# Abstract

Modulo arithmetic is the essential building block for many cryptographic algorithms in post quantum cryptographic (PQC) schemes. There are several fundamental operations including modulo reduction, modulo addition, modulo subtraction, modulo multiplication and modulo exponentiation. These arithmetic operations plays a critical role in the performance of PQC schemes. In the recent open literature, there are several approaches and algorithms to optimize these execution units to speed up the post quantum cryptographic schemes. However, it is not clear which algorithm performs better when considering a parameter set of a PQC scheme. If prime numbers used in these modulo operations is constant and fixed, then a better computational performance in terms of performance can be achieved with a prime-specific hardware design.

We have had research on various acceleration and efficiency enhancement techniques within the scope of modular arithmetic, as well as algorithms related to these techniques. As a result of these studies, we focused on Barrett Reduction and reduction for a fixed prime.

Until this process, the application of modular arithmetic operations by working with Barret Reduction logic with 48-bit numbers as 24x24 bits is implemented. In this part of this project, tests of the Barret Reduction algorithm for 64-bit numbers as 32x32 bits and reduction or a fixed prime as 64-bit which is prepared software by creating hardware algorithms are conducted. As a result of these tests, the most appropriate algorithm for this project by making our observations about efficiency has determined, which is the main goal of the project.

# Introduction

Modular arithmetic core of Barrett Reduction in our algorithm consists of a 32-bit adder, a 32-bit substractor and a 32-bit multiplier along with associated modular reduction logic. The modular design involves a pair of adder and subtractor. In the operations, it uses carry, borrow and difference bit (c, b, d, s). Conditional addition and subtraction operations are performed. The purpose of these operations is to perform the modular reduction operation. These processes are computed in the same cycle, as this avoids side-channels. If many instances of x.y mod q for a fixed module q (0≤xy<q2) wanted to be calculated, reduction operations using Barrett reduction algorithm can be made faster. One of the main reasons for applying this method is that these operations are faster operations than division. A similar method is followed by Barrett Reduction logic in the application of reduction operations for fixed primes. Modular multiplier implementations for some special prime numbers used by NIST Round 1 lattice-based candidates. Some of these special fixed primes are: 7681 (CRYSTALS-Kyber), 12289 (NewHope), 40961 (R.EMBLEM), 65537 (pqNTRUSign), 120833 (Ding Key Exchange), 133121 / 184321 (LIMA), 8380417 (CRYSTALS-Dilithium).

# Method

At the first stage, the Barrett Reduction algorithm with 32x32 bit-length numbers and reduction for a fixed prime as 64-bit in accordance with the algorithm has applied that founded and selected in accordance with researches on post-quantum cryptography schema parameters. This application is provided by converting our algorithm to VHDL. As applying the Barrett Reduction and reduction for a fixed prime algorithm VHDL code for the algorithms has produced. The hardware design tests has been completed and applied to the zedboard development board, using the Vivado Design package. By this way, at the final it is applied and observed on oled screen of Xilinx Zynq-7000 SoC architecture zedboard development kit.

## 2.1. Barret Reduction

A 32-bit multiplier is used followed by a prime q of size up to 32 bits. Let z be the 64-bit product to be reduced to Zq, Barrett computes z mod q with the quotient [z/q] without performing any division. Barrett reduction contains two multiplications, one subtraction, one bit-shift and one conditional subtraction. 1/q value is approximated as m/2k. The error of approximated as e = 1/q − m/2k, hence the reduction is valid as long as ze < 1. While z < q2 , k is set to be the smallest number accordingly e = 1/q − ([2k/q]/2k) < 1/q2. Ordinarily, k value is very close to 2[lg q], that means the bit-size of q2. In process of reduction for fixed prime numbers, if the modulus of x mod q desired to be calculated.

