

## Overview

MultiMediaCard (MMC) and Secure Digital (SD) Card are flash based memories, designed to meet security, capacity and performance requirements of various audio and video consumer electronic devices like MP3 players, cell phones, PDAs, digital still and video cameras, data loggers, and more. This design example describes the implementation of SDIO Controller using iCE65 FPGAs, which allows you to easily connect a processor to SDIO cards using a generic processor interface.

## Features

- Meets SD standard spec version 2.0
- Auto Card detect and SDIO card detection.
- Forced initialization from processor.
- Operates under SD 1-bit, SD 4-bit and SPI mode.
- Operates at the normal mode of 25 MHz and also at an optional high speed mode of 50MHz.
- Features a 32 bit processor bus interface so that the processor isn't tied to the controller for the low speed transfers on the SDIO physical layer.
- Features an 8K Read/Write buffer for processor initiated writes and reads.
- Single and multi byte read/write for SDIO cards.
- Single and multi block read/write for combo cards.
- In case of an SDIO transaction, the controller supports single byte Read/Write access via CMD52 and a multi-byte access mode at a configurable byte count of 2 bytes to 512 bytes via CMD53.
- CRC7 and CRC16 support for CMD and DATA.
- Support SDIO card interrupts.
- IP-XACT 1.2 compliant

## Resource Utilization

*Table 1: Resource Utilization*

| LUTs | Registers | Memory | I/Os | GBs |
|------|-----------|--------|------|-----|
| 3164 | 1545      | 17     | 0    | 0   |

