RISC Based Myth Workshop, Lab Day 2

Arijit Sengupta

July 31, 2020

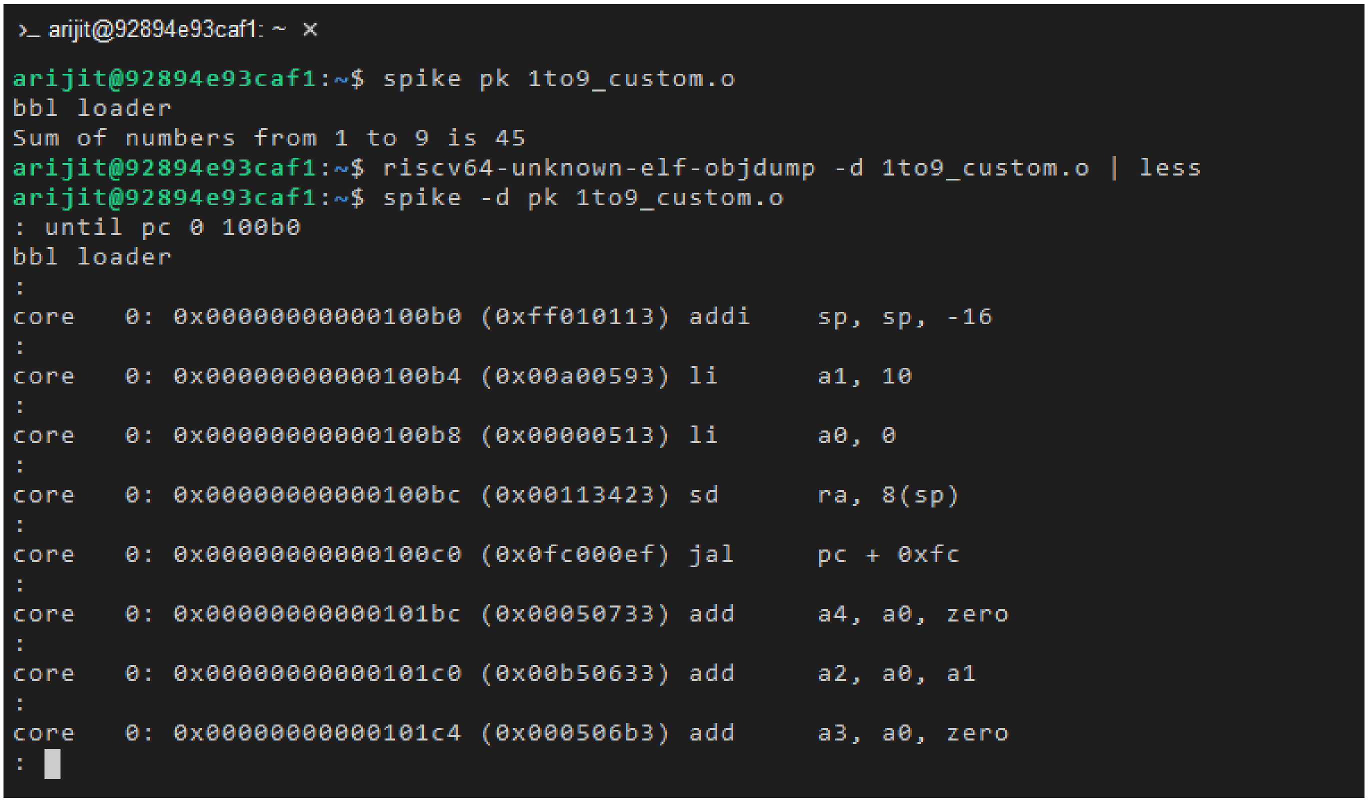
1 1to9_custom.c

```
#include⟨stdio.h⟩
extern int load(int x, int y);
int main() {
    int result = 0;
    int count = 9;
    result = load(0x0, count+1);
    printf(”Sum of numbers from 1 to %d is %d”, count, result);
}
```

