NUC505从内部SPI flash拨放WAV檔

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

代码简述	本范例代码从NUC505内部SPI flash读取WAV档案拨放
BSP 版本	NUC505 Series BSP CMSIS V3.02.000
开发平台	NuTiny-EVB-NUC505 V1.6

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 功能介绍

1.1 简介

范例程序中，NUC505使用内部音讯译码器，读取存在内部SPI flash的WAV档并且拨放。使用者需先刻录WAV档至内部SPI flash，并从config.h文件中的WAVFile_address去指定WAV文件储存位置。请注意WAV档案不可超过NUC505内存位置。程序会先判断WAV文件头起始位置是否为RIFF，以及第8字节是否为WAVE，来验证储存是否为WAV檔。

WAV档案存放地址范围	播放WAV档案位置参数
0x2000~0x1F_FFFF	WAVFile_address

NUC505支持单/双声道，数据长度为16 bits格式，取样率为8k, 11025, 12k, 16k, 22050, 24k, 32k, 44100和48k，程序代码会自动依照不同取样率设定APLL数值。

1.2 执行结果

执行后会打印出以下信息

```
Relocate vector table in SRAM (0x20000000) for fast interrupt handling, size = 0xbc.
+------------------------------------------------------------+
|      Play WAV file from internal Flash Sample Code         |
+------------------------------------------------------------+
  Please store WAV file first in the internal Flash.
  WAV file storage location cannot exceed flash size
  NOTE: Need head-phone.

wav.bin pre-stored in SPI Flash address from 0x00002000
wav: sampling rate = 44100
wav: Data size = 864 KB
wav: Channel Num = 2

Configure Internal CODEC ...[OK]
Start Playing ...
Play finish... enter while 1
```

2 代码介绍

首先在main.c中对NUC505做初始化设定，并且把程序代码放入SRAM中执行以增加其执行速度，根据WAVFile_address所设定位置读取WAV文件拨放。

```
int32_t main(void)
{
    /* Init System, IP clock and multi-function I/O */
    SYS_Init();

    /* Init UART0 to 115200-8n1 for print message */
    UART0_Init();

    /* Relocate vector table in SRAM for fast interrupt handling. */
    {
        extern uint32_t __Vectors[];
        extern uint32_t __Vectors_Size[];
        extern uint32_t Image$$ER_VECTOR2$$ZI$$Base[];

        printf("Relocate vector table in SRAM (0x%08X) for fast interrupt handling, size =
0x%x.\n", Image$$ER_VECTOR2$$ZI$$Base, (unsigned int) __Vectors_Size);
        memcpy((void *) Image$$ER_VECTOR2$$ZI$$Base, (void *) __Vectors, (unsigned int)
__Vectors_Size);
        SCB->VTOR = (uint32_t) Image$$ER_VECTOR2$$ZI$$Base;
    }
    /* Init I2S */
    I2S_Init();

    printf("+------------------------------------------------------------+\n");
    printf("|      Play WAV file from internal Flash Sample Code         |\n");
    printf("+------------------------------------------------------------+\n");
    printf("  Please store WAV file first in the internal Flash.\n");
    printf("  WAV file storage location cannot exceed flash size\n");
    printf("  NOTE: Need head-phone. \n\n");

    /* Play wav from internal flash at WAVFile_address */
    ReadFlashPlay(WAVFile_address);

    while (1);
}
```

在ReadFlashPlay(uint32_t FlashAddr)函式中，程序会先判断WAV文件头起始位置是否为RIFF，以及第8字节是否为WAVE，来验证储存是否为WAV檔，并且读取WAV文件相关信息，并根据取样率对APLL做设定。

```
    /* Parsing WAV file header */
    SPIFlash_DMARead(SPIM, FlashAddr, 0, 44, aWavHeader, SPIM_CTL0_CMDCODE_FAST_READ);

    /* Read sampling rate from WAV header */
    u32WavSamplingRate = aWavHeader[6];

    /* Check if the WAV file is correct */
    if (aWavHeader[0] != 0x46464952 | aWavHeader[2] != 0x45564157) {
        printf("WAV File ERROR, please store Wav file at 0x%08x\n", FlashAddr);
        return -1;
    }

    /* Read WAV file total size from WAV header */
    PlayFrameSize = aWavHeader[1] / (PCM_BUFFER_SIZE * 4);
    WavRealSize = aWavHeader[1] / 1024;

    /* Read Channel number from WAV header */
    ChannelNum = (aWavHeader[5] & 0x00030000) >> 16;
    AudioFormat = ChannelNum == 1 ? I2S_MONO : I2S_STEREO;
    /* Check if the MP3 SamplRate frequency is 11025 multiples */
    if (u32WavSamplingRate % 11025) {
        /* APLL is 49152031Hz for 12k, 16k, 24k, 32k, 48k, and 96k sampling rate */
        CLK_SET_APLL(CLK_APLL_49152031);

        if (u32WavSamplingRate == 8000)
            CLK_SetModuleClock(I2S_MODULE, CLK_I2S_SRC_APLL, 1);
    }
```

