

ID: ITC SIU22056

The image displays the QtSPIM MIPS simulator interface, which is divided into several panes. The top-left pane shows the assembly code being executed, with instructions like `li $v0, 0`, `beq $a1, $0, EXIT`, and `addi $a1, $a1, -1`. The top-right pane shows the register file, with `$v0` containing the value `0`. The bottom-left pane shows the memory dump, which is currently empty. The bottom-right pane shows the console output, which includes the text `Phan Tran Thanh Huy` and `Enter the range: 5`.

The assembly code in the top-left pane is as follows:

```

li $v0, 0
beq $a1, $0, EXIT
lw $t0, 0($a0)
move $s0, $t0
addi $a0, $a0, 4
addi $a1, $a1, -1
jal SUM
add $v0, $v0, $s0

EXIT:
lw $ra, 0($sp)
lw $s0, 4($sp)
addi $sp, $sp, 8
jr $ra

```

The register file in the top-right pane shows the following values:

Register	Value
\$v0	0
\$a1	0
\$a0	0
\$s0	0
\$t0	0
\$t1	0
\$t2	0
\$t3	0
\$t4	0
\$t5	0
\$t6	0
\$t7	0
\$a2	0
\$a3	0
\$a4	0
\$a5	0

The console output in the bottom-right pane shows the following text:

```

Phan Tran Thanh Huy
Enter the range: 5
Enter the array: 2
0
5
4
Your array is: 2 2 0 5 6
The sum all elements of the array: 15

```

Code:

```
1 .data
2 nameId: .asciiz "Phan Tran Thanh Huy\nITCSI022056"
3 inputRange: .asciiz "Enter the range: "
4 inputArray: .asciiz "Enter the arrays: "
5 outputArray: .asciiz "Your array is: "
6 stringSpace: .asciiz " "
7 stringLine: .asciiz "\n"
8 printResult: .asciiz "The sum all elements of the array: "
9 n: .word 0
10 arr: .space 100
11
12 .text
13 .globl main
14
15 main:
16     la $a0, nameId      #Print the information of student
17     li $v0, 4
18     syscall
19
20     la $a0, stringLine
21     li $v0, 4
22     syscall
23
24     la $a0, inputRange
25     li $v0, 4
26     syscall
27
28     li $v0, 5           #Enter the range of array
29     syscall
30     sw $v0, n           #Save v0 to n
31
32     la $a0, inputArray
33     li $v0, 4
34     syscall
35
36     la $t0, arr         #load the array to t0
37     lw $s0, n           #load n to s0
38
39 INPUT:                 #Enter the array
40     li $v0, 5
41     syscall
42     sw $v0, 0($t0)      #save v0 to t0
43     addi $t0, $t0, 4    #move to the next element of the array
44     addi $s0, $s0, -1   #decrease counter
45     bne $s0, $0, INPUT  #return if s0 = 0
46
47     la $a0, outputArray
48     li $v0, 4
49     syscall
50
51     la $t0, arr         #load the array to t0
52     lw $s0, n           #load n to s0
53
54 OUTPUT:               #Print the array
55     lw $t1, 0($t0)      #Load t0 to t1
56     move $a0, $t1       #Print the element of the array
57     li $v0, 1
58     syscall
59     la $a0, stringSpace
60     li $v0, 4
61     syscall
62     addi $t0, $t0, 4    #move to the next element of the array
63     addi $s0, $s0, -1   #decrease counter
64     bne $s0, $0, OUTPUT #return if s0 = 0
65
66     la $a0, stringLine
67     li $v0, 4
68     syscall
69
70     la $a0, printResult
71     li $v0, 4
72     syscall
73
74     la $a0, arr         #load the array to t0
75     lw $a1, n           #load n to a1
76     jal SUM             #Call the recursive
77
78     move $a0, $v0        #move v0 to a0
79     li $v0, 1           #Print the result
80     syscall
81
82     li $v0, 10
83     syscall
84
85 SUM:                  #The recursive function
86     addi $sp, $sp, -8
87     sw $ra, 0($sp)
88     sw $s0, 4($sp)
89
90     li $v0, 0           #load 0 to v0. Using v0 as the sum
91     beq $a1, $0, EXIT   #to the EXIT if a1 = 0
92     lw $t0, 0($a0)      #load the element of the array to t0
93     move $s0, $t0       #move t0 to s0
94     addi $a0, $a0, 4    #move to the next element of the array
95     addi $a1, $a1, -1   #Decrease counter
96     jal SUM
97     add $v0, $v0, $s0   #Compute the sum by adding s0 to v0
98
99 EXIT:
100     lw $ra, 0($sp)
101     lw $s0, 4($sp)
102     addi $sp, $sp, 8
103     jr $ra
```

