Concerned Products:	**Customer Notification**	Date: 16 March 2000
		NEC-Electronics (Europe) GmbH EAD -Technical Product Support
µPD784224 ***µPD784224Y*** ***µPD784225*** ***µPD784225Y*** ***µPD78F4225*** ***µPD78F4225Y***	**Bug Report**	Source Doc: SBG-T-0483 SBG-T-0504 SIF-T-?? (TQ-TH970820/1) SIF-T-10395 (TQ-TH971215/2) SBG-T-0619 NN
		Author: A. Twardy
Date of initial issue: 07 August 1998		Doc. No.: TPS-LE-B-4223

(A) BUG LIST

Bug No.	Outline		μPD784224 μPD784224Y				μPD784225 Note3 μPD784225Y				μPD78F4225 μPD78F4225Y						
		Rev.									DS1.0	ES1.0	ES1.2	ES2.0	CS1.0	CS3.2	
		Standard									I	I	K	K	K	E	
1	IDLE mode release						✋	✓			✋	✋	✓	✓	✓	✓	
2	ADC initial conversion result						💣	💣			💣	💣	💣	💣	💣	💣	
3	ADC conversion result after stop						💣	💣			💣	💣	💣	💣	💣	💣	
4	I2C restart procedure Note1						✋	✓			✋	✋	✓	✓	✓	✓	
5	Low power consumption mode						✓	✓			✋	✋	✓	✋	✓	✓	
6	TIMER compare register access						✋	✓			✋	✋	✓ Note4	✋	✓ Note4	✓ Note4	
7	UART2 clock sources						✋	✓			✋	✋	✓	✋ Note2	✓	✓	
8	Restrictions						✓	✓			✋	✋	✋	✋	✓ Note5	✓ Note5	

Explanation:

✓: No problem, ✋: Bug (will be corrected by version upgrade), 💣: Bug (restriction, not corrected by version upgrade)

Note1: Applicable for μPD784224Y, μPD784225Y, μPD78F4225Y subseries only.

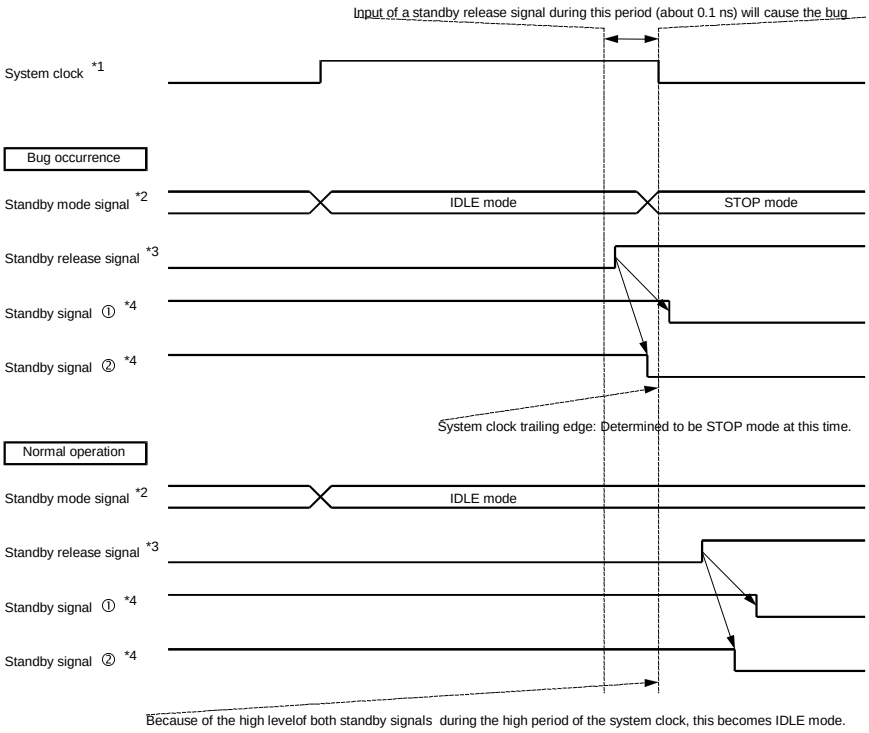
Note2: Under investigation.

