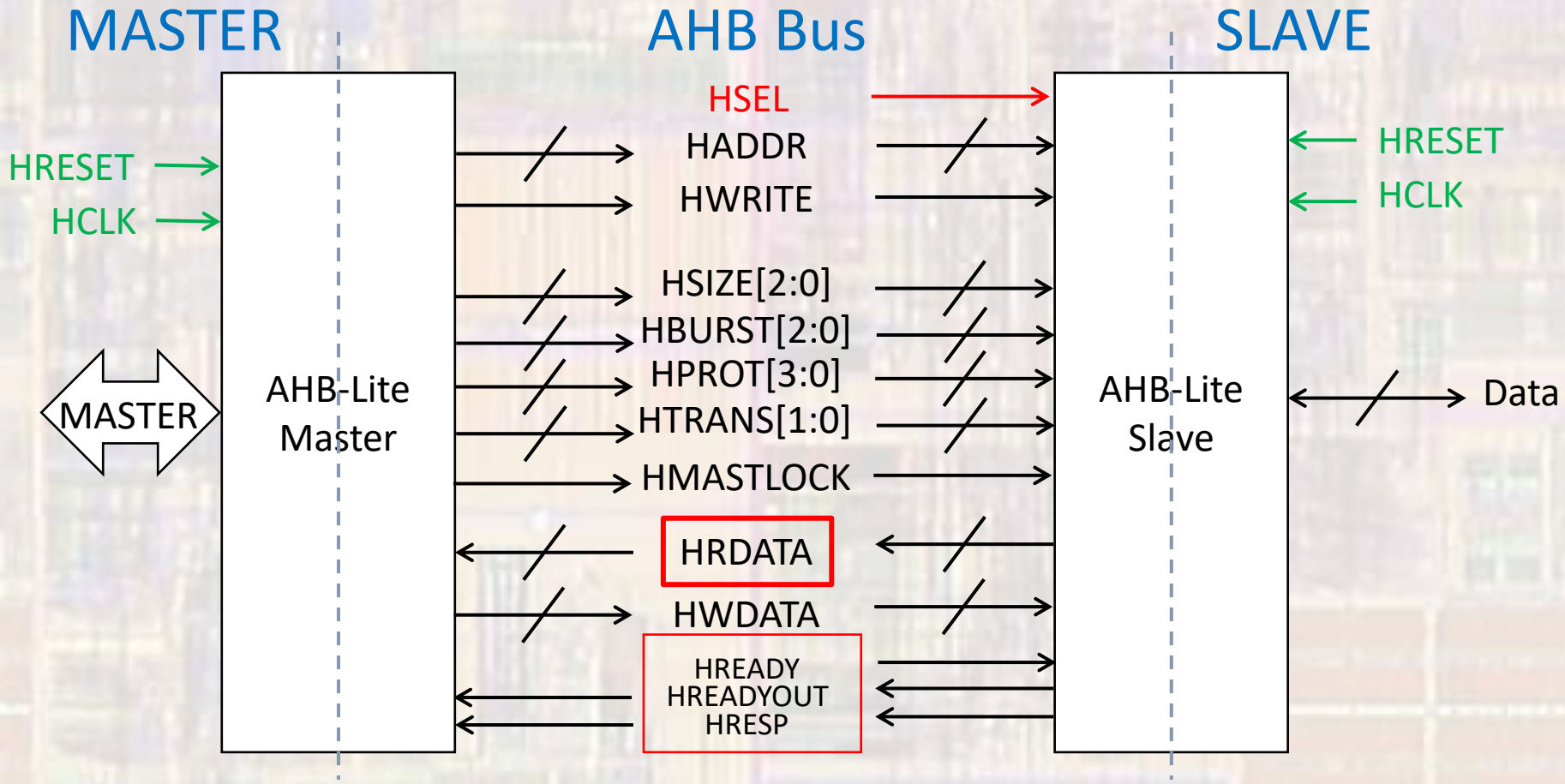
- AHB-Lite Slave Interface
  - Each component tied to the AHB



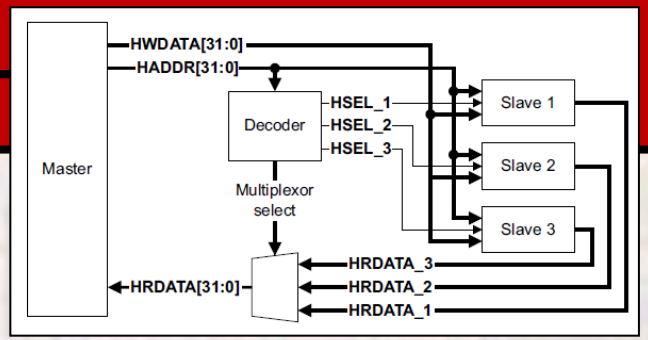
Parallel Communications -

- AHB-Lite - Connections





Parallel Communications -



- AHB-Lite Signals

  - HCLK – Derived system clock – gated, frequency
  - HRESET – Derived system reset

  - HADDR – Memory mapped address –32 bits
  - HSEL – Slave select – via decoder

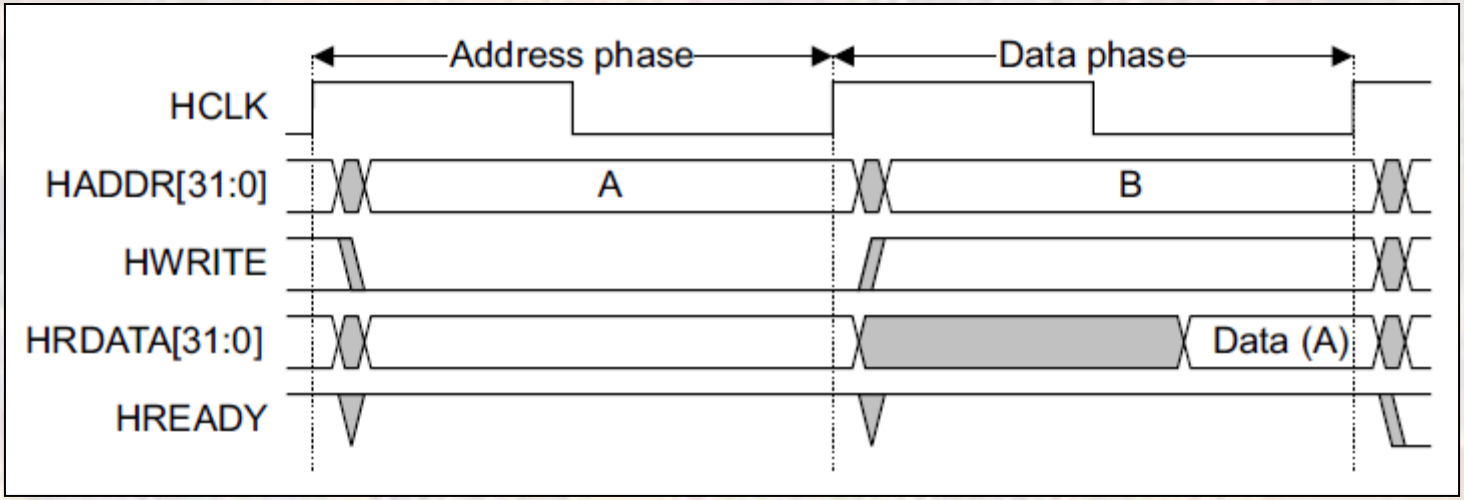
  - HWDATA – Write Data – ? bits – to slave
  - HRDATA – Read Data – ? bits – from slave via multiplexor
  - HWRITE – Read/Write

  - HBURST – Burst mode select (3 bits)
  - HMASTLOCK – Locked Transaction
  - HPROT – Protected access control (4 bits)
  - HSIZE – Size of data transfer (3 bits)
  - HTRANS – Transfer Type (2 bits)

  - HREADY – Transfer is complete – to slaves
  - HREADYOUT – Transfer is complete – from slave via multiplexor
  - HRESP – Transfer error – from slave via multiplexor

