ECE 545 – Project

Aaron Joe Parrish

# Build Results

# SPARTAN 6

## Build #1:

A build of my entire AES\_GCM Core, implementing GHASH as a 128x128 single-cycle multiply (ghash\_datapath.vhd).

I did not apply any timing constraints; I wanted to get an initial number to use for further builds. However, the build did not complete, it was stuck in PAR for at least 45 minutes. I believe it was having trouble routing the large multiply.

Resource Utilization (from MAP stage):

|  |  |  |
| --- | --- | --- |
| Flip Flop | 2,621 | 2% |
| LUT | 9,979 | 21% |
| Slice | Unknown, PAR did not finish |  |

I did not calculate throughput for this build because it did not complete the build process.

## Build #2

A build of entire AES\_GCM Core, implementing GHASH as a 128x32 four-cycle multiply (ghash\_datapath2.vhd).

I used synthesis and build constraints to target a 50 MHz clock speed. This time the build completed.

Resource Utilization:

|  |  |  |
| --- | --- | --- |
| Flip Flop | 2,875 | 3% |
| LUT | 4,960 | 10% |
| Slice | 1,732 | 14% |

Minimum Clock Period:

----------------------------------------------------------------------------------------------------------

TS\_clk\_i = PERIOD TIMEGRP "clk\_i" 20 ns H | SETUP | 7.850ns| **12.150ns**| 0| 0

IGH 50% | HOLD | 0.293ns| | 0| 0

----------------------------------------------------------------------------------------------------------

I did not calculate throughput for this build because the next build has better performance.

## Build #3

A build of entire AES\_GCM Core, implementing GHASH as a 128x32 four-cycle multiply (ghash\_datapath2.vhd).

I used synthesis and build constraints to target a 100 MHz clock speed. Since the previous build had so much slack, I decided to try and see if the tool could perform better with tighter constraints.

Resource Utilization

|  |  |  |
| --- | --- | --- |
| Flip Flop | 2,875 | 3% |
| LUT | 4,913 | 10% |
| Slice | 1,682 | 14% |

Minimum Clock Period:

----------------------------------------------------------------------------------------------------------

TS\_clk\_i = PERIOD TIMEGRP "clk\_i" 10 ns H | SETUP | 0.234ns| **9.766ns**| 0| 0

IGH 50% | HOLD | 0.374ns| | 0| 0

----------------------------------------------------------------------------------------------------------

Minimum Latency: Time between inputs = 13 clock cycles / (1000000 cycles /second) = **13 us**

Maximum Throughput: Using formula from Step 7, Throughput = (1.23) \* (1000000) bytes / second = **1.23 Mbits / second**

Maximum Throughput/Area: (1.23 Mbits / second) / (1,682 slices) = **(0.73 Kbits / second) / slice**

# Virtex 6

## Build #1

A build of entire AES\_GCM Core, implementing GHASH as a 128x32 four-cycle multiply (ghash\_datapath2.vhd).

I used synthesis and build constraints to target a 100 MHz clock speed. I feel that the Virtex-6 should be able to at least match Spartan-6.

Resource Utilization

|  |  |  |
| --- | --- | --- |
| Flip Flop | 2,869 | 3% |
| LUT | 4,662 | 10% |
| Slice | 1,398 | 14% |

Minimum Clock Period:

----------------------------------------------------------------------------------------------------------

TS\_clk\_i = PERIOD TIMEGRP "clk\_i" 10 ns H | SETUP | 3.868ns| **6.132ns**| 0| 0

IGH 50% | HOLD | 0.081ns| | 0| 0

----------------------------------------------------------------------------------------------------------

I did not calculate throughput for this build because the next build has better performance.

## Build #2

A build of entire AES\_GCM Core, implementing GHASH as a 128x32 four-cycle multiply (ghash\_datapath2.vhd).

I used synthesis and build constraints to target a 200 MHz clock speed. Since the previous build had so much slack, I decided to try and see if the tool could perform better with tighter constraints.

Resource Utilization

|  |  |  |
| --- | --- | --- |
| Flip Flop | 2,869 | 3% |
| LUT | 4,492 | 9% |
| Slice | 1,854 | 15% |

Minimum Clock Period:

----------------------------------------------------------------------------------------------------------

TS\_clk\_i = PERIOD TIMEGRP "clk\_i" 5 ns HI | SETUP | 0.077ns| **4.923ns**| 0| 0

GH 50% | HOLD | 0.036ns| | 0| 0

----------------------------------------------------------------------------------------------------------

Minimum Latency: Time between inputs = 13 clock cycles / (2000000 cycles /second) = **6.5 us**

Maximum Throughput: Using formula from Step 7, Throughput = (1.23) \* (2000000) bytes / second = **2.46 Mbits / second**

Maximum Throughput/Area: Maximum Throughput/Area: (2.46 Mbits / second) / (1,854 slices) = **(1.33 Kbits / second) / slice**

# Analysis

Overall, I think my results for both FPGA families are good considering the amount of time I had to design the circuit. In my Static Timing Analysis, the majority of paths are limited by routing resources. The improved routing in Virtex-6 achieved double the clock speed for the same circuit.

I would make further improvements to the design by pipelining the AES circuit. It would decrease my input latency and improve routing to run at a faster clock speed.