Note3: The Mask-Level of the devices will be given in the next version of the Bug Report. In case of unclearness please ask NEC-EE EAD-TPS.

Note4: The upper and lower compare registers have always to be set after stopping in cascade connection mode.

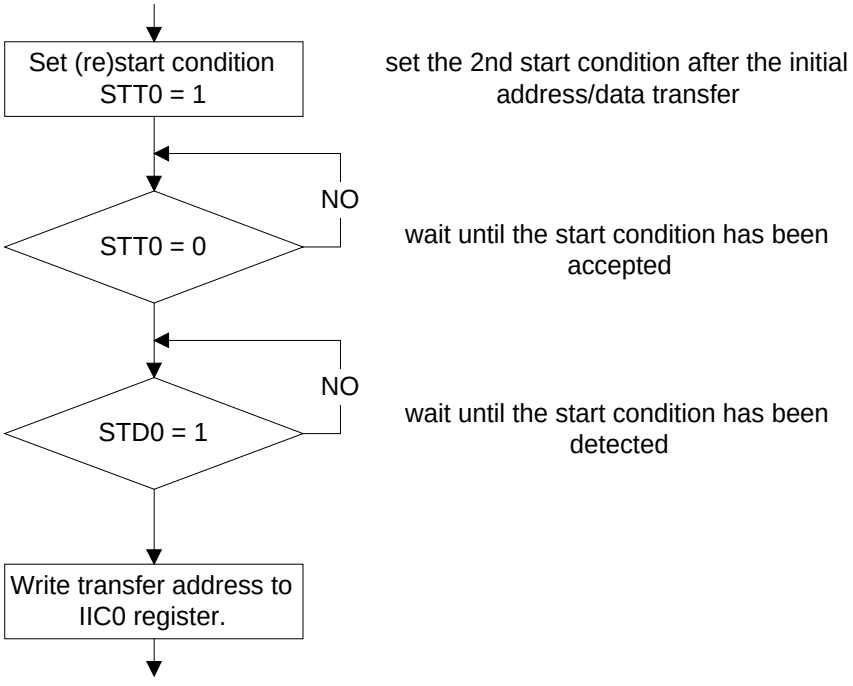
Note5: Please refer always to the actual Data-Sheet and the restrictions listed under point 8.

