2026 Technical Documentation: FIRMWARE

2026 Technical Documentation: FIRMWARE

February 15, 2026

MISSION: **Explorer**

FROM: **California Polytechnic State University**

LEAD: **Jason K Tov**

MEMBERS: **Marie Klassen, Raymond Tran, Adit Srikumar, Colby Cordoba, Evan Lee**

Abstract

The ROV firmware system is built around an STM32 microcontroller that interacts with all onboard peripherals through a custom circuit board and connector network. The STM32 generates PWM signals to drive the thrusters and servos, communicates with the IMU over an I2C bus, and exchanges command and telemetry data with the topside computer through a UART connection converted to USB. This setup allows the microcontroller to handle real-time control while maintaining reliable communication with both sensors and the operator.

CAL POLY

Table of Contents

Why STM32F439ZI?

Overall the STM family is selected due to strong development support and a wide range of integrated peripherals suited for our control application. From a functional standpoint, STM32 allowed for hardware timers with PWM Outputs for thruster and servo controls, UART peripherals for a reliable serial communication to our topside, and I2C support for sensor and peripherals integration. The STM32F439ZI offers many features that we like and appreciated, All devices offer three 12-bit ADCs, two DACs, a low-power RTC, 12 general-purpose 16-bit timers including two PWM timers for motor control, two general-purpose 32-bit timers. The STM32F439ZI is likely overkill for the current application; it offers significant ability for future expansion. In hindsight, a lower power STM32 L series device may meet the current requirements but also provide a robust framework that will minimize power consumption and risk during development.

Why UART over USB to topside?

Communication between the ROV and topside computer is implemented using UART, converted to USB via a UART to USB interface which was chosen for simplicity, reliability and consistency. UART is simple to debug using standard serial tools and is already used extensively during firmware testing and development.

USB 2.0 Extender and USB 3.0 Tradoffs

Because the USB protocol has inherent distance limitations, the MCU's USB line is routed through a four port USB 2.0 to Ethernet extender along with the ROV camera feeds. This approach allows for reliable communication of up to 50 meters while keeping the system low profile and power efficient. USB 3.0 was considered to support a higher bandwidth camera but not pursued due to the increased power requirements of extenders. Instead USB 2.0 cameras were selected and provide a clear enough video feed without adding unnecessary complexity.

Our chosen system architecture allows for the cameras to be routed through the microcontroller which allows for both data and video to share a single USB connection; however this option adds significant complexity with little benefit given that the USB 2.0 extender already supports multiple independent USB connections.

Topside ROV Communication protocol

The topside ROV communication system was designed around simplicity and reliability. UART provides a clean and robust command interface while USB conversion enables easy integration with topside computers and USB 2.0 ethernet extender overcomes distance limitations with minimap power and size impact, the higher bandwidth processing and vision are internally handled though topside via the USB splitter.

The communication protocol currently sends a fixed number of byte commands along with a checksum byte for error detection. We will implement error correction for corrupted or missing bytes in the future via a TCP style PDU. It will incorporate the checksum byte, header length byte, and data length byte. If the recv() call does not return the correct header+payload size, then we will trash the current packet and wait for a new one.

