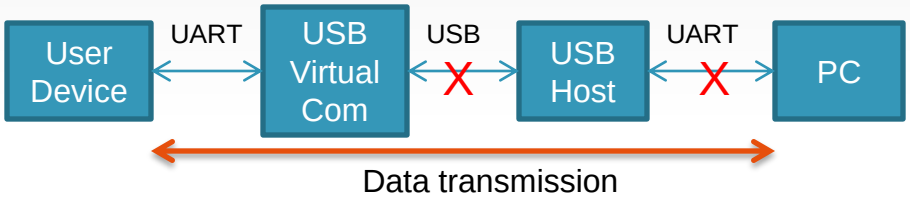
USB CDC Host application

Issue : USB Host not working

MCU FAE David Chen

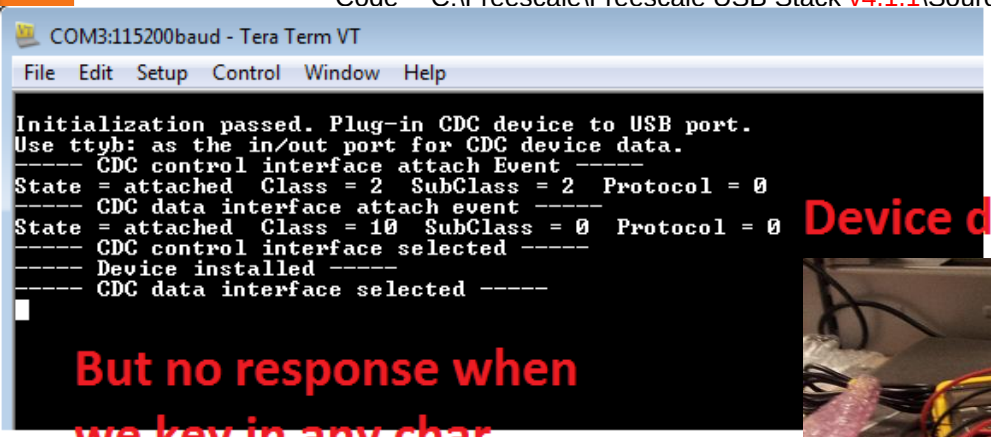
Issue



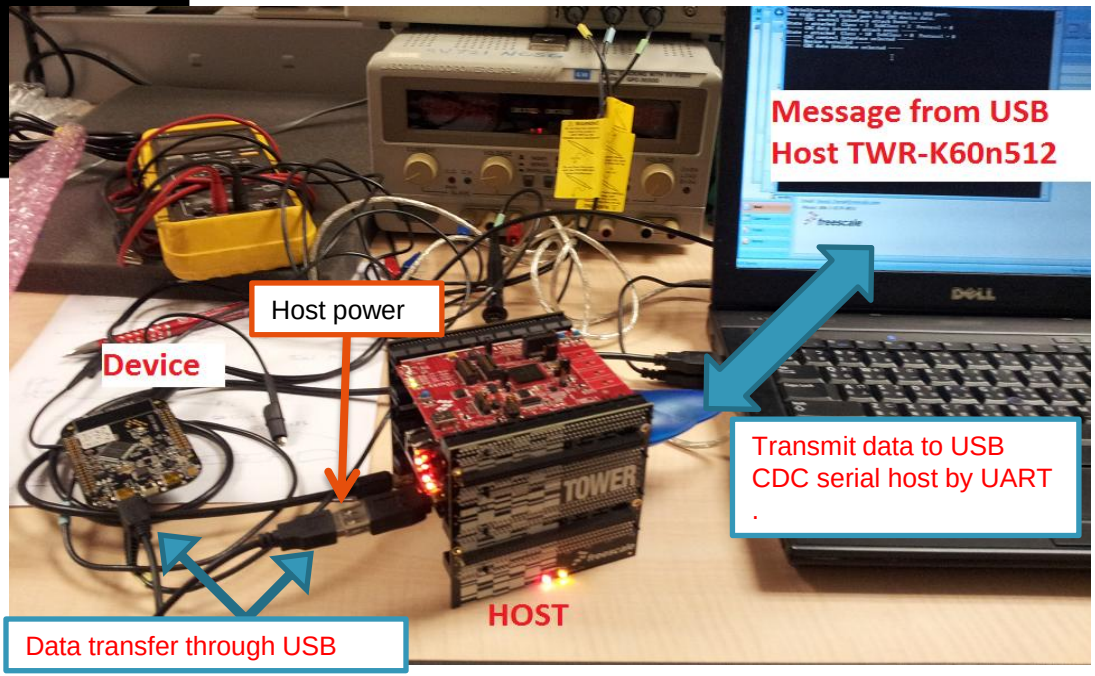
- Stack revision – 4.0.3
- State
  - Virtual Com USB Device detectable .
  - **Data transfer not working** .
- Expect
  - PC transfer data to Embedded USB Host Device by UART then the data passed to USB Virtual Com Device by embedded USB Host .
  - USB Virtual Com Device transfer data to embedded USB Host then the data passed to PC by UART of embedded USB Host Device .
- Issue
  - USB Host can't transfer data to Virtual Com device from UART
  - There is no response when Virtual Com transfer data to USB Host

