UNIVERSITÄT DES SAARLANDES

Programming 2 - SS24

Project 1 - Rock Paper Scissors

Authors: Christopher Cohnen, Chi Fang, Elvira Mayer

30. April 2024

Universität des Saarlandes

Overview

Git introduction

Configuration

`$ git config` is used to configure Git repositories.

- `--global` sets up the global configuration.
    - `user.name "firstname lastname"`
    - `user.email "...@stud.uni-saarland.de"`

Configuration

`$ git config` is used to configure Git repositories.

- `--global` sets up the global configuration.
  - `user.name "firstname lastname"`
  - `user.email "...@stud.uni-saarland.de"`

Example

```
$ git config --global user.name "Konrad Klug"
```

Git project repository

We can obtain the project using `$ git clone` and the following url:

```
ssh://git@dgit.cs.uni-saarland.de:2222/prog2/2024/students/
project-1-<NUMBER>.git
```

<NUMBER> = your matriculation number

Git project repository

We can obtain the project using `$ git clone` and the following url:

```
ssh://git@dgit.cs.uni-saarland.de:2222/prog2/2024/students/
project-1-<NUMBER>.git
```

<NUMBER> = your matriculation number

Caution

You must have created and uploaded an ssh-key to dgit.cs.uni-saarland.de beforehand.

▸ Check out the git section in the installation guide.

Submitting the project

- `$ git status` - list modified files
- `$ git add <file>` - stage the modified files
- `$ git commit -m "message"` - commit all the staged files
- `$ git push` - submit commits

[1] anywhere on Earth, AoE

Submitting the project

- `$ git status` - list modified files
- `$ git add <file>` - stage the modified files
- `$ git commit -m "message"` - commit all the staged files
- `$ git push` - submit commits

Caution

Only the changes you submitted onto the server by Monday, 13th May 2024, end of day[1], are tested and counted as valid submissions.

[1] anywhere on Earth, AoE

MARS/MIPS Introduction

Setting up MARS

- Clone your repository
- Download the MARS executable from the dCMS Materials page
- Place Mars4_5.jar into the root directory of your repository

Starting MARS via commandline

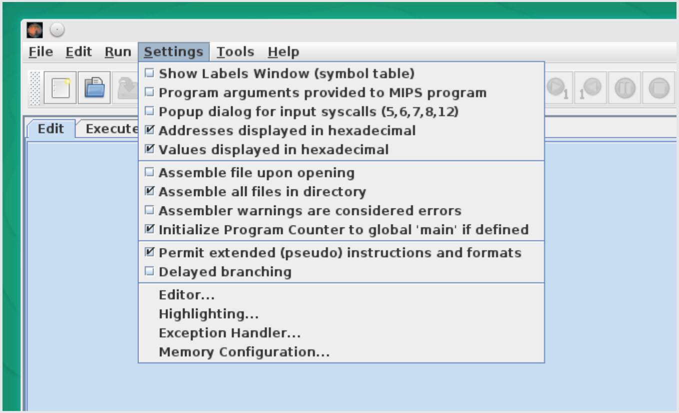
Navigate to the directory in the commandline and execute

```
$ java -jar Mars4_5.jar
```

MARS Settings

Caution

We have to adjust two settings:



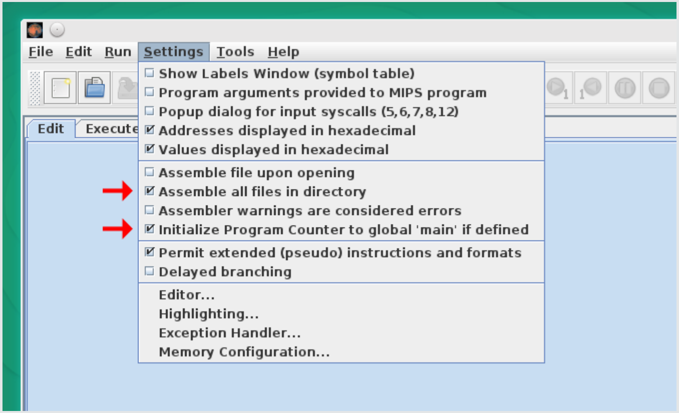
- [ ] Show Labels Window (symbol table)
- [ ] Program arguments provided to MIPS program
- [ ] Popup dialog for input syscalls (5,6,7,8,12)
- [✓] Addresses displayed in hexadecimal
- [✓] Values displayed in hexadecimal
- [ ] Assemble file upon opening
- [✓] Assemble all files in directory
- [ ] Assembler warnings are considered errors
- [✓] Initialize Program Counter to global 'main' if defined
- [✓] Permit extended (pseudo) instructions and formats
- [ ] Delayed branching

