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 CMSIS 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.

_____

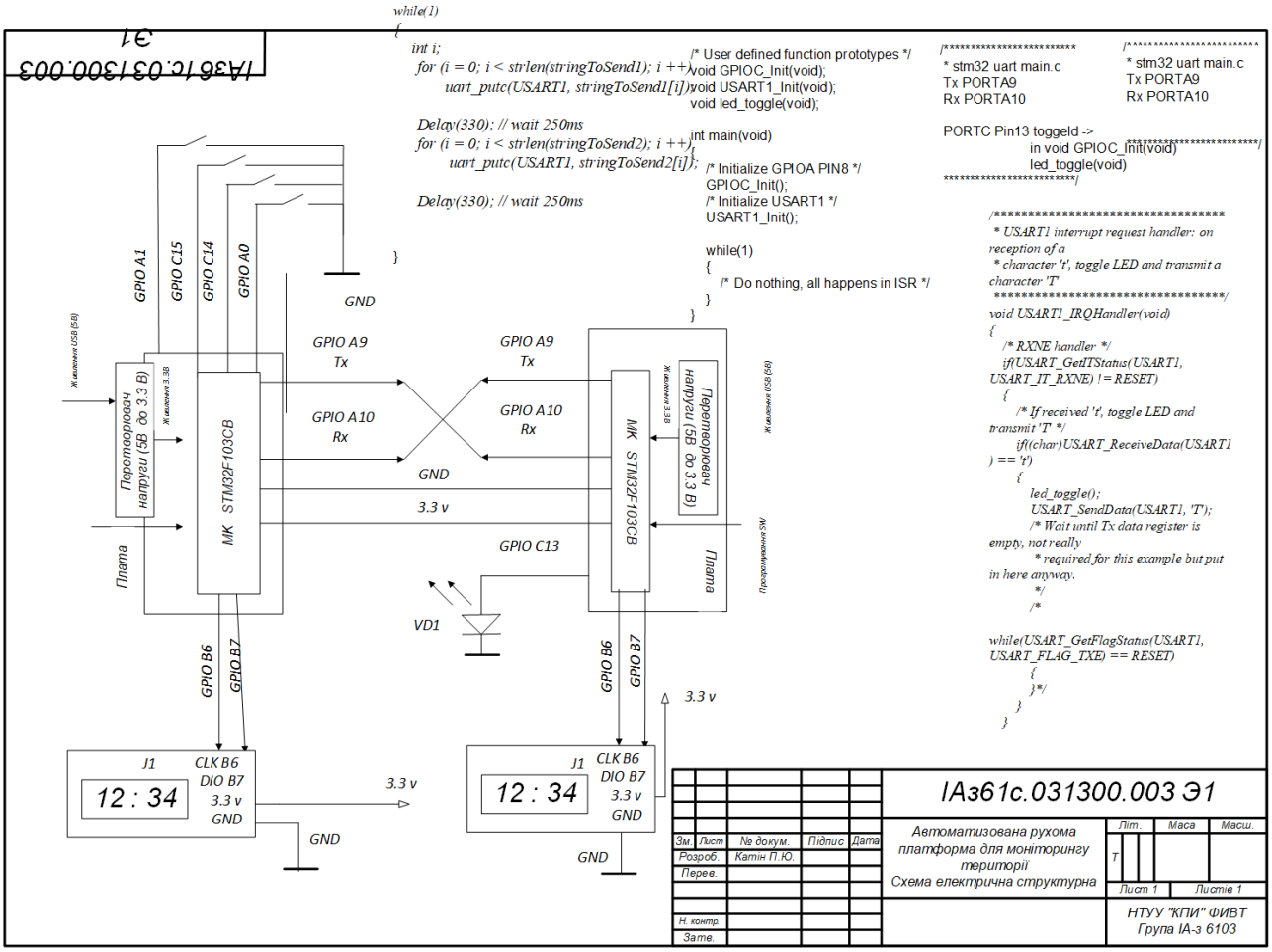
(підпис, дата)

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');
                            }
    }
}
```

2. Для передавача створюємо метод ініціалізації uart_open(), де підключаємо альтернативну функції AF_PushPull, IN_FLOATING до послідовного порту USART1 на шині A 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;
}
```

3. 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;}
```

4. 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.