SD/eMMC Host Controller IP DFI Interface

Internal Name: SDHC
Part Number: IP6061

NDA # _________

Interfaces

Revision: 0.3

Confidentiality Notice

Table of Contents

List of Figures

List of Tables

1. *SDHC* DFI Interfaces

1.1. Overview

This document describes the interface protocol that defines the connectivity between a *SDHC* and the Cadence Combo PHY and SD/eMMC devices. The protocol defines the signals, signal relationships and timing parameters required to transfer control information, read and write data to and from the *SDHC* devices over the DFI. This document mainly focus on the DFI interface. For other signals not presented in this document please refer to the user guide.

1.1.1. Architecture

All signals defined by the DFI are required to be driven by registers clocked on the rising edge of the DFI clock (sdmclk). The DFI places no restrictions on the source of the DFI clock. The only requirement is that the DFI clock must exist and all DFI related signals must be referenced from this clock. Compatibility between the *SDHC* and the PHY at given frequencies is dependent on the specification of both the output timing for signals driven and the setup and hold requirements for reception of these signals on the DFI.

Figure 1.1. Top Level Interfaces of the *SDHC* to the PHY

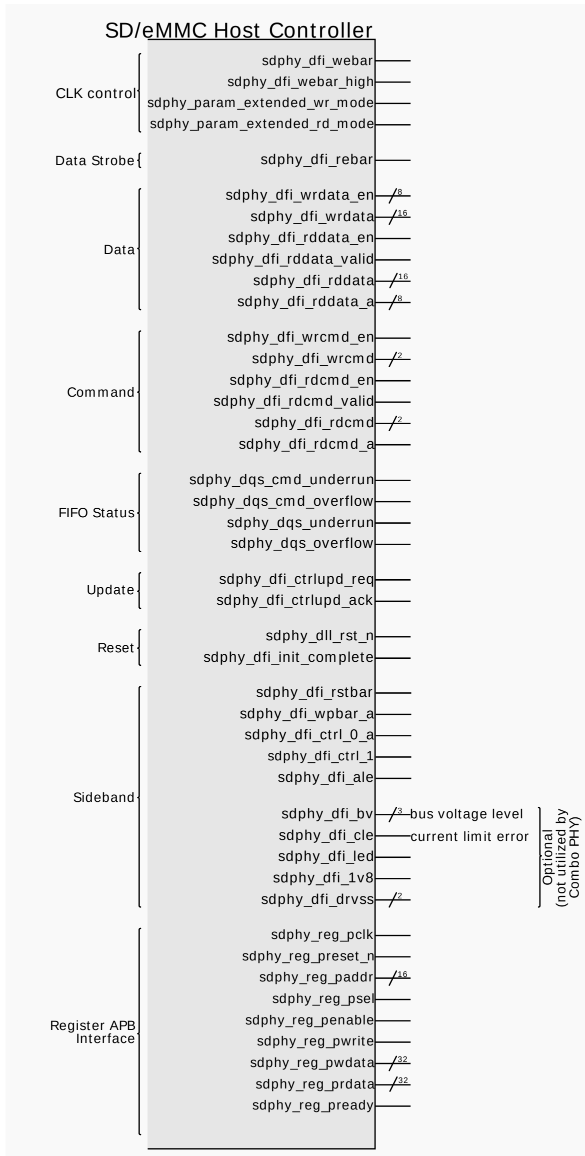


Figure 1.2. Top Level Interfaces of the Combo PHY

1.1.2. Interface Signal Groups

The DFI is subdivided into the following interface groups:

- CLK control interface
- Data Strobe interface
- Data Interface
- Command Interface
- FIFO Status Interface
- Update Interface
- Reset Interface
- Side-band Interface
- Register APB Interface

CLK control interface is to enable, disable and halt the SDCLK (card clock) generation logic in the PHY. Command interface is used to pass valid command and receive valid response. Data interface is used to pass valid write and receive valid read data across the DFI. Data Strobe interface backs up Command and Data interface to adjust strobe point in the PHY. FIFO Status Interface informs about unexpected events on the PHY internal Command and Data FIFOs. Update Interface enables to re-calibrate PHY master DLL. Reset Interface resets PHY DLL logic and confirms its readiness. Side-band interface controls eMMC reset, Write Protect, Power Supply Enable, Power Supply Voltage, Card Detect, DAT2 Pull-Up/Pull-Down, Current Limit Error, LED, 1.8V Signaling Enable, Drive Strength. Register APB Interface accesses to PHY registers through its APB Slave interface.

1.1.3. Interface

Table 1.1. Combo PHY Interface

Signal	Width	Dir	Src clk	Description
sdphy_dfi_rebar	1	output	sdmclk	Port utilized for Phony DQS generation.
sdphy_dfi_rdcmd_a	1	input	N/A	CMD line value read from IO Cell without any synchronization.
sdphy_dfi_rdcmd_valid	1	input	sdmclk	Value read from CMD line is ready for read.
sdphy_dfi_rdcmd_en	1	output	sdmclk	Request CMD line read.
sdphy_dfi_rdcmd	1	input	sdmclk	Value read from CMD line and synchronized to target clock domain.
sdphy_dfi_wrcmd_en	1	output	sdmclk	Request CMD line write.
sdphy_dfi_wrcmd	2	output	sdmclk	Value written to CMD line.
sdphy_dfi_rddata_a	8	input	sdmclk	DAT lines value read from IO Cells without any synchronization.
sdphy_dfi_rddata_valid	1	input	sdmclk	Value read from DAT lines is ready for read.
sdphy_dfi_rddata_en	1	output	sdmclk	Request Read DAT lines.
sdphy_dfi_rddata	16	input	sdmclk	Value read from DAT lines and synchronized to target clock domain.
sdphy_dfi_wrdata_en	8	output	sdmclk	Request DAT lines write.
sdphy_dfi_wrdata	16	output	sdmclk	Value written to DAT lines.
sdphy_dfi_webar	1	output	sdmclk	Controls SDCLK clock generation.
sdphy_dfi_webar_high	1	output	sdmclk	Park SDCLK clock in HIGH.
sdphy_dfi_wpbar_a	1	input	sdmclk	Value read from Write Protect.
sdphy_param_extended_rd_mode	1	output	sdmclk	Changes clock mode generation: (0) SDCLK=SDMCLK and (1) SDCLK=SDMCLK/2*N.
sdphy_param_extended_wr_mode	1	output	sdmclk	Changes clock mode generation: (0) SDCLK=SDMCLK and (1) SDCLK=SDMCLK/2*N.
sdphy_dfi_ctrlupd_ack	1	input	sdmclk	Confirmation that requested DLL update is completed.
sdphy_dfi_ctrlupd_req	1	output	sdmclk	Confirm DLL update.
sdphy_dll_rst_n	1	output	sdmclk	Software reset for DLL logic
sdphy_dfi_dqs_overflow	1	input	sdmclk	Internal FIFO on DAT path has been written when full.
sdphy_dfi_dqs_underrun	1	input	sdmclk	Internal FIFO on DAT path has been read when empty.
sdphy_dfi_dqs_cmd_overflow	1	input	sdmclk	Internal FIFO on CMD path has been written when full.
sdphy_dfi_dqs_cmd_underrun	1	input	sdmclk	Internal FIFO on CMD path has been read when empty.

