DATASHEET

SD3.0/SDIO3.0/eMMC4.51 -Host Controller

SD Host Spec Ver 3.0 compliant
SD Physical Layer Spec Ver 3.0 compliant
SDIO Spec Ver 3.0 compliant
e•MMC Spec Ver 4.51 compliant

Arasan Chip Systems, Inc
2010 North First Street, Suite #510
San Jose, CA 95131
Ph:408-282-1600
Fx:408-282-7800
www.arasan.com

June 21, 2012

DISCLAIMER

This document is written in good faith with the intent to assist the readers in the use of the product. Circuit diagrams and other information relating to Arasan Chip Systems' products are included as a means of illustrating typical applications. Although the information has been checked and is believed to be accurate, no responsibility is assumed for inaccuracies. Information contained in this document is subject to continuous improvements and developments.

Arasan Chip Systems' products are not designed, intended, authorized or warranted for use in any life support or other application where product failure could cause or contribute to personal injury or severe property damage. Any and all such uses without prior written approval of an Officer of Arasan Chip Systems Inc. will be fully at the risk of the customer.

Arasan Chip Systems Inc. disclaims and excludes any and all warranties, including without limitation any and all implied warranties of merchantability, fitness for a particular purpose, title, and infringement and the like, and any and all warranties arising from any course or dealing or usage of trade.

This document may not be copied, reproduced, or transmitted to others in any manner. Nor may any use of information in this document be made, except for the specific purposes for which it is transmitted to the recipient, without the prior written consent of Arasan Chip Systems, Inc. This specification is subject to change at any time without notice. Arasan Chip Systems Inc. is not responsible for any errors contained herein.

In no event shall Arasan Chip Systems Inc. be liable for any direct, indirect, incidental, special, punitive, or consequential damages; or for lost of data, profits, savings or revenues of any kind; regardless of the form of action, whether based on contract; tort; negligence of Arasan Chip Systems or others; strict liability; breach of warranty; or otherwise; whether or not any remedy of buyers is held to have failed of its essential purpose, and whether or not Arasan Chip Systems Inc. has been advised of the possibility of such damages.

Restricted Rights
Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Copyright Notice

Questions or comments may be directed to:
Arasan Chip Systems Inc
2010 North First Street, Suite #510
San Jose, CA 95131.
Ph: 408-282-1600
Fax: 408-282-7800
Email: sales@arasan.com
www.arasan.com

CONTENTS

TABLES

FIGURES

1 INTRODUCTION

1.1 Overview

Arasan Chip Systems' SD3.0/SDIO3.0/eMMC4.51 Host Controller IP is a highly integrated Intellectual Property (IP) solution that supports three key memory and I/O technologies:

1. Secured Digital (SD)3.0
2. Secured Digital Input Output (SDIO)3.0
3. Embedded Multi Media Card (eMMC)4.5

This IP handles all of the timing and interface protocol requirements to access these media as well as processing the commands in hardware, thereby scaling in both performance and access speeds.

The IP supports connection to a single slot and performs multi-block writes and erases that lower access overhead. In addition, a host can utilize this IP to boot directly from an attached eMMC memory, thereby simplifying system initialization during power up. The host interface is based on a standard 32-bit Advanced High-Performance Bus (AHB) bus which is used to transfer data and configure the IP.

1.2 Features

- Compliant with the following specification versions:
  - Part A2 SD Host controller Version 3.00
  - Part 1 Physical Layer specification Version 3.00
  - Part E1 SDIO specification Version 3.00
  - eMMC Specification Version 4.5
  - Advanced Microprocessor Bus Architecture (AMBA), AHB Specification Version 2. 00
  - AMBA, Advanced Extensible Interface (AXI) Specification Version 1.00 (optional)
  - Open Core Protocol (OCP) specification Version 2.2 (Optional)

- The core feature supports:
  - 32-bit and 64-bit system bus
  - Interrupts and wake up functionality
  - Internal Clock divider for various card operational modes
  - One of the AHB, AXI or OCP System/Host bus
  - The data is transferred using:
    - Programmed Input Output (PIO) mode on the Host Bus Slave interface
    - Direct Memory Access (DMA) mode on the Host Bus Master interface

Note: The Host Bus is AHB or AXI or OCP

  - Configurable First In First Out (FIFO) size to support different block sizes

- UHS- I features (SD3.0/SDIO3.0) supports:
  - 1.8V voltages switch operation
  - Tuning for SDR104 and SDR50
  - Host clock rate variable between 0 and 208 MHz
  - Up to 832Mbps data rate using 4 parallel data lines (SDR104 mode)
  - Transfers the data in 1-bit and 4-bit SD modes and SPI mode
  - Transfers the data in SDR104, DDR50, SDR50, SDR25, SDR12, DS and HS modes
  - Cyclic Redundancy Check (CRC) CRC7 for command and CRC16 for data integrity
  - Variable-length data transfers
  - Performs Read wait Control, Suspend/Resume operation with SDIO CARD
  - Designed to work with I/O cards, Read-only cards and Read/Write cards
  - Card Detection (Insertion / Removal)

- eMMC4.51 feature supports:
  - eMMC4.51 Security Protocol Commands
  - Primary and alternate boot modes
  - Packed commands, Data Tags, Discard and Sanitize features
  - 4KB block support
  - Tuning for HS200 mode
  - MMC Plus and MMC Mobile
  - Host clock rate variable between 0 and 200 MHz
  - Up to 1.6Gbps (HS200) data rate using 8 bit parallel data lines
  - Transfers the data in 1-bit, 4-bit and 8-bit modes
  - CRC7 for command and CRC16 for data integrity
  - Password protection of cards

2 ARCHITECURE

2.1 Functional Description

The **Arasan SD3.0 / SDIO3.0 / eMMC4.51 Host Controller** is a Host Controller with an AHB/AXI/OCP processor interface. This product conforms to SD Host Controller Standard Specification Version 3.00.

