AN0013
Application Note

AT32 USB Audio Development Note

Introduction

This application note introduces the implementation of USB Audio Demo, including the USB transfer types (Control and Isochronous).

This application note is written to help users develop USB devices using Artery AT32Fxx series USB peripherals.

The following AT32Fxx USB related contents are detailed:

- usbd_drivers: the library at USB protocol layer
- usbd_class\audio: application of isochronous transfer to realize USB Speaker and USB MicroPhone

References:

- AT32F403A_407_Firmware_Library\project\at_start_f403a\examples\usb_device
- Sections about Flash memory USB device in RM_AT32F4xx
- Universal Serial Bus Device Class Definition for Audio Devices

Note: The corresponding code in this application note is developed on the basis of V2.x.x BSP provided by Artery. For other versions of BSP, please pay attention to the differences in usage.

Applicable products:

Part number	AT32 MCU with USB

Contents

List of Tables

List of Figures

1 usbd_drivers protocol library

This section mainly introduces the AT32 USB library structure and application method. The AT32 USB is based on USB2.0 full-speed devices and does not support USB2.0 high-speed devices. This library is used to manage USB peripherals and implement USB basic protocol, so that users can start developing faster.

Figure 1. USB application structure

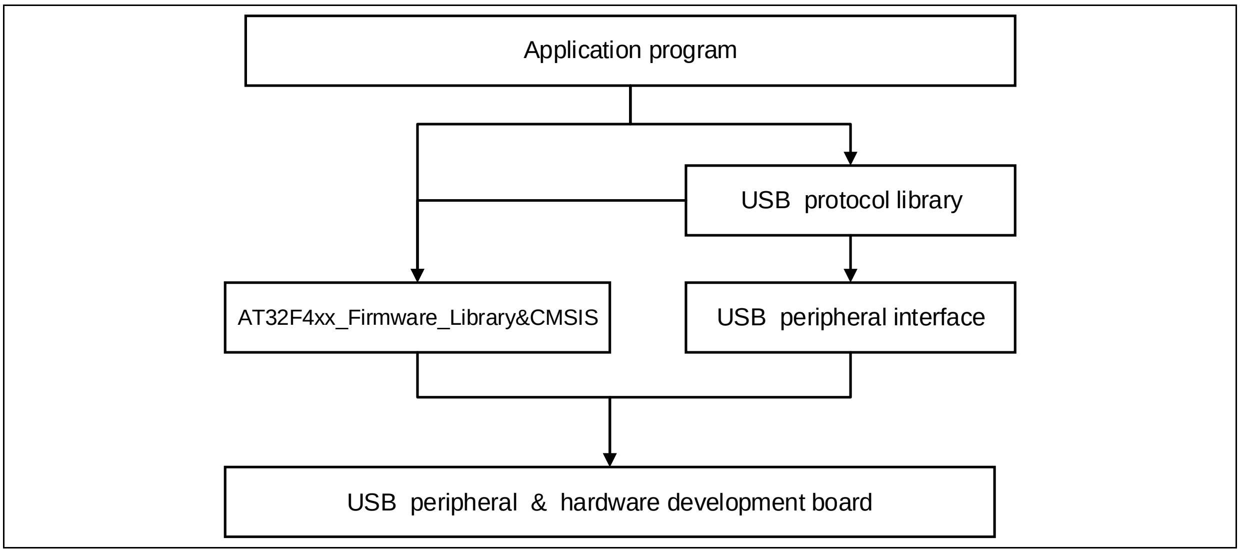
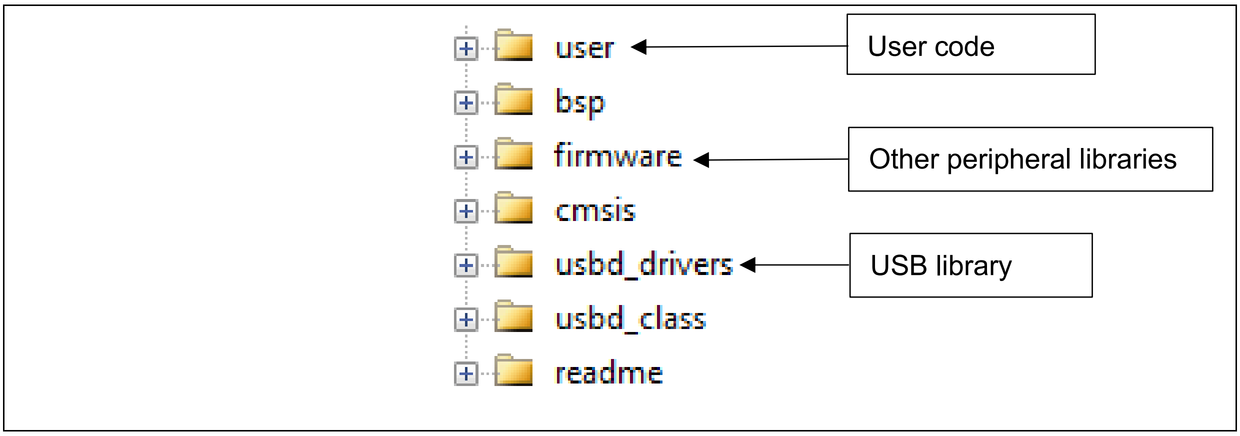

Application program
USB protocol library
AT32F4xx_Firmware_Library&CMSIS
USB peripheral interface
USB peripheral & hardware development board

Figure 2. AT32 project structure



2 AT32 USB library file

Table 1. Files in USB library

File name	Content
usbd_core	Implementation of USB Specification protocol
usbd_int	USB interrupt handler
usbd_sdr	USB standard device request

Table 2. USB user interface files

File name	Content
audio_class	Audio class related handlers
audio_ desc	Implement description of audio device by the application

3 Audio_Composite_Speaker_MicroPhone

The audio routine runs on the AT-START evaluation board, and the Audio Speaker and MicroPhone are implemented based on Audio Arduino Demo Board. The AT-START is connected to Audio Arduino Board during test. For more information, see UM_Audio Arduino Daughter Board_V1.0. Audio device adheres to USB Audio Device 1.0, and please refer to Universal Serial Bus Device Class Define for Audio Device V1.0 for details.

