Design and Implementation of Efficient FSM for AHB Slave

Design Document

.

….

VIVARTAN
TECHNOLOGIES
Revision 1.0 -

Contents

List of Figures

List of Tables

1. Introduction:

Advanced Microcontroller Bus Architecture (AMBA)is a protocol that is used as an open standard, on chip interconnect specification for the connection and management of functional blocks in a system-on-chip (SoC).

- AMBA assists the progress of right-first-time development of multiprocessor designs with a large number of controllers & peripherals.
- The Advanced Microcontroller Bus Architecture (AMBA) has the ability to re-use designs and here it means it has the ability to re-use IP. IP re-use in today's technology is an important factor in reducing the development costs and timescales for System-on-chip (SoC).
- AMBA is a standard interface specification that makes sure of the compatibility between IP components provided by different design teams or vendors.
- The worldwide reception of AMBA specifications all over the semiconductor industry has driven a comprehensive market in third party IP products and tools to support the development of AMBA based systems.

1.1 Overview of AMBA Specifications:

There are three different buses defined within the AMBA specification.

- Advanced High Performance Bus (AHB),
- Advanced System Bus (ASB), and
- Advanced Peripheral Bus (APB).

This document mainly deals with AMBA AHB and particularly AHB master.

1.10 Advanced High Performance Bus (AHB)

The AMBA AHB is for high-performance, high clock frequency system modules. The AHB acts as the high performance system backbone bus. AHB supports the efficient connection of processors, on-chip memories and off-chip external memory interfaces with low-power peripheral macro cell functions. AHB is also specified to ensure ease of use in an efficient design flow using synthesis and automated test techniques.

1.11 Advanced System Bus (ASB)

The AMBA ASB is for high-performance system modules. AMBA ASB is an alternative system bus suitable for use where the high-performance features of AHB are not required. ASB also supports the efficient connection of processors, on-chip memories and off-chip external memory interfaces with low-power peripheral macro cell functions.

1.12 Advanced Peripheral Bus (APB)

The AMBA APB is for low-power peripherals. AMBA APB is optimized for minimal power consumption and reduced interface complexity to support peripheral functions. APB can be used in conjunction with either version of the system bus.

1.2 Objectives of the AMBA specification:

The AMBA specification has been derived to satisfy four key requirements:

- To facilitate the right-first-time development of embedded microcontroller products with one or more CPUs or signal processors.
- To be technology-independent and ensure that highly reusable peripheral and system macrocells can be migrated across a diverse range of IC processes and be appropriate for full-custom, standard cell and gate array technologies.
- To encourage modular system design to improve processor independence, providing a development road-map for advanced cached CPU cores and the development of peripheral libraries
- To minimize the silicon infrastructure required to support efficient on-chip and off-chip communication for both operation and manufacturing tests.

1.3 A typical AMBA-based microcontroller.

An AMBA-based microcontroller typically consists of a high-performance system backbone bus (AMBA AHB or AMBA ASB), able to sustain the external memory bandwidth, on which the

CPU, on-chip memory and other Direct Memory Access(DMA) devices reside. This bus provides a high-bandwidth interface between the elements that are involved in the majority of transfers. Also located on the high performance bus is a bridge to the lower bandwidth APB, where most of the peripheral devices in the system are located (see figure 1.1)