(B) BUG DESCRIPTION

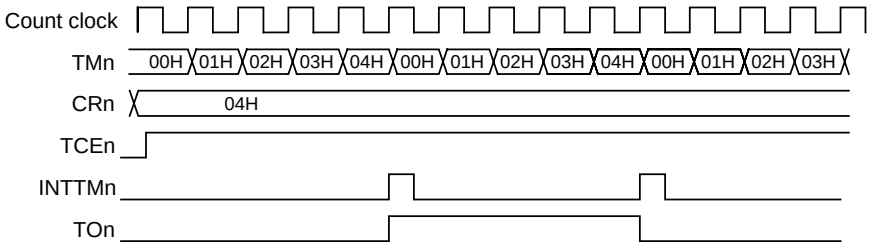
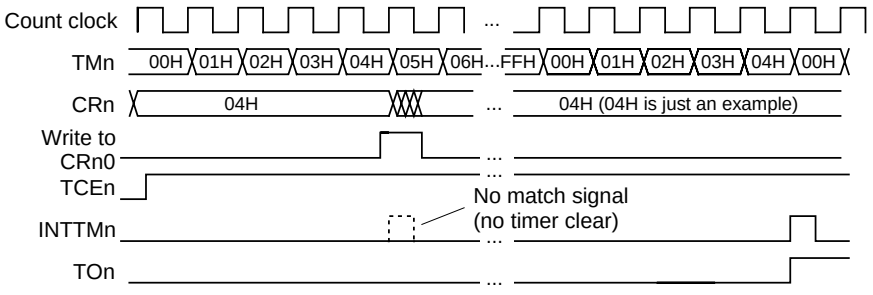
Bug No.	Outline	Description
1	Cancelling the IDLE mode may cause a wait during the oscillation stabilization time that has been set by oscillation stabilization time specification register (OSTS).	*Details* When a contention occurs between the execution of an IDLE mode instruction and an interrupt to release the IDLE mode, STOP mode is released after executing STOP mode, rather than IDLE mode being released after executing IDLE mode as it should. Therefore, despite the fact that the program has set the IDLE mode, this bug results in a wait for the length of oscillation stabilization time that is set by the oscillation stabilization time register (OSTS), when cancelling this mode. Normally, when cancelling the IDLE mode, there is no need to wait for the duration of the oscillation stabilization time. Specifically, when the bug occurs, an operation as shown in the following timing chart is performed.  *1: Internal system clock *2: The signal notifying entry to entering each standby mode, While the system clock is high, the values of standby signals ① and ② are latched, and signals to select which standby mode to enter are created. *3: The standby mode release signal created by the microprocessor by inputting the standby mode release factor. *4: STOP mode: Signal ① = 1, Signal ② = 0 HALT mode: Signal ① = 0, Signal ② = 1 IDLE mode: Signal ① = 1, Signal ② = 1

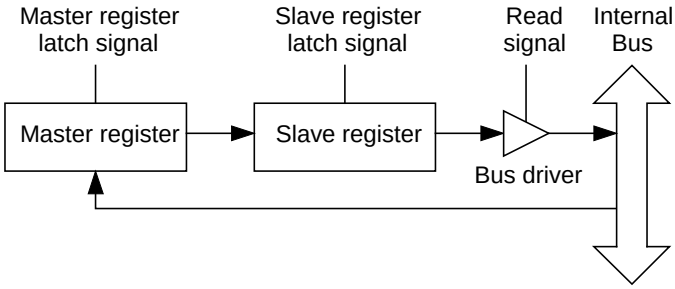
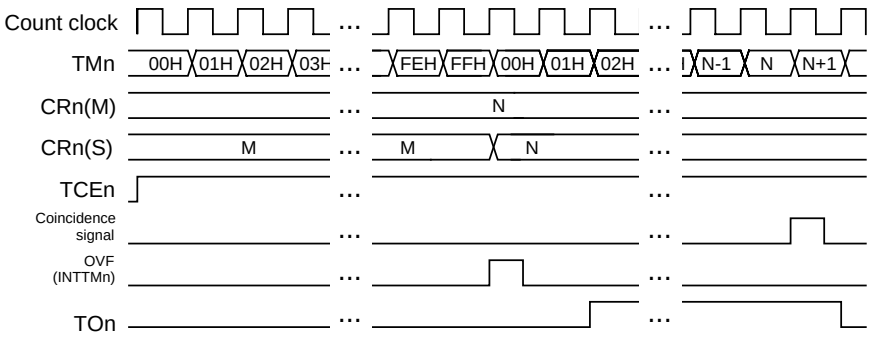
Bug No.	Outline	Description
1	(continuation)	*Description of timing chart* First, standby signal ① and ② is changed by executing the IDLE instruction. Then, while the system clock has high level, the values of standby signals are latched. Selection of the standby mode to enter is set by falling edge of the system clock. The standby mode signal is placed in IDLE mode if the standby signals ① and ② are high level respectively while the system clock is high. The bug is generated when the standby mode release interrupt occurs after the standby mode instructions (MOV STBC, #byte) and immediately before the falling edge of the initial system clock. Because of the time difference generated after the standby release interrupt until the signals ① and ② are cleared, the system is placed in the STOP mode signal status when the system clock falls, which is neither IDLE mode nor normal mode. Consequently, the system is determined to be in the STOP mode, thus resulting in the wait for the duration of oscillation stabilization time upon cancellation of the standby. *Workaround* Unfortunately, it is impossible to take perfect countermeasures in software. As it was explained above, in order to stop this bug from occurring, it is necessary to make sure that the external interrupts and input to the IDLE instruction do not conflict with each other. So it is recommended, when using the IDLE mode, to **shorten the value of the oscillation stabilization time**, which is set by the OSTS register. This bug doesn't occur unless an external interrupt is generated during the short time of about 0.1 ns when executing the IDLE instruction. Therefore, the rate of occurrence is considered extremely low.