Experiment -- Environment

- USB CDC Host :
  - EVB – TWR-K60n512
  - Code -- C:\Freescale\Freescale USB Stack v4.0.3\Source\Host\examples\cdc_serial\cw10\kinetis_k60
- USB virtual Com Device :
  - EVB – FRDM-KL25Z
  - Code -- C:\Freescale\Freescale USB Stack v4.1.1\Source\Device\app\cdc\cw10\kinetis_l2k



Device detected



USB Host information declaration

cdc_serial.c | sci.c | poll.c | khci_kinetis.c | usb_host_cdc.c | usbms

```
81 void CDC_Task();
82 extern void _usb_khci_task(void);
83
84 static const USB_HOST_DRIVER_INFO DriverInfoTable[] = {
85    {
86       {0x00,0x00},                /* Vendor ID per USB-IF             */
87       {0x00,0x00},                /* Product ID per manufacturer      */
88       USB_CLASS_COMMUNICATION,    /* Class code                       */
89       USB_SUBCLASS_COM_ABSTRACT,  /* Sub-Class code                   */
90       0xFF,                       /* Protocol                         */
91       0,                          /* Reserved                         */
92       usb_host_cdc_acm_event      /* Application call back function   */
93    },
94    {
95       {0x00,0x00},                /* Vendor ID per USB-IF             */
96       {0x00,0x00},                /* Product ID per manufacturer      */
97       USB_CLASS_DATA,             /* Class code                       */
98       0xFF,                       /* Sub-Class code                   */
99       0xFF,                       /* Protocol                         */
100      0,                          /* Reserved                         */
101      usb_host_cdc_data_event         /* Application call back function   */
102   },
103   /* USB 1.1 hub */
104   {
105
106      {0x00,0x00},                /* Vendor ID per USB-IF             */
107      {0x00,0x00},                /* Product ID per manufacturer      */
108      USB_CLASS_HUB,              /* Class code                       */
109      USB_SUBCLASS_HUB_NONE,      /* Sub-Class code                   */
110      USB_PROTOCOL_HUB_LS,        /* Protocol                         */
111      0,                          /* Reserved                         */
112      usb_host_hub_device_event   /* Application call back function   */
113   },
114   {
115      {0x00,0x00},                /* All-zero entry terminates        */
116      {0x00,0x00},                /* driver info list.                */
117      0,
118      0,
119      0,
120      0,
121      NULL
122   }
123 };
```

- Detail please refer to USBHOSTUG.pdf Chapter 4.2.2 [**Define a driver info table**].

/* Information for one class or device driver */

typedef struct driver_info

{

uint_8 idVendor[2]; /* Vendor ID per USB-IF */

uint_8 idProduct[2]; /* Product ID per manufacturer */

uint_8 bDeviceClass; /* Class code, 0xFF if any */

uint_8 bDeviceSubClass; /* Sub-Class code, 0xFF if any */

uint_8 bDeviceProtocol; /* Protocol, 0xFF if any */

uint_8 reserved; /* Alignm

event_callback attach_call; /* event callback function*/

} USB_HOST_DRIVER_INFO, _PTR_ USB_HOST_DRIVER_INFO_PTR; ent padding */

Initial USB Host

cdc_serial.c | sci.c | poll.c | khci_kinetis.c

```
/*FUNCTION*-----------------------------------------
*
* Function Name  : Main
* Returned Value : none
* Comments       :
*
*
*END*-----------------------------------------------
#ifdef __GNUC__
int main(void)
#else
void main(void)
#endif
{
   USB_STATUS           status = USB_OK;
   _usb_host_handle     host_handle;

   /* Initialize the current platform. Call for the
   _bsp_platform_init();
#ifdef MCU_MK70F12
  sci2_init();
#else
  sci1_init();
#endif
   TimerInit();

   /* Init polling global variable */
   POLL_init();

   DisableInterrupts;
   #if (defined _MCF51MM256_H) || (defined _MCF51JE
   usb_int_dis();
   #endif
```