3.1 Function overview

Implement a composite USB audio device that contains a speaker and a MicroPhone to play and record audio simultaneously. The speaker uses isochronous OUT transfer mode, and the MicroPhone uses isochronous IN transfer mode.

Speaker:

- Support 16 K and 48 K sampling rates (only support 16 K sampling rate, by default)
- Support sampling rate switching
- Support 16-bit sampling
- Support mute mode
- Support volume control
- Support Feedback

MicroPhone:

- Support 16 K and 48 K sampling rates (only support 16 K sampling rate, by default)
- Support sampling rate switching
- Support 16-bit sampling
- Support mute mode
- Support volume control

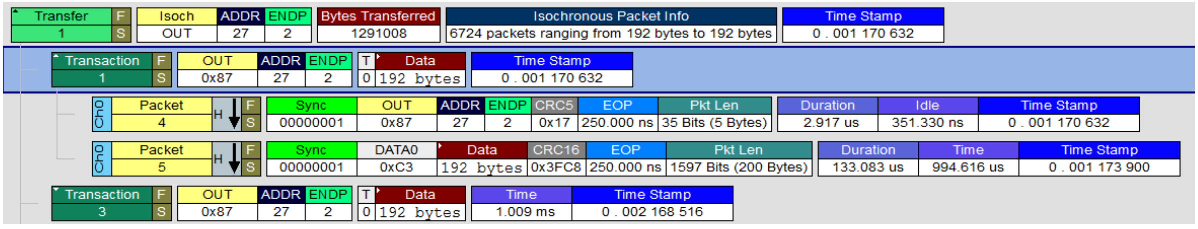
Note: The sampling rate of Speaker and Microphone must be the same; otherwise, the Speaker or the MicroPhone cannot work normally.

3.2 Isochronous transfer

The isochronous transfer mode is mainly used for highly real-time transfer of large amount of data form Host to Device or Device to Host, without handshake packet ACK during transmission.

The isochronous OUT transfer process is shown below:

Figure 3. Isochronous OUT transfer



Transfer	F S	Isoch	ADDR	ENDP	Bytes Transferred	Isochronous Packet Info	Time Stamp
1		OUT	27	2	1291008	6724 packets ranging from 192 bytes to 192 bytes	0 . 001 170 632

Transaction	F S	OUT	ADDR	ENDP	T	Data	Time Stamp
1		0x87	27	2	0	192 bytes	0 . 001 170 632

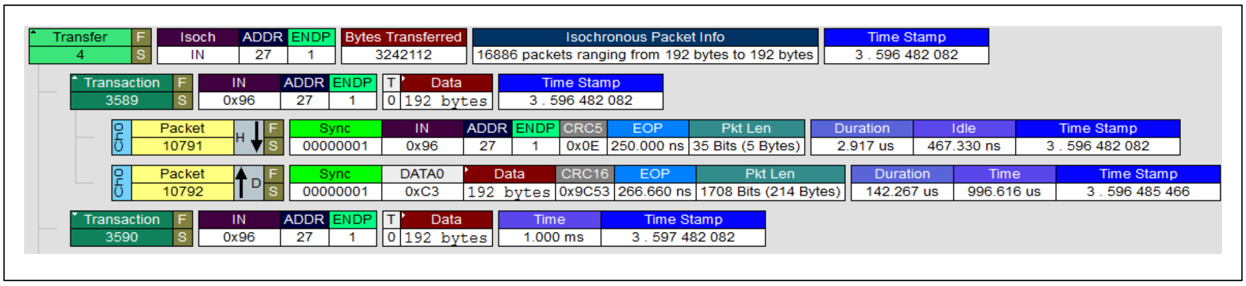
Ch0	Packet	H	F S	Sync	OUT	ADDR	ENDP	CRC5	EOP	Pkt Len	Duration	Idle	Time Stamp
	4			00000001	0x87	27	2	0x17	250.000 ns	35 Bits (5 Bytes)	2.917 us	351.330 ns	0 . 001 170 632

Ch0	Packet	H	F S	Sync	DATA0	Data	CRC16	EOP	Pkt Len	Duration	Time	Time Stamp
	5			00000001	0xC3	192 bytes	0x3FC8	250.000 ns	1597 Bits (200 Bytes)	133.083 us	994.616 us	0 . 001 173 900

Transaction	F S	OUT	ADDR	ENDP	T	Data	Time	Time Stamp
3		0x87	27	2	0	192 bytes	1.009 ms	0 . 002 168 516

The isochronous IN transfer process is shown below:

Figure 4. Isochronous IN transfer



Transfer	F S	Isoch	ADDR	ENDP	Bytes Transferred	Isochronous Packet Info	Time Stamp
4		IN	27	1	3242112	16886 packets ranging from 192 bytes to 192 bytes	3 . 596 482 082

Transaction	F S	IN	ADDR	ENDP	T	Data	Time Stamp
3589		0x96	27	1	0	192 bytes	3 . 596 482 082

Cho	Packet	H	F S	Sync	IN	ADDR	ENDP	CRC5	EOP	Pkt Len	Duration	Idle	Time Stamp
	10791			00000001	0x96	27	1	0x0E	250.000 ns	35 Bits (5 Bytes)	2.917 us	467.330 ns	3 . 596 482 082

Cho	Packet	D	F S	Sync	DATA0	Data	CRC16	EOP	Pkt Len	Duration	Time	Time Stamp
	10792			00000001	0xC3	192 bytes	0x9C53	266.660 ns	1708 Bits (214 Bytes)	142.267 us	996.616 us	3 . 596 485 466

Transaction	F S	IN	ADDR	ENDP	T	Data	Time	Time Stamp
3590		0x96	27	1	0	192 bytes	1.000 ms	3 . 597 482 082

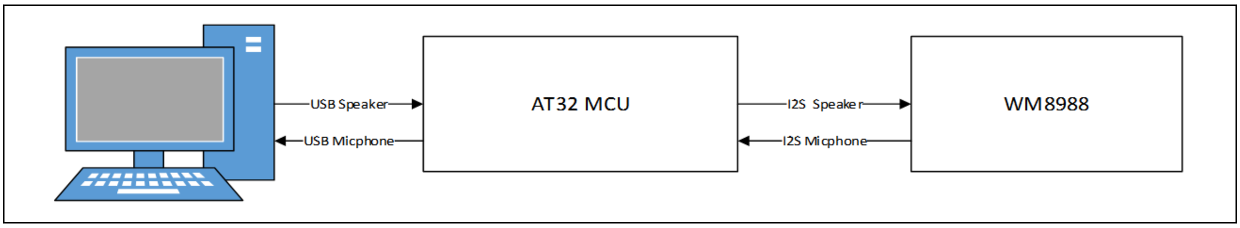
3.3 USB Audio Device

The *Universal Serial Bus Device Class Define for Audio Device* defines the implementation of USB Audio Device and introduces how to implement an audio device and controls (such as sound control).

