

Chisel 初体验 67-20



Chisel 初步探索与实验

PLCT Lab 每周技术分享

熊家辉 浙江工商大学 2024年2月21日



Chisel 初体验 67-Verilog 语言简介 2004-70

<sup>第1节</sup> Verilog 语言简介

第1节

Verilog 语言简介

第1节 Verilog 语言简介 第11小节 Verilog 语言

第1节

Verilog 语言简介

第 1.1 小节

Verilog 语言

**◆ロ▶ ◆昼▶ ◆昼▶ ◆昼▶ 昼 り**�� 4/54

## Verilog 语言

- Verilog 是一种用于描述、设计电子系统(特别是数字电路)的硬件描述 主要用于在集成电路设计,特别是超大规模集成电路的计算机辅助设计。
- Verilog 是电气电子工程师学会(IEEE)的 1800-2009 号标准。
- Verilog 的设计初衷是成为一种基本语法与 C 语言相近的硬件描述语

◆□▶ ◆□▶ ◆□▶ ◆□▶ ■ りゅ○ 5/54

Chisel 初体验 2024-02-29 -Verilog 语言简介 └─Verilog 语言 └─Verilog 语言

Verilog 语言

2024-02-29

#### Verilog 代码示例

Verilog 语言

Chisel 初体验

```
一段由 Chisel 吐出的 Verilog 代码
 module ModuleSample(
              clock, reset.
   input
   input [7:0] io_a,io_b,
   output [7:0] io_minnum,io_maxnum
   wire _io_maxnum_T = io_a <= io_b;</pre>
   assign io_minnum = _io_maxnum_T ? io_a : io_b;
   assign io_maxnum = _io_maxnum_T ? io_b : io_a;
9 endmodule
```

∢□▶ ∢□▶ ∢□▶ ∢□▶ ▼□ ♥ ♀○ 6/54

萨塞克斯人工智能学院

Chisel 初体验 Verilog 代码示例 -Verilog 语言简介 module ModuleSample( input clock,reset, input [7:0] io\_a,io\_b, output [7:0] io minnum.io maxnum └─Verilog 语言 └─Verilog 代码示例 assign io\_minnum = \_io\_maxnum\_T 7 io\_a : io\_b; aggier to maxeum = to maxeum T 7 to b : to a:

Chisel 初体验 ─Verilog 语言简介 ─运行 Verilog

第1节 Verilog 语言简介 第12小节 运行 Verilog

第1节

# Verilog 语言简介

第 1.2 小节

运行 Verilog

◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 ∽Q♀ 7/54

Verilog 语言简介 ○○○○○●○○○ 运行 Verilog

#### Verilator 比较

#### Verilator

- 接受 Verilog 或 SystemVerilog。
- 执行 lint 代码质量检查。
- 编译为多线程 C++ 或 SystemC。
- 优于许多闭源商业模拟器,单线程和 多线程输出模型。
- 最广泛的使用。

# 1 iverilog

- 接受 Verilog 或 SystemVerilog (不完 全支持)。
- 广泛的兼容性(包括 Windows®)。
- 部分指令支持不完全。(见 Unsupported Constructs)
- 速度较慢。

2024-03

Chisel 初体验 67-70-Verilog 语言简介 └─ 运行 Verilog └─ Verilator 比较

Chisel 初体验 萨塞克斯人工智能学院

#### Verilator 简介

#### 提示

该报告的重点为 Chisel 相关内容, Verilator 相关内容仅供参考。

- 用途 Verilator 是一款开源的支持 Verilog 和 System Verilog 仿真工具,它支持 代码质量检查等功能,能够将给定的电路设计翻译成 C++ 或者 System C 的库 等中间文件,最后使用 C/C++ 编写 testbench ,去调用前面生成的中间文件, 由 C 编译器编译执行,来完成仿真。
- 主要功能 Verilator 的主要功能就是将 Verilog 代码转化为 SystemC 或 C++ 代 码。

◆□▶ ◆問▶ ◆臺▶ ◆臺▶ 臺 ♡Q♡ 9/54

Chisel 初体验 -Verilog 语言简介 └─运行 Verilog └─Verilator 简介

Verilog 语言简介 Chisel 和 RISC-V Scala 简介 Chisel 简介

运行 Verilog

#### Verilator 工作原理

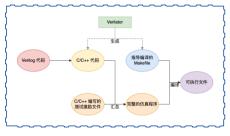


图: Verilator 工作原理

#### Verilator 工作流程

- 硬件设计转化为一个处理器模拟器的 软件。
- ② 使用 C/C++ 编写激励文件。
- 3 利用 GCC 等编译器将生成的 C/C++ 文件和我们编写的激励文件编译成成 用于仿真的可执行文件。