Result:

```
Phan Tran Thanh Huy
ITCSIU22056
Enter the range: 5
Enter the array: 2
2
0
5
6
Your array is: 2 2 0 5 6
The sum all elements of the array: 15|
```

Explanation of the address:

Before calling the procedure SUM, the initial return address is stored in memory address [00400018] and the initial stack pointer address is stored in memory address [7ffff19c].

Int Regs [16]	#	x	Text
R2 [v0] = 4			[004000b8] 8d090000 lw \$9, 0(\$8) ; 55: lw \$t1, 0(\$t0) #Load t0 to t1
R3 [v1] = 0			[004000bc] 00092021 addu \$4, \$0, \$9 ; 56: move \$a0, \$t1 #Print the element of the array
R4 [a0] = 10010080			[004000c0] 34020001 ori \$2, \$0, 1 ; 57: li \$v0, 1
R5 [a1] = 7ffff1a0			[004000c4] 0000000c syscall ; 58: syscall
R6 [a2] = 7ffff1a8			[004000c8] 3c011001 lui \$1, 4097 [stringSpace]; 59: la \$a0, stringSpace
R7 [a3] = 0			[004000cc] 34240054 ori \$4, \$1, 84 [stringSpace]
R8 [t0] = 10010094			[004000d0] 34020004 ori \$2, \$0, 4 ; 60: li \$v0, 4
R9 [t1] = 6			[004000d4] 0000000c syscall ; 61: syscall
R10 [t2] = 0			[004000d8] 21080004 addi \$8, \$8, 4 ; 62: addi \$t0, \$t0, 4 #move to the next element of the array
R11 [t3] = 0			[004000dc] 2210ffff addi \$16, \$16, -1 ; 63: addi \$s0, \$s0, -1 #decrease counter
R12 [t4] = 0			[004000e0] 1600ffff bne \$16, \$0, -40 [OUTPUT-0x004000e0]
R13 [t5] = 0			[004000e4] 3c011001 lui \$1, 4097 [stringLine]; 66: la \$a0, stringLine
R14 [t6] = 0			[004000e8] 34240056 ori \$4, \$1, 86 [stringLine]
R15 [t7] = 0			[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
R16 [s0] = 0			[004000f0] 0000000c syscall ; 68: syscall
R17 [s1] = 0			[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R18 [s2] = 0			[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R19 [s3] = 0			[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R20 [s4] = 0			[00400100] 0000000c syscall ; 72: syscall
R21 [s5] = 0			[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R22 [s6] = 0			[00400108] 34240080 ori \$4, \$1, 128 [arr]
R23 [s7] = 0			[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R24 [t8] = 0			[00400110] 8c25007c lw \$5, 124(\$1)
R25 [t9] = 0			[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R26 [k0] = 0			[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R27 [k1] = 0			[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R28 [gp] = 10008000			[00400120] 0000000c syscall ; 80: syscall
R29 [sp] = 7ffff19c			[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R30 [s8] = 0			[00400128] 0000000c syscall ; 83: syscall
R31 [ra] = 400018			[0040012c] 23bdfbf8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
			[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
			[00400134] 3c011001 lui \$1, 4097 ; 88: sw \$a0, 4(\$sp)
			[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
			[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
			[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
			[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
			[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
			[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
			[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
			[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
			[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
			[0040015c] 8fbf0004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
			[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
			[00400164] 03e00008 jr \$31 ; 103: jr \$ra
			Kernel Text Segment [80000000]..[80010000]
			[80000180] 0001d821 addu \$27, \$0, \$1 ; 90: move \$k1 \$at # Save \$at
			[80000184] 3c011001 lui \$1, 4097 ; 92: li \$v0, 1 #Print the result

After calling the procedure SUM, the return address is stored in memory address [00400118], which is the return address of the callee to the main function.

