

I spent 2h for this lab.

Divide by 9

Code:

```
1 main:
2 addi t0, zero, 9 # t0 = 9
3 addi a1, zero, 3 # a1 = 3
4 addi t1, zero, a0 # t1 = a0
5 addi t2, zero, 8 # t2 = 8
6 jal subtract # call function
7
8 subtract:
9 sub t1, t1, t0 # t1 = a0 - 9
10 beq t1, zero, equal # t1 = h + i
11 ble t1, t2, nequal # 9 > t1 nequal
12 j subtract # subtract again
13
14 equal:
15 addi a0, zero, 1 # a0 is divisible by 9
16 j end
17
18 nequal:
19 addi a0, zero, 0 # a0 is not divisible by 9
20 j end
21
22 end:
23 #finished
```



Simulation:  
a0 = 90

RunStepPrevResetDump

Machine Code	Basic Code	Original Code
0x00900293	addi x5 x0 9	addi t0, zero, 9 # t0 = 9
0x00300593	addi x11 x0 3	addi a1, zero, 3 # a1 = 3
0x00a00333	add x6 x0 x10	add t1, zero, a0 # t1 = a0
0x00800393	addi x7 x0 8	addi t2, zero, 8 # t2 = 8
0x004000ef	jal x1 4	jal subtract # call function
0x40530333	sub x6 x6 x5	sub t1, t1, t0 # t1 = a0 - 9
0x00030663	beq x6 x0 12	beq t1, zero, equal # t1 = h + i
0x0043d8e3	bge x7 x6 16	ble t1, t2, nequal # 9 > t1 nequal
0xffff006f	jal x0 -12	j subtract # subtract again
0x00100513	addi x10 x0 1	addi a0, zero, 1 # a0 is divisible by 9
0x00c0006f	jal x0 12	j end
0x00000513	addi x10 x0 0	addi a0, zero, 0 # a0 is not divisible by 9
0x0040006f	jal x0 4	j end

console output

RegistersMemory

zero	0
ra (x1)	0
sp (x2)	2147483632
gp (x3)	200435406
tp (x4)	0
t0 (x5)	9
t1 (x6)	0
t2 (x7)	0
a0 (x8)	0
a1 (x9)	0
a0 (x10)	90
a1 (x11)	0
a2 (x12)	0
a3 (x13)	0
a4 (x14)	0
a5 (x15)	0
a6 (x16)	0
a7 (x17)	0

Display SettingsDecimal

a0 = 1, divisible

RunStepPrevResetDump

Machine Code	Basic Code	Original Code
0x00900293	addi x5 x0 9	addi t0, zero, 9 # t0 = 9
0x00300593	addi x11 x0 3	addi a1, zero, 3 # a1 = 3
0x00a00333	add x6 x0 x10	add t1, zero, a0 # t1 = a0
0x00800393	addi x7 x0 8	addi t2, zero, 8 # t2 = 8
0x004000ef	jal x1 4	jal subtract # call function
0x40530333	sub x6 x6 x5	sub t1, t1, t0 # t1 = a0 - 9
0x00030663	beq x6 x0 12	beq t1, zero, equal # t1 = h + i
0x0043d8e3	bge x7 x6 16	ble t1, t2, nequal # 9 > t1 nequal
0xffff006f	jal x0 -12	j subtract # subtract again
0x00100513	addi x10 x0 1	addi a0, zero, 1 # a0 is divisible by 9
0x00c0006f	jal x0 12	j end
0x00000513	addi x10 x0 0	addi a0, zero, 0 # a0 is not divisible by 9
0x0040006f	jal x0 4	j end

console output

RegistersMemory

zero	0
ra (x1)	0
sp (x2)	2147483632
gp (x3)	200435406
tp (x4)	0
t0 (x5)	9
t1 (x6)	0
t2 (x7)	8
a0 (x8)	0
a1 (x9)	0
a0 (x10)	1
a1 (x11)	3
a2 (x12)	0
a3 (x13)	0
a4 (x14)	0
a5 (x15)	0
a6 (x16)	0
a7 (x17)	0

