Mask Set Errata for Mask 0N91Z

This report applies to mask 0N91Z for these products:
- MC56F81xxx

### Table 1. Errata and Information Summary

<table>
<thead>
<tr>
<th>Erratum ID</th>
<th>Erratum Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR050631</td>
<td>QDC: A possible speed measurement issue when CTRL3[PMEN]=1</td>
</tr>
<tr>
<td>ERR050527</td>
<td>ROM: After ROM execution and jumping to application, ROM does not restore PIT0 registers to reset values</td>
</tr>
</tbody>
</table>

### Table 2. Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initial revision</td>
</tr>
</tbody>
</table>

**ERR050631: QDC: A possible speed measurement issue when CTRL3[PMEN]=1**

**Description:** When CTRL3[PMEN]=1, and reading POSD occurs simultaneously with any edge of phase A or phase B signal, the captured position difference value in POSDH register may not match the time period captured in POSDPERH register, which causes inaccuracy in speed measurement.

**Workaround:** This issue is caused by the fact that the edge of phase A and phase B are delayed 3 QDC clock cycles and then they are used to drive position counter. When reading POSD occurs simultaneously with the edge of phase A or phase B, the captured time period in POSDPERH is correct, but position counter is not increased or decreased yet, so the captured position difference value in POSDH is one count less or more than the correct value depending on the direction. To avoid such an issue, follow the steps below to get a correct position difference value:
1. Read POSD register to trigger capture of position difference and time period between each reading of POSD. Now the position difference is captured into POSDH (inaccurate value), and time period is captured into POSDPERH (accurate value).

2. Read LPOS register after 3 QDC clock cycles. Make sure it is exactly 3 QDC clock cycles between reading POSD in step 1 and reading LPOS in this step.

3. Store the value of LPOSH and UPOSH into a 32bit variable, such as u32PosCnt. Another variable u32PosCnt_1 is used to store LPOSH and UPOSH value of last time.

4. Define variable u32PosDiff = u32PosCnt - u32PosCnt_1. Now u32PosDiff represents the accurate position difference. Use u32PosDiff and the value in POSDPERH to calculate speed.

5. Update u32PosCnt_1 with the value of u32PosCnt.

Note: repeat these steps each time the speed is calculated.

ERR050527: ROM: After ROM execution and jumping to application, ROM does not restore PIT0 registers to reset values

Description: When jumping from ROM to application, it is expected to restore all registers (used by ROM) to reset values. However, PIT0 registers PIT0_CTRL, PIT0_CNTR_L and PIT0_CNTR_H are not restored to their reset values.

Workaround: Set 0x0000 to PIT0_CTRL at the beginning of user’s application. Since PIT0_CTRL[CNT_EN] bit is cleared, the counter registers PIT0_CNTR_L and PIT0_CNTR_H returns to 0x0000 value automatically.