2 load.S

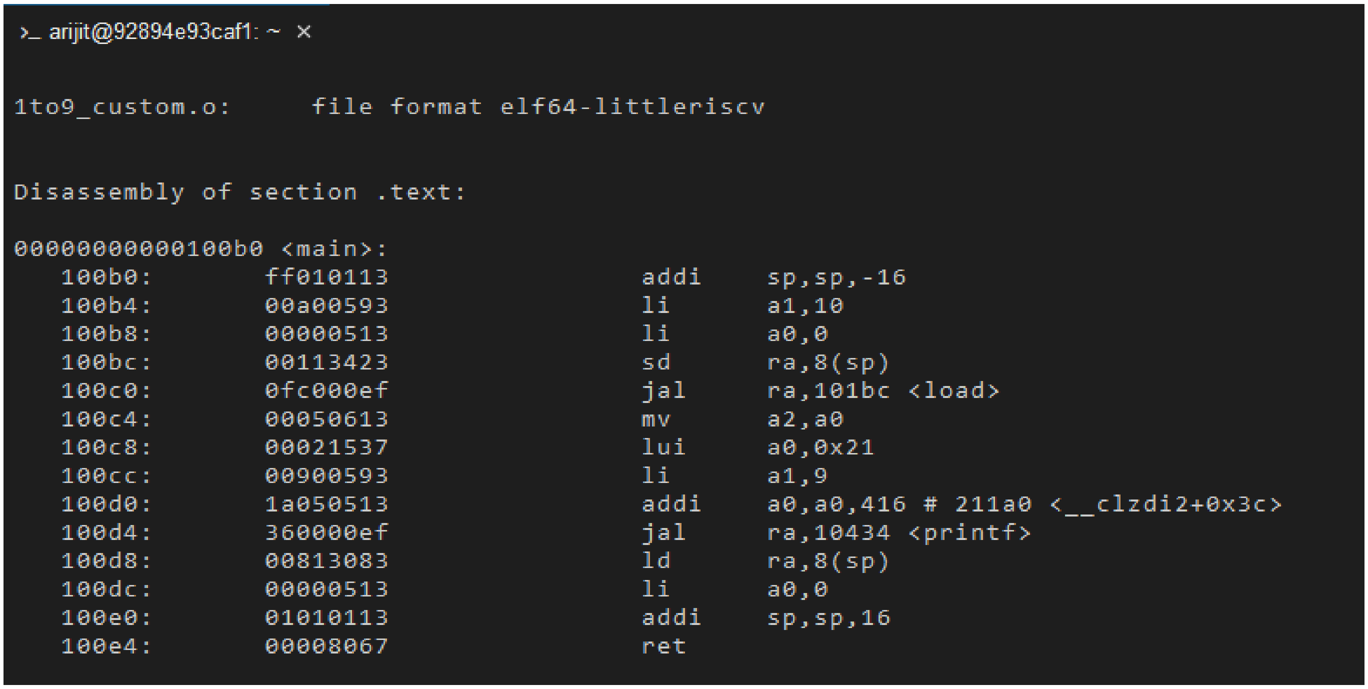
```
.section .text
.global load
.type load, @function
load:
    add    a4, a0, zero //Initialize sum register a4 with 0x0
    add    a2, a0, a1 //Store count of 10 in register a2. Register a1 is loaded with 0xa (decimal 10) from main
    add    a3, a0, zero //Initialize intermediate sum register a3 by 0
loop:
    add    a4, a3, a4 //Incremental addition
    addi   a3, a3, 1 //Increment intermediate register by 1
    blt    a3, a2, loop //If a3 is less than a2, branch to label named ¡loop¿
    add    a0, a4, zero //Store final result to register a0 so that it can be read by main program
    ret
```

3 Spike Debugger & Objdump



```
arijit@92894e93caf1:~$ spike pk 1to9_custom.o
bbl loader
Sum of numbers from 1 to 9 is 45
arijit@92894e93caf1:~$ riscv64-unknown-elf-objdump -d 1to9_custom.o | less
arijit@92894e93caf1:~$ spike -d pk 1to9_custom.o
: until pc 0 100b0
bbl loader
:
core   0: 0x00000000000100b0 (0xff010113) addi    sp, sp, -16
:
core   0: 0x00000000000100b4 (0x00a00593) li      a1, 10
:
core   0: 0x00000000000100b8 (0x00000513) li      a0, 0
:
core   0: 0x00000000000100bc (0x00113423) sd      ra, 8(sp)
:
core   0: 0x00000000000100c0 (0x0fc000ef) jal     pc + 0xfc
:
core   0: 0x00000000000101bc (0x00050733) add     a4, a0, zero
:
core   0: 0x00000000000101c0 (0x00b50633) add     a2, a0, a1
:
core   0: 0x00000000000101c4 (0x000506b3) add     a3, a0, zero
:
```

Figure 1: Spike Debugger output



```

1to9_custom.o:     file format elf64-littleriscv


Disassembly of section .text:

00000000000100b0 <main>:
   100b0:       ff010113                addi    sp,sp,-16
   100b4:       00a00593                li      a1,10
   100b8:       00000513                li      a0,0
   100bc:       00113423                sd      ra,8(sp)
   100c0:       0fc000ef                jal     ra,101bc <load>
   100c4:       00050613                mv      a2,a0
   100c8:       00021537                lui     a0,0x21
   100cc:       00900593                li      a1,9
   100d0:       1a050513                addi    a0,a0,416 # 211a0 <__clzdi2+0x3c>
   100d4:       360000ef                jal     ra,10434 <printf>
   100d8:       00813083                ld      ra,8(sp)
   100dc:       00000513                li      a0,0
   100e0:       01010113                addi    sp,sp,16
   100e4:       00008067                ret
```

Figure 2: Objdump output