

MiMC in Halo2

Abstract

We give a specification of the MiMC block ciphers for implementation in the Pasta elliptic curve fields. The specification closely follows the MiMC implementation in `circom`. We also outline the proposed implementation strategy in Halo2.

1 Introduction

The MiMC block ciphers were designed to have low multiplicative complexity [1]. Two variants were proposed: one based on a cubic round function and the other having a Feistel network structure (with a cubic round function). We will refer to them as `MiMC` and `MiMC-Feistel` for convenience. The MiMC hash functions are obtained from these block ciphers by setting the key value to zero.

We are interested in MiMC implementations when the input, output, and key are from a prime field \mathbb{F}_p .

1.1 MiMC over a prime field

A MiMC block cipher implementation over \mathbb{F}_p is specified as below:

- Set s to be smallest integer greater than one that satisfies $\gcd(s, p-1) = 1$. The gcd condition ensures that x^s is a permutation in \mathbb{F}_p . We want s to be the smallest such integer to keep the multiplicative complexity minimal.
- The cipher has r rounds where $r = \left\lceil \frac{\log_2 p}{\log_2 s} \right\rceil$.
- Let $\{c_i \in \mathbb{F}_p \mid i = 0, 1, \dots, r-1\}$ be the r round constants. By convention, the first round constant is set to zero, i.e. $c_0 = 0$. The remaining round constants are derived in a pseudorandom manner and are fixed for a particular implementation.
- The i th round function is given by $F_i(x) = (x + k + c_i)^s$ where $k \in \mathbb{F}_p$ is the secret key.
- The encryption of $x \in \mathbb{F}_p$ is given by

$$\begin{aligned} E_k(x) &= F_{r-1}(\cdots F_2(F_1(F_0(x))) \cdots) + k \\ &= (\cdots ((x + k + c_0)^s + k + c_1)^s \cdots + k + c_{r-1})^s + k. \end{aligned}$$

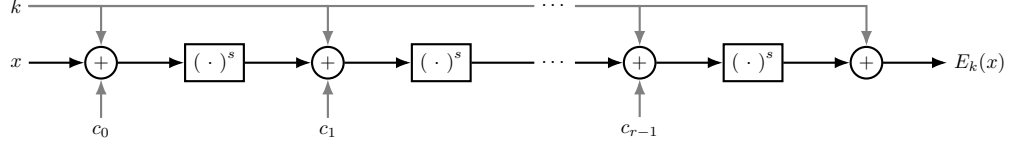


Figure 1: MiMC Encryption

This is illustrated in Figure 1.

1.2 MiMC-Feistel over a prime field

A MiMC-Feistel implementation over \mathbb{F}_p is specified as below:

- Once again, set s to be smallest integer greater than one that satisfies $\gcd(s, p-1) = 1$.
- The cipher has r rounds where $r = 2 \times \left\lceil \frac{\log_2 p}{\log_2 s} \right\rceil$. Note that the number of rounds is twice as large as the number used in MiMC.
- Let $\{c_i \in \mathbb{F}_p \mid i = 0, 1, \dots, r-1\}$ be the r round constants. By convention, the first and last round constants are set to zero, i.e. $c_0 = c_{r-1} = 0$. The remaining round constants are derived in a pseudorandom manner and are fixed for a particular implementation.
- In the first $r-1$ rounds, the i th round function is given by

$$F_i(x_L \| x_R) = (x_R + (x_L + k + c_i)^s) \| x_L, \quad i = 0, 1, \dots, r-1,$$

where $x_L, x_R \in \mathbb{F}_p$ are the inputs and $k \in \mathbb{F}_p$ is the secret key.

- The last round function is given by

$$F_{r-1}(x_L \| x_R) = x_L \| (x_R + (x_L + k + c_{r-1})^s).$$

It does not perform the swap operation seen in previous rounds.

