2014

Exo Terra Monsoon RS400: Misting Frequency Flaw Analysis and Resolution

BY: SCOTT RISING

Contents

List of Figures 1
Introduction 2
Background 2
Hardware Analysis 2
Frequency Flaw Analysis 3
Frequency Flaw Analysis Results 4
Frequency Flaw Root Cause Analysis 5
Frequency Flaw Resolution 6
Digital Upgrade 7
Works Cited 9

List of Figures

Figure 1: RS400 Main Logic Board, Top 2
Figure 2: RS400 Main Logic Board, Bottom 3
Figure 3: RS400 Main Logic Board Testing Setup 4
Figure 4: RS400 Digital Upgrade 8

Introduction

The Exo Terra Monsoon RS400 is a high-pressure, programmable misting system designed for animal- or plant-based terrariums. According to the product page (1), "The system can be programmed to mist multiple times per 24-hour cycle, for time periods ranging from one second to two hours. The large 9.5 liter (2.5 gallon) reservoir ensures continuous operation for several days without the need for refilling (ideal for vacations!), and the reservoir can be easily replenished without needing to uninstall the entire system."

Background

The Monsoon RS400 suffers from a flaw that makes the system unusable: it will mist far more frequently than programmed. Initial observations showed that the system would completely drain its 2.5 gallon reservoir within a day, even when the system was configured to mist for 30 seconds in a 24-hour period. Discussion with other owners of the system revealed similar results; additionally, a local pet store reported that many of the Monsoon RS400 units are returned due to the frequency flaw.

Hardware Analysis

The electrical/electronic hardware external to the Monsoon RS400 consists of a power supply, which accepts 120VAC and converts it to 12VAC, where it connects to the top of the mister unit. Internally, the mister unit consists of a small power supply and pump driver board, which then connects to the main logic board. The main logic board appears to be driven by a PIC microcontroller. Since the microcontroller has no markings on it, it wasn't immediately identifiable; however, pin placement matches several products in the PIC16F18xx microcontroller family. This board also has three momentary-contact buttons, two LEDs, two potentiometers, and an IR-receiver. The board is fed with 5VDC power, and one of the output pins on the microcontroller drives a relay on the pump driver board, which then turns the pump on and off.

The microcontroller and potentiometers appear to be properly de-coupled using capacitors. Images of both sides of the logic board are shown below.

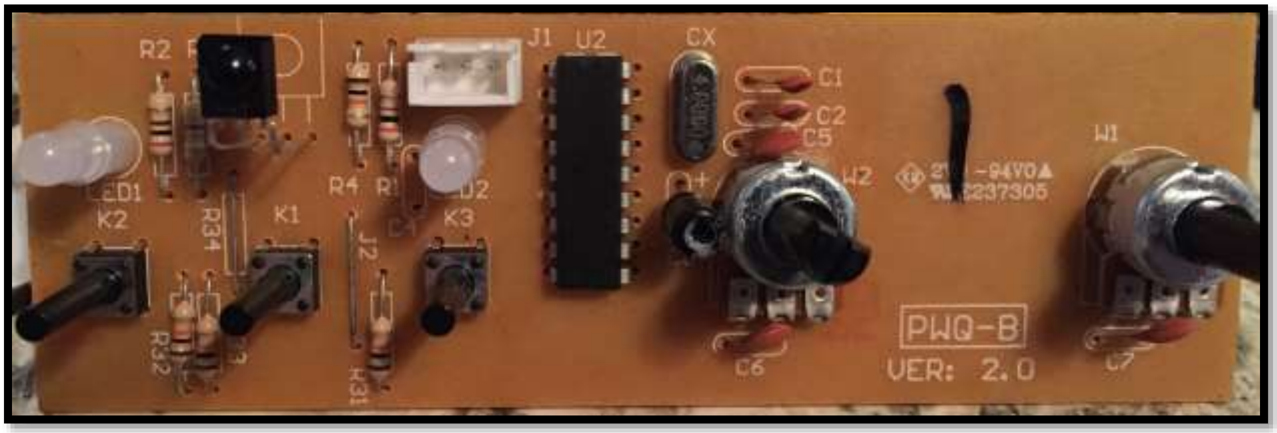

Figure 1: RS400 Main Logic Board, Top

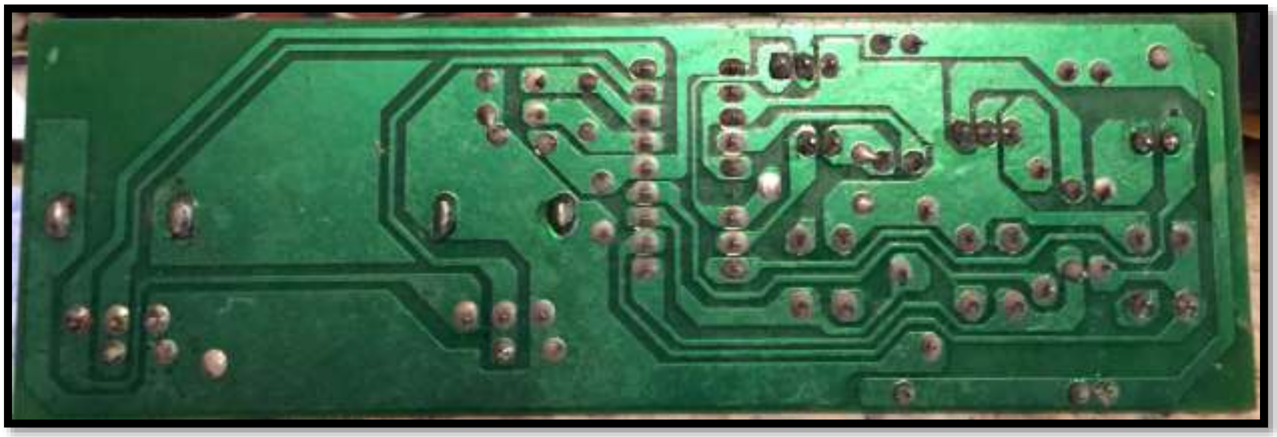

Figure 2: RS400 Main Logic Board, Bottom

Frequency Flaw Analysis

There are two main points in the system that could cause the frequency flaw. The first would be a case where the pump would get stuck in an enabled state, and the second would be a flaw in the logic that controls the pump. In order to test each major system component, the main logic board was removed from the mister unit. The pump portion of the mister unit was connected to an Arduino Nano v3.0, loaded with a sketch that would activate the mister for 30 seconