Experiment 2: Serial Communication

In this experiment, student will learn how to configurate the GPIO pins for the alternative function for a serial communication. By connecting KU Lab board to computer or notebook, the communication will be established. Student will learn how to use program on computer for communication and learn the importance of Baud rate and how to adjust it. Finally, student will learn how to create two ways communication.

1. Serial Communication Module

Unlike the old microcontroller technology, STM32F103C8T6 came with USART module which stands for the Universal Synchronous / Asynchronous Receiver / Transmitter. This module allows microcontroller to communication with other devices with both synchronous and asynchronous communication. In this topic, only UART (Universal Asynchronous Receiver / Transmitter) is focus because it is the simplest. The new notebook or desktop do not come with serial communication port or DB9, thus, the USB-to-Serial Module is required in this experiment as shown in Figure 1. This module has FTDI232RL as the converter and it will change the USB protocol into Serial protocol (UART) and become virtual com port device in the windows OS environment. The output of this module is serial signals, Tx and Rx, that can be connected into the STM32 GPIO pins directly. Tx is the signal for transmitting while Rx is the signal for receiving which can be considered Tx as the alternative output and Rx as the input for GPIO. According to Figure 2, Tx pin is needed to be set as alternative push-pull while Rx is needed to be set as input floating.

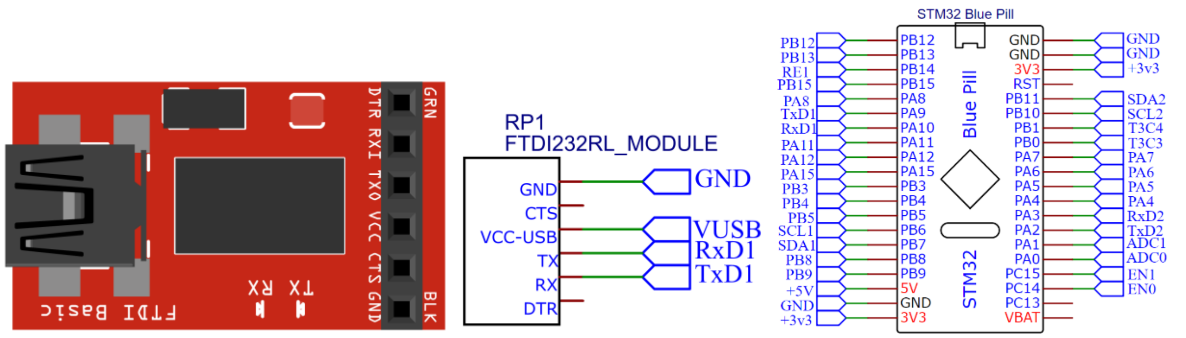
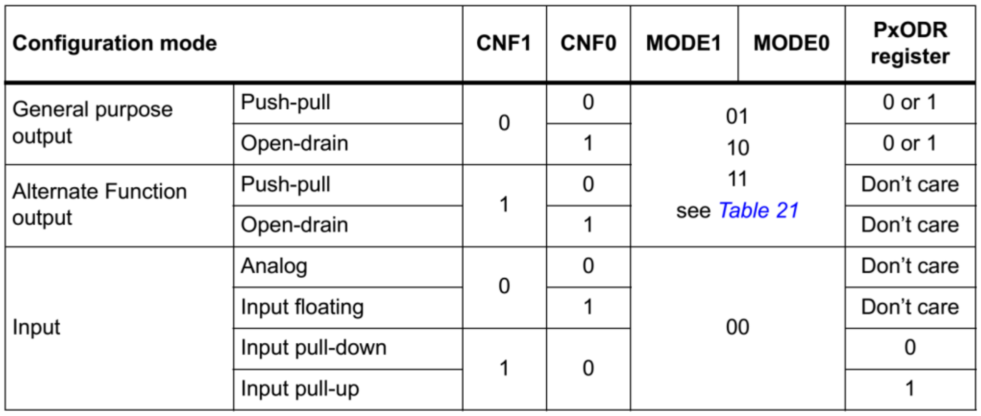