3.4 AT32 USB Audio Speaker and MicroPhone

AT32 MCU transfers data to PC through USB and communicates with WM8988 through I^{2}S.

Figure 5. AT32 MCU USB Audio



3.5 Operating environment and resources

MCU peripheral resources:

USB:

- Endpoint 0 Control: used for USB enumeration and USB control (volume and sampling rate, etc.)
- Use endpoint 1 as isochronous IN: AT32 MCU sends data to PC; MicroPhone function
- Use endpoint 2 as isochronous OUT: PC sends data to AT32 MCU, Speaker function

I^{2}C:

- Use I^{2}C1 to send control information to audio board

I^{2}S:

- Use I^{2}S1 to send control information to audio board (Speaker)
- Use I^{2}S2 to receive data from audio board (MicroPhone)

DMA1:

- Use DMA1_Channel3 to transmit data of I^{2}S1
- Use DMA1_Channel4 to transmit data of I^{2}S2

TMR1:

- Use TMR1 to generate 12 MHz Clock for audio board

Audio properties:

- Audio format: TYPE I/PCM8 Format /Stereo
- Audio Resolution: 16-bit
- Audio Sample Frequency:16 KHz/48 KHz

Hardware environment:

- Refer to *UM_Audio Arduino Daughter Board_V1.0*

3.6 Software implementation

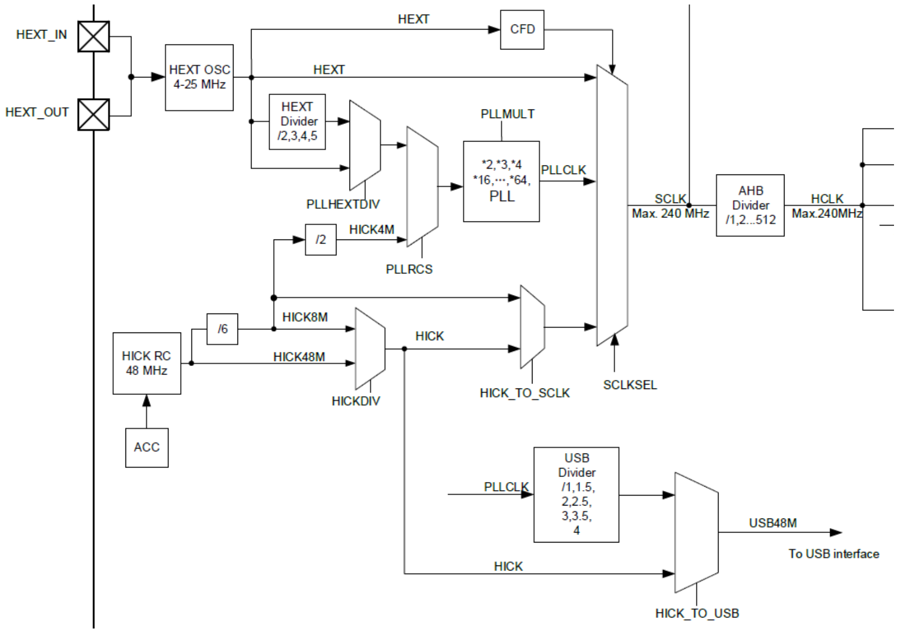
- Initialization of USB peripherals, such as clock configuration, endpoint and endpoint buffer configurations
- Description of USB as a Speaker device
- Description of USB as a MicroPhone device

3.6.1 USB clock configuration

USB 48 MHz clock source:

USB is clocked by a divided PLLCLK, with the division factor being 0, 1, 1.5, 2, 2.5, 3, 3.5 or 4.

Figure 6. USB clocked by PLLCLK



Code implementation:

Configure the division factor for USB according to the system frequency.

Figure 7. USB division factor configuration

```
void usb_clock48m_select(usb_clk48_s clk_s)
{
  if(clk_s == USB_CLK_HICK)
  {
    crm_usb_clock_source_select(CRM_USB_CLOCK_SOURCE_HICK);
```

```
        /* enable the acc calibration ready interrupt */
        crm_periph_clock_enable(CRM_ACC_PERIPH_CLOCK, TRUE);

        /* update the c1\c2\c3 value */
        acc_write_c1(7980);
        acc_write_c2(8000);
        acc_write_c3(8020);

        /* open acc calibration */
        acc_calibration_mode_enable(ACC_CAL_HICKTRIM, TRUE);
    }
    else
    {
        switch(system_core_clock)
        {
            /* 48MHz */
            case 48000000:
                crm_usb_clock_div_set(CRM_USB_DIV_1);
                break;

            /* 72MHz */
            case 72000000:
                crm_usb_clock_div_set(CRM_USB_DIV_1_5);
                break;
         …/* other main frequencies */
            default:
                break;

        }
    }
}
```

3.6.2 USB endpoint initialization

Initialize the endpoints to be used, including the endpoint type and the maximum length supported. The endpoint initialization is performed after the device receives a Reset signal from the host.