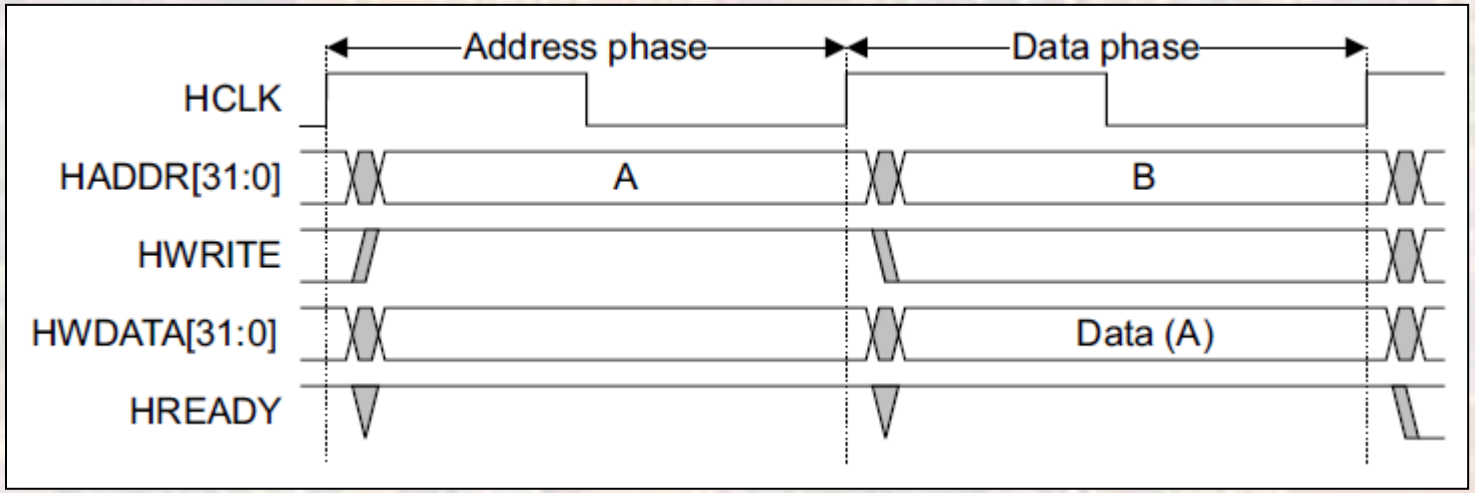
Parallel Communications - AHB

- AHB-Lite Basic Data Transfer
  - Pipelined
  - 2 clock cycles
  - Read



Parallel Communications - AHB

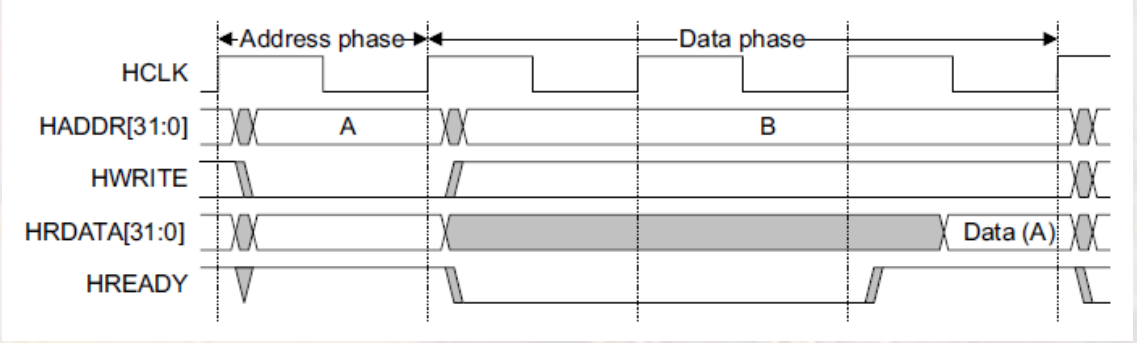
- AHB-Lite Basic Data Transfer
  - Pipelined
  - 2 clock cycles
  - Write



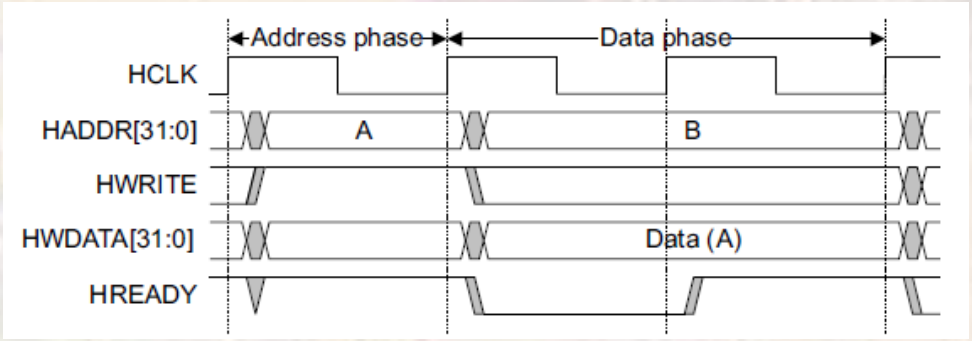
Parallel Communications - AHB

- AHB-Lite Basic Data Transfer
  - 2 clock cycles
  - With wait states

READ



WRITE



Parallel Communications - AHB

- AHB-Lite Transfer Size
  - AHB supports multiple sizes of data transfer
  - Allows transfer of LESS than full bus width data
    - Eg. A 32 bit data bus would only allow
      [000], [001], [010] states
  - 1 transfer is called a Beat

HSIZE[2]	HSIZE[1]	HSIZE[0]	Size (bits)	Description
0	0	0	8	Byte
0	0	1	16	Halfword
0	1	0	32	Word
0	1	1	64	Doubleword
1	0	0	128	4-word line
1	0	1	256	8-word line
1	1	0	512	-
1	1	1	1024	-

Parallel Communications - AHB

- AHB-Lite Burst Operation
  - AHB supports multiple beat transfers – called a burst
  - Bursts of 1,4,8,16 or an undefined # of beats are allowed
  - Incrementing burst
    - Advance the address by HSIZE for each beat
  - Wrapping burst

Advances the address by HSIZE for each beat
UNTIL
The address reaches an address boundary (defined as MOD(HSIZE * Burst length))
THEN
The address wraps back around to the beginning of the boundary and continues

All addresses must be aligned with beat boundaries

Parallel Communications - AHB

- AHB-Lite Burst Operation
  - Examples

HBURST[2:0]	Type	Description
b000	SINGLE	Single burst
b001	INCR	Incrementing burst of undefined length
b010	WRAP4	4-beat wrapping burst
b011	INCR4	4-beat incrementing burst
b100	WRAP8	8-beat wrapping burst
b101	INCR8	8-beat incrementing burst
b110	WRAP16	16-beat wrapping burst
b111	INCR16	16-beat incrementing burst

- An 8-beat incrementing burst of half word (2-byte) accesses with a start address of 0x34 then consists of eight transfers to addresses 0x34, 0x36, 0x38, 0x3A, 0x3C, 0x3E , 0x40 and 0x42

- A four-beat wrapping burst of word (4-byte) accesses wraps at 16-byte boundaries. Therefore, if the start address of the transfer is 0x34, then it consists of four transfers to addresses 0x34, 0x38, 0x3C, and 0x30

