SD Flash Controller Using SD Bus

April 2011 **Reference Design RD1088**

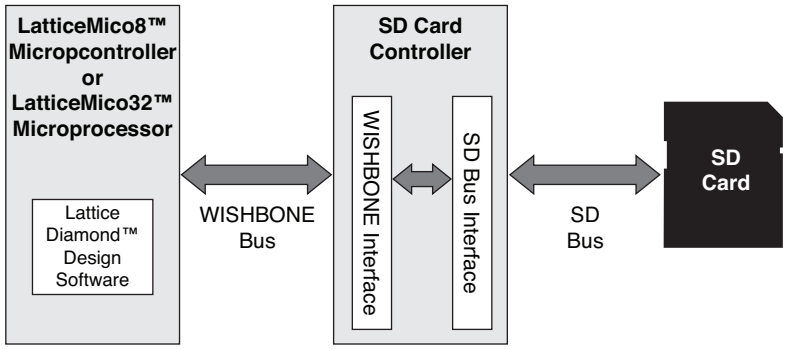
Introduction

The Security Digital (SD) memory card has become a standard data and media storage medium for mobile electronic devices. In order to access the SD memory, a SD Flash Controller is required for communication between the host system and the SD card socket. The SD card supports three different transfer modes: 1-bit SD mode, 4-bit SD mode, and SPI mode. The SD Flash Controller must be able to support at least one of these modes in order to access the memory.

This design implements a SD Flash Controller based on the OpenCores SD Card Mass Storage Controller design. The controller connects the SD card on one side and the WISHBONE bus on the other. All transmission and reception between the host and SD card comply with the SD Physical Layer Simplified Specification 2.0 released by the SD Card Association. This reference design is designed to work with a file system where a SD card can be recognized as a system disk.

This design supports DMA, interrupts, and buffered read/write features and includes a simplified model of a SD card for simulation purposes. The design can be used for controlling a SD card, micro SD card or SDHC card. Figure 1 shows a typical application environment for this reference design.

Figure 1. SD Flash Controller Application Environment



Features

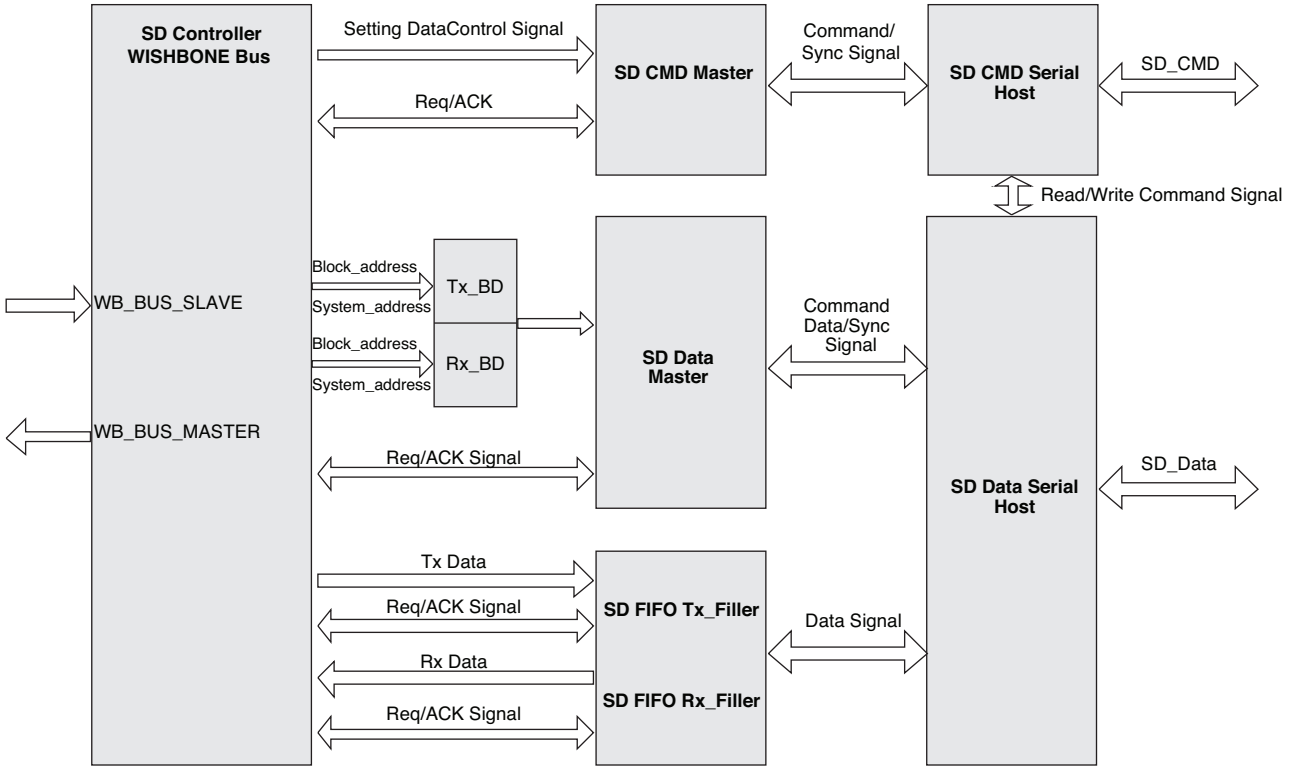
The SD Flash Controller includes the features listed below:

- 32-bit WISHBONE interface
- DMA
- Buffer Descriptor (BD)
- Compliant with SD Physical Layer Simplified Specification 2.0
- Supports 4-bit SD mode
- Write/read FIFO with variable size
- Internal implementation of CRC16 for data lines and CRC7 for command lines

Functional Description

The SD Flash Controller reference design consists of several modules. These modules provide the necessary logic for the processor to fully access the SD Flash memory card. A high-level description of the function of each module is provided later in this document. Figure 2 gives an overview of the design architecture.

