n: the number of dimensions in space (n = 2).
m: the order of the curve passing through a space.
N: the number of bits in a *derived-key*, $N = nm$.
i: the number of the iteration of the algorithm, $i \in [1, m]$.
byte: a word containing n bits.
$\omega^{\mathbf{i}}$: A byte of n bits where

$$\omega^i = \omega^{i-1} \oplus \bar{\tau}^{i-1}, \omega^1 = 0\ 0\ \cdots\ 0\ 0.$$

$\alpha^{\mathbf{i}}$: A byte of n bits where

$$\alpha^i = \omega^i \oplus \bar{\sigma}^i.$$

$\mathbf{a_j}$: a coordinate in dimension j of the point, $(a_1, a_2, \cdots, a_n)$ whose *derived-key* is r. A coordinate is also expressed as a real number in the range $[0, 1)$.
$\alpha_{\mathbf{j}}^{\mathbf{i}}$: a binary digit in a coordinate a_j, such that

$$\alpha_j = a_j^i \cdots \alpha_n^i = a_n^i.$$

principal position: the last, or least significant, bit position, j, in ρ^i such that $\rho_j^i \neq \rho_n^i$. If all bits in ρ^i are equal, the principal position is the nth, or least significant. The most significant bit position is considered to occupy position 1.
parity: the number of bits in a byte which are set to 1 modulo 2.
$\mathbf{J_i}$: An integer between 1 and n equal to the subscript of the principal position of ρ^i.

$\bar{\sigma}^{\mathbf{i}}$: A byte of n bits where

$$\bar{\sigma}^i = \alpha^i \oplus \omega^i, \bar{\sigma}^1 = \alpha^1$$

There is no shift in $\bar{\sigma}^1$.
$\sigma^{\mathbf{i}}$: A byte of n bits, such that

$$(J_1 - 1) + (J_2 - 1) + \cdots + (J_{i-1} - 1)$$

ρ^i represents the ith byte of n bits in r, such that

$$\rho_1^i = \rho_1^i, \rho_2^i = \sigma_2^i \oplus \sigma_1^i, \cdots, \rho_n^i = \sigma_n^i \oplus \sigma_{n-1}^i.$$

$\tau^{\mathbf{i}}$: A byte of n bits obtained by complementing σ^i in the nth position and then, if and only if the resulting byte is of odd parity, complementing in the principal position. Hence, τ^i is always of even parity. Note that the parity of σ^i is given by the bit ρ_n^i and that a mask for performing the second complementation may be set up in the same process which calculates J_i.
An algorithm for finding τ^i follows:

1. **if** $\rho^i < 3$ **then**
2. $\quad\tau^i := 0$
3. **else**

4. **if** $\rho^i \% 2 == 0$ **then**

5. $\quad \tau^i := (\rho^i - 1) \oplus (\rho^i - 1)/2$

6. **else**

7. $\quad \tau^i := (\rho^i - 2) \oplus (\rho^i - 2)/2$

8. **end if**

9. **end if**

$\bar{\tau}^{\mathbf{i}}$: A byte of n bits obtained by shifting τ^i in exactly the same way as σ^i is derived from $\bar{\sigma}^i$.

$\bar{\tau}^{\mathbf{i}}$: A byte of n bits obtained by shifting τ^i in exactly the same way as σ^i is derived from $\bar{\sigma}^i$.