Philox Engine Correction Proposal

Visual inspection of various features of the framework

Document #:	D0000R1
Date:	2025-01-10
Project:	Programming Language C++
Audience:	Library Evolution Library
Revises:	D0000R0

Reply-to:	Author Juan Lucas Rey <juanlucasrey@gmail.com>

1 Philox Corrections

This document aims to correct certain flaws in the philox engine description as it is written in "p2075r6.pdf"

The STD currently says:

For philox4x32 only:
-the const template parameters are 0xD2511F53, 0x9E3779B9, 0xCD9E8D57, 0xBB67AE85.

For philox4x64 only:
-the const template parameters are 0xD2E7470EE14C6C93, 0x9E3779B97F4A7C15, 0xCA5A826395121157, 0xBB67AE8584CAA73B.

For both philox4x32 and philox4x64:
-the word permutation table is (0, 3, 2, 1)
-Lastly the update equations are
$X_{2k} = \text{mullo}(V_{2.k+1}, M_k, w)$
$X_{2.k+1} = \text{mulhi}(V_{2.k+1}, M_k, w)$ xor $((K_k + q.C_k) \bmod 2^w)$ xor $V_{2.k}$

The implementation following these indications has been written in this repo However, as shown in this test, the random numbers do not respect the condition "If the default-constructed engine is of type std::philox4x64, the 10000th consecutive invocation of it produces the value 3409172418970261260."

The following modifications to the description are able to generate this number:
For philox4x32 only:
-the const template parameters are 0xCD9E8D57, 0x9E3779B9, 0xD2511F53, 0xBB67AE85.

For philox4x64 only:
-the const template parameters are 0xCA5A826395121157, 0x9E3779B97F4A7C15, 0xD2E7470EE14C6C93, 0xBB67AE8584CAA73B.

Note that only the multipliers are inverted. This is in line with the multipliers in the original paper, where, in page 7 it says: "This procedure yielded the following multipliers: for Philox-4x32, 0xCD9E8D57 and 0xD2511F53; for Philox-2x64, 0xD2B74407B1CE6E93, and for Philox 4x64, 0xCA5A826395121157 and 0xD2E7470EE14C6C93." In the original paper, the multiplier 0xCD9E8D57 comes BEFORE 0xD2511F53 and

0xCA5A826395121157 comes BEFORE 0xD2E7470EE14C6C93.

For both philox4x32 and philox4x64:
-Also, the word permutation table should be (2, 1, 0, 3)
-Lastly the update equations should be
$X_{2k} = \text{mulhi}(V_{2.k}, M_k, w)$ xor $((K_k + q.C_k) \bmod 2^w)$ xor $V_{2.k+1}$
$X_{2.k+1} = \text{mullo}(V_{2.k}, M_k, w)$

This implementation has been written and, as shown in a test, it satisfies the condition "If the default-constructed engine is of type std::philox4x64, the 10000th consecutive invocation of it produces the value 3409172418970261260."

2 Cases n=8 and n=16

Given that the original paper about philox has no informatin about n=8 or n=16, I would also like to inquire as to the origin of the permuation parameters for these 2 final cases. Would it be possible to include 10000th consecutive invocations for a particular implementations of Philox with n=8 and n=16?