**◆ロ > ◆御 > ◆草 > ◆草 > 草 り**�� 10/54

Chisel 初体验 -Verilog 语言简介 └─运行 Verilog └─Verilator 工作原理



- 1. 首先, Verilator 将 Verilog 代码中并行的各个逻辑部件以合适的顺序串行化, 使硬件设 计转化为一个类似于处理器模拟器的软件;
- 2. 接着, 我们需要使用 C/C++ 编写激励文件。Verilator 为我们提供了顶层模块输入/输出 引脚的接口,使我们得以对顶层模块的输入信号赋值或读取其输出信号。与使用 Vivado 仿真不同的是,我们可以实时地输出一些调试信息;要查看波形图时,需要将波形图导 出到文件:
- 3. 最后, Verilator 会生成一个 Makefile 脚本, 利用 GCC 等编译器将生成的 C/C++ 文件 和我们编写的激励文件编译成成用于仿真的可执行文件。

Chisel 初体验 萨塞克斯人工智能学院

Chisel 和 RISC-V Scala 简介 Chisel 简介

Verilog 语言简介 运行 Verilog

#### Verilator 激励代码

节选自 Verilator 官方激励代码示例。

```
节选
 VerilatedContext* const contextp = new VerilatedContext;
 contextp->commandArgs(argc, argv);
3 Vtop* const top = new Vtop{contextp};
4 while (!contextp->gotFinish()) {
     top->eval();
 top->final();
```

**◆ロ > ◆御 > ◆草 > ◆草 > 草 ・ 夕**Qで 11/54

Chisel 初体验 └─Verilog 语言简介 └─运行 Verilog └─Verilator 激励代码

Verilator 激励代码 节读自 Verilator 官方激励代码示例 vtop\* const top = new Vtop(contextp
while (!contextp->gotFinish()) {

- 1. 构造一个 VerilatedContext 来保存模拟时间等。
- 2. 传递参数,以便经过验证的代码可以看到它们,在创建任何模型之前需要调用它.
- 3. 根据 Verilating "top.v" 生成的 Vtop.h 构建 Verilated 模型
- 4. 模拟直到finish
- 5. 评估模型
- 6. 最终模型清理

萨塞克斯人工智能学院 Chisel 初体验

Chisel 初体验 └─Scala 简介

第2节 Scala **简介** 

# 第2节 **Scala 简介**

◆ロト ◆母 ト ◆ 章 ト ◆ 章 ト 章 りゅつ 12/54

Chisel 初体验 └─Scala 简介 └─Scala 语言简介

第2节 Scala 简介 第2.1 小节 Scala 语言简介

第2节

Scala 简介

第 2.1 小节

Scala 语言简介

◆□ ▶ ◆□ ▶ ◆□ ▶ □ ◆○ \$\text{13/54}

Chisel 初体验 萨塞克斯人工智能学院

Scala 简介

Chisel 和 RISC-V

#### Scala 语言简介 Scala 语言

Verilog 语言简介

- 1 目的 Scala 是一门多范式的编程语言,设计初衷是要集成面向对象编程和函数 式编程的各种特性。
- 2 编译 Scala 运行于 Java 平台(Java 虚拟机),并兼容现有的 Java 程序。
- Scala是一种纯面向对象的语言、函数式语言、强静态类型语言。