The SD3.0/SDIO3.0/eMMC4.51 Host Controller handles SDIO/SD Protocol at transmission level, packing data, adding CRC, Start/End bit, and checking for transaction format correctness. This Host Controller provides Programmed IO method and DMA data transfer method. In programmed IO method, the Host processor transfers data using the Buffer Data Port Register.

The SD3.0/SDIO3.0/eMMC4.51 Host Controller support for DMA can be determined by checking the DMA support in the Capabilities register. DMA allows a peripheral to read or write memory without the intervention from the CPU. This Host Controller's Host Controller system address register points to the first data address, and data is then accessed sequentially from that address. It supports connection to a single slot and performs multi-block writes and erases the lower access.

2.2 Functional Block Diagram

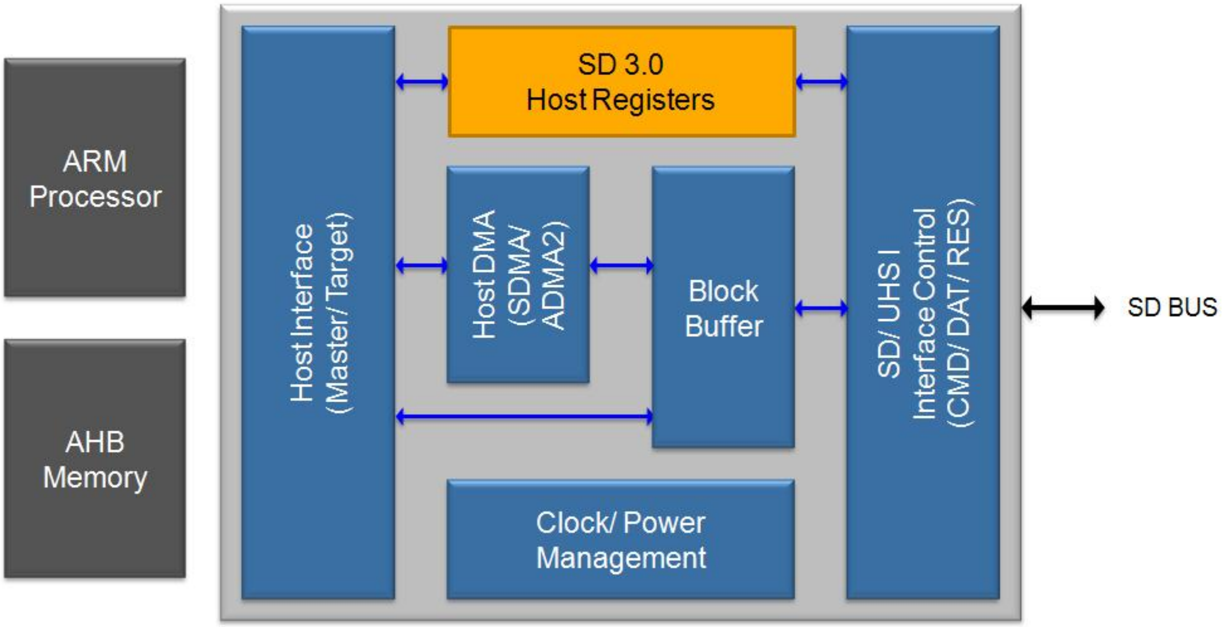

Figure 1: SD3.0/SDIO3.0/eMMC4.51 Host Controller Functional Block Diagram

2.3 Functional Block Diagram Description

2.3.1 Host Interface (Master/Target)

The Master Bus Interface is used to access the DMA Controller (when using DMA or Advanced Direct Memory Access (ADMA)2 Modes). The DMA Controller module interfaces to the Host (AHB/AXI/OCP) Master Module to generate Transfers and on the other side it interfaces with the Block Buffer to store/fetch block data. The DMA Controller implements a separate DMA for Simple Direct Memory Access (SDMA) Operation and Separate DMA for the ADMA2 Operation. In addition it implements Host Transaction Generator that generates controls for the Host Master Interface Module.

The DMA Controller uses the Master DMA Interfaces to transfer data between the Host Controller and the System Memory and vice-versa and also to fetch the Descriptors while operating in ADMA2 mode.

The Host Controller interfaces to the System bus using the AHB, AXI, or OCP Master and Slave Interface. The Slave Interface is used to access the Registers inside the Host controller. The Slave Interface supports only single transfer access (no Burst Support) and only one outstanding read/write transaction in case of AXI or OCP interface.

2.3.2 Host DMA

The PIO/DMA Controller Module implements the SDMA and ADMA2 Engines as defined in the SD Host Controller Specification and maintains the block transfer counts for PIO operation. It interacts with the Registers Set and starts the DMA Engine when a Command with Data Transfer is involved.

The DMA Controller interfaces to the Host (AHB/AXI/OCP) Master Module to generate Transfers and on the other side it interfaces with the Block Buffer to store/fetch block data. It implements a separate DMA for SDMA Operation and Separate DMA for the ADMA2 Operation and in addition implements Host Transaction Generator that generates controls for the Host Master interface module.

