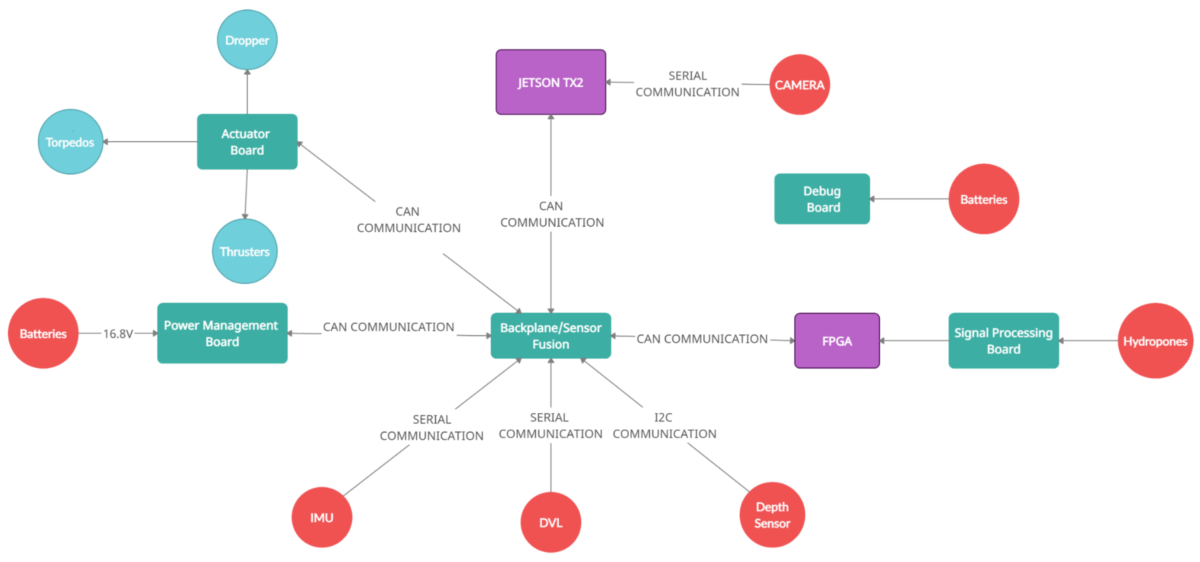

OVERVIEW:
Though the previous version of our electronics stack met all the requirements perfectly, but, it also has its issues that we have addressed in this year's rebuild.

One major issue was power management, previously we used to power the thrusters using one battery and rest of the circuit using another. This caused the thruster power to run out quickly than the other which reduced our bots runtime and efficiency, so this year we switched to a single power source. We now use a single battery at a time and after it runs out we switch to another using hot swap controllers.This results in increased runtime and better power management.

This year, we have decided to change the Microcontroller platform from ATMEL to STM32 line of microcontrollers giving us the flexibility to use High processing power where required or Low power usage if needed.

Last year's stack missed critical security features like Under and overvoltage protection, current spike and overcurrent protection. So we have added this protection using eFuses and TVS diodes. The thought of security leads us to the Kill switch, last year we used a REED switch as a kill switch which was extremely unreliable, and had a lot of false firing issues, this year we switched to a better and more reliable waterproof physical button as a kill switch.

Another major issue was the lack of any "Debugging data" making any sort of debugging a major pain in the buttocks, so we have introduced an independently powered (2x18650 cells) custom made debugging board, which allow us to keep track of Voltages of various nodes of the circuit along with current going to each thruster and temperatures of ESCs. This data will be sent to the main CPU using CAN protocol (which will also be the standard protocol all over the stack) then the CPU can use this data in intelligent decision making like shutting down parts of the stack in case of local errors and saving the rest of the circuit and AUV. the board will also have an EEPROM to store all this data for later use in analytics.

Following the debugging line of thought, in our previous stack making any kind of repair was also difficult, so this time we have decided to replace a single PCB with various modularized PCBs making it easy to just replace the problematic board with a replacement much like any other commercial electronics gadget. Also, this allows our debugging board to execute the local shutdown of unimportant components in case of an error in the board.

POWER MANAGEMENT BOARD:
As already discussed due to galvanic isolation, we powered the Thrusters with one battery and rest with the another. The Thruster battery discharged fast, decreasing runtime and efficiency. So, this year a smart battery management system was incorporated which used 2 16000 mAh LIPO battery packs but one battery at a time.After a battery discharges to a minimum preset level we switch batteries with help of a hot swap controller and a STM32f103c8t6. This board is also responsible for generation of different voltage levels,i.e. 12V,5V and 3.3V to meet the requirements of different components across our circuit.

BACKPLANE/MAIN BOARD:
All the different boards are connected to the main board using high density connectors which are controlled by STMH745ZI-Q. This board communicates with JETSON TX2.
A high-quality, reliable Inertial Measurement Unit (IMU) is a crucial part of any autonomous underwater vehicle (AUV). Previously our AUV had to rely on accelerometer and gyroscope signals to obtain velocity and position values, which was giving lots of error in determining the exact position. So to assist our previous IMU, we are installing Doppler Velocity Log (DVL) in Makara which serially communicates to the CPU. Erroneous velocity values that are impossible or unlikely are filtered out. For samples when no values are received, the program extrapolates until a real value is received.

Furthermore, to provide an accurate way of measuring depth, a pressure-based depth sensor was also added to our AUV which communicates with STM32 microcontroller through I2C.

ACTUATOR BOARD:
The AUV has a support for 10 thrusters and also we have incorporated MOSFETs to drive solenoid valves for torpedoes. We use a mix of T200 and T100 thrusters, controlled using BlueESC. We are also planning to introduce current sensors to measure the current going to individual thrusters thus helping us to get an approximate idea for speed of thrusters as we can't have an encoder on the thrusters.

DEBUGGING BOARD:
In our previous circuit, we used LEDs to debug circuits which made troubleshooting difficult and time consuming. So, in the new circuit we introduced an independent debug/protection board with an onboard OLED screen powered with an independent power supply. This board contains a STM32f103c8t6 microcontroller which monitors voltage, current and temperature at all important nodes using LTC2990. The data acquired is stored in an EEPROM which can be used for later debugging. Our circuits have also been incorporated with TPS2660x efuses to ensure circuit protection. These efuses can also be controlled by the debugging board in case of any emergency.

ACOUSTIC BOARD:
In our previous bot, we tried making passive sonar for locating the pinger with hydrophone and RaspberryPi with external ADC but that did not turn out to be working. So This time we planned the whole Acoustic signal processing part based on FPGA. We designed our own Analog frontend board which Has Adjustable Bandpass filters, Amplifiers with Adaptive gain control, and High-speed ADCs which connect to the Zybo Z7 FPGA board. We find out the pinger heading by finding the Time Delay of Arrival of signals. FPGA can communicate to onboard Computer via Ethernet or CAN bus
There is an onboard AVR microcontroller ATMEGA328p that controls the Frontend gain and filtering also controls various power rails within the board. This board has its own local power distribution to meet the special requirements of the board. The microcontroller also interfaces with the debugging board via CAN and can regulate high-performance and

power-saving modes. And power down partially or fully in case of fault when signaled from debugging board.