МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА АВТОМАТИЗОВАНОГО УПРАВЛІННЯ ТЕХНІЧНИМИ СИСТЕМАМИ

Лабораторна робота № 2

Варіант 1

По дисципліні «Програмування мікроконтролерних систем»

Тема: «Програмування мікроконтролера з використанням типової бібліотеки СMSIS і дослідження послідовного інтерфейсу USART (UART)»

Виконали:

студенти групи ІТ-51

Цитовцева А.С.

Бессмертний Р.С.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис, дата)

Перевірив:

ст. викладач кафедри АУТС Катін П. Ю.

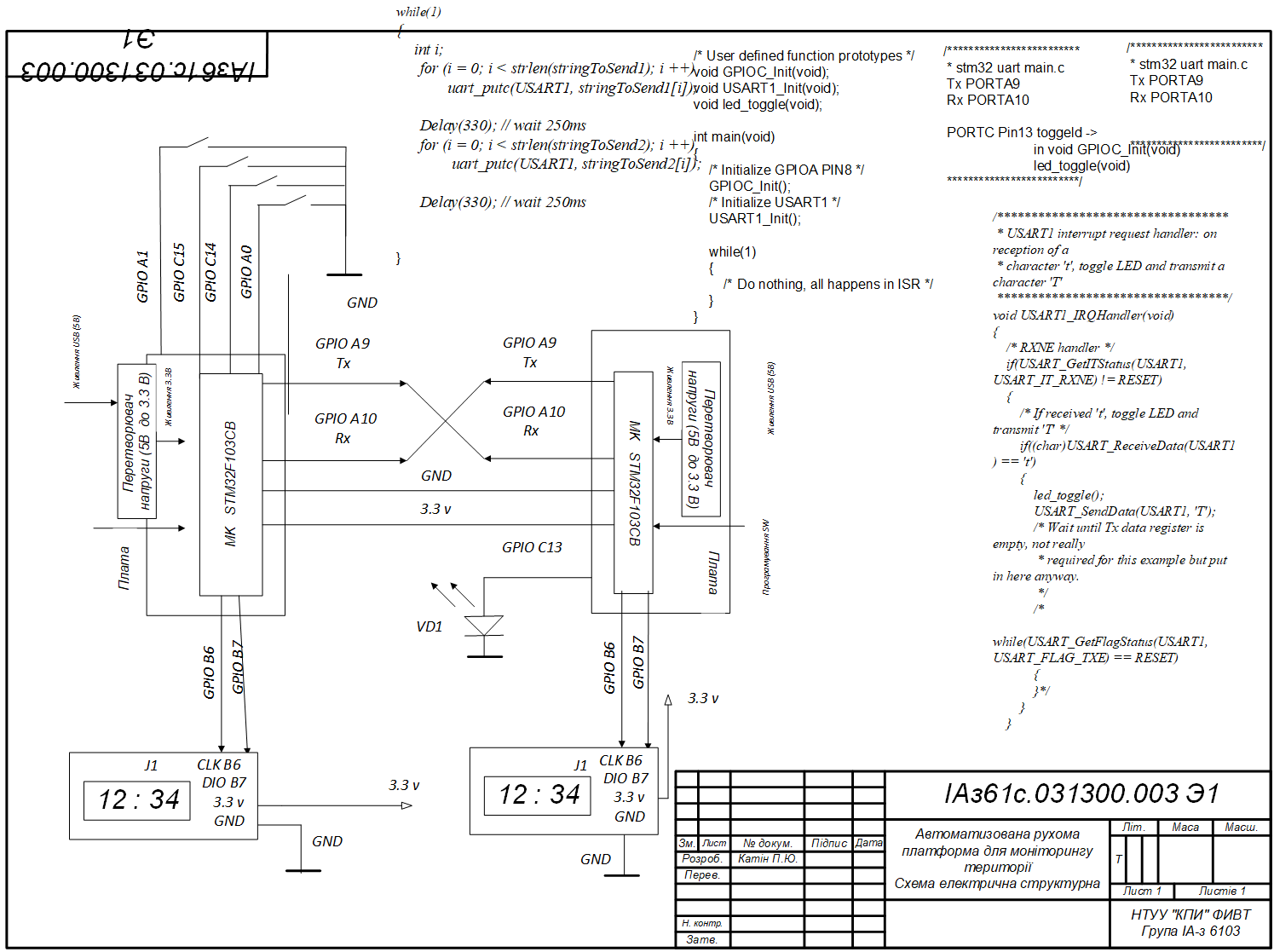
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис, дата)

Київ-2018

**Мета:** Розробити структурну схему апаратної частини і програмну частину і дослідити процес роботи системи передачі даних на базі мікроконтролера з використанням послідовного інтерфейсу. Вдосконалити програму, користуючись шаблоном кінцевого автомату.

**Хід роботи:**

Структурна схема включає дві плати налагодження для stm32.



1. Опис вихідного коду для приймача. У приймачі ініціалізуємо USART за допомогою методу USART1\_Init(). В ньому ж вмикаємо таймер та налаштовуємо сам USART1, використовуючи структуру.

void USART1\_Init(void)

{

   /\* USART configuration structure for USART1 \*/

   USART\_InitTypeDef USART1\_init\_struct;

   /\* Bit configuration structure for GPIOA PIN9 and PIN10 \*/

   GPIO\_InitTypeDef gpioa\_init\_struct;

   /\* Enalbe clock for USART1, AFIO and GPIOA \*/

   RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_USART1 | RCC\_APB2Periph\_AFIO |

                          RCC\_APB2Periph\_GPIOA, ENABLE);

   /\* GPIOA PIN9 alternative function Tx \*/

   gpioa\_init\_struct.GPIO\_Pin = GPIO\_Pin\_9;

   gpioa\_init\_struct.GPIO\_Speed = GPIO\_Speed\_50MHz;

   gpioa\_init\_struct.GPIO\_Mode = GPIO\_Mode\_AF\_PP;

   GPIO\_Init(GPIOA, &gpioa\_init\_struct);

   /\* GPIOA PIN10 alternative function Rx \*/

   gpioa\_init\_struct.GPIO\_Pin = GPIO\_Pin\_10;

   gpioa\_init\_struct.GPIO\_Speed = GPIO\_Speed\_50MHz;

   gpioa\_init\_struct.GPIO\_Mode = GPIO\_Mode\_IN\_FLOATING;

   GPIO\_Init(GPIOA, &gpioa\_init\_struct);

   /\* Enable USART1 \*/

   USART\_Cmd(USART1, ENABLE);

   /\* Baud rate 9600, 8-bit data, One stop bit

    \* No parity, Do both Rx and Tx, No HW flow control

    \*/

   USART1\_init\_struct.USART\_BaudRate = 9600;

   USART1\_init\_struct.USART\_WordLength = USART\_WordLength\_8b;

   USART1\_init\_struct.USART\_StopBits = USART\_StopBits\_1;

   USART1\_init\_struct.USART\_Parity = USART\_Parity\_No ;

   USART1\_init\_struct.USART\_Mode = USART\_Mode\_Rx | USART\_Mode\_Tx;

   USART1\_init\_struct.USART\_HardwareFlowControl = USART\_HardwareFlowControl\_None;

   /\* Configure USART1 \*/

   USART\_Init(USART1, &USART1\_init\_struct);

   /\* Enable RXNE interrupt \*/

   USART\_ITConfig(USART1, USART\_IT\_RXNE, ENABLE);

   /\* Enable USART1 global interrupt \*/

   NVIC\_EnableIRQ(USART1\_IRQn);

}

1. Метод main() приймача для отримання та відображення прийнятих даних, у якому відповідно до значення змінної tem\_res відображуються певні сегменти на дисплеї.

