

STUDY OF AMBA PROTOCOL AND VERIFICATION IPs

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF

MASTER OF TECHNOLOGY

IN

VLSI DESIGN AND EMBEDDED SYSTEM

Submitted by:

ANUKARNA SINHA
(2K20/VLS/02)

Under the supervision of

Prof. Neeta Pandey



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of
Engineering) Bawana Road,
Delhi-110042

MAY 2022

**DEPARTMENT OF ELECTRONICS AND
COMMUNICATION ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY**

(Formerly Delhi College of
Engineering) Bawana Road,
Delhi-110042

CANDIDATE'S DECLARATION

I, Anukarna Sinha, Roll No. 2K20/VLS/02 student of M.Tech (VLSI Design & Embedded system), hereby declare that the project dissertation titled “**Study of AMBA protocols and Verification IPs** ” which is submitted by me to the Department of Electronics and Communication Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation.

Place: Delhi
Date : 13.05.2022



**Anukarna Sinha
(2K20/VLS/02)**

**DEPARTMENT OF ELECTRONICS AND
COMMUNICATION ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY**


(Formerly Delhi College of
Engineering) Bawana Road,
Delhi-110042

CERTIFICATE

I hereby certify that the Project Dissertation titled “**Study of AMBA protocol and Verification IPs**” which is submitted by **Anukarna Sinha**, 2K20/VLS/02 (Department of Electronics & Communication Engineering), Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the student under my supervision.

Place: Delhi

Date: 13.05.2022


Prof Neeta Pandey
Department of ECE
DTU , Delhi

ACKNOWLEDGEMENT

A successful project requires not only the efforts of the person to whom the project is assigned, but also the assistance and supervision of others who assisted in the project's completion. I'd like to express my gratitude to everyone who has assisted us in this research and motivated me throughout my studies.

With profound sense of gratitude, I thank Prof. Neeta Pandey, my Research Supervisor, for her encouragement, support, patience and guidance in this project work. I heartily appreciate the guidance given by her in the project presentation that has improved my presentation skills with her comments and advices. I am overjoyed to express my gratitude to my family and friends for their assistance throughout this undertaking.

Anukarna Sinha

Anukarna Sinha

Roll no: 2K20/VLS/02

ABSTRACT

The ARM Advanced Microcontroller Bus Architecture (AMBA) is an open-standard interconnect protocol that connects and manages functional blocks in a system-on-a-chip design. It is easy to build multiprocessor systems with a large number of controllers and components using this bus design. High-performance microprocessors, high-level caches, memory management units, decoders, arbiters, controllers, and other IPs were provided by ARM. However, in order to connect them, an interconnect standard that is simple to design and build for low-power applications is required. The AMBA architecture is the de facto industry standard. Later, the standard was made accessible to the public, allowing others to design and integrate their own IPs. When establishing an interconnection standard, significant design concerns include chip pricing, latency, bandwidth, and the number of IPs that may be linked to the bus.

The report presents a detailed study of AMBA APB and AHB protocol and explains its features supported . It also explains about the various signals involved in the protocol .

The report also talks about the Intellectual Property and Verification Intellectual property and its importance in today's modern SOC technology used . In the semiconductor industry, IP (intellectual property) refers to a reusable unit of logic that is licenced to numerous suppliers for use as a building block in various chip designs. It could be logic, functionality, a cell, or even a layout design.

More and more functionality is being built into a single chip, resulting in SOC (silicon on chip) designs . Because these IPs are reusable units that may be reused an unlimited number of times once designed, they have become quite popular in the SOC. These IPs may include microprocessors, microcontrollers, and a variety of additional features.

JTAG is one of the VIPs which is used in the modern SOC's for verification . It can be used to verify the internal connection of chip as well as the interconnections between the chips when multiple chips are mounted on the board. The report includes the detailed study of JTAG and its register and after studying that the simulation results of bypass and idcode instructions are shown using Verdi tool .

TABLE OF CONTENTS

CERTIFICATE	v
ACKNOWLEDGEMENT	v
ABSTRACT	v
TABLE OF CONTENTS	v
LIST OF FIGURES	v
LIST OF TABLES	v
LIST OF ABBREVIATIONS	v
CHAPTER 1 INTRODUCTION	1-8
1.1 AMBA Protocol	1
1.1.1 Evolution of AMBA protocol	3
1.2 Goals of the thesis organization	7
1.3 Thesis organization	7
CHAPTER 2 STUDY OF APB AND AHB AMBA PROTOCOL	9-29
2.1 Advanced peripheral bus protocol (APB)	9
2.1.1 Operating states of APB	12
2.1.2 APB Write and Read cycle	13
2.1.3 APB ERROR response	16
2.1.4 APB limitations	18
2.2 Advanced High performance bus (AHB)	19
2.2.1 AHB master	21
2.2.2 AHB slave	21
2.2.3 Interconnect	22
2.2.3.1 Decoder	22
2.2.3.2 Multiplexor	22
2.2.3.3 Arbiter	22
2.2.4 Operation	23
2.2.5 Signal descriptions	25
2.2.5.1 Global signals	25
2.2.5.1 Master signals	26
2.2.5.2 Slave signals	26
2.2.5.3 Decoder signals	27
2.2.5.4 Multiplexor signals	27

2.2.5	Transfer of data	27
CHAPTER 3 VERIFICATION IPs		30 - 36
3.1	IPs and VIPs	30
3.2	Jtag	32
3.2.1	Working of JTAG	34
3.2.2	JTAG registers	35
CHAPTER 4 SIMULATIONS		37-38
CHAPTER 5 CONCLUSION AND FUTURE SCOPE		39
REFERENCES		40-41

LIST OF FIGURES

S.no	Figure details	Page no.
Fig 1.1	SOC system block diagram	1
Fig 1.2	Evolution of AMBA over time	3
Fig 1.3	AMBA bus architecture diagram	4
Fig 2.1	APB master	9
Fig 2.2	APB slave	10
Fig 2.3	Interfacing of APB master & slave	11
Fig 2.4	State diagram	12
Fig 2.5	APB Operating states in timing diagram	13
Fig 2.6	APB Write cycle timing diagram	14
Fig 2.7	APB Write cycle with wait states	15
Fig 2.8	APB Read cycle timing diagram	15
Fig 2.9	APB Read cycle with wait states	16
Fig 2.10	APB Error response for read cycle	17
Fig 2.11	APB Error response for write cycle	17
Fig 2.12	AHB block diagram	19
Fig 2.13	AHB lite block diagram	20
Fig 2.14	AHB Master interface	21
Fig 2.15	AHB Slave interface	22
Fig 2.16	AHB Read Transfer	28
Fig 2.17	AHB Write Transfer	28
Fig 2.18	AHB Read transfer with wait states	29
Fig 2.19	AHB Write transfer with wait states	29
Fig 3.1	An example of IP	30
Fig 3.2	Architecture of JTAG	32
Fig 3.3	System with boundary scan	33
Fig 3.4	Finite state machine TAP controller	34
Fig 4.1	Simulation result of bypass scan instruction	37
Fig 4.2	Simulation result of idcode instruction	38

LIST OF TABLES

Table no	Name of Table	Page no
2.1	List of APB Signals	12
2.2	Transfer types in AHB protocol	24
2.3	Burst types in AHB protocol	24
2.4	AHB Global signals	25
2.5	AHB Master signals	26
2.6	AHB Slave signals	27
2.7	AHB Decoder signals	27
2.8	AHB Multiplexor signals	27

List of Abbreviations

1. **SOC** - System on Chip
2. **AMBA** - Advance Microcontroller Bus Architecture
3. **APB** - Advance Peripheral Bus
4. **AHB** - Advanced High Performance Bus
5. **AXI** – Advanced Extensible Interface
6. **ASIC** - Application Specific Integrated Circuit
7. **IP** – Intellectual Property
8. **IC** - Integrated Circuit
9. **FPGA** –Field Programmable Gate Array
10. **RTL** – Register Transfer Level
11. **CPU** - Central Processing Unit
12. **GPU** - Graphics Processing Unit
13. **PS** - Processing System
14. **IOP** - Input Output Peripherals
15. **BW** - Bandwidth
16. **STA** - Static Timing Analysis
17. **VIP** – Verification Intellectual property
18. **JTAG** – Joint Test Action Group

19. UART – Universal Asynchronous receiver transmitter

20. USB – Universal Serial Bus

21. PLL – Phase Locked Loop

22. TAP – Test Access Port

23. TMS – Test Mode Select

24. TDI – Test Data Input

25. TDO – Test Data Output

CHAPTER 1

INTRODUCTION

1.1 AMBA PROTOCOL

The Advanced Microcontroller Bus Architecture (AMBA) is a well-known open standard, on-chip communicating standard, and on-chip connection for the alternate organizing and courting of squares in a design. AMBA as a structure aids a wide range of designs, regardless of how many controllers and peripherals they include. System on-chip designs (SOCs), Application-Specific Integrated Circuits (ASICs), and Anomalous state embedded tiny scale controllers all use AMBA for on-chip delivery [1].

The AMBA protocol is used for the interconnection and management of functional block in ASIC (application specific integrated circuit) and SOCs (silicon on chips). It helps in the development of multiple processor design connected with large number of peripherals and controllers . One of the main feature of AMBA protocol is that it is not dependent on the platform and also it is used along with any sort of processor architecture , for example , the Application processors which are used in devices such as IOT subsystem .

In a modern SOC design there is a high level integration of many design components including Intellectual property (IP) [2]. Micro controllers or microprocessors, as well as additional components such as internal memory or external memory bridge, all of which are present on a single platform(or chip) in current SOC. Figure 1.1 includes the block diagram of AMBA enabled SOC .

In the SOCs since there are many components present , there is a need to identify for an effective mechanism for these components to interact with each other . In order to fulfil these needs we need to study various protocols . Also, the type of protocol used affects the overall performance of the device so the correct selection of protocol according to our requirement is also important. AMBA protocol integrates well with the SOCs thereby improving its performance .

Usually the IP cores are designed with different interfaces and communication protocol which might lead to a problem when we integrate the SOCs . For avoiding such problems several standard on chip protocols have been developed. One more advantage of using AMBA is that it minimizes the amount of silicon needed for both on-chip and off-chip communications.

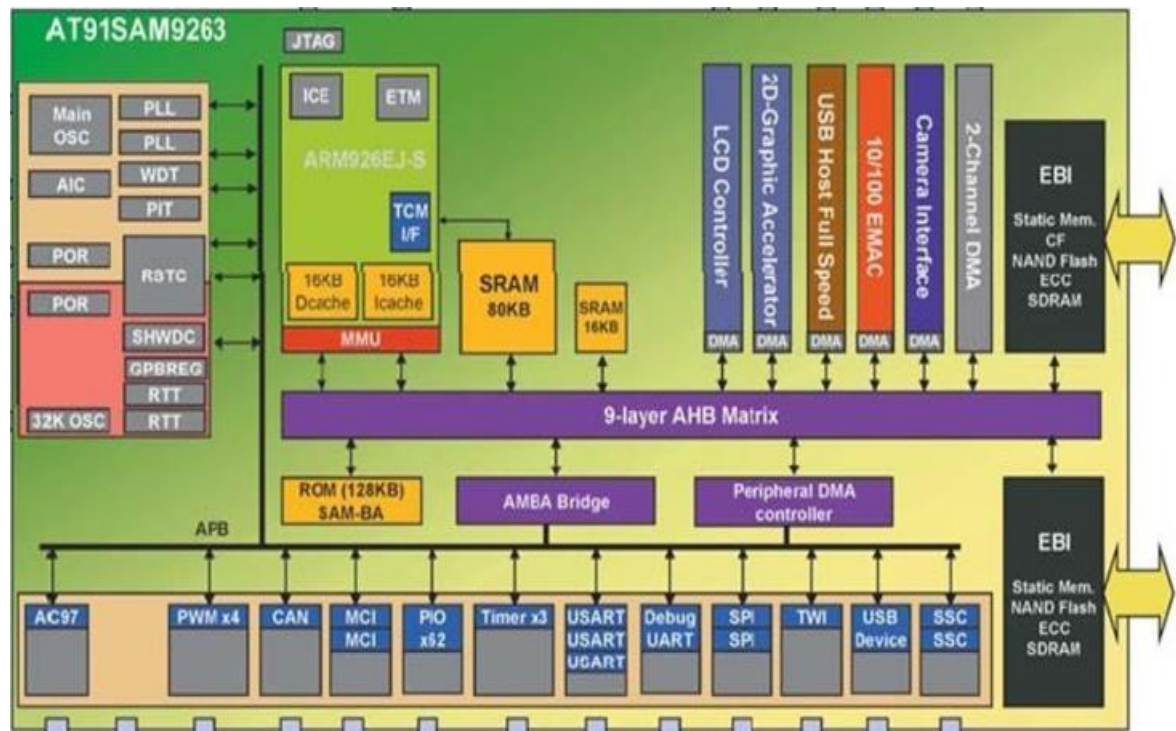


Fig 1.1 SOC system block diagram [2]

ADVANTAGES OFFERED BY AMBA PROTOCOL

Flexibility : AMBA provides with the ability to use it with various SOC's . In order to implement IP reuse we need a common standard as well as supporting various types of SOC's differing in area, power and performance requirements . AMBA offers various specifications to satisfy the needs of various SOC's .