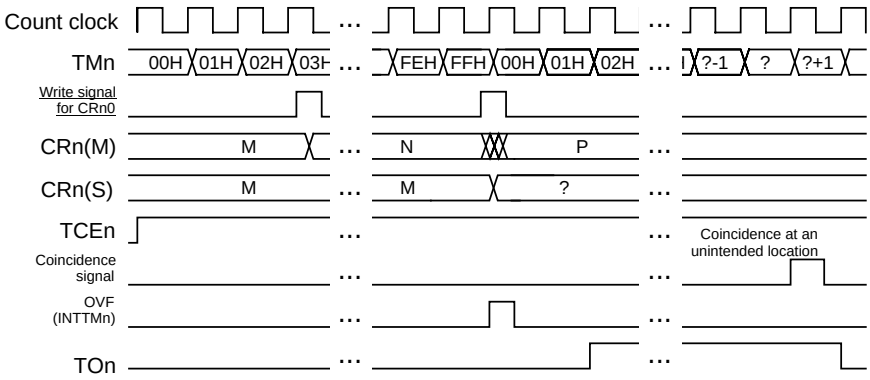
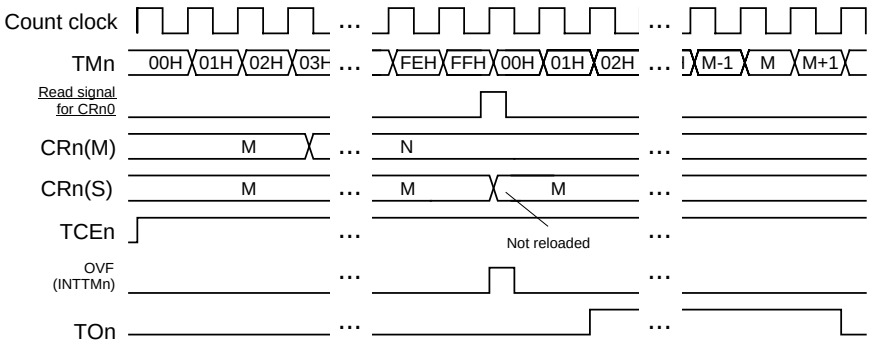
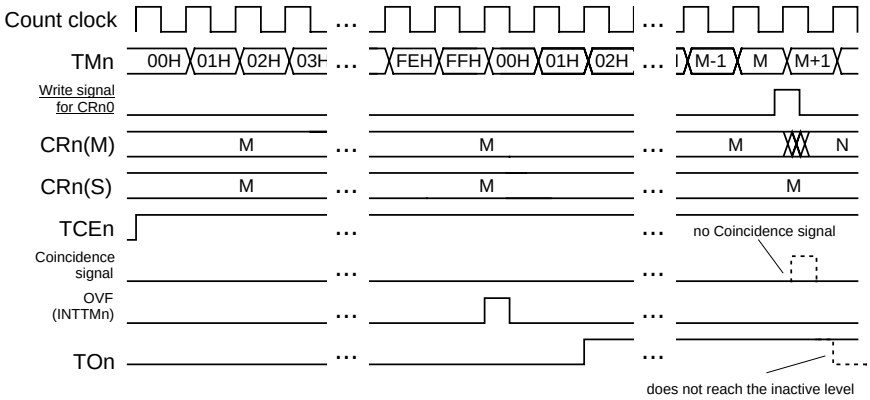
Bug No.	Outline	Description
2	The initial conversion result immediately after the A/D conversion operation startup may not satisfy the specification.	*Details* Immediately after the A/D conversion is started, the reference voltage of the A/D converter is unstable. The normal operation of the sampling circuit being controlled is affected by this voltage, thus increasing the conversion error. Because the conversion error at this time cannot be confirmed due to differences in conversion speed and uneven product quality, etc., the conversion result becomes an undefined value. *Workaround* Do not use the initial result (contents of ADCR register) immediately after the A/D converter is started. Instead, use the second or later conversion results subsequently generated (ref. to the diagram below). When the A/D conversion end interrupt (INTAD) of the initial conversion occurs, either accept it or clear the appropriated interrupt request flag ADIF.
3	If the conversion result is read after halting the A/D conversion operation, the value may be undefined.	*Details* If the timing for the A/D conversion end conflicts with that for halting the A/D conversion operation, an undefined value is stored into the ADCR. Reading the conversion result afterwards will result in reading an undefined value. *Workaround* Read the A/D conversion result register ADCR while the A/D converter is operating. When reading the conversion result after halting the A/D conversion, make sure that timing for halting A/D conversion does not conflict with that for the end of the A/D conversion.

