LPTPWM Control Breathing Light in NPD3 Mode

Example Code Introduction for 32-bit NuMicro® Family

Document Information

Application	In the M2L31 series microcontroller, the Low Power Timer (LPTMR) in PWM mode, referred to as LPTPWM, can periodically generate the desired output waveform in both Normal and Power-down mode. In this example, the LPTPWM leverages this capability to create a breathing light effect in Normal Power-down Mode 3 (NPD3). Additionally, the M2L31 can wake up from NPD3 mode as needed.
BSP Version	M2L31_Series_BSP_CMSIS_V3.01.000
Hardware	NuMaker-M2L31KI V2.2

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

The M2L31 series microcontroller supports Power-down mode for certain peripherals, including the LPTMR, LPSPI, LPUART and so on. In this example, the LPTMR operates in PWM mode (LPTPWM), regularly generating an output signal via the pulse waveform generator in NPD3 mode.

In NPD3 mode, the LPTPWM can wake up the system at a specified Compare Data (CMPDAT) or Period Data (PERIOD) point. Regardless of whether the system remains in or exits NPD3 mode, the periodic output pulse from the LPTPWM remains unaffected.

In this example, the output signal drives an LED to create a breathing light effect.

1.1 Principle

The LPTPWM clock source can be either LXT or LIRC. Given both CMPDAT and PERIOD values to the LPTMRx_PWMCMPDAT and LPTMRx_PWMPERIOD registers respectively, the LPTPWM pulse generator uses these two values to generate PWM output pulse in the up-count mode.

In periodic mode, the LPTPWM counter starts from zero, with its current value stored in the LPTMRx_PWMCNT register. Once the LPTMRx_PWMCNT value exceeds the PERIOD value, the counter resets and restarts from zero.

As shown in Figure 1-1, the LPTPWM pulse generator sets the output signal to a high voltage state when the LPTMRx_PWMCNT value is less than the assigned CMPDAT value. Once the LPTMRx_PWMCNT value reaches or exceeds CMPDAT, the output signal switches to a low voltage state.

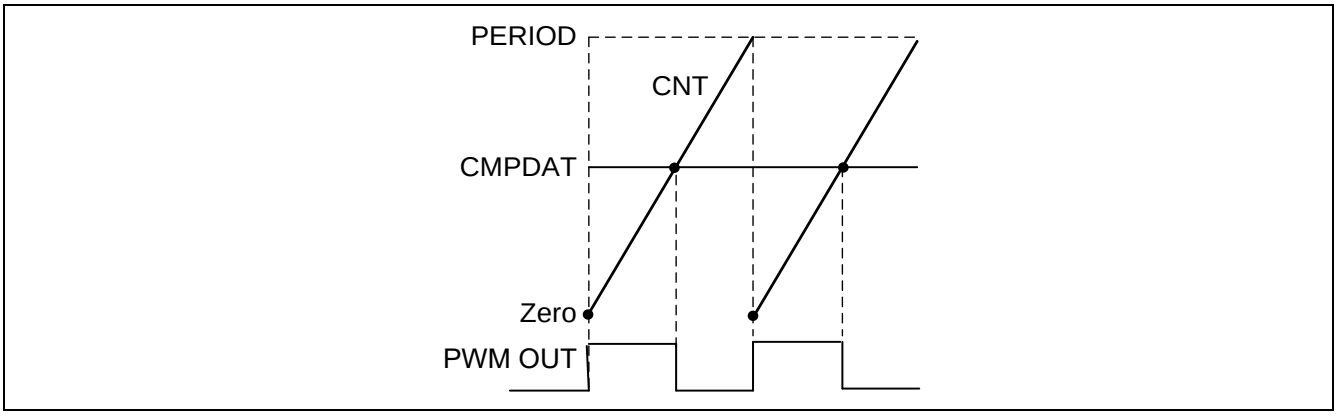

Figure 1-1 LPTPWM Pulse Generator

When the PDCLKEN register (LPTMRx_CTL[16]) is enabled, the LPTPWM internal counter continues running in NPD3 Mode, allowing the pulse generator to produce the assigned waveform in NPD3 mode. Additionally, if a CMPDAT or PERIOD wake-up event is set and triggered, the system can exit NPD3 mode.