Efficient IP reuse : IP reuse is a major component to reduce the development cost and time frame of the SOC's . To facilitate this the AMBA protocol supports with various interface standards that allows for IP reuse . Hence , many of ASIC and SOC's uses AMBA protocols.

Compatibility : AMBA supports with standard specifications allowing its compatibility among various IP components from vendors .

Latency: Latency stands for the delay among the initiation and completion of a transaction. In a burst-based system, the latency figure often refers to the completion of the first transfer rather than the entire burst. The performance of the interface relies upon the volume to which it achieves the maximum bandwidth with zero latency.

Support: AMBA is well supported. It is widely implemented and supported throughout the semiconductor industry, including support from third-party IP products and tools. Bus interface requirements like AMBA, are differentiated via the overall performance that they enable. The main characteristics of bus interface performance are:

Bandwidth: The speed at which data can be pushed across the interface. The maximum bandwidth in a synchronous system is limited by the product of the clock speed and the data bus width.

1.1.1 EVOLUTION OF AMBA

AMBA has evolved over the years to meet the demands of processors and new technologies, as shown in the fig 1.2 .

AMBA : Arm introduced AMBA in the late 1990s. During the first phases some of the protocols that were introduced are the Advanced Peripheral Bus (APB) and the Advanced System Bus (ASB). ASB has been superseded by more recent protocols, while APB is still widely used today.[1]

APB is non pipelined protocol so it is used for low bandwidth accesses. The bus has communication splitted into address and data phase and it involves signal list which are of less complicated nature .

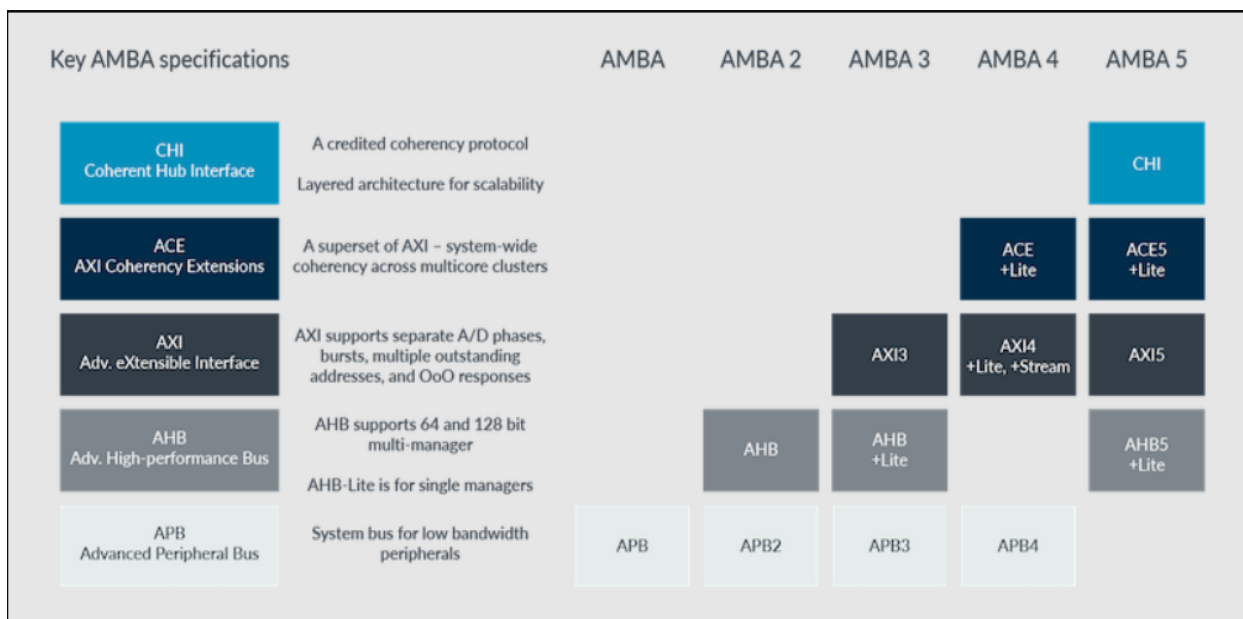


Fig 1.2 Evolution of AMBA over time [2]

AMBA2: The AMBA High-performance Bus (AHB), which supports one clock-edge transfers, was introduced in 1999 with AMBA 2. At the AHB, a simple transaction consists of an address phase after which the data phase comes. During the address phase the address of slave is sent and while the data phase is happening the data is sent. An arbiter is present to select one of the master for the transfers thereby access is provided to a single master at any given time. The APB protocol is designed as non pipelined protocol and hence is a simpler protocol but the AHB protocol supports pipelining for overall performance.[2]

AMBA3: The third generation AMBA protocol was introduced in 2003, AMBA 3, which has ATB and AHB-Lite[1]. The ATB protocol can be used for trace answer.

AHB-Lite could be a set of AHB. This set simplifies the planning for a bus with one master. Advanced extensible Interface (AXI), introduced during the the third generation of AMBA interface is a well known and used protocol which is specifically targeted for good performance , high speed system styles. AXI includes options that makes it suitable for good performance and higher speed interconnections on the chip .

AMBA4: The AMBA 4 development came into picture during the year 2010 , beginning with AMBA AXI4 followed by AMBA 4 AXI Coherency Extensions (ACE) during the year 2011.

ACE adds more signals to AXI, resulting in system-wide coherency. Multiple processors can share memory which is an added advantage supported under system wide coherency feature of AMBA .The ACE-Lite protocol offers unidirectional coherency at constant time.

The AXI4 protocol is designed for one-way data transfers from master to slave with reduced signal mapping from master to slave , that is good for implementation in FPGAs.

AMBA5: AMBA 5 CHI (Coherent Hub Interface) which was given within 2013 by ARM , to empower a superior and versatile framework on-chip innovation.[1]

Figure 1.3 includes AMBA bus architecture showing high performance ARM processor , bridge , Timer , AHB or ASB protocol used for on chip connections , DMA bus master , UART, and an external memory interface .

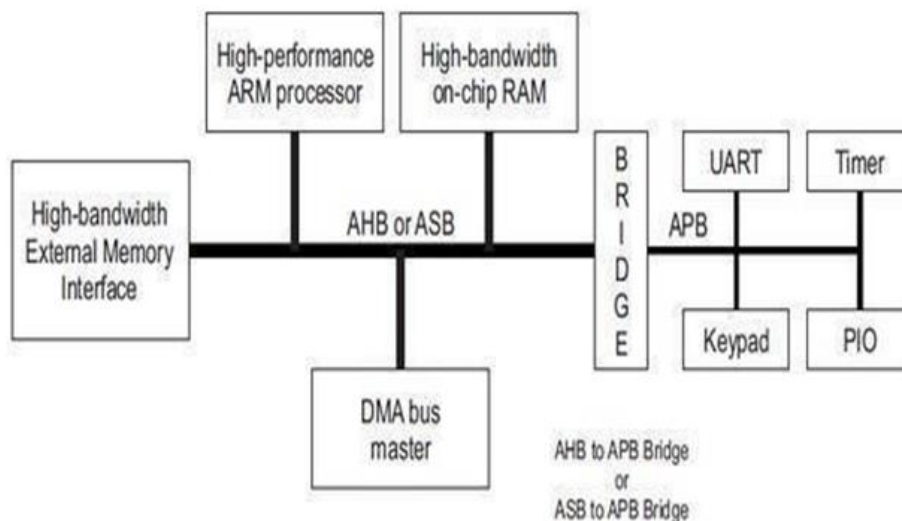


Fig 1.3 Amba bus architecture diagram [7]

Advanced Peripheral Bus (APB)

Advanced peripheral bus forms the subset of AMBA based protocol design which helps in reducing

the complexity majorly in terms of interfacing as well as power utilization and ease of understanding since it involves only a single master. It is used at places where low transfer speeds is sufficient for connecting the peripherals. For the transportation the main segments used are APB bridge and APB slave. The master is present for single transportation in AMBA based outlines. Also , in the case of APB the framework execution is improved .

Address hooking, a strobe flag PENABLE, and a choice flag PSELx are the main components of APB Bridge. PCLK , PWRITE , PRDATA, PRESETn, PWDATA, PADDR [31:0], PENABLE and PSELx are the fundamental flags that govern APB's task. PREADY and PSLVERR, in addition to alternate signs of APB, are used in these two supplementary signals. PWRITE is an important signal to determine whether the master would be writing the data on the slave or it would be reading the data from slave . For data that is to be written on to the slave , PREADY signal must be high . If the PREADY signal is low it means that wait states are inserted and the master has to wait for that many number of cycles till PREADY goes high .

Advanced System Bus (ASB)

The ASB protocol is mostly incorporated into areas where various microcontrollers are inserted for pipelined transport . This protocol helps in connecting various processors, external memory interface and on chip memories . This protocol offers several advantages including burst exchange , transport matter assistance and a better pipelined activity.

ASB master , ASB judge , ASB slave and ASB decoder would constitute the major portion of the protocol. The ASB master generates browsing workouts using the address and control information .There are various transportation professionals available in case of ASB but only one of them is granted access and ASB choose assists professionals in gaining access to ASB . There are different type of exchanges that can happen through ASB namely non sequential , sequential and address only. BWAIT, BLAST, BnRES, BWRITE, BTRAN[1:0], DSELx, BPROT [1:0], BSIZE [1:0], BLOK, BERROR, BD[31:0], BA[31:0], BCLK. AGNTx, and AREQx are characteristic signals used in this transport .

Advanced High-Performance Bus (AHB)

AHB is a widely used protocol of the AMBA family due its features like pipelining , multi master multi slave support .Due to its pipelining feature it supports increased data transmission activity.

It also provides several other highlights such as split exchanges, higher information transport setup, burst exchange, activity which works on single clock changes . AHB is typically utilized on ARM7, ARM Cortex-M, and ARM9-based designs. AHB master, slave, decoder, arbiter and AHB authority forms the part of AHB framework setup. It supports transfer of data divided into two phases: setup phase followed by access phase. During setup phase the information of address is passed and in access phase data information is sent.

Although AHB supports multiple master, for any particular instance of time only one master is granted the right to read from or write to the slave. The arbiter chooses which master will be granted access and that master is selected for transmission. It supports four different types of transfer namely idle, sequential, non sequential and busy . HSELx ,HWDATA [31:0], HREADY, HRDATA [31:0], HRESP [1:0], HMASTLOCK , HGRANTx , HSPLITx [15:0], HLOCKx, HMASTER [3:0] and HBUSREQx are the major signals in AHB protocol. AHB-Lite v1.0 features a high-speed exchange motion. Apart from the basic AHB signals, this adjustment employs a variety of banners to increase activity .

Advanced Extensible Interface (AXI)

AXI V1.0 was launched during the third phase of AMBA by ARM which is a burst based convention. It provides higher level of execution , a higher rate of return and a faster task . The data and address information is taken is taken in two phases . It also supports burst transfer of data . Some distinctive classifications in AXI includes compose information carrier signal , compose address ,carrier signal , compose reaction carrier signals , read address carrier signal , read information carrier flags and low power interface signals . AXI4 light is a more powerful version of AXI , which modifies the essential AXI indications and the most recent modification of AXI is AXI-stream v1.0 .

AXI Coherency Extensions (ACE)

An AXI upgrade that includes third-level reserves, peripherals, on-chip RAM, and external memory. The AXI read and compose channels are built for a 64-bit or 128-piece interface in this case. It is the foundation for 1:1 clock proportions in processor clocks. It will also run with a large number of CPU clocks. Professional is an ARM Cortex-A processor that includes the Cortex-A7 and Cortex-A15. ACE lite masters, ACE masters, and ACE Lite/AXI slaves are interconnected portions within the master. It offers coherency at the framework level a structure. The essential flags of ACE are read data channel signals, browse address carrier signals, snoop carrier signals, write address carrier signals, and response signals.

Advanced Trace Bus (ATB)

To debug the framework, the ATB supports an interchange of information around the Core Sight. It supports bite-sized bundles and, as a result, the control signals used to display the number of bytes significant in each cycle. This mode of transportation uses the signals ATCLKEN, ATCLK, ATREADY, ATRESETn , ATID[6:0], ATVALID, AFVALID, ATBYTES [m:0], knowledge [n:0], and AFREADY.

1.1.2 GOALS OF THE AMBA IMPLEMENTATION

The goals of the AMBA implementation are :

- It will enhance the reusability of IPs , make the design simpler and reduce the time to market of SOC .
- It will configure fewer potential interfaces .

1.2 VERIFICATION IP

In the semiconductor industry, IP (intellectual property) refers to a reusable unit of logic that is licenced to numerous suppliers for use as a building block in various chip designs. It could be logic, functionality, a cell, or even a layout design.