Bug No.	Outline	Description
4	Using the I^2C restart condition causes a faulty address transfer.	*Details* If the transfer address is set to the shift register (IIC0) immediately after the (re)start is set (STT = 1), the transfer address will be sent simultaneously by rising edge of the serial clock (SCL). The serial clock is at low level normally before a restart condition will be set, due to the preceded address/data transfer. To generate a (re)start condition the serial clock (SCL) has to be first on high level (recessive state). By falling edge of the SDA line, when SCL has high level, the (re)start condition can be detected. Afterwards the SCL line will fall to low level and the slave address can be transferred by normal operation. Now, if the slave address is written into the shift register (IIC0) immediately after the start condition trigger (STT0) is set, the rising edge on the SCL line resulting in data output before the start condition is set. This causes the bug. *Workaround* The transfer address must be set after the confirmation of start condition detection (STT=0, STD = 1). 

Bug No.	Outline	Description
5	The low power consumption mode, where the device operates by sub-system clock, cannot be used.	*Details* Only flash devices are concerned by this bug. The problem is caused by the integrated charging pump, which is needed for reading the flash memory. At higher temperature a malfunction occurs, and the CPU cannot fetch the program code from the flash memory correctly. Therefore the program execution becomes undefined. *Workaround* No. Do not use the low power consumption mode for flash devices. Just the operation at room temperature (about 25°C) is permitted, in order to carry out an evaluation for revised devices.

Bug No.	Outline	Description
6	Timer compare register access while corresponding timer register matches	*Details* The Bugs described below concern all 8-Bit Timers of the µPD784216/4216Y subseries. In detail the timers concerned are Timer 1, 2, 5, 6, 7 and Timer 8. 6.1 Interval timer mode Writing to a compare register while the corresponding timer register matches does not generate an interrupt request. Under conditions like above the timerregister is not cleared in the “Timer Clear & Start” mode. 6.2 Square wave output mode Writing to a compare register while the corresponding timer register matches does not invert the output signal of the timer output. Please refer to the timing diagrams shown below. Please note, however, that the timing charts below are schematic representations of operations and therefore do not indicate a precise operation timing. *Normal operation*  *Bug occurrence (6.1 and 6.2)* 