2.3.3 SD3.0 Host Registers

The Host Controller Register Set implements the Registers defined by the SD Host Controller Specification. The Registers are Byte/DWORD accessible from the Slave interface. The Host Controller Register Set also implements the Data Port Registers for the PIO Mode transfers.

The SD/SDIO Host Controller uses a Dual Port Block Buffer (read/write on both ports) that is used to store the Block Data during SD Transfers. The size of the Block Buffer is Configurable and has to be a minimum of 1 Block Size (Block Size is 512 Bytes in SD Memory and up to 2K Bytes in SDIO).

The Register Set provides the control signals and monitors the status signals from the blocks to set Interrupt Status Bits and eventually generate Interrupt signal to the Host Bus.

2.3.4 Block Buffer

The SD/SDIO Host Controller uses a Dual Port Block Buffer (read/write on both ports) that is used to store the Block Data during SD Transfers. The size of the Block Buffer is Configurable and has to be a minimum of 1 Block Size (Block Size is 512 Bytes in SD Memory and up to 2K Bytes in SDIO).

To achieve maximum performance the Block buffer has to be sized to twice the maximum Block Size supported by Host Controller. The Block Buffer uses Circular Buffer Architecture. One side of the Block Buffer is interfaced with the DMA Controller and operates on the Host Clock. The other side of the Block Buffer interfaces with SD Control Logic and operates on SD Clock. During a write transaction (data transferred from ARM Processor to SD3.0 / SDIO3.0 / eMMC4.51 card), the data is fetched from the System Memory and is stored in the Block Buffer. When a Block of data is available, the SD Control logic will transfer it onto the SD Interface.

The DMA Controller continues to fetch additional block of data when the Block Buffer has space. During a read transaction (data transferred from SD3.0 / SDIO3.0 / eMMC4.51 card to ARM Processor), the data from SD3.0 / SDIO3.0 / eMMC4.51 card will be written in to block buffer and at the end when the CRC of the Block is valid, the data is committed. When a block of data is available, then the DMA Controller transfers this data to the System Memory. The SD Control logic meanwhile receives the next block of data provided there is space in the Block Buffer. If the Host controller cannot accept any data from SD3.0 / SDIO3.0 / eMMC4.51 card, then it will issue read wait (if card supports read wait mechanism) to stop the data transfer from card or by stopping the clock.

Note: FIFO depth can be varied using parameter passed to the Core using the 'dot parameter instantiation'. When the Block Buffer size is twice that of the Block Size, the Block Buffer behaves like a ping-pong buffer.

2.3.5 SD/UHS-I Interface Control (CMD/DAT/RES)

The SD Interface Control block maps the internal signals to the External SD Interface and vice versa. Based on the Bus Width (1/4/8) the internal signals are driven out appropriately. In case of DS, the outputs are driven on the negative edge of the sd_clk.

The input from RxFlops module are latched on the rx_clk (looped back or tuned clock) and is output to the Receive Control Module for further processing.

2.3.6 Clock/ Power Management

The SD Clock Generator module generates the SD Clock from the Reference Clock (xin_clk), based on the Controls programmed in the Clock Control Register. These include the Clock Divide Value, SD Clock Enable and so on. The outputs from this module are the SD_CLK and the SD_CARD Clock. The SD_CLK is used by the SD Control Logic and the SD_CARD Clock connected to the "CLK" Pin on the SD Interface. This module also generates system resets to various clock domains.

3 PINOUTS

3.1 I/O Description

The Arasan SD3.0/SDIO3.0/eMMC4.51 Host Controller has following interface groups:

- System (AHB/AXI/OCP) Bus Interface Signals
- SD3.0 / SDIO3.0 / eMMC4.51 Interface that forms the main card interface
- Power Control Signals
- Clock, Special Controls and Test Mode Signals
- Block RAM, Static Random Access Memory (SRAM) Interface Signals
- Core Configuration Signals

Table 1: AHB Bus Interface Signals

Pin	Direction	Description
ahb_clk	IN	AHB System Clock.
ahb_reset_n	IN	AHB System Reset (Active Low)
ahbmaster_hbusreq	OUT	AHB Bus request
ahbmaster_hgrant	IN	AHB Bus Grant
ahbmaster_haddr[31:0]	OUT	DWord Address (Assuming 32-bit Address Bus)
ahbmaster_hwdata[31:0]	OUT	AHB master write data (Assuming 32-bit Data Bus width)
ahbmaster_hrdata[31:0]	IN	Read data (Assuming 32-bit Data Bus width)
ahbmaster_hwrite	OUT	Write / Read Direction Indication
ahbmaster_hsize[2:0]	OUT	Size (byte, half word or word)
ahbmaster_hburst[2:0]	OUT	Burst Size
ahbmaster_hready	IN	Ready signal
ahbmaster_htrans[1:0]	OUT	Transfer type
ahbmaster_hresp[1:0]	IN	Transfer response
ahb_intr	OUT	Interrupt to the ARM
ahb_wkup	OUT	Wakeup Indication to ARM
ahbtarget_hsel	IN	Slave Select
ahbtarget_haddr[15:0]	IN	DWord Address (256 bytes)
ahbtarget_hwdata[31:0]	IN	Write Data
ahbtarget_hrdata[31:0]	OUT	Read Data
ahbtarget_hwrite	IN	Write / Read Direction Indication
ahbtarget_hsize[2:0]	IN	Size (Byte, Half Word or Word)
ahbtarget_htrans[1:0]	IN	Transfer Type
ahbtarget_hready_in	IN	Slave Ready Input
ahbtarget_hready	OUT	Slave Ready

Pin	Direction	Description
ahbtarget_hresp[1:0]	OUT	Transfer Response

Note: Target Interface doesn't support BURST transaction.