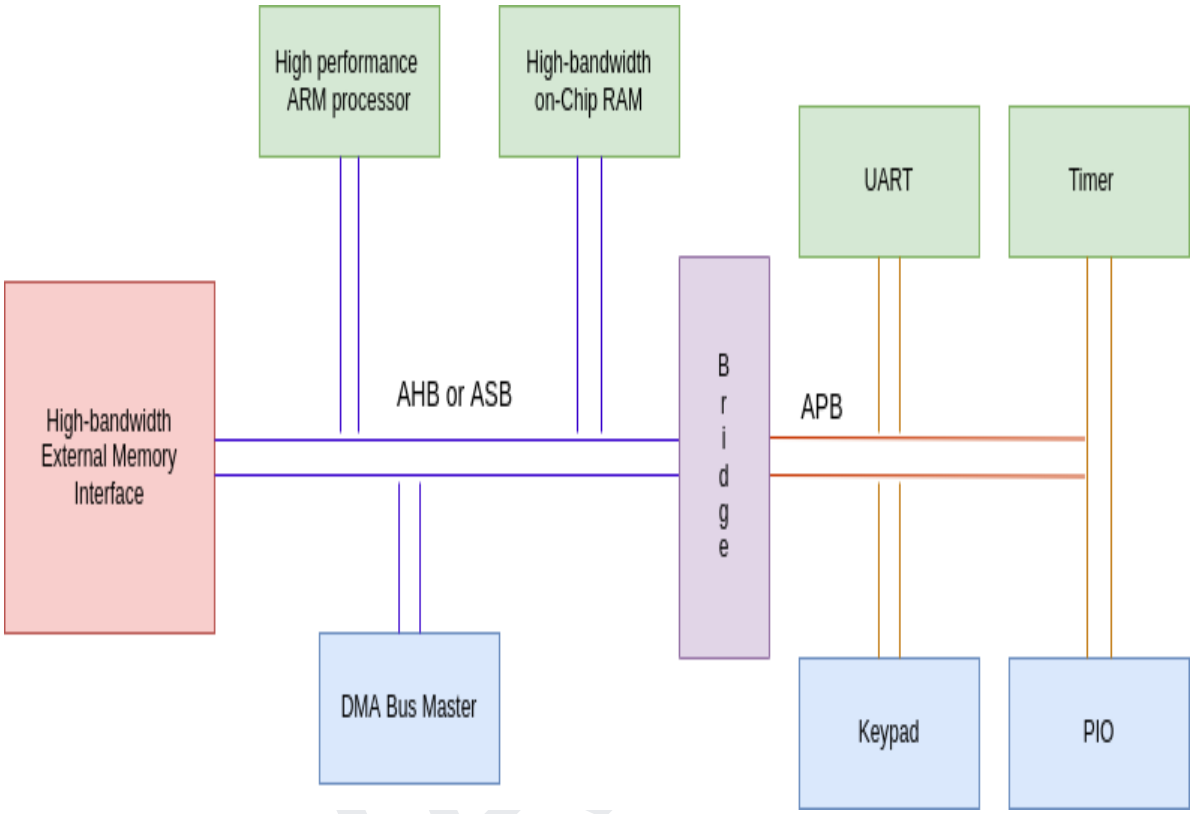

Figure 1.1 . Typical AMBA-based System.

AMBA AHB	AMBA ASB	AMBA APB
*High performance	High performance	low power
*Pipelined operation	Pipelined operation	latched address & control
*Multiple bus masters	Multiple bus masters	simple transfer
* Burst transfer		suitable for many peripherals
*Split transactions		

AMBA APB provides the basic peripheral macrocell communications infrastructure as a secondary bus from the higher bandwidth pipelined main system bus. Such peripherals typically:

- have interfaces which are memory-mapped registers
- have no high-bandwidth interfaces
- are accessed under programmed control.

The external memory interface is application-specific and may only have a narrow data path, but may also support a test access mode which allows the internal AMBA AHB, ASB and APB modules to be tested in isolation with system-independent test sets.

1.4 Terminology:

The following terms are used throughout this specification.

Bus cycle: A bus cycle is a basic unit of one bus clock period and for the purpose of AMBA AHB or APB protocol descriptions is defined from rising-edge to rising-edge transitions. An ASB bus cycle is defined from falling-edge to falling-edge transitions. Bus signal Timing is referenced to the bus cycle clock.

Bus transfer: An AMBA ASB or AHB bus transfer is a read or write operation of a data object, which may take one or more bus cycles. The bus Transfer is terminated by a completion response from the addressed slave.

Burst operation: A burst operation is defined as one or more data transactions, initiated by a bus master, which have a consistent width of transaction to an incremental region of address space. The increment step per transaction is determined by the width of transfer (byte, halfword, word). No burst operation is supported on the APB.

1.5 Features.

- Supports read, write, Retry and split operations.
- Supports single-edge clock protocol.
- Supports multiple bus masters and slaves.
- Supports burst transfers.
- Supports pipelined address phase operations.

AMBA APB provides the basic peripheral macrocell communications infrastructure as a secondary bus from the higher bandwidth pipelined main system bus. Such

peripherals typically:

- have interfaces which are memory-mapped registers
- have no high-bandwidth interfaces
- are accessed under programmed control.

