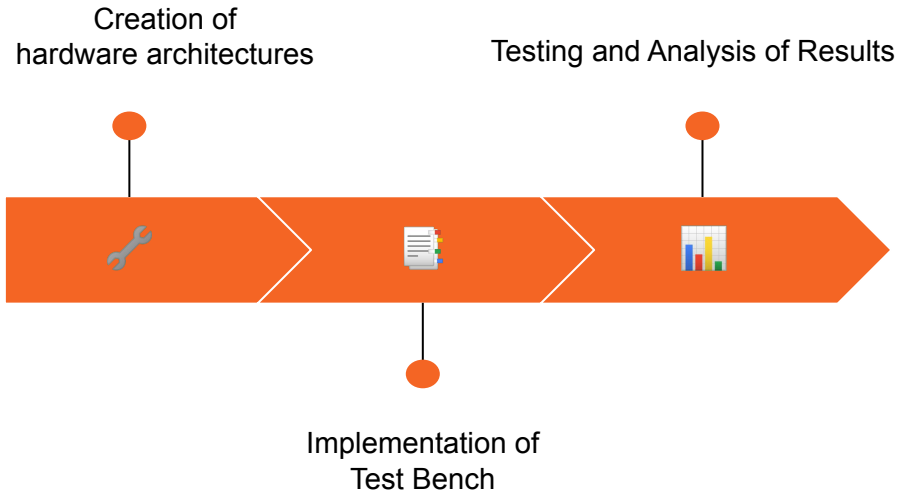
Computer Engineering Project

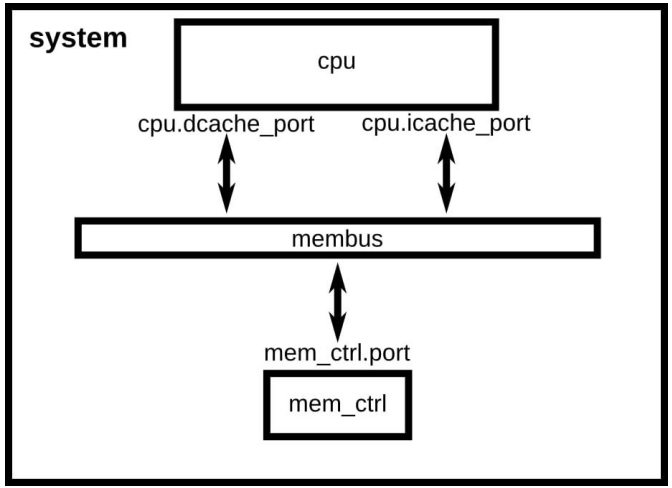


The aim of the project is to instantiate hardware architectures that differ both in structure and in computational capacity in order to test the impact of the AES algorithm on system performance and how performance varies with cryptographic and system characteristics variation.

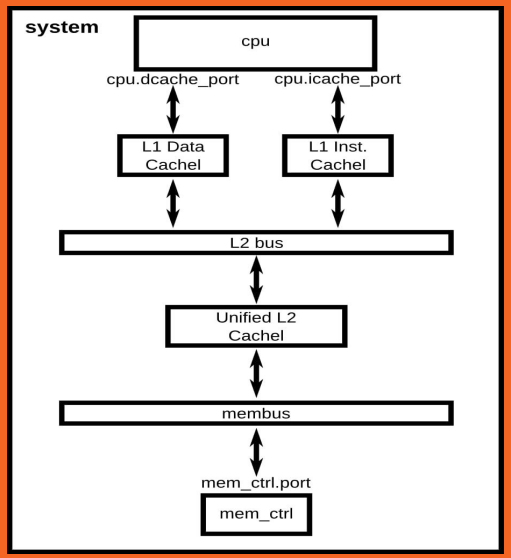
Giuseppe Calcagno

Simple

The first "Simple" architecture is composed of a processor directly connected to a central memory via a bus