Table 2: OCP Bus Interface Signals

Pins	Direction	Description
clk_ocp	IN	OCP System Clock.
OCPMaster_MAddr	OUT	OCP Master read/write address. The address bus width is based on the selected OCP Address width using the SDHC_MSTOCP_AW parameter at the SDHC_OCP_TOP Instantiation
OCPMaster_MCmd[2:0]	OUT	Indicates the type of transaction that the OCP Master has initiated
OCPMaster_MData	OUT	Write data from OCP Master to the slave The Data Bus width can be 32 or 64 bits wide and is based on the selected bus width using the SDHC_MSTOCP_DW parameter at the SDHC_OCP_TOP Instantiation.
OCPMaster_MDataValid	OUT	Is the qualifier for OCPMAster_MData
OCPMaster_SCmdAccept	IN	Indicates that the OCP Slave has accepted the command
OCPMaster_SData	IN	Read data from OCP Slave The Data Bus width can be 32 or 64 bits wide and is based on the selected bus width using the SDHC_MSTOCP_DW parameter at the SDHC_OCP_TOP Instantiation.
OCPMaster_SDataAccept	IN	Asserted by OCP slave to indicate that the current Master write data is accepted
OCPMaster_SResp[1:0]	IN	Response signal for Master write transfers
OCPMaster_MByteEn[3:0]	OUT	Byte enable from the master for write/read transactions
OCPMaster_MBurstLength[4:0]	OUT	Indicates the burst length of the transaction
OCPMaster_MBurstPrecise	OUT	Indicates that the given burst length is precise
OCPMaster_MBurstSeq[2:0]	OUT	Indicates the type of burst
OCPMaster_MBurstSingleReq	OUT	Indicates the number of requests associated with the burst
OCPMaster_MDataLast	OUT	Last Data of the burst
OCPMaster_MReqLast	OUT	Last request in a burst
OCPMaster_SRespLast	IN	Last response in a burst
OCPMaster_MDataByteEn[3:0]	OUT	Write Byte enables to the OCP slave during Data handshake phase
OCPMaster_max_burst_size_config	IN	Configurable burst length
OCPSlave_MCmd[2:0]	IN	Type of transaction from the Master
OCPSlave_MAddr[31:0]	IN	Transfer address from Master

Pins	Direction	Description
OCPSlave_MData[31:0]	IN	Write data from OCP Master
OCPSlave_SCmdAccept	OUT	Acceptance signal to the external OCP master for the request phase
OCPSlave_SResp[1:0]	OUT	Response signal from OCP Slave
OCPSlave_SData[31:0]	OUT	Read data from OCP Slave
OCPSlave_MByteEn[3:0]	IN	Byte Enable from OCP Master
OCPSlave_MReset_n	IN	Reset signaling from OCP Master
OCPSlave_MRespAccept	IN	Master accepts response

Table 3: AXI Bus Interface Signals

Pins	Direction	Description
aximst_wstrb	OUT	Write strobes. This signal indicates which byte lanes to update in memory. There is one write strobe for each eight bits of the write data bus. Therefore, WSTRB[n] corresponds to WDATA[(8 Þ n) + 7:(8 Þ n)].
aximst_wlast	OUT	Write last. This signal indicates the last transfer in a write burst.
aximst_wvalid	OUT	Write valid. This signal indicates that valid write data and strobes are available: 1 = write data and strobes available 0 = write data and strobes not available.
aximst_wready	IN	Write ready. This signal indicates that the slave can accept the write data: 1 = slave ready 0 = slave not ready.
aximst_bid[3:0]	IN	Response ID. The identification tag of the write response. The BID value must match the AWID value of the write transaction to which the slave is responding.
aximst_bresp[1:0]	IN	Write response. This signal indicates the status of the write transaction. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
aximst_bvalid	IN	Write response valid. This signal indicates that a valid write response is available: 1 = write response available 0 = write response not available.
aximst_bready	IN	Response ready. This signal indicates that the master can accept the response information. 1 = master ready 0 = master not ready.
aximst_arid[3:0]	OUT	Read address ID. This signal is the identification tag for the read address group of signals.
aximst_araddr[31:0]	OUT.	Read address. The read address bus gives the initial address of a read burst transaction. Only the start address of the burst is provided and the control signals that are issued alongside the address detail how the address is calculated for the remaining transfers in the burst
aximst_arlen[3:0]	OUT	Burst length. The burst length gives the exact number of