Due to the rising popularity of IP designs, a demand for Intellectual Property Verification arose (VIPs).

The Verification IPs, like the IPs, are pre-defined functional blocks that may be introduced into test benches to check a specific design.

One of such verification intellectual property is JTAG which is used for verifying a design . The JTAG is used to verify designs and test printed circuit boards after they have been manufactured. When many chips are put on the board, it can be used to verify the internal connection of each chip as well as the interconnections between them. There are many public instruction out of which IDCODE and BYPASS instructions are implemented and the simulations results are attached .

1.3 THESIS ORGANISATION

The thesis organisation is made up as :

Chapter 1 deals with a brief introduction AMBA protocol and its use in the modern SOC . It also explains the evolution of AMBA protocol . It briefly explains about the major AMBA evolved protocol like APB , AHB and AXI .

Chapter 2 provides a deep knowledge on APB and AHB protocol citing the various signals involved in the master and slave . Also, explains the features of this protocol in detail. It provides an insight to which protocol might be suitable at any situation on the chip according to the features which are supported by them .

Chapter 3 provides the knowledge of IP and VIPs and its importance in verification environment .It talks about JTAG VIP in details .

Chapter 4 provides the simulations results and relevant analysis.

Chapter 5 provides conclusion and the future scope of the project .

CHAPTER 2

STUDY OF APB AND AHB AMBA PROTOCOL

2.1 Advance Peripheral Bus Protocol (APB)

In the current VLSI design environment, the desirable features are high speed, complex functionality and shorter time to market. To promote these features a reuse based SoC design is required and the APB protocol helps in promoting these features. In the APB protocol interfacing for transfer of data is achieved between a single master and there are multiple slaves present. It is used at places where low bandwidth peripheral interconnect and low power consumption is a requirement. It is an unpipelined protocol indicating that it cannot have any outstanding transactions (second transaction will start only when the response of the first transaction has come). The APB protocol can be used to interface with the AHB and AXI protocol as well. The transfer of data is achieved with the help of various signals and the important ones among them being the address signal, select signal, write signal, ready signal and error signal. PSLVERR and PREADY are the two main signals which protect lost data when the transfer of data is occurring. The detailed description of these signals can be found in table 2.1. The APB block diagram can be found in figure 2.3 showing the master slave communication. It consists of APB master and slave, unlike the AHB protocol where additional components like arbiter, multiplexor are required.

APB Master :

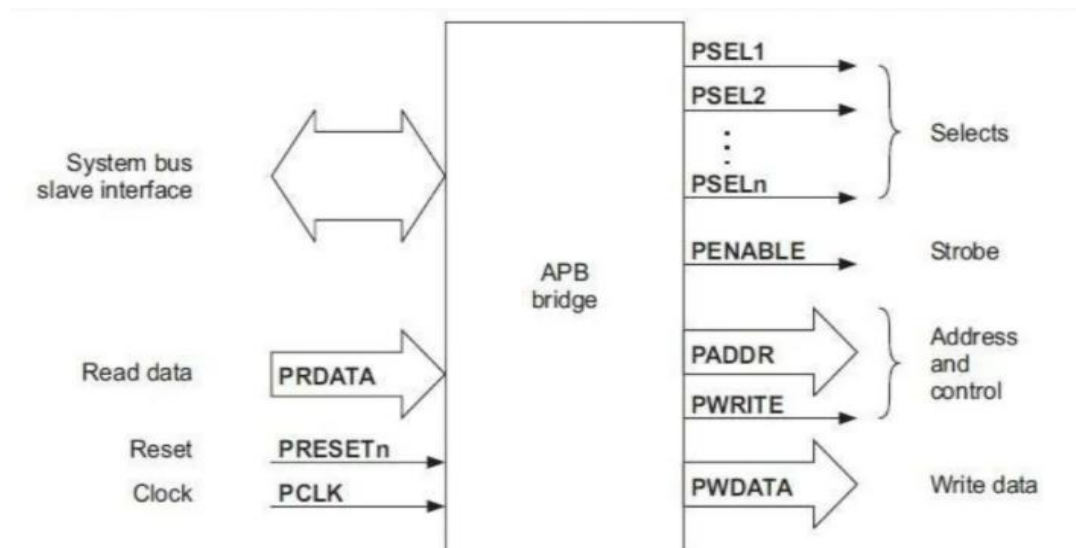


Fig 2.1 APB master interface [3]

Fig 2.1 includes the block diagram of APB master . It has single bus master and hence there is no need of arbiter . In the APB protocol, the master is in charge of driving the address and write buses, as well as performing combinatorial decoding of the address to determine which PSELx signal is active. It's also in charge of controlling the PENABLE signal, which is used in timing of the transfer. During a read transfer, it transfers APB data to the system bus.

APB Slave

APB slave supports a very flexible and simple interface. Fig 2.2 shows APB slave block diagram. PSELx signal is used to select one of the slaves out of others available slaves . PWDATA is the signal that master uses to write data onto the slave whereas PRDATA is the signal that slave uses to write onto the master. In this the two main signals which are used to protect the data from getting lost when the data transfer is occurring , is PSLVERR and PREADY . A detailed analysis on PSLVERR is given in the section 2.1.3 .

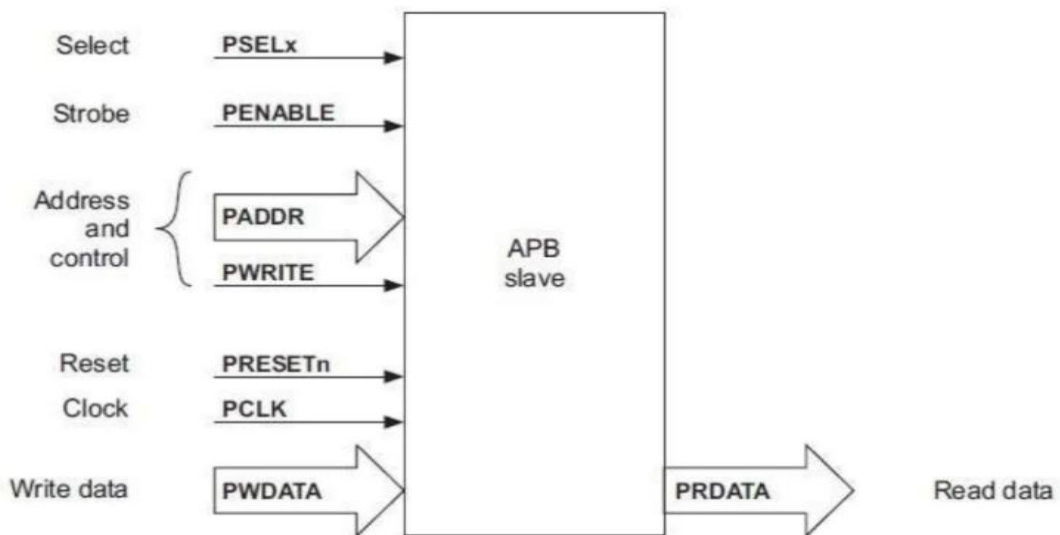


Fig 2.2 APB slave interface [3]

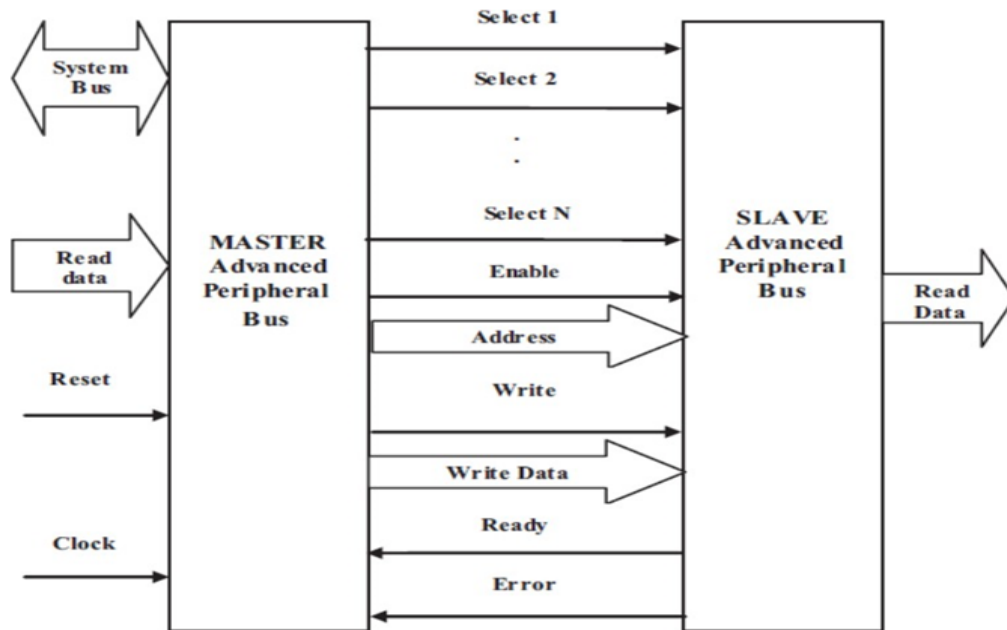


Fig 2.3 Interfacing of APB Master & Slave [5]

For the APB communication to occur, firstly proper select signal is given to choose the slave from which information exchange has to happen (as there are multiple slaves present). After establishing the connection we check for the PWRITE signal, if PWRITE is high it indicates master would be in the charge of transferring the information to the slave and when PWRITE is low, it indicates that the transfer of information is scheduled from the slave to the master on the PRDATA signal.

The transfer of data occurs in two cycles, one cycle is known as the setup phase and the second cycle is known as the access phase. When it is in the first phase the address information is being sent, following which we have the access phase in which the data is sent. If the slave holds PREADY low during the first cycle of the access phase, the peripheral bus would be remaining in the second phase until it is driven high. If the slave drives PREADY high in the second cycle of the access phase, the access state ends and the bus would be returning to the idle state when no additional transfers are necessary. When the requirement for other transfer arises, the bus would again be driven to the setup phase and the cycle repeats.[2]

SIGNALS	SOURCE	DESCRIPTION
PCLK	Clock source	Clock. All signals changes occurs on its edges (positive or negative)
PRESETn	System bus equivalent	Reset. The APB reset signal is active low
PADDR	APB Bridge	APB address bus. It can be exceeded till 32 bits wide

PSELx	APB Bridge	Select . It is used to select the slave from which the data transfer has to happen
PWRITE	APB Bridge	Direction .When high it indicates data transfer direction from master to slave and when low it indicates data direction from the slave to the master .
PWDATA	APB Bridge	Write data . It is peripheral bus on which the data is sent from the master and it can be exceeded uptill 32 bits wide in size
PREADY	Slave interface	Ready . It is utilized by slave to insert wait states in the transaction .
PRDATA	Slave interface	Read data . It is driven by slave to transfer the data to the master whenever the pwrite is driven as low . It can be upto 32 bits wide .
PSLVERR	Slave interface	It indicates that some error has occurred during the transfer and the data might not have been faithfully recovered
PENABLE	APB Bridge	Enable . It denotes the start of access phase

Table 2.1 List of APB signals [5]

2.1.1 Operating states of APB

There are a total of three states in APB : idle , setup and access. Idle is the most common condition of the APB .The bus enters the SETUP state and asserts the relevant select signal PSELx where a transfer is necessary .

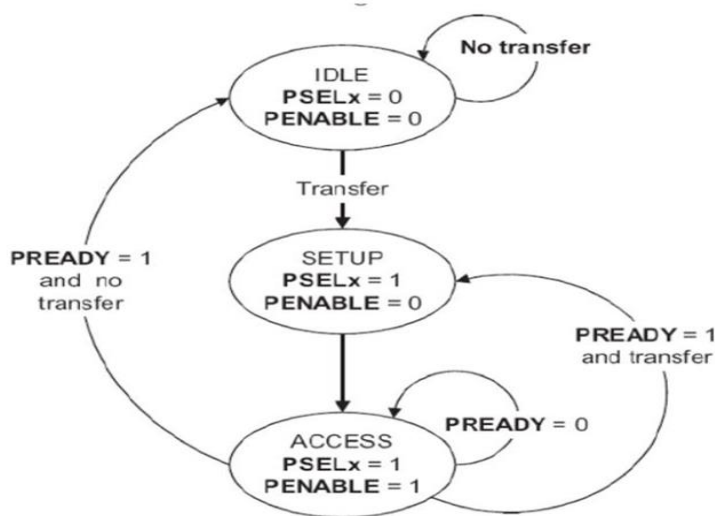


Fig 2.4 State Diagram of APB

In the first clock cycle the bus would remain in the setup phase , after which it will enter into the access state whenever the next clock signal would arrive . During the access state , Penable signal would be asserted high . All the other signals such as the Pwrite, Pwdata , select, and address signals should be staying constant whenever the transition is occurring from the first phase to the second phase . The access state is known to control when it needs to depart when the slave gives the Pready signal. The first criterion is that the peripheral bus would be remaining in the access phase whenever the slave has PREADY asserted as LOW. It also remains in the setup phase when Pready is kept in the high state by the slave and after the transfer it moves to the setup phase . If no additional transfers are present, then the slave drives PREADY HIGH, exits the ACCESS state, after which it returns to the IDLE state. The cycle is then repeated. [2][3].

