Microcontroller Technical Information

<table>
<thead>
<tr>
<th>32-Bit Microcontrollers V850E2/ME3</th>
<th>Document No.</th>
<th>ZBG-CC-06-0039</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage Restrictions</td>
<td>Date issued</td>
<td>November 2, 2006</td>
</tr>
<tr>
<td>Issued by</td>
<td>2nd Solution Development Group</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Multipurpose Microcomputer Systems Division</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4th Systems Operations Unit</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NEC Electronics Corporation</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Notification classification</td>
<td>✓ Usage restriction</td>
<td>Upgrade</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Document modification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Other notification</td>
</tr>
</tbody>
</table>

1. Affected products

V850E2/ME3
- µPD703500

2. New restrictions

The following restrictions (No. 1 and No. 2) have been newly found. For details, refer to the attachment.

No. 1 Restriction of DMA transfer by an internal USB interrupt

DMA transfer may not be correctly executed by a USB interrupt.

No. 2 USB-related registers cannot be accessed unless an external clock is supplied to the UCLK pin (the CPU must be reset because it keeps waiting for a response of a USB register that is not operating).

3. Workarounds

- **Workaround for restriction No. 1**

  When starting DMA by a USB interrupt, set the DTFRn register to 0x7EH and execute DMA transfer in the single-step transfer mode (setting of 0x7EH is prohibited in the User’s Manual, but it was confirmed that the operation is performed without problem even when the register is set to 0x7EH instead of 0x7FH).

- **Workaround for restriction No. 2**

  If the clock is not supplied to UCLK, set a register (to be specified) that supplies the internal clock as the USB clock.

4. Action

The above restrictions No. 1 and No. 2 will be added to usage restrictions. We apologize for the inconvenience.
5. List of restrictions

The restriction history and detailed information is described in the attachment.

6. Document revision history

<table>
<thead>
<tr>
<th>Document Number</th>
<th>Date Issued</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZBG-CC-06-0039</td>
<td>November 2, 2006</td>
<td>Addition of new bug (No. 1 and 2)</td>
</tr>
</tbody>
</table>
List of Usage Restrictions in V850E2/ME3

1. Product Version

μPD703500: Rank K, E

* The rank is indicated by the letter appearing as the 5th digit from the left in the lot number marked on each product.

2. Product History

• μPD703500

<table>
<thead>
<tr>
<th>No.</th>
<th>Bugs</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Restriction of DMA transfer by an internal USB interrupt</td>
<td>Δ</td>
</tr>
<tr>
<td>2</td>
<td>Restriction of USB register access when USB clock is not connected</td>
<td>Δ</td>
</tr>
</tbody>
</table>

√: Bug does not occur, Δ: Bug will also apply in future, ×: Bug occurs

3. Restriction Details

No. 1 Restriction of DMA transfer by an internal USB interrupt

[Description]

DMA transfer may not be correctly executed by a USB interrupt because of the combination of the following problems <1> and <2>.

Problem <1>

If a USB interrupt is specified (if the DTFRn register is set to 0x7FH) as a transfer trigger for two or more DMA channels, the USB interrupt source set for the DMA channel of the highest priority is transferred to the other DMA channel in the CPU, and all the DMA requests that set the DTFRn register to 0x7FH are generated.

Problem <2>

If a DMA request for USB is generated in the CPU while DMA transfer for internal peripheral I/O is executed, the CPU cannot correctly control the DMA request and acknowledge the request; consequently, DMA transfer is not executed in some cases.

[Workaround]

When starting DMA by a USB interrupt, set the DTFRn register to 0x7EH and execute DMA transfer in the single-step transfer mode. This can avoid both of the above problems <1> and <2> (a flowchart is shown on the next page). The User’s Manual prohibits setting of 0x7EH, but it was confirmed that the operation is performed without problem even when the register is set to 0x7EH instead of 0x7FH.
DMA transfer EP2 → dRAM (DMA: ch2)/dRAM → EP1 (DMA: ch3) (1/2)

<Classification of operation>

START Hardware operation
↓ Software operation

* Set single-step mode to Ch2 and Ch3.

DMA Ch2 setting
POWER(DC)=0x01
DSMC=0x0C
DDA2=dRAM (address)
DSA2=UTF0BO1(address)
DADC2=0xE800

DMA Ch3 setting
DDA3=UTF0B11 (address)
DSA3=dRAM (address)
DBC3L=0x3F
DADC3=0xB200

USB setting (DMA-related)
UF0CS=0x45XX
DMAEDM=UTF0IM0=0
BK11NLMD=UTF0IM2=0
BK01NLMD=UTF0IM3=0

* Cancel the mask setting for necessary interrupts.

