Interfacing an SD Card System with the AJIT Processor System

M.Tech Project Report I

Semester: Autumn, 2020-21

Submitted by

Name: Arghya Kamal Dey

Roll No. :193070053

Integrated Circuits and Systems

Under Guidance of:

Prof. Madhav P. Desai

Department of Electrical Engineering

INDIAN INSTITUTE OF TECHNOLOGY BOMBAY

Powai, Mumbai-40076

December 2020

Dissertation Approval

The report entitled

"**Interfacing an SD Card System with the AJIT Processor System**"

by

Arghya Kamal Dey

(Roll No. 193070053)

is approved for the degree of

Master of Technology with Integrated Circuit and System specialization

_____ _____

(Examiner) (Chairperson)

_____

(Supervisor)

Date:

Place:

Declaration

I declare that this written submission represents my own ideas in my own words and where other's ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misinterpreted or fabricated or falsified any idea/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

_____

Arghya Kamal Dey

Roll No 193070053

Date :

Acknowledgments

I would like acknowledge my supervisor, Prof. Madhav P. Desai for his guidance and mentorship throuhgout the period, which helped me in the persuit of excellence.

Abstract

We want to build a system for an SD card around the AJIT core. This system will help us integrating a new memory system with the already existing system for the AJIT core. An SD card can be used to transfer file system in much faster speed and can have higher capacity. The aim of this project is to build a system which can transfer data at a high speed with the core. In this report the AJIT processor system currently available is described. The memory systems currently used for storing data like flash memory and DRAM is explained. How in the current system the communication between the memory and the processor occurs is shown. Then SD card and its different types are discussed. The design choices are made such that the system gives the required performance as well as high capacity of memory. There are different modes of communication with an SD card. All of them are discussed and a reason behind the choice made is explained. For communicating with an SD card an SD host controller needs to be designed. The design specifications and the internals of the host controller are discussed with block diagrams. The design needs a verification plan which will verify the functionality of the SD card host controller. The validation of the system needs to be done in an FPGA. The choice of the FPGA and logistics to be used are discussed at the end.

