CS 8803: Compilers: theory and practice
Project Phase 2: Back end
Testing and output report

Team: Gang Liao, Rachna Saxena

This document describes the assembly code output of Tiger language programs. In the second phase of project, back end components like naive register allocation, control flow graph, intra block allocation, and whole function register allocation using Briggs algorithm using live range analysis is implemented. It also implements instruction selection and code generation. Bonus2 i.e. function calls are also part of this implementation.

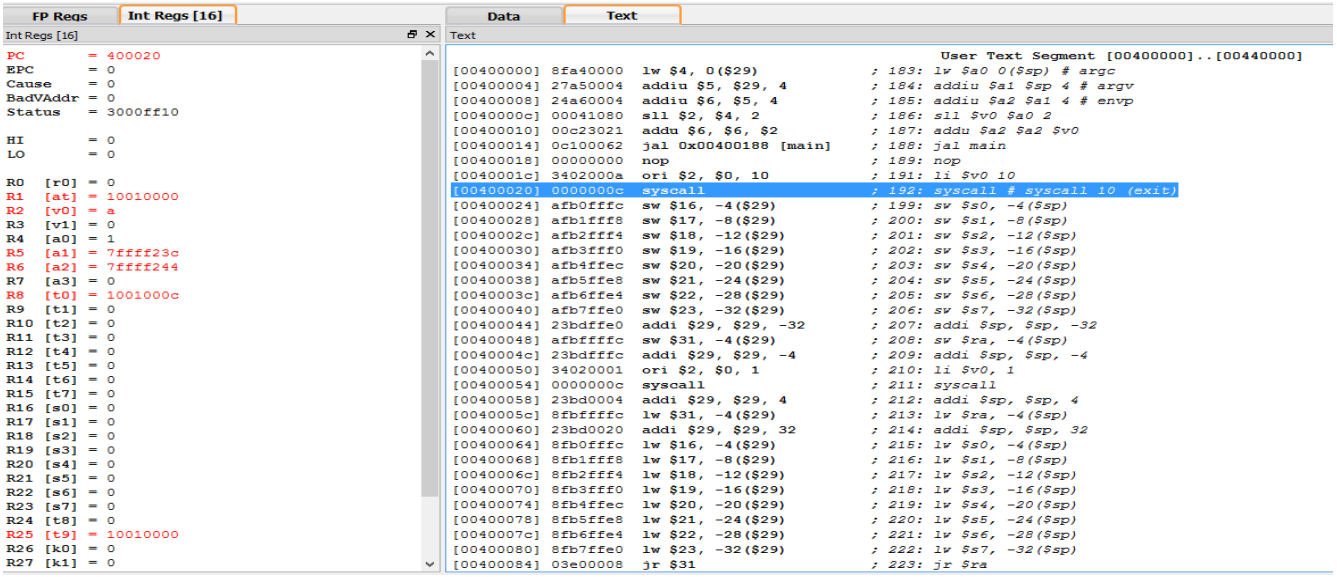
This report contains test case results for generated assembly code executed on SPIM, MIPS based simulator. spim-keepstats is used to generate execution level details.

Test cases Output: All outputs are present in the directory testcases2. Below table describes the purpose and result of different each test cases.

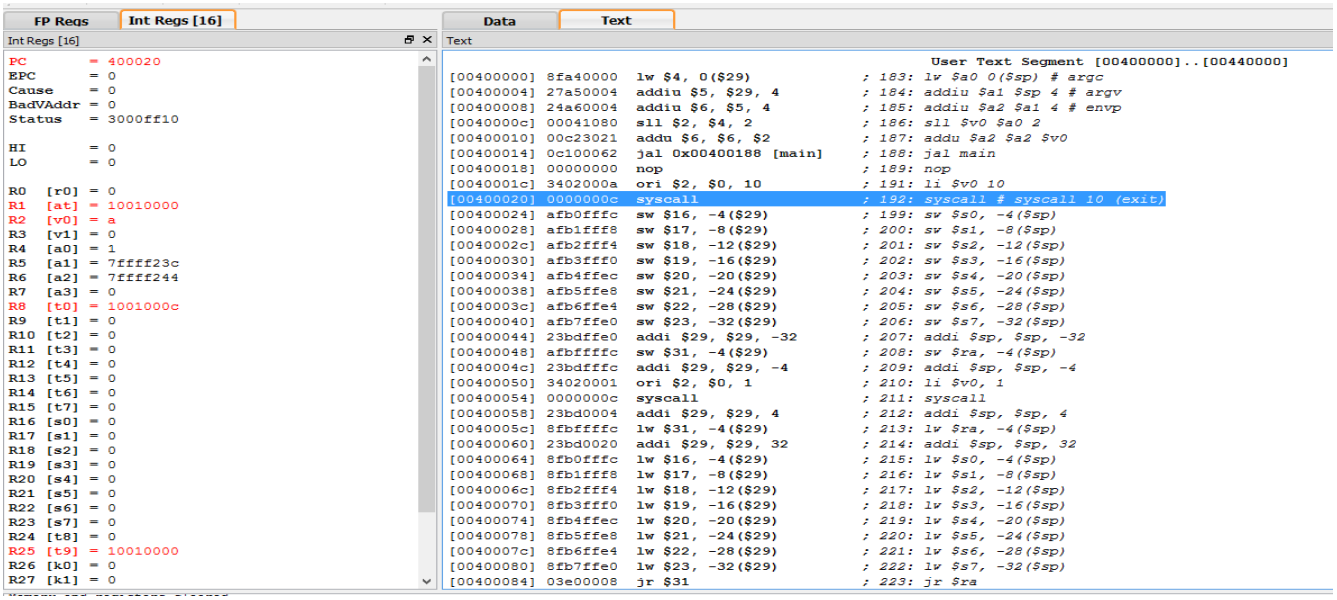
Test case	**Test Scenario**	**Status**	**Comments**
Float.tiger	Tests for floating point parameter to a function and float addition	Pass	
Float_arr.tiger	Tests for floating point array element passed as parameter to a function and float addition	Pass	
Func.tiger	Test for function call, argument passing	Pass	
Func_ret.tiger	Test for function call, argument passing and return value implementation	Pass	
Test1.tiger	Type and variable declaration, for loop, array usage	Pass	
Test2.tiger	Type declaration, function definition, function call	Pass	
Test3.tiger	Variable declaration (float type), function definition, function call	Pass	
Test5.tiger	Variable declaration, if-then-else statement	Pass	
factorial.tiger	Calculate and print factorial of a number	Pass	

