Timer 撷取 IR 资料

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

代码简述	使用 Timer 2 capture 功能来撷取 IR 数据
BSP 版本	NUC230/240 Series BSP CMSIS v3.01.002
开发平台	NuTiny – EVB – NUC240 – LQFP100 V1.0

1 功能介绍

1.1 简介

此示例代码将演示如何使用 Timer 2 capture 功能来撷取符合 NEC IR 协议的 IR 数据。

1.2 原理

在图 1-1 内显示了此范例程序的示意图，包含了 NUC240 MCU，IR 接收模块和 IR 讯号发送器。

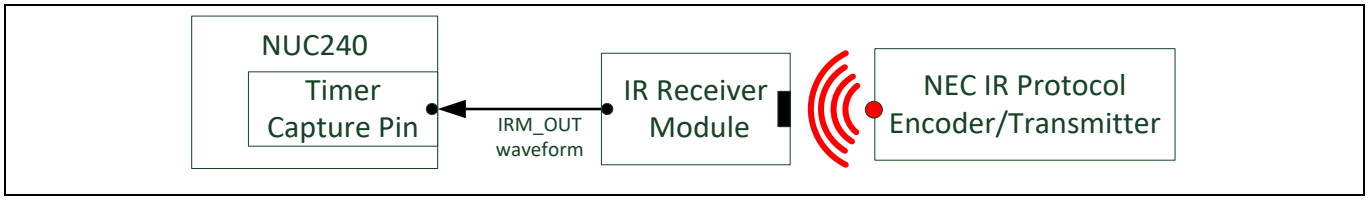


图 1-1 范例示意图

NEC 红外线协议 (NEC IR Protocol) 特性为：

1. 由一个 Leader 讯号表示启动传输，包含 9 ms 的低电位输出的和 4.5 ms 的高电为输出
2. 接下来为 8-bit 的 Address 码和 8-bit 的 Address 反向码
3. 再来为 8-bit 的 Command 码和 8-bit 的 Command 反向码
4. 最后为一个 562.5 us 低电位的 Stop 讯号，表示传输结束

图 1-2 为 NEC IR 协议的数据型态 (NEC_IR) 和 IR 接收器收到讯号后所输出波形 (IRM_OUT) 的示意图。

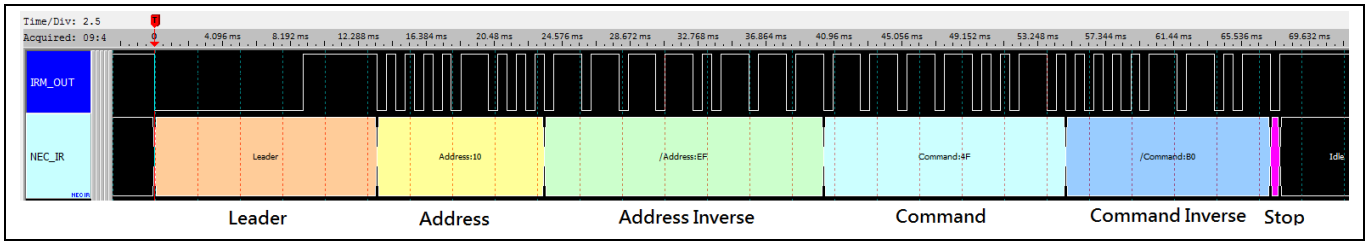


图 1-2 NEC IR 协议和 IR 接收器模块输出波形

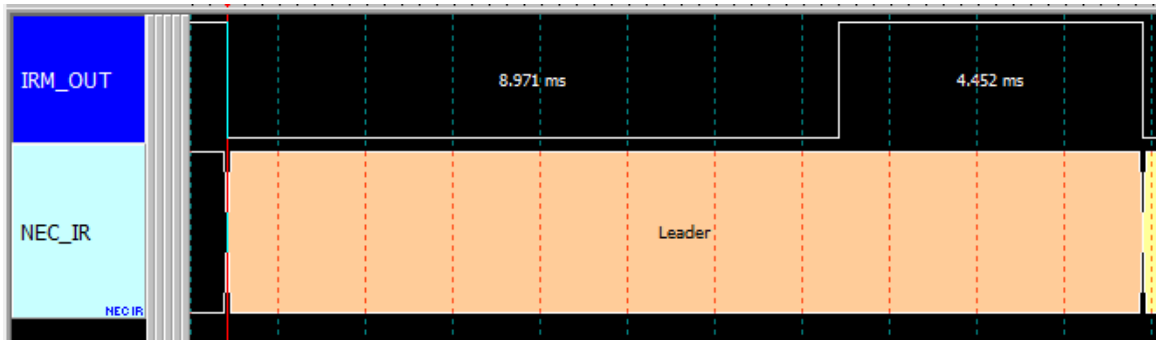
图示 Address/Command 内每一个 bit 的值是由接收到的波形 (IRM_OUT) 状态和长度来决定的。

