1401 Collegiate Circle, Apt 301
Raleigh, NC 27606
(469)-456-4858

Sounder Rajendran

srajend4@ncsu.edu
https://sounder.world
linkedin.com/in/sounder-dev/

Education

North Carolina State University **Raleigh, NC** **Aug 2022 – Dec 2023 (Expected)**

- M.S. in Computer Engineering; GPA: 4.00

Coimbatore Institute of Technology **Coimbatore, India** **Aug 2015 – May 2019**

- Bachelor of Engineering in Electrical and Electronics Engineering, First Class Graduate

Relevant Coursework

- Advanced Microarchitecture
- Data Structures and Algorithms
- ASIC and FPGPA Design With Verilog
- ASIC Verification
- Architecture of Parallel Computers
- Embedded System Design

Languages and Technologies

- **Languages:** Python, C, C++, JS, Dart, SQL, LabVIEW, Verilog, System Verilog, Unix scripting
- **Tools:** Synopsys, Multisim, Cadence Virtuoso, Xilinx Vivado, MATLAB
- **Technologies/Frameworks:** Git, JIRA, Confluence, Rational Jazz, AUTOSAR, Azure IoT, ROS2, MIPS, MPI

Employment

Senior Embedded HMI Engineer **Bosch Global Software Technologies** **Jan 2022 - June 2022**

- Involved in the design and development of device drivers and SDK toolkit for low-cost, low-power HMI Embedded Systems based on Xtensa dual-core 32-bit LX6 microprocessors.
- Involved in researching and developing a generic automated IoT test bed integrated with robotic test framework of cloud-connected HMI systems. This resulted in the growth of the automated testing business by 15%.
- Initiated research and development of AI security threat mitigation layer in Bosch IoT gateways to detect and kickoff connected nodes with mismatching digital device signatures.

Embedded HMI Software Engineer **Bosch Global Software Technologies** **Sept 2019 - Dec 2021**

- Involved in the design and programming of sensor fusion to extract enhanced positional data for use in tri-axial ultra-low power acceleration sensors (BMA series).
- Proposed, designed, and developed a new generation HMI test automation framework - Closed Loop Testing of Voice and Extended Reality Systems (python and LabVIEW). Resulted in the expansion of HMI business scope by 20%.
- Performed testing and debugging of Embedded HMI hardware on Automated Test Equipment (ATE) and Hardware-in-the-loop (HIL) systems utilizing tools including Digital Storage Oscilloscope, current transformers, and hall-effect sensors.

Notable Industry Projects

- **Plug and Play device driver development:** Programmed and tested python and C++ based drivers for USB, ethernet, RS232 and wifi module connections for SIMANTIC IoT2020 gateway for Siemens Industrial Automation
- **Interactive Humanoid Robot Swarm:** Designed, built and developed a swarm of robots that upon reciving audio/visual stimulus, follows and interacts with humans. Used ROS2, python, C++, Embedded C, CV2 and tensorflow

Notable Academic Projects

- **Neural Network Accelerator using Verilog:** Designed and developed a multi-stage neural network, including a convolution layer, a fully connected layer, and a max pooling layer with ReLu activation functions with functional parallelism in Verilog for ECE 564 course under Dr. Paul Franzon
- **Cache Simulation and Memory Hierarchy Design:** Designed and developed a flexible cache and memory hierarchy simulator to compare the performance, area, and energy of different memory hierarchy configurations testing against microbenchmarks; WBWA & WTWNA policies are used along with LRU replacement policy to update data into cache.
- **Multi-processor system with bus-based coherence:** Designed and developed a trace-based simulator for comparing MSI, MESI, MOESI based cache coherence protocols in multicore systems with generic memory hierarchies in C++.
- **Branch predictor Simulator:** Developed branch prediction simulator with C++ that supports different configurations like GShare, BiModal, and Hybrid with an option to add Branch Target Buffer, for ECE 563 course under Dr. Eric Rotenberg.
- **16-bit RISC Processor using Verilog:** Curently involved in the development of a simple 16-bit RISC processor (Harvard architecture) using Verilog. It uses a 24-bit instruction set which can be decoded into 16 different instructions including arithmetic (ADD, SUB, MUL), logical (AND, OR, XOR, NOT), data transfer (LOAD, STORE), and branching (JMP) operations.