The external memory interface is application-specific and may only have a narrow data path, but may also support a test access mode which allows the internal AMBA AHB, ASB and APB modules to be tested in isolation with system-independent test sets.

1.6 AHB AMBA with Multiplexer Interconnection.

AHB is a new generation of AMBA bus which is intended to address the requirements of high performance synthesizable designs. It is a high performance system bus that supports multiple bus masters and provides high bandwidth operation. AMBA AHB implements the features required for high-performance, high clock frequency systems including:

- Burst transfers.
- Split transactions.
- Single-cycle bus master handover.
- Single clock edge operation.
- Wider data bus configurations (64/128, up to1024 bits).
- SEQ, NON-SEQ, BUSY IDLE transfer types.
- Address decoding.

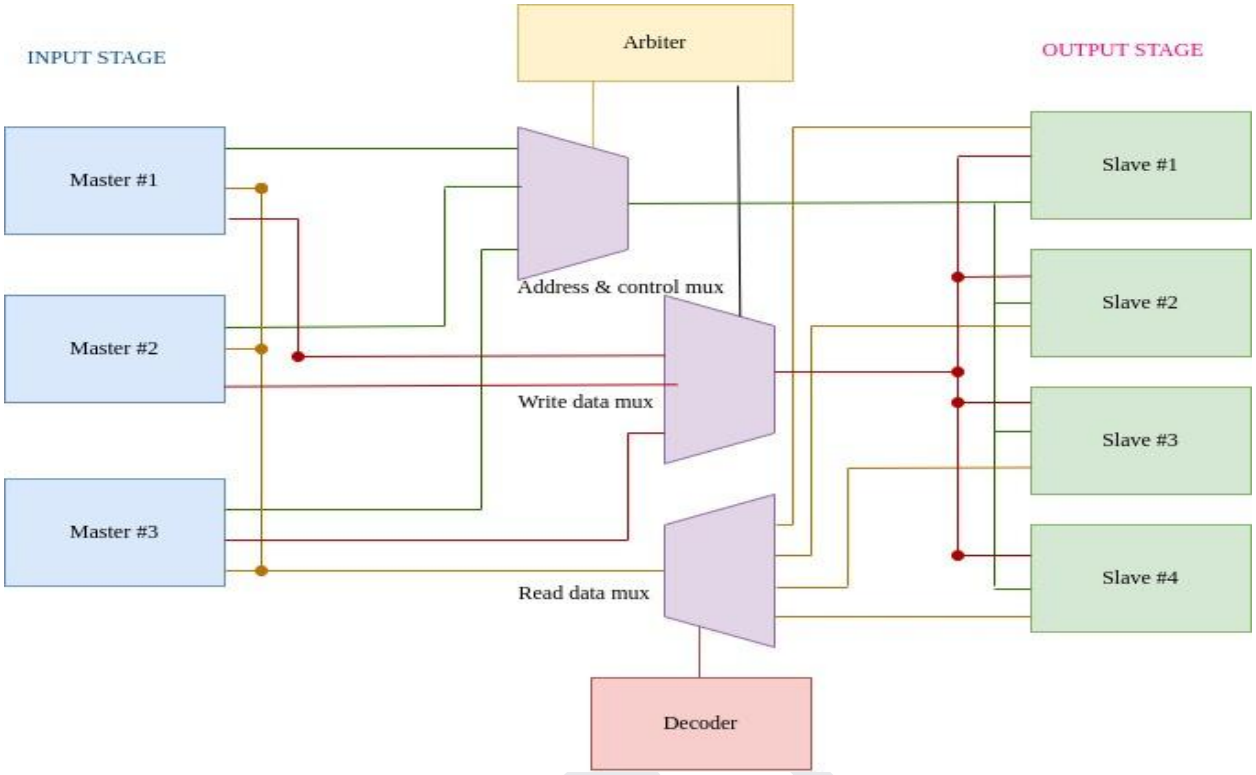

Figure 1.2. AHB AMBA with Multiplexer Interconnection.

An AMBA AHB with multiplexer is having following components:

AMBA AHB MASTER: An AMBA AHB bus master is able to initiate read and write operations by making use of an address and control information. Only one bus master at a time is allowed to actively use the bus.

AMBA AHB SLAVE: An AMBA AHB bus slave responds to read and write operation initialized by master within a given address space range. The bus slave signals back to active master about the success, failure or waiting of the data transfer.