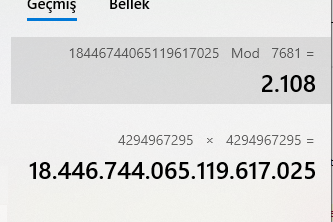


Figure 1: X= 429496725, Y= 4294967295

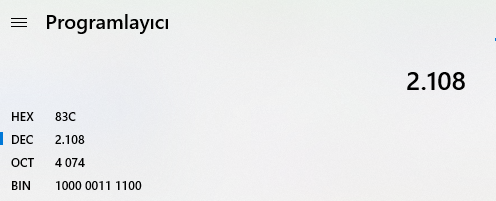


Figure 2: Result for Z = 18446744065119617025

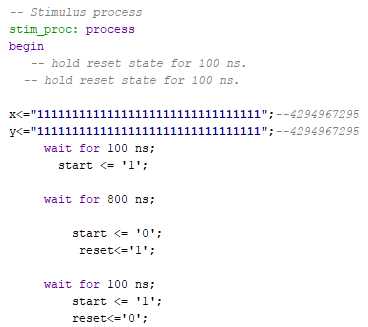


Figure 3: Testbench of the Barret Reduction

## Reduction for a Fixed Prime

If many instances of x mod q desired to be calculated, based on the reduction for fixed prime method can be done with the help of fixed prime numbers that are previously determined. There is a special reduction block for each of the special fixed primes mentioned for reduction for a fixed prime. Reduction blocks have parameters q, m and k encoded in digital logic, and in addition, these blocks do not need open multipliers. This can create a reduction in flexibility and the need for additional field, but because of this, the reduction method for fixed primes involves fewer calculations than the classic Barrett reduction circuit.The reduction becomes particularly efficient when at least one of m and q or both can be written in the form 2a1 ± 2a2 ± 2a3… ± 1. As we implement in the project, for example, if we define the value q as 7681 that equals 213 − 29 + 1. k value equals 21 and m value equals 273 which is 28+24+1. For this reason, the multiplications by q and m can be converted to significantly cheaper bit-shifts and additions or subtractions. Implementation details and reduction parameters for 7681 modular block are in Figure 5.

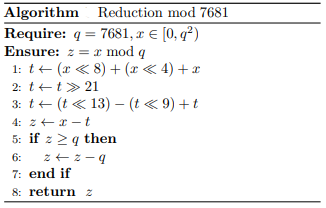


Figure 4: Pseudo-code of Reduction mod 7681

# Empirical Setup

In Vivado design studio, Modular Multiplication with Barrett Reduction algorithm has been developed and implemented in vhdl language at RTL level then simulation has developed with reference to the Xilinx Zynq-7000 SoC architecture zedboard development kit.

In Vivado design studio, Reduction for Fixed Prime Number (7681) at 64 Bits algorithm has been developed and implemented in vhdl language at RTL level and with reference to the Xilinx Zynq-7000 SoC architecture zedboard development kit.

# Empirical Results

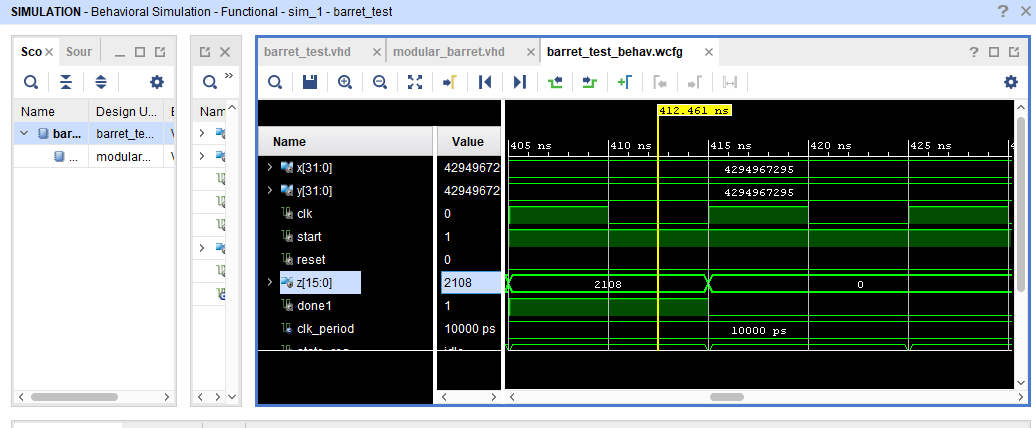


Figure 5: Simulation of Modular Multiplication with Barrett Reduction

In this simulation, the result of the 264 mod 7681 modular multiplication operation with barret reduction is given in the variable z.