FTDI Basic
TX RX
DTR RXI TXO VCC CTS GND
GRN
BLK
RP1
FTDI232RL_MODULE
GND
CTS
VCC-USB
TX
RX
DTR
GND
VUSB
RxD1
TxD1
STM32 Blue Pill
Blue Pill
STM32
PB12 PB13 RE1 PB15 PA8 TxD1 RxD1 PA11 PA12 PA15 PB3 PB4 PB5 SCL1 SDA1 PB8 PB9 +5V GND +3v3
PB12 PB13 PB14 PB15 PA8 PA9 PA10 PA11 PA12 PA15 PB3 PB4 PB5 PB6 PB7 PB8 PB9 5V GND 3V3
GND GND 3V3 RST PB11 PB10 PB1 PB0 PA7 PA6 PA5 PA4 PA3 PA2 PA1 PA0 PC15 PC14 PC13 VBAT
GND GND +3v3 SDA2 SCL2 T3C4 T3C3 PA7 PA6 PA5 PA4 RxD2 TxD2 ADC1 ADC0 EN1 EN0

Figure 1. USB to Serial Module (FTDI232RL) and KU lab Board connection.



Configuration mode		CNF1	CNF0	MODE1	MODE0	PxODR register
General purpose output	Push-pull	0	0	01 10 11 see *Table 21*		0 or 1
	Open-drain		1			0 or 1
Alternate Function output	Push-pull	1	0			Don't care
	Open-drain		1			Don't care
Input	Analog	0	0	00		Don't care
	Input floating		1			Don't care
	Input pull-down	1	0			0
	Input pull-up					1

Figure 2. GPIO pin configuration.

Moveover, since serial communication module in STM32F103C8T6 belongs to the alternative function therefore it requires clock input via RCC_APB2PeriphClockCmd( ) like GPIOs.

Experiment 2.1

1. First, new project according to experiment 0 and assign project name as Lab2.
2. Write down the **DelayMC( )** function to the **main.c** file in the project before main( ) function as below:

```c
void DelayMC(unsigned int ms)
{
            volatile unsigned int nCount;
            nCount = (unsigned int)((72000000/4000000)*ms*1000 - 10);
            while(nCount--);
}
```

3. Write down the **GPIO_configuration** function to the **main.c** file in the project before main( ) function as below:

```c
void GPIO_Configuration()
{
 GPIO_InitTypeDef GPIO_InitStruct;
 // Enable clock for GPIOA and GPIO C and AFIO and USART1
 RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA | RCC_APB2Periph_GPIOC,ENABLE);
 RCC_APB2PeriphClockCmd(RCC_APB2Periph_AFIO | RCC_APB2Periph_USART1,ENABLE);
 // Configure PA9 as alternative output push pull
 GPIO_InitStruct.GPIO_Pin = GPIO_Pin_9;
 GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;
 GPIO_InitStruct.GPIO_Mode = GPIO_Mode_AF_PP;
 GPIO_Init(GPIOA, &GPIO_InitStruct);

// Configure PA10 as input floating
 GPIO_InitStruct.GPIO_Pin = GPIO_Pin_10;
 GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;
 GPIO_InitStruct.GPIO_Mode = GPIO_Mode_IN_FLOATING;
 GPIO_Init(GPIOA, &GPIO_InitStruct);
// Configure PC13 as output push pull 50MHz
 GPIO_InitStruct.GPIO_Pin = GPIO_Pin_13;
 GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;
 GPIO_InitStruct.GPIO_Mode = GPIO_Mode_Out_PP;
 GPIO_Init(GPIOC, &GPIO_InitStruct);

}
```

4. Add the **USART_setup( )** and **USART1_sendC( )** functions in **main.c** as follow:

```
void USART_setup()
{
 USART_InitTypeDef usart_init_struct;
 /* Baud rate 115200, 8-bit data, One stop bit
 * No parity, Do both Rx and Tx, No HW flow control*/
 usart_init_struct.USART_BaudRate = 115200;
 usart_init_struct.USART_WordLength = USART_WordLength_8b;
 usart_init_struct.USART_StopBits = USART_StopBits_1;
 usart_init_struct.USART_Parity = USART_Parity_No ;
 usart_init_struct.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
 usart_init_struct.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
 /* Configure USART1*/
 USART_Init(USART1, &usart_init_struct);
 /* Enable USART1 */
 USART_Cmd(USART1, ENABLE);
}
```