- 逻辑 0：562.5 us 低电位 + 562.5 us 高电位，总计 1125 us
- 逻辑 1：562.5 us 低电位 + 1687.5 us 高电位，总计 2250 us

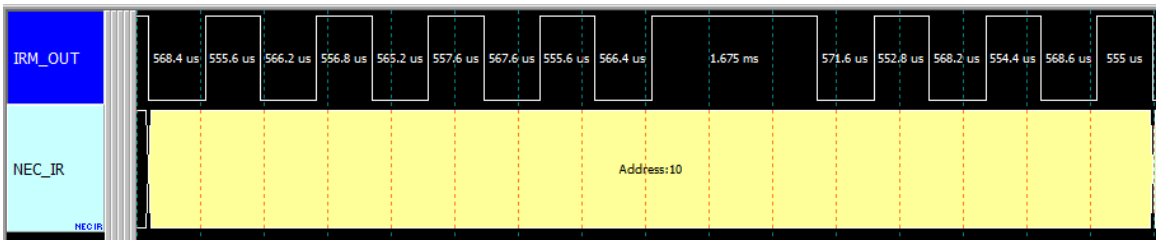
根据以上的说明和图标，当 IR 接收模块收到 IR 信号时，会产生下降沿触发。只要用户可以抓取连续的两个下降沿触发间隔，便可以由这间隔时间知道传入的讯号为何。
因此，此示例代码使用 NUC240 Timer 的撷取功能来得到 IR 接收模块收到的连续下降沿触发间隔时间。

根据图 1-2，NEC IR 协议的每个部分描述如下：

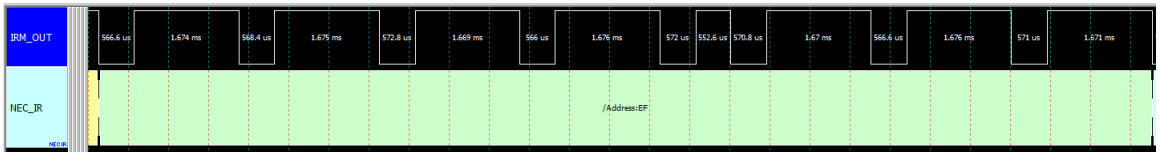
- Leader 讯号间隔



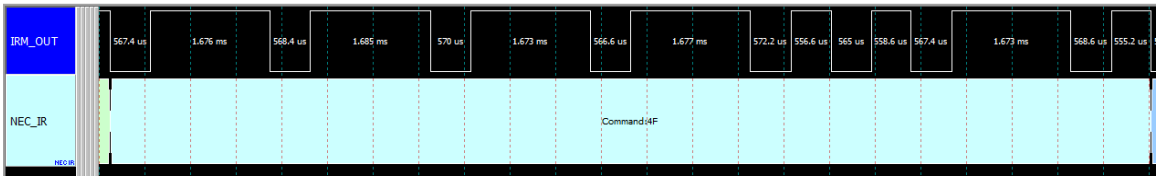
- Address 码 (0x10)



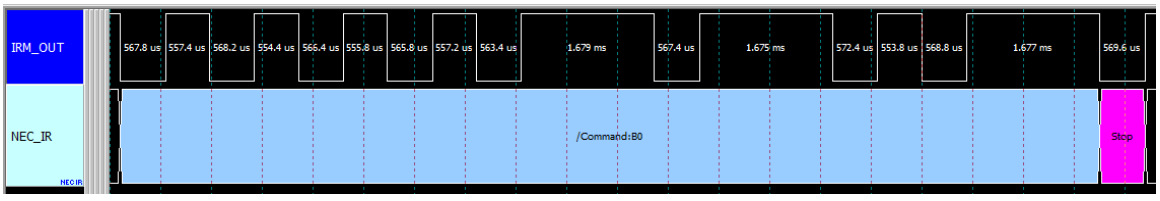
- Address 反向码 (0xEF)



