Search Medium

Pasindu_Sandima Follow
Aug 10, 2021 · 4 min read · Listen

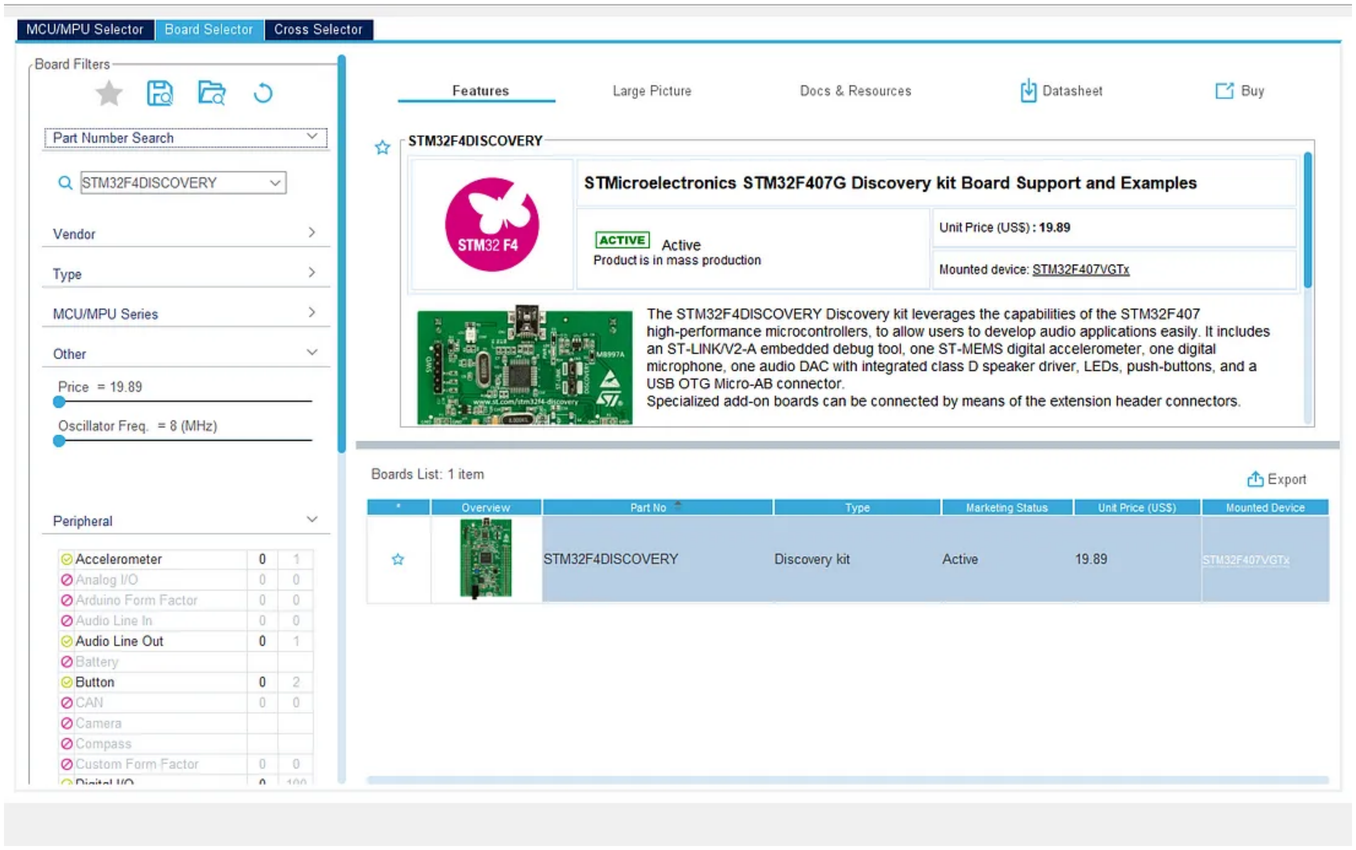
Save

STM32 USB Virtual COM Port (VCP)

This article is a tutorial on how to configure the USB peripheral of STM32 microcontrollers as a virtual COM port using the USB communication device class.