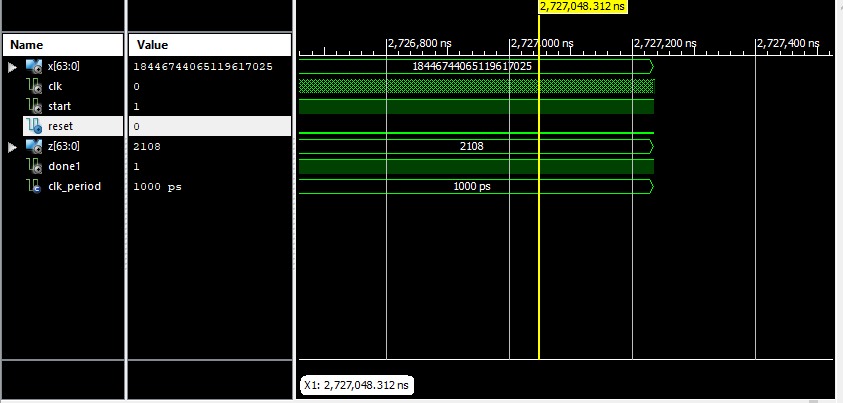


Figure 6: Simulation of Reduction for Fixed Prime Number (7681) at 64 Bits

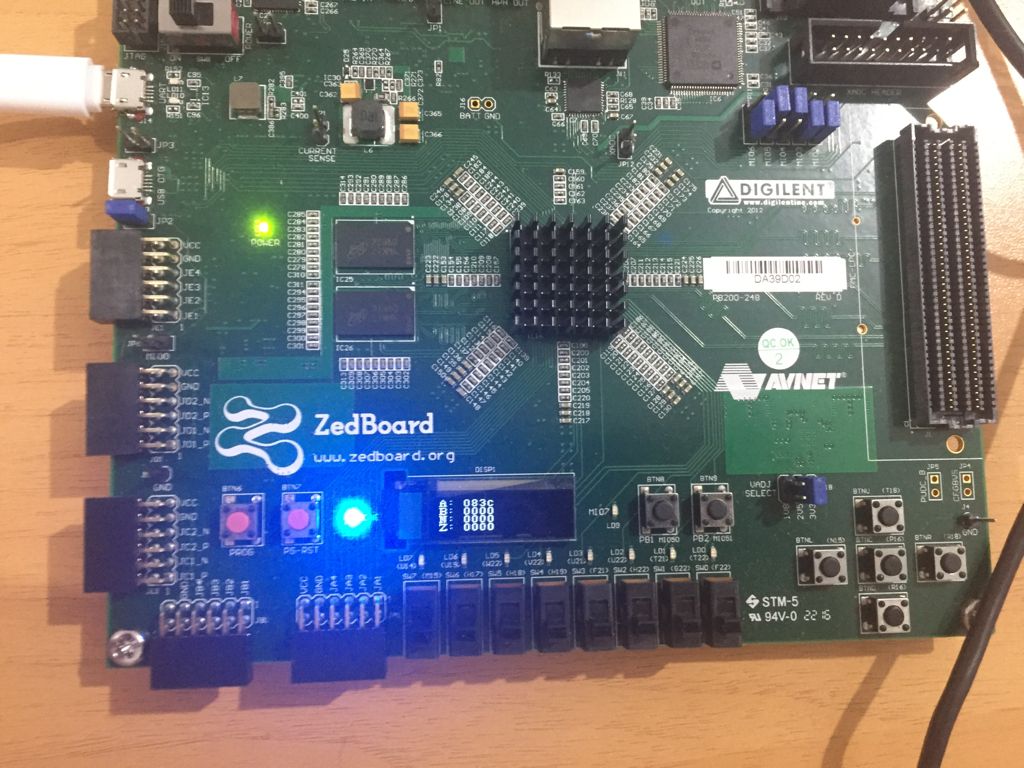


Figure 7: Oled Screen Suppressed Out Output of Z (083c)

In this simulation, the vhdl code has been developed and implemented for the 7681 value chosen as a fixed prime number. Simulation tried in 64 bits yielded successful results. In the simulation in the Figure-6 and Figure-7, the correct result, which is 2108, was found by performing the 18446744073709551615 mod 7681 operation.