AMBA AHB ARBITER: An AMBA AHB bus arbiter gives an assurance that only one bus master at a time is allowed to initiate the data transfers.Even though the arbitration scheme is fixed, any arbitration scheme can be used like Round Robin, Fair Chance etc. depending on the application requirement.

AMBA AHB DECODER: The AMBA AHB decoder is used to decode the address of each transfer and provide a select signal for the slave that is involved in the transfer. A single centralized decoder is required in all AHB implementations.

Figure 1.2 shows the working principle of AHB Bus. Before starting the AMBA AHB transfer, the bus master must have to be granted access to the bus. In this process first of all the master asserts a request signal to an arbiter. Now the arbiter will indicate when the master will get the grant of the bus. This decision of granting the access to the bus is achieved using some arbitration mechanism like priority based or round robin mechanism etc. A granted bus master then starts the AHB transfer by first driving an address and control signals. These address and control signals provide information about an address, direction and width of the transfer, burst transfer information if the transfer forms the part of the burst.

2. AHB Slave Operation with Block Diagram.

An AHB bus slave responds to transfers initiated by bus masters within the system. The slave uses a HSELx select signal from the decoder to determine when it should respond to a bus transfer. All other signals required for the transfer, such as the address and control information, will be generated by the bus master.

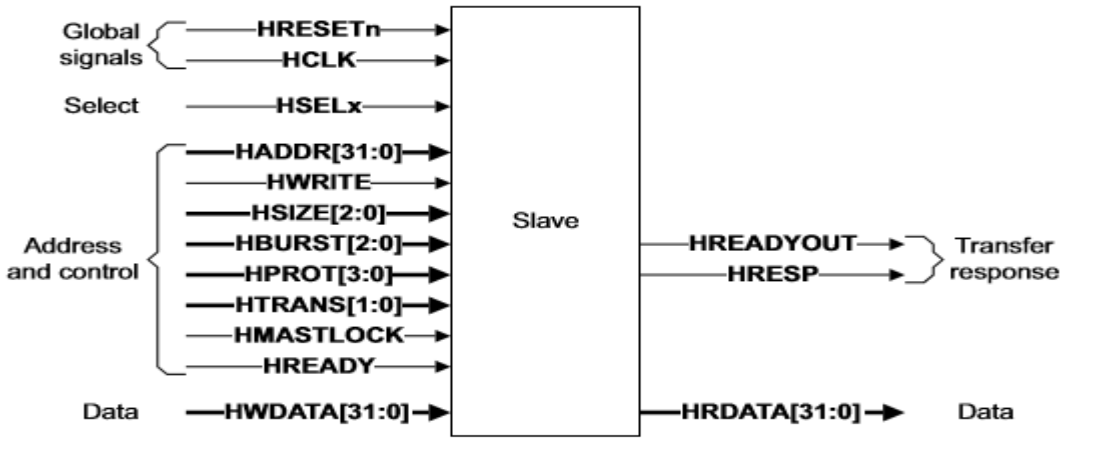

Figure 2.1 AHB Slave block diagram.

2.1 Overview of AHB Operations:

- ➢ Masters must be granted bus access before transfer.
  - Masters assert request signals .
  - Arbiter indicates when the master will be granted use of the bus.
- ➢ An AHB-bus transfer
  - Address phase: A single cycle in which the master drives the address and control signals indicate direction, width of transfer, and if the transfer forms part of a burst Each address correspond to 1-byte data (byte-addressable)
  - Data phase: One or more cycles controlled by HREADY from slave .The slaves sample and process the data
  - Address phase and data phase of different transactions are overlapped.
- ➢ Response HRESP [1:0]:
  - OKAY
  - ERROR
  - RETRY and SPLIT.

The master starts a transfer by driving the address and control signals. These signals provide information about the address, direction, width of the transfer, and indicate if the transfer forms part of a burst.

Transfers can be:

- single
- incrementing bursts that do not wrap at address boundaries
- wrapping bursts that wrap at particular address boundaries.

The write data bus moves data from the master to a slave, and the read data bus moves data from a slave to the master

2.2 Burst Operation.

- 4/8/16-beat bursts, as well as undefined-length burst's.
- Burst size indicates the number of beats in the burst.
- Valid data transferred = (# of beats) x (data size in each beat).

