DESIGN INTERNSHIP (DI-26)

FOR

MAVEN SILICON

Centre of Excellence in VLSI

AHB TO APB BRIDGE DESIGN

PROJECT REPORT

DONE BY -

Vetriashwath S

Introduction - AHB to APB Bridge:

The AHB bus is a high-performance and high-bandwidth bus protocol commonly used in complex SoC (System-on-Chip) designs. It is typically used for connecting high-speed and critical components such as processors, memories, and high-speed peripherals.

On the other hand, the APB bus is a low-power and low-bandwidth bus protocol commonly used for connecting low-speed peripherals in an SoC. It is designed to minimize power consumption and is suitable for connecting slower and less critical components.

The purpose of an AHB to APB bridge is to enable communication between the high-performance AHB bus and the low-power APB bus. The bridge typically handles address decoding, data transfer, protocol translation, and bus synchronization between the AHB and APB buses. It ensures that transactions initiated on the AHB bus are correctly translated and communicated to the APB bus, and vice versa.

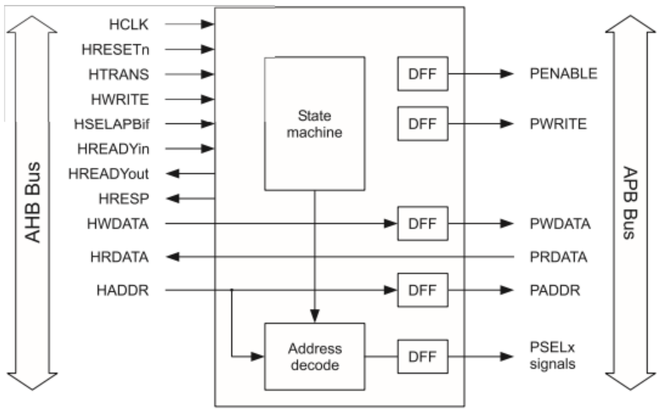


Fig1: simple diagram of AHB to APB Bridge

AMBA AHB Signals

Name	Source	Description
HCLK	Clock source	This clock times all bus transfers. All signal timings are related to the rising edge of HCLK.
HRESETn	Reset controller	The bus reset signal is active LOW and is used to reset the system and the bus.This is the only active LOW signal.
HADDR[31:0]	Master	The 32-bit system address bus.
HTRANS[1:0]	Master	Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, IDLE or BUSY.

Name	Source	Description
HWRITE	Master	When HIGH this signal indicates a write transfer and when LOW a read transfer.
HSIZE[2:0]	Master	Indicates the size of the transfer, which is typically byte (8-bit), halfword (16-bit) or word (32-bit). The protocol allows for larger transfer sizes up to a maximum of 1024 bits.
HBURST[2:0]	Master	Indicates if the transfer forms part of a burst. Four, eight and sixteen beat bursts are supported and the burst may be either incrementing or wrapping.
HPROT[3:0]	Master	The protection control signals provide additional information about a bus access and are primarily intended for use by any module that wishes to implement some level of protection. This is optional
HWDATA[31:0]	Master	The write data bus is used to transfer data from the master to the bus slaves during write operations. A minimum data bus width of 32 bits is recommended.
HSELx	Decoder	Each AHB slave has its own slave select signal and this signal indicates that the current transfer is intended for the selected slave. This signal is simply a combinatorial decode of the address bus.
HRDATA[31:0]	Slave	The read data bus is used to transfer data from bus slaves to the bus master during read operations. A minimum data bus width of 32 bits is recommended.
HREADY	Slave	When HIGH the HREADY signal indicates that a transfer has finished on the bus. This signal may be driven LOW to extend a transfer. Note: Slaves on the bus require HREADY as both an input and an output signal.
HRESP[1:0]	Slave	The transfer response provides additional information on the status of a transfer. Four different responses are provided, OKAY, ERROR, RETRY and SPLIT.

AMBA APB Signals

Name	Source	Description
PCLK	Clock Source	The rising edge of PCLK is used to time all transfers on the APB.
PRESETn	Reset Controller	The APB bus reset signal is active LOW and this signal will normally be connected directly to the system bus reset signal.
PADDR[31:0]	Master	This is the APB address bus, which may be up to 32-bits wide and is driven by the peripheral bus bridge unit.
PSELx	Decoder	A signal from the secondary decoder, within the peripheral bus bridge unit, to each peripheral bus slave x. This signal indicates that the slave device is selected and a data transfer is required. There is a PSELx signal for each bus slave.

Name	Source	Description
PENABLE	Master	This strobe signal is used to time all accesses on the peripheral bus. The enable signal is used to indicate the second cycle of an APB transfer. The rising edge of PENABLE occurs in the middle of the APB transfer.
PWRITE	Master	When HIGH this signal indicates an APB write access and when LOW a read access.
PRDATA[31:0]	Slave	The read data bus is driven by the selected slave during read cycles (when PWRITE is LOW). The read data bus can be up to 32-bits wide.
PWDATA[31:0]	Master	The write data bus is driven by the peripheral bus bridge unit during write cycles (when PWRITE is HIGH). The write data bus can be up to 32-bits wide.

