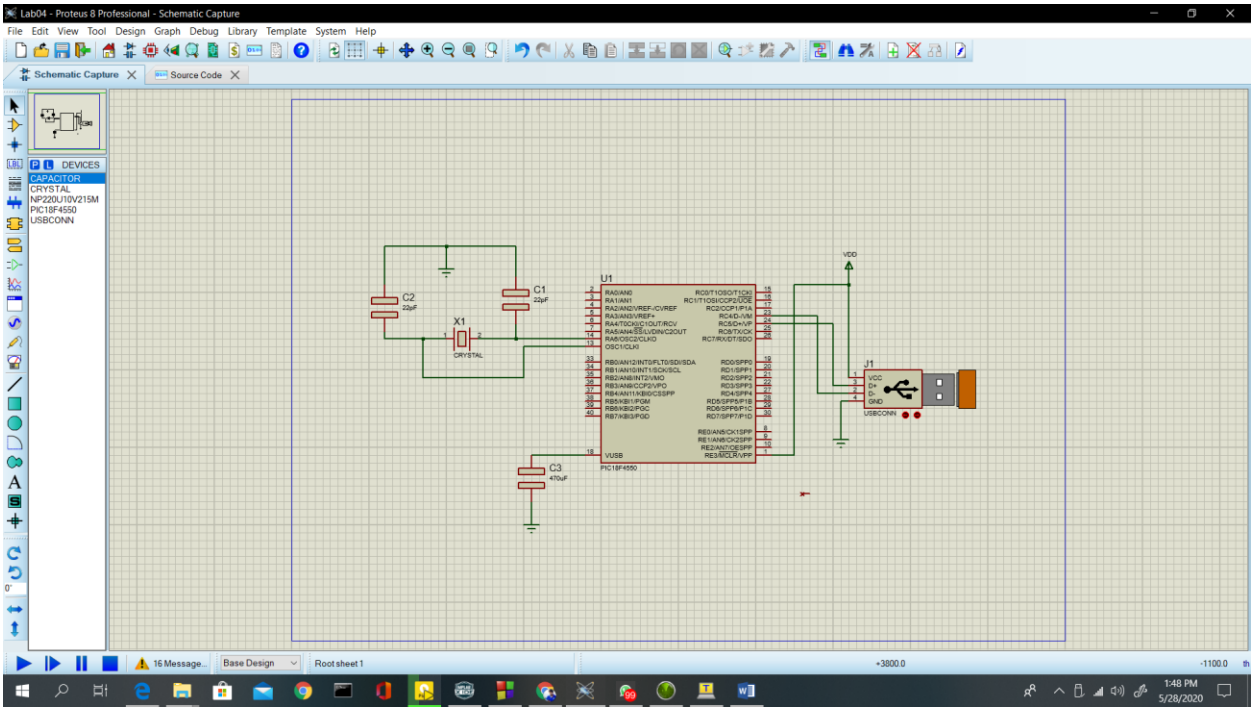
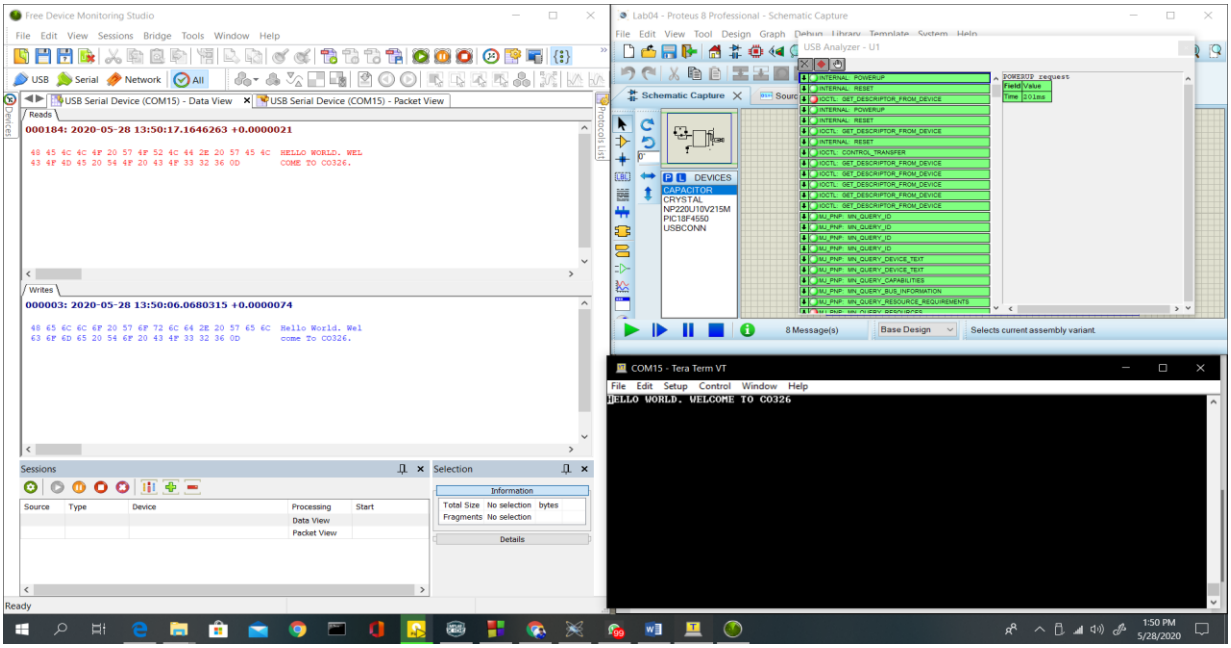
Report

