M480 UAC1.0 Sample-rate 48kHz, 24-bit and 2-channel on both playback and record ways

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This example code uses M480 MCU and NAU88L25 audio codec device to implement the UAC1.0 based on sample-rate 48kHz, 24-bit data length and 2-channel on both playback and record ways.
BSP Version	M480 Series BSP CMSIS V3.03.001
Hardware	NuMaker-PFM-M487 V3.0

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

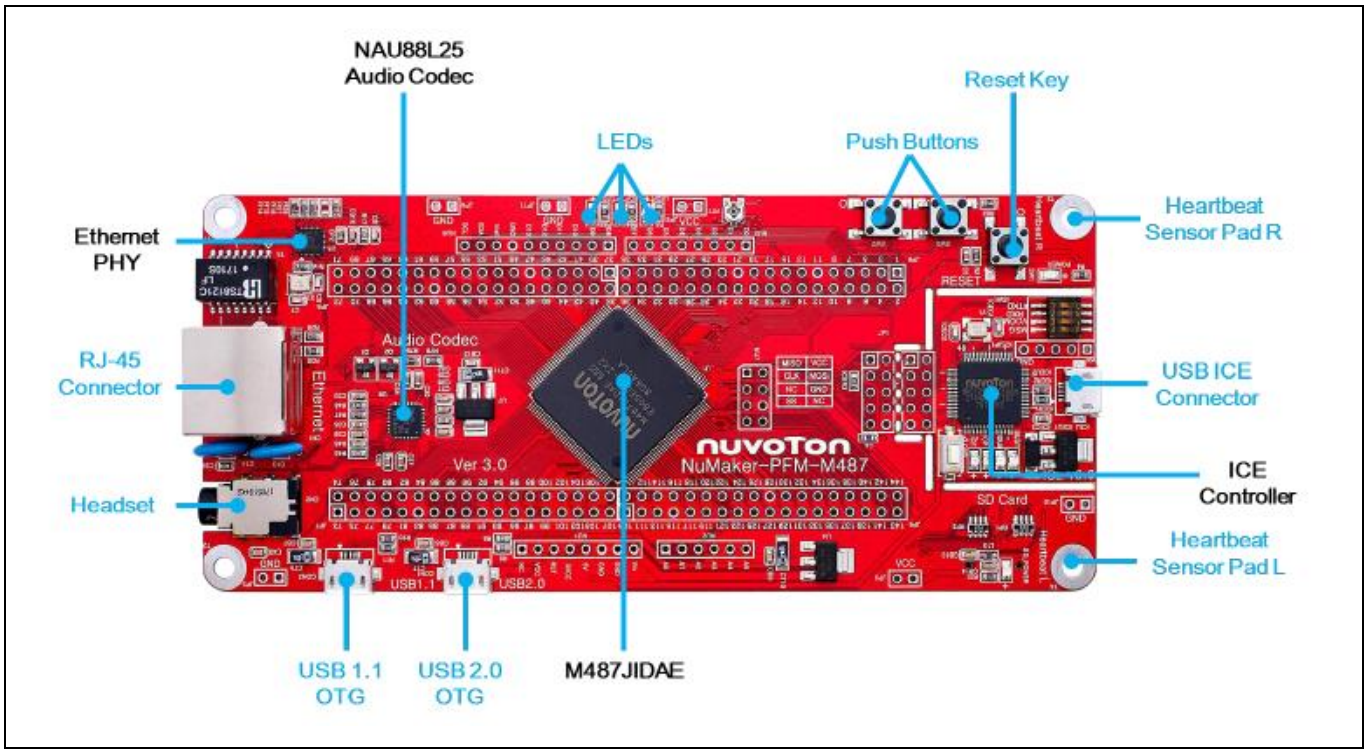
1.1 Introduction

This example code uses M480 MCU and on-board NAU88L25 audio codec device to implement the UAC1.0 (USB Audio Class 1.0) based on sample-rate 48kHz, 24-bit data length and 2-channel on both playback and record ways on the NuMaker-PFM-M487 V3.0 board.

For more information on the NuMaker-PFM-M487 board for the ARM® mbed™ Device Platform, please visit the ARM® mbed™ hardware board website or Nuvoton website to download the User Manual document UM_NuMaker-PFM_M487_User_Manual_EN:

https://developer.mbed.org/platforms/NUMAKER-PFM-M487/

https://www.nuvoton.com/hq/products/microcontrollers/arm-cortex-m4-mcus/User-Manual/?__locale=en&resourcePage=Y&category=&pageIndex=3



NuMaker-PFM-M487 V3.0 Board

1.2 Demo Result

After executed this example code, user can plug-in the USB cable that connect the USB1.1 connector (FS USB OTG) with PC Host and plug-in a headset to the CN2 connector on the NuMaker-PFM-M487 V3.0 board to listen the playing music from PC or to record the voice from the microphone of the headset to the PC recorder.

2 Code Description

This EP3_Handler() function is used to transfer the 24-bit playback PCM data of Left and Right channels from PC Host and store these data to buffers (g_au32PcmPlayBuf[]) at 32-bit format for I2S0 port to transmit to the external NAU88L25 audio codec device.

```
/**
 * @brief       EP3 Handler
 *
 * @param[in]   None
 *
 * @return      None
 *
 * @details     This function is used to process EP3 event
 */
/* Play */
void EP3_Handler(void)
{
    uint32_t u32Len;
    int32_t i;
    uint8_t *pu8Src;
    uint8_t *pu8Buf;
    uint32_t u32Idx;
    uint8_t  Samples;

    /* Get the address in USB buffer */
    pu8Src = (uint8_t *)((uint32_t)USBD_BUF_BASE + USBD_GET_EP_BUF_ADDR(EP3));

    /* Get the byte size of play data */
    u32Len = USBD_GET_PAYLOAD_LEN(EP3);

    /* Calculate samples */
    Samples = u32Len/6;  // 48 = 288/(2ch * 3bytes)

    for(i = 0; i < Samples; i++) {
        /* Check ring buffer turn around */
        u32Idx = g_u32PlayPos_In + 1;
        if(u32Idx >= BUF_LEN)
            u32Idx = 0;

        /* Check if buffer full */
```

```
        if(u32Idx != g_u32PlayPos_Out) {
            /* Update play ring buffer only when it is not full */
            /* Each Word stores a 24-bit Left or Right PCM data */
            pu8Buf = (uint8_t *)&g_au32PcmPlayBuf[u32Idx];
            /* L-data */
            *pu8Buf++ = 0;
            *pu8Buf++ = *pu8Src++;
            *pu8Buf++ = *pu8Src++;
            *pu8Buf++ = *pu8Src++;
            /* R-data */
            *pu8Buf++ = 0;
            *pu8Buf++ = *pu8Src++;
            *pu8Buf++ = *pu8Src++;
            *pu8Buf   = *pu8Src++;

            /* Update IN index */
            u32Idx++;
            g_u32PlayPos_In = u32Idx;
        } else {
            printf("full\n");
        }
    }

    /* Prepare for next OUT packet */
    USBD_SET_PAYLOAD_LEN(EP3, EP3_MAX_PKT_SIZE);

    if(g_u8PlayEn == 0) {
        /* Start play data output through I2S only when we have enough data in buffer */
        if(GetSamplesInBuf() > BUF_LEN / 2)
            g_u8PlayEn = 1;
    }

}
```

This UAC_SendRecData() function is used to transfer the 32-bit record PCM data of Left and Right channels through the data buffers (g_au32PcmRecBuf[]) receiving from the external NAU88L25 audio codec device and store these data into the internal SRAM of Full Speed USB Device at 24-bit format for endpoint 2 (EP2).

```
/**
  * @brief  SendRecData, prepare the record data for next ISO transfer.
```

```
  * @param  None.
  * @retval None.
  */
void UAC_SendRecData(void)
{
    uint8_t *pu8Buf;
    uint8_t *pu8Src;
    int32_t i;

    /* Get the address in USB buffer */
    pu8Buf = (uint8_t *)((uint32_t)USBD_BUF_BASE + USBD_GET_EP_BUF_ADDR(EP2));

    /* Prepare the data to USB IN buffer */
    for(i = 0; i < EP2_MAX_PKT_SIZE/6; i++) {
        pu8Src = (uint8_t *)&g_au32PcmRecBuf[g_u32RecPos_Out];
        /* L-data */
        pu8Src++;
        *pu8Buf = *pu8Src++;
        pu8Buf++;
        *pu8Buf = *pu8Src++;
        pu8Buf++;
        *pu8Buf = *pu8Src++;
        pu8Buf++;
        /* R-data */
        pu8Src++;
        *pu8Buf = *pu8Src++;
        pu8Buf++;
        *pu8Buf = *pu8Src++;
        pu8Buf++;
        *pu8Buf = *pu8Src++;
        pu8Buf++;

        /* Update Out index */
        g_u32RecPos_Out += 2;
        if(g_u32RecPos_Out >= BUF_LEN) g_u32RecPos_Out = 0;
    }
    /* Trigger ISO IN */
    USBD_SET_PAYLOAD_LEN(EP2, EP2_MAX_PKT_SIZE);
}
```

This I2S0_IRQHandler() function is the interrupt service routine of IS20 port and it is used to transmit the playback PCM data to NAU88L25 codec device thru the playback buffers (g_au32PcmPlayBuf[]) by writing the internal FIFO of I2S0 port and receive the record PCM data from NAU88L25 codec device thru the record buffers (g_au32PcmRecBuf[]) by reading the internal FIFO of I2S0 port.

```
/**
  * @brief  I2S0_IRQHandler, I2S0 interrupt service routine to transmit and receive data to/from NAU88L25 codec.
  * @param  None.
  * @retval None.
  */
void I2S0_IRQHandler(void)
{
    uint32_t u32I2SIntFlag;
    uint32_t i, u32Idx;

    u32I2SIntFlag = I2S_GET_INT_FLAG(I2S0, I2S_STATUS0_TXTHIF_Msk | I2S_STATUS0_RXTHIF_Msk);

    if(u32I2SIntFlag & SPI_I2SSTS_TXTHIF_Msk) {
        /* Fill 2 word data when it is TX threshold interrupt */
        for(i = 0; i < 1; i++) {
            /* Check buffer empty */
            if(g_u8PlayEn) {
                if((g_u32PlayPos_Out != g_u32PlayPos_In)) {
                    /* Check ring buffer turn around */
                    u32Idx = g_u32PlayPos_Out + 2;
                    if(u32Idx >= BUF_LEN)
                        u32Idx = 0;

                    /* Transmit data to codec */
                    I2S_WRITE_TX_FIFO(I2S0, g_au32PcmPlayBuf[u32Idx]);
                    I2S_WRITE_TX_FIFO(I2S0, g_au32PcmPlayBuf[u32Idx+1]);

                    /* Update OUT index */
                    g_u32PlayPos_Out = u32Idx;
                } else {
                    /* Fill 0x0 when buffer is empty */
                    I2S_WRITE_TX_FIFO(I2S0, 0x00);
                    I2S_WRITE_TX_FIFO(I2S0, 0x00);

                    /* Buffer underrun. Disable play */
```

```
                    g_u8PlayEn = 0;
                    g_u32PlayPos_In = 0; //BUF_LEN / 2;
                    g_u32PlayPos_Out = 0;
                }
            } else {
                /* Fill 0x0 when buffer is empty */
                I2S_WRITE_TX_FIFO(I2S0, 0x00);
                I2S_WRITE_TX_FIFO(I2S0, 0x00);
            }
        }
    }

    if(u32I2SIntFlag & SPI_I2SSTS_RXTHIF_Msk) {
        if(g_u8RecEn == 1) {
            /* Receive data from codec */
            g_au32PcmRecBuf[g_u32RecPos_In++] = I2S_READ_RX_FIFO(I2S0);
            g_au32PcmRecBuf[g_u32RecPos_In++] = I2S_READ_RX_FIFO(I2S0);

            /* Check In index */
            if(g_u32RecPos_In >= BUF_LEN) g_u32RecPos_In = 0;
        } else {
            I2S_READ_RX_FIFO(I2S0);
            I2S_READ_RX_FIFO(I2S0);
        }
    }
}
```