Figure 2. SD Flash Controller Simplified Internal Architecture



SD Controller Top

This module is the interface between the SD Controller core and the WISHBONE bus. Two WISHBONE interfaces (slave and master) are used in this design. The WISHBONE slave interface is used to access the internal registers, setting and buffer descriptors and the WISHBONE master interface is used by the internal DMA to fetch and store data to and from an external memory without going through the host CPU.

SD CMD Master

This module receives the command from the host and monitors the status of command transmitting and data/response receiving units. It performs the following tasks:

- Receives the command from the host and combines it with data for the transmitting module to process
- Receives responses from the SD card and generates the necessary flags for the host
- Monitors the status of the host module of the controller

SD CMD Host

This module is the command transmitting and response receiving control unit. It adds start bits, stop bits and the CRC checksum to the command stream in addition to transmitting and receiving command between the host and the SD card.

SD Data Master

This module handles new data transfer requests. If a new data transfer request is received, it generates the data transfer command for the command module first. It then checks the status of the SD card and the CRC when transmission is complete to determine if a stop command is necessary.

SD Data Host

This module interfaces directly to the SD card device data port. The interface consists of five signals, including a clock signal (SDCLK) and a 4-bit bi-directional data bus (DAT). The module synchronizes requests for write and read data, adds a CRC-16 checksum on sent data, and checks for the correct CRC-16 on received commands.

SD Buffer Descriptor (BD)

Two sequential writings to the register TX_BD or RX_BD will generate a data transfer command, or buffer descriptor. The first write provides the source address (memory location) and the second write provides the data lock destination address.

SD FIFO Rx/Tx Filler

These modules are responsible for the data stream FIFO for transmit and receive. The FIFO status is also monitored.

Interface

The SD Flash Controller has two WISHBONE interfaces and one SD physical interface. Tables 1 and 2 provide the port descriptions for the design.

Table 1. WISHBONE Interfaces

Port	Width	Direction	Description
WB_CLK_I	1	Input	Slave WISHBONE Clock Input
WB_RST_I	1	Input	Slave WISHBONE Reset Input
WB_SEL_I	4	Input	Slave WISHBONE Select Inputs
WB_DAT_I	32	Input	Slave WISHBONE Data Inputs
WB_DAT_O	32	Output	Slave WISHBONE Data Output
WB_ADR_I	8	Input	Slave WISHBONE Address Input
WB_WE_I	1	Input	Slave WISHBONE Write Enable
WB_CYC_I	1	Input	Slave WISHBONE Cycle
WB_STB_I	1	Input	Slave WISHBONE Strobe
WB_ACK_O	1	Output	Slave WISHBONE Acknowledgment
M_WB_ADR_O	32	Output	Master WISHBONE Address
M_WB_SEL_O	1	Output	Master WISHBONE Select
M_WB_WE_O	1	Output	Master WISHBONE Write Enable
M_WB_DAT_O	32	Output	Master WISHBONE Data Output
M_WB_DAT_I	32	Input	Master WISHBONE Data Input
M_WB_CYC_O	1	Output	Master WISHBONE Cycle
M_WB_ACK_I	1	Input	Master WISHBONE Acknowledgment Input
M_WB_CTI_O	1	Output	Master WISHBONE Cycle Type Identifier
M_WB_BTE_O	1	Output	Master WISHBONE Burst Type Extension

Table 2. SD I/O Ports

Port	Width	Direction	Description
SD_CMD_DAT_I	1	Input	SDC/MMC CMD Input
SD_CMD_OUT_O	1	Output	SDC/MMC CMD Output
SD_CMD_OE_O	1	Output	SDC/MMC CMD Output Enable
SD_DAT_DAT_I	4	Input	SDC/MMC Data Input
SD_DAT_OUT_O	4	Output	SDC/MMC Data Output
SD_DAT_OE_O	1	Output	SDC/MMC Data Output Enable
SD_CLK_O_PAD	1	Output	SDC/MMC CLK Output
SD_CLK_I_PAD	1	Input	SDCLK Input
INT_A	1	Output	Interrupt A Output
INT_B	1	Output	Interrupt B Output
INT_C	1	Output	Interrupt C Output

Note: Command and data ports of the SD interface can be implemented as bi-directional I/O ports by combining the input/output/OE into a single port in the source code.

Design Module Descriptions

SD_CMD_SERIAL_HOST

An interface to the CMD port of the external SD card is implemented in this module. Through this interface the user commands and configures parameters sent to the SD card and the corresponding response data received from the SD card. The external interface consists of two signals, CLK and a bi-directional CMD signal. The CMD_OUT_O, CMD_DAT_I and CMD_OE_O signals must be combined in an additional module (preferably the test bench top module).

Signals and State Machine

After power-up, this module is in the init state. In init state, it waits for 64 cycles and then a '1' is written to the SD card in the CMD port. The state machine then shifts to idle state. The SD card does not respond in this case.

In idle state, the module waits for the request (REQ_IN) from the master. Once the request is received, the first task is to synchronize the REQ_IN into this clock domain. It then decodes the settings (user configuration information) into the following signals:

- RESPONSE_SIZE
- CRC_CHECK_ON
- DELAY_CYCLER
- BLOCK_WRITE
- BLOCK_READ
- WORD_SELECT

The RESPONSE_SIZE signal is the response size for the SD bus mode. It has two values: 48 and 136. If the RESPONSE_SIZE is larger than 48, the user should use the WORD_SELECT signal to select the different words in the response data.

CRC_CHECK_ON is the CRC ON/OFF configuration signal. When this signal is valid, the CRC checksum should be verified for the received response data.

DELAY_CYCLER is a delay counter configuration signal. Its default value is 7. In SD bus mode, there is no response for some commands. After transmitting commands of this type, the module should wait for eight cycles to enter next idle state.

The block write and block read signals are the starting signals for reading and writing data commands.