Contents

List of Figures

List of Tables

1. Introduction

1.1 Introduction to AJIT processor

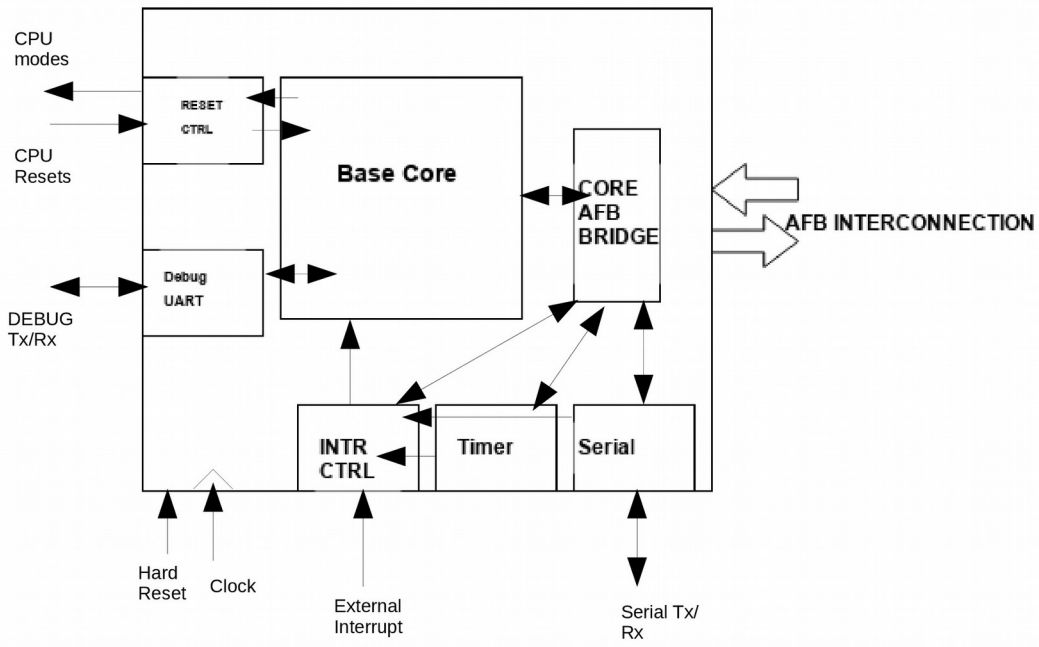

Figure 1.1 : Generic AJIT Processor Core

The AJIT processor is a completely indigineous processor made in IIT Bombay using tools made in IIT Bombay. The AJIT processor central processing unit is an implementation of SPARC-V8 instruction set architecture. It has 32 kB instruction cache and 32 kB data cache. Both are direct mapped. It has a

Memory Management Unit (MMU), which helps in translating virtual memory address to physical memory address. It has a 64 bit AJIT core FIFO bus interface. This 64 bit interface is translated into a 32 bit AJIT FIFO BUS protocol. Also there are some integrated peripherals like a serial device, a timer and a interrupt controller. The processor also includes a debug support unit with UART. This helps in remote debugging and internal state monitoring. There can be a CPU reset or soft reset or a power on reset or hard reset.

1.2 Introduction to the Project

To run an application in the current system we prepare a FLASH image for booting the application. This application is then loaded into the RAM by a progaram which resides in the FLASH memory. FLASH memory is a read-only memory, which is connected to the processor using a SPI FLASH controller.

The main disadvantage of this system is that the random access memory is volatile here, that is the data is lost every time power is disconnected. So every time the application or the file system needs to be loaded into the FLASH and then transferred into the RAM. This is a costly operation and takes time. Therefore the solution is using a non-volatile kind of memory to store a file system.

An SD card gives that flexibility in storing file system in non-volatile maanner. Also a much higher storage capacity can be achieved. Performance and speed of the SD card can be much higher also. Therfore the SD card sytem is being integrated into the AJIT processor core system.

The SD card needs a host controller to communicate with the processor. SD card host controllers are specified by the SD Assosciation. The version which we are going to follow and the reason behind it is discussed in the report. The specification of the SD card and the host controller is discussed here in the report. All the design decisions made are explained and the performane goals are mentioned too. The functioning needs to be verified using a testbench. Also a validation methodology is planned for validating the performance of the system.

