



PULKIT AGRAWAL

+1-858-943-8811 | p3agrawal@ucsd.edu
in/pulkitag22 |  /pulkitag22 |  /pulkitag22.github.io

OBJECTIVE

Seeking an internship position to explore career options in Computer Architecture, RTL Design and VLSI domain.

EDUCATION

University of California, San Diego, CA

MS in Electrical & Computer Engineering (Major: Computer Engineering)

Class of 2023

Sept 2021 – Present

Coursework: Low-power VLSI Implementation for Machine Learning, Principles in Computer Architecture, VLSI Digital System Algorithms and Architecture.

Birla Institute of Technology and Sciences (BITS) Pilani, Goa Campus, India

BE (Hons.) in Electronics & Instrumentation Engineering

CGPA: 8.39/10

Aug 2015 – May 2019



Coursework: Computer Architecture, Digital Design, Microprocessors and Interfacing, Operating Systems, Real-Time Systems, Data Structures and Algorithms, Network Embedded Applications, C Programming.

EXPERIENCE

Research Assistant (Remote) - Prof. Akash Kumar |

Oct 2020 – June 2021

CFAED, Technische Universität Dresden, Germany 

- Worked on Logic Synthesis and Boolean Network classification. Used ABC Logic Synthesis tool  and MockTurtle (logic network representation and manipulation library) .
- Created libraries for evaluating self-duality of a boolean function and added functional support to return the Self-dual equivalent class for n-input boolean functions.

Firmware Engineer

Sept 2020 – July 2021


Amplify Mobility Pvt. Ltd. 

Hyderabad, India


- Developed **C/C++** based firmware to implement application protocol, **Open Charge Point Protocol (OCPP)** for 4 different AC & DC Electric Vehicle Charging Stations product lines.
- The code base developed is catering to 1000+ EV Charging station across the country India.

ASIC Design Intern

July 2018 – Dec 2018

Nvidia Hardware 

Bangalore, India

- Worked with the **SOC Clocks Team** worked on generating Register-transfer-level (RTL) for Tegra SOC  - 'Orin' using a novel automated framework.
- Used **Synopsys SpyGlass** to generate lint reports to categorize about 35,000 errors. Wrote **Python/Perl** based scripts to automate the rectification process.
- Worked on porting of existing verification test plans (of the old clocking scheme) for simulation of new Clocks RTL.

TECHNICAL SKILLS

Languages: Python, Perl, C/C++, Embedded C, Verilog, System Verilog, MASM (Assembly Language)

Developer Tools: Git, PyCharm, Arduino IDE, CubeMX (STM32) IDE, Keil μ Vision, Eclipse, ModelSim Altera

PROJECTS

RISC Processor Synthesis | Computer Architecture, Verilog

March 2018 – April 2018

- Implemented 32-bit **MIPS Architecture** with a simple Arithmetic Logic Unit (ALU) and Control Unit in **Verilog**.
- Designed a Hazard Unit to take care of any potential hazards and wrote test benches to test the pipeline architecture.

Approximate Adder Circuits | Cadence RTL Compiler, Verilog |

Jan 2018 – April 2018

- Implemented various 16-bit single-level and multi-level approximation adder architectures in **Verilog**.
- The designs were synthesized using **Cadence RTL Compiler** and the comparison with 16-bit exact Ripple-Carry adder showed the approximate architectures were **46% faster** and saving upto **30% in area**, and **45% in power**.

Multiplications of 'N' complex numbers | Multi-threaded Programming, C |

Oct 2019 – Nov 2019

- Implemented a multi-threaded **C** program using **POSIX Threads** library, for multiplying 'N' complex numbers.
- Created pairs of 'N' complex numbers and performed concurrent multiplication using threads. Stored the result of each pair using dynamic memory allocation.
- Values from the multiplication of each pair were then used to create new pairs until the result was obtained.

Scheduling Algorithms for Real-Time System | ADA Language, Cheddar Simulator |

Aug 2017 – Sept 2017

- Implemented user-defined scheduling algorithms for real-time systems in **ADA Language** used by **Cheddar Simulator**.
- Designed a hybrid algorithm, dynamic-priority scheduling class, to give tasks with zero laxity the highest priority.