Figure 8. Endpoint initialization

```
usb_sts_type class_init_handler(void *udev)
{
    usb_sts_type status = USB_OK;
    usbd_core_type *pudev = (usbd_core_type *)udev;


    /* enable microphone in endpoint double buffer mode */
    usbd_ept_dbuffer_enable(pudev, USBD_AUDIO_MIC_IN_EPT);
    /* open microphone in endpoint */
    usbd_ept_open(pudev, USBD_AUDIO_MIC_IN_EPT, EPT_ISO_TYPE,
AUDIO_MIC_IN_MAXPACKET_SIZE);

    /* enable speaker out endpoint double buffer mode */
    usbd_ept_dbuffer_enable(pudev, USBD_AUDIO_SPK_OUT_EPT);

    /* open speaker out endpoint */
    usbd_ept_open(pudev, USBD_AUDIO_SPK_OUT_EPT, EPT_ISO_TYPE,
AUDIO_SPK_OUT_MAXPACKET_SIZE);

    /* enable speaker feedback endpoint double buffer mode */
    usbd_ept_dbuffer_enable(pudev, USBD_AUDIO_FEEDBACK_EPT);

    /* open speaker feedback endpoint */
    usbd_ept_open(pudev, USBD_AUDIO_FEEDBACK_EPT, EPT_ISO_TYPE,
AUDIO_FEEDBACK_MAXPACKET_SIZE);

    /* start receive speaker out data */
    usbd_ept_recv(pudev, USBD_AUDIO_SPK_OUT_EPT, audio_struct.audio_spk_data,
AUDIO_SPK_OUT_MAXPACKET_SIZE);

    return status;
}
```

```
#define EPT0_TX_ADDR                    0x40      /*!< usb endpoint 0 tx buffer address offset */
#define EPT0_RX_ADDR                    0x80      /*!< usb endpoint 0 rx buffer address offset */

#define EPT1_TX_ADDR                    0xC0      /*!< usb endpoint 1 tx buffer address offset */
#define EPT1_RX_ADDR                    0x100     /*!< usb endpoint 1 rx buffer address offset */

#define EPT2_TX_ADDR                    0x140     /*!< usb endpoint 2 tx buffer address offset */
#define EPT2_RX_ADDR                    0x180     /*!< usb endpoint 2 rx buffer address offset */
```

3.6.3 Speaker and MicroPhone configurations

Implement description of the corresponding device, and the host can get the specific functions of devices through the description. Users can select the Speaker or MicroPhone or both functions as required, and the description is modified according to configurations.

Function configuration

Configure the required functions through audio_conf.h, including whether to implement Speaker and MicroPhone, and whether to support 16 KHz and 48 KHz sampling rates. The configuration code is as follows:

Figure 9. Function configuration

```
#define AUDIO_SUPPORT_SPK                       1
#define AUDIO_SUPPORT_MIC                       1
#define AUDIO_SUPPORT_FEEDBACK                  1
/* Whether or not to support Speaker, Micrphone, Feedback */
#define AUDIO_SUPPORT_FREQ_16K                  1
#define AUDIO_SUPPORT_FREQ_48K                  1
/*Whether or not to support 16 K or 48 K sampling rate*/
#define AUDIO_SUPPORT_FREQ                (AUDIO_SUPPORT_FREQ_16K + \
                                           AUDIO_SUPPORT_FREQ_48K \   )

#define AUDIO_FREQ_16K                          16000
#define AUDIO_FREQ_48K                          48000
#define AUDIO_BITW_16                           16
/*Current default configuration is as follows */
#define AUDIO_MIC_CHANEL_NUM                    2
#define AUDIO_MIC_DEFAULT_BITW                  AUDIO_BITW_16

#define AUDIO_SPK_CHANEL_NUM                    2
#define AUDIO_SPK_DEFAULT_BITW                  AUDIO_BITW_16

#define AUDIO_SUPPORT_MAX_FREQ                  48
#define AUDIO_DEFAULT_FREQ                      AUDIO_FREQ_48K
#define AUDIO_DEFAULT_BITW                      AUDIO_BITW_16
```

Audio device description->usbd_desc.c

Table 3. Audio device description->usbd_desc.c

Field	Value	Description
bLength	0x12	Size of this descriptor in bytes
bDescriptorType	0x01	DEVICE Descriptor type
bcdUSB	0x0200	USB Specification Release Number
bDeviceClass	0x00	Device defined at interface level
bDeviceSubClass	0x00	Unused
bDeviceProtocol	0x00	Unused
bMaxPacketSize0	0x40	Maximum packet size for endpoint 0 0x40 bytes
idVendor	0x2E3C	Vendor ID
idProduct	0x5730	Product ID
bcdDevice	0x0200	Device release number
iManufacturer	0x01	Index of string descriptor describing Manufacturer
iProduct	0x02	Index of string descriptor describing product
iSerialNumber	0x03	Index of string descriptor describing the device serial number
bNumConfigurations	0x01	Number of possible configurations

Audio device description->usbd_desc.c

Table 4. Audio configuration description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor in bytes
bDescriptorType	0x02	CONFIGURATION Descriptor type
wTotalLength	0xC0 (Note 1)	Total length of data returned for this configuration
bNumberInterface	0x02 (Note 2)	Number of interfaces supported of this configuration
bConfigurationValue	0x01	Value to use as SetConfiguration() argument
iConfiguration	0x00	Index of string descriptor describing this configuration
bmAttributes	0xc0	Configuration characteristics
bMaxPower	0x32	bMaxPower

Note 1: The wTotalLength varies depending on the configuration (sampling rate supported; whether or not to support Microphone/Speaker).

Note 2: The bNumberInterface varies depending on the function to be implemented (whether or not to support Micpone/Speaker).

Standard AC interface description->usbd_desc.c

Table 5. Standard AC interface description->usbd_desc.c

Field	Value	Description
bLength	0x0A	Size of this descriptor in bytes
bDescriptorType	0x04	INTERFACE Descriptor type
bInterfaceNumber	0x00	Number of this interface
bAlternateSetting	0x00	Index of this setting
bNumEndpoints	0x00	0 endpoint
bInterfaceClass	0x01	AUDIO

Field	Value	Description
bInterfaceSubClass	0x01	AUDIO CONTROL
bInterfaceProtocol	0x00	Protocol code(assigned by the USB)
iInterface	0x00	Index of string descriptor describing this interface

Audio Class-specific AC interface description->usbd_desc.c

Table 6. Audio Class-specific AC interface description->usbd_desc.c

Field	Value	Description
bLength	0x0A	Size of this descriptor
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x01	HEADER descriptor subtype
bcdADC	0x0100	Audio Device Class Specification Release Number
wTotalLength	0x46	Total number of bytes returned for class-specific
bIncollection	0x02	The number of AudioStreaming and MIDIStreaming interfaces
baInterfaceNr	0x02	Interface number of the first AudioStreaming
baInterfaceNr	0x01	Interface number of the second AudioStreaming

