

RISC-V-Playground

0.1

Gerado por Doxygen 1.9.1

1 Algoritmo para Simulador RISC-V	1
1.1 Objetivo	1
1.2 Descrevendo a experiência	1
1.2.1 Pontos positivos	1
1.2.2 Pontos negativos	1
1.3 Descrição da dinâmica de trabalho do grupo	1
2 Índice Hierárquico	3
2.1 Hierarquia de Classes	3
3 Índice dos Componentes	5
3.1 Lista de Classes	5
4 Índice dos Arquivos	7
4.1 Lista de Arquivos	7
5 Classes	9
5.1 Referência da Classe ArithmeticLogicUnit	9
5.1.1 Descrição detalhada	9
5.2 Referência da Classe Instruction	9
5.2.1 Descrição detalhada	10
5.2.2 Construtores e Destrutores	10
5.2.2.1 Instruction() [1/2]	10
5.2.2.2 Instruction() [2/2]	10
5.2.3 Funções membros	11
5.2.3.1 getFunc3()	11
5.2.3.2 getOpcode()	11
5.2.3.3 getRegDest()	11
5.2.3.4 getRegSrc1()	12
5.3 Referência da Classe InstructionTypeR	12
5.3.1 Descrição detalhada	12
5.3.2 Construtores e Destrutores	13
5.3.2.1 InstructionTypeR() [1/2]	13
5.3.2.2 InstructionTypeR() [2/2]	13
5.3.3 Funções membros	13
5.3.3.1 instructionToBin()	13
5.4 Referência da Classe MainWindow	14
5.4.1 Descrição detalhada	14
5.5 Referência da Classe Memory	15
5.5.1 Descrição detalhada	15
5.5.2 Funções membros	15
5.5.2.1 get()	15
5.5.2.2 set()	16
5.6 Referência da Classe ProgramCounter	16

5.6.1 Descrição detalhada	16
5.7 Referência da Classe Registers	16
5.7.1 Descrição detalhada	17
5.7.2 Funções membros	17
5.7.2.1 get()	17
5.7.2.2 set()	17
5.8 Referência da Classe Simulator	18
5.8.1 Descrição detalhada	18
6 Arquivos	19
6.1 Referência do Arquivo arithmeticlogicunit.cpp	19
6.1.1 Descrição detalhada	19
6.2 Referência do Arquivo arithmeticlogicunit.h	19
6.2.1 Descrição detalhada	20
6.3 Referência do Arquivo instruction.cpp	20
6.3.1 Descrição detalhada	20
6.4 Referência do Arquivo instruction.h	20
6.4.1 Descrição detalhada	21
6.5 Referência do Arquivo instructiontyper.cpp	21
6.5.1 Descrição detalhada	21
6.6 Referência do Arquivo instructiontyper.h	21
6.6.1 Descrição detalhada	22
6.7 Referência do Arquivo main.cpp	22
6.7.1 Descrição detalhada	22
6.8 Referência do Arquivo memory.cpp	23
6.8.1 Descrição detalhada	23
6.9 Referência do Arquivo memory.h	23
6.9.1 Descrição detalhada	23
6.10 Referência do Arquivo programcounter.cpp	24
6.10.1 Descrição detalhada	24
6.11 Referência do Arquivo programcounter.h	24
6.11.1 Descrição detalhada	24
6.12 Referência do Arquivo registers.cpp	24
6.12.1 Descrição detalhada	25
6.13 Referência do Arquivo registers.h	25
6.13.1 Descrição detalhada	25
6.14 Referência do Arquivo simulator.cpp	25
6.14.1 Descrição detalhada	26
6.15 Referência do Arquivo simulator.h	26
6.15.1 Descrição detalhada	26
Índice Remissivo	27

Capítulo 1

Algoritmo para Simulador RISC-V

1.1 Objetivo

Aplicativo tem como objetivo simular o comportamento de uma máquina RISC-V.

