Poseidon2 哈希算法的 Circom 电路实现报告

一、项目背景与目的

在零知识证明领域，哈希算法的电路实现具有至关重要的意义。它能够在保护隐私的前提下，实现对数据哈希值的验证等功能，广泛应用于身份认证、数据完整性校验等场景。

本次项目旨在基于 Poseidon2 哈希算法，使用 Circom 语言实现相应的电路。具体要求为参考文档中(n,t,d)=(256,3,5)的参数配置，将 Poseidon2 哈希值作为电路的公开输入，哈希原象作为隐私输入，且哈希算法的输入只考虑一个 block，最后采用 Groth16 算法生成证明。

二、Poseidon2 哈希算法概述

Poseidon2 哈希算法是一种基于 SPN（Substitution-Permutation Network 结构的哈希函数，其主要操作步骤包括 AddRoundConstant、SubWords（S-box）和 MixLayer。

1. AddRoundConstant：在每一轮操作中，向状态的每个元素添加特定的轮常量，用于增加算法的随机性和安全性。
2. SubWords（S-box）：对状态元素进行非线性变换，在本实现中，采用指数为 5 的幂运算作为 S-box，即对元素进行 x^5的运算，符合 d=5 的参数要求。
3. MixLayer：通过线性变换对状态元素进行混合，进一步扩散输入信息，增强算法的抗攻击能力。

本次实现采用的参数配置为(n,t,d)=(256,3,5)，其中 n 表示哈希输出长度为 256bits,t 表示状态元素数量为 3,d 表示 S-box 指数为 5。

三、Circom 电路实现

1、整体结构

本次实现的 Circom 电路主要包含 Poseidon2Hash 模板、Poseidon2Circuit 模板以及主组件实例化部分。

2、各模板功能说明

Poseidon2Hash 模板

该模板实现了 Poseidon2 哈希算法的核心计算过程。

输入输出定义：定义了隐私输入 preimage[2]作为哈希原象（因 t-1=2），以及输出 hash [1]作为哈希结果。
状态初始化：将隐私输入作为速率部分，容量部分初始化为 0，即 state[0]=preimage[0]，state[1]=preimage[1]，state[2]=0。
轮常量与混合矩阵：轮常量和混合矩阵在实际应用中应使用文档 1 中指定的官方参数，此处为示例设置了临时值。
轮操作：包括前半部分完整轮、部分轮和后半部分完整轮的操作。

完整轮：在每一轮中，先进行 AddRoundConstant 操作，为每个状态元素添加轮常量；然后进行 SubWords 操作，对所有状态元素应用 S-box；最后进行 MixLayer 操作，通过混合矩阵对状态进行线性混合。

部分轮：与完整轮的区别在于 SubWords 操作只对第一个元素应用 S-box。
哈希结果输出：将状态的容量部分 state[2]作为哈希结果输出。
Poseidon2Circuit 模板
该模板将哈希计算与约束条件相结合。
输入定义：定义了公开输入 expectedHash 作为预期的哈希值，以及隐私输入 preimage[2]作为哈希原象。
哈希计算：通过实例化 Poseidon2Hash 模板的 hasher 组件，将隐私输入传递给 hasher 进行哈希计算。
约束条件：约束计算得到的哈希值 hasher.hash[0]与公开输入的预期哈希值 expectedHash 相等，即 expectedHash===hasher.hash[0]。
主组件实例化
通过 component main=Poseidon2Circuit()实例化主电路组件，作为整个电路的入口。

四、总结

本次项目成功基于 Poseidon2 哈希算法和 Circom 语言实现了相应的电路，实现了将哈希原象作为隐私输入、哈希值作为公开输入，并采用 Groth16 算法生成证明的功能。

通过本次实现，深入理解了 Poseidon2 哈希算法的原理 Circom 电路的设计方法。但在实际应用中，还需要严格替换轮常量和混合矩阵等参数，并进行更全面的测试，以确保电路的安全性和可靠性。
未来，可以进一步优化电路的性能，减少计算复杂度，提高证明生成和验证的效率。同时，也可以探索将该电路应用于更多实际的零知识证明场景中，如隐私交易、数据共享等领域。