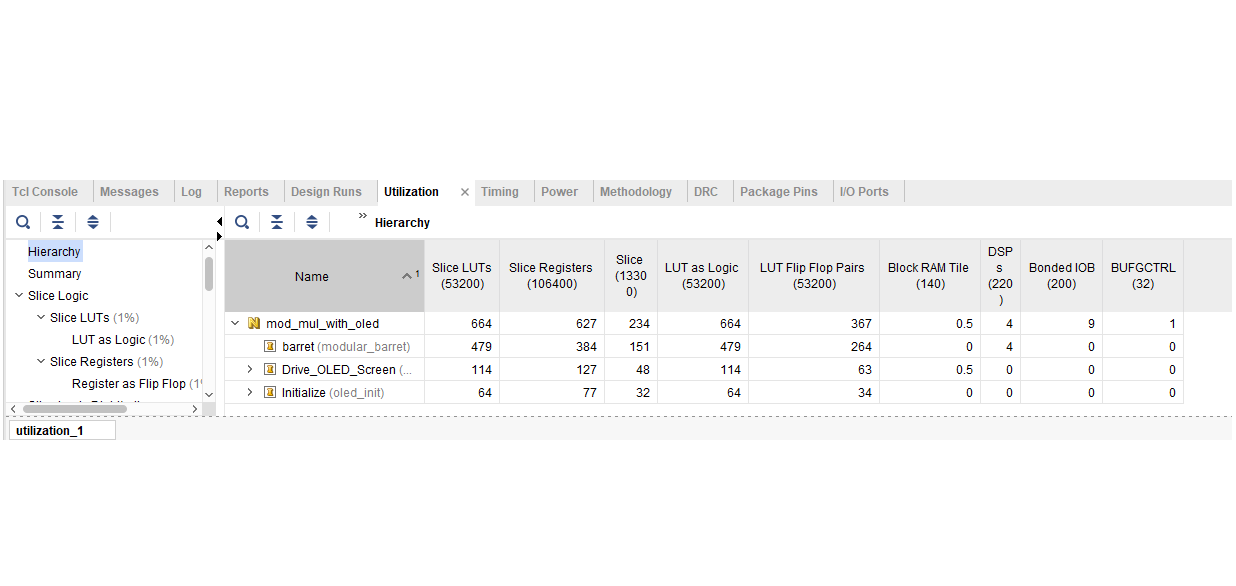


Figure 8: Utilization of Barret Reduction

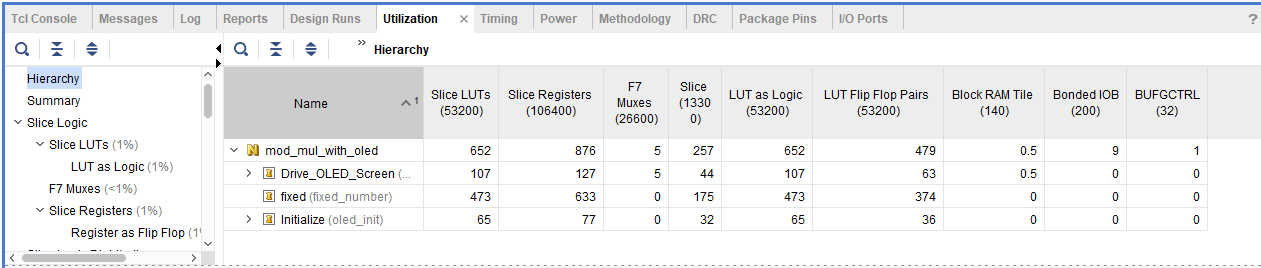


Figure 9: Utilization of Reduction of a Fixed Prime

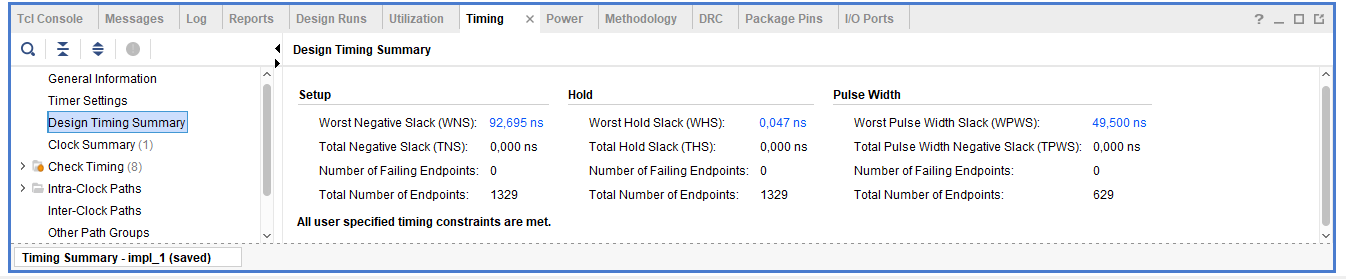


Figure 10: Timing Summary of Barret Reduction

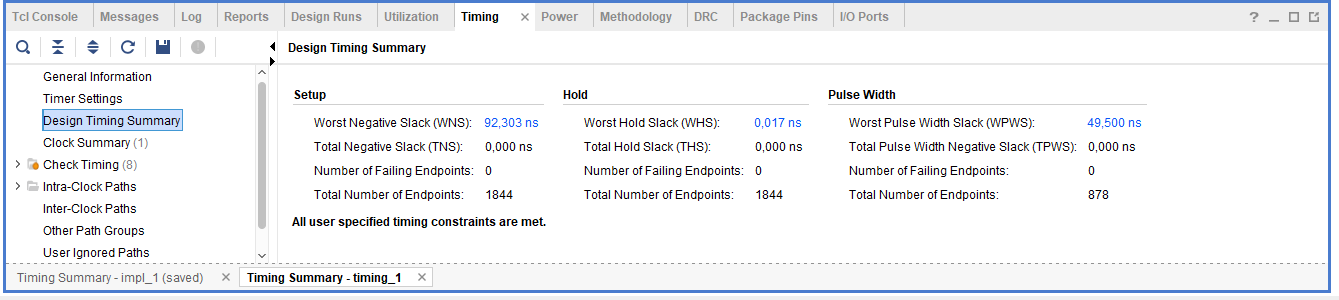


Figure 11: Timing Summary of Reduction for a Fixed Prime

# Conclusion

By evaluating the existing barret reduction and reduction for a fixed prime algorithms and converting them to VHDL code at the RTL level, the first phase of the accelerator hardware design process has been completed. Trials of these algorithm were provided on Xilinx Zynq-7000 SoC architecture zedboard development kit, and it was observed on oled screen that on this development kit.

In line with the directions provided by the barret reduction and reduction for a fixed prime algorithms, algorithms have been developed for a constant prime number. This prime number 7681 was chosen and arranged according to the modular operation of 64 bit numbers. The algorithm written in VHDL language at RTL level was tested and it was seen that the code worked correctly. The main purpose of the project was that creating an efficient modulo arithmetic accelerator. Therefore, as a result of observations of software and hardware tests, it was concluded that the reduction for a fixed prime algorithm was more advantageous when the efficiency of the algorithms applied was evaluated in terms of time, while the barret reduction algorithm was a more advantageous method when evaluated in terms of field.

# References

[1] Amanor D. (2005). “Efficient Hardware Architectures for Modular Multiplication”. The University of Applied Sciences Offenburg, Germany.

[2] Zimmermann R. (1999). “Efficient VLSI Implementation of Modulo (2^n+- 1) Addition and Multiplication”. Swiss Federal Institute of Technology (ETH), Zürich.

[3] Banerjee U., Ukyab T., Chandrakasan A.(2019)” A Configurable Crypto-Processor for Post-Quantum Lattice-based Protocols”. Dept. of EECS, Massachusetts Institute of Technology, Cambridge.

[4] Wikipedia(2021). Montgomery modular multiplication, Erişim tarihi: 25.04.2021,

<https://en.wikipedia.org/wiki/Montgomery_modular_multiplication>