- Command 码 (0x4F)



- Command 反向码 (0xB0) 和 Stop 讯号间隔



1.3 执行结果

程序执行结果可经由串口鲍率 115200 的输出到 PC 做显示，如下图 1-3 所示。

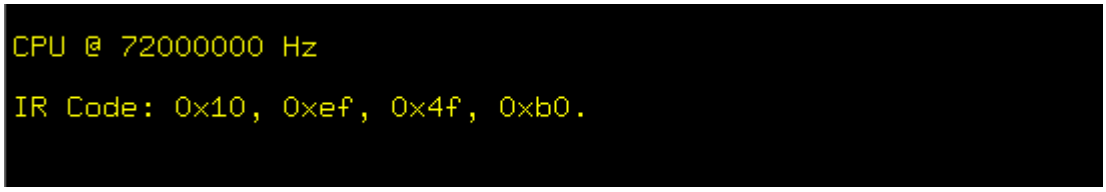


图 1-3 串口执行结果

2 代码介绍

宣告在擷取功能运作时用于储存下降沿的间隔时间和擷取数量的变量。

```
#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( ) 内实作 Timer 2 擷取 PB.2 的下降沿间隔。

```
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，用户需要初始化 Timer 2 撷取功能并配置一个 Timer 2 计数器，其单元为 1 us。在执行 TIMER_Start( ) 后，便可在 TMR2_IRQHandler( ) 内撷取 IR 接收模块收到的讯号间隔。

```
    ……
    /* 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);
    ……
```

收到一组完整的 IR 讯号间隔数据后，便可在 main.c 代码内解析出所有间隔的意义，以获得实际的 IR 代码，包括 Address 码，反向 Address 码，Command 码和反向 Command 码。

```
    ……
    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 软件与硬件环境

- 软件环境
  - BSP 版本
    - NUC230/240 Series BSP CMSIS v3.01.002
  - IDE 版本
    - Keil uVersion 5.28

- 硬件环境
  - 电路组件
    - NuTiny–EVB–NUC240–LQFP100 V1.0
    - IR module, IRM-2638
  - 示意图
    - 将 IR 接收模块输出讯号连接到 NUC240 Timer 2 撷取引脚 (PB.2)，以撷取 IR 讯号间隔时间。
    - 将 NUC240 UART TX (PB.1) 引脚连接到 PC UART RX，以在 PC 上显示示例代码的执行结果。

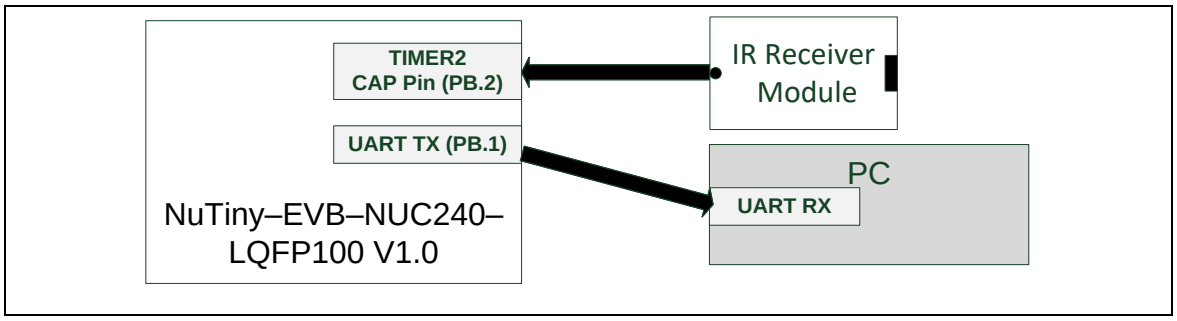


Figure 3-1 将 NUC240 引脚连接到 IR 接收模块输出和 PC UART RX

4 目录信息

- 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 如何执行范例程序

1. 根据目录信息章节进入 ExampleCode 路径中的 KEIL 文件夹，双击 TIMER_Capture_IR.uvproj。
2. 进入编译模式接口
   a. 编译
   b. 下载代码至内存
   c. 进入 / 离开除错模式
3. 进入除错模式接口
   a. 执行代码

6 修订纪录

Date	Revision	Description
Sep. 18, 2019	1.00	1. 初始发布。

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.