Considerando um conjunto limitado de instruções:

- ADDI
- ADD
- SUB
- AND
- OR
- LW
- SW
- BEQ
- BNE

1.2 Descrevendo a experiência

Escrever algo aqui

1.2.1 Pontos positivos

- Alguma coisa

1.2.2 Pontos negativos

- Alguma coisa

1.3 Descrição da dinâmica de trabalho do grupo

Dentro da documentação de cada método e classe possui o nome do author

Capítulo 2

Índice Hierárquico

2.1 Hierarquia de Classes

Esta lista de hierarquias está parcialmente ordenada (ordem alfabética):

ArithmeticLogicUnit	9
Instruction	9
InstructionTypeR	12
Memory	15
ProgramCounter	16
QMainWindow	
MainWindow	14
Registers	16
Simulator	18

Capítulo 3

Índice dos Componentes

3.1 Lista de Classes

Aqui estão as classes, estruturas, uniões e interfaces e suas respectivas descrições:

ArithmeticLogicUnit	
A classe ArithmeticLogicUnit :	9
Instruction	
A classe Instruction : Classe abstrata a qual é base para todos os modos de instrução	9
InstructionTypeR	
A classe InstructionTypeR :	12
MainWindow	14
Memory	
A classe Memory :	15
ProgramCounter	
A classe ProgramCounter :	16
Registers	
A classe Registers :	16
Simulator	
A classe Simulator :	18

Capítulo 4

Índice dos Arquivos

4.1 Lista de Arquivos

Esta é a lista de todos os arquivos documentados e suas respectivas descrições:

arithmeticlogicunit.cpp	Arquivo que implementa os métodos da classe ArithmeticLogicUnit	19
arithmeticlogicunit.h	Arquivo que define a classe ArithmeticLogicUnit	19
instruction.cpp	Arquivo que implementa os métodos da classe Instruction	20
instruction.h	Arquivo que define a classe Instruction	20
instructiontyper.cpp	Arquivo que implementa os métodos da classe InstructionTypeR	21
instructiontyper.h	Arquivo que define a classe InstructionTypeR	21
main.cpp	Arquivo principal	22
mainwindow.cpp	??
mainwindow.h	??
memory.cpp	Arquivo que implementa os métodos da classe Memory	23
memory.h	Arquivo que define a classe Memory	23
programcounter.cpp	Arquivo que implementa os métodos da classe ProgramCounter	24
programcounter.h	Arquivo que define a classe ProgramCounter	24
registers.cpp	Arquivo que implementa os métodos da classe Registers	24
registers.h	Arquivo que define a classe Registers	25
simulator.cpp	Arquivo que implementa os métodos da classe Simulator	25
simulator.h	Arquivo que define a classe Simulator	26

Capítulo 5

Classes

5.1 Referência da Classe ArithmeticLogicUnit

A classe [ArithmeticLogicUnit](#):

```
#include <arithmeticlogicunit.h>
```

5.1.1 Descrição detalhada

A classe [ArithmeticLogicUnit](#):

Definição na linha 14 do arquivo arithmeticlogicunit.h.

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

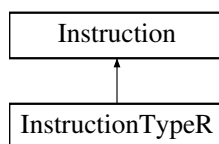
- [arithmeticlogicunit.h](#)
- [arithmeticlogicunit.cpp](#)

5.2 Referência da Classe Instruction

A classe [Instruction](#): Classe abstrata a qual é base para todos os modos de instrução.

```
#include <instruction.h>
```

Diagrama de hierarquia para Instruction:



Membros Públicos

- `Instruction ()`
Inicializa todos os campos com zero.
- `Instruction (int op, int rd, int f3, int rs1)`
- `int getOpcode ()`
Obtêm o valor do opcode para alguém fora da classe.
- `int getRegDest ()`
Obtêm o valor do registrador destino para alguém fora da classe.
- `int getFunc3 ()`
Obtêm o valor do Funct 3 para alguém fora da classe.
- `int getRegSrc1 ()`
Obtêm o valor do registrador source 1 para alguém fora da classe.
- `virtual void printInfo ()=0`
- `virtual bool * instructionToBin ()=0`

5.2.1 Descrição detalhada

A classe `Instruction`: Classe abstrata a qual é base para todos os modos de instrução.

Definição na linha 32 do arquivo `instruction.h`.

5.2.2 Construtores e Destrutores

5.2.2.1 `Instruction()` [1/2]

```
Instruction::Instruction ( )
```

Inicializa todos os campos com zero.

Vamos colocar algo mais detalhado

Definição na linha 15 do arquivo `instruction.cpp`.

5.2.2.2 `Instruction()` [2/2]

```
Instruction::Instruction (
    int op,
    int rd,
    int f3,
    int rs1 )
```

Esse é um método provido por conveniência. Ele difere do método acima apenas na lista de argumentos que devem ser utilizados.

Parâmetros

<i>op</i>	valor do opcode
<i>rd</i>	valor do registrador destino
<i>f3</i>	valor do funct 3
<i>rs1</i>	valor do registrador fonte 1

Definição na linha 31 do arquivo instruction.cpp.

5.2.3 Funções membros

5.2.3.1 getFunc3()

```
int Instruction::getFunc3 ( )
```

Obtém o valor do Funct 3 para alguém fora da classe.