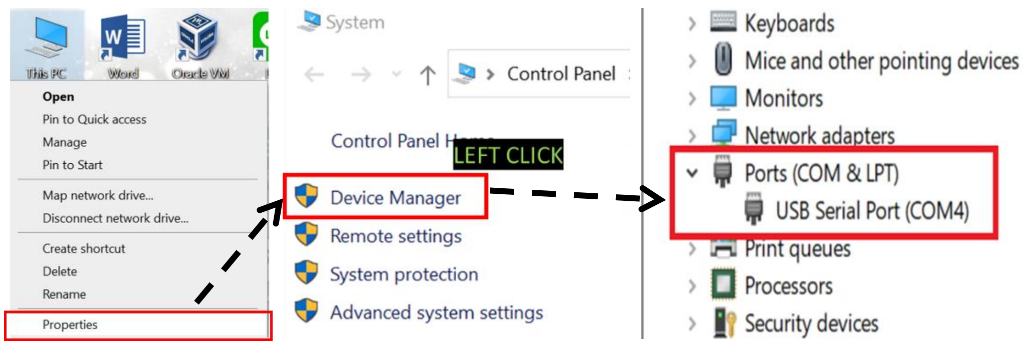
```
void USART1_sendC(unsigned char c)
{
   while(USART_GetFlagStatus(USART1,USART_FLAG_TXE)==RESET);
   USART_SendData(USART1,(unsigned char) c);
}
```

5. Edit the **main( )** function in **main.c** as follow:

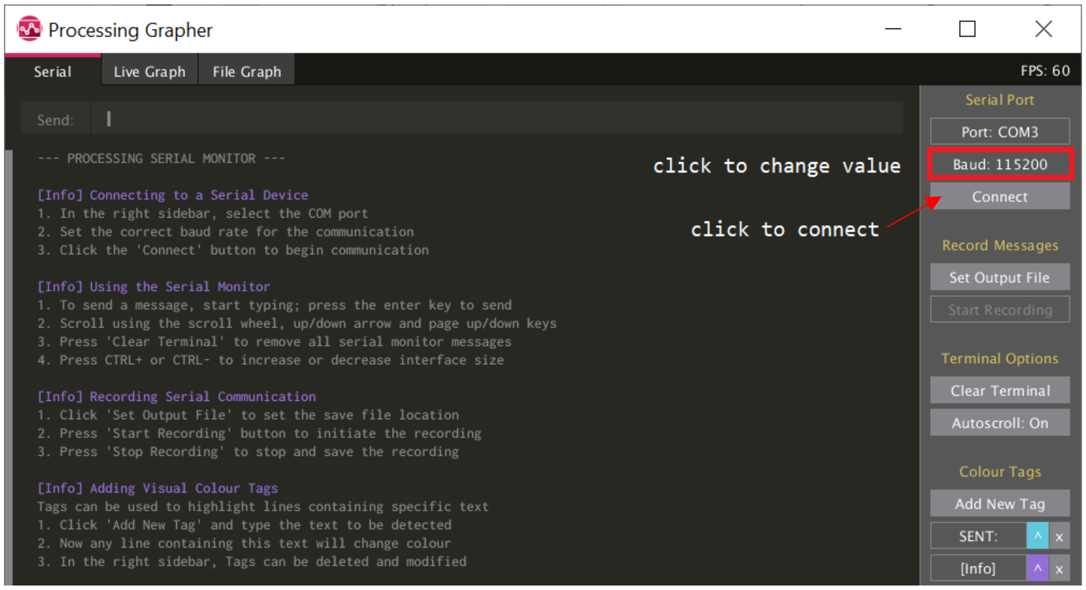
```
int main(void)
{
  unsigned char sdata = 0;
  GPIO_Configuration();
  GPIO_WriteBit(GPIOC,GPIO_Pin_13,0);
  USART_setup();
  sdata = 33;

```

```
    while(1)
    {
       USART1_sendC(sdata);
       sdata++;
       if (sdata >=  127)
       {
           sdata = 33;
           USART1_sendC('\n');
           USART1_sendC('\r');
        }
    }
}
```

6. Compile and program into the board. Connect the USB-to-Serial to desktop or notebook and find the virtual com port number via device manager.



Open ProcessingGrapher program, set to com port number and Baud rate to 115200 and click connect. Reset the STM32F103C8T6 and write down the result from the experiment 2.1 that show on ProcessingGrapher screen.



```
--- PROCESSING SERIAL MONITOR ---

[Info] Connecting to a Serial Device
1. In the right sidebar, select the COM port
2. Set the correct baud rate for the communication
3. Click the 'Connect' button to begin communication

[Info] Using the Serial Monitor
1. To send a message, start typing; press the enter key to send
2. Scroll using the scroll wheel, up/down arrow and page up/down keys
3. Press 'Clear Terminal' to remove all serial monitor messages
4. Press CTRL+ or CTRL- to increase or decrease interface size

[Info] Recording Serial Communication
1. Click 'Set Output File' to set the save file location
2. Press 'Start Recording' button to initiate the recording
3. Press 'Stop Recording' to stop and save the recording

[Info] Adding Visual Colour Tags
Tags can be used to highlight lines containing specific text
1. Click 'Add New Tag' and type the text to be detected
2. Now any line containing this text will change colour
3. In the right sidebar, Tags can be deleted and modified
```

____

____

____

____

____

____

Lab instructor / TA signature

2. Two ways communication

Two ways communication (not a full-duplex communication) is experimented in this topic. Microcontroller will receive the numbers that are sent from the desktop or notebook and convert them into the number. Microcontroller also echo each received character back to desktop or notebook in order to display on the screen. Finally, microcontroller will calculate and convert the result and sent it back to desktop or notebook screen. The following code demonstrates only an add operation.

Experiment 2.2

1. Use the same project as experiment 2.1. Add **USART1_getC( )**, **toi( )** , **USART1_putS( )** and **flush_USART1( )** function into the **main.c** as below:

```c
unsigned char USART1_getC()
{
   while(USART_GetFlagStatus(USART1,USART_FLAG_RXNE)==RESET);
   return(USART_ReceiveData(USART1));
}
```

```c
unsigned int toi(unsigned char c)
{
   unsigned int num=-1;
   if(c=='0') num=0;
   if(c=='1') num=1;
```

```
    if(c=='2') num=2;
    if(c=='3') num=3;
    if(c=='4') num=4;
    if(c=='5') num=5;
    if(c=='6') num=6;
    if(c=='7') num=7;
    if(c=='8') num=8;
    if(c=='9') num=9;
    return(num);
}
```

```
void USART1_putS(unsigned char *s)
{
   while (*s)
   {
       USART1_sendC(*s);
       s++; }
}
```

```
void flush_USART1(void)
{
   while(USART_GetFlagStatus(USART1,USART_FLAG_RXNE)==SET)
   // There are some data left in RX
   {
   USART_ReceiveData(USART1);  //Flush that data
   }
}
```

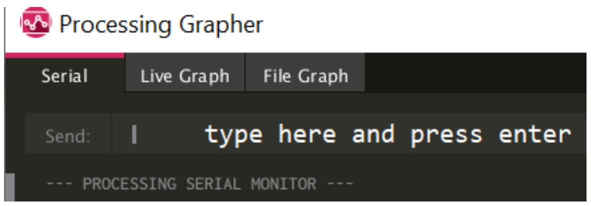
2. Edit the **main( )** function in **main.c** as follow:

```
int main(void)
{
  unsigned char sdata1[4]= {'0','0','0','0'}, sdata2[4]= {'0','0','0','0'};
  unsigned char *text1, sdata=0;
  unsigned int number1=0, number2=0, result =0, i1=0, i2=0, temp =0;
  static const char digits [] = "0123456789ABCDEF";
  GPIO_Configuration();
  GPIO_WriteBit(GPIOC,GPIO_Pin_13,0);
  USART_setup();
  text1 = "Please type number 1 and enter \n\r\0";
  USART1_putS(text1);
  i1=0;
  while(1)
  {
     sdata = USART1_getC();
     if ((sdata=='\n') || (sdata=='\r'))
         break;
     sdata1[i1] = sdata;
```

```
        i1++;
        if (i1>=4)
            break;
    }
    text1 = "Please type number 2 and enter \n\r\0";
    USART1_putS(text1);

    flush_USART1();

    i2=0;
    while(1)
    {
       sdata = USART1_getC();
       if ((sdata=='\n') || (sdata=='\r'))
            break;
        sdata2[i2] = sdata;
        i2++;
        if (i2>=4)
            break;
    }

    if(i1==1)
        number1=toi(sdata1[0]);
    if(i1==2)
        number1=toi(sdata1[0])*10+toi(sdata1[1]);
    if(i1==3)
        number1=toi(sdata1[0])*100+toi(sdata1[1])*10+toi(sdata1[2]);
    if(i1==4)
    {
        number1=toi(sdata1[0])*1000+toi(sdata1[1])*100;
        number1=number1+toi(sdata1[2])*10+toi(sdata1[3]);
    }

    if(i2==1)
        number2=toi(sdata2[0]);
    if(i2==2)
        number2=toi(sdata2[0])*10+toi(sdata2[1]);
    if(i2==3)
        number2=toi(sdata2[0])*100+toi(sdata2[1])*10+toi(sdata2[2]);
    if(i2==4)
    {
        number2=toi(sdata2[0])*1000+toi(sdata2[1])*100;
        number2=number2+toi(sdata2[2])*10+toi(sdata2[3]);
    }

    result=number1+number2;

    text1 = "The summation result is: \0";
    USART1_putS(text1);

    temp = (int) (result/10000)%10;
    USART1_sendC(digits[temp]);
    temp = (int) (result/1000)%10;
    USART1_sendC(digits[temp]);
    temp = (int) (result/100)%10;
    USART1_sendC(digits[temp]);
    temp = (int) (result/10)%10;
```

```
USART1_sendC(digits[temp]);
temp = (int) (result)%10;
USART1_sendC(digits[temp]);
USART1_sendC('\n');
USART1_sendC('\r');
while(1);

}
```

3. Compile and program into the board. Connect the USB-to-Serial to desktop or notebook and find the virtual com port number via device manager. Open ProcessingGrapher program, set to com port number and Baud rate to 115200 and click connect. Reset the STM32F103C8T6 and send number 1234 and 5678 to microcontroller using ProcessingGrapher, explain the result from the experiment 2.2 and explain the limitation of the experiment code 2.2 below.



____

____

____

____

____

____

____

4. Explain the function of **flush_USART1( )** and why we need this function in the experiment 2.2.

____

____

Lab instructor / TA signature

3. Chatbot

A chatbot program is experimented in this section. In place of direct communication with a live human agent, a chatbot is a software application that conducts an online chat conversation using text or text-to-speech. Experiment 2.3 will create a simple chatbot software via the serial communication. First start by user, type a question and send to microcontroller via ProcessingGrapher and if it is identical to the predefined question the microcontroller will response by specific answers, otherwise, microcontroller will response that it doesn't understand.

Experiment 2.3

1. Use the same project as experiment 2.2. Add **CheckQuestion( )** function into **main.c** as follow:

```c
unsigned char CheckQuestion(unsigned char *Ques, unsigned char *In)
{
   unsigned char result = 1, i = 0;
   while(Ques[i]!='\0')
   {
          if (Ques[i] != In[i])
                 result = 0;
          i++;
   }
   return(result);
}
```