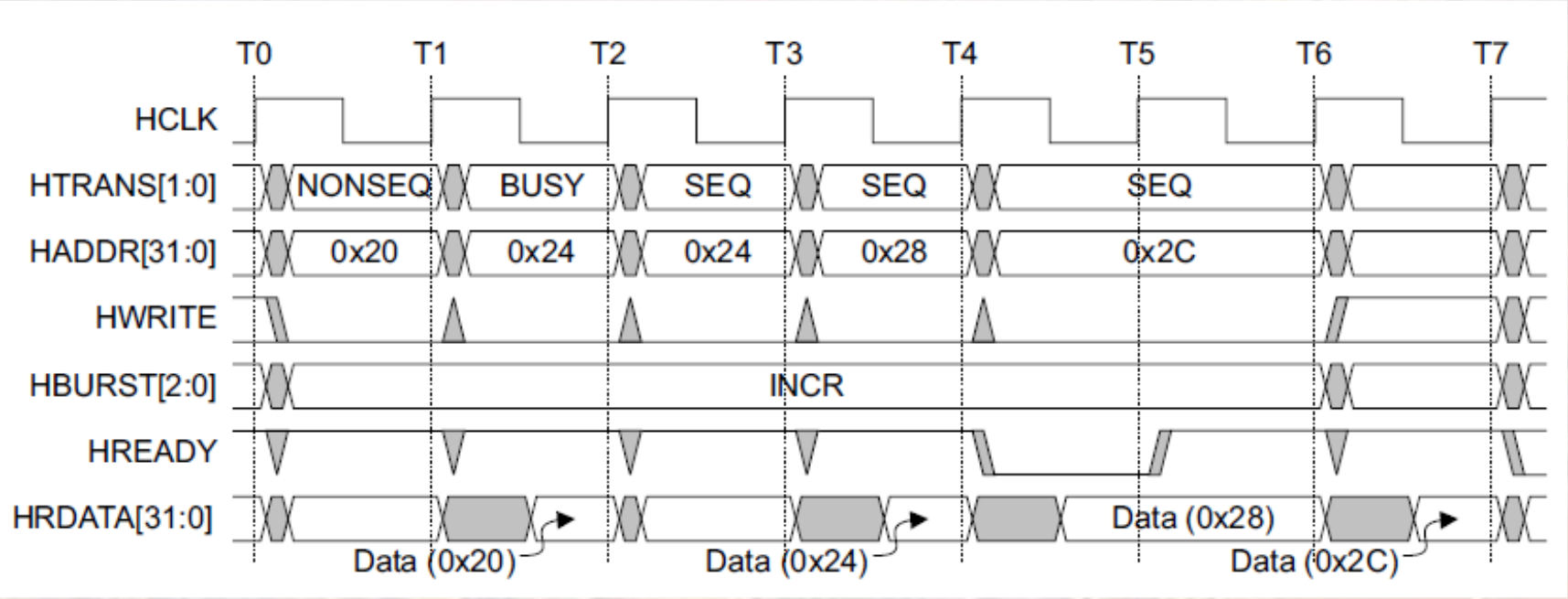
Parallel Communications - AHB

- AHB-Lite Transfer Types
  - IDLE
    - No transfer is required
  - BUSY
    - Master requests a wait
  - NONSEQ
    - Single transfer or first transfer of a burst
  - SEQ
    - 2nd and subsequent transfers of a burst