设定完毕之后使用Ping-pong buffer机制拨放WAV文件，其流程如下：

```
    while (1) {
        /* Set ping-pong buffer to transmit data */
        if ((aPCMBuffer_Full[0] == 1) && (aPCMBuffer_Full[1] == 1)) {
            /* In beginning, wait all buffers full and start play */
            if (!bAudioPlaying) {
                bAudioPlaying = 1;
                /* I2S DMA settings */
```

```
                I2S_SET_TX_TH_LEVEL(I2S, I2S_FIFO_TX_LEVEL_WORD_15);
                I2S_SET_RX_TH_LEVEL(I2S, I2S_FIFO_RX_LEVEL_WORD_16);
                /* Tx Start Address */
                I2S_SET_TXDMA_STADDR(I2S, (uint32_t) &aPCMBuffer[0][0]);
                /* Tx Threshold Address */
                I2S_SET_TXDMA_THADDR(I2S, (uint32_t) &aPCMBuffer[0][PCM_BUFFER_SIZE - 1]);
                /* Tx End Address */
                I2S_SET_TXDMA_EADDR(I2S, (uint32_t) &aPCMBuffer[1][PCM_BUFFER_SIZE - 1]);
                /* Enable setting -> start to play */
                I2S_ENABLE_TXDMA(I2S);
                I2S_ENABLE_TX(I2S);
                /* Enable interrupt : TX DMA Threshold and TX DMA End Interrupt */
                I2S_EnableInt(I2S, (I2S_IEN_TDMATIEN_Msk | I2S_IEN_TDMAEIEN_Msk));
                NVIC_EnableIRQ(I2S_IRQn);
                printf("Start Playing ...\n");
            }
            /* All buffers are full, wait */
            while ((aPCMBuffer_Full[0] == 1) && (aPCMBuffer_Full[1] == 1));
        }
        /* Read WAV data from internal flash */
        SPIFlash_DMARead(SPIM, FlashAddr + 44, 0, PCM_BUFFER_SIZE * 4,
_UNVMP(&aPCMBuffer[u8PCMBufferTargetIdx][0]), SPIM_CTL0_CMDCODE_FAST_READ);
        /* Move address pointer to next address */
        FlashAddr = FlashAddr + PCM_BUFFER_SIZE * 4;
        count_frame++;
        NVIC_DisableIRQ(I2S_IRQn);
        /* Set buffer full flag */
        aPCMBuffer_Full[u8PCMBufferTargetIdx] = 1;
        NVIC_EnableIRQ(I2S_IRQn);
        if (bAudioPlaying) {
            /* Make sure the current ping-pong buffer is empty */
            while (aPCMBuffer_Full[u8PCMBufferTargetIdx ^ 1]) {
                /* Break if the last audio data played */
                if (count_frame >= PlayFrameSize) {
                    goto stop;
                }
            }
        }
        /* Switch to another buffer */
        u8PCMBufferTargetIdx ^= 1;
    }
```

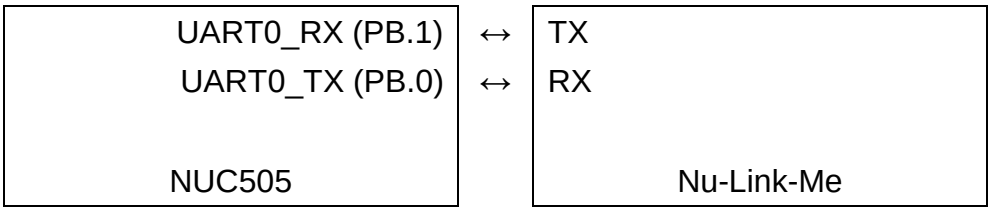
在nuc505_isr.c，当数据传送完毕时清除缓冲区(buffer)满溢旗标

```
void I2S_IRQHandler(void)
{
    uint32_t u32I2SIntFlag;

    /* Check interruput flag */
    u32I2SIntFlag = I2S_GET_INT_FLAG(I2S, I2S_STATUS_TDMATIF_Msk |
I2S_STATUS_TDMAEIF_Msk);

    if (u32I2SIntFlag & I2S_STATUS_TDMATIF_Msk) {
        I2S_CLR_INT_FLAG(I2S, I2S_STATUS_TDMATIF_Msk);

        /* Buffer 0 transfer completed, clear buffer full flag */
        aPCMBuffer_Full[0] = 0;
    } else if (u32I2SIntFlag & I2S_STATUS_TDMAEIF_Msk) {
        I2S_CLR_INT_FLAG(I2S, I2S_STATUS_TDMAEIF_Msk);
        /* Buffer 1 transfer completed, clear buffer full flag */
        aPCMBuffer_Full[1] = 0;
    }
}
```

3 软件与硬件环境

- 软件环境
  - BSP 版本
    - NUC505 Series BSP CMSIS V3.02.000
  - IDE 版本
    - Keil uVersion 5.26
- 硬件环境
  - 电路组件
    - NuTiny-EVB-NUC505 V1.6
    - 3.5 mm 接头耳机
  - 示意图

使用 3.5 mm 接头耳机连接板上的 HP OUT 底座，并透过 NUC505 的 UART0_RX(PB.1)、UART0_TX(PB.0)连接至 Nu-Link Me，打印讯息。设置终端机的 COM Port 与 Baud，COM Port 的编号可在设备管理器中找到「NuBridge Virtual Com Port (COMX)」，Baud 设置为 115200。



4 目录信息

- EC_NUC505_WavPlayerSPIFlash_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 WavPlayerSPIFlash.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
Jul. 03, 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.

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.