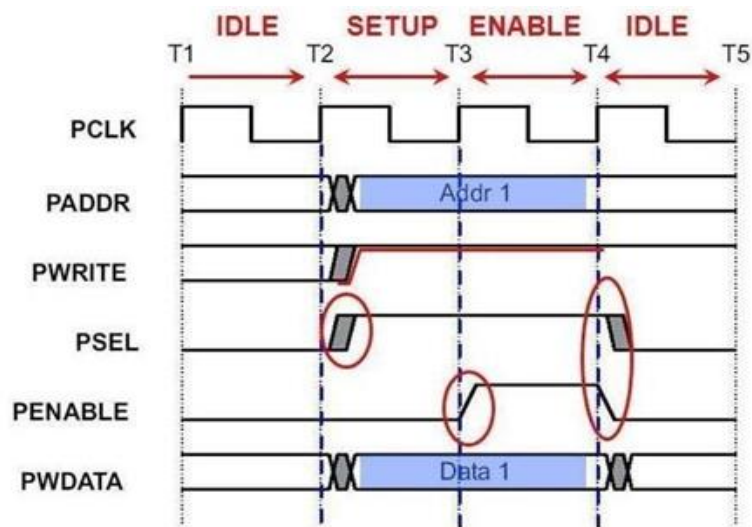


Fig 2.5 APB Operating states in timing diagram [2]

Fig 2.5 shows general timing diagram including the various operating states . Initially it is in the idle state . When PSELx is asserted ,it marks the start of setup phase in which the address is sent . When PENABLE goes high it enters the access phase in which transfer of data starts . When the transfer is complete the PENABLE goes low indicating the end of access phase . Again it enters the idle state and the loop continues .

2.1.2 APB Write and Read cycle

Write cycle :

When the master wants to write data on to the slave , the PWRITE signal must be high . The PCLK signal indicates the clock and Pwdata, Paddr , Pwrite and Psel signal changes are sampled at the positive edges of clock .

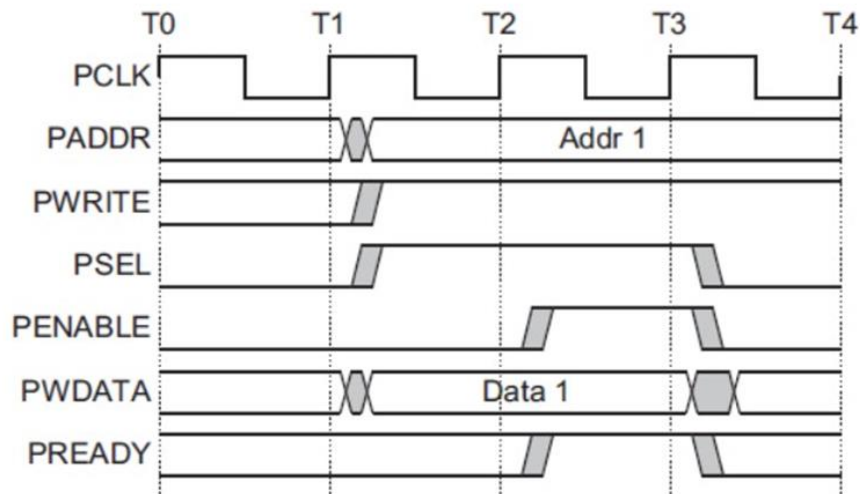


Fig 2.6 APB Write Cycle Timing diagram [2]

PSEL goes high at T1 indicating the starting of transaction. The first cycle is setup phase asserted by the master and after one clock cycle PENABLE goes high and now transfer of data takes place and after the data transfer has occurred, PENABLE will go low. PADDR should remain valid throughout the PSEL assertion.

From fig 2.6 we can see that at time instance T1, the setup phase begins at the positive edge of PCLK, and the information about the address of slave is sent in this phase. After successfully sending the address information at the time instance T2 Access phase begins. At the rising edge of the clk (T2), the signals Penable and Pready are sampled. Since the PREADY is high it shows the readiness of the slave to accept the data transfer and there are no wait states inserted by the slave. During the next rising edge (T3), the Penable signal goes low indicating the completion of transfer and it marks the completion of access phase. When the transfer gets completed, the Penable signal would be disabled. Similarly, the signal PSEL is likewise suppressed at the end of transfer but in situations where immediate transfer has to be made to the same slave device it will continue to be in the same state [2-3].

The figure 2.7 shows write cycle with two wait states inserted as the PREADY signal goes high after two clock pulses.

When the PENABLE signal is high during the access phase, the slave extends the transfer by driving the PREADY signal low. While the PREADY signal is asserted low, the PADDR, PWRITE, PSEL, PENABLE, and PWDATA should remain stable. When PENABLE is low, PREADY can take any value. It is advised that the address and write signals be left unchanged until another access occurs.

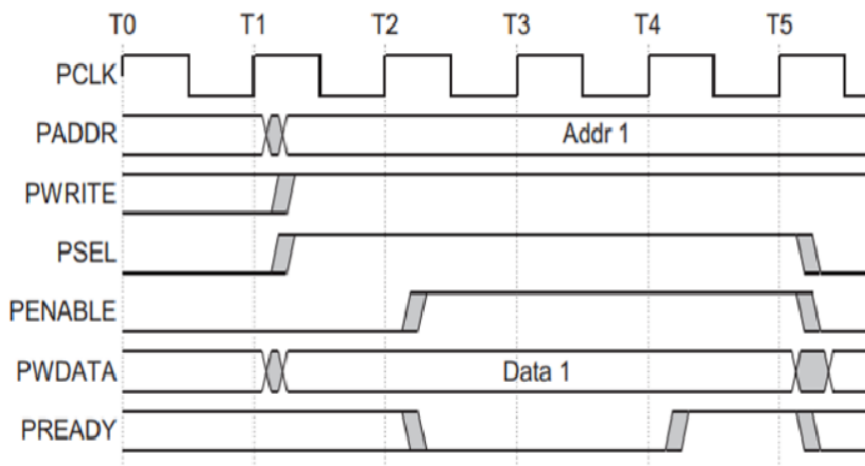


Fig 2.7 APB Write cycle with wait states [2]

Read cycle :

When the PWRITE signal goes low it indicates read cycle (slave wants to write the data to master and master reads it). The Psel ,Penable , Paddr and Pwrite signals are the signals that are sampled at the rising edge of clock (T1) during the setup phase . The rising edge of the clock at T1 marks the start of transfer .

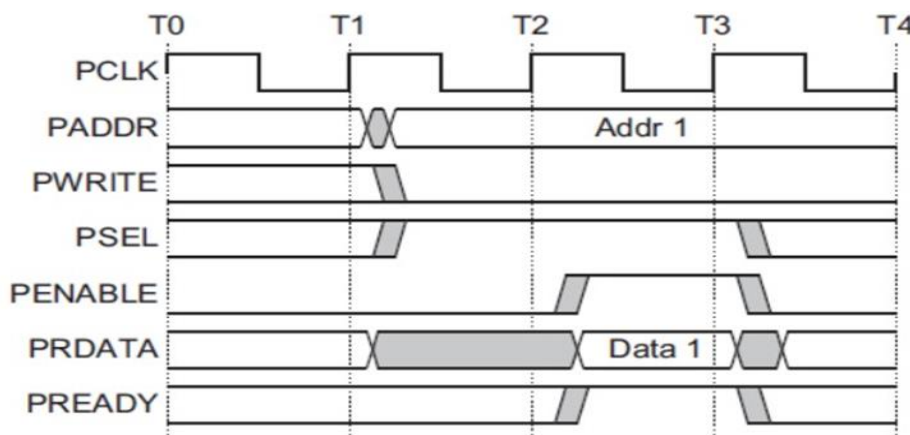


Fig 2.8 APB Read cycle timing diagram [2]

From fig 2.8 we can observe that the PENBLE signal goes high at the T3 clock cycle indicating the start of access phase . Since PREADY is driven high at T2 clock cycle it indicates that there are no wait states inserted by slave here . The PRDATA signal is driven in tandem with the PREADY signal. When asserted, PREADY signals that the slave can complete the transfer by providing the data on PRDATA on the next rising edge of PCLK. Before the completion of read transfer, i.e. before the T3 cycle, the slave must ensure data transfer.

Similar to the write cycle, even in the read cycle wait states can be inserted when PREADY signal goes low as shown in fig 2.9 . When PREADY is low the other signals like PADDR, PWRITE, PSEL, PENABLE, PRDATA must be held stable until PREADY goes high for successful read of data to occur .

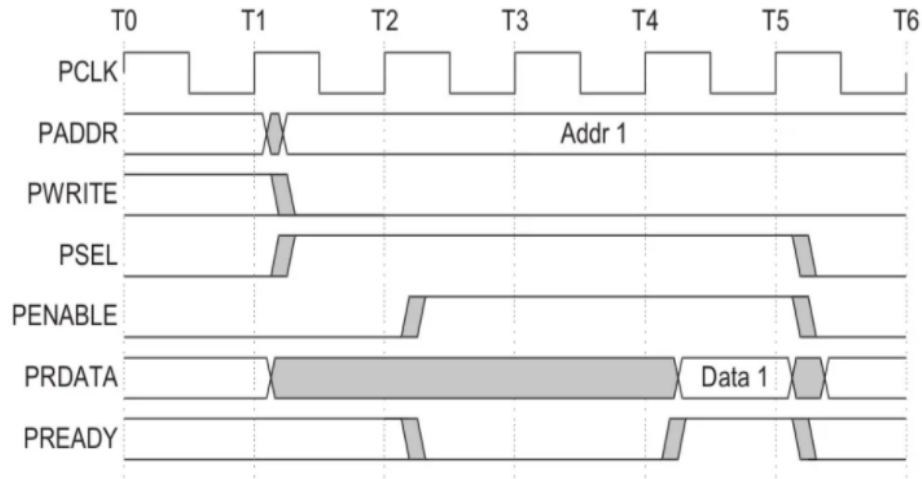


Fig 2.9 APB Read cycle with wait states[2]

2.1.3 APB Error Response

If an issue occurs during data transmission, the slave asserts the PSLVERR signal, which shows the error response for the transfer. When the PENABLE, PSEL and PREADY signals are all high when transfer of data is achieved and the transfer is at the last phase , PSLVERR is considered valid and legitimate. When the PSLVERR is not being sampled, it is also advisable to drive it low. Fig 2.10 shows the error response for the read cycle . The PSLVERR is high indicating that error response and data is not faithfully received by the master.

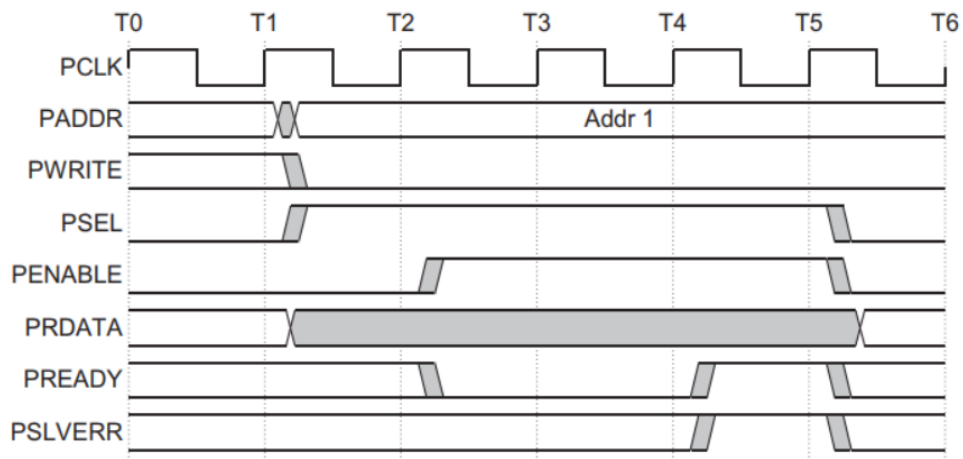


Fig 2.10 APB Error response for read cycle

The status of peripheral may or may not have been modified by transactions that generate an error response. It is not guaranteed that data will not be written on the slave peripheral if an APB master initiates a write transaction to an APB slave and receives an error response.