2. SD Card

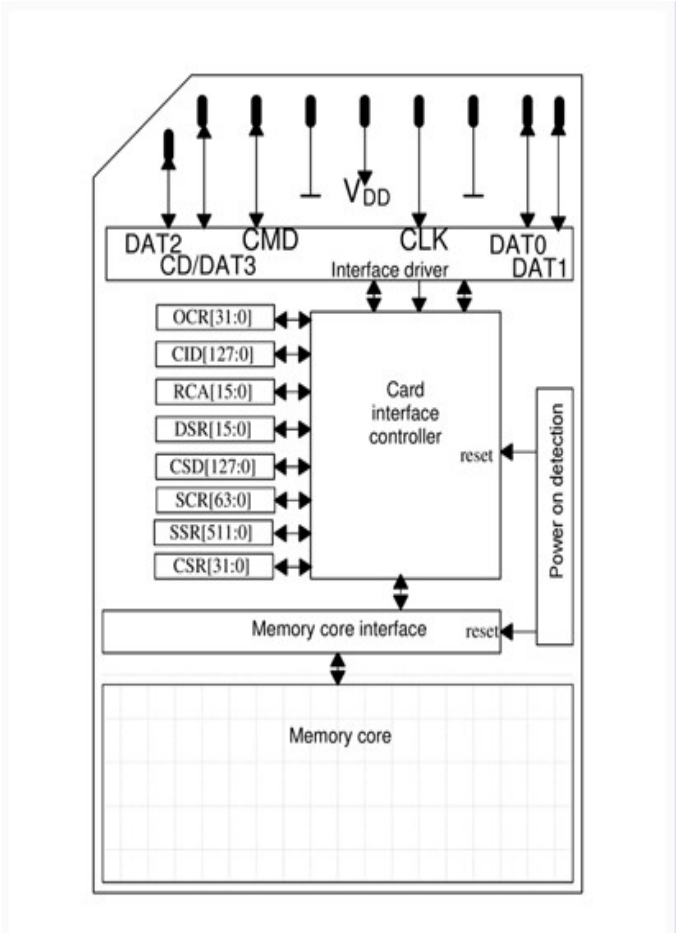

Figure 2.1 : SD card internals

SD card or Secure Digital card is a non-volatile portable memory card. This is developed by the SD Assosciation (SDA). The SDA creates SD card standards and host controller specification standards and some other standards to assure compatibility of specifications.

2.1 SD Card Architecture

The SD card internal architecture is shown which conforms to physical layer 3.01 in Figure 2.1.

2.1.1 SD Card Pins

- CMD: The bidirectional line through which commands are sent to the SD card in serial sequential manner is the CMD line. Also the responses from the SD card is sent to the host through line only. This pin is an open-drain connection.
- DAT[0:3] : There are four bidirectional lines which are used to transfer data from the host to the card and vice-versa. When 1-bit mode is used, data is transferred only through DAT0 line. In the 4 -bit transfer mode all the data lines send or recieve data simultaneously.

- CLK : The clock is sent by the host to the card. So the clock is generated by the host controller.
- VDD & VSS: There are one vdd line and two vss lines connected in the interface. The voltage of VDD depends on the type of the card and mode of operation.

2.1.2 SD Card data packet format

There are two types of data packet formats :

1. Byte-width
2. Wide-width data

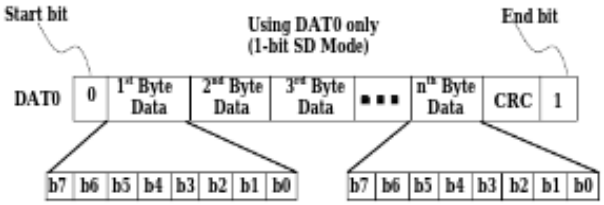

Figure 2.2 : Byte-width data-packet format when using 1-bit SD mode

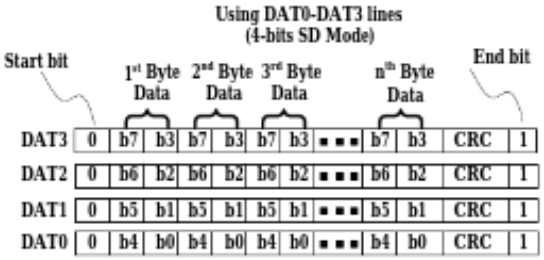

Figure 2.3 : Byte-width data-packet format when using 4-bit SD mode

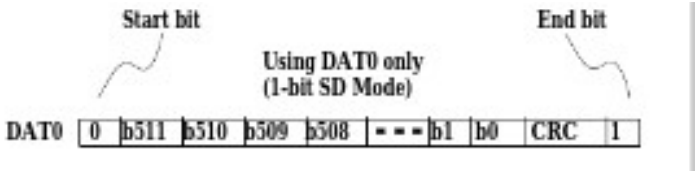

Figure 2.4 : Wide-width data-packet format when using 1-bit SD mode

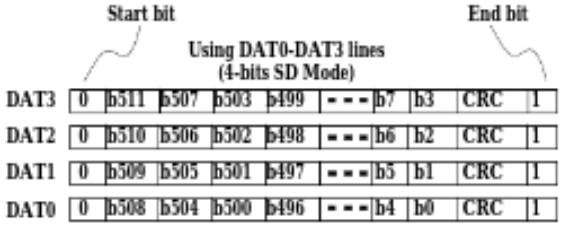

Figure 2.5 : Wide-width data-packet format when using 4-bit SD mode

2.1.3 SD Card Registers

An SD card has its own registers to carry card or content specific data. These registers can only be accessed by some specific commands. There is command for accessing each of the registers.

