Timer 擷取 IR 資料

NuMicro® 32 位系列微控制器範例代碼介紹

文件資訊

代碼簡述	使用 Timer 2 capture 功能來擷取 IR 資料
BSP 版本	NUC230/240 Series BSP CMSIS v3.01.002
開發平台	NuTiny–EVB–NUC240–LQFP100 V1.0

1 功能介紹

1.1 簡介

此示例代碼將演示如何使用 Timer 2 capture 功能來擷取符合 NEC IR 協議的 IR 資料。

1.2 原理

在圖 1-1 內顯示了此範例程式的示意圖，包含了 NUC240 MCU，IR 接收模組和 IR 訊號發送器。

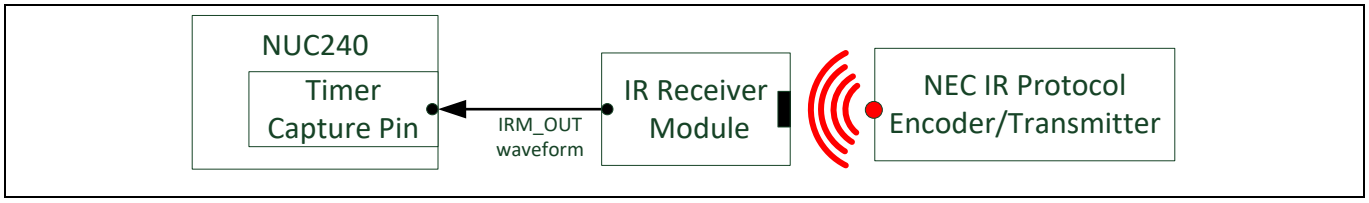


圖 1-1 範例示意圖

NEC 紅外線協議 (NEC IR Protocol) 特性為：

1. 由一個 Leader 訊號表示啟動傳輸，包含 9 ms 的低電位輸出的和 4.5 ms 的高電為輸出
2. 接下來為 8-bit 的 Address 碼和 8-bit 的 Address 反向碼
3. 再來為 8-bit 的 Command 碼和 8-bit 的 Command 反向碼
4. 最後為一個 562.5 us 低電位的 Stop 訊號，表示傳輸結束

圖 1-2 為 NEC IR 協議的資料型態 (NEC_IR) 和 IR 接收器收到訊號後所輸出波形 (IRM_OUT) 的示意圖。

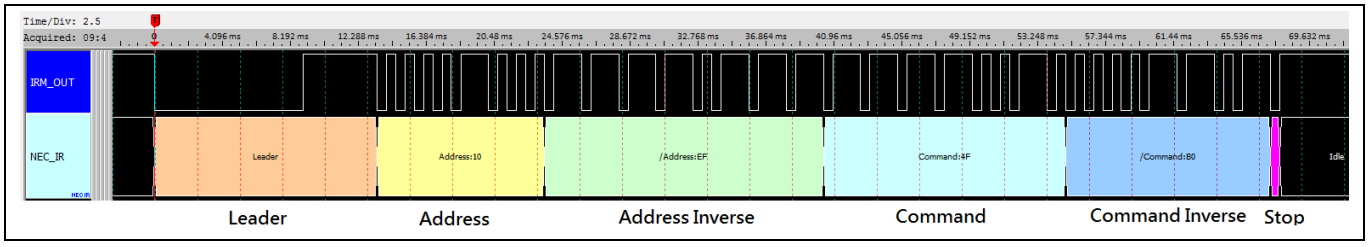


圖 1-2 NEC IR 協議和 IR 接收器模組輸出波形

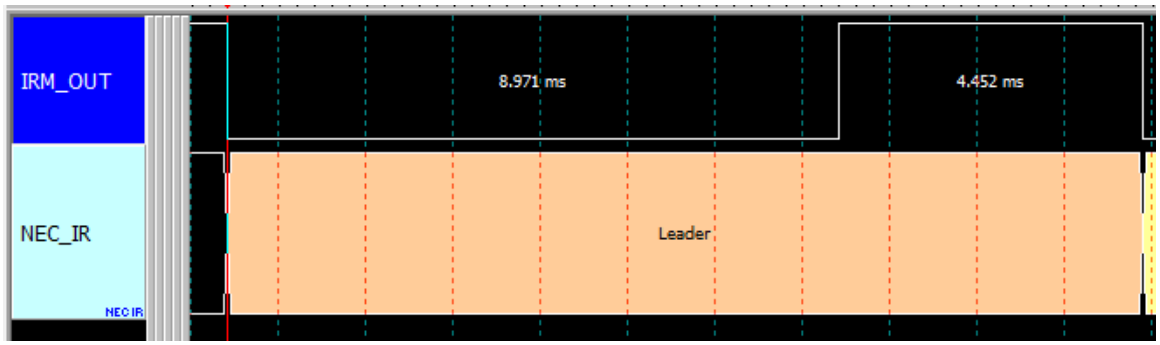
圖示 Address/Command 內每一個 bit 的值是由接收到的波形 (IRM_OUT) 狀態和長度來決定的。