Float.tiger:

Float.tiger.naive.s:



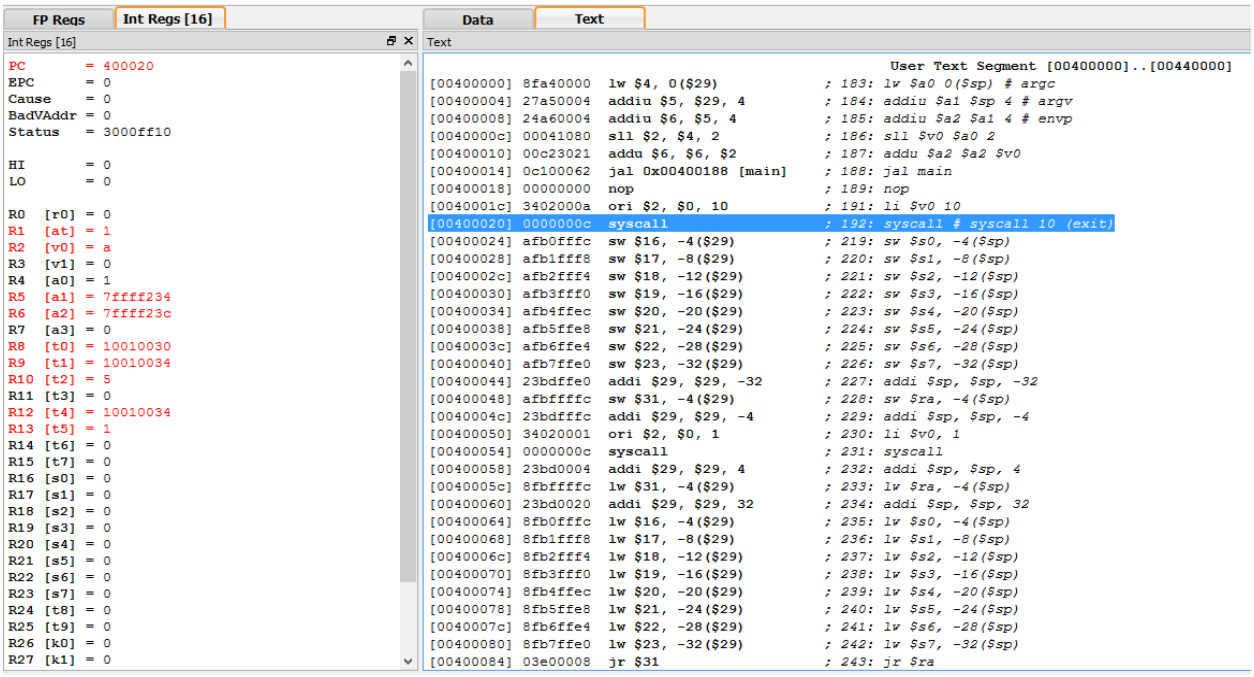
Float.tiger.cfg.s:



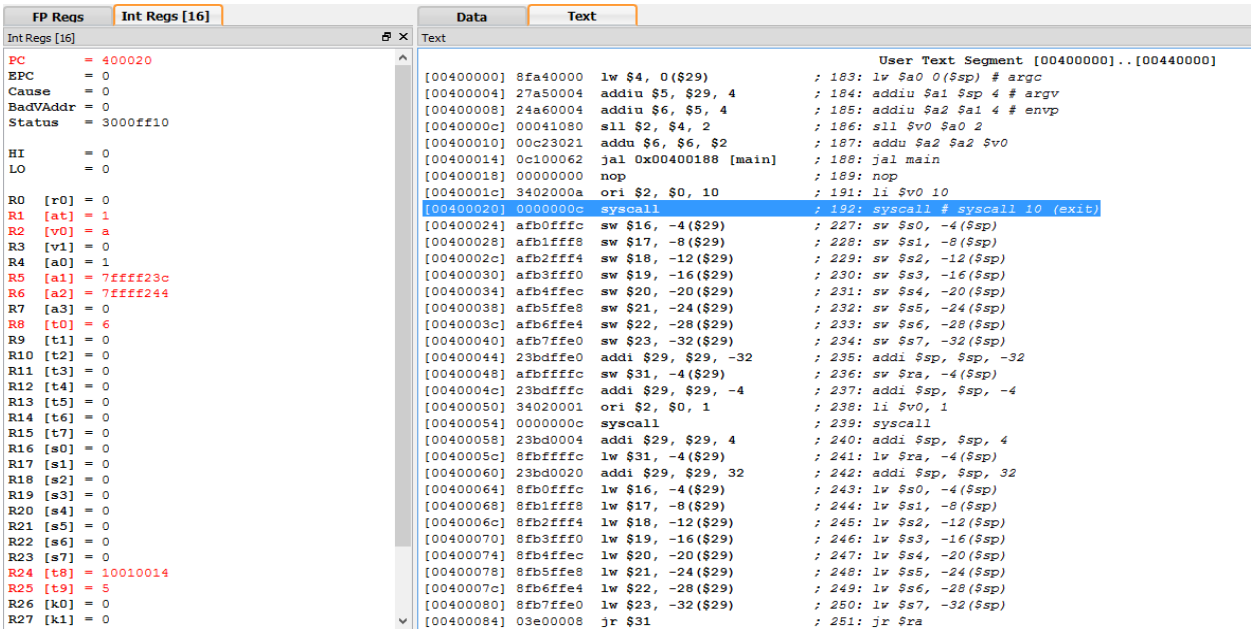
Code/Files	Instructions	Read	Write	Branches	Other
Float.tiger.naive.s	132	29	28	5	70
Float.tiger.cfg.s	134	29	28	5	72

Float_arr.tiger:

Float_arr.tiger.naive.s



Float_arr.tiger.cfg.s

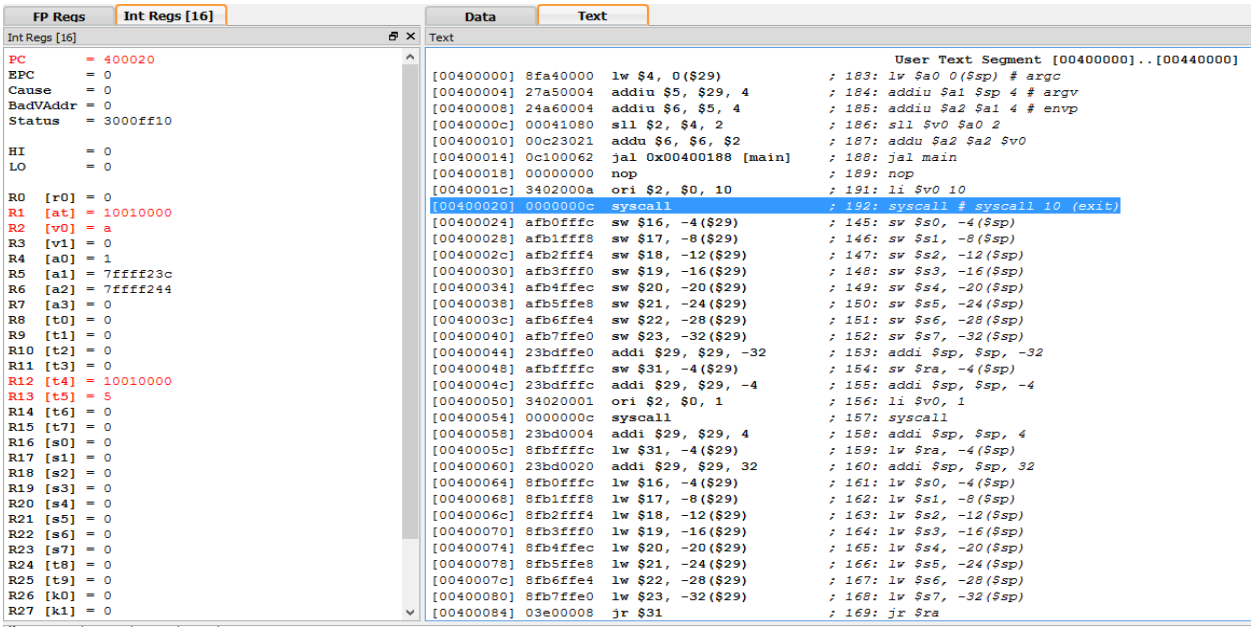


Code/Files	Instructions	Read	Write	Branches	Other
Float_arr.tiger.naive.s	691	159	131	28	373
Float_arr.tiger.cfg.s	680	148	131	28	373

There is 6.92% improved in number of operations. There is 1.62% improvement in the number of instructions in cfg implementation.

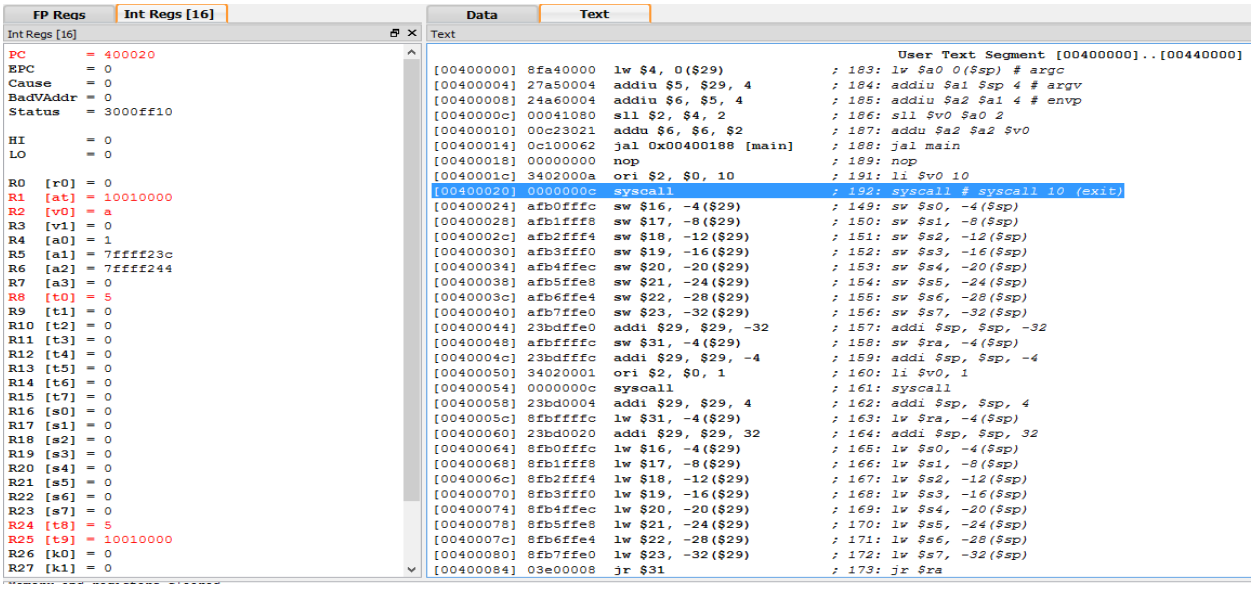
func.tiger:

func.tiger.naive.s



Console output: 5

Func.tiger.cfg.s



Console output: 5

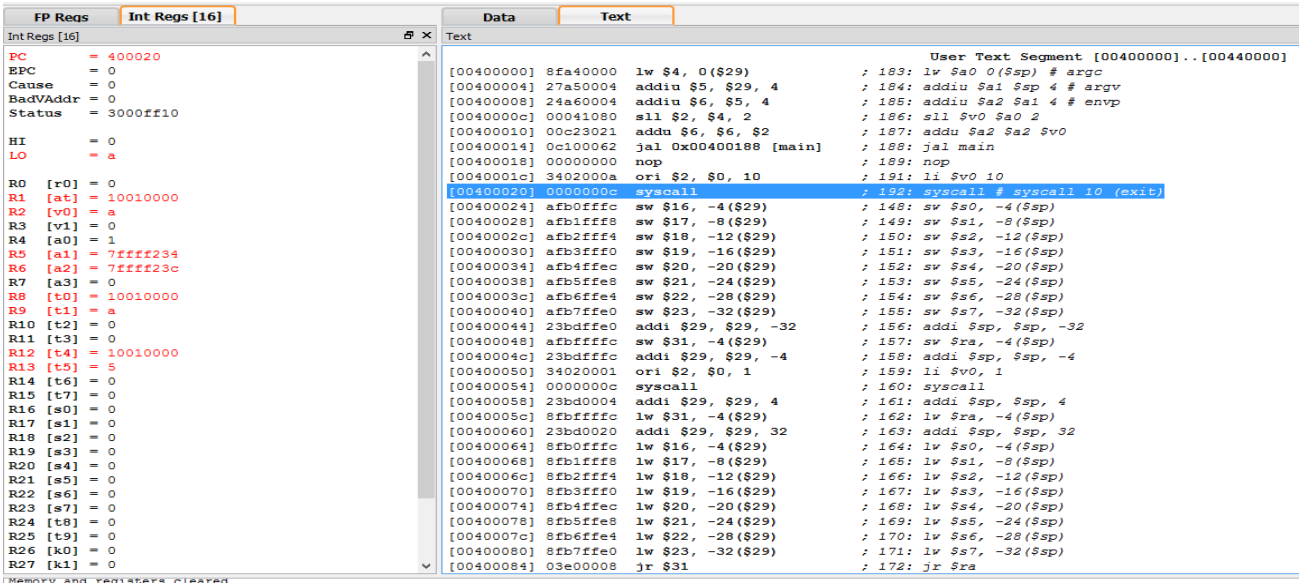
Code/Files	Instructions	Read	Write	Branches	Other

func.tiger.naive.s	153	50	48	7	48
func.tiger.cfg.s	157	50	49	7	51

In this case, there is 2.04% improvement in the number of write operations.

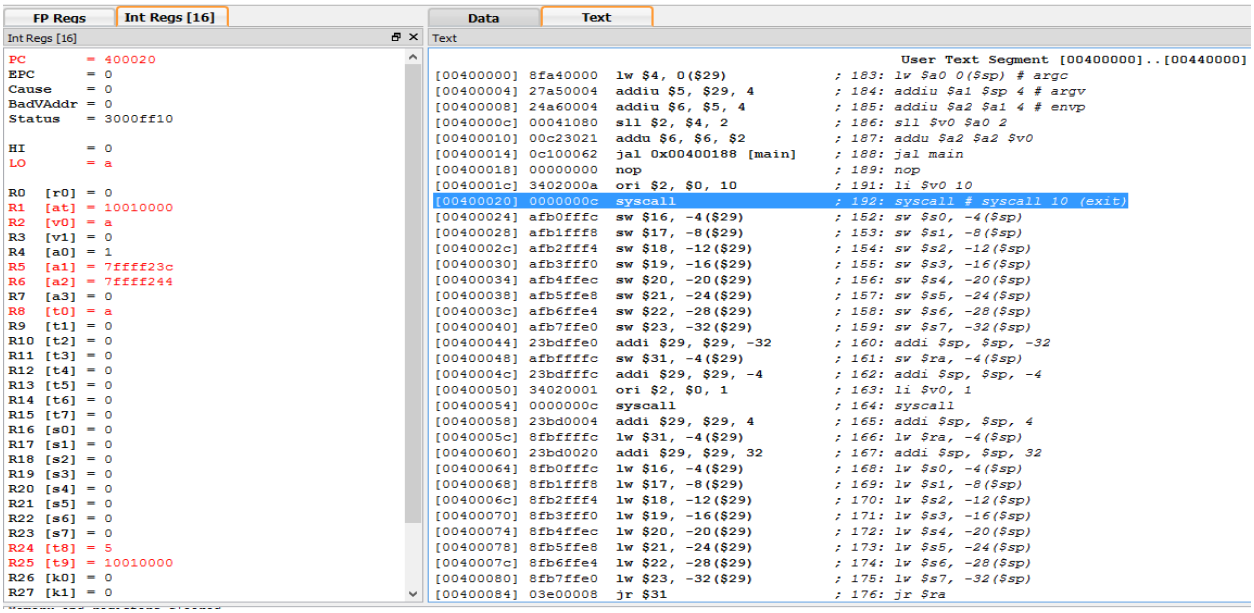
Func_ret.tiger:

Func_ret.tiger.naive.s



Console output: 10

Func_ret.tiger.cfg.s



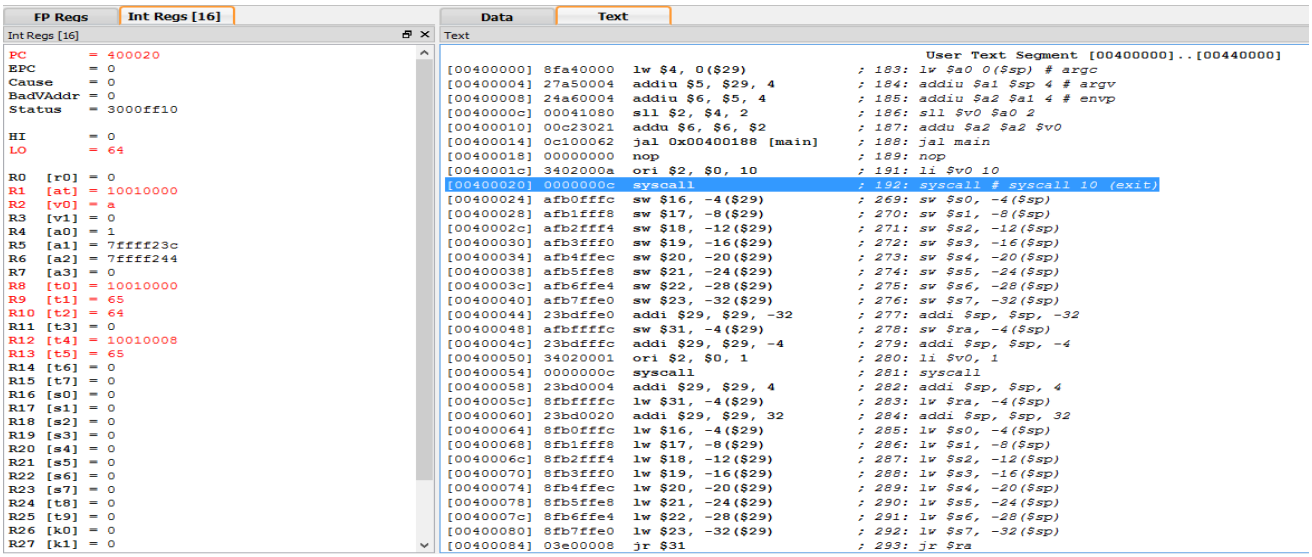
Console output: 10

Code/Files	Instructions	Read	Write	Branches	Other
Func_ret.tiger.naive.s	169	53	50	7	59
Func_ret.tiger.cfg.s	170	52	50	7	61

In this case, there is 1.92% improvement in the number of read operation in the cfg implementation.