#### 语言提示

scale 一词指规模、尺度、比例,而 scala 没有意思。请在搜索时注意拼写。

**◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 ♡९♡ 14/54** 

2024-02-29

Chisel 初体验 -Scala 简介 —Scala 语言简介 └─Scala 语言

Scala 语言 いた 一切後収穫、尺度、比例、而 いた 没有音楽、清在探索財注音拼写、 Scala 语言简介

### Scala 代码示例

```
Hullo World!

1 object HelloWorld extends App {
2    println("Hello, world!")
3 }
```

萨塞克斯人工智能学院



Chisel 初体验 67-70-8cala 简介 —Scala 实践

第2节 Scala 简介 第22小节 Scala 实践

第2节

Scala 简介

第 2.2 小节

Scala 实践

◆□ ▶ ◆□ ▶ ◆ □ ▶ ◆ □ ▶ ○
 ○○ 16/54

Chisel 初体验 萨塞克斯人工智能学院

Scale 安装

参考 Installation。

- Java Development Kit: 安装 openjdk-21-jdk 包。
- SBT:安装 sbt 包。

◆ロト ◆御ト ◆草ト ◆草ト 草 からで 17/54

Chisel 初体验 2024-02-29 └─Scala 简介 └─Scala 实践 ─Scale 安装

Scale 安装

参考 Installation。 • Java Development Kit: 安装 openjdk-21-jdk 包。 • SBT: 安装 abt 包。

Chisel 初体验 萨塞克斯人工智能学院 Scala 实践

#### Scala 编译

将代码示例存入 HelloWorld.scala。运行下列命令编译。

#### 编译 Scala 源代码

- 1 scalac HelloWorld.scala // 把源码编译为字节码
- 2 scala HelloWorld // 把字节码放到虚拟机中解释运行

Scala **集等** 株代研究教介入 Relieferid, scala, 近行下列を含葉は、 単作 Scala 現代同 in scala 現代同 in scala Relieferid anala // ビ皮明線を含字可可 in scala Relieferid anala // ビ皮明線を含す可可 in scala Relieferid anala // ビ皮明線の表現を in scala Relieferid anala // ビ皮明線の表現を in scala Relieferid anala // ビ皮明線を in scala Relieferid anala // Excelation anal

Chisel 初体验
—Scala 简介
—Scala Test

第2节 Scala 簡介 第23小节 ScalaTest

第2节

Scala 简介

第 2.3 小节

ScalaTest

Scala 简介 Chisel 简介

Verilog 语言简介 ScalaTest

#### ScaleTest 简介

Scala Test 是 Scala 生态系统中最灵活、最受欢迎的测试工具。 与构建它的 Scala 语言一样, Scala Test 旨在随着用户的需求而增长: 您可以轻松扩 展和组合 Scala Test 的核心组件,以满足您可能有的任何特殊需求。因此, ScalaTest 可以扩展到各种规模的项目,从探索新想法的个人到协作开发关键任务软 件的大型团队。

ScalaTest 缩小规模的一种方式是,尽管ScalaTest 具有丰富的功能集,但它很容易 上手。基于您从其他测试框架的经验中获得的知识,您可以通过 ScalaTest快速提高 工作效率。

◆□▶ ◆□▶ ◆□▶ ◆□▶ □ 夕○○ 20/54

Chisel 和 RISC-V

Chisel 初体验 -Scala 简介 -ScalaTest └─ScaleTest 简介

ScaleTest 简介

Chisel 初体验 萨塞克斯人工智能学院 Verilog 语言简介

#### ScaleTest 代码示例

在包含了 "org.scalatest" % "scalatest" % "3.2.18" % "test" 库后,可以使用下列代码进行测试。

```
Hullo ScalaTest!

1 class ExampleTest extends AnyFlatSpec with Matchers {
2    "Integers" should "add" in {
3         val i = 2
4         val j = 3
5         i + j should be (5)
6     }
7 }
```

```
◆ロ▶ ◆母▶ ◆臺▶ ◆臺▶ 臺 か�� 21/54
```

Chisel 初体验 萨塞克斯人工智能学院



Chisel 初体验 Chisel 简介

第3节 Chisel 简介

第3节 **Chisel 简介** 

Chisel 初体验 Chisel 简介 Chisel 库简介

第3节 Chisel 简介 第31小节 Chisel 库简介

第3节

Chisel 简介

第 3.1 小节

Chisel 库简介

Verilog 语言简介 ○○○○○○○○ Chisel 库简介

#### Chisel 简介

- 1 用途 Constructing Hardware in a Scala Embedded Language (Chisel) 是一种开源 硬件描述语言,用于在寄存器传输级别描述数字电子设备和电路。
- ② 语言 Chisel 将硬件构造添加到 Scala 编程语言中。
- 3 目标 Chisel 可以编写复杂的、可参数化的电路生成器,从而生成可综合的 Verilog。
- 4 手段 Chisel 可以创建可重用的组件和库,提高了设计的抽象级别,同时保留了细粒度的控制。

