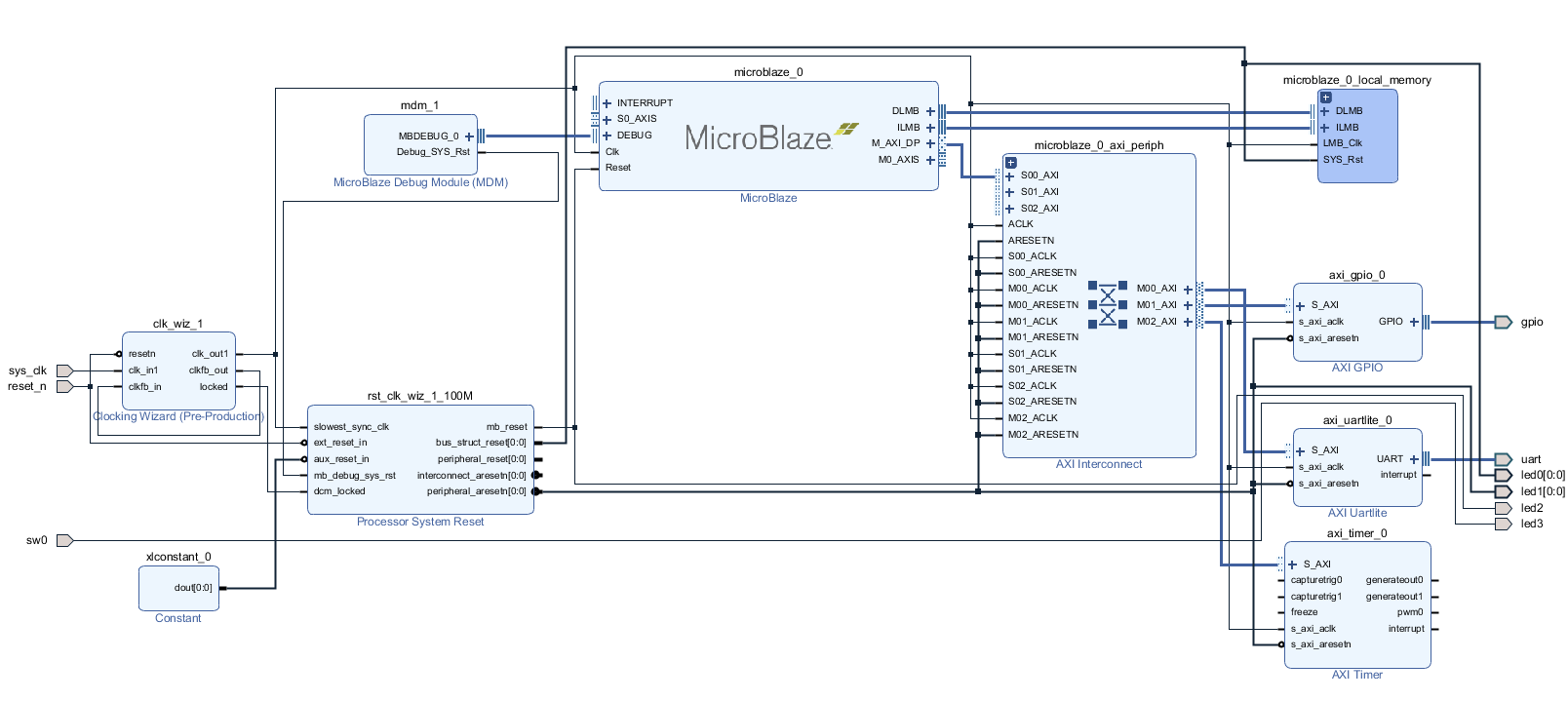
Abstract

Field Programmable Gate Array (FPGA) designs have become widely adopted with their flexibility in accelerating performance of specific applications at a low power cost. This flexibility comes with the desire to also have customizable designs to suit the application. In this survey, we will perform tests on the Xilinx MicroBlaze processor within a 7-Series FPGA. Each of these tests enables some features on the processor while disabling others. Minimum area, high performance, maximum frequency, and frequency optimized implementations will be produced in order to survey tradeoffs that exist within these various architectures. The application that will be chosen to survey these tradeoffs will be an encryption and decryption algorithm. Our tests will demonstrate the tradeoff of power and area for performance, with an emphasis on tradeoffs for embedded cryptography applications.

