Customer Training Workshop
Traveo™ II SDHC Host Controller
## Target Products

### Target product list for this training material

<table>
<thead>
<tr>
<th>Family Category</th>
<th>Series</th>
<th>Code Flash Memory Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traveo™ II Automotive Body Controller High</td>
<td>CYT3BB/4BB</td>
<td>Up to 4160KB</td>
</tr>
<tr>
<td>Traveo II Automotive Body Controller High</td>
<td>CYT4BF</td>
<td>Up to 8384KB</td>
</tr>
</tbody>
</table>
Introduction to Traveo II Body Controller High

- SDHC Host Controller is located in the Peripheral blocks

CPU Subsystem

- eCT Flash 8384KB Code flash + 256KB Work flash
- Arm Cortex-M7 350 MHz
- 512KB SRAM0
- 256KB SRAM1
- 256KB SRAM2
- 16KB Flash Controller
- 16KB TCM
- 16KB DTCM

System Interconnect (Multi Layer AXI/AHB, IPC, MPU/SMPU)

Peripheral Interconnect (MMIO, PPU)

- 11x TCPWM
- 2x LIN
- 10x SBC
- 10x CANFD
- 1x FLEXRAY
- 1x GMCAN
- 2x Ethernet
- 8x SMIF
- 5x Smart I/O
- 256KB SRAM1
- 128KB SRAMMUX

High-Speed I/O Matrix, Smart I/O, Boundary Scan

I/O Subsystem

- Up to 196x GPIO_STD, 4x GPIO_ENH, 4x HSIO

System Resources

- Power
- Sleep Control
- DCM
- DVP
- PWRSYS

- Clock
- Clock Control
- PLL
- 4xPLL

- Reset
- Reset Control
- XRES

- Test
- Test Mode Entry
- Digital DFT
- Analog DFT
- SWJ

- WCO
- RTC

Power Modes

- Active/Idle
- LowPower
- Active/Sleep
- DeepSleep
- Hibernate

Review TRM chapter 33 for additional details

Copyright © Infineon Technologies AG 2020. All rights reserved.
SDHC Host Controller Overview

› Overview
  - SDHC Host Controller enables interface with:
    - Secure Digital (SD) cards
    - Secure Digital Input Output (SDIO) cards or WiFi products (e.g. CYW4343W)
    - Embedded Multimedia Card (eMMC)–based memory devices

› Features
  - SD 6.0, SDIO 4.10, and eMMC 5.1 standards
  - Host Controller Interface (HCI) 4.2
  - eMMC/SD/SDIO Interface for 3.3 V
  - Three DMA modes
  - I/O interfaces
  - Card detection
  - Mechanical write protection

Review the following documents for additional details
- Datasheet
- TRM chapter 33
- SD Specifications Part 1 Physical Layer Specification Version 6.00
- SD Specifications Part A2 SD Host Controller Standard Specification Version 4.20
- SD Specifications Part E1 SDIO Specifications Version 4.10
- Embedded Multi-Media Card (eMMC) Electrical Standard 5.1
SDHC Host Controller Block Diagram

- **SDHC Host Controller components**

  - **DMA Engine (SDMA, ADMA2, ADMA3)**
    - Handles direct data transfer between the SDHC logics and system memory.

  - **AHB Master Interface**
    - Transfer data to and from the system memory.
    - Provides access to the configuration registers.

  - **AHB Slave Interface**
    - Interrupts to CPU Sub-System.
    - CLK_HFx
    - CLK_GRx
    - CLK_SLOW

  - **Configuration Registers**

  - **SRAM Controller**
    - Stores data packets while carrying out data transfer to and from the card.

  - **eMMC/SD/SDIO Interface**
    - To I/O Sub-System.

  - **1KB Packet Buffer SRAM**

Review TRM section 33.2 for additional details.
SDHC Host Controller Block Diagram

- SDHC Host Controller components
  - eMMC/SD/SDIO Bus Protocol
  - eMMC/SD/SDIO Interface

Review TRM section 33.1 for additional details
eMMC/SD/SDIO Bus Protocol

Each message is represented by one of the following tokens
- Command: Command is a token that is sent from the host to the card for operation start
- Response: Response is a token that is sent from the card to the host as an answer to a previously received command
- Data: Data is transferred via the data line
- Busy: Block write operation uses a simple busy signaling of the write operation on the DAT0 line

Block Read Operation
- Command
- Response
- Data block
- CRC
- Single Block Read Operation
- Multiple Block Read Operation

Block Write Operation
- Command
- Response
- Data block
- CRC
- Busy
- Single Block Write Operation
- Multiple Block Write Operation
eMMC/SD/SDIO Interface

Data Bus and Speed Modes

<table>
<thead>
<tr>
<th>Signal</th>
<th>Function</th>
<th>eMMC</th>
<th>SD</th>
<th>SDIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDHC_CLK_CARD</td>
<td>Clock output</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SDHC_CARD_CMD</td>
<td>Command (bi-directional)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SDHC_CARD_DAT_3TO0_[3:0]</td>
<td>Data (bi-directional)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SDHC_CARD_DAT_7TO4_[7:4]</td>
<td>Data (bi-directional)</td>
<td>Yes</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SDHC_CARD_DETECT_N</td>
<td>Card detect signal input, Active low</td>
<td>-</td>
<td>Yes</td>
<td>-</td>
</tr>
<tr>
<td>SDHC_CARD_MECH_WRITE_PROT</td>
<td>Mechanical write protect signal input, Active low</td>
<td>-</td>
<td>Yes</td>
<td>-</td>
</tr>
<tr>
<td>SDHC_CARD_IF_PWR_EN</td>
<td>Card interface power enable output</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

- SDHC_CARD_DETECT_N and SDHC_CARD_MECH_WRITE_PROT should be connected to ground if an eMMC or an embedded SDIO device is connected.

Max. Speed Modes
- Legacy: 26 MBps @ 26 MHz, 8-bit
- High Speed SDR: 52 MBps @ 52 MHz, 8-bit
- High Speed DDR: 104 MBps @ 52 MHz, 8-bit
- Default Speed (DS): 12.5 MBps @ 25 MHz, 4-bit

I/O Interface

Review TRM section 33.1 and 33.7, and Register TRM for additional details.
SDHC Host Controller Block Diagram

- SDHC Host Controller components
  - SRAM
    - SRAM Controller
    - Packet Buffer SRAM

> Review TRM section 33.8 for additional details
Packet Buffer SRAM

- Packet buffer SRAM stores the data packets while carrying out data transfer to and from the card
  - SRAM size is 1KB to support buffering of two 512-byte blocks
    - Read Transfer:
      - Read Transfer:
    - Write Transfer: DMA writes data into a packet buffer that is subsequently read by the card interface

Data Flow in Read Transfer

1. Received data from the card interface is written into packet buffer
2. When one block of data is received, DMA starts transmitting that data to the system by reading it from the packet buffer.

Review TRM section 33.8 and Register TRM for additional details.
SDHC Host Controller Block Diagram

- SDHC Host Controller components
  - DMA Engine
    - SDMA
    - ADMA2
    - ADMA3

Review TRM section 33.9 for additional details.

Single Operation DMA (SDMA)
Advanced DMA (ADMA)
DMA Engine

- DMA engine handles data transfer between SDHC and system memory

Features

- Supports SDMA, ADMA2, and ADMA3 modes

<table>
<thead>
<tr>
<th>Modes</th>
<th>Description</th>
<th>Max. Transfer Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDMA</td>
<td>SDMA transfers a data boundary by a read/write command</td>
<td>512KB</td>
</tr>
<tr>
<td>ADMA2</td>
<td>ADMA2 transfers multiple data boundaries by a read/write command</td>
<td>4GB</td>
</tr>
<tr>
<td>ADMA3</td>
<td>ADMA3 performs multiple ADMA2 operations</td>
<td>4EB</td>
</tr>
</tbody>
</table>

- DMA engine enables new task descriptor fetches\(^1\) while DMA is moving data during task execution\(^2\)

\(^1\) For CMD44 and CMD45  
\(^2\) For CMD46 and CMD47
SDMA Operation

- SDMA transfers a data boundary by a read/write command
  - Writing to the command register\(^1\) triggers SDMA transfer
  - Configuration registers for transfer
    - **System Memory Address or Block Count\(^2\):**
      - SDMA uses ADMA System Address Register\(^3\) when Block Count is specified\(^4\)
    - **Data Block Size\(^5\):**
      - 1 byte to 2048 bytes
    - **Size of Contiguous Buffer\(^6\):**
      - 4 KB, 8 KB, 16 KB, 32 KB, 64 KB, 128 KB, 256 KB, 512 KB
ADMA2 Operation

- ADMA2 transfers multiple data boundaries by a read/write command
  - Procedure
    1. Setting ADMA System Memory Address\(^1\) for Descriptor Table start address
    2. Writing to the command register\(^2\) triggers ADMA2 transfer
    3. ADMA2 fetches one descriptor line and executes it in the Descriptor Table
      - This is repeated until the end of the descriptor is found\(^3\)
  - Data Address and Length
    - Minimum unit of address: 4 bytes
    - Maximum data length of each descriptor line: less than 64KB

### Descriptor Table

<table>
<thead>
<tr>
<th>Address</th>
<th>Length</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address 1</td>
<td>Length 1</td>
<td>Transfer</td>
</tr>
<tr>
<td>Address 2</td>
<td>Length 2</td>
<td>Transfer</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Address n</td>
<td>Length n</td>
<td>Transfer &amp; End</td>
</tr>
</tbody>
</table>

### System Memory Map

1. ADMA_SA_LOW_R
2. CMD_R
3. End = 1 in attribute
ADMA3 Operation: Writing to Integrated Address Register

- ADMA3 performs multiple ADMA2 operations. The following slides describe each of the operations.

- Procedure:
  1. Writing to the ADMA3 Integrated Address register\(^1\) triggers ADMA3

\(^1\) ADMA_ID_LOW_R
ADMA3 Operation: Fetching Pointer in Descriptor

Procedure:

1. Writing to the ADMA3 Integrated Address register\(^1\) triggers ADMA3
2. ADMA3 fetches pointers one by one in the Integrated Descriptor and executes descriptors designated by the pointer

---

\(^1\)ADMA_ID_LOW_R
ADMA3 Operation: Issuing a Command

Procedure:

1. Writing to the ADMA3 Integrated Address register\(^1\) triggers ADMA3
2. ADMA3 fetches pointers one by one in the Integrated Descriptor and executes Descriptors designated by the pointer
3. ADMA3 writes the Command Descriptor contents to the Host Controller registers to issue a command and then executes the ADMA2 Descriptor

\(^1\) ADMA_ID_LOW_R
ADMA3 Operation: Executing Next Descriptor

Procedure:

1. ADMA3 fetches the next pointer in the Integrated Descriptor and executes descriptors designated by the pointer.
2. ADMA3 writes the next Command Descriptor contents to the Host Controller registers to issue a command and then executes the ADMA2 descriptor.
ADMA3 Operation: Executing until the end of Descriptor

Procedure:

6. ADMA3 fetches the next pointer in the Integrated Descriptor and executes descriptors designated by the pointer until the end of the Command Descriptor\(^2\)

7. ADMA3 writes the end of the Command Descriptor contents to the Host Controller registers to issue a command and then executes the ADMA2 descriptor
SDHC Host Controller Block Diagram

- SDHC Host Controller components
  - Interrupts
    - Wakeup Interrupts
    - General Interrupts

Review TRM section 33.6 for additional details
Wakeup Interrupts (only for SD/SDIO mode)

- Wakeup SDHC Host Controller from Active/Sleep mode

<table>
<thead>
<tr>
<th>Wakeup Interrupt</th>
<th>Set Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD Card Insertion</td>
<td>Card Insertion</td>
</tr>
<tr>
<td>SD Card Removal</td>
<td>Card Removal</td>
</tr>
<tr>
<td>SDIO Card Interrupt</td>
<td>DAT[1] = 1</td>
</tr>
</tbody>
</table>

Wakeup interrupts cannot wake up from DeepSleep and Hibernate mode.

In DeepSleep mode, only CTL register is retained.

Review TRM section 33.6 and Register TRM for additional details.
General Interrupts (for eMMC/SD/SDIO mode)

- Triggered on all other events, in either normal conditions or error conditions

<table>
<thead>
<tr>
<th>Normal Interrupt</th>
<th>Set Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Complete</td>
<td>Command Complete</td>
</tr>
<tr>
<td>Transfer Complete</td>
<td>Command execution is completed</td>
</tr>
<tr>
<td>Block Gap Event</td>
<td>Transaction stopped at block gap</td>
</tr>
<tr>
<td>DMA Interrupt</td>
<td>DMA Interrupt is generated</td>
</tr>
<tr>
<td>Buffer Write Ready</td>
<td>Ready to write buffer</td>
</tr>
<tr>
<td>Buffer Read Ready</td>
<td>Ready to read buffer</td>
</tr>
<tr>
<td>Card Insertion (SD mode only)</td>
<td>Card Inserted</td>
</tr>
<tr>
<td>Card Removal (SD mode only)</td>
<td>Card Removed</td>
</tr>
<tr>
<td>Card Interrupt (SDIO mode only)</td>
<td>DAT[1] = 1</td>
</tr>
<tr>
<td>FX Event</td>
<td>R[14] = 1 (Response register) and Response Type R1/R5 = 0 (Transfer Mode register)</td>
</tr>
<tr>
<td>Command Queuing Engine Event</td>
<td>Command Queuing related event has occurred (depends on Command Queuing Interrupt Status register)</td>
</tr>
</tbody>
</table>
General Interrupts (for eMMC/SD/SDIO mode)

- Triggered on all other events, in either normal conditions or error conditions

<table>
<thead>
<tr>
<th>Error Interrupt</th>
<th>Set Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Timeout Error</td>
<td>No response is returned within 64 SD clock cycles from the end bit of the Command</td>
</tr>
<tr>
<td>Command CRC Error</td>
<td>Command CRC error occurred</td>
</tr>
<tr>
<td>Command End Bit Error</td>
<td>Detected that the end bit of a command response is 0</td>
</tr>
<tr>
<td>Command Index Error</td>
<td>Command Index error occurred in the command response</td>
</tr>
<tr>
<td>Data Timeout Error</td>
<td>Detected one of the following timeout conditions:</td>
</tr>
<tr>
<td></td>
<td>- Busy timeout for R1b, R5b type</td>
</tr>
<tr>
<td></td>
<td>- Busy timeout after Write CRC status</td>
</tr>
<tr>
<td></td>
<td>- Write CRC Status timeout</td>
</tr>
<tr>
<td></td>
<td>- Read Data timeout</td>
</tr>
<tr>
<td>Data CRC Error</td>
<td>Detected CRC error when transferring read data, which uses the DAT line, when detecting the Write CRC status having a value of other than 010 or when write CRC status timeout occurs.</td>
</tr>
<tr>
<td>Data End Bit Error</td>
<td>Detected 0 at the end bit position of read data that uses the DAT line or at the end bit position of the CRC status.</td>
</tr>
<tr>
<td>Current Limit Error</td>
<td>Power fail occurred</td>
</tr>
<tr>
<td>Auto CMD Error</td>
<td>Detected that any of the bits D00 to D05 in Auto CMD Error Status register have changed from 0 to 1.</td>
</tr>
<tr>
<td>ADMA Error</td>
<td>ADMA error occurred</td>
</tr>
<tr>
<td>Response Error (SD/SDIO mode only)</td>
<td>An error is detected in a response</td>
</tr>
</tbody>
</table>
SDHC Host Controller Block Diagram

- SDHC Host Controller components
  - Clock
    - CLK_HFx\(^1\)
    - CLK_GRx\(^2\)
    - CLK_SLOW

1 It is connected to CLK_HF6 in CYT4BF. For other devices, refer to the respective device datasheet.
2 It is connected to CLK_GR4 in CYT4BF. For other devices, refer to the respective device datasheet.
Clocks in SDHC Host Controller Block

The following clocks are used in the SDHC Host Controller block:

<table>
<thead>
<tr>
<th>Source</th>
<th>SDHC Host Controller Clock</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLK_SLOW</td>
<td>Core SDHC Clock</td>
<td>Used for core SDHC functions including the packet buffer SRAM; it is sourced from the slow clock (CLK_SLOW); it must be ≥ AHB slave clock.</td>
</tr>
<tr>
<td></td>
<td>AHB Master Interface Clock</td>
<td>Used by the AHB master interface; it is sourced from the slow clock (CLK_SLOW); it must be ≥ AHB slave clock.</td>
</tr>
<tr>
<td>CLK_GRx</td>
<td>AHB Slave Interface Clock</td>
<td>Used by the AHB slave interface; it is clocked by the PERI group clock (CLK_GRx); it must be ≥ CLK_CARD.</td>
</tr>
<tr>
<td>CLK_HFx</td>
<td>Base Clock / Card Clock</td>
<td>Used for sourcing the SD/eMMC interface clock (CLK_CARD); it is derived from CLK_HFx; it must be set to 100 MHz to be compatible with the Capabilities register¹.</td>
</tr>
<tr>
<td>Timer Clock</td>
<td></td>
<td>Used for command and data timeout functions; it is derived from CLK_HFx.</td>
</tr>
</tbody>
</table>

CLK_SLOW ≥ CLK_GRx ≥ CLK_CARD

¹ This register provides the Host Driver with information specific to the Host Controller.

Review TRM section 33.3 and Register TRM for additional details
Appendix
Unsupported Features

- SD/SDIO operation in UHS-II mode
- Command queuing engine (CQE)
- eMMC boot operation in dual data rate mode
- Read wait operation by DAT[2] signaling in an SDIO card
- Suspend/resume operation in an SDIO card
- Interrupt input pins for embedded SD systems
- SPI protocol mode of operation
- SD UHS-I mode using 1.8-V signal voltage: SDR, SDR25, SDR50, and DDR50
Part of your life. Part of tomorrow.
## Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>ECN</th>
<th>Submission Date</th>
<th>Description of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>6401030</td>
<td>12/04/2018</td>
<td>Initial release</td>
</tr>
<tr>
<td>*A</td>
<td>6606574</td>
<td>06/28/2019</td>
<td>Added note descriptions in all pages. Updated page 3, 5, 6, 8, 9, 11, 15 to 20, 21, 23, 24.</td>
</tr>
<tr>
<td>*B</td>
<td>7044761</td>
<td>10/29/2020</td>
<td>Updated page 2.</td>
</tr>
</tbody>
</table>