Display SettingsDecimal

a0 = 98

RunStepPrevResetDump

Machine Code	Basic Code	Original Code
0x00900293	addi x5 x0 9	addi t0, zero, 9 # t0 = 9
0x00300593	addi x11 x0 3	addi a1, zero, 3 # a1 = 3
0x00a00333	add x6 x0 x10	add t1, zero, a0 # t1 = a0
0x00800393	addi x7 x0 8	addi t2, zero, 8 # t2 = 8
0x004000ef	jal x1 4	jal subtract # call function
0x40530333	sub x6 x6 x5	sub t1, t1, t0 # t1 = a0 - 9
0x00030663	beq x6 x0 12	beq t1, zero, equal # t1 = h + i
0x0043d863	bge x7 x6 16	ble t1, t2, unequal # 9 > t1 unequal
0xfffff06f	jal x0 -12	j subtract # subtract again
0x00100513	addi x10 x0 1	addi a0, zero, 1 # a0 is divisible by 9
0x00c0006f	jal x0 12	j end
0x00000513	addi x10 x0 0	addi a0, zero, 0 # a0 is not divisible by 9
0x0040006f	jal x0 4	j end

console output

RegistersMemory

zero	0
ra (x1)	0
sp (x2)	2147483632
gp (x3)	202435456
tp (x4)	0
t0 (x5)	0
t1 (x6)	0
t2 (x7)	0
s0 (x8)	0
s1 (x9)	0
a0 (x10)	98
a1 (x11)	0
a2 (x12)	0
a3 (x13)	0
a4 (x14)	0
a5 (x15)	0
a6 (x16)	0
a7 (x17)	0

Display SettingsDecimal

a0 = 0 not divisible

RunStepPrevResetDump

Machine Code	Basic Code	Original Code
0x00900293	addi x5 x0 9	addi t0, zero, 9 # t0 = 9
0x00300593	addi x11 x0 3	addi a1, zero, 3 # a1 = 3
0x00a00333	add x6 x0 x10	add t1, zero, a0 # t1 = a0
0x00800393	addi x7 x0 8	addi t2, zero, 8 # t2 = 8
0x004000ef	jal x1 4	jal subtract # call function
0x40530333	sub x6 x6 x5	sub t1, t1, t0 # t1 = a0 - 9
0x00030663	beq x6 x0 12	beq t1, zero, equal # t1 = h + i
0x0043d863	bge x7 x6 16	ble t1, t2, unequal # 9 > t1 unequal
0xfffff06f	jal x0 -12	j subtract # subtract again
0x00100513	addi x10 x0 1	addi a0, zero, 1 # a0 is divisible by 9
0x00c0006f	jal x0 12	j end
0x00000513	addi x10 x0 0	addi a0, zero, 0 # a0 is not divisible by 9
0x0040006f	jal x0 4	j end

console output

RegistersMemory

zero	0
ra (x1)	20
sp (x2)	2147483632
gp (x3)	202435456
tp (x4)	0
t0 (x5)	9
t1 (x6)	8
t2 (x7)	8
s0 (x8)	0
s1 (x9)	0
a0 (x10)	0
a1 (x11)	3
a2 (x12)	0
a3 (x13)	0
a4 (x14)	0
a5 (x15)	0
a6 (x16)	0
a7 (x17)	0

Display SettingsDecimal

big2little

code:

```
1 lui s0, 0xABCD00 # s0 = 0xABCD0000
2 addi s0, s0, 0x123 # s0 = 0xABCD0123
3 addi t0, zero, 0
4 addi t1, zero, 0
5 addi t2, zero, 32
6
7
8 main:
9 j load
10
11 load:
12 sw s0, 0x300(t0) # memory[300+t0] = 0xABCD0123
13 lb a0, 0x300(t0) # a0 = memory[300+t0]
14 lb a1, 0x301(t0) # a1 = memory[301+t0]
15 lb a2, 0x302(t0) # a2 = memory[302+t0]
16 lb a3, 0x303(t0) # a3 = memory[303+t0]
17 addi t0, t0, 4
18 j swap
19
20 swap:
21 sb a3, 0x300(t1) # a3 = memory[300+t1]
22 sb a2, 0x301(t1) # a2 = memory[301+t1]
23 sb a1, 0x302(t1) # a1 = memory[302+t1]
24 sb a0, 0x303(t1) # a0 = memory[303+t1]
25 addi t1, t1, 4
26 j eightwuds # word counter
27
28 eightwuds:
29 beq t0, t2, finish
30 j load
31
32 finish:
33
```