Editor...
Highlighting...
Exception Handler...
Memory Configuration...

MARS Settings

Caution

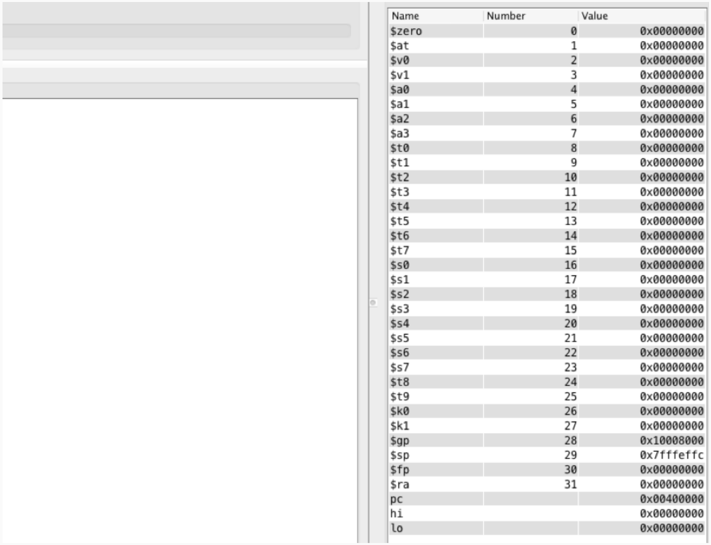
We have to adjust two settings:



☐ Show Labels Window (symbol table)
☐ Program arguments provided to MIPS program
☐ Popup dialog for input syscalls (5,6,7,8,12)
☑ Addresses displayed in hexadecimal
☑ Values displayed in hexadecimal
☐ Assemble file upon opening
→ ☑ Assemble all files in directory
☐ Assembler warnings are considered errors
→ ☑ Initialize Program Counter to global 'main' if defined
☑ Permit extended (pseudo) instructions and formats
☐ Delayed branching
Editor...
Highlighting...
Exception Handler...
Memory Configuration...

Calling Conventions

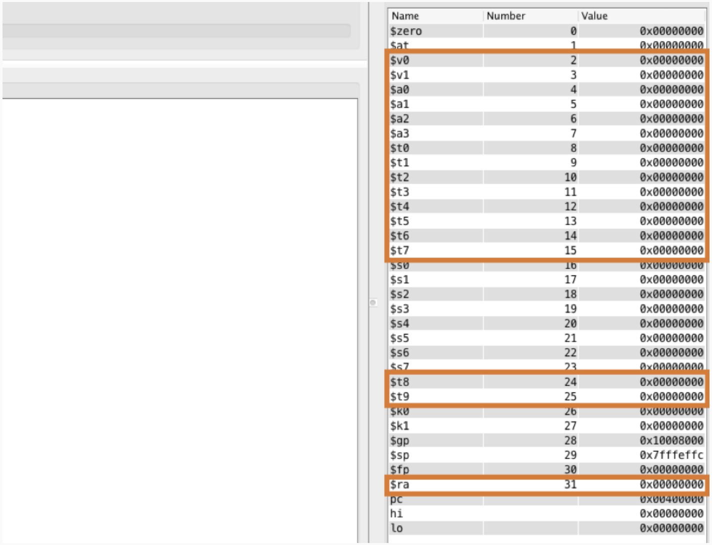
Registers



Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000000
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00000000
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400000
hi		0x00000000
lo		0x00000000