MicroPhone Input Terminal description->usbd_desc.c

Table 7. MicroPhone Input Terminal description->usbd_desc.c

Field	Value	Description
bLength	0x0C	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x02	INPUT_TERMINAL descriptor subtype
bTerminalID	0x01	ID of this Input Terminal
wTerminalType	0x0201	Terminal is Microphone
bAssocTerminal	0x00	No association
bNrChannels	0x02	Two channel
wChannelConfig	0x0003	Stereo
iChannelNames	0x00	Unused
iTerminal	0x00	Unused

MicroPhone Feature Unit description->usbd_desc.c

Table 8. MicroPhone Feature Unit description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x06	FEATURE Unit ID descriptor subtype
bUnitID	0x02	ID of this Unit ID
bSourceID	0x01	Input Terminal ID
bControlSize	0x01	Control size
bmaControl	0x01	MUTE is Support
bmaControl	0x02	Volume is Support
iTerminal	0x00	Unused

MicroPhone Output Terminal description->usbd_desc.c

Table 9. MicroPhone Output Terminal description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x03	OUTPUT_TERMINAL descriptor subtype
bTerminalID	0x03	ID of this Output Terminal
wTerminalType	0x0101	USB Streaming
bAssocTerminal	0x00	Unused
bSourceID	0x02	Feature Unit ID
iTerminal	0x00	Unused

Speaker Input Terminal description->usbd_desc.c

Table 10. Speaker Input Terminal description->usbd_desc.c

Field	Value	Description
bLength	0x0C	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x02	INPUT_TERMINAL descriptor subtype
bTerminalID	0x04	ID of this Input Terminal
wTerminalType	0x0101	USB Streaming
bAssocTerminal	0x00	No association
bNrChannels	0x02	Two channel
wChannelConfig	0x0003	Stereo
iChannelNames	0x00	Unused
iTerminal	0x00	Unused

Speaker Feature Unit description->usbd_desc.c

Table 11. Speaker Feature Unit description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x06	FEATURE Unit ID descriptor subtype
bUnitID	0x05	ID of this Unit ID
bSourceID	0x04	Input Terminal ID
bControlSize	0x01	Control size
bmaControl	0x01	MUTE is Support
bmaControl	0x02	Volume is Support
iTerminal	0x00	Unused

Speaker Output Terminal description->usbd_desc.c

Table 12. Speaker Output Terminal description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x03	OUTPUT_TERMINAL descriptor subtype
bTerminalID	0x06	ID of this Output Terminal
wTerminalType	0x0301	Speaker
bAssocTerminal	0x00	Unused
bSourceID	0x05	Feature Unit ID
iTerminal	0x00	Unused

MicroPhone Standard AS interface description->usbd_desc.c

Table 13. MicroPhone Standard AS interface description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x04	CS_INTERFACE descriptor type
bInterfaceNumber	0x01	Index of this interface
bAlternateSetting	0x00	Index of this alternate setting
bNumEndpoint	0x00	0 endpoint
bInterfaceClass	0x01	AUDIO
bInterfaceSubClass	0x02	AUDIO_STREAMING
bInterfaceProtocol	0x00	Unused
iInterface	0x00	Unused

MicroPhone Standard AS interface description->usbd_desc.c

Table 14. MicroPhone Standard AS interface description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x04	CS_INTERFACE descriptor type
bInterfaceNumber	0x01	Index of this interface
bAlternateSetting	0x01	Index of this alternate setting
bNumEndpoint	0x01	1 endpoint
bInterfaceClass	0x01	AUDIO
bInterfaceSubClass	0x02	AUDIO_STREAMING
bInterfaceProtocol	0x00	Unused
iInterface	0x00	Unused

MicroPhone Class-Specific AS Interface description->usbd_desc.c

Table 15. MicroPhone Class-Specific AS Interface description->usbd_desc.c

Field	Value	Description
bLength	0x07	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x01	GENERAL subtype
bTerminalLink	0x03	Unit ID of the Output Terminal
bDelay	0x01	Interface delay
wFormatTag	0x0001	PCM Format

MicroPhone Type I Format type description->usbd_desc.c

Table 16. MicroPhone Type I Format type description->usbd_desc.c

Field	Value	Description
bLength	0x0E	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x02	FORMAT_TYPE subtype
bFormatType	0x01	FORMAT_TYPE_I
bNrChannels	0x02	Two Channel
bSubFrameSize	0x02	Two bytes per audio subframe
bBitResolution	16	16 bits per sample
bSamFreqType	2	Two frequency supported
tSamFreq	0x003E80	16000HZ
tSamFreq	0x00BB80	48000HZ

MicroPhone Standard Endpoint description->usbd_desc.c

Table 17. MicroPhone Standard Endpoint description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x05	ENDPOINT descriptor
bEndpointAddress	0x81	IN Endpoint 1
bmAttributes	0x05	Isochronous
wMaxPacketSize	0x0120	288 bytes per packet
bInterval	0x01	One packet per frame
bRefresh	0x00	Unused
bSynchAddress	0x00	Unused

MicroPhone Class-specific iso Endpoint description->usbd_desc.c

Table 18. MicroPhone Class-specific iso Endpoint description->usbd_desc.c

Field	Value	Description
bLength	0x07	Size of this descriptor, in bytes.
bDescriptorType	0x25	ENDPOINT descriptor
bDescriptorSubtype	0x01	GENERAL subtype
bmAttributes	0x00	No sampling frequency control
bLockDelayUnits	0x00	Unused
wLockDelay	0x0000	Unused

Speaker Standard AS interface description->usbd_desc.c

Table 19. Speaker Standard AS interface description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x04	CS_INTERFACE descriptor type
bInterfaceNumber	0x02	Index of this interface
bAlternateSetting	0x00	Index of this alternate setting
bNumEndpoint	0x00	0 endpoint
bInterfaceClass	0x01	AUDIO
bInterfaceSubClass	0x02	AUDIO_STREAMING
bInterfaceProtocol	0x00	Unused
iInterface	0x00	Unused

Speaker Standard AS interface description->usbd_desc.c

Table 20. Speaker Standard AS interface description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x04	CS_INTERFACE descriptor type
bInterfaceNumber	0x02	Index of this interface
bAlternateSetting	0x01	Index of this alternate setting
bNumEndpoint	0x02	2 endpoint, feed back
bInterfaceClass	0x01	AUDIO
bInterfaceSubClass	0x02	AUDIO_STREAMING
bInterfaceProtocol	0x00	Unused
iInterface	0x00	Unused

