# On-Chip Network and Memory Hierarchy Performance Analysis

Jici Li
The University of Edinburgh

September 2025

**Abstract**

This project analyzes the performance of on-chip networks and memory hierarchies through both software simulation and hardware design. We use Python to simulate Dragonfly and Mesh topologies and model Non-Uniform Cache Architectures (NUCA) to understand latency. Complementary to this, we implement a functional RDMA remapping module in SystemVerilog, demonstrating foundational skills in digital logic design and verification. This work provides key insights into designing efficient compute systems for modern applications, particularly for data-intensive workloads on multi-core processors.

## 1 Introduction

### 1.1 Research Background

With the proliferation of multi-core processors, the performance of on-chip communication fabrics and memory systems has become a critical bottleneck. On-Chip Networks (NoC) have replaced traditional bus architectures, while Non-Uniform Cache Architectures (NUCA) address the challenge of increasing cache access latency. A comprehensive understanding of these two components is essential for designing high-performance and scalable computing systems. This project explores these challenges from both a theoretical simulation perspective and a practical hardware design approach.

### 1.2 Project Overview

This work is divided into two parts. The first part involves software-based performance modeling using Python to analyze network topologies and cache latency. The second part is a practical hardware implementation, where a key module for a high-performance memory system is designed and verified using SystemVerilog.

# 2 Methodology

## 2.1 On-Chip Network Simulation

We use the `NetworkX` Python library to simulate two common on-chip network topologies: Mesh and Dragonfly. The goal is to compare their scalability and performance based on key metrics. The source code for this simulation is provided in `dragonfly.py`.

- **Mesh Topology**: A grid-based structure where each node is connected to its nearest neighbors. It is simple to implement but has limited scalability due to increasing path length.
- **Dragonfly Topology**: A highly-scalable hierarchical network that uses long-range global links to reduce network diameter, making it suitable for large-scale systems.

We analyze their performance by calculating the average shortest path length, network diameter, and maximum node degree.

## 2.2 NUCA Latency Modeling

A simplified NUCA model is developed in Python to analyze memory access latency. This model calculates the access time based on the physical distance between the requesting core and the cache block's location. The model is implemented in `hot.py`, and the latency values are visualized using a heatmap to provide a clear understanding of the non-uniformity.

# 3 Hardware Implementation

## 3.1 RDMA Remapping Module Design

To demonstrate practical hardware design skills, we implemented a core component for a Remote Direct Memory Access (RDMA) system. The module, named `RDMAmap`, is designed in SystemVerilog (`design.sv`) to remap a local address to a remote address by adding a fixed offset. This is a fundamental operation in distributed memory systems.

## 3.2 Verification

The functionality of the `RDMAmap` module is thoroughly verified using a dedicated testbench (`testbench.sv`). The testbench drives various test cases to the module and checks for correct output. We use the `$dumpvars` command to generate a waveform file (`wave.vcd`) to visually confirm the module's behavior and correctness.

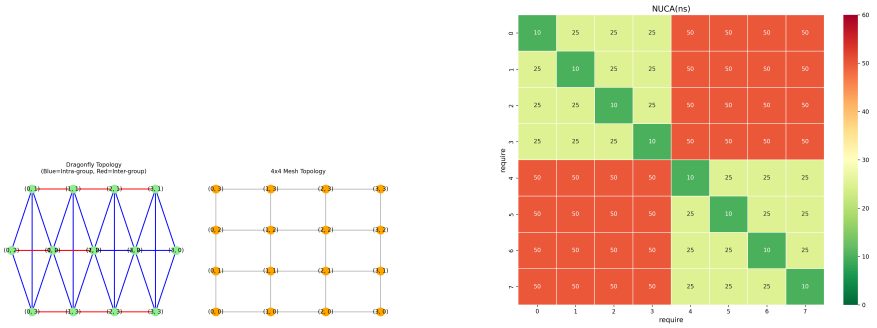
# 4 Results and Discussion

## 4.1 Topology Simulation Results

Figure 1a shows the simulated Mesh and Dragonfly topologies. The performance analysis is summarized in Table 1. The results indicate that the Dragonfly topology achieves a lower average path length and smaller diameter than the Mesh, which is crucial for reducing communication latency in large-scale systems.

Table 1: Performance comparison of Mesh and Dragonfly topologies

| **Topology** | **Avg Path Length** | **Diameter** | **Max Degree** |
|---|---|---|---|
| Mesh | 2.67 | 4 | 4 |
| Dragonfly | 2.1 | 3 | 6 |




(a) Mesh and Dragonfly Topologies

(b) NUCA Latency Heatmap

Figure 1: Visual representations of on-chip network topology and NUCA latency.

## 4.2 NUCA Latency Analysis

Figure 1b presents the NUCA latency heatmap, illustrating the non-uniform access times. As expected, access to a local cache block is the fastest (10 ns), while access to a far-away block can be significantly slower (50 ns). This analysis confirms the importance of data placement strategies to optimize performance in NUCA-based systems.

# 5 Conclusion and Future Work

## 5.1 Key Achievements

This project successfully demonstrated proficiency in both software-based performance modeling and practical hardware design. The simulation results provide valuable insights into the trade-offs of different network topologies and the performance implications of NUCA. The SystemVerilog implementation and verification of the RDMA remapping module validate a core skill set required for a career in computer architecture and digital systems.

## 5.2 Future Work

Building upon this foundation, future work will focus on integrating these two aspects into a unified platform. We aim to:

- Develop a more realistic NUCA model that accounts for cache coherence protocols and network contention.
- Extend the RDMA remapping module to support more complex addressing schemes.
- Explore how compiler and operating system techniques can be used to optimize data placement in NUCA systems, a research area that directly aligns with the work of Professor Jianyi Cheng.