Since the pulse generator counter remains active, the frequency of the periodic output pulse remains unchanged, regardless of whether the system is in NPD3 mode or has been woken up.

This example demonstrates the use of LPPDMA in scatter-gather mode to modify the CMPDAT setting when a PERIOD event occurs. Since the CMPDAT is updated during the PERIOD event, it affects the output duty cycle of the LPTPWM pulse generator in the next cycle. As a result, any change in duty cycle alters the LED's on-time.

To achieve a breathing light effect, two LPPDMA scatter-gather tables are used. In one table, the LED's on-time gradually increases, while in the other, it gradually decreases.

1.2 Demo Result

This example code configures the M2L31 in NPD3 mode. On the NuMaker-M2L31KI board, a red LED cathode is connected to pin PC.14, which is assigned to the LPTM1 pin. Each time the LPTPWM pulse generator completes a cycle, the LED flashes once. The red LED remains off until the counter value reaches the CMPDAT setting, at which point the LED turns on. The LED stays on until the counter value equals or exceeds the PERIOD setting.

This example code provides two versions, which are distinguished by the definition of OPT_WAKEUP. In the first version, if the OPT_WAKEUP declaration is set to zero, the M2L31 remains in NPD3 mode while LPTPWM1 is running. In the second version, if the OPT_WAKEUP declaration is set to a value other than zero, the M2L31 wakes up on each LPTPWM1 CMPDAT and PERIOD event as shown in Figure 1-2.

In Figure 1-2, the M2L31 is woken up when CMPDAT or PERIOD event occurred. Once the M2L31 is woken up, the system HCLK is re-enabled. Channel 1 displays the HCLK output from PA.3, and channel 2 shows the pulse generator output from PC.14. After the M2L31 wakes up briefly, it is set to re-enter NPD3 mode. Regardless of whether the system is in NPD3 mode or has been woken up, the periodic output pulse generated by the LPTPWM remains unchanged.

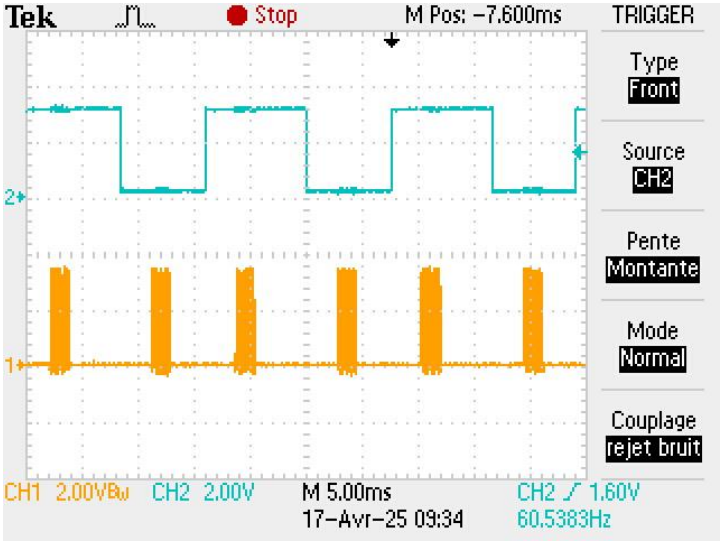

Figure 1-2 Outputs of Pulse Generator and System HCLK Clock Signal

In Figure 1-3, the debug window shows messages indicating when the M2L31 is woken up by the CMPDAT and PERIOD events, respectively.