Speaker Class-Specific AS Interface description->usbd_desc.c

Table 21. Speaker Class-Specific AS Interface description->usbd_desc.c

Field	Value	Description
bLength	0x07	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x01	GENERAL subtype
bTerminalLink	0x04	ID
bDelay	0x01	Interface delay
wFormatTag	0x0001	PCM Format

Speaker Type I Format type description->usbd_desc.c

Table 22. Speaker Type I Format type description->usbd_desc.c

Field	Value	Description
bLength	0x0E	Size of this descriptor, in bytes.
bDescriptorType	0x24	CS_INTERFACE descriptor type
bDescriptorSubtype	0x02	FORMAT_TYPE subtype
bFormatType	0x01	FORMAT_TYPE_I
bNrChannels	0x02	Two Channel
bSubFrameSize	0x02	Two bytes per audio subframe
bBitResolution	16	16 bits per sample
bSamFreqType	2	Two frequency supported
tSamFreq	0x003E80	16000HZ
tSamFreq	0x00BB80	48000HZ

Speaker Standard Endpoint description->usbd_desc.c

Table 23. Speaker Standard Endpoint description->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x05	ENDPOINT descriptor
bEndpointAddress	0x02	OUT Endpoint 2
bmAttributes	0x05	Isochronous
wMaxPacketSize	0x0120	288 bytes per packet
bInterval	0x01	One packet per frame
bRefresh	0x00	Unused
bSynchAddress	0x83	Feed back endpoint 3

Speaker Class-specific iso Endpoint description->usbd_desc.c

Table 24. Speaker Class-specific iso Endpoint description->usbd_desc.c

Field	Value	Description
bLength	0x07	Size of this descriptor, in bytes.
bDescriptorType	0x25	ENDPOINT descriptor
bDescriptorSubtype	0x01	GENERAL subtype
bmAttributes	0x00	No sampling frequency control
bLockDelayUnits	0x00	Unused
wLockDelay	0x0000	Unused

Speaker Feed back Endpoint description->usbd_desc.c

Table 25. Speaker Feed back Endpoint description ->usbd_desc.c

Field	Value	Description
bLength	0x09	Size of this descriptor, in bytes.
bDescriptorType	0x05	ENDPOINT descriptor
bEndpointAddress	0x83	IN Endpoint 3
bmAttributes	0x11	Isochronous
wMaxPacketSize	0x0003	3 bytes per packet
bInterval	0x01	One packet per frame
bRefresh	0x08	Refresh time
bSynchAddress	0x00	Unused

3.6.4 MicroPhone data handling

Microphone data is sent from USB device to the HOST, at the data transmission rate of 1 frame/ms, and the size of one frame of data is determined by the sampling rate and bit width (formula: size of one frame of data = (sampling rate / 1000) * (bit width / 8) * number of channels). Demo performs IN transmission through endpoint 1.

The software implementation process is as follows:

Figure 10. MicroPhone data handling

```
usb_sts_type class_in_handler(void *udev, uint8_t ept_num)
{
  usb_sts_type status = USB_OK;
  uint32_t len = 0;

  /* ...user code...
     trans next packet data
  */
  if((ept_num & 0x7F) == (USBD_AUDIO_MIC_IN_EPT & 0x7F)) /*if endpoint 1 */
  {
     len = audio_codec_mic_get_data(audio_struct.audio_mic_data); /*Get the data length of the
 current frame */
```

```
        usbd_ept_send(udev, USBD_AUDIO_MIC_IN_EPT, audio_struct.audio_mic_data, len);
/*Send data to the HOST*/
    }

    else if((ept_num & 0x7F) == (USBD_AUDIO_FEEDBACK_EPT & 0x7F)) /*if endpoint 3 */
    {
        len = audio_codec_spk_feedback(audio_struct.audio_feed_back); /* Get the data length of
the current frame */
        usbd_ept_send(udev, USBD_AUDIO_FEEDBACK_EPT, audio_struct.audio_feed_back,
len); /*Send back the current feedback*/
    }

    return status;
}
```

3.6.5 Speaker data handling

Speaker data is sent from the HOST to the device, at the data transmission rate of 1 frame/ms, and the size of one frame of data is determined by the sampling rate and bit width (formula: size of one frame of data = (sampling rate / 1000) * (bit width / 8) * number of channels). Demo performs OUT transmission through endpoint 2. Due to the clock asynchronization, the Speaker has a Feedback feature that can reflect the actual sampling rate over a period of time. Then, the host can adjust the rate of data transmission according to the actual sampling rate fed back by the device. For details, see class_in_handler(void *udev, uint8_t ept_num).

The software implementation process is as follows:

Figure 11. Speaker data handling

```
usb_sts_type class_out_handler(void *udev, uint8_t ept_num)
{
    usb_sts_type status = USB_OK;
    usbd_core_type *pudev = (usbd_core_type *)udev;
    uint16_t g_rxlen;

    /* get endpoint receive data length    */
    g_rxlen = usbd_get_recv_len(pudev, ept_num);

    if((ept_num & 0x7F) == (USBD_AUDIO_SPK_OUT_EPT & 0x7F)) /*if endpoint 2 */
    {
        /* speaker data*/
        audio_codec_spk_fifo_write(audio_struct.audio_spk_data, g_rxlen); /*write data to the
playback buffer*/

```

```
        /* get next data */
        usbd_ept_recv(pudev, USBD_AUDIO_SPK_OUT_EPT, audio_struct.audio_spk_data,
AUDIO_SPK_OUT_MAXPACKET_SIZE); /*get Speaker data*/
    }

    return status;
}
```

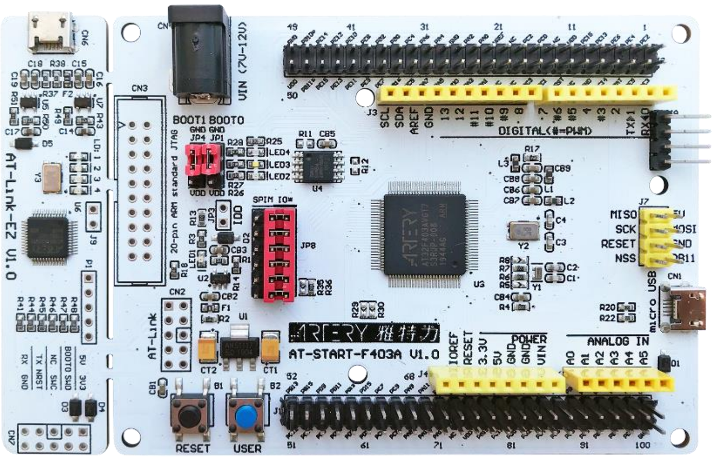
4 USB Audio test

This section introduces how to use USB Audio Demo for audio playback and recording.