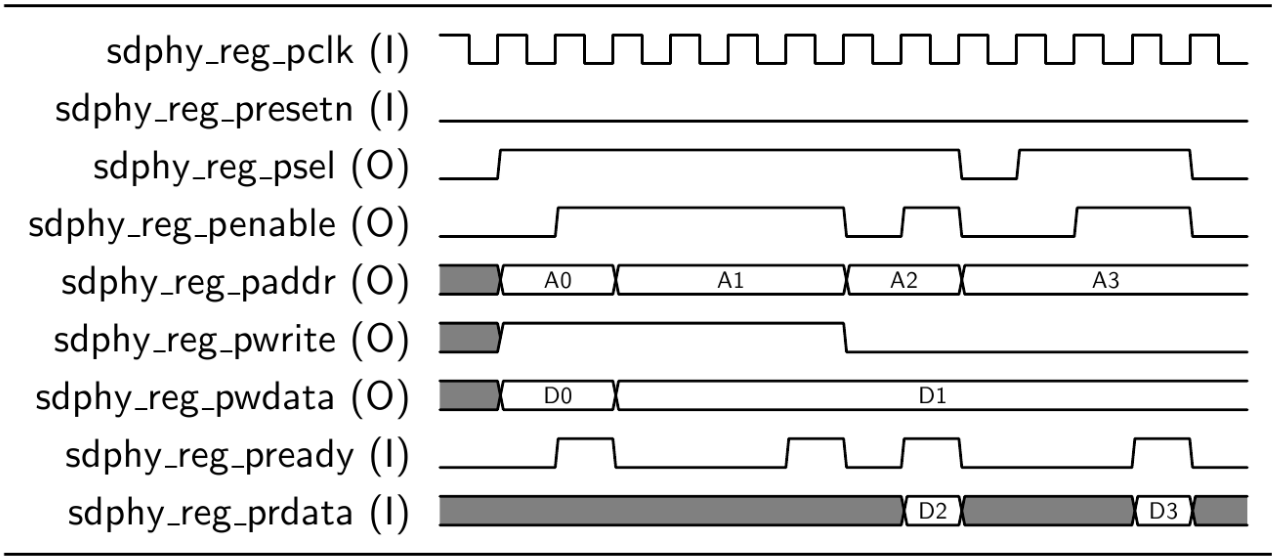
Signal	Width	Dir	Src clk	Description
sdphy_dfi_init_complete	1	input	sdmclk	Indicates that PHY has finished initialization stage and is ready to work.
sdphy_dfi_rstbar	1	output	clk	eMMC device reset request.
sdphy_dfi_ctrl_0_a	1	input	N/A	Card Detect signal coming from SD slot. Signal is asynchronous.
sdphy_dfi_ctrl_1	1	output	clk	Bus Power enables power supply for SD/ eMMC device.
sdphy_dfi_ale	1	output	clk	Control over pull-up/pull-down resistor required in LVSI procedure on DAT2 line.
sdphy_dfi_bv	3	output	clk	Bus Voltage goes optionally along with Bus Power and indicates Power Supply voltage level.
sdphy_dfi_cle	1	input	clk	Optional signal informs whether the external Power Supply for SD/eMMC exceed its limits.
sdphy_dfi_led	1	output	clk	Optional signal to enable/disable LED information on PCB.
sdphy_dfi_1v8	1	output	clk	Switches between signalization in 3.3V and 1.8V. Applicable only for SD.
sdphy_dfi_drvss	2	output	clk	Switches drive strength of IOs.
sdphy_reg_psel	1	output	sdphy_reg_pclk	Selector for APB slave. Starts APB transaction.
sdphy_reg_penable	1	output	sdphy_reg_pclk	Indicates second consecutive part of APB transaction.
sdphy_reg_paddr	16	output	sdphy_reg_pclk	Register address
sdphy_reg_pwrite	1	output	sdphy_reg_pclk	Indicates direction of the transaction.
sdphy_reg_pwdata	32	output	sdphy_reg_pclk	32-bit of written data
sdphy_reg_pready	1	input	sdphy_reg_pclk	Slave acknowledge transaction.
sdphy_reg_prdata	32	input	sdphy_reg_pclk	32-bit read data

1.2. Functional Use

1.2.1. APB Master Interface

Master APB Interface enables to map the PHY registers into the Host Controller register set and access those through HRS04/ HRS05. Read from or write to HRS05 creates APB read/write transactions. This interface is general purpose - it may be utilized to connect Controller to PHY but it is not mandatory.

Figure 1.3. APB Interface



1.2.2. DFI Initialization

Host Controller waits for dfi_init_complete signal to be released (High). While the signal is asserted, the DFI signals remains at default value, and PHY settings can be reprogrammed. Once the dfi_init_complete signal is asserted, all other DFI signals are able to assert in accordance with the device protocol.

dfi_init_complete is allowed to be asserted only on Power-On-Reset or dll_rst_n.

1.2.3. Normal operation

Host Controller DFI operates in four modes:

- SDR with clock divider (a.k.a. extended mode)
- DDR with clock divider (a.k.a. extended mode)
- SDR without clock divider (a.k.a. normal clock mode)
- DDR without clock divider (a.k.a. normal clock mode)

1.2.4. Clock control

DFI controls the SDCLK (pad_mem_webar_t) clock that supplies SD / eMMC device.

1.2.4.1. Normal clock mode (extended mode = 0)

The output clock generation starts in 2 SDMCLK clock cycles after dfi_webar gets to Low (shown in Figure 1.4). The clocks generation stops in 1.5 SDMCLK clock cycles after the dfi_webar gets back to High (shown in Figure 1.5).

Figure 1.4. SDCLK clock enable in normal clock mode

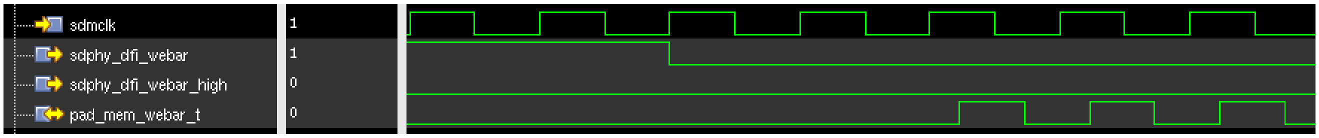
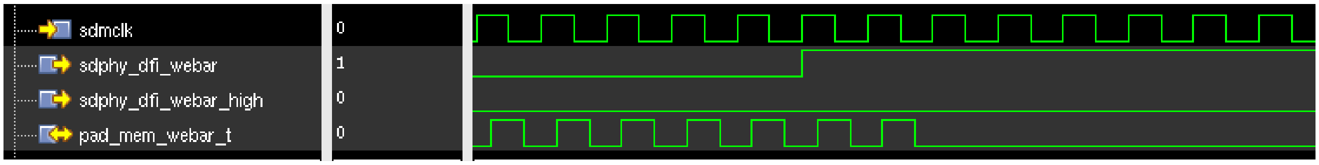


Figure 1.5. SDCLK clock disable in normal clock mode



The clock can be paused/resumed during the data transfer as an flow-control mechanism. This is shown in Figure 1.6 and Figure 1.7. The clock is paused in one SDMCLK cycle after dfi_webar_high gets to High. Clock resumes in 1.5 SDMCLK clock cycle.

Figure 1.6. SDCLK clock pause in normal clock mode

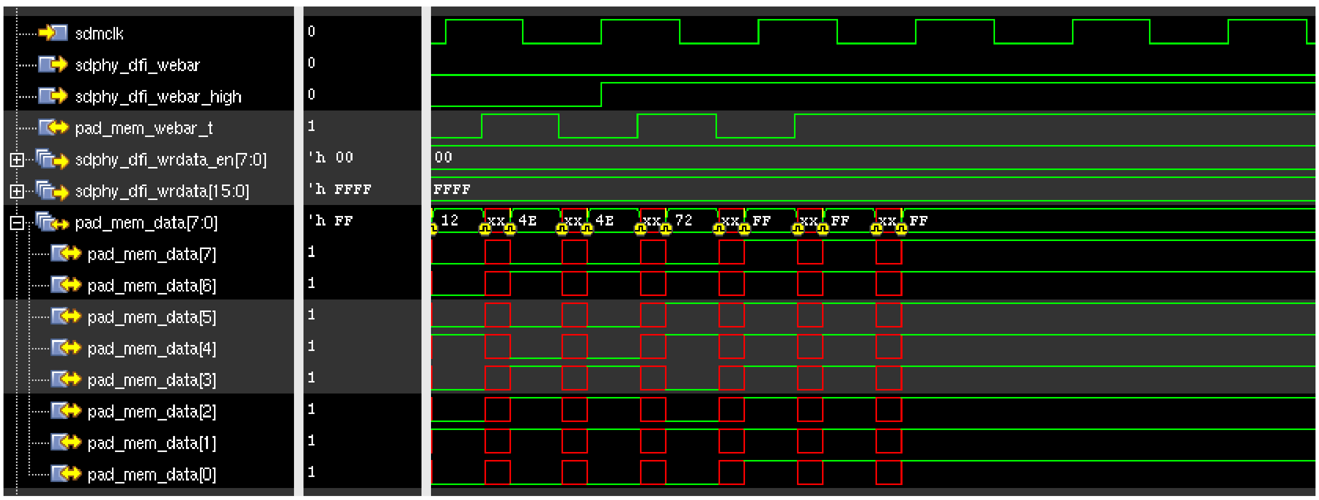
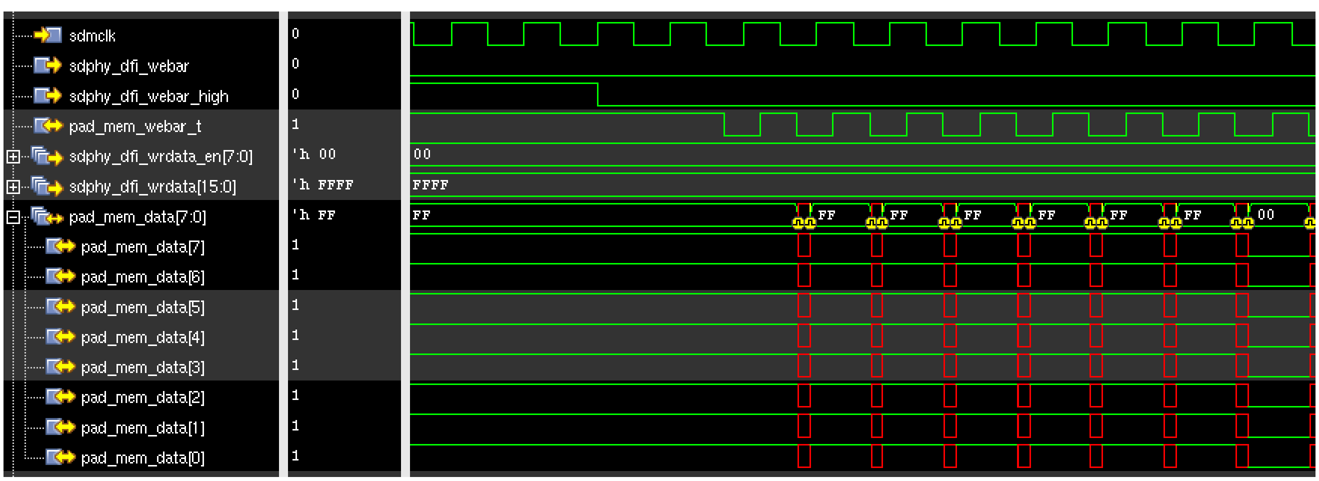


Figure 1.7. SDCLK clock resume in normal clock mode



1.2.4.2. Extended clock mode

The SDCLK clock frequency is controlled rather by the host controller. The host creates a pattern of 0's and 1's that is mimicked on the clock output with 2 SDMCLK delay. The output clock transition to High is shown in Figure 1.8 and the transition to Low is shown in Figure 1.9.

