EIE3810 Microprocessor System Design Laboratory

**Laboratory Report #2**

Name: 关嘉俊

Student ID: 116010060

Date: January 19th

The Chinese University of Hong Kong, Shenzhen

 Experiment A: e.g. learn to set up the USART2 by setting the corresponding register, and observe the image on the oscilloscope when sending the message in the serial port.

 Experiment B: e.g. Similarly, change the corresponding register used in the first experiment to set up USART1, with 9600bps, 8N1 with 72MHz. Also we need to figure out how to solve the USB-to-Serial occupation’s conflict.

 Experiment C: e.g. Instead of using the delay function, we can check the TXE bit to ensure the last string has been sent. Also we would modify the USART\_print function to simplify the code.

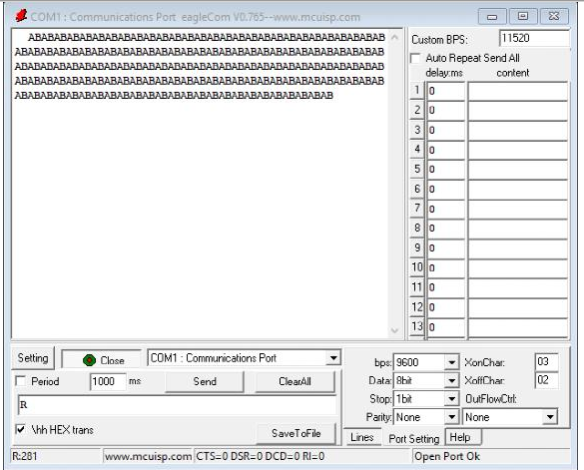
 Experiment D: e.g Modify the program from experiment C and create files EIE3810\_Clock.c and EIE3810\_USART.c. Finally meet the requests and compile the download.

**I. Experiment A**

**(a)Basic procedure**

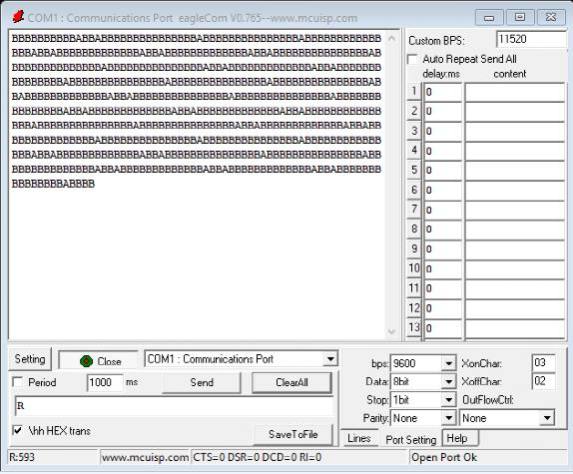
1.Create new project, download main.c from Blackboard and add the code 2.Connect a USB-to-Serial cable to the board and the computer, run the program and check the eagleCom. 3.Observe the eagleCom interface and use the Oscilloscope to measure it. 4.Change the delay time with smaller interval, observe the eagleCom interface.

**(b)Raw data**



In the figure we can see that, the letter A and B are sent and received in turns repeatly. The experimental result meet the expectation. What’s more, we change the delay time from 50000 to 100 to see what effect would that make.

Figure 1 eagleCom Interface record



From the figure we can see that, A B are sent irregularly compared with the last figure. The reason is that actually the register USART2->DR are still not ready when it turn to A, and then B would be sent continuously.

Figure 2 Delay time reduce to 100

(c) Experiment code:

#include "stm32f10x.h"

void EIE3810\_clock\_tree\_init(void);

void EIE3810\_USART2\_init(u32, u32);

void Delay(u32);

int main(void)

{

EIE3810\_clock\_tree\_init();

EIE3810\_USART2\_init(36, 9600);

while(1)

{

USART2->DR = 0x41;//0x41 represent A in ASCII

Delay(50000); //to make sure the former has been transformed

USART2->DR = 0x42;//0x42 represent "B"

Delay(50000);

Delay(1000000);} }

void Delay(u32 count)

{

u32 i;

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

}

void EIE3810\_clock\_tree\_init(void)

{

u8 PLL=7; //which is 0b0111

u8 temp=0;

RCC->CR |= 0x00010000; //HSE\_ON is at the bit-16,set it to 1 to enable HSE

while(!((RCC->CR>>17)&0x1));//wait until HSE is ready(the HSERDY flag to be 1)

RCC->CFGR &= 0xFFFDFFFF; //set PLLXTPRE(bit-17 in CFGR) to 0, HSE not divided.(16-19 is 0b1101 = 0xD)

RCC->CFGR |= 1<<16; //set bit-16 in CFGR(PLLSRC) to 1, select HSE as PLL input clock

RCC->CFGR |= PLL<<18; //set PLLMUL to 0111 to multiple the clock by 9

RCC->CR |=0x01000000;//set bit-24 in CR to 1 to enable PLL

while(!(RCC->CR>>25));//wait until the PLLRDY flag to be 1 when PLL is locked

RCC->CFGR &=0xFFFFFFFE;//set bit-0 in CFGR to 0 (con.)

RCC->CFGR |=0x00000002;//set bit-1 in CFGR to 1 to select PLL as system clock ?

//CFGR == 0b\*\*\*\* \*\*\*\* \*\*01 1101 \*\*\*\* \*\*\*\* \*\*\*\* \*\*10

while(temp !=0x02) //Use a while loop to make sure bit-3 and bit-2 equal to 10(PLL selected as system clock)

{

temp=RCC->CFGR>>2;

temp &= 0x03; //assign bit-3 and bit-2 to temp in binary

}

RCC->CFGR &= 0xFFFFFC0F;//set from bit-4 to bit-9 to 0

RCC->CFGR |= 0x00000400;//set bit-10 to 1

//PPRE1 become 100, HCLK devided by 2; HPRE become 0000, SYSCLK not divided

FLASH->ACR = 0x32;//Set FLASH with 2 wait states

RCC->APB1ENR |= 1<<17; //to enable USART2}

void EIE3810\_USART2\_init(u32 pclk1, u32 baudrate)

{ //USART2

float temp;

u16 mantissa;

u16 fraction;

temp=(float) (pclk1\*1000000)/(baudrate\*16); //calculate UARTDIV in decimal

mantissa=temp; //save integer portion (12 bits)

fraction=(temp-mantissa)\*16;//take fraction portion (4 bits) and save it as interger

mantissa<<=4;

mantissa+=fraction;//Convert them together

RCC->APB2ENR |= 1<<2; //enable GPIOA

GPIOA->CRL &= 0xFFFF00FF; //initialize PA2 and PA3

GPIOA->CRL |= 0x00008B00; //set PA2 altermate function ouput push-pull with 50MHz; PA3 to input with pull-up/pull-down

RCC->APB1RSTR |= 1<<17; //reset USART2 by setting bit-17 to 1

RCC->APB1RSTR &= ~(1<<17); //clear the reset status by settign bit-17 to 0

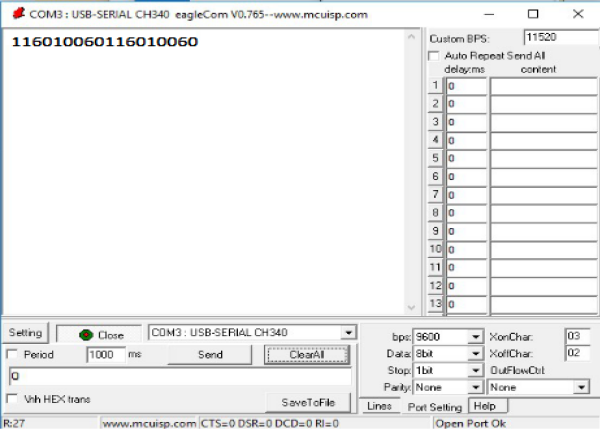
USART2->BRR=mantissa;//set baud rate register according to the binary number converted from UARTDIV

USART2->CR1=0x2008; //enable USART2,set word length, disable parity, enable transmitter by assign 0b0010 0000 0000 1000 to CR1

}

**II. Experiment B**

1. Procedure (1)By checking the register table, modify the code in the last program to initialize the clock tree and the USART1 (2) Select the method 1 or the method 2 mentioned in the handout. (3)Send the CUHK(SZ) student ID through USART1 approximately one second after “RESET” key released.
2. Raw material



From the result in the eaglecom we can see that, the initialization and the set up work ideally.

****

Figure 3 Oscilloscope figure

1. Code information:

#include "stm32f10x.h"

void EIE3810\_clock\_tree\_init(void);

//void EIE3810\_USART2\_init(u32, u32);

void EIE3810\_USART1\_init(u32, u32);

void Delay(u32);

int main(void)

{

EIE3810\_clock\_tree\_init();

//EIE3810\_USART2\_init(36, 9600);

EIE3810\_USART1\_init(72, 9600);

while(1)

{

Delay(1000000)

USART1->DR = 0x31;//send "1"

Delay(50000);

USART1->DR = 0x31;//send "1"

Delay(50000);

USART1->DR = 0x36;//send "6"

Delay(50000);

USART1->DR = 0x30;//send "0"

Delay(50000);

USART1->DR = 0x31;//send "1"

Delay(50000);

USART1->DR = 0x30;//send "0"

Delay(50000);

USART1->DR = 0x30;//send "0"

Delay(50000);

USART1->DR = 0x36;//send "6"

Delay(50000);

USART1->DR = 0x30;//send "0"

Delay(50000);

break;

}

}

void Delay(u32 count)

{

u32 i;

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

}

void EIE3810\_clock\_tree\_init(void)

{

u8 PLL=7; //0b0111

u8 temp=0;

RCC->CR |= 0x00010000; //set bit-16 in CR to 1 to enable HSE

while(!((RCC->CR>>17)&0x1));//wait until the HSERDY flag to be 1 when HSE is ready,the loop will break

RCC->CFGR &= 0xFFFDFFFF; //set bit-17 in CFGR(PLLXTPRE) to 0, HSE not divided.

RCC->CFGR |= 1<<16; //set bit-16 in CFGR(PLLSRC) to 1, select HSE as PLL input clock

RCC->CFGR |= PLL<<18; //set PLLMUL to 0111 to multiple the clock by 9

RCC->CR |=0x01000000;//set bit-24 in CR to 1 to enable PLL

while(!(RCC->CR>>25));//wait until the PLLRDY flag to be 1 when PLL is locked

RCC->CFGR &=0xFFFFFFFE;//set bit-0 in CFGR to 0 (con.)

RCC->CFGR |=0x00000002;//set bit-1 in CFGR to 1 to select PLL as system clock

//CFGR == 0b\*\*\*\* \*\*\*\* \*\*01 1101 \*\*\*\* \*\*\*\* \*\*\*\* \*\*10

while(temp !=0x02) //check if bit-3 and bit-2 equal to 10(PLL selected as system clock)

{

temp=RCC->CFGR>>2;

temp &= 0x03; //assign bit-3 and bit-2 to temp in binary

}

RCC->CFGR &= 0xFFFFFC0F;//set from bit-4 to bit-9 to 0

RCC->CFGR |= 0x00000400;//set bit-10 to 1

//PPRE1 become 100, HCLK devided by 2(36MHz); HPRE become 0000, SYSCLK not divided

RCC->CFGR &= 0xFFFFDFFF;//Important chane: set bit-13 to 0 to PPRE2 to 0xx (no divided, 72MHz)

FLASH->ACR = 0x32;//Set FLASH with 2 wait states

//RCC->APB1ENR |= 1<<17; //to enable USART2

RCC->APB2ENR |= 1<<14; //to enable USART1

}

void EIE3810\_USART1\_init(u32 pclk2, u32 baudrate)

{

//USART1

float temp;

u16 mantissa;

u16 fraction;

temp=(float) (pclk2\*1000000)/(baudrate\*16); //calculate UARTDIV in decimal

mantissa=temp; //take integer portion (12 bits)

fraction=(temp-mantissa)\*16;//take fraction portion (4 bits)

mantissa<<=4;

mantissa+=fraction;//Convert them together

RCC->APB2ENR |= 1<<2; //enable GPIOA

GPIOA->CRH &= 0xFFFFF00F; //initialize PA9 and PA10

GPIOA->CRH |= 0x000008B0; //set PA9 altermate function ouput push-pull with 50MHz;

//PA10 to input with pull-up/pull-down

RCC->APB2RSTR |= 1<<14; //reset USART1 by setting bit-14 to 1

RCC->APB2RSTR &= ~(1<<14); //clear the reset status

USART1->BRR=mantissa;//set baud rate register according to the binary number converted from UARTDIV

USART1->CR1=0x2008; //enable USART1,set word length(1 start bit, 8 data bit, n stop bit);

//disable parity, enable transmitter by assigning 0b0010 0000 0000 1000 to CR1

}

**III. Experiment C**

1. Procedure: 1.Similar to the last two experiments, we first set up the tree-clock initialization and USART initialization. 2. Different form the last two experiments, without using delay function which can assure that the last letter has been sent, we check the TXE bit and create a new function USART\_print and send the ID number.
2. The flowchart is shown as follow:

i=st

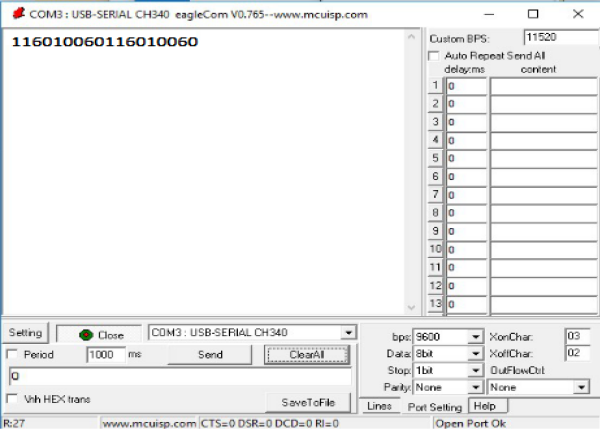
Sent st[i]

Check bit TXE

i == 255

break

i = i + 1



1. Experiment Code:

#include "stm32f10x.h"

void EIE3810\_clock\_tree\_init(void);

//void EIE3810\_USART2\_init(u32, u32);

void EIE3810\_USART1\_init(u32, u32);

void Delay();

void USART\_print(u8 USARTport, char \*st)

{

u8 i = 0;

while(st[i] != 0x00)

{

if (USARTport == 1){

USART1->DR = st[i];

while (!((USART1->SR>>7)&0x01));

}

if (USARTport == 2){

USART2->DR = st[i];

while (!((USART2->SR>>7)&0X01));

}

if (i==255) break;

i++;

}

}

int main(void)

{

EIE3810\_clock\_tree\_init();

//EIE3810\_USART2\_init(36, 9600);

EIE3810\_USART1\_init(72, 9600);

Delay(10000000);

USART\_print(1,"116010060");

}

void Delay(u32 count)

{

u32 i;

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

}

void EIE3810\_clock\_tree\_init(void)

{

u8 PLL=7; //0b0111

u8 temp=0;

RCC->CR |= 0x00010000; //set bit-16 in CR to 1 to enable HSE

while(!((RCC->CR>>17)&0x1));//wait until the HSERDY flag to be 1 when HSE is ready,the loop will break

RCC->CFGR &= 0xFFFDFFFF; //set bit-17 in CFGR(PLLXTPRE) to 0, HSE not divided.