- The encryption of $x_L \| x_R \in \mathbb{F}_p^2$ is given by

$$E_k(x) = F_{r-1}(\dots F_2(F_1(F_0(x_L \| x_R))) \dots).$$

This is illustrated in Figure 2.

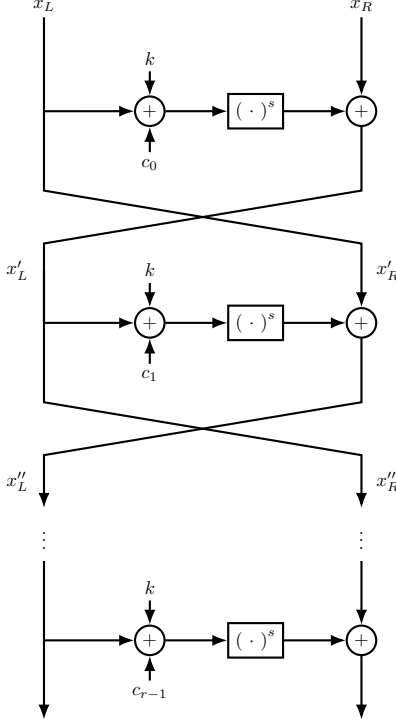


Figure 2: MiMC Feistel Encryption

2 MiMC in circom

The `circom` library [2] has implementations of both MiMC and MiMC-Feistel ciphers. The MiMC implementation in `circom` uses $s = 7$ while the MiMC-Feistel implementation uses $s = 5$.¹

The MiMC implementation has 91 rounds where the round constants are elements in \mathbb{F}_p where p is the order of the `alt_bn128` curve. Figure 3 shows pseudocode of the round constant calculation.² It hashes the string “mimc”

¹It is not clear why $s = 7$ was chosen for MiMC in `circom`. The field where the MiMC calculations will occur has order equal to the group order of the `alt_bn128` (BN254) curve. This order is given by the prime

$$p = 2188824287183927522246405745257275088548364400416034343698204186575808495617.$$

While $\gcd(p - 1, 3) = 3$, we have $\gcd(p - 1, 5) = 1$. So $s = 5$ is a valid choice. In a document describing EdDSA on the Baby Jubjub curve [3], the authors mention that $s = 7$ is the optimal choice since $\gcd(l - 1, 3) = 3$ where l is group order of the Baby Jubjub curve. It turns out that $\gcd(l - 1, 5) = 5$ and $\gcd(l - 1, 7) = 1$. But the order of elliptic curve group should not matter.

²See <https://github.com/iden3/circomlibjs/blob/main/src/mimc7.js> for a Javascript implementation of the round constant calculation.

The number of MiMC rounds in both Pasta fields will be 110, as $\left\lceil \frac{\log_2 p}{\log_2 5} \right\rceil = \left\lceil \frac{\log_2 q}{\log_2 5} \right\rceil = 110$. And the number of MiMC-Feistel rounds will be 220.

3.1 Round Constant Generation

We propose to use the same algorithm as `circom` to generate round constants (with the Pasta fields substituted for the `alt_bn128` scalar field).

Sage codes for generating the Pallas field round constants for MiMC and MiMC-Feistel are given in Appendix A and Appendix B. These programs and the generated constants are available at <https://github.com/avras/pasta-mimc>.

4 MiMC implementation in Halo2

4.1 MiMC in Halo2

The MiMC cipher and hash implementations are available at <https://github.com/avras/mimc-halo2/tree/main/src/mimc>.

