Austin Petersen, Ayden Dauenhauer

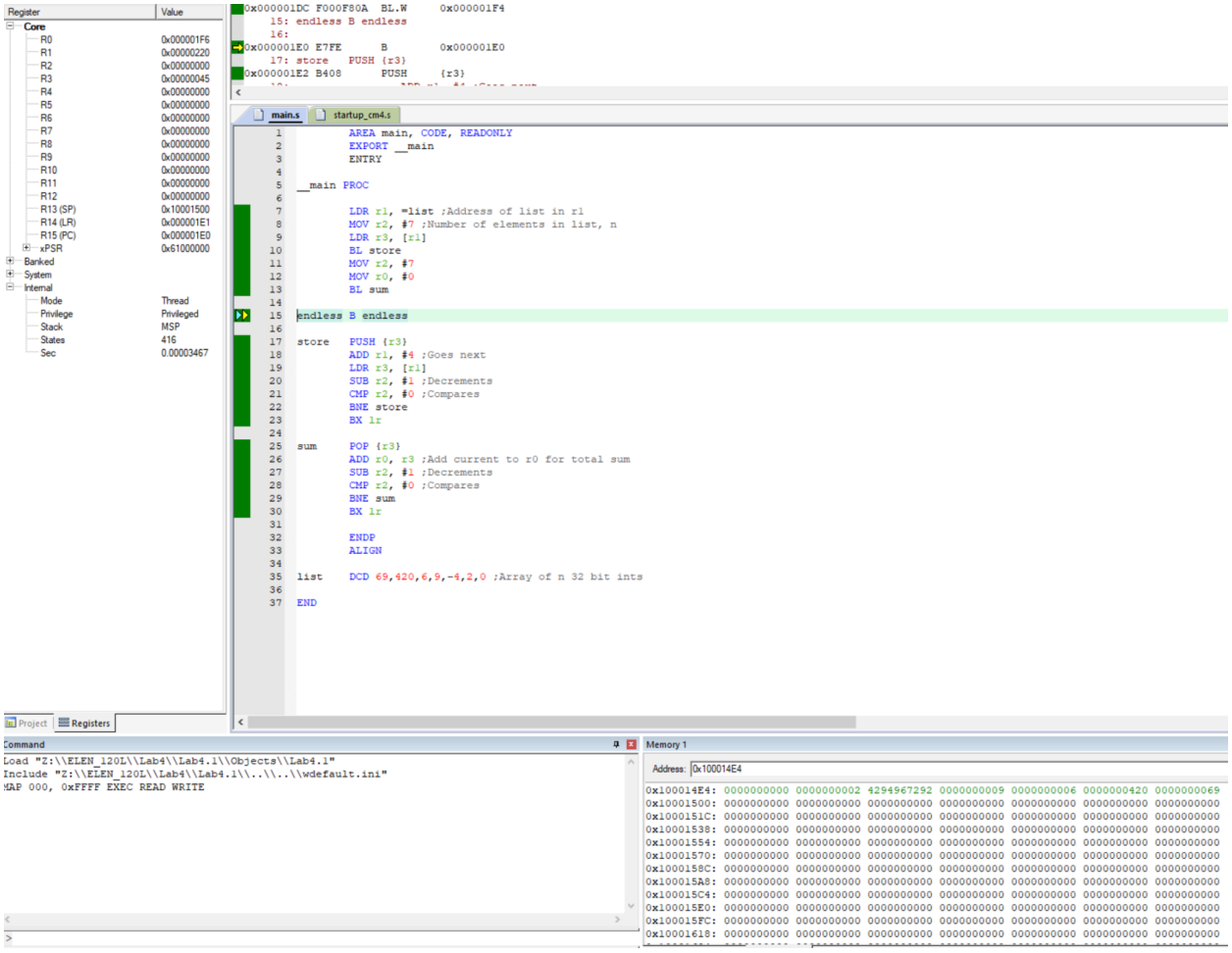
Prof. Wolfe

ELEN 120L Tuesday 2:15-5:00

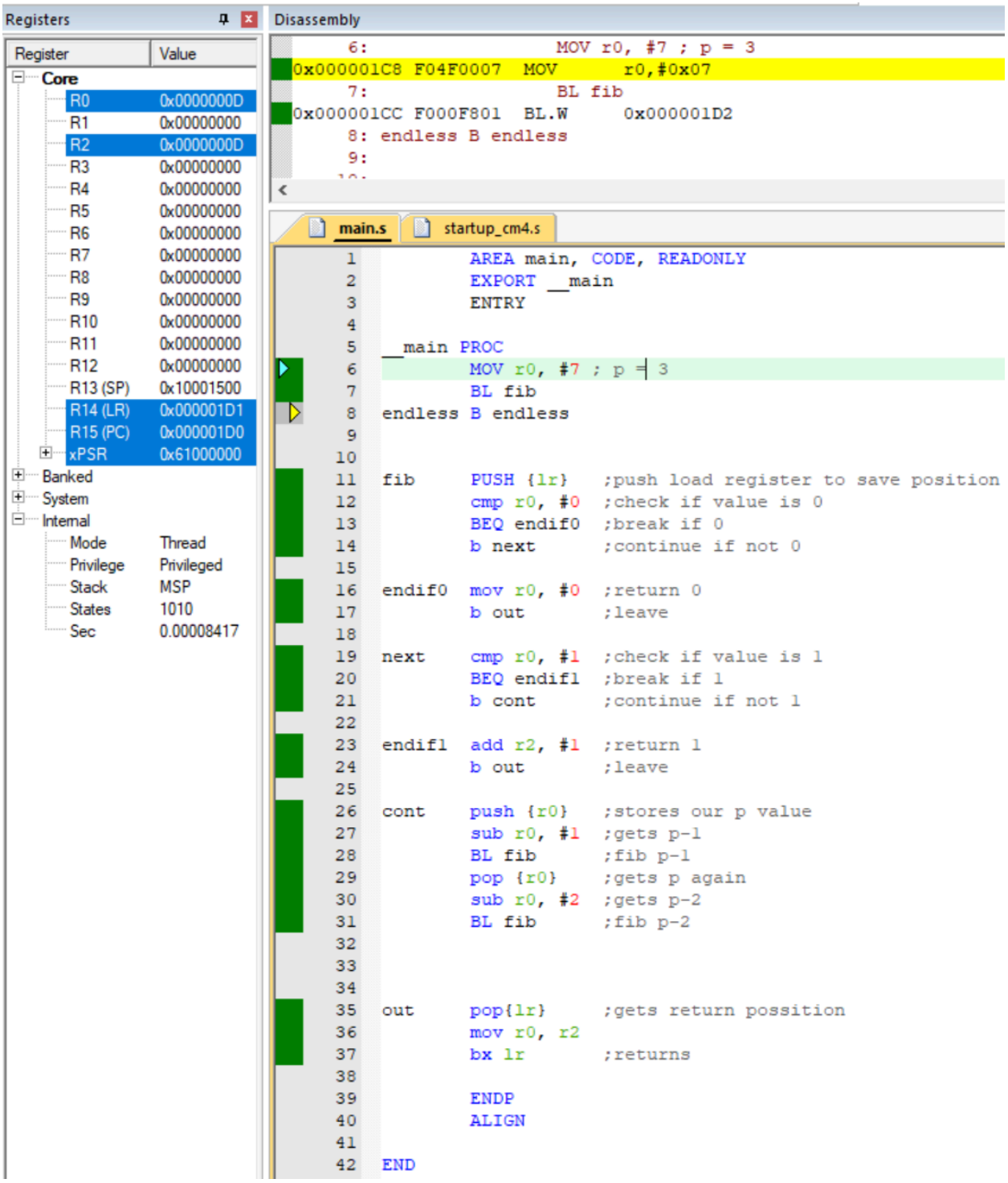
October 24, 2023

Report 4

Problem 1:



Problem 2:


Registers
Register
Value
Core
R0 0x0000000D
R1 0x00000000
R2 0x0000000D
R3 0x00000000
R4 0x00000000
R5 0x00000000
R6 0x00000000
R7 0x00000000
R8 0x00000000
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x10001500
R14 (LR) 0x000001D1
R15 (PC) 0x000001D0
xPSR 0x61000000
Banked
System
Internal
Mode Thread
Privilege Privileged
Stack MSP
States 1010
Sec 0.00008417
Disassembly
6: MOV r0, #7 ; p = 3
0x000001C8 F04F0007 MOV r0,#0x07
7: BL fib
0x000001CC F000F801 BL.W 0x000001D2
8: endless B endless
9:
main.s
startup_cm4.s
1 AREA main, CODE, READONLY
2 EXPORT __main
3 ENTRY
4
5 __main PROC
6 MOV r0, #7 ; p = 3
7 BL fib
8 endless B endless
9
10
11 fib PUSH {lr} ;push load register to save position
12 cmp r0, #0 ;check if value is 0
13 BEQ endif0 ;break if 0
14 b next ;continue if not 0
15
16 endif0 mov r0, #0 ;return 0
17 b out ;leave
18
19 next cmp r0, #1 ;check if value is 1
20 BEQ endif1 ;break if 1
21 b cont ;continue if not 1
22
23 endif1 add r2, #1 ;return 1
24 b out ;leave
25
26 cont push {r0} ;stores our p value
27 sub r0, #1 ;gets p-1
28 BL fib ;fib p-1
29 pop {r0} ;gets p again
30 sub r0, #2 ;gets p-2
31 BL fib ;fib p-2
32
33
34
35 out pop{lr} ;gets return possition
36 mov r0, r2
37 bx lr ;returns
38
39 ENDP
40 ALIGN
41
42 END

