# Cryptology complementary TP

2018-03-29

The goal of this "TP" lab session is to implement the multiplication of two numbers modulo the "MSIAM" prime $p = 2^{111} - 37$, in a reasonably efficient way that exploits the fact that $p$ is close to a power of two.

## Instructions & grading

All functions and programs must be written in C or C++

This TP is graded as *contrôle continu*. You must send a written report (in a portable format) detailing your answers to the questions, and the corresponding source code **with compilation and execution instructions** by April 16 (2018-04-06T23:59+0200) to:

pierre.karpman@univ-grenoble-alpes.fr.

Working in teams of two is allowed (but not mandatory), in which case only one report needs to be sent, with the name of both students clearly mentioned.

By default, this mandatory contrôle continu grade will count for one third of the final grade for this course, the other two thirds being the grade of a final exam.

## Objective

As stated above, we wish to implement the multiplication in the field $\mathbb{F}_{2^{111}-37}$. This will be done through a function `mul111_37` with the following signature:

```
void mul111_37(uint64_t f[4], uint64_t g[4], uint64_t h[4])
```

and the following semantics: `f` and `g` are 4-quadword arrays containing a representation of numbers in $\mathbb{F}_{2^{111}-37}$, and after executing `mul111_37(f,g,h)`, `h` contains a representation of their product. These representations are of the following form: if one writes $0 \leq f < 2^{111} - 37$ as $2^{84}f_3 + 2^{56}f_2 + 2^{28}f_1 + f_0$, with $f_i < 2^{28}$ for all $i$ (and $f_3 < 2^{27}$), then $f$ is represented as `f` with `f[i]` $= f_i$ for all $i$.

## Question 1

By noticing that one has $2^{111} \equiv 37 \mod p$ and $2^{112} \equiv 74 \mod p$, give a simple expression of $h = (2^{84}f_3 + 2^{56}f_2 + 2^{28}f_1 + f_0) \times (2^{84}g_3 + 2^{56}g_2 + 2^{28}g_1 + g_0) = 2^{84}h_3 + 2^{56}h_2 + 2^{28}h_1 + h_0 \mod 2^{111} - 37$. In particular, we take $h_0$ to be equal to $f_0g_0 + 74f_1g_3 + 74f_2g_2 + 74f_3g_1$, and we want to find a similar expression for $h_{1...3}$. Note that we do not require this expression to be already reduced mod $2^{111} - 37$, that is we do not ask (yet) for $2^{84}h_3 + 2^{56}h_2 + 2^{28}h_1 + h_0$ to be smaller than this modulus. We do not require either all of the $h_i$ to be smaller than $2^{28}$.

## Question 2

Let again $h$ be of the form $2^{84}h_3 + 2^{56}h_2 + 2^{28}h_1 + h_0$. Give a procedure to compute a representation $2^{84}h'_3 + 2^{56}h'_2 + 2^{28}h'_1 + h'_0$ for the same number where $h'_0, h'_1, h'_2 < 2^{28}$.

## Question 3

Given a number of the form $2^{84}h_3 + 2^{56}h_2 + 2^{28}h_1 + h_0$, where $h_0, h_1, h_2 < 2^{28}$, give a simple procedure to reduce this number modulo $2^{111} - 37$.

## Question 4

By combining the procedures of the two previous questions, give an algorithm that reduces $2^{84}h_3+2^{56}h_2+2^{28}h_1+h_0$ modulo $2^{111}-37$ and stores the result as $2^{84}h'_3+2^{56}h'_2+2^{28}h'_1+h'_0$ where $h'_0, h'_1, h'_2 < 2^{28}$, $h'_3 < 2^{27}$.

## Question 5

Implement `mul111_37` by first using the expression of the (non-reduced) product of Question 1, and then reducing it as in Question 4. It is advised to test your algorithm on random inputs, and e.g. compare its results with Sage.

## Question 6

Explain why all of the quantities used in your implementation are less than $2^{64}$, and thus why it may safely use arithmetic with variables of type `uint64_t`.

## Remark

Implementations similar to this one may be used in certain cryptographic algorithms, for instance *polynomial MACs* or schemes based on elliptic curves. They would however probably use a slightly "more efficient" prime modulus, and exploit the fact that modern CPUs have $64 \times 64 \to 128$ bit multipliers, or vectorized multiplication. The basic approach could however be the same.