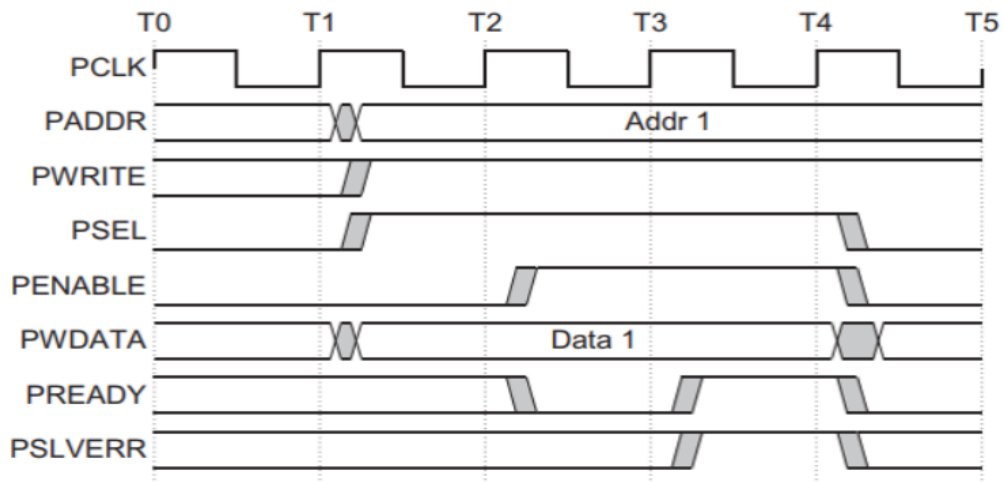


Fig 2.11 APB Error response for write cycle

Fig 2.11 shows error response for write cycle , indicating that a successful write of data from master to slave could not occur . It is a one cycle error response .

PSLVERR mapping to AHB / AXI

If we have a system where the initiator is a AHB initiator but slave is supporting only APB protocol so there needs to be a bridge between them for converting AHB to APB and similarly map the response also . In another scenario , if initiator is supporting AXI and issuing a burst transfer of 8 bits (it will complete 8 transfers) but if slave side is APB (single transfer) so bridge will break the transaction into 8 different transactions to slave interface . When we are getting error response from APB slave we must be able to map it to AXI and AHB error response .

From AHB to APB

For transfers between initiator supporting AHB protocol while the receiver supporting APB protocol mapping is achieved by mapping PSLVERR signal to HRESP (2 bits) for both the read and the write operation .

From AXI to APB

For transfers between initiator supporting AXI protocol while the receiver supporting APB protocol mapping is achieved by mapping PSLVERR to the signals RRESP[1] for reads and BRESP[1] of AXI for write transfers .

2.1.4 APB limitations

Some of the limitations of APB protocol are

- Low bandwidth protocol so it does not support multiple transfers at a time
- Cannot be scaled in terms of performance .
- A single outstanding request causes a bottleneck in throughput and multi-threading performance.

2.2 Advanced High- performance Bus (AHB)

The AHB protocol is a new protocol that provides some additional benefits over the APB protocol and incorporates the capabilities needed for high-performance, high-clock-frequency systems including :

- Pipelined transfer : It need not wait for the completion of the previous transfer i.e while the data of first cycle is being transmitted the address of the next cycle can be transferred .
- Burst transfer : It offers more data words read/write access in a single access .
- Split transaction : If the slave is unable to complete the response of a given transaction it may split the response into multiple phases .
- Single clock edge operation .
- Data bus is made wider and it can support upto 64/128 bit wide data .

Due to these advantages over APB protocol , AHB protocol is widely used . The various interconnection components of AHB protocol are master, slave , arbiter , decoder and multiplexor .

In a typical multi master and multi slave scenario AHB protocol is one of the best supported protocol. Any master should be able to connect to any slave but at any given time only one master is given access by the arbiter to communicate with the slave and hence the need for arbiter arises which was not present in the case of APB protocol.

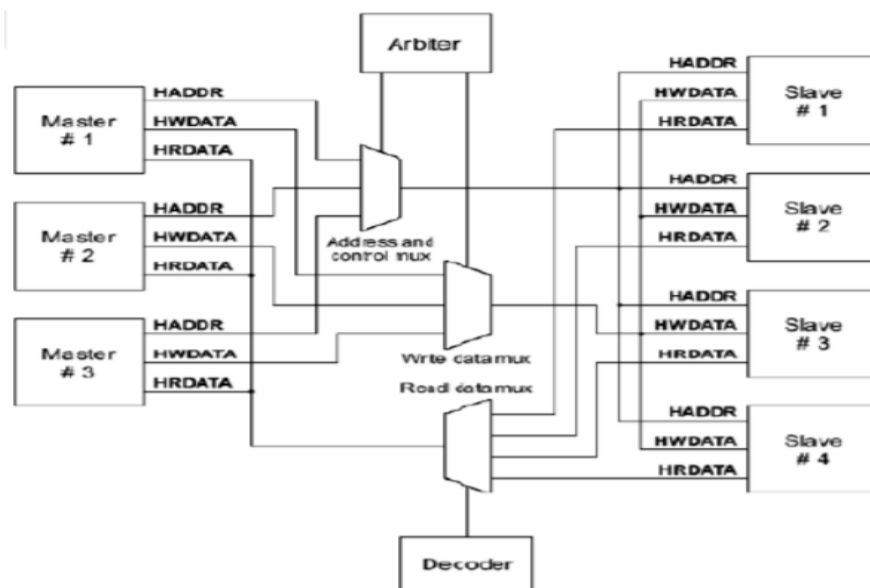


Fig 2.12 AHB Block diagram

In AHB to start any transaction , the approval of arbiter is needed and after the grant is given the transaction can start . Hselect signal is given for each slave available and for whichever it is high that will receive the data from master . AMBA AHB bus protocol is designed to be connected with a central multiplexor subsystem . Fig 2.12 depicts the block diagram of AHB transmission of data . There are multiple master and depending on the arbiter a slave is selected for the transfer . Two multiplexors are used , one for the address and the other for data transfer . The transfer of data from slave to master is controlled by a mux and a decoder .

AHB Lite is advanced version of the AHB protocol in which there is a single master present and hence there is no requirement of arbitration making the design simpler. However, there is a requirement of decoder to select the slave . Fig 2.13 shows the block diagram of AHB lite .

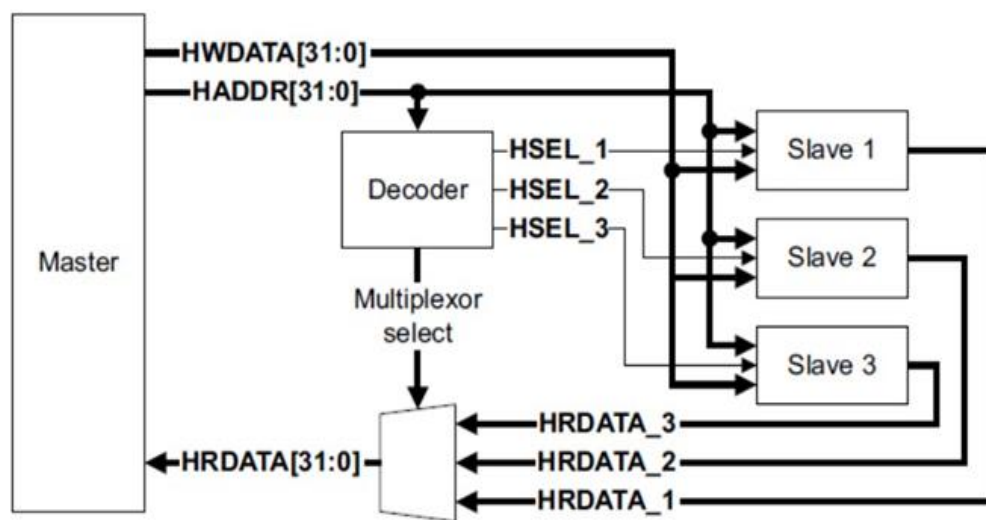


Fig 2.13 AHB Lite block diagram [16]

- It defines for a single master and multiple slaves.
- Decoder provides select signal to slaves and control signals to the multiplexor
- Multiplexor is used to route response from slaves to master.

The transfer is similar to the transfers done in AHB . No request/ grant handshake is required as only single master is present . All sizes , transfer types and burst type are same as done in AHB . Slave uses HREADYOUT to insert wait states . Only okay and error responses are supported by slave and error response has the same two cycle response .

2.2.1 AHB master

The master of AHB protocol provides addressing and control information for read / write operation. Out of the multiple master present at any given time only a single master is given the access by the arbiter to communicate with the slave . Fig 2.14 shows the master interface.

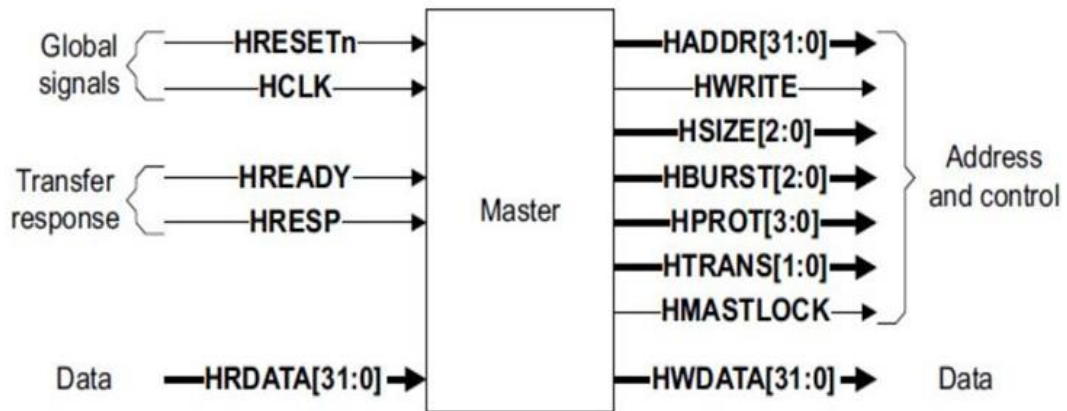


Fig 2.14 AHB Master interface

2.2.2 AHB slave

The AHB slave would react to the master initiated requests . When HWRITE is 1 then master writes data onto the slave and when it is 0 it reads data from the slave via the PRDATA signal . As there are many slaves present for communication , to choose between them decoder HSELx signal is present .

It is the responsibility of the slave to ensure that the master receives the following messages :

- If the bus transfer is completed or it is expanded .
- When a transfer has completed whether it was a successful transfer or due to some error it was a failure .

A block diagram of slave interface is present in fig 2.15

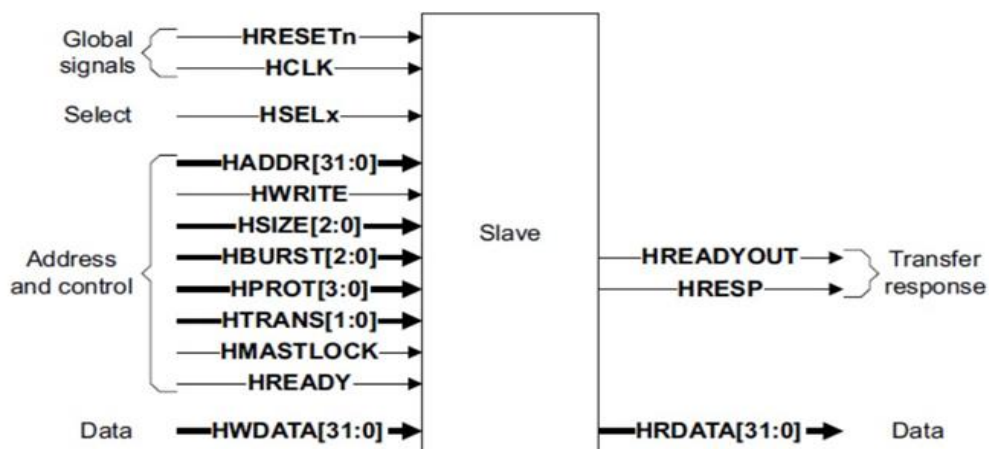


Fig. 2.15 AHB slave interface

2.2.3 Interconnect

The connection between the master and slave is made with the help of interconnect, particularly decoder and multiplexor which is discussed in detail in the below section. In AHB protocol where multiple masters are present there is a need of arbiter to select one master for transaction to the slave. The address, control and write data signals all require this route. The decoder is responsible for providing the select signal for selecting the slave.

2.2.3.1 Decoder

The decoder provides the respective select signal to the multiplexor for choosing the correct slave in order to provide HRDATA to the master. In all configurations with more number of slaves involved, a single centralized decoder is required.

2.2.3.2 Multiplexor

While transfer of data from master to slave, the arbiter is responsible for giving select signal to the multiplexor to choose the appropriate master for the transaction. The arbiter has the appropriate information on which master has to be selected. Also while the transfer from slave to master happens, the multiplexor chooses which slave has to be selected depending on the decoder select signal. In all cases with more slaves (more than two), centralized multiplexor should be present.

2.2.3.4 Arbiter

Any master should be able to connect to any slave in the AHB protocol, but at a given time only one master can communicate with one slave, hence we need to utilise an arbitrator to control which master can communicate with which slave. The arbiter is used to ensure that the bus master is only accessible to one master. The HBURSREQx is used by the master to seek bus access. In this scenario, HLOCKx signifies exclusive transfers, and the arbitrator will not allow bus to another master. After the grant is given, the master is not needed to keep HBURSREQx high. Arbiter uses HMASTER[3:0] to identify which master has bus access. When a bus is available but not necessary, an idle transfer should be made.

