

# APPLICATION NOTE

## **AN10167**

Autocalibration of internal low power oscillator for clocking the UART

Author: Christian Kulig

2003 Jan 20

# Autocalibration of internal low power oscillator for clocking the UART

AN10167

## INTRODUCTION

The LPC932 provides a number of features, which help to reduce power consumption as well as to minimize external components.

For example, it might be an option for applications in battery powered and handheld devices to clock the microcontroller with the internal watchdog timer (WDT). (See figure 1)

This lowers the power consumption considerably compared to the use of an external crystal or the internal RC oscillator. Also, there's no more need for the crystal and its two capacitors. Thus, applications, which don't need a high clock frequency, can be developed more economically. Although the WDT oscillator has a low initial accuracy, it's possible to achieve a highly accurate time base if required. For example to drive the integrated UART, it's necessary to set up a baud rate with a deviation smaller than  $\pm 2.5\%$ . Since the watchdog oscillators frequency is in a range between  $-30\%$  and  $+20\%$  of 400kHz, the actual frequency must be measured and considered before setting up the particular baud rate. This means the watchdog oscillator has to be calibrated.

Fortunately, the actual frequency is very stable, so that the WDT only needs to be calibrated once. The results are written into the EEPROM and can be used from then on.

This description shows in detail how to setup the UART as an example, but the calibration itself and its results can be used for any other purpose requiring an accurate time base.

## CALIBRATION PRINCIPLE (See figure 2)

A convenient way to generate a baud rate for the UART is to use the internal baud rate generator. It divides the CCLK by a factor that is defined by two registers BRG0 and BRG1. If CCLK is  $f_{WDT} = 400\text{ kHz}$  and the intended baud rate is 9600 the factor should be 42. Obviously, this factor is proportional to CCLK and  $f_{WDT}$ .

To determine the frequency of the WDT we just have to count watchdog timer events (counts, overflows) during a certain period. It's easy to use overflows, because they cause an interrupt, which can be used to increment a counter. To maximize the accuracy we set up the highest overflow frequency, by writing a 0 into the watchdog timer reload register. Now every 80  $\mu\text{s}$  an overflow occurs. The resulting count value could be used directly as the factor for the baud rate generator, but it's advantageous to count a value that's 4 times the required factor. By shifting the result by 2 digits to right, the LSB error can be removed.

Now the measurement time is  $4 \times 42 \times 80\ \mu\text{s} = 13.44\text{ms}$ . The result of the calibration is as accurate as this time can be measured.

When the LPC932 is powered up for the first time, it uses the internal RC oscillator as its clock source. It has a tolerance of  $\pm 2.5\%$  over the whole temperature range and is better than  $\pm 0.8\%$  when used at room temperature. This tolerance is well suited for performing the calibration. The RC oscillator has a frequency of 7.373 MHz, this means an internal timer (e.g. Timer0) increments every 0.271  $\mu\text{s}$ . To get a total measurement time of 13.44ms it has to count 49547 times until it overflows and causes an interrupt. This interrupt must stop all timers (T0 and WDT) and write the current count value into the EEPROM (In-Application Programming).

Now that the WDT is calibrated, an IAP routine is called to set up the microcontroller to use the WDT as its clock source. The last step is to execute a software reset to let the new clock settings take effect.

On further power-ups this calibration procedure can be skipped, since the EEPROM contains the necessary information. It can be read, shifted right by 2 digits and used as the factor for the baud rate generator to clock the UART.

It can also be used for the prescaler of any other peripheral of the LPC932 that needs an accurate time base.