Peripheral initial

```
/*
** It means that we are going to act like host, so we initialize the
** host stack. This call will allow USB system to allocate memory for
** data structures, it uses later (e.g pipes etc.).
*/
status = _usb_host_init (
      HOST_CONTROLLER_NUMBER,   /* Use value in header file */
      MAX_FRAME_SIZE,           /* Frame size per USB spec  */
      &host_handle);            /* Returned pointer */
if (status != USB_OK)
{
   printf("\nUSB Host Initialization failed. STATUS: %x",(unsigned int) s
   fflush(stdout);
   exit(3);
}
status = _usb_host_driver_info_register (
                                  host_handle,
                                  (void *)DriverInfoTable
                                  );
if (status != USB_OK)
{
   printf("\nDriver Registration failed. STATUS: %x", (unsigned int)statu
   fflush(stdout);
   exit(4);
}

EnableInterrupts;
#if (defined _MCF51MM256_H) || (defined _MCF51JE256_H)
usb_int_en();
#endif

printf("\fInitialization passed. Plug-in CDC device to USB port.\nUse ttyb

usb_event_init(&device_registered);
uart2usb_num = usb2uart_num = 0; /* reset number of bytes in buffers */

f_usb = (FILE_CDC_PTR)malloc(sizeof(FILE_CDC));
memset(f_usb, 0, sizeof(FILE_CDC));

f_usb->DEV_PTR = (IO_DEVICE_STRUCT_PTR)malloc(sizeof(IO_DEVICE_STRUCT));
```

USB Host initial

Register USB driver

Initial event

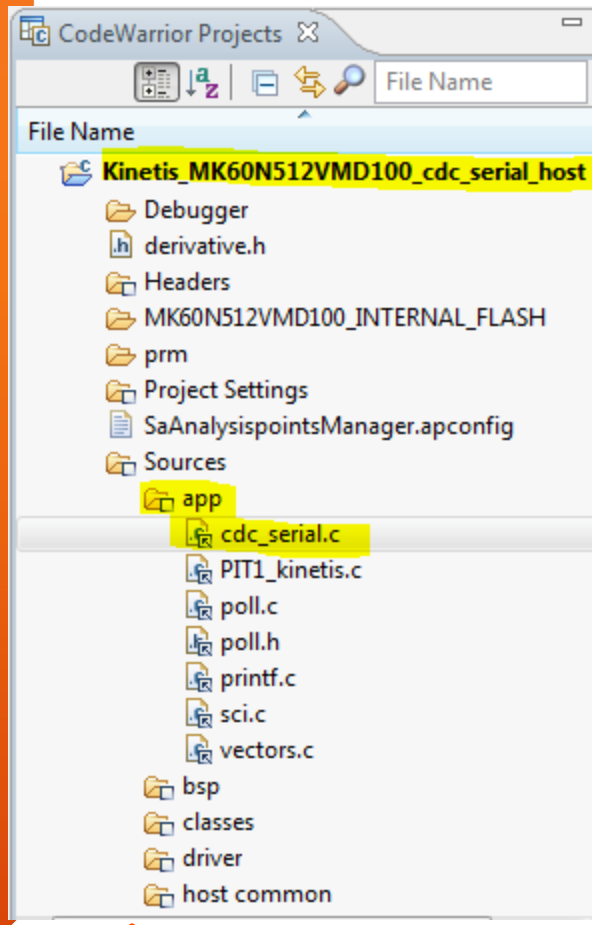
freescale™

USB Host application function flow

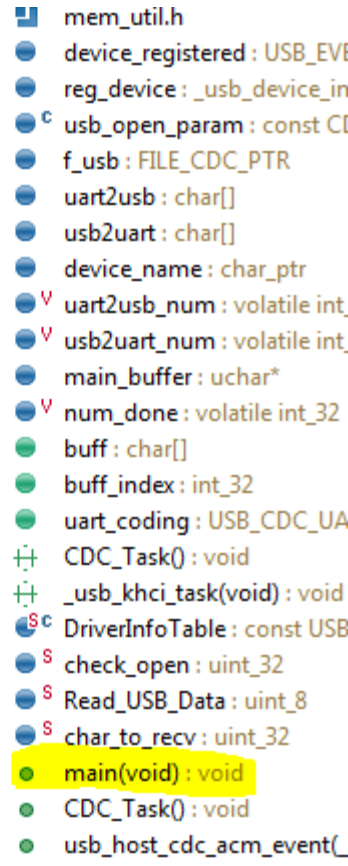
- Please refer to USBHOSTUG.pdf
  - Chapter 4.2.3 [Main application function flow]
    - Initializing hardware
    - Initializing the host controller
    - Registering service
    - Calling tasks in a forever loop