4.1 Hardware resources

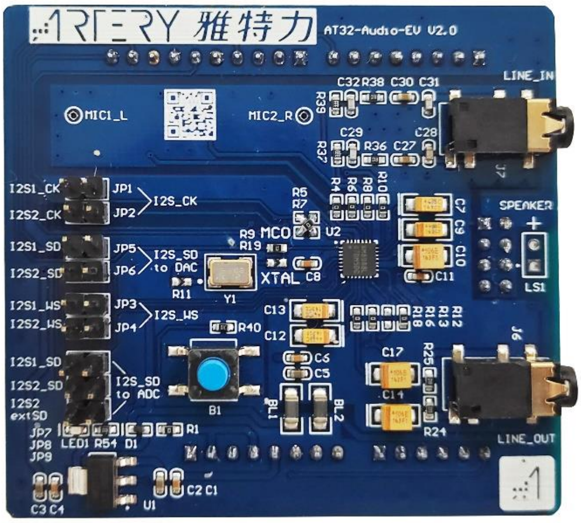
- LED2/LED3/LED4
- USB(PA11/PA12)
- AT-START-F403A V1.0 demo board
- Earphone

Figure 12. AT-START-F403A V1.0 demo board



- AT32-Audio-EV V2.0

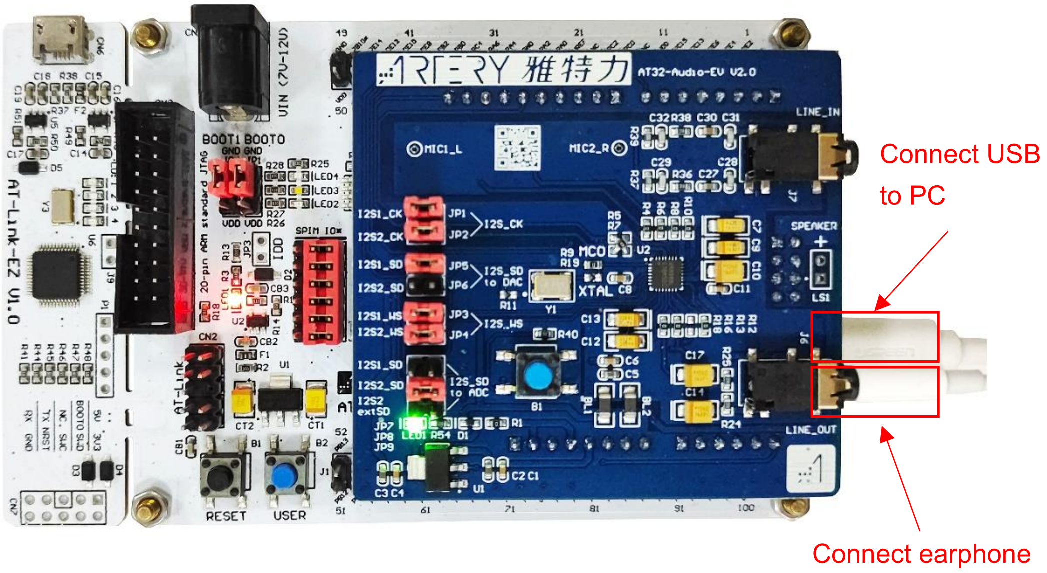
Figure 13. AT32-Audio-EV V2.0



4.2 Test procedures

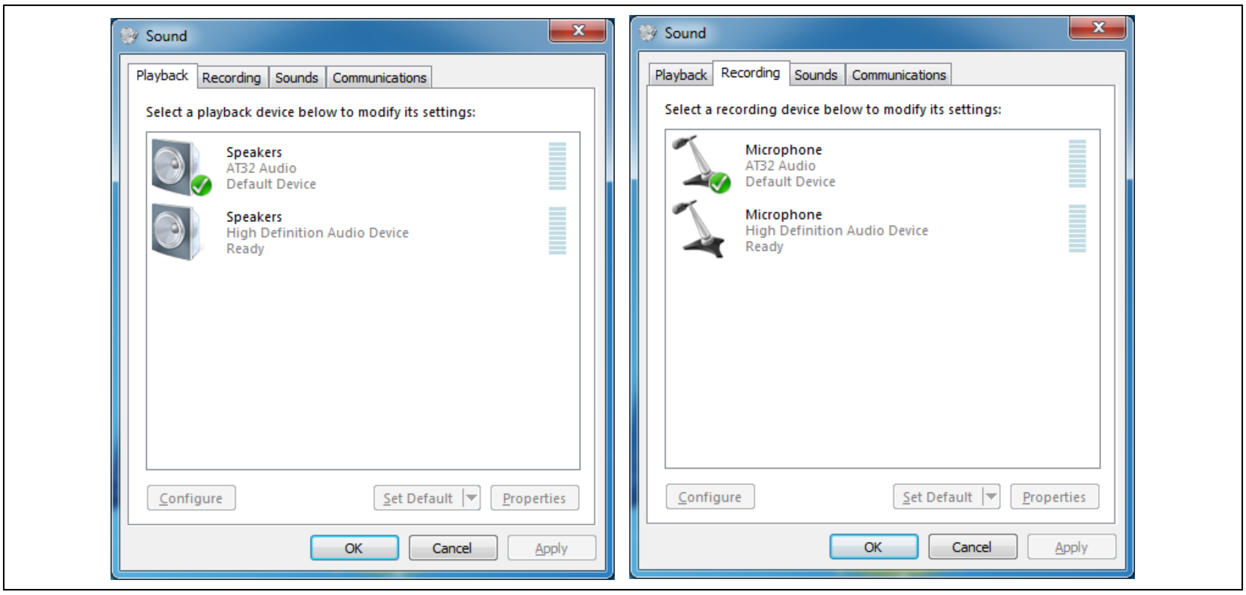
Connect AT-START-F403A V1.0 and AT32-Audio-EV V2.0;
Download Audio Demo to the demo board (project path: AN0013_SourceCode_V2.0.0\utilities\AN0013_demo)
Connect USB to PC Host (test environment: WIN7)

Figure 14. Connect USB to PC Host



- If PC successfully recognizes the Audio device, AT32 Audio devices (a speaker and a microphone) will be found in the sound controller.