## Simulation:

### First word big-endian

Run Step Prev Reset Dump

Machine Code	Basic Code	Original Code
0xab0de437	lui x8 703710	lui x0, 0xab0de # x0 = 0xab0de000
0x12340413	addi x8 x8 291	addi x0, x0, 0x123 # x0 = 0xab0de123
0x00000293	addi x5 x0 0	addi t0, zero, 0
0x00000313	addi x6 x0 0	addi t1, zero, 0
0x02000393	addi x7 x0 32	addi t2, zero, 32
0x0040006f	jai x0 4	j load
0x3082a023	sw x8 768(x5)	sw x0, 0x300(t0) # memory[300+t0] = 0xab0de123
0x30028503	lb x0 768(x5)	lb a0, 0x300(t0) # a0 = memory[300+t0]
0x30128583	lb x1 768(x5)	lb a1, 0x301(t0) # a1 = memory[301+t0]
0x30228603	lb x12 770(x5)	lb a2, 0x302(t0) # a2 = memory[302+t0]
0x30328663	lb x13 771(x5)	lb a3, 0x303(t0) # a3 = memory[303+t0]
0x00428293	addi x5 x5 4	addi t0, t0, 4
0x0040006f	jai x0 4	j swap
0x30d30023	sb x13 768(x6)	sb a3, 0x300(t1) # a3 = memory[300+t1]

console output

Registers Memory

Address	+0	+1	+2	+3
0x00000318	00	00	00	00
0x00000314	00	00	00	00
0x00000310	00	00	00	00
0x0000030c	00	00	00	00
0x00000308	00	00	00	00
0x00000304	00	00	00	00
0x00000300	23	e1	cd	ab
0x000002fc	00	00	00	00
0x000002f8	00	00	00	00
0x000002f4	00	00	00	00
0x000002f0	00	00	00	00
0x000002ec	00	00	00	00
0x000002e8	00	00	00	00

Jump to -- choose -- Up Down

Display Settings Hex

### First word small-endian

Run Step Prev Reset Dump

0x00000313	addi x6 x0 0	addi t1, zero, 0
0x02000393	addi x7 x0 32	addi t2, zero, 32
0x0040006f	jai x0 4	j load
0x3082a023	sw x8 768(x5)	sw x0, 0x300(t0) # memory[300+t0] = 0xab0de123
0x30028503	lb x10 768(x5)	lb a0, 0x300(t0) # a0 = memory[300+t0]
0x30128583	lb x11 769(x5)	lb a1, 0x301(t0) # a1 = memory[301+t0]
0x30228603	lb x12 770(x5)	lb a2, 0x302(t0) # a2 = memory[302+t0]
0x30328663	lb x13 771(x5)	lb a3, 0x303(t0) # a3 = memory[303+t0]
0x00428293	addi x5 x5 4	addi t0, t0, 4
0x0040006f	jai x0 4	j swap
0x30d30023	sb x13 768(x6)	sb a3, 0x300(t1) # a3 = memory[300+t1]
0x30c300a3	sb x12 769(x6)	sb a2, 0x301(t1) # a2 = memory[301+t1]
0x30b30123	sb x11 770(x6)	sb a1, 0x302(t1) # a1 = memory[302+t1]
0x30a301a3	sb x10 771(x6)	sb a0, 0x303(t1) # a0 = memory[303+t1]
0x00400313	addi x6 x6 4	addi t1, t1, 4

console output

Registers Memory

Address	+0	+1	+2	+3
0x00000318	00	00	00	00
0x00000314	00	00	00	00
0x00000310	00	00	00	00
0x0000030c	00	00	00	00
0x00000308	00	00	00	00
0x00000304	00	00	00	00
0x00000300	ab	cd	e1	23
0x000002fc	00	00	00	00
0x000002f8	00	00	00	00
0x000002f4	00	00	00	00
0x000002f0	00	00	00	00
0x000002ec	00	00	00	00
0x000002e8	00	00	00	00