Name	Width	Description
CID	128	Card Identification number is stored
RCA	16	Relative Card Address or the local system address of a Card. It is dynamically suggested by the card
DSR	16	Driver Stage Register is used to configure the card's output driver
CSD	128	Card Specific Data is used to store card operation conditions
SCR	64	SD Configuration Register stores card's feature capabilities
OCR	32	Operation Condition Register
SSR	512	SD Status Register has information about card proprietary features
CSR	32	Card Status Register stores card status data

Table 2.1: SD Card Registers

2.2 Classification of SD Card

An SD card can be classified in two ways : capacity and speed. They are described as below.

2.2.1 Capacity

There are four different families in which the SD card can be divided in terms of capacity. These are SDSC, SDHC, SDXC and SDUC. They are explained below.

- **SDSC :** It stands for Secure Digital Standard Capacity. SDSC can store upto 2 GB of memory. Typical file system stored in the SDSC type of card is in FAT16 format.
- **SDHC :** It stands for Secure Digital High Capacity. These cards have capacity upto 32 GB. Physically and electrically they are the same as SDSC. Typical file format in this type of card is FAT32. The standard clock speed was increased to 25 MB/s.
- **SDXC :** It stands for Secure Digital eXtended Capacity. An SDXC card can have minimum of 32 GB capacity and maximum of 2 TB of memory. SDXC file system is made mandatory to Microsoft's exFAT format. New bus speed modes were introduced too in this version. The interface speeds can be 50MB/s to 104 MB/s.
- **SDUC :** It stands for Secure Digital Ultra Capacity. It can support upto 128 TB of storage from a minimum of 2 TB.

2.2.2 Bus Speed

SD card speed is rated by sequential read or write speed. This card speed is improved by increasing the SD card bus speed. Therefore the classification of the SD card speed is made in terms of the bus speed.

- **Default Speed**

  SD card in this speed mode works at the speed of 12.5 MB/s.

- **High Speed**

  This speed mode works works at the speed of 25 MB/s.

- **Ultra High Speed (UHS)**
  - UHS-I : This mode supports transfer of data at a speed of 104 MB/s. This mode is also called UHS104 or SDR104. It supports a clock frequency of 208 Mhz at four-bit transfer mode. It also supports slower 50 MB/s speed. It requires 1.8 volt operating voltage.
  - UHS-II: In this mode maximum bus speed is further increased. In this mode the SD card has an additional row of pins. The maximum speed can be 312 MB/s.

2.3 SD Card Commands

SD Card takes commands from the host and responds accordingly. The registers are accessed through the commands, Each command has its own arguments. A standard command is of 48-bits which is sent serially through the CMD line of the SD card.

2.4 SD Card Responses

There are five types of response types. The responses are sent by the SD card to the host controller. The responses can be of length 48 bits or 136 bits depending on the response type.

2.5 SD Card Functional Description

The commands are sent by the host to the SD card and accordingly the card may change its state. There is a dependency between the host operation mode and the card state. The physical layer specification version 3.01 is followed for the functional description.

Card State	**Operation Mode**
Inactive state	Inactive
Idle State	Card Identification mode
Ready State	
Identification State	
Stand-by State	Data Transfer State
Transfer State	

Sending-data State	
Receive-data State	
Programming State	
Disconnect State	

Table 2.2 : Card States Vs Operation Modes

2.6 Different modes of communication with SD card

There are two modes of communication through which host can communicate with the SD card. They are SPI mode and SD mode. According to the mode we choose to implement the host controller should be designed.

2.6.1 SPI Mode

The pinout relationship for SD card in SPI mode is shown.