2.2.4 Operation

The transactions in the case of AHB protocol is split in two phases address phase and data phase.

Address phase : This is first phase of transmission in which the information of slave's address is sent.

Data phase : During this phase, the data is placed on to the bus. Since it supports pipelining so the address phase of next cycle can be fetched during this state.

In AHB to start any transaction , the approval of arbiter is required . After the grant is provided by the arbiter the transaction can start . HSELECT signal for each slave is available and for whichever it is high that slave will receive the data .

The Pwdata signal is responsible for transmission of data from the master to the slave and for reading the data from the slave Prdata signal is responsible. For starting the transfer firstly the master is selected and then it provides the information on the address and other control signals . Some of the information like transfer's address , transfer type , burst type is provided by these signals .

The transfer of data can take place in one of the following forms ;

- Single transfers
- Burst that wrap at address boundaries
- Burst that do not wrap when the address boundaries are incremented .

When the slave is not ready to receive the data it can choose to extend the transfer by asserting the Hready signal as low . The address and data signal should be extended as it is when HREADY goes low . If Hready goes low it indicates that the slave wanted to add some wait states during the transfer as it was not ready to receive the data . It allows slave to have more time to supply or sample the data.

Transfer types

There are four different types of transfer supported by AHB protocol given by the HTRANS [1:0] signal . It is a two bit signal and hence four different types of transfers are possible .

HTRANS[1:0]	TYPE	DESCRIPTION
00	IDLE	No data transfer is required
01	BUSY	Burst of Idle cycles .Master inserts idle cycle in the middle of large sequence of transfer . If it has to serve some immediate request , the master can do so by putting in busy mode and later on it can drive the remaining burst .
10	NONSEQ	This informs about the first data transfer from a burst of data or an individual transfer . The HBURST signal can determine either of them.
11	SEQ	For a burst transfer the first transfer would be indicated as nonseq and the remaining ones would be indicated by SEQ .

Table 2.2 Transfer types in AHB protocol

Burst type

The transfer of data can be in any of the modes either an individual transfer or burst of transfer indicated through the HBURST[2:0] signal . For the burst transfer to happen it can happen in incrementing type or wrap around type .

HBURST[2:0]	TYPE	DESCRIPTION
000	Single	Individual transfer
001	INCR	Incrementing burst where the length of data is not specified .
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

Table 2.3 Burst types in AHB protocol

For an incrementing burst the address after each of the transfer can be found by incrementing the previous transfer's address .

For the wrapping burst the wrap boundaries are defined and when address crosses that boundary then the address wraps around to the start address .

Slave transfer response

Using the Hresp[1:0] along with Hready signal the slave signal will indicate the response of the transfer .

In one of the following ways the slave would be completing the transfer :

- Completion of the transfer immediately
Okay transfer response provided (HRESP =2'b00)
- By adding wait states
HREADY low used to introduce wait states

- Signal an error .
Error response provided (HRESP = 2'b01)
Two cycle response
- Delay completion but allows bus to be released
Split/Retry response provided
Two cycle response

Two cycle response :

ERROR, SPLIT and RETRY uses 2 cycle response .

- It allows master to cancel the following transfer .
- HREADY signal is low in first cycle and high in the second cycle .
- HRESP[1:0] signal indicates required error , split , retry response in both the cycles .

2.2.5 Signal Descriptions

2.2.5.1 Global signals

NAME	SOURCE	DESCRIPTION
HCLK	Clock source	All bus transfers happens in synchronization to this clock (rising edge)
HRESETn	Reset controller	This active LOW signal is used to reset the bus.

Table 2.4 AHB Global signals

2.2.5.2 Master signals

NAME	Destination	DESCRIPTION
HADDR[31:0]	Slave and decoder	Address bus which is 32 bit wide
HBURST[2:0]	Slave	Indicates individual transfer or a burst of data transfer .
HMASTLOCK	Slave	If HIGH, it means that the current transfer is part of a locked sequence.

HPROT[3:0]	Slave	Protection control signals
HSIZE[2:0]	Slave	Size of the transfer
HTRANS[1:0]	Slave	The type of transfer being done .
HWDATA[31:0]	Slave	During write operations, the write data bus is utilized to transmit data from the bus master
HWRITE	Slave	When this signal is high , the data is transmitted from master to slave and when low the data is read from the slave .

Table 2.5 AHB Master signals

2.2.5.3 Slave signals

NAME	Destination	DESCRIPTION
HRDATA[31:0]	Multiplexer	Using this bus , the data is been sent from the slave to master .
HREADYOUT	Multiplexer	Whenever the HREADY signal becomes HIGH, it means that a bus transfer is complete.
HRESP	Multiplexer	The transfer response contains extra information about a transfer's status.

Table 2.6 AHB Slave signals

2.2.5.4 Decoder signals

NAME	Destination	DESCRIPTION
HSELx	Slave	HSELx is the slave select signal for each slave, indicating that the current transfer is for the slave now selected.

Table 2.7 Decoder Signals

2.2.5.5 Multiplexor signals

NAME	SOURCE	DESCRIPTION
HRDATA[31:0]	Master	The decoder selects a data bus to read.
HREADY	Master and slave	When this signal is high it indicates that the previous transfer was complete .
HRESP	Master	The decoder chooses the transfer answer.

Table 2.8 Multiplexor signals

2.2.5 Transfer of data

There are two stages to a transfer:

- The address is valid for one HCLK cycle and it might get extended due to previous transfer.
- HCLK cycles may be extended according to the data being sent . It determines how many clock cycles are necessary to complete the transfer using the HREADY signal.

The flow of data is determined by the Hwrite signal. As a result, when:

- When HWRITE is HIGH, the master provides data on the HWDATA [31:0] write data bus, indicating that a write transfer is in progress.
- Date can be read from the slave when Hwrite is driven low . The data is sent through the HRDATA [31:0] read databus.

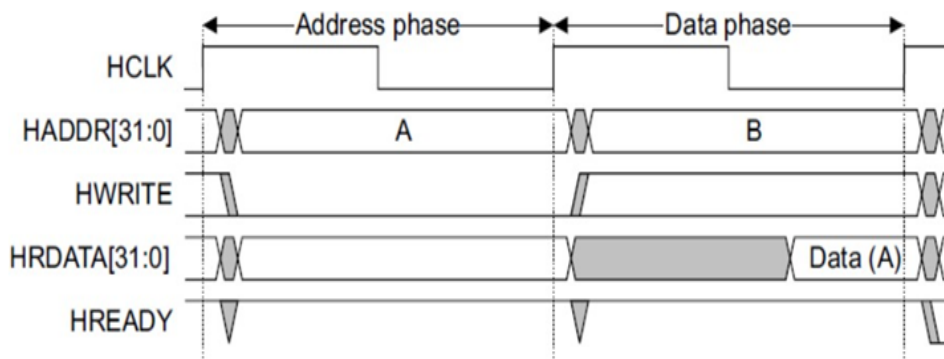


Fig 2.16 AHB Read Transfer[17]

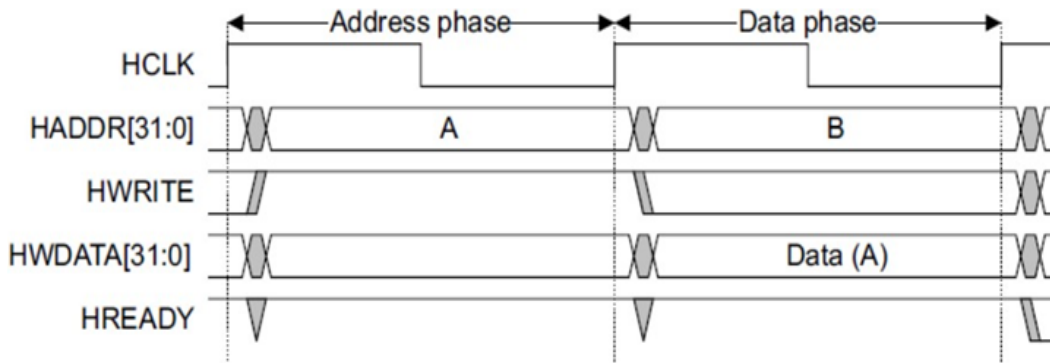


Fig 2.17 AHB Write transfer [17]

When the transfer does not involve any wait states :

- On the positive edge of Hclk , the master is responsible for providing the address and the control signals on the bus.
- On the next positive edge of Hclk , the slave is responsible for sampling the address and control information .
- After receiving the address and control information by the slave , it will drive the Hready signal appropriately . The master samples this response on the third rising edge of HCLK.

The data and address phases of the transfer are presented over several clock cycles. Since it is a pipelined protocol the data phase of the current transfer overlaps with the next transfer's address phase .This concurrent nature of the protocol helps in achieving good performance .

Appropriate time interval is given for slave to complete transfer and if it needs additional time it can do so by driving the Hready signal low.

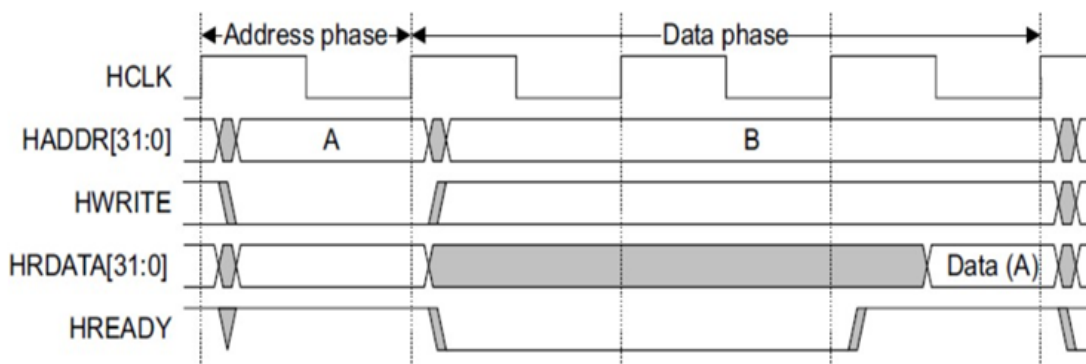


Fig 3.18 AHB Read transfer with wait states [16]

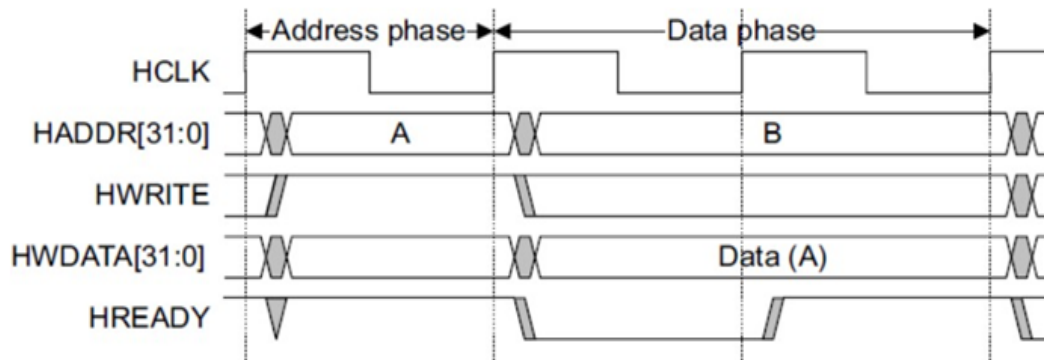


Fig 3.19 AHB Write transfer with wait states [16]

In the case of write operations, the master is held responsible for maintaining the data stability for long interval of time. As shown in fig 3.18 and 3.19 the read and write transfers are shown where the slave has inserted some wait states by making the Hready signal low . In fig 3.18 the Hready is driven low for two clock cycles so it indicates that two wait states have been inserted in the read transfer whereas in fig 3.19 one wait states is inserted since Hready is made low for one clock cycle during write transfer .

Chapter 3

Verification IPs

3.1 IPs and VIPs

An IP (intellectual property) in semiconductor stands for a reusable unit of logic which is intended to license to multiple vendors for the purpose of reusability as a building block in different chip design . It can be a logic or a functionality or a cell or a layout design.

In the ongoing IP trend , more and more functionality is getting embedded into the single chip i.e SOC (silicon on chip) design [15]. The use of these IPs became very prominent in the SOC since they are reusable unit and when once designed can be reused as many number of times . These IPs may contain microprocessors , microcontrollers and other lot of functionality as designed .