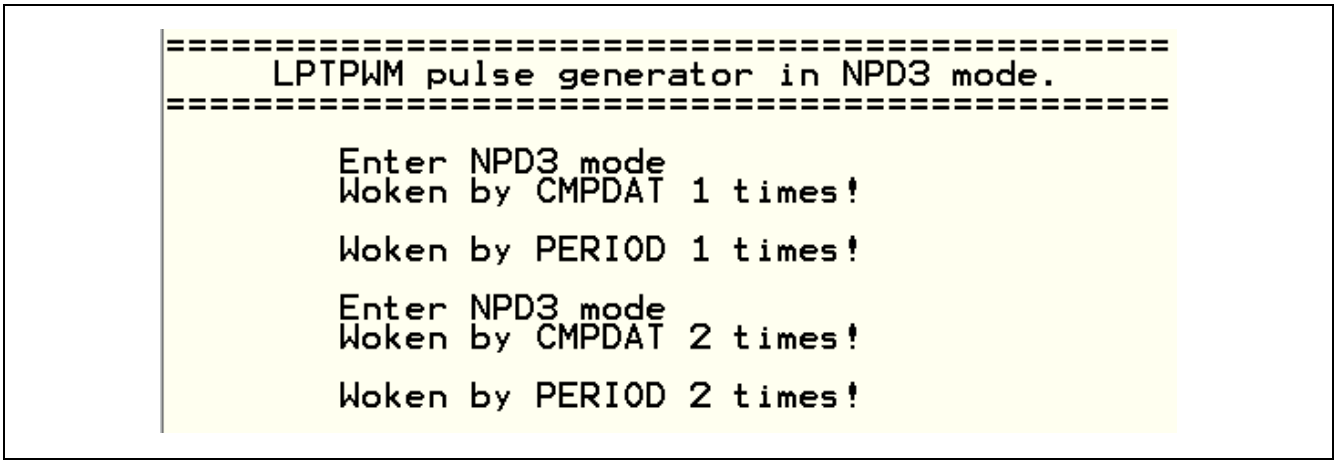

Figure 1-3 Debug Window

2. Code Description

This example code begins by initializing several features, including general system functions, HCLK clock output, LPTMR1, and LPPDMA, using the SYS_Init(), CLK_EnableCKO(), LPTPWM1_Init(), and LPPDMA_Init() functions, respectively. After these features are initialized, the LPTPWM1 counter is enabled to start the pulse generator.

Before calling the LPPDMA_Init() function, two array buffers must be set up properly for the LPPDMA scatter-gather tables. These arrays store different CMPDAT settings: one array contains values that gradually increase, while the other contains values that gradually decrease.

Since the CMPDAT value changes by two with each LPPDMA trigger, the size of each array must be at least half of the PERIOD setting to ensure proper configuration.

```
int main(void)
{
    ...

    /* Init System, IP clock and multi-function I/O. */
    SYS_Init();
    ...
    /* Output selected clock to CKO, CKO Clock = HCLK / 2^(15+1) */
    CLK_EnableCKO(CLK_CLKSEL1_CLKOSEL_HCLK, 15, 1);

    /* Init LPTPWM1 */
    LPTPWM1_Init();

    /* Get the given value PERIOD register to be 0x214 */
    g_u32Period = LPTPWM_GET_PERIOD(LPTMR1);
    g_u32BufLength = g_u32Period/CMPDAT_VAR;

    /* Set two arrays for LPPDMA scatter-gather tables */
    for (i=0; i<g_u32BufLength; i++)
    {
        g_au16SrcArray0[i] = CMPDAT_VAR*i;
        g_au16SrcArray1[i] = g_u32Period - CMPDAT_VAR*i;
    }

    /* Init LPPDMA */
    LPPDMA_Init(g_u32BufLength);

    /* Start LPTPWM1 counter */
    LPTPWM_START_COUNTER(LPTMR1);

    ...
}
```

Main loop

In the while(1){…} main loop, the M2L31 first enters NPD3 mode and waits to be woken up if OPT_WAKEUP is set to non-zero value. When the M2L31 is woken up, the code checks whether the wake-up source is the CMPDAT or PERIOD event.

```
int main(void)
{
    ...

    while(1)
    {
        /* System enters NPD3 mode */
        printf("\tEnter NPD3 mode\n");
        while(!UART_IS_TX_EMPTY(UART0));
        PowerDown_Enter();

#if OPT_WAKEUP
        /* Check which is the wakeup source */
        if (g_u32IntFlag & LPTMR_PWMINTSTS0_CMPUIF_Msk)
        {
            g_u32IntFlag &= ~LPTMR_PWMINTSTS0_CMPUIF_Msk;
            printf("\tWoken by CMPDAT %d times!\n\n", ++i);
        }

        if (g_u32IntFlag & LPTMR_PWMINTSTS0_PIF_Msk)
        {
            g_u32IntFlag &= ~LPTMR_PWMINTSTS0_PIF_Msk;
            printf("\tWoken by PERIOD %d times!\n\n", ++j);
        }
        while(!UART_IS_TX_EMPTY(UART0));
#endif
    }
    ...
}
```