Note: Resource Utilization is based on iCECube2 2010.12.14671

## System Block Diagram

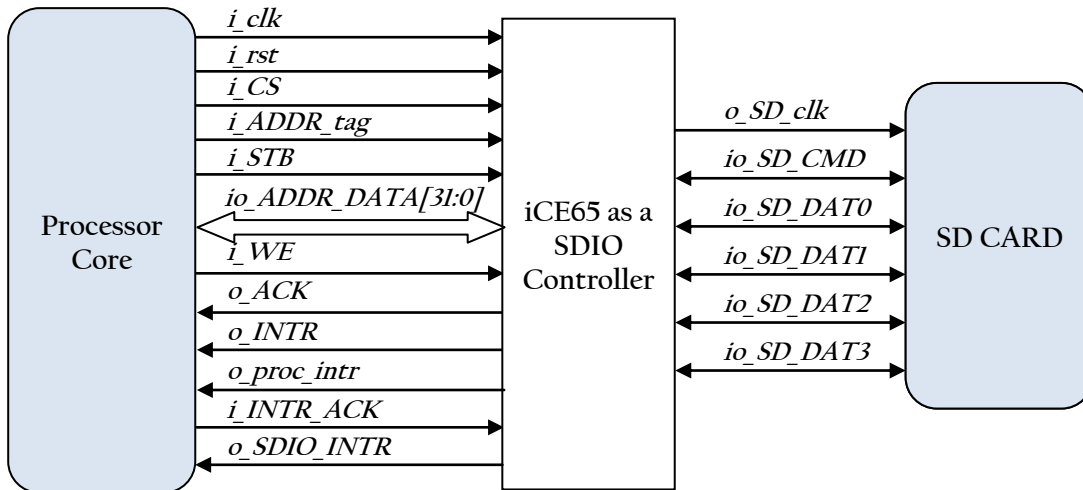


Figure 1: System Block Diagram

## Functional Block Diagram

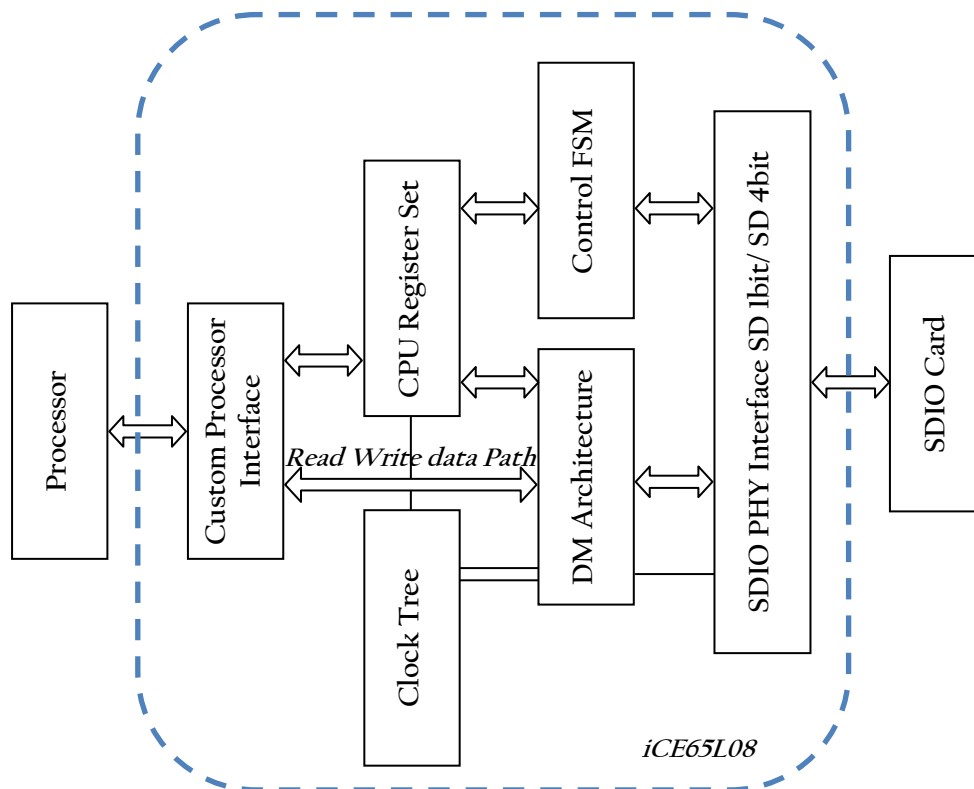


Figure 2: Functional Block Diagram

## Design Interface

*Table 2: Pin Description*

| Signal Name               | Pin Type | Signal Description  |
|---------------------------|----------|---|
| <b>i_clk</b>              | Input    | System clock .  |
| <b>i_rst</b>              | Input    | Asynchronous active high system reset.  |
| <b>i_CS</b>               | Input    | Active low chip select signal from the processor interface which shall be asserted for all transactions on the processor interface. |
| <b>i_ADDR_tag</b>         | Input    | Active high Address latch enable signal indicating that the value on the io_ADDR_DATA bus is an address for the transaction.        |
| <b>i_STB</b>              | Input    | Active low strobe input which indicates that a valid transaction is on the processor interface bus.                                 |
| <b>io_ADDR_DATA[31:0]</b> | Output   | Multiplexed 32-bit address/data bus which can be used to transfer data and address for processor initiated transactions.            |
| <b>i_WE</b>               | Output   | Active high acknowledge indicating that the host Controller has accepted the transaction.   |
| <b>o_ACK</b>              | Output   | Active high acknowledge indicating that the host Controller has accepted the transaction.   |
| <b>o_INTR</b>             | Output   | Active high interrupt signal that may be asserted for conveying error/information to the processor.                                 |
| <b>i_INTR_ACK</b>         | Input    | Active high interrupt acknowledge to de-assert the interrupt that was raised by the Host Controller.                                |
| <b>i_SEL [3:0]</b>        | Input    | Byte enables select for indicating the bytes that are enabled in the io_ADDR_DATA bus.  |
| <b>o_SDIO_INTR</b>        | Output   | Active high interrupt signal, asserted due to interrupt from SDIO card.   |
| <b>o_SD_clk</b>           | Output   | SDIO card clock input. This line is at 400KHz during initialization, and afterwards at data transfer frequency (<25 or <50 MHz).    |
| <b>io_SD_CMD</b>          | InOut    | The command/response bus of the SDIO bus interface.   |
| <b>io_SD_DAT0</b>         | InOut    | The data line 0 of the SDIO bus interface.  |
| <b>io_SD_DAT1</b>         | InOut    | The data line 1 of the SDIO bus interface.  |
| <b>io_SD_DAT2</b>         | InOut    | The data line 2 of the SDIO bus interface.  |
| <b>io_SD_DAT3</b>         | InOut    | The data line 3 of the SDIO bus interface.  |
| <b>i_Card_detect</b>      | Input    | The card detect pin of the SDIO bus interface.  |
| <b>o_proc_intr</b>        | Output   | Interrupt after completion of operations  |

## Configurable Parameters

None

## Register Map

The SDIO Host Controller Configuration can be performed on run-time as well as during compile time depending on the designs requirements and also resource consumption. Table 3 lists the SDIO HCI registers and their address mapping.

*Table 3: SDIO Host Control registers*

| Register Name               | Address | Width | Mode | Register Description  |
|-----------------------------|---------|-------|------|---|
| <b>Block Size Register</b>  | 0x04    | 16    | RW   | The register can be used to indicate the Block size of transfer.                                  |
| <b>Block Count Register</b> | 0x06    | 16    | RW   | The Register is used for configuring the number of Blocks that are to be transferred to/ from the |

|  |      |     |    | processor from/to the SDIO Card  |
|--|------|-----|----|--|
| <b>SDIO Configuration Register</b>                 | 0x07 | 8   | NA | The register is used to configure SDIO data transactions to the SD card                          |
| <b>SiliconBlue Specific Configuration Register</b> | 0x64 | 8   | RW | The register is used for configuring the SDIO Host Controller operations                         |
| <b>SiliconBlue Specific Status Register</b>        | 0x80 | 8   | RW | The register is used for identifying the Status of the SDIO Controller Initialization modules.   |
| <b>Explicit Command Register</b>                   | 0x0E | 16  | RW | The register is used to configure any commands that the processor wants to execute from its side |
| <b>SD Argument Register (LS Word)</b>              | 0x08 | 16  | RW | The Argument 0 of the SDIO command shall be programmed in this register                          |
| <b>SD Argument Register (MS Word)</b>              | 0x0A | 16  | RW | The Argument 1 of the SDIO command shall be programmed in this register.                         |
| <b>Response Register</b>                           | 0x10 | 128 | RO | The Response register can be read by the processor once the command is posted.                   |

The Table 4 lists the bit mapping of the CPU registers provided in the SDIO Controller. Here are the details on various bits of these registers.

