Poseidon2 电路实现说明文档

密码二班：钟铭圳. 董俊豪
密码一班：张瑞豪

2025 年 8 月 9 日

目 录

1 算法参数

确定参数：选择 (n, t, d) = (256, 3, 5) 。n 是安全级别，t 是状态大小，d 是 S-box 的指数。

2 核心组件说明

2.1 矩阵乘法

实现 3×3 MDS 矩阵乘法

```
template MatrixMul() {
    signal input in[3];
    signal output out[3];

    var M[3][3] = [
        [5, 7, 1],
        [3, 4, 6],
        [1, 1, 4]
    ];

    out[0] <== M[0][0]*in[0] + M[0][1]*in[1] + M[0][2]*in[2];
    out[1] <== M[1][0]*in[0] + M[1][1]*in[1] + M[1][2]*in[2];
    out[2] <== M[2][0]*in[0] + M[2][1]*in[1] + M[2][2]*in[2];
}
```

2.2 完整轮次

计算流程： AddRoundConstants S-box 应用（x 非线性变换）线性层变换（M_E 矩阵）

```
template FullRound(roundIndex) {
    // 1. 加轮常数
    afterARC[i] <== in[i] + rc[roundIndex][i];

    // 2. S-box (x )
    afterSBOX[i] <== afterARC[i]^5;

    // 3. 矩阵乘法
    component matmul = MatrixMul();
}
```

2.3 部分轮次

```
template PartialRound(roundIndex) {
    // 仅处理第一个元素
    afterARC <== in[0] + rc[roundIndex];
    afterSBOX <== afterARC^5;

    // 其他元素直通
    midState[0] <== afterSBOX;
    midState[1] <== in[1];
}
```

3 主电路架构

3.1 初始处理

应用初始线性层 M_E（文档 2 第 4 节安全增强）

```
component initMatMul = MatrixMul();
initMatMul.in <== inputs;
```

3.2 前四轮

```
    // 前4个完整轮次 (R_F/2)
    component fullRounds1[4];
    for (var i = 0; i < 4; i++) {
        fullRounds1[i] = FullRound(i);
        if (i == 0) {
            fullRounds1[i].in[0] <== initMatMul.out[0];
            fullRounds1[i].in[1] <== initMatMul.out[1];
            fullRounds1[i].in[2] <== initMatMul.out[2];
        } else {
            fullRounds1[i].in[0] <== fullRounds1[i-1].out[0];
            fullRounds1[i].in[1] <== fullRounds1[i-1].out[1];
            fullRounds1[i].in[2] <== fullRounds1[i-1].out[2];
        }
    }
```

3.3 56 个部分轮次

```
    component partialRounds[56];
    for (var i = 0; i < 56; i++) {
        partialRounds[i] = PartialRound(i);
        if (i == 0) {
            partialRounds[i].in[0] <== fullRounds1[3].out[0];
            partialRounds[i].in[1] <== fullRounds1[3].out[1];
            partialRounds[i].in[2] <== fullRounds1[3].out[2];
        } else {
            partialRounds[i].in[0] <==
                partialRounds[i-1].out[0];
            partialRounds[i].in[1] <==
                partialRounds[i-1].out[1];
            partialRounds[i].in[2] <==
                partialRounds[i-1].out[2];
        }
    }
```

3.4 后 4 个完整轮次

```
    component fullRounds2[4];
    for (var i = 0; i < 4; i++) {
        fullRounds2[i] = FullRound(i+4); // 使用索引4-7的常数
        if (i == 0) {
            fullRounds2[i].in[0] <== partialRounds[55].out[0];
            fullRounds2[i].in[1] <== partialRounds[55].out[1];
            fullRounds2[i].in[2] <== partialRounds[55].out[2];
        } else {
            fullRounds2[i].in[0] <== fullRounds2[i-1].out[0];
            fullRounds2[i].in[1] <== fullRounds2[i-1].out[1];
            fullRounds2[i].in[2] <== fullRounds2[i-1].out[2];
        }
    }
```