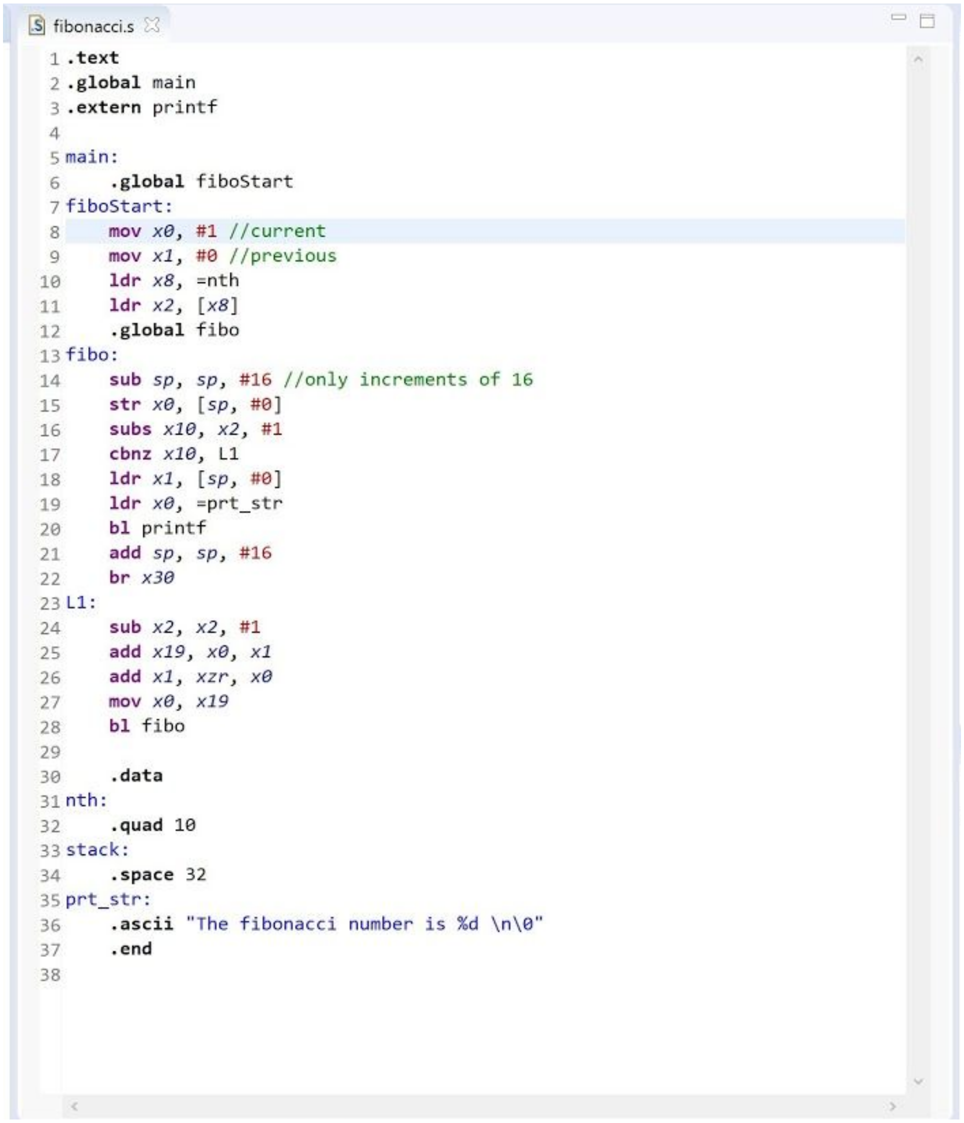
Semal Shah
I pledge my honor that I have abided by the Stevens Honor System.

