Project Report for ECE 755: GNN Implementation

Members: Ishan Moondra & Cullen Krasselt

**Summary:**

Our project has implemented the required GNN, via a 4-stage pipeline, incorporating a balanced pipelined enhanced with advanced features such as power gating, clock gating, with a focus on high operating frequency. We are pleased to announce that our GNN achieves timing closure at a minimal clock period of just 759 picoseconds, allowing for an operating frequency well above 1.31 GHz.

Implementing on the advanced 7nm process node, the ASAP 7nm Standard Cell PDK has allowed us with standard cells that operate reliably, and with low power dissipation, even when pushed to beyond 1 GHz clock speeds. In the table below, we are presenting the key design metrics of our GNN implementation, as reported by the Synopsys Family of tools.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Area (sq mm)** | **Frequency (MHz)** | **Min Latency (ns)** | **Power (mW)** | **Energy (pJ)** | **EDAP** |
| 0.13 | 1317.5 | 5.27 | 9.201 | 48.49 | 33.22 |

**Optimizations:**

Balanced Pipeline Load: We have implemented certain architectural advancements into the GNN & the lower-level modules themselves to balance the pipeline load. The Neuron itself, which performs a simple MAC (Multiply & Accumulate Operation), has been pipelined, to help achieve lower critical path delays. Thus, the basic Neuron itself, has a multiply operation in the first cycle, followed by a pipeline flop, which allows the accumulation to be performed in the second cycle. This optimization has allowed us to achieve faster timings, without having to use bulky & power-hungry Carry Select Adders. According to our calculations, this alone allows us to save 28% power on the same clock speed. Since we have designed a very high-speed data path, this allowed us to go faster without burning significantly more energy.

Power Gating: We utilize the Input Ready signal as a convenient power gating signal for the various downstream components. Our design cleverly understands that downstream components need not be powered down, they only need to be passed an input string of 0s, which our Input Layer does for us, based on the Input Ready signal. Since this provides us with a functional power gating itself, we save power without having to invest in Multiplexors at every stage of the pipeline. The functional level power gating has allowed us power savings upwards of 15% over a naïve design.

Clock Gating: We also explored the utilization clock gating techniques in our many tool workflows. Since our design style heavily favors separating combinational & sequential blocks from each other, we are allowing the Synopsys Family of tools to be able to re-structure & better utilize the various blocks that we instantiate. As such, our clock gating potential, has yielded significant results. With clock gating we are happy to report that we save upwards of 5+ mW in power, with reductions in area & no reduction in maximum clock frequency of the system.

Asynchronous Falling Edge Reset: We recognized the need for asynchronous reset triggers for the GNN, as multi-clock domain based heterogenous systems that will utilize our GNN should be able to have a master control on the GNN sub-block itself. Thus, keeping in line with good design practices, we have added the Asynchronous Falling Edge Resets to the entire GNN sub-system. Thus, we improve the robustness of our design by ensuring proper resets for the entire system.

**Explorations:**

Understanding the Limits of the ASAP 7nm PDK: We have extensively explored the various limits of performance provided to us by the ASAP 7nm PDK, by varying the speed, area & power targets of the GNN. We have discovered through our testing, an inflection point while graphing the speeds targeted and the resulting areas. On further inspection, we realized that, beyond this sweet spot, (inflection point), the basic Ripple Carry Adders are no longer meeting timing, meaning that the tools must change to bulkier, more power-hungry CSAs, to meet stricter timings. Power dissipation, grows linearly with area, as expected. Thus, with this understanding, we can provide better performance figures for the system, as per the power budget of the ASIC.

Optimizing Critical Path Delays: We made conscious design efforts to balance our pipeline load, and we made architectural decisions that allowed us to balance out the loads, making sure the tools use downsized multipliers & adders to satisfy timing requirements. We are also conscious of the number of pipeline stages, and have such limited it to 4, allowing good throughput while also attaining high speeds.

Extensive Testing: We have performed extensive pre & post synthesis testing of the GNN block, at various clock speeds to ensure that we are functionally fully verified for potential applications in the future without needing significant re-work. We tested all styles of possible input conditions, including inputs that are out of range for the GNN block itself. We are delighted to report that the GNN passes all the tests with flying colors, hitting the required specifications effortlessly.

Good Design Practices: Keeping in mind the various good design practices of the industry, we have modularized many of the functions of the GNN, while also having parameterizable modules, allowing easy scaling of the system as needed for the specifications. Our coding style has allowed for plenty of code re-use. We also ran code coverage for the GNN, and we report a healthy 93% for the whole GNN system.

**Additional Data:**

Functional Power Gating Runs:

|  |  |  |  |
| --- | --- | --- | --- |
| **Latency (ns)** | **Speed (MHz)** | **Area (sq microns)** | **Power (mW)** |
| 676 | 1.479 | 4143 | 11.638 |
| 690 | 1.449 | 3983 | 6.622 |
| 700 | 1.429 | 3852 | 6.444 |
| 750 | 1.333 | 3639 | 5.789 |
| 800 | 1.250 | 3472 | 5.309 |
| 900 | 1.111 | 3030 | 4.488 |
| 955 | 1.047 | 2829 | 4.112 |
| 1000 | 1.000 | 2816 | 3.913 |
| 1111 | 0.900 | 2787 | 3.487 |
| 1250 | 0.800 | 2787 | 3.099 |

Clock Gating & Functional Power Gating Runs:

|  |  |  |  |
| --- | --- | --- | --- |
| **Latency (ns)** | **Speed (MHz)** | **Area (sq microns)** | **Power (mW)** |
| 690 | 1.449 | 3913 | 5.315 |
| 750 | 1.333 | 3651 | 4.670 |
| 800 | 1.250 | 3166 | 5.309 |

**Additional Observations:**

While we succeeded in getting superb results from our clock gating exploratory efforts, we were let down by the tools’ inability to handle hold times properly. With such large complexity of the tool flows in such a limited time span, we were unable to narrow down the required scripting changes needed to achieve proper timing closure with clock gating enabled. While this directly re-enforces the age-old advice of “Not touching Clock Nets”, it does point out to the massive potential our design still possess. With additional time & some more practical expertise, we are sure to iron out these issues & improve our design with a frozen RTL code as well.

While the current specifications require a fixed weight array, future work can look to make this entire block more adaptable to larger set of domains, however, the overhead of re-configurable hardware might cut into some of the savings our system will be able to provide. Downclocking to lower speeds & applying DVFS abilities can also push our designs to much wider utility & coalesce with our high speed RTL designs.