SD Argument Register (LS Word), SD Argument Register (MS Word), and Response Register are 16, 16 and 128-bit registers respectively.

- Block Count Registers(8-bits)

- Block Count: Allowed Values are 0x00–0x0F. A maximum of 16 blocks can be transmitted.

- Block Size Register (16 bits)

- Block Size: The register is a 16 bit register and the value in this register indicates the number of bytes that accessed to/from the SDIO card. A 0x0000 value in the register would imply an access of 512 bytes to/from the card. A 0x0200 implies an access of 1024 bytes per access on the SD interface provided the SDIO Access bit in the SDIO Configuration register is a 0. Any value between 0x0200 and 0x0000 should not be programmed during an SD operation on an SD card.

For an SDIO access, the Host Controller can be programmed for multi-byte access in the range 2bytes to 512 bytes using this register. A 0x0000 value in this register implies a 512 byte access to the SDIO register space provided the SDIO Access bit in the SDIO Configuration register is set to a 1. A value of 0x0010 would provide a 15 byte access to the SDIO register space under the same configuration.

- SiliconBlue Specific Configuration Register (16 bits)

- Bus Width : A 00 indicates SDIO 1 bit mode, 01 indicates SDIO 4 bit mode, 11 indicates SPI mode.
  - Reset\_init\_stats : A 1 written to this bit shall reset all internal flags of the Host Controller and processor should initiate an initialization all over again.
  - Force\_init: This bit shall be made to 1 for forcible initialization of the SDIO card.
  - Speed Class: This bit shall be programmed to 1 for High Speed mode of operation for both SD and 0 for a normal mode of operation.

- SiliconBlue Specific Status Register (16 bits)

- Init\_done: This bit indicates an initialization done indication to the processor
  - Err\_in\_init: This bit indicates that the initialization was unsuccessful
  - Unusable Card: This bit shall indicate a voltage mismatch or continuous ERC error observed during initialization and indicate it as an unusable card
  - SD Standard Legacy Card : Indicates that the card is SD and a Legacy card

- SD Normal Capacity Card : Indicates that the card is SD and normal capacity (upto 2Gb)
- SD High Capacity Card : Indicates that the card is SD and High Capacity(over 2Gb)
- High Voltage MMC Card : When 1 indicates that the card is a High voltage MMC card else it represents Dual voltage MMC card when all SD status are 0 and init done is 1
- MMC Version 4.lx : Indicates that the MMC is of version4.lx
- SDIO Flag: Indicates that there is an SDIO section in the card that is inserted
- Memory Flag: Indicates that there is a memory section in the card that is inserted. This flag can be used with the SDIO Flag bit to check if the card inserted is a Combo Card.
- Explicit Command Register (16 bits)
  - SDMMC\_Command: The command that has to be posted. The SD specification shall be referred for command op-codes
  - Command\_Request: The bit shall indicate the validity of the command value in the Command Index section of the register

The register set information for the design of the Software driver is shown in Table 4

**Table 4: SDIO Host Controller Register set breakup**

| Sl. No. | Register                    | Register Address | Register Size (bits) | Bit Range (Access) | Register Description  |
|---------|-----------------------------|------------------|----------------------|--------------------|---|
| 1       | Block Size Register         | 0x04             | 16                   | [7:0] RW           | Block size register MS byte (address=0x04)  |
|         |                             |                  |                      | [15:0] RW          | Block size register LS byte (address=0x05)  |
| 2       | Block Count Register        | 0x06             | 8                    | [7:4] RO           | Reserved for future use   |
|         |                             |                  |                      | [3:0] RW           | Block count which indicates the number of blocks of data (each of size indicated by the Block Size register) for a read/write transaction on the SD interface. This is only applicable in case of SD accesses. For SDIO accesses the value is treated as “don’t care” |
| 3       | SDIO Configuration Register | 0x07             | 8                    | [7:4] RO           | Reserved for future use   |
|         |                             |                  |                      | [3] NA             | SDIO access flag. This bit shall be set to one before the processor posts a read/write access to the SDIO register space. If this bit is not set, then the processor transaction is interpreted to be targeted to the SD/MMC space of the card                        |
|         |                             |                  |                      | [2:0] NA           | SDIO function number: The function number to which the processor wants to issue the access to. This field is used in the argument for the SDIO commands CMD53/CMD52.  |
| 4       | SD Argument Register        | 0x08             | 32                   | [7:0] RW           | LS byte of LS word of the argument register(address 0x08)   |
|         |                             |                  |                      | [15:8] RW          | MS byte of LS word of the argument register(address 0x09)   |
|         |                             |                  |                      | [23:16] RW         | LS byte of MS word of the argument register(address 0xA)  |
|         |                             |                  |                      | [31:24] RW         | MS byte of MS word of the argument register(address 0xB)  |
|         |                             |                  |                      | NA                 | The argument register can be used to send a processor intended command to the SD card   |

