# SIDDHARTH Jayashankar

PhD student, Computer Science, Carnegie Mellon University
B.Tech, Computer Science & Engineering, IIT Kanpur 2021

✉ sidjay64 [at] gmail.com
✉ sidjay [at] cmu.edu
in linkedin.com/in/sidjay10
○ github.com/sidjay10
⬀ sidjay10.github.io

## 🎓 EDUCATION

**Carnegie Mellon University**                                   *2022 - Present*
Doctor of Philosophy, Computer Science

**Indian Institute of Technology, Kanpur**                       *2017 - 2021*
Bachelor of Technology in Computer Science and Engineering  |  Grade: **9.8**/10

## 🖥 WORK EXPERIENCE

**Carnegie Mellon University**                                   *August 2022 - Present*
Doctoral Research Assistant  |  Advisors: Prof. Wenting Zheng & Prof. Dimitrios Skarlatos

▸ My research focus is on hardware and software systems for secure and private multi party computation
▸ I'm currently working on designing compiler and accelerator for Deep Fully Homomorphic Encryption (FHE) to support running large ML models like transformers in the encrypted domain
▸ In this project, I've implemented the RTL for a scalable FHE acclerator and built cycle simulation for the acclerator
▸ I've also built an optimizing compiler for FHE and am using it to implement models like Resnet, HELR and Bert in FHE

**NVIDIA - Architecture Research Group**                         *May 2024 - Aug 2024*
Doctoral Research Intern  |  Mentors: Dr. Michael Sullivan & Dr. David Nellans

▸ I worked on characterizing Fully Homomorphic Encryption on GPUs and designing potential FHE architectural extensions for NVIDIA's future GPUs

**Microsoft Research India - Systems Research Group**            *July 2021 - July 2022*
Research Fellow  |  Mentors: Dr. Kapil Vaswani & Dr. Akash Lal

▸ I designed the attested TLS protocol that seamlessly integrates for establishing secure communication channels between services in Trusted Execution Environments
▸ I designed an implementation of the attested TLS protocol for easy deployment using network proxies and service meshes for azure confidential containers.

**IIT Kanpur -**

**Computer Architecture for Reliable, Secure, and Scalable Systems (CAR3S) Research Group**   *Oct 2020 - May 2021*
Undergraduate Researcher  |  Advisor: Prof. Biswabandan Panda

▸ I Ported Valgrind to Android and built tracing tool- cstracer to collect program execution traces of android apks and collected instruction and memory acess traces of mobile benchmarks - geekbench5 and Antutu
▸ I measured the performance difference between the big and little cores in ARM's big.LITTLE architecture and investigated the potential for cache block compression on a custom modification of the champSim simulator using the traces collected
▸ ○ **cstracer**  |  **Presentation**

**EPFL - Parallel Systems Architecture Lab (PARSA)**             *March 2020 - Aug 2020*
External Student Researcher  |  Mentor: Dr. Yunho Oh

▸ I studied the performance of container-level horizontal and vertical scaling of Cloudsuite benchmark workloads on server CPUs.
▸ I analysed performance bottlenecks caused by front end pipeline stalls in server CPUs, lock contention and thread scheduling overheads.
▸ I performed experiments to measure the speedup caused by scale-out and scale-up workloads on CloudSuite and identify configurations for optimal resource utilisation.

**Intel - Processor Architecture Research Lab (PARL)**           *Dec 2019 - Feb 2020*
Architecture Research Intern  |  Mentors: Anant V Nori, Sreenivas Subramoney

- I worked on designing a new prefetcher to mitigate the performance reduction caused by downsizing the L2 cache to extend the CATCH Microarchitecture proposed by Nori et al [ISCA'18]
- I first identified the causes for the drop in performance on benchmark workloads and used the insights to develop a new prefetcher.
- My prefetcher was able to significantly offset the performance drop caused by downsizing the L2 cache on several benchmark workloads.

