Austin Petersen, Ayden Dauenhauer

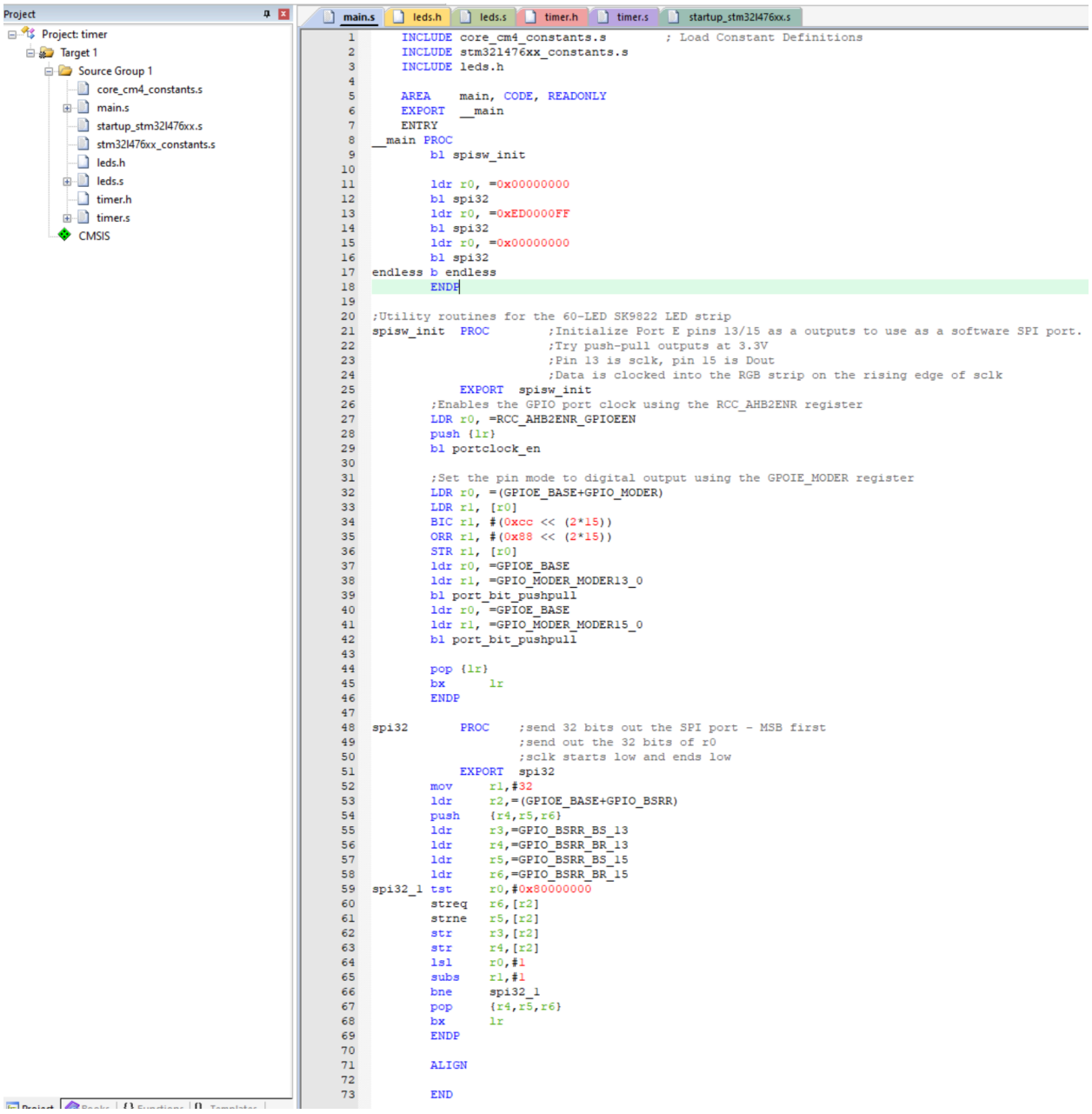
Prof. Wolfe

ELEN 50L Tuesday 2:15 p.m.

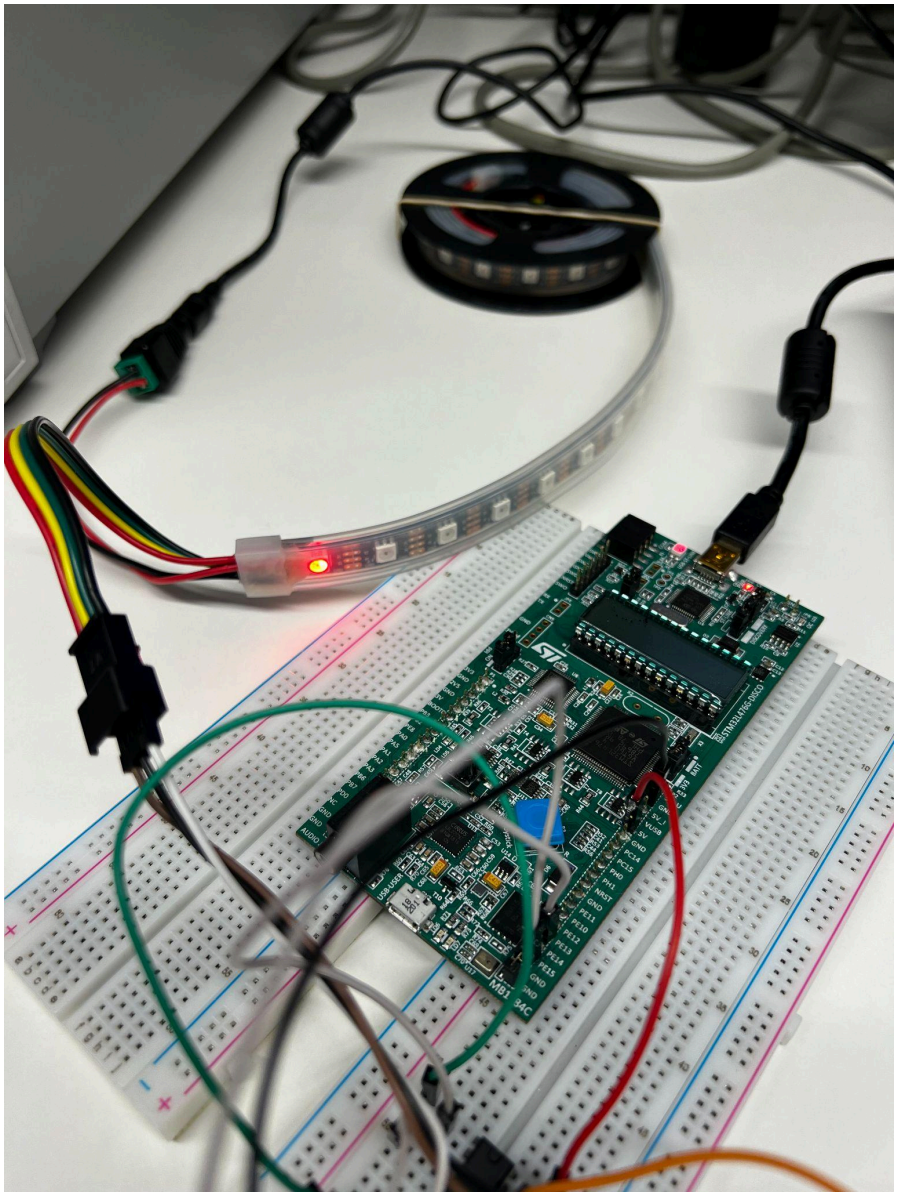
7 November 2023

Lab 6 - Serial Communications

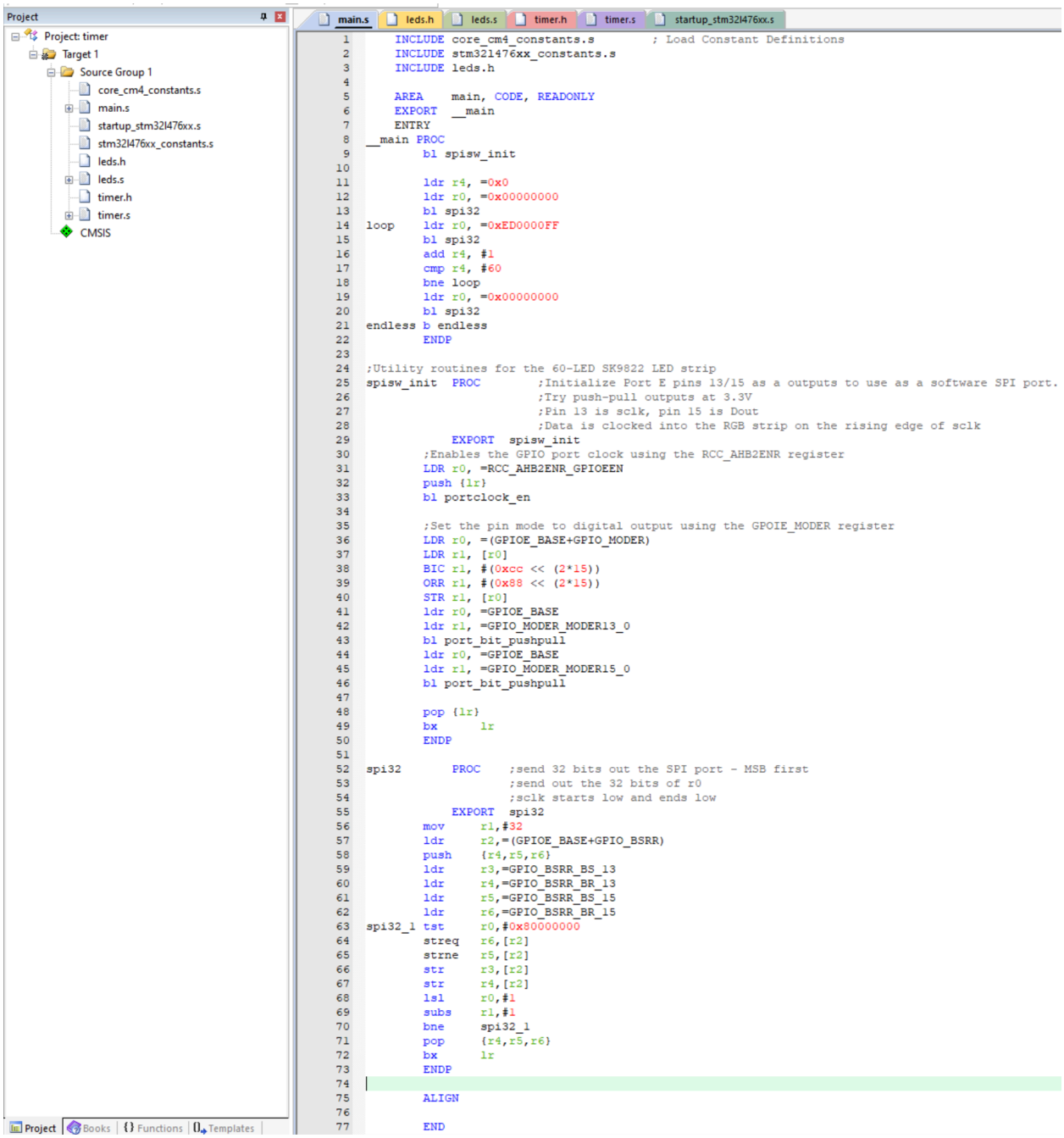
Problem 1:



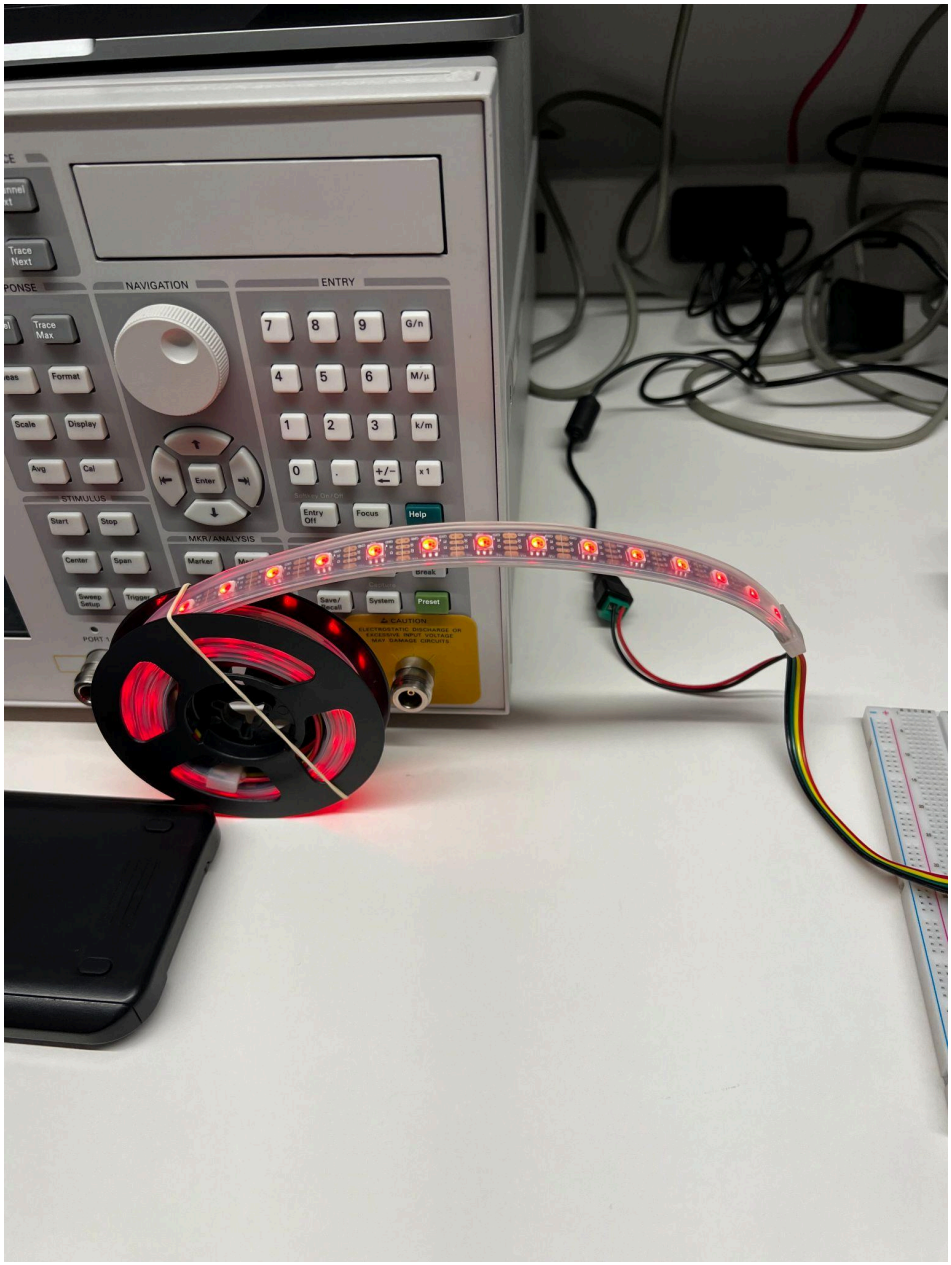

```
    INCLUDE core_cm4_constants.s        ; Load Constant Definitions
    INCLUDE stm32l476xx_constants.s
    INCLUDE leds.h

    AREA    main, CODE, READONLY
    EXPORT  __main
    ENTRY
__main PROC
        bl spisw_init

        ldr r0, =0x00000000
        bl spi32
        ldr r0, =0xED0000FF
        bl spi32
        ldr r0, =0x00000000
        bl spi32
endless b endless
        ENDP

;Utility routines for the 60-LED SK9822 LED strip
spisw_init  PROC        ;Initialize Port E pins 13/15 as a outputs to use as a software SPI port.
                        ;Try push-pull outputs at 3.3V
                        ;Pin 13 is sclk, pin 15 is Dout
                        ;Data is clocked into the RGB strip on the rising edge of sclk
            EXPORT  spisw_init
        ;Enables the GPIO port clock using the RCC_AHB2ENR register
        LDR r0, =RCC_AHB2ENR_GPIOEEN
        push {lr}
        bl portclock_en

        ;Set the pin mode to digital output using the GPOIE_MODER register
        LDR r0, =(GPIOE_BASE+GPIO_MODER)
        LDR r1, [r0]
        BIC r1, #(0xcc << (2*15))
        ORR r1, #(0x88 << (2*15))
        STR r1, [r0]
        ldr r0, =GPIOE_BASE
        ldr r1, =GPIO_MODER_MODER13_0
        bl port_bit_pushpull
        ldr r0, =GPIOE_BASE
        ldr r1, =GPIO_MODER_MODER15_0
        bl port_bit_pushpull

        pop {lr}
        bx      lr
        ENDP

spi32       PROC    ;send 32 bits out the SPI port - MSB first
                    ;send out the 32 bits of r0
                    ;sclk starts low and ends low
            EXPORT  spi32
        mov     r1,#32
        ldr     r2,=(GPIOE_BASE+GPIO_BSRR)
        push    {r4,r5,r6}
        ldr     r3,=GPIO_BSRR_BS_13
        ldr     r4,=GPIO_BSRR_BR_13
        ldr     r5,=GPIO_BSRR_BS_15
        ldr     r6,=GPIO_BSRR_BR_15
spi32_1 tst     r0,#0x80000000
        streq   r6,[r2]
        strne   r5,[r2]
        str     r3,[r2]
        str     r4,[r2]
        lsl     r0,#1
        subs    r1,#1
        bne     spi32_1
        pop     {r4,r5,r6}
        bx      lr
        ENDP

        ALIGN

        END
```



Problem 2:



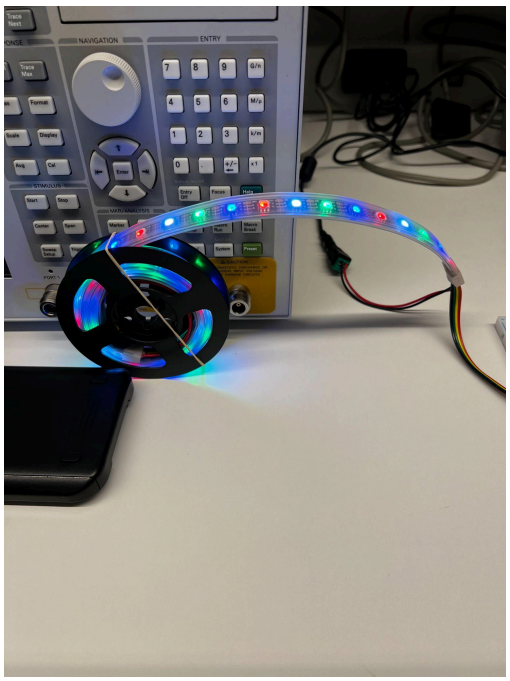

```
    INCLUDE core_cm4_constants.s        ; Load Constant Definitions
    INCLUDE stm32l476xx_constants.s
    INCLUDE leds.h

    AREA    main, CODE, READONLY
    EXPORT  __main
    ENTRY
__main PROC
        bl spisw_init

        ldr r4, =0x0
        ldr r0, =0x00000000
        bl spi32
loop    ldr r0, =0xED0000FF
        bl spi32
        add r4, #1
        cmp r4, #60
        bne loop
        ldr r0, =0x00000000
        bl spi32
endless b endless
        ENDP

;Utility routines for the 60-LED SK9822 LED strip
spisw_init  PROC        ;Initialize Port E pins 13/15 as a outputs to use as a software SPI port.
                        ;Try push-pull outputs at 3.3V
                        ;Pin 13 is sclk, pin 15 is Dout
                        ;Data is clocked into the RGB strip on the rising edge of sclk
            EXPORT  spisw_init
        ;Enables the GPIO port clock using the RCC_AHB2ENR register
        LDR r0, =RCC_AHB2ENR_GPIOEEN
        push {lr}
        bl portclock_en

        ;Set the pin mode to digital output using the GPOIE_MODER register
        LDR r0, =(GPIOE_BASE+GPIO_MODER)
        LDR r1, [r0]
        BIC r1, #(0xcc << (2*15))
        ORR r1, #(0x88 << (2*15))
        STR r1, [r0]
        ldr r0, =GPIOE_BASE
        ldr r1, =GPIO_MODER_MODER13_0
        bl port_bit_pushpull
        ldr r0, =GPIOE_BASE
        ldr r1, =GPIO_MODER_MODER15_0
        bl port_bit_pushpull

        pop {lr}
        bx      lr
        ENDP

spi32       PROC    ;send 32 bits out the SPI port - MSB first
                    ;send out the 32 bits of r0
                    ;sclk starts low and ends low
            EXPORT  spi32
        mov     r1,#32
        ldr     r2,=(GPIOE_BASE+GPIO_BSRR)
        push    {r4,r5,r6}
        ldr     r3,=GPIO_BSRR_BS_13
        ldr     r4,=GPIO_BSRR_BR_13
        ldr     r5,=GPIO_BSRR_BS_15
        ldr     r6,=GPIO_BSRR_BR_15
spi32_1 tst     r0,#0x80000000
        streq   r6,[r2]
        strne   r5,[r2]
        str     r3,[r2]
        str     r4,[r2]
        lsl     r0,#1
        subs    r1,#1
        bne     spi32_1
        pop     {r4,r5,r6}
        bx      lr
        ENDP

        ALIGN

        END
```


Trace Next
NAVIGATION
ENTRY
Trace Max
Format
Scale
Display
Avg
Cal
Enter
STIMULUS
Start
Stop
Center
Span
Sweep Setup
MKR/ANALYSIS
Marker
Save/ Recall
System
Preset
Entry Off
Focus
Help
7
8
9
G/n
4
5
6
M/µ
1
2
3
k/m
0
.
+/−
x1

Problem 3:

main.s | leds.h | leds.s | timer.h | timer.s | startup_stm32l476xx.s

```
1     INCLUDE core_cm4_constants.s        ; Load Constant Definitions
2     INCLUDE stm321476xx_constants.s
3     INCLUDE leds.h
4
5     AREA    main, CODE, READONLY
6     EXPORT  __main
7     ENTRY
8  __main PROC
9          bl spisw_init
10
11         ldr r4, =0x0
12         ldr r0, =0x00000000
13         bl spi32
14 loop    ldr r0, =0xE50000FF
15         bl spi32
16         ldr r0, =0xE5FF0000
17         bl spi32
18         ldr r0, =0xE500FF00
19         bl spi32
20         ldr r0, =0xE5FFFFFF
21         bl spi32
22         add r4, #1
23         cmp r4, #60
24         bne loop
25         ldr r0, =0x00000000
26         bl spi32
27 endless b endless
28         ENDP
```