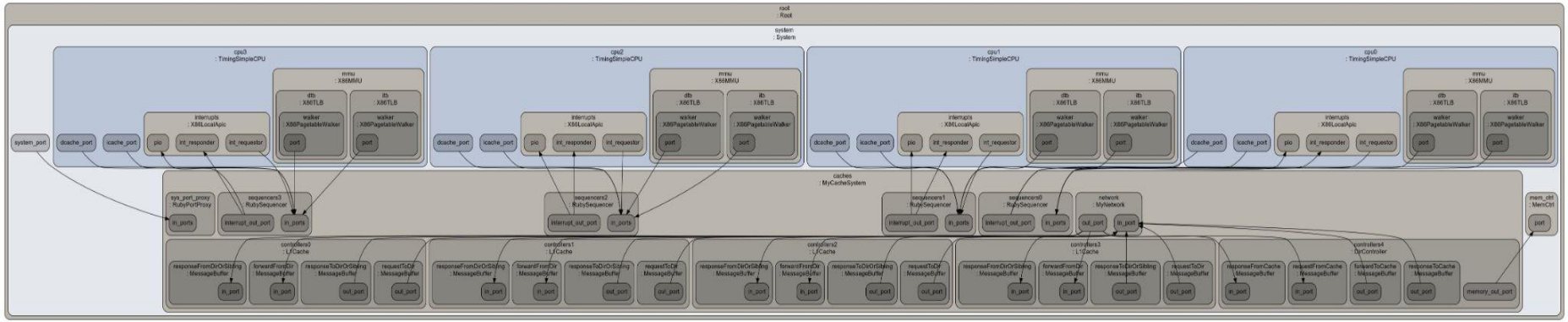
Two Level Cache



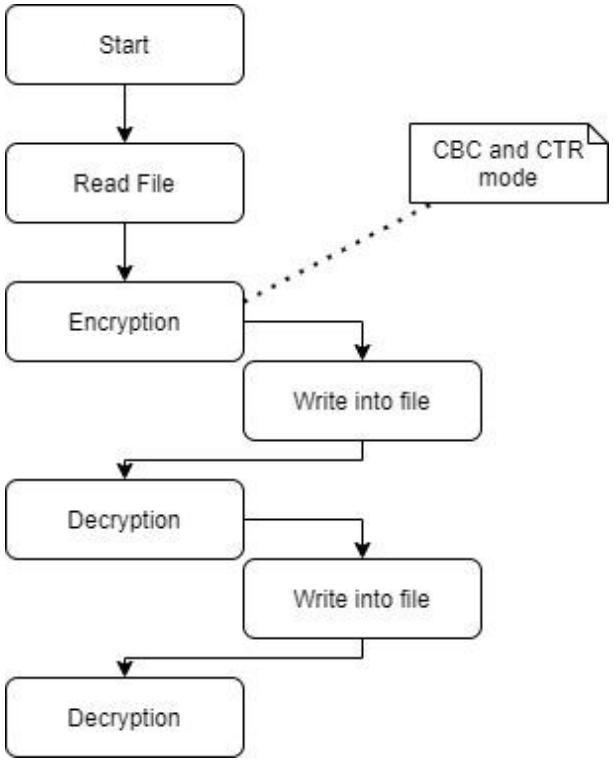
The architecture “two level cache" consists of a processor connected to 2 cache levels and a central memory. Caches have different performance for access and data transfer times

Ruby System Architecture

The Ruby architecture is composed of 4 processors connected to a system of 4 small caches managed by a consistency protocol. The architecture takes its name from the language used to define the cache system. Memories are linked together through a simple point-to-point connection



Test Bench



The test bench reads a file from memory, the content is encrypted and the result is written to a new file, then it is decrypted and the plain text is written into the same file for a consistency check with the original content.

Two other programs were created to test the time used for file management and parallelization of the AES algorithm

Executed Test

1 Standard Test

I started using the test bench program with files of 64, 2048 and 16384 bytes using different sizes of the encryption key (128, 192 and 256 bits) on all architectures

2 File-Time Test

For greater accuracy I decided to modify the test bench to collect data regarding the time required to manage the file (without encryption)

3 Multi-Threading and CBC-CTR differences

I used the threads in the test bench to see the improvement in performance. For a more accurate analysis, I also tested the differences between the 2 encryption modes used

Analysis of the results

The analysis of the results focused on these points:

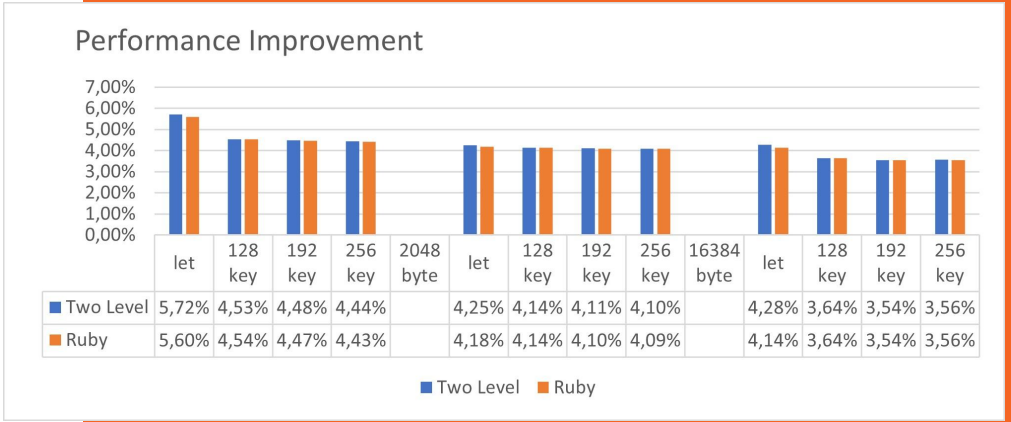
- difference in performance on different architectures.
- Impact on the system of an AES algorithm
- variation in performance as the file and key size varies.
- performance improvement thanks to the parallelization of operations
- performance difference in based on the type of encryption used.