System Architecture



The system architecture consists of a typical FPGA design utilizing a soft processor such as the MicroBlaze. The MicroBlaze system, along with many other designs, utilizes the AMBA Advanced eXtensible Interface (AXI) as the bus. The processor acts a master in order to control peripherals and access memory. The data and instruction memory lies on block RAM acting as a slave on the AXI bus. Additional slaves utilized for the purpose of testing include the AXI Uartlite and AXI timer submodules. The uartlite acts as an interface between AXI and UART and allows tests to occur by loading in files to encrypt, and reading out the result for verification. The AXI timer is enabled by the processor at the start of the algorithm under test, and is critical for giving cycle-accurate readings of the execution time. A clock generator instantiates a phase locked loop (PLL) in order to produce the system’s clock, which is set to the maximum achievable frequency in the particular implementation.

Implementations

Six distinct implementations of the processor were chosen in order to achieve a variety of tradeoffs for analysis. Note that in each of these implementations, regardless of target, the debug mode feature was included in the MicroBlaze, this increases area by a constant amount among all the tests (about 500 LUTs/Registers).

**1) Three Stage Pipeline**

This build targets the minimum area achievable. It utilizes a three stage pipeline. This type of processor build is frequently used among FPGA designers working in the embedded field, because it allows easy entry into building a software system with minimum sacrifice of chip space. This build sacrifices performance by giving up additional hardware features, but will remain cheap in terms of power and area.

**2) Three Stage Pipeline with Multiplier & Divider**

This build augments the normal three stage pipeline by adding a multiplier and divider within the execution stage. This uses up additional space on the chip, but also decreases the maximum achievable frequency to 115 MHz.

**3) Five Stage Pipeline**

The five stage pipeline represents the most used MicroBlaze pipeline. It is the pipelined used for the maximum performance of the MicroBlaze. In this implementation, we use the five stage pipeline with no multiplier, divider, or branch table buffer (BTB). This barebones pipeline provides a baseline for the performance of the five stage pipeline.

**4) Five Stage Pipeline with Multiplier**

This implementation augments the five stage pipeline to include a multiplier unit in the execution stage. This reduces the execution time of the program signifcantly. Due to the DSP slice resource on Xilinx FPGAs, the multiplier does not add signficantly higher utilization.

**5) Five Stage Pipeline with Multiplier & Divider**

This build adds a multiplier and a divider unit in the execution stage of the five stage pipeline. This greatly reduces the execution time of the program. Resource utilization goes up a moderate amount from the addition of the divider.

**6) Five Stage Pipeline with Multiplier, Divider, & Branch Target Buffer**

Uses the same five stage pipeline with the multiplier and divider, but adds an additional branch target buffer. The BTB enhances the branch prediction capabilities of the microblaze by adding an additional cache to store the results of previous branches. In a vacuum, this would increase performance for most designs, but the MicroBlaze implementation of the BTB does reduce the maximum frequency by about 20%. This means that to see a performance increase, the addition of the BTB must make up for the lost maximum frequency.

Software & Test

In order to give an accurate depiction of the tradeoffs of the MicroBlaze system, a software application was produced that could take advantage of additional hardware acceleration. Encrypting and decrypting messages takes significant computational resources, and can be scaled with better hardware. The RSA algorithm was used, which uses prime numbers as a basis for encoding and decoding. One public key is generated for encryption, while one private key is used for decrypting the encrypted message.