4.1.1 MiMC cipher

The MiMC cipher implementation has the following columns (illustrated in Table 1):

1. One fixed column to hold the round constants; the i th row holds c_i
2. One advice column to hold the key k which is used in all the rounds
3. One advice column to hold the cipher state x_i after the i th round. The first row of this column holds the message x to be encrypted.

$$x_i = \begin{cases} x & \text{if } i = 0, \\ (x_{i-1} + k + c_{i-1})^5 & \text{if } 1 \leq i \leq 110, \\ x_{i-1} + k & \text{if } i = 111. \end{cases}$$

4. Two selectors: `s_in_rounds`, `s_post_rounds`

$$\begin{aligned} \text{s_in_rounds} = 1 &\iff 1 \leq i \leq 110 \\ \text{s_post_rounds} = 1 &\iff i = 111 \end{aligned}$$

4.1.2 MiMC hash

The MiMC hash implementation has the following columns (illustrated in Table 2):

1. One fixed column to hold the round constants; the i th row holds c_i

state	key column	round constants	s_in_rounds	s_post_rounds
x_0	k	c_0	0	0
$x_1 = (x_0 + k + c_0)^5$	k	c_1	1	0
$x_2 = (x_1 + k + c_1)^5$	k	c_2	1	0
$x_3 = (x_2 + k + c_2)^5$	k	c_3	1	0
\vdots	\vdots	\vdots	\vdots	\vdots
$x_{109} = (x_{108} + k + c_{108})^5$	k	c_{109}	1	0
$x_{110} = (x_{109} + k + c_{109})^5$	k		1	0
$x_{110} + k$			0	1

Table 1: MiMC cipher layout

state	round constants	s_in_rounds
x_0	c_0	0
$x_1 = (x_0 + c_0)^5$	c_1	1
$x_2 = (x_1 + c_1)^5$	c_2	1
$x_3 = (x_2 + c_2)^5$	c_3	1
\vdots	\vdots	\vdots
\vdots		
$x_{109} = (x_{108} + c_{108})^5$	c_{109}	1
$x_{110} = (x_{109} + c_{109})^5$		1

Table 2: MiMC hash layout

2. One advice column to hold the cipher state x_i after the i th round. The first row of this column holds the message x to be hashed.

$$x_i = \begin{cases} x & \text{if } i = 0, \\ (x_{i-1} + c_{i-1})^5 & \text{if } 1 \leq i \leq 110 \end{cases}$$

3. One selector: **s_in_rounds**

$$\mathbf{s_in_rounds} = 1 \iff 1 \leq i \leq 110$$

4.2 MiMC-Feistel in Halo2

Not implemented yet.

A MiMC constant generation for Pallas field

[illegible]

B MiMC-Feistel constant generation for Pallas field

```
from sage.all import *
from Crypto.Hash import keccak

prime = 0x400000000000000000000000000000000224698fc094cf91b992d30ed00000001
num_rounds = 2*ceil(log(prime,2)/log(5,2))
print('Number of rounds =', num_rounds)

F = GF(prime)
round_constants = [F(0)]

seed_value = b'mimcsponge';
keccak_hash = keccak.new(data=seed_value, digest_bits=256)
hash_val = keccak_hash.digest()

for i in range(1,num_rounds-1):
    keccak_hash = keccak.new(data=hash_val, digest_bits=256)
    hash_val = keccak_hash.digest()
    hash_val_in_hex = keccak_hash.hexdigest()

    field_element = F('0x'+hash_val_in_hex)
    round_constants.append(field_element)
round_constants.append(F(0))

print ('round_constants = [')
for i in range(num_rounds):
    print(' ', hex(round_constants[i])+','+')
print('']')
```


References

- [1] Martin Albrecht, Lorenzo Grassi, Christian Rechberger, Arnab Roy, and Tyge Tiessen. MiMC: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. Cryptology ePrint Archive, Paper 2016/492, 2016. <https://eprint.iacr.org/2016/492>.
- [2] CircomLib. <https://github.com/iden3/circomlib>.
- [3] Jordi Baylina and Marta Bellés. EdDSA for Baby Jubjub elliptic curve with MiMC-7 hash. https://iden3-docs.readthedocs.io/en/latest/_downloads/a04267077fb3fdbf2b608e014706e004/Ed-DSA.pdf.