Int Regs [16]	#	x	Text
R2 [v0] = 4			[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R3 [v1] = 0			[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R4 [a0] = 10010080			[00400100] 0000000c syscall ; 72: syscall
R5 [a1] = 5			[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R6 [a2] = 7ffff1a8			[00400108] 34240080 ori \$4, \$1, 128 [arr]
R7 [a3] = 0			[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R8 [t0] = 10010094			[00400110] 8c25007c lw \$5, 124(\$1)
R9 [t1] = 6			[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R10 [t2] = 0			[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R11 [t3] = 0			[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R12 [t4] = 0			[00400120] 0000000c syscall ; 80: syscall
R13 [t5] = 0			[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R14 [t6] = 0			[00400128] 0000000c syscall ; 83: syscall
R15 [t7] = 0			[0040012c] 23bdfbf8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R16 [s0] = 0			[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R17 [s1] = 0			[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$a0, 4(\$sp)
R18 [s2] = 0			[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R19 [s3] = 0			[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R20 [s4] = 0			[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R21 [s5] = 0			[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R22 [s6] = 0			[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R23 [s7] = 0			[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R24 [t8] = 0			[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R25 [t9] = 0			[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R26 [k0] = 0			[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R27 [k1] = 0			[0040015c] 8fbf0004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R28 [gp] = 10008000			[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
R29 [sp] = 7ffff19c			[00400164] 03e00008 jr \$31 ; 103: jr \$ra
R30 [s8] = 0			
R31 [ra] = 400118			
			Kernel Text Segment [80000000]..[80010000]
			[80000180] 0001d821 addu \$27, \$0, \$1 ; 90: move \$k1 \$at # Save \$at
			[80000184] 3c011001 lui \$1, 4097 ; 92: li \$v0, 1 #Print the result

And the stack pointer is stored in memory address [7ffff194]

Int Regs [16]	Text
R2 [v0] = 4	[004000b8] 8d090000 lw \$9, 0(\$8) ; 55: lw \$t1, 0(\$t0) #Load t0 to t1
R3 [v1] = 0	[004000bc] 00092021 addu \$4, \$0, \$9 ; 56: move \$a0, \$t1 #Print the element of the array
R4 [a0] = 10010080	[004000c0] 34020001 ori \$2, \$0, 1 ; 57: li \$v0, 1
R5 [a1] = 5	[004000c4] 0000000c syscall ; 58: syscall
R6 [a2] = 7ffff1a8	[004000c8] 3c011001 lui \$1, 4097 [stringSpace]; 59: la \$a0, stringSpace
R7 [a3] = 0	[004000cc] 34240054 ori \$4, \$1, 84 [stringSpace]
R8 [t0] = 10010094	[004000d0] 34020004 ori \$2, \$0, 4 ; 60: li \$v0, 4
R9 [t1] = 6	[004000d4] 0000000c syscall ; 61: syscall
R10 [t2] = 0	[004000d8] 21080004 addi \$8, \$8, 4 ; 62: addi \$t0, \$t0, 4 #move to the next element of the array
R11 [t3] = 0	[004000dc] 2210ffff addi \$16, \$16, -1 ; 63: addi \$s0, \$s0, -1 #decrease counter
R12 [t4] = 0	[004000e0] 1600ffff bne \$16, \$0, -40 [OUTPUT-0x004000e0]
R13 [t5] = 0	[004000e4] 3c011001 lui \$1, 4097 [stringLine]; 66: la \$a0, stringLine
R14 [t6] = 0	[004000e8] 34240056 ori \$4, \$1, 86 [stringLine]
R15 [t7] = 0	[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
R16 [s0] = 0	[004000f0] 0000000c syscall ; 68: syscall
R17 [s1] = 0	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R18 [s2] = 0	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R19 [s3] = 0	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R20 [s4] = 0	[00400100] 0000000c syscall ; 72: syscall
R21 [s5] = 0	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R22 [s6] = 0	[00400108] 34240080 ori \$4, \$1, 128 [arr]
R23 [s7] = 0	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R24 [s8] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R25 [t9] = 0	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R26 [k0] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R27 [k1] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R28 [gp] = 10008000	[00400120] 0000000c syscall ; 80: syscall
R29 [sp] = 7ffff194	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R30 [s8] = 0	[00400128] 0000000c syscall ; 83: syscall
R31 [ra] = 400118	[0040012c] 23bdffff addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)

In the first save, the stack pointer is stored in memory address [7ffff18c] and the return address is stored in memory address [00400154], which is the return address of the add operation.