No	SD	SPI
8	DAT1	-
7	DAT0	DO
6	VSS2	VSS2
5	CLK	SCLK
4	VCC	VCC
3	VSS1	VSS1
2	CMD	DI
1	DAT3	CS
9	DAT2	-

Table 2.3 : SD vs SPI pinout for SD card

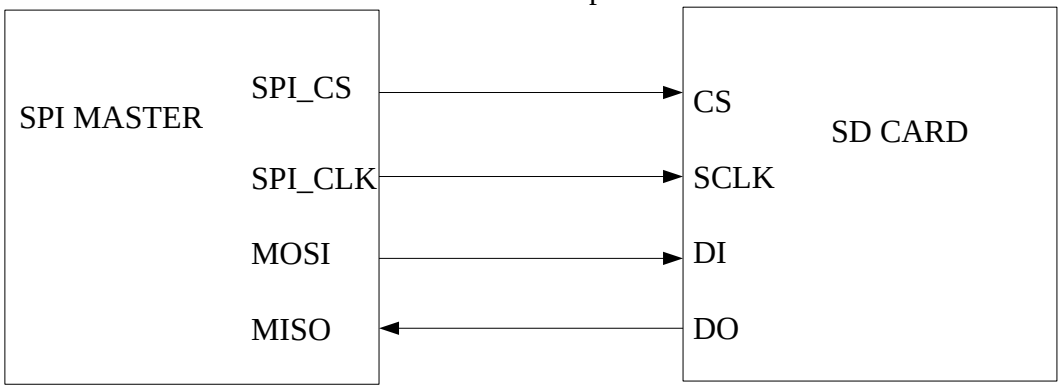

Figure 2.6 : Spi configuration with the SPI master

SPI mode for communication has its advantages and disadvantages.

- Advantages
  - It does not need a separate dedicated host controller for transferring data. Any SPI master can communicate with SD card in SPI mode.
  - It is much simpler than the SD mode.
  - Data direction on every line is fixed. None of them are bidirectional.
  - It is a low cost solution
- Disadvantages
  - Data is tranferred as a byte-oriented serial communication.
  - It uses only one DAT line for data transfer.
  - The trasfer is very slow. The maximum clock frequency at which it can work is 25MHz.

2.6.2 SD mode

There are two types of SD mode operation.

- 1-bit mode
- 4-bit mode

In this mode all transactions start with a command. The command is sent by the host to the SD card. This command is transferred in serial manner sequentially.

To answer that command a response is sent by the SD card to the host. This is also done trhrough the command line serially. The host has to wait for the response to arrive. Some commands do not send any response.

Data can be sent or received through 1 DAT line or all the four DAT lines for faster data transfer.

The SD mode can send data faster compared to SPI mode as data is sent as blocks. Typical block length is 512 bytes. Therefore the host controller has to be made to support the block transfers of a specific length.

Design Decision made here is that a SD host controller will be made which can work in both the SD modes.

3. SD Card Host Controller

3.1 SD Host Architecture

The SD Host controller is the hardware interface which helps connect the SD card with the host software. SD Host controller being implemented here follows SD Host controller specification version 3.00. This helps in developing a SD host driver for linux without looking at the specific host controller specification.

The host driver cannot directly communicate with the SD card. It can only read or write the SD host controller registers. That is why it is important to standardise the registers. The driver here works both as sd host bus driver and host controller driver. Therefore driver is made specifically for ajit peripheral bus which can support the data transfer.