LPTPWM1 initialization

The LPTPWM1_Init() function configures LPTMR1 in PWM mode and enables several features, including setting the pulse generator frequency, configuring the LPTMR1 output pin, enabling CMPDAT and PERIOD event interrupts, allowing LPTPWM1 wake-up, managing LPTMR1 clock control in Power-down mode, and triggering LPPDMA by the PERIOD event. After applying these settings, the LPTPWM1 counter is enabled upon exiting the function.

```
void LPTPWM1_Init(void)
{
    /* Change LPTMR1 to PWM mode */
    LPTPWM_ENABLE_PWM_MODE(LPTMR1);

    /* Set LPTPWM1 TMx frequency to 60 Hz with duty 0% */
    LPTPWM_ConfigOutputFreqAndDuty(LPTMR1, 60, 0);

    /* Set LPTPWM1 output to LPTM1 pin */
    LPTPWM_SELECT_TOUT_PIN(LPTMR1, LPTPWM_TOUT_PIN_FROM_TX);

    /* Enable LPTPWM1 output */
    LPTPWM_ENABLE_OUTPUT(LPTMR1, 1);

#if OPT_WAKEUP
    /* Enable CMPDAT event interrupt */
    LPTPWM_ENABLE_CMP_UP_INT(LPTMR1);
```

```
    /* Enable PERIOD event interrupt */
    LPTPWM_ENABLE_PERIOD_INT(LPTMR1);

    /* Enable LPTPWM1 wake-up function */
    LPTPWM_EnableWakeup(LPTMR1);

    /* Enable LPTPWM1 interrupt */
    NVIC_EnableIRQ(LPTMR1_IRQn);
#endif

    /* Enable LPTPWM1 clock in NPD3 mode */
    LPTMR_EnablePDCLK(LPTMR1);

    /* Enable LPTPWM1 PERIOD event to trigger LPPDMA */
    LPTPWM_EnableTriggerLPPDMA(LPTMR1, LPTPWM_TRIGGER_AT_PERIOD_POINT);
}
```

LPPDMA initialization

The LPPDMA_Init() function configures two scatter-gather tables to control the LPTPWM pulse generator. Each time a PERIOD event occurs, LPTPWM1 triggers LPPDMA to transfer one 16-bit data value from the assigned array buffer to the LPTMR1_PWMCMPDAT register. In the next pulse generator cycle, the updated CMPDAT value takes effect, adjusting the duty cycle accordingly.

```
void LPPDMA_Init(uint32_t g_u32BufLength)
{
    LPPDMA_Open(LPPDMA0,1 << LPPDMA_CH);
    DMA_DESC[0].u32Ctl = LPPDMA_OP_SCATTER |
                         LPPDMA_REQ_SINGLE |
                         LPPDMA_SAR_INC |           /* source address -> incremented */
                         LPPDMA_DAR_FIX |           /* destination address -> fixed  */
                         LPPDMA_WIDTH_16 |          /* transfer width -> 16-bit      */
                         LPPDMA_TBINTDIS_DISABLE |  /* Table Interrupt -> disabled   */
                         ((g_u32BufLength - 1) << LPPDMA_DSCT_CTL_TXCNT_Pos);
    DMA_DESC[0].u32Src = (uint32_t)g_au16SrcArray0;
    DMA_DESC[0].u32Dest = (uint32_t)&(LPTMR1->PWMCMPDAT);
    DMA_DESC[0].u32Offset = (uint32_t)&DMA_DESC[1] - (PDMA0->SCATBA);

    DMA_DESC[1].u32Ctl = LPPDMA_OP_SCATTER |
                         LPPDMA_REQ_SINGLE |
                         LPPDMA_SAR_INC |           /* source address -> incremented */
                         LPPDMA_DAR_FIX |           /* destination address -> fixed  */
                         LPPDMA_WIDTH_16 |          /* transfer width -> 16-bit      */
                         LPPDMA_TBINTDIS_DISABLE |  /* Table Interrupt -> disabled   */
                         ((g_u32BufLength - 1) << LPPDMA_DSCT_CTL_TXCNT_Pos);
    DMA_DESC[1].u32Src = (uint32_t)g_au16SrcArray1;
    DMA_DESC[1].u32Dest = (uint32_t)&(LPTMR1->PWMCMPDAT);
    DMA_DESC[1].u32Offset = (uint32_t)&DMA_DESC[0] - (PDMA0->SCATBA);

    LPPDMA_SetTransferMode(LPPDMA0, LPPDMA_CH, LPPDMA_LPTMR1, 1, (uint32_t)&DMA_DESC[0]);
}
```