Bug No.	Outline	Description
6	Timer compare register access while corresponding timer register matches (continuation)	6.3 PWM output mode The compare registers of the concerned timers consist of two buffers the master and the slave register. The simplified diagram is show below.  In PWM mode the data transfer from the master to the slave register is performed when an overflow occurs. This bug occurs when performing a data transfer from the concerned master register to the slave register when an overflow occurs in an attempt to rewrite the value of the master register, thus damaging the data to be transferred. The description of the bugs are in detail as following: 6.3.1 Writing to a compare register while the corresponding timer register matches does not change the output signal of the timer to the inactive level. 6.3.2 Writing to CRn0 (master register) once and then writing again when the corresponding timer register (TMn) overflows causes an undefined value to be transferred to CRn0 (slave register). 6.3.3 Reading the compare register when an overflow of the corresponding timer occurs stops the transfer of data form CRn0 (master register) to CRn0 (slave register). The transfer is performed when overflow of the timer occurs. Please refer to the timing diagrams shown below. Please note, however, that the timing charts below are schematic representations of operations and therefore do not indicate a precise operation timing. *Normal operation as PWM timer* 

Bug No.	Outline	Description
6	Timer compare register access while corresponding timer register matches (continuation)	*Bug occurrence in PWM timer mode (6.3.1)*  *Bug occurrence in PWM timer mode (6.3.2)*  *Bug occurrence in PWM timer mode (6.3.3)* 

Bug No.	Outline	Description
6	Timer compare register access while corresponding timer register matches (continuation)	*Workaround* In general the countermeasure is to avoid writing to the compare register by coincedence between timer register and compare register. This can be realised by using the interrupt processing (INTTMn) of each timer. In the cases mentioned under 6.1 and 6.2 Use the interrupt which is generated by coincedence between the timer register and the compare register. This interrupt should be used to write to the compare register before the next coincedence occurs. In the cases mentioned under 6.3.1, 6.3.2 and 6.3.3 Use the interrupt which is generated when the overflow of the timer occurs. In this interrupt process avoid the correspondences between the timer register, the compare register and the next timeroverflow. Another countermeasure for all bugs mentioned under 6, is to stop the timer operation once before resuming it.

Bug No.	Outline	Description
7	UART2 clock sources	*Details* The UART2 can not work with the internal Timer1 as clock source. *Workaround* No. Do not use the timer1 as clock source for the UART2.
8	Restrictions	*Details* Only the flash devices are concerned by these restrictions. 8.1 The flash ROM of the devices has to be programmed one time in a row. Use PREWRITE with 00 (HEX) before erasing. Single Block operation is not guaranteed. 8.2 Samples DS1.0, ES1.0 and ES2.0 Operating temperature: -10 to +60 deg C Programming temperature: +10 to +40 deg C Operating voltage: 2.7V to 5.5V Programming voltage: 9.7V - 10.3V Number of W/E cycles: no guarantee Data retention: no guarantee 24 hours operating: no guarantee programmer: FlashMaster V1.12 or higher write-back function: no needed 8.3 Samples ES1.2 Operating temperature: -40 to +85 deg C Programming temperature: +10 to +40 deg C Operating voltage: 1.9V to 5.5V Programming voltage: 9.7V - 10.3V Number of W/E cycles: no guarantee Data retention: no guarantee 24 hours operating: no guarantee programmer: FlashMaster V2.00 or higher write-back function: no needed 8.4 Samples CS1.0 Operating temperature: -40 to +85 deg C Programming temperature: +10 to +40 deg C Storage temperature: -40 to +125 deg C Operating voltage: 1.9V to 5.5V Programming voltage: 9.7V - 10.3V Number of W/E cycles: 5 Data retention: 10 years 24 hours operating: no guarantee programmer: FlashMaster V2.00 or higher write-back function: not needed

Bug No.	Outline	Description
8	Restrictions	*Details* 8.4 Samples CS3.2 Operating temperature: -40 to +85 deg C Programming temperature: +10 to +40 deg C Storage temperature: -40 to +125 deg C Operating voltage: 1.9V to 5.5V Programming voltage: 9.7V - 10.3V Number of W/E cycles: 20 Data retention: 10 years 24 hours operating: guaranteed programmer: FlashMaster V3.00 or higher write-back function: needed internal verify: needed *Workaround* No.