All simulations were performed in gem5 using timing mode: ”reflect our best effort for realistic timing and include the modeling of queuing delay and resource contention. Once a timing request is successfully sent at some point in the future the device that sent the request will either get the response or a NACK if the request could not be completed”.

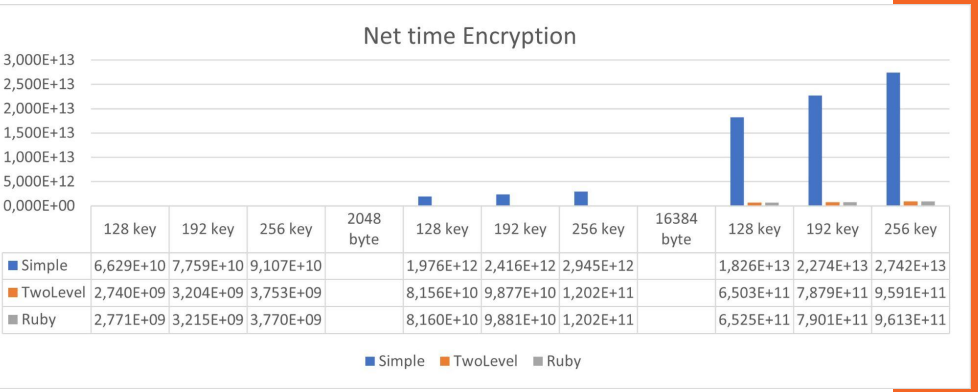
All results are expressed in ticks.
(ticks per second: 10^12)

Some Results

In the graph below we see a representation of the time taken by the different architectures to complete the test bench. As expected the simple architecture has the worst performance, while the use of a single processor by the Ruby architecture brings the other 2 architectures similar results



	let	128 key	192 key	256 key	2048 byte	let	128 key	192 key	256 key	16384 byte	let	128 key	192 key	256 key
Two Level	5,72%	4,53%	4,48%	4,44%		4,25%	4,14%	4,11%	4,10%		4,28%	3,64%	3,54%	3,56%
Ruby	5,60%	4,54%	4,47%	4,43%		4,18%	4,14%	4,10%	4,09%		4,14%	3,64%	3,54%	3,56%