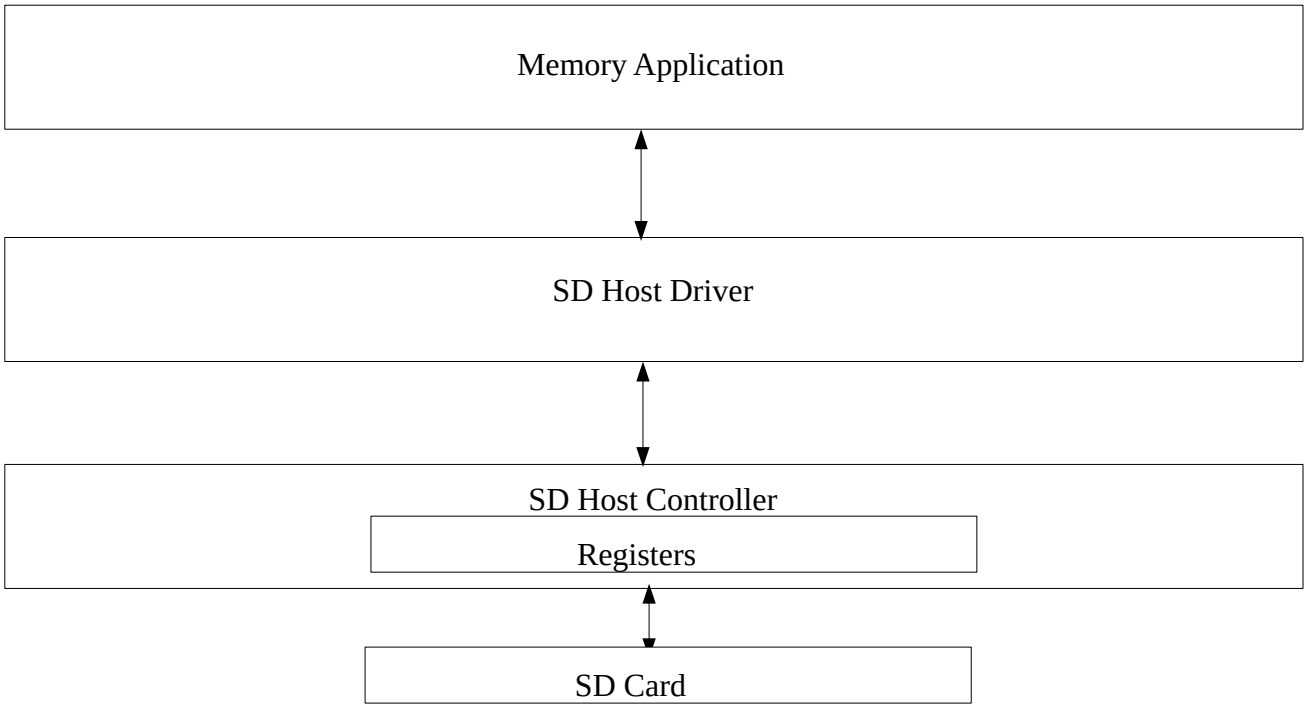

Figure 3.1 : Host hardware and driver architecture

3.2 Host Controller Interfaces

3.2.1 Clock, Reset Interfaces

- A single clock signal Clk comes as input. The controller is a positive edge-triggered.
- A single active-high synchronous reset signal as input.

