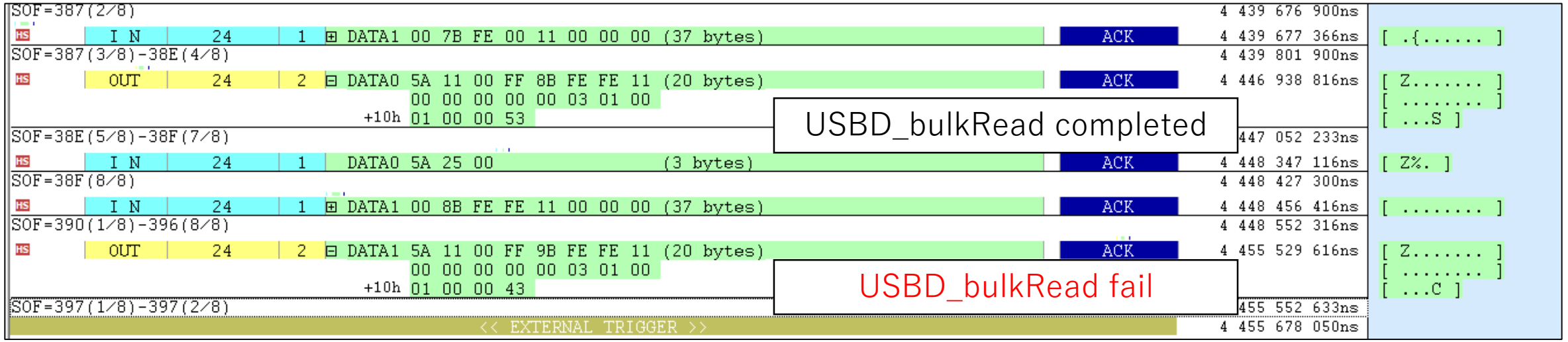
Issue: USB Read/Write sometimes fail



Information of protocol analyzer

Cause of failure

C:¥ti¥pdk_am57xx_1_0_11¥packages¥ti¥drv¥usb¥src¥dwc¥ usb_dwc_dcd.c

B::List `usbDwcDcdEpEvntHandler`

オーバー | Diverge | 出口へ | 呼出元 | 実行 | 停止 | 表示切替 | 検索: | usb_dwc_dcd.c

```
ine source

void usbDwcDcdEpEvntHandler(usbDwcDcdDevice_t *dwc3, usbDwcDcdEvnt_t *usbDwcDcdEvnt)
996 {
    debug_printf("%s:%d. evntType=%d. ¥n",
                 __FUNCTION__, __LINE__,
                 usbDwcDcdEvnt->dEpEvnt.evntType
                 );

    /* Logical endpoints 0 and 1 are handled seperately */
003 if((1U == usbDwcDcdEvnt->dEpEvnt.phEpNum) ||
       (0U == usbDwcDcdEvnt->dEpEvnt.phEpNum))
    {
006     usbDwcDcdEp0EvntHandler(dwc3, usbDwcDcdEvnt);
    }
    else
    {
        /* all other endpoint events are handled here */
011     switch(usbDwcDcdEvnt->dEpEvnt.evntType)
        {
            case USB_DWC_DCD_EVNT_TYPE_XFERCOMPLETE:
014         if(3U == usbDwcDcdEvnt->dEpEvnt.phEpNum)
                {
                    /* This is a request for in transfer complete */
                    /* Update length to be equal to length transferred */
018                 dwc3->inEpReq.length = dwc3->inEpReq.length - bulkInTrb.bufSize;
019                 dwc3->inEpReq.reqComplete(dwc3->pDcdCore->pGadgetObject, &dwc3->inEpReq);
                }
021             else if(2U == usbDwcDcdEvnt->dEpEvnt.phEpNum)
                {
023                 dwc3->outEpReq.length = dwc3->outEpReq.length - bulkOutTrb.bufSize;
024                 dwc3->outEpReq.reqComplete(dwc3->pDcdCore->pGadgetObject, &dwc3->outEpReq);
                }
```

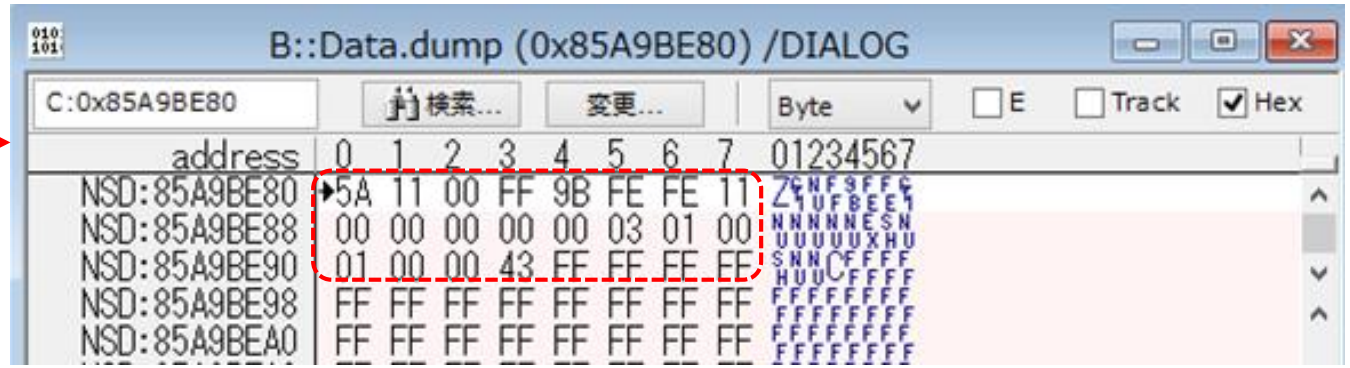
C:¥ti¥pdk_am57xx_1_0_11¥packages¥ti¥drv¥usb¥s