Calling Conventions

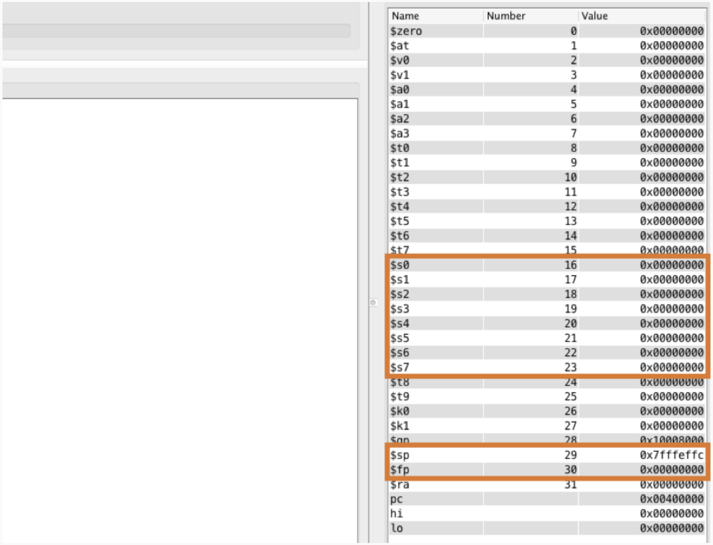
Registers - Caller save



Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000000
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00000000
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400000
hi		0x00000000
lo		0x00000000

Calling Conventions

Registers - Callee save



Name	Number	Value
$zero	0	0x00000000
$at	1	0x00000000
$v0	2	0x00000000
$v1	3	0x00000000
$a0	4	0x00000000
$a1	5	0x00000000
$a2	6	0x00000000
$a3	7	0x00000000
$t0	8	0x00000000
$t1	9	0x00000000
$t2	10	0x00000000
$t3	11	0x00000000
$t4	12	0x00000000
$t5	13	0x00000000
$t6	14	0x00000000
$t7	15	0x00000000
$s0	16	0x00000000
$s1	17	0x00000000
$s2	18	0x00000000
$s3	19	0x00000000
$s4	20	0x00000000
$s5	21	0x00000000
$s6	22	0x00000000
$s7	23	0x00000000
$t8	24	0x00000000
$t9	25	0x00000000
$k0	26	0x00000000
$k1	27	0x00000000
$gp	28	0x10008000
$sp	29	0x7fffeffc
$fp	30	0x00000000
$ra	31	0x00000000
pc		0x00400000
hi		0x00000000
lo		0x00000000

Tests and Debugging

Tests

- Public Tests - come with the project, can run locally
- Regular Tests - run on the server after you have pushed your code and pass a sufficient number of public tests
- Eval Tests - run after submission

Tests

- Public Tests - come with the project, can run locally
- Regular Tests - run on the server after you have pushed your code and pass a sufficient number of public tests
- Eval Tests - run after submission

Caution

All public tests for a subtask must be passed in order to receive points for that subtask.

Public Tests

We can run the Public Tests in our project folder using `$ python run_tests`[2].

Caution

- You need to have Python installed on your machine.

[2] or `$ python3 run_tests`, alternative: `$ ./run_tests`

Public Tests

We can run the Public Tests in our project folder using `$ python run_tests`[3].

Most important arguments

- `-h` : list of all possible arguments
- `<path_to_test.s>` : execute only the test specified in the path
- `<directory>` : execute tests in directory
- `-v` : additionally shows the printed output of your program

[3] or `$ python3 run_tests`, alternative: `$ ./run_tests`

Writing own tests

We can create our own tests in a `tests/custom`-folder:

- Create *.s*-file containing the test (must have a global `main` label)
- Create *ref*-file containing the expected output
- Run `$ python run_tests tests/custom`

Debug tests

We can debug Public Tests using `$ python run_tests <path_to_test.s> --debug`.

We can then execute and debug the files in the `debugbox` directory.

Example - Printing

We want to print an ASCII-string to the console.