Figure 1.8. SDCLK clock enable in extended clock mode

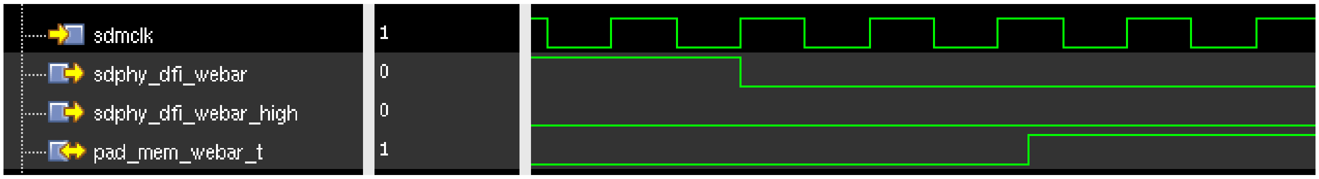
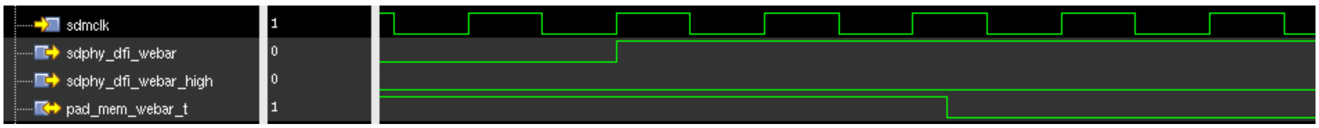


Figure 1.9. SDCLK clock disable in extended clock mode



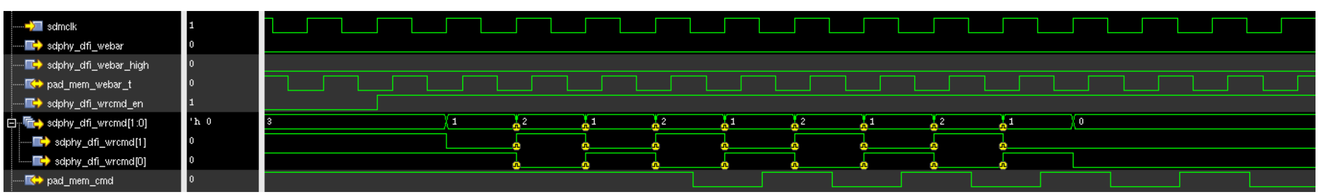
The clock can be paused/resumed for data flow-control. In this case, the host controller stimulates dfi_webar to pause clock edge generation.

1.2.5. Command Line

1.2.5.1. Normal clock mode

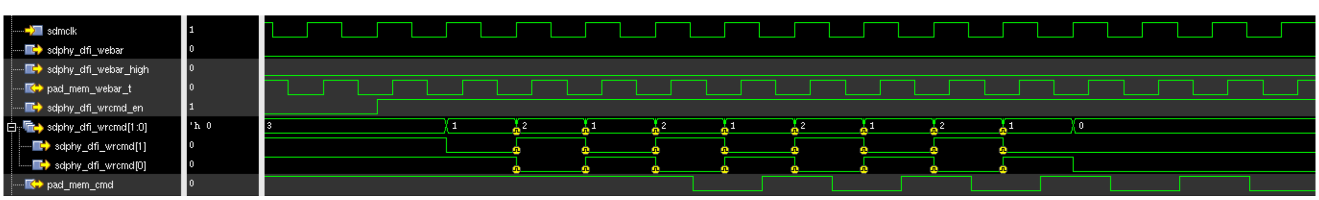
The CMD line is a bidirectional port. When Host wants to transfer a command frame to device, it sets the dfi_wrcmd_en 1 SDMCLK clock cycle before the expected command start bit (Figure 1.10)

Figure 1.10. CMD output enable in normal clock mode



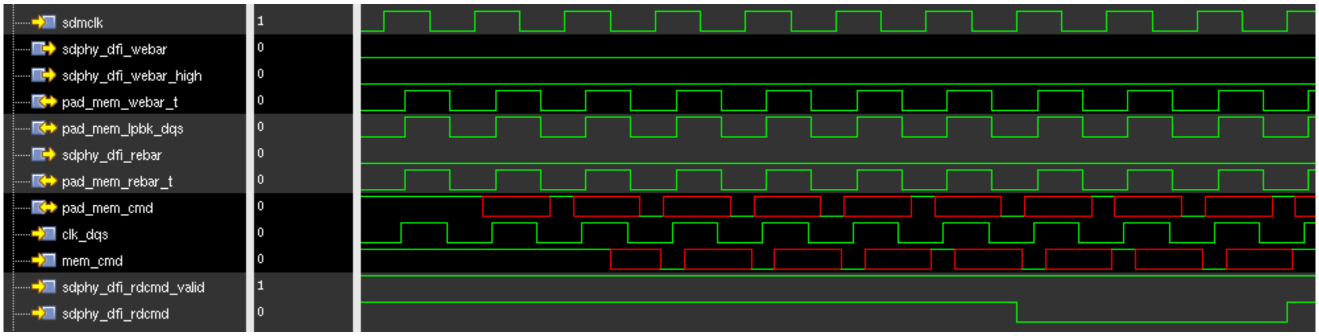
The CMD line is driven by the dfi_wrcmd when dfi_wrcmd_en is equal to 1. dfi_wrcmd values appears on the CMD line after 3 SDMCLK clock cycles. The host controller is able to delay the data by N*0.5 SDMCLK clock cycle where N is programmable. Figure 1.11 shows an example with 3.5 SDMCLK clock cycle delay.

Figure 1.11. CMD output in normal clock mode



Host Controller configures the PHY to utilize rebar_t/lpbk_dqs to sample receiving data. Data from the CMD line (pad_mem_cmd) goes into the PHY (mem_cmd) and is sampled by clk_dqs (delayed pad_mem_dqs_t). Sampled data, after being synchronized, is provided from the PHY to Host on the read data bus (dfi_rdcmd). The dfi_rdcmd carries valid data only when the dfi_rddata_valid is high.

Figure 1.12. CMD input in normal clock mode

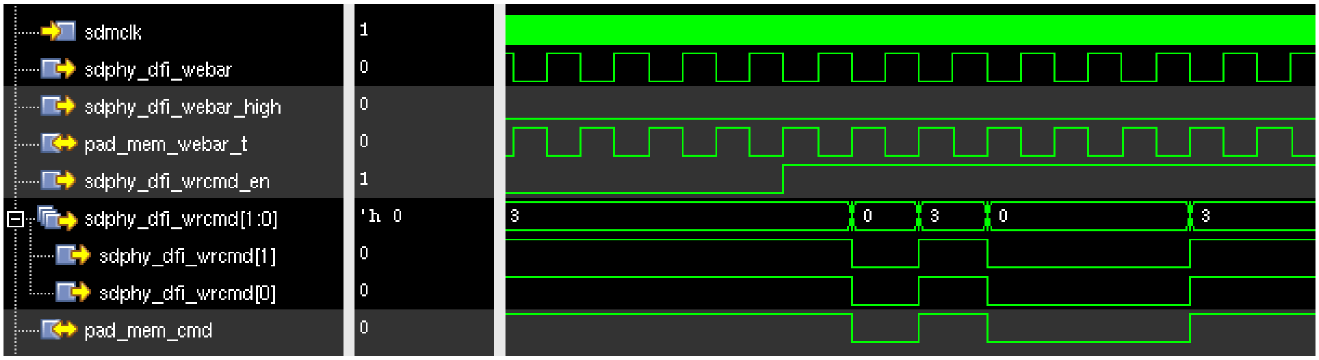


1.2.5.2. Extended clock mode

The CMD line is a bidirectional port. When Host wants to transfer a command frame to device, it sets the dfi_wrcmd_en 1 SDMCLK clock cycle before the expected command start bit (Figure 1.13)