Jump to -- choose -- Up Down

Display Settings Hex

## Finish

Run
Step
Prev
Reset
Dump

```

0x00000313      addi x6 x0 0              addi t1, zero, 0
0x02000393      addi x7 x0 32          addi t2, zero, 32
0x0400006f      jal x0 4              j load
0x3082a023      sw x8 768(x5)         sw a0, 0x300(t0) # memory[300+t0] = 0xABCDEF123
0x30028503      lb x10 768(x5)        lb a0, 0x300(t0) # a0 = memory[300+t0]
0x30128583      lb x11 769(x5)        lb a1, 0x301(t0) # a1 = memory[301+t0]
0x30228603      lb x12 770(x5)        lb a2, 0x302(t0) # a2 = memory[302+t0]
0x30328683      lb x13 771(x5)        lb a3, 0x303(t0) # a3 = memory[303+t0]
0x00428293      addi x5 x5 4          addi t0, t0, 4
0x0040006f      jal x0 4              j swap
0x30300023      sb x13 768(x6)        sb a3, 0x300(t1) # a3 = memory[300+t1]
0x30300043      sb x12 769(x6)        sb a2, 0x301(t1) # a2 = memory[301+t1]
0x30b30123      sb x11 770(x6)        sb a1, 0x302(t1) # a1 = memory[302+t1]
0x30a301a3      sb x10 771(x6)        sb a0, 0x303(t1) # a0 = memory[303+t1]
0x00430313      addi x6 x6 4          addi t1, t1, 4
        
```

Registers
Memory

Address	+0	+1	+2	+3
0x00000330	00	00	00	00
0x0000032c	00	00	00	00
0x00000328	00	00	00	00
0x00000324	00	00	00	00
0x00000320	00	00	00	00
0x0000031c	ab	cd	e1	23
0x00000318	ab	cd	e1	23
0x00000314	ab	cd	e1	23
0x00000310	ab	cd	e1	23
0x0000030c	ab	cd	e1	23
0x00000308	ab	cd	e1	23
0x00000304	ab	cd	e1	23
0x00000300	ab	cd	e1	23

Jump to
-- choose --
Up
Down

Display Settings
Hex

console output

# Bubblesort

## Code:

```
1 addi s0, zero, 89
2 addi s1, zero, 63
3 addi s2, zero, -55
4 addi s3, zero, -187
5 addi s4, zero, 42
6 addi s5, zero, 98
7 addi s6, zero, -425
8 addi s7, zero, 283
9 addi s8, zero, 0
10 addi s9, zero, 303
11
12 main:
13 jal arrayinit
14
15 arrayinit: #initialize array
16 sw s0, 0x400(zero)
17 sw s1, 0x404(zero)
18 sw s2, 0x408(zero)
19 sw s3, 0x40c(zero)
20 sw s4, 0x410(zero)
21 sw s5, 0x414(zero)
22 sw s6, 0x418(zero)
23 sw s7, 0x41c(zero)
24 sw s8, 0x420(zero)
25 sw s9, 0x424(zero)
26
27 addi s0, zero, 0 #counter for len(array)
28 addi s1, zero, 0 #counter for element in array
29 addi s2, zero, 36
30 addi s3, zero, 10
31 j sortcount
32
33 sortcount:
34 addi s1, zero, 0
35 addi s0, s0, 1
36 beq s0, s3, finish # if the sort has gone 9 times
37 j bubblesort # start bubblesort
38
```

```
21 sw s5, 0x414(zero)
22 sw s6, 0x418(zero)
23 sw s7, 0x41c(zero)
24 sw s8, 0x420(zero)
25 sw s9, 0x424(zero)
26
27 addi s0, zero, 0 #counter for len(array)
28 addi s1, zero, 0 #counter for element in array
29 addi s2, zero, 36
30 addi s3, zero, 10
31 j sortcount
32
33 sortcount:
34 addi s1, zero, 0
35 addi s0, s0, 1
36 beq s0, s3, finish # if the sort has gone 9 times
37 j bubblesort # start bubblesort
38
39 bubblesort:
40 lw t0, 0x400(s1)
41 lw t1, 0x404(s1)
42 bge t0, t1, swap # if front element is larger than behind, swap these two
43 addi s1, s1, 4 # go to next word
44 beq s1, s2, sortcount # if 10 words are go through
45 j bubblesort # bubble sort again
46
47 afterswap:
48 addi s1, s1, 4 # go to next word
49 beq s1, s2, sortcount # if 10 words are go through
50 j bubblesort
51
52 swap: # swap front and behind element
53 sw t1, 0x400(s1)
54 sw t0, 0x404(s1)
55 j afterswap
56
57 finish:
58
```