Retorna

o valor do funct3

Definição na linha 63 do arquivo instruction.cpp.

5.2.3.2 getOpcode()

```
int Instruction::getOpcode ( )
```

Obtém o valor do opcode para alguém fora da classe.

Retorna

o valor do opcode

Definição na linha 43 do arquivo instruction.cpp.

5.2.3.3 getRegDest()

```
int Instruction::getRegDest ( )
```

Obtém o valor do registrador destino para alguém fora da classe.

Retorna

o valor do registrador de destino

Definição na linha 53 do arquivo instruction.cpp.

5.2.3.4 getRegSrc1()

```
int Instruction::getRegSrc1 ( )
```

Obtém o valor do registrador source 1 para alguém fora da classe.

Retorna

valor do registrador fonte 1

Definição na linha 72 do arquivo instruction.cpp.

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

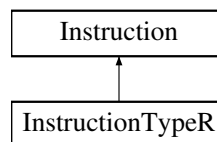
- [instruction.h](#)
- [instruction.cpp](#)

5.3 Referência da Classe InstructionTypeR

A classe [InstructionTypeR](#):

```
#include <instructiontyper.h>
```

Diagrama de hierarquia para InstructionTypeR:



Membros Públicos

- [InstructionTypeR](#) ()
Inicializa todos os valores com zero.
- [InstructionTypeR](#) (int op, int rd, int f3, int rs1, int rs2, int f7)
- bool * [instructionToBin](#) ()
Cria um vetor contendo o valor binário da instrução.
- void [printInfo](#) ()
Imprime na tela todas as informações dos membros da classe.

5.3.1 Descrição detalhada

A classe [InstructionTypeR](#):

Definição na linha 19 do arquivo instructiontyper.h.

5.3.2 Construtores e Destrutores

5.3.2.1 InstructionTypeR() [1/2]

```
InstructionTypeR::InstructionTypeR ( )
```

Inicializa todos os valores com zero.

Explicar mais (Se remover isso quebra)

Definição na linha 15 do arquivo instructiontyper.cpp.

5.3.2.2 InstructionTypeR() [2/2]

```
InstructionTypeR::InstructionTypeR (
    int op,
    int rd,
    int f3,
    int rs1,
    int rs2,
    int f7 )
```

Esse é um método provido por conveniência. Ele difere do método acima apenas na lista de argumentos que devem ser utilizados.

Parâmetros

<i>op</i>	valor do opcode
<i>rd</i>	valor do registrador destino
<i>f3</i>	valor do funct 3
<i>rs1</i>	valor do registrador fonte 1
<i>rs2</i>	valor do registrador fonte 1
<i>f7</i>	valor do funct 7

Definição na linha 31 do arquivo instructiontyper.cpp.

5.3.3 Funções membros

5.3.3.1 instructionToBin()

```
bool * InstructionTypeR::instructionToBin ( ) [virtual]
```

Cria um vetor contendo o valor binário da instrução.

Dá para escrever mais aqui

Aviso

o vetor dinâmico de valores booleanos deve ser desalocado por quem chamou.

Pré-condição

Todos os membros da classe devem estar inicializados.

Pós-condição

cria o vetor dinâmico de valores booleanos, contendo o valor binário da instrução.

Retorna

vetor de valores booleanos, contendo o valor binário da instrução.

Implementa [Instruction](#).

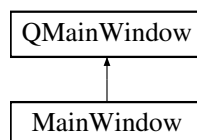
Definição na linha 46 do arquivo `instructiontyper.cpp`.

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- [instructiontyper.h](#)
- [instructiontyper.cpp](#)

5.4 Referência da Classe MainWindow

Diagrama de hierarquia para MainWindow:



Membros Públicos

- **MainWindow** (QWidget *parent=nullptr)

5.4.1 Descrição detalhada

Definição na linha 10 do arquivo `mainwindow.h`.

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- `mainwindow.h`
- `mainwindow.cpp`

5.5 Referência da Classe Memory

A classe `Memory`:

```
#include <memory.h>
```

Membros Públicos

- `Memory ()`
Inicializa todas as células de memória com zero.
- `int get (int position)`
Lê o valor contido na memória.
- `void set (int position, int value)`
Escreve o valor passado por referência na memória.

5.5.1 Descrição detalhada

A classe `Memory`:

Definição na linha 16 do arquivo `memory.h`.

5.5.2 Funções membros

5.5.2.1 `get()`

```
int Memory::get (  
    int position )
```

Lê o valor contido na memória.