Type	HTRANS[1:0]
IDLE	b 0 0
BUSY	b 0 1
NONSEQ	b 1 0
SEQ	b 1 1

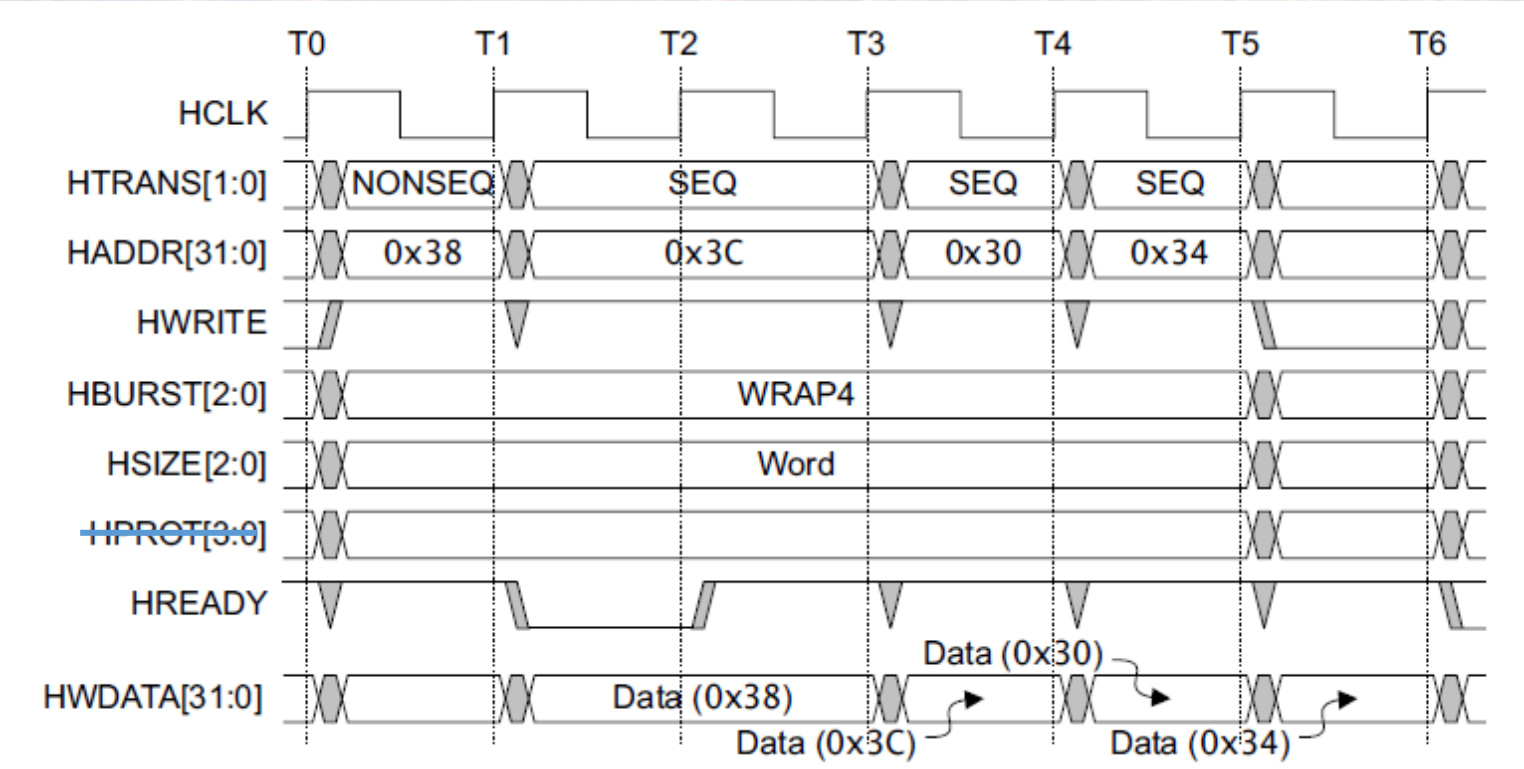
Parallel Communications - AHB

- AHB-Lite Transfer Types
  - Example



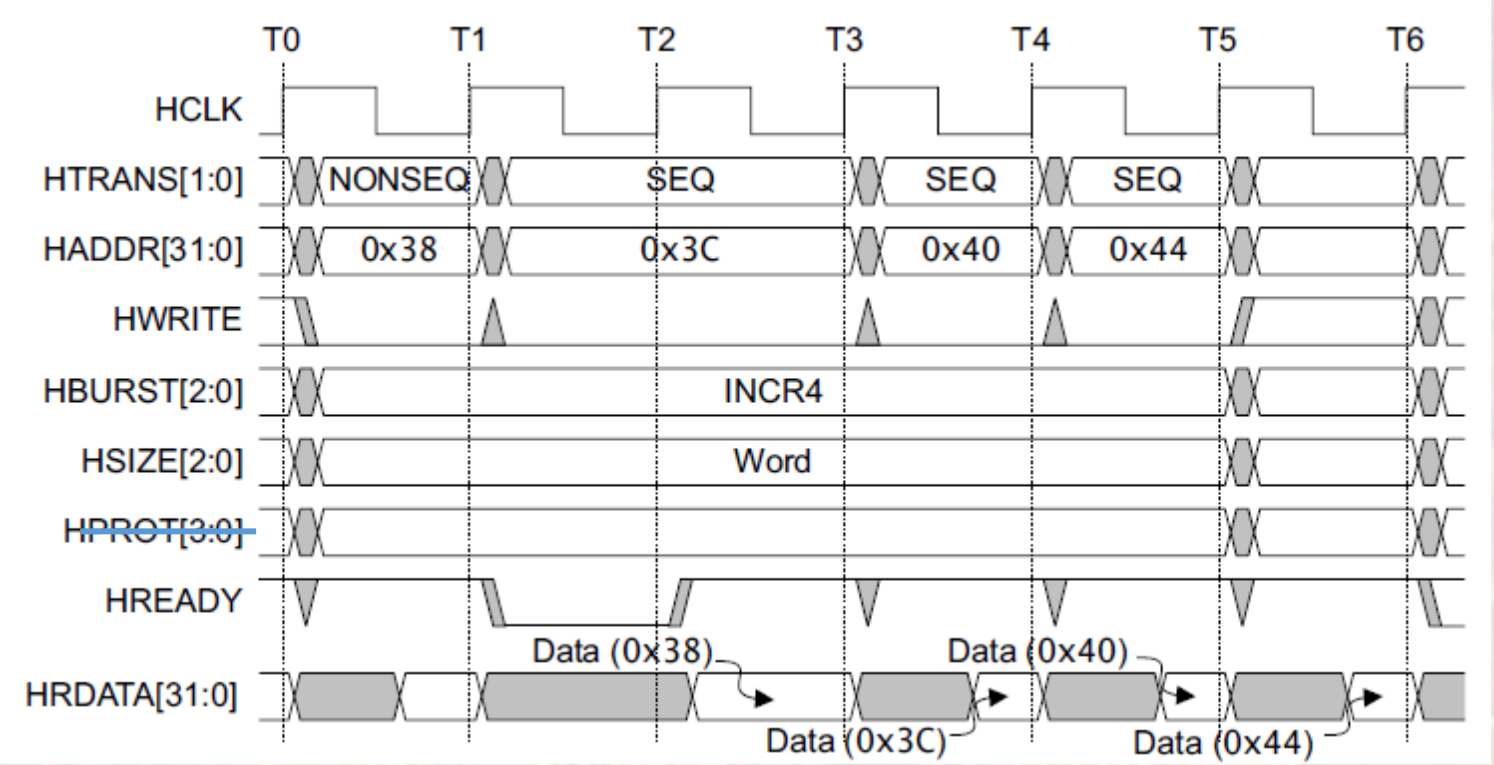
Parallel Communications - AHB

- AHB-Lite Transfer Types
  - Example – wrapping burst – 4 beats



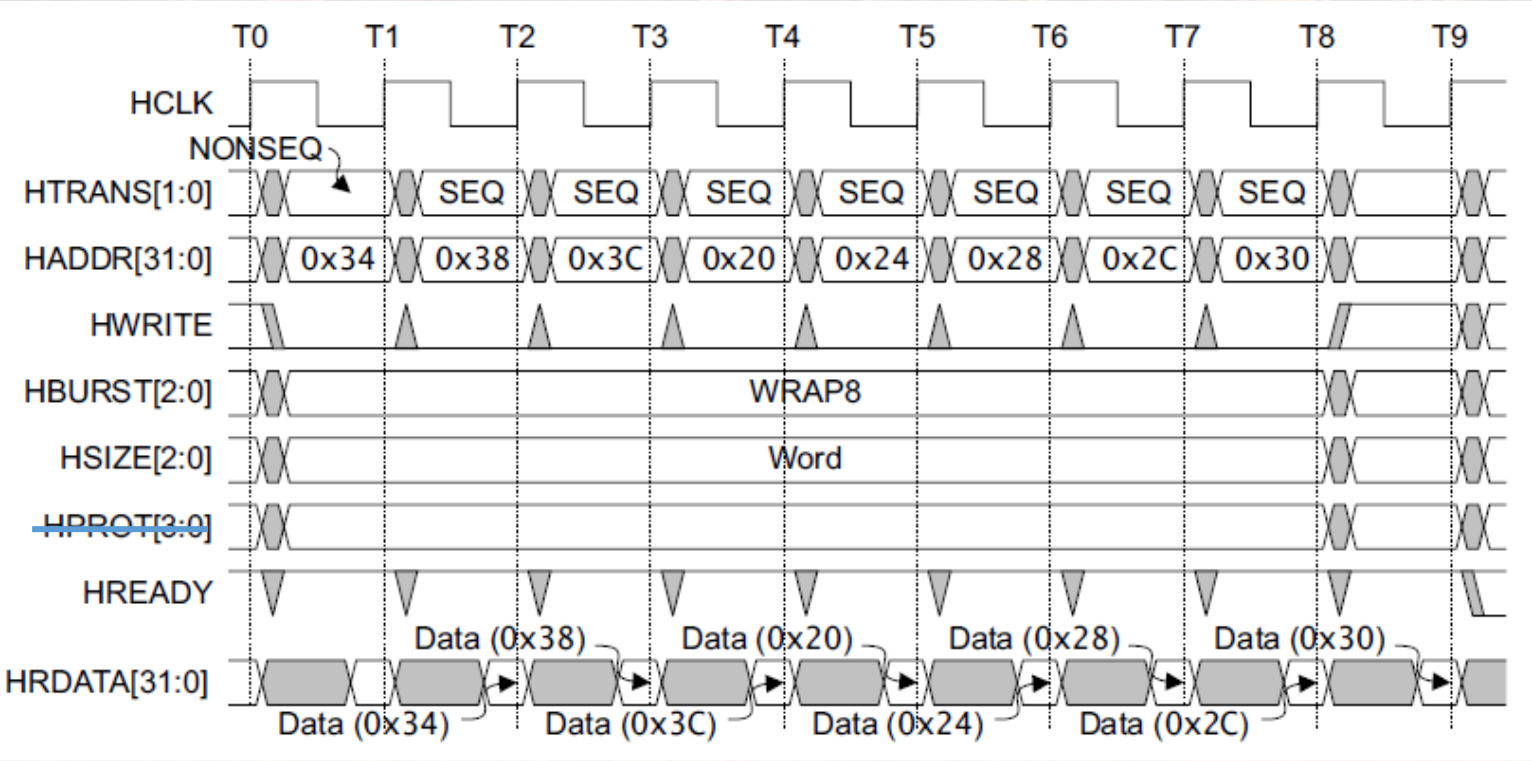
Parallel Communications - AHB

- AHB-Lite Transfer Types
  - Example – incrementing burst – 4 beats w/wait



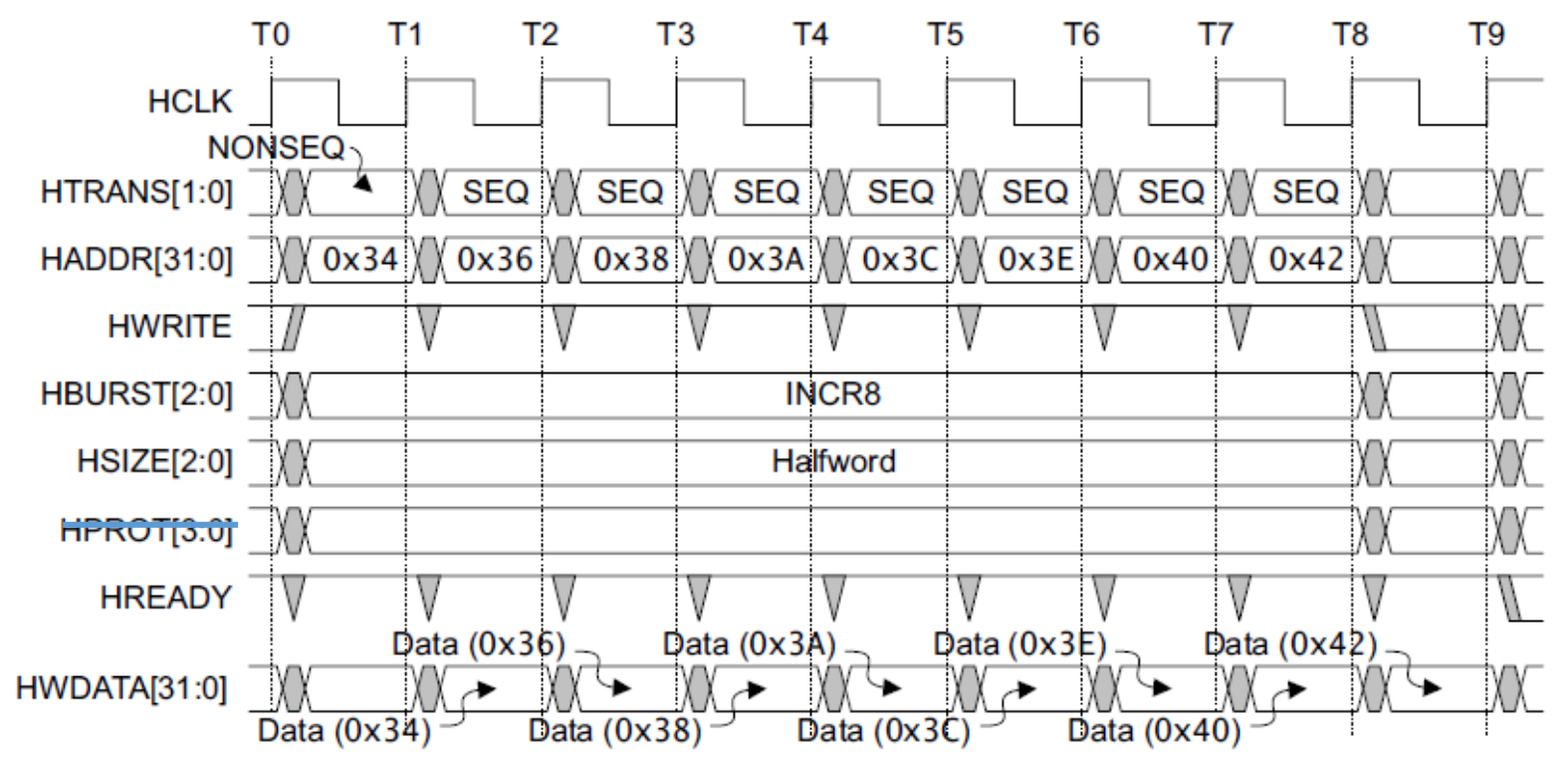
Parallel Communications - AHB

- AHB-Lite Transfer Types
  - Example – wrapping burst – 8 beats



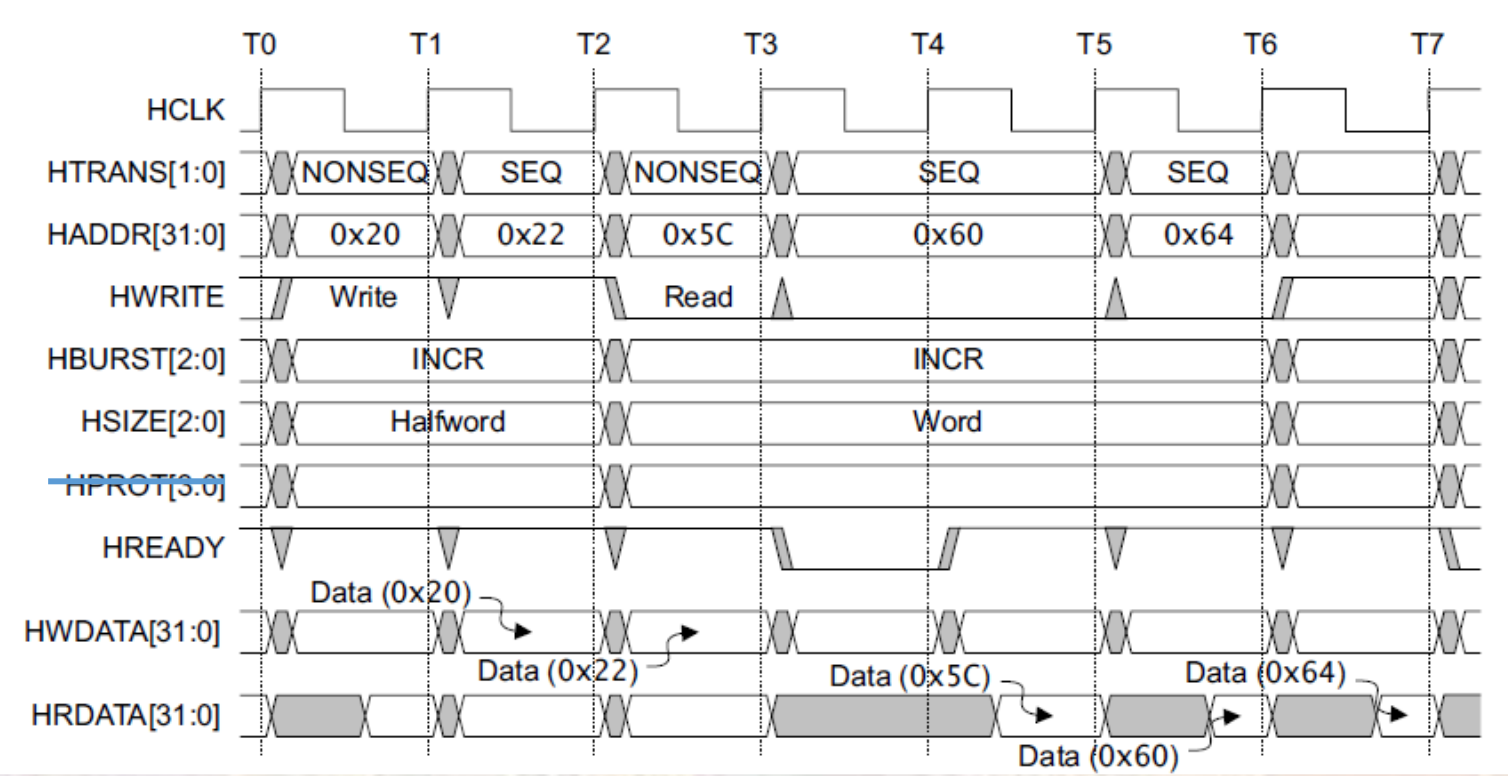