Bridge Architecture:

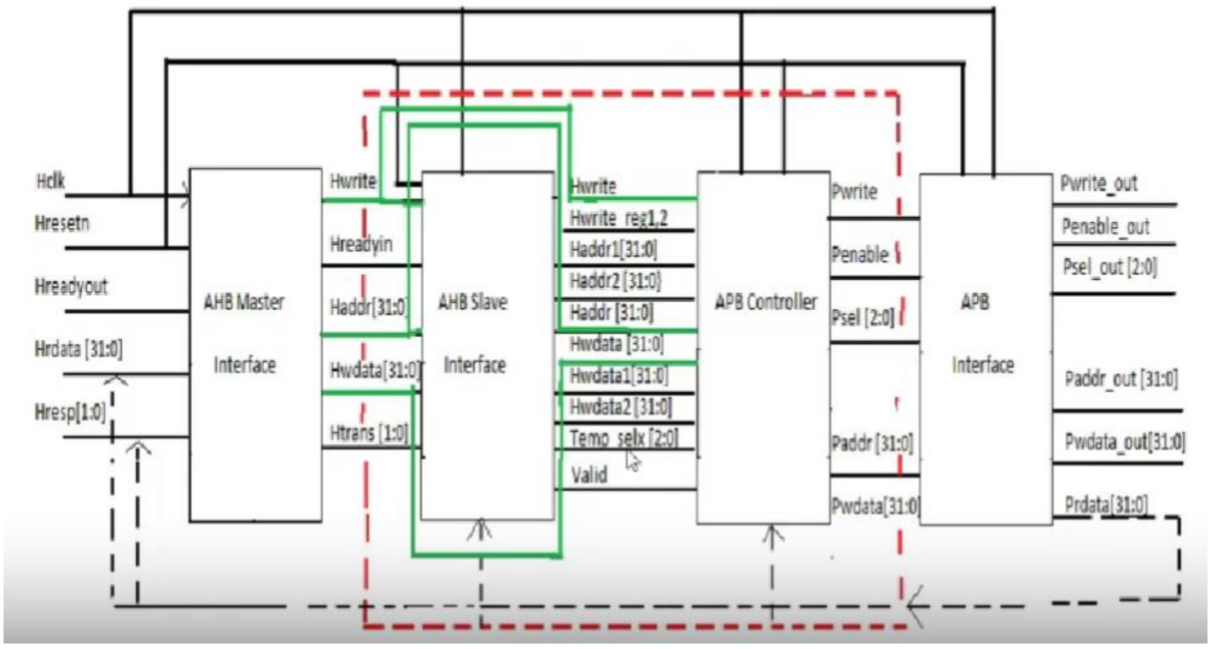


Fig 2: Bridge Architecture

This Bridge Architecture include modules like AHB Master, Top Module(AHB Slave Interface, APB Controller), APB Interface.

AHB Master:

An AHB master is responsible for controlling and initiating transactions on the AHB bus. It acts as a bus master and can communicate with other modules or peripherals in the system that are connected as AHB slaves.

It is responsible for the action to be performed by the AHB to APB Bridge. Some of the actions includes

- Single read
- Single write
- Burst 4 increment read
- Burst 4 wrap read
- Burst 4 increment write
- Burst 4 wrap write

Top module Tb:

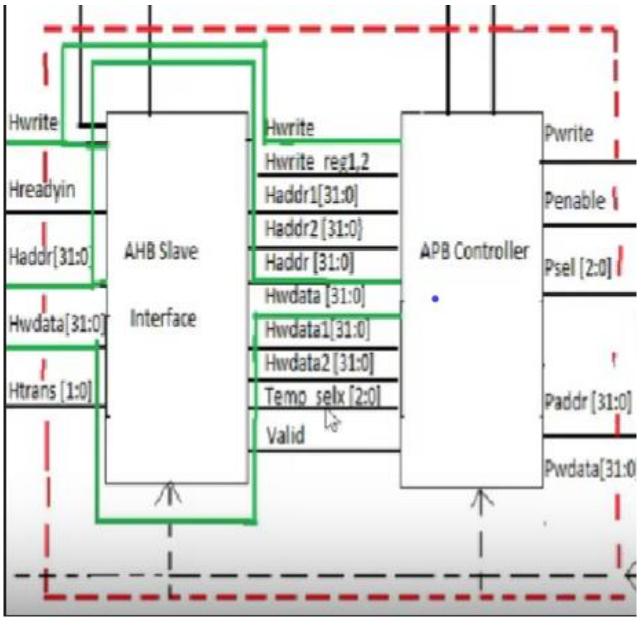


Fig 3: Top module

Top Tb is a module which includes both AHB Slave interface and APB controller. This module contains the input of AHB slave interface and the output of APB controller.

AHB Slave Interface:

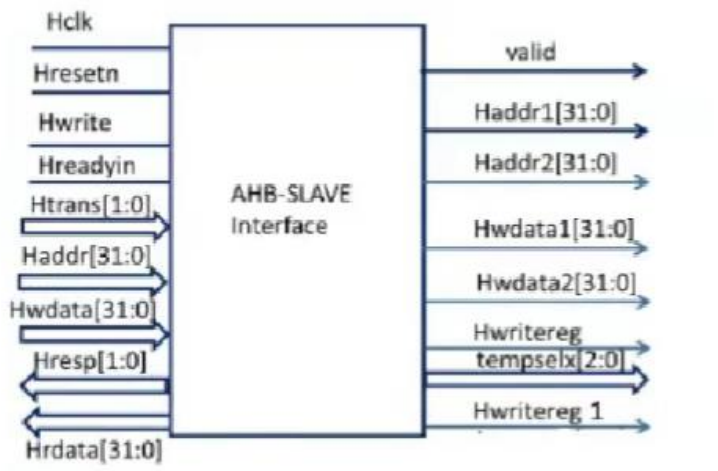


Fig 4: AHB Slave Interface

An AHB slave interface refers to the interface through which a peripheral or module connects to the AHB bus as a slave. The AHB slave interface allows the module to receive read and write transactions initiated by AHB bus masters.