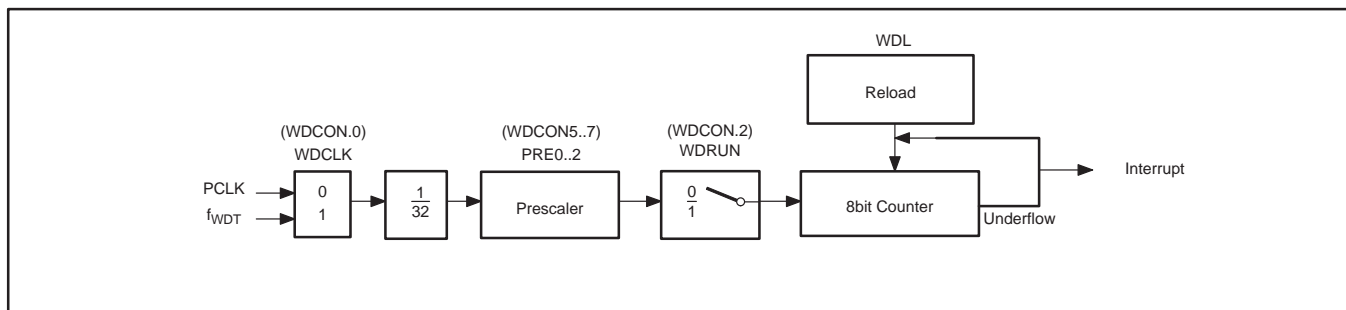


Figure 1. Watchdog timer in timer mode

# Autocalibration of internal low power oscillator for clocking the UART

AN10167

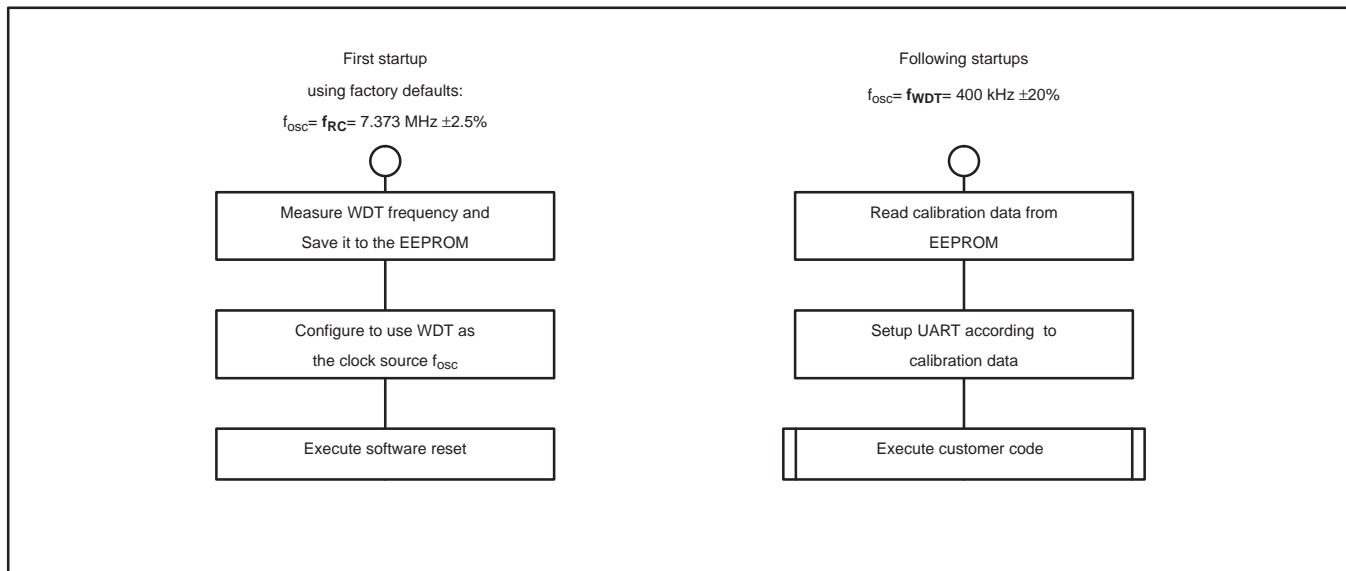


Figure 2. Calibration flowchart

## SOFTWARE EXAMPLE

The following code implements the calibration of the WDT. It's divided into two major parts, so that it's easy to include it in existing projects.

