```c
#include "common.h"

#include <stdbool.h>
#include <stdint.h>

#include "usblib.h"
#include "usblib/usblib.h"
#include "usb_descriptors.h"
#include "inc/hw_types.h"
#include "driverlib/usb.h"
#include "usblib/usblib.h"
#include "usblib/usbcdc.h"
#include "usblib/usb-ids.h"
#include "usblib/device/usbdevice.h"
#include "usblib/device/usbdcdc.h"




//-----------------------
//  LANGUAGE SUPPORTED  |
//-------------------------------------------------------------------------------------------------------------------
const u8 UsbLanguageDescriptor[] = {

    4,
    USB_DTYPE_STRING,
    USBShort(USB_LANG_EN_US)
};
//-------------------------------------------------------------------------------------------------------------------




//-----------------
//  MANUFACTURER  |
//-------------------------------------------------------------------------------------------------------------------
//  "Multiverse Amps"

const u8 UsbMfgString[] = {

    2 + (15 * 2),
    USB_DTYPE_STRING,

    'M', 0, 'u', 0, 'l', 0, 't', 0, 'i', 0, 'v', 0, 'e', 0, 'r', 0, 's', 0, ' ', 0,
    'A', 0, 'm', 0, 'p', 0, 's', 0,
};
//-------------------------------------------------------------------------------------------------------------------




//------------
//  PRODUCT  |
//-------------------------------------------------------------------------------------------------------------------
//  "Multiverse Proto 2"

const u8 UsbProductString[] = {

    2 + (18 * 2),
    USB_DTYPE_STRING,

    'M', 0, 'u', 0, 'l', 0, 't', 0, 'i', 0, 'v', 0, 'e', 0, 'r', 0, 's', 0, 'e', 0, ' ', 0,
    'P', 0, 'r', 0, 'o', 0, 't', 0, 'o', 0, ' ', 0, '2', 0,
};
//-------------------------------------------------------------------------------------------------------------------




//------------------
//  SERIAL NUMBER  |
```

```
//-------------------------------------------------------------------------------------------------------------
//  "12345678"

const u8 UsbSerNmbString[] = {

    2 + (8 * 2),
    USB_DTYPE_STRING,

    '1', 0, '2', 0, '3', 0, '4', 0, '5', 0, '6', 0, '7', 0, '8', 0
};
//-------------------------------------------------------------------------------------------------------------




//-----------------------------------
//  CONTROL INTERFACE DESCRIPTION  |
//-------------------------------------------------------------------------------------------------------------
//  "ACM Control Interface"   *FIXME* ???

const u8 UsbCtlInterfaceString[] = {

    2 + (21 * 2),
    USB_DTYPE_STRING,

    'A', 0, 'C', 0, 'M', 0, ' ', 0, 'C', 0, 'o', 0, 'n', 0, 't', 0,
    'r', 0, 'o', 0, 'l', 0, ' ', 0, 'I', 0, 'n', 0, 't', 0, 'e', 0,
    'r', 0, 'f', 0, 'a', 0, 'c', 0, 'e', 0
};
//-------------------------------------------------------------------------------------------------------------




//-------------------------------
//  CONFIGURATION DESCRIPTION  |
//-------------------------------------------------------------------------------------------------------------
//  "Self Powered Configuration"   *FIXME* ???

const u8 UsbConfigString[] = {

    2 + (26 * 2),
    USB_DTYPE_STRING,

    'S', 0, 'e', 0, 'l', 0, 'f', 0, ' ', 0, 'P', 0, 'o', 0, 'w', 0,
    'e', 0, 'r', 0, 'e', 0, 'd', 0, ' ', 0, 'C', 0, 'o', 0, 'n', 0,
    'f', 0, 'i', 0, 'g', 0, 'u', 0, 'r', 0, 'a', 0, 't', 0, 'i', 0,
    'o', 0, 'n', 0
};
//-------------------------------------------------------------------------------------------------------------




//----------------------
//  DESCRIPTOR TABLE  |
//-------------------------------------------------------------------------------------------------------------
const u8 * const UsbStringsStruct[] = {

    UsbLanguageDescriptor,
    UsbMfgString,
    UsbProductString,
    UsbSerNmbString,
    UsbCtlInterfaceString,
    UsbConfigString
};
//-------------------------------------------------------------------------------------------------------------
#define NUM_STRING_DESCRIPTORS (sizeof(UsbStringsStruct) / sizeof(u8 *))
//-------------------------------------------------------------------------------------------------------------
```

```
//---------------------------------
//  CALLBACK FUNCTION PROTOTYPES  |
//-------------------------------------------------------------------------------------------------------------------

uint32_t RxHandler(void *pvCBData, uint32_t ui32Event, uint32_t ui32MsgValue, void *pvMsgData);
uint32_t TxHandler(void *pvCBData, uint32_t ui32Event, uint32_t ui32MsgValue, void *pvMsgData);
uint32_t ControlHandler(void *pvCBData, uint32_t ui32Event, uint32_t ui32MsgValue, void *pvMsgData);

//-------------------------------------------------------------------------------------------------------------------




//-------------------------------------
//  INITIALIZATION AND CUSTOMIZATION  |
//-------------------------------------------------------------------------------------------------------------------
#define USB_VID_MV_FFFF 0xFFFF          // Vendor ID, dummy value since this has not been assigned by USB-IF
#define USB_PID_SERIAL  0x0002          // Product ID
//-------------------------------------------------------------------------------------------------------------------
tCDCSerInstance CdcInstanceStruct;

tUSBDCDCDevice CdcDeviceStruct = {

    USB_VID_MV_FFFF,            // VID
    USB_PID_SERIAL,             // PID
    0,                          // Power used in MA
    USB_CONF_ATTR_SELF_PWR,     // Self Powered, no Remote Wakeup.
    ControlHandler,             // Callback Function for Asynchronous Control Events
    (void *)&CdcDeviceStruct,   // Pointer passed as first parameter to ControlHandler function, this structure.
    USBBufferEventCallback,     // Callback Function for Receive Channel events.
    (void *)&UsbRxBuffStruct,   // Pointer passed as first parameter to USBBufferEventCallback, (USB) Rx Buffer.
    USBBufferEventCallback,     // Callback Function for Transmit Channel events.
    (void *)&UsbTxBuffStruct,   // Pointer passed as first parameter to USBBufferEventCallback, (USB) Tx Buffer.
    UsbStringsStruct,           // Pointer to the Array of string descripter pointers.
    NUM_STRING_DESCRIPTORS,     // Numner of entries in the Array of string descripter pointers.
    CdcInstanceStruct           // Instance of private data structure used in the TivaWare USB code.
};
//-------------------------------------------------------------------------------------------------------------------




//-------------------------------------
//  RECEIVE BUFFER (USB PERSPECTIVE)  |
//-------------------------------------------------------------------------------------------------------------------
u8 UsbRxBuffer[USB_RX_BUFF_SIZE];

tUSBBuffer UsbRxBuffStruct = {

    false,                          // This is a receive buffer.
    RxHandler,                      // pfnCallback
    (void *)&CdcDeviceStruct,          // Callback data is our device pointer.
    USBDCDCPacketRead,              // pfnTransfer
    USBDCDCRxPacketAvailable,       // pfnAvailable
    (void *)&CdcDeviceStruct,          // pvHandle
    UsbRxBuffer,               // pui8Buffer
    USB_RX_BUFF_SIZE,               // ui32BufferSize
};
//-------------------------------------------------------------------------------------------------------------------




//-------------------------------------
//  TRANSMIT BUFFER (USB PERSPECTIVE)  |
//-------------------------------------------------------------------------------------------------------------------
u8 UsbTxBuffer[USB_TX_BUFF_SIZE];

tUSBBuffer UsbTxBuffStruct = {

    true,                           // This is a transmit buffer.
```

```
    TxHandler,                      // pfnCallback
    (void *)&CdcDeviceStruct,          // Callback data is our device pointer.
    USBDCDCPacketWrite,             // pfnTransfer
    USBDCDCTxPacketAvailable,       // pfnAvailable
    (void *)&CdcDeviceStruct,          // pvHandle
    UsbTxBuffer,                    // pcBuffer
    USB_TX_BUFF_SIZE,               // ulBufferSize
};
//--------------------------------------------------------------------------------------------------------------
```