

Assignment 3

Adarsh Tiwari (B21ES002)

September 2023

1 Question 1

1.1 Part a

```
.data

input: .string "Hello World!"

.text
.globl main

print_string:
    addi sp sp -16
    sw ra 0(sp)
    sw a0 4(sp)
    sw a1 8(sp)      #what if we know that we have only one input argument

    add t0 x0 a0
Loop:    lbu a1,0(t0)
        beq a1,x0,Return
        addi a0,x0,11
        ecall
        addi t0 t0 1
    j Loop

    lw ra 0(sp)
    lw a0 4(sp)
    lw a1 8(sp)
    addi sp sp 12

Return:    ret

main:

    la a0 input
    jal ra print_string

    addi a0 x0 10
    ecall
```

VenusEditorSimulatorChocopy

RunStepPrevResetDumpTraceRe-assemble from Editor

0x8	0x00A12223	sw x10 4(x2)	sw a0 4(sp)
0xc	0x00B12423	sw x11 8(x2)	sw a1 8(sp) #what if we know that we have only one input argument
0x10	0x00A002B3	add x5 x0 x10	add t0 x0 a0
0x14	0x0002C583	lbu x11 0(x5)	Loop: lbu a1,0(t0)
0x18	0x02058263	beq x11 x0 36	beq a1,x0,Return
0x1c	0x00B00513	addi x10 x0 11	addi a0,x0,11
0x20	0x00000073	ecall	ecall
0x24	0x00128293	addi x5 x5 1	addi t0 t0 1
0x28	0xFEDFF06F	jal x0 -20	j Loop
0x2c	0x00012083	lw x1 0(x2)	lw ra 0(sp)

Copy!Download!Clear!

Hello World!

RegistersMemoryCacheVDB

Integer (R)Floating (F)

zero	0x00000000
ra (x1)	0x0000004C
sp (x2)	0x7FFFFFF0
gp (x3)	0x10000000
tp (x4)	0x00000000
t0 (x5)	0x1000000C
t1 (x6)	0x00000000
t2 (x7)	0x00000000
s0 (x8)	0x00000000
s1 (x9)	0x00000000
a0 (x10)	0x0000000A
a1	0x00000000

Display Settings

Hex

Figure 1: Output

1.2 Part b

```
.data
input: .string "Hello World!"

.text
.globl main

print_string:
    addi sp sp -16
    sw ra 0(sp)
    sw a0 4(sp)
    sw a1 8(sp)

    add t0 x0 a1    # copy the length of string
    add t1 x0 a0     # copy the base address of string
Loop: beq t0 x0 Return
    addi t0 t0 -1
    addi a0 x0 11
    lbu a1 0(t1)
    ecall
    addi t1 t1 1
    j Loop

Return:    lw ra 0(sp)
    lw a0 4(sp)
    lw a1 8(sp)
    addi sp sp +16
    ret

main:
    la a0 input
    addi a1 x0 12
    jal ra print_string

    addi a0 x0 10
    ecall
```

Venus
Editor
Simulator
Chocopy

Run
Step
Prev
Reset
Dump
Trace
Re-assemble from Editor

0x20	0x00B00513	addi x10 x0 11	addi a0 x0 11
0x24	0x00034583	lbu x11 0(x6)	lbu a1 0(t1)
0x28	0x00000073	ecall	ecall
0x2c	0x00130313	addi x6 x6 1	addi t1 t1 1
0x30	0xFE9FF06F	jal x0 -24	j Loop
0x34	0x00012083	lw x1 0(x2)	Return: lw ra 0(sp)
0x38	0x00412503	lw x10 4(x2)	lw a0 4(sp)
0x3c	0x00812583	lw x11 8(x2)	lw a1 8(sp)
0x40	0x01010113	addi x2 x2 16	addi sp sp +16
0x44	0x00008067	jalr x0 x1 0	ret
0x48	0x10000517	auipc x10 65536	la a0 input

Copy!
Download!
Clear!

Hello World!