The first four files are a simple software example. It uses the UART for receiving a command ('S' = Send Data) from the PC and sending back some data.

The last two files contain the necessary code, which calibrates the watchdog timer.

To test this example, the LPC932 needs a RS232 interface to a PC, which may be provided by the Keil MCB900 evaluation board.

Start the program 'Hyperterminal' (Start / Programs / Accessories / Communications) and set the configuration to 9600 baud, 8 data bits, 1 stop bit, no parity bit and no flow control. Then click on 'Wait for Call'.

After a power up, the programmed device blinks the LEDs on Port 2. That means, it has already calibrated the watchdog timer and is using it as the clock source.

By pressing 'S' in Hyperterminal, the LPC932 sends out some data to the PC, which are displayed by Hyperterminal.

# Autocalibration of internal low power oscillator for clocking the UART

AN10167

## CustomerCode.c

```
#include <Reg932.h>
#include "CustomerCode.h"

unsigned char dat[]="ABCDEFGHJ";
unsigned char calStat=0;

void main(void)
{
    init();
    UART_init();
    WDcal();
    EA=1;
    while(1)
    {
        P2=0x00;
        msec(5);
        P2=BRGR0; // blinks BRGR0
        msec(5);
        if (sendRQ)
        {
            sendtoPC(dat);
            sendRQ=0;
        }
    }
}
```

## CustomerCode.h

```
#ifndef CUSTOMERCODE
#define CUSTOMERCODE

// includes for customer software
#include "CSTM_lib.h"
#include "WDCAL_lib.h"
#endif
```

## Cstm\_lib.h

```
#ifndef CSTM_LIB
#define CSTM_LIB

// customer specific functions
extern void init();
extern void UART_init();
extern void UART_ISR(void);
extern void sendtoPC(char* dat);
extern unsigned char sendRQ;
#endif
```

## Cstm\_lib.c

```
#include <REG932.H>
#include "CSTM_lib.h"
#include "WDCAL_lib.h"
unsigned char sendRQ=0;

void UART_ISR(void) interrupt 4
{
    RI = 0;
    if (SBUF=='S') sendRQ=1;
}

void sendtoPC(char* dat)
```

```
{
    int j=0;
    while(dat[j])
    {
        TI=0;
        SBUF=dat[j];
        while (!TI);
        j++;
    }
}

void init(void)
{
    P1M1 = 0x00; //push pull
                //except RX
    P1M2 = 0xFD;
    P2M1 = 0x00;
    P2M2 = 0xFF;
    ES = 1;
    EA = 1;
}

void UART_init()
{
    unsigned char presc=0;
    SCON=0x50; // select BRG as UART
                // for 9600 baud
    SSTAT=0x60; // separate Rx/Tx
                //interrupts
    // calculate presc.
    presc=(EEPROMread(ADRWDFQ)>>2)-16;
    BRGR0=(unsigned char)presc;
    BRGR1=(unsigned char)(presc>>8);
    BRGCON = 0x03; //enable BRG
}
}
```

## WDCal\_lib.h

```
#ifndef WDCAL_LIB
#define WDCAL_LIB

#define ADRSTAT 510
#define ADRWDFQ 511
#define MESWD 0
#define USERC 1
#define WDOSC 0x44
#define RCOSC 0x43
#define CRYOSC 0x40

extern void msec(int x);
extern void WDT_ISR();
extern void T0_ISR();
extern void EEPROMwrite(unsigned int adr, unsigned char dat);
extern unsigned char EEPROMread(unsigned int adr);
extern void WDcal();
extern void setUCFG(unsigned char ucfg);

#endif
```

