RUDRENDU MAHINDAR

rudrendumahindar@gmail.com | https://www.linkedin.com/in/rudrendu-mahindar/ | https://github.com/RudrenduMahindar | 323-413-7376 | Los Angeles, CA 90007

EDUCATION

Master of Science in Electrical Engineering **Aug 2019-May 2021**
University of Southern California, Los Angeles, CA **GPA: 3.65/4**
Bachelor of Technology in Electronics & Communication Engineering **Jul 2014-Jul 2018**
Maulana Abul Kalam Azad University of Technology, India **GPA: 8.77/10**
Relevant coursework: Operating Systems, Computer Architecture, Parallel and Distributed Computing, Computing and Software for Systems Engineers

SKILLS

Programming languages: C, C++, Python, Verilog
Software tools: Bazel, Visual Studio, Gdb, Git, Atmel AVR Studio, Proteus, QuestaSim, Arduino IDE, MATLAB, Xilinx Vivado
Development skills: Kernel programming, data structures, bare-metal firmware, FreeRTOS, CUDA, MPI, Google Test

PROFESSIONAL EXPERIENCE

Rhoman Aerospace - Embedded Software Intern **Jun 2020–Jul 2020**

- Integrated scalable firmware application in C++ with PX4 middleware layer for adding new sensor data to flight telemetry logs
- Leveraged background of Amazon Web Services (AWS) to customize data storage of flight telemetry logs from Pixhawk
- Automated telemetry data flow in python to transfer data from ground control station application to storage system

Electron - Assistant Embedded Firmware Developer **Feb 2019–Jun 2019**

Temperature measurement and display system for thermocouples (types- J, K, R, S)

- Increased functionality of system with flexible hardware and calibrated firmware in C to support different thermocouples
- Simulated hardware in Proteus to integrate thermocouple, instrumentation amplifier, ADC, with AVR microcontroller
- Updated firmware in C, coded SPI driver in AVR MCU to read ADC module for measuring amplified voltage

Automatic detection and indication of wrong/correct sequence of wires in 3-phase (R-Y-B) alternating current source

- Designed low-voltage circuit to interface 3-phase supply with AVR microcontroller and tested simulated design in Proteus
- Introduced intelligent mechanism by coding firmware in C for AVR MCU to direct input to load for correct sequence

PROJECTS

Implementation of exokernel style operating system- JOS for x86 architecture emulated by QEMU

- Wrote memory management code that included physical memory allocator, and set up MMU's page tables for virtual memory
- Enhanced kernel with data structures for running user environments, and set up IDT with handlers of interrupts/exceptions
- Implemented preemptive multitasking with round-robin scheduling, added IPC to support communication of user-environments

Implementation of lightweight monolithic UNIX-based operating system- Weenix for x86 architecture emulated by QEMU

- Built basic blocks of OS– processes, threads, synchronization primitives (mutex), scheduler based on FIFO
- Interfaced VFS using polymorphism for underlying file systems (S5FS, RAMFS), file and directory lookup implementation
- Implemented abstraction for user-space mapping onto physical memory using virtual memory with different memory objects

IoT-based Home Appliances Control System- ARM Cortex M4-based STM32 MCU, NodeMCU, and Blynk Server

- Programmed firmware in C on ARM Cortex-based STM32 to read SPI, I2C, ADC for sensors, send UART data to NodeMCU
- Coded GPIO driver on STM32 microcontroller board to signal actuators, UART driver to read data over serial terminal

Token Bucket Emulation- multithreading in C using POSIX Threads

- Implemented token generator, token bucket, packet generator, packet queues, and servers to emulate traffic shaper in UNIX OS
- Prevented race conditions in multithreading for shared resources and busy-waiting by using mutex and conditional variable
- Synchronized signal-handling in emulated multithreaded system to gracefully terminate program with printed emulation trace

Brightness-controlled LED dimmer in FreeRTOS on ARM cortex-based STM32 MCU

- Created tasks in Real-time Operating System to read brightness on LDR from ADC, and control intensity of LED through PWM
- Communicated between tasks in RTOS through notifications to save CPU cycles and update PWM value

High-performance computing: K-means clustering in C++

- Parallelized K-means clustering by using C++ multithreading, handled synchronization using mutex, and condition variable
- Loaded image as input data of 800x800 matrix, clustered input matrix through concurrent programming into K-clusters

5-stage pipelined CPU in Verilog and simulation on QuestaSim

- Designed CPU to perform operations like ADD1, ADD4, SUB3, and MOV on 16-bit data in Verilog (RTL) on QuestaSim
- Divided execution stage into 2 parts- subtractor stage (subtracted 3 from operand) followed by adder stage (added 4 to operand)
- Incorporated logic to generate Stall signal in pipeline, reduced stalling instances through forwarding MUX and IFRF

Matrix multiplication on GPU- C, CUDA

- Applied CUDA programming model to multiply two 1024 x 1024 matrices on GPU by using 2-D grid and blocks of threads
- Leveraged use of shared memory in block matrix multiplication to increase performance by 70%

LEADERSHIP & INVOLVEMENT

- Associate Chair at Graduate Committee in IEEE branch of University of Southern California