

Name: Breona Pizzuta

Partner (if any): Ben Carpenter

Pledge: "I pledge my honor that I have abided by the Stevens Honor System."

CS 382 Lab 4 Task 2

Start by using the `b _start` command:

```
[ Register Values Unavailable ]

lab4.s
b+> 10  MOV X0, 0 //Set register X0 to 0
      11  MOV X1, 0 //Set register X1 to 0
      12
      13  ADR X2, vec1 //Loads vec1 into register X2
      14  ADR X3, vec2 //Loads vec2 into register X3
      15  ADR X4, dot //Loads dot into register X4
      16
      17  LDR X5, [X2, 0] //Load the first element of vec1
      18  LDR X6, [X3, 0] //Load the first element of vec2
      19
      20  MUL X7, X5, X6 //Multiplies X5 and X6 storing in X7
      21  ADD X0, X0, X7 //Dest X0 add X0 and X7
      22

remote Thread 1.5002 In: _start
(gdb) b _start
Breakpoint 1 at 0x4000b0: file lab4.s, line 10.
(gdb) █
```

Step through the first 19 lines:

Here we are loading in the first elements of our vectors.

```
Register group: general
x0      0x0      0
x1      0x0      0
x2      0x410104 4260100
x3      0x41011c 4260124
x4      0x410134 4260148
x5      0xa      10
x6      0x1      1
x7      0x0      0
x8      0x0      0
x9      0x0      0
x10     0x0      0
x11     0x0      0
x12     0x0      0

lab4.s
b+ 10     MOV X0, 0 //Set register X0 to 0
    11     MOV X1, 0 //Set register X1 to 0
    12
    13     ADR X2, vec1 //Loads vec1 into register X2
    14     ADR X3, vec2 //Loads vec2 into register X3
    15     ADR X4, dot //Loads dot into register X4
    16
    17     LDR X5, [X2, 0] //Load the first element of vec1
    18     LDR X6, [X3, 0] //Load the first element of vec2
    19
> 20     MUL X7, X5, X6 //Multiplies X5 and X6 storing in X7
    21     ADD X0, X0, X7 //Dest X0 add X0 and X7
    22

remote Thread 1.5002 In: start L20 PC: 0x400000
Breakpoint 1 at 0x4000b0: file lab4.s, line 10.
(gdb) step
=> 0x00000000004000b4 <_start+4>: 01 00 80 d2 mov x1, #0x0
/ #0
(gdb) step
=> 0x00000000004000b8 <_start+8>: 62 02 08 10 adr x2, 0x410104
(gdb) step
=> 0x00000000004000bc <_start+12>: 03 03 08 10 adr x3, 0x41011c
(gdb) step
=> 0x00000000004000c0 <_start+16>: a4 03 08 10 adr x4, 0x410134
(gdb) step
=> 0x00000000004000c4 <_start+20>: 45 00 40 f9 ldr x5, [x2]
(gdb) step
=> 0x00000000004000c8 <_start+24>: 66 00 40 f9 ldr x6, [x3]
(gdb) step
=> 0x00000000004000cc <_start+28>: a7 7c 06 9b mul x7, x5, x6
(gdb) █
```

Step through next 2 lines:

Here we store the multiplication and add the product so far

```
Register group: general
x0      0xa      10
x1      0x0      0
x2      0x410104  4260100
x3      0x41011c  4260124
x4      0x410134  4260148
x5      0xa      10
x6      0x1      1
x7      0xa      10
x8      0x0      0
x9      0x0      0
x10     0x0      0
x11     0x0      0
x12     0x0      0

lab4.s
16
17     LDR X5, [X2, 0] //Load the first element of vec1
18     LDR X6, [X3, 0] //Load the first element of vec2
19
20     MUL X7, X5, X6 //Multiplies X5 and X6 storing in X7
21     ADD X0, X0, X7 //Dest X0 add X0 and X7
22
23
> 24     LDR X8, [X2, 8] //Load the second element of vec1
25     LDR X9, [X3, 8] //Load the second element of vec2
26
27     MUL X7, X8, X9 //Multiplies X5 and X6 storing in X7
28     ADD X0, X0, X7 //Dest X0 add X0 and X7

Remote Thread 1.5141 In: start L24 PC: 0x40
gdb) step
> 0x0000000004000b8 <_start+8>: 62 02 08 10 adr x2, 0x410104
gdb) step
> 0x0000000004000bc <_start+12>: 03 03 08 10 adr x3, 0x41011c
gdb) step
> 0x0000000004000c0 <_start+16>: a4 03 08 10 adr x4, 0x410134
gdb) step
> 0x0000000004000c4 <_start+20>: 45 00 40 f9 ldr x5, [x2]
gdb) step
> 0x0000000004000c8 <_start+24>: 66 00 40 f9 ldr x6, [x3]
gdb) step
> 0x0000000004000cc <_start+28>: a7 7c 06 9b mul x7, x5, x6
gdb) step
> 0x0000000004000d0 <_start+32>: 00 00 07 8b add x0, x0, x7
gdb) step
> 0x0000000004000d4 <_start+36>: 48 04 40 f9 ldr x8, [x2, #8]
gdb)
```

Step through the next 2 lines:

Here we are loading in the 2nd elements of our vectors.

```
Register group: general
x0      0xa      10
x1      0x0      0
x2      0x410104 4260100
x3      0x41011c 4260124
x4      0x410134 4260148
x5      0xa      10
x6      0x1      1
x7      0xa      10
x8      0x14     20
x9      0x2      2
x10     0x0      0
x11     0x0      0
x12     0x0      0

Lab4.s
22
23
24     LDR X8, [X2,8] //Load the second element of vec1
25     LDR X9, [X3,8] //Load the second element of vec2
26
> 27     MUL X7, X8, X9 //Multiplies X5 and X6 storing in X7
28     ADD X0, X0, X7 //Dest X0 add X0 and X7
29
30     LDR X10, [X2,16] //Load the third element of vec1
31     LDR X11, [X3,16] //Load the thrid element of vec2
32
33     MUL X7, X10, X11 //Multiplies X5 and X6 stored in X7
34     ADD X0, X0, X7 //Dest X0 add X0 and X7

Remote Thread 1.5141 In: _start L27
gdb) step
> 0x00000000004000c0 <_start+16>: a4 03 08 10 adr x4, 0x410134
gdb) step
> 0x00000000004000c4 <_start+20>: 45 00 40 f9 ldr x5, [x2]
gdb) step
> 0x00000000004000c8 <_start+24>: 66 00 40 f9 ldr x6, [x3]
gdb) step
> 0x00000000004000cc <_start+28>: a7 7c 06 9b mul x7, x5, x6
gdb) step
> 0x00000000004000d0 <_start+32>: 00 00 07 8b add x0, x0, x7
gdb) step
> 0x00000000004000d4 <_start+36>: 48 04 40 f9 ldr x8, [x2, #8]
gdb) step
> 0x00000000004000d8 <_start+40>: 69 04 40 f9 ldr x9, [x3, #8]
gdb) step
Show Applications 000dc <_start+44>: 07 7d 09 9b mul x7, x8, x9
gdb) step
```

Step through next 2 lines:

Here we store the multiplication of the second vector and add the product so far

```
Register group: general
x0      0x32      50
x1      0x0       0
x2      0x410104  4260100
x3      0x41011c  4260124
x4      0x410134  4260148
x5      0xa       10
x6      0x1       1
x7      0x28      40
x8      0x14      20
x9      0x2       2
x10     0x0       0
x11     0x0       0
x12     0x0       0

lab4.s
22
23
24     LDR X8, [X2,8] //Load the second element of vec1
25     LDR X9, [X3,8] //Load the second element of vec2
26
27     MUL X7, X8, X9 //Multiplies X5 and X6 storing in X7
28     ADD X0, X0, X7 //Dest X0 add X0 and X7
29
> 30     LDR X10, [X2,16] //Load the third element of vec1
31     LDR X11, [X3,16] //Load the thrid element of vec2
32
33     MUL X7, X10, X11 //Multiplies X5 and X6 stored in X7
34     ADD X0, X0, X7 //Dest X0 add X0 and X7

remote Thread 1.5141 In: _start L30 PC: 0x40
(gdb) step
=> 0x00000000004000c8 <_start+24>: 66 00 40 f9    ldr    x6, [x3]
(gdb) step
=> 0x00000000004000cc <_start+28>: a7 7c 06 9b    mul    x7, x5, x6
(gdb) step
=> 0x00000000004000d0 <_start+32>: 00 00 07 8b    add    x0, x0, x7
(gdb) step
=> 0x00000000004000d4 <_start+36>: 48 04 40 f9    ldr    x8, [x2, #8]
(gdb) step
=> 0x00000000004000d8 <_start+40>: 69 04 40 f9    ldr    x9, [x3, #8]
(gdb) step
=> 0x00000000004000dc <_start+44>: 07 7d 09 9b    mul    x7, x8, x9
(gdb) step
=> 0x00000000004000e0 <_start+48>: 00 00 07 8b    add    x0, x0, x7
(gdb) step
=> 0x00000000004000e4 <_start+52>: 4a 08 40 f9    ldr    x10, [x2, #16]
(gdb) |
```

Step through the next 2 lines:

Here we are loading in the 3rd elements of our vectors.

```
Register group: general
x0      0x32      50
x1      0x0       0
x2      0x410104  4260100
x3      0x41011c  4260124
x4      0x410134  4260148
x5      0xa       10
x6      0x1       1
x7      0x28      40
x8      0x14      20
x9      0x2       2
x10     0x1e      30
x11     0x3       3
x12     0x0       0

lab4.s
28      ADD X0, X0, X7 //Dest X0 add X0 and X7
29
30      LDR X10, [X2,16] //Load the third element of vec1
31      LDR X11, [X3,16] //Load the thrid element of vec2
32
> 33      MUL X7, X10, X11 //Multiplies X5 and X6 stored in X7
34      ADD X0, X0, X7 //Dest X0 add X0 and X7
35
36      STR X0, [X4] //Store the result into X4
37
38
39
40

Remote Thread 1.5141 In: _start L33 PC:
(gdb) step
=> 0x00000000004000d0 <_start+32>: 00 00 07 8b add x0, x0, x7
(gdb) step
=> 0x00000000004000d4 <_start+36>: 48 04 40 f9 ldr x8, [x2, #8]
(gdb) step
=> 0x00000000004000d8 <_start+40>: 69 04 40 f9 ldr x9, [x3, #8]
(gdb) step
=> 0x00000000004000dc <_start+44>: 07 7d 09 9b mul x7, x8, x9
(gdb) step
=> 0x00000000004000e0 <_start+48>: 00 00 07 8b add x0, x0, x7
(gdb) step
=> 0x00000000004000e4 <_start+52>: 4a 08 40 f9 ldr x10, [x2, #16]
(gdb) step
=> 0x00000000004000e8 <_start+56>: 6b 08 40 f9 ldr x11, [x3, #16]
(gdb) step
=> 0x00000000004000ec <_start+60>: 47 7d 0b 9b mul x7, x10, x11
(gdb)
```

Step through next 2 lines:

Here we store the multiplication of the third vector and add the product so far

```
Register group: general
x0      0x8c      140
x1      0x0       0
x2      0x410104  4260100
x3      0x41011c  4260124
x4      0x410134  4260148
x5      0xa       10
x6      0x1       1
x7      0x5a      90
x8      0x14      20
x9      0x2       2
x10     0x1e      30
x11     0x3       3
x12     0x0       0

lab4.s
28      ADD X0, X0, X7 //Dest X0 add X0 and X7
29
30      LDR X10, [X2,16] //Load the third element of vec1
31      LDR X11, [X3,16] //Load the thrid element of vec2
32
33      MUL X7, X10, X11 //Multiplies X5 and X6 stored in X7
34      ADD X0, X0, X7 //Dest X0 add X0 and X7
35
> 36      STR X0, [X4] //Store the result into X4
37
38
39
40

remote Thread 1.5141 In: _start L36 PC: 0x4000
(gdb) step
=> 0x00000000004000d8 <_start+40>: 69 04 40 f9 ldr x9, [x3, #8]
(gdb) step
=> 0x00000000004000dc <_start+44>: 07 7d 09 9b mul x7, x8, x9
(gdb) step
=> 0x00000000004000e0 <_start+48>: 00 00 07 8b add x0, x0, x7
(gdb) step
=> 0x00000000004000e4 <_start+52>: 4a 08 40 f9 ldr x10, [x2, #16]
(gdb) step
=> 0x00000000004000e8 <_start+56>: 6b 08 40 f9 ldr x11, [x3, #16]
(gdb) step
=> 0x00000000004000ec <_start+60>: 47 7d 0b 9b mul x7, x10, x11
(gdb) step
=> 0x00000000004000f0 <_start+64>: 00 00 07 8b add x0, x0, x7
(gdb) step
=> 0x00000000004000f4 <_start+68>: 80 00 00 f9 str x0, [x4]
(gdb)
```