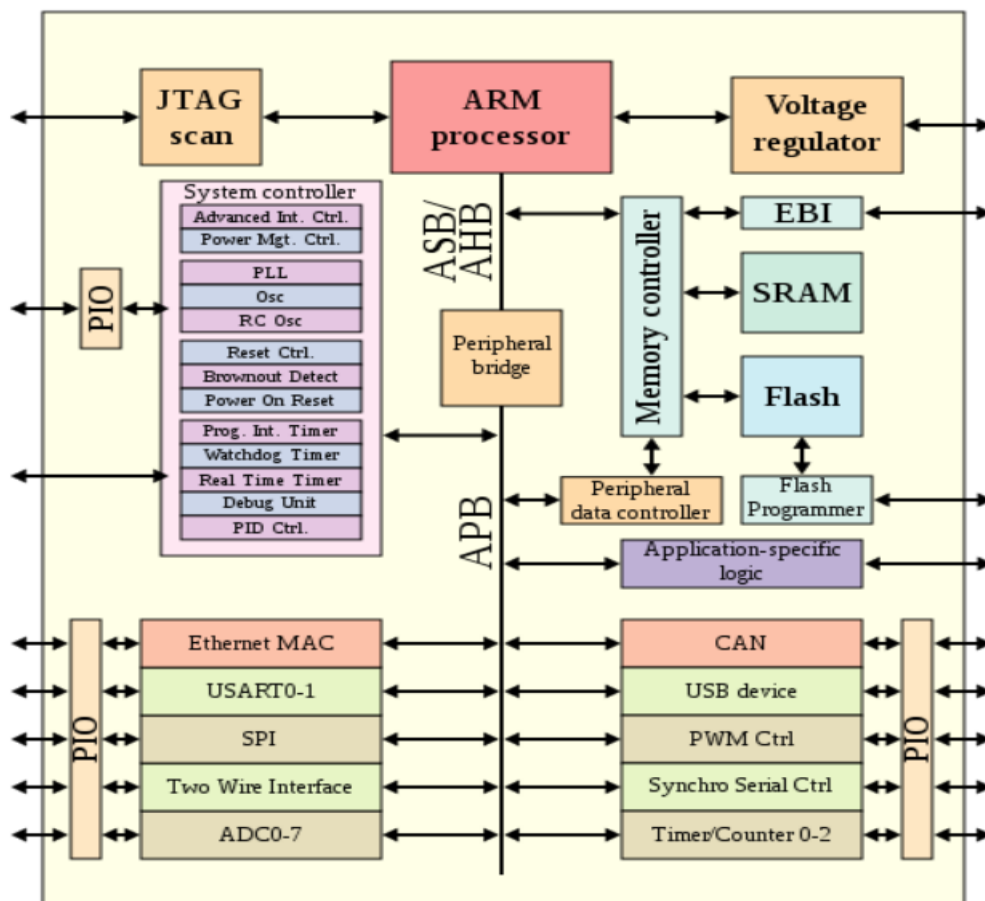


Fig 4.1 An example of IP

In the above figure an example of intellectual property is shown . It might contain components like processors , memory and some standard ARM protocols like AHB , APB , the designs like ethernet

, SPI, USB , UART core etc . All of the above mentioned components can be integrated into a single IP which can be licensed to multiple vendors .

The IPs can be further classified as two types , as mentioned below :

Soft IP core

These IPs blocks are basically those that can be used as a synthesizable RTL models which are developed using Hardware descriptive language like Verilog or SV . Also in cases where IP core are synthesizable and also can be provided as generic gate level netlist that are mapped to the process technology comes under the category of soft IP cores. One offer that it provides is in the customization in the back end placement and routing by a customer to any process technologies as required .

Hard IP core

These IPs are provided as a layout design in a layout format as GDS that can be mapped to process technology which can be given to customer to the final layout of the chip . Due to this , these cores could not be customized according to different process technologies .

Usually the digital logic cores are designed and developed as soft IP core since it offers an advantage of mapping to different process technology according to the customer .

Due to the rising popularity of IP designs, a demand for Intellectual Property Verification arose (VIPs).

The Verification IPs, like the IPs, are pre-defined functional blocks that may be introduced into testbenches to check a specific design.

Verification of SOCs often takes more than half of the full project cycle and is done in stages, starting with smaller logical blocks, then moving on to checking logic components at a subsystem level, and finally certifying the entire SOC chip.

The VIPs can be used to verify at all these levels of verification as simulation models for actual design IP .

Bus functional models, stimulus generators, protocol monitors, and functional coverage blocks are common VIP blocks. As a result of the current technological trend, testbenches use a variety of languages and methodologies, and these VIPs are meant to be readily customised and incorporated into verification environments.

Most of the complex SOCs involves the use of IP core based design and verification IPs for easier verification and also a thorough functional coverage . It also reduces the time to market and also results in successful products .

3.2 JTAG

JTAG (Joint Test Action Group) is used for verifying the designs and also testing the printed circuit boards after manufacture . It can be used to verify the internal connection of chip as well as the interconnections between the chips when multiple chips are mounted on the board[14].

Due to shrinkage of chip size , its testability became difficult . In order to solve the complexity of testing , JTAG was devised using boundary scan technology which helped the engineers in debugging the right cause of failure on a system using a small number of dedicated pins . The signals are sent on the appropriate input pin of the JTAG serially and correspondingly output is observed under various conditions and it is verified with the input signals sent . Boundary scan technology is an effective method for design for test used in the industry . The below figure 4.2 includes the architecture of JTAG .

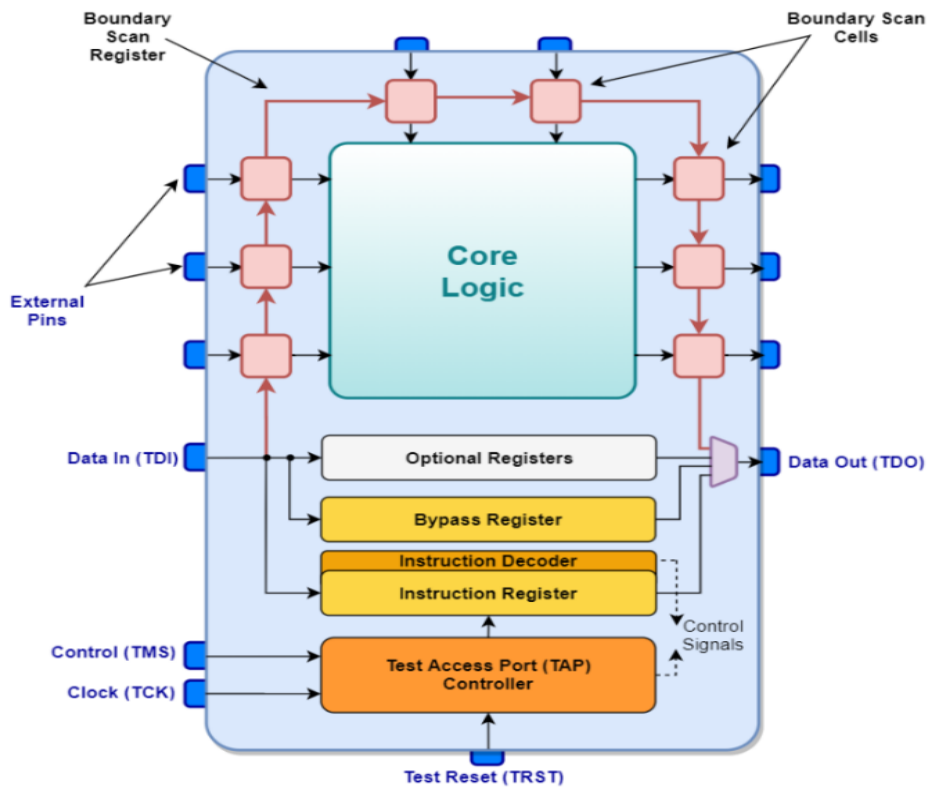


Fig 4.2 Architecture of JTAG [13]

It consists of four signals for defining its functionality :

- Test data input (TDI) pins
- Test data output (TDO) pins
- A test clock pin (TCK)
- A test mode select pin (TMS) for controlling the TAP state machine

There is an optional pin called as reset pin (TRST), which forces the state machine into the reset state. It is an optional pin since reset state can also be obtained by holding the TMS low and clocking TCK five times.

Using the TDI pin the input data is serially sent into for testing . Through the TDO pin the data comes out after being shifted serially through the input pin. All the activity is synchronized with the help of clock pin TCK . Using the TMS pin the tap controller is controlled and we can enter into loading the instruction into the instruction register or enter into loading the data onto the data register for achieving the desired operation .

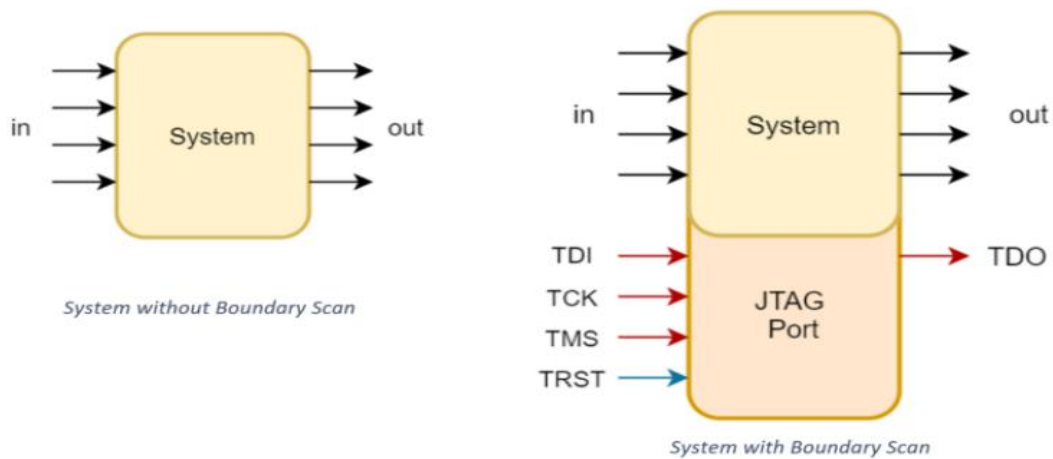


Fig 4.3 System with boundary scan [13]

THE TAP CONTROLLER

The test access port (TAP) consists of the TAP controller, an instruction register for holding the instruction and also data register.

The TAP controller composes of a finite state machine which operates based on the TCK and TMS signal which is clocked by TCK . The FSM operates only in two modes : instruction mode or data mode . The decision to operate on the instruction or data operations in the FSM is based on making the TMS signal 0 or 1 appropriately. The state machine then either progresses (TMS HIGH) or goes toward the reset state when in a particular mode (TMS LOW).[15]

The input pin is responsible for feeding the input to the boundary scan cell from the physical pins , and also load the data on to the instruction register or data register depending on whichever operation is selected by the TMS signal . The output pin is used for reading the data from the boundary scan cells as well as receive the data from instruction and data registers .

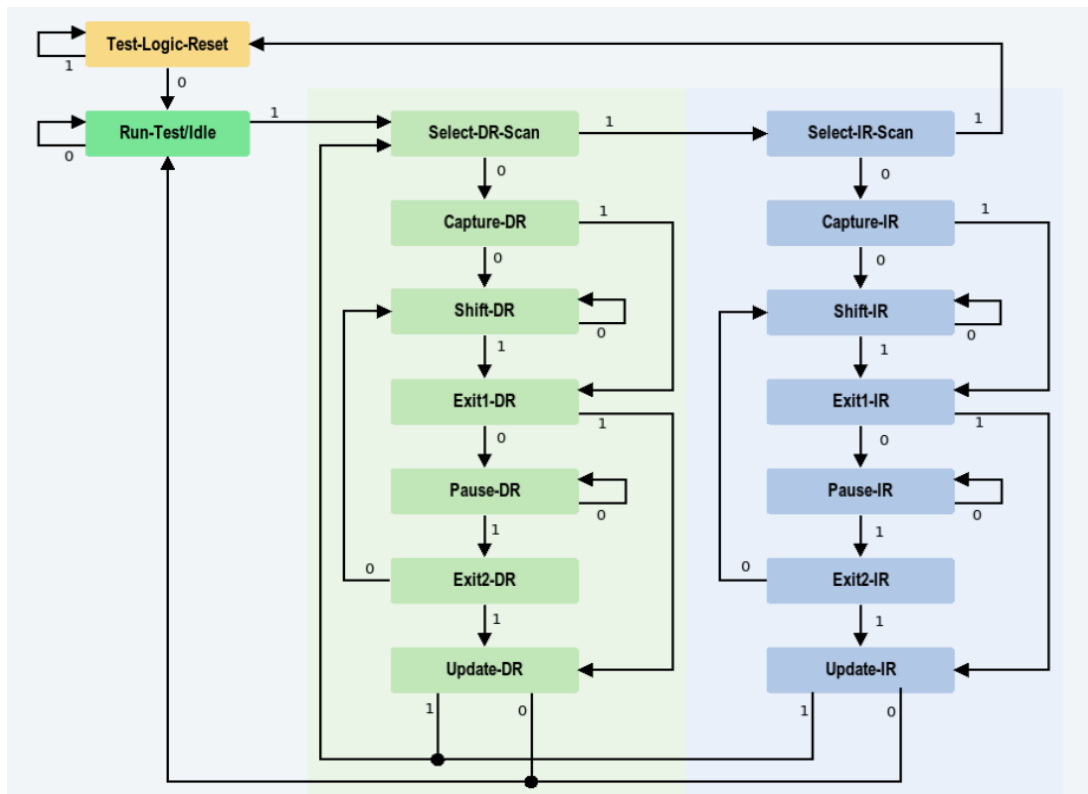


Fig 4.4 Finite state machine of the TAP controller [15]

3.1.1 WORKING OF JTAG

The working of JTAG is summarized in below points :

1. The instruction mode is chosen first. The user can clock in an instruction using TDI in Shift IR state inside the instruction mode path.
2. The state machine then continues until it resets.
3. The following step in most instructions is to choose the data mode. Data is loaded via TDI and/or read via TDO in this mode. The data routes for TDI and TDO will be configured in line with the clocked-in instruction. Data is loaded on in the Shift DR state of the TAP controller .
4. The state machine returns to the reset state after the read or write operation is completed.

