# Detect UART Baud Rate by PWM Capture

Example Code Introduction for 32-bit NuMicro® Family

## Information

| | |
|---|---|
| Application | This code uses NUC123 PWMA channel 0 (PA.12) to detect UART1 baud rate by PWM capture function. |
| BSP Version | NUC123 Series BSP CMSIS v3.01.001 |
| Hardware | NuTiny-SDK-NUC123AE V1.0<br>Nu-Link2-Me |

*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 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 Function Description

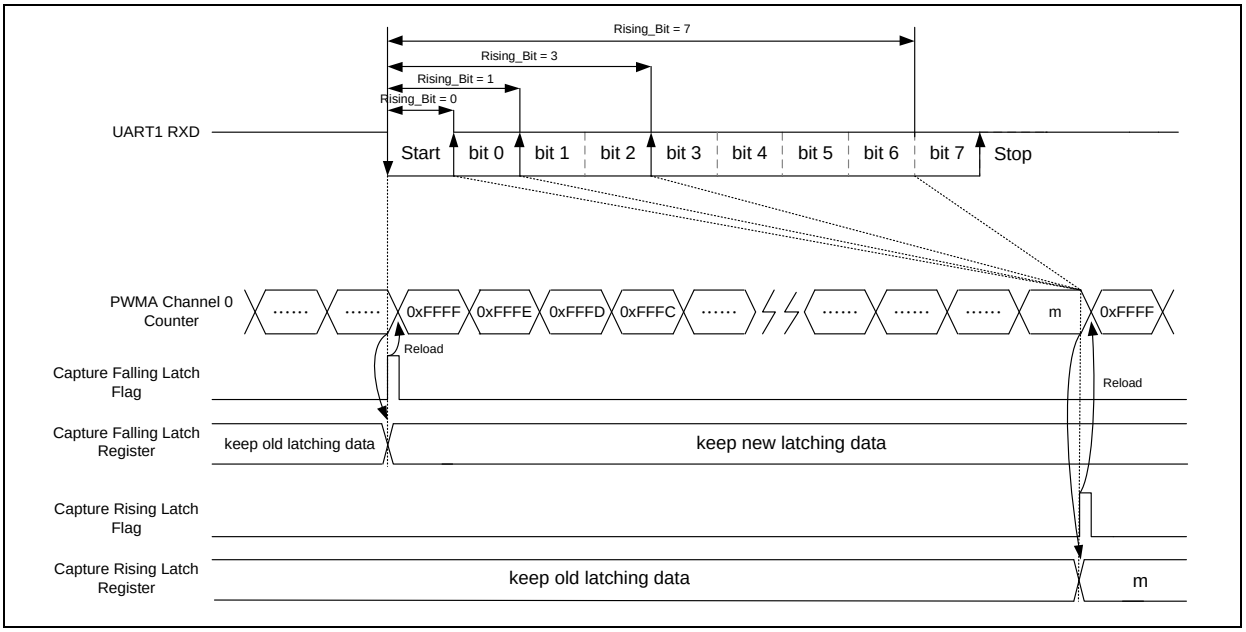
## 1.1 Introduction

This example code uses PWMA channel 0 (PA.12) to detect UART1 baud rate by PWM capture function. PWM capture function can reload the counter value when it detects falling edge and latch the counter value when it detects rising edge. User can transmit specified data on RXD and use this method to detect UART baud rate by calculating the low-level time of UART data.

To use this example code, user should connect PWMA channel 0 (PA.12) to UART1 RXD (PB.4). It also has to connect another Nu-Link2-Me ICE_TX and GND to NUC123 UART1 RXD and GND to transmit the specified data.