Parâmetros

<i>position</i>	indica o registrador a ser lido
-----------------	---------------------------------

Retorna

o valor contido na memória na posição[`position`]

Definição na linha 25 do arquivo `memory.cpp`.

5.5.2.2 set()

```
void Memory::set (
    int position,
    int value )
```

Escreve o valor passado por referência na memória.

Parâmetros

<i>position</i>	indica o registrador a ser lido
<i>value</i>	valor a ser escrito

Definição na linha 35 do arquivo memory.cpp.

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- [memory.h](#)
- [memory.cpp](#)

5.6 Referência da Classe ProgramCounter

A classe [ProgramCounter](#):

```
#include <programcounter.h>
```

5.6.1 Descrição detalhada

A classe [ProgramCounter](#):

Definição na linha 14 do arquivo programcounter.h.

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- [programcounter.h](#)
- [programcounter.cpp](#)

5.7 Referência da Classe Registers

A classe [Registers](#):

```
#include <registers.h>
```

Membros Públicos

- [Registers](#) ()
Inicializa todos os registradores com zero.
- `int` [get](#) (`int position`)
Lê o valor contido em um registrador.
- `void` [set](#) (`int position`, `int value`)
Escreve o valor passado por referência em um registrador.

5.7.1 Descrição detalhada

A classe [Registers](#):

Definição na linha 16 do arquivo registers.h.

5.7.2 Funções membros

5.7.2.1 get()

```
int Registers::get (  
    int position )
```

Lê o valor contido em um registrador.

Parâmetros

<i>position</i>	indica o registrador a ser lido
-----------------	---------------------------------

Retorna

o valor contido no registrador `x[position]`

Definição na linha 25 do arquivo registers.cpp.

5.7.2.2 set()

```
void Registers::set (  
    int position,  
    int value )
```

Escreve o valor passado por referência em um registrador.

Parâmetros

<i>position</i>	indica o registrador a ser lido
<i>value</i>	valor a ser escrito

Definição na linha 35 do arquivo registers.cpp.

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- [registers.h](#)
- [registers.cpp](#)

5.8 Referência da Classe Simulator

A classe [Simulator](#):

```
#include <simulator.h>
```

5.8.1 Descrição detalhada

A classe [Simulator](#):

Definição na linha 21 do arquivo simulator.h.

A documentação para essa classe foi gerada a partir dos seguintes arquivos:

- [simulator.h](#)
- [simulator.cpp](#)

Capítulo 6

Arquivos

6.1 Referência do Arquivo arithmeticlogicunit.cpp

Arquivo que implementa os métodos da classe [ArithmeticLogicUnit](#).

```
#include "arithmeticlogicunit.h"
```

6.1.1 Descrição detalhada

Arquivo que implementa os métodos da classe [ArithmeticLogicUnit](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.2 Referência do Arquivo arithmeticlogicunit.h

Arquivo que define a classe [ArithmeticLogicUnit](#).

Componentes

- class [ArithmeticLogicUnit](#)

A classe [ArithmeticLogicUnit](#):

6.2.1 Descrição detalhada

Arquivo que define a classe [ArithmeticLogicUnit](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.3 Referência do Arquivo instruction.cpp

Arquivo que implementa os métodos da classe [Instruction](#).

```
#include "instruction.h"
```

6.3.1 Descrição detalhada

Arquivo que implementa os métodos da classe [Instruction](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.4 Referência do Arquivo instruction.h

Arquivo que define a classe [Instruction](#).

```
#include <iostream>
#include <cstdlib>
```

Componentes

- class [Instruction](#)

A classe [Instruction](#): Classe abstrata a qual é base para todos os modos de instrução.

Definições e Macros

- `#define OP_SIZE 7`
- `#define RD_SIZE 5`
- `#define F3_SIZE 3`
- `#define RS1_SIZE 5`
- `#define OP_BEGIN 0`
- `#define RD_BEGIN 7`
- `#define F3_BEGIN 12`
- `#define RS1_BEGIN 15`
- `#define INSTRUCTION_SIZE 32`

6.4.1 Descrição detalhada

Arquivo que define a classe [Instruction](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.5 Referência do Arquivo instructiontyper.cpp

Arquivo que implementa os métodos da classe [InstructionTypeR](#).

```
#include "instructiontyper.h"
```

6.5.1 Descrição detalhada

Arquivo que implementa os métodos da classe [InstructionTypeR](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.6 Referência do Arquivo instructiontyper.h

Arquivo que define a classe [InstructionTypeR](#).

```
#include "instruction.h"
```

Componentes

- class [InstructionTypeR](#)
A classe [InstructionTypeR](#):

Definições e Macros

- #define **RS2_BEGIN** 20
- #define **F7_BEGIN** 25

6.6.1 Descrição detalhada

Arquivo que define a classe [InstructionTypeR](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.7 Referência do Arquivo main.cpp

Arquivo principal.

```
#include "mainwindow.h"
#include "instruction.h"
#include "instructiontyper.h"
#include <QApplication>
#include <QFile>
```

Funções

- int **main** (int argc, char *argv[])

6.7.1 Descrição detalhada

Arquivo principal.