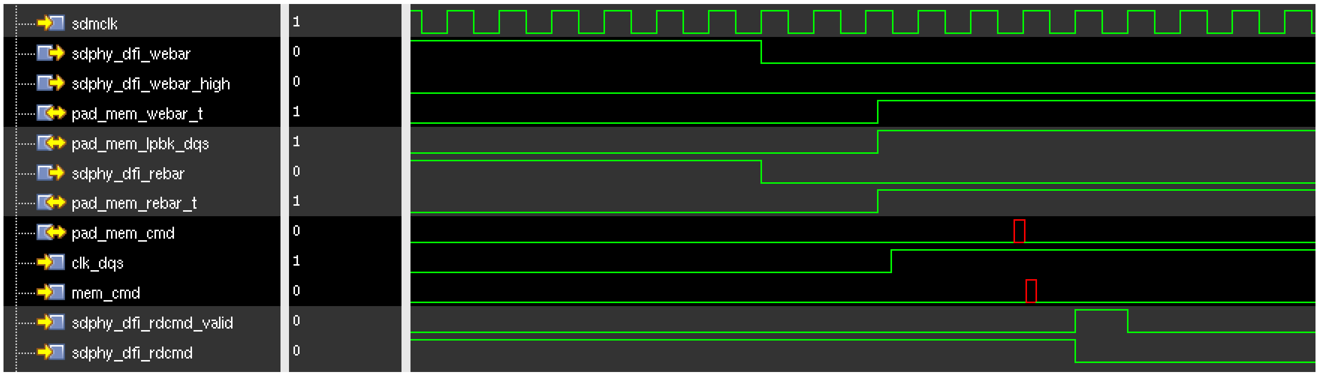
The CMD line is driven by the dfi_wrcmd when dfi_wrcmd_en is equal to 1. dfi_wrcmd values appears on the CMD line after 3 SDMCLK clock cycles. The host controller is able to delay the data by N*0.5 SDMCLK clock cycle where N is programmable.

Figure 1.13. CMD output in extended clock mode



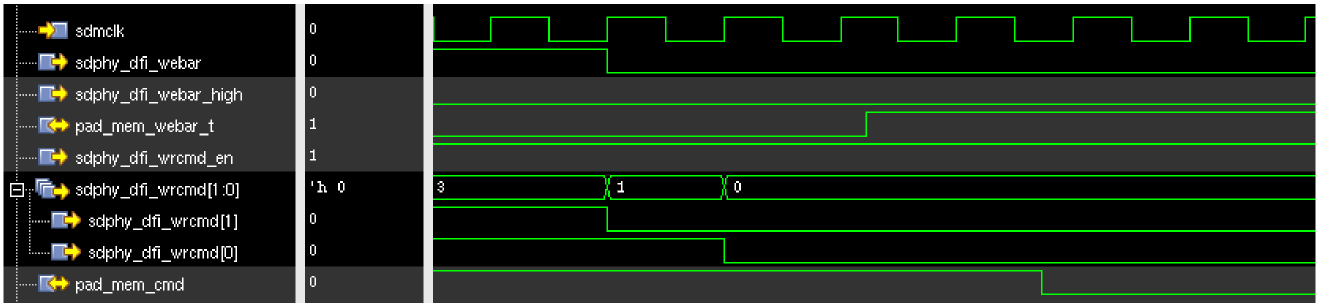
Host Controller configures the PHY to utilize rebar_t/lpbk_dqs to sample receiving data. Data from the CMD line (pad_mem_cmd) goes into the PHY (mem_cmd) and is sampled by clk_dqs (delayed pad_mem_dqs_t). Sampled data, after being synchronized, is provided from the PHY to Host on the read data bus (dfi_rdcmd). The dfi_rdcmd carries valid data only when the dfi_rddata_valid is high.

Figure 1.14. CMD input in extended clock mode



Host Controller configures the PHY to utilize rebar_t/lpbk_dqs to sample receiving data. Data from the CMD line (pad_mem_cmd) goes into the PHY (mem_cmd) and is sampled by clk_dqs (delayed pad_mem_dqs_t). Sampled data, after being synchronized, is provided from the PHY to Host on the read data bus (dfi_rdcmd). The dfi_rdcmd carries valid data only when the dfi_rddata_valid is high.

Figure 1.15. CMD output in extended clock mode



1.2.6. Data Line

1.2.6.1. Normal clock mode

1.2.6.1.1. DDR mode

Host Controller sets the write data output enable (dfi_wrdata_en) on the falling edge of the dfi_webar, and one SDCLK clock cycle before actual data transfer starts on the write data bus (dfi_wrdata).

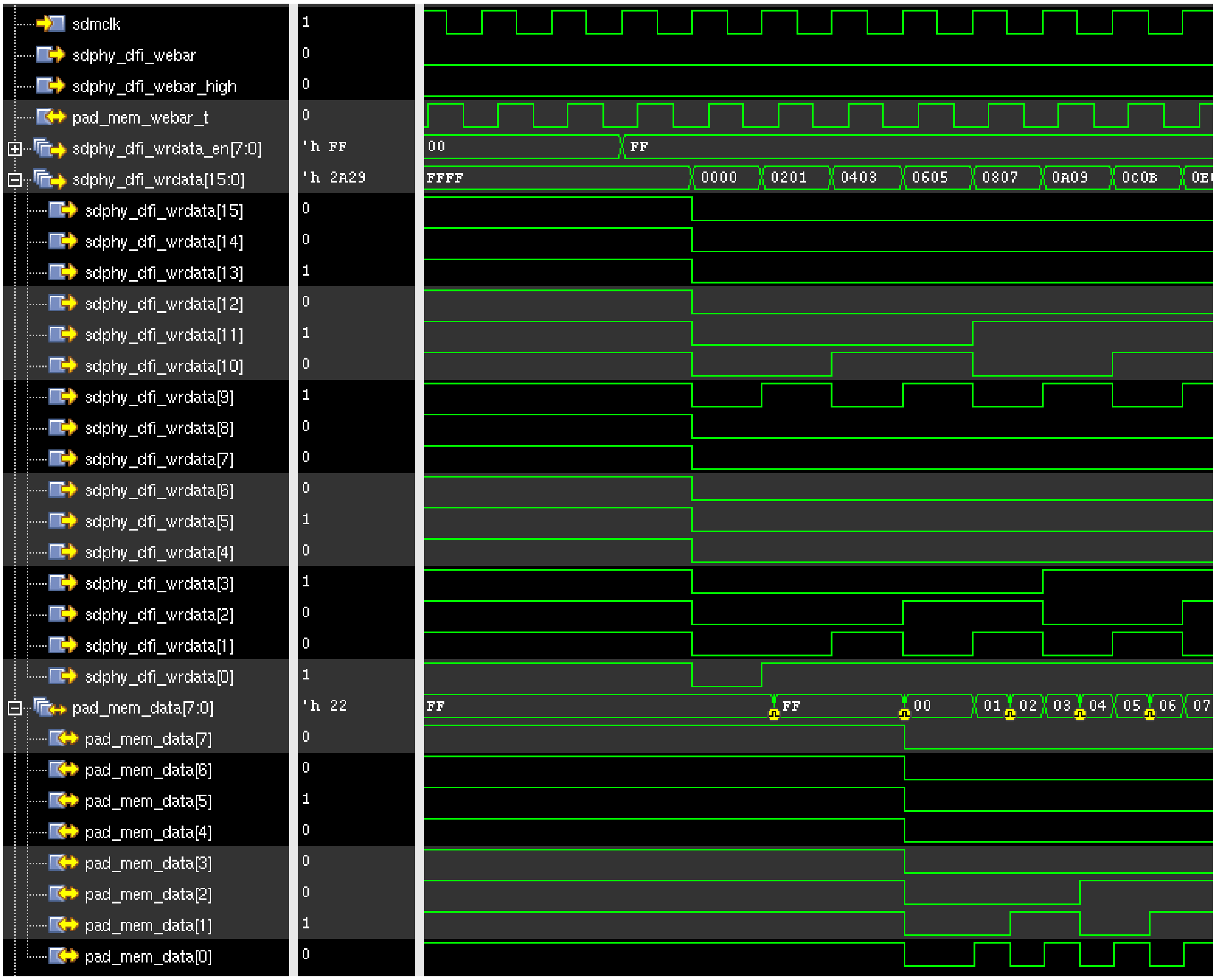
The dfi_wrdata_en value depends on the Bus Width settings - 0x1 (1-bit width), 0xf (4-bit width), and 0xff (8-bit width).

Host Controller sends a data sequence over the write data bus (dfi_wrdata). The data sequence comprise of a start bit(s), a data block, a CRC status and an end bit. Number of utilized write data bus (dfi_wrdata) bits depends on the Bus Width setting and can be 1, 4 or 8.

In DDR mode and normal clock mode, new sets of data appears every single SDCLK (dfi_webar) clock cycle.