Both encryption and decryption require modulo and multiplication to encode and decode. These operations can be accelerated with multiply and divide operations, which require additional hardware to be present in processors. Without this hardware, the performance of the encoding and decoding will be reduced. This tradeoff of area and power in return for performance was the basis for the tests done using this algorithm.

The test consisted of encrypting and decrypting two scripts. One of the scripts consisted of 32kB of latin text, and the other script consisted of 32kB of numbers. Different scripts were used in order to verify that the algorithm was working as intended with all characters and to record any potential variations in performance with a variety of characters. The test data was large in order to give a more accurate result. All the data from the scripts was loaded in over UART into the internal block RAM of the FPGA before the algorithm starting encrypting. This was done in order to remove any performance impact from external memory or serial communication links. The AXI Timer submodule kept a cycle-precise timer that was initialized directly before starting the encryption. After the decryption finished, the timer was stopped. This ensured that no serial communication time for reading out was included in the final time count.

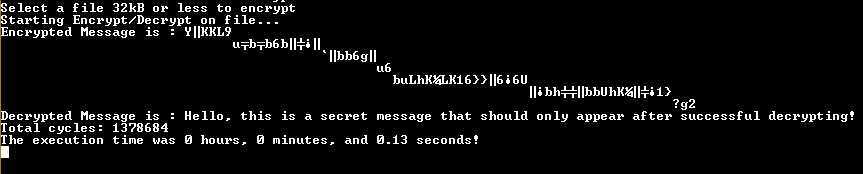


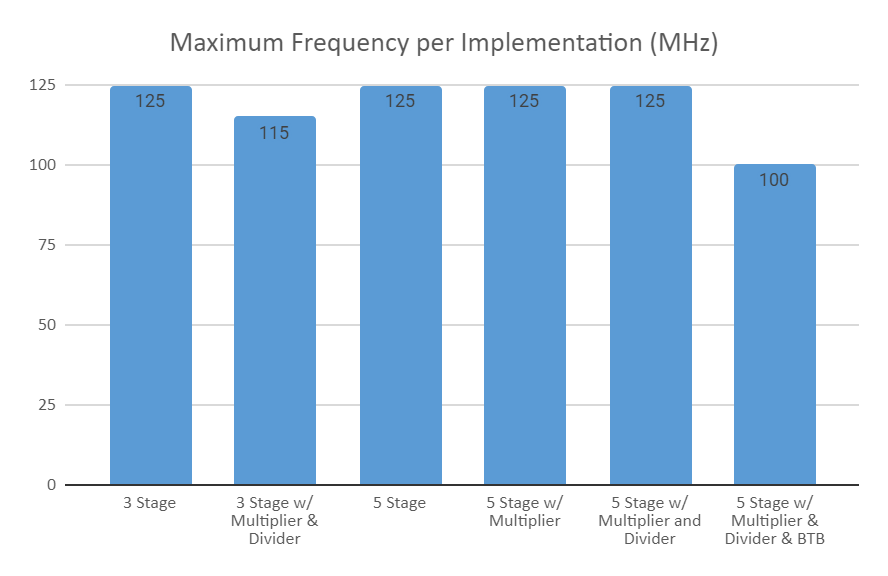
Figure: Example of a small file containing one sentence being encrypted and then subsequently decrypted.

Results

All tests and implementations were performed the Xilinx 7 series chip7s50csga324. The following details the results among different categories.

**Maximum Frequency**

For each implementation, the maximum frequency was found and used for the corresponding test. The five stage pipeline, with its increased pipeline stages, is able to hold 125 MHz even with the addition of multipliers and dividers. However, due to the longer signal paths within the three stage pipeline, the maximum frequency drops when adding multipliers and dividers. In addition to this, the branch target buffer also adds significantly to the critical path of the system. The result is that when implementing the BTB, the frequency must also be reduced by about 20%. Therefore, systems implementing the BTB must be able to take advantage of improved branch prediction in order to make up for the loss in frequency.