LPTMR1 interrupt handler

If the OPT_WAKEUP declaration is set to a value other than zero, the M2L31 can be woken up from NPD3 mode when a CMPDAT or PERIOD event occurs, provided its corresponding interrupt is enabled. In this case, the LPTMR1 Interrupt Service Routine (ISR) is triggered first.

As the M2L31 prepares for another Power-down to wake-up cycle, all wake-up, CMPDAT, and PERIOD event flags must be cleared within the ISR to ensure the M2L31 can wake up again.

```
void LPTMR1_IRQHandler(void)
{
    /* Clear LPTPWM1 wakeup flag */
    LPTPWM_ClearWakeupFlag(LPTMR1);

    /* Check if system woken up by CMPDAT event */
    if (LPTPWM_GET_CMP_UP_INT_FLAG(LPTMR1))
    {
        /* Clear CMPDAT interrupt flag */
        LPTPWM_CLEAR_CMP_UP_INT_FLAG(LPTMR1);

        g_u32IntFlag |= LPTMR_PWMINTSTS0_CMPUIF_Msk;
    }

    /* Check if system woken up by PERIOD event */
    if (LPTPWM_GET_PERIOD_INT_FLAG(LPTMR1))
    {
        /* Clear PERIOD interrupt flag */
        LPTPWM_CLEAR_PERIOD_INT_FLAG(LPTMR1);

        g_u32IntFlag |= LPTMR_PWMINTSTS0_PIF_Msk;
    }
}
```

3. Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - M2L31_Series_BSP_CMSIS_V3.01.000
- IDE version
  - Keil uVersion 5.38.0.0

3.2 Hardware Requirements

- NuMaker-M2L31KI V2.2
- Pin Connection
  - LEDG1 is the power LED.
  - PC.14 is connected to LEDR1 cathode.