I'm using the STM32F4 Discovery board which has STM32F407VG micro-controller and an on-board micro USB port. I'm using the STM32 Cube Framework and the HAL libraries and USB Device middle-ware for this application

First open STM32 CubeIDE and start a new STM32 project and select the microcontroller of your preference. Here I will select STM32F407VG microcontroller.



STMicroelectronics STM32F407G Discovery kit Board Support and Examples

ACTIVE Active
Product is in mass production

Unit Price (US$) : 19.89

Mounted device: STM32F407VGTx

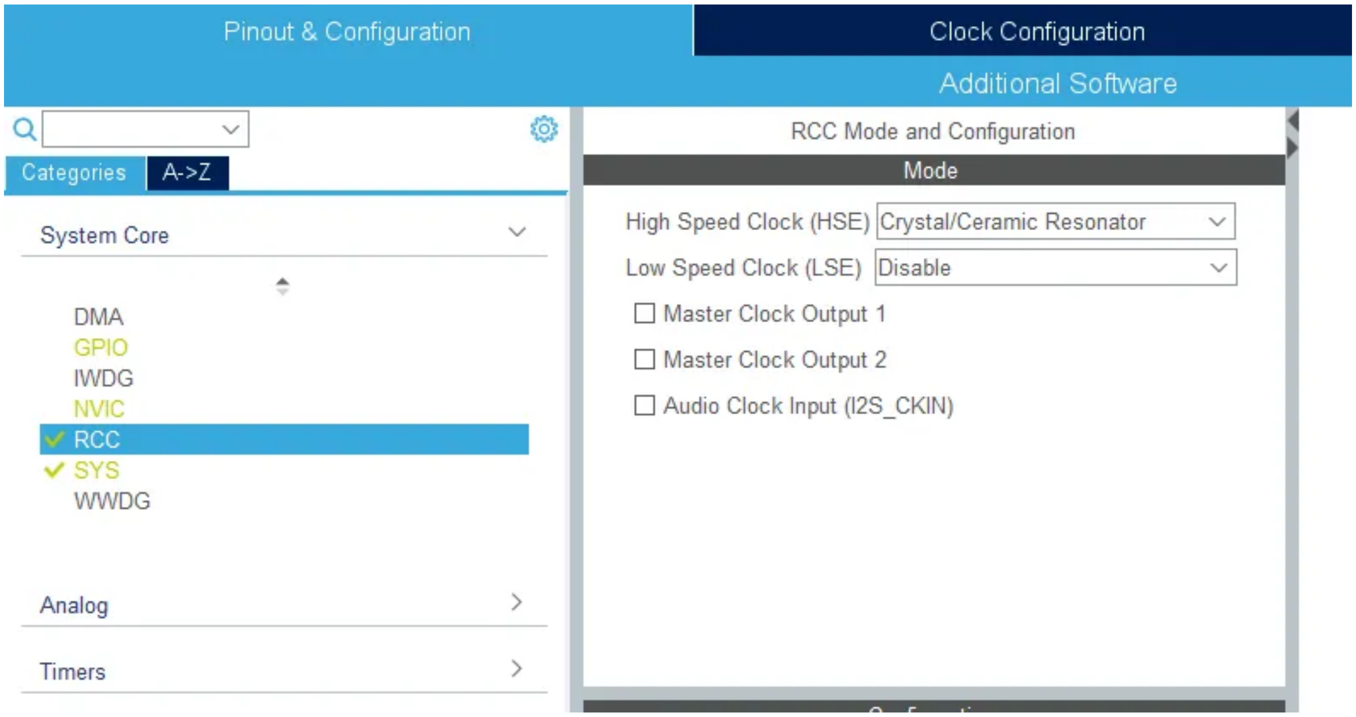
The STM32F4DISCOVERY Discovery kit leverages the capabilities of the STM32F407 high-performance microcontrollers, to allow users to develop audio applications easily. It includes an ST-LINK/V2-A embedded debug tool, one ST-MEMS digital accelerometer, one digital microphone, one audio DAC with integrated class D speaker driver, LEDs, push-buttons, and a USB OTG Micro-AB connector.
Specialized add-on boards can be connected by means of the extension header connectors.

