Braden Everson

bradeneverson@gmail.com | (608) 628-0067
linkedin.com/in/braden-everson | github.com/BradenEverson | bradeneverson.github.io

SUMMARY

Computer Science Major, Computer Engineering Minor student at MSOE with hands-on experience in firmware engineering, systems level programming, and hobbyist language design. Strong background in Rust, C, and Zig with a focus on solving challenging problems in the world of lower-level programming. Passionate about roles involving embedded systems, operating systems and mission critical applications in unique areas of focus. Seeking opportunities that align with career goals of contributing to impactful solutions in unique fields while always continuing to learn.

EDUCATION

B.S. Computer Science – Computer Engineering Minor | Milwaukee School of Engineering | GPA: 3.93 | Exp. 06/2027

INTERNSHIP EXPERIENCE

Milwaukee Tool - Firmware Engineering Intern **May 2025 - August 2025**

- Worked on the Platform Electronics - Tool Intelligence FW team.
- Investigated BLE features including Periodic Advertising and Broadcast Isochronous Streams and their uses in Tool BLE Firmware.
- Proposed and Designed systems for complex Firmware BLE systems and topologies.
- Implemented systems on STM32 target boards using IAR embedded workbench.

Cognex Corporation – Software Engineering Intern **May 2024 - May 2025**

- Researched and integrating Rust into a large CMake-based C++/C codebase, replacing an in-house HTTP server implementation responsible for multiple user-facing crashes due to memory related issues.
- Utilized the Corrosion crate and cxx for seamless C++ and Rust interoperability, ensuring smooth integration into the existing codebase.
- Leveraging Rust's memory safety to architect a robust and crash-resistant HTTP server, improving system stability while maintaining performance deltas.
- Implemented WebSocket upgrade protocols in Rust, enabling more modern features within the server.
- Utilized asynchronous Rust for thread-safe, optimized server management, ensuring efficient handling of multiple connections.
- Developed and executed unit and integration tests in both Rust and C++ to validate compatibility with existing FFI consumers.
- Reported daily findings and progress to scrum team of engineers.

FitX On Demand – Full Stack .NET Developer Intern **January 2023 – August 2023**

- Developed cloud based IoT solution for remote access QR codes that can scan into fitness content.
- Implemented secure zone identifier that bans video access if scan location is out of range from the designated hotel/gym.
- Leveraged AWS s3 buckets to host video content securely with limited endpoints open.
- Developed data visualization through QR code scan location statistics on a map along with dynamic QR code disabling based on how much variety these scan locations carried.

BrightBean Labs – Full Stack .NET Contract Developer **November 2020 – January 2023**

- Developed Asp.Net web applications and Xamarin Forms mobile applications for clients seeking unique software solutions in the .Net space.
- Developed Denver Health DHREM scheduling platform: an Asp.Net built application providing continuously integrated scheduling services in real time for Denver Health resident interns.
- Used Xamarin forms and MVC architecture to develop an IoT mobile solution for smart truck engine warmers that could be dynamically controlled through the app.

PERSONAL PROJECT EXPERIENCE

RTOS Preemptive and Cooperative Scheduler From Scratch | Independent **July 2025 – Present**

- Implemented a basic scheduler that can register tasks as concurrent processes, providing a yield API for cooperatively or preemptively causing a context switch
- Created a modular backend system that allows different context switching assembly logic for different CPU architectures that is determined at compile time
- Created a compile time scheduler algorithm that can be swapped out to follow different schedulers, including Round Robin or a random task chooser.
- Made a demo Stm32 Nucleo project that uses the scheduler to concurrently run 2 tasks

GUNTHER - Autonomous Nerf Turret with Real Time Tracking | Team of 2 **November 2024 – Present**

- Developed the control systems software for GUNTHER, a 3-foot autonomous Nerf Turret.
- Designed a modular binary communication protocol for specifying up/down and left/right movement as well as starting and stopping shooting.
- Implemented GUNTHER across two processes connected by a UNIX pipe: the commander and the controller. The controller implemented the communication protocol and controlled peripherals as such. The commander was responsible for administering these commands.
- Created an OpenCV and ONNX Runtime process for detecting poses using the YoloV8 Pose model. Calculated pose position in the screen and sent varying movement commands to center the pose based on magnitude of distance from center.
- Implemented multi-person tracking where the system wouldn't hang or jump between people when multiple poses are found.Check out GUNTHER in our video

MaoLang – Interpreted Language with Randomized Syntax Rules | Independent **March 2025 – May 2025**

- Inspired by the card game, created an interpreted language with dynamic syntax rules that change based on the Sha256 hash of the file being interpreted (as an April fool's joke).
- Implemented if/else conditional branching, printing, variable assignment and reassignment, while and for style loops and primitive types including booleans, floats and strings.
- Created dynamic rules that affect if parenthesis are required and what is necessary to end a statement (;, 'done', or '.').
- Managed to actually generate a few working Mao scripts.

Real Time Multiplayer Web Game from Scratch | Independent **July 2024 – August 2024**

- Developed a real-time, WebSocket-enabled tower defense game with a focus on multiplayer interactions.
- Utilized Rust and TypeScript to provide a robust backend server and a responsive gameplay experience on the frontend.
- Utilized Asynchronous Rust runtimes in a performant and optimized manner to ensure scalable connection handling.
- Implemented a WebSocket server in Rust to handle real-time game state synchronization across multiple clients.Created a protocol built from WebSockets to convey important game information using Rust's algebraic types that could map via serialized binary to TypeScript objects.
- Utilized TypeScript for the frontend to manage game state, render animations, and handle user inputs on an HTML5 canvas.

Multiplayer Uno in the Terminal | Independent **July 2024 – July 2024**

- Developed a WebSocket-based server for playing Uno, written in Rust to ensure high performance and low latency.
- Implemented the game logic, including rules enforcement and turn management, in a scalable, event-driven architecture.
- Integrated with a custom WebSocket client to provide a seamless multiplayer Uno experience.
- Developed an intuitive and charming TUI for playing Uno directly from the terminal.

OPEN-SOURCE CONTRIBUTION

St Jude Rust Labs – Bioinformatics Toolchains and Execution Engines

- Working alongside St Jude software engineers on the Crankshaft project, a headless task execution engine in Rust capable of performing concurrent workflow tasks on local, HPC and Cloud-based systems with the goal of making large scale bioinformatics processing accessible and efficient.
- Developed backend-agnostic configuration service in which backend behavior could be completely decided by a .toml configuration file. Utilized this format to create an HPC backend that can submit jobs to a cluster.
- Competed and presented Crankshaft as a part of the St Jude KIDS24 BioHackathon, where Crankshaft was ranked in the top 6 of final projects.