Code for my fibonacci:



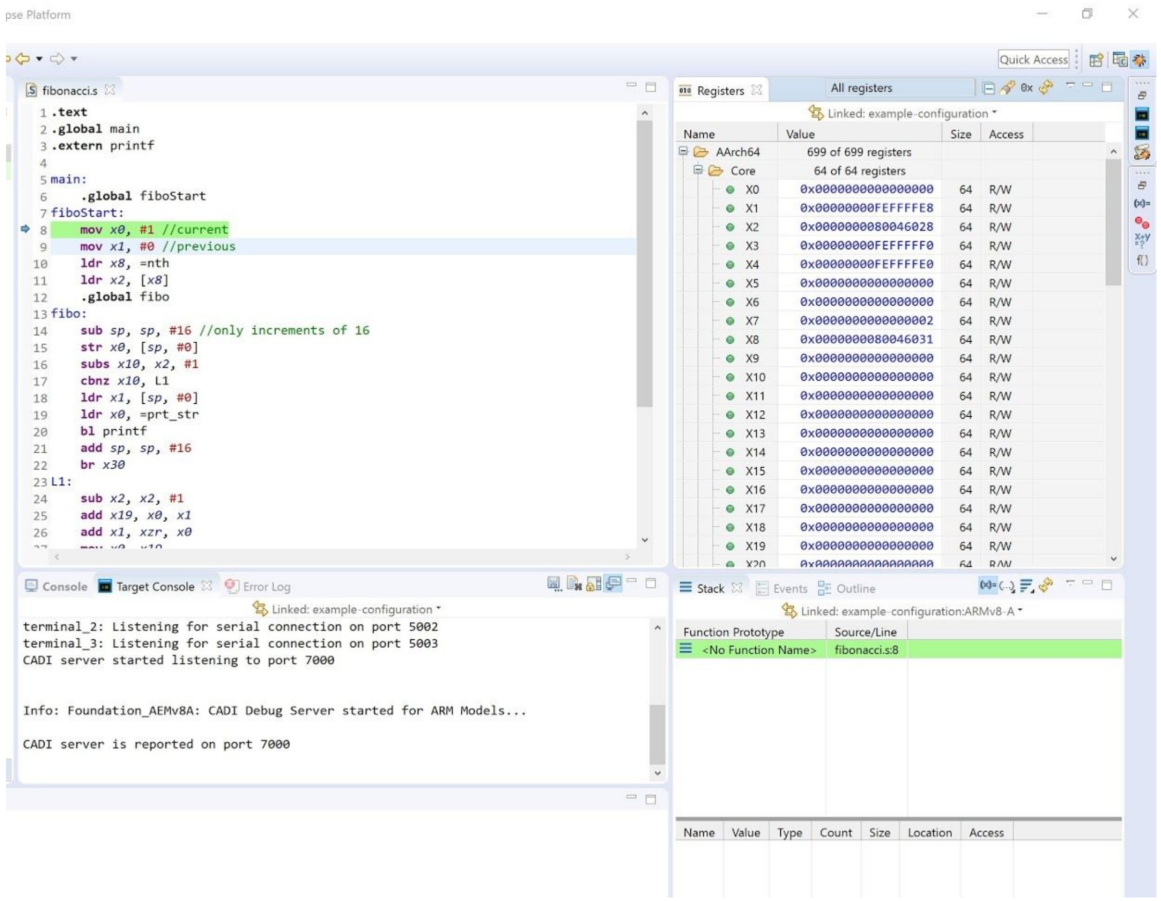
```
.text
.global main
.extern printf

main:
    .global fiboStart
fiboStart:
    mov x0, #1 //current
    mov x1, #0 //previous
    ldr x8, =nth
    ldr x2, [x8]
    .global fibo
fibo:
    sub sp, sp, #16 //only increments of 16
    str x0, [sp, #0]
    subs x10, x2, #1
    cbnz x10, L1
    ldr x1, [sp, #0]
    ldr x0, =prt_str
    bl printf
    add sp, sp, #16
    br x30
L1:
    sub x2, x2, #1
    add x19, x0, x1
    add x1, xzr, x0
    mov x0, x19
    bl fibo

    .data
nth:
    .quad 10
stack:
    .space 32
prt_str:
    .ascii "The fibonacci number is %d \n\0"
    .end
```

Explanation:
I started with the wrapper function shown in the powerpoint and called fibo from it. In fibo I made space on the stack first and then stored x0 on the stack. Then I checked whether x2 is equal to 1. If not then it just goes through the recursion of making current previous + current and making previous current. After the recursion is done (if it was needed in the first place), the program loads the last current value into x1 and then loads the address of prt_str into x0. The %d in the string will call on x1’s value; therefore, at the end, displaying “The fibonacci number is (the correct answer)”