# Autocalibration of internal low power oscillator for clocking the UART

AN10167

## WDcal\_lib.c

```

//This file contains asm directives:
// Set the Compiler options
// (Keil uVision2) for this file to:
// - Generate Assembler SRC File
// - Assemble SRC File
#include <REG932.H>
#include "WDCAL_lib.h"
#include "CSTM_lib.h"

unsigned char WDTcnt=0;

void WDcal()
{
    unsigned char calStat=0;
    calStat=EEPROMread(ADRSTAT);
    switch(calStat)
    {
        case MESWD:
            EEPROMwrite(ADRSTAT,calStat+1);
            // Set Timer0 16bit Mode 1
            TAMOD=0x00;
            TMOD=0x01;
            TH0=0x3E;
            TL0=0x74;
            IP0H|=0x02; // lvl 2
            IP0&=0xFD;
            // Setup Watchdog
            // WDT reload=0 -> OFper=80us
            WDL=0x00;
            // Start WDT, use WDesc
            WDCON=0x05;
            IP0H&=0xBF; // lvl 1
            IP0|=0x40;
            // Enable both timers
            msec(11); // wait for
            ET0=1; // WDT update
            EC=1;
            EA=1;
            TR0=1;
            WDTcnt=0;
            while(EC);
            // set WDT osc and reset
            setUCFG(WDTOSC);
            break;
        case USERC:
            break; // Execute User code
    }
}

void EEPROMwrite(unsigned int adr,
    unsigned char dat)
{x

```

```

EA=0;
DEECON=(unsigned char)((adr>>8)&0x01);
DEEDAT=dat;
DEEADR=(unsigned char) adr;
EA=1;
while((DEECON&0x80)==0);
}

unsigned char EEPROMread(unsigned int adr)
{
    DEECON=(unsigned char)((adr>>8)&0x01);
    DEEADR=(unsigned char) adr;
    while((DEECON&0x80)==0);
    return DEEDAT;
}

void WDT_ISR() interrupt 10
{
    WDTcnt++;
    WDCON&=0x0FD;
}

void T0_ISR() interrupt 1
{
    EC=0;
    ET0=0;
    EEPROMwrite(ADRWFQ, WDTcnt);
}

void setUCFG(unsigned char ucfg)
{
    EA=0;
    ACC=ucfg; // 0x44: WDTosc
    #pragma asm
    MOV R5,A
    MOV A,#02h
    MOV R7,#00h
    CALL 0FF00h ;call IAP
    MOV AUXR1, #08h ;reset
    #pragma endasm
    EA=1;
}

void msec(int x) // @ 7.3 MHz
{
    int j=0;
    while(x>=0)
    {
        for (j=0; j<1350; j++);
        x--;
    }
}

```

# Autocalibration of internal low power oscillator for clocking the UART

AN10167

## Definitions

**Short-form specification** — The data in a short-form specification is extracted from a full data sheet with the same type number and title. For detailed information see the relevant data sheet or data handbook.

**Limiting values definition** — Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 60134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.

**Application information** — Applications that are described herein for any of these products are for illustrative purposes only. Philips Semiconductors make no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

## Disclaimers

**Life support** — These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips Semiconductors customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips Semiconductors for any damages resulting from such application.

**Right to make changes** — Philips Semiconductors reserves the right to make changes in the products—including circuits, standard cells, and/or software—described or contained herein in order to improve design and/or performance. When the product is in full production (status 'Production'), relevant changes will be communicated via a Customer Product/Process Change Notification (CPCN). Philips Semiconductors assumes no responsibility or liability for the use of any of these products, conveys no license or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

## Contact information

For additional information please visit  
<http://www.semiconductors.philips.com>. Fax: +31 40 27 24825

For sales offices addresses send e-mail to:  
[sales.addresses@www.semiconductors.philips.com](mailto:sales.addresses@www.semiconductors.philips.com)

© Koninklijke Philips Electronics N.V. 2003  
All rights reserved. Printed in U.S.A.

Date of release: 01-03

Document order number:

9397 750 10992

*Let's make things better.*