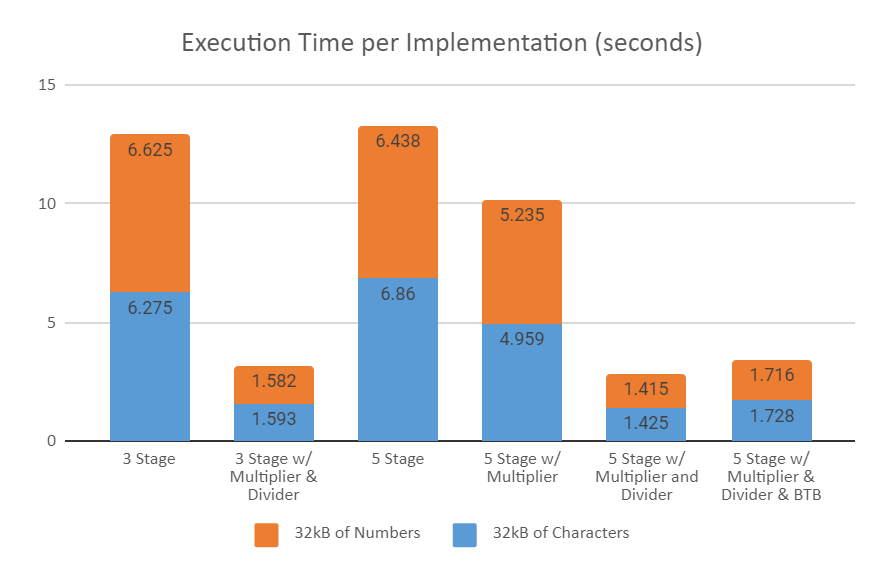


**Execution Time**

The execution time represents the total time of both encrypting and decrypting a file. The overall performance of our implementations is given by this execution time. The baseline runs, 3 & 5 stage pipelines without additional hardware had the longest execution times. This is a result of the extra cycles incurred from producing multiply and divide operations without the extra hardware. As a consequence of not having these resources, more instructions and cycles were needed to accomplish those operations.

Among all tests, the multiplier unit improved performance by about 20%. This is due to the frequency of non-power of two multiplies present in the encryption and decryption algorithm. Even greater performance benefit was seen from the divider unit, which improved performance by 60%. The modulus operator is repeated many times per each character within our cryptographic algorithm, which incurs many instructions without a true divider. The addition of the divider and multiplier drove both our 3 stage and five stage tests down almost 80% from their baseline.

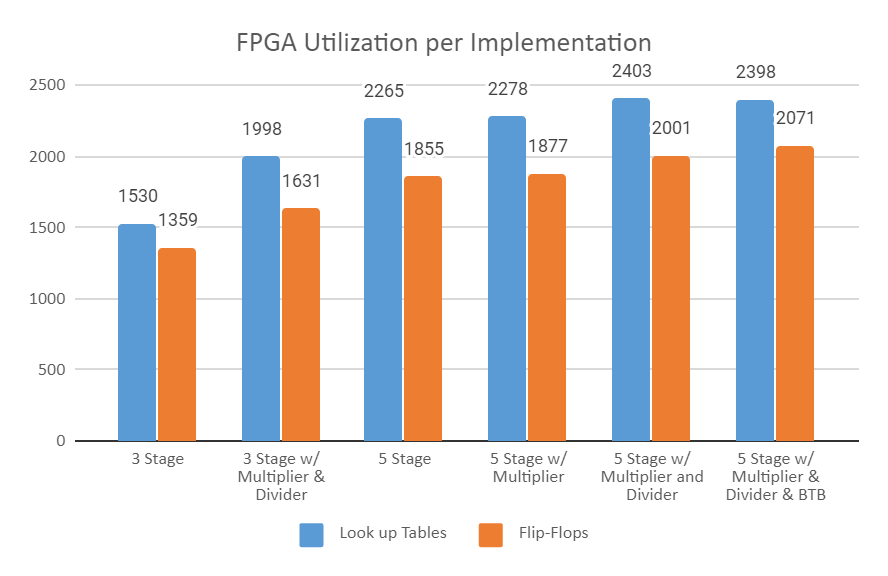
The branch target buffer is also notable, since execution time seemingly goes up slightly with the addition of the BTB. In reality, the BTB itself likely helps performance slightly, but the BTB added an additional critical path which reduced the frequency by 20%. This impact to the clock speed directly affected our design’s performance by the same amount, which is why adding the BTB did not see a system level performance increase.