Problem 3:

main.s | startup_stm32l476xx.s

```
7               ENTRY
8
9   __main  PROC
10              bl porta
11              ldr r4, =list
12              ldr r5, =eoll
13  loop        bl read_jstick
14              bl decode_jstick
15              cmp r0, #0
16              movne r6, r0
17              b loop
18  endless     b endless
19
20
21          ;enables the clock in GPIO A
22  porta   LDR r0, =(RCC_BASE+RCC_AHB2ENR)
23          LDR r1, [r0]
24          ORR r1, #RCC_AHB2ENR_GPIOAEN
25          STR r1, [r0]
26
27          ldr r0, =(GPIOA_BASE+GPIO_MODER)
28          ldr r1, [r0]
29          mov r2, #0xCFF
30          bic r1, r1, r2
31          str r1, [r0]
32
33          ldr r0, =(GPIOA_BASE+GPIO_PUPDR)
34          ldr r1, [r0]
35          mov r2, #0xCFC
36          bic r1, r1, r2
37          mov r3, #2_100010101010
38          orr r1, r3
39          str r1, [r0]
40          bx lr
41
42  read_jstick ldr r0, =(GPIOA_BASE+GPIO_IDR)  ;loads IDR from gpio a in r0
43              ldr r1, [r0]                ;loads the value
44              mov r2, #2_101111
45              and r0, r1, r2  ; clears all but bits 0, 1, 2, 3, 5
46              bx lr                           ;returns
47
48
49  decode_jstick cmp r0, #2_000001
50                beq center
51                cmp r0, #2_000010
52                beq left
53                cmp r0, #2_000100
54                beq right
55                cmp r0, #2_001000
56                beq up
57                cmp r0, #2_100000
58                beq down
59                bx lr
60
61  center      mov r0, #'c'
62              b store
63
64  right       mov r0, #'r'
65              b store
66
67  left        mov r0, #'l'
68              b store
69
70  up          mov r0, #'u'
71              b store
72
73  down        mov r0, #'d'
74              b store
75
76  out         bx lr
77
78  store       cmp r0, r6
79              beq out ; moveq pc. lr
```

```
store       cmp r0, r6
            beq out ; moveq pc. lr
            str r0, [r4]
            add r4, #1
            cmp r4, r5
            beq endless
            bx lr
            |



        ENDP
            ALIGN
            AREA    myData, DATA, READWRITE
            ALIGN

list    dcb 0,0,0,0,0
eoll

    END
```

main.s* startup_stm32l476xx.s

```
25          STR r1, [r0]
26
27          ldr r0, =(GPIOA_BASE+GPIO_MODER)
28          ldr r1, [r0]
29          mov r2, #0xCFF
30          bic r1, r1, r2
31          str r1, [r0]
32
33          ldr r0, =(GPIOA_BASE+GPIO_PUPDR)
34          ldr r1, [r0]
35          mov r2, #0xCFC
36          bic r1, r1, r2
37          mov r3, #2_100010101010
38          orr r1, r3
39          str r1, [r0]
40          bx lr
41
42  read_jstick ldr r0, =(GPIOA_BASE+GPIO_IDR)  ;loads IDR from gpio a in r0
43              ldr r1, [r0]                ;loads the value
44              mov r2, #2_101111
45              and r0, r1, r2  ; clears all but bits 0, 1, 2, 3, 5
46              bx lr                           ;returns
47
48
49  decode_jstick cmp r0, #2_000001
50                beq center
51                cmp r0, #2_000010
52                beq left
53                cmp r0, #2_000100
54                beq right
55                cmp r0, #2_001000
56                beq up
57                cmp r0, #2_100000
58                beq down
59                bx lr
60
61  center      mov r0, #'c'
62              b store
63
64  right       mov r0, #'r'
65              b store
66
67  left        mov r0, #'l'
68              b store
69
70  up          mov r0, #'u'
71              b store
72
73  down        mov r0, #'d'
```

ects\\joystick.axf"

Memory 1

Address: 0x20000000

0x20000000: dlruc.....