FAQ0118

Frequently Asked Questions

USB data cannot display on host

Question:

When using USB device to send data, if the number of bytes to send is always equal to the Bulk IN endpoint's maximum packet size (such as 64 Byte), the host tool will fail to display data.

For example: for V1.3.0 or older BSP version, in the demo of "VirtualComPort_loopback", the host transmits a 64-byte data to USB device, which then receives it and then sends back to the host. In this case, the host is likely to be unable to receive such data in a real time way.

Answer:

This problem is a result of USB2.0 protocol bulk transfer definition. See below about Bulk transfers defined in Section 5.8 of USB2.0 specification.

> An endpoint must always transmit data payloads with a data field less than or equal to the endpoint's reported *wMaxPacketSize* value. When a bulk IRP involves more data than can fit in one maximum-sized data payload, all data payloads are required to be maximum size except for the last data payload, which will contain the remaining data. A bulk transfer is complete when the endpoint does one of the following:
>
> - Has transferred exactly the amount of data expected
> - Transfers a packet with a payload size less than *wMaxPacketSize* or transfers a zero-length packet

Simply put it, a bulk transfer is complete when an endpoint does one of the following:

1. Has transferred exactly the amount of data expected
2. Transfer a packet with a payload size less than a maximum packet size
3. Transfer a zero-length packet

Back to the above-mentioned problem, if USB always transmit data payloads with a data field equal to the maximum packet size, the current bulk transfer is regarded as incomplete by host, so that the data cannot be displayed on host.

Solution:

1. Use the latest version of BSP
2. For V1.3.0 or lower BSP version, this problem can be fixed by transferring a zero-length packet. The following example is based on AT32F403A's BSP DEMO VirtualComPort_loopback.

Add red font in the code (see red font below, which is used to judge whether the current data transfer is complete or not). If there is no data pending for transmit, USB device then sends a zero-length packet to tell the host that the current transfer has been complete.

```
void EP1_IN_Callback(void)
{
  uint16_t dwSendLen = 0;
  uint32_t SendPtr = 0;
  static uint8_t send0packet = 0;
  /*no data need send*/
  if (usb_txfifo.wrpointer == usb_txfifo.curpointer )
```

```
  {
    if ( send0packet == 1 )
    {
      SetEPTxCount(ENDP1, 0);
      SetEPTxValid(ENDP1);
      send0packet = 0;
    }
    return;
  }
  if ( usb_txfifo.wrpointer > usb_txfifo.curpointer )
  {
    dwSendLen = usb_txfifo.wrpointer - usb_txfifo.curpointer;
  }
  else
  {
    dwSendLen = USB_FIFO_MAX - usb_txfifo.curpointer;
  }
  SendPtr = usb_txfifo.curpointer;
  if ( dwSendLen > VIRTUAL_COM_PORT_DATA_SIZE )
  {
    dwSendLen = VIRTUAL_COM_PORT_DATA_SIZE;
  }
  usb_txfifo.curpointer += dwSendLen;
  if ( usb_txfifo.curpointer >= USB_FIFO_MAX )
  usb_txfifo.curpointer = 0;
  send0packet = 1;
  /* send   packet to PMA*/
  UserToPMABufferCopy(&usb_txfifo.fifo[SendPtr], ENDP1_TXADDR, dwSendLen);
  SetEPTxCount(ENDP1, dwSendLen);
  SetEPTxValid(ENDP1);
}
```

The above-mentioned modifications (red font descriptions) have already been added to V1.3.0 or newer BSP versions.

For 2.x BSP version, proceed as lows (note: these contents have already been added to BSP. Here we just use them for further explanation. User do not need to do settings any more)

```
while(1)
{
  /* get usb vcp receive data */
  data_len = usb_vcp_get_rxdata(&usb_core_dev, usb_buffer);

  if(data_len > 0 || send_zero_packet == 1)
  {
```

```
    /* bulk transfer is complete when the endpoint does one of the following
    1 has transferred exactly the amount of data expected
    2 transfers a packet with a payload size less than wMaxPacketSize or transfers a zero-length packet
        */
  if(data_len > 0)
  send_zero_packet = 1;

  if(data_len == 0)
  send_zero_packet = 0;

  timeout = 50000;
  do
  {
    /* send data to host */
    if(usb_vcp_send_data(&usb_core_dev, usb_buffer, data_len) == SUCCESS)
    {
      break;
    }
  }while(timeout --);
}
```

According to code logic, a zero-length packet will be sent in the end every time the USB device receives data payload and then sends back to host.

Type: MCU application

Applicable products: AT32F403，AT32F413，AT32F403A，AT32F407

Main function: USB

Other function: None

Document revision history

Date	Revision	Changes
2022.3.3	2.0.0	Initial release
2022.4.28	2.0.1	Updated settings of V2.x BSP

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Purchasers are solely responsible for the selection and use of ARTERY's products and services, and ARTERY assumes no liability whatsoever relating to the choice, selection or use of the ARTERY products and services described herein

No license, express or implied, to any intellectual property rights is granted under this document. If any part of this document deals with any third party products or services, it shall not be deemed a license granted by ARTERY for the use of such third party products or services, or any intellectual property contained therein, or considered as a warranty regarding the use in any manner of such third party products or services or any intellectual property contained therein.

Unless otherwise specified in ARTERY's terms and conditions of sale, ARTERY provides no warranties, express or implied, regarding the use and/or sale of ARTERY products, including but not limited to any implied warranties of merchantability, fitness for a particular purpose (and their equivalents under the laws of any jurisdiction), or infringement on any patent, copyright or other intellectual property right.

Purchasers hereby agree that ARTERY's products are not designed or authorized for use in: (A) any application with special requirements of safety such as life support and active implantable device, or system with functional safety requirements; (B) any aircraft application; (C) any aerospace application or environment; (D) any weapon application, and/or (E) or other uses where the failure of the device or product could result in personal injury, death, property damage. Purchasers' unauthorized use of them in the aforementioned applications, even if with a written notice, is solely at purchasers' risk, and Purchasers are solely responsible for meeting all legal and regulatory requirements in such use.

Resale of ARTERY products with provisions different from the statements and/or technical characteristics stated in this document shall immediately void any warranty grant by ARTERY for ARTERY's products or services described herein and shall not create or expand any liability of ARTERY in any manner whatsoever.