Waking up MCU Only When LPADC Data Exceeded

Example Code Introduction for 32-bit NuMicro® Family

Document Information

Application	LPADC in the M2L31 series microcontroller (MCU) can work itself, regularly sampling and comparing with threshold, and waking up MCU only when the data exceeds the threshold.
BSP Version	M2L31_Series_BSP_CMSIS_V3.01.000
Hardware	NuMaker-M2L31KI V2.1

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1. Overview

Some peripherals in the M2L31 series microcontroller (MCU) can still work in sleep mode, such as LPTimer, LPADC, LPUART and so on. In this example code, LPADC is triggered by LPTimer periodically. If the result of LPADC exceeds the threshold, the MCU will be woken up to deal with the subsequent matters. If the conversion result does not exceed the threshold, the MCU remains in sleep mode.

This code can be used to monitor sensor at the lowest power consumption.

1.1 Principle

Set the LPTIMER of the M2L31 counting in periodic mode to trigger LPADC while MCU works in NPD4 sleep mode. Then the LPADC data is compared with the specified threshold, and the MCU will be woken up when exceeded.

1.2 Demo Result

If the LPADC data exceeds the specified value (2048 in the code), the MCU will be woken up, the red LED on pin PC14 flashes once and then the MCU goes into NPD4 sleep mode again. LED will not flash if LPADC data does not exceed the threshold .

To verify the code performing properly, the LPADC input pin PB2 can be connected to V_{DD} or ground. Then you can see the red LED in PC14 flashing or measure the waveform at PC14 with an oscilloscope. A microammeter can be used to measure the current of the MCU when the LED does not flash. See the Pin Connect diagram in Chapter 3 for details.

2. Code Description

To reduce power consumption, MIRC is configured at 1 MHz and used as CPU, LPTimer0 and LPADC clocks.

The LPADC initialization code is as follows. The first three sentences of code are LPADC is configured working as single-ended mode and triggered by LPTimer0. The middle two sentences of code are LPADC's value comparator wakes up the MCU when the value exceeds 0x800. The last two sentences of code are interrupt initialization.

```
void LPADC_Init(void)
{
    SYS->VREFCTL |= SYS_VREFCTL_VBGFEN_Msk;

    /* Enable LPADC converter */
    LPADC_POWER_ON(LPADC0);

    LPADC_Open(LPADC0, LPADC_ADCR_DIFFEN_SINGLE_END, LPADC_ADCR_ADMD_SINGLE, BIT2);

    /* LPTMR0 triggered */
    LPADC_SelectAutoOperationMode(LPADC0, LPADC_AUTOCTL_TRIGSEL_LPTMR0);

    LPADC_ENABLE_CMP0(LPADC0, 2,  LPADC_ADCMPR_CMPCOND_GREATER_OR_EQUAL, 0x800, 1);

    /*Enable LPADC wakeup */
    LPADC0->AUTOCTL |= LPADC_AUTOCTL_CMP0WKEN_Msk;

    /* Enable interrupt bit */
    LPADC_ENABLE_INT(LPADC0, LPADC_CMP0_INT);

    /* Enable NVIC */
    NVIC_EnableIRQ(LPADC0_IRQn);
}
```

If LPADC is awakened, it must clear the wake flag before it can enter sleep mode again. This code is in the interrupt.

```
void LPADC0_IRQHandler(void)
{
    /* Clear wakeup flag */
    LPADC0->AUTOSTS = 7;

    /* Clear interrupt flag */
    LPADC0->ADSR0 = 7;
}
```

After configuring the clock and pin in the main function, LPTimer0 is configured. The LPTimer0 count value can be modified to change the LPADC data-flow speed. Finally, LED flashes two times indicating that the code is executed. Also, the LED will not be bright if LPADC data is not exceeded.

```
int32_t main(void)
{
      ……

    /* 1MHz/2500 = 400sps,  8MHz/625 = 12.8Ksps */
    LPTMR0->CMP = 2500 - 1;
    LPTMR0->TRGCTL = LPTMR_TRGCTL_TRGEN_Msk;
    LPTMR_Start(LPTMR0);

    LPADC_Init();

    GPIO_ReInit();
    GPIO_SetMode(PB, BIT2,  GPIO_MODE_INPUT);
    GPIO_SetMode(PC, BIT14, GPIO_MODE_OUTPUT);

    /* LED flashes 2 times indicating MCU restart */
    PC14 &= ~1;

    for (Cnt = 0; Cnt < 50;  ++Cnt) __NOP();

    PC14 |= 1;

    for (Cnt = 0; Cnt < 50000; ++Cnt) __NOP();

    PC14 &= ~1;

    for (Cnt = 0; Cnt < 50;  ++Cnt) __NOP();

    PC14 |= 1;

    SYS_ENABLE_LVR();
    SYS_DISABLE_BOD();
    SYS_DISABLE_POR();

    /* NPD013 & SPD0 need this */
    SYS_SetPowerLevel((0x2UL << SYS_PLCTL_PLSEL_Pos));
    CLK_SetPowerDownMode(CLK_PMUCTL_PDMSEL_NPD4);

    while (1)
    {
        CLK_PowerDown();
        PC14 &= ~1;

        /* LED light about 20us when MIRC = 1MHz */
        for (Cnt = 0; Cnt < 5;  ++Cnt) __NOP();

        PC14 |= 1;
    };
}
```

In the main loop, MCU will go to sleep. If MCU is woken up, LED will light 20 us and it can be seen. In other ways the oscilloscope can measure the waveform.

3. Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - M2L31_Series_BSP_CMSIS_V3.01.000
- IDE version
  - Keil uVersion 5.28

3.2 Hardware Requirements

- Circuit components
  - NuMaker-M2L31KI V2.1
- Pin Connect
  - When PB2 is connected to V_{DD}, the red LED will blink continuously due to the excessive LPADC data exceeded, and the PB2 grounded, LED will not blink.

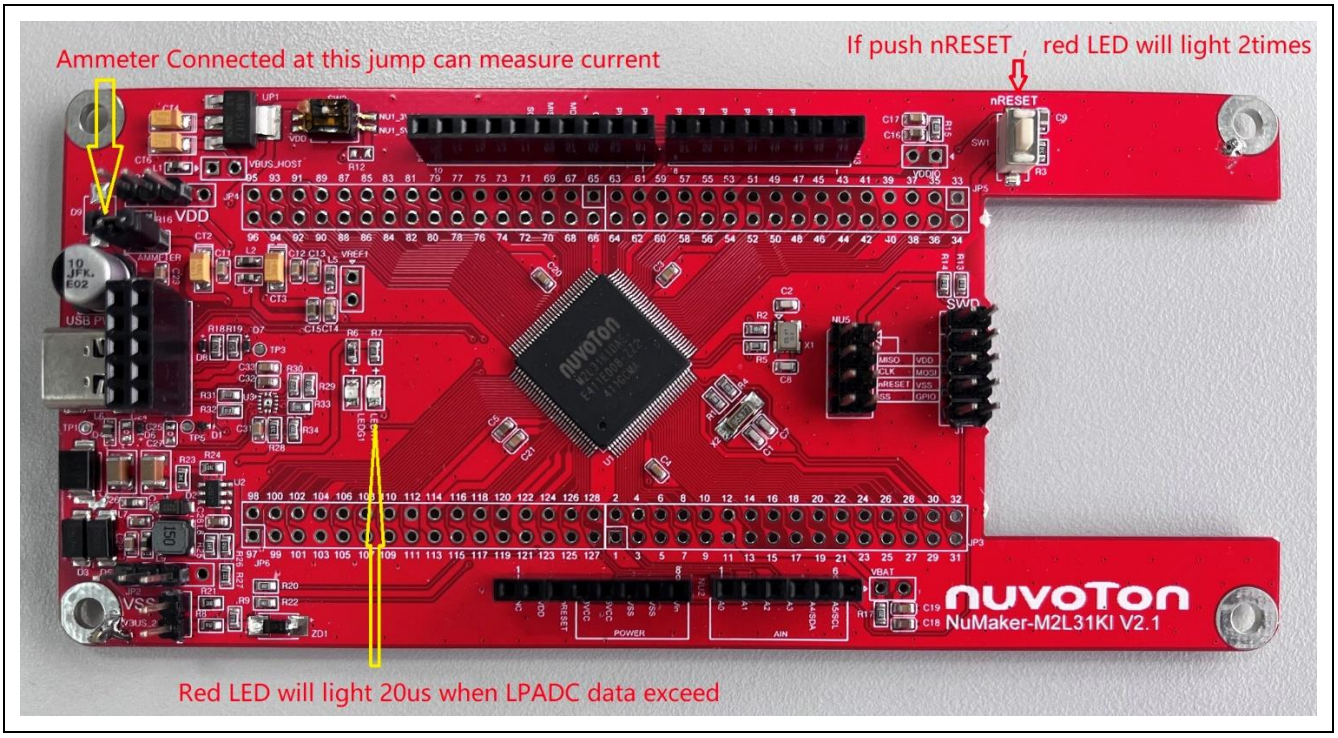

Figure 3-1 Pin Connect

4. Directory Information

EC_M2L31_LPADC_ResultMonitor_Wakeup_NPD4_V1.00	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
SampleCode	
ExampleCode	
Project	Source file of example code

Figure 4-1 Directory Structure

5. Example Code Execution

1. Browse the sample code folder as described in the Directory Information section and double-click *EC_M2L31_LPADC_ResultMonitor_Wakeup_NPD4.uvprojx*.
2. Enter Keil compile mode.
   - Build
   - Download
   - Start/Stop debug session
3. Enter debug mode.
   - Run

6. Revision History

Date	Revision	Description
2024.06.06	1.00	Initial version.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.