Chisel 初体验 67-70 ← Chisel 简介 ← Chisel 库简介 ← Chisel 简介



Chisel 和 RISC-V

Verilog 语言简介 ○○○○○○○○ Chisel 库简介

#### Chisel 代码示例

以下是来自 LED blink 的示例。

```
LED blink
 class Blinky(freq: Int, startOn: Boolean = false) extends Module {
     val io = IO(new Bundle {val led0 = Output(Bool())})
     val led = RegInit(startOn.B)
     val (_, counterWrap) = Counter(true.B, freq / 2)
     when(counterWrap) {
         led := ~led
     io.led0 := led
9
```

```
◀□▶◀圖▶◀臺▶◀臺▶ 臺 ∽Qⓒ 25/54
```

Chisel 初体验 萨塞克斯人工智能学院

Chisel 初体验 Chisel 简介 Chisel 库简介 └─Chisel 代码示例

# Chical 代码示例 CT是集员 LED Mode, 例示例。 LED Mode (The American Control Con

Chisel 库简介

#### Chisel 输出示例

来自 LED blink 的示例可以生成如下 Verilog 代码。

```
LED blink 的 Verilog 代码节选
```

```
if (reset) begin led <= 1'h0; counterWrap_c_value <= 9'h0; end</pre>
```

- 2 else begin
- automatic logic counterWrap = counterWrap\_c\_value == 9'h1F3; led <=
   counterWrap ^ led;</pre>
- 4 if (counterWrap) counterWrap\_c\_value <= 9'h0; else counterWrap\_c\_value
   <= counterWrap\_c\_value + 9'h1;</pre>
- 5 end

```
◆ロ▶ ◆母▶ ◆夏▶ ◆夏▶ 夏 からで 26/54
```

Chisel 初体验 67-70 ← Chisel 简介 Chisel 库简介 Chisel 输出示例

Chieck 輸出示例

素質 LED wink 形示例可述或如下 Verlag 代码。

LED Man W Verlag 代码可见。

Lift (Formal Verlag 16 of 12%; constratings, value < 9700; cod

1 size legits

constrating - Legit constrating - constratings, value < 97007; led <constrainty - Legit constrainty - constrainty, value = 97007; led <constrainty - Legit constrainty - constrainty, value = 97007; led <-

Chisel 初体验

萨塞克斯人工智能学院

Chisel 初体验 —Chisel 简介 —Chisel 实践

第3节 Chisel 简介 第32小节 Chisel 实践

第3节

Chisel 简介

第 3.2 小节

Chisel 实践

Scala 简介 0000000000 Chisel 简介 ○○○○○●○○○○○○○○○○○○○

Verilog 语言简介 000000000 Chisel 实践

#### Chisel 安装

参考 Installation。

- Java Development Kit: 安装 openjdk-21-jdk 包。
- SBT:安装 sbt 包。
- Verilator: 安装 verilator 包。

**◆ロ▶ ◆御▶ ◆夏▶ ◆夏▶ 夏 か**�� 28/54

Chisel 初体验 2024-02-29 -Chisel 简介

Chisel 安装

Chisel 安装

• Java Development Kit: 安装 openjdk-21-jdk 包。 • SBT: 安装 abt 包。 • Verilator: 安装 verilator 包。

Chisel 初体验 萨塞克斯人工智能学院 Scala 简介 Chisel 简介 ○○○○○○○○

Chisel 和 RISC-V 00000000

 Verilog 语言简介
 Sc

 000000000
 00

 Chisel 实践
 00

#### Chisel 编译

使用 Chisel Project Template 作为模板,可以快速开始设计硬件。 在目录中运行 sbt test 可以测试当前安装是否正确。环境正常时,测试应当通过。 在目录中运行 sbt run 可以运行主程序。

2024.02.20

Chisel 初体验 67-70 Chisel 简介 Chisel 实践 Chisel 编译

Chisel 编译

el Project Template 作为模板,可以快速开始设计硬件。 运行 abt teat 可以测试高前安装是否正确。环境正常时,测试应高通过。 运行 abt rus 可以运行主程序。