## IISc Bangalore - Algorithms, Complexity and Optimisations Group  *May 2019 - July 2019*
Summer Research Intern | Advisor: Prof. Arindam Khan

- I worked on devising a polynomial time approximation algorithm for 2D strip packing to improve the absolute approximation ratio of Steinberg's algorithm

## 📰 PUBLICATIONS

### Cinnamon: A Scale Out Framework for Encrypted AI  *ASPLOS 2025*
- Siddharth Jayashankar, Edward Chen, Tom Tang, Wenting Zheng, Dimitrios Skarlatos. Proceedings of the 30th Intl. Conference on Architectural Support for Programming Languages and Operating Systems, Rotterdam, The Netherlands, March 2025

## ★ AWARDS & ACHIEVEMENTS

- Academic Excellence Award, IIT Kanpur for the years 2020, 2019, 2018 and 2017
- A* grade for outstanding performance in 8 courses at IIT Kanpur
- Perfect GPA (10/10) in five semesters at IIT Kanpur
- Selected for the uArch Workshop at MICRO 2020
- Ranked 1348 in JEE(Advanced), 2017 and 2481 in JEE(Main), 2017
- Selected for the KVPY Fellowship Award 2016 by IISc Bangalore

## ⚙ PROJECTS

### Empirical Evaluation of State-of-the-art cache replacement policies  *Nov 2021*
- A Study to compare the performance of state-of-the-art LLC replacement policies like SHiP, DIP, Hawk-Eye on SPEC CPU benchmarks using the ChampSim simulator | ⧉ **Project Report**

### Cache Simulator to study effects of architecture and replacement policies  *Aug 2019*
- Built a cache simulator to study the effect of replacment policy, associativity and cache type (Inclusive, Exclusive, NINE) on hit and miss rates | ⧉ **Repository** | ⧉ **Project Report**

### C to MIPS32 compiler  *May 2021*
- A C source to MIPS32 target compiler written in **C++** | ⧉ **Source Code**
- Compiler supports language features like **variables, control statements, recursion, custom data types, multi level pointers, multi dimensional arrays**. It can perform code optimisations like **constant folding** and **basic dead code elimination** and also provide detailed error reports.

### Cache miss analysis on loop nests  *Sep 2020*
- A program that takes as input loop nests and cache sizes, parses the loop nest and then reports the number of cache misses for the loop ordering | ⧉ **Source Code**

### Mozart Oz Interpreter  *Oct 2020*
- A minimal interpreter for the Oz programming language that implements the kernel language | ⧉ Source Code

### Programming Operating System syscall and exception handlers  *Oct 2019*
- Implemented syscalls like **mmap, munmap, mprotect, page fault exception handling, fork, copy on write fork, file and pipe reads, writes, open, close** on gemOS - a teaching OS | ⧉ Source Code 1, ⧉ Source Code 2

### FPGA synthesis of a basic processor  *Apr 2019*
- Synthesised a simple processor that can perform instructions like add, sub, compare, branch and load on an FPGA. This processor can execute simple programs like loops and report results. | ⧉ Source Code

## 📘 COURSEWORK

| | | |
|---|---|---|
| Parallel Computer Architecture (A) | Operating Systems (A) | Compiler Design (A) |
| Advanced Computer Architecture[†] (6) | Software Security[†] (5) | Advanced Algorithms (A) |
| Machine Learning (A*) | Paralellism & Concurrency[†] (5.25) | Theory of Computation (A) |
| Formal Logic (A) | Programming for Performance (A) | Data Structures and Algorithms (A) |
| Database Systems[†] (6) | Learning Theory[†] (5.75) | Discrete Mathematics (A) |
| Principles of Programming Languages (B) | Computer Networks (A) | Multivariable Calculus (A) |
| Linear Algebra (A*) | Introductory Economics (A*) | Macroeconomics (A) |

† Course done at EPFL

## ☰ SKILLS

| | |
|---|---|
| **PROGRAMMING & SCRIPTING** | C++ \| C \| Systems Verilog \| Verilog \| C \| Python \| CUDA \| Rust \| Bash |
| **ASSEMBLY** | x86-64 \| MIPS \| aarch64 |
| **OPERATING SYSTEM** | Linux \| Android \| Windows |
| **CONTAINERS** | Docker \| Kubernetes \| runc \| Microsoft hcsshim |
| **SOFTWARE** | Envoy Proxy \| Istio \| Valgrind \| OpenSSL |
| **MISC** | Latex \| Git \| Markdown \| Jekyll |