Registers
Memory
Cache
VDB

Integer (R)
Floating (F)

zero	0x00000000
ra (x1)	0x00000058
sp (x2)	0x7FFFFFF0
gp (x3)	0x10000000
tp (x4)	0x00000000
t0 (x5)	0x00000000
t1 (x6)	0x1000000C
t2 (x7)	0x00000000
s0 (x8)	0x00000000
s1 (x9)	0x00000000
a0 (x10)	0x0000000A
a1	0x0000000C

Display Settings
Hex

Figure 2: Output

1.3 Part c

```
.data
input: .string "Hello World!"

.text
.globl main

print_string:
    addi sp sp -16
    sw ra 0(sp)
    sw a0 4(sp)
    sw a1 8(sp)

    add t1 x0 a0    # copy the base address of string
    lbu t0 0(t1)
    beq t0 x0 Return
Loop:
    lbu a1 0(t1)
    addi a0 x0 11
    ecall
    addi t1 t1 1
    beq a1 x0 Return
    j Loop

Return:    lw ra 0(sp)
    lw a0 4(sp)
    lw a1 8(sp)
    addi sp sp +16
    ret

main:

    la a0 input
    jal ra print_string

    addi a0 x0 10
    ecall
```

VenusEditorSimulatorChocopy

RunStepPrevResetDumpTraceRe-assemble from Editor

PC	Instruction	Assembly	Disassembly
0x24	0x00B00513	addi x10 x0 11	addi a0 x0 11
0x28	0x00000073	ecall	ecall
0x2c	0x00130313	addi x6 x6 1	addi t1 t1 1
0x30	0xFEDFF06F	jal x0 -20	j Loop
0x34	0x00012083	lw x1 0(x2)	Return: lw ra 0(sp)
0x38	0x00412503	lw x10 4(x2)	lw a0 4(sp)
0x3c	0x00812583	lw x11 8(x2)	lw a1 8(sp)
0x40	0x01010113	addi x2 x2 16	addi sp sp +16
0x44	0x00008067	jalr x0 x1 0	ret
0x48	0x10000517	auipc x10 65536	la a0 input
0x4c	0xFB850513	addi x10 x10 -72	la a0 input

Copy!Download!Clear!

Hello World!

RegistersMemoryCacheVDB

Integer (R)Floating (F)

zero	0x00000000
ra (x1)	0x00000054
sp (x2)	0x7FFFFFF0
gp (x3)	0x10000000
tp (x4)	0x00000000
t0 (x5)	0x00000048
t1 (x6)	0x1000000C
t2 (x7)	0x00000000
s0 (x8)	0x00000000
s1 (x9)	0x00000000
a0 (x10)	0x0000000A
a1	0x7FFFFFF0

Display SettingsHex

Figure 3: Output

2 Question 3

```
.text
.globl main

next_half_int:

    addi a0 x0 0
    addi a0 a0 1

    addi t0 t0 1
    slli t0 t0 15

    slt t1 a0 t0
    bne t1 x0 True1
    addi a0 x0 0

True1:    slli t0 t0 1
    sub t0 x0 t0

    slt t1 t0 a0

    bne t1 x0 True2
    addi a0 x0 0

True2:    jr ra


main:
    jal next_half_int

    add a1 a0 x0
    addi a0 x0 1

    ecall

    addi a0 x0 10

    ecall
```

Venus
Editor
Simulator
Chocopy

Run
Step
Prev
Reset
Dump
Trace
Re-assemble from Editor

0x18	0x00000513	addi x10 x0 0	addi a0 x0 0
0x1c	0x00129293	slli x5 x5 1	True1: slli t0 t0 1
0x20	0x405002B3	sub x5 x0 x5	sub t0 x0 t0
0x24	0x00A2A333	slt x6 x5 x10	slt t1 t0 a0
0x28	0x00031463	bne x6 x0 8	bne t1 x0 True2
0x2c	0x00000513	addi x10 x0 0	addi a0 x0 0
0x30	0x00008067	jalr x0 x1 0	True2: jr ra
0x34	0xFCDF0EF	jal x1 -52	jal next_half_int
0x38	0x000005B3	add x11 x10 x0	add a1 a0 x0
0x3c	0x00100513	addi x10 x0 1	addi a0 x0 1
0x40	0x00000073	ecall	ecall

Copy!
Download!
Clear!