Boards List: 1 item

*	Overview	Part No	Type	Marketing Status	Unit Price (US$)	Mounted Device
☆		STM32F4DISCOVERY	Discovery kit	Active	19.89	STM32F407VGTx

Selecting the microcontroller

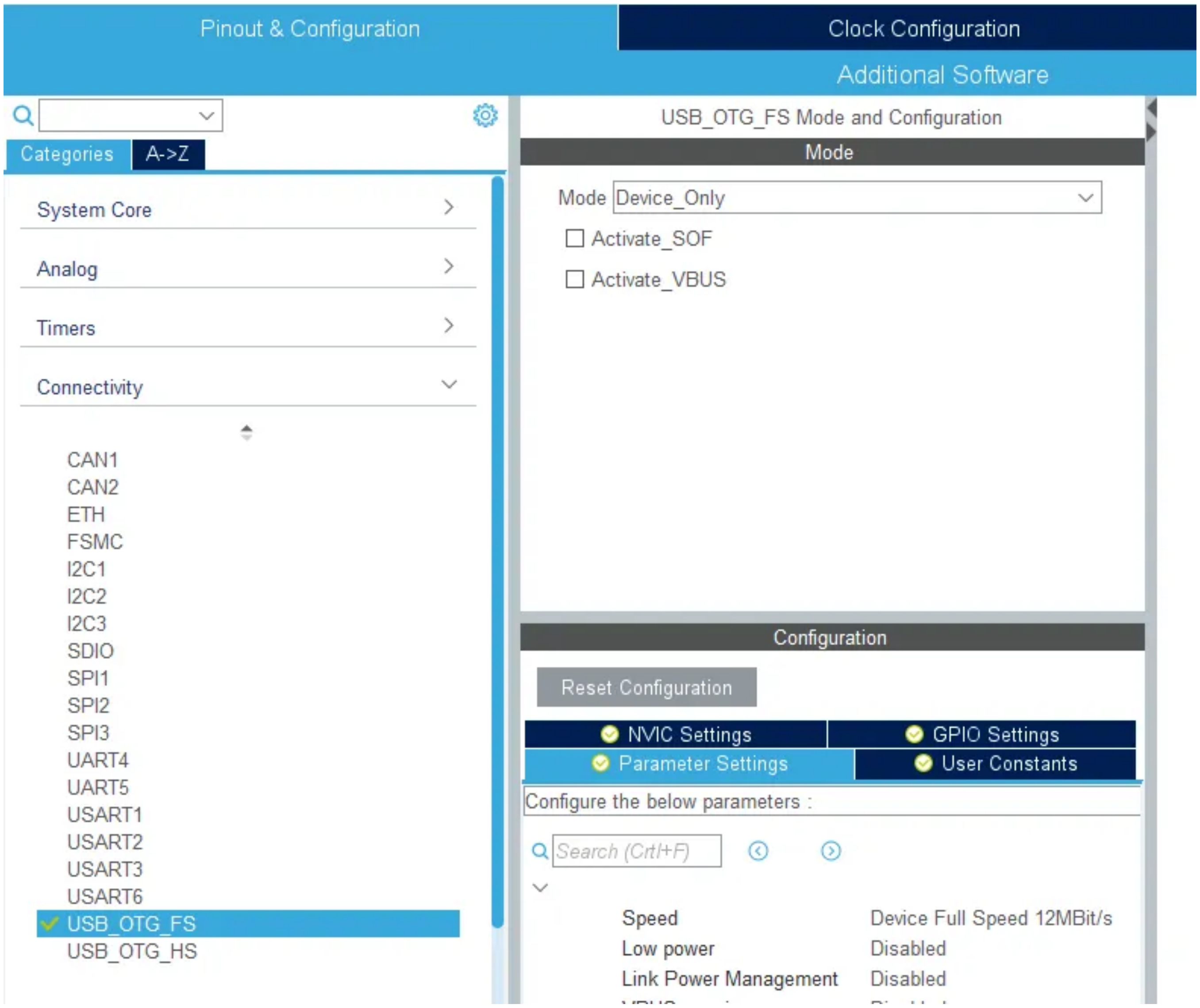
Next in the STM32 CubeMX perspective, enable the HSE (High Speed External) Clock and select Ceramic/Crystal Resonator (This will depend on your development board. If your board has a bypass clock source select that option) in the RCC settings of the System core section.