|          |   |        |   |            |  |  |
|----------|---|--------|---|------------|--|--|
|          |   |        |   |            | without the interference of the controller. The processor can use the command register for sending an explicit command. The register is byte as well as word accessible  |  |
| <b>5</b> | SiliconBlue Specific Configuration Register | 0x64   | 16  | [15:6] RO  | Reserved for future use  |  |
|          |   |        |   | [5] WO     | Auto Detect, Asserted if Auto Detect is required   |  |
|          |   |        |   | [4] WO     | Speed class, 0 for Standard Mode, 1 for High Speed Mode  |  |
|          |   |        |   | [3:2] RW   | Bus width configuration on the PHY interface A 00 indicates a 1 bit mode, 01 indicates a 4 bit mode, a 11 indicates SPI mode of operation  |  |
|          |   |        |   | [1] RW     | Reset initialization Statistics: The bit is used as a soft reset for resetting any initialization flags that are set by the host controller when it was used to initialize any other card. The processor is expected to set this bit at least once before a second initialization is forced on the Host controller initialization state machines |  |
|          |   |        |   | [0] RW     | Force initialization: This bit is used to control the initialization flow of the Host controller. Only when the software driver sets this bit, the host controller starts the initialization process.  |  |
| <b>6</b> | SiliconBlue Specific Status Register        | 0x80   | 16  | [15:11] RO | Reserved for future use  |  |
|          |   |        |   |            | [10] NA  | Memory Flag: When set indicates that the card inserted has a memory section. (The bit is of importance in case of SDIO operation to make sure if the card is a combo card)   |
|          |   |        |   |            | [9] NA   | SDIO Flag: When set indicates that the card inserted has an SDIO section   |
|          |   |        |   |            | [8] NA   | MMC Card version above 4.1x. Indicates that the card inserted is above MMC version 4.1x  |
|          |   |        |   |            | [7] RO   | Unusable card: This is an error bit and indicates that the card is unusable for various reasons such as repeated errors during etc   |
|          |   |        |   |            | [6] RO   | Card Capacity: When set indicates that the card is a high capacity card.   |
|          |   |        |   |            | [5] NA   | MMC Dual Voltage supported: When Set indicates that the card supports dual voltage.  |
|          |   |        |   |            | [4] RO   | SD Legacy Card: When set indicates that the card is a legacy SD card   |
|          |   |        |   |            | [3] NA   | MMC Card: Indicates that the card inserted is an MMC Card  |
|          |   |        |   |            | [2] RO   | CMD Sent: This bit is an acknowledge bit indicating that the explicit command is sent by the Host Controller and the received response is updated in the Response register. The processor may have to poll this bit at regular intervals after posting a request for an explicit command to enquire the status of the request. |
|          |   | [1] RO | Error in Initialization: Indicates there was some |            |  |  |

|          |                           |      |     |            |  |
|----------|---------------------------|------|-----|------------|--|
|          |                           |      |     |            | error during initialization of the card.   |
|          |                           |      |     | [0] RO     | Initialization Done: This bit when set indicates that the initialization process was successful and all the status register bits are valid for the software driver to interpret  |
| <b>7</b> | Response Register         | 0x10 | 128 | [127:0] RO | This register is the response register for an explicit command posted by the processor.  |
| <b>8</b> | Explicit Command Register | 0x0E | 16  | [15:8] RO  | Reserved for future use  |
|          |                           |      |     | [7] RW     | Command Request: This bit when set indicates that the processor is posting a explicit command whose index is indicated by the Command index section of the Explicit Command Register. The processor must set this bit only after setting all the command argument and command index sections of the register set. As soon as the request bit is set the Host controller frames a command and sends it to the Card. The response is updated in to the Response Register |
|          |                           |      |     | [6] RO     | Reserved for future use  |
|          |                           |      |     | [5:0] RW   | Command Index Register: This section is the command index of the explicit command that the processor intends to post to the card. The section is evaluated only when the Command Request bit of the Explicit Command register is set.  |

Note:

RW : Read and Write

RO: Read Only

NA: Not Applicable

Run-time/Compile time Configurations details for the hardware IP are as shown below:

1. Initialization : The Initialization can be forced by the processor by writing a value 1 to force\_init bit of the SiliconBlue Specific Configuration Register (Address = 0x64). The package file shall have CNTRL\_FSM\_CARD\_DETECT\_SUPPORTED parameter set to 0. For an auto initialization feature, the CNTRL\_FSM\_CARD\_DETECT\_SUPPORTED parameter shall be set to a 1.
2. SiliconBlue Specific Status Register : This is the SDMMC Controller Status register(Address=0x80) which shall indicate a Initialization done on its init\_done bit. The processor shall read this register to check if initialization is done. The Register also has the err\_int\_init bit indicating that the initialization was unsuccessful.
3. Bus width for SD operation : The processor can run-time configure the bus width for the SDIO operation by writing the Bus Width section bit of the SiliconBlue Specific Configuration Register (Address = 0x64). A value of "00" indicates a 1 bit mode of operation, a value of "01" indicates a 4-bit mode of operation and a value of "11" shall indicate a SPI mode of operation. The Controller automatically detects if the card is a SD card. The registers shall be programmed before performing the forced initialization. For compile time configuration of the bus width only the CNTRL\_FSM\_BUS\_PKG\_CONFIGURABLE parameter in the package shall be set to a 1 and CNTRL\_FSM\_BUS\_WIDTH\_SUPPORTED shall be set to corresponding values that point to 1 bit, 4 bit, SPI mode of operation.

4. High speed operation : The SDIO Host Controller can be configured to operate at normal data rate(25MHz) and also in a High Speed data rate mode(50MHz). For run time configuration, the CNTRL\_FSM\_SPEED\_CLASS\_PKG\_CONFIGURABLE parameter in the package shall be set to a 0, and the processor can program the Speed Class section of SiliconBlue Specific Configuration Register. A “1” in this register indicates a switch to a high speed operation after initialization and a “0” indicates a switch to a normal transfer rate of 25MHz. For compile time configuration, the CNTRL\_FSM\_SPEED\_CLASS\_PKG\_CONFIGURABLE shall be set to a “1” and parameter CNTRL\_FSM\_SPEED\_CLASS\_SUPPORTED shall be set to what mode is required.
5. Reset initialization statistics : The processor can reset the initialization statistics by programming the SiliconBlue Specific Configuration Register bit 1 to a 1. The processor shall de-assert it once it is written. The bit acts as a soft reset to the status registers.
6. Clocking factors : The SDIO Controller shall be operating in 3 different frequencies. When the Controller is in the initialization mode, it operates at 400KHz, and under data transfer, depending on the Speed class that is configured in the SiliconBlue Specific Configuration Register (Address = 0x64), the Host Controller operates at around 25MHz for normal data rate, and at 50MHz for High Speed operation. The controller uses the i\_clk provided to the system to derive the 3 clocks. The division factors are defined in the package and can be compile time programmable and depending on the frequency that can be provided to the system. The CLK\_TREE\_INITIALIZATION\_CLK\_DIV\_FACTOR defined in the package shall be set to a value which is used to divide the system clock to extract the initialization clock of 400 KHz. The CLK\_TREE\_DEFAULT\_SPEED\_CLK\_DIV\_FACTOR shall be used to extract the default speed clock for normal data rate operation. The CLK\_TREE\_HIGH\_SPEED\_CLK\_DIV\_FACTOR shall be used to extract the high speed clock for High speed operation.
7. VDD window configuration : The VDD voltage window can be configured depending on the board VDD supply voltage. This is necessary since the SD/MMC card shall operation only under matching VDD conditions which are negotiated by the SD Controller. The CNTRL\_FSM\_VDD\_WINDOW\_SUPPORTED parameter shall be set depending on the board VDD supply. The value is set to “0100” which indicates a 2.4–3.6v VDD range.
8. Command registers : The processor can execute a SDIO command using a system driver environment using this register. The processor shall program the Argument0 [15:0](Address=0x08) and Argument1[15:0](Address = 0x0A) with the command arguments that shall be inserted for the command that will be posted. The processor then shall program the Command register [15:0](Address=0x0E). For posting the command, the processor can then check the response on the Response Register[127:0](Address 0x10) by using the single block read transactions multiple times to read the entire 16 bytes of response register using the single block read transactions multiple times to read the entire 16 bytes of response register.
9. SDHC support : The SD Host Controller shall identify a SDHC Card (High Capacity SD Card). The information is conveyed to the processor via the CPU registers. The SiliconBlue Specific Status Register shall contain Card related flags and information. When the Card is detected to be SDHC card, the processor shall make sure that the BLOCK SIZE Register is always set to a value corresponding to a 512Byte block size.

### Design Details

**Custom Processor Interface :** The processor interface used here is a general purpose 32-bit multiplexed address and data bus interface. The processor shall issue the address and data for read and write operation on the same multiplexed bus. The processor is informed about the Host Controller status using “read only” status registers in the CPU registers map. The processor can configure the Host Controller by writing different configuration registers.

**CPU Register Set :** Consisting of the SDIO Controller register set which can be used by the Host Driver running on the Processor.

**DMA Architecture :** Used for buffering the data to be read/written from/to SDIO Card through the Host Controller. The processor may hence may operate at a higher rate compared to data transfer between the Host Controller and the SDIO Card.

