Synthesis Summary

# Area-Delay Product

Table 1 shows the different values of clock period and corresponding area, and area-delay products.

|  |  |  |
| --- | --- | --- |
| **Area (µm2)** | **Delay (ns)** | **Area\*Delay (ns** ·**µm2)** |
| 926598.6 | 2.56 | 2372092.372 |
| 845136.6 | 2.7 | 2281868.694 |
| 845136.6 | 2.7 | 2281868.694 |
| 926598.6 | 2.56 | 2372092.372 |
| 817226.8 | 3.12 | 2549747.653 |
| 813349 | 3.71 | 3017524.741 |
| 815359.9 | 4.17 | 3400050.93 |
| 773002.8 | 4.53 | 3501702.674 |
| 753269.2 | 5 | 3766346.014 |
| 745961.7 | 5.51 | 4110249.148 |
| 749387.2 | 6 | 4496322.921 |
| 730186.2 | 6.57 | 4797323.423 |

Table 1 – Area\*Delay products

Fig-1 shows a graph for the values listed in Table 1.

Fig-1 – Delay vs. Area\*Delay product

* The delay as reported by Synopsys PrimeTime was **2.7 ns**
* The area reported by Synopsys DC Compiler was **845,136.55 µm2**
* The power reported by Synopsys DC Compiler was **79.4254 mW**

When optimizing the area-delay product, we were able to achieve a good reduction in this product by trying several different options within design compiler. By using the compile -map\_effort high -area\_effort high command, we noticed that design compiler is able to automatically map operators within our alu such that unused blocks within one operation are shared with another operation, thus reducing area. Using a for loop within the design compiler tcl script, we obtained the above results for optimizing our area delay product.

# STA

The critical path reported was from *node0\_d\_in[15]* to *proc\_0/WB\_alu\_result\_reg[0]*.

The histograms generated from PrimeTime are shown in Fig. 2 and 3. Fig-2 shows the details for the worst group for path slack, and Fig-3 shows the details for endpoint slack.