3.1.2 JTAG REGISTERS

The Boundary scan is related with two types of registers. One instruction register and two or more data registers are present in each conforming device.

INSTRUCTION REGISTER

The present instruction is recorded in the instruction register. The TAP controller decides what to do with signals based on its content. Also, which of the data registers signals should be routed to is usually decided by the content of this register.

DATA REGISTER

The three principal data registers are the Boundary Scan Register (BSR), the BYPASS register, and the IDCODES register. Some additional registers is also provided as they may be required , however Jtag standard does not requires it.

Boundary scan register

The major testing data register is the BSR. It is involved in transferring of data to and from a device's I/O pins.

Bypass register

BYPASS register transfers data from TDI to TDO. It is a single bit register . It enables the testing of devices in a circuit with minimum overhead since it bypasses the device whose testing need not be performed .

IDCODE register

IDCODES registers the device's ID code and are stored in this register. It is used for checking whether the device being tested is correct or not .The device can be linked to its Boundary Scan Description Language (BSDL) file using this information. The settings for the Boundary Scan is detailed in the file .

Boundary scan instructions

For a device to be regarded compliant, the IEEE 1149.1 standard specifies a set of instructions that should be present . These are the instructions:

- Bypass

The TDI and TDO pins are connected via a single bit register when this instruction is executed . It is basically used at a situation where there are multiple chips connected on a board but we need to test only a few of them . So , at such situations in order to avoid overhead we employ this instruction and the chips are bypassed .

- EXTEST

This instruction connects the TDI and TDO to the Boundary Scan Register (BSR). This instruction is basically used to test the external connections between the chips present on a board . Device pins

are selected with capture state and the new values will be moving to the BSR using the shift state .

- INTEST

When we use the INTEST instruction, the TDI and TDO lines are connected to the Boundary Scan Register (BSR). Using this state the device's core logic is tested involving the various states that occur in JTAG .

Chapter 4

Simulations

In this chapter the simulation results for public data register: Bypass and Idcode are shown .

BYPASS Scan instruction implementation

Using the bypass instruction there establishes a direct connection between the TDI and the TDO signal . The structure of this register is very basic . It's merely a one-bit storage register . We can shift our test data from TDI to TDO when the ShiftDR state equals logic-1. In this approach, we give a bypass through this chip, so if it has already been confirmed or we trust the chip, JTAG will not test it.

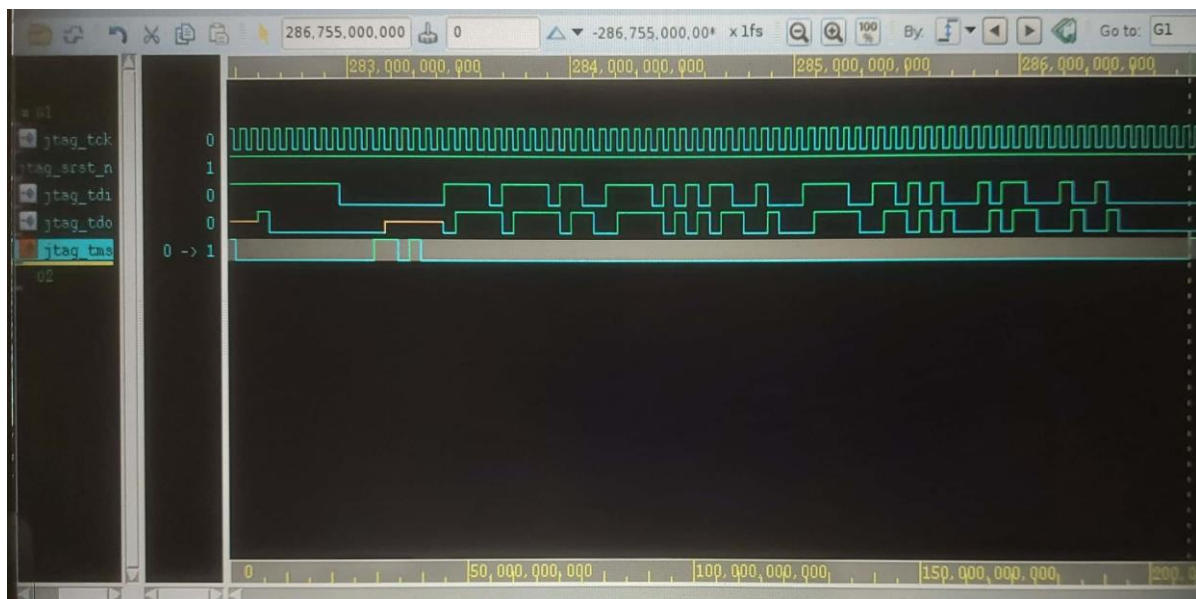


Fig 4.1 Simulation results of Bypass scan instruction

As shown in fig 4.1 it can be seen that the inputs that we applied through the TDI pin is appearing after one clock cycle at the output pin , thus manifesting the Bypass operation in the JTAG .

IDCODE instruction implementation

This instruction is used to identify the device identification code of the device being connected for verification . Here, the device identification code is a 32 bit value being sent on the TDI pin but we mask the upper bits and check for lower 12 bits which should be equal to 0E1 H . Upon receiving

this code on the TDO pin we match it with this value and if it matches we can proceed with the operation . It is shown in fig 4.2 .

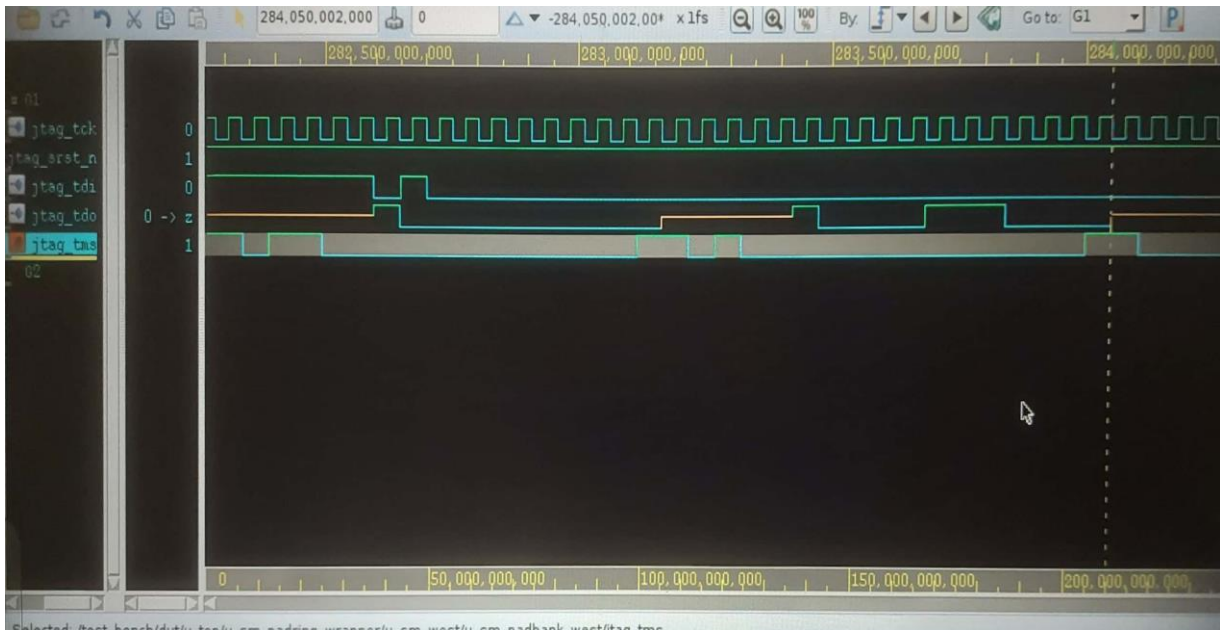


Fig 4.2 Simulation results for IDCODE instruction

Conclusion & Future scope

The AMBA protocol was studied in detail under this project majorly focusing on the APB and AHB protocol. All the signals related to the master and slave interface was also studied . The project also showcases the evolution of AMBA protocol over time and the added features it brought with each evolution . The importance of having a protocol for faster SOC integration was mentioned . After realizing the use of protocol in SOC environment , the study of IP and VIPs is presented . An important verification IP , JTAG is studied in which its architecture is described in details to understand the functionality of JTAG . Also , the implementation and simulation results for bypass instruction and idcode instruction is included under this project .

As a part of future scope under AMBA protocol several other important advanced protocol like AXI, ACE , CHI and other protocols can be studied and implemented . Under JTAG several other verifications like boundary scan output and input tests can be simulated for a given number of chips being connected . Also , operations like INTEST and EXTEST can be implemented in order to check for any stuck at faults .

References

- [1] “AMBA Specification (Rev 2.0)”, available at <http://www.arm.com>.
- [2] ARM. “AMBA Open Specifications” <http://www.arm.com/products/system-ip/amba/ambaopen-specifications.php>.
- [3] “Advanced Microcontroller Bus Architecture (AMBA)” http://en.wikipedia.org/wiki/Advanced_Microcontroller_Bus_Architecture.
- [4] Kiran Rawat *et al.* (2015). “RTL Implementation for AMBA ASB APB Protocol at System on Chip Level” *2nd International Conference on Signal Processing and Integrated Networks (SPIN)*, pp. 927- 930.
- [5] Kiran Rawat *et al.* (2014). “Design of AMBA APB Bridge with Reset Controller for Efficient Power Consumption” *9th International Conference on Industrial and Information Systems (ICIIS)*, pp.1-5.
- [6] Kiran Rawat *et al.* (2015). “Implementation of AMBA APB Bridge with Efficient Deployment of System Resources” *International Conference on Computer, Communication and Control (IC4)*, pp. 1- 4.
- [7] Ashutosh Gupta *et al.* (2016). “Physical Design Implementation of 32-bit AMBA ASB APB module with improved performance” *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pp. 3121-3124.
- [8] Jasmine Chhikara *et al.* (2015). “Implementing Communication Bridge between I2C and APB” *IEEE International Conference on Computational Intelligence & Communication Technology*, pp. 235-238.
- [9] Chenghai Ma *et al.* (2011) “Design and Implementation of APB Bridge based on AMBA 4.0” *International Conference on Consumer Electronics, Communications and Networks (CECNet)*, pp. 193-196.
- [10] L. Benini, A. Macii *et al.* (2000). “Architectures and Synthesis Algorithms for Power-efficient Bus Interfaces,” *IEEE Trans. Computer-Aided Design*, vol. 19, pp. 969–980.
- [11] J. Chhikara, R. Sinha and S. Kaila, "Implementing Communication Bridge between I2C and APB," 2015 IEEE International Conference on Computational Intelligence & Communication Technology, 2015, pp. 235-238, doi: 10.1109/CICT.2015.19.

- [12] P. Jain and S. Rao, "Design and Verification of Advanced Microcontroller Bus Architecture-Advanced Peripheral Bus (AMBA-APB) Protocol," 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), 2021, pp. 462-467, doi: 10.1109/ICICV50876.2021.9388549.
- [13] "DFT scan techniques" <https://technobyte.org/jtag-boundary-scan-structured-techniques-dft/>
- [14]. C. Elakkiya, N. S. Murty, C. Babu and G. Jalan, "Functional Coverage - Driven UVM Based JTAG Verification," 2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 2017, pp. 1-7, doi: 10.1109/ICCIC.2017.8524556.
- [15] Milna M. J, Deepa N. R , "Development of JTAG Verification IP in UVM Methodology", IJSRD - International Journal for Scientific Research & Development| Vol. 1, Issue 8, 2013 | ISSN (online): 2321-0613
- [16] Guo Jian-min and Luo De-lin, "A functional enhancement methodology to JTAG controller in complex SOC," 2009 4th International Conference on Computer Science & Education, 2009, pp. 1128-1131, doi: 10.1109/ICCSE.2009.5228481.
- [17] Acasandrei, Laurentiu, and Angel Barriga. "AMBA bus hardware accelerator IP for Viola-Jones face detection." IET Computers & Digital Techniques 7, no. 5 (2013): 200-209.