Parallel Communications - AHB

- AHB-Lite Transfer Types
  - Example – incrementing burst – 8 beats - halfword



Parallel Communications - AHB

- AHB-Lite Transfer Types
  - Example – undefined burst



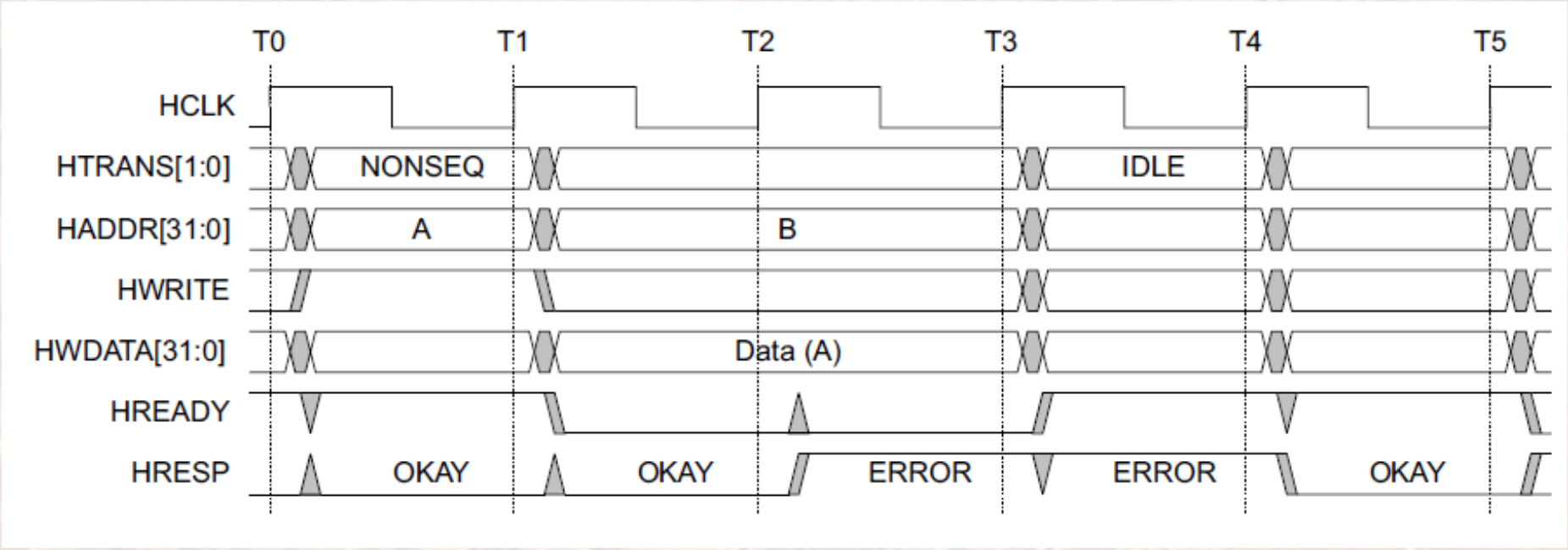
Parallel Communications - AHB

- AHB-Lite Slave Response
  - The slave uses the HRESP signal and the HREADY signal to confirm transactions
    - HRESP = 0 → OK (no known problem)
      - Unless an error is detected – the slave must respond with OK (idle, busy, …)
    - HRESP = 1 → ERROR (something has gone wrong)
      - A two-cycle response is required for an error condition with **HREADY** being asserted in the second cycle