Int main(void)  
{  
    sys\_tick\_ini();  
    TM1637\_init();   
    TM1637\_brightness(BRIGHT\_DARKEST);   
    TM1637\_clearDisplay();  
    /\* Initialize GPIOA PIN8 \*/  
   // GPIOC\_Init();  
    /\* Initialize USART1 \*/  
    USART1\_Init();  
 tem\_res='4';  
 char old\_tem\_res=tem\_res;  
 uint8\_t a=0x75;  
  
    while(1)  
    {  
 if(old\_tem\_res!=tem\_res){  
        switch (tem\_res) {  
 case 0:  
 Display\_Loop\_1\_effect(5);  
 break;  
 case 1:  
 Display\_Loop\_4\_effect(5);  
 break;  
 case 2:  
 Display\_Loop\_2\_effect(5);  
 break;  
 case 3:  
 Display\_Loop\_dash\_effect\_1(5);  
 break;  
 default:  
 //1  
 break;  
 }  
 old\_tem\_res = tem\_res;  
 }else{  
        TM1637\_display\_all('T');  
 }  
    }  
}

1. Для передавача створюємо метод ініціалізації uart\_open(), де підключаємо альтернативну функції AF\_PushPull, IN\_FLOATING до послідовного порту USART1 на шині А зі встановленим системним таймером. Виконуємо налаштування USART1 через структуру.

int uart\_open (USART\_TypeDef \* USARTx, uint32\_t baud, uint32\_t flags)

{

// This function assumes GPIOA is used for USART

GPIO\_InitTypeDef GPIO\_InitStructure;

USART\_InitTypeDef USART\_InitStructure;

GPIO\_StructInit (& GPIO\_InitStructure);

// Enable GPIOA Clock

RCC\_APB2PeriphClockCmd (RCC\_APB2Periph\_GPIOA, ENABLE);

if (USARTx == USART1)

{

// Enable USART1 Clock

/ \* Enalbe clock for USART1, AFIO and GPIOA \* /

   RCC\_APB2PeriphClockCmd (RCC\_APB2Periph\_USART1 | RCC\_APB2Periph\_AFIO |

                          RCC\_APB2Periph\_GPIOA, ENABLE);

/ \* GPIOA PIN9 alternative function Tx \* /

   GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_9;

   GPIO\_InitStructure.GPIO\_Speed ​​= GPIO\_Speed\_50MHz;

   GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AF\_PP;

   GPIO\_Init (GPIOA, & GPIO\_InitStructure);

   / \* GPIOA PIN10 alternative function Rx \* /

  GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_10;

   GPIO\_InitStructure.GPIO\_Speed ​​= GPIO\_Speed\_50MHz;

   GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_IN\_FLOATING;

   GPIO\_Init (GPIOA, & GPIO\_InitStructure);

};

// Initialise USART structure

USART\_StructInit (& USART\_InitStructure);

// Modify USART\_InitStructure for non-default values, e.G.

USART\_InitStructure.USART\_BaudRate = Baud;

USART\_InitStructure.USART\_Mode = flags;

USART\_Init (USARTx, & USART\_InitStructure);

USART\_Cmd (USARTx, ENABLE);

return 0;

}

1. Користування послідовним портом проходить через методи отримання та відправки даних.

char uart\_getc (USART\_TypeDef \* USARTx)

{

while (USART\_GetFlagStatus (USARTx, USART\_FLAG\_RXNE) == RESET);

return USART\_ReceiveData (USARTx);

}

int uart\_putc (USART\_TypeDef \* USARTx, char c)

{

while (USART\_GetFlagStatus (USARTx, USART\_FLAG\_TXE) == RESET);

USART\_SendData (USARTx, c);

return 0;}

1. Метод Main для початку роботи передавача та відправки даних на інший пристрій, а також отримання відповіді від нього.

int main (void)

{

    sys\_tick\_ini ();

    btn\_init\_in\_c15 ();

    btn\_init\_RED ();

    TM1637\_init ();

    TM1637\_brightness (BRIGHT\_DARKEST);

    // uint16\_t j = 55;

    TM1637\_display\_all (55);

    delay\_from\_Brown (1000);

    TM1637\_display\_all (77);

    delay\_from\_Brown (1000);

    TM1637\_clearDisplay ();

uart\_open (USART1, 9600, USART\_Mode\_Rx | USART\_Mode\_Tx);

while (1)

{

TM1637\_display\_all (1234);

buttonHandler ();

// receive answer

char answer = uart\_getc (USART1);

// show answer

TM1637\_display\_all ((uint16\_t)

}}

Висновки: Під час виконання лабораторної роботи ми поглибили знання в роботі Serial port типового мікроконтролера на базі STM32 і розібрали можливість зв’язку двох пристроїв та передачі даних один одному для їх відображення на LED дисплеях.