Cairo University
Faculty of Computers and artificial intelligence

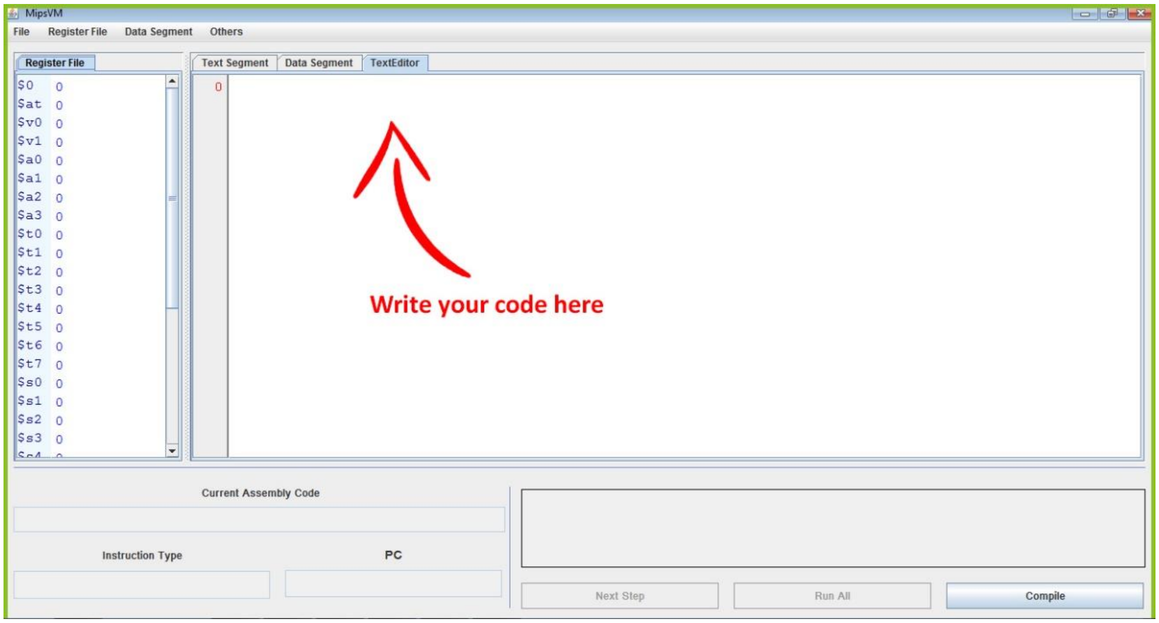
CS332
Computer Architecture and Organization

Assignment - 4 (Option 2)

Name	ID
Ehab Fawzy	20170072
Hatem Mamdooh	20170085
Hussien Ashraf	20170093
Khaled Ezzat	20170098