High Speed Clock (HSE) Crystal/Ceramic Resonator
Low Speed Clock (LSE) Disable
☐ Master Clock Output 1
☐ Master Clock Output 2
☐ Audio Clock Input (I2S_CKIN)

Selecting the HSE Clock Source

Next select the USB_OTG_FS under the connectivity section and select the mode as Device Only.



USB_OTG_FS Mode and Configuration

Mode

Mode Device_Only

☐ Activate_SOF

☐ Activate_VBUS

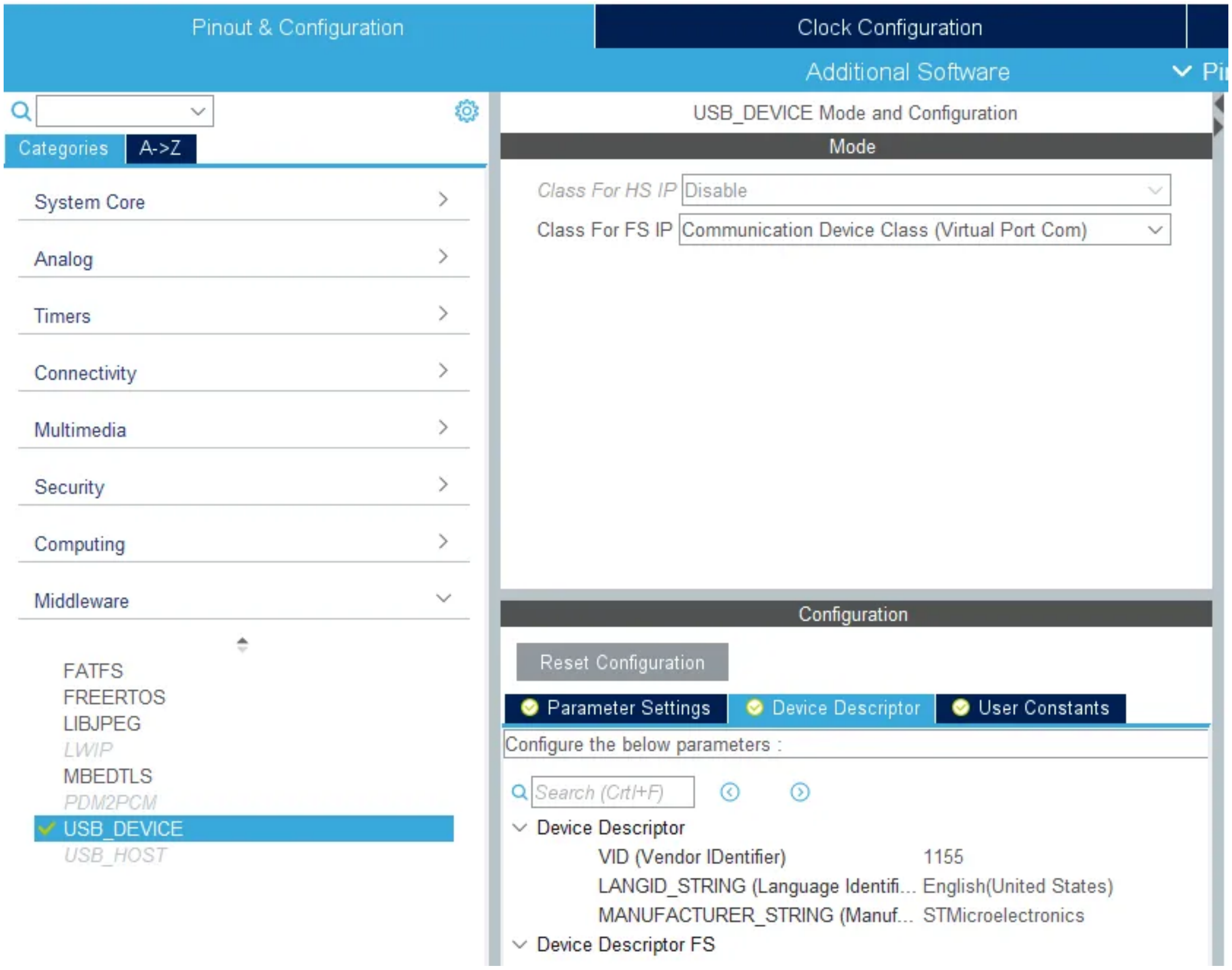
Configuration

Configure the below parameters :

Speed	Device Full Speed 12MBit/s
Low power	Disabled
Link Power Management	Disabled

Selecting the USB Device only mode

Finally, select the USB_DEVICE under the middleware section and select the class for FS IP as Communication Device Class (Virtual Port Com).



USB_DEVICE Mode and Configuration

Mode

Class For HS IP: Disable