USB CDC Serial Host – main task

- Major task of USB CDC serial Host application




```
195   _usb_event_init(&device_registered);
196   uart2usb_num = usb2uart_num = 0; /* reset number of
197
198   f_usb = (FILE_CDC_PTR)malloc(sizeof(FILE_CDC));
199   memset(f_usb, 0, sizeof(FILE_CDC));
200
201   f_usb->DEV_PTR = (IO_DEVICE_STRUCT_PTR)malloc(sizeo
202
203
204   for(;;)
205   {
206     Poll();
207     CDC_Task();
208     __RESET_WATCHDOG(); /* feeds the dog */
209   } /* loop forever */
210   /* please make sure that you never leave main */
211 #ifdef __GNUC__
212   return 0;
213 #endif
214 }
215
216 /*FUNCTION*-----------------------------------------
217 *
218 * Function Name  : CDC_Task
219 * Returned Value : none
220 * Comments       :
221 *     Execution starts here
222 *
```



Major process of main task – Enumerate Device

cdc_serial.c | sci.c | poll.c | khci_kinetis.c | usb_host_cdc.c | usbmsgq.c | virtual_com.c | PIT1_kine

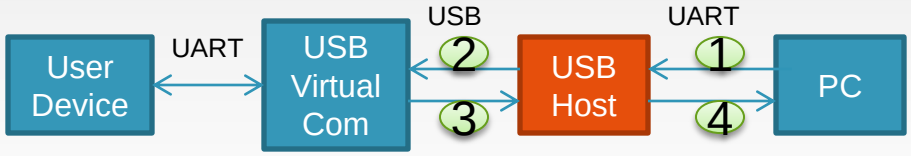
```
222 /*FUNCTION*----------------------------------------------------------------
223 *
224 * Function Name  : CDC_Task
225 * Returned Value : none
226 * Comments       :
227 *     Execution starts here
228 *
229 *END*----------------------------------------------------------------------*/
230 void CDC_Task ()
231 { /* Body */
232     USB_STATUS          status = USB_OK;
233     uint_32             i = 0;
234     //static uint_32       char_to_recv = 0;
235
236     /* due to the fact that uart driver blocks task, we will check if char is available and then we read it */
237     /* write them to USB */
238     if(USB_EVENT_NOT_SET == _usb_event_wait_ticks(&device_registered, 0x01, TRUE, 0))
239       return;
240
241     if (0 == check_open)
242     {
243         _io_cdc_serial_open(f_usb, device_name, (char *)&usb_open_param);
244         check_open = 1;
245     }
246     else
247     {
248 #ifdef MCU_mcf51jf128
249         /* todo AI: change this for MCU_mcf51jf128 */
250         char temp;
251
252         temp = TERMIO_GetChar();
253         if(temp)
254         {
255             buff_index = 1;
256             buff[0] = temp;
257         }
258 #endif /* MCU_mcf51jf128 */
259
260         /* Read data from UART */
```