User Manual

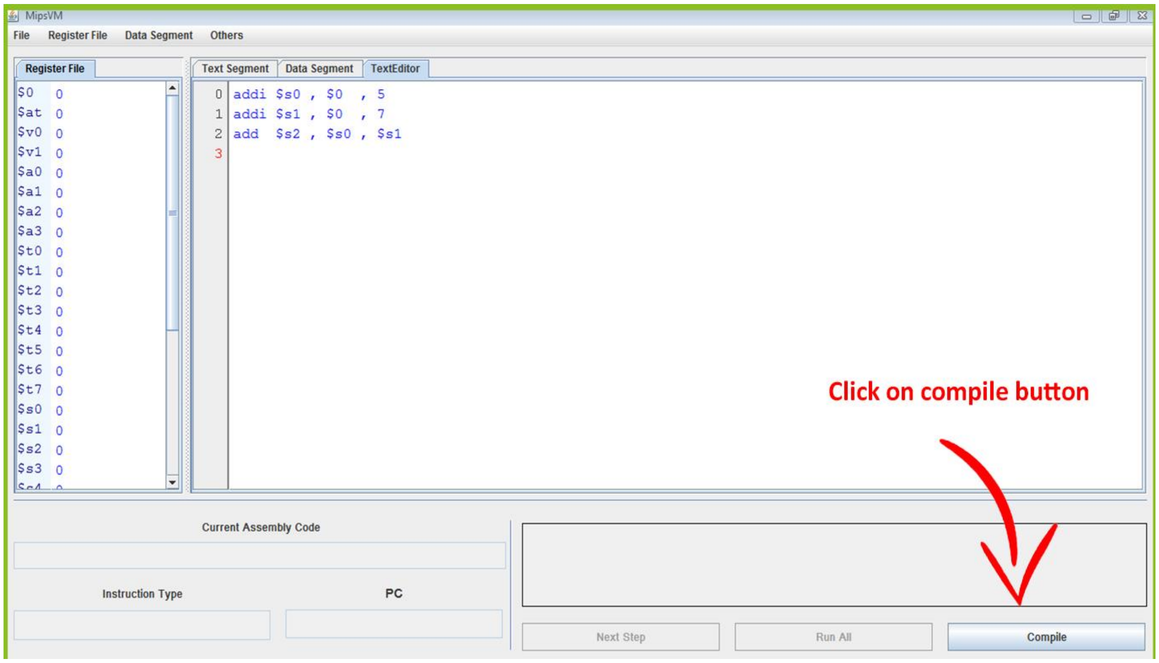
Step 1: Write MIPS Assembly code



$0	0
$at	0
$v0	0
$v1	0
$a0	0
$a1	0
$a2	0
$a3	0
$t0	0
$t1	0
$t2	0
$t3	0
$t4	0
$t5	0
$t6	0
$t7	0
$s0	0
$s1	0
$s2	0
$s3	0

Write your code here

Step 2: Compile to parse and make sure syntax is correct

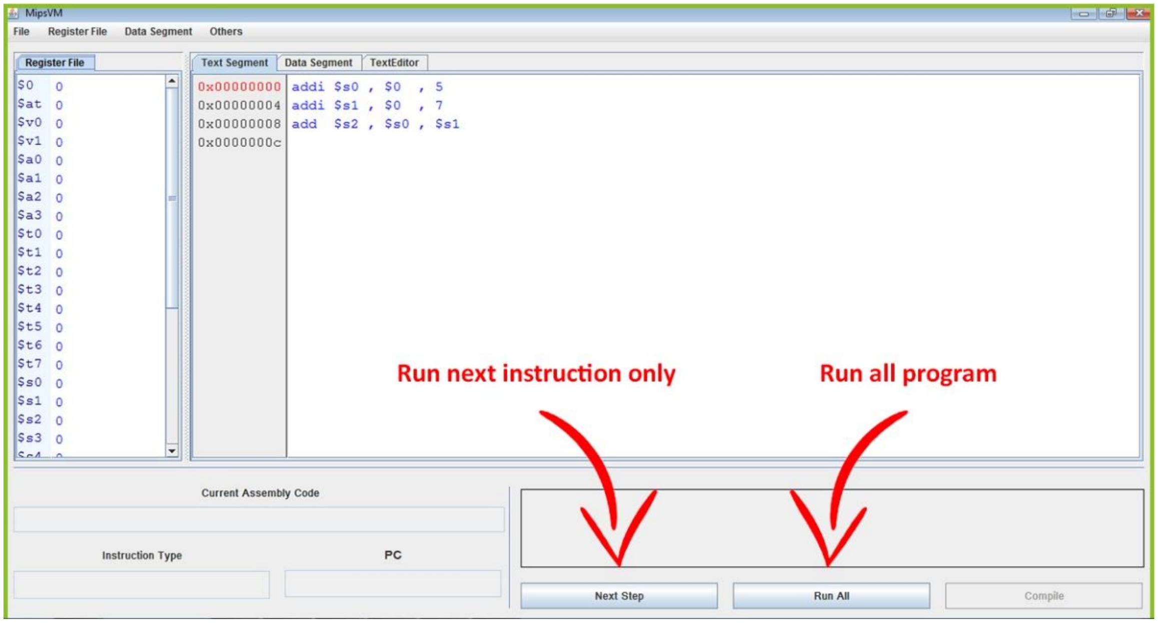


$0	0
$at	0
$v0	0
$v1	0
$a0	0
$a1	0
$a2	0
$a3	0
$t0	0
$t1	0
$t2	0
$t3	0
$t4	0
$t5	0
$t6	0
$t7	0
$s0	0
$s1	0
$s2	0
$s3	0

```
addi $s0 , $0  , 5
addi $s1 , $0  , 7
add  $s2 , $s0 , $s1
```