Class For FS IP: Communication Device Class (Virtual Port Com)

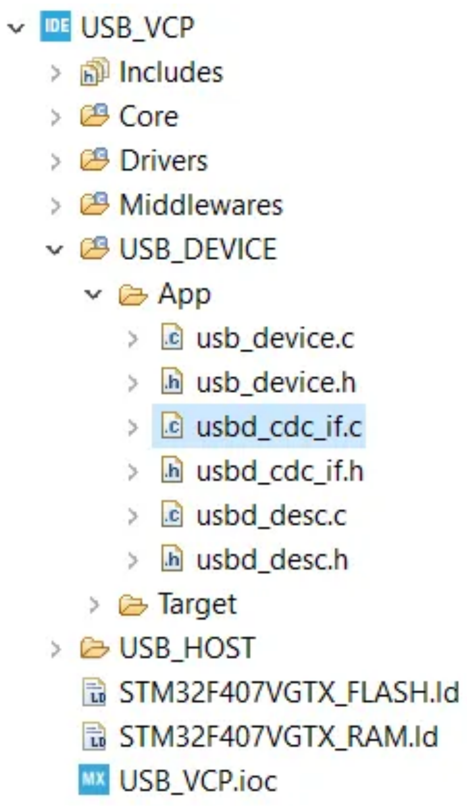
Configure the below parameters :

Device Descriptor
VID (Vendor IDentifier) 1155
LANGID_STRING (Language Identifi... English(United States)
MANUFACTURER_STRING (Manuf... STMicroelectronics
Device Descriptor FS

Selecting the USB Device Class

Next, save the CubeMX (.ioc) file to generate the code. Now all the peripherals are enabled for the USB communication over VCP. But as the HAL USB stack is heavy on the memory, the minimum heap size needed has to be changed. This can be changed by opening the STM32F407VGTX_FLASH.ld linker script file and changing the line 59 to `_Min_Heap_Size = 0x600;` .

Now as the code is generated let's make a simple echo functionality using the USB CDC. First open the usbd_cdc_if.c file.