The CMD_IN input signal is the command signal and it includes the command index and argument parameter. It is stored in the IN_BUFF register for transmitting.

After processing these signals, the DECODER_ACK signal is asserted low. The acknowledge signal is combined with two signals, DECODER_ACK and FSM_ACK. When the main state machine transfers data to the SD card, FSM_ACK is at the low level until the data transfer is complete.

Depending on the size of the response, the state machine will shift to two different branches. One branch is for write-only commands and the other for write-read commands.

For write-only commands, the SD card has no response. Once the command is transmitted completely, the state machine enters the turn-around state and waits eight cycles before releasing the ack signal. It then enters the idle state to wait for the next request signal.

For write-read commands, the SD card transmits the response to the controller. The output response is saved in out-buff registers according to the RESPONSE_SIZE. If the CRC_CHECK_ON is valid, the controller will verify the CRC checksum of the received response. After verification, the state machine enters the turn-around state to release the ack signal. At this point, the state machine enters the idle state.

A status signal is used to monitor the state machine, the response and the CRC checksum. The lower four bits are used to monitor the state. The status [5] is used to monitor the verified CRC result. The status [6] is used to assert the data valid for the transmitting command.

Table 3. CMD_SERIAL_HOST Setting In Register

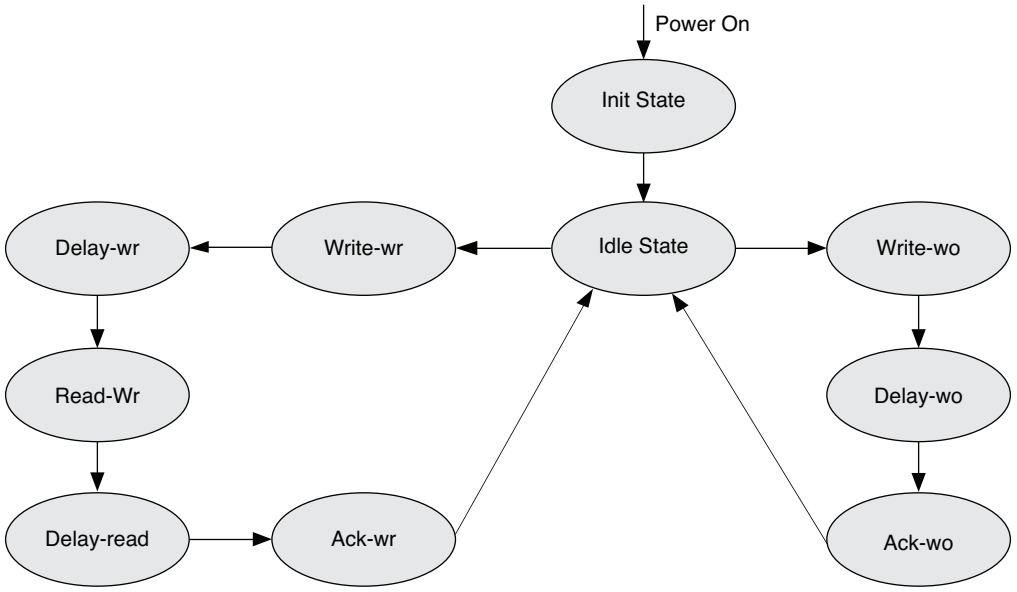
Bit	15	[14:13]	12	11	[10:8]	7	[6:0]
Width	1	2	1	1	3	1	7
Definition	Reserved	Word select	Block Read	Block Write	Timing Value	CRC Check On/Off	Response Size

Table 4. CMD_SERIAL_HOST Status Register

Bit	[15:7]	6	5	[3:0]
Width	9	1	1	4
Definition	Reserved	Data Available	CRC Valid	State

Figure 3 shows the state machine of this module.

Figure 3. CMD_SERIAL_HOST State Machine



SD_DATA_SERIAL_HOST

This module transfers data to the SD card. It provides a data path between the SD card and the master. The external interface consists of two signals, CLK and the bi-directional signal DAT. The DAT_OE_O, DAT_DAT_O and DAT_DAT_I signals must be combined into a bi-directional signal in an additional module.

The module performs the following actions:

- Synchronizes requests for write and read data
- Adds a CRC-16 checksum on sent data and checks the CRC-16 on received data

Signals and State Machine

In default status, this module is in the idle state. It waits for the start signal (START_DAT). The START_DAT signal is asserted at the beginning of reading data, writing data or stop operations. The '01' of the START_DAT signal means that the next operation is to write the data to SD card. The '10' of the START_DAT means that the next operation is to read the data from the SD card. If the START_DAT is '11', the module will stop the current operation.

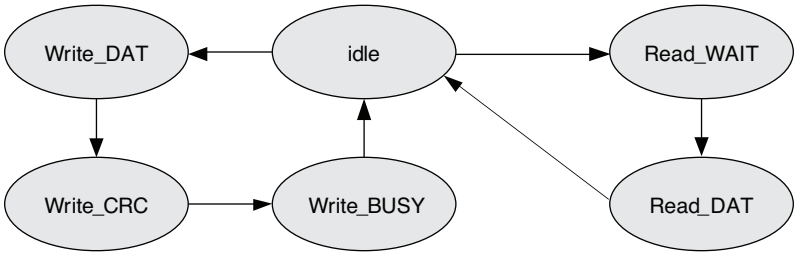
If the start signal is writing a data request signal, the state machine enters the WRITE_DAT state. In this state, the first task is to generate the data token and send the reading TX_FIFO request. Because the output data from TX_FIFO is 32 bits wide, it needs eight cycles to transfer one block of data. In this design, the block size is 512 bytes, so it reads one data from TX_FIFO every eight cycles until one block is transmitted completely. In the mean time, it calculates the 4-data-line CRC data. After transmitting the data block, it transmits the CRC checksum to the SD card. Once a block of data is transferred completely, this module generates one TRANSM_COMPLETE signal for the host. It then enters the turn-around state and the first 4-bit data line is at the low level. At the same time the SD card is in the programming state. If the SD card has finished programming the Flash, it releases the data line from '0' to '1'. The state machine will then check the ACK_IN signal and enter the idle state to wait for next operation if the ACK_IN signal is asserted.