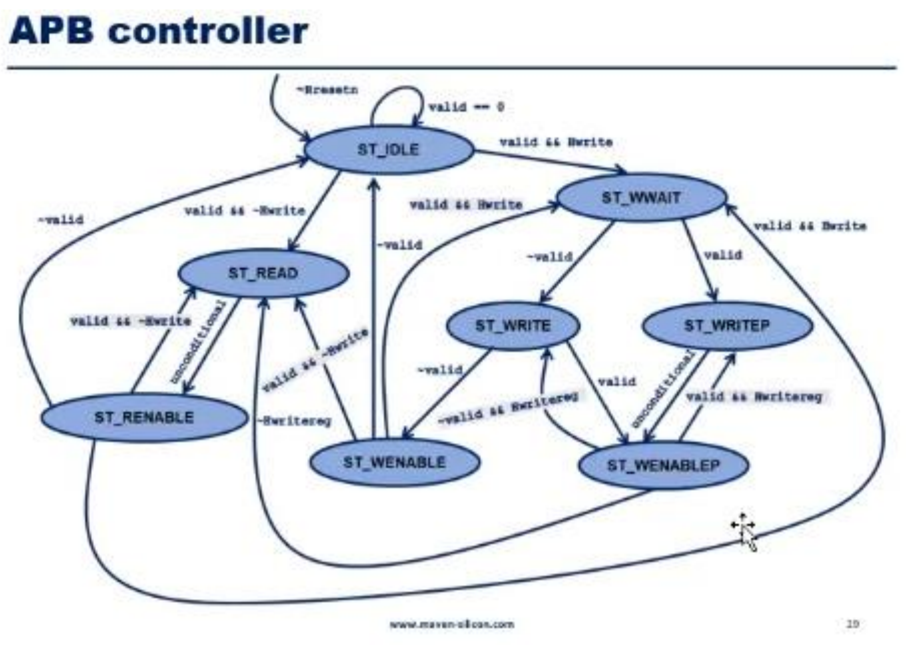
The AHB slave interface typically consists of the following signals:

- HRESETn: This is the active-low reset signal for the AHB slave. When asserted, it resets the slave and brings it into a known state.
- HCLK: The clock signal for the AHB bus. It provides the timing reference for the AHB slave to synchronize its internal operations with the bus.
- HSEL: The slave select signal that indicates whether the current transfer is intended for the slave. When this signal is asserted, the slave should decode the address and respond accordingly.
- HTRANS: The transfer type signal that specifies the type of transfer being initiated by the AHB master. It can indicate single transfers, multiple transfers, or non-sequential transfers.
- HADDR: The address signal that carries the target address for the current transfer. The slave uses this address to determine if the transfer is intended for it.

- HWDATA: The write data signal for write transfers. The AHB master provides the data to be written to the slave on this bus.
- HRDATA: The read data signal for read transfers. The slave drives the read data on this bus in response to a read request.
- HWRITE: The write signal that indicates whether the current transfer is a write operation. The slave should latch the data on the HWDATA bus when this signal is asserted.
- HSIZE: The size signal that specifies the size of the current transfer, indicating the number of bytes being transferred.
- HREADY: The ready signal that indicates the slave's readiness to accept the transfer. The slave asserts this signal to acknowledge the completion of the current transfer.
- HRESP: The response signal that indicates the status of the transfer. It can be used to indicate if the transfer was successful (OKAY) or encountered an error (ERROR).

These signals enable the AHB slave to interface with the AHB bus and handle transactions initiated by the bus masters. The AHB slave interface allows for high-performance communication between the AHB bus and the connected peripheral or module.

APB Controller:



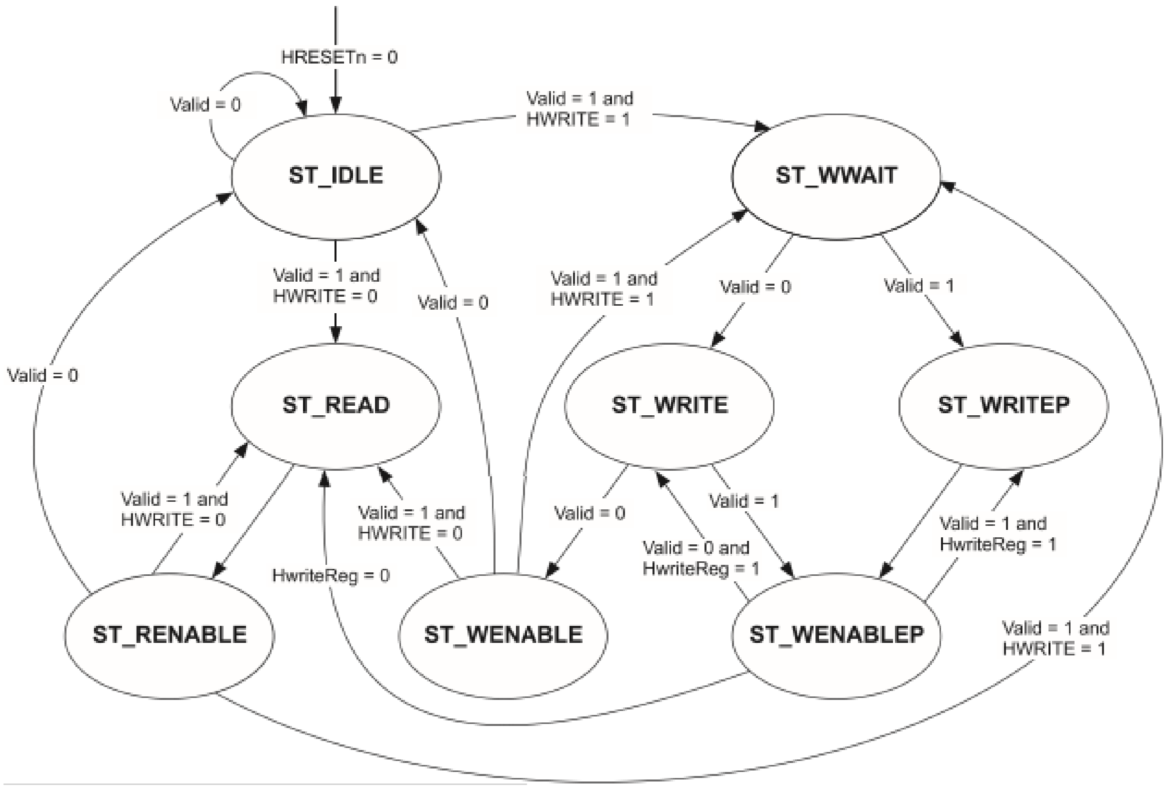


Fig 5: FSM logic of APB controller

APB controller works based on FSM (Finite State Machine) logic.

An FSM consists of a set of states, a set of inputs, a set of outputs, and a set of transitions that define the behaviour and operation of the system. The FSM operates based on its current state and the inputs it receives, and it produces outputs and transitions to new states based on predefined logic.

ST_IDLE:

During this state the APB buses and PWRITE are driven with the last values they had, and PSEL and PENABLE lines are driven LOW.

The ST_IDLE state is entered from:

- reset, when the system is initialized
- ST_RENABLE, ST_WENABLE, or ST_IDLE, when there are no peripheral transfers to perform.

The next state is:

- ST_READ, for a read transfer, when the AHB contains a valid APB read transfer

- ST_WWAIT, for a write transfer, when the AHB contains a valid APB write transfer.

ST_READ:

During this state the address is decoded and driven onto PADDR, the relevant PSEL line is driven HIGH, and PWRITE is driven LOW. A wait state is always inserted to ensure that the data phase of the current AHB transfer does not complete until the APB read data has been driven onto HRDATA.

- The ST_READ state is entered from ST_IDLE, ST_RENABLE, ST_WENABLE, or ST_WENABLEP during a valid read transfer.
- The next state will always be ST_RENABLE.

ST_WWAIT:

This state is needed due to the pipelined structure of AHB transfers, to allow the AHB side of the write transfer to complete so that the write data becomes available on HWDATA. The APB write transfer is then started in the next clock cycle.

- The ST_WWAIT state is entered from ST_IDLE, ST_RENABLE, or ST WENABLE, during a valid write transfer.
- The next state will always be ST_WRITE.

ST_WRITE:

During this state the address is decoded and driven onto PADDR, the relevant PSEL line is driven HIGH, and PWRITE is driven HIGH.

A wait state is not inserted, as a single write transfer can complete without affecting the AHB.

The ST_WRITE state is entered from:

- ST_WWAIT, when there are no further peripheral transfers to perform
- ST WENABLEP, when the currently pending peripheral transfer is a write, and there are no further transfers to perform.

The next state is:

- ST WENABLE, when there are no further peripheral transfers to perform
- ST_WENABLEP, when there is one further peripheral write transfer to perform.

ST_WRITEP:

During this state the address is decoded and driven onto PADDR, the relevant PSEL line is driven HIGH, and PWRITE is driven HIGH. A wait state is always inserted, a there must only ever be one pending transfer between the currently performed APB transfer and the currently driven AHB transfer. See the write transfer timing diagrams in the AMBA Specification (Rev 2.0) for more details.

The ST_WRITEP state is entered from:

- ST_WWAIT, when there is a further peripheral transfer to perform.
- ST_WENABLEP, when the currently pending peripheral transfer is a write, and there is a further transfer to perform.
- The next state will always be ST_WENABLEP

ST_RENABLE:

During this state the PENABLE output is driven HIGH, enabling the current APB transfer. All other APB outputs remain the same as the previous cycle.

- The ST_RENABLE state is always entered from ST_READ.

The next state is:

- ST_READ, when there is a further peripheral read transfer to perform
- ST_WWAIT, when there is a further peripheral write transfer to perform
- ST_IDLE, when there are no further peripheral transfers to perform.

ST_WENABLE:

During this state the PENABLE output is driven HIGH, enabling the current APB transfer. All other APB outputs remain the same as the previous cycle.

- The ST WENABLE state is always entered from ST_WRITE,

The next state is:

- ST_READ, when there is a further peripheral read transfer to perform
- ST_WWAIT, when there is a further peripheral write transfer to perform
- ST_IDLE, when there are no further peripheral transfers to perform.

ST WENABLEP:

A wait state is inserted if the pending transfer is a read because, when a read follows a write, an extra wait state must be inserted to allow the write transfer to complete on the APB before the read is started.

The ST WENABLEP state is entered from:

- ST_WRITE, when the currently driven AHB transfer is a peripheral transfer
- ST_WRITEP, when there is a pending peripheral transfer following the current write.

The next state is:

- ST_READ, when the pending transfer is a read
- ST_WRITE, when the pending transfer is a write, and there are no further transfers to perform
- ST_WRITEP, when the pending transfer is a write, and there is a further transfer to perform.

APB Interface:

The APB (Advanced Peripheral Bus) interface is used as a communication bus to connect peripheral devices or components to a system-on-chip (SoC), microcontroller, or other integrated circuits. It is a simple and low-power bus protocol designed for connecting slower peripherals to a central processing unit (CPU) or a higher-performance bus like the AHB (Advanced High-performance Bus).