	HREADY	
HRESP	**0**	**1**
0	Transfer pending	Successful transfer completed
1	ERROR response, first cycle	ERROR response, second cycle

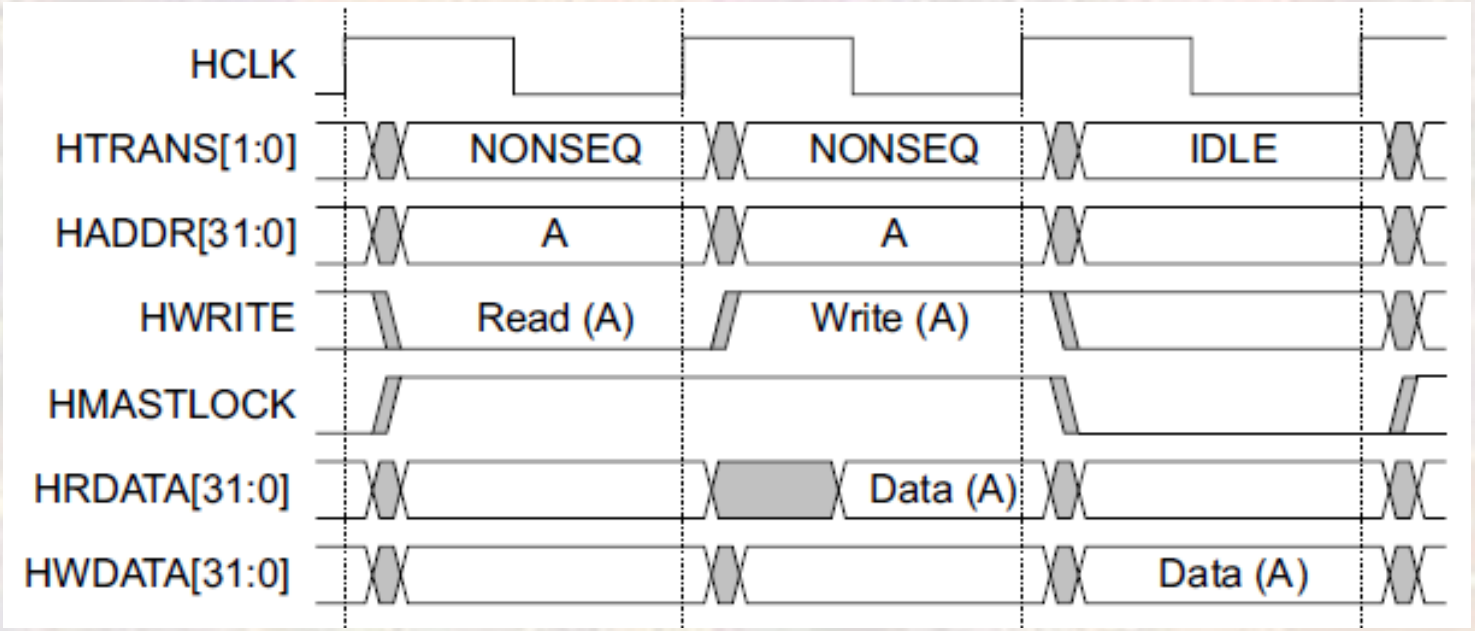
Parallel Communications - AHB

- AMBA – AHB-Lite
  - Example – Error condition



Parallel Communications - AHB

- AMBA – AHB-Lite
  - Locked Transaction
    - HMASTLOCK can be used to indicate the sequence is indivisible and to ensure slaves operate IN ORDER



Parallel Communications - AHB

- AMBA – AHB-Lite
  - Transfer Protection
    - HPROT[3:0} is used for transaction protection

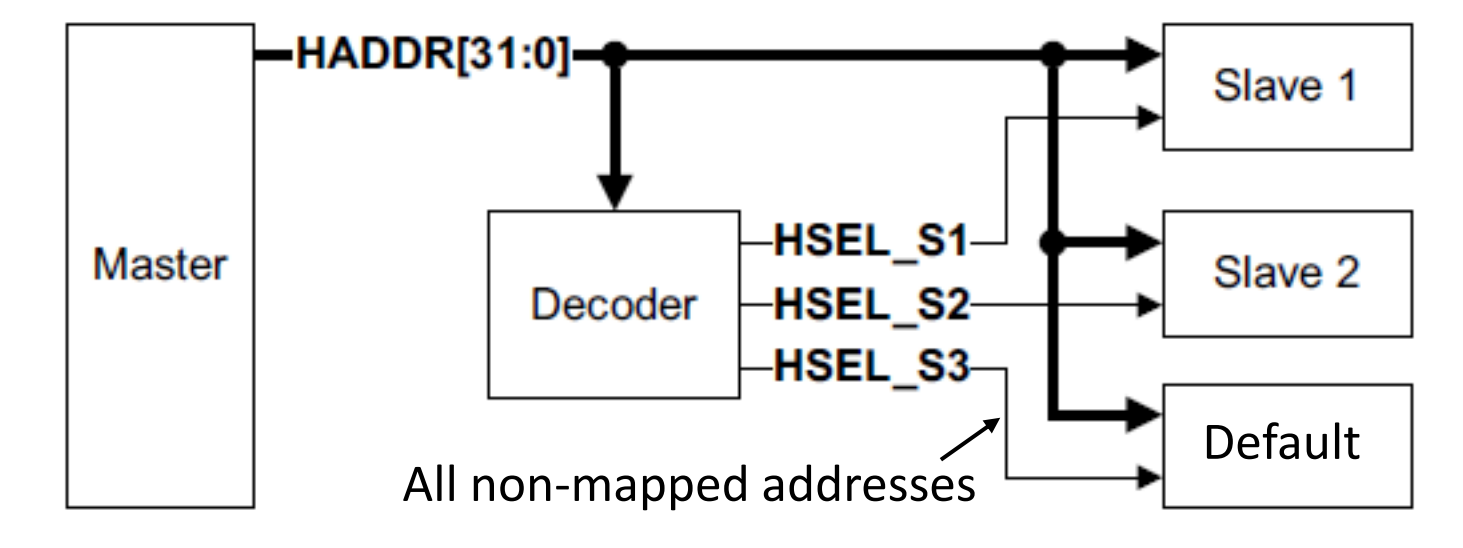
HPROT[3] Cacheable	HPROT[2] Bufferable	HPROT[1] Privileged	HPROT[0] Data/Opcode	Description
-	-	-	0	Opcode fetch
-	-	-	1	Data access
-	-	0	-	User access
-	-	1	-	Privileged access
-	0	-	-	Non-bufferable
-	1	-	-	Bufferable
0	-	-	-	Non-cacheable
1	-	-	-	Cacheable

Parallel Communications - AHB

- AMBA – AHB-Lite
  - Default Slave
    - Any time the address map is not full a default slave must be created
      - All non-filled addresses must point to the slave
      - Any Sequential or non-sequential accesses must respond with an ERROR
      - Any Idle or busy accesses must respond with OK