Int Regs [16]	Text
R2 [v0] = 0	[004000f0] 0000000c syscall ; 68: syscall
R3 [v1] = 0	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R4 [a0] = 10010084	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R5 [a1] = 4	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R6 [a2] = 7ffff1a8	[00400100] 0000000c syscall ; 72: syscall
R7 [a3] = 0	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R8 [t0] = 2	[00400108] 34240080 ori \$4, \$1, 128 [arr]
R9 [t1] = 6	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R10 [t2] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R11 [t3] = 0	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R12 [t4] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R13 [t5] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R14 [t6] = 0	[00400120] 0000000c syscall ; 80: syscall
R15 [t7] = 0	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R16 [s0] = 2	[00400128] 0000000c syscall ; 83: syscall
R17 [s1] = 0	[0040012c] 23bdffff addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R18 [s2] = 0	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R19 [s3] = 0	[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R20 [s4] = 0	[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R21 [s5] = 0	[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R22 [s6] = 0	[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R23 [s7] = 0	[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R24 [t8] = 0	[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R25 [t9] = 0	[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R26 [k0] = 0	[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R27 [k1] = 0	[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R28 [gp] = 10008000	[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R29 [sp] = 7ffff18c	[0040015c] 8fbf0004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R30 [s8] = 0	[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
R31 [ra] = 400154	[00400164] 03e00008 jr \$31 ; 103: jr \$ra

Kernel Text Segment: [80000000]..[80010000]

In the second save, the stack pointer is stored in memory address [7ffff184]

Int Regs [16]	#	X	Text
R2 [v0] = 0			[004000f0] 0000000c syscall ; 68: syscall
R3 [v1] = 0			[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R4 [a0] = 10010088			[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R5 [a1] = 3			[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R6 [a2] = 7ffff1a8			[00400100] 0000000c syscall ; 72: syscall
R7 [a3] = 0			[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R8 [t0] = 2			[00400108] 34240080 ori \$4, \$1, 128 [arr]
R9 [t1] = 6			[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R10 [t2] = 0			[00400110] 8c25007c lw \$5, 124(\$1)
R11 [t3] = 0			[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R12 [t4] = 0			[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R13 [t5] = 0			[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R14 [t6] = 0			[00400120] 0000000c syscall ; 80: syscall
R15 [t7] = 0			[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R16 [s0] = 2			[00400128] 0000000c syscall ; 83: syscall
R17 [s1] = 0			[0040012c] 23bdffff addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R18 [s2] = 0			[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R19 [s3] = 0			[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R20 [s4] = 0			[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R21 [s5] = 0			[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R22 [s6] = 0			[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R23 [s7] = 0			[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R24 [t8] = 0			[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R25 [t9] = 0			[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R26 [k0] = 0			[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R27 [k1] = 0			[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R28 [gp] = 10008000			[00400158] 8bf00000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R29 [sp] = 7ffff184			[0040015c] 8bf00004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R30 [s8] = 0			[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
R31 [ra] = 400154			[00400164] 03e00008 jr \$31 ; 103: jr \$ra
Kernel Text Segment [800000001..f800100001]			

In the third save, the stack pointer is stored in memory address [7ffff17c]

Int Regs [16]	#	X	Text
R2 [v0] = 0			[004000f0] 0000000c syscall ; 68: syscall
R3 [v1] = 0			[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R4 [a0] = 1001008c			[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R5 [a1] = 2			[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R6 [a2] = 7ffff1a8			[00400100] 0000000c syscall ; 72: syscall
R7 [a3] = 0			[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R8 [t0] = 0			[00400108] 34240080 ori \$4, \$1, 128 [arr]
R9 [t1] = 6			[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R10 [t2] = 0			[00400110] 8c25007c lw \$5, 124(\$1)
R11 [t3] = 0			[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R12 [t4] = 0			[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R13 [t5] = 0			[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R14 [t6] = 0			[00400120] 0000000c syscall ; 80: syscall
R15 [t7] = 0			[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R16 [s0] = 0			[00400128] 0000000c syscall ; 83: syscall
R17 [s1] = 0			[0040012c] 23bdffff addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R18 [s2] = 0			[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R19 [s3] = 0			[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R20 [s4] = 0			[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R21 [s5] = 0			[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R22 [s6] = 0			[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R23 [s7] = 0			[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R24 [t8] = 0			[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R25 [t9] = 0			[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R26 [k0] = 0			[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R27 [k1] = 0			[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R28 [gp] = 10008000			[00400158] 8bf00000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R29 [sp] = 7ffff17c			[0040015c] 8bf00004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R30 [s8] = 0			[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
R31 [ra] = 400154			[00400164] 03e00008 jr \$31 ; 103: jr \$ra
Kernel Text Segment [800000001..f800100001]			