Waiting for USB Device attached

Open USB Device if not yet

Read / Write Data between USB and UART, after USB device attached

USB Data handler



*cdc_serial.c | sci.c | poll.c | khci_kinetis.c | usb_host_cdc.c | usbmsgq.c | virtual_com.c | PIT1_kinetis.c | host_ch9.c | usbev

```
240
241     if (0 == check_open)
242     {
243         _io_cdc_serial_open(f_usb, device_name, (char *)&usb_open_param);
244         check_open = 1;
245     }
246     else
247     {
248
249         DisableInterrupts;
250         uart2usb_num = buff_index;
251         for (i = 0; i < buff_index; i++)
252             uart2usb[i] = buff[i];
253         buff_index = 0;
254         EnableInterrupts;
255
256         /* Write data to USB */
257         if (uart2usb_num)
258         {
259            num_done = _io_cdc_serial_write(f_usb, uart2usb, (int_32)(sizeof(uart2usb[0]) * uart2usb_num));
260            if(num_done > 0)
261            {
262                for (i = (uint_32)num_done; i < uart2usb_num; i++) /* move buffer data, remove the written ones */
263                    uart2usb[i - num_done] = uart2usb[i];
264                uart2usb_num -= num_done;
265                char_to_recv += num_done;
266            }
267         }
268
269
270         /* Read data from USB */
271         if(char_to_recv > 0 || Read_USB_Data)
272         {
273             Read_USB_Data = 0 ;
274             num_done = _io_cdc_serial_read(f_usb, usb2uart + usb2uart_num, (int_32)(sizeof(uart2usb) / sizeof(uart2usb[0]) - usb2uart_num));
275             if(num_done > 0 && usb2uart[usb2uart_num] != 0 )
276             {
277                 usb2uart_num += num_done;
278                 if(char_to_recv >= num_done)
279                 {
280                     char to recv -= num done;
306                 }
307                 else
308                 {
309                     char_to_recv  = 0;
310                 }
311             }
312             else
313             {
314                 char_to_recv  = 0;
315             }
316         }
317
318         /* write them to UART */
319         if (usb2uart_num > 0)
320         {
321             for (i = 0; i < usb2uart_num; i++)
322             {
323 #ifde MCU_mcf51jf128
324                 printf(" --received--> %c\n\r",usb2uart[i]);
325 #else
326                 sci2_PutChar(usb2uart[i]);
327
328 #endi
329             }
330             usb2uart_num = 0;
331
332         }
333     }
334 } /* Endbody */
335
```

1 UART to USB buffer index handler

2 Write data to USB

3 Read data from USB

4 Write data to UART

- UART to USB buffer
  - PC terminal data to USB CDC Host
- Write Data to USB
  - USB CDC Host transfer data to CDC Device
- Read data from USB
  - USB CDC Host receive data from CDC device
- Write data to UART
  - USB CDC Host transfer data to PC terminal

Debug 1– Can't receive data from PC terminal

Only handle buffer index but no one to receive data

UART to USB buffer index handler

Write data to USB

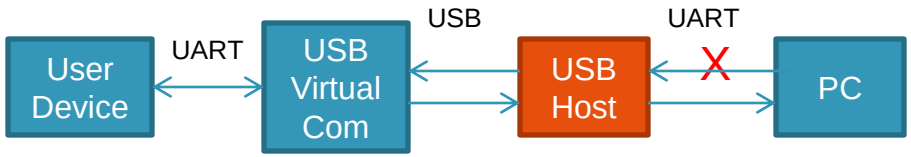
Read data from USB

Write data to UART

```
*cdc_serial.c | sci.c | poll.c | khci_kinetis.c | usb_host_

240
241     if (0 == check_open)
242     {
243         _io_cdc_serial_open(f_usb, device_name, (char *)&usb_op
244         check_open = 1;
245     }
246     else
247     {
248
249         DisableInterrupts;
250         uart2usb_num = buff_index;
251         for (i = 0; i < buff_index; i++)
252             uart2usb[i] = buff[i];
253         buff_index = 0;
254         EnableInterrupts;
255
256         /* Write data to USB */
257         if (uart2usb_num)
258         {
259            num_done = _io_cdc_serial_write(f_usb, uart2usb, (int_32)(sizeof(uart2usb[0]) * uart2usb_num));
260            if(num_done > 0)
261            {
262                for (i = (uint_32)num_done; i < uart2usb_num; i++) /* move buffer data, remove the written ones */
263                    uart2usb[i - num_done] = uart2usb[i];
264                uart2usb_num -= num_done;
265                char_to_recv += num_done;
266            }
267         }
268
269
270         /* Read data from USB */
271         if(char_to_recv > 0 || Read_USB_Data)
272         {
273             Read_USB_Data = 0 ;
274             num_done = _io_cdc_serial_read(f_usb, usb2uart + usb2uart_num, (int_32)(sizeof(uart2usb) / sizeof(uart2usb[0]) - usb2uart_num));
275             if(num_done > 0 && usb2uart[usb2uart_num] != 0 )
276             {
277                 usb2uart_num += num_done;
278                 if(char_to_recv >= num_done)
279                 {
280                     char to recv -= num done;
306                 }
307                 else
308                 {
309                     char_to_recv  = 0;
310                 }
311             }
312             else
313             {
314                 char_to_recv  = 0;
315             }
316         }
317
318         /* write them to UART */
319         if (usb2uart_num > 0)
320         {
321             for (i = 0; i < usb2uart_num; i++)
322             {
323 #ifde MCU_mcf51jf128
324                 printf(" --received--> %c\n\r",usb2uart[i]);
325 #else
326                 sci2_PutChar(usb2uart[i]);
327
328 #endi
329             }
330             usb2uart_num = 0;
331
332         }
333     }
334 } /* Endbody */
335
```

- Issue :
  - Data NOT received from PC terminal
- Root cause :
  - No bubby to receive data from UART
- Workaround :
  - Call uart_getchar() to receive data
  - Detail pls refer to next slide .