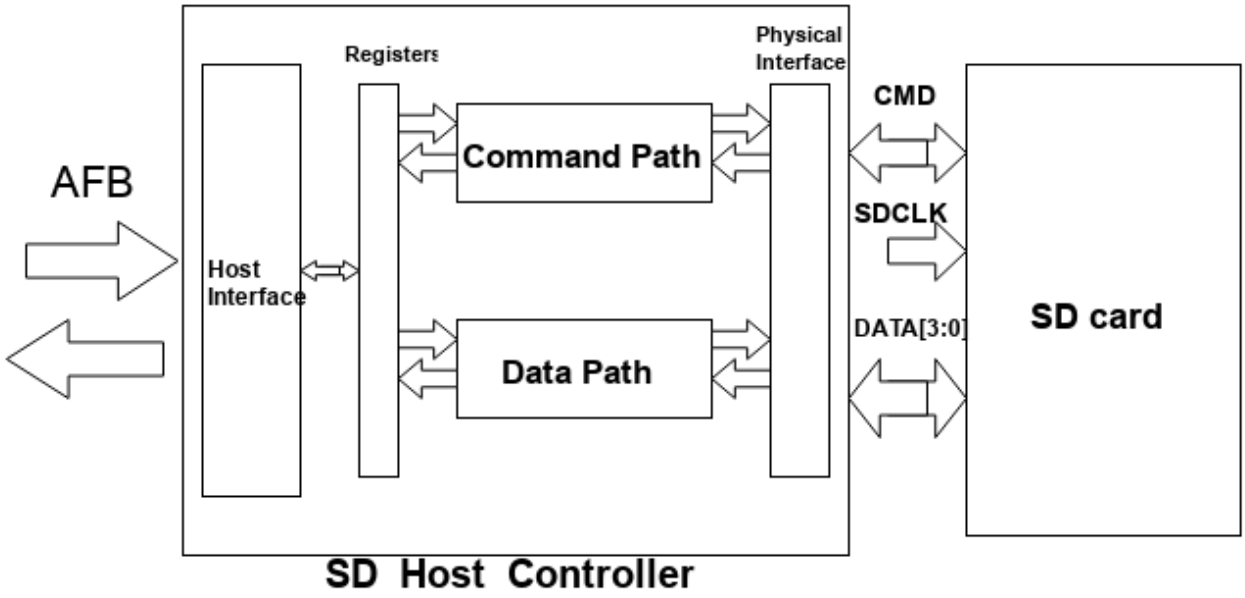

Figure 3.2 : Block Diagram of Host Controller

3.2.2 Processor Side Interface

On the processor side, there is peripheral bus interface.

1. **Request FIFO Interface**

peripheral_bridge_to_sdhc_request_data :IN (63: 0);

peripheral_bridge_to_sdhc_request_req: IN(0:0);

peripheral_bridge_to_sdhc_request_ack : OUT(0:0) ;

peripheral_bridge_to_sdhc_request_data data fields are given below:

63	62:59	58:56	55:32	31:0
rwbar	bytemask	reserved	address	data

Table 3.1 : Request data break-up

2. **Response FIFO interface**

sdhc_to_peripheral_bridge_response_data: OUT (31:0);

sdhc_to_peripheral_bridge_response_req : IN(0:0);

sdhc_to_peripheral_bridge_response_ack : OUT(0:0);

3.2.3 SD Card Bus Interface

SD_CLK: OUT (0:0) :Provides clock to the SD card.

CMD: INOUT(0:0) :A bidirectional wire that is used to send commands serially to the card and receive their corresponding responses from the addressed card to the host.

DAT : INOUT(3:0) : A bidirectional bus used to transmit data from the host to the card and vice versa. The DAT signal operates in push-pull mode.

3.3 SD Host Standard Registers

Following are the registers of the SD device as described in the SD host controller version 3.00.

Sl. No	Registers	Width	Address Offset
1	Argument 2 register	32	0x0
2	Block Size Register	16	0x4
3	Block Count Register	16	0x6
4	Argument Register	32	0x8
5	Transfer Mode Register	16	0xC
6	Command Register	16	0xE
7	Response0 Register	32	0x10
8	Response2 Register	32	0x14
9	Response4 Register	32	0x18
10	Response6 Register	32	0x1c
11	Buffer Data Port Register	32	0x20
12	Present State Register	32	0x24
13	Host Control Register	8	0x28
14	Power Control Register	8	0x29
15	Block Gap Control Register	8	0x2A
16	Wakeup Control Register	8	0x2B
17	Clock Control Register	16	0X2C
18	Timeout Control Register	8	0X2E
19	Software Reset Register	8	0X2F
20	Normal Interrupt Status Register	16	0x30
21	Error Interrupt Status Register	16	0x32
22	Normal Interrupt Status Enable Register	16	0x34
23	Error Interrupt Status Enable Register	16	0x36
24	Normal Interrupt Signal Enable Register	16	0x38
25	Error Interrupt Signal Enable Register	16	0x3A
26	Capabilities Register	32	0X40
27	Host Controller Version Register	16	0XFE