Chisel 初体验

#### Chisel 基础语法 I

```
GCD.scala
  class ModuleSample extends Module {
    val io = IO(new Bundle {
     val a = Input(UInt(8.W))
     val b = Input(UInt(8.W))
     val minnum = Output(UInt(8.W))
     val maxnum = Output(UInt(8.W))
    })
    io.minnum := Mux(io.a <= io.b, io.a, io.b)</pre>
    io.maxnum := Mux(io.a <= io.b, io.b, io.a)</pre>
10
```

```
◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 ∽Q♡ 30/54
```

Chisel 初体验 2024-02-29 -Chisel 简介 └─Chisel 实践 └─Chisel 基础语法 Chisel 基础语法 I val a = Input(UInt(8.W)) val b = Input(UInt(8.W)) val minnum = Dotnot (HTat (S W)) val maxnum = Dutput(UInt(8.W)) Chisel 实践

Chisel 初体验

Verilog 语言简介

#### Chisel 基础语法 II

上述代码可以运行 sbt run 生成对应的 SystemVerilog 代码。

```
GCD.scala 对应 SystemVerilog 代码节选
module ModuleSample(
             clock, reset,
  input
 input [7:0] io_a,io_b,
  output [7:0] io_minnum,io_maxnum
 wire _io_maxnum_T = io_a <= io_b;</pre>
  assign io_minnum = _io_maxnum_T ? io_a : io_b;
  assign io_maxnum = _io_maxnum_T ? io_b : io_a;
endmodule
```

```
◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 ∽ 90° 31/54
```

萨塞克斯人工智能学院

2024-02-29

Chisel 初体验

Chisel 简介

Chisel 实践

Chisel 基础语法

Chisel 初体验 Chisel 简介 Chisel 设计验证

第3节 Chisel 简介 第33小节 Chisel 设计验证

第3节

# Chisel 简介

第 3.3 小节

Chisel 设计验证

◆□ ▶ ◆□ ▶ ◆ 臺 ▶ ◆ 臺 ▶ ○ ● ○ 32/54

 Chisel 和 RISC-V

Chisel 初体验 ☐ Chisel 简介

Chisel 设计验证

Chisel 设计验证

Chisel 设计验证

参见 Design Verification。

 chiseltest (通用于 Chied 50 及之前版本) 是用于基于 Chied 的 RTL 设计的 合定地的测试和形式验证证法, 并且是以前的 PosePoseTests 的替代品, 提供 同的基本构造, 但具有简化的接口和并发支持 fork 和 join 和 Verilator 集成证 模拟。

#### Chisel 设计验证

Chisel 设计验证

参见 Design Verification。

- svsim 是 Chisel 的轻量级测试库,包含在此存储库中。
- chiseltest(适用于 Chisel 5.0 及之前版本)是用于基于 Chisel 的 RTL 设计的包含电池的测试和形式验证库,并且是以前的 PeekPokeTester 的替代品,提供相同的基本构造,但具有简化的接口和并发支持 fork 和 join 和 Verilator 集成进行模拟。

◆□ ▶ ◆□ ▶ ◆ ■ ▶ ◆ ■ りへ○ 33/54

Chisel 初体验 萨塞克斯人工智能学院

Verilog 语言简介 Scala 简介 Chisel 设计验证

Chisel 简介  Chisel 和 RISC-V

svsim I

svsim 是 Chisel 的轻量级测试库,包含在 Chisel 中。它是一个用于编译和控制 SystemVerilog 模拟的低级库,目前以 Verilator 和 VCS 作为后端。 Chisel 中可以使用 expect 来进行断言。通过对输入进行操作,然后让时钟前进一个 周期即可获取结果。

**◀□▶ ◀鬪▶ ◀臺▶ ◀臺▶** 臺 ∽9,0~ 34/54

Chisel 初体验 -Chisel 简介 └─Chisel 设计验证 ∟svsim

Chisal 中可以使用 expect 来进行断言。通过对输入进行操作,然后让时钟前进一

萨塞克斯人工智能学院 Chisel 初体验

 Scala 简介
 Chisel 简介
 Chisel 和 RISC-V

 ○○○○○○○○
 ○○○○○○○○

Verilog 语言简介 ○○○○○○○○ Chisel 设计验证

#### svsim II

```
GCDSpec.scala
class GCDSpec extends AnyFreeSpec with Matchers {
  "min and max check" in {
   simulate(new ModuleSample) { dut =>
     var a,b : Int =100;
     dut.io.a.poke(a);dut.io.b.poke(b);dut.clock.step()
     dut.io.minnum.expect(min(a,b), "Min Value test failed.");dut.io.
         maxnum.expect(max(a,b), "Max Value test failed.")
```

