

In-EVM Mina State Verification Circuit Description

Cherniaeva Alisa

a.cherniaeva@nil.foundation

=nil; Crypto3 (<https://crypto3.nil.foundation>)

Shirobokov Ilia

i.shirobokov@nil.foundation

=nil; Crypto3 (<https://crypto3.nil.foundation>)

November 7, 2021

1 Introduction

WIP

High level description according to RfP¹

1. Computing several hash values from the data of the proof. This involves using the Poseidon hash function with 63 full rounds both over \mathbb{F}_p and \mathbb{F}_q with round constants and MDS matrix specified for \mathbb{F}_p ² and for \mathbb{F}_q ³.
2. Checking arithmetic equations.
3. Performing one multi-scalar multiplication (MSM) of size $2n_2 + 4 + (2 + 25) = 63$, for which some of the bases are fixed and some are variable.
4. For each $i \in \{1, 2\}$, performing a multi-scalar multiplication over \mathbb{G}_i of size 2^{n_i} with a fixed array of bases, and with scalars that can be very efficiently computed from the proof.

Note that for MSM in Step 4:

$$\sum_{i=0}^{2^{n_k}-1} s_i \cdot G_i = H$$
$$s_i := \prod_{\substack{0 \leq j \leq n_k \\ \text{bits}(i)[j]=1}} \phi(c_j),$$

where:

- $\phi: \{0, 1\}^{128} \rightarrow \mathbb{F}$ is defined as `to_field` in the implementation⁴.
- Given an integer $i < 2^{n_k}$, $\text{bits}(i)$ is defined as the little-endian bit array of length n representing the binary expansion of i .
- $G_0, \dots, G_{2^{n_k}-1} \in \mathbb{G}_k$ is a fixed sequence of group elements⁵.
- $c_0, \dots, c_{n_k-1} \in \{0, 1\}^{128}$ is a sequence of challenges.

We use the same 15-wires PLONK circuits that are designed for Mina.⁶

2 Preliminaries

WIP

¹https://hackmd.io/u_2Ygx8XS5Ss1a0bgOFjkA

²<https://github.com/o1-labs/proof-systems/blob/master/oracle/src/pasta/fp.rs>

³<https://github.com/o1-labs/proof-systems/blob/master/oracle/src/pasta/fq.rs>

⁴<https://github.com/o1-labs/proof-systems/blob/49f81edc9c86e5907d26ea791fa083640ad0ef3e/oracle/src/sponge.rs#L33>

⁵<https://github.com/o1-labs/proof-systems/blob/master/dlog/commitment/src/srs.rs#L70>

⁶https://o1-labs.github.io/mina-book/specs/15_wires/15_wires.html

2.1 Pasta Curves

Let $n_1 = 17$, $n_2 = 16$. Pasta curves parameters:

- $p = 2^{254} + 45560315531419706090280762371685220353$
- $q = 2^{254} + 45560315531506369815346746415080538113$
- Pallas:

$$\mathbb{G}_1 = \{(x, y) \in \mathbb{F}_p \mid y^2 = x^3 + 5\}$$

$$|\mathbb{G}_1| = q$$

- Vesta:

$$\mathbb{G}_2 = \{(x, y) \in \mathbb{F}_q \mid y^2 = x^3 + 5\}$$

$$|\mathbb{G}_2| = p$$

2.2 Verification Algorithm

Notations:

N_{wires}	Number of wires ('advice columns')
N_{perm}	Number of wires that are included in the permutation argument
N_{prev}	Number of previous challenges
$S_{\sigma_i}(X)$	Permutation polynomials for $0 \leq i < N_{\text{perm}}$
$pub(X)$	Public input polynomial
$w_i(X)$	Witness polynomials for $0 \leq i < N_{\text{wires}}$
$\eta_i(X)$	Previous challenges polynomials for $0 \leq i < N_{\text{prev}}$
ω	n -th root of unity

Proof π contains (here \mathbb{F}_r is a scalar field of \mathbb{G}):