**Control FSM :** Used for initialization and command posting mechanism.

**SDIO PHY Interface :** for SDIO memory side physical interface that handles the SDIO data transfer modes (SD 1 bit Mode, SD 4-bit Mode, SPI Mode).

## Timing Diagram

**Single Word Read Transaction :** This transaction can be used to access a single 32 bit word from the SDIO Controller. This transaction is mainly used for accessing the CPU register set for reading response bytes, status registers, read write register values. The timing diagram of the transaction is shown in Figure 3

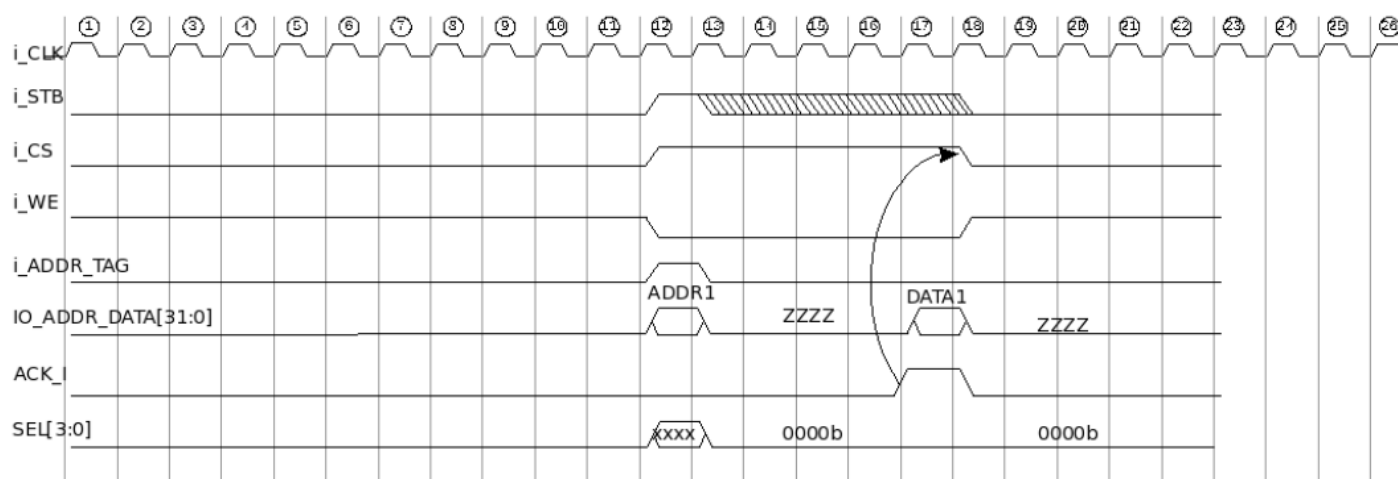


Figure 3: Single word read cycle on the processor interface

**Single word write transaction :** This transaction can be used to access a single 32 bit word from the SDIO Controller for programming purposes. This transaction is mainly used for writing configuration registers for forcing initialization of the SDIO memory, for executing SD commands from processor if required. For clearing status registers using soft resets. The timing diagram of the Single Word Write transaction is shown in Figure 4