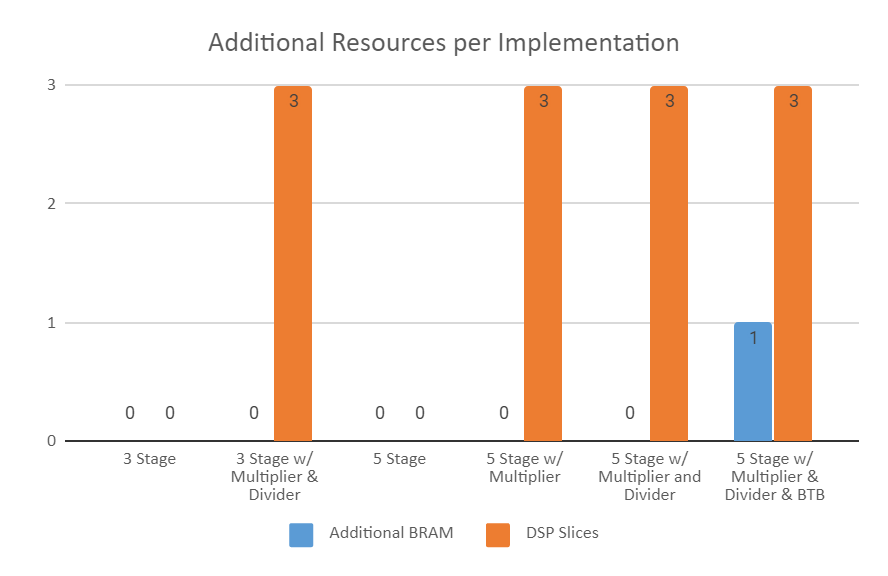


**Resource Utilization**

Resource utilization sees a significant increase when going from the three stage pipeline to the five stage pipeline. Utilization increases about 33%, which correlates to the addition of 2 extra stages with extra logic and registers added with each stage.

The benefit of Xilinx DSP slices can be seen from these results. Adding a multiplier in a pipeline sees virtually no increase in LUT or FF utilization, since the DSP slice performs the entire multiply. The additional benefit of these slices is that they have much smaller critical paths than if they were built out of traditional LUT or FF hardware. The dividers increase utilization by about 6%, but see a very large performance benefit compared to implementations without the divider. Finally, the addition of the BTB does not add significant logic, but does incur one block RAM in order to store the branch table.





**Power Utilization**

As the ability to increase speed and logic increases, power becomes more of a concern for digital designs. In this example with one MicroBlaze on a relatively low end FPGA, these power numbers appear small. However, many systems will realize multi-core architectures or chip architectures with the potential to consume more power. The results shown from this study are important due to this scalability. Although there is significant overhead shown with static power and PLL power, this is with only 5% of the FPGA being used. With 100% utilization of the FPGA, the dynamic power will play a large role in determining the overall power.

The less logic and slower the clock speed, the less power will be used. The 5 stage pipeline with BTB now demonstrates some advantage for certain systems. Due to its slower frequency, it was lower in performance even with the additional hardware. However, this also makes it consume less power, giving it a commendable performance per watt.