Pins	Direction	Description
		transfers in a burst. This information determines the number of data transfers associated with the address.
aximst_arsize[2:0]	OUT	Burst size. This signal indicates the size of each transfer in the burst.
aximst_arburst[1:0]	OUT	Burst type. The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated.
aximst_arvalid	OUT	Read address valid. This signal indicates, when HIGH, that the read address and control information is valid and will remain stable until the address acknowledge signal, ARREADY, is high. 1 = address and control information valid 0 = address and control information not valid.
aximst_rid[3:0]	IN	Read ID tag. This signal is the ID tag of the read data group of signals. The RID value is generated by the slave and must match the ARID value of the read transaction to which it is responding.
aximst_rdata	IN	Read data. The read data bus can be 8, 16, 32, 64, 128, 256, 512, or 1024 bits wide.
aximst_rresp[1:0]	IN	Read response. This signal indicates the status of the read transfer. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
aximst_rlast	IN	Read last. This signal indicates the last transfer in a read burst.
aximst_rvalid	IN	Read valid. This signal indicates that the required read data is available and the read transfer can complete: 1 = read data available 0 = read data not available
aximst_rready	OUT	Read ready. This signal indicates that the master can accept the read data and response information: 1= master ready 0 = master not ready.
axislv_awid[7:0]	IN	Write address ID. This signal is the identification tag for the write address group of signals.
axislv_awaddr[31:0]	IN	Write address. The write address bus gives the address of the first transfer in a write burst transaction. The associated control signals are used to determine the addresses of the remaining transfers in the burst.
axislv_awlen[3:0]	IN	Burst length. The burst length gives the exact number of transfers in a burst. This information determines the number of data transfers associated with the address.
axislv_awsize[2:0]	IN	Burst size. This signal indicates the size of each transfer in the burst. Byte lane strobes indicate exactly which byte lanes to update.
axislv_awburst[1:0]	IN	Burst type. The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated.
axislv_awvalid	IN	Write address valid. This signal indicates that valid write address and control information are available:

Pins	Direction	Description
		1 = address and control information available 0 = address and control information not available The address and control information remain stable until the address acknowledge signal, AWREADY, goes HIGH.
axislv_awready	OUT	Write address ready. This signal indicates that the slave is ready to accept an address and associated control signals: 1 = slave ready 0 = slave not ready.
axislv_wid[7:0]	IN	Write ID tag. This signal is the ID tag of the write data transfer. The WID value must match the AWID value of the write transaction.
axislv_wdata[31:0]	IN	Write data. The write data bus can be 8, 16, 32, 64, 128, 256, 512, or 1024 bits wide.
axislv_wstrb[3:0]	IN	Write strobes. This signal indicates which byte lanes to update in memory. There is one write strobe for each eight bits of the write data bus. Therefore, WSTRB[n] corresponds to WDATA[(8 Þ n) + 7:(8 Þ n)].
axislv_wlast	IN	Write last. This signal indicates the last transfer in a write burst.
axislv_wvalid	IN	Write valid. This signal indicates that valid write data and strobes are available: 1 = write data and strobes available 0 = write data and strobes not available.
axislv_wready	OUT	Write ready. This signal indicates that the slave can accept the write data: 1 = slave ready 0 = slave not ready.
axislv_bid[7:0]	OUT	Response ID. The identification tag of the write response. The BID value must match the AWID value of the write transaction to which the slave is responding.
axislv_bresp[1:0]	OUT	Write response. This signal indicates the status of the write transaction. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
axislv_bvalid	OUT	Write response valid. This signal indicates that a valid write response is available: 1 = write response available 0 = write response not available.
axislv_bready	IN	Response ready. This signal indicates that the master can accept the response information. 1 = master ready 0 = master not ready
axislv_arid[7:0]	IN	Read address ID. This signal is the identification tag for the read address group of signals.
axislv_araddr[31:0]	IN	Read address. The read address bus gives the initial address of a read burst transaction. Only the start address of the burst is provided and the control signals that are issued alongside the address detail how the address is calculated for the remaining transfers in the burst.
axislv_arlen[3:0]	IN	Burst length. The burst length gives the exact number of

Pins	Direction	Description
		transfers in a burst. This information determines the number of data transfers associated with the address.
axislv_arsize[2:0]	IN	Burst size. This signal indicates the size of each transfer in the burst.
axislv_arburst[1:0]	IN	Burst type. The burst type, coupled with the size information, details how the address for each transfer within the burst is calculated.
axislv_arvalid	IN	Read address valid. This signal indicates, when HIGH, that the read address and control information is valid and will remain stable until the address acknowledge signal, ARREADY, is high. 1 = address and control information valid 0 = address and control information not valid.
axislv_arready	OUT	Read address ready. This signal indicates that the slave is ready to accept an address and associated control signals: 1 = slave ready 0 = slave not ready.
axislv_rid[7:0]	OUT	Read ID tag. This signal is the ID tag of the read data group of signals. The RID value is generated by the slave and must match the ARID value of the read transaction to which it is responding.
axislv_rdata[31:0]	OUT	Read data. The read data bus can be 8, 16, 32, 64, 128, 256, 512, or 1024 bits wide.
axislv_rresp[1:0]	OUT	Read response. This signal indicates the status of the read transfer. The allowable responses are OKAY, EXOKAY, SLVERR, and DECERR.
axislv_rlast	OUT	Read last. This signal indicates the last transfer in a read burst.
axislv_rvalid	OUT	Read valid. This signal indicates that the required read data is available and the read transfer can complete: 1 = read data available 0 = read data not available.
axislv_rready	IN	Read ready. This signal indicates that the master can accept the read data and response information: 1= master ready 0 = master not ready.
int_to_arm	OUT	Interrupt to the ARM
cfg_mstid	IN	programmable ID for master interface

Table 4: SD3.0 / SDIO3.0 / eMMC4.51 Interface

Pins	Direction	Description
sdif_cd_n	IN	Active Low. Card Detection for single Slot
sdif_wp	IN	Active High. SD Card Write Protect
sdif_clkout	OUT	SD/SDIO/MMC Clock to Card (CLK)
rxclk_in	IN	SD/SDIO/MMC Clock looped back from PAD
sdif_cmdin	IN	SD1/SD4/MMC8 : Command Input

