

Austin Petersen, Ayden Dauenhauer

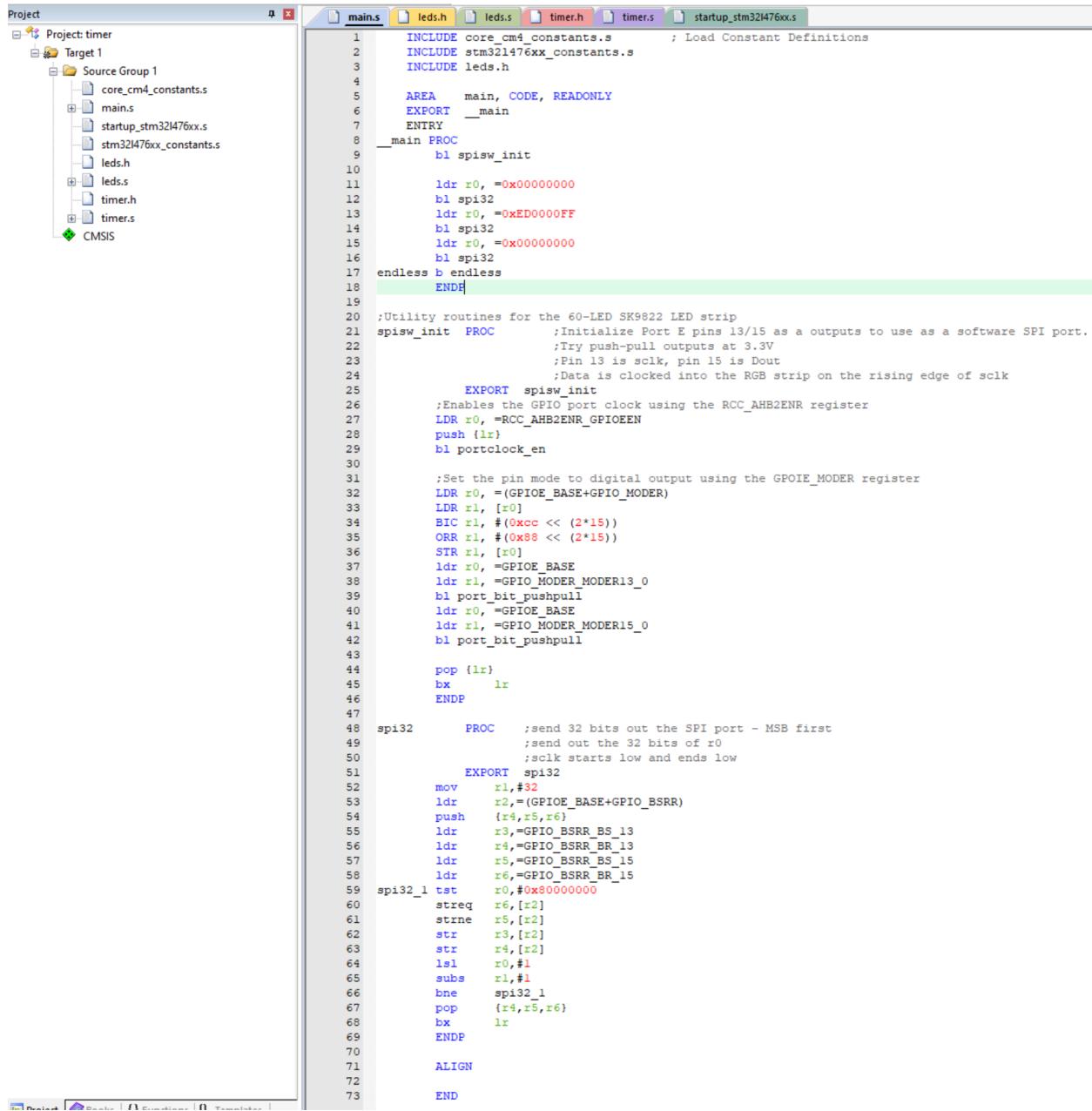
Prof. Wolfe

ELEN 50L Tuesday 2:15 p.m.

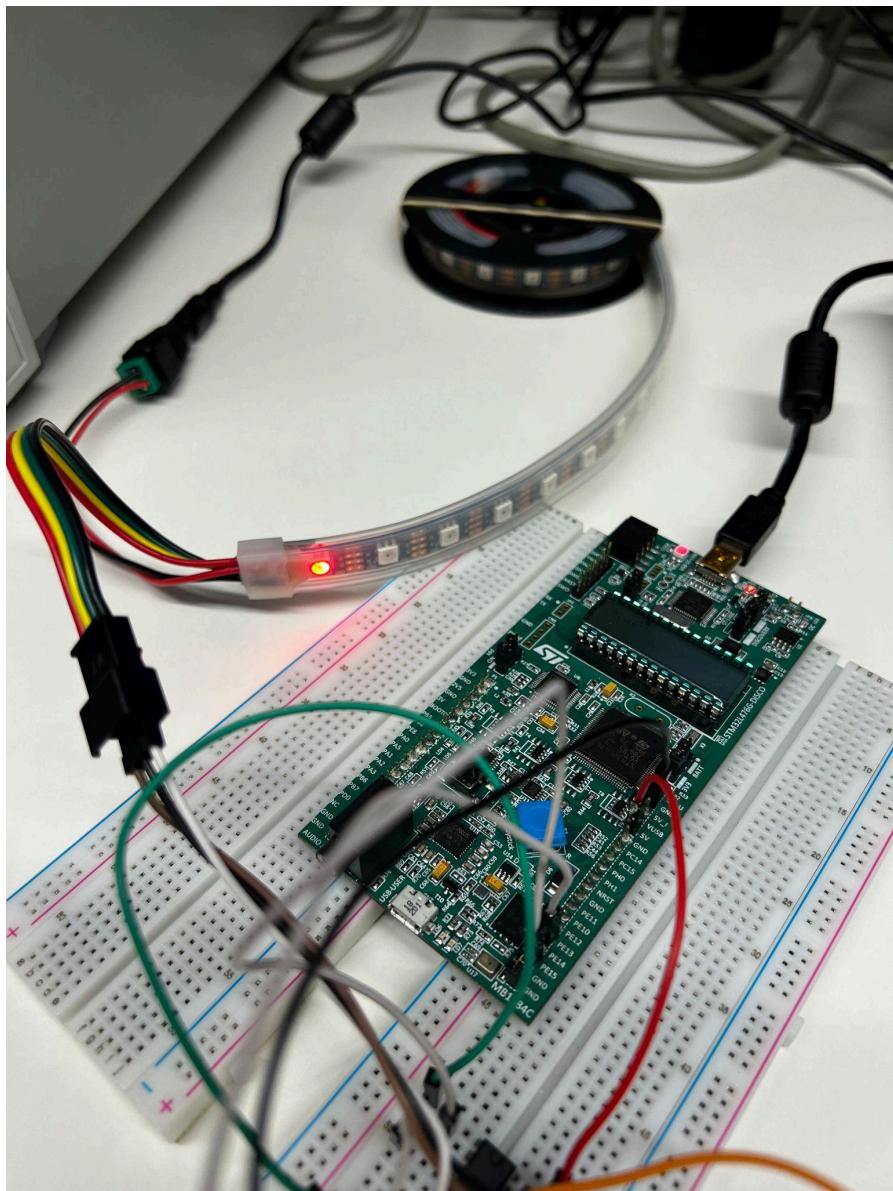
7 November 2023

Lab 6 - Serial Communications

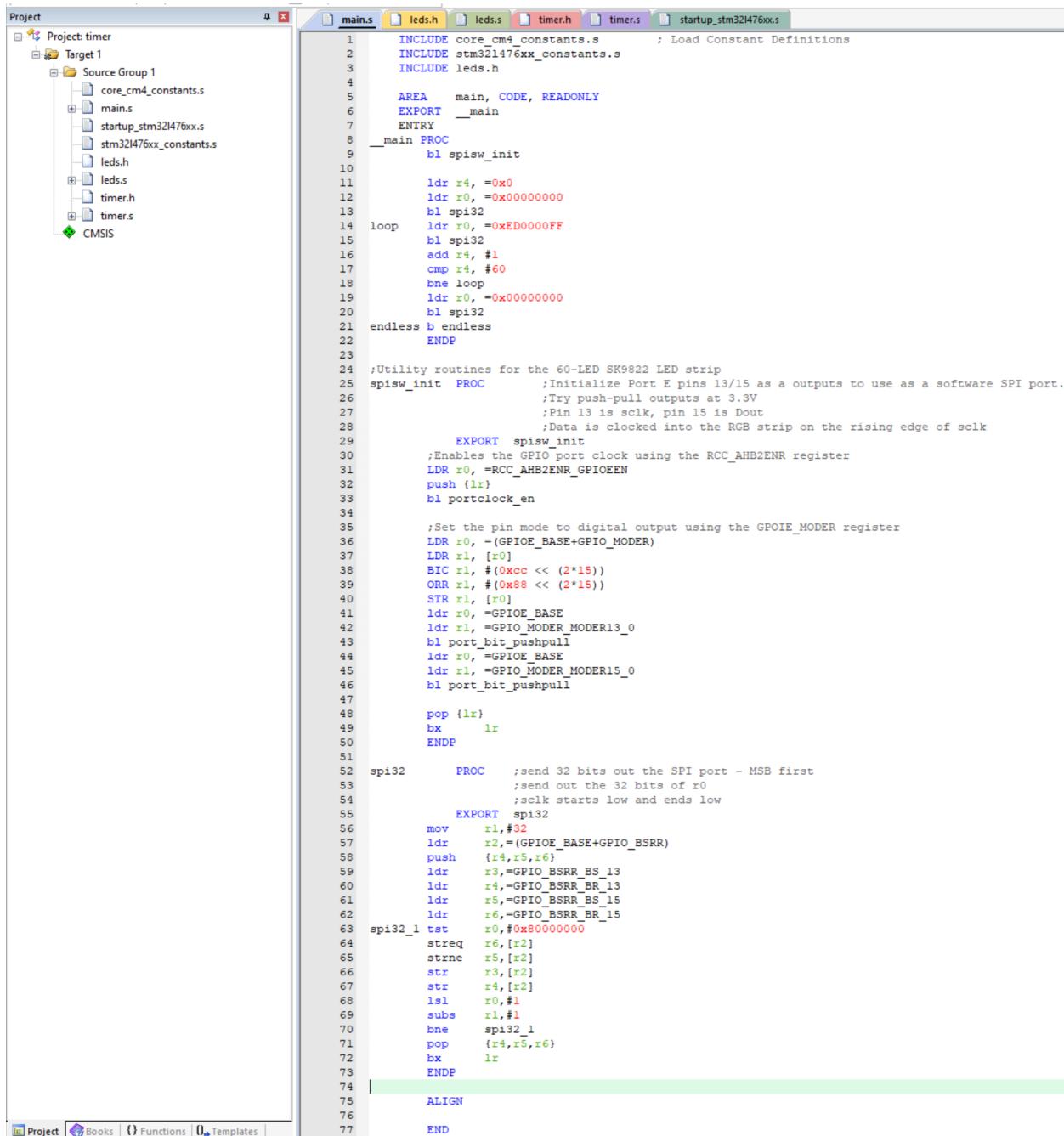
Problem 1:



```
1 INCLUDE core_cm4_constants.s           ; Load Constant Definitions
2 INCLUDE stm32l476xx_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    ldr r0, =0x00000000
11    bl spi32
12    ldr r0, =0xED0000FF
13    bl spi32
14    ldr r0, =0x00000000
15    bl spi32
16    b endless
17    ENDP
18
19
20 ;Utility routines for the 60-LED SK9822 LED strip
21 spisw_init PROC      ;Initialize Port E pins 13/15 as outputs to use as a software SPI port.
22 ;Try push-pull outputs at 3.3V
23 ;Pin 13 is sclk, pin 15 is Dout
24 ;Data is clocked into the RGB strip on the rising edge of sclk
25     EXPORT spisw_init
26     ;Enables the GPIO port clock using the RCC_AHB2ENR register
27     LDR r0, =RCC_AHB2ENR_GPIOEN
28     push {lr}
29     bl portclock_en
30
31     ;Set the pin mode to digital output using the GPIOE_MODER register
32     LDR r0, =(GPIOE_BASE+GPIO_MODER)
33     LDR r1, [r0]
34     BIC r1, #(0xcc <> (2*15))
35     ORR r1, #(0x88 <> (2*15))
36     STR r1, [r0]
37     LDR r0, =GPIOE_BASE
38     LDR r1, =GPIOE_MODER_MODER13_0
39     bl port_bit_pushpull
40     LDR r0, =GPIOE_BASE
41     LDR r1, =GPIOE_MODER_MODER15_0
42     bl port_bit_pushpull
43
44     pop {lr}
45     bx    lr
46     ENDP
47
48 spi32    PROC      ;send 32 bits out the SPI port - MSB first
49 ;send out the 32 bits of r0
50 ;sclk starts low and ends low
51     EXPORT spi32
52     mov   r1,#32
53     ldr   r2,=(GPIOE_BASE+GPIO_BSRR)
54     push  {r4,r5,r6}
55     ldr   r3,=GPIOE_BSRR_BS_13
56     ldr   r4,=GPIOE_BSRR_BR_13
57     ldr   r5,=GPIOE_BSRR_BS_15
58     ldr   r6,=GPIOE_BSRR_BR_15
59     spi32_l tst   r0,#0x80000000
60     streq r6,[r2]
61     strne r5,[r2]
62     str   r3,[r2]
63     str   r4,[r2]
64     lsl   r0,#1
65     subs  r1,#1
66     bne   spi32_l
67     pop   {r4,r5,r6}
68     bx    lr
69     ENDP
70
71     ALIGN
72
73 END
```

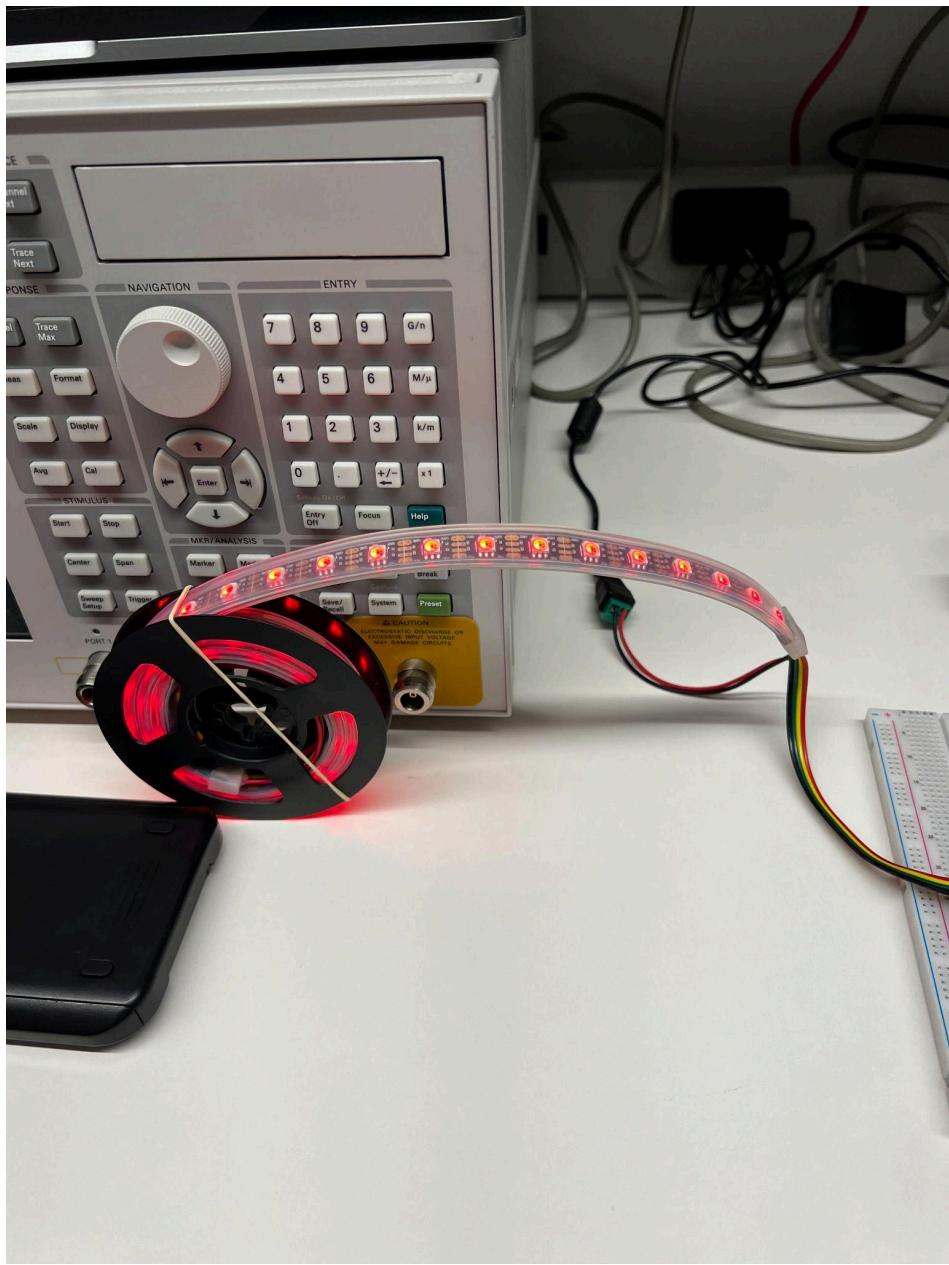


Problem 2:

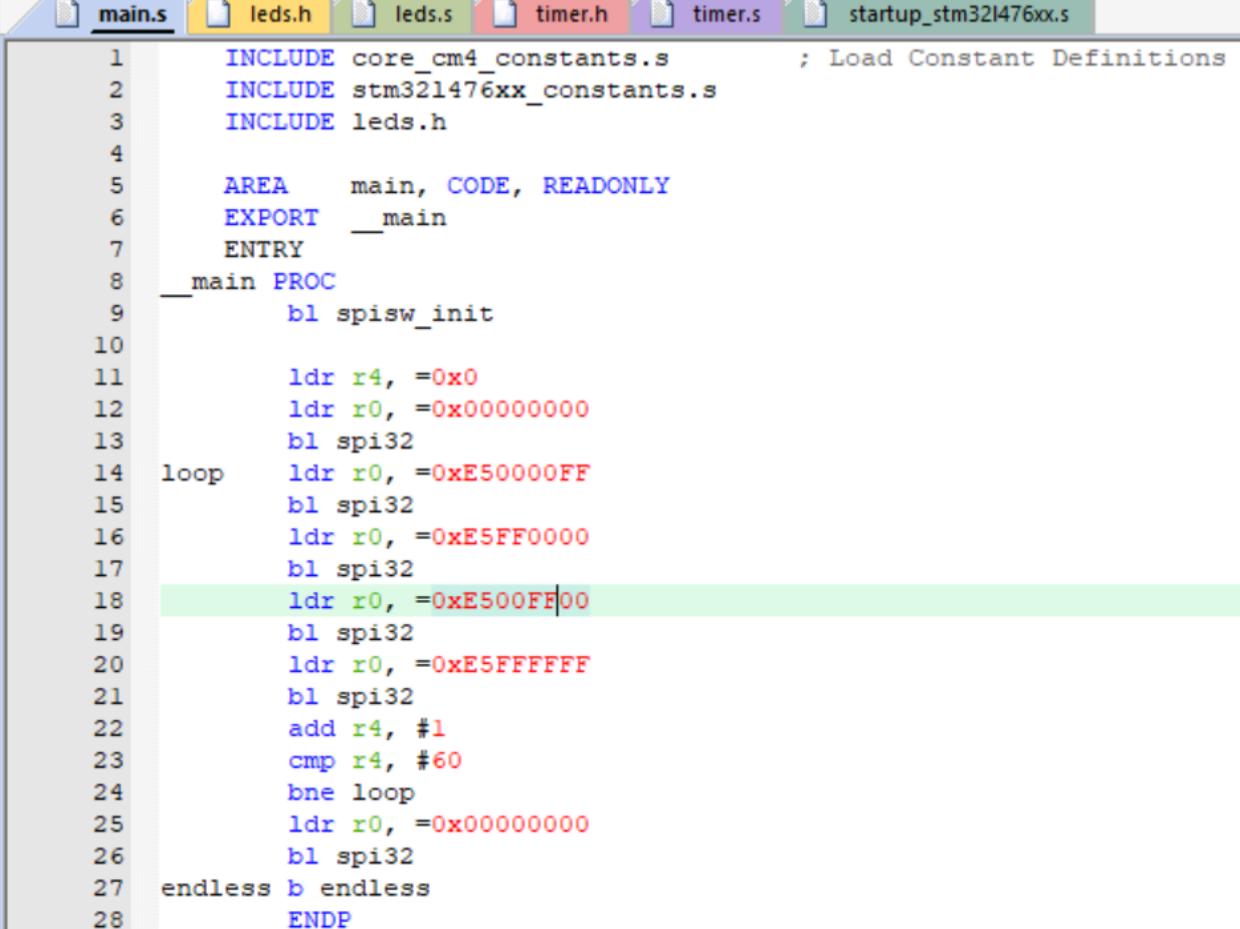


The screenshot shows the Keil MDK-ARM IDE interface with the main.s file open in the editor. The project tree on the left shows files like core_cm4_constants.s, main.s, startup_stm32l476xx.s, stm32l476xx_constants.s, leds.h, leds.s, timer.h, timers.s, and CMSIS. The main.s file contains assembly code for the STM32L476xx microcontroller. The code includes includes for core_cm4_constants.s and stm32l476xx_constants.s, defines for AREA main, EXPORT __main, ENTRY, and main PROC. It also includes a spisw_init routine and a spisw_init PROC. The spisw_init PROC initializes Port E pins 13/15 as outputs for a software SPI port, sets pin 13 as sclk and pin 15 as Dout, and configures GPIOE_MODER registers. The code ends with an ENDP directive.

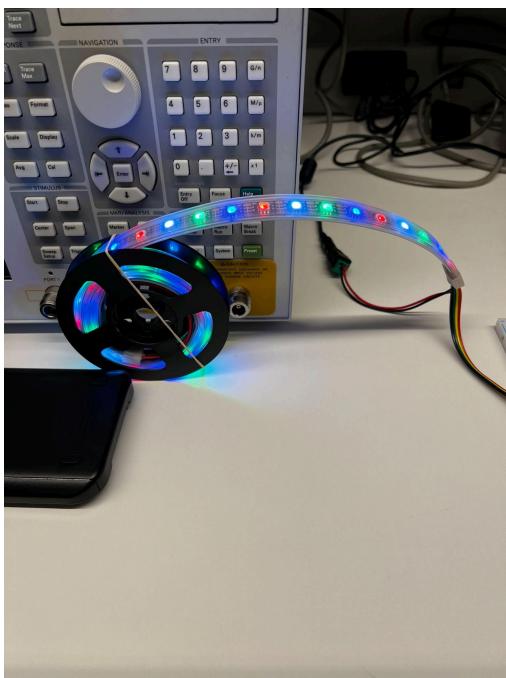
```
1 INCLUDE core_cm4_constants.s ; Load Constant Definitions
2 INCLUDE stm32l476xx_constants.s
3 INCLUDE leds.h
4
5 AREA main, CODE, READONLY
6 EXPORT __main
7 ENTRY
8 main PROC
9     bl spisw_init
10
11    ldr r4, =0x0
12    ldr r0, =0x00000000
13    bpi32
14 loop    ldr r0, =0xED0000FF
15    bpi32
16    add r4, #1
17    cmp r4, #60
18    bne loop
19    ldr r0, =0x00000000
20    bpi32
21    endless b endless
22    ENDP
23
24 ;Utility routines for the 60-LED SK9822 LED strip
25 spisw_init PROC ;Initialize Port E pins 13/15 as a outputs to use as a software SPI port.