Data size in each beat is indicated in HSIZE[2:0]

- Incrementing bursts:

Access sequential locations with the address of each transfer in the burst being an increment of the previous address.

An incremental burst can be of any length, but the upper limit is set by the fact that the address must not cross a 1KB boundary.

- Wrapping bursts:

If the start address of the transfer is not aligned to the total number of bytes in a burst (size x beats) then the address of the transfers in the burst will wrap when the boundary is reached.

A 4-beat wrapping burst of word (4-byte) access.

3. AHB Slave Signal Descriptions:

All signals are prefixed with the letter H which represents the master signals.

- **HCLK:** This signal contains a width of 1-bit and it is driven by an external clock source. The data can be transferred at each rising edge of HCLK.

- **HRESET_n** : The signal width 1 bit and driven by reset source . rest is active low signal resets the system and the bus . This is the only active low AHB signal.

- **HBUSREQ:** Signal width is 1-bit driven by master ,A signal from bus master x to the bus arbiter which indicates that the bus master requires the bus. There is an HBUSREQx signal for each bus master in the system.

- **HGRANT:** signal width is 1-bit, which is driven by an arbiter. This signal indicates that bus master x is currently the highest priority master.Master gets access to the bus when both HREADY and HGRANT are HIGH.

- **HADDR [32:0]** : Width of this signal is 32-bit and driven by the master to assign address.

- **HLOCK:** Signal width is 1 bit and driven by master .When HIGH this signal indicates that the master requires locked access to the bus and no other master should be granted the bus until this signal is LOW.

- **HTRANS:** Width of this signal is 2-bit and driven by master. It indicates the type of the current transfer happening.

HTRANS[1:0]	Type	Description
b'00	IDLE	indicates that no data transfer is required. the master uses IDLE transfer when it does not want to perform a data transfer slave must always provide a zero wait state OKAY response to idle transfer. the transfer must be ignored by slave
b'01	BUSY	The busy transfer type enables the master to insert idle cycles in the middle of a burst . only undefined length burst can have a busy transfer as the last cycle of a burst slave must always provide a zero wait state OKAY response to idle transfer.
b'10	NON-SEQ	Indicates a single transfer or the first transfer of a burst. The address and control signals are unrelated to the previous transfer.
b'11	SEQ	The remaining transfers in a burst are SEQUENTIAL and the address is related to the previous transfer. The control information is identical to the previous transfer.

Table 3.1. HTRANS Signals

- **HWRITE:** Width of this signal is 1-bit it is also driven by the master. HWRITE controls the direction of data transfer to or from the master. Therefore,

  when:

  HWRITE is HIGH, it indicates a write transfer and the master broadcasts data on the write data bus, HWDATA[31:0]

HWRITE is LOW, a read transfer is performed and the slave must generate the data on the read data bus, HRDATA[31:0].

- **HSIZE :** Width of this signal is 3-bit and driven by master . It indicates the size of the transfer.

HSIZE[2:0]	Size (bits)	Description
3'b000	8	byte
3'b001	16	half-word
3'b010	32	word
3'b011	64	double-word
3'b100	128	4-word line
3'b101	256	8-word line
3'b110	512	-
3'b111	1024	-

Table 3.2. HSIZE Signals

The size is used in conjunction with the HBURST[2:0] signals to determine the address boundary for wrapping bursts.

- **HBURST :** Width of this signal is 3-bit, and driven by master. It indicates if the transfer forms part of a burst.

HBURST [2:0]	Type	Description
b'000	SINGLE	single burst
b'001	INCR	increment burst of undefined length
b'010	WRAP4	4-beat wrapping burst
b'011	INC4	4 beat incrementing burst
b'100	WRAP8	8-beat wrapping burst
b'101	INC8	8-bit incrementing burst
b'110	WRAP16	16-beat wrapping burst
b'111	INC16	16-beat incrementing burst

Table 3.3 HBURST Signal Description.

- **HWDATA [31:0] :** Width of this signal is 32-bit and driven by the master. It is used to transfer data from the master to the bus slaves during write operations.

- **HRDATA [31:0] :** Width of this signal is 32-bit and driven by slave. It is used to transfer data from slaves to the bus master during read operations.