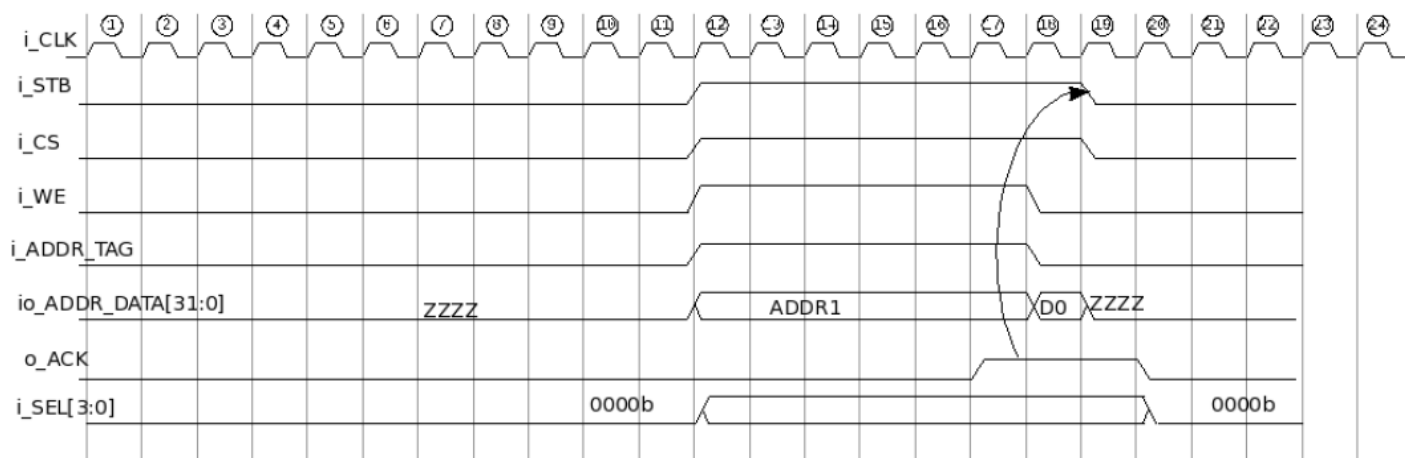


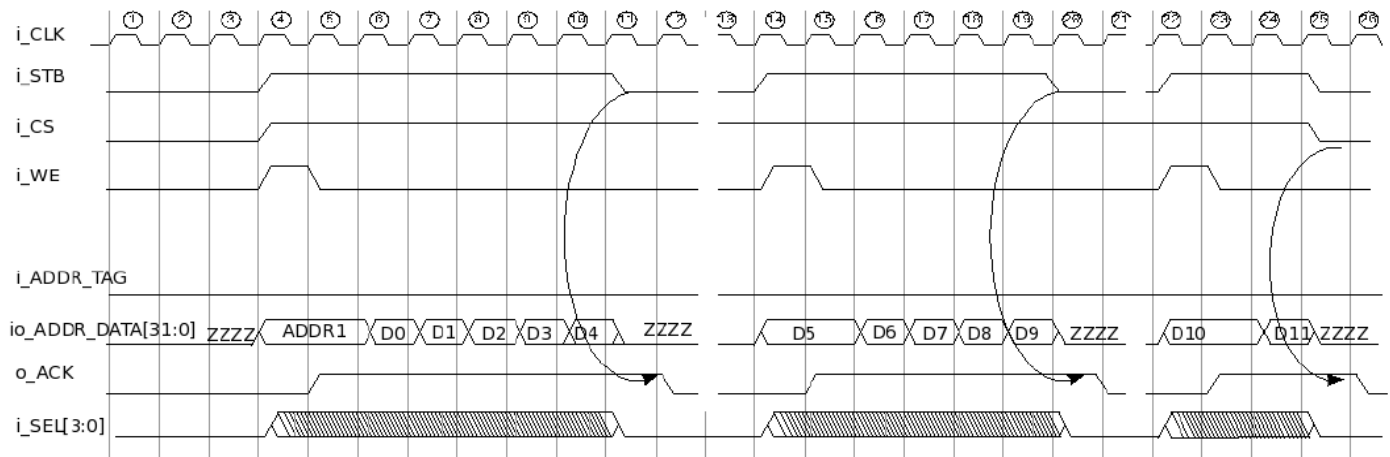
Figure 4: Single word write transaction on the processor interface

**Multiblock Read Transaction :** The Processor can transfer multiple blocks of data from the SDIO card via the Host Controller using this transaction. The processor shall write the Block count register before performing this transaction. The host Controller shall transfer the number of blocks configured in the Block Count Register, from the SDIO card to the Read Write buffer and then transfer it to the processor. The sequence of operation for a multi/single block read from the SDIO card is as follows:

- The processor shall configure the Block Count Register [15:0] (Address= 0x06) with the number of blocks to be fetched. The maximum block count can be 16(“0000010”) for an 8Kbytes Read Write buffer. The register is configured using the single word write transaction.

- The process shall also program the Block Size Register[15:0](Address= 0x04) with a 1 for a 1024Byte block configuration, and a 0 for 512 Byte block size configuration. By default, the register is configured to 512 byte mode.
- The processor shall also program the address from which of the SD Host Controller shall read the blocks of data into the Argument0 Register [31:0] (Address = 0x08) with a 32 bit address value. The SD host controller shall post multi block or single block read commands with the address argument being the value of the register Argument0.
- The processor can now read the entire blocks of data that it configured for using the Multi Block Read transaction.
- The processor can abort the read transaction in before it reads out the entire block length of data by de-asserting the chip select.

Figure 5 shows a Multi Block Read Transaction on the processor bus.



**Figure 5: Multi block read transaction on the processor interface**

**MultiBlock Write Transaction :** The Processor can transfer multiple blocks of data to the SDIO card via the Host Controller using this transaction. The processor shall write the Block count register before performing this transaction. The host Controller shall transfer the number of blocks configured in the Block Count Register, from the Read Write buffer to the SDIO card. The sequence of operation for a multi/single block write transaction to the SDIO card is as follows :

- The processor shall configure the Block Count Register [15:0](Address= 0x06) with the number of blocks to be transferred. The maximum block count can be 16(x"0000010") for an 8Kbytes Read Write buffer. The register is configured using the single word write transaction.
- The process shall also program the Block Size Register [15:0](Address= 0x04) with a 1 for a 1024Byte block configuration, and a 0 for 512 Byte block size configuration. By default, the register is configured to 512 byte mode.
- The processor shall also program the address from which of the SD Host Controller shall read the blocks of data into the Argument0 Register [31:0] (Address = 0x08) with a 32 bit address value. The SD host controller shall post multi block or single block write commands with the address argument being the value of the register Argument0.
- The processor can now write the entire blocks of data that it configured for, using the Multi Block Write transaction.
- The processor can abort a write transaction before the minimum block length of data is transferred from the processor(less than 512 bytes), by de-asserting the chip select on the processor interface.