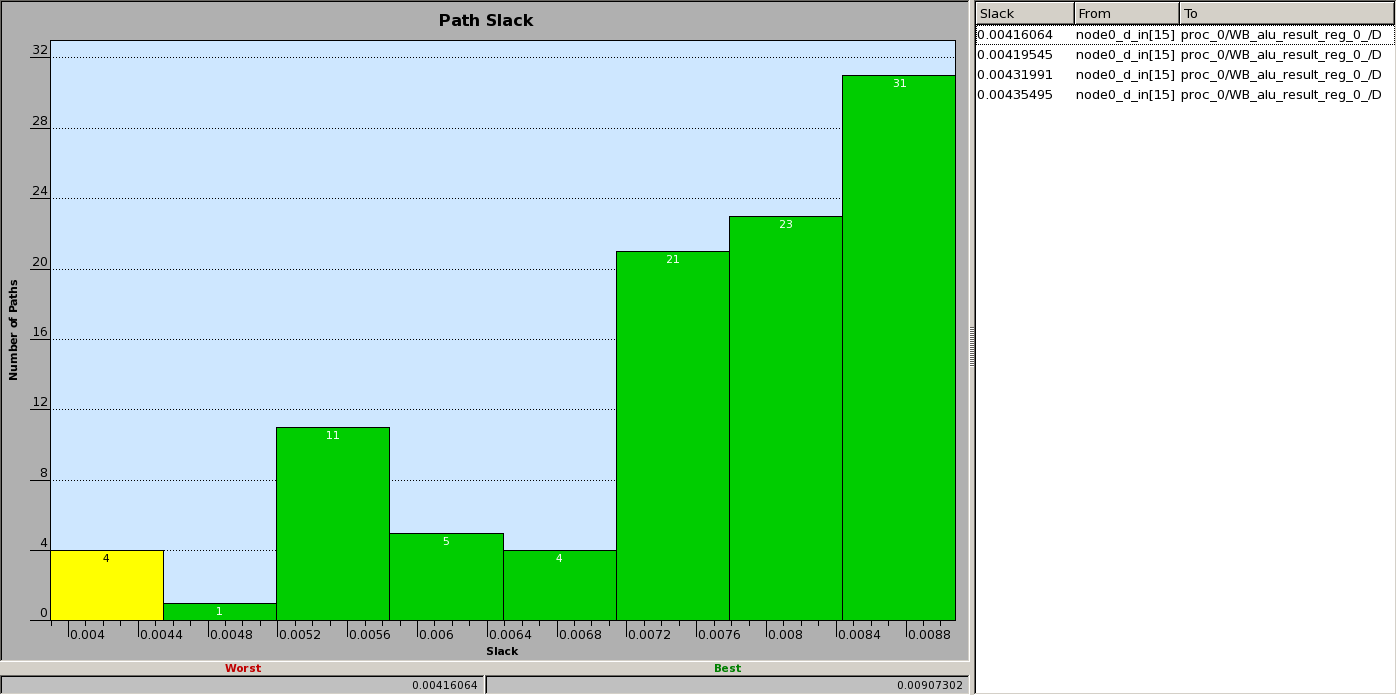


Fig-2 – Path slack histogram from PrimeTime

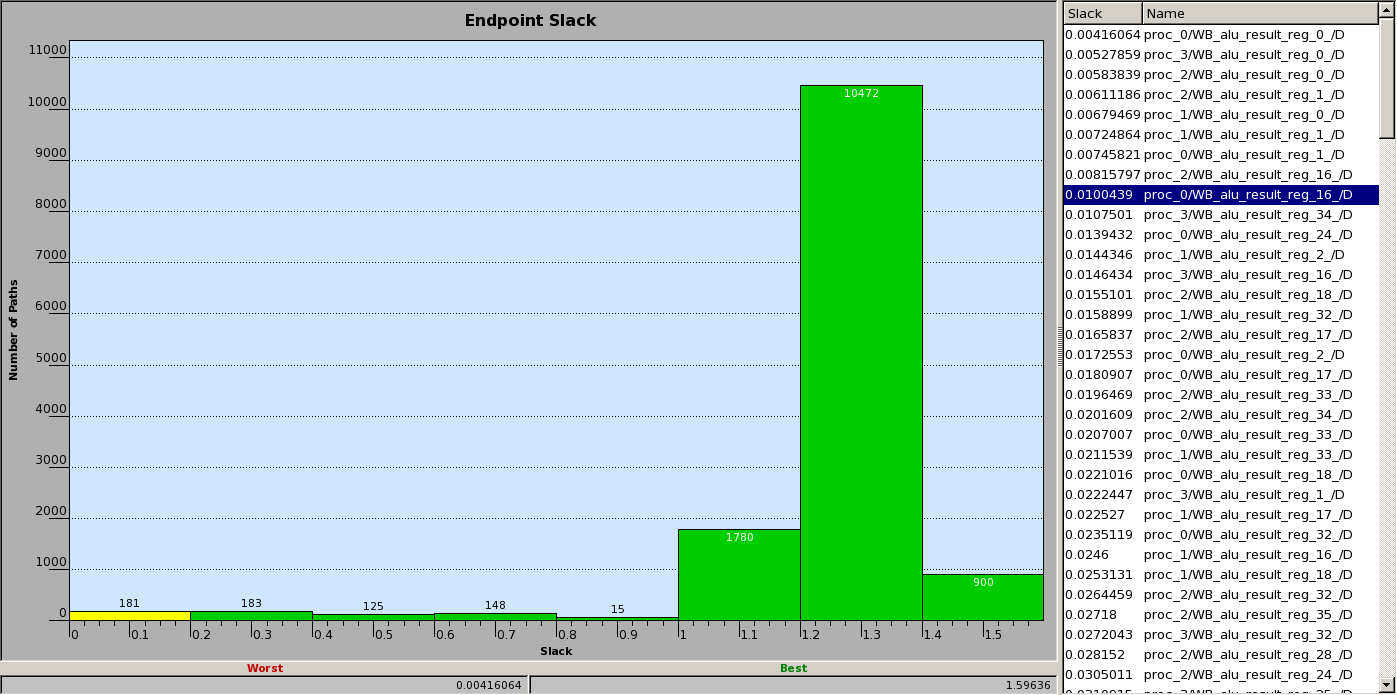


Fig-3 – Endpoint slack histogram from PrimeTime