

# ERRATA SHEET

**Date:** 2009 May 8  
**Document Release:** Version 1.7  
**Device Affected:** LPC2468

This errata sheet describes both the known functional problems and any deviations from the electrical specifications known at the release date of this document.

Each deviation is assigned a number and its history is tracked in a table at the end of the document.

2009 May 8

**Document revision history**

Rev	Date	Description
1.7	May 8 2009	Added Rev D
1.6	June 2 2008	Added Errata Note 1
1.5	April 8 2008	Vbat.1 was added
1.4	February 12 2008	1. USB.4 was added
1.3	September 21 2007	1. Added Rev B 2. Removed Rev A from ESD.1. ESD.1 does not appear in Rev A. It was accidentally listed in version 1.2
1.2	July 20 2007	1. Ethernet.3 was added 2. Ethernet.1 was updated 3. Flash.1 was updated
1.1	June 7 2007	1. Added Rev A 2. Added Flash.1, MAM.1 and CAN.1
1.0	March 21 2007	First version

## Identification

The typical LPC2468 devices have the following top-side marking:

LPC2468xxx

xxxxxxx

xxYYWW R[x]

The last/second to last letter in the third line (field 'R') will identify the device revision. This Errata Sheet covers the following revisions of the LPC2468:

Revision Identifier (R)	Comment
'-'	Initial device revision
'A'	Second device revision
'B'	Third device revision
'D'	Fourth device revision

Field 'YY' states the year the device was manufactured. Field 'WW' states the week the device was manufactured during that year.

**Errata Overview - Functional Problems**

Functional Problem	Short Description	Device Revision the problem occurs in
ADC.1	ADDRx read conflicts with hardware setting of DONE bit	-
Ethernet.1	Setting up the Ethernet interface in RMII mode	-
Ethernet.2	Ethernet SRAM disabled	-
Ethernet.3	RxDescriptor number cannot be greater than 4	-
I <sup>2</sup> S.1	I <sup>2</sup> S DMA can stall	-
PLL.1	PLL output is limited to 290 MHz	-
SRAM.1	16 kB SRAM can not be used for code execution	-
USB.1	USB_NEED_CLK is always asserted	-
USB.2	U1CONNECT is not functional	-
USB.3	V <sub>BUS</sub> status input is not functional	-
USB.4	USB_PWRDx pin(s) does not function as intended	-
WDT.1	Accessing non-Watchdog APB registers in the middle of the feed sequence causes a reset	-
Core.1	Incorrect update of the Abort Link register in Thumb state	-, A, B, D
Flash.1	Operating speed out of on-chip flash is restricted	-, A
MAM.1	Code execution failure can occur with MAM Mode 2	-, A
CAN.1	Data overrun condition can lock the CAN controller	-, A, B
Vbat.1	Increased power consumption on Vbat when Vbat is powered before the 3.3 V supply used by rest of device	-, A, B

**Errata Overview - AC/DC Deviations**

AC/DC Deviation	Short Description	Device Revision the deviation occurs in
ESD.1	2kV ESD requirements are not met on the RTCX1 pin	-

### Errata Notes

Notes	Short Description	Device Revision the note applies to
Note 1	When the input voltage is $V_i \geq V_{dd} I/O + 0.5\text{ v}$ on each of the following port pins P0.23, P0.24, P0.25, P0.26, P1.30, P1.31, P0.12, and P0.13 (configured as general purpose input pin (s)), current must be limited to less than 4 mA by using a series limiting resistor.	-, A, B, D

### Functional Problems of LPC2468

#### ADC.1: ADDR<sub>x</sub> read conflicts with hardware setting of the DONE bit

**Introduction:** The LPC2468 has a 10-bit ADC, which can be used to measure analog signals and convert the signals into a 10-bit digital result. There are eight A/D channels and each channel has its own individual A/D Data Register (ADDR0 to ADDR7). The A/D Data Register holds the result when an A/D conversion is complete, and also includes the flags that indicate when a conversion has been completed (DONE bit) and when a conversion overrun has occurred. The DONE bit is cleared when the respective A/D Data Register is read.

**Problem:** If a software read of ADDR<sub>x</sub> conflicts with the hardware setting of the DONE bit in the same register (once a conversion is completed) then the DONE bit gets cleared automatically, thereby clearing the indication that a conversion was completed.

**Workarounds:** For software controlled mode or burst mode with only one channel selected, the DONE bit in the A/D Global Data Register (located at 0xE003 4004) can be used instead of the individual ADDR<sub>x</sub> result register with no impact on performance.

For burst mode with multiple channels selected, the DONE bit together with the CHN field in the A/D Global Data Register can be used with some impact on throughput.