dwc3-> outEpReq.length “0”
dwc3->outEpReq.length = 0x200
bulkOutTrb.bufSize = 0x200 (Not updated)
-> If success “read”, bufSize is updated

“bulkOutTrb” and “Receive buffer”

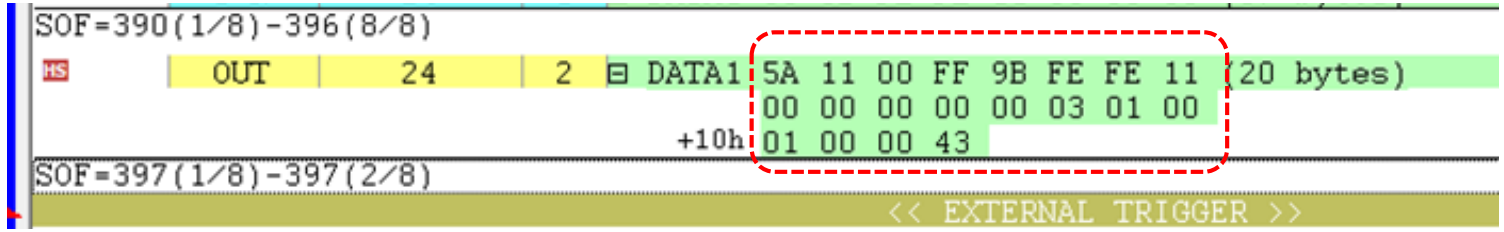
```
bulkOutTrb = (
    bufPtrLow = 0x85A9BE80,
    bufPtrHigh = 0x0,
    bufSize = 0x0200,
    packetCntM1 = 0x0,
    rsvd1 = 0x0,
    trbSts = 0x0,
    hwo = 0x1,
    lst = 0x1,
    chn = 0x0,
    csp = 0x0,
    trbCtrl = 0x1,
    ispImi = 0x0,
    ioc = 0x0,
    rsvd2 = 0x0,
    streamId = 0x0,
    rsvd3 = 0x0)
```

Memory dump(Receive buffer)



*FF: Buffer filled before receiving

USB protocol analyzer



Data that could not be received was in the buffer.

TRB area is overwritten (Write back) with cache data. (Probably)

Therefore, the TRB area was set outside the cache area.

usb_dwc_dcd.c

```
/** ¥brief usb endpoint command parameters */
static usbDwcDcdDEpCmdParms_t dEpCmdParm __attribute__ ((aligned (ALIGNED_SIZE), section
(".bss:extMemNonCache:usbXhci")));

/** ¥brief TRB for transfer of control data */
static usbDEpTrb_t ctrlDataTrb __attribute__ ((aligned (ALIGNED_SIZE), section (".bss:extMemNonCache:usbXhci")));

/** ¥brief TRB for control status commands */
static usbDEpTrb_t ctrlStatusTrb __attribute__ ((aligned (ALIGNED_SIZE), section (".bss:extMemNonCache:usbXhci")));

/** ¥brief TRB for bulk in transfers */
static usbDEpTrb_t bulkInTrb __attribute__ ((aligned (ALIGNED_SIZE), section (".bss:extMemNonCache:usbXhci")));

/** ¥brief TRB for bulk out transfers */
static usbDEpTrb_t bulkOutTrb __attribute__ ((aligned (ALIGNED_SIZE), section (".bss:extMemNonCache:usbXhci")));

/** ¥brief USB event buffer */
static usbDwcDcdEvnt_t usbEvnt[USB_DCD_DWC_EVNT_BUF_MAX] __attribute__ ((aligned (ALIGNED_SIZE), section
(".bss:extMemNonCache:usbXhci")));

/** ¥brief Control data buffer */
static uint32_t usbCtrlDataBuf[32U] __attribute__ ((aligned (ALIGNED_SIZE), section (".bss:extMemNonCache:usbXhci")));
```

Added section (“.bss: extMemNonCache: usbXhci”)) to __attribute__ of each memory definition.

* SUCCESS ALL
(READ 1000000 times, WRITE 2000000 times)

From this hypothesis and experiment,
we thought the TRB area was wrong.