Communication Diagram MCU – > Thrusters, Servos, IMU, UART USB

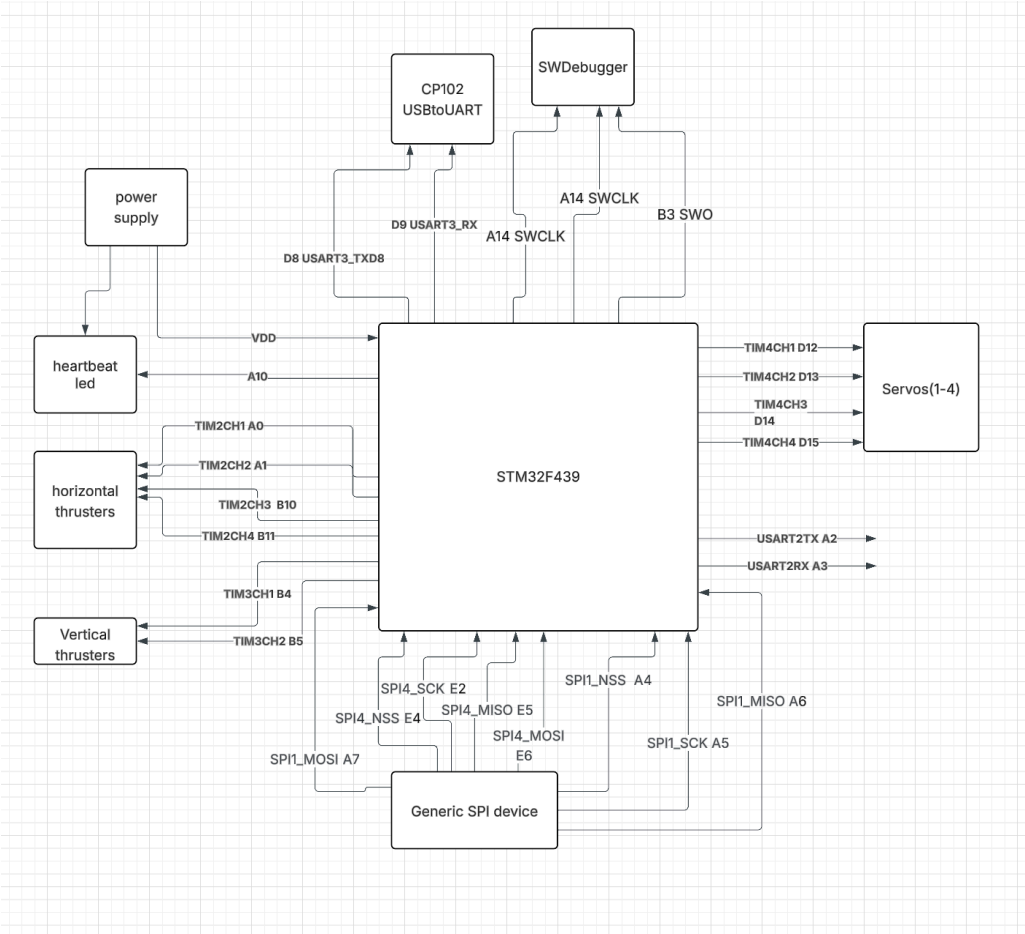

Figure 1. MCU Communication Diagram

This diagram illustrates the complete communication and control configuration for the STM32 based ROV controller. The STM32 microcontroller serves as the central processing unit, interfacing with topside systems, actuators, and onboard sensors through a combination of serial communication buses, timer driven PWM output and general purpose I/O.

Topside communication is implemented using a CP2102 USB to UART bridge rather than a STM32 native USB peripheral. The STM32 transmits and receives serial data using USART6 this also is USART_TX and USART_RX signals from the MCUconnect directly to CP2102. The

CP2102 converts UART data into USB D+ and D- signals, which connect to the USB port. Communication remains UART based while transported over USB and is later extended though a USB to ethernet extender in the tether.

Programming and debugging access to the MCU is provided through the Serial Wire Debug though interface using an ST link debugger.

- **3.3 V** reference
- **Ground**
- **SWDIO (PA13)** – bidirectional data line
- **SWCLK (PA14)** – debug clock
- **SWO (PB3)** – optional trace output

The MCU is powered from a regulated 3.3 V supply provided by the system power board. Power and ground connections are shown separate from signal lines to emphasize the separation between power distribution and control communication signaling.

A heartbeat LED is connected to **GPIO PA10** to provide a simple visual indication of system status.

- Periodic toggling confirms that the firmware is running

Thrusters and servos are controlled using **hardware PWM outputs generated by three independent timers**. Using dedicated timers ensures precise timing, consistent update rates, and minimal CPU overhead.

Horizontal Thrusters: **TIM2**

Thruster	Timer Channel	MCU Pin
Horizontal 0	TIM2_CH1	PA0
Horizontal 1	TIM2_CH2	PA1
Horizontal 2	TIM2_CH3	PB10
Horizontal 3	TIM2_CH4	PB11

Vertical Thrusters: **TIM3**

Thruster	Timer Channel	MCU Pin
Vertical 0	TIM3_CH1	PB4
Vertical 1	TIM3_CH2	PB5

Servo PWM Outputs: **TIM4**

Servo	Timer Channel	MCU Pin
Servo 0	TIM4_CH1	PD12
Servo 1	TIM4_CH2	PD13
Servo 2	TIM4_CH3	PD14
Servo 3	TIM4_CH4	PD15

The sensor subsystem is designed for flexibility and expandability, supporting multiple communication protocols.

UART Interface

Signal	MCU Pin
USART2_TX	PA2
USART2_RX	PA3

SPI Interfaces

Two SPI buses are provided to support high-speed sensors such as IMUs.

SPI1

Signal	MCU Pin
NSS	PA4
SCK	PA5
MISO	PA6
MOSI	PA7

SPI4

Signal	MCU Pin
NSS	PE4
SCK	PE2
MISO	PE5
MOSI	PE6

Additional IMU control signals

Signal	MCU Pin	Purpose
IMU_WAKE	PE0	Power / sleep control
IMU_RST	PE1	Hardware reset
IMU_INT	PE3	Interrupt output

I²C Interfaces

Two I²C buses are included to support lower-speed sensors and future expansion.

I2C1

Signal	MCU Pin
SCL	PB6
SDA	PB7

I2C2

Signal	MCU Pin
SDA	PF0
SCL	PF1

Multiple I²C buses reduce bus loading and allow logical separation of sensor groups.

Embedded-Software Interface

- https://github.com/CalPoly-UROV/Firmware25-26/blob/main/Documentation/Embedded-Software%20Interface.md