- Commitments:
 - Witness polynomials: $w_{0,\text{comm}}, \dots, w_{N_{\text{wires}},\text{comm}} \in \mathbb{G}$
 - Permutation polynomial: $z_{\text{comm}} \in \mathbb{G}$
 - Quotient polynomial: $t_{\text{comm}} = (t_{\text{comm},1}, t_{\text{comm},2}) \in (\mathbb{G}^{N_{\text{perm}}} \times \mathbb{G})$
- Evaluations:
 - $w_0(\zeta), \dots, w_{N_{\text{wires}}}(\zeta) \in \mathbb{F}_r$
 - $w_0(\zeta\omega), \dots, w_{N_{\text{wires}}}(\zeta\omega) \in \mathbb{F}_r$
 - $z(\zeta), z(\zeta\omega) \in \mathbb{F}_r$
 - $S_{\sigma_0}(\zeta), \dots, S_{\sigma_{N_{\text{perm}}}}(\zeta) \in \mathbb{F}_r$
 - $S_{\sigma_0}(\zeta\omega), \dots, S_{\sigma_{N_{\text{perm}}}}(\zeta\omega) \in \mathbb{F}_r$
 - $\bar{L}(\zeta\omega) \in \mathbb{F}_r$
- Opening proof o_π :
 - $(L_i, R_i) \in \mathbb{G} \times \mathbb{G}$ for $0 \leq i < \text{lr_rounds}$
 - $\delta, \hat{G} \in \mathbb{G}$
 - $z_1, z_2 \in \mathbb{F}_r$
- previous challenges:
 - $\{\eta_i(\xi_j)\}_{j, \eta_{i,\text{comm}}}$, for $0 \leq i < \text{prev}$

Denote multi-scalar multiplication $\sum_{s_i \in \mathbf{s}, G_i \in \mathbf{G}} [s_i]G_i$ by $\text{MSM}(\mathbf{s}, \mathbf{G})$ for $l_{\mathbf{s}} = l_{\mathbf{G}}$ where $l_{\mathbf{s}} = |\mathbf{s}|$, $l_{\mathbf{G}} = |\mathbf{G}|$.

If $l_{\mathbf{s}} < l_{\mathbf{G}}$, then we use only first $l_{\mathbf{s}}$ elements of \mathbf{G}

Remark: For simplicity, we do not use distinct proofs index i for each element in the algorithm below. For instance, we write pub_{comm} instead of $pub_{i,\text{comm}}$.