Figure 6 shows a multiblock write transaction on the processor bus

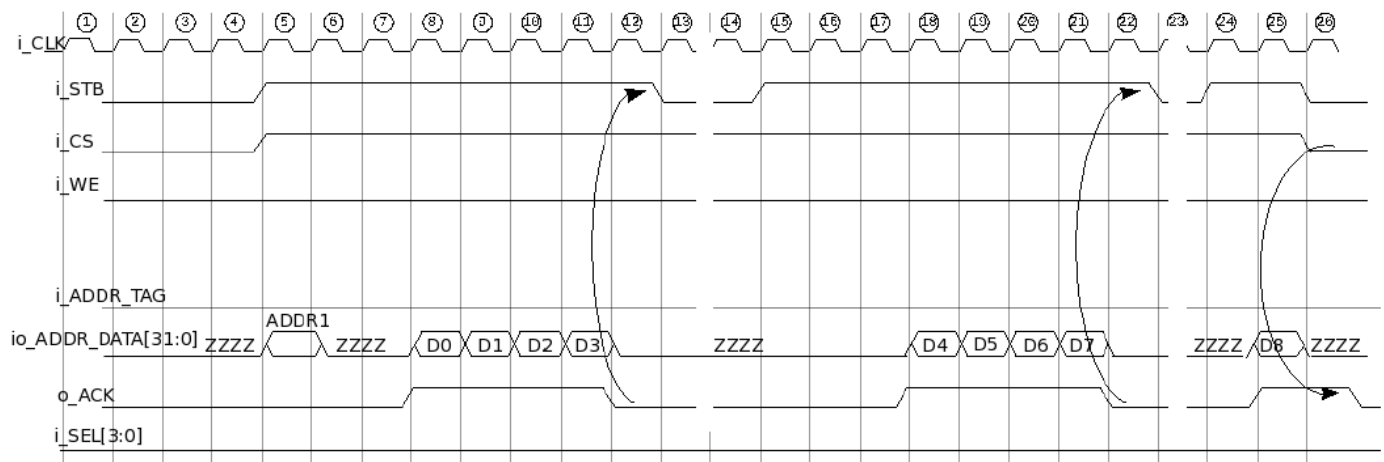


Figure 6: Multiblock write transaction on the processor bus

## System Designer Flow

SDIO Controller is compatible with System Designer/IP-XACT 1.2. The System Designer flow is as follows,

- 1) Launch the System Designer from Synplify Pro using menu 'Import -> Launch System Designer'.
- 2) Create a new project (open an existing old project, as necessary) and import the IP-XACT XML file
- 3) Drag and place the component from the 'Library' pane to the 'Design' pane
- 4) Click on the "Generate Files" button, which generates the necessary files required for synthesis and simulation.
- 5) Go to Synplify Pro, click on the "Run" button to synthesize the System Designer generated files. Synplify Pro generates all the necessary files for P&R in iCECube2.

## References

The following references were used in the creation of this design:

- SiliconBlue Technologies, Inc. "[iCE65 Ultra Low-Power mobileFPGA Family](#)" datasheet (26-May-2010).
- SD Specifications: Part 1, Physical Layer Specification, Version 2.00, May 9, 2006
- SD Specifications: Part A2, SD Host Controller Standard Specification, Version 2.00, January 30, 2007
- SD Specifications: Part E1 SDIO Simplified Specification Version 2.00 Feb8, 2007

## Revision History

| Version | Date        | Description            |
|---------|-------------|------------------------|
| 1.0     | 15-SEP-2010 | Initial Draft Document |
| 1.1     | 08-DEC-2010 | IP-XACT Format Update  |

## Disclaimer

Copyright © 2007–2009 by SiliconBlue Technologies LTD. All rights reserved. SiliconBlue is a registered trademark of SiliconBlue Technologies LTD in the United States. Specific device designations, and all other words and logos that are identified as trademarks are, unless noted otherwise, the trademarks of SiliconBlue Technologies LTD. All other product or service names are the property of their respective holders. SiliconBlue products are protected under numerous United States and foreign patents and pending applications, maskwork rights, and copyrights. SiliconBlue warrants performance of its semiconductor products to current specifications in accordance with SiliconBlue's standard warranty, but reserves the right to make changes to any products and services at any time without notice. SiliconBlue assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by SiliconBlue Technologies LTD. SiliconBlue customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.



SiliconBlue Technologies Corporation

3255 Scott Blvd.,  
Building 7, Suite 101  
Santa Clara, CA 95054

Tel: 408-727-6101  
Fax: 408-727-6085

[www.SiliconBlueTech.com](http://www.SiliconBlueTech.com)