- 邏輯 0：562.5 us 低電位 + 562.5 us 高電位，總計 1125 us
- 邏輯 1：562.5 us 低電位 + 1687.5 us 高電位，總計 2250 us

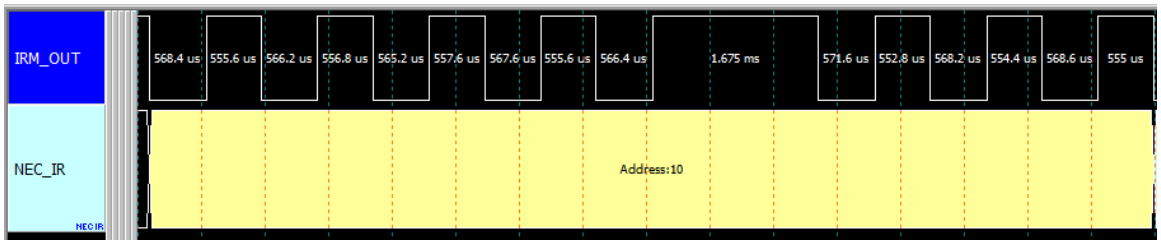
根據以上的說明和圖示，當 IR 接收模組收到 IR 信號時，會產生下降沿觸發。只要用戶可以抓取連續的兩個下降沿觸發間隔，便可以由這間隔時間知道傳入的訊號為何。
因此，此示例代碼使用 NUC240 Timer 的擷取功能來得到 IR 接收模組收到的連續下降沿觸發間隔時間。

根據圖 1-2，NEC IR 協議的每個部分描述如下：

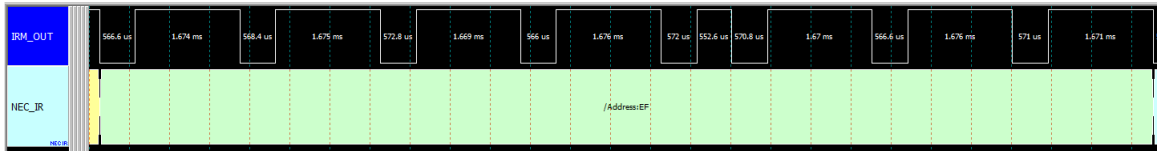
- Leader 訊號間隔



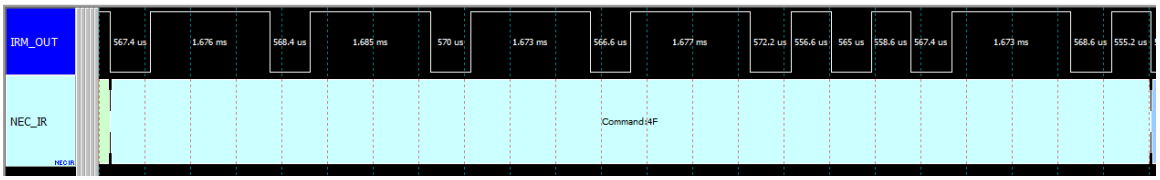
- Address 碼 (0x10)



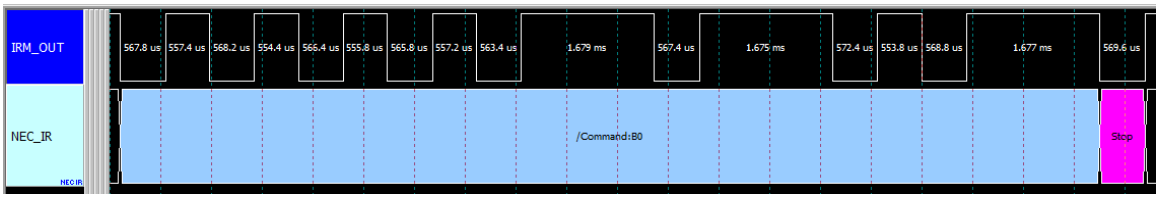
- Address 反向碼 (0xEF)



- Command 碼 (0x4F)