#### Ethernet.1: Setting up the Ethernet interface in RMII mode

**Introduction:** The LPC2468 has an Ethernet interface, which can be interfaced with an off-chip PHY using the RMII interface.

**Problem:** The default configuration of the device does not enable the RMII interface.

**Workaround:** To use the Ethernet interface in RMII mode write a 1 to bit 12 (P1.16) in PINSEL2 register (located at 0xE002 C008). This workaround only applies for Rev '- ' devices and does not apply for Rev 'A' and newer devices. In order to have both Rev '- ' and other revisions coexist in the same piece of software, the MAC module ID can be used to identify the part and determine if port pin P1.6 needs to be set or not.

Here are the steps (along with some sample code) to initialize the MAC based on the module ID:

1. In master header file ILPC24xx.h, make sure Module ID is defined (Please note, this ID register is not documented in the User's Manual).

```
#define MAC_BASE_ADDR          0xFFE00000
#define MAC_MODULEID (*(volatile unsigned long *) (MAC_BASE_ADDR +
0xFFC)) /* Module ID reg (RO) */
```

2. In the beginning of the MAC initialization file, add below definition:

```
#define OLD_EMAC_MODULE_ID    ((0x3902 << 16) | 0x2000)
```

3. In MAC initialization routine, right after setting the EMAC clock in the PCONP register, add a few lines as below:

```
/* Turn on the ethernet MAC clock in PCONP, bit 30 */
regVal = PCONP;
regVal |= PCONP_EMAC_CLOCK;
PCONP = regVal;

/*-----
* Write to PINSEL2/3 to select the PHY functions on P1[17:0]
* P1.6, ENET-TX_CLK, has to be set for Rev '-' devices and it
* must not be set for Rev 'A' and newer devices
*-----*/
regVal = MAC_MODULEID;
if ( regVal == OLD_EMAC_MODULE_ID )
    {
        /* On Rev. '-', MAC_MODULEID should be equal to
        OLD_EMAC_MODULE_ID, P1.6 should be set. */
        PINSEL2 = 0x50151105;
        /* selects P1[0,1,4,6,8,9,10,14,15] */
    }
else
    {
        /* on rev. 'A', MAC_MODULEID should not equal to
        OLD_EMAC_MODULE_ID, P1.6 should not be set. */
        PINSEL2 = 0x50150105;
        /* selects P1[0,1,4,8,9,10,14,15] */
    }
PINSEL3 = 0x00000005; /* selects P1[17:16] */
```

### Ethernet.2: Ethernet SRAM disabled

Introduction: The LPC2468 has an Ethernet interface, which has a dedicated 16 kB SRAM.

Problem: When the Ethernet block is disabled (in the PCONP register located at 0xE01F C0C4), the Ethernet SRAM is also disabled.

Workaround: Enable the Ethernet block by setting the PCENET bit (bit no 30) in the PCONP register. The Ethernet SRAM is now enabled.

### Ethernet.3 Receive Status registers will not function correctly if RxDescriptor number is greater than 4

Introduction: The Receive number of Descriptors register (RxDescriptor-0xFFE0 0110) defines the number of descriptors in the Descriptor array. Each receive descriptor element in the Descriptor array has an

associated status field which consists of the HashCRC word and Status Information word.

**Problem:** The status words are updated incorrectly if the number of Descriptors set in the Receive number of Descriptors register is greater than or equal to 5.

**Workaround:** Define 4 or less in the Receive number of Descriptors register.

### **I<sup>2</sup>S.1: I<sup>2</sup>S DMA interface is non-operational**

**Introduction:** The LPC2468 has an I<sup>2</sup>S interface, which can be used for audio devices. The I<sup>2</sup>S interface was initially designed to operate with the general purpose DMA controller.

**Problem:** The DMA controller cannot access the I<sup>2</sup>S interface.

**Workaround:** No known workaround.

### **PLL.1: PLL output (F<sub>CCO</sub>) is limited to 290 MHz**

**Introduction:** The PLL input, in the range of 32 KHz to 50 MHz, may initially be divided down by a value "N", which may be in the range of 1 to 256. Following the PLL input divider is the PLL multiplier. This can multiply the input divider output through the use of a Current Controlled Oscillator (CCO) by a value "M", in the range of 1 through 32768. The resulting frequency, F<sub>CCO</sub> must be in the range of 275 MHz to 550 MHz. This frequency can be divided down (using the Clock Divider registers) to get the desired clock frequencies for the core and peripherals.