In the fourth save, the stack pointer is stored in memory address [7ffff174]

Int Regs [16]	Text
R2 [v0] = 0	[004000f0] 0000000c syscall ; 68: syscall
R3 [v1] = 0	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R4 [a0] = 10010090	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R5 [a1] = 1	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R6 [a2] = 7ffff1a8	[00400100] 0000000c syscall ; 72: syscall
R7 [a3] = 0	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R8 [t0] = 5	[00400108] 34240080 ori \$4, \$1, 128 [arr]
R9 [t1] = 6	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R10 [t2] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R11 [t3] = 0	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R12 [t4] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R13 [t5] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R14 [t6] = 0	[00400120] 0000000c syscall ; 80: syscall
R15 [t7] = 0	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R16 [s0] = 5	[00400128] 0000000c syscall ; 83: syscall
R17 [s1] = 0	[0040012c] 23bdffff addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R18 [s2] = 0	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R19 [s3] = 0	[00400134] afbf0000 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R20 [s4] = 0	[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R21 [s5] = 0	[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R22 [s6] = 0	[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R23 [s7] = 0	[00400144] 00080021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R24 [t8] = 0	[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R25 [t9] = 0	[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R26 [k0] = 0	[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R27 [k1] = 0	[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R28 [gp] = 10008000	[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R29 [sp] = 7ffff174	[0040015c] 8fbf0000 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R30 [s8] = 0	[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
R31 [ra] = 400154	[00400164] 03e00008 jr \$31 ; 103: jr \$ra

In the fifth save, the stack pointer is stored in memory address [7ffff16c]

Int Regs [16]	Text
R2 [v0] = 0	[004000f0] 0000000c syscall ; 68: syscall
R3 [v1] = 0	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R4 [a0] = 10010094	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R5 [a1] = 0	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R6 [a2] = 7ffff1a8	[00400100] 0000000c syscall ; 72: syscall
R7 [a3] = 0	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R8 [t0] = 6	[00400108] 34240080 ori \$4, \$1, 128 [arr]
R9 [t1] = 6	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R10 [t2] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R11 [t3] = 0	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R12 [t4] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R13 [t5] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R14 [t6] = 0	[00400120] 0000000c syscall ; 80: syscall
R15 [t7] = 0	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R16 [s0] = 6	[00400128] 0000000c syscall ; 83: syscall
R17 [s1] = 0	[0040012c] 23bdffff addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R18 [s2] = 0	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R19 [s3] = 0	[00400134] afbf0000 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R20 [s4] = 0	[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R21 [s5] = 0	[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R22 [s6] = 0	[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R23 [s7] = 0	[00400144] 00080021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R24 [t8] = 0	[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R25 [t9] = 0	[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R26 [k0] = 0	[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R27 [k1] = 0	[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R28 [gp] = 10008000	[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R29 [sp] = 7ffff16c	[0040015c] 8fbf0000 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R30 [s8] = 0	[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
R31 [ra] = 400154	[00400164] 03e00008 jr \$31 ; 103: jr \$ra

Now using all the value contain in the stack. The stack pointer is stored back to memory address [7ffff174]

Int Regs [16]	Text
R2 [v0] = 0	[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
R3 [v1] = 0	[004000f0] 0000000c syscall ; 68: syscall
R4 [a0] = 10010094	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R5 [a1] = 0	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R6 [a2] = 7ffff1a8	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R7 [a3] = 0	[00400100] 0000000c syscall ; 72: syscall
R8 [t0] = 6	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R9 [t1] = 6	[00400108] 34240080 ori \$4, \$1, 128 [arr]
R10 [t2] = 0	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R11 [t3] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R12 [t4] = 0	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R13 [t5] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R14 [t6] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R15 [t7] = 0	[00400120] 0000000c syscall ; 80: syscall
R16 [s0] = 6	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R17 [s1] = 0	[00400128] 0000000c syscall ; 83: syscall
R18 [s2] = 0	[0040012c] 23bdfff8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R19 [s3] = 0	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R20 [s4] = 0	[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R21 [s5] = 0	[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R22 [s6] = 0	[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R23 [s7] = 0	[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R24 [t8] = 0	[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R25 [t9] = 0	[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R26 [k0] = 0	[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R27 [k1] = 0	[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R28 [gp] = 10008000	[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R29 [sp] = 7ffff174	[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R30 [s8] = 0	[0040015c] 8fbf0004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R31 [ra] = 400154	[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
	[00400164] 03e00008 jr \$31 ; 103: jr \$ra