Chisel 初体验 萨塞克斯人工智能学院

Chisel 初体验 67-70 Chisel 简介 — Chisel 设计验证 — svsim

```
im II

**CCCGos acid

**CCCGos acid

**Cass acid per extends AnyTendoper with Batchers {

**Ass and acc clear* is {

**Ass and ac
```

svsim III

Chisel 简介 ○○○○○○○○○○○○●●●○○○○○○○

Chisel 中可以使用 println 输出任何东西。

◆ロト ◆御ト ◆草ト ◆草ト 草 からで 36/54

Chisel 初体验 2024-02-29 -Chisel 简介 └─Chisel 设计验证 svsim

Chisal 中可以使用 println 输出任何东西。

Scala 简介Chisel 简介Chisel 和 RISC-V○○○○○○○○○○○○○○○○○○○○○○○○○

Verilog 语言简介 ○○○○○○○○ Chisel 设计验证

#### svsim IV

```
GCDSpec.scala
class GCDSpec extends AnyFreeSpec with Matchers {
  "min and max check" in {
   simulate(new ModuleSample) { dut =>
     var a,b : Int =100;
     dut.io.a.poke(a);dut.io.b.poke(b);dut.clock.step()
     println("Result: " +a.toString+" & "+b.toString+" = "+dut.io.minnum
         .peek().toString)
```

◆ロト ◆母ト ◆ 喜ト ◆ 喜 ト 喜 ・ 夕 Q で 37/54

Chisel 初体验 萨塞克斯人工智能学院

Chisel 初体验 Chisel 简介 Chisel 设计验证 └─svsim

```
Arism IV

CONSTRUCTOR

CONSTRUC
```

Scala 简介 Chisel 简介 

Chisel 和 RISC-V

Verilog 语言简介

#### Chiseltest I

Chisel 设计验证

#### 版本支持

Chisel 6.0.0 在 2024 年 1 月 19 日发布。根据 Chisel Project Versioning 的约定,主 要版本更新不会提供向前的兼容性。而根据 ucb-bar/chiseltest#699. 该库尚未对 Chisel 6.0.0 提供支持。下列内容未经验证、仅供参考。

Chiseltest 是用于基于 Chisel 的 RTL 设计的包含电池的测试和形式验证库。 Chiseltest 强调测试的轻量级(最小化样板代码)、易于读写(可理解性)和组合 (为了更好的测试代码重用)。同时,它提供更加复杂的多线程测试等。 以下代码为队列测试的示例,见QueueTest.scala。

**◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 ∽**9<**○** 38/54

Chisel 初体验

-Chisel 简介

—Chisel 设计验证

Chiseltest

Chiseltest

要版本更新不全提供向前的基定性。而網報 urb.har/rhicebast#600 该市份表现 Chisel 6.0.0 提供支持。下列内容未经验证、仅供参考。

Chiseltest 是用于基于 Chisel 的 RTL 设计的包含电池的测试和形式验证库

Chisel 和 RISC-V Scala 简介 Chisel 简介 

Verilog 语言简介 000000000 Chisel 设计验证

#### Chiseltest II

```
GCDSpec.scala
1 it should "work with a combinational queue" in {
  test(new PassthroughQueue(UInt(8.W))) { c =>
    c.in.initSource();c.out.initSink()
    fork {
      c.in.enqueueSeq(Seq(42.U, 43.U, 44.U))
    }.fork {
      c.out.expectDequeueSeq(Seq(42.U, 43.U, 44.U))
    }.join()
```

```
◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 ∽9,0° 39/54
```

Chisel 初体验

-Chisel 简介

└─Chisel 设计验证

\_\_Chiseltest

2024-02-29

Chiseltest II

Chisel 初体验 Chisel 简介 Chisel 和 VS Code

第3节 Chisel 简介 第3.4 小节 Chisel 和 VS Code

第3节

Chisel 简介

第 3.4 小节

Chisel 和 VS Code

◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 から○ 40/54

Verilog 语言简介 ○○○○○○○○ Chisel 和 VS Code

### VS Code 对 Chisel 的支持

Chisel 是一个 Scala 库。因此 Chisel 主要使用 Scala 的相关插件。为支持 Chisel 的, VS Code 主要使用 3 个插件。

- scalameta.metals: Scala 感知和 sbt 相关支持。
- scala-lang.scala: Scala 语法高亮
- aaronduino.chisel: Chisel 语法和代码块支持。

