Tyler Neal

Embedded Software Engineer

Chesterfield, MI 48047 | +1 810-278-2749 | tylerneal.dev@gmail.com | github.com/tp-neal | linkedin.com/in/tyler-neal-dev

SUMMARY

Recent Michigan Tech Computer Science graduate specializing in embedded systems and low-level C/C++ programming. Passionate about the hardware-software interface, with hands-on project experience in firmware development for STM32 microcontrollers using FreeRTOS and bare-metal techniques. Eager to apply my skills to solving real-world engineering challenges in the automotive or robotics sector.

EDUCATION

Bachelor's of Science in Computer Science, Major GPA: 3.43 — 2020 – 2024
Michigan Technological University — *Houghton, MI*

PROJECTS

RTOS-Based Self-Balancing Robot | C, FreeRTOS, PID Control, STM32, ARM Cortex-M — June 2025 – August 2025

- Developed real-time firmware in C for a two-wheeled self-balancing robot on an STM32 microcontroller.
- Implemented a PID (Proportional-Integral-Derivative) control loop to maintain the robot's upright balance by modulating motor speed based on sensor feedback.
- Engineered a sensor fusion algorithm using a complementary filter to combine accelerometer and gyroscope data from an MPU6050 IMU for an accurate pitch estimate.
- Architected a multithreaded system using FreeRTOS, employing mutexes and semaphores to ensure thread-safe data sharing between sensor data acquisition and control logic tasks.

Bare-Metal "Simon Says" Game | C, Bare-Metal, STM32, ARM Cortex-M — May 2025 – June 2025

- Developed a "Simon Says" memory game on an STM32L476RG microcontroller, programming directly against hardware registers in C without the use of HAL or vendor libraries.
- Engineered peripheral drivers from scratch for GPIO pin configuration, SysTick timer for precise millisecond delays, and ADC for generating a random seed from pin noise.
- Configured the system clock by enabling the PLL to scale the HSI oscillator to 80MHz through direct register manipulation.

Multi-Layer Cache Simulator | **Research Project** | C, Spec95 — March 2024 – April 2024

- Designed and implemented a multi-layer (L1, L2, L3) direct-mapped write-back trace-based cache simulator in C
- Analyzed cache hit/miss rate, and memory access time using trace-based simulation with the Spec95 benchmark suite; validating trace files up to 6.15GB.
- Implemented multiple bash scripts to automatically run traces and print detailed cache access information based on cache configuration.
- Authored a technical paper detailing simulator design, performance metrics, and cache management techniques.

TECHNICAL SKILLS

Embedded Development: C, C++, Bare-Metal & RTOS (FreeRTOS) Programming, ARM Cortex-M, STM32
Hardware & Protocols: I2C, UART, GPIO, PWM, ADC
Software & Tools: Git, GDB, Make, Valgrind, Linux, Bash
Other Languages: Python, C#, Java, JavaScript, SQL