Registers before change:



fibonacci.s

```
.text
.global main
.extern printf

main:
    .global fiboStart
fiboStart:
    mov x0, #1 //current
    mov x1, #0 //previous
    ldr x8, =nth
    ldr x2, [x8]
    .global fibo
fibo:
    sub sp, sp, #16 //only increments of 16
    str x0, [sp, #0]
    subs x10, x2, #1
    cbnz x10, L1
    ldr x1, [sp, #0]
    ldr x0, =prt_str
    bl printf
    add sp, sp, #16
    br x30
L1:
    sub x2, x2, #1
    add x19, x0, x1
    add x1, xzr, x0
```

Target Console

```
terminal_2: Listening for serial connection on port 5002
terminal_3: Listening for serial connection on port 5003
CADI server started listening to port 7000


Info: Foundation_AEMv8A: CADI Debug Server started for ARM Models...

CADI server is reported on port 7000
```

Registers — All registers — Linked: example-configuration

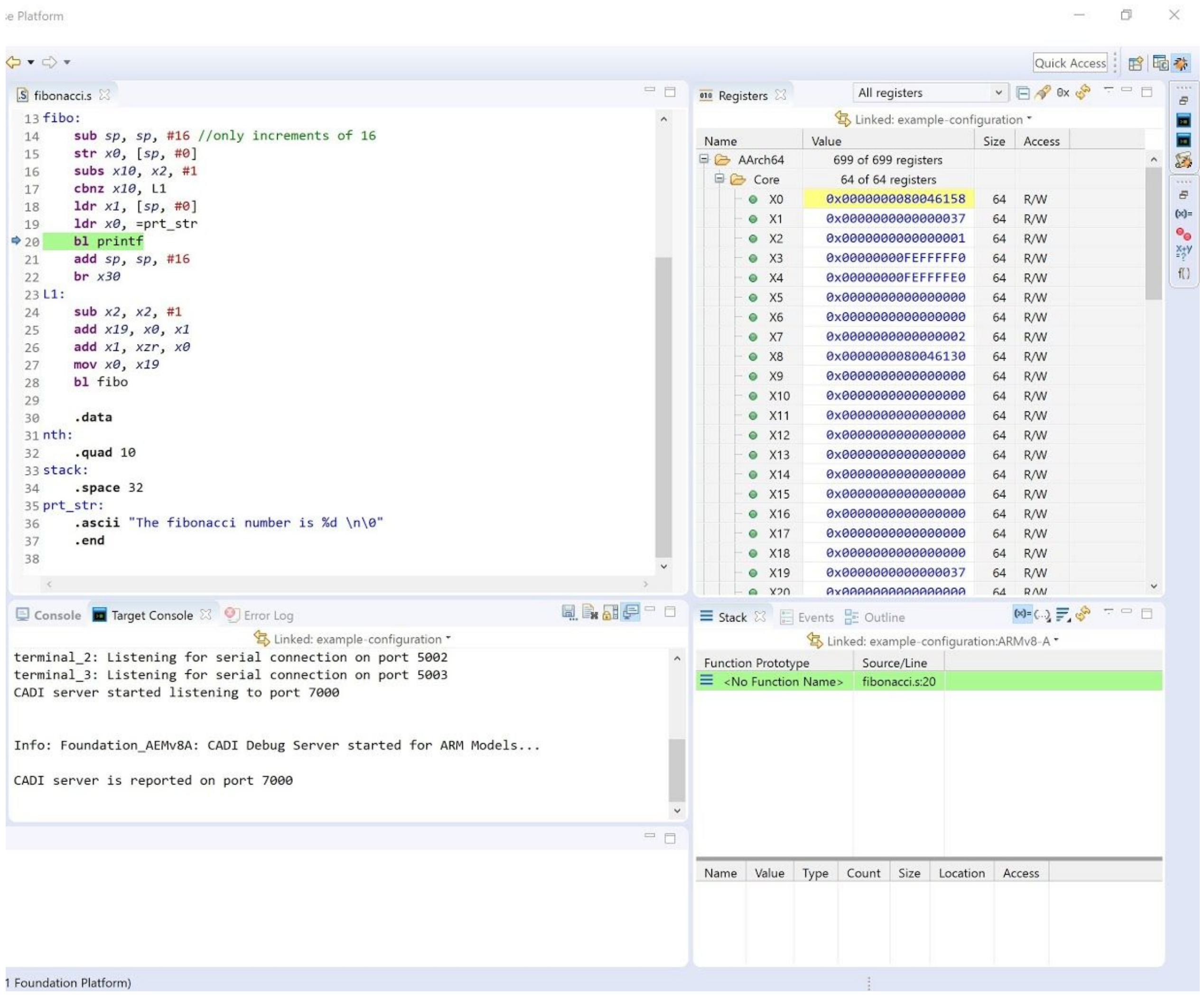
Name	Value	Size	Access
AArch64	699 of 699 registers		
Core	64 of 64 registers		
X0	0x0000000000000000	64	R/W
X1	0x00000000FEFFFFE8	64	R/W
X2	0x0000000080046028	64	R/W
X3	0x00000000FEFFFFF0	64	R/W
X4	0x00000000FEFFFFE0	64	R/W
X5	0x0000000000000000	64	R/W
X6	0x0000000000000000	64	R/W
X7	0x0000000000000002	64	R/W
X8	0x0000000080046031	64	R/W
X9	0x0000000000000000	64	R/W
X10	0x0000000000000000	64	R/W
X11	0x0000000000000000	64	R/W
X12	0x0000000000000000	64	R/W
X13	0x0000000000000000	64	R/W
X14	0x0000000000000000	64	R/W
X15	0x0000000000000000	64	R/W
X16	0x0000000000000000	64	R/W
X17	0x0000000000000000	64	R/W
X18	0x0000000000000000	64	R/W
X19	0x0000000000000000	64	R/W
X20	0x0000000000000000	64	R/W