**◆□▶◆□▶◆臺▶◆臺▶** 臺 **夕**♀○ 41/54

Chisel 和 RISC-V

Chisel 初体验 萨塞克斯人工智能学院

Chisel 初体验
Chisel 简介
Chisel 简介
Chisel 和 VS Code

VS Code 对 Chisel 的支持

VS Code 对 Chisel 的支持

Chisel 是一个 Scala 库。因此 Chisel 主要使用 Scala 的相关操作。为支持 Chisel 的

- scalameta.metals: Scala 感知和 sbt 相关支
- scala-lang.scala: Scala 语法高亮

 Scala 简介
 Chisel 简介
 Chisel 和 RISC-V

 ○○○○○○○○○
 ○○○○○○○○○
 ○○○○○○○○

Verilog 语言简介 000000000 Chisel 和 VS Code

# VS Code 的 Chisel 效果

图: VS Code 的 Chisel 智能感知和静态检查演示

**◆ロ ▶ ◆ □ ▶ ◆ ■ ▶ ◆ ■ ・ 夕 ○ ○ 42/54** 

Chisel 初体验 67-70-70 └─ Chisel 简介 └─ Chisel 和 VS Code └─ VS Code 的 Chisel 效果



Chisel 初体验 —Chisel 简介 2024-02-29 └─测试 Chisel

第3节 Chisel 简介 第3.5 小节 **测试 Chisel** 

第3节

# Chisel 简介

第 3.5 小节

测试 Chisel

◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 りへで 43/54

Chisel 和 RISC-V

测试 Chisel

Verilog 语言简介

#### 测试 Chisel I

参考 Installation, 首先安装所需的依赖项才能在本地构建 Chisel, 安装 openjdk-21-jdk 包、安装 sbt 包、安装 verilator 包。 运行下列代码克隆并构建 Chisel。

#### 构建 Chisel

- git clone https://github.com/chipsalliance/chisel.git
- 2 cd chisel
- 3 sbt compile

为了运行单元测试,PATH 中应包含 verilator、yosys 和 espresso。

**◆ロ ▶ ◆ 御 ▶ ◆ 草 ▶ ◆ 草 ・ 夕 ♀ ○ 44/54** 

萨塞克斯人工智能学院 Chisel 初体验

Chisel 初体验 -Chisel 简介 └─测试 Chisel └─测试 Chisel

測试 Chisel I 参考 Installation, 首先安装所需的依赖项才能在本地构建 Chisel, 安装 openjdk-21-jdk 包、安装 abt 包、安装 verilator 包。 运行下列代码克隆并构建 Chisel。

Verilog 语言简介 ○○○○○○○○

#### 测试 Chisel II

如果编译成功并且安装了上述依赖项,则可以通过调用以下命令来运行包含的单元测试。

## 构建 Chisel

1 sbt test

2024-02-29

Chisel 初体验 67-Chisel 简介 ─测试 Chisel ─测试 Chisel

測試 Chisel II
加斯特理政府上述情報項,例可以通过周用以下中令未进行哲술的申元 测试。
同语 Chinal
Lake Start

2024-02-29 Chise

Chisel 初体验 └─Chisel 和 RISC-V

<sup>第4节</sup> Chisel 和 RISC-V

第4节

Chisel 和 RISC-V

◆□ ▶ ◆□ ▶ ◆□ ▶ □ ♥ ♀
 ◆○ 46/54

Chisel 初体验 -Chisel 和 RISC-V └─在 openEuler RISC-V 上运行 Chisel

第4节 Chisel 和 RISC-V 第4.1 小节 在 openEuler RISC-V 上运行 Chisel

第4节

## Chisel 和 RISC-V

第 4.1 小节

在 openEuler RISC-V 上运行 Chisel

**◆ロ▶ ◆御▶ ◆夏▶ ◆夏▶ 夏 か**�� 47/54

在 openEuler RISC-V 上运行 Chisel

#### 运行 Verilator I

注意到 openEuler RISC-V 尚未提供 verilator 包。参阅 Git Quick Install 的说明。

#### 构建 Verilator

- 2 git clone https://github.com/verilator/verilator --depth=1
- 3 cd verilator
- 4 autoconf
- 5 ./configure
- 6 make -j 'nproc'
- 7 sudo make install