Algorithm 1 Verification

Input: $\pi_0, \dots, \pi_{\text{batch_size}}$ (see 2.2)**Output:** acc or rej

1. for each π_i :
 - 1.1 $\text{pub}_{\text{comm}} = \text{MSM}(\mathbf{L}, \text{pub}) \in \mathbb{G}$, where \mathbf{L} is Lagrange bases vector
 - 1.2 **random_oracle**(p_{comm}, π_i):
 - 1.2.1 $H_{\mathbb{F}_q}.\text{absorb}(\text{pub}_{\text{comm}} || w_{0,\text{comm}} || \dots || w_{N_{\text{wires}},\text{comm}})$
 - 1.2.2 $\beta, \gamma = H_{\mathbb{F}_q}.\text{squeeze}()$
 - 1.2.3 $H_{\mathbb{F}_q}.\text{absorb}(z_{\text{comm}})$
 - 1.2.4 $\alpha = \phi(H_{\mathbb{F}_q}.\text{squeeze}())$
 - 1.2.5 $H_{\mathbb{F}_q}.\text{absorb}(t_{1,\text{comm}} || \infty || \dots || \infty || t_{2,\text{comm}})$
 - 1.2.6 $\zeta = \phi(H_{\mathbb{F}_q}.\text{squeeze}())$
 - 1.2.7 Transform $H_{\mathbb{F}_q}$ to $H_{\mathbb{F}_r}$
 - 1.2.8 $\bar{\alpha} = [\alpha^2, \dots, \alpha^{59}]$
 - 1.2.9 $H_{\mathbb{F}_r}.\text{absorb}(\text{pub}(\zeta) || w_0(\zeta) || \dots || w_{N_{\text{wires}}}(\zeta) || S_0(\zeta) || \dots || S_{N_{\text{perm}}}(\zeta))$
 - 1.2.10 $H_{\mathbb{F}_r}.\text{absorb}(\text{pub}(\zeta\omega) || w_0(\zeta\omega) || \dots || w_{N_{\text{wires}}}(\zeta\omega) || S_0(\zeta\omega) || \dots || S_{N_{\text{perm}}}(\zeta\omega))$
 - 1.2.11 $H_{\mathbb{F}_r}.\text{absorb}(\bar{L}(\zeta\omega))$
 - 1.2.12 $v = \phi(H_{\mathbb{F}_r}.\text{squeeze}())$
 - 1.2.13 $u = \phi(H_{\mathbb{F}_r}.\text{squeeze}())$
 - 1.2.14 Compute evaluation of $\eta_i(\zeta), \eta_i(\zeta\omega)$ for $0 \leq i < N_{\text{prev}}$
 - 1.2.15 Compute evaluation of $\bar{L}(\zeta)$
 - 1.3 $\mathbf{f}_{\text{base}} = \{S_{\sigma_{N_{\text{perm}}-1},\text{comm}}, \text{gate}_{\text{mult},\text{comm}}, w_{0,\text{comm}}, w_{1,\text{comm}}, w_{2,\text{comm}}, q_{\text{const},\text{comm}}, \text{gate}_{\text{psdn},\text{comm}}, \text{gate}_{\text{rc},\text{comm}}, \text{gate}_{\text{ec_add},\text{comm}}, \text{gate}_{\text{ec_dbl},\text{comm}}, \text{gate}_{\text{ec_endo},\text{comm}}, \text{gate}_{\text{ec_vbase},\text{comm}}\}$
 - 1.4 $s_{\text{perm}} := (w_0(\zeta) + \gamma + \beta \cdot S_{\sigma_0}(\zeta)) \cdot \dots \cdot (w_5(\zeta) + \gamma + \beta \cdot S_{\sigma_{N_{\text{perm}}}}(\zeta))$
 - 1.5 $\mathbf{f}_{\text{scalars}} = \{-z(\zeta\omega) \cdot \beta \cdot \alpha_0 \cdot \text{zkp}(\zeta) \cdot s_{\text{perm}}, w_0(\zeta) \cdot w_1(\zeta), w_0(\zeta), w_1(\zeta), 1, s_{\text{psdn}}, \alpha^0, \dots, \alpha^{14}, s_{\text{ec_add}}, s_{\text{ec_dbl}}, s_{\text{ec_endo}}, s_{\text{ec_vbase}}\}$
 - 1.6 $f_{\text{comm}} = \text{MSM}(\mathbf{f}_{\text{base}}, \mathbf{f}_{\text{scalars}})$
 - 1.7 $\bar{L}_{\text{comm}} = f_{\text{comm}} - t_{\text{comm}} \cdot (\zeta^n - 1)$
 - 1.8 **PE** is a set of elements of the form $(f_{\text{comm}}, f(\zeta), f(\zeta\omega))$ for the following polynomials:
 $\eta_0, \dots, \eta_{N_{\text{prev}}}, \text{pub}, w_0, \dots, w_{N_{\text{wires}}}, z, S_{\sigma_0}, \dots, S_{\sigma_{N_{\text{perm}}}}, \bar{L}$
 - 1.9 $\mathcal{P}_i = \{H_{\mathbb{F}_q}, \zeta, v, u, \mathbf{PE}, o_{\pi_i}\}$
 2. **final_check**($\mathcal{P}_0, \dots, \mathcal{P}_{\text{batch_size}}$)
-

Algorithm 2 Final Check

Input: $\pi_0, \dots, \pi_{\text{batch_size}}$, where $\pi_i = \{H_{i, \mathbb{F}_q}, \zeta_i, \zeta_i \omega, v_i, u_i, \mathbf{PE}_i, o_{\pi_i}\}$ **Output:** acc or rej