Figure 15. PC recognizes the speaker and microphone



Sound

Select a playback device below to modify its settings:

Speakers
AT32 Audio
Default Device

Speakers
High Definition Audio Device
Ready

Sound

Select a recording device below to modify its settings:

Microphone
AT32 Audio
Default Device

Microphone
High Definition Audio Device
Ready

4.3 Audio playback

Open Windows Media Player (or other media players) and play a piece of music, and then users can hear the sound through earphones.

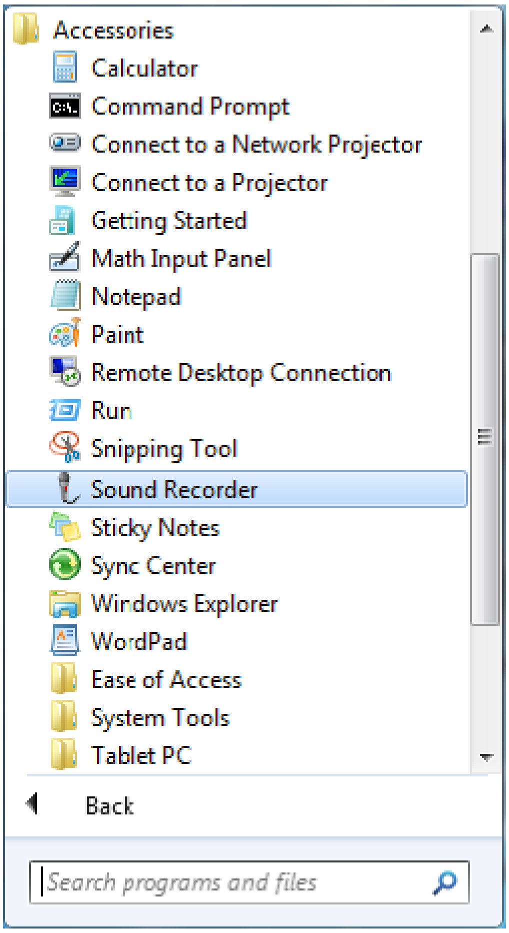
Figure 16. Audio playback

4.4 Audio recording test

Open the Windows “Sound recorder” to conduct test. The test procedures are as follows:

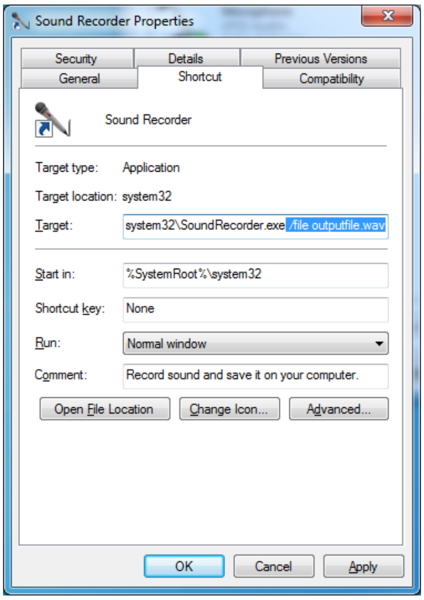
- Select Windows->Accessories->Sound recorder; right click->Property

Figure 17. Open sound recorder



- Add "/file outputfile.wav" to the target (T), and then click OK.

Figure 18. Sound recorder property configuration



Sound Recorder

Target type: Application

Target location: system32

Target: system32\SoundRecorder.exe /file outputfile.wav

Start in: %SystemRoot%\system32

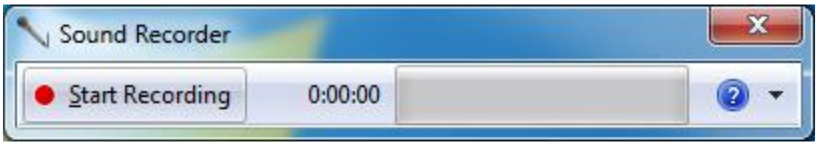
Shortcut key: None

Run: Normal window

Comment: Record sound and save it on your computer.

- Open sound recorder->start recording

Figure 19. Start recording



Save and then play the recorded audio.

5 Revision history

Table 26. Document revision history

Date	Version	Revision note
2022.01.17	2.0.0	Initial release

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Purchasers are solely responsible for the selection and use of ARTERY's products and services; ARTERY assumes no liability for purchasers' selection or use of the products and the relevant services.

No license, express or implied, to any intellectual property right is granted by ARTERY herein regardless of the existence of any previous representation in any forms. If any part of this document involves third party's products or services, it does NOT imply that ARTERY authorizes the use of the third party's products or services, or permits any of the intellectual property, or guarantees any uses of the third party's products or services or intellectual property in any way.

Except as provided in ARTERY's terms and conditions of sale for such products, ARTERY disclaims any express or implied warranty, relating to use and/or sale of the products, including but not restricted to liability or warranties relating to merchantability, fitness for a particular purpose (based on the corresponding legal situation in any unjudicial districts), or infringement of any patent, copyright, or other intellectual property right.

ARTERY's products are not designed for the following purposes, and thus not intended for the following uses: (A) Applications that have specific requirements on safety, for example: life-support applications, active implant devices, or systems that have specific requirements on product function safety; (B) Aviation applications; (C) Aerospace applications or environment; (D) Weapons, and/or (E) Other applications that may cause injuries, deaths or property damages. Since ARTERY products are not intended for the above-mentioned purposes, if purchasers apply ARTERY products to these purposes, purchasers are solely responsible for any consequences or risks caused, even if any written notice is sent to ARTERY by purchasers; in addition, purchasers are solely responsible for the compliance with all statutory and regulatory requirements regarding these uses.

Any inconsistency of the sold ARTERY products with the statement and/or technical features specification described in this document will immediately cause the invalidity of any warranty granted by ARTERY products or services stated in this document by ARTERY, and ARTERY disclaims any responsibility in any form.