Stack — Linked: example-configuration:ARMv8-A

Function Prototype	Source/Line
<No Function Name>	fibonacci.s:8

Name	Value	Type	Count	Size	Location	Access

Registers after change:



```
13 fibo:
14     sub sp, sp, #16 //only increments of 16
15     str x0, [sp, #0]
16     subs x10, x2, #1
17     cbnz x10, L1
18     ldr x1, [sp, #0]
19     ldr x0, =prt_str
20     bl printf
21     add sp, sp, #16
22     br x30
23 L1:
24     sub x2, x2, #1
25     add x19, x0, x1
26     add x1, xzr, x0
27     mov x0, x19
28     bl fibo
29
30     .data
31 nth:
32     .quad 10
33 stack:
34     .space 32
35 prt_str:
36     .ascii "The fibonacci number is %d \n\0"
37     .end
38
```

Name	Value	Size	Access
AArch64	699 of 699 registers		
Core	64 of 64 registers		
X0	0x0000000080046158	64	R/W
X1	0x0000000000000037	64	R/W
X2	0x0000000000000001	64	R/W
X3	0x00000000FEFFFFF0	64	R/W
X4	0x00000000FEFFFFE0	64	R/W
X5	0x0000000000000000	64	R/W
X6	0x0000000000000000	64	R/W
X7	0x0000000000000002	64	R/W
X8	0x0000000080046130	64	R/W
X9	0x0000000000000000	64	R/W
X10	0x0000000000000000	64	R/W
X11	0x0000000000000000	64	R/W
X12	0x0000000000000000	64	R/W
X13	0x0000000000000000	64	R/W
X14	0x0000000000000000	64	R/W
X15	0x0000000000000000	64	R/W
X16	0x0000000000000000	64	R/W
X17	0x0000000000000000	64	R/W
X18	0x0000000000000000	64	R/W
X19	0x0000000000000037	64	R/W
X20	0x0000000000000000	64	R/W

```
terminal_2: Listening for serial connection on port 5002
terminal_3: Listening for serial connection on port 5003
CADI server started listening to port 7000


Info: Foundation_AEMv8A: CADI Debug Server started for ARM Models...

CADI server is reported on port 7000
```

Function Prototype	Source/Line
<No Function Name>	fibonacci.s:20

Name	Value	Type	Count	Size	Location	Access

For the first picture I just screenshotted at the beginning of the program, and for the second picture I screenshotted before the program ended. As shown the highlighted register is that of x0 where the program loaded the address of prt_str into the register. Also, you can see the register x2 after the screenshot has 0x000...001. That is the value in x2 in this case which is 1 as it should be to pass line 17.