Chisel 初体验 萨塞克斯人工智能学院

Chisel 初体验 └─Chisel 和 RISC-V └─在 openEuler RISC-V 上运行 Chisel └─运行 Verilator

行 Verilator I
記載 pomine RECV 商業提供 vertaue 色、参阅 Ca Quodo install 的認用。
記述 Verifier

"you tended "y git heighten perl pythend ands settement giv files bisen
combe sensett."

git time Maps://githab.com/wrilator/verilator -depth-1

git time Maps://githab.com/wrilator/verilator -depth-1

state of the person of th

在 openEuler RISC-V 上运行 Chisel

### 运行 Verilator II

按照有关提示,运行 make test 进行有关测试,部分测试可以通过,部分测试由于缺少包而无法通过或跳过测试。这说明 Verilator 能够在该系统上运行。

#### Verilator 验证结果

1 ==SUMMARY: Passed 1 Failed 0 Time 2:55

2 ==SUMMARY: Passed 1 Failed 0 Time 2:55

Chisel 初体验 萨塞克斯人工智能学院

Chisel 初体验 ├─Chisel 和 RISC-V ├─在 openEuler RISC-V 上运行 Chisel ├─运行 Verilator

Verilog 语言简介

Scala 简介 0000000000

Chisel 和 RISC-V ○○○○●○○○

在 openEuler RISC-V 上运行 Chisel

运行 Scale

参考 17 安装并测试 Hullo World 程序, 能够运行通过。

Chisel 初体验 萨塞克斯人工智能学院

Chisel 初体验 2024-02-29 -Chisel 和 RISC-V └─在 openEuler RISC-V 上运行 Chisel └─运行 Scale

运行 Scale

参考 17 安装并测试 Hullo World 程序,能够运行通过

在 openEuler RISC-V 上运行 Chisel

#### 运行 Chisel

#### 参考 Did it work? 的有关说明, 在目录中运行

sbt -Dsbt.override.build.repos=true test --debug,得到如下结果。猜想问题为 JVM 虚拟机内部实现可能不符合预期,需进一步排查或对 Chisel 进行进一步测试。

#### 

◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 からで 51/54

Chisel 初体验 萨塞克斯人工智能学院

Chisel 初体验 └─Chisel 和 RISC-V └─在 openEuler RISC-V 上运行 Chisel └─运行 Chisel

\$ancofun\\$7.apply(Macros.scala:186)
at scala.tools.msc.typechecker.Macros\\$MacroImplBinding\\$'
\$ancofun\\$7.apply(Macros.scala:186)

Chisel 初体验 -Chisel 和 RISC-V └使用 Chisel 的 RISC-V 项目

第4节 Chisel 和 RISC-V 第42 小节 使用 Chisel 的 RISC-V 項目

第4节

Chisel 和 RISC-V

第 4.2 小节

使用 Chisel 的 RISC-V 项目

◆ロ → ◆母 → ◆ 章 → 章 り へ ○ 52/54

使用 Chisel 的 RISC-V 项目

#### 使用 Chisel 的 RISC-V 项目

- The Sodor Processor Collection: RV32I 处理器的简易实现
- NutShell: OSCPU(大学开源芯片项目)团队开发的处理器,目前支持RV64IMAC 和 RV32IMAC。
- Rocket Chip Generator: 一款开源片上系统设计生成器,可生成 RTL。
- 香山: 一款开源的高性能 RISC-V 处理器, 架构为 RV64GCBK。
- BOOM: Christopher Celio 的 RV64 乱序处理器实现。
- BottleRocket: RV32IMC 微处理器。

**◆□▶ ◆□▶ ◆臺▶ ◆臺▶ 臺 ∽**♀◎ 53/54

Chisel 初体验 萨塞克斯人工智能学院

Chisel 初体验 67-Chisel 和 RISC-V ──使用 Chisel 的 RISC-V 项目 ──使用 Chisel 的 RISC-V 项目

#### 使用 Chisel 的 RISC-V 項目

- The Sodor Processor Collection: RV32I 处理器的商易实现
- RV64IMAC 和 RV32IMAC。
- 晋山: 一款开源的高性能 RISC-V 处理器、架构为 RV64GC
- BOOM: Christopher Celio 的 RV64 乱序处理器实

# 谢谢