Proteus Setup



Tera Term Terminal



MpLab Code

```
#include "system.h"
#include <stdint.h>
#include <string.h>
#include <stddef.h>
#include "usb.h"
#include "app_led_usb_status.h"
#include "app_device_cdc_basic.h"
#include "usb_config.h"

/** VARIABLES ******************************************************/

static bool buttonPressed;
static char buttonMessage[] = "Button pressed.\r\n";
static uint8_t readBuffer[CDC_DATA_OUT_EP_SIZE];
static uint8_t writeBuffer[CDC_DATA_IN_EP_SIZE];
uint8_t count = 0;

/*********************************************************************
 * Function: void APP_DeviceCDCBasicDemoInitialize(void);
 *
 * Overview: Initializes the demo code
 *
 * PreCondition: None
 *
 * Input: None
 *
 * Output: None
 *
```

```
 *****************************************************************/
void APP_DeviceCDCBasicDemoInitialize() {
   line_coding.bCharFormat = 0;
   line_coding.bDataBits = 8;
   line_coding.bParityType = 0;
   line_coding.dwDTERate = 9600;

   buttonPressed = false;
}

/*****************************************************************
 * Function: void APP_DeviceCDCBasicDemoTasks(void);
 *
 * Overview: Keeps the demo running.
 *
 * PreCondition: The demo should have been initialized and started via
 *   the APP_DeviceCDCBasicDemoInitialize() and APP_DeviceCDCBasicDemoStart() demos
 *   respectively.
 *
 * Input: None
 *
 * Output: None
 *
 *****************************************************************/
void APP_DeviceCDCBasicDemoTasks() {
   /* If the USB device isn't configured yet, we can't really do anything
    * else since we don't have a host to talk to.  So jump back to the
    * top of the while loop. */
```

```
if (mUSBUSARTIsTxTrfReady() == true) {
   putrsUSBUSART("\0");
}
while (1) {
   if (USBGetDeviceState() < CONFIGURED_STATE) {
      return;
   }

   /* If we are currently suspended, then we need to see if we need to
    * issue a remote wakeup.  In either case, we shouldn't process any
    * keyboard commands since we aren't currently communicating to the host
    * thus just continue back to the start of the while loop. */
   if (USBIsDeviceSuspended() == true) {
      return;
   }

   /* Check to see if there is a transmission in progress, if there isn't, then
    * we can see about performing an echo response to data received.
    */
   if (USBUSARTIsTxTrfReady() == true) {
      uint8_t i;
      uint8_t numBytesRead;

      numBytesRead = getsUSBUSART(readBuffer, sizeof (readBuffer));

      /* For every byte that was read... */
      for (i = 0; i < numBytesRead; i++, count++) {
         switch (readBuffer[i]) {
            case 'a' ... 'z':
```

```
                writeBuffer[count] = readBuffer[i] - 32;
                break;
              default:
                writeBuffer[count] = readBuffer[i];
                break;
            }
          }
          if (readBuffer[i] == 0X0A || readBuffer[i] == 0x0D) {
            /* After processing all of the received data, we need to send out
             * the "echo" data now.
             */
            putUSBUSART(writeBuffer, count);
            count = 0;
          }
        }
        CDCTxService();
      }
    }
```