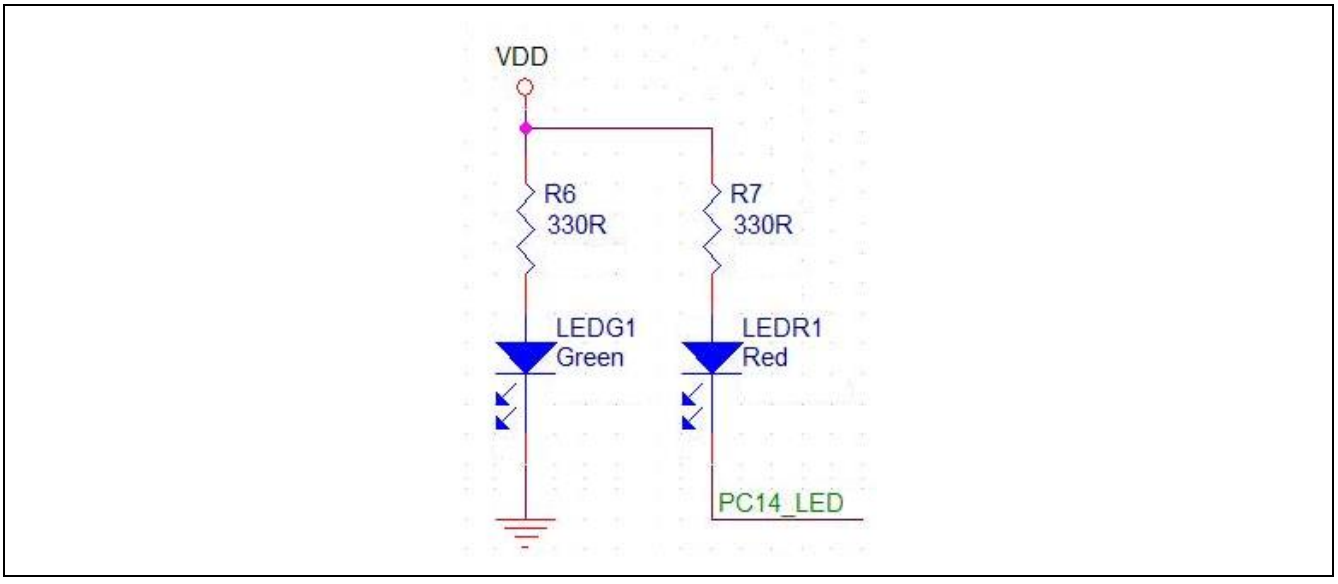

Figure 3-1 Pin Connection

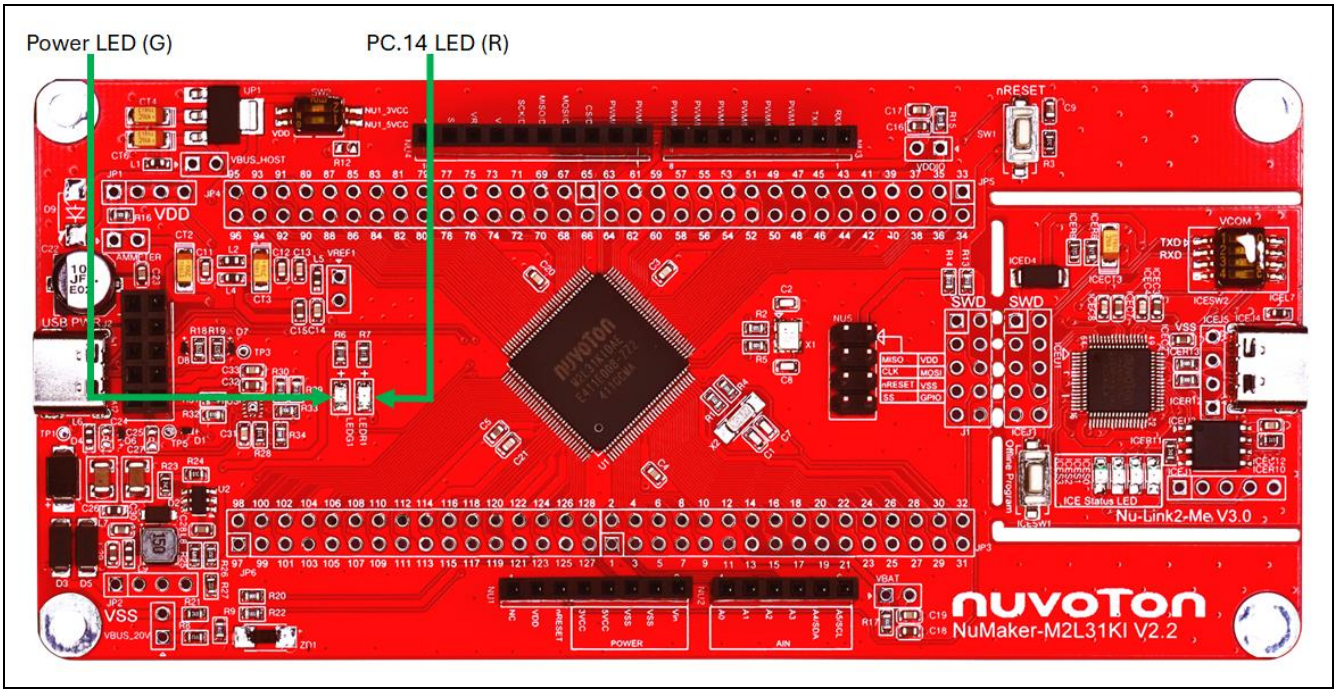

Figure 3-2 NuMaker-M2L31KI V2.2

4. Directory Information

The directory structure is shown below.

EC_M2L31_LPTPWM_BreathingLight_NPD3_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_LPTPWM_BreathingLight_NPD3.uvprojx*.
2. Enter Keil compile mode.
   - Build
   - Download
   - Start/Stop debug session
3. Enter debug mode.
   - Run

6. Revision History

Date	Revision	Description
2025.03.10	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.