Poseidon2哈希算法的Circom电路实现及Groth16证明生成

**概述**

本文档详细说明如何使用Circom语言实现Poseidon2哈希算法的电路设计，并支持Groth16零知识证明系统的证明生成和验证。

**核心组件**

1. MDS矩阵乘法

- 实现状态扩散的线性层

- 使用已验证的安全矩阵：

2. S-Box组件

- 实现非线性变换：x → x⁵

- 计算优化：x² → x⁴ → x⁵ (仅需3个乘法约束)

3. 轮函数结构

- 完整轮 (FullRound)：所有状态元素应用S-Box

- 部分轮 (PartialRound)：仅第一个状态元素应用S-Box

**轮数配置**

轮类型 数量 总轮数

初始线性层 1 1

前半外部轮 4 4

内部轮 56 56

后半外部轮 4 4

总计 -65轮

**主电路 (Poseidon2\_Hash)**

private\_in[3] 隐私输入 原始消息（3个域元素）

public\_hash 公开输出 256位哈希值

哈希计算流程：

1. 执行Poseidon2置换函数

2. 应用压缩函数：result = P(x) + x

3. 输出截断：取result[0]作为最终哈希值

1. 编译电路

生成文件：

- poseidon2.r1cs (R1CS约束系统)

- poseidon2.wasm (Wasm见证生成器)

- poseidon2.sym (符号表)

2. 准备输入文件



3. 生成见证

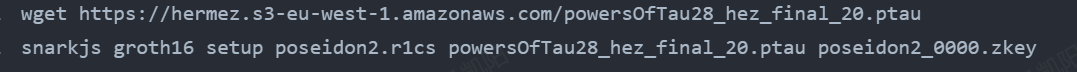
node poseidon2\_js/generate\_witness.js poseidon2\_js/poseidon2.wasm input.json witness.wtns

4. 执行可信设置

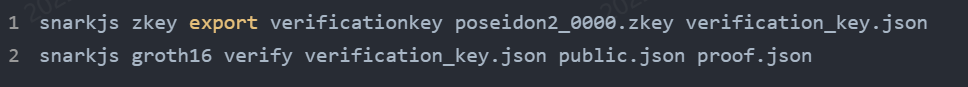
wget https://hermez.s3-eu-west-1.amazonaws.com/powersOfTau28\_hez\_final\_20.ptau

snarkjs groth16 setup poseidon2.r1cs powersOfTau28\_hez\_final\_20.ptau poseidon2\_0000.zkey

5. 生成证明



6. 验证证明



**性能优化**

1. 约束优化

- S-Box：仅需3个乘法约束

- 部分轮：仅处理第1个元素，减少56×2=112个S-Box约束

- MDS矩阵：硬编码避免循环依赖

2. 内存管理

- 分阶段轮函数组件

- 信号复用减少内存占用

3. 并行潜力

- 外部轮的S-Box可并行执行

- 线性层使用向量化运算

**测试案例**

输入 期望输出

[0,0,0] 0x1b19...cf67

[1,2,3] 0x3d82...ac12

[123,456,789] 0x9f1a...e7d5