DMA Ch2 setting
DTFR2=0x7E

DMA Ch3 setting
EN3(DCHC3)=0x01
DTFR3=0x7E
DQBI1MS=UTF0IDR=1

<1>
DMA ch3 transfer completed
TC3(DCHC3)=1
DQBI1MS=UTF0IDR=0

Yes
No

Moves to INTDMA3 interrupt vector

DMA ch3 request clear
DTFR3=0x7E

Clears DMARQ signal by setting FCLR3 bit.

DMA ch3 request clear
TC3(DCHC3)=0
FCLR3(DCHC3)=1

Number of DMA ch3 transfers changed?

<2>
Yes
No

Change DBC3L set value

 DMA Ch3 setting
EN3(DCHC3)=1
DQBI1MS=UTF0IDR=1

Sets DMA ch3 again.
DMA transfer EP2 → dRAM (DMA: ch2)/dRAM → EP1 (DMA: ch3) (2/2)

- DMA ch2 Transfer end?
  - Yes: DMA ch2 transfer completed
    - TC2(DCHC2)=1
    - DQBO1MS(UF0IDR)=0
    - Moves to INTDMA3 interrupt vector
  - No: DMA ch2 request clear
    - DTFR2=0x7E
    - DMA ch2 request clear
      - TC2(DCHC2)=0
      - FCLR2(DCHC2)=1
      - <1>
      - Moves to INTUSB1B interrupt vector
    - Interrupt occurred during EP2 data
      - Confirm data reception.
      - BKO1DT=1?
        - Yes: Cancels transfer operation if a NULL package is received.
        - No: BKO1NL=1?
          - Yes: Starts DMA transfer after setting number of DMA transfers.
          - No: Reads UF0BO1L register
            - Yes: Starts DMA ch2 transfer
            - No: DMA Ch2 setting
              - DBC2L=UF0BO1L-1
              - TC2(DCHC2)=1
              - DQBO1MS(UF0IDR)=1
              - Starts DMA ch2 transfer

- INTUSB1B interrupt occurred?
  - Yes: Moves to INTUSB1B interrupt vector
    - Reads UF0IS3 register
      - BKO1DT=1?
        - Yes: Cancels transfer operation if a NULL package is received.
        - No: BKO1NL=1?
          - Yes: Starts DMA transfer after setting number of DMA transfers.
          - No: Reads UF0BO1L register
            - Yes: Starts DMA ch2 transfer
            - No: DMA Ch2 setting
              - DBC2L=UF0BO1L-1
              - TC2(DCHC2)=1
              - DQBO1MS(UF0IDR)=1
              - Starts DMA ch2 transfer

- Interrupt occurred during EP2 data
  - Confirm data reception.
  - BKO1DT=1?
    - Yes: Cancels transfer operation if a NULL package is received.
    - No: BKO1NL=1?
      - Yes: Starts DMA transfer after setting number of DMA transfers.
      - No: Reads UF0BO1L register
        - Yes: Starts DMA ch2 transfer
        - No: DMA Ch2 setting
          - DBC2L=UF0BO1L-1
          - TC2(DCHC2)=1
          - DQBO1MS(UF0IDR)=1
          - Starts DMA ch2 transfer
No. 2  Restriction of USB register access when USB clock is not connected

[Description]
When an external clock is not supplied to the UCLK pin*, the registers related to USB cannot be accessed (the CPU must be reset because it keeps waiting for a response of a USB register that is not operating).

* The external clock is not supplied to the UCLK pin in either of the following cases.
  - The external clock is not connected.
  - Before the PFC10 bit of the PFC1 register is switched to the UCLK input mode with the external clock connected

[Workaround]
The USB macro of the above product is of a configuration always requiring clock supply. As a register that supplies the internal clock of ME3 to the USB macro, therefore, the USB clock control register (UCKC) is stipulated. Set this register in the following procedure, depending on how the USB function is used.

[USB clock control register (UCKC)]
This register selects the USB clock source. When USB is not used, set the UCDIV1 bit to 1 to supply the internal clock. This register can be read or written in 8- or 1-bit units. To write this register, be sure to clear bits 0 and 2 to 7 to "0". The operation cannot be guaranteed if a value other than 0 is set to these bits.

**Caution** Be sure to supply the external clock when using USB. Do not use the USB function while the internal clock is supplied.
• When USB is not used
  Set the UCKC register in the following procedure in the initialization routine.

  <1> UCKC.UCDIV1 = 1; Sets USB clock as internal clock.
  <2> After that, execute other initialization processing.

• When USB is used
  The above register does not have to be set.
  As described in the User’s Manual, however, set the PFC1 and PMC1 registers using the following procedure to enable supply of the external clock, and then use the USB function (including accesses to registers related to USB).

  <1> PM1.PM10 = 1; Sets P10 in the input mode
  <2> PFC1.PFC10 = 1; Sets the UCLK pin in the input mode.
  <3> PMC1.PMC10 = 1; Specifies the alternate function of the P10 pin as the operation mode.
  <4> After that, execute other initialization processing.

  Note  The PM1 register does not have to be set after reset, because its default value is FFH.