Step until line 40:

Use x/1dg &dot: The result of the dot product should be stored here.

We see it prints 140 which is the dot product

```
Register group: general
x0      0x8c      140
x1      0x0       0
x2      0x410104  4260100
x3      0x41011c  4260124
x4      0x410134  4260148
Files   0xa       10
        0x1       1
x7      0x5a      90
x8      0x14      20
x9      0x2       2
x10     0x1e      30
x11     0x3       3
x12     0x0       0

lab4.s
31      LDR X11, [X3,16] //Load the thrid element of vec2
32
33      MUL X7, X10, X11 //Multiplies X5 and X6 stored in X7
34      ADD X0, X0, X7   //Dest X0 add X0 and X7
35
36      STR X0, [X4]     //Store the result into X4
37
38
39
40
> 41      MOV X0, 0      // Set register X0 to 0 (return code)
42      MOV X8, 93      // Set register X8 to 93 (syscall number for exit)
43      SVC 0           // Invoke syscall to exit

remote Thread 1.5141 In: start L41 PC: 0>
=> 0x00000000004000e0 <_start+48>: 00 00 07 8b add x0, x0, x7
(gdb) step
=> 0x00000000004000e4 <_start+52>: 4a 08 40 f9 ldr x10, [x2, #16]
(gdb) step
=> 0x00000000004000e8 <_start+56>: 6b 08 40 f9 ldr x11, [x3, #16]
(gdb) step
=> 0x00000000004000ec <_start+60>: 47 7d 0b 9b mul x7, x10, x11
(gdb) step
=> 0x00000000004000f0 <_start+64>: 00 00 07 8b add x0, x0, x7
(gdb) step
=> 0x00000000004000f4 <_start+68>: 80 00 00 f9 str x0, [x4]
(gdb) step
=> 0x00000000004000f8 <_start+72>: 00 00 80 d2 mov x0, #0x0
/ #0
(gdb) x/1dg &dot
0x410134: 140
(gdb)
```


Step to end of program:
Ends program

Register group: general		
x0	0x0	0
x1	0x0	0
x2	0x410104	4260100
x3	0x41011c	4260124
x4	0x410134	4260148
x5	0xa	10
x6	0x1	1
x7	0x5a	90
x8	0x5d	93
x9	0x2	2
x10	0x1e	30
x11	0x3	3
x12	0x0	0

lab4.s	
43	SVC 0 // Invoke syscall to exit
44	
45	.data
46	vec1: .quad 10, 20, 30
47	vec2: .quad 1, 2, 3
48	dot: .quad 0
49	
50	
51	
52	
53	
54	
55	

exec No process in:

L??

```
(gdb) step
=> 0x00000000004000f0 <_start+64>:      00 00 07 8b      add     x0, x0, x7
(gdb) step
=> 0x00000000004000f4 <_start+68>:      80 00 00 f9      str     x0, [x4]
(gdb) step
=> 0x00000000004000f8 <_start+72>:      00 00 80 d2      mov     x0, #0x0
/ #0
(gdb) x/1dg &dot
0x410134:      140
(gdb) step
=> 0x00000000004000fc <_start+76>:      a8 0b 80 d2      mov     x8, #0x5d
/ #93
(gdb) step
=> 0x0000000000400100 <_start+80>:      01 00 00 d4      svc     #0x0
(gdb) step
[Inferior 1 (process 1) exited normally]
(gdb)
```