Click on compile button

Step 3: Choose whether to run next instruction only or run the whole program at once



$0	0
$at	0
$v0	0
$v1	0
$a0	0
$a1	0
$a2	0
$a3	0
$t0	0
$t1	0
$t2	0
$t3	0
$t4	0
$t5	0
$t6	0
$t7	0
$s0	0
$s1	0
$s2	0
$s3	0

```
0x00000000 addi $s0 , $0  , 5
0x00000004 addi $s1 , $0  , 7
0x00000008 add  $s2 , $s0 , $s1
0x0000000c
```

Current Assembly Code

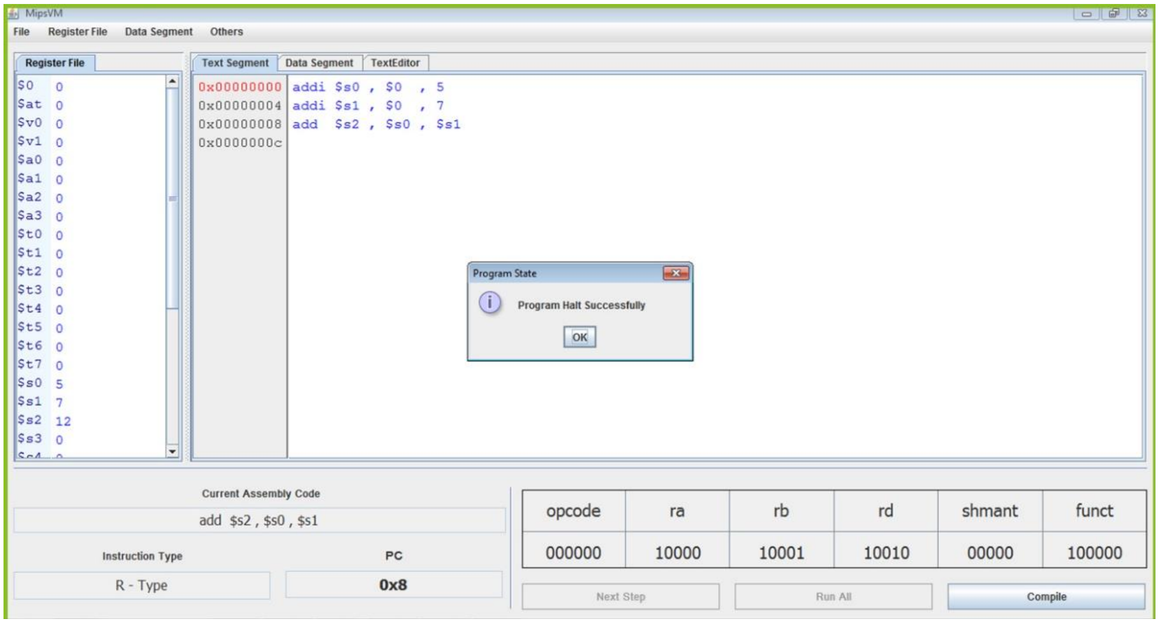
Instruction Type

PC

Run next instruction only

Run all program

Step 4: Program halts successfully after finishing all the instructions.



$0	0
$at	0
$v0	0
$v1	0
$a0	0
$a1	0
$a2	0
$a3	0
$t0	0
$t1	0
$t2	0
$t3	0
$t4	0
$t5	0
$t6	0
$t7	0
$s0	5
$s1	7
$s2	12
$s3	0

```
0x00000000 addi $s0 , $0  , 5
0x00000004 addi $s1 , $0  , 7
0x00000008 add  $s2 , $s0 , $s1
0x0000000c
```