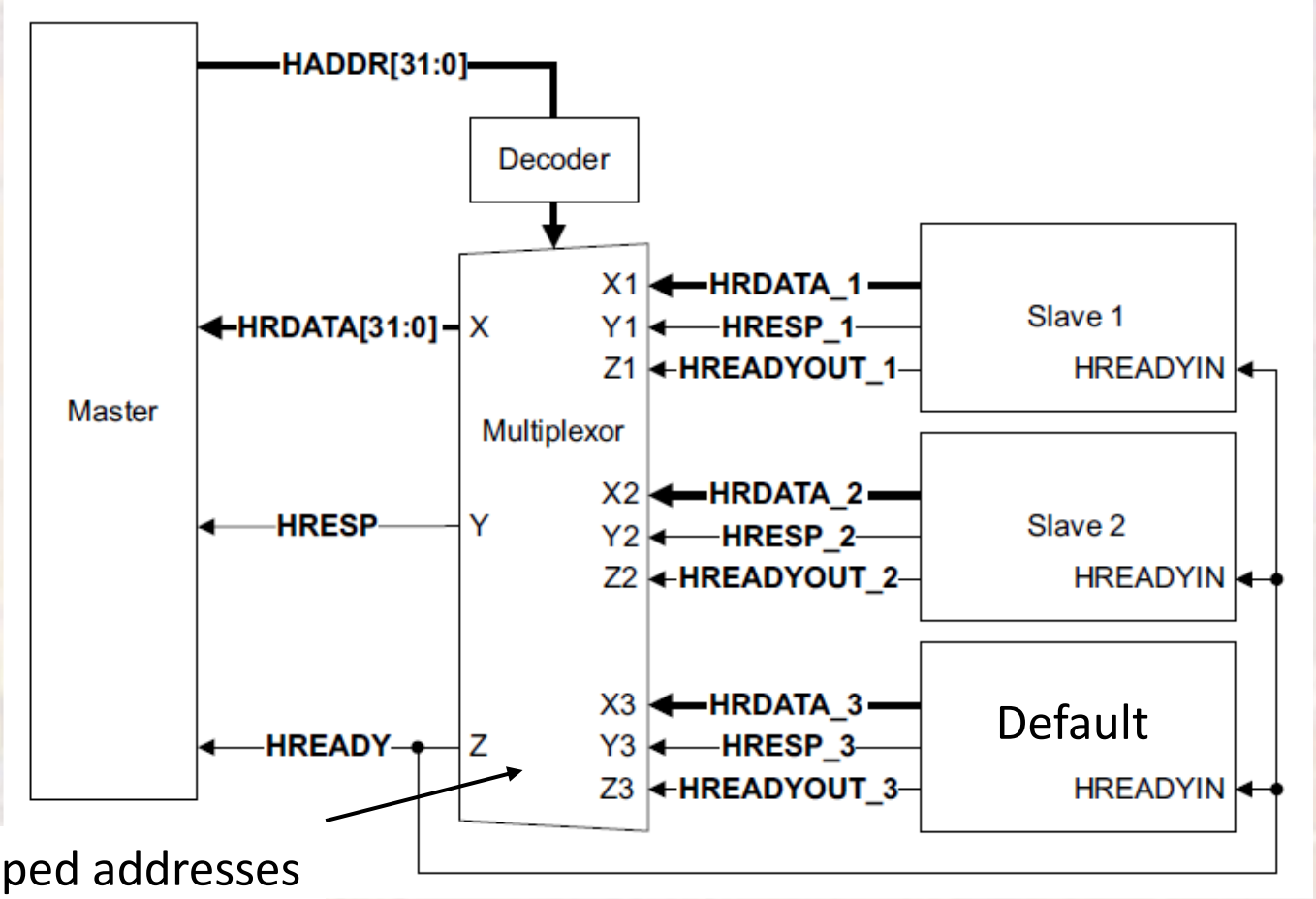
Parallel Communications - AHB

- AMBA – AHB-Lite
  - Decoder



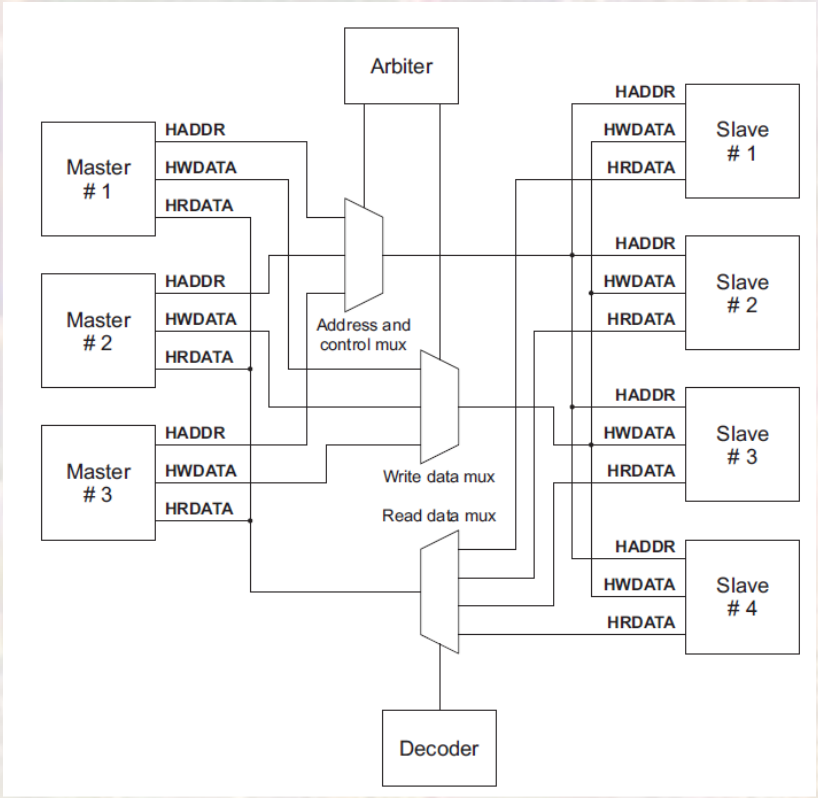
Parallel Communications - AHB

- AMBA – AHB-Lite
  - Multiplexor



Parallel Communications - AHB

- AMBA – AHB
  - The full version of AHB supports multiple masters on a single bus fabric

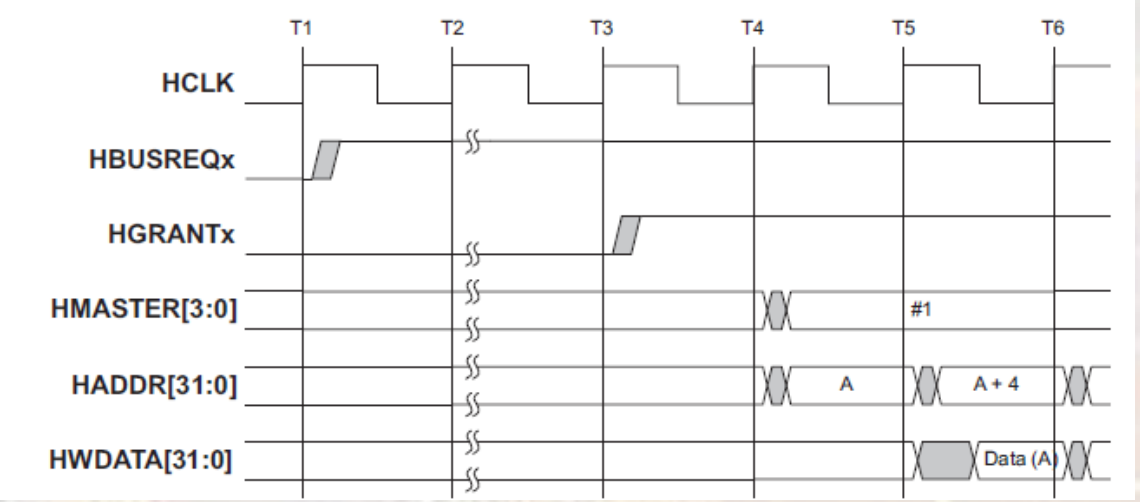


Parallel Communications - AHB

- AMBA – AHB
  - Additional Signals
    - HBUSREQx – Bus request from master (x) – 16 max
    - HGRANTx - Bus grant to master (x) from the arbiter
    - HMASTER[3:0] – Identifies which master currently has bus access
    - HSPLITx[15:0] – Slave signal to indicate which master may have split access
    - HLOCKx – Master (x) would like locked access to the bus
    - HRESP[1:0] – Slave response signals
      - OKAY, ERROR
      - RETRY – Master should retry the transfer
      - SPLIT – Master should release the bus, slave will inform the arbiter when it can continue