If the start signal is reading a data request signal, the state machine enters the READ_WAIT state. In the READ_WAIT state, it checks the first data line. If the data line changes to the low level, it shifts to the READ_DAT state. In the READ_DAT state, it stores the received data into the Rx FIFO. In the mean time, it enables the CRC module to calculate the CRC result. After receiving one block of data, it begins to check the CRC checksum with

the received checksum and sends one TRANSM_COMPLETE signal to the host. If the checksum is correct, it sets the CRC_OK signal to '1'; otherwise, it sets the TRANS_FAILED signal to '1' for the host. Finally, it enters the idle state if the ACK_IN signal is received.

The module includes six states. Figure 4 shows the state machine of this module.

Figure 4. Finite State Machine Data Serial Host



CRC7 and CRC16

The CRC7 and CRC16 modules are used to calculate the input commands or data checksum. CRC7 has the following description:

$G(x) = X^7+X^3+1$.

For every command, CRC7 calculates the START_BIT, TRANSMITTING_BIT, command index and argument parameters.

CRC16 has the following description:

$G(x) = X^{16}+X^{12}+X^5+1$.

CRC16 calculates the payload data.

SD_CMD_MASTER

This module is a bridge between the WISHBONE bus and the SD_CMD_SERIAL_HOST module. It implements the following tasks:

- Reads a set of registers from the user-accessible register and sets up the corresponding command format.
- Reads response messages from the SD_CMD_SERIAL_HOST and forwards them to the user register for the master.
- Keeps track of the status of the SD_CMD_SERIAL_HOST and reflects the status for the master.

Signals and Operation

In default status, the state machine is in idle state. The user can configure the ARG_REG to generate the NEW_CMD signal. Once the NEW_CMD is asserted, a new command is available in the ARG_REG. The state machine enters the setup state immediately. In the setup state, it builds the command according to the command format defined in the SD Physical Layer Simplified Specification 2.0. It then enters the execute state. In this state, it waits for the complete signal from the SD_CMD_SERIAL_HOST module. If the complete signal is asserted, it enters the idle state. In setup state, it can generate the CMD_OUT signal and set out the signal for the SD_CMD_SERIAL_HOST module. In the execute state, it reads the input status from the SD_CMD_SERIAL_HOST. According to the status signal, it outputs the NORMAL_INT_REG to the user. The user can read this register to understand the status of the controller.

SD_DATA_MASTER

This module is used to connect the WISHBONE bus, the SD_CMD_SERIAL_HOST module and the SD_DATA_SERIAL_HOST module. It completes the following tasks:

- Generates the READ_SINGLE_BLOCK command (CMD17) and the WRITE_SINGLE_BLOCK command (CMD24) for the controller to process
- Checks the card status
- Checks the data transfer state and compares this to the data CRC checksum.
- Generates the stop (STOP_TRANSMISSION) command (CMD12) to the host if the data transfer fails

Signals and Operation

In the idle state, the state machine checks the TX_BD module or RX_BD module. If the TX_BD FIFO or RX_BD FIFO is not empty, it enters the GET_TX_BD state or GET_RX_BD state to read the argument parameter. In this module, the priority of the TX_BD is higher than RX_BD. In the GET_TX_BD state, it sends the write request to the TX_BD module and generates a CMD24 command for the SD_CMD_SERIAL host to process. It then enters the SEND_CMD state. In the GET_RX_BD state, it sends the read request to the RX_BD module and generates a CMD17 command for the SD_CMD_serial host to process. It then enters the SEND_CMD state.

In the SEND_CMD state, it transfers the WRITE_CMD request for the SD_CMD_SERIAL_HOST to process. Once it receives the ACK signal from the SD_CMD_SERIAL_HOST, it moves to the receive CMD state.

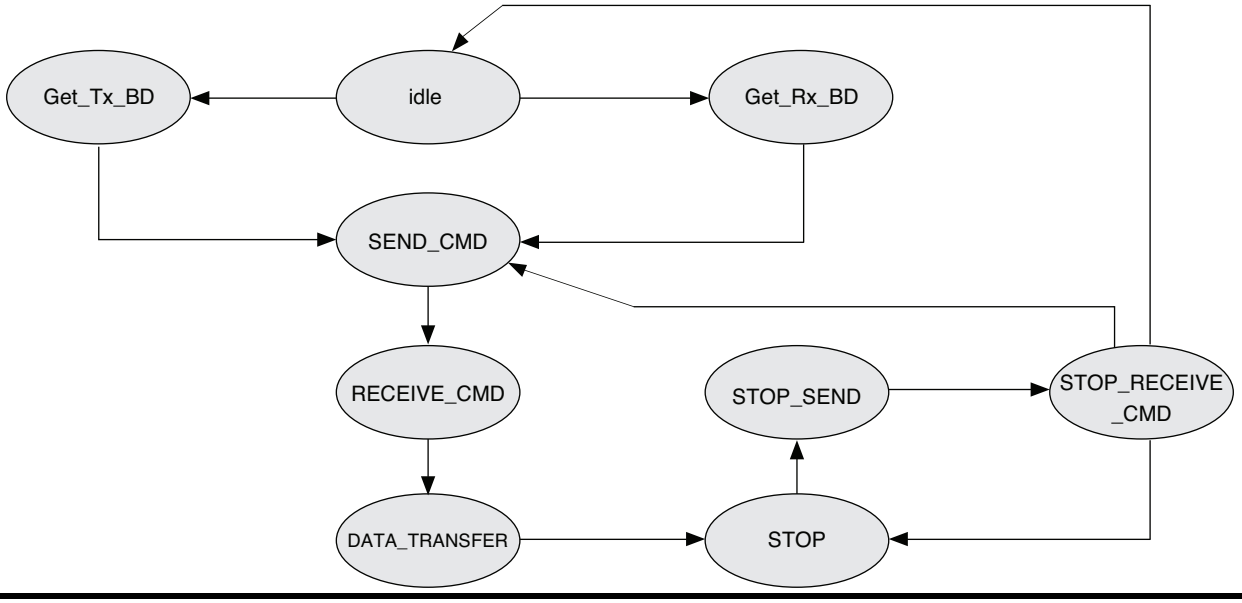
In the receive CMD state, it checks the CARD_STATUS after the CMD has been sent. If the CARD_STATUS indicates the former command is correct, it enters the data transfer state. The card status is defined in the SD Physical Layer Simplified Specification 2.0. If the card status indicates that the card is in an abnormal state, it will re-send the previous command to the SD card.