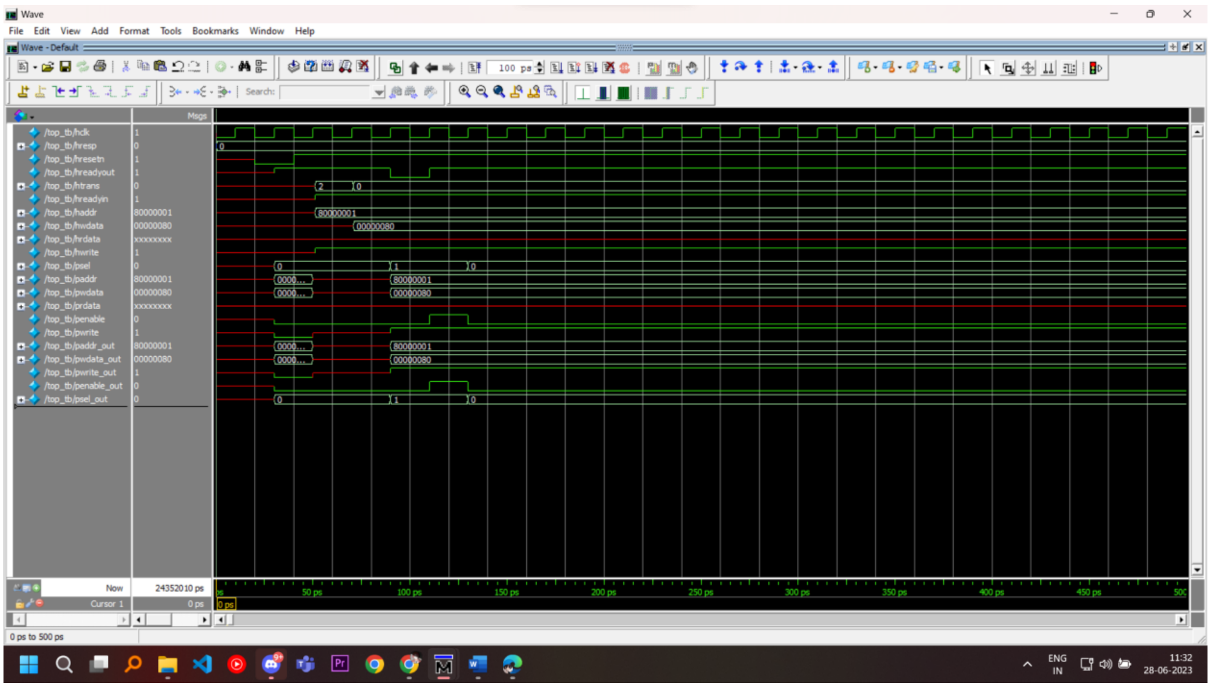
The APB interface serves as an interface between the CPU or bus master and the peripheral devices, enabling data transfer and control signals between them.

Output Wave Forms:

Single Write Transfer:

A single write transfer refers to a data transfer operation where a bus master initiates a write request to a target slave device to store a single piece of data.

A single write transfer typically involves storing a single piece of data at a specific address within a peripheral device. The bus master initiates the write operation by providing the address and the data to be stored. The target slave device receives the data and performs the necessary storage operation.



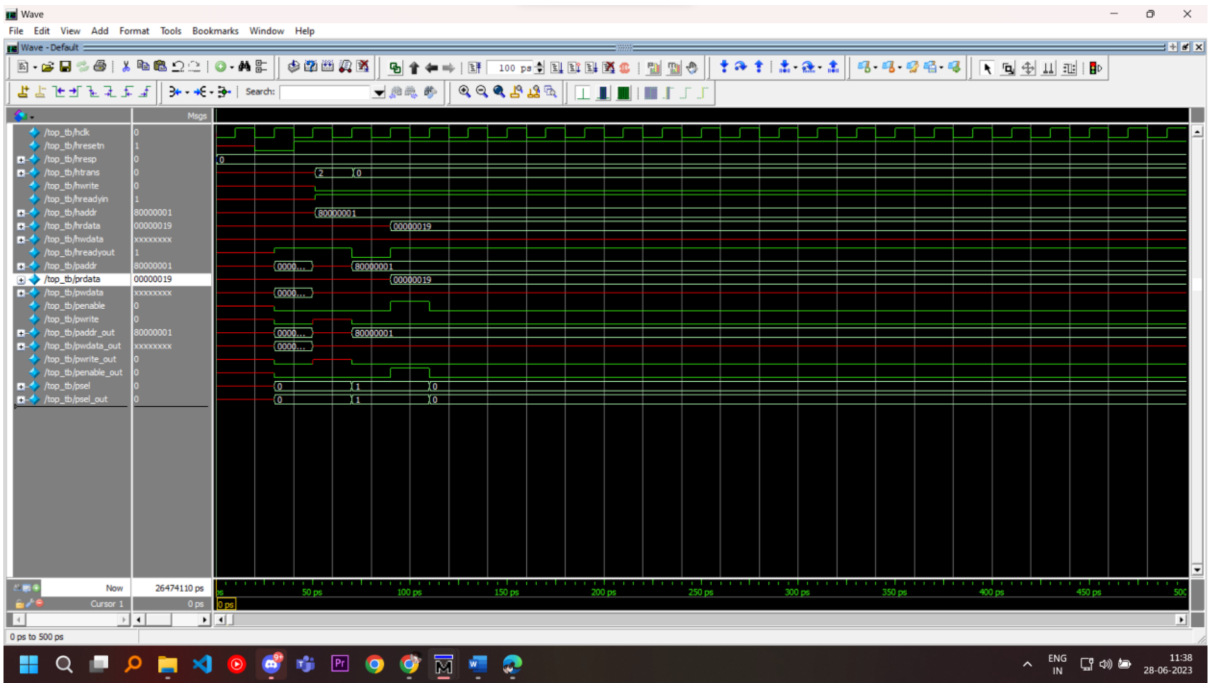
```
Msgs
/top_tb/hclk          1
/top_tb/hresp         0
/top_tb/hresetn       1
/top_tb/hreadyout     1
/top_tb/htrans        0
/top_tb/hreadyin      1
/top_tb/haddr         80000001
/top_tb/hwdata        00000080
/top_tb/hrdata        xxxxxxxx
/top_tb/hwrite        1
/top_tb/psel          0
/top_tb/paddr         80000001
/top_tb/pwdata        00000080
/top_tb/prdata        xxxxxxxx
/top_tb/penable       0
/top_tb/pwrite        1
/top_tb/paddr_out     80000001
/top_tb/pwdata_out    00000080
/top_tb/pwrite_out    1
/top_tb/penable_out   0
/top_tb/psel_out      0
Now                   24352010 ps
Cursor 1              0 ps
```