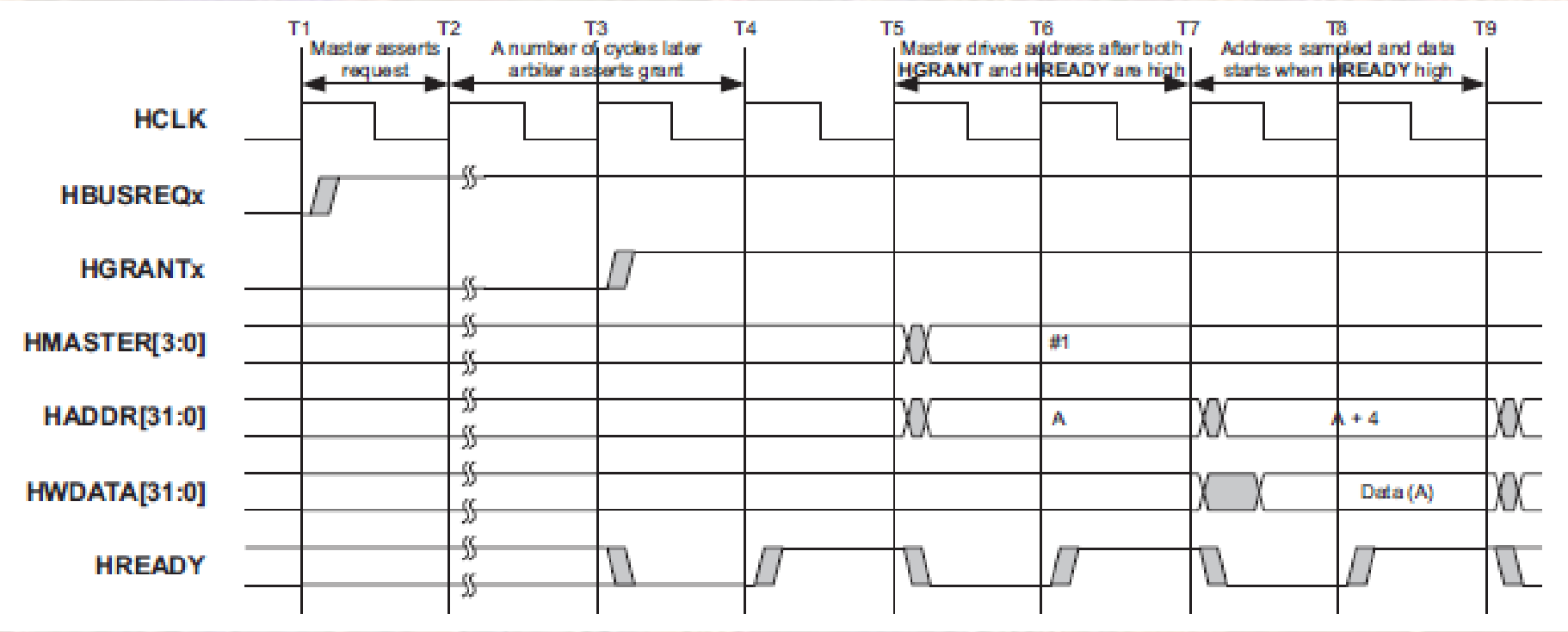
Parallel Communications - AHB

- AMBA – AHB
  - Bus Request/Grant
    - Master must request access to the bus
    - Arbiter grants request
      - Round robin
      - Fixed priority



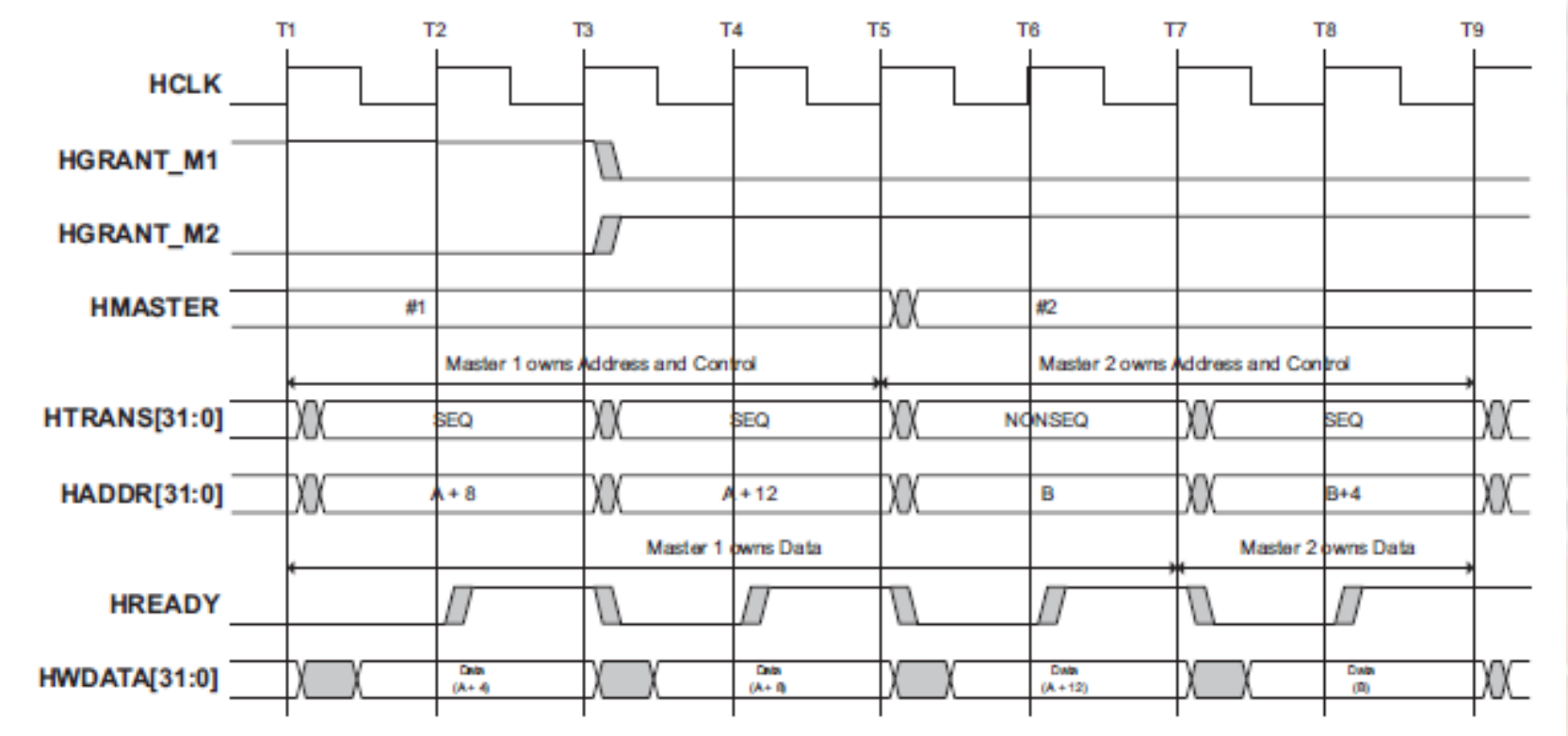
Parallel Communications - AHB

- AMBA – AHB
  - Bus Request/Grant



Parallel Communications - AHB

- AMBA – AHB
  - Bus Request/Grant



Parallel Communications - AHB

- AMBA – AHB
  - Split Access

1. The master starts the transfer in an identical way to any other transfer and issues address and control information

2. If the slave is able to provide data immediately it may do so. If the slave decides that it may take a number of cycles to obtain the data it gives a SPLIT transfer response. During every transfer the arbiter broadcasts a number, or tag, showing which master is using the bus. The slave must record this number, to use it to restart the transfer at a later time.

3. The arbiter grants other masters use of the bus and the action of the SPLIT response allows bus master handover to occur. If all other masters have also received a SPLIT response then the default master is granted.

Parallel Communications - AHB

- AMBA – AHB
  - Split Access – cont'd

4. When the slave is ready to complete the transfer it asserts the appropriate bit of the **HSPLITx** bus to the arbiter to indicate which master should be regranted access to the bus.

5. The arbiter observes the **HSPLITx** signals on every cycle, and when any bit of **HSPLITx** is asserted the arbiter restores the priority of the appropriate master.

6. Eventually the arbiter will grant the master so it can re-attempt the transfer. This may not occur immediately if a higher priority master is using the bus.

7. When the transfer eventually takes place the slave finishes with an OKAY transfer response.