**Problem:** The maximum output of the CCO within the PLL block is limited to 290 MHz.

**Workaround:** Care should be taken while programming the PLL so that F<sub>CCO</sub> resides in the desired range. The suggested setting is to use a 12 MHz external crystal. Use a PLLdivider (N) of 1 and PLL multiplier (M) of 12. Putting the values in the equation:

$$F_{CCO} = (2 \times M \times F_{IN}) / N$$

$$F_{CCO} = 288 \text{ MHz}$$

The CPU Clock Configuration register (located at 0xE01F C104) can then be used to divide this frequency by 4 to produce the maximum CPU speed of 72 MHz (except on Rev '-' and Rev 'A', see Flash.1).

### **SRAM.1: 16 kB SRAM cannot be used for code execution**

**Introduction:** The LPC2468 has 16 kB of SRAM on the AHB2 bus, which would generally be used by the Ethernet block.

**Problem:** The 16 kB of SRAM can only be used as data RAM. Code can not be executed from this memory.

**Workaround:** No known workaround.

### **USB.1: USB\_NEED\_CLK is always asserted**

**Introduction:** The USB\_NEED\_CLK signal is used to facilitate going into and waking up from chip Power Down mode. USB\_NEED\_CLK is asserted if any of the bits of the USBClkSt register are asserted.

**Problem:** The USB\_NEED\_CLK bit of the USBIntSt register (located at 0xE01F C1C0) is always asserted, preventing the chip from entering Power Down mode when the USBWAKE bit is set in the INTWAKE register (located at 0xE01F C144).

**Workaround:** After setting the PCUSB bit in PCONP (located at 0xE01F C0C4), write 0x1 to address 0xFFE0C008. The USB\_NEED\_CLK signal will now function correctly. Writing to address 0xFFE0C008 only needs to be done once after each chip reset.

**USB.2: U1CONNECT signal is not functional**

Introduction: U1CONNECT Signal (alternate function of P2.9) is part of the SoftConnect USB feature, which is used to switch an external 1.5 K $\Omega$  resistor under the software control.

Problem: The USB U1CONNECT alternate function does not work as expected.

Workaround: Configure P2.9 as a GPIO pin, and use it to enable the pull-up resistor on the U1D+ pin.

**USB.3: V<sub>BUS</sub> status input is not functional**

Introduction: The V<sub>BUS</sub> signal indicates the presence of USB bus power.

Problem: The V<sub>BUS</sub> status input is not functional.

Workaround: Configure P1.30 as a GPIO pin, and poll it to determine when V<sub>BUS</sub> goes to 0, signalling a disconnect event.

**USB.4: USB\_PWRDx pin(s) does not function as intended.**

Introduction: The device has a USB\_PWRD1 signal for USB port 1 and a USB\_PWRD2 signal for USB port 2. Both signals monitor the status of V<sub>BUS</sub> (USB bus power) and are active high signals.

Problem: On the Rev -, the USB\_PWRDx signals are implemented as active low signals and as a result, they are unable to monitor the status of V<sub>BUS</sub> without external inverter.

Workaround: An external inverter is needed on the USB\_PWRDx pin(s) to be able to monitor the V<sub>BUS</sub> status.

**WDT.1: Accessing non-Watchdog APB registers in the middle of the feed sequence causes a reset.**

Introduction: The Watchdog timer can reset the microcontroller within a reasonable amount of time if it enters an erroneous state.

Problem: After writing 0xAA to WDFEED, any APB register access other than writing 0x55 to WDFEED may cause an immediate reset.

Workaround: Avoid APB accesses in the middle of the feed sequence. This implies that interrupts and the GPDMA should be disabled while feeding the Watchdog.

**Core.1 Incorrect update of the Abort Link register in Thumb state**

Introduction: If the processor is in Thumb state and executing the code sequence STR, STMIA or PUSH followed by a PC relative load, and the STR, STMIA or PUSH is aborted, the PC is saved to the abort link register.

Problem: In this situation the PC is saved to the abort link register in word resolution, instead of half-word resolution.

Conditions:

The processor must be in Thumb state, and the following sequence must occur:

<any instruction>

<STR, STMIA, PUSH> <---- data abort on this instruction

LDR rn, [pc,#offset]

In this case the PC is saved to the link register R14\_abt in only word resolution, not half-word resolution. The effect is that the link register holds an address that could be #2 less than it should be, so any abort handler could return to one instruction earlier than intended.

Work around: In a system that does not use Thumb state, there will be no problem.

In a system that uses Thumb state but does not use data aborts, or does not try to use data aborts in a recoverable manner, there will be no problem.

Otherwise the workaround is to ensure that a STR, STMIA or PUSH cannot precede a PC-relative load. One method for this is to add a NOP before any PC-relative load instruction. However this is would have to be done manually.