Table 3.2 : Register Set in Host controller

4. Verification and Implementation

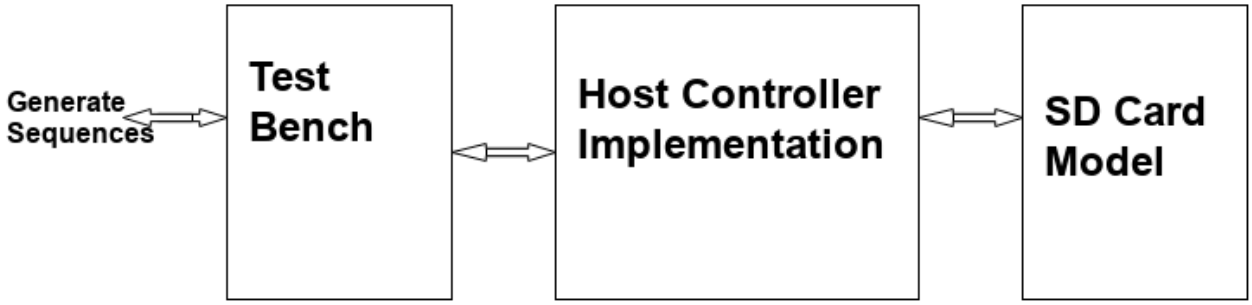

Figure 4.1 : Verification plan

4.1 Verification

- Implement a behavioural model of the SD card (In C programming language).
- The SD card model will be written following physical layer specification version 3.01.
- Host Controller to be implemented compliant to version 3.0 in Aa language.
- Communicate with the SD card model using the controller by generating sequence of commands using testbench written in C and the ccontroller is converted from Aa to C.
- For verifying the memory first we are writing to every address the same value as the address, and then reading from the address and checking whether the value is same or not. This is called a memory march test.

4.1.1 Testbench

The testbench follows the following sequence:

1. Initilization done and check for errrors. If successful, go to step 2 else to step 5.
2. Memory blocks are written with serially incremented values, if no error then go to step 3 else to step 5.
3. Memory block are read and matched with expected values, if any error is found go to step 5.
4. Test is successful and passes functionality check.
5. Test is not successful.

4.2 Validation

This process will be implemented on a Basys-3 FPGA platform.

- An SD breakout board is used for connecting the SD card with the Basys-3 FPGA.
- The breakout board has all the pins for SD mode of communication.
- Load the binary file of the implemented host controller on the FPGA card.
- Connect the desired SD card to the board for verification.
- An SD card of 1TB storage and of SDXC type is decided to be used for validating.

4.3 Performance Goals

With this controller, the SD card is expected to operate at a maximum speed of 104MB/s for SDXC cards(Capacity 32GB-2TB), 50MB/s for cards other than SDHC (Capacity 32GB), and a minimum speed of 12.5MB.s (applicable to all cards). The host controller system should run in 4-bit mode and 1-bit SD mode both. The decision was made to follow the SD Host controller version 3.00. Therefore the host controller should be overall compliant to version 3.00.

4.4 Future Work

The actual implementation of the SD host controller needs to be made. The controller would be written in Aa language and then tested for functionality using the testbench already written in C using the Aa-to-C model of the controller.

It needs to be validated that it can achieve the performance it is aimed for. For that the controller will be converted into vhdl code and tested.

Finally it should run on the operating system using the SD specific drivers.

A. Abbreviations Used

Abbreviation	Full Form
SD	Secure Digital
FPGA	Field Programmable Gate Array
SPI	Serial Peripheral Interface

References

[1] Generic AJIT processor core description and user guide by Prof. Madhav P. Desai

[2] SD Specifications Part 1 PHYSICAL LAYER Simplified Specification Version 3.01 by SD Assosciation.

[3] SD Specifications Part A2 SD Host Controller Simplified Specification Version 3.00 by SD Assosciation

[4] SD card Wikipidia

[5] How to Use MMC/SDC – elm-chan.org/docs/mmc/