Pins	Direction	Description
sdif_cmdout **sdif_cmdena**	OUT OUT	SD1/SD4/MMC8: Command Output SPI : Command output and write data SD1/SD4/MMC8 mode: Command Output Enable SPI mode : Command output enable and write data enable
sdif_dat0in **sdif_dat0out** **sdif_dat0en**	IN OUT OUT	SD1/SD4/MMC8 mode: Data0 Input SPI mode : Command response input, read data and CRC status for write data SD1/SD4/MMC8 mode: Data0 Output SD1/SD4/MMC8 mode: Data0 Output Enable
sdif_dat1in **sdif_dat1out** **sdif_dat1en**	IN OUT OUT	SD1 mode: Interrupt SD4 mode: Data1 Input or Interrupt (optional) MMC8 mode: Data1 Input SD4/MMC8 mode: Data1 Output SD4/MMC8 mode: Data1 Output Enable
sdif_dat2in **sdif_dat2out** **sdif_dat2en**	IN OUT OUT	SD4/MMC8 mode: Data2 Input SD1 mode: Read Wait(optional) SD4 mode: Data2 Output or Read Wait (optional) MMC8 mode: Data2 Output SD1 mode: Read Wait Enable(optional) SD4 mode: Data2 Output Enable or Read Wait Enable (optional) MMC8 mode: Data2 Output Enable
sdif_dat3in **sdif_dat3out** **sdif_dat3en**	IN OUT OUT	SD4/MMC8 mode: Data3 Input SD4/MMC8 mode: Data3 Output SPI mode : chip select SD4/MMC8 mode: Data3 Output Enable SPI mode : chip select enable
sdif_dat4in **sdif_dat4out** **sdif_dat4en**	IN OUT OUT	MMC8 mode: Data4 Input MMC8 mode: Data4 Output MMC8 mode: Data4 Output Enable
sdif_dat5in **sdif_dat5out** **sdif_dat5en**	IN OUT OUT	MMC8 mode: Data5 Input MMC8 mode: Data5 Output MMC8 mode: Data5 Output Enable
sdif_dat6in **sdif_dat6out** **sdif_dat6en**	IN OUT OUT	MMC8 mode: Data6 Input MMC8 mode: Data6 Output MMC8 mode: Data6 Output Enable
sdif_dat7in **sdif_dat7out** **sdif_dat7en**	IN OUT OUT	MMC8 mode: Data7 Input MMC8 mode: Data7 Output MMC8 mode: Data7 Output Enable

Note:
CMD/ DATA output enables are active high signals.

Table 5: Power Control Signals

Pins	Direction	Description
sdhc_ledcontrol	OUT	LED ON: To Caution the user not to remove the card while the SD card is being accessed.
sdhc_sdbuspower	OUT	Control SD Card Power Supply.
sdhc_sdbusvoltage[2:0]	OUT	SD Bus voltage select.
sdhc_1p8vsigenable	OUT	1.8V Signaling Enable
sdhc_driverstrength[1:0]	OUT	Driver Strength Select 00b Driver Type B is Selected 01b Driver Type A is Selected 10b Driver Type C is Selected 11b Driver Type D is Selected

Table 6: Clock, Special Controls and Test Mode Signals

Pins	Direction	Description
xin_clk	IN	This clock input is used to generate SD Clock. For maximum efficiency this should be around 50MHz (for SD)/ 208MHz (for SD3.0)
corectrl_itapdlyena	IN	Used to enable selective Tap delay line on the Looped back SD Clock (rxclk_in). This signal along with the corectrl_itapdlysel[4:0] selects the amount of delay to be inserted on the line. When Tuning is enabled (for SDR104 and optionally for SDR50), this signal is ignored and internal controls are used instead. This should not be asserted when operating in DS mode.
corectrl_itapdlysel[4:0]	IN	Selects one of the 32 Taps on the rxclk_in line. This is effective only when corectrl_itapdlyena is asserted and Tuning is not enabled.
corectrl_itapchgwin	IN	This is used to gate the output of the Tap Delay lines so as to avoid glitches being propagated into the Core. This signal should be asserted few clocks before the corectrl_itapdlysel changes and should be asserted for few clocks after.
corectrl_otapdlyena	IN	Used to enable the selective Tap delay on the sdcard_clk so as to generate the delayed sdcard_clk. This is used to latch the CMD/DAT outputs to generate delay on them w.r.t CLK going out. This signal along with corectrl_otapdlysel[3:0] selects the amount of delay to be inserted on the Clock line. This signal should not be asserted when operating in DS mode
corectrl_otapdlysel[3:0]	IN	Selects one of the 16 Taps on the sdcard_clk. This is effective only when corectrl_otapdlyena is asserted.

Pins	Direction	Description
test_mode	IN	Test mode signal is used for DFT purpose. Muxes in the AXI_reset_n signal for all internally generated resets. (Active High)
scan_mode	IN	Scan Mode signal for selecting Scan Clocks for internally generated clocks
scan_clk1	IN	Scan Clock#1 used to mux in for the internally generated sd_clk
scan_clk2	IN	Scan Clock#2 used to mux in for the final rxclk_in (after the tap delay etc)