Problem 4:

main.s | leds.h | leds.s | timer.h | timer.s | startup_stm32l476xx.s

```
    INCLUDE core_cm4_constants.s        ; Load Constant Definitions
    INCLUDE stm32l476xx_constants.s
    INCLUDE leds.h

    AREA    main, CODE, READONLY
    EXPORT  __main
    ENTRY
__main PROC
        bl spisw_init
        bl clean

        bl waiter
loopr   ldr r4, =0x0
        ldr r0, =0x00000000
        bl spi32
loop    ldr r0, =0xE50000FF
        bl spi32
        bl waiter
        ldr r0, =0xE500FF10
        bl spi32
        bl waiter
        ldr r0, =0xE5FFFFFF
        bl spi32
        bl waiter
        add r4, #1
        cmp r4, #60
        bne loop
        bl clean
        b loopr
        ldr r0, =0x00000000
        bl spi32
endless b endless
        ENDP

;Utility routines for the 60-LED SK9822 LED strip
spisw_init  PROC        ;Initialize Port E pins 13/15 as a outputs to use as a software SPI port.
                        ;Try push-pull outputs at 3.3V
                        ;Pin 13 is sclk, pin 15 is Dout
                        ;Data is clocked into the RGB strip on the rising edge of sclk
            EXPORT  spisw_init
        ;Enables the GPIO port clock using the RCC_AHB2ENR register
        LDR r0, =RCC_AHB2ENR_GPIOEEN
        push {lr}
        bl portclock_en

        ;Set the pin mode to digital output using the GPOIE_MODER register
        LDR r0, =(GPIOE_BASE+GPIO_MODER)
        LDR r1, [r0]
        BIC r1, #(0xcc << (2*15))
        ORR r1, #(0x88 << (2*15))
        STR r1, [r0]
        ldr r0, =GPIOE_BASE
        ldr r1, =GPIO_MODER_MODER13_0
        bl port_bit_pushpull
        ldr r0, =GPIOE_BASE
        ldr r1, =GPIO_MODER_MODER15_0
        bl port_bit_pushpull

        pop {lr}
        bx      lr
        ENDP
```

```
 62
 63  spi32       PROC    ;send 32 bits out the SPI port - MSB first
 64                      ;send out the 32 bits of r0
 65                      ;sclk starts low and ends low
 66              EXPORT  spi32
 67          mov     r1,#32
 68          ldr     r2,=(GPIOE_BASE+GPIO_BSRR)
 69          push    {r4,r5,r6}
 70          ldr     r3,=GPIO_BSRR_BS_13
 71          ldr     r4,=GPIO_BSRR_BR_13
 72          ldr     r5,=GPIO_BSRR_BS_15
 73          ldr     r6,=GPIO_BSRR_BR_15
 74  spi32_1 tst     r0,#0x80000000
 75          streq   r6,[r2]
 76          strne   r5,[r2]
 77          str     r3,[r2]
 78          str     r4,[r2]
 79          lsl     r0,#1
 80          subs    r1,#1
 81          bne     spi32_1
 82          pop     {r4,r5,r6}
 83          bx      lr
 84          ENDP
 85
 86
 87  waiter  PROC
 88
 89          EXPORT waiter
 90          mov r8, #0
 91  loop5   add r8, #1
 92          cmp r8, #0x00010001
 93          bne loop5
 94          bx lr
 95          ENDP
 96
 97  clean   PROC
 98
 99          EXPORT clean
100          push {lr}
101          mov r8, #0
102          ldr r0, =0x00000000
103          bl spi32
104  loop6   ldr r0, =0xE0000000
105          bl spi32
106          add r8, #1
107          cmp r8, #60
108          bne loop6
109          ldr r0, =0x00000000
110          bl spi32
111          pop {lr}
112          bx lr
113          ENDP
114
115
116          ALIGN
117
118          END
```

https://drive.google.com/file/d/1OTDTV2FYf5TetFrfxuuKeiAUruQ3L1B2/view?usp=drive_link