Code

```
        .globl print
        .text
    print:
        move $a0 $a1
        li $v0 4
        syscall

        jr    $ra
```

Example - Testing

test.s

```
.data
  greetings:
   .asciiz "Hello␣world!"
.text
  .globl main
main:
  la $a1 greetings
  jal print
  li $v0 10
  syscall
```

test.ref

```
Hello world!
```

Questions?

About the project

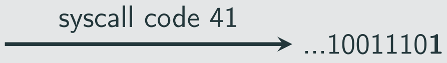
Assignment 1 - gen_bit

In the file `random.s`:

Generate 1 random bit

- $a0 - address of configuration
- $v0 - result

Functionality



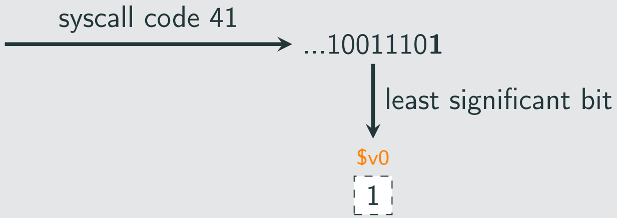
Assignment 1 - gen_bit

In the file `random.s`:

Generate 1 random bit

- $a0 - address of configuration
- $v0 - result

Functionality



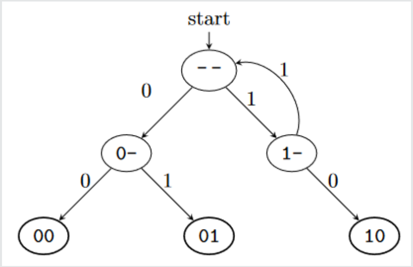
Assignment 2 - gen_byte

In the file `random.s`:

Generate random move

- $a0 - address of configuration
- $v0 - result

Functionality



rock paper scissors

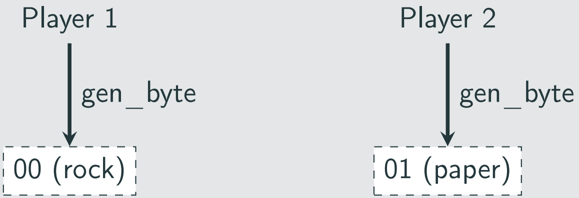
Assignment 3 - play_game_once

In the file `rps.s`:

Generate moves for two players and announce who wins

- $a0 - address of configuration

Functionality



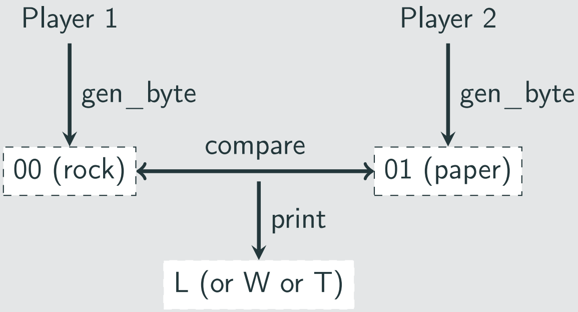
Assignment 3 - play_game_once

In the file `rps.s`:

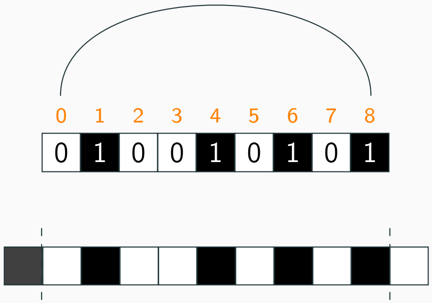
Generate moves for two players and announce who wins