Debug 1 -- Workaround – Can't receive data from PC terminal

```
260     /* Read data from UART */
261     char temp;
262
263     temp = uart_getchar();  //for CodeWarrior
264     //temp = TERMIO_GetCharNB();    // for IAR
265     if(temp)
266     {
267         buff[0] = temp;
268         buff_index = 1;
269     }
270
271      DisableInterrupts;
272      uart2usb_num = buff_index;
273      for (i = 0; i < buff_index; i++)
274          uart2usb[i] = buff[i];
275      buff_index = 0;
276      EnableInterrupts;
```

Call uart_getchat() to receive data from PC terminal

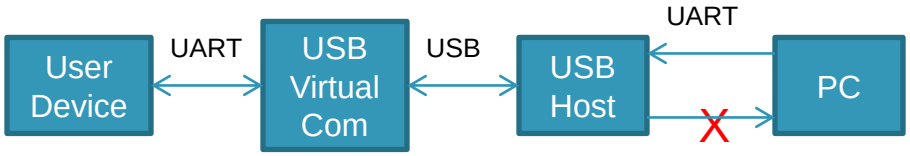
*cdc_serial.c | *sci.c | PIT1_kinetis.c | host_ch9.c

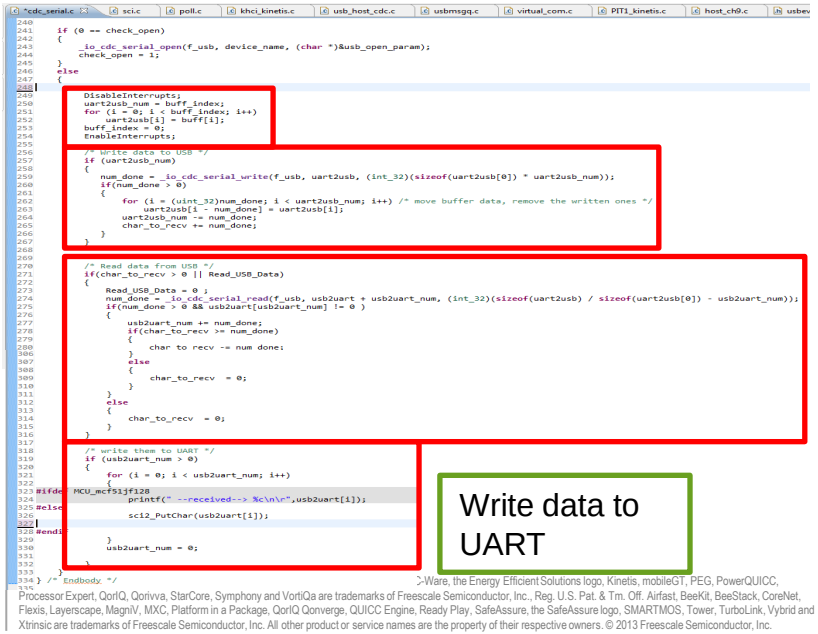
```
200
201 #ifdef __CC_ARM
202 int getkey(void)
203 #else
204 char uart_getchar (void)
205 #endif
206 {
207     /* Wait until character has been received */
208     //while (!(UART3_S1 & UART_S1_RDRF_MASK));
209     if(UART3_S1 & UART_S1_RDRF_MASK)
210     /* Return the 8-bit data from the receiver */
211         return UART3_D;
212     else
213         return 0 ;
214 }
```

A modify required of uart_getchat() in sci.c
Must change
"while(!(UART3_S1&UART_S1_RDRF_MASK))"
To
"if(UART3_S1&UART_S1_RDRF_MASK)"
If NOT
Code will be stop here to wait data from PC
.

Debug 2 – Can’t transfer data to PC terminal

- Issue :
  - After bug 1 fixed
  - Data transfer to USB CDC Host from PC terminal is working
  - When USB CDC Device receive data from host that will return the data to Host , but there is nothing display on PC terminal .
- Root cause:
  - Something wrong at “write data to UART” block .
  - Write data to UART by “sci2_PutChar(usb2uart[i])” when usb2uart_num not zero .
  - **But “sci2_PutChar() “ is empty that decelerated in sci.c .**
- Workaround :
  - To call “uart_putchar()“ instead .
  - “uart_putchar(usb2uart[i]);” decelerated in sci.c too .
  - Detail please refer to next page .