- **HREADY :** Width of this signal is 1-bit and driven by slave. When high the HREADY signal indicates that transfer has finished on the bus. This signal may be driven low to extend a transfer (insert idle cycles) .
- **HSELx :** 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.

- **HRESP:-** Width of this signal is 2-bit and driven by slave. It provides additional information on the status of a transfer.

HRESP[1:0]	Response	Description
2'b00	OKAY	The transfer has completed successfully. the hready signal indicates whether the transfer is pending or complete
2'b01	ERROR	An error has occurred during the transfer. the error signal condition must be signaled to master so that it is aware of the transfer has been unsuccessfully A two cycle response is required for an error condition with hready being asserted in the second cycle
2'b10	RETRY	The RETRY response shows the transfer has not yet completed, so the bus master should retry the transfer. The master should continue to retry the transfer until it completes. A two-cycle RETRY response is required.
2'b11	SPLIT	The transfer has not yet been completed successfully. The bus master must retry the transfer when it is next granted access to the bus. The slave will request access to the bus on behalf of the master when the transfer can complete. A two-cycle SPLIT response is required.

Table 3.4. HRESP Signal Description.

When it is necessary for a slave to insert a number of wait states prior to deciding what response will be given then it must drive the response to OKAY.

4. Finite state machine for AHB Slave

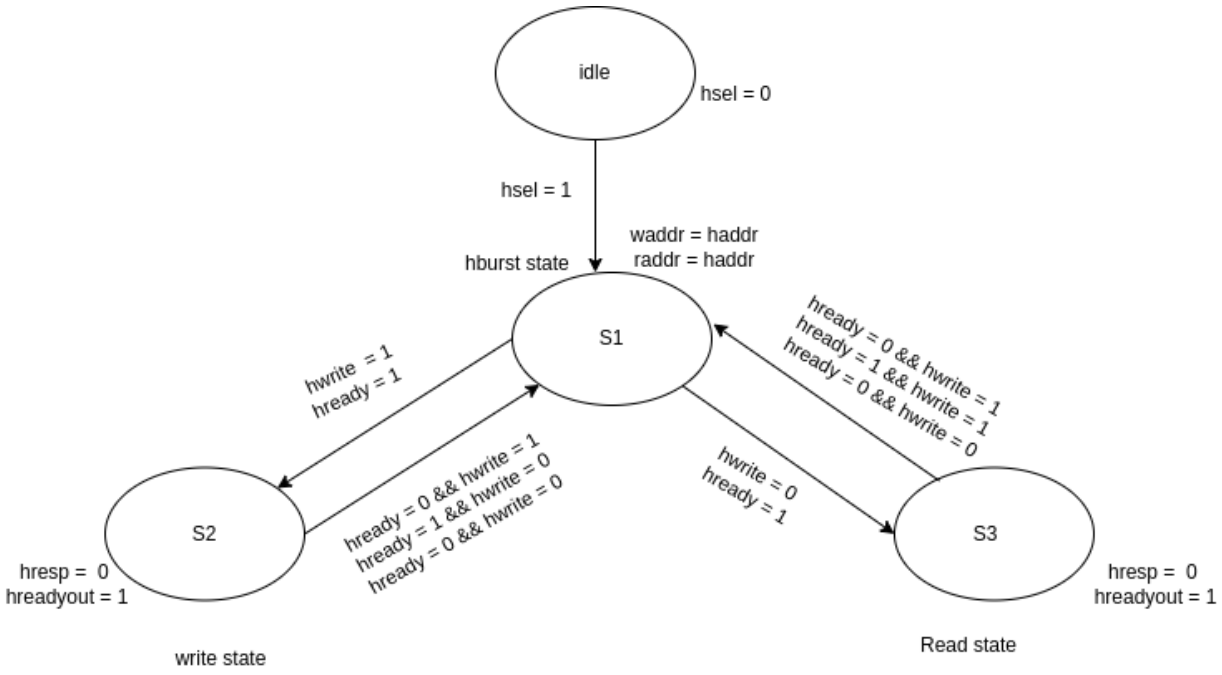

Figure 4.1 FSM for Write and Read operation.

The FSM for the AHB Slave – Burst Operation is developed based on its operation and is shown in the Figure 4.1.