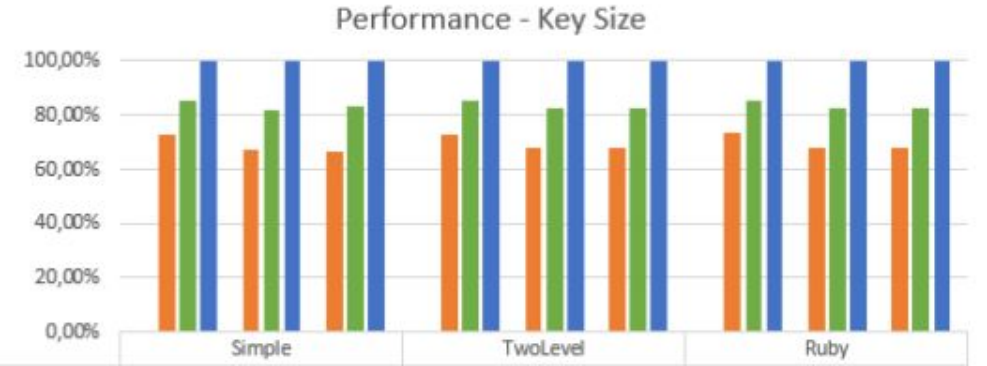


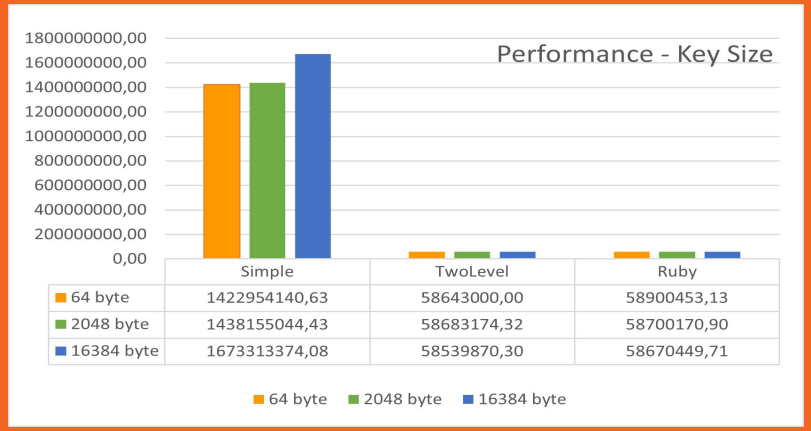
	128 key	192 key	256 key	2048 byte	128 key	192 key	256 key	16384 byte	128 key	192 key	256 key
Simple	6,629E+10	7,759E+10	9,107E+10		1,976E+12	2,416E+12	2,945E+12		1,826E+13	2,274E+13	2,742E+13
TwoLevel	2,740E+09	3,204E+09	3,753E+09		8,156E+10	9,877E+10	1,202E+11		6,503E+11	7,879E+11	9,591E+11
Ruby	2,771E+09	3,215E+09	3,770E+09		8,160E+10	9,881E+10	1,202E+11		6,525E+11	7,901E+11	9,613E+11

At the top instead we see a percentage comparison between the different architectures and the “simple” one. We note that the 2 most powerful take less than 10% of the time taken by the simple architecture to finish the test

Some Results

In the graph below we see the percentage difference between the tests carried out with 256 keys compared to 192 and 128 bit keys

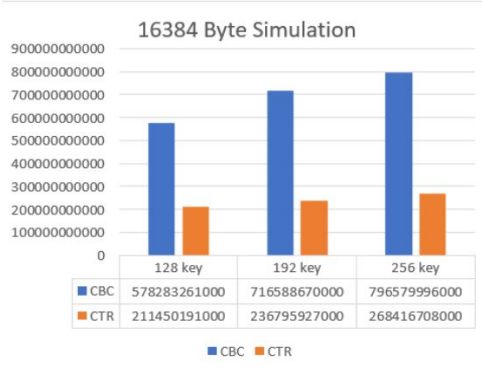


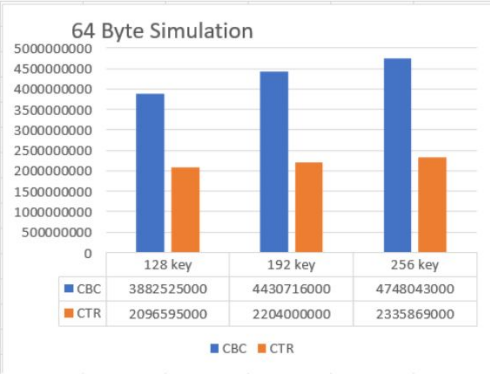


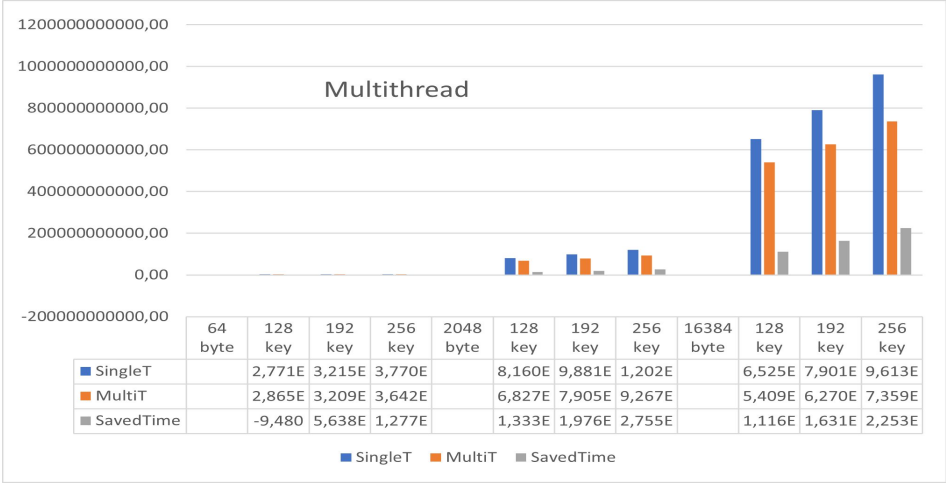
Above instead we see the difference between the test benches carried out with files of different sizes

Some Results

The graph below shows the differences between the tests performed in single or multi thread on the Ruby architecture (the only one that benefits from multithreading)







Above, on the other hand, the graph highlights the differences between the 2 cryptographic methods used (CBC and CTR)

All the data, code and scripts mentioned in these slides can be found on GitHub together with a more in-depth analysis of the data at the link:
Giuseppe-Calcagno/Progetto-IngInf (github.com)

Special thanks go to Prof. Christian Pilato for his support and help during the (many) difficult moments.