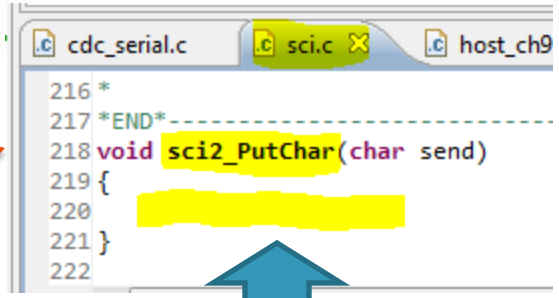


Debug 2 – Workaround –Can't transfer data to PC terminal

CDC_Task()

```
/* write them to UART */
if (usb2uart_num > 0)
{
    for (i = 0; i < usb2uart_num; i++)
    {
        //sci2_PutChar(usb2uart[i]);
        uart_putchar(usb2uart[i]);
        usb2uart[i] = 0 ; //clear buffer
    }
    usb2uart_num = 0;

}
```



Not thing done here

To call "uart_putchar()" instead

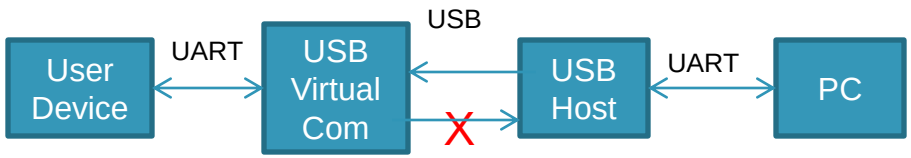
Sci.c

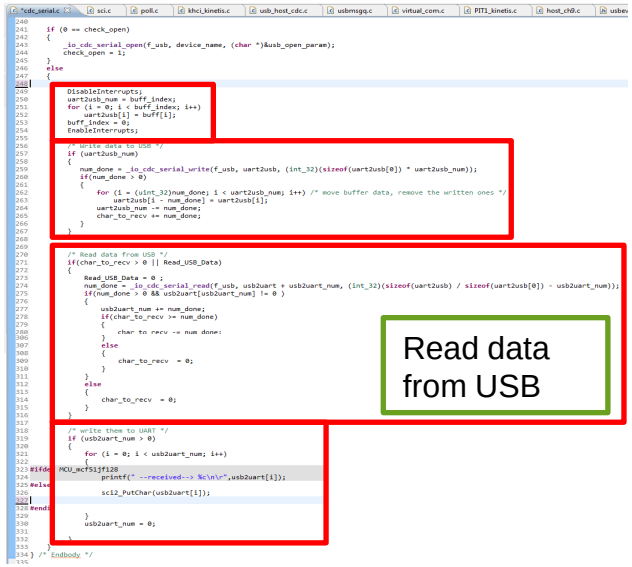
```
* Function Name    : TERMIO_PutChar
* Returned Value   :
* Comments         :
*                      This function sends a char via SCI.
*
*END*----------------------------------------------------------
void uart_putchar (char ch)
{
    /* Wait until space is available in the FIFO */
    while(!(UART3_S1 & UART_S1_TDRE_MASK)){};

    /* Send the character */
    UART3_D = (uint_8)ch;
}
```

Debug 3– Can’t receive data from CDC Device

- Issue :
  - After bug 1 fixed ,
  - When USB CDC Host transfer a data to CDC Device , we can receive one response from CDC Device
  - If CDC device transfer data to USB CDC Host **voluntarily** , there hasn’t any response at Host side .
- Root cause :
  - Something wrong at “Read data from USB”
  - Read data from USB by “**_io_cdc_serial_read()**” , but it only called when **“char_to_recv”** asserted not periodicity .
  - And “char_to_recv” only asserted after USB CDC Host send data to USB CDC device . It will be cleared after two byte received from USB CDC device when one byte sent by host .
- Workaround :
  - To execute “_io_cdc_serial_read()” periodicity .
  - One more condition added to inter “read data from USB” .
  - Detail please refer to next page .