In the data transfer state, it waits for the TRANSM_COMPLETE signal until the SD_DATA_SERIAL_HOST has finished the reading or writing task. If the transfer fails, a one-stop command for the SD_CMD_SERIAL_HOST module is generated. It then enters the idle state.

If the previous data transfer fails, a STOP_TRANSMISSION command will be sent to the SD card. This command is CMD12.

Figure 5 shows the state machine of this module.

Figure 5. Finite State Machine Data Master



FIFO_RX_FILLER.v

This module implements a DMA function using the WISHBONE bus and controls the RX_FIFO for the data stream. The module contains the submodule SD_RX_FIFO.v. It stores the data from the SD card and outputs it to the external memory.

Functional Description

The FIFO_RX_FILLER.v module contains the master WISHBONE interface signals. The signal En is used to start a read operation from the master. Once the En signal is valid, it checks the RX_FIFO empty signal and the WISHBONE bus. If the Rx FIFO is not empty and the WISHBONE is free, it generates a read-enable signal to the RX_FIFO. Once the read-enable signal is valid, it outputs the Rx FIFO data to the external memory.

Writing RX_FIFO is driven by the SD_DATA_SERIAL_HOST module. As long as the controller has the data required to send from the SD card, it writes these data into the Rx FIFO.

FIFO_TX_FILLER.v

The FIFO_TX_FILLER.v module implements a DMA function using a WISHBONE bus and controls the TX_FIFO for the data stream. The module contains the submodule SD_TX_FIFO.v. It stores the data from the master and outputs it to the SD card.

Signals and Operation

The FIFO_TX_FILLER.v module contains the master WISHBONE interface signals. The signal En is from the SD_DATA_MATER module. Once the host sends the write-data command to the SD card, the En signal is valid until one block of data is transferred completely.

If the TX FIFO is not full, the module sends the read request to the external memory using the WISHBONE bus. The external memory outputs data directly into the Tx FIFO. The read operation to the TX_FIFO is driven by the SD_DATA_SERIAL_HOST module.

SD_RX_FIFO.v / SD_TX_FIFO.v

The FIFO in this module is a simple FIFO with a read port, write port and logic to signal full and empty states. It is designed to be implemented as a register and not as a RAM block, so the size of the FIFO is small.

SD_BD.v

The transmission and reception processes are based on the descriptors. Two sequential writings to this module are required to create one buffer descriptor. First, the source address (memory location) of the data is written, then the card block address is written. Depending on the specified RAM width (16 or 32 bits), two or four writings are required to forge a complete buffer descriptor.

Signals and Operation

The signal WE_M is asserted when the master side is writing the DAT_IN_M data to the buffer descriptor RAM. The master side can also read a buffer descriptor by asserting RE_M. Data is then available on the DAT_OUT_M port. When the read side needs to read a buffer descriptor, RE_S is asserted and data will be available on DAT_OUT_S when ACK_O_S is set.

SD_CONTROLLER_TOP.v

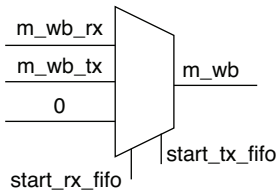
The host interface connects the reference design to the rest of the system (RISC, memory) via the WISHBONE bus. The configuration registers and the external memory can be accessed through the WISHBONE bus. At present, only DMA transfers are supported for transferring the data from/to the external memory.

Signals and Operation

All the WISHBONE signals, SD card interface and interrupt signals are included in the SD_CONTROLLER_TOP.v module and all the major submodules are instantiated. The connections between different modules are given in Figure 2.

Some muxes in the module are used by other modules. This is the case with the Tx and Rx FIFO filler where both need to drive the WISHBONE master signals cyc, stb, we and ADR_O. The mux uses the START_RS_FIFO and START_TX_FIFO from the data master module to describe which modules needs access to the bus. If neither needs access, it is set to 0.

Figure 6. Master WISHBONE Mux



SD_CONTROLLER_WB.v

This module describes the WISHBONE slave interface used to control the operations for reading from and writing to all user registers.

Signals and Operation

This module uses a slave WISHBONE interface for the register writing and reading. When the WISHBONE is accessed, a WB_ACK_O is generated. When an access to the Tx or Rx registers is performed the WE_M_TX_BD or WE_M_RX_BD is asserted and the WB_DAT_I is clocked in. Depending on the width of the block RAM this procedure takes one or two cycles. The WB_ACK_O is not generated until the entire buffer descriptor has been written.

The signal CMD_INT_BUSY is set whenever the DATA_MASTER is doing a CMD_ARG or CMD_SET accession. INT_BUSY is set whenever the WISHBONE accesses the CMD_ARG register.

Operation

This design can be customized by the user through the setting of parameters. The parameters are described in Table 5.