26 ;Try push-pull outputs at 3.3V
27 ;Pin 13 is sclk, pin 15 is Dout
28 ;Data is clocked on the RGB strip on the rising edge of sclk
29     EXPORT spisw_init
30 ;Enables the GPIO port clock using the RCC_AHB2ENR register
31     LDR r0, =RCC_AHB2ENR_GPIOEEN
32     push {lr}
33     bpi portclock_en
34
35 ;Set the pin mode to digital output using the GPOE_MODER register
36     LDR r0, =(GPIOE_BASE+GPIO_MODER)
37     LDR r1, [r0]
38     BIC r1, #(0xcc << (2*15))
39     ORR r1, #(0x88 << (2*15))
40     STR r1, [r0]
41     LDR r0, =GPIOE_BASE
42     LDR r1, =GPIOE_MODER_MODER13_0
43     bpi port_bit_pushpull
44     LDR r0, =GPIOE_BASE
45     LDR r1, =GPIOE_MODER_MODER15_0
46     bpi port_bit_pushpull
47
48     pop {lr}
49     bx lr
50     ENDP
51
52 spis2 PROC ;send 32 bits out the SPI port - MSB first
53 ;send out the 32 bits of r0
54 ;sclk starts low and ends low
55     EXPORT spis2
56     mov r1,#32
57     ldr r2,=(GPIOE_BASE+GPIO_BSRR)
58     push {r4,r5,r6}
59     ldr r3,=GPIO_BSRR_BS_13
60     ldr r4,=GPIO_BSRR_BS_13
61     ldr r5,=GPIO_BSRR_BS_15
62     ldr r6,=GPIO_BSRR_BR_15
63 spis2_l   tst r0,#0x80000000
64     streq r6,[r2]
65     strne r5,[r2]
66     str r3,[r2]
67     str r4,[r2]
68     lsl r0,#1
69     subs r1,#1
70     bne spis2_l
71     pop {r4,r5,r6}
72     bx lr
73     ENDP
74
75     ALIGN
76
77     END
```



Problem 3:



```
main.s
1 INCLUDE core_cm4_constants.s           ; Load Constant Definitions
2 INCLUDE stm32l476xx_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
1 INCLUDE core_cm4_constants.s      ; Load Constant Definitions
2 INCLUDE stm32l476xx_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    bl clean
11
12    bl waiter
13    ldr r4, =0x0
14    ldr r0, =0x00000000
15    bl spi32
16    loop    ldr r0, =0xE50000FF
17    bl spi32
18    bl waiter
19    ldr r0, =0xE500FF10
20    bl spi32
21    bl waiter
22    ldr r0, =0xE5FFFFFF
23    bl spi32
24    bl waiter
25    add r4, #1
26    cmp r4, #60
27    bne loop
28    bl clean
29    b loop
30    ldr r0, =0x00000000
31    bl spi32
32    endless b endless
33    ENDP
34
35 ;Utility routines for the 60-LED SK9822 LED strip
36 spisw_init PROC      ;Initialize Port E pins 13/15 as outputs to use as a software SPI port.
37             ;Try push-pull outputs at 3.3V
38             ;Pin 13 is sclk, pin 15 is Dout
39             ;Data is clocked into the RGB strip on the rising edge of sclk
40     EXPORT spisw_init
41     ;Enables the GPIO port clock using the RCC_AHB2ENR register
42     LDR r0, =RCC_AHB2ENR_GPIOEEN
43     push {lr}
44     bl portclock_en
45
46     ;Set the pin mode to digital output using the GPOIE_MODER register
47     LDR r0, =(GPIOE_BASE+GPIO_MODER)
48     LDR r1, [r0]
49     BIC r1, #(0xcc << (2*15))
50     ORR r1, #(0x88 << (2*15))
51     STR r1, [r0]
52     ldr r0, =GPIOE_BASE
53     ldr r1, =GPIO_MODER_MODER13_0
54     bl port_bit_pushpull
55     ldr r0, =GPIOE_BASE
56     ldr r1, =GPIO_MODER_MODER15_0
57     bl port_bit_pushpull
58
59     pop {lr}
60     bx lr
61     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/1OTDTV2FYf5TetFrfrxuuKeiAUruQ3L1B2/view?usp=drive_link