Figure 1.16 shows an example write transfer.

Figure 1.16. DAT output in normal clock mode, DDR mode

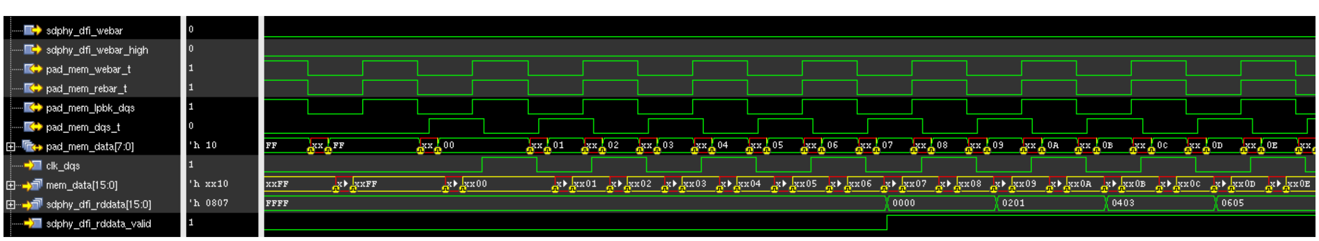


Host Controller configures the PHY to utilize rebar_t/lpbk_dqs to sample receiving data. Data from the DAT line (pad_mem_data) goes into the PHY (mem_data) and is sampled by clk_dqs (delayed pad_mem_dqs_t). Sampled data, after being synchronized, is provided from the PHY to Host on the read data bus (dfi_rddata). The dfi_rddata carries valid data only when the dfi_rddata_valid is high.

In DDR mode and normal clock mode, dfi_rddata[7:0] carries data sampled on rising edge and dfi_rddata[15:0] carries data sampled on falling edge.

Figure 1.17 shows an example read transfer.

Figure 1.17. DAT input in normal clock mode, DDR mode

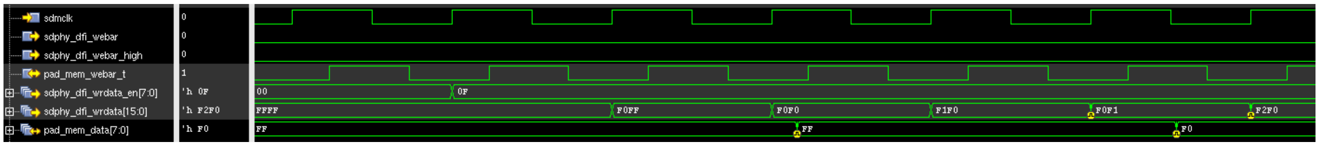


1.2.6.1.2. SDR mode

Host Controller sets the write data output enable (dfi_wrdata_en) on the falling edge of the dfi_webar, and one SDCLK clock cycle before actual data transfer starts on the write data bus (dfi_wrdata) (Figure 1.18).

The dfi_wrdata_en value depends on the Bus Width settings - 0x1 (1-bit width), 0xf (4-bit width), and 0xff (8-bit width).

Figure 1.18. DAT output enable in normal clock mode, SDR mode

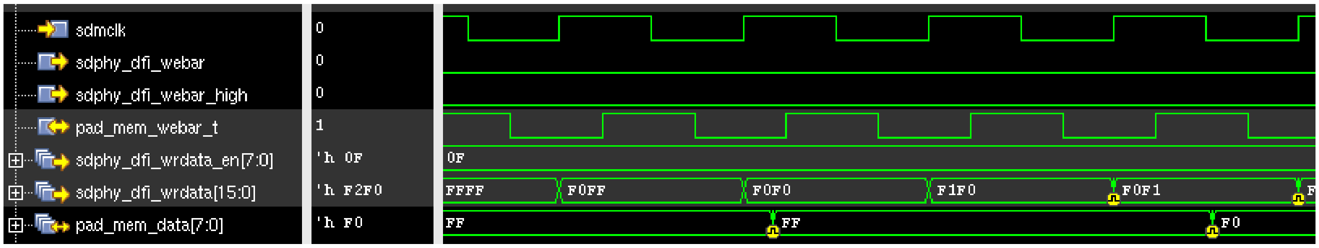


Host Controller sends a data sequence over the write data bus (dfi_wrdata). The data sequence comprise of a start bit(s), a data block, a CRC status and an end bit. Number of utilized write data bus (dfi_wrdata) bits depends on the Bus Width setting and can be 1, 4 or 8.

In SDR mode and normal clock mode, new set of data appears every single SDCLK (dfi_webar) clock cycle.

Figure 1.19 shows an example write transfer.

Figure 1.19. DAT output in normal clock mode, SDR mode

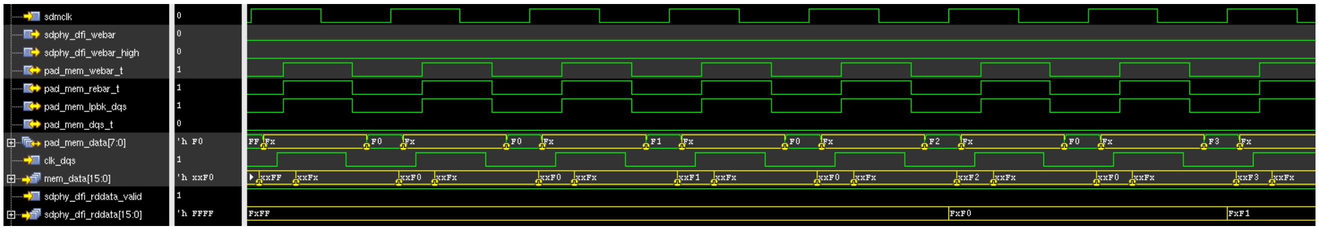


Host Controller configures the PHY to utilize rebar_t/lpbk_dqs to sample receiving data. Data from the DAT line (pad_mem_data) goes into the PHY (mem_data) and is sampled by clk_dqs (delayed pad_mem_dqs_t). Sampled data, after being synchronized, is provided from the PHY to Host on the read data bus (dfi_rddata). The dfi_rddata carries valid data only when the dfi_rddata_valid is high.

In DDR mode and extended clock mode, dfi_rddata[7:0] carries data sampled on rising and falling edge.

Figure 1.20 shows an example write transfer.

Figure 1.20. DAT input in normal clock mode, SDR mode



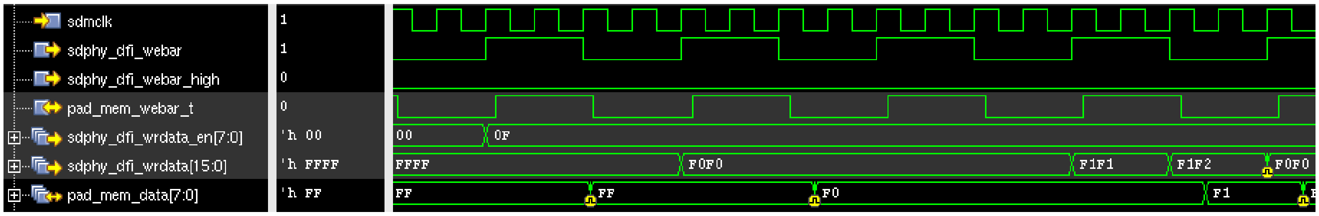
1.2.6.2. Extended clock mode

1.2.6.2.1. DDR mode

Host Controller sets the write data output enable (dfi_wrdata_en) on the falling edge of the dfi_webar, and one SDCLK clock cycle before actual data transfer starts on the write data bus (dfi_wrdata) (Figure 1.21).

The dfi_wrdata_en value depends on the Bus Width settings - 0x1 (1-bit width), 0xf (4-bit width), and 0xff (8-bit width).

Figure 1.21. DAT output enable in extended clock mode, DDR mode

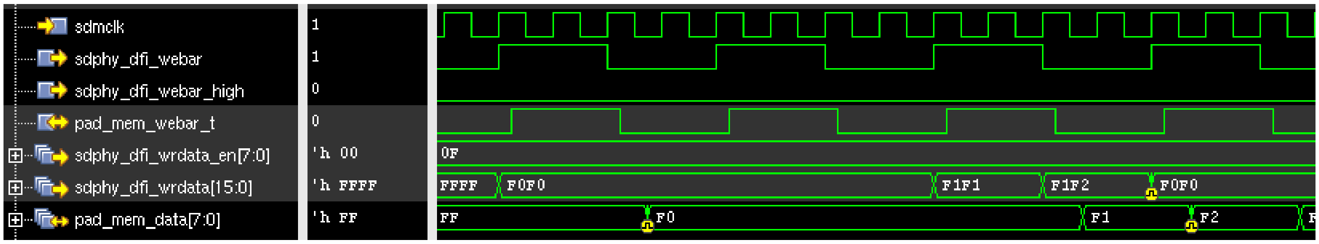


Host Controller sends a data sequence over the write data bus (dfi_wrdata). The data sequence comprise of a start bit(s), a data block, a CRC status and an end bit. Number of utilized write data bus (dfi_wrdata) bits depends on the Bus Width setting and can be 1, 4 or 8.