Debug 3 – Workaround – Can't receive data from CDC Device

One more condition "Read_USB_Data" added to inter "read data from USB "

CDC_Task()

cdc_serial.c | sci.c | usb_host_cdc.c | usbmsgq.c | virtual_com.c

```
291
292         /* Read data from USB */
293         if(char_to_recv > 0 || Read_USB_Data)
294         {
295             Read_USB_Data = 0 ;
296             num_done = _io_cdc_serial_read(f_usb, usb2uart + usb2uart_num,
297             if(num_done > 0 && usb2uart[usb2uart_num] != 0 )
298             {
299                 usb2uart_num += num_done;
300                 if(char_to_recv >= num_done)
301                 {
302                     char_to_recv -= num_done;
303
304
305
306                     char_to_recv  = 0;
307                 }
308             }
309             else
310             {
311                 char_to_recv  = 0;
312             }
313         }
314
```

To execute "_io_cdc_serial_read()" periodicity

Main()

cdc_serial.c | usb_host_cdc.c

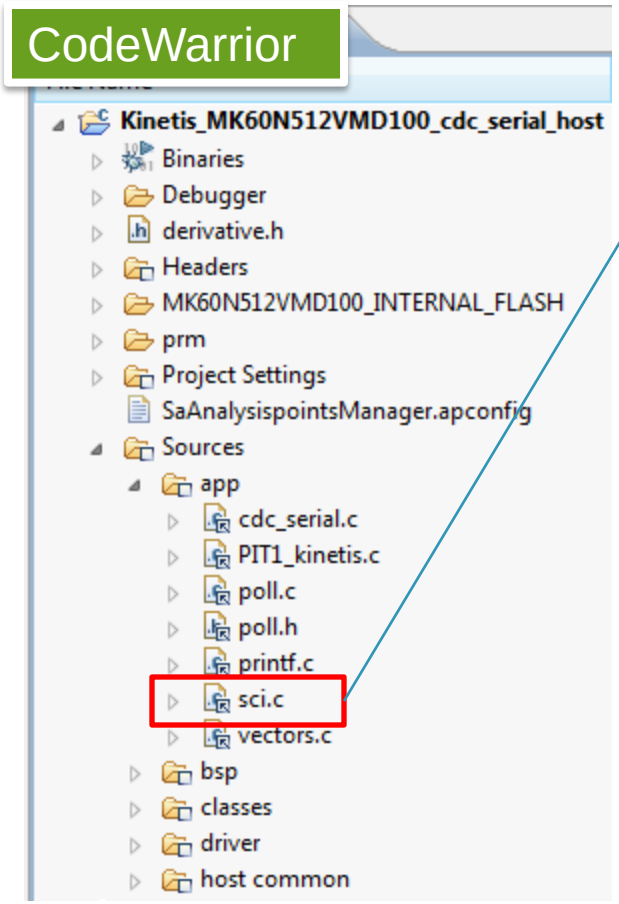
```
204     for(;;)
205     {
206       Poll();
207       CDC_Task();
208       if(char_to_recv == 0)
209       {
210          Read_USB_Data = 1 ;
211          time_delay(10);
212       }
213
214       __RESET_WATCHDOG(); /* feeds the dog */
215     } /* loop forever */
```

To set flag "Read_USB_Data" periodicity

- For this issue , please configure CDC device to send the data to USB Host voluntarily
  - May be press switch to send data to USB CDC host .

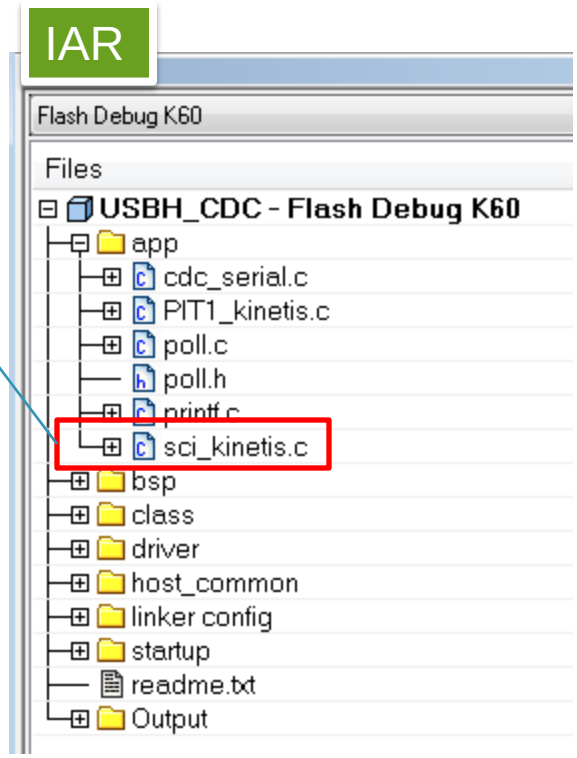
Backup

- There are a little different between IAR and CodeWarrior environment



SCI driver source code is different

Something we modified in “sci.c” please do it on “sci_kinetis.c”




freescale™