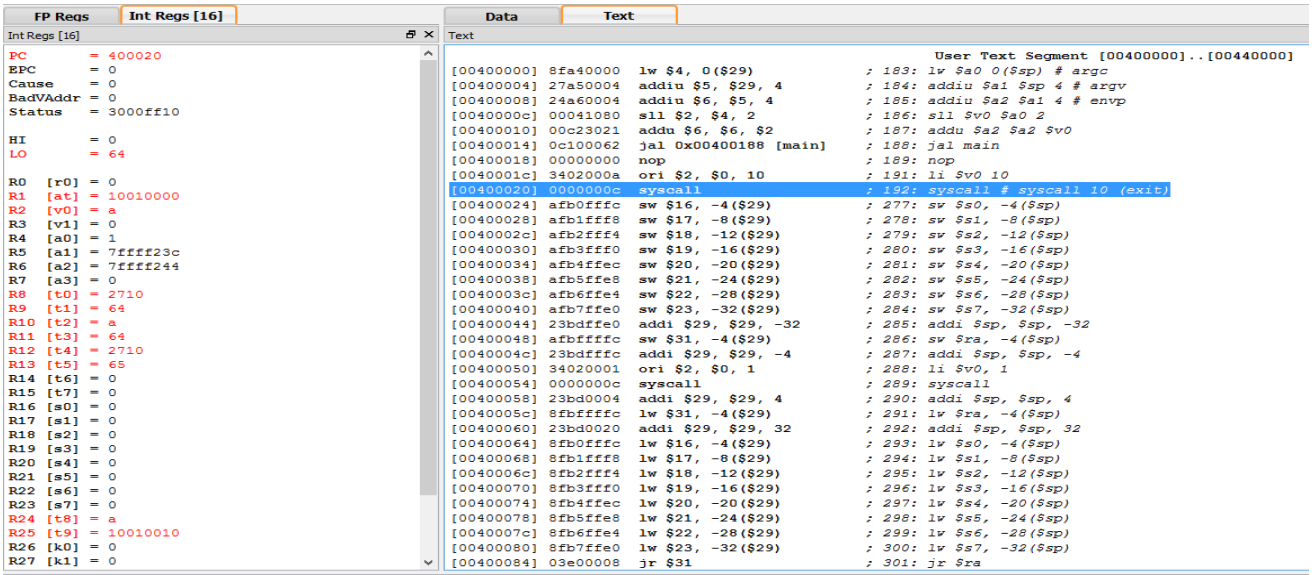
Test1.tiger:

Test1.tiger.naive.s



Console output: 10000

Test1.tiger.cfg.s



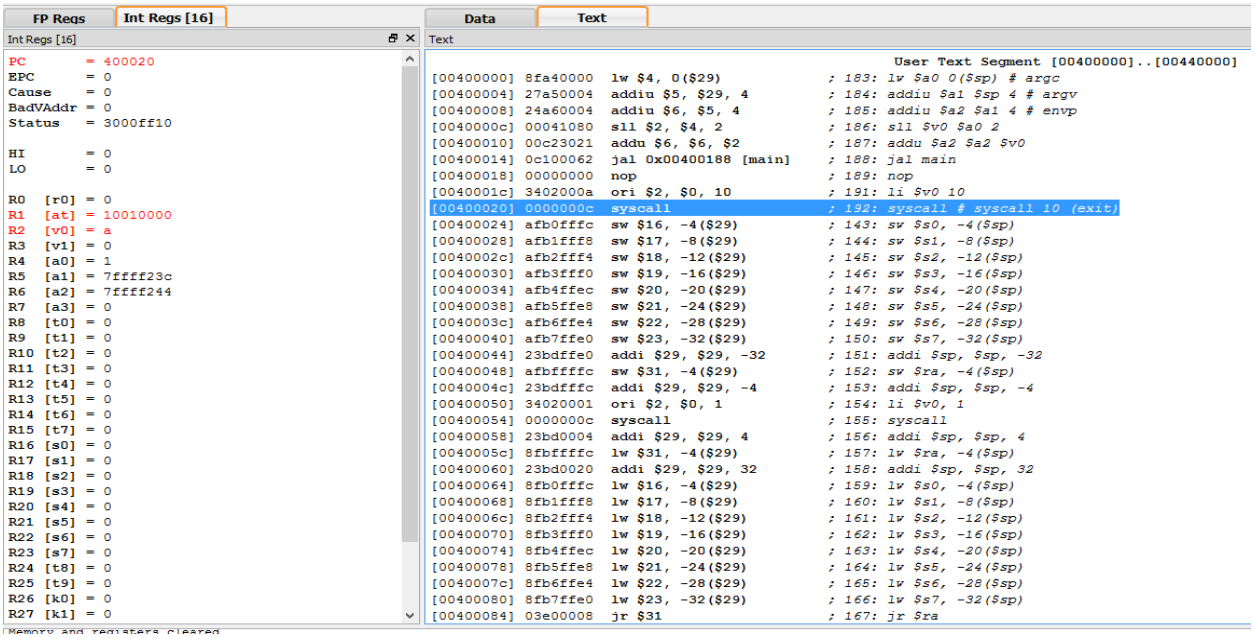
Console output: 10000

Code/Files	Instructions	Read	Write	Branches	Other
Test1.tiger.naive.s	7296	1350	837	207	4902
Test1.tiger.cfg.s	7016	1053	838	207	4918

There is 0.12% improvement in the write performance in the improved implementation. There is 3.99% improvement in the number of instructions.

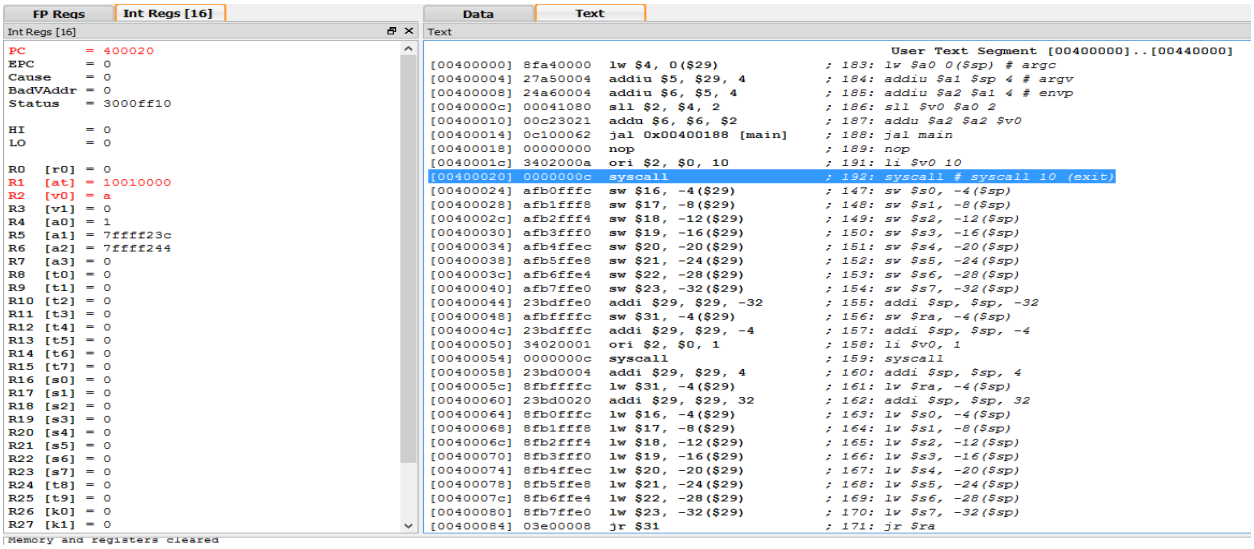
Test2.tiger:

Test2.tiger.naive.s



Console output: 5

Test2.tiger.cfg.s

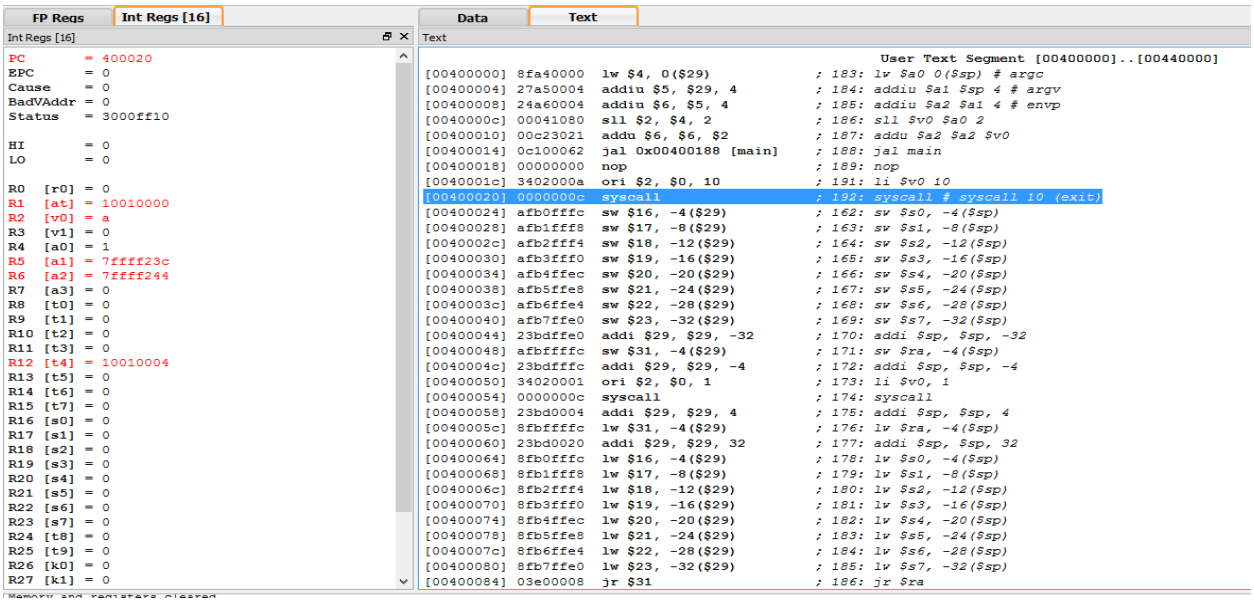


Console output: 5

Code/Files	Instructions	Read	Write	Branches	Other
Test2.tiger.naive.s	149	49	47	7	46
Test2.tiger.cfg.s	151	49	47	7	48

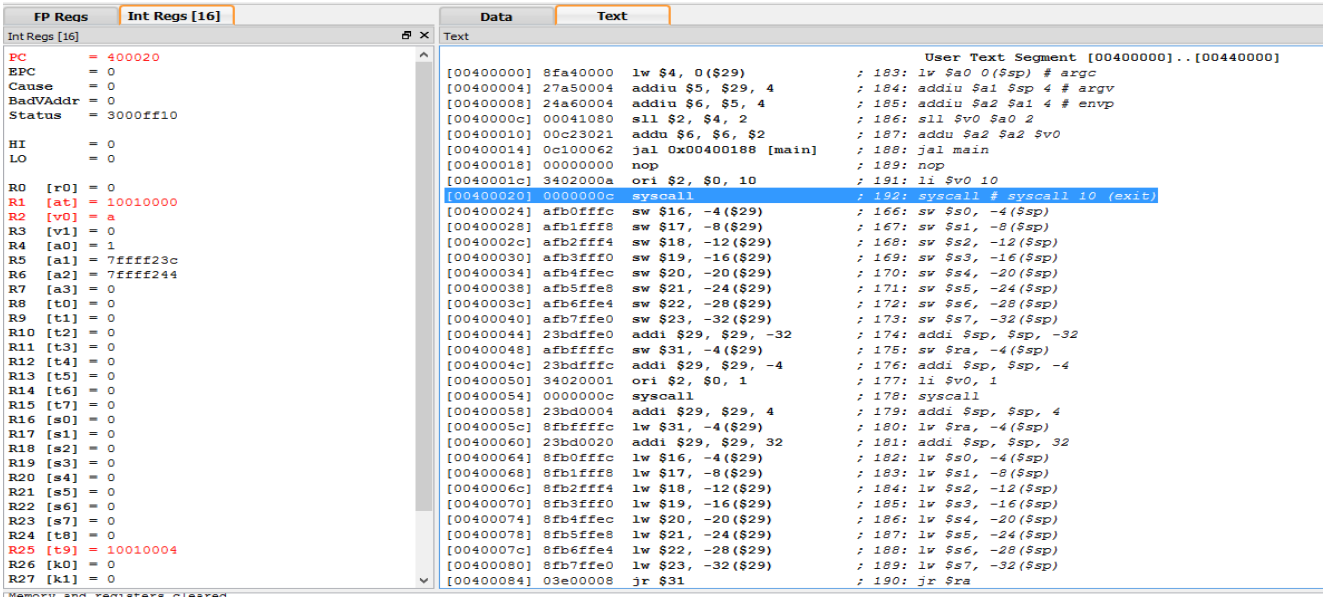
Test3.tiger

Test3.tiger.naive.s



Console output: 5

Test3.tiger.cfg.s

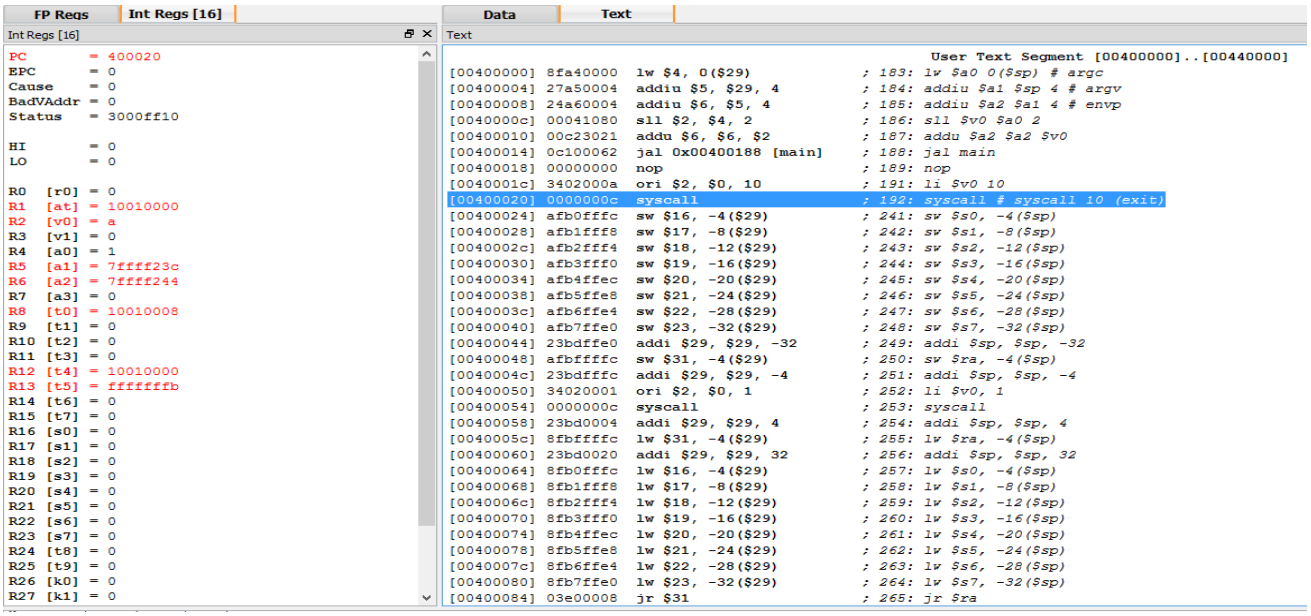


Console output:5

Code/Files	Instructions	Read	Write	Branches	Other
Test3.tiger.naive.s	166	49	47	7	63
Test3.tiger.cfg.s	172	49	47	7	69

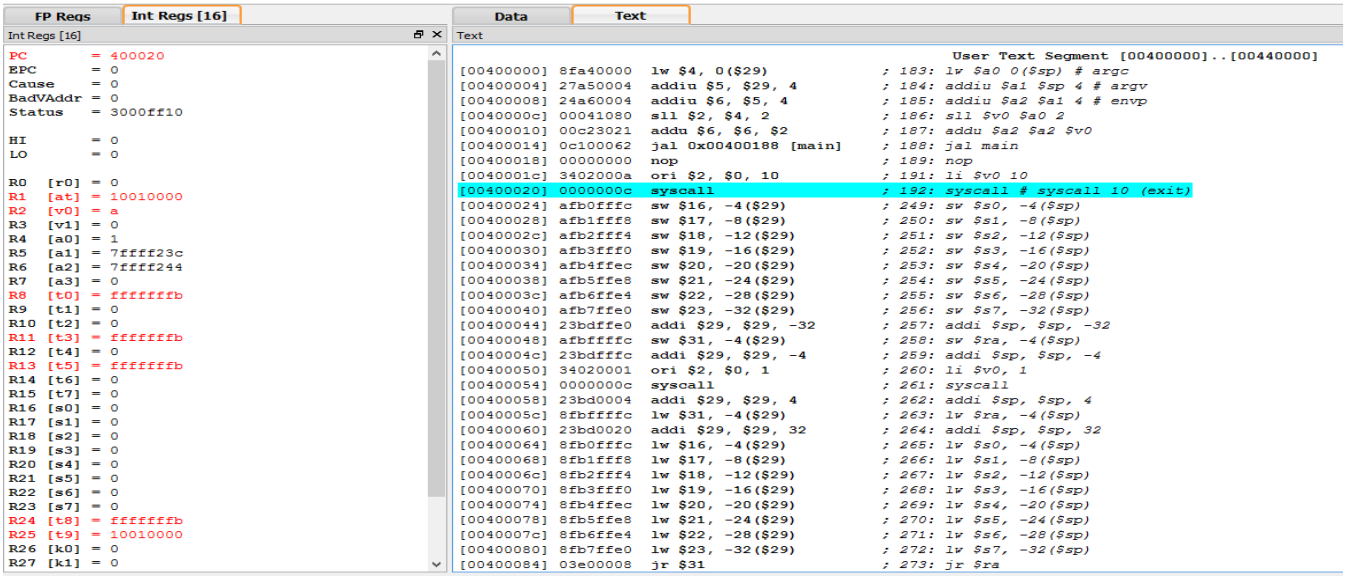
Test5.tiger:

Test5.tiger.naive.s



Console output: -5

Test5.tiger.cfg.s



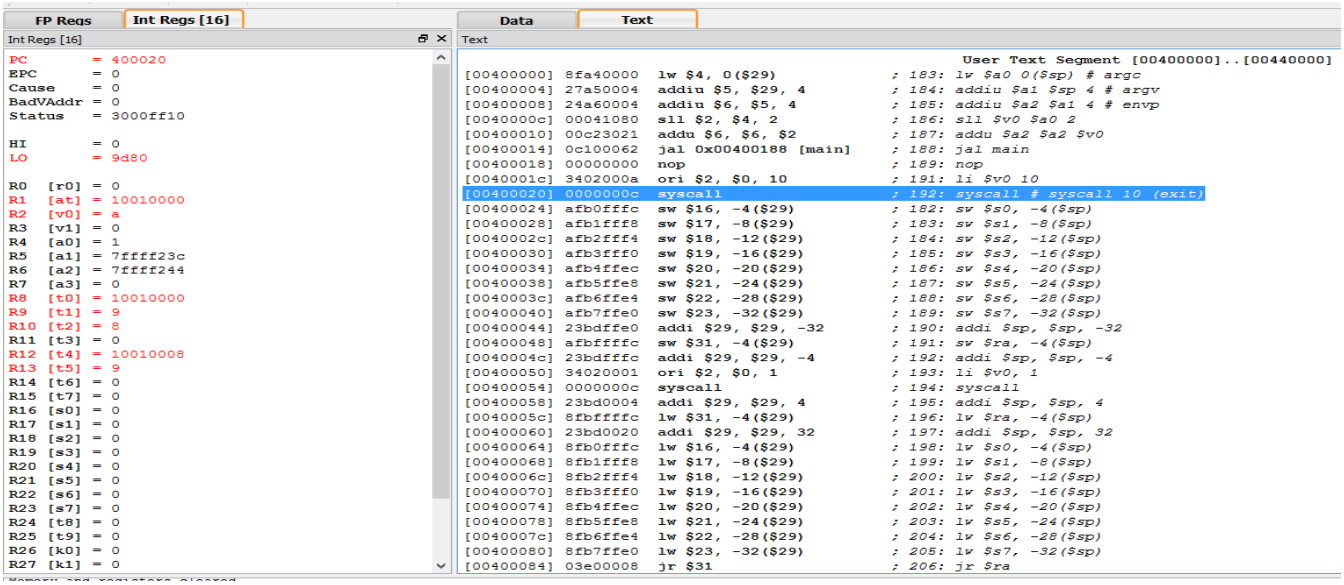
Console output: -5

Code/Files	Instructions	Read	Write	Branches	Other
Test5.tiger.naive.s	175	47	37	6	85
Test5.tiger.cfg.s	177	43	39	6	89

There is 9.3% improvement in the number of instruction in improved implementation.

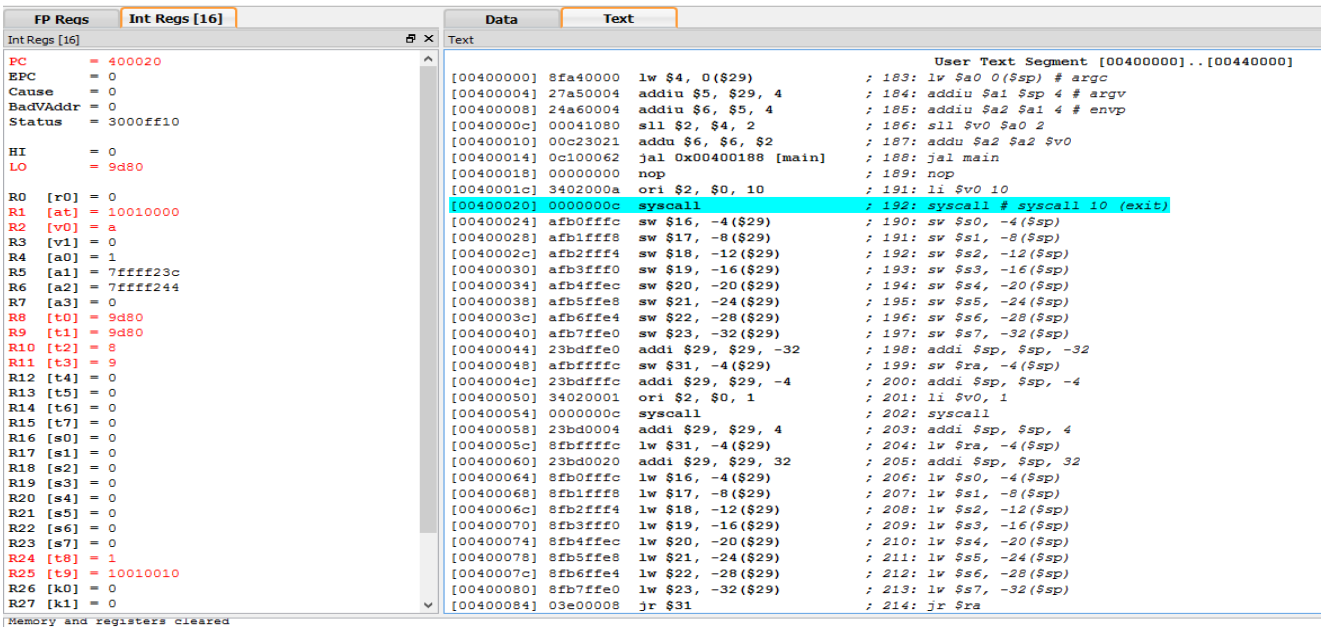
factorial.tiger:

factorial.tiger.naive.s



Console output: 40320

factorial.tiger.cfg.s



Console output: 40320

Code/Files	Instructions	Read	Write	Branches	Other
factorial.tiger.naive.s	402	92	56	21	233
factorial.tiger.cfg.s	408	79	65	21	243

There is 16.46% improvement in the number of read operations in the cfg implementation.