In DDR mode and extended clock mode, new set of data appears twice every single SDCLK (dfi_webar) clock cycle.

Figure 1.22 shows an example write transfer.

Figure 1.22. DAT output in extended clock mode, DDR mode

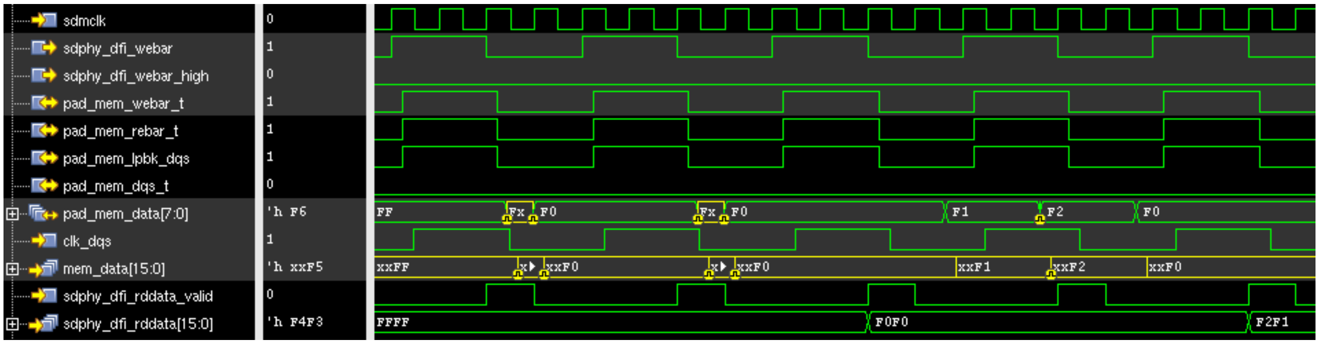


Host Controller configures the PHY to utilize rebar_t/lpbk_dqs to sample receiving data. Data from the DAT line (pad_mem_data) goes into the PHY (mem_data) and is sampled by clk_dqs (delayed pad_mem_dqs_t). Sampled data, after being synchronized, is provided from the PHY to Host on the read data bus (dfi_rddata). The dfi_rddata carries valid data only when the dfi_rddata_valid is high.

In DDR mode and extended clock mode, dfi_rddata[7:0] carries data sampled on rising and falling edge.

Figure 1.23 shows an example read transfer.

Figure 1.23. DAT input in extended clock mode, DDR mode

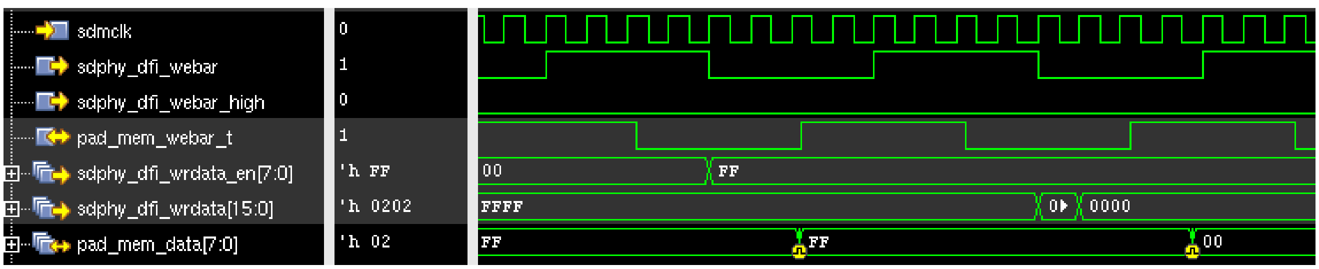


1.2.6.2.2. SDR mode

Host Controller sets the write data output enable (dfi_wrdata_en) on the falling edge of the dfi_webar, and one SDCLK clock cycle before actual data transfer starts on the write data bus (dfi_wrdata) (Figure 1.24).

The dfi_wrdata_en value depends on the Bus Width settings - 0x1 (1-bit width), 0xf (4-bit width), and 0xff (8-bit width).

Figure 1.24. DAT output enable in extended clock mode, SDR mode

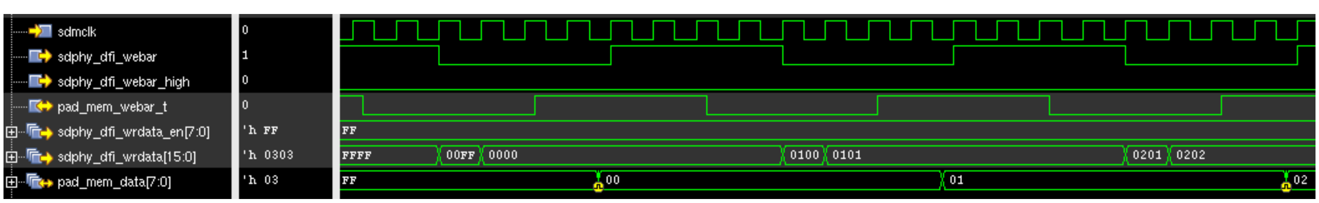


Host Controller sends a data bit sequence over the write data bus (dfi_wrdata). The data sequence comprise of a start bit(s), a data block, a CRC status and an end bit. Number of utilized write data bus (dfi_wrdata) bits depends on the Bus Width setting and can be 1, 4 or 8.

In SDR mode and extended clock mode, new set of data appears every single SDCLK (dfi_webar) clock cycle.

Figure 1.25 shows an example write transfer.

Figure 1.25. DAT output in extended clock mode, SDR mode

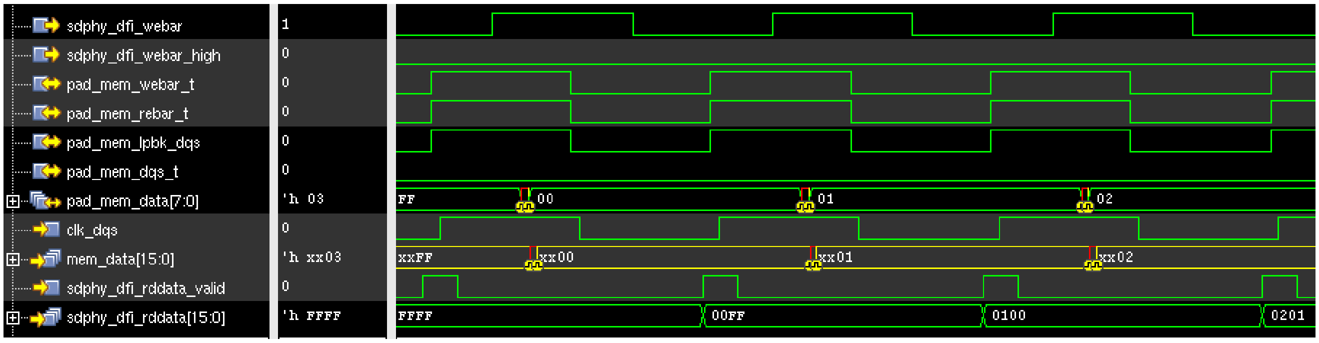


Host Controller configures the PHY to utilize rebar_t/lpbk_dqs to sample receiving data. Data from the DAT line (pad_mem_data) goes into the PHY (mem_data) and is sampled by clk_dqs (delayed pad_mem_dqs_t). Sampled data, after being synchronized, is provided from the PHY to Host on the read data bus (dfi_rddata). The dfi_rddata carries valid data only when the dfi_rddata_valid is high.

In SDR mode and extended clock mode, dfi_rddata[7:0] carries data sampled on rising edge.

Figure 1.26 shows an example read transfer.

Figure 1.26. DAT input in extended clock mode, SDR mode



1.2.7. Command/Data lines timing adjustment

The Host Controller can increase the PHY command/data output hold time (pad_mem_webar_t to pad_mem_cmd/pad_mem_data period) by delaying the write data bus (dfi_wrcmd/dfi_wrdata). Hold time can be increased by (N*0.5 SDMCLK) clock cycle. Delay by N*SDMCLK clock cycle can be applied by adding N*SDMCLK clock cycle delay on the write data bus (dfi_wrcmd/dfi_wrdata). Additional +0.5 SDMCLK clock cycle can be applied when dfi_wrcmd[0]/dfi_wrdata[7:0] is delayed by (N+1) SDMCLK clock cycle. The host has registers to control the delay:

- WRCMD0_DLY - delay of dfi_wrcmd[0] in SDMCLK clock cycles
- WRCMD1_DLY - delay of dfi_wrcmd[1] in SDMCLK clock cycles
- WRDATA0_DLY - delay of dfi_wrdata[7:0] in SDMCLK clock cycles
- WRDATA1_DLY - delay of dfi_wrdata[15:8] in SDMCLK clock cycles

1.2.8. SD Features

1.2.8.1. Flow-control

When the Host Controller reads data from device, it can happen that Host Controller internal buffer is occasionally filled. This for example can be a result of system interface bandwidth limitation. When DMA/SRS08 transfer is not as quick as on SD/eMMC interface transfer, host controller activates the flow-control mechanism. The mechanism stops the SDCLK clock (clock provided to the device) between data block.