1. $\rho_1 \rightarrow \mathbb{F}_r$
 2. $\rho_2 \rightarrow \mathbb{F}_r$
 3. $r_0 = r'_0 = 1$
 4. for $0 \leq i < \text{batch_size}$:
 - 4.1 $\text{cip}_i = \text{inner_product}(\zeta_i, \zeta_i \omega, v_i, u_i, \mathbf{PE}_i)$
 - 4.2 $H_{i, \mathbb{F}_q}.\text{absorb}(\text{cip}_i - 2^{255})$
 - 4.3 $U_i = (H_{i, \mathbb{F}_q}.\text{squeeze}()).\text{to_group}()$
 - 4.4 Calculate opening challenges $\xi_{i,j}$ from o_{π_i}
 - 4.5 $h_i(X) := \prod_{k=0}^{\log(d+1)-1} (1 + \xi_{\log(d+1)-k} X^{2^k})$, where $d = \text{lr_rounds}$
 - 4.6 $b_i = h_i(\zeta) + u_i \cdot h_i(\zeta \omega)$
 - 4.7 $C_i = \sum_j (\sum_k v_i^k f_{k, \text{comm}})$, where $f_{k, \text{comm}}$ from \mathbf{PE}_i .
 - 4.8 $Q_i = \sum (\xi_{i,j} \cdot L_{i,j} + \xi_{i,j}^{-1} \cdot R_j) + \text{cip}_i \cdot U_i + C_i$
 - 4.9 $c_i = \phi(H_{i, \mathbb{F}_q}.\text{squeeze}())$
 - 4.10 $r_i = r_{i-1} \cdot \rho_1$
 - 4.11 $r'_i = r'_{i-1} \cdot \rho_2$
 - 4.12 Check $\hat{G}_i = \langle s, G \rangle$, where s is set of $h(X)$ coefficients.
Remark: This check can be done inside the MSM below using r'_i .
 5. $\text{res} = \sum_i r^i (c_i Q_i + \text{delta}_i - (z_{i,1}(\hat{G}_i + b_i U_i) + z_{i,2} H))$
 6. return $\text{res} == 0$
-

Algorithm 3 Inner Product

Input: $\zeta_1, \zeta_2, \xi, r, \{f_0(\zeta_1), \dots, f_k(\zeta_1)\}, \{f_0(\zeta_2), \dots, f_k(\zeta_2)\}$ **Output:** s

1. $s = \sum_{i=0}^k \xi^i \cdot (f_i(\zeta_1) + r \cdot f_i(\zeta_2))$
-

3 Elliptic Curve Arithmetic

WIP

3.1 Addition

Row	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i	x_1	y_1	x_2	y_2	x_3	y_3	r	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

Constraints:

- $(x_2 - x_1) \cdot (y_3 + y_1) - (y_1 - y_2) \cdot (x_1 - x_3)$
- $(x_1 + x_2 + x_3) \cdot (x_1 - x_3) \cdot (x_1 - x_3) - (y_3 + y_1) \cdot (y_3 + y_1)$
- $(x_2 - x_1) \cdot r = 1$

3.2 Doubling and Tripling

Row	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i	x_1	y_1	x_2	y_2	x_3	y_3	r_1	r_2	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

Constraints:

- Doubling:

$$\begin{aligned}
& - 4 \cdot y_1^2 \cdot (x_2 + 2 \cdot x_1) = 9 \cdot x_1^4 \\
& - 2 \cdot y_1 \cdot (y_2 + y_1) = (3 \cdot x_1^2) \cdot (x_1 - x_2) \\
& - y_1 \cdot r_1 = 1
\end{aligned}$$

- Addition (for tripling):

$$\begin{aligned}
& - (x_2 - x_1) \cdot (y_3 + y_1) - (y_1 - y_2) \cdot (x_1 - x_3) \\
& - (x_1 + x_2 + x_3) \cdot (x_1 - x_3) \cdot (x_1 - x_3) - (y_3 + y_1) \cdot (y_3 + y_1) \\
& - (x_2 - x_1) \cdot r_2 = 1
\end{aligned}$$

3.3 Variable Base Scalar Multiplication

For $S = [r]T$, where $r = 2^n + k$ and $k = [k_n \dots k_0]$, $k_i \in \{0, 1\}$:⁷

1. $S = [2]T$

2. for i from $n - 1$ to 0:

- 2.1 $Q = k_{i+1} ? T : -T$

- 2.2 $R = S + Q$

- 2.3 $S = R + S$

3. $S = k_0 ? S - T : S$

Row	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i	x_T	y_T	x_S	y_S	x_P	y_P	$n = 0$	x_R	y_R	s_1	s_2	b_1	s_3	s_4	b_2
$i + 1$	s_5	b_3	x_S	y_S	x_P	y_P	n	x_R	y_R	x_V	y_V	s_1	b_1	s_3	b_2
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
$i + 100$	x_T	y_T	x_S	y_S	x_P	y_P	n	x_R	y_R	s_1	s_2	b_1	s_3	s_4	b_2
$i + 101$	s_5	b_3	x_S	y_S	x_P	y_P	n	x_R	y_R	x_V	y_V	s_1	b_1	s_3	b_2

Constraints for $i + z$, where $z \bmod 2 = 0$:

- $b_1 \cdot (b_1 - 1) = 0$
- $b_2 \cdot (b_2 - 1) = 0$
- $(x_P - x_T) \cdot s_1 = y_P - (2b_1 - 1) \cdot y_T$
- $s_1^2 - s_2^2 = x_T - x_R$
- $(2 \cdot x_P + x_T - s_1^2) \cdot (s_1 + s_2) = 2y_P$
- $(x_P - x_R) \cdot s_2 = y_R + y_P$
- $(x_R - x_T) \cdot s_3 = y_R - (2b_2 - 1) \cdot y_T$
- $s_3^2 - s_4^2 = x_T - x_S$
- $(2 \cdot x_R + x_T - s_3^2) \cdot (s_3 + s_4) = 2 \cdot y_R$
- $(x_R - x_S) \cdot s_4 = y_S + y_R$
- $n = 32 \cdot \text{next}(n) + 16 \cdot b_1 + 8 \cdot b_2 + 4 \cdot \text{next}(b_1) + 2 \cdot \text{next}(b_2) + \text{next}(b_3)$

Constraints for $i + z$, where $z \bmod 2 = 1$:

⁷Using the results from <https://arxiv.org/pdf/math/0208038.pdf>

- $b_1 \cdot (b_1 - 1) = 0$
- $b_2 \cdot (b_2 - 1) = 0$
- $b_3 \cdot (b_3 - 1) = 0$
- $(x_P - x_T) \cdot s_1 = y_P - (2b_1 - 1) \cdot y_T$
- $(2 \cdot x_P + x_T - s_1^2) \cdot ((x_P - x_R) \cdot s_1 + y_R + y_P) = (x_P - x_R) \cdot 2y_P$
- $(y_R + y_P)^2 = (x_P - x_R)^2 \cdot (s_1^2 - x_T + x_R)$
- $(x_T - x_R) \cdot s_3 = (2b_2 - 1) \cdot y_T - y_R$
- $(2x_R - s_3^2 + x_T) \cdot ((x_R - x_V) \cdot s_3 + y_V + y_R) = (x_R - x_V) \cdot 2y_R$
- $(y_V + y_R)^2 = (x_R - x_V)^2 \cdot (s_3^2 - x_T + x_V)$
- $(x_T - x_V) \cdot s_5 = (2b_3 - 1) \cdot y_T - y_V$
- $(2x_V - s_5^2 + x_T) \cdot ((x_V - x_S) \cdot s_5 + y_S + y_V) = (x_V - x_S) \cdot 2y_V$
- $(y_S + y_V)^2 = (x_V - x_S)^2 \cdot (s_5^2 - x_T + x_S)$

3.4 Variable Base Endo-Scalar Multiplication

For $S = [r]T$, where $r = [r_n \dots r_0]$ and $r_i \in \{0, 1\}$:⁸

1. $S = [2](\phi(T) + T)$
2. for i from $\frac{\lambda}{2} - 1$ to 0:
 - 2.1 $Q = r_{2i+1} ? \phi([2r_{2i} - 1]T) : [2r_{2i} - 1]T$
 - 2.2 $R = S + Q$
 - 2.3 $S = R + S$