- Command 反向碼 (0xB0) 和 Stop 訊號間隔



1.3 執行結果

程序執行結果可經由串口鮑率 115200 的輸出到 PC 做顯示，如下圖 1-3 所示。

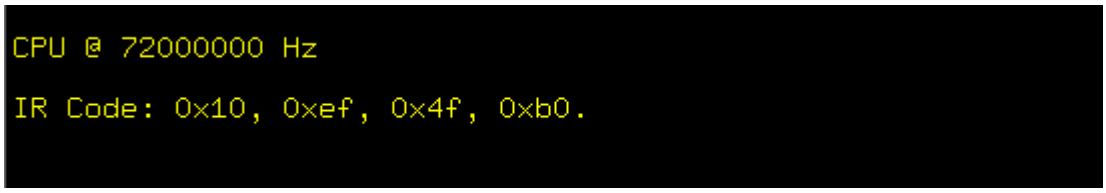


圖 1-3 串口執行結果

2 代碼介紹

宣告在擷取功能運作時用於儲存下降沿的間隔時間和擷取數量的變數。

```
#define IR_FRAME_TOUT       (100000) // Received time-out interval: 100 ms

volatile uint32_t gu32IRINRIdx = 0;     // To indicate the interval index of IR frame
volatile uint32_t gu32IsIRINRDone = 0;  // To indicate an IR frame is completed or not
volatile uint32_t gau32IRINR[33] = {0}; // To save all intervals of IR frame
```

在 TMR2_IRQHandler( ) 內實作 Timer 2 擷取 PB.2 的下降沿間隔。

```
void TMR2_IRQHandler(void)
{
    uint32_t interval;

    /* To check if receiving IR signal interval has time-out */
    if (TIMER_GetIntFlag(TIMER2) == 1)
    {
        /* IR received has time-out and reset the receive state */
        gu32IRINRIdx = 0;
        gu32IsIRINRDone = 0;
        TIMER_ClearIntFlag(TIMER2);
        return ;
    }

    /* To capture the two falling edge-triggered interval */
    if (TIMER_GetCaptureIntFlag(TIMER2) == 1)
    {
        /* Clear Timer2 capture interrupt flag */
        TIMER_ClearCaptureIntFlag(TIMER2);

        if (PB2 == 0) // capture pin status
        {
            /* Reload TIMER2 counter to 0 */
            TIMER_SET_CMP_VALUE(TIMER2, IR_FRAME_TOUT);

            if (gu32IRINRIdx == 0)
            {
                /* It's starting signal to capture the 1st interval - "Leader" */
```

```
                gu32IRINRIdx++;
            }
            else
            {
                /* Save to interval to gau32IRINR[] */
                interval = TIMER_GetCaptureData(TIMER2);
                //printf("INR: %d(%d)\n", interval, gu32IRINRIdx);
                gau32IRINR[(gu32IRINRIdx - 1)] = interval;
                gu32IRINRIdx++;

                if (gu32IRINRIdx > 33)
                {
                    /* Get a complete IR frame interval data */
                    gu32IRINRIdx = 0;
                    gu32IsIRINRDone = 1;
                }
            }
        }
        else
        {
            /* Reset the receive state */
            gu32IRINRIdx = 0;
            gu32IsIRINRDone = 0;
            printf("\nERROR: unexpected capture event !!!\n\n");
        }
    }
}
```

在 main.c 中，用戶需要初始化 Timer 2 擷取功能並配置一個 Timer 2 計數器，其單元為 1 us。在執行 TIMER_Start( ) 後，便可在 TMR2_IRQHandler( ) 內擷取 IR 接收模組收到的訊號間隔。

```
    ……
    /* Initial Timer2 default setting */
    TIMER_Open(TIMER2, TIMER_PERIODIC_MODE, 1);

    /* Configure Timer2 setting for external capture function, and one Timer2 counter is 1
us. */
    TIMER_SET_PRESCALE_VALUE(TIMER2, ((SystemCoreClock / 1000000) - 1));
    TIMER_SET_CMP_VALUE(TIMER2, IR_FRAME_TOUT);
    TIMER_EnableCapture(TIMER2, TIMER_CAPTURE_FREE_COUNTING_MODE,
TIMER_CAPTURE_FALLING_EDGE);
```

```
    TIMER_EnableCaptureInt(TIMER2);

    /* Enable Timer2 NVIC */
    NVIC_EnableIRQ(TMR2_IRQn);

    /* Start Timer2 capture function */
    TIMER_Start(TIMER2);
    ……
```