Program Halt Successfully

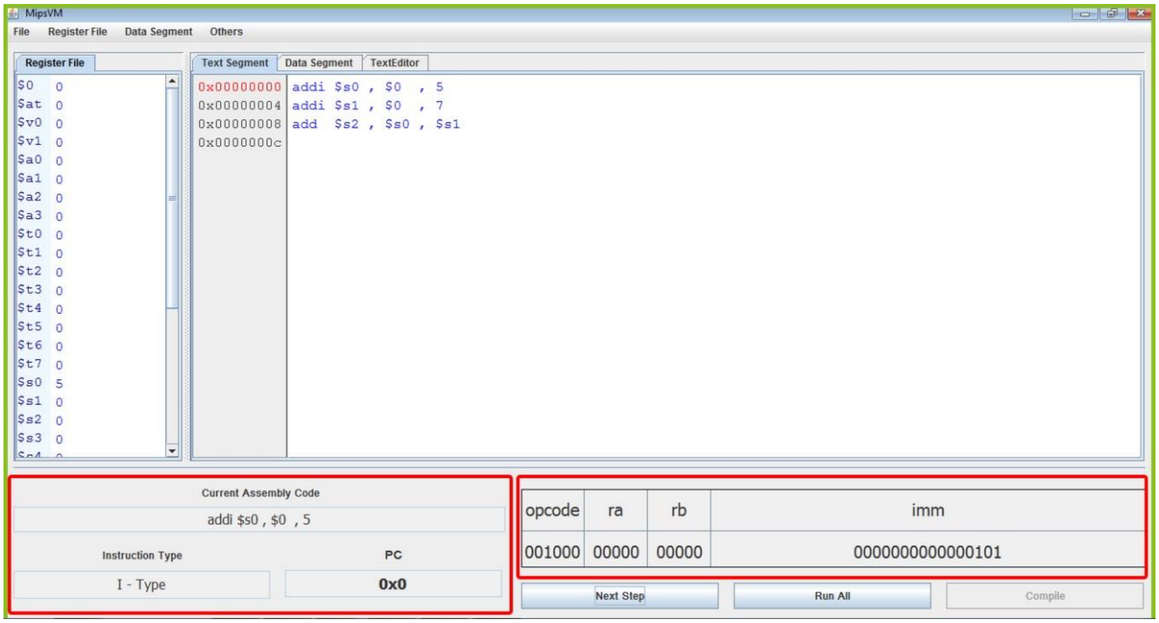
Current Assembly Code: add $s2 , $s0 , $s1

Instruction Type: R - Type

PC: 0x8

opcode	ra	rb	rd	shmant	funct
000000	10000	10001	10010	00000	100000

Note 1: Instruction type and program counter is displayed in the left red box showing which instruction is currently being executed, the right red box shows the current instruction in the binary format.



```
$0   0
$at  0
$v0  0
$v1  0
$a0  0
$a1  0
$a2  0
$a3  0
$t0  0
$t1  0
$t2  0
$t3  0
$t4  0
$t5  0
$t6  0
$t7  0
$s0  5
$s1  0
$s2  0
$s3  0
```

```
0x00000000 addi $s0 , $0  , 5
0x00000004 addi $s1 , $0  , 7
0x00000008 add  $s2 , $s0 , $s1
0x0000000c
```

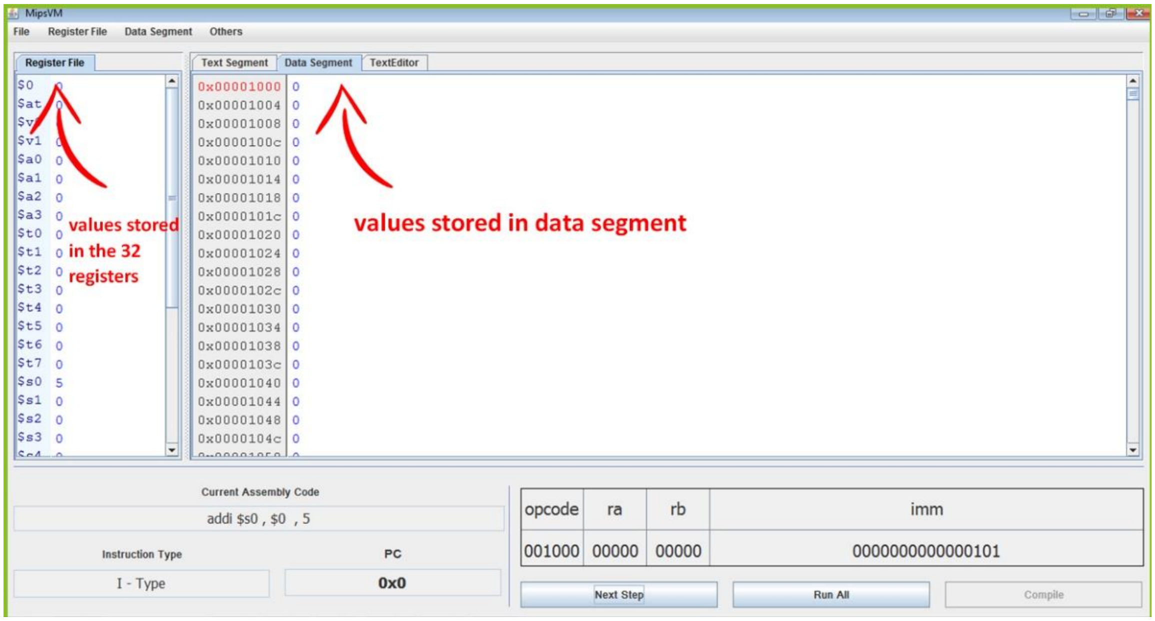
Current Assembly Code: addi $s0 , $0 , 5

Instruction Type: I - Type

PC: 0x0

opcode	ra	rb	imm
001000	00000	00000	0000000000000101

Note 2: The values stored in the registers and the memory can be viewed when needed.



```
$0   0
$at  0
$v0  0
$v1  0
$a0  0
$a1  0
$a2  0
$a3  0
$t0  0
$t1  0
$t2  0
$t3  0
$t4  0
$t5  0
$t6  0
$t7  0
$s0  5
$s1  0
$s2  0
$s3  0
```

```
0x00001000 0
0x00001004 0
0x00001008 0
0x0000100c 0
0x00001010 0
0x00001014 0
0x00001018 0
0x0000101c 0
0x00001020 0
0x00001024 0
0x00001028 0
0x0000102c 0
0x00001030 0
0x00001034 0
0x00001038 0
0x0000103c 0
0x00001040 0
0x00001044 0
0x00001048 0
0x0000104c 0
```

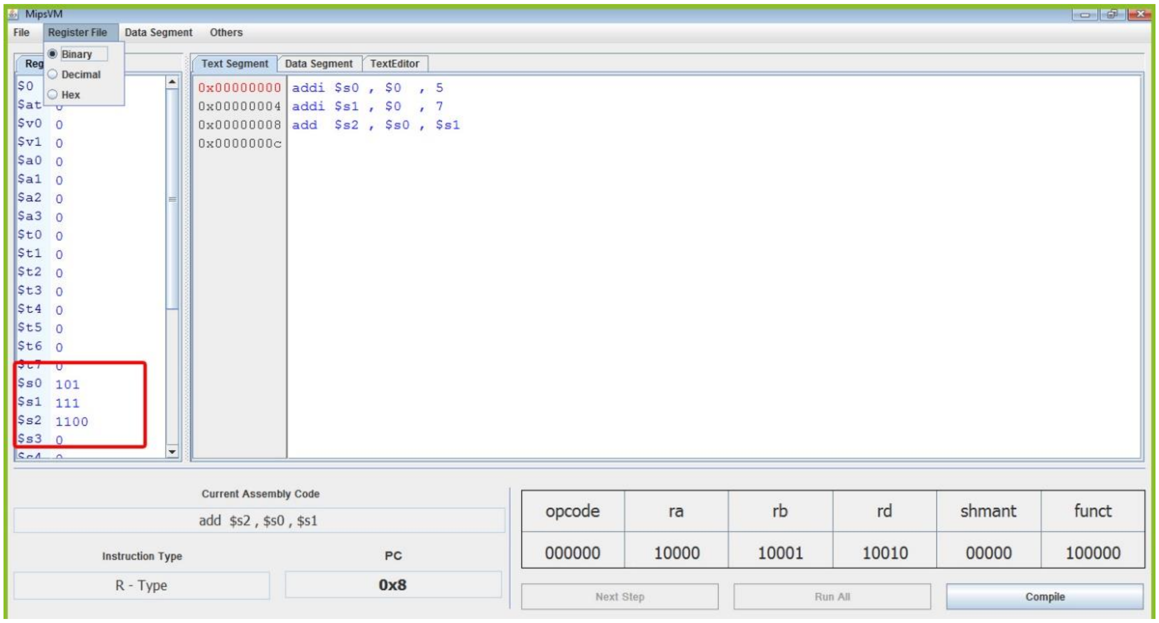
Current Assembly Code: addi $s0 , $0 , 5

Instruction Type: I - Type

PC: 0x0

opcode	ra	rb	imm
001000	00000	00000	0000000000000101

Note 3: You can view the register file values and data segment values using any of the different numbering-system bases suitable.



```
0x00000000 addi $s0 , $0  , 5
0x00000004 addi $s1 , $0  , 7
0x00000008 add  $s2 , $s0 , $s1
0x0000000c
```

```
$0
$at  0
$v0  0
$v1  0
$a0  0
$a1  0
$a2  0
$a3  0
$t0  0
$t1  0
$t2  0
$t3  0
$t4  0
$t5  0
$t6  0
$t7  0
$s0  101
$s1  111
$s2  1100
$s3  0
$s4  0
```

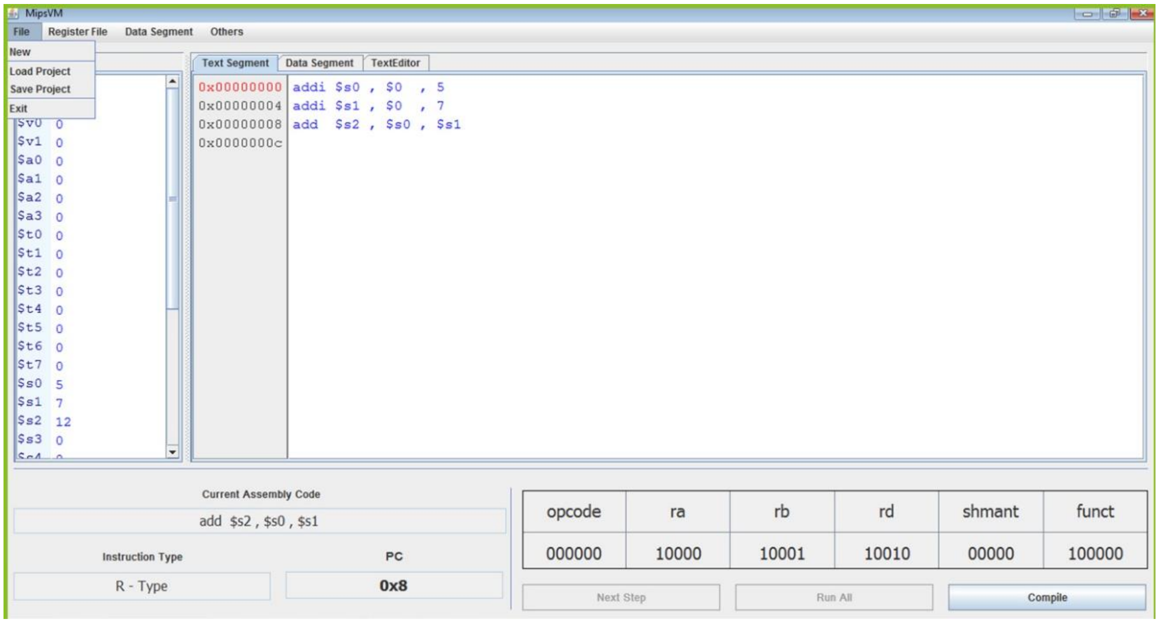
Current Assembly Code: add $s2 , $s0 , $s1

Instruction Type: R - Type

PC: 0x8

opcode	ra	rb	rd	shmant	funct
000000	10000	10001	10010	00000	100000

Note 4: You can load a file to compile or save the current assembly code from the text editor to a file.



```
0x00000000 addi $s0 , $0  , 5
0x00000004 addi $s1 , $0  , 7
0x00000008 add  $s2 , $s0 , $s1
0x0000000c
```

```
$v0  0
$v1  0
$a0  0
$a1  0
$a2  0
$a3  0
$t0  0
$t1  0
$t2  0
$t3  0
$t4  0
$t5  0
$t6  0
$t7  0
$s0  5
$s1  7
$s2  12
$s3  0
$s4  0
```

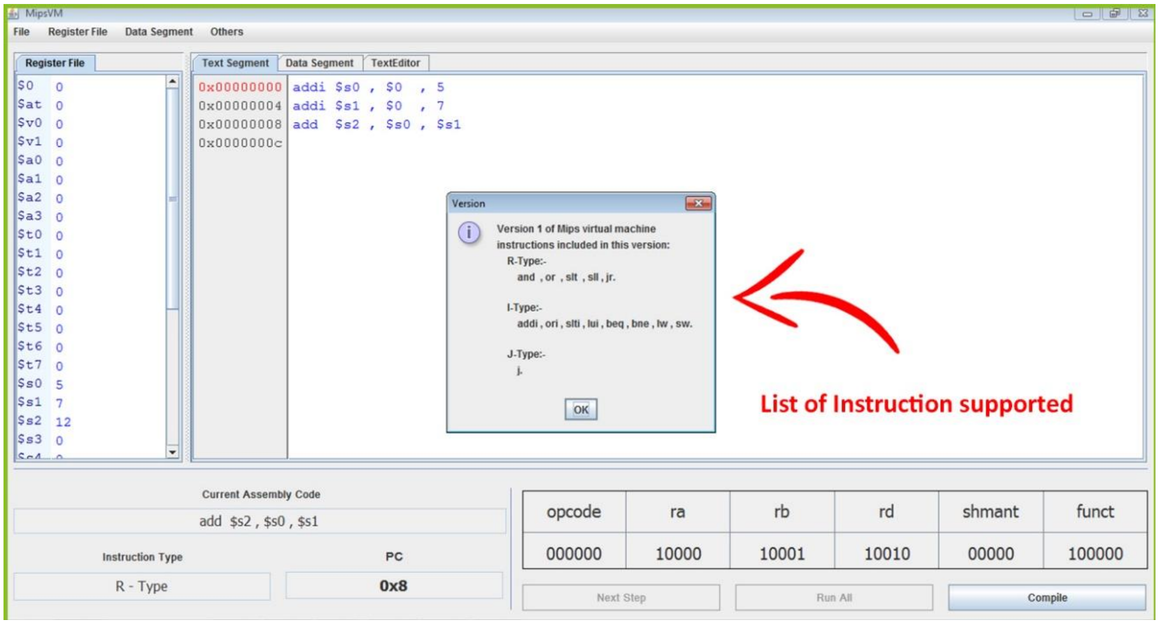
Current Assembly Code: add $s2 , $s0 , $s1

Instruction Type: R - Type

PC: 0x8

opcode	ra	rb	rd	shmant	funct
000000	10000	10001	10010	00000	100000

List of supported instructions:



$0	0
$at	0
$v0	0
$v1	0
$a0	0
$a1	0
$a2	0
$a3	0
$t0	0
$t1	0
$t2	0
$t3	0
$t4	0
$t5	0
$t6	0
$t7	0
$s0	5
$s1	7
$s2	12
$s3	0

```
0x00000000 addi $s0 , $0  , 5
0x00000004 addi $s1 , $0  , 7
0x00000008 add  $s2 , $s0 , $s1
0x0000000c
```

Version

Version 1 of Mips virtual machine
instructions included in this version:
R-Type:-
and , or , slt , sll , jr.

I-Type:-
addi , ori , slti , lui , beq , bne , lw , sw.

J-Type:-
j.

OK

List of Instruction supported

Current Assembly Code: add $s2 , $s0 , $s1

Instruction Type: R - Type

PC: 0x8

opcode	ra	rb	rd	shmant	funct
000000	10000	10001	10010	00000	100000