Isaac Adam Tourner

(248) 938-2918 | itourner@umich.edu | linkedin.com/in/itourner | github.com/itourner

EDUCATION

University of Michigan, Ann Arbor, MI

M.S.E., Electrical and Computer Engineering (GPA: 3.78/4.0) *Aug. 2024 - Dec. 2025*

B.S.E., Biomedical Engineering; minor: Electrical Engineering (GPA: 3.59/4.0) *Aug. 2020 - May. 2024*

Relevant Coursework: Embedded Control Systems; Signals & Systems; Probability & Random Processes; Matrix Methods for Signal Processing & ML; Circuits & Systems for Medical Instrumentation; ML for Embedded Systems (Mobile IoT)

TECHNICAL SKILLS

Languages: C++, C, Python (Pytorch), MATLAB/Simulink; C#/.NET, CUDA (in progress)

Real-time and concurrency: FreeRTOS (tasks, vTaskDelayUntil), semaphores/mutexes, ISR-to-task handoff, interrupt-driven software, multi-threading, synchronization, deadlock avoidance

Interfaces: SPI, I2C, CAN, GPIO, ADC; register-level bring-up and debug

Tools: Git/GitHub, VS Code, gdb, Make/CMake; Linux; oscilloscope, logic analyzer, DMM; KiCad, Altium, LTspice

RESEARCH AND PROFESSIONAL EXPERIENCE

Neurobionics Lab - Embedded System for Ankle-Foot Orthosis | *University of Michigan* *Aug. 2025 - Dec. 2025*

- Led design of a size-constrained embedded PCB; validated components through schematic/PCB review and bench testing.
- Wrote and validated sensor and ADC drivers, plus a bring-up/regression framework emphasizing interrupt-driven acquisition and debuggability.

Willsey Lab - Real-Time Finger BCI Simulator | *University of Michigan* *Aug. 2025 - Dec. 2025*

- Built a closed-loop real-time simulator with a focus on timing, reliability, and logging.
- Trained and evaluated movement prediction models and demonstrated end-to-end closed-loop decoding.
- Implemented a multi-threaded data pipeline; created lightweight real-time visualization to monitor telemetry during experiments.

U.S. Army DEVCOM Ground Vehicle Systems Center | *Warren, MI* *May 2023 - Aug. 2023*

Electrical Engineering Intern, Energy Storage Team

- Executed Li-ion simulation tests and analysis to improve cycle life by 54% under heavy loads; produced repeatable test logs.
- Validated BMS behavior using wireless CAN tooling and investigated anomalies by correlating logs, sensor data, and test conditions.

PROJECTS

FreeRTOS Real-Time Robot Scheduling + Deferred Interrupts | *Arduino, C/C++*

- Built periodic tasks with RM-style priority assignment using xTaskCreate() and vTaskDelayUntil(), then instrumented releases/execution via GPIO and verified timing/jitter on a logic analyzer across 500+ samples (mean/min/max/std).
- Implemented minimal-work ISR (task handoff using a binary semaphore with xSemaphoreGiveFromISR), keeping higher-priority deadlines intact while moving heavy work into a worker task (classic real-time best practice).

ADS114S08 SPI ADC Driver + Data Logger | *RaspberryPi CM5, C/Python, SPI/GPIO, interrupts*

- Developed a register-level driver for the ADS114S08 ADC (using SPI) with DRDY-based acquisition, configurable single-shot sampling, and optional STATUS/CRC checks to improve data integrity and bring-up reliability during bench debugging.
- Built a reusable data-logging pipeline (Python CLI + CSV output) that converts raw codes to calibrated voltages and sensor units, enabling repeatable bench tests and faster validation of divider and Hall sensor measurements.

Embedded Linux Device Drivers + Vision-Guided Motor Control | *RaspberryPi, C/C++*

- Built and validated LKMs with Kbuild, used insmod/rmmod/lsmod, and debugged with printk and dmesg; implemented a character device with /dev node creation (major/minor via mknod), exercised via open/read/write, and extended functionality with ioctl (swap L/R, invert polarity).
- Combined a GPIO-controlled H-bridge driver (sysfs + driver interface) with OpenCV tennis-ball tracking and converted ball center coordinates into serial motor commands for closed-loop steering; validated interrupts and system behavior using /proc/interrupts and timing measurements on a logic analyzer.

SSVEP EEG Controller Pipeline | *STM32, Embedded C/C++, Python, SPI/I2C*

- Performed microcontroller hardware bring-up, interface debugging, and end-to-end data capture tooling with emphasis on repeatability and robust integration.
- Architected an EEG-driven control pipeline: data acquisition to signal processing/classification to command output; emphasized latency and repeatability.

Adaptive Cruise Control and Steering | *NXP S32K, MATLAB/Simulink, Embedded C, CAN*

- Modeled vehicle dynamics in MATLAB/Simulink and implemented an Adaptive Cruise Control (ACC) controller in C, including position and velocity control with CAN-based communication.

- Built and integrated NXP-based vehicle subsystems using GPIO, PWM, QD, and ADC to support potentiometer/dipswitch inputs and a haptic wheel; implemented automatic steering using Pick Lead Logic for lead-vehicle following in platoon scenarios.