Sne Samal

London, UK | +44 7398892448 | sne.samal@gmail.com

EDUCATION

Imperial College London **London, UK**

Electronic and Information Engineering *Graduation Date: Jul 2026*

- Year 1 Mark: 76.12 (1st Class). Dean's List (top 10% of cohort)
- Year 2 Mark: 72.44 (1st Class).
- Year 1 Modules of Interest: Maths, Digital Electronics & Computer Architecture, Programming in C++
- Year 2 Modules of Interest: Instruction Architecture & Compilers, Software Systems, Discrete Maths, Information Processing
- Societies: Data Science, Electrical Engineering, Department of Computing.

WORK EXPERIENCE

Imperial College London **London, UK**

Undergraduate Tutor *Oct 2023 – present*

- Tutoring digital electronics and computer architecture to first year electrical and electronic undergraduates
- Arranged hourly 1:1 sessions, going through course content and ensuring that students leave with a stronger understanding.

Embedd.it **London, UK**

Electronics Engineer Intern *Jul 2023 – Sep 2023*

- Led analysis of electronic component databases to identify the most popular digital interfaces and vendors.
- Automated the creation of JSON files, later read by Large Language Models, from pdf datasheets using Python and Pandas, streamlining the documentation process and reducing manual effort.
- Analysed datasheets analysis to extract critical information, enabling faster decision-making in the development process.

PROJECT EXPERIENCE

FPGA HFT Accelerator *Jul 2024 – Aug 2024*

Technologies Used: System Verilog, C++, Python, Vivado, Git, Cocotb, Verilator, Icarus Verilog

- Implemented the Avellaneda-Stoikov market making strategy on a PYNQ Z1 FPGA in System Verilog. Wrote code for trading logic and a functional order book capable of processing input ITCH data for 4 stocks.
- Wrote a market simulation in Python to randomly execute, send and receive orders to and from the FPGA via TCP.
- Comprehensively simulated and tested each sub module using Cocotb testbenches and C++ testbenches using Verilator.

FPGA Voxel Integer Ray Tracer

Technologies Used: C++, System Verilog, Python, Shell Scripting, Git, Verilator, Vivado *May 2024 – Jun 2024*

- Developed an integer-based voxel ray tracer that was accelerated via hardware, achieving speeds 1000 times faster than Python and 116 times faster than C++ software equivalents
- Wrote System Verilog logic to perform ray tracing, including the generation of rays from a given camera position and direction and wrote logic for rays to traverse a scene and return a correct pixel colour depending on whether an object was hit.
- Physically implemented the logic on a PYNQ Z1 FPGA, used MMIO via Python to pass in parameters like camera position.

C90 to RISC-V Compiler

Technologies Used: C++, C, RISC-V Assembly, Make, Bash, Git, GitHub *Feb 2024 – Mar 2024*

- Developed a functional C90 to RISC-V assembly, in C++ with support for integers, floats, pointers, arrays, chars and structs.
- Added support for control-flow including if-else, for and while, enums, nested and recursive functions and the built-in sizeof.
- Made use of Make for build automation and created Bash scripts to automate testing and validation of outputs.

DotsApp: Real Time FPGA Morse Code Chatting App

Technologies Used: C, Python, Verilog, Quartus, AWS, EC2, DynamoDB. Git, GitHub *Mar 2024*

- Produced a real time chatting application where users can use buttons, switches and tilt gestures on board a DE10-Lite FPGA to communicate via Morse code to different chat rooms.
- FPGA part built on Quartus, using Verilog and C to process outputs from the board and NIOS II processor.
- Communications between users processed using a Tkinter client and a python server running on an EC2 instance on AWS, with message history being stored on DynamoDB.

RISC-V CPU

Technologies Used: C++, System Verilog, Python, Verilator, WLS, Git, GitHub *Oct 2023 – Dec 2023*

- Programmed a functional multi-stage pipe-lined processor in a team of 4, using System Verilog, Verilator, WSL and C++, compliant with the RISC-V instruction set specification (RV32I 2.0).
- Involved in all aspects of implementation, and was responsible for the majority of features, specifically CPU design, C++ testbenching and debugging, workflow automation and documentation through a team GitHub repository.
- Implemented functional and verified pipelining and data cache within the processor for improved performance.

SKILLS & INTERESTS

Skills: C/C++, Python, Assembly, System Verilog, JavaScript, Git, GitHub, SQL, Markdown, LaTeX, Arduino, Django, Flutter

Interests: Computer Architecture, Micro-Architecture, FPGAs, Digital Logic Design, Machine Learning, Digital Electronics, Embedded Systems, Compiler Design.