- $a0 - address of configuration

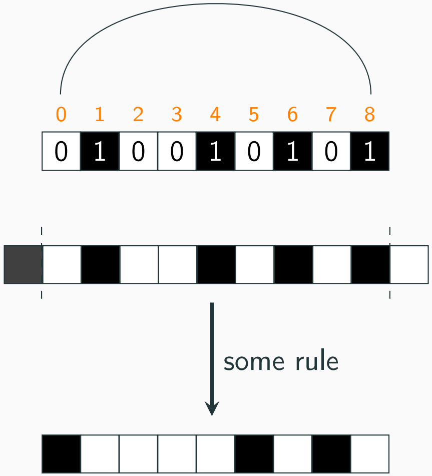
Functionality



Cellular Automaton



Cellular Automaton



Assignment 4 - print_tape

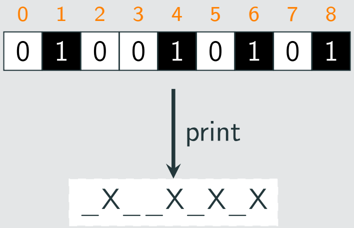
In the file `automaton.s`:

Print the tape

Configuration

eca	tape	tape_len	rule	skip	column
($a0)	4($a0)	8($a0)	9($a0)	10($a0)	11($a0)

Functionality



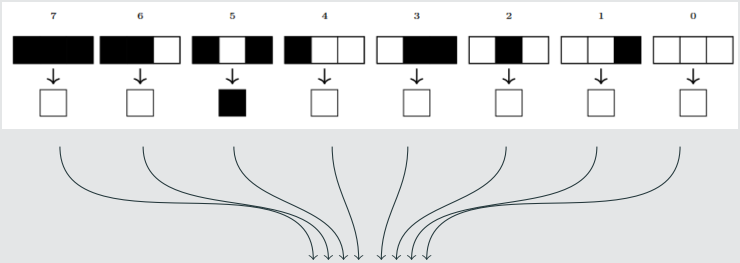
Assignment 5 - simulate_automaton

In the file `automaton.s`:

Simulate one step of the automaton

eca	tape	tape_len	rule	skip	column
(\$a0)	4(\$a0)	8(\$a0)	9(\$a0)	10(\$a0)	11(\$a0)

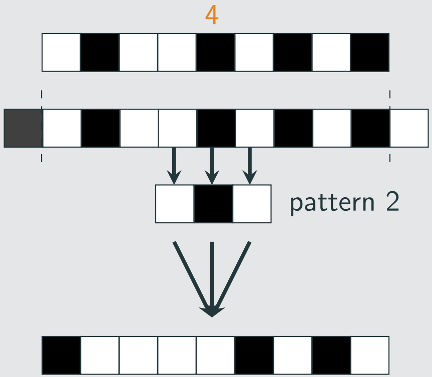
Rule



9(\$a0) 00100000

Assignment 5 - simulate_automaton

One step for position 4



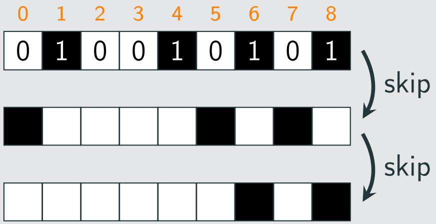
Assignment 6 - gen_bit

In the file `random.s`:

Simulate automaton and return one position

eca	tape	tape_len	rule	skip	column
(\$a0)	4(\$a0)	8(\$a0)	9(\$a0)	10(\$a0)	11(\$a0)

Functionality



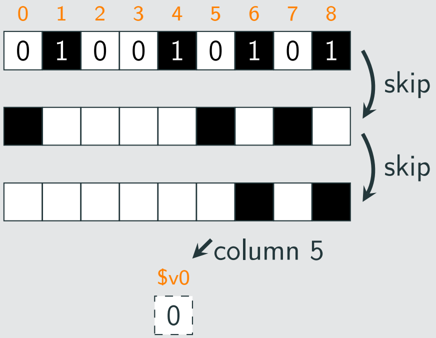
Assignment 6 - gen_bit

In the file `random.s`:

Simulate automaton and return one position

eca	tape	tape_len	rule	skip	column
($a0)	4($a0)	8($a0)	9($a0)	10($a0)	11($a0)

Functionality



(Probably) Frequent Mistakes

- Calling Convention!
- The tape has no end!
- The argument of all functions is an address!
- The tape cells are counted from left to right, but are on the right (i.e. least significant) side of the register!
- Calling Convention!

Questions?

If you have any problems, use the forum or come to the Office Hours!