Fig 6: Single Write Transfer

Single Read Transfer:

A single read transfer refers to a data transfer operation where a bus master initiates a read request to a target slave device to retrieve a single piece of data.

A single read transfer typically involves a single data access or retrieval from a specific address in a peripheral device. It allows the bus master to obtain the requested data from the target slave device and use it for further processing, such as loading it into registers or performing calculations.



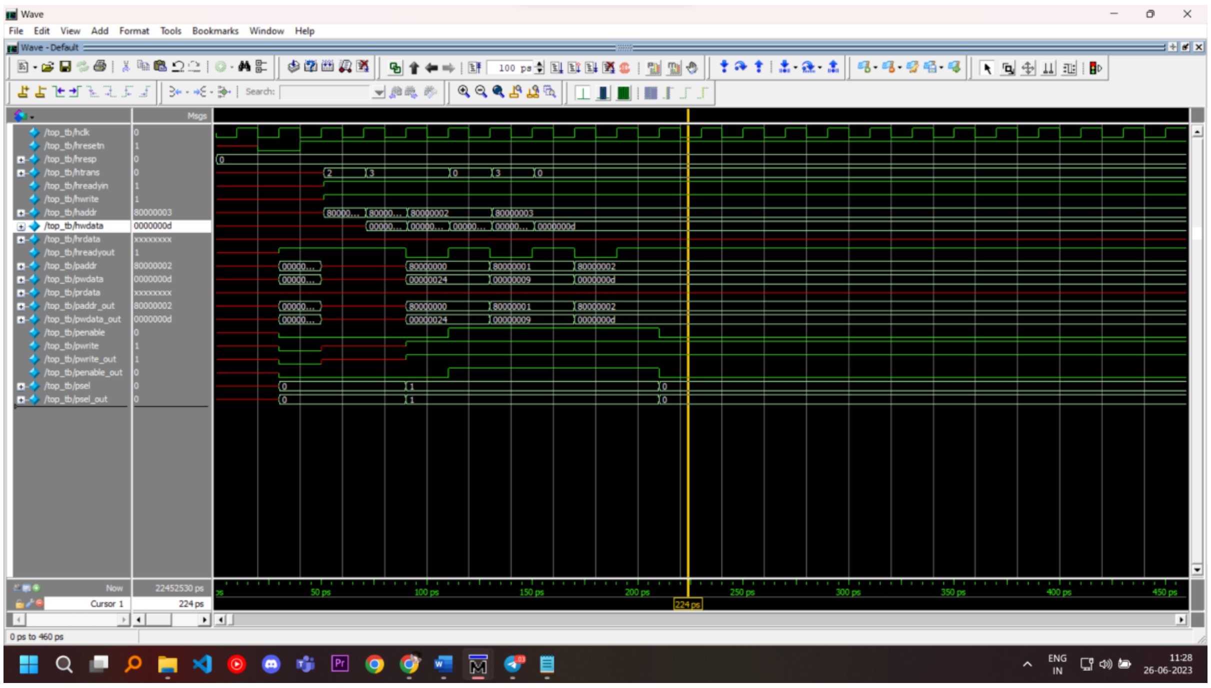
```
Msgs
/top_tb/hclk          0
/top_tb/hresetn       1
/top_tb/hresp         0
/top_tb/htrans        0
/top_tb/hwrite        0
/top_tb/hreadyin      1
/top_tb/haddr         80000001
/top_tb/hrdata        00000019
/top_tb/hwdata        xxxxxxxx
/top_tb/hreadyout     1
/top_tb/paddr         80000001
/top_tb/prdata        00000019
/top_tb/pwdata        xxxxxxxx
/top_tb/penable       0
/top_tb/pwrite        0
/top_tb/paddr_out     80000001
/top_tb/pwdata_out    xxxxxxxx
/top_tb/pwrite_out    0
/top_tb/penable_out   0
/top_tb/psel          0
/top_tb/psel_out      0
Now                   26474110 ps
Cursor 1              0 ps
```

Fig 7: Single Read Transfer

Burst Increment Write Transfer:

A burst increment write transfer is a specific type of data transfer operation in bus protocols. The burst increment write transfer allows for efficient writing of multiple data values to consecutive addresses without the need for individual address and data transactions for each value. It reduces the overhead associated with initiating separate write operations for each data value.

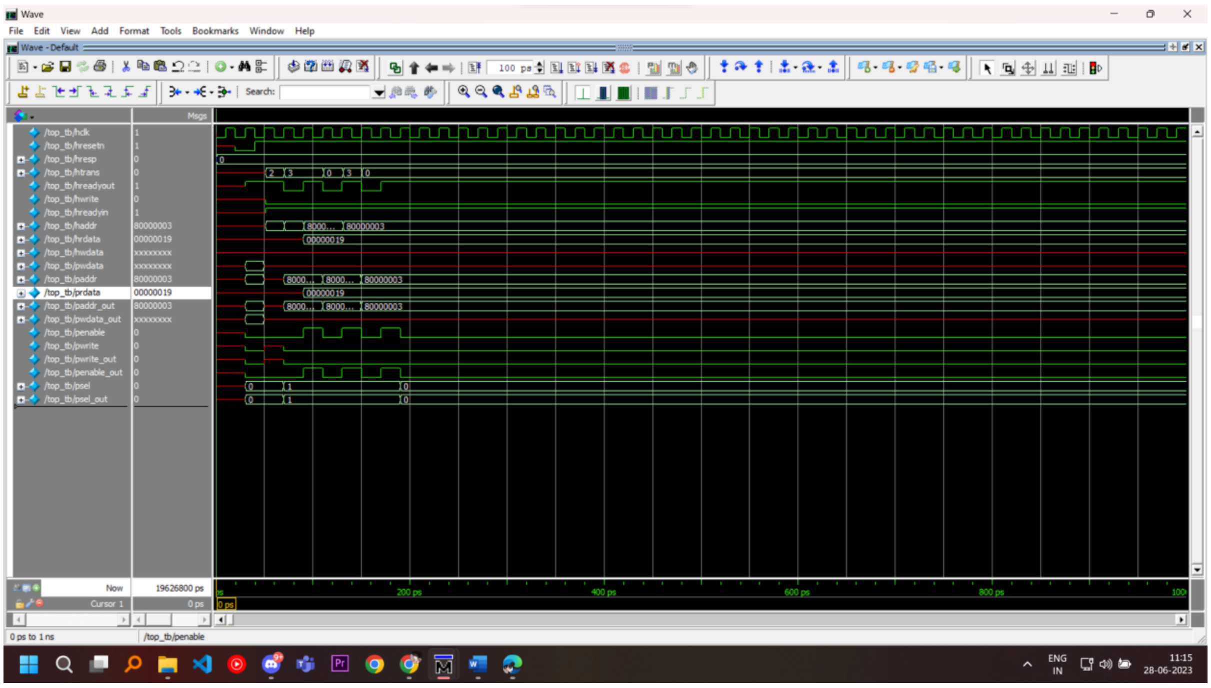
The number of data values written during a burst transfer can be predefined or dynamically determined based on system requirements. The burst length specifies the total number of data values to be written, and the address automatically increments after each write to enable sequential writes.



```
                          Msgs
/top_tb/hclk              0
/top_tb/hresetn           1
/top_tb/hresp             0
/top_tb/htrans            0
/top_tb/hreadyin          1
/top_tb/hwrite            1
/top_tb/haddr             80000003
/top_tb/hwdata            0000000d
/top_tb/hrdata            xxxxxxxx
/top_tb/hreadyout         1
/top_tb/paddr             80000002
/top_tb/pwdata            0000000d
/top_tb/prdata            xxxxxxxx
/top_tb/paddr_out         80000002
/top_tb/pwdata_out        0000000d
/top_tb/penable           0
/top_tb/pwrite            1
/top_tb/pwrite_out        1
/top_tb/penable_out       0
/top_tb/psel              0
/top_tb/psel_out          0

Now       22452530 ps
Cursor 1  224 ps
0 ps to 460 ps
```

Fig 8: Burst Increment Write Transfer



	Msgs
/top_tb/hclk	1
/top_tb/hresetn	1
/top_tb/hresp	0
/top_tb/htrans	0
/top_tb/hreadyout	1
/top_tb/hwrite	0
/top_tb/hreadyin	1
/top_tb/haddr	80000003
/top_tb/hrdata	00000019
/top_tb/hwdata	xxxxxxxx
/top_tb/pwdata	xxxxxxxx
/top_tb/paddr	80000003
/top_tb/prdata	00000019
/top_tb/paddr_out	80000003
/top_tb/pwdata_out	xxxxxxxx
/top_tb/penable	0
/top_tb/pwrite	0
/top_tb/pwrite_out	0
/top_tb/penable_out	0
/top_tb/psel	0
/top_tb/psel_out	0

Now 19626800 ps
Cursor 1 0 ps
0 ps to 1 ns /top_tb/penable

Fig 9. Burst Increment Write Transfer

Burst Wrap Write Transfer:

A burst wrap write transfer is another type of data transfer operation in bus protocols. The burst wrap write transfer is useful when there is a need to write multiple data values to consecutive addresses, and the address space wraps around to the starting address once the end is reached. It allows for efficient utilization of the address space and avoids the need for separate transactions to wrap around and continue writing.

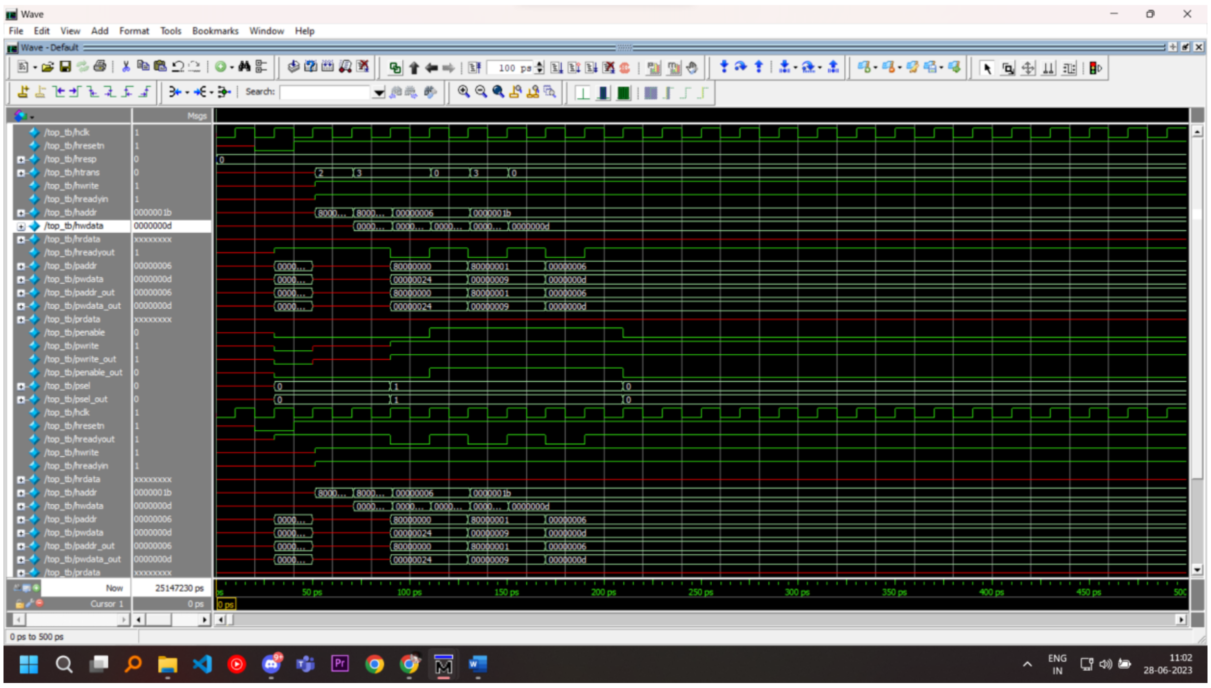


Fig 10: Burst Wrap Write Transfer

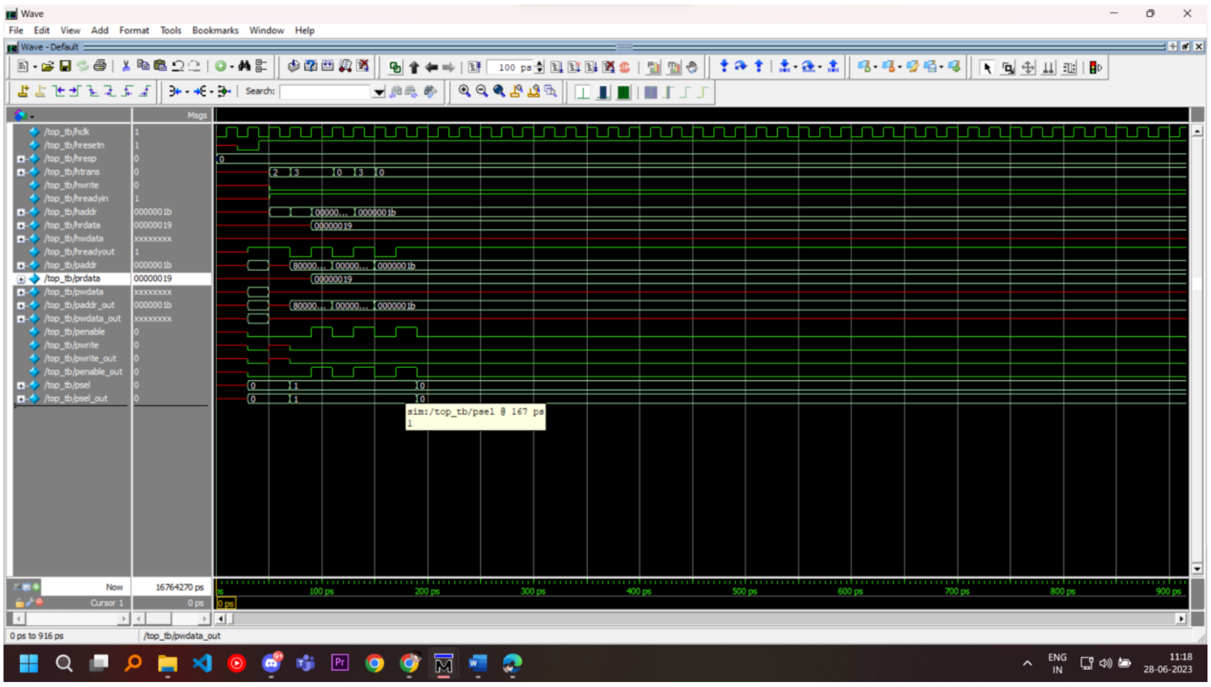


Fig 11: Burst Wrap Read Transfer

Conclusion:

In conclusion, the AHB to APB bridge design plays a critical role in integrating higher-performance AHB bus systems with lower-performance APB peripherals in a system-on-chip (SoC) or microcontroller design. The bridge acts as an interface, enabling seamless communication between the two bus protocols.

The AHB to APB bridge design allows for the efficient transfer of data and control signals between the AHB and APB buses. It handles protocol conversion, address decoding, data transfer, and timing synchronization to ensure proper communication between the higher-performance AHB bus and the lower-performance APB peripherals.

The bridge facilitates the initiation of read and write transfers from the AHB bus to the APB peripherals and vice versa. It ensures the translation of AHB transactions into APB transactions and vice versa, ensuring compatibility and interoperability between the two bus protocols.

Overall, the AHB to APB bridge design provides a reliable and efficient solution for integrating lower-performance APB peripherals into systems that utilize the higher-performance AHB bus. It enables seamless communication, efficient data transfer, and effective control between the AHB and APB domains, making it an essential component in modern system-on-chip designs.