Problems and issues

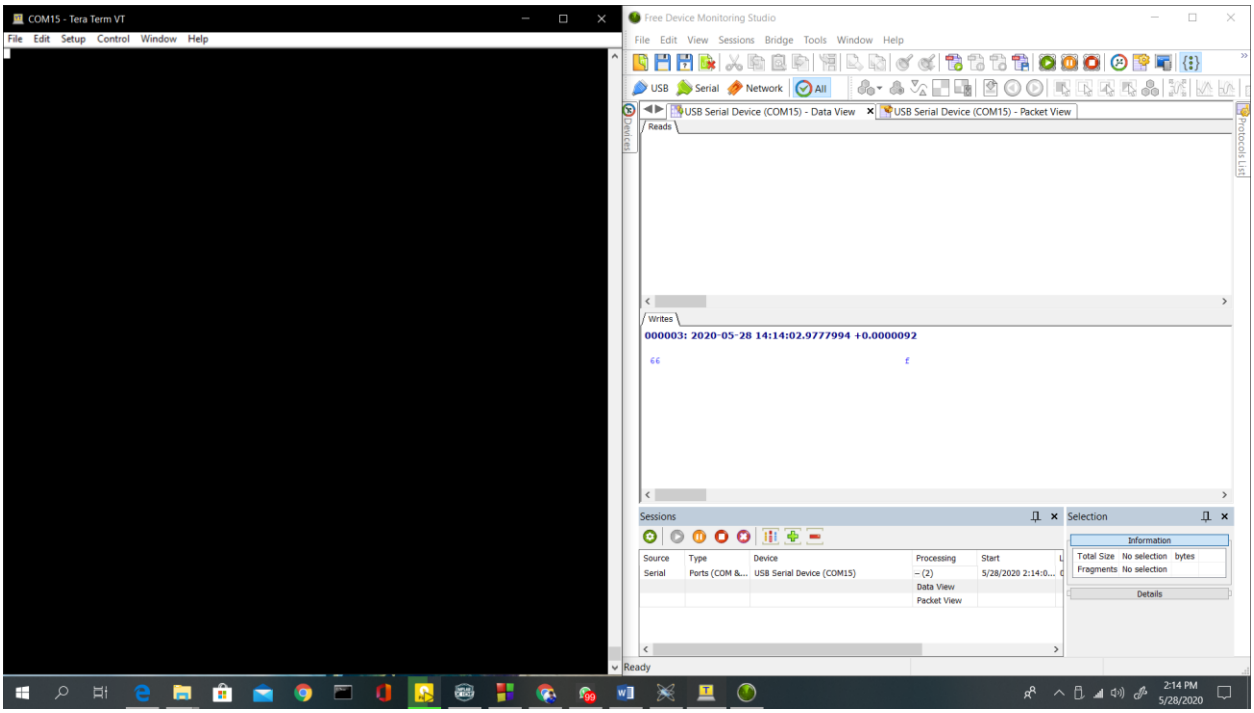
- Initially when building the MPLAB code some errors occurred. After changing the compiler to c90 standards the problem was resolved
- After a PC shutting down when the proteus setup run the USB device isn't shown at device manager and a warning is occurred in proteus saying the Virtual USB setup is unavailable. After reinstalling the virtual USB setup, the problem was resolved.

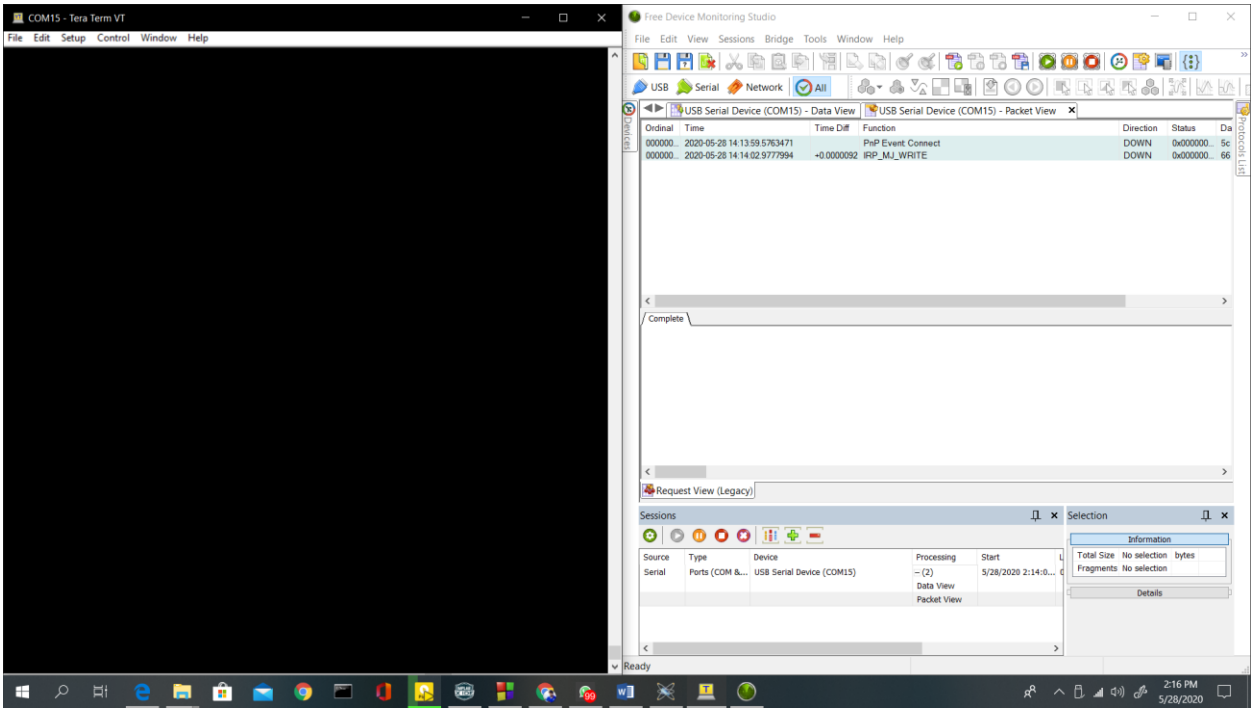
Explain followings

- Give a letter you typed and what is observed on the Tera Term

  The letter isn't shown at tera term, but in device monitoring studio, it is shown in writes that the letter is typed. Reason is the code upload to PIC microcontroller doesn't print until a enter is pressed

- Give screenshots of the USB monitor relevant to the letter you type and the letter displayed on the Tera Term.





- One type of packet is IN and other is OUT. Explain each case discussing why they become IN and OUT packets

In USB the LSB of the packet is transmitted first. An USB packet contains different fields. They are:

- **Sync**: It is a mandatory field occurring at starting of the packet. This field synchronizes the clock of the receiver with the transmitter. For low and full speed mode, this field is 8 bytes long and for high speed mode it is 32 bytes long.
- **PID**: PID means Packet ID. It indicates the packet type that is being sent. This field is of 8 bits long. The upper four bits identifies the type of packet and lower four bits are bit-wise compliment of upper four bits. The lower four bits helps in detecting errors.

Packet Type	PID Value	Packet Identifier
Token	0001	OUT Token
	1001	IN Token
	0101	SOF Token
	1101	SETUP Token
Data	0011	DATA0
	1011	DATA1
	0111	DATA2
	1111	MDATA
Handshake	0010	ACK
	1010	NAK
	1110	STALL
	0110	NYET
Special	1100	Preamble
	1100	ERR
	1000	Split
	0100	Ping

- **ADDR**: This field contains the designation address of the USB device. It is of 7 bits; this means it can support 27 ¬ = 127 devices.
- **ENDP**: This field specifies the endpoint number. It is of 4 bits; this means it can indicate 2¬¬4¬¬ ¬¬= 16 possible endpoints.
- **CRC:** CRC stands for Cyclic Redundancy Check. This field is used to check data in the packet for any error using CRC process
  For token packets, 5-bit CRC is used and for data packets 16-bit CRC is used

- **EOP**: EOP stands for End of Packet. This field signals the data lines for Single Ended Zero(SE0) for approximately 2 bit times, followed by J state (idle state) for 1-bit time

There are four types of data packets, in token packets:

- In – This packet notifies the USB device that host wants to read information.
- Out – This packet notifies the USB device that host wants to write information.