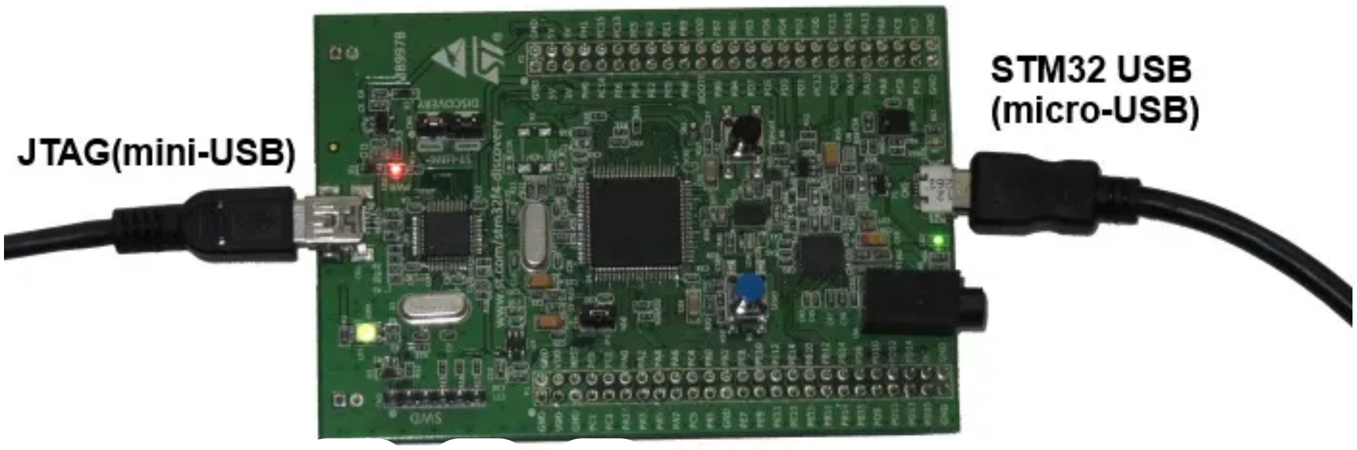
File Hierarchy

Navigate to the CDC_Receive_FS function and edit the function as follows.

```
static int8_t CDC_Receive_FS(uint8_t* Buf, uint32_t *Len)
{
  /* USER CODE BEGIN 6 */
  USBD_CDC_SetRxBuffer(&hUsbDeviceFS, &Buf[0]);
  USBD_CDC_ReceivePacket(&hUsbDeviceFS);
  CDC_Transmit_FS(Buf,*Len);
  return (USBD_OK);
  /* USER CODE END 6 */
}
```

CDC_Receive_FS hosted with ❤ by **GitHub** **view raw**

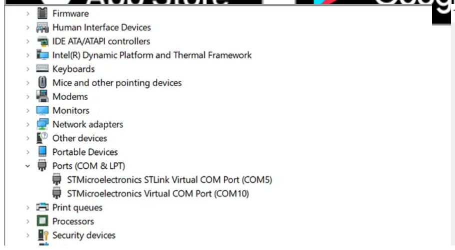
Now build the project and download the program to the device using the built in ST-Link. Then connect the board and the PC using the micro-USB port in the other end of the board.



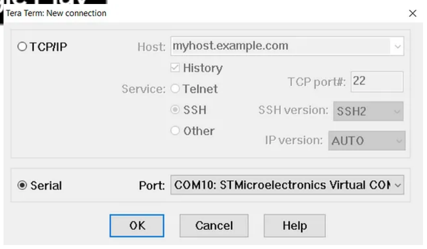
source (visual gdb)

Open your preferred serial monitor (Here I am using the Tera Term application) and select the COM port of the ST-microelectronics Virtual COM Port by using the device manager in Windows. Now you can [illegible] e terminal and the those keys will be echoed back by the device.

> *If the COM port is not detected,*
>
> *1. install the virtual COM port drivers from ST.*
>
> *2. Try reconnecting the USB cable and resetting the device.*



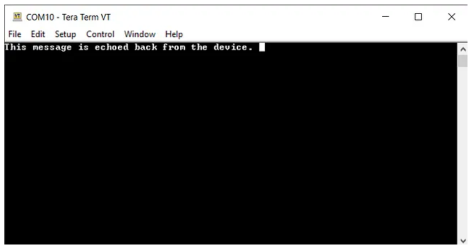
Firmware
Human Interface Devices
IDE ATA/ATAPI controllers
Intel(R) Dynamic Platform and Thermal Framework
Keyboards
Mice and other pointing devices
Modems
Monitors
Network adapters
Other devices
Portable Devices
Ports (COM & LPT)
STMicroelectronics STLink Virtual COM Port (COM5)
STMicroelectronics Virtual COM Port (COM10)
Print queues
Processors
Security devices



Tera Term: New connection

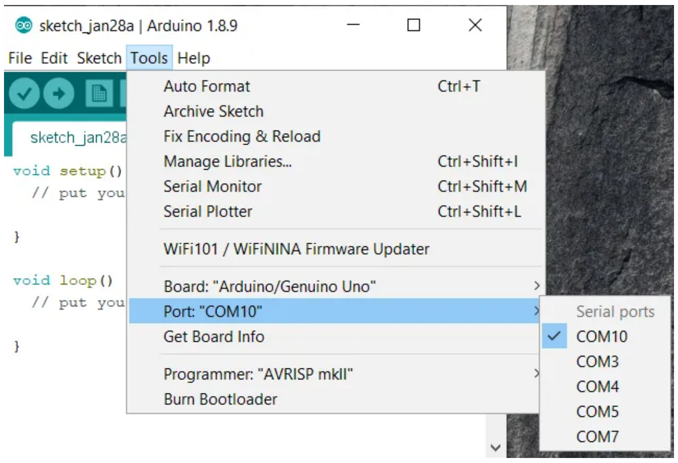
TCP/IP Host: myhost.example.com
History
Service: Telnet TCP port#: 22
SSH SSH version: SSH2
Other IP version: AUTO

Serial Port: COM10: STMicroelectronics Virtual CO



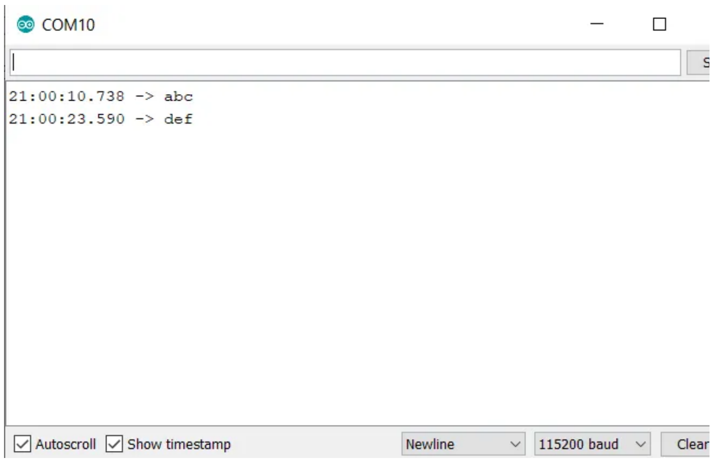
```
This message is echoed back from the device.
```

If you do not have any other serial monitors, you can still use the Arduino IDE's built in serial monitor. Select the port in the tools menu and open the serial monitor. Type a message in the serial monitor and press send. Then the device will echo back the message you typed.


sketch_jan28a | Arduino 1.8.9
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino/Genuino Uno"
Port: "COM10"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
COM10
COM3
COM4
COM5
COM7

```
void setup()
  // put you
}

void loop()
  // put you
}
```



```
21:00:10.738 -> abc
21:00:23.590 -> def
```

Controlling the built-in LED using VCP

To control the built in LED using virtual com port, first open the CubeMX (.ioc) file again t generate the code for the LED. In the CubeMX perspective, select the pin for the LED (whic is PD13 for the orange LED of the F4 discovery board) and set as GPIO_Output. Now save th file to generate the code again. Next change the CDC_Receive_FS function in the usbd_cdc_ file as follows.

```
static int8_t CDC_Receive_FS(uint8_t* Buf, uint32_t *Len)
{
  /* USER CODE BEGIN 6 */
  USBD_CDC_SetRxBuffer(&hUsbDeviceFS, &Buf[0]);
  USBD_CDC_ReceivePacket(&hUsbDeviceFS);
  if(Buf[0] == '1'){
         HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_SET);
  }
  else if(Buf[0] == '0'){
          HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_RESET);
  }
  return (USBD_OK);
  /* USER CODE END 6 */
}
```

CDC_Receive_FS hosted with ❤ by GitHub view raw

Build the project and download the program to the device. Open the serial monitor again ar send ‘1’ to light up the LED and ‘0’ to turn off the LED.