3 Software and Hardware Environment

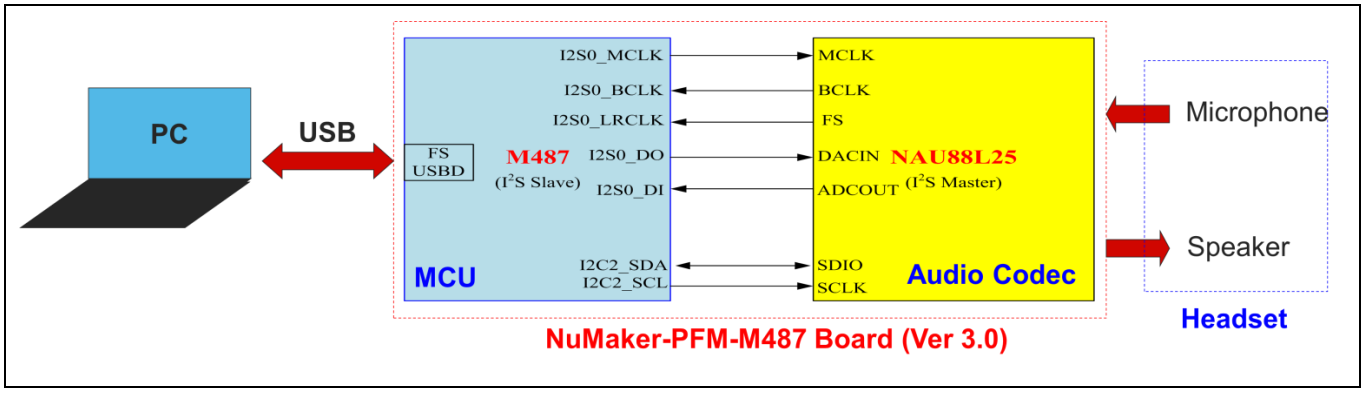
Software Environment

- BSP version
  - M480 Series BSP CMSIS V3.03.001
- IDE version
  - Keil uVersion 4.74

Hardware Environment

- Circuit components
  - NuMaker-PFM-M487 V3.0 board
- Diagram

The M480 MCU uses the I2C2 port to configure the NAU88L25 for I^2S clock settings of I^2S master, I2S0 port to transfer data with NAU88L25 audio codec device and Full Speed USB Device (FS USBD) to transfer PCM streams with PC Host.



Hardware Block Diagram

4 Directory Information

- EC_M480_UAC10_SR48kHz_24bit_2ch_Dual_V1.00
  - Library — Sample code header and source code files
    - CMSIS — Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
    - Device — CMSIS compliant device header files
    - StdDriver — All peripheral driver header and source code files
  - SampleCode
    - ExampleCode — Source code files of this example code

5 How to Execute Example Code

1. Browsing into the sample code folder by Directory Information (section 4) and double click this sample code project M480_UAC10_SR48kHz_24bit_2ch_Dual.uvproj.
2. Enter Keil compile mode.
   a. Build.
   b. Download.
   c. Start/Stop debug session.
3. Enter debug mode.
   a. Run.

6 Revision History

Date	Revision	Description
June 28, 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.