In IDLE state hsel will be made high, slave will go S1 state, in S1 selects hburst, and haddr updated to waddr and raddr. when HWrite =1 and HReady = 1 slave will go S2 (WRITE) state and check for the address (waddr). The address will get increment only when the data is stored in the memory.When hready=0 , it goes S1 state or stay in same state.

In the same way, when HWrite = 0, slave will go S3 (READ) state and the data is fetched from memory. Once the fetching process is over, HReadyout is made high, HResp is set to low (OKAY). When hready=0 , it goes S1 state or stay in same state.

5. Verification goals :

- Minimum 32-bit transfer size is recommended. It supports only 32 bit size .
- During burst transfer hwrite and hready must be constant .
- When write=1 and hready = 1, it should perform a write operation and write the data into memory. if write=0 and hready = 1, it performs a read operation and read the data from memory .
- If hreadyout is high and hresp is low indicates that read or write transfer is completed and if hreadyout is low extend the transfer .
- It supports single , increment and wrap burst transfers.

6. Simulation Result :

Figure shows the operation of write operation for wrap 8 burst transfer, here master transfers 8 data's with one address then the address will get increment and data stored in the memory

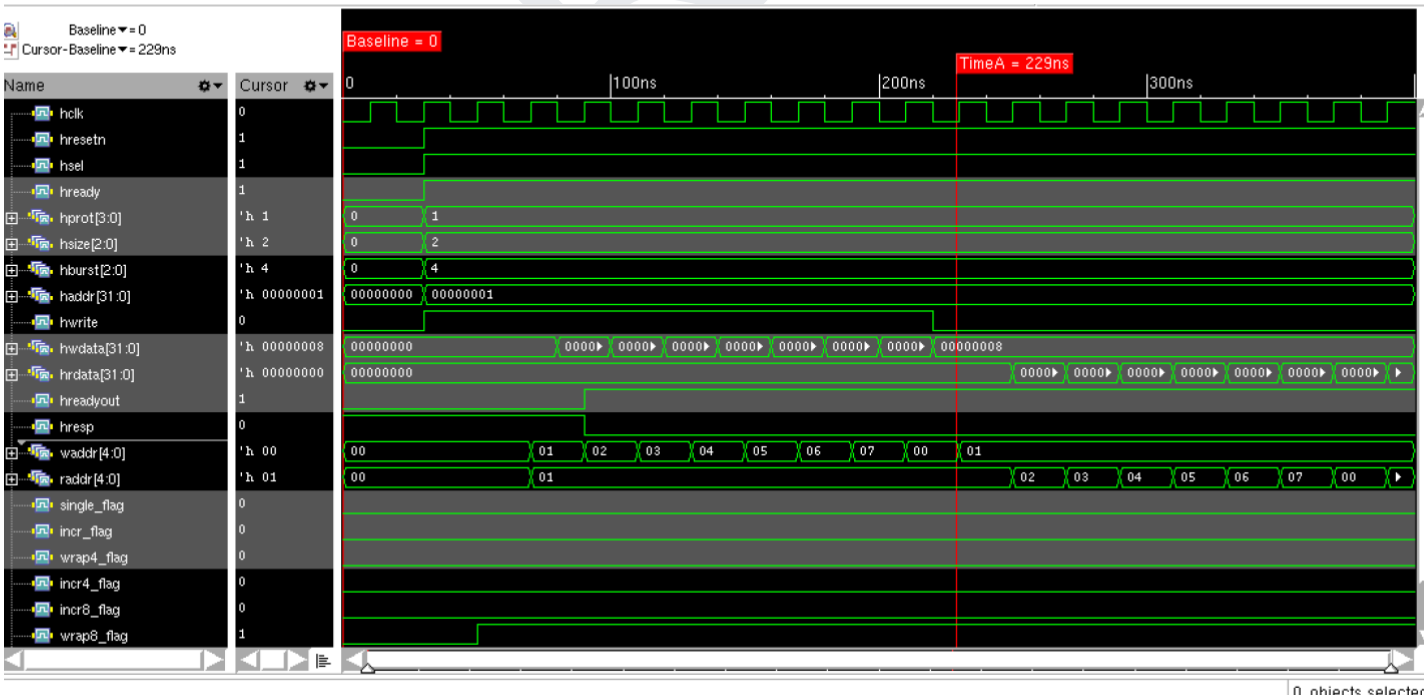

Figure 6.1. Simulation waveform for burst transfer.