Row	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i	x_T	y_T	x_S	y_S	x_P	y_P	n	x_R	y_R	s_1	s_3	b_1	b_2	b_3	b_4
$i + 1$	s_5	b_3	x_S	y_S	x_P	y_P	n	x_R	y_R	s_1	s_3	b_1	b_2	b_3	b_4
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
$i + 62$	x_T	y_T	x_S	y_S	x_P	y_P	n	x_R	y_R	s_1	s_3	b_1	b_2	b_3	b_4
$i + 63$	s_5	b_3	x_S	y_S	x_P	y_P	n	x_R	y_R	s_1	s_3	b_1	b_2	b_3	b_4

Constraints:

- $b_1 \cdot (b_1 - 1) = 0$
- $b_2 \cdot (b_2 - 1) = 0$
- $b_3 \cdot (b_3 - 1) = 0$
- $b_4 \cdot (b_4 - 1) = 0$
- $((1 + (\text{endo} - 1) \cdot b_2) \cdot x_T - x_P) \cdot s_1 = (2 \cdot b_1 - 1) \cdot y_T - y_P$
- $(2 \cdot x_P - s_1^2 + (1 + (\text{endo} - 1) \cdot b_2) \cdot x_T) \cdot ((x_P - x_R) \cdot s_1 + y_R + y_P) = (x_P - x_R) \cdot 2 \cdot y_P$
- $(y_R + y_P)^2 = (x_P - x_R)^2 \cdot (s_1^2 - (1 + (\text{endo} - 1) \cdot b_2) \cdot x_T + x_R)$
- $((1 + (\text{endo} - 1) \cdot b_2) \cdot x_T - x_R) \cdot s_3 = (2 \cdot b_3 - 1) \cdot y_T - y_R$
- $(2 \cdot x_R - s_3^2 + (1 + (\text{endo} - 1) \cdot b_4) \cdot x_T) \cdot ((x_R - x_S) \cdot s_3 + y_S + y_R) = (x_R - x_S) \cdot 2 \cdot y_R$
- $(y_S + y_R)^2 = (x_R - x_S)^2 \cdot (s_3^2 - (1 + (\text{endo} - 1) \cdot b_4) \cdot x_T + x_S)$
- $n = 16 \cdot \text{next}(n) + 8 \cdot b_1 + 4 \cdot b_2 + 2 \cdot b_3 + b_4$

4 Multi-Scalar Multiplication Circuit

WIP

Input: $G_0, \dots, G_{k-1} \in \mathbb{G}, s_0, \dots, s_{k-1} \in \mathbb{F}_r$, where \mathbb{F}_r is scalar field of \mathbb{G} .

Output: $S = \sum_{i=0}^k s_i \cdot G_i$

⁸Using the results from <https://eprint.iacr.org/2019/1021.pdf>

4.1 Naive Algorithm

Using endomorphism:

1. $A = \infty$
2. for j from 0 to $k - 1$:
 - 2.1 $r := s_j, T := G_j$
 - 2.2 $S = [2](\phi(T) + T)$
 - 2.3 for i from $\frac{\lambda}{2} - 1$ to 0:
 - 2.3.1 $Q = r_{2i+1} ? \phi([2r_{2i} - 1]T) : [2r_{2i} - 1]T$
 - 2.3.2 $R = S + Q$
 - 2.3.3 $S = R + S$
 - 2.4 $A = A + S$

$$\text{rows} \approx k \cdot (\text{sm_rows} + 1 + 2) \approx 67k,$$

where **sm_rows** is the number of rows in the scalar multiplication circuit.

Without endomorphism:

1. $A = \infty$
2. for j from 0 to $k - 1$:
 - 2.1 $r := s_j, T := G_j$
 - 2.2 $S = [2]T$
 - 2.3 for i from $n - 1$ to 0:
 - 2.3.1 $Q = k_{i+1} ? T : -T$
 - 2.3.2 $R = S + Q$
 - 2.3.3 $S = R + S$
 - 2.4 $S = k_0 ? S - T : S$
 - 2.5 $A = A + S$

$$\text{rows} \approx k \cdot (\text{sm_rows} + 1 + 1) \approx 105k,$$

where **sm_rows** is the number of rows in the scalar multiplication circuit.