Table 5. SD Flash Controller Design Parameters

Parameter	Valid Value	Description
BIG_ENDIAN		Big Endian system
LITLE_ENDIAN		Little Endian system
SIM		To ease up simulation
SYN		For synthesizing
IRQ_ENABLE		Three extra interrupt pins will be added
BD_WIDTH	<= 8	2nlog2 BD-SIZE
BD_SIZE	<= 255	Size of the BD buffer
RAM_MEM_WIDTH_16		Width of block RAM = 16
RAM_MEM_WIDTH_16	16, 32	Width of block RAM
RESEND_MAX_CNT	<255	Number of retries to send data
MEM_OFFSET		Memory address offset between two words
SD_CLK_BUS_CLK		Use the same clock as the WISHBONE Bus
SD_CLK_SEP		Use SD_CLK_I_PAD as SD CLK
SD_CLK_STATIC		SD CLK = IN clock
BLOCK_SIZE	512	Block size
SD_BUS_WIDTH_4		Supports 4 bits
SD_BUS_W	4	Supports 4 bits
FIFO_RX_MEM_DEPTH		Width of Rx FIFO
FIFO_RX_MEM_ADR_SIZE		FIFO_RX_MEM_DEPTH 2nlog2 +1
FIFO_TX_MEM_DEPTH		Width of Tx FIFO
FIFO_TX_MEM_ADR_SIZE		FIFO_TX_MEM_DEPTH 2nlog2 +1

The SD Flash Controller uses the WISHBONE bus as the process bus. User registers must be configured in order to start every operation. The WISHBONE bus serves to access the configuration registers and the memory. The slave WISHBONE bus is used to control all user registers. The master WISHBONE bus is used to access the external memory.

- **Configuration Registers** – The function of the configuration registers is transparent. Before every operation, the user can configure the following three registers to set up the SD Flash Controller:
  - Reset software register
  - Set the timeout register
  - Set the software register
- **Sending a Command** – The sending of a command to the SDC/MMC card is performed in two steps:
  - Configure the command index setting register with command index and transmission settings.
  - Configure the argument registers with the argument bits of the command to initiate the transfer. The Normal Interrupt register gives the status. The Error Interrupt Status register makes any failure visible to the host.
- **Data Block Transmission and Reception** – To transmit or receive a block of data, the CPU must perform several steps:
  - Confirm that the card is initiated correctly and that it is ready for data with a block size of 512 bytes and all four data bits are enabled.
  - Configure the TX_BD register or RX_BD register to generate the WRITE_BLOCK command (CMD24) and the READ_SINGLE_BLOCK command (CMD17).

– Wait for the correct response from the SD card and check the SD card status.
– Once the data transfer command is valid, data transmission or reception begins.

Registers

From the host's perspective, the SD Flash Controller is a set of accessible registers. Proper setting of the writable registers allows successful initialization and command/data transfers. The readable registers provide information to monitor the status and quality of the transfer.

Table 6. Registers

Name	Address	Width	Access	Description
Argument	0x00	32	R/W	Command Argument Register
Command Setting	0x04	16	R/W	Command Setting Register
Card Status	0x08	16	R	Card Status Register
Response	0x0c	32	R	Command Response
Controller Setting	0x1c	16	R	Controller Setting
Block Size	0x20	16	R	Block Size Register
Power Control	0x24	8	R	Power Control Register
Software reset	0x28	8	R/W	Software Reset Register
Timeout	0x2c	16	R/W	Timeout Register
Normal Int Status	0x30	16	R/W	Normal Interrupt Status Register
Error Int Status	0x34	16	R/W	Error Interrupt Status Register
Normal Int Enable	0x38	16	R/W	Normal Interrupt Enable
Error Int Enable	0x3c	16	R/W	Error Interrupt Enable Register
Capability	0x48	16	R	Capability Register
Clock Divider	0x4c	8	R/W	Clock Divider Register
BD buffer Status	0x50	16	R/W	BD Status Register
Dat Int Status	0x54	16	R/W	Data Interrupt Status Register
Dat Int Enable	0x58	16	R/W	Data Interrupt Enable Register
BD RX	0x60	64	W	BD RX
BD TX	0x80	64	W	BD TX

Argument Register

Table 7. Argument Register

Bit #	Access	Description
[31:0]	W	CMDA – Command Argument Command data, when writing to this register, transmission begins.

Reset value: 0000000h.

Command Setting Register

Table 8. Command Setting Register

Bit #	Access	Description
[15:14]		Reserved
[13:8]	R/W	CMDI – Command index of the next command
[7:6]	R/W	CMDW – Command Word Selects word to be read when the size of the response is > 48 bits.
5		Reserved
4	R/W	CICE – Command index check 0: Do not perform an index check on the response CMD 1: Perform an index check on the response CMD
3	R/W	CIRC – Command CRC check 0: Do not perform a CRC check on the response CMD 1: Perform a CRC check on the response CMD
2		Reserved
[1:0]	R/W	RTS – Response type 0: No response 01: Response length 136 10: Response length 48 11: Response length 48

Reset Value: 0000h.

Status Register

Table 9. Status Register

Bit #	Access	Description
[15:12]	R	CST – CMD Host Serial Status 0: Reset 1: Write only state 2: Write to read state 3: Delay after write to read 4: Delay after write only state 5: Read CMD 6: Delay after read
[11:1]		Reserved
0	R	CICMD – Command Inhibit 1: Busy 0: Ready

Reset value: 0000h.

Response Register

Table 10. Status Register

Bit #	Access	Description
[31:0]	R	CRSP – Command Response Response of last command

Reset value: 00000000h.

Controller Settings

Not in use.

Block Size

Table 11. Block Size

Bit #	Access	Description
[15:12]		Reserved
[11:0]	R	BS – Block Size Hard coded to 512. The value has no effect on the operation.

Reset value: 0200h.

Power Control

Table 12. Power Control

Bit #	Access	Description
[7:3]		Reserved
[3:0]	R	SDBP – SD Bus Power Voltage provided to the bus 111: 3.3 V 110: 3.0V 101: 1.8V

Reset value: 0007h.

