

Trabalho 2: Simulador RISC-V

Aluno: Adriano Ulrich do Prado Wiedmann – 202014824

Objetivo:

Este código consiste em implementar um simulador da arquitetura RV321 usando a linguagem de alto nível C.

O código:

Foram implementados 4 arquivos. O arquivo '*simulador.c*' contém a função *main* do código. No arquivo '*instructions.c*', estão implementadas todas as funções do programa. O arquivo '*instructions.h*' é o header para o arquivo '*instructions.c*'. Por fim, o arquivo '*globals.h*' contém as variáveis globais.

O código lê dois arquivos .bin gerados pelo RARS, um arquivo '*data.bin*' e outro '*text.bin*'. Ambos são armazenados em um *array* '*mem[MEM_SIZE]*'. Levando em consideração que o segmento de código (*text*) começa no endereço *0x00000000* de memória e o segmento de data (*data*) começa no endereço *0x00002000* de memória.

Após isso, o código chama a função *run()* que será executado enquanto o valor do *pointer counter (pc)* $< MEM_SIZE$ (4096) e se não houver uma *syscall* para finalizar o programa. Quando é executado uma *syscall* para encerrar o código, é utilizada a função *exit(0)* da biblioteca padrão do C, '*<stdlib.h>*'.

Durante o *loop while* na função *run()*, é chamado a função *step()*, que executa as funções, *fetch()*, *decode()* e *execute()*, respectivamente.

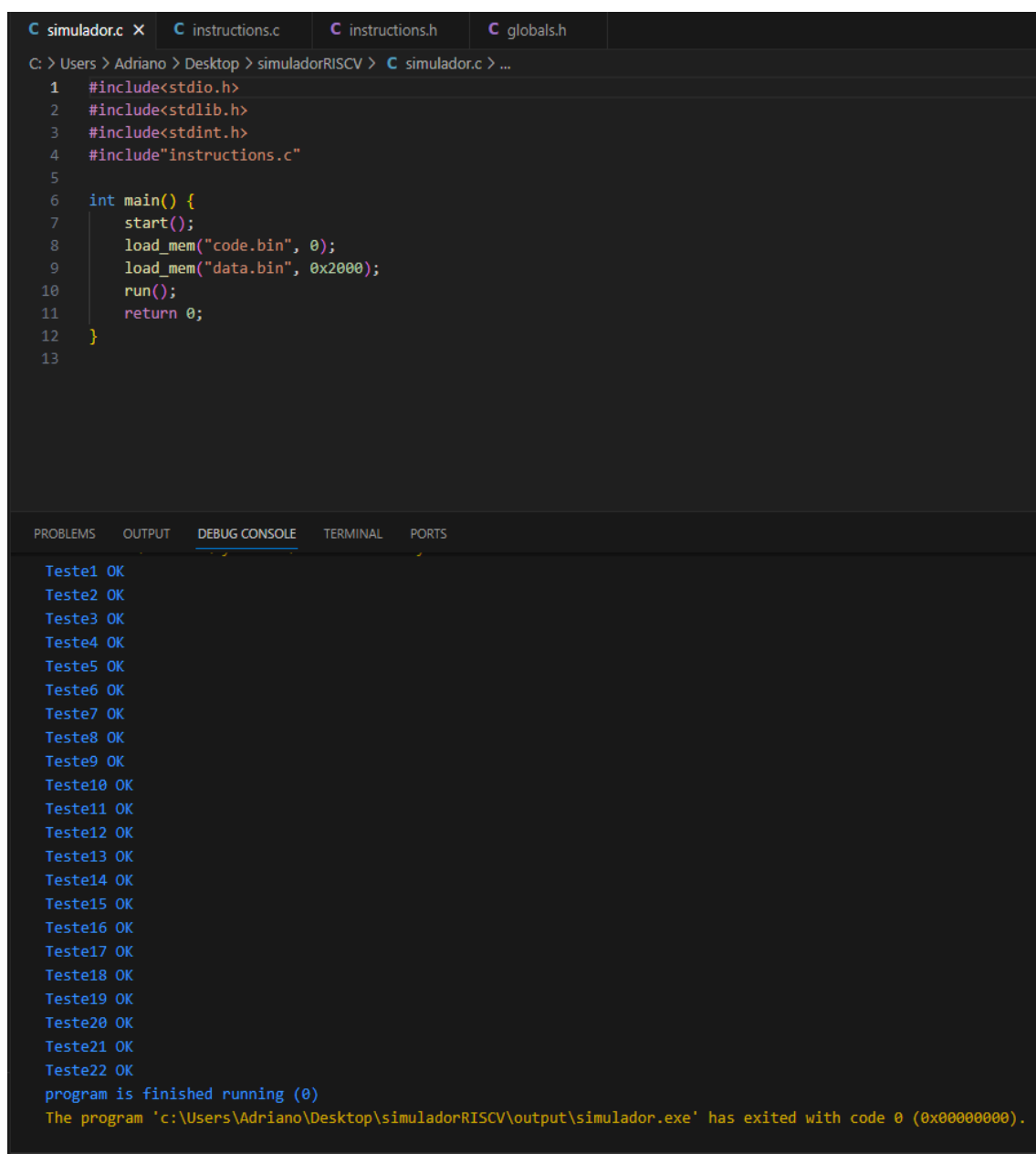
A função *fetch()* busca a instrução a ser executada da memória e atualiza o valor do pc,

A função *decode()* extrai todos os campos da instrução. Para gerar o imediato é chamado a função *geralmm()*, que ao ser executada, produz um valor imediato em 32 bits a partir do código da instrução.

A função *execute()*, toda vez que executada, é atribuído ao registrador x0 o valor 0. A função executa uma instrução de acordo com o *opcode*, *funct3* e *funct7*.

Verificação do Simulador:

Para verificar o simulador, é utilizado o arquivo “*testador.asm*” fornecido junto à tarefa no Moodle. É extraído os arquivos binários *text.bin* e *data.bin* no RARS do arquivo *testador.asm*. Portanto, o simulador empregado executa esses arquivos *.bin*. E fornece os resultados dos testes, conforme na Figura 1.



```
C simulador.c x C instructions.c C instructions.h C globals.h
C: > Users > Adriano > Desktop > simuladorRISCv > C simulador.c > ...
1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<stdint.h>
4  #include"instructions.c"
5
6  int main() {
7      start();
8      load_mem("code.bin", 0);
9      load_mem("data.bin", 0x2000);
10     run();
11     return 0;
12 }
13

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
Teste1 OK
Teste2 OK
Teste3 OK
Teste4 OK
Teste5 OK
Teste6 OK
Teste7 OK
Teste8 OK
Teste9 OK
Teste10 OK
Teste11 OK
Teste12 OK
Teste13 OK
Teste14 OK
Teste15 OK
Teste16 OK
Teste17 OK
Teste18 OK
Teste19 OK
Teste20 OK
Teste21 OK
Teste22 OK
program is finished running (0)
The program 'c:\Users\Adriano\Desktop\simuladorRISCv\output\simulador.exe' has exited with code 0 (0x00000000).
```

Figura 1. Verificação dos testes do arquivo *testador.asm*.

Informações Gerais sobre o código fonte:

- O código pode ser testado e compilado usando a IDE, estando no arquivo principal *simulador.c*
- Compilador empregado: GCC (MinGW.org GCC Build-2) 9.2.0
- Sistema Operacional: Windows 10
- IDE: Visual Studio Code
- Extensões do Visual Studio Code:
 - C/C++ v1.17.5
 - C/C++ Compile Run