After the add operation, the stack pointer is stored back to memory address [7ffff17c]

Int Regs [16]	Text
R2 [v0] = 6	[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
R3 [v1] = 0	[004000f0] 0000000c syscall ; 68: syscall
R4 [a0] = 10010094	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R5 [a1] = 0	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R6 [a2] = 7ffff1a8	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R7 [a3] = 0	[00400100] 0000000c syscall ; 72: syscall
R8 [t0] = 6	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R9 [t1] = 6	[00400108] 34240080 ori \$4, \$1, 128 [arr]
R10 [t2] = 0	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R11 [t3] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R12 [t4] = 0	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R13 [t5] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R14 [t6] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R15 [t7] = 0	[00400120] 0000000c syscall ; 80: syscall
R16 [s0] = 5	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R17 [s1] = 0	[00400128] 0000000c syscall ; 83: syscall
R18 [s2] = 0	[0040012c] 23bdfff8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R19 [s3] = 0	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R20 [s4] = 0	[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R21 [s5] = 0	[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R22 [s6] = 0	[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R23 [s7] = 0	[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R24 [t8] = 0	[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R25 [t9] = 0	[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R26 [k0] = 0	[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R27 [k1] = 0	[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R28 [gp] = 10008000	[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R29 [sp] = 7ffff17c	[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R30 [s8] = 0	[0040015c] 8fbf0004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R31 [ra] = 400154	[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
	[00400164] 03e00008 jr \$31 ; 103: jr \$ra

After the add operation, the stack pointer is stored back to memory address [7ffff184]

Int Regs [16]	#	×	Text
R2 [v0] = b			[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
R3 [v1] = 0			[004000f0] 0000000c syscall ; 68: syscall
R4 [a0] = 10010094			[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R5 [a1] = 0			[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R6 [a2] = 7ffff1a8			[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R7 [a3] = 0			[00400100] 0000000c syscall ; 72: syscall
R8 [t0] = 6			[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R9 [t1] = 6			[00400108] 34240080 ori \$4, \$1, 128 [arr]
R10 [t2] = 0			[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R11 [t3] = 0			[00400110] 8c25007c lw \$5, 124(\$1)
R12 [t4] = 0			[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R13 [t5] = 0			[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R14 [t6] = 0			[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R15 [t7] = 0			[00400120] 0000000c syscall ; 80: syscall
R16 [s0] = 0			[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R17 [s1] = 0			[00400128] 0000000c syscall ; 83: syscall
R18 [s2] = 0			[0040012c] 23bdfff8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R19 [s3] = 0			[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R20 [s4] = 0			[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R21 [s5] = 0			[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R22 [s6] = 0			[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R23 [s7] = 0			[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R24 [t8] = 0			[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R25 [t9] = 0			[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R26 [k0] = 0			[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R27 [k1] = 0			[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R28 [gp] = 10008000			[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R29 [sp] = 7ffff184			[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R30 [s8] = 0			[0040015c] 8fb00004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R31 [ra] = 400154			[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
			[00400164] 03e00008 jr \$31 ; 103: jr \$ra

After the add operation, the stack pointer is stored back to memory address [7ffff18c]

Int Regs [16]	#	×	Text
R2 [v0] = b			[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
R3 [v1] = 0			[004000f0] 0000000c syscall ; 68: syscall
R4 [a0] = 10010094			[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R5 [a1] = 0			[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R6 [a2] = 7ffff1a8			[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R7 [a3] = 0			[00400100] 0000000c syscall ; 72: syscall
R8 [t0] = 6			[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R9 [t1] = 6			[00400108] 34240080 ori \$4, \$1, 128 [arr]
R10 [t2] = 0			[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R11 [t3] = 0			[00400110] 8c25007c lw \$5, 124(\$1)
R12 [t4] = 0			[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R13 [t5] = 0			[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R14 [t6] = 0			[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R15 [t7] = 0			[00400120] 0000000c syscall ; 80: syscall
R16 [s0] = 2			[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R17 [s1] = 0			[00400128] 0000000c syscall ; 83: syscall
R18 [s2] = 0			[0040012c] 23bdfff8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R19 [s3] = 0			[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R20 [s4] = 0			[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R21 [s5] = 0			[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R22 [s6] = 0			[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R23 [s7] = 0			[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R24 [t8] = 0			[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R25 [t9] = 0			[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R26 [k0] = 0			[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R27 [k1] = 0			[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R28 [gp] = 10008000			[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R29 [sp] = 7ffff18c			[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R30 [s8] = 0			[0040015c] 8fb00004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R31 [ra] = 400154			[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
			[00400164] 03e00008 jr \$31 ; 103: jr \$ra