The Host prevents device from starting a new data block by stopping the SDMCLK. This mechanism requires a calibration. There is an uncertainty due to output clock path and input data pat which is related to PHY delays, IO Pad delays, IO Pad to device propagation time, SD/eMMC timings, etc.

Provided script calculates correction coefficient based on input factors.

Invalid correction coefficient (HSSDCLKADJ) value may lead to protocol errors such as Data CRC Error, Data End Bit Error, Data Timeout Error.

Figure 1.27 and Figure 1.27 shows an example read transfer.

- t_{OUTDLY} - output pad delay
- t_{INDLY} - input pad delay
- t_{CLKDLY} - PHY clock path delay
- t_{PHYDLY} - PHY data path delay; internal synchronization mechanism cause uncertainty - the delay in N or N+1 sdmclk clock cycles
- CLKADJ - HSSDCLKADJ register value that adjust moment of disabling the clock

Figure 1.27. Flow-control (2 clock cycles CDC)

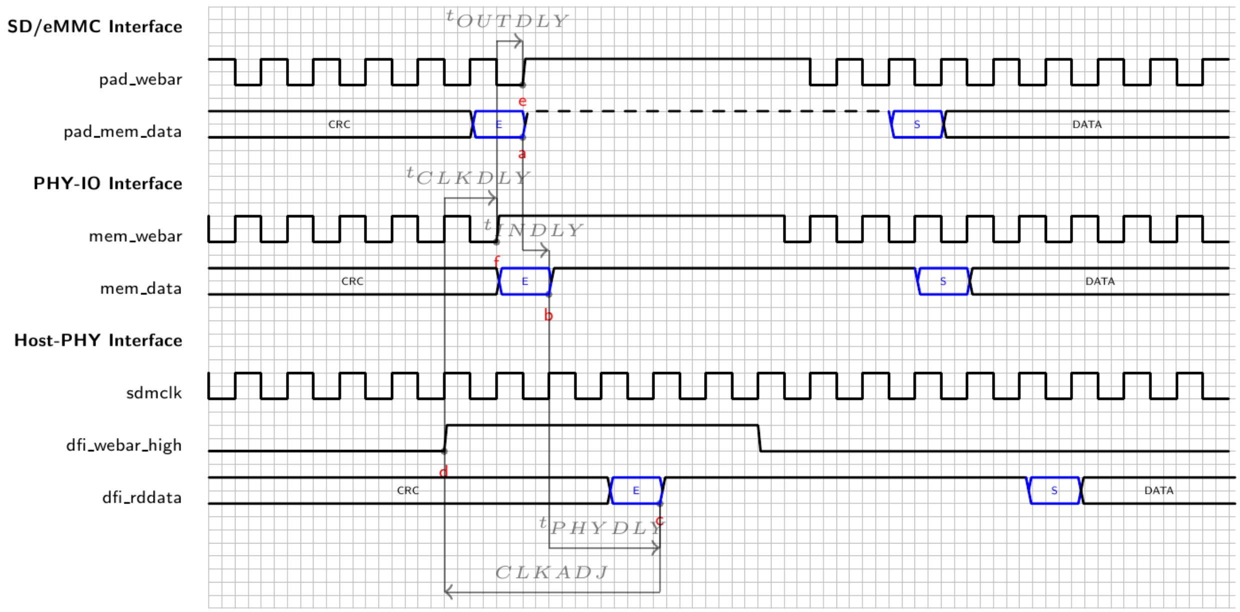
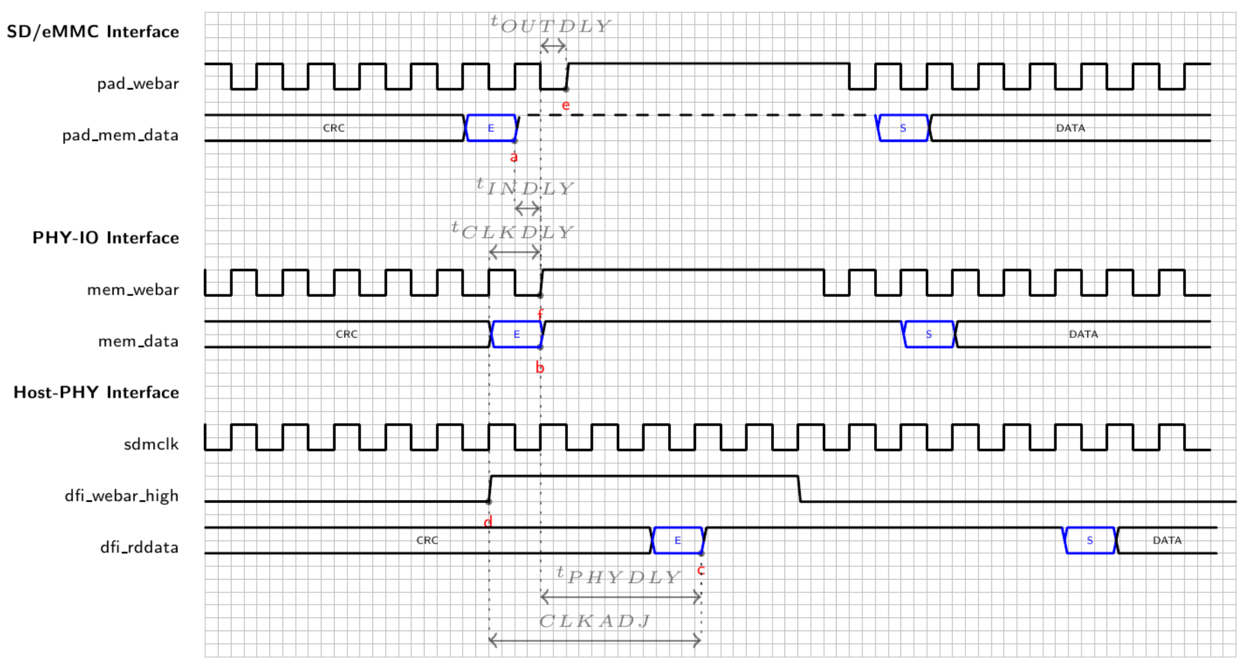


Figure 1.28. Flow-control (3 clock cycles CDC)



1.2.8.2. Interrupt

SD protocol enables a device to send an interrupt signal to request Host/SoC reaction.

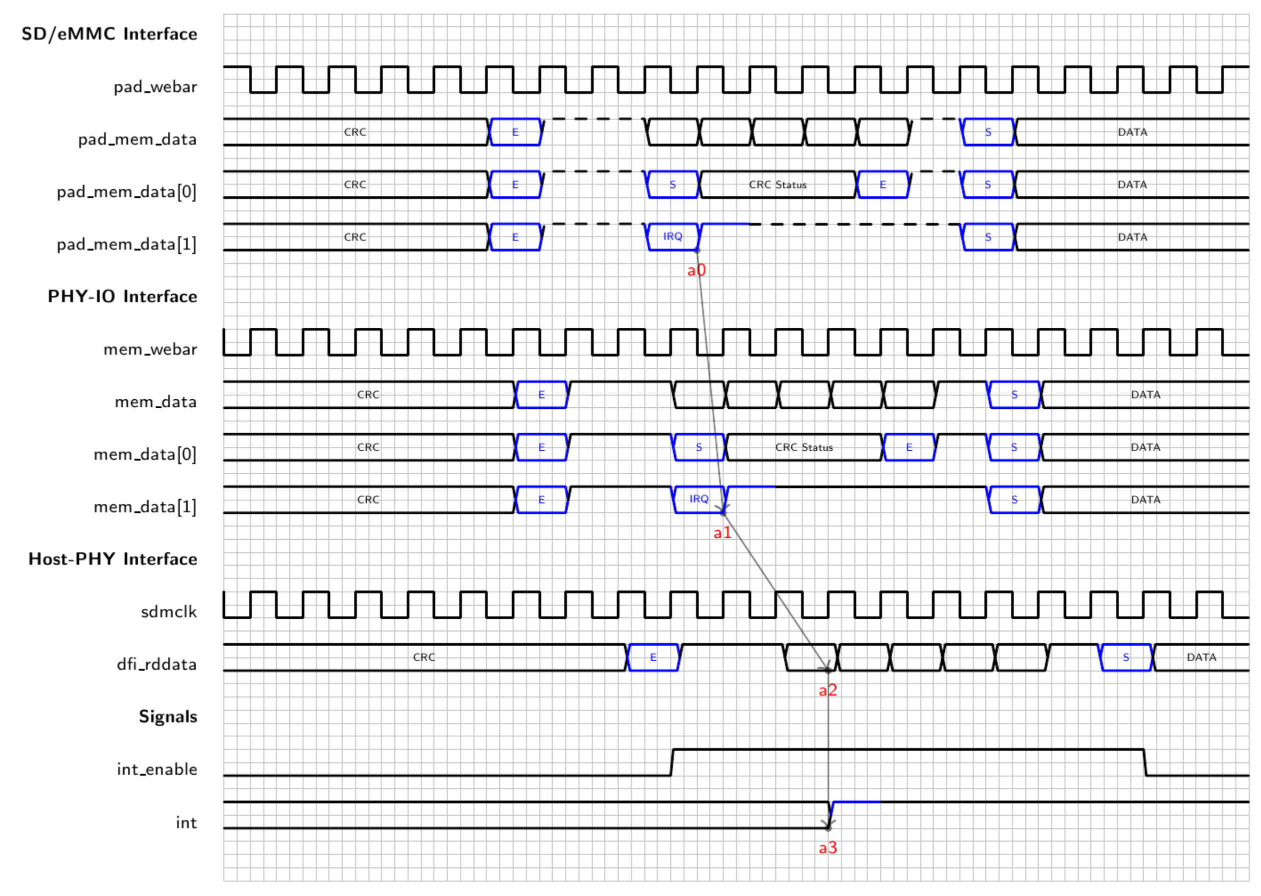
SDIO (device) sends an interrupt through DAT[1] line. The line is utilized for a data transfer and the interrupt signaling. To prevent collision the interrupt is signalized only when no data transfer is active, 1-bit DAT bus width setting is enabled, or withing a data block gaps.

This section covers the last case where Host Controller receives information about pending interrupt during a data transfer. Device sends the interrupt signal in the read/write data block gap, between receiving/transmitting data frames. Example write transfer with the active interrupt information (IRQ) is shown in Figure 1.29. IRQ appears 2 clock cycles after the end bit (E) for a single SDCLK (pad_webar) clock cycle.

Receiving data (dfi_rddata) includes the interrupt information are delayed due to PCB, IO Pads, PHY, etc. propagation delay. Host Controller requires a setting value which tells how to deal with the delays and recognize the valid IRQ instead of e.g. data/CRC.

The timing calculation script provides the setting value to update the IDELAY_VAL (HRS07) register. This register value adjusts the interrupt enable (int_enable) to activate interrupt detection logic for the specific period and deactivates the logic (masks input line) to prevent false interrupt detection.

Figure 1.29. Interrupt



1.2.8.3. Read-Wait

SDIO (device) has an alternative flow-control mechanism that can be enabled by software. This mechanism pauses a multiple read data block transfer sending the Read-Wait signal on DAT[2] line.

Both, data transfer and Read-Wait indication utilize DAT[2] line. The setting adjustment is required to prevent overlapping on the common line. An invalid setting may lead to protocol violation and the signal misinterpretation or data corruption.

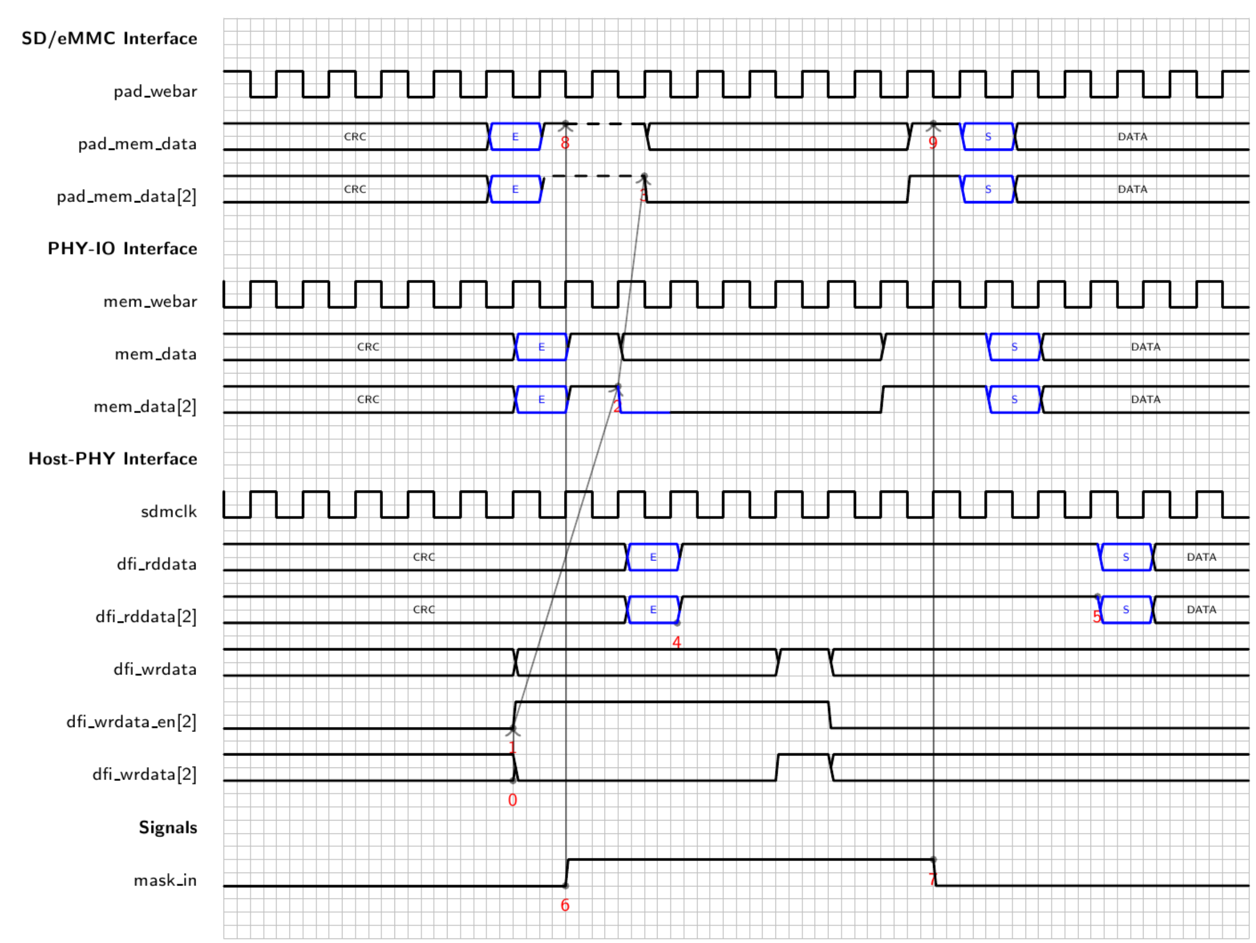
The timing calculation script provides the setting value to update the RW_COMPENSATE (HRS07) register. This register value adjusts the start moment of the Read-Wait signal. The standard defines that Read-Wait begins 2 clock cycles after the end bit (DS, HS, SDR12, SDR25) and 2 to 5 clock cycles after the end bit (SDR50, SDR104, DDR50).

The parameter compensates existing DAT[2] output delay (responsible for Read-Wait signalization) and DAT input delay (responsible for the end bit detection). The Read-Wait starts 2 clock cycles after the data end bit. The Host Controller detects the end bit with latency of PCB, IO Pads, PHY, etc. The DAT[2] is driven with a latency of path from Host Controller, through PHY, IO Pads and PCB line delay to the device.

Figure 1.30 shows an example read data transfer with Read-Wait function enabled. The compensation settings moves dfi_wrdata[2] 3 clock before the pad_mem_data[2] goes L (Read-Wait activation). This enables to start the Read-Wait exactly 2 clock cycles after the end bit on the pad_mem_data[2] (DAT[2] line).

In the figure, transfers on write data (dfi_wrdata) and read data (dfi_rddata) overlap. It may happen that driven Read-Wait interferes with a read data block. PHY can prevent this situation using internal mask_in signal. Software responsibility is to adjustment the mask_in signal position by applying the io_mask_start script output setting to the PHY register setting.

Figure 1.30. Read-Wait



1.2.9. Update Interface

Combo DLL PHY requires the slave DLLs periodical re-synchronization to track the master DLL values to compensate PVT variations. Software driver is obliged to run a sequence that requests DLL update (dfi_ctrlupd_req) and waits for DLL update acknowledge (dfi_ctrlupd_ack). Software controls the dfi_ctrlupd_req signal over the Host Controller Register Set (HRS07.PHY_DLL_UPDREQ) and monitors the dfi_ctrlupd_ack signal over the HRS07.PHY_DLL_UPDACK.

The re-synchronization sequence execution affects temporarily the SD / eMMC interface thus it is permitted when the Host Controller SD/eMMC interface is idling (no ongoing CMD or DAT transfer). Transfers can be resumed after PHY DLL update acknowledge.

Appendix A. Document Revision History

Table A.1. Document Revision History

Revision	Modification	Data	Author
0.3	Update Functional Use Section - added description of clock, command, data lines and SD Features	5 April 2019	mbarb@cadence.com
0.2	First review	15 March 2019	mbarb@cadence.com
0.1	First Draft Release	30 July 2018	mbarb@cadence.com