Software Reset

Table 13. Software Reset

Bit #	Access	Description
[7:1]		Reserved
0	R	SRST – Software Reset 0: 1: Reset the hardware

Reset value: 0000h.

Timeout Register

Table 14. Timeout Register

Bit #	Access	Description
[15:0]	RW	CTO – Command Timeout. Time before a timeout signal is generated when sending, counted by the system clock.

Reset value: 00000000h.

Normal Interrupt Status Register

Table 15. Normal Interrupt Status Register

Bit #	Access	Description
15	RW	EI – Error Interrupt If any of the bits in the Error Interrupt Status register are set, then this bit is 1.
[14:1]		Reserved
0	RW	CC – Command Complete This bit is set when the end bit of the command response is received.

Reset value: 0000h.

Error Interrupt Status Register

Table 16. Error Interrupt Status Register Control Settings

Bit #	Access	Description
[15:4]		Reserved
3	RW	CIE – Command Index Error. This bit is set if a command index error occurs.
2		Reserved
1	RW	Command CRC error has occured
0	RW	Command time out indicated

Reset value: 0000h.

Normal Interrupt Enable Register

Table 17. Normal Interrupt Enable Register Control Settings

Bit #	Access	Description
15	RW	EEI – Enable Error Interrupt 1: Enable interrupt generation on EI 0: Disable interrupt generation on EI
[14:1]		Reserved
0	RW	ECC – Enable Command Complete 1: Enable interrupt generation on ECC 0: Disable interrupt generation on ECC

Reset value: 0000h.

Error Interrupt Enable Register

Table 18. Error Interrupt Enable Register

Bit #	Access	Description
[15:4]		Reserved
3	RW	ECIE – Command Index Error 1: Enable interrupt generation on CIE 0: Disable interrupt generation on CIE
2		Reserved
1	RW	ECCRC – Command CRC Error 1: Enable interrupt generation on CCRC 0: Disable interrupt generation on CCRC
0	RW	ECTE – Command Timeout 1: Enable interrupt generation on CTE 0: Disable interrupt generation on CTE

Reset value: 0000h.

Capability Register

Not in use.

Data Interrupt Status Register

Table 19. Data Interrupt Status Register

Bit #	Access	Description
[7:6]		Reserved
5	RW	TRE – Transmission Error 1: CRC check failed during transmission
4	RW	CMDE – Command Error 1: Error in the command response
2	RW	FIFOE – FIFO Error 1: FIFO underflow/overflow
1	RW	MRC – Maximum Retry Attempts 1: Unable to send after N attempts
0	RW	TRS – Transmission Successful 1: One data block has been sent/received

Reset value: 0000h.

Data Interrupt Enable Register

Table 20. Data Interrupt Enable Register

Bit #	Access	Description
[7:6]		Reserved
5	RW	ETRE – Transmission Error 1: Enable interrupt generation on TRE 0: Disable interrupt generation on TRE
4	RW	ECMDE – Command Error 1: Enable interrupt generation on CMDE 0: Disable interrupt generation on CMDE
2	RW	EFIFOE – FIFO Error 1: Enable interrupt generation on FIFOE 0: Disable interrupt generation on FIFOE
1	RW	EMRC – Maximum Retry Attempts 1: Enable interrupt generation on MRC 0: Disable interrupt generation on MRC
0	RW	ETRS – Transmission Successful 1: Enable interrupt generation on TRS 0: Disable interrupt generation on TRS

Reset value: 0000h.

BD_RX

Table 21. BD_RX

Bit #	Access	Description
[63:32]	W	Memory location were data is stored.
[31:0]	W	Block address to read from.

Reset value: 0000h.

BD_TX

Table 22. BD_TX

Bit #	Access	Description
[63:32]	W	Memory location were data is stored.
[31:0]	W	Block address to read from.

Reset value: 0000h.

Test Bench Description

The test bench for this design includes the following modules:

- SD_CONTROLLER_TOP_TB.v
- SD_MODEL.v
- WB_BUS_MON.v
- WB_MASTER32.v
- WB_MASTER_BEHAVIORAL.v
- WB_SLAVE_BEHAVIORAL.v
- WB_MODEL_DEFINES.v

SD_CONTROLLER_TOP_TB.v

This module is used to generate test vectors. It includes six test task items as described below.

- **IRQ_TEST_SEND_CMD**
  This task tests the register reading, writing, software reset and sending commands to the SD card.
- **TEST_INIT_SEQUENCE**
  This task tests the init procedure.
- **TEST_SEND_DATA**
  This task tests the writing of data to the SD card. The test command is CMD24.
- **TEST_SEND_REC_DATA**
  This task tests the writing of data and the sending of the reading commands to the SD card. The test commands are CMD17 and CMD24.
- **TEST_SEND_CMD_ERROR_RSP**
  This task tests the sending of an error command to the SD card.
- **TEST_SEND_REC_DATA_ERROR_RSP**
  This task reports an error if an incorrect data command is sent.

This module also generates the reset and process clock signals. If the test fails, it records the error information in the following files:

- ETH_TB_HOST.log
- ETH_TB_PHY.log
- ETH_TB_WB_M_MON.log
- ETH_TB_WB_S_MON.log
- SD_MODEL.log
- SD_TB_MEMORY.log
- SDC_TB.log

SD_MODEL.v

This module is a simple simulation model of the SD card. It receives the command and data from the host and responds to the corresponding data and response for the master. It supports the following commands: CMD0, CMD2, CMD3, CMD7, CMD8, CMD14, CMD16, CMD17, CMD24, CMD33, ACMD41, and CMD55.

WB_BUS_MON.v

This module connects to the WISHBONE master signals and monitors for any illegal combinations appearing on the bus.

WB_MASTER_BEHAVIORAL.v and WB_MASTER32.v

These modules make up the master model of the WISHBONE bus. They simulate a microprocessor and the sending command for the controller.