Autor

Victor Emanuel Almeida

Versão

0.1

6.8 Referência do Arquivo memory.cpp

Arquivo que implementa os métodos da classe [Memory](#).

```
#include "memory.h"
```

6.8.1 Descrição detalhada

Arquivo que implementa os métodos da classe [Memory](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.9 Referência do Arquivo memory.h

Arquivo que define a classe [Memory](#).

Componentes

- class [Memory](#)
A classe [Memory](#):

Definições e Macros

- #define **MEMORY_SIZE** 1024

6.9.1 Descrição detalhada

Arquivo que define a classe [Memory](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.10 Referência do Arquivo programcounter.cpp

Arquivo que implementa os métodos da classe [ProgramCounter](#).

```
#include "programcounter.h"
```

6.10.1 Descrição detalhada

Arquivo que implementa os métodos da classe [ProgramCounter](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.11 Referência do Arquivo programcounter.h

Arquivo que define a classe [ProgramCounter](#).

Componentes

- class [ProgramCounter](#)
A classe [ProgramCounter](#):

6.11.1 Descrição detalhada

Arquivo que define a classe [ProgramCounter](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.12 Referência do Arquivo registers.cpp

Arquivo que implementa os métodos da classe [Registers](#).

```
#include "registers.h"
```

6.12.1 Descrição detalhada

Arquivo que implementa os métodos da classe [Registers](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.13 Referência do Arquivo registers.h

Arquivo que define a classe [Registers](#).

Componentes

- class [Registers](#)

A classe [Registers](#):

Definições e Macros

- `#define QUANTITY_REGISTERS 32`

6.13.1 Descrição detalhada

Arquivo que define a classe [Registers](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.14 Referência do Arquivo simulator.cpp

Arquivo que implementa os métodos da classe [Simulator](#).

```
#include "simulator.h"
```

6.14.1 Descrição detalhada

Arquivo que implementa os métodos da classe [Simulator](#).

Autor

Victor Emanuel Almeida

Versão

0.1

6.15 Referência do Arquivo simulator.h

Arquivo que define a classe [Simulator](#).

```
#include "arithmeticlogicunit.h"
#include "instruction.h"
#include "instructiontyper.h"
#include "memory.h"
#include "programcounter.h"
#include "registers.h"
```

Componentes

- class [Simulator](#)

A classe [Simulator](#):

6.15.1 Descrição detalhada

Arquivo que define a classe [Simulator](#).

Autor

Victor Emanuel Almeida

Versão

0.1

Índice Remissivo

ArithmeticLogicUnit, [9](#)
arithmeticlogicunit.cpp, [19](#)
arithmeticlogicunit.h, [19](#)

get
 Memory, [15](#)
 Registers, [17](#)
getFunc3
 Instruction, [11](#)
getOpcode
 Instruction, [11](#)
getRegDest
 Instruction, [11](#)
getRegSrc1
 Instruction, [11](#)

Instruction, [9](#)
 getFunc3, [11](#)
 getOpcode, [11](#)
 getRegDest, [11](#)
 getRegSrc1, [11](#)
 Instruction, [10](#)
instruction.cpp, [20](#)
instruction.h, [20](#)
instructionToBin
 InstructionTypeR, [13](#)
InstructionTypeR, [12](#)
 instructionToBin, [13](#)
 InstructionTypeR, [13](#)
instructiontyper.cpp, [21](#)
instructiontyper.h, [21](#)

main.cpp, [22](#)
MainWindow, [14](#)
Memory, [15](#)
 get, [15](#)
 set, [15](#)
memory.cpp, [23](#)
memory.h, [23](#)

ProgramCounter, [16](#)
programcounter.cpp, [24](#)
programcounter.h, [24](#)

Registers, [16](#)
 get, [17](#)
 set, [17](#)
registers.cpp, [24](#)
registers.h, [25](#)

set

Memory, [15](#)
 Registers, [17](#)
Simulator, [18](#)
simulator.cpp, [25](#)
simulator.h, [26](#)