### **Flash.1 Operating speed out of on-chip Flash is restricted**

Introduction: The operating speed of this device out of internal Flash/ SRAM is specified at 72MHz.

Problem: Code execution from internal Flash is restricted depending upon the device revision:

1. Rev 'A' devices: Code execution from internal flash is restricted to a maximum of 60MHz. For example, use a PLL output frequency of  $F_{cco}=360\text{MHz}$  and divide it by 6 (CCLKSEL=5) to generate 60MHz CPU clock (Do not use even values for CCLKSEL).
2. Rev '-' devices: Code execution from internal flash is restricted to a maximum of 60MHz also. However, this device revision has one more restriction in terms of the PLL output frequency ( $F_{cco}$ - Please refer to PLL.1 above).  $F_{cco}$  is limited to 290MHz.

Considering the same example in PLL.1 (Input crystal-12MHz,  $N=1$ ,  $M=12$ ):  $F_{cco} = 288\text{ MHz}$   
The CPU Clock Configuration register (located at 0xE01F C104) can then be used to divide this frequency by 6 (CCLKSEL=5) to achieve a maximum of 48MHz. Since this register only accepts odd values for CCLKSEL, a division by 5 (CCLKSEL=4) is not a valid option.

In both the above revisions, code can still execute out of SRAM at up to 72 MHz.

Work around: None.

### **MAM.1: Under certain conditions in MAM Mode 2 code execution out of internal Flash can fail.**

Introduction: The MAM block maximizes the performance of the ARM processor when it is running code in Flash memory. It includes three 128-bit buffers called the Prefetch Buffer, the Branch Trail Buffer and the data buffer. It can operate in 3 modes; Mode 0 (MAM off), Mode 1 (MAM partially enabled) and Mode 2 (MAM fully enabled).

Problem: Under certain conditions when the MAM is fully enabled (Mode 2) code execution from internal Flash can fail. The conditions under which the problem can occur is dependent on the code itself along with its positioning within the Flash memory.

Workaround: If the above problem is encountered then Mode 2 should not be used. Instead, partially enable the MAM using Mode 1.

### **CAN.1: Data Overrun condition can lock the CAN controller**

Introduction: Each CAN controller provides a double Receive Buffer (RBX) per CAN channel to store incoming messages until they are processed by the CPU. Software task should read and save received data as soon as a message reception is signaled.

In cases, where both receive buffers are filled and the contents are not read before the third message comes in, a CAN Data Overrun situation is signaled. This condition is signaled via the Status

register and the Data Overrun Interrupt (if enabled).

**Problem:** In a Data Overrun condition, the CAN controller is locked from further message reception.

**Workaround:**

1. Recovering from this situation is only possible with a soft reset to the CAN controller.
2. If software cannot read all messages in time before a third message comes in, it is recommend to change the acceptance filtering by adding further acceptance filter group(s) for messages, which are normally rejected. With this approach, the third incoming message is accepted and the Data Overrun condition is avoided. These additional messages are received with the corresponding group index number can be easily identified and rejected by software.

**Vbat.1: Increased power consumption on Vbat when Vbat is powered before the 3.3 V supply used by rest of the device.**

**Introduction:** The device has a Vbat pin which provides power only to the RTC and Battery RAM. Vbat can be connected to a battery or the same 3.3 V supply used by rest of the device (VDD(3V3) pin, VDD(DCDC)(3V3) pin).

**Problem:** If Vbat is powered before the 3.3 V supply, Vbat is unable to source the start-up current required for the Battery RAM. Therefore, power consumption on the Vbat pin will be high and will remain high until 3.3 V supply is powered up. Once 3.3 V supply is powered up, power consumption on the Vbat pin will reduce to normal and subsequent power cycle on the 3.3 V supply will not cause an increased power consumption on the Vbat pin.

**Workaround:** Provide 3.3 V supply used by rest of the device first and then provide Vbat voltage.

## AC/DC Deviations

**ESD.1:           The LPC2468 does not meet the 2 kV ESD requirements on the RTCX1 pin**

Introduction:    The LPC2468 is rated for 2 kV ESD. The RTCX1 pin is the input pin for the RTC oscillator circuit.

Problem:         The LPC2468 does not meet the required 2 kV ESD specified.

Workarounds:    Observe proper ESD handling precautions for the RTCX1 pin.

**Errata Notes**

**Note 1:** On each of the following port pins P0.23, P0.24, P0.25, P0.26, P1.30, P1.31, P0.12, and P0.13 (when configured as general purpose input pin (s)), leakage current increases when the input voltage is  $V_i \geq V_{dd\ I/O} + 0.5\text{ v}$ . Care must be taken to limit the current to less than 4 mA by using a series limiting resistor.