4.2 Simultaneous Doubling

Using endomorphism:

1. $A = \sum_{j=0}^k [2](\phi(G_j) + G_j)$
2. for i from $\frac{\lambda}{2} - 1$ to 0:
 - 2.1 for j from 0 to $k - 1$:
 - 2.1.1 $r := s_j, T := G_j$
 - 2.1.2 $Q = r_{2i+1} ? \phi([2r_{2i} - 1]T) : [2r_{2i} - 1]T$
 - 2.1.3 $A = A + Q$
 - 2.2 if $i \neq 0$:
 - 2.2.1 $A = 2 \cdot A$

$$\text{rows} \approx \frac{\lambda}{2} \cdot (k \cdot \text{add_rows} + \text{dbl_rows}) + 2k \approx 64 \cdot (k + 1) \approx 66k + 64,$$

where

- **add_rows** is the number of rows in the addition circuit.
- **dbl_rows** is the number of rows in the doubling circuit.

Without endomorphism:

1. $A = \sum_{j=0}^k [2]G_j$
2. for i from $n - 1$ to 0:
 - 2.1 for j from 0 to $k - 1$:
 - 2.1.1 $r := s_j, T := G_j$
 - 2.1.2 $Q = k_{i+1} ? T : -T$
 - 2.1.3 $A = A + Q$
 - 2.2 if $i \neq 0$:
 - 2.2.1 $A = 2 \cdot A$
3. $A = A + \sum_{j=0}^k [1 - s_{j,0}]G_j$

$$\text{rows} \approx \frac{2}{5}n \cdot (k \cdot \text{add_rows} + \text{dbl_rows}) + k \approx 103 \cdot (k + 1) + 2k \approx 104k + 103,$$

where

- **add_rows** is the number of rows in the addition circuit.
- **dbl_rows** is the number of rows in the doubling circuit.

5 Poseidon Circuit

WIP

Mina uses Poseidon hash with width = 3. Therefore, each permutation state is represented by 3 elements and each row contains 5 states.

Denote i -th permutation state by $T_i = (T_{i,0}, T_{i,1}, T_{i,2})$.

Row	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
i	$T_{0,0}$	$T_{0,1}$	$T_{0,2}$	$T_{4,0}$	$T_{4,1}$	$T_{4,2}$	$T_{1,0}$	$T_{1,1}$	$T_{1,2}$	$T_{2,0}$	$T_{2,1}$	$T_{2,2}$	$T_{3,0}$	$T_{3,1}$	$T_{3,2}$
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
$i + 10$	$T_{50,0}$	$T_{50,1}$	$T_{50,2}$	$T_{54,0}$	$T_{54,1}$	$T_{54,2}$	$T_{51,0}$	$T_{51,1}$	$T_{51,2}$	$T_{52,0}$	$T_{52,1}$	$T_{52,2}$	$T_{53,0}$	$T_{53,1}$	$T_{53,2}$
$i + 11$	$T_{55,0}$	$T_{55,1}$	$T_{55,2}$	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots	\vdots

State change constraints:

$$\text{STATE}(i + 1) = \text{STATE}(i)^\alpha \cdot \text{MDS} + \text{RC}$$

Denote the index of the first state in the row by **start** (e.g. **start** = 50 for 10-th row). We can expand the previous formula to:

- For i from **start** to **start** + 5:
 - $T_{i+1,0} = T_{i,0}^5 \cdot \text{MDS}[0][0] + T_{i,1}^5 \cdot \text{MDS}[0][2] + T_{i,2}^5 \cdot \text{MDS}[0][2] + \text{RC}_{i+1,0}$
 - $T_{i+1,1} = T_{i,0}^5 \cdot \text{MDS}[1][0] + T_{i,1}^5 \cdot \text{MDS}[1][2] + T_{i,2}^5 \cdot \text{MDS}[1][2] + \text{RC}_{i+1,1}$
 - $T_{i+1,2} = T_{i,0}^5 \cdot \text{MDS}[2][2] + T_{i,1}^5 \cdot \text{MDS}[2][2] + T_{i,2}^5 \cdot \text{MDS}[2][2] + \text{RC}_{i+1,2}$

Notice that the constraints above include the state from the next row (**start** + 5).

6 Other Circuits

WIP

7 Bringing it all together

WIP

References