Semiconductors

# AN10029_1

## Odd or Even Byte Indicator in the ISP1161A1

July 2003

# Application Note
# Rev. 1.0

***Revision History:***

| Version | Date | Description | Author |
|---|---|---|---|
| 1.0 | May 2003 | First release. | Alvin Lim |

We welcome your feedback. Send it to wired.support@philips.com.

Philips Semiconductors - Asia Product Innovation Centre
Visit www.semiconductors.philips.com/buses/usb or www.flexiusb.com

PHILIPS

This is a legal agreement between you (either an individual or an entity) and Philips Semiconductors. By accepting this product, you indicate your agreement to the disclaimer specified as follows:

# DISCLAIMER

PRODUCT IS DEEMED ACCEPTED BY RECIPIENT. THE PRODUCT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, PHILIPS SEMICONDUCTORS FURTHER DISCLAIMS ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANT ABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT. THE ENTIRE RISK ARISING OUT OF THE USE OR PERFORMANCE OF THE PRODUCT AND DOCUMENTATION REMAINS WITH THE RECIPIENT. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL PHILIPS SEMICONDUCTORS OR ITS SUPPLIERS BE LIABLE FOR ANY CONSEQUENTIAL, INCIDENTAL, DIRECT, INDIRECT, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THIS AGREEMENT OR THE USE OF OR INABILITY TO USE THE PRODUCT, EVEN IF PHILIPS SEMICONDUCTORS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

# CONTENTS

# 1. Introduction

The odd or even byte indicator allows firmware to read the exact number of data bytes that are transferred across the direct memory access (DMA) transfer counter from the ISP1161A1 Device Controller (DC) to the external DMA bus. The odd or even byte indicator is valid only for the OUT token data.

The implementation will track whether an odd number has been transferred from the last OUT token packet to the DMA bus.

# 2. Location of the odd or even byte indicator

The odd or even byte indicator is implemented in bit 8 (ODD_EVEN_IND) of the DcDMAConfiguration register. The bit is read-only; see Table 2-1.

**Table 2-1: DcDMAConfiguration register: bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | CNTREN | SHORTP | reserved | reserved | reserved | reserved | reserved | ODD_ EVEN_IND |
| Reset | 0[1] | 0[1] | 0[1] | 0[1] | 0[1] | 0[1] | 0[1] | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | EPDIX[3:0] | | | | DMAEN | reserved | BURSTL[1:0] | |
| Reset | 0[1] | 0[1] | 0[1] | 0[1] | 0 | 0 | 0[1] | 0[1] |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] Unchanged by a bus reset.

# 3. Behavior of the ODD_EVEN_IND bit

The ODD_EVEN_IND bit is logic 0 when the last DMA access is a byte (LSB Byte valid, MSB Byte invalid.). This bit is logic 1 when the last DMA access is a word (LSB Byte valid and MSB Byte valid.). You can treat this bit as **MSB Byte valid** for the last DMA access of 16 bits.

**Table 3-1: Payload size for the ODD_EVEN_IND bit**

| ODD_EVEN_IND bit | Payload size |
|---|---|
| 1 | Even bytes |
| 0 | Odd bytes |

# 4. Using the odd or even byte indicator for DMA

The odd or even byte indicator must be used together with the DcDMACounter register during SP_EOT to determine the correct amount of data transferred during a DMA read (OUT token).

The following formula can be used to calculate the number of bytes transferred by DMA.

**Actual DMA transfer count** = Internal DMA count – Bit 8 of the DcDMAConfiguration register

**Number of bytes transferred by DMA** = Programmed transfer count - Actual DMA transfer count

Where,

**Internal DMA count** is the value in the DcDMACounter register at the end of DMA (SP_EOT or EOT).

**Bit 8 of the DcDMAConfiguration register** is the ODD_EVEN_IND bit.

**Programmed transfer count** is the 'Actual DMA transfer count' programmed by the programmer before starting DMA.

The result (**Number of bytes transferred by DMA**) will give the exact amount of data transferred by DMA (up to byte accuracy). You will be able to differentiate between whether the MSB Byte of the last word transferred by DMA is valid or not. If the result is odd, the MSB Byte is invalid; and if it is even, the MSB Byte is valid.