Table 7: Block RAM (SRAM) Interface Signals

Pins	Direction	Description
sram_clka	OUT	Clock for PORT A
sram_addra [N-1:0]	OUT	Address bus for PORT A. The width of the Address bus is based on the size of the SRAM (SDHC_BUFFER_SIZE)
sram_writea	OUT	Write Enable for PORT A
sram_reada	OUT	Read Enable for PORT A
sram_wrdata[N-1:0]	OUT	Write Data for PORT A. N is based on the SDHC_MSTAXI_DW parameter
sram_rddataa [N-1:0]	IN	Read Data from SRAM on PORT A. N is based on the SDHC_MSTAXI_DW parameter
sram_clkb	OUT	Clock for PORT B
sram_addrb [N-1:0]	OUT	Address bus for PORT B. The width of the Address bus is based on the size of the SRAM (SDHC_BUFFER_SIZE)
sram_writeb	OUT	Write Enable for PORT B
sram_readb	OUT	Read Enable for PORT B
sram_wrdatab [N-1:0]	OUT	Write Data for PORT B. N is based on the SDHC_MSTAXI_DW parameter
sram_rddatab [N-1:0]	IN	Read Data from SRAM on PORT B. N is based on the SDHC_MSTAXI_DW parameter

Table 8: Core Configuration Signals

Pins	Direction	Description
corecfg_tuningcount[5:0]	IN	Configures the Number of Taps (Phases) of the rxclk_in that is supported. The Tuning State machine uses this information to select one of the Taps (Phases) of the rxclk_in during the Tuning Procedure.
corecfg_timeoutclkfreq[5:0]	IN	Timeout Clock Frequency Suggested Value is 1. (KHz or MHz). Internally the 1msec /1usecTimer is used for Timeout Detection. The 1msec Timer is generated from the xin_clk.
corecfg_timeoutclkunit	IN	Timeout Clock Unit Suggested value is 1'b1 to Select MHz Clock.
corecfg_baseclkfreq[7:0]	IN	Base Clock Frequency for SD Clock. This is the frequency of the xin_clk.
corecfg_maxblklength[1:0]	IN	Max Block Length Maximum Block Length supported by the Core/Device 00: 512 (Bytes) 01: 1024 10: 2048 11: Reserved
corecfg_8bitsupport	IN	8-bit Support for Embedded Device Suggested Value is 1'b1 (The Core supports 8-bit Interface). Optionally an be set to 1'b0 if the Application supports only 4-bit SD Interface.
corecfg_adma2support	IN	ADMA2 Support Suggested Value is 1'b1 (The ADMA2 is supported by Core). Optionally can be set to 1'b0 if the application doesn't want to support ADMA2 Mode
corecfg_highspeedsupport	IN	High Speed Support Suggested Value is 1'b1 (The High Speed mode is supported by Core).
corecfg_sdmasupport	IN	SDMA Support Suggested Value is 1'b1 (The SDMA is supported by Core). Optionally can be set to 1'b0 if the application doesn't want to support SDMA Mode
corecfg_suspressupport	IN	Suspend/Resume Support Suggested Value is 1'b1 (The Suspend/Resume is supported by Core). Optionally can be set to 1'b0 if the application doesn't want to support Suspend/Resume Mode
corecfg_3p3voltsupport	IN	3.3V Support Suggested Value is 1'b1 as the 3.3 V is the default voltage on the SD Interface.

Pins	Direction	Description
corecfg_3p0voltsupport	IN	3.0V Support Should be set based on whether 3.0V is supported on the SD Interface.
corecfg_1p8voltsupport	IN	1.8V Support Suggested Value is 1'b1 (The 1.8 Volt Switching is supported by Core). Optionally can be set to 1'b0 if the application doesn't want 1.8V switching (SD3.0)
corecfg_64bitsupport	IN	64-bit System Bus Support This should be set based on the System Address Bus. When set to 1'b0 the Core supports only 32-bit System Bus. When set to 1'b1 the Core supports 64-bit System Address.
corecfg_asyncintrsupport	IN	Asynchronous Interrupt Support Suggested Value is 1'b1 (The Core supports monitoring of Asynchronous Interrupt)
corecfg_slottype[1:0]	IN	Slot Type Should be set based on the final product usage 00 - Removable SCard Slot 01 - Embedded Slot for One Device 10 - Shared Bus Slot 11 – Reserved
corecfg_sdr50support	IN	SDR50 Support Suggested Value is 1'b1 (The Core supports SDR50 mode of operation) Optionally can be set to 1'b0 if the application doesn't want to support SDR50
corecfg_sdr104support	IN	SDR104 Support Suggested Value is 1'b1 (The Core supports SDR104 mode of operation) Optionally can be set to 1'b0 if the application doesn't want to support SDR104
corecfg_ddr50support	IN	DDR50 Support Suggested Value is 1'b1 (The Core supports DDR50 mode of operation) Optionally can be set to 1'b0 if the application doesn't want to support DDR50
corecfg_adriversupport	IN	Driver Type A Support This bit should be set based on whether Driver Type A for 1.8 Signaling is supported or not.
corecfg_cdriversupport	IN	Driver Type C Support This bit should be set based on whether Driver Type C for 1.8 Signaling is supported or not.
corecfg_ddriversupport	IN	Driver Type D Support This bit should be set based on whether Driver Type D for 1.8 Signaling is supported or not.
corecfg_retuningtimercnt[3:0]	IN	Timer Count for Re-Tuning