2. Edit the **main( )** function in **main.c** as follow:

```c
int main(void)
{
  unsigned char quest[12]= {'0','0','0','0','0','0','0','0','0','0','0','0'};
  unsigned char *text1, sdata=0;
  unsigned char Q1[] = "Hello\0";
  unsigned char Q2[] = "Your name?\0";
  unsigned char Q3[] = "How old?\0";
  unsigned char Q4[] = "Bye\0";
  unsigned char A1[] = "Hello, too!\r\n\0";
  unsigned char A2[] = "I am Talkative.\r\n\0";
  unsigned char A3[] = "I am 9999 years old.\r\n\0";
  unsigned char A4[] = "Please Do not Go !\r\n\0";
  unsigned char Ax[] = "I do not Understand.\r\n\0";
  unsigned char result_compare = -1, iq=0, found = 0;

  GPIO_Configuration();
  GPIO_WriteBit(GPIOC,GPIO_Pin_13,0);
  USART_setup();
while(1)
 {
   text1 = "Please Ask the Question \n\r\0";
   USART1_putS(text1);
   flush_USART1();
   iq=0;
   while(1)
   {
      sdata = USART1_getC();
      if ((sdata=='\n') || (sdata=='\r'))
            break;
      quest[iq] = sdata;
      iq++;
      if (iq>=12)
            break;
   }
   found = 0;
   USART1_putS("Looking for Answer .... \r\n\0");
   result_compare = CheckQuestion(Q1,quest);
   if(result_compare)
   {  USART1_putS(A1);
      found = 1; }
   result_compare = CheckQuestion(Q2,quest);
   if(result_compare)
   {  USART1_putS(A2);
      found = 1; }
   result_compare = CheckQuestion(Q3,quest);
   if(result_compare)
   {  USART1_putS(A3);
      found = 1; }
   result_compare = CheckQuestion(Q4,quest);
   if(result_compare)
   {  USART1_putS(A4);
      found = 1; }
   if(found ==0)
      USART1_putS(Ax);
 }
}
```

3. Compile and program into the board. Explain the result from the experiment 2.3

____

____

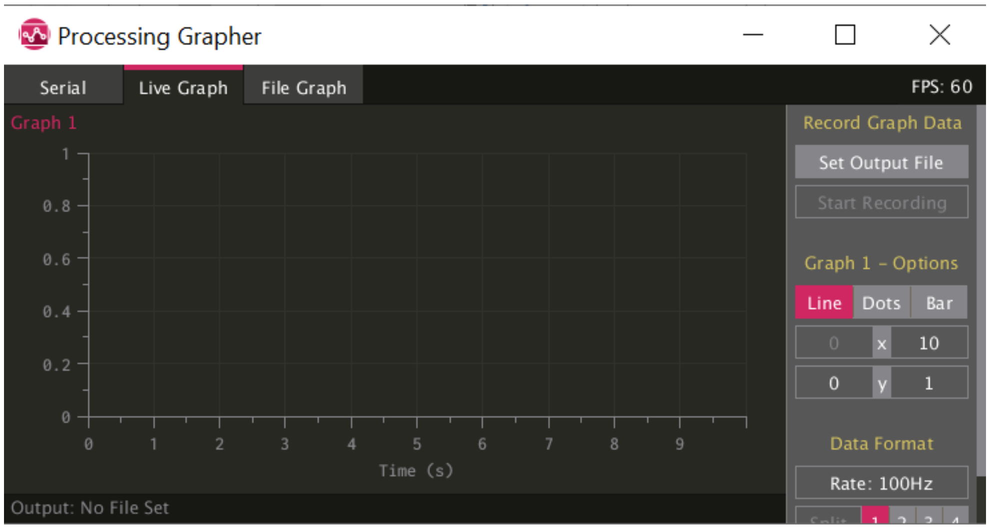
____

____

Lab instructor / TA signature

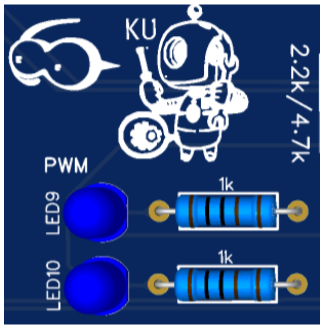
Homework

1. Write a program to generate sine wave that have amplitude of 1.2v with any frequency and send it's amplitude via serial communication to ProcessingGrapher to plot in Live Graph tab. Show the graph to the instructor or TA.



Lab instructor / TA signature

2. Write a program to control LED 9 and LED 10 on KU Lab board. When user send '9 on' then LED 9 will turn on and LED 9 will turn off when user send '9 off'. When user send '10 on' then LED 10 will turn on and LED 10 will turn off when user send '10 off'.



Lab instructor / TA signature