## 1.2 Principle

When receiving the Start bit of specified data, PWM capture will reload the counter value if user enables the interrupt. PWM capture will reload the value to 0xFFFF that is CNR maximum value. Once PWM detects the rising edge, PWM capture will latch the counter value to register. The operation is shown below figure:




User can transmit different data which rising in different data bit to change the low-level time of UART data. The result will be more precise with longer low-level time. Based on Rising_Bit defined in example code, the data should correspond to below table:

| Rising_Bit | UART Data | Rising_Bit | UART Data |
|---|---|---|---|
| 0 | 0xFF | 4 | 0xF0 |
| 1 | 0xFE | 5 | 0xE0 |
| 2 | 0xFC | 6 | 0xC0 |
| 3 | 0xF8 | 7 | 0x80 |

The UART baud rate equation is shown below:

$$UART\ baud\ rate\ = \frac{(Rising_Bit\ +\ 1)}{(0xFFFF - Rising\ Latch\ Counter\ Value) * (Counter\ Unit\ Time)}$$

Counter Unit Time equation is shown below:

$$Counter\ Unit\ Time\ = \frac{PWM\ Prescaler}{PWM\ Clock\ Source\ Frequency}$$

## 1.3 Demo Result

This example code detects the baud rate every time when Nu-Link2-Me transmits the data. It will set UART1 baud rate with new calculation results. The result is shown below picture:

```
CPU @ 71884800 Hz
+------------------------------------------------------------+
|    Detect UART Baud Rate by PWM Capture Sample Code        |
|    Use PWMA Channel 0 to detect UART1 RXD                  |
|    Please connect PWMA Channel 0 (PA.12) to UART1 RXD (PB.4)    |
+------------------------------------------------------------+
Rising_Bit is 4.

PWM Capture counter value is 104.
Set UART1 baud rate to 192307.

PWM Capture counter value is 524.
Set UART1 baud rate to 38167.

PWM Capture counter value is 2092.
Set UART1 baud rate to 9560.

PWM Capture counter value is 16743.
Set UART1 baud rate to 1194.
```

Nu-Link2-Me uses UART baud rate 194000, 38400, 9600 and 1200 to send data 0xF0 in sequential. The deviation of calculation results are -0.873%, -0.607%, -0.417% and -0.500% individually.

# 2 Code Description

Initialize PWMA channel 0 capture function. Set counter unit time to 250 ns.

```
void PWMA_Init(void)
{
    /* Set PWMA Channel 0 input clock divider */
    PWM_SET_DIVIDER(PWMA, 0, PWM_CLK_DIV_1);

    /* Set PWMA Channel 0 auto-reload mode */
    PWMA->PCR |= PWM_PCR_CH0MOD_Msk;

    /* Set PWMA Channel 0 prescaler value to make unit count time is 250 ns */
    /* PWMA clock input is HCLK = 72 MHz, prescaler value is HCLK*250/(10^9) - 1 = 17 */
    PWM_SET_PRESCALER(PWMA, 0, 17);

    /* Set PWMA Channel 0 CNR to max value */
    PWM_SET_CNR(PWMA, 0, 0xFFFF);

    /* Enable capture rising and falling edge interrupt for PWMA channel 0 */
    PWM_EnableCaptureInt(PWMA, PWM_CH0, PWM_CAPTURE_INT_RISING_LATCH |
PWM_CAPTURE_INT_FALLING_LATCH);

    /* Enable PWMA NVIC interrupt */
    NVIC_EnableIRQ(PWMA_IRQn);

    /* Enable Timer for PWMA channel 0  */
    PWM_Start(PWMA, BIT0);

    /* Enable Capture Function for PWMA channel 0 */
    PWM_EnableCapture(PWMA, BIT0);

    /* Clear detect flag */
    g_u8DetectFlag = 0;
}
```