After the add operation, the stack pointer is stored back to memory address [7ffff194]

Int Regs [16]	Text
R2 [v0] = d	[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
R3 [v1] = 0	[004000f0] 0000000c syscall ; 68: syscall
R4 [a0] = 10010094	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R5 [a1] = 0	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R6 [a2] = 7ffff1a8	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R7 [a3] = 0	[00400100] 0000000c syscall ; 72: syscall
R8 [t0] = 6	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R9 [t1] = 6	[00400108] 34240080 ori \$4, \$1, 128 [arr]
R10 [t2] = 0	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R11 [t3] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R12 [t4] = 0	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R13 [t5] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R14 [t6] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R15 [t7] = 0	[00400120] 0000000c syscall ; 80: syscall
R16 [s0] = 2	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R17 [s1] = 0	[00400128] 0000000c syscall ; 83: syscall
R18 [s2] = 0	[0040012c] 23bdfff8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R19 [s3] = 0	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R20 [s4] = 0	[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R21 [s5] = 0	[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R22 [s6] = 0	[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R23 [s7] = 0	[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R24 [s8] = 0	[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R25 [t9] = 0	[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R26 [k0] = 0	[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R27 [k1] = 0	[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R28 [gp] = 10008000	[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R29 [sp] = 7ffff194	[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R30 [s8] = 0	[0040015c] 8fbf0004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R31 [ra] = 400154	[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
	[00400164] 03e00008 jr \$31 ; 103: jr \$ra

After the add operation, the stack pointer is stored back to memory address [7ffff19c] and the return address is stored back to memory address [00400118] as callee of the main function.

Int Regs [16]	Text
R2 [v0] = f	[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
R3 [v1] = 0	[004000f0] 0000000c syscall ; 68: syscall
R4 [a0] = 10010094	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R5 [a1] = 0	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R6 [a2] = 7ffff1a8	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R7 [a3] = 0	[00400100] 0000000c syscall ; 72: syscall
R8 [t0] = 6	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R9 [t1] = 6	[00400108] 34240080 ori \$4, \$1, 128 [arr]
R10 [t2] = 0	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R11 [t3] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R12 [t4] = 0	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R13 [t5] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R14 [t6] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R15 [t7] = 0	[00400120] 0000000c syscall ; 80: syscall
R16 [s0] = 2	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R17 [s1] = 0	[00400128] 0000000c syscall ; 83: syscall
R18 [s2] = 0	[0040012c] 23bdfff8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R19 [s3] = 0	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R20 [s4] = 0	[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R21 [s5] = 0	[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R22 [s6] = 0	[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R23 [s7] = 0	[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R24 [s8] = 0	[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R25 [t9] = 0	[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R26 [k0] = 0	[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R27 [k1] = 0	[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R28 [gp] = 10008000	[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R29 [sp] = 7ffff194	[00400158] 8fbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R30 [s8] = 0	[0040015c] 8fbf0004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R31 [ra] = 400154	[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
	[00400164] 03e00008 jr \$31 ; 103: jr \$ra

Int Regs [16]	Text
R2 [v0] = 0	[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
R3 [v1] = 0	[004000f0] 0000000c syscall ; 68: syscall
R4 [a0] = 10010094	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
R5 [a1] = 0	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
R6 [a2] = 7ffff1a8	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
R7 [a3] = 0	[00400100] 0000000c syscall ; 72: syscall
R8 [t0] = 6	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
R9 [t1] = 6	[00400108] 34240080 ori \$4, \$1, 128 [arr]
R10 [t2] = 0	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R11 [t3] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R12 [t4] = 0	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R13 [t5] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R14 [t6] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R15 [t7] = 0	[00400120] 0000000c syscall ; 80: syscall
R16 [s0] = 0	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R17 [s1] = 0	[00400128] 0000000c syscall ; 83: syscall
R18 [s2] = 0	[0040012c] 23bdfbf8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R19 [s3] = 0	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R20 [s4] = 0	[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R21 [s5] = 0	[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R22 [s6] = 0	[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R23 [s7] = 0	[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R24 [t8] = 0	[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R25 [t9] = 0	[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R26 [k0] = 0	[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R27 [k1] = 0	[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R28 [sp] = 10008000	[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R29 [sp] = 7ffff19c	[00400158] efbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R30 [s8] = 0	[0040015c] efbf0004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R31 [ra] = 400118	[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
	[00400164] 03e00008 jr \$31 ; 103: jr \$ra