1

RegistersMemoryCacheVDB

Integer (R)Floating (F)

zero0x00000000
ra (x1)0x00000038
sp (x2)0x7FFFFFFE
gp (x3)0x10000000
tp (x4)0x00000000
t0 (x5)0xFFFFF000
t1 (x6)0x00000001
t2 (x7)0x00000000
s0 (x8)0x00000000
s1 (x9)0x00000000
a0 (x10)0x0000000A
a10x00000001

Display Settings
Hex

Figure 4: Output

3 Question 5

```
.data
neg3:  .ascii "f(-3) is 11, and it is: "
neg2:  .ascii "f(-2) is 21, and it is: "
neg1:  .ascii "f(-1) is 45, and it is: "
zero:  .ascii "f(0) is -3, and it is: "
pos1:  .ascii "f(1) is 12, and it is: "
pos2:  .ascii "f(2) is 62, and it is: "
pos3:  .ascii "f(3) is 16, and it is: "

output: .word 11, 21, 45, -3, 12, 62, 16
.text
main:
    la    a0, neg3
    jal   print_str
    li    a0, -3
    jal   f                # f(-3); is 11
    jal   print_int
    jal   newline

    la    a0, neg2
    jal   print_str
    li    a0, -2
    jal   f                # f(-2); is 21
    jal   print_int
    jal   newline

    la    a0, neg1
    jal   print_str
    li    a0, -1
    jal   f                # f(-1); is 45
    jal   print_int
    jal   newline

    la    a0, zero
    jal   print_str
    li    a0, 0
    jal   f                # f(0); is -3
    jal   print_int
    jal   newline

    la    a0, pos1
    jal   print_str
    li    a0, 1
    jal   f                # f(1); is 12
    jal   print_int
    jal   newline

    la    a0, pos2
    jal   print_str
    li    a0, 2
    jal   f                # f(2); is 62
    jal   print_int
```

```

        jal        newline

        la         a0, pos3
        jal        print_str
        li         a0, 3
        jal        f                # evaluate C(4,0); is 16
        jal        print_int
        jal        newline

        li         a0, 10
        ecall

# calculate f(a0)
f:
        la         t0, output      # Why is this a good idea?

        addi a0 a0 3
        slli a0 a0 2

        add t0 t0 a0
        lw a0 0(t0)

        # YOUR CODE

        jr         ra              # Remember to jr ra after your function!

print_int:
        mv a1, a0
        li         a0, 1
        ecall
        jr         ra

print_str:
        mv a1, a0
        li         a0, 4
        ecall
        jr         ra

newline:
        li         a1, '\n'
        li         a0, 11
        ecall
        jr         ra

```

Run

Step

Prev

Reset

Dump

Trace

Re-assemble from Editor

0xec8	0x00000000	addi x11, x10, 0	mv a1, a0
0xec	0x00100513	addi x10, x0, 1	li a0, 1
0xf0	0x00000073	ecall	ecall
0xf4	0x00000067	jalr x0, x1, 0	jr ra
0xf8	0x00050593	addi x11, x10, 0	mv a1, a0
0xfc	0x00400513	addi x10, x0, 4	li a0, 4
0x100	0x00000073	ecall	ecall
0x104	0x00000067	jalr x0, x1, 0	jr ra
0x108	0x00A00593	addi x11, x0, 10	li a1, '\n'
0x10c	0x00B00513	addi x10, x0, 11	li a0, 11
0x110	0x00000073	ecall	ecall
0x114	0x00000067	jalr x0, x1, 0	jr ra

Copy!

Download!

Clear!

```

f(-3) is 11, and it is: 11
f(-2) is 21, and it is: 21
f(-1) is 45, and it is: 45
f(0) is -3, and it is: -3
f(1) is 12, and it is: 12
f(2) is 62, and it is: 62
f(3) is 16, and it is: 16

```

Registers

Memory

Cache

VDB

Integer (R)

Floating (F)

zero	0x00000000
ra (x1)	0x000000C4
sp (x2)	0x7FFFFFFE0
gp (x3)	0x10000000
tp (x4)	0x00000000
t0 (x5)	0x100000C3
t1 (x6)	0x00000000
t2 (x7)	0x00000000
s0 (x8)	0x00000000
s1 (x9)	0x00000000
a0 (x10)	0x0000000A
a1 (x11)	0x0000000A
a2 (x12)	0x00000000
a3 (x13)	0x00000000

Display

Settings

Hex

Figure 5: Output