Simulation:

Before bubble sort:

Run Step Prev Reset Dump

0x0c00b93	addi x23, x0, 203	addi a7, zero, 203
0x00000c13	addi x24, x0, 0	addi s6, zero, 0
0x12f00c93	addi x25, x0, 303	addi s9, zero, 303
0x004000ef	jal x1, 4	j al arrayinit
0x40802023	sw x8, 1024(x0)	sw s0, 0x400(zero)
0x40902223	sw x9, 1028(x0)	sw s1, 0x404(zero)
0x41202423	sw x18, 1032(x0)	sw s2, 0x408(zero)
0x41302623	sw x19, 1036(x0)	sw s3, 0x40c(zero)
0x41402823	sw x20, 1040(x0)	sw s4, 0x410(zero)
0x41502a23	sw x21, 1044(x0)	sw s5, 0x414(zero)
0x41602c23	sw x22, 1048(x0)	sw s6, 0x418(zero)
0x41702e23	sw x23, 1052(x0)	sw s7, 0x41c(zero)
0x43802023	sw x24, 1056(x0)	sw s8, 0x420(zero)
0x43902223	sw x25, 1060(x0)	sw s9, 0x424(zero)
0x00000413	addi s0, x0, 0	addi s0, zero, 0 #counter for len(array)

console output

Registers Memory

Address	+0	+1	+2	+3
0x00000430	00	00	00	00
0x0000042c	00	00	00	00
0x00000428	00	00	00	00
0x00000424	2f	01	00	00
0x00000420	00	00	00	00
0x0000041c	cb	00	00	00
0x00000418	57	fe	ff	ff
0x00000414	62	00	00	00
0x00000410	2a	00	00	00
0x0000040c	95	ff	ff	ff
0x00000408	c9	ff	ff	ff
0x00000404	3f	00	00	00
0x00000400	59	00	00	00

Jump to -- choose -- Up Down

Display Settings Hex

After first bubble sort:

Run Step Prev Reset Dump

0x00000993	addi x19, x0, 10	addi s3, zero, 10
0x004000ef	j al x0, 4	j sortcount
0x00000493	addi s9, x0, 0	addi s1, zero, 0
0x00140413	addi s8, x0, 1	addi s0, s0, 1
0x03340c63	beq s0, s3, finish # if the sort has gone 9 times	
0x004000ef	j al x0, 4	j bubblesort # start bubblesort
0x4004a293	lw s5, 1024(x9)	lw t0, 0x400(s1)
0x4044a303	lw s6, 1028(x9)	lw t1, 0x404(s1)
0x0042de63	bge s5, s6, 28	bge t0, t1, swap # if front element is larger than behind, swap these two
0x00448493	addi s9, s9, 4	addi s1, s1, 4 # go to next word
0xff2480e3	beq s9, s18, -32	beq s1, s2, sortcount # if 10 words are go through
0xfedff06f	j al x0, -20	j bubblesort # bubble sort again
0x00448493	addi s9, s9, 4	addi s1, s1, 4 # go to next word
0xf024f0e3	beq s9, s18, -44	beq s1, s2, sortcount # if 10 words are go through

console output

Registers Memory

Address	+0	+1	+2	+3
0x00000430	00	00	00	00
0x0000042c	00	00	00	00
0x00000428	00	00	00	00
0x00000424	2f	01	00	00
0x00000420	00	00	00	00
0x0000041c	cb	00	00	00
0x00000418	57	fe	ff	ff
0x00000414	62	00	00	00
0x00000410	2a	00	00	00
0x0000040c	95	ff	ff	ff
0x00000408	c9	ff	ff	ff
0x00000404	59	00	00	00
0x00000400	3f	00	00	00

Jump to -- choose -- Up Down

Display Settings Hex

Finish:



