

<<IERG3810>>

<<Microcontroller and Embedded Systems Laboratory>>

Report on Experiment <<2>>

<<Universal Synchronous/Asynchronous

Receiver/Transmitter (USART) >>

Group: 19

Member: Chan Kai Yin (CUID: 1155124983)

Ip Tsun Yu (CUID: 1155144668)

Submission Date: 16-02-2022

Disclaimer

I declare that the assignment here submitted is original except for source material explicitly acknowledged, and that the same or related material has not been previously submitted for another course. I also acknowledge that I am aware of University policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the website <http://www.cuhk.edu.hk/policy/academichonesty/>

Jesse	Chan Kai Yin	16-02-2022
Signature	Name	Date
Derek	Ip Tsun Yu	16-02-2022
Signature	Name	Date

I. OBJECTIVES

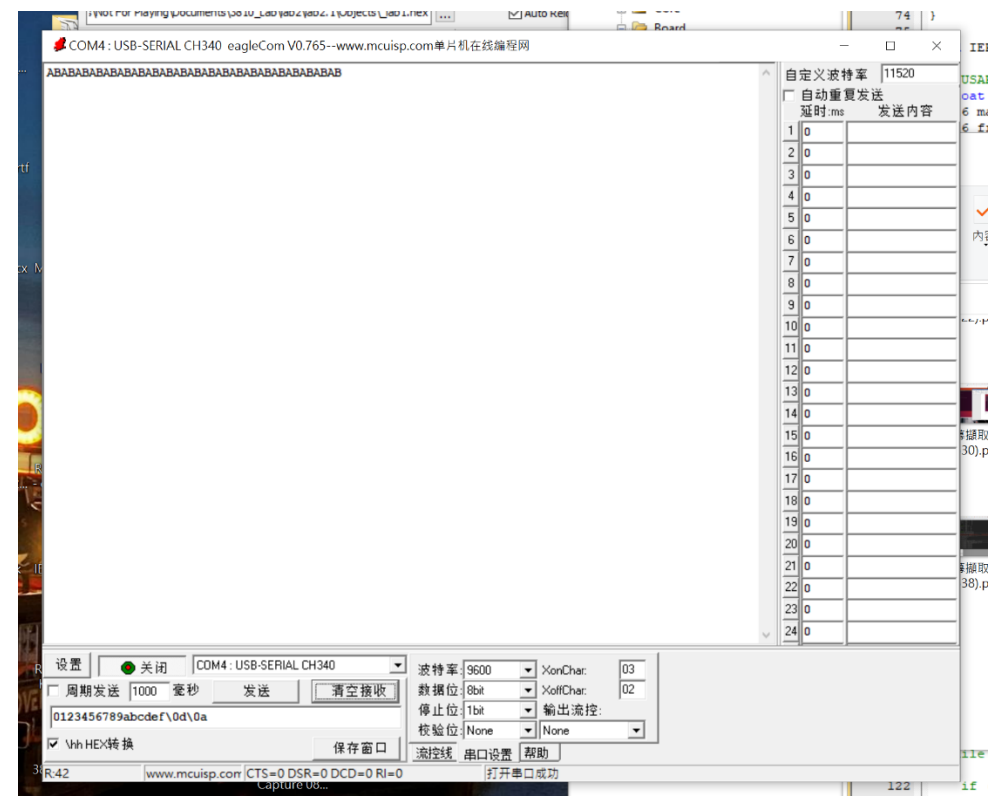
To study the clock tree of Cortex-M3.

To study the settings of USART transmitting.

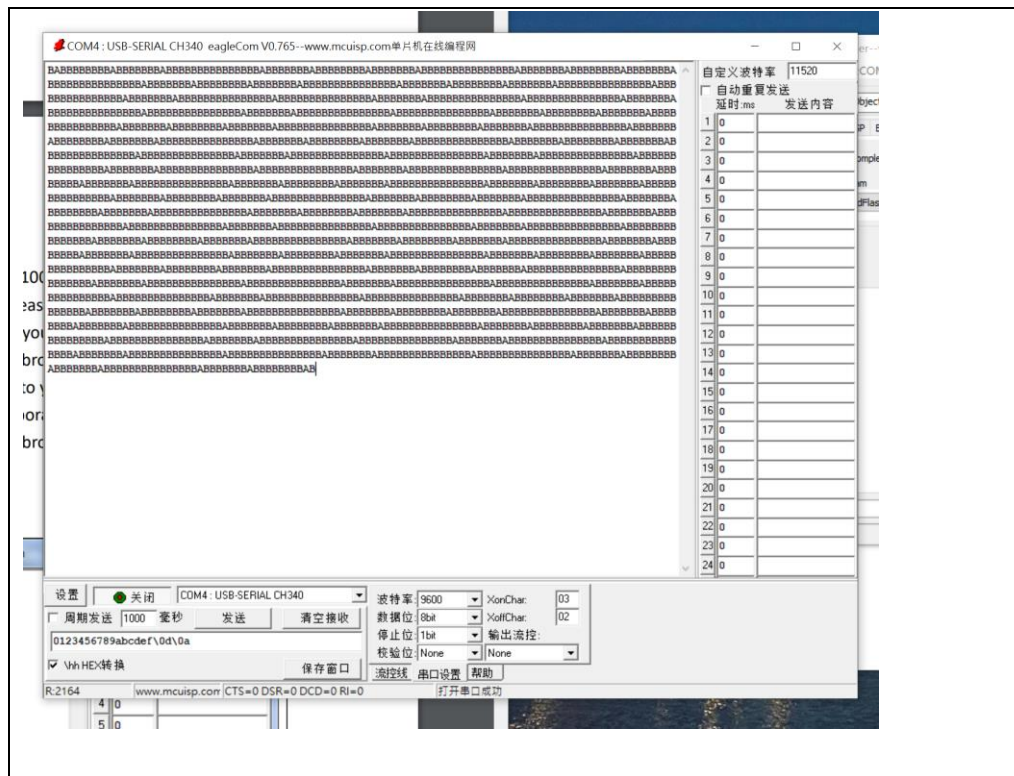
II. DATA ANALYSIS

Experiment 2.1

The alphabets AB shown on the software:



After modifying the figure of 'Delay(50000)' to 'Delay(100)':



Experiment 2.2

WithSID1155124983:

SART2 in experiment 2.1. USART1

The screenshot shows a serial terminal window titled "COM3: USB-SERIAL CH340 eagleCom V0.765--www.mcuisp.com单片机在线编程网". The main text area displays a long list of hexadecimal values, all appearing to be "1155124983" repeated many times. To the right of the text area is a table for configuring data transmission:

延时,ms	发送内容
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0

Below the table is a configuration panel with the following settings:

- 波特率: 9600
- 数据位: 8bit
- 停止位: 1bit
- 校验位: None
- 流控: None
- 串口设置: 帮助
- 串口已关闭

At the bottom, there is a status bar showing "ing TXE R:740" and a message "Undefined identifier - function 'Message'" in the background code editor.

```
#include "stm32f10x.h"
#include "IERG3810_LED.h"
#include "IERG3810_Buzzer.h"
#include "IERG3810_KEY.h"

void IERG3810_clock_tree_init(void);
void IERG3810_USART2_init(u32, u32);
void Delay(u32);

void Delay(u32 count)
{
    u32 i;
```

```

        for(i=0;i<count;i++);
    }

void IERG3810_clock_tree_init(void){
    u8 PLL=7;
    unsigned char temp=0;
    RCC->CFGR &= 0xF8FF0000;

    RCC->CR &= 0XFEF6FFFF;
    RCC->CR |= 0x00010000;
    while(!(RCC->CR>>17));
    RCC->CFGR=0x00000400;
    RCC->CFGR|=PLL<<18;
    RCC->CFGR|=1<<16;

    FLASH->ACR|=0x32; // 72
    RCC->CR|=0x01000000;
    while(!(RCC->CR>>25));
    RCC->CFGR|=0x00000002;

    while(temp!=0x02)
    {
        temp=RCC->CFGR>>2;
        temp&=0x03;
    }
}

void IERG3810_USART2_init(u32 pclk1, u32 bound)
{
    //USART2
    float temp;
    u16 mantissa;
    u16 fraction;
    temp= (float) (pclk1*1000000)/(bound*16);
    mantissa = temp;
    fraction = (temp - mantissa)*16;
    mantissa <= 4;
}

```

```

    mantissa += fraction;
    RCC -> APB2ENR |= 1<<2;
    RCC -> APB1ENR |= 1<<17;
    GPIOA -> CRL &= 0xFFFF00FF;
    GPIOA -> CRL |= 0X00008B00;
    RCC -> APB1RSTR |= 1<<17;
    RCC -> APB1RSTR &= ~(1<<17);
    USART2 -> BRR = mantissa;
    USART2 -> CR1 |= 0X2008;
}

void IERG3810_USART1_init(u32 pclk2, u32 bound)
{
    float temp;
    u16 mantissa;
    u16 fraction;
    temp = (float) (pclk2*1000000)/(bound*16);
    mantissa = temp;
    fraction = (temp - mantissa)*16;
    mantissa<<= 4;
    mantissa += fraction;
    RCC->APB2ENR |= 1<<2;
    RCC->APB2ENR |= 1<<14;
    GPIOA->CRH &= 0xFFFFF00F;
    GPIOA->CRH |= 0X000008B0;
    RCC->APB2RSTR |= 1<<14;
    RCC->APB2RSTR &= ~(1<<14);
    USART1->BRR = mantissa;
    USART1->CR1 |=0X2008;
}

int main(void)
{
    IERG3810_clock_tree_init();
    IERG3810_USART2_init(36, 9600);
    IERG3810_USART1_init(72, 9600);
}

```

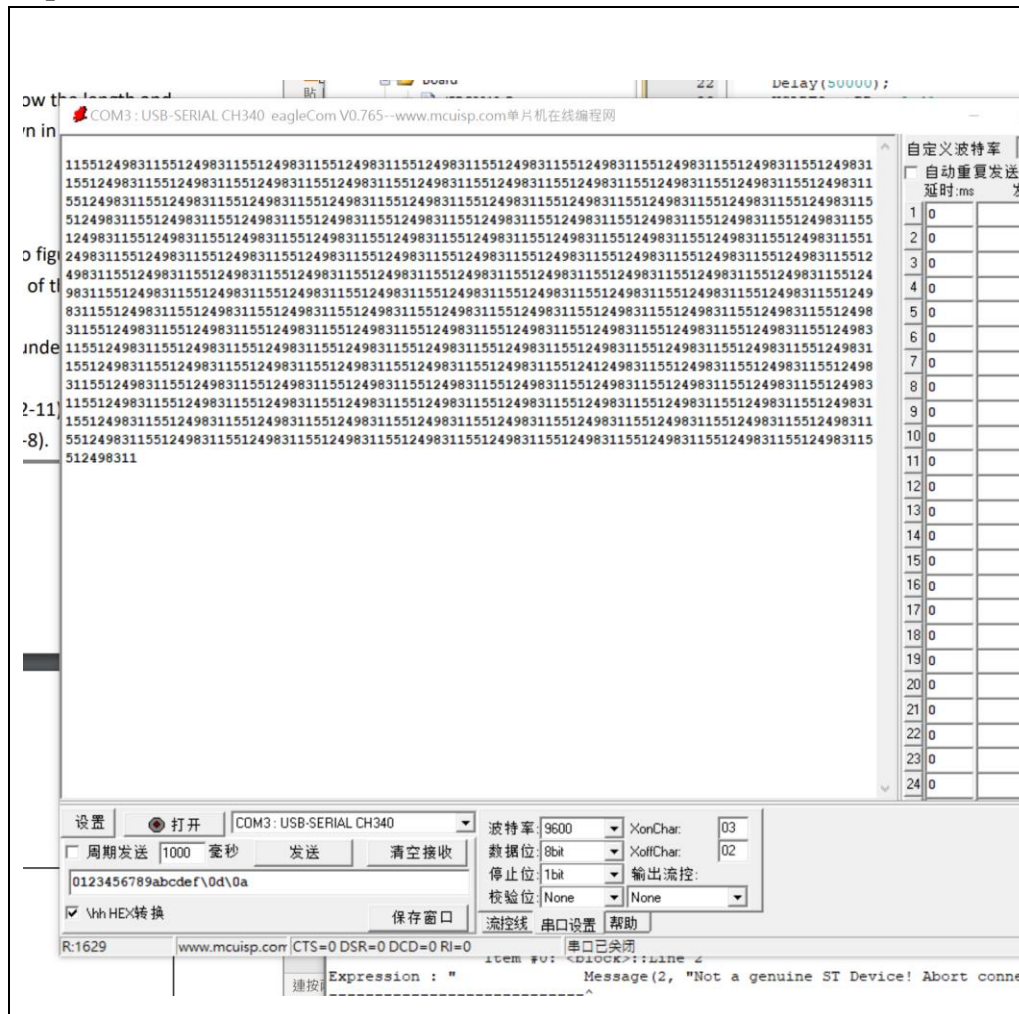
```
while(1)
{
    //USART2 ->DR = 0x41; // A
    Delay(50000);
    //USART2 ->DR = 0x42; // B

    // 1155124983
    USART1 -> DR = 0x31;
    Delay(50000);
    USART1 -> DR = 0x31;
    Delay(50000);
    USART1 -> DR = 0x35;
    Delay(50000);
    USART1 -> DR = 0x35;
    Delay(50000);
    USART1 -> DR = 0x31;
    Delay(50000);
    USART1 -> DR = 0x32;
    Delay(50000);
    USART1 -> DR = 0x34;
    Delay(50000);
    USART1 -> DR = 0x39;
    Delay(50000);
    USART1 -> DR = 0x38;
    Delay(50000);
    USART1 -> DR = 0x33;
    Delay(50000);

    Delay(1000000);

}
}
```

Experiment 2.3



Replace the Delay(50000) command in USART_print()

```
void USART_print(u8 USARTport, char *st)
```

```
{
```

```
    u8 i=0;
```

```
    while (st[i] != 0x00)
```

```
    {
```

```
        if (USARTport == 1) USART1->DR = st[i];
```

```
        if (USARTport == 2) USART2->DR = st[i];
```

```
        while (((USART1 -> SR) & (0x00000080))==0);
```

```
        while (((USART2 -> SR) & (0x00000080))==0);
```



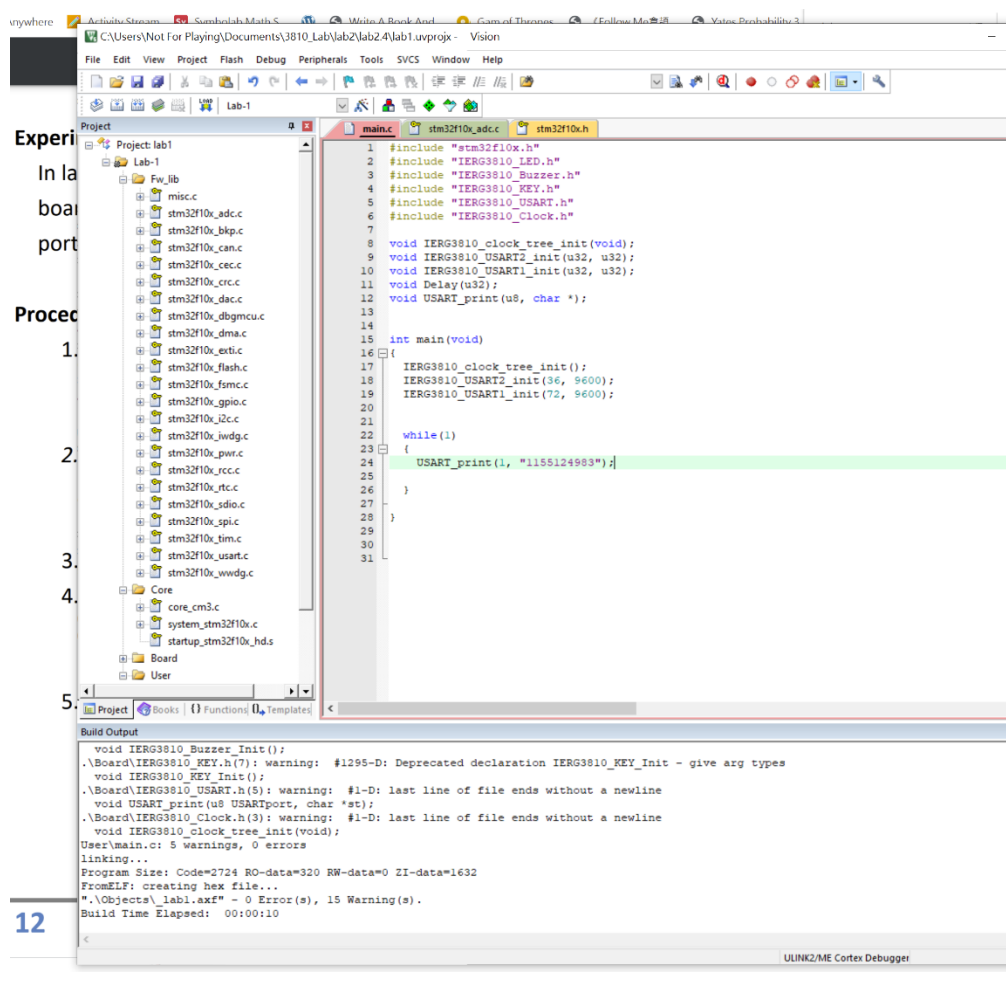
```

//Delay(50000);
while (!(USART1->SR << 7));
if (i == 255) break;
i++;
}
}

```

Experiment 2.4

Only keep 'main()' function in 'main.c' file. Compile the program and download it to the board without error.



The screenshot shows the STM32CubeIDE IDE with the project 'lab1' open. The 'main.c' file is selected, and the code is as follows:

```

1 #include "stm32f10x.h"
2 #include "IERG3810_LED.h"
3 #include "IERG3810_Buzzer.h"
4 #include "IERG3810_KEY.h"
5 #include "IERG3810_USART.h"
6 #include "IERG3810_Clock.h"
7
8 void IERG3810_clock_tree_init(void);
9 void IERG3810_USART2_init(u32, u32);
10 void IERG3810_USART1_init(u32, u32);
11 void Delay(u32);
12 void USART_print(u8, char *);
13
14
15 int main(void)
16 {
17     IERG3810_clock_tree_init();
18     IERG3810_USART2_init(36, 9600);
19     IERG3810_USART1_init(72, 9600);
20
21     while(1)
22     {
23         USART_print(1, "1155124983");
24     }
25 }
26
27
28
29
30
31

```

The 'Build Output' window shows the following messages:

```

void IERG3810_Buzzer_Init();
.\Board\IERG3810_KEY.h(7): warning: #1295-D: Deprecated declaration IERG3810_KEY_Init - give arg types
void IERG3810_KEY_Init();
.\Board\IERG3810_USART.h(5): warning: #1-D: last line of file ends without a newline
void USART_print(u8 USARTport, char *s);
.\Board\IERG3810_Clock.h(3): warning: #1-D: last line of file ends without a newline
void IERG3810_Clock_tree_init(void);
User\main.c: 5 warnings, 0 errors
linking...
Program Size: Code=2724 RO-data=320 RW-data=0 ZI-data=1632
FromELF: creating hex file...
".\Objects\lab1.axf" - 0 Error(s), 15 Warning(s).
Build Time Elapsed: 00:00:10

```

12

ULINK2/ME Cortex Debugger

III. DISCUSSION

Exp 2.1

<Step 8 : Describe what has happened and the reason of an error of the serial communication (check eagleCom's screen.)>

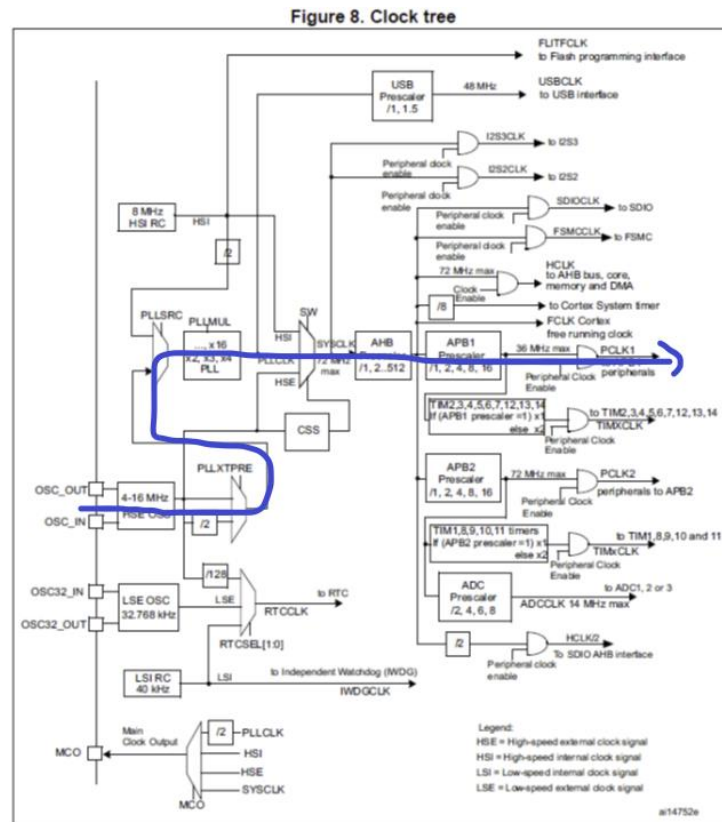
The speed of popping is faster than before. Also, A is popping up once combining with multiple B and repeated. It should have the same effect as previous which is repeating "ABAB". It may due to data corruption because of the pervious character has not been sent and the new character already launch.

< Step 9: Describe the setting and procedures of the subroutine

'IERG3810_clock_tree_init()>

It clears first 16 bits and conf of clock output on CFGR, setting PLL off, Clock detector OFF, HSE oscillator ON. Check for internal 8 MHz RC oscillator ready, HCLK divided by 2, PLL input clock x 9, HSE clock divided by 2, Two wait states, if $48 \text{ MHz} < \text{SYSCLK} \leq 72 \text{ MHz}$. PLL ON, check PLL clock ready flag, PLL selected as system clock. While loop until the PLL used as system clock, set the temp = System clock switch status and keep last two bits.

<The signal path>



< Step 10: Describe the setting and procedures of the subroutine
'IERG3810_USART2_init()>

Baud is calculated by the above formula. It further divided into integer and fraction and let them as mantissa and fraction. GPIOA and USART2 are enabled. Setting USART2 as alternative functions with PA2 and PA3. Set the mantissa of USARTDIV and Transmitter enable.

Exp 2.3

<Step 4 : Try to understand the function of USART_print().>

The function takes the port number of USART and the target printed char. Then it iterates the string char by char. It sends to the specify port number then wait for data is transferred to the shift register. It continues to send the next char in the string until the last char is send.

IV. SUMMARY

We learn the setting of clock tree and USART transmitting of Cortex-M3.

V. DIVISION OF WORK

<Lab work: Jesse & Derek, Report writing: Jesse>

VI. REFERENCES