Reset g_u32CaptureCounter value when detecting falling edge and get counter value when detecting rising edge in PWM interrupt handler.

```
void PWMA_IRQHandler(void)
{
    uint32_t u32CaptureFlag = PWM_GetCaptureIntFlag(PWMA, 0);
```

```
    /* Detect rising edge */
    if (u32CaptureFlag == 1)
    {
        /* Clear rising edge flag */
        PWM_ClearCaptureIntFlag(PWMA, 0, PWM_CAPTURE_INT_RISING_LATCH);

        /* Detect rising bit, get counter value */
        g_u32CaptureCounter = 0xFFFF - PWM_GET_CAPTURE_RISING_DATA(PWMA, 0);

        /* Set detect flag */
        g_u8DetectFlag = 1;
    }
    /* Detect falling edge */
    else if (u32CaptureFlag == 2)
    {
        /* Clear falling edge flag */
        PWM_ClearCaptureIntFlag(PWMA, 0, PWM_CAPTURE_INT_FALLING_LATCH);

        /* Detect start bit, clear g_u32CaptureCounter to get counter value */
        g_u32CaptureCounter = 0;
    }
    /* Detect both rising and falling edge */
    else if (u32CaptureFlag == 3)
    {
        /* Clear rising edge flag */
        PWM_ClearCaptureIntFlag(PWMA, 0, PWM_CAPTURE_INT_RISING_LATCH);

        /* Clear falling edge flag */
        PWM_ClearCaptureIntFlag(PWMA, 0, PWM_CAPTURE_INT_FALLING_LATCH);
    }
}
```

Calculate new UART baud rate and set it to UART1.

```
    while (1)
    {
        /* Wait until detection finish */
        while (!g_u8DetectFlag);

        /* Print PWM Capture result */
        printf("PWM Capture counter value is %d.\n", g_u32CaptureCounter);

        /* Set UART1 baud rate */
```

```
        u32Baudrate = (uint32_t)((Rising_Bit + 1) * 100000000) / (g_u32CaptureCounter *
25);
        printf("Set UART1 baud rate to %d.\n\n", u32Baudrate);

        /* Init UART1 */
        UART1_Init(u32Baudrate);

        /* Clear detect flag */
        g_u8DetectFlag = 0;
    }
```

# 3 Software and Hardware Environment

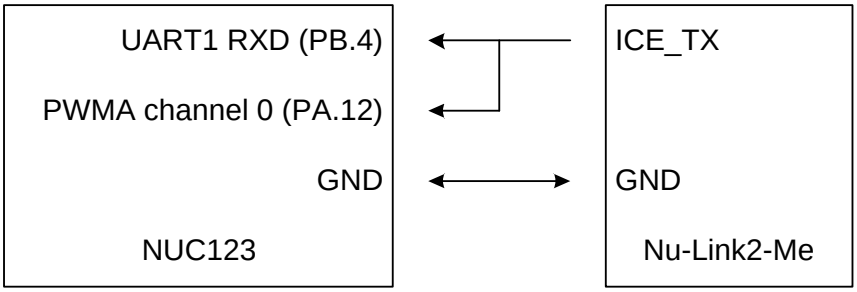
## Software Environment

- BSP version
  - NUC123Series BSP CMSIS v3.01.001
- IDE version
  - Keil uVersion 5.26

## Hardware Environment

- Circuit components
  - NuTiny-SDK-NUC123AE V1.0
  - Nu-Link2-Me
- Diagram

NUC123 uses UART1 RXD (PB.4) to connect with Nu-Link2-Me ICE_TX pin to receive data. NUC123 also needs to connect PWMA channel0 (PA.12) with UART1 RXD.



# 4 Directory Information

- EC_NUC123_Detect_Baudrate_by_PWM_Capture_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 — Source file of example code

# 5 How to Execute Example Code

1. Browsing into sample code folder by Directory Information (section 4) and double click NUC123_Detect_Baudrate_by_PWM_Capture.uvproj.
2. Enter Keil compile mode
   a. Build
   b. Download
   c. Start/Stop debug session
3. Enter debug mode
   a. Run
4. Use terminal tool to send specified data from Nu-Link2-Me to NUC123 UART1 RXD

# 6 Revision History

| Date | Revision | Description |
|---|---|---|
| July 1, 2019 | 1.00 | 1. Initially issued. |

## 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.*