收到一組完整的 IR 訊號間隔資料後，便可在 main.c 代碼內解析出所有間隔的意義，以獲得實際的 IR 代碼，包括 Address 碼，反向 Address 碼，Command 碼和反向 Command 碼。

```
    ……
    while (1)
    {
        /* In NEC IR protocol, an IR frame format consists of
            [Leader interval] +
                [8-bit Address] + [8-bits Address Inverse] +
                [8-bit Command] + [8-bit Command Inverse] +
            [Stop interval]
        */
        /* Get one IR frame data */
        if (gu32IsIRINRDone)
        {
            /* Check if is valid "Leader" signal, 13 ~ 14 ms*/
            if ((gau32IRINR[0] > 13000) && (gau32IRINR[0] < 14000))
            {
#if 0 // enable for debugging

                for (i = 0; i < 33; i++)
                {
                    printf("INR: %d (%d)\n", gau32IRINR[i], i);
                }

                printf("---------------\n\n");
#endif

                /* Parse IR code[0] */
                offset = 1;

                for (i = 0; i < 8; i++)
```

```
{
    interval = gau32IRINR[(i + offset)];

    if (interval > 2000)
        au8IRCode[0] |= (1 << i);  // get data "1"
    else
        au8IRCode[0] |= (0 << i);  // get data "0"
}

/* Parse IR code[1] */
offset += 8;

for (i = 0; i < 8; i++)
{
    interval = gau32IRINR[(i + offset)];

    if (interval > 2000)
        au8IRCode[1] |= (1 << i);  // get data "1"
    else
        au8IRCode[1] |= (0 << i);  // get data "0"
}

/* Parse IR code[2] */
offset += 8;

for (i = 0; i < 8; i++)
{
    interval = gau32IRINR[(i + offset)];

    if (interval > 2000)
        au8IRCode[2] |= (1 << i);  // get data "1"
    else
        au8IRCode[2] |= (0 << i);  // get data "0"
}

/* Parse IR code[3] */
offset += 8;

for (i = 0; i < 8; i++)
{
    interval = gau32IRINR[(i + offset)];
```

```
                if (interval > 2000)
                    au8IRCode[3] |= (1 << i);  // get data "1"
                else
                    au8IRCode[3] |= (0 << i);  // get data "0"
            }
        }

        printf("IR Code: 0x%x, 0x%x, 0x%x, 0x%x.\n",
               au8IRCode[0], au8IRCode[1], au8IRCode[2], au8IRCode[3]);

        /* Clear au8IRCode[] data */
        for (i = 0; i < 4; i++)
            au8IRCode[i] = 0x0;

        /* Clear gau32IRINR[] data */
        for (i = 0; i < 33; i++)
            gau32IRINR[i] = 0x0;

        /* Reset the receive state */
        gu32IRINRIdx = 0;
        gu32IsIRINRDone = 0;
    }
}
```

3 軟體與硬體環境

● 軟體環境

- ■ BSP 版本
  - ◆ NUC230/240 Series BSP CMSIS v3.01.002
- ■ IDE 版本
  - ◆ Keil uVersion 5.28

● 硬體環境

- ■ 電路元件
  - ◆ NuTiny–EVB–NUC240–LQFP100 V1.0
  - ◆ IR module, IRM-2638
- ■ 示意圖
  - ◆ 將 IR 接收模組輸出訊號連接到 NUC240 Timer 2 擷取引腳 (PB.2)，以擷取 IR 訊號間隔時間。
  - ◆ 將 NUC240 UART TX (PB.1) 引腳連接到 PC UART RX，以在 PC 上顯示示例代碼的執行結果。

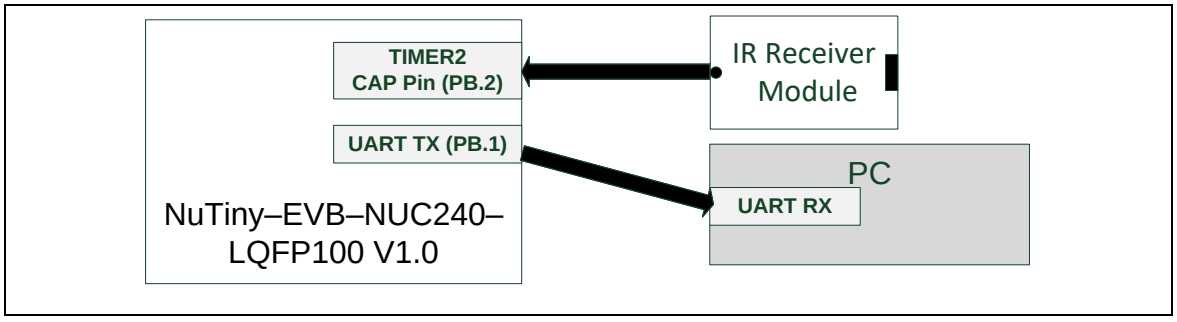


Figure 3-1 將 NUC240 引腳連接到 IR 接收模塊輸出和 PC UART RX

4 目錄資訊

- EC_NUC240 _TIMER_Capture_IR_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 如何執行範例程式

1. 根據目錄資訊章節進入 ExampleCode 路徑中的 KEIL 資料夾，雙擊 TIMER_Capture_IR.uvproj。
2. 進入編譯模式介面
   a. 編譯
   b. 下載代碼至記憶體
   c. 進入 / 離開除錯模式
3. 進入除錯模式介面
   a. 執行代碼

6 修訂紀錄

Date	Revision	Description
Sep. 18, 2019	1.00	1. 初始發布。

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.