RCC->CFGR |= 1<<16; //set bit-16 in CFGR(PLLSRC) to 1, select HSE as PLL input clock

RCC->CFGR |= PLL<<18; //set PLLMUL to 0111 to multiple the clock by 9

RCC->CR |=0x01000000;//set bit-24 in CR to 1 to enable PLL

while(!(RCC->CR>>25));//wait until the PLLRDY flag to be 1 when PLL is locked

RCC->CFGR &=0xFFFFFFFE;//set bit-0 in CFGR to 0 (con.)

RCC->CFGR |=0x00000002;//set bit-1 in CFGR to 1 to select PLL as system clock

//CFGR == 0b\*\*\*\* \*\*\*\* \*\*01 1101 \*\*\*\* \*\*\*\* \*\*\*\* \*\*10

while(temp !=0x02) //check if bit-3 and bit-2 equal to 10(PLL selected as system clock)

{

temp=RCC->CFGR>>2;

temp &= 0x03; //assign bit-3 and bit-2 to temp in binary

}

RCC->CFGR &= 0xFFFFFC0F;//set from bit-4 to bit-9 to 0

RCC->CFGR |= 0x00000400;//set bit-10 to 1

//PPRE1 become 100, HCLK devided by 2(36MHz); HPRE become 0000, SYSCLK not divided

RCC->CFGR &= 0xFFFFDFFF;//Important chane: set bit-13 to 0 to PPRE2 to 0xx (no divided, 72MHz)

FLASH->ACR = 0x32;//Set FLASH with 2 wait states

//RCC->APB1ENR |= 1<<17; //to enable USART2

RCC->APB2ENR |= 1<<14; //to enable USART1

}

void EIE3810\_USART1\_init(u32 pclk2, u32 baudrate)

{

//USART1

float temp;

u16 mantissa;

u16 fraction;

temp=(float) (pclk2\*1000000)/(baudrate\*16); //calculate UARTDIV in decimal

mantissa=temp; //take integer portion (12 bits)

fraction=(temp-mantissa)\*16;//take fraction portion (4 bits)

mantissa<<=4;

mantissa+=fraction;//Convert them together

RCC->APB2ENR |= 1<<2; //enable GPIOA

GPIOA->CRH &= 0xFFFFF00F; //initialize PA9 and PA10

GPIOA->CRH |= 0x000008B0; //set PA9 altermate function ouput push-pull with 50MHz;

//PA10 to input with pull-up/pull-down

RCC->APB2RSTR |= 1<<14; //reset USART1 by setting bit-14 to 1

RCC->APB2RSTR &= ~(1<<14); //clear the reset status

USART1->BRR=mantissa;//set baud rate register according to the binary number converted from UARTDIV

USART1->CR1=0x2008; //enable USART1,set word length(1 start bit, 8 data bit, n stop bit);

//disable parity, enable transmitter by assigning 0b0010 0000 0000 1000 to CR1

}

**IV. Experiment D**

1. Procedure 1.Summary the code above and create the own library. 2.Use the function in the library to achieve the goal that USART1 send 1234567890 and USART2 116010060 (my student ID)
2. Raw data

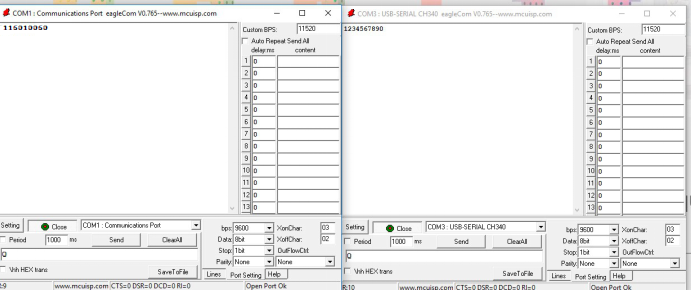


Figure 4 Raw data of the serial information

1. Experiment code

#include "stm32f10x.h"

#include "EIE3810\_CLOCK.h"

#include "EIE3810\_USART.h"

int main(void)

{

EIE3810\_clock\_tree\_init();

EIE3810\_USART2\_init(36, 9600);

EIE3810\_USART1\_init(72, 9600);

USART\_print(1,"1234567890");

USART\_print(2,"115010184");

}