WB_SLAVE_BEHAVIORAL.v

This module is a simple slave WISHBONE bus model. It models an external SSRAM. The controller can write the received data to this module.

Timing Specifications

This section shows the simulation results of four different operations of the SD Flash Controller. The four operations include sending a command, an initialization sequence, sending data to the SD card, and receiving data from the SD card.

Figure 7. TEST_SEND_CMD Waveform

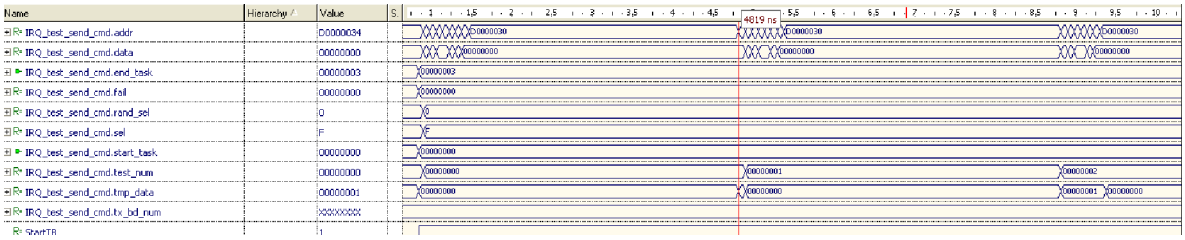


Figure 8. TEST_INIT_SEQUENCE Waveform

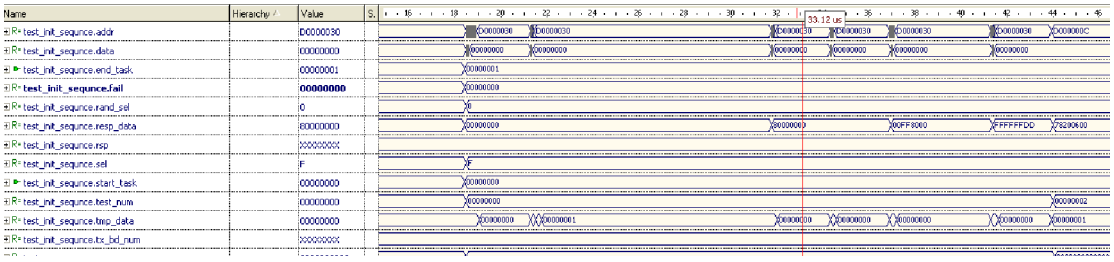


Figure 9. TEST_SEND_DATA Waveform

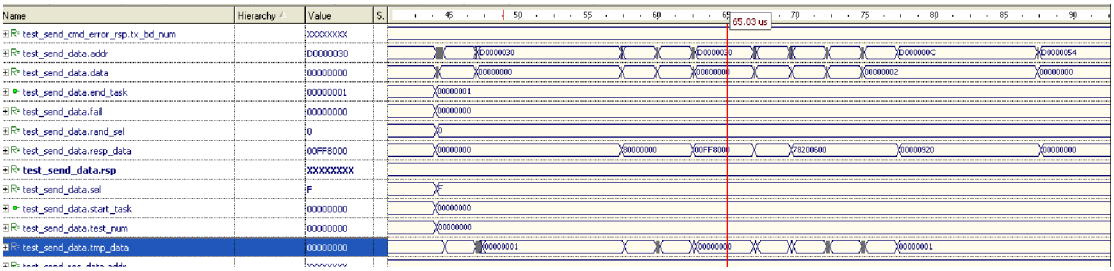
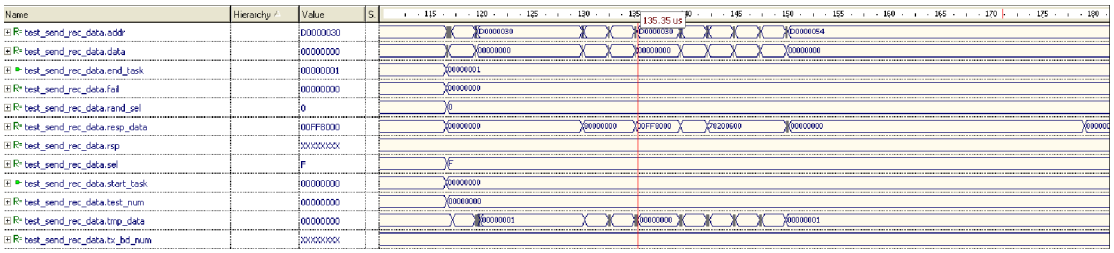


Figure 10. TEST_SEND_REC_DATA Waveform



Implementation

This design is implemented in Verilog. When using this design in a different device, density, speed, or grade, performance and utilization may vary. Default settings are used during the fitting of the design.

Table 23. Performance and Resource Utilization

Device Family	Language	Speed Grade	Utilization (LUTs)	f_{MAX} (MHz)	I/Os	Architecture Resources
LatticeECP3™ [1]	Verilog	-7	1520	>75	203	N/A
MachXO2™ [2]	Verilog	-6	1528	>50	203	N/A
MachXO™ [3]	Verilog	-4	1562	>50	203	N/A

1. Performance and utilization characteristics are generated using LFE3-95EA-7FN1156C with Lattice Diamond 1.2 design software.
2. Performance and utilization characteristics are generated using LCMXO2-7000HC-6BG332C with Lattice Diamond 1.2 design software.
3. Performance and utilization characteristics are generated using LCMXO2280C-4FT324C with Lattice Diamond 1.2 design software.

References

- SD Physical Layer Simplified Specification 2.0
- WISHBONE System-on-Chip Interconnection Architecture for Portable IP Cores
- OpenCores SD Controller web page
  www.opencores.org/project,sdcard_mass_storage_controller

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
April 2010	01.0	Initial release.
November 2010	01.1	Added support for MachXO2 device family and Lattice Diamond design software.
April 2011	01.2	Added support for LatticeECP3 device family and Lattice Diamond 1.2 design software.