Proof of Concept Test Demo Sheet

**TEAM NAME: Oceans 7 Test Date & Time:**

**Test # and Name (from PoC plan) 2B: Communications stress Test Test Type: Feasability**

1. **Purpose of Test and How it Relates to Project (Brief, concise, but complete, description):**

Critical Functionality Demonstrated: UART communications and PWM signals will function in an environment with the high levels of EM noise that will be generated by the eight motors drawing upwards of 200A.

Relation to Overall System: Communications testing will ensure that the various submodules of the CU RoboSub are able to reliably communicate with each other. If digital communications are untenable in the noisy environment that is expected then alternate designs will have to be pursued for communications and or power systems.

1. **Test Setup, Pre-conditions, and Procedure (Brief, concise, but complete, description):**

Set Up and Preconditions

- Two microcontroller dev boards (the STM32 Nucleo-144 and an Arduino Uno, in place of the sensors that use I2C comms) with firmware (Nucleo\_AXIM\_FLASH project in SW4STM) loaded to support sending and receiving data via UART communication protocols at 38400 Baud

- A desktop computer with a serial terminal to communicate with the microcontroller

- Electromagnet, or power drill to simulate electromagnetic noise

- Backplane test board

- BlueRobotics ESC 30A (electronic speed controller)

Note: The *control set* of strings is defined as {‘UART’, ‘motor-on’, ‘motor-off’}. These are used throughout the test.

Testing Procedures:

1. Connect the Nucleo and Arduino boards to the computer via USB
2. Connect the TX pin of the Nucleo to the RX pin on the backplane test board
3. Connect the RX pin of the Nucleo to the TX pin on the backplane test board
4. Connect the TX pin of the Arduino to the RX pin on the backplane test board
5. Connect the RX pin of the Arduino to the TX pin on the backplane test board
6. Send the control string ‘UARTXXX’, where ‘XXX’ is the number of tests. This tells the Nucleo board to send the poem “Good Night Moon” to the Arduino and compare the returned string *ntests* times.
7. The terminal will the number of error bytes received (bytes that don’t match the data sent). Repeat step F as many times as desired and record any instances of error.
8. Connect the 14.8V line from the voltage supply to the 14.8V pin on the backplane
9. Connect the ESC PWM header to the male ESC header on the backplane
10. Using jumper wires, connect pin PA6 of the Nucleo to the PWM pin of the female ESC header on the backplane, and connect the other 2 pins on the ESC header to 5V and Ground (they are labeled)
11. Connect the power and ground cables of the ESC to the connectors labeled as ESC power and ESC ground
12. Plug the 3-phase cables into the motor
13. Send the string ‘motor-on’ to turn on motor, then place it near the backplane test board (note: ‘motor-off’ will turn off the motor). Control the motor in the same manner as in test 3A, and place near the communication traces on the backplane test board.
14. Additionally, for more noise, run a power drill near the data lines at full speed.
15. Repeat steps F and G recording the data separately
16. Turn off the voltage source, disconnect the microcontrollers, and remove the power systems test board from the backplane test board
17. Tally the number of errors encountered and calculate the rate of error relative to the number of bytes transmitted and received

1. **Success Criteria (feasibility) / Selection Criteria (alternatives) / Design Info (gathering): (Concise, complete, quantitative):**

Fidelity of data sent via I2C is 99% or higher

Fidelity of data sent via UART is 99% or higher

Fidelity of data sent via I2C with the motor running is >95%

Fidelity of data sent via UART with the motor running is >95%

1. **Instructional Team Notes:**
2. **Test outcome and what was learned (toward finalizing design):**