Pins	Direction	Description
		This is the Timer Count for Re-Tuning Timer for Re-Tuning Mode 1 to 3. Setting to 4'b0 disables Re-Tuning Timer.
corecfg_tuningforsdr50	IN	Use Tuning for SDR50 This bit should be set if the Application wants Tuning be used for SDR50 Modes. The Core operates with or without tuning for SDR50 mode as long as the Clock can be manually tuned using tap delay.
corecfg_retuningmodes[1:0]	IN	Re-Tuning Modes Should be set to 2'b00 as the Core supports only the Mode0 Retuning.
corecfg_spisupport	IN	SPI Mode Support Suggested Value is 1'b1 (The Core supports SPI mode of operation) Optionally can be set to 1'b0 if the application doesn't want to support SPI Mode
corecfg_spiblkmode	IN	SPI Block Mode Reserved and should be set to1'b0
corecfg_initpresetval[12:0]	IN	Preset Value for Initialization.
corecfg_dsppresetval[12:0]	IN	Preset Value for Default Speed
corecfg_hsppresetval[12:0]	IN	Preset Value for High Speed
corecfg_sdr12presetval[12:0]	IN	Preset Value for SDR12
corecfg_sdr25presetval[12:0]	IN	Preset Value for SDR25
corecfg_sdr50presetval[12:0]	IN	Preset Value for SDR50
corecfg_sdr104presetval[12:0]	IN	Preset Value for SDR104
corecfg_ddr50presetval[12:0]	IN	Preset Value for DDR50
corecfg_maxcurrent1p8v[7:0]	IN	Maximum Current for 1.8V
corecfg_maxcurrent3p0v[7:0]	IN	Maximum Current for 3.0V
corecfg_maxcurrent3p3v[7:0]	IN	Maximum Current for 3.3V
corecfg_asyncwkupena	IN	Determines the Wakeup Signal Generation Mode. **0: Synchronous Wakeup Mode:** The xin_clk has to be running for this mode. The Card Insertion/Removal/Interrupt events are detected synchronously on the xin_clk and the Wakeup Event is generated. The Assertion and deassertion of the wakeup Event signal synchronous to xin_clk. **1: Asyncrhonous Wakeup Mode:** The xin_clk and the host_clk can be stopped in this mode and the Wake up Event is asynchronously generated based on the Card Insertion/Removal/Interrupt Events. The Assertion and de-assertion of the wakeup Event signal is asynchronous.

4 SoC LEVEL INTEGRATION

4.1 IP Deliverables for RTL Version

- Verilog HDL of the IP Core
- Synthesis scripts
- Test Environment and test scripts
- User guide

4.2 Verification Environment

This section provides information on the architecture of the SD Host Controller Verification Environment.

The SD Host Controller Design Under Test (DUT) is written in synthesizable Verilog. On the processor side it interfaces with AXI/AHB/OCP Master BFM and AXI/AHB/OCP Slave BFM. On the device side it interfaces with a user selectable device BFM (SD, SDIO, eMMC). The Host Controller Capability is selected by connecting the capability pins to power and ground. The device side interface is 8 bits wide bi-directional data lane supporting DDR (double data rate). The width of the bus is user selectable (1, 4 or 8).The data can be sent or received either by processor input and output or by DMA. The data flow can be aborted by the processor side sending an abort command. The data transfer goes through a ping-pong buffer which achieves back to back frames transfer.

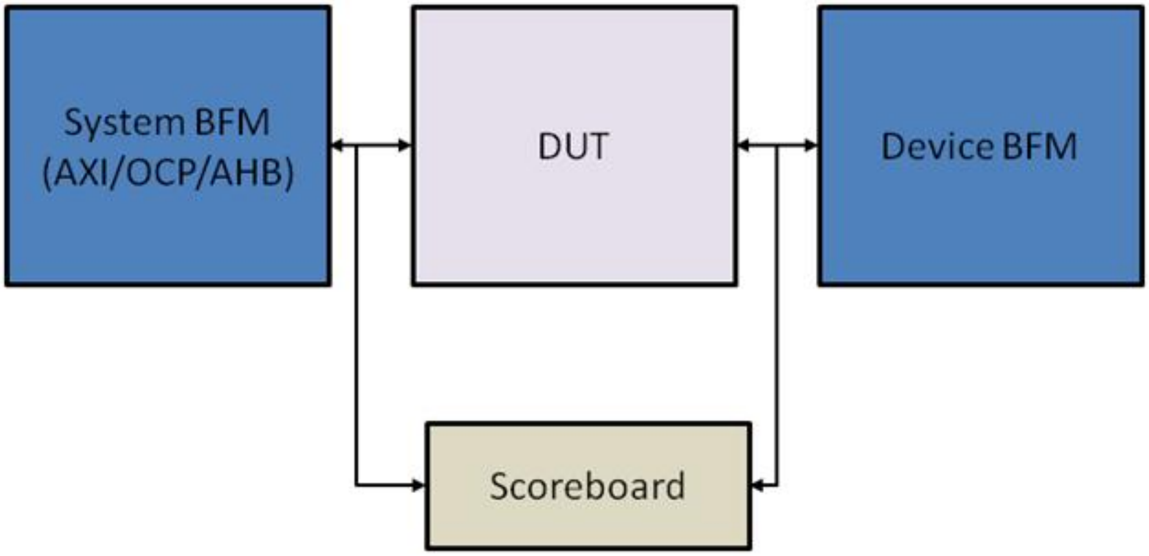

Figure 2: Verification Environment

4.3 Related Items for Total Solution

- SD3.0 Device IP
- eMMC4.51 Device IP
- SD3/eMMC4.51 Hardware Validation Platform
- SD4.0 Device IP
- SD4/SDIO4/eMMC4.51 Host Controller IP

Note: Arasan Chip Systems Inc. is an Executive Member of the SD Card Association since 2001 and participates in the SD/SDIO Standards and Compliance Workgroups.