The register v0 and a0 now contain the result

Int Regs [16]	Text
PC = 40011c	[004000ec] 34020004 ori \$2, \$0, 4 ; 67: li \$v0, 4
EPC = 0	[004000f0] 0000000c syscall ; 68: syscall
Cause = 0	[004000f4] 3c011001 lui \$1, 4097 [printResult]; 70: la \$a0, printResult
BadVAddr = 0	[004000f8] 34240058 ori \$4, \$1, 88 [printResult]
Status = 3000ff10	[004000fc] 34020004 ori \$2, \$0, 4 ; 71: li \$v0, 4
	[00400100] 0000000c syscall ; 72: syscall
HI = 0	[00400104] 3c011001 lui \$1, 4097 [arr] ; 74: la \$a0, arr #load the array to t0
LO = 0	[00400108] 34240080 ori \$4, \$1, 128 [arr]
	[0040010c] 3c011001 lui \$1, 4097 ; 75: lw \$a1, n #load n to a1
R0 [r0] = 0	[00400110] 8c25007c lw \$5, 124(\$1)
R1 [at] = 10010000	[00400114] 0c10004b jal 0x0040012c [SUM] ; 76: jal SUM #Call the recursive
R2 [v0] = 0	[00400118] 00022021 addu \$4, \$0, \$2 ; 78: move \$a0, \$v0 #move v0 to a0
R3 [v1] = 0	[0040011c] 34020001 ori \$2, \$0, 1 ; 79: li \$v0, 1 #Print the result
R4 [a0] = 0	[00400120] 0000000c syscall ; 80: syscall
R5 [a1] = 0	[00400124] 3402000a ori \$2, \$0, 10 ; 82: li \$v0, 10
R6 [a2] = 7ffff1a8	[00400128] 0000000c syscall ; 83: syscall
R7 [a3] = 0	[0040012c] 23bdfbf8 addi \$29, \$29, -8 ; 86: addi \$sp, \$sp, -8
R8 [t0] = 6	[00400130] afbf0000 sw \$31, 0(\$29) ; 87: sw \$ra, 0(\$sp)
R9 [t1] = 6	[00400134] afbf0004 sw \$16, 4(\$29) ; 88: sw \$s0, 4(\$sp)
R10 [t2] = 0	[00400138] 34020000 ori \$2, \$0, 0 ; 90: li \$v0, 0 #load 0 to v0. Using v0 as the sum
R11 [t3] = 0	[0040013c] 10a00007 beq \$5, \$0, 28 [EXIT-0x0040013c]
R12 [t4] = 0	[00400140] 8c880000 lw \$8, 0(\$4) ; 92: lw \$t0, 0(\$a0) #load the element of the array to t0
R13 [t5] = 0	[00400144] 00088021 addu \$16, \$0, \$8 ; 93: move \$s0, \$t0 #move t0 to s0
R14 [t6] = 0	[00400148] 20840004 addi \$4, \$4, 4 ; 94: addi \$a0, \$a0, 4 #move to the next element of the array
R15 [t7] = 0	[0040014c] 20a5ffff addi \$5, \$5, -1 ; 95: addi \$a1, \$a1, -1 #Decrease counter
R16 [s0] = 0	[00400150] 0c10004b jal 0x0040012c [SUM] ; 96: jal SUM
R17 [s1] = 0	[00400154] 00501020 add \$2, \$2, \$16 ; 97: add \$v0, \$v0, \$s0 #Compute the sum by adding s0 to v0
R18 [s2] = 0	[00400158] efbf0000 lw \$31, 0(\$29) ; 100: lw \$ra, 0(\$sp)
R19 [s3] = 0	[0040015c] efbf0004 lw \$16, 4(\$29) ; 101: lw \$s0, 4(\$sp)
R20 [s4] = 0	[00400160] 23bd0008 addi \$29, \$29, 8 ; 102: addi \$sp, \$sp, 8
R21 [s5] = 0	[00400164] 03e00008 jr \$31 ; 103: jr \$ra