**Note**: The odd or even byte indicator is valid only for OUT endpoints.

## 4.1. Examples on the use of the odd or even byte indicator

### 4.1.1. Example 1

The host wishes to send **8 bytes** of data. The DC DMA counter is set to 8 bytes.

**Actual DMA transfer** = 1 – 1 (Even)

**Number of bytes transferred by DMA** = 8 – Actual DMA transfer

**Number of bytes transferred by DMA** = **8**

### 4.1.2. Example 2

The host wishes to send **7 bytes** of data. The DC DMA counter is set to 8 bytes.

**Actual DMA transfer** = 1 – 0 (Odd)

**Number of bytes transferred by DMA** = 8 – Actual DMA transfer

**Number of bytes transferred by DMA** = **7**

### 4.1.3. Example 3

The host wishes to send **34 bytes** of data. The DC DMA counter is set to 128 bytes.

**Actual DMA transfer** = 95 – 1 (Even)

**Number of bytes transferred by DMA** = 128 – Actual DMA transfer

**Number of bytes transferred by DMA** = **34**

### 4.1.4. Example 4

The host wishes to send **33 bytes** of data. The DC DMA counter is set to 128 bytes.

**Actual DMA transfer** = 95 – 0 (Odd)

**Number of bytes transferred by DMA** = 128 – Actual DMA transfer

**Number of bytes transferred by DMA** = **33**

# 5. Reading the DMA counter between transfers

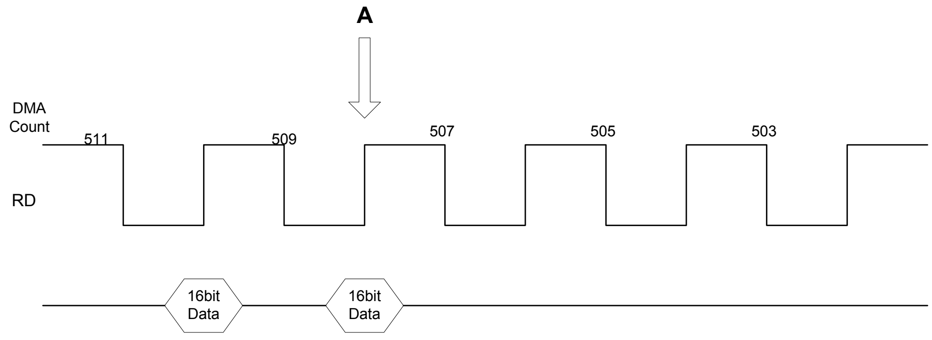
The odd or even byte indicator is designed to give the correct value at the SP_EOT condition. If you, however, decide to read the DMA counter in between DMA transfers, see Table 5-1.

**Table 5-1: Reading DMA counter between transfers**

| Device Controller DMA counter | USB packet | Formula[1] |
|---|---|---|
| Even counter value | Even | Use Formula |
| Even counter value | Odd | Use Formula + 1 |
| Odd counter value | Odd | Use Formula + 1 |
| Odd counter value | Even | Use Formula |

**[1] Formula: Actual transfer** = Programmed DC DMA counter – (read DC DMA counter – odd or even byte indicator)

## 5.1. Even counter examples




### 5.1.1. Example 1

Consider that:

- DC DMA counter is 512 bytes
- USB transfers 10 bytes (OUT token)
- Odd or even byte indicator is 1(even byte).

If the micro wishes to read the ISP1161A1 DC DMA counter at A, then according to Table 5-1:

**Actual transfer**: 512 – (509 – 1) = 4 bytes

### 5.1.2. Example 2

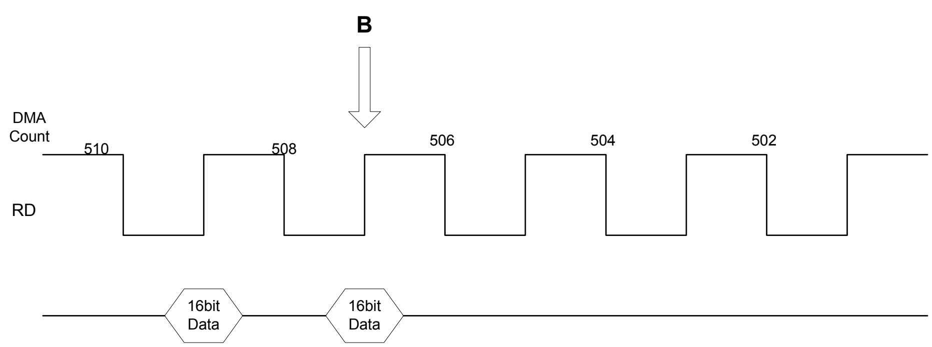
Consider that:

- DC DMA counter is 512 bytes
- USB transfers 9 bytes (OUT token)
- Odd or even byte indicator is 0 (odd byte).

If the micro wishes to read the ISP1161A1 DC DMA counter at A, then according to Table 5-1:

**Actual transfer**: 512 – (509 – 0) + 1 = 4 bytes

## 5.2. Odd counter examples




### 5.2.1. Example 1

Consider that:

- DC DMA counter is 511 bytes

- USB transfers 9 bytes (OUT token)
- Odd or even byte indicator is 0 (odd byte)

If the micro wishes to read the ISP1161A1 DC DMA counter at B, then as per Table 5-1:

**Actual Transfer**: 511 – (508 – 0) + 1 = 4 bytes

### 5.2.2. Example 2

Consider that:

- DC DMA counter is 511 bytes
- USB transfers 10 bytes (OUT token)
- Odd or even byte indicator is 1 (even byte).

If the micro wishes to read the ISP1161A1 DCDMA counter at B, then according to Table 5-1:

**Actual Transfer**: 511 – (508 – 1) = 4 bytes

# 6. Sending a zero-length packet

In the case of a zero-length packet, the Device Controller will generate an EOT interrupt.

**Note**: The interrupt is not SP_EOT.

## 6.1. Behavior of DREQ

If the ISP1161A1 Device Controller receives a zero-length packet, DMA will be terminated and the EOT interrupt will be asserted.

For example, consider that the Device Controller DMA counter is 512 bytes. The ISP1161A1 Device Controller will receive three packets of 64 bytes each, followed by a zero-length packet for termination. On receiving the zero-length packet, DREQ will not be asserted because DREQ assertion is based on whether the OUT buffer has data or not. Since it is a zero packet, it implies that it does not have data.

## 6.2. Behavior of the DMA counter

Consider a situation similar to that given in Section 6.1. If the ISP1161A1 Device Controller receives three full packets followed by an empty packet, and then an EOT Interrupt, it will result in correct DMA counter reading.

For example, consider that the DC DMA counter is 512 bytes. If the ISP1161A1 Device Controller receives three packets of 64 bytes each, followed by a zero-length packet for termination, reading the Device Controller DMA Counter will provide a value of 320 bytes (512 – 192 = 320). Therefore, the SP_EOT formula (Section 5) need not be applied.

## 6.3. Behavior of the ODD_EVEN_IND bit

On receiving the zero-length packet, the ODD_EVEN_IND bit will reflect as even.

# 7. References


- *Universal Serial Bus Specification Rev. 2.0*
- *ISP1161A1 Full-speed Universal Serial Bus single-chip host and device controller* datasheet.

# Philips Semiconductors

Philips Semiconductors is a worldwide company with over 100 sales offices in more than 50 countries. For a complete up-to-date list of our sales offices please e-mail
sales.addresses@www.semiconductors.philips.com.
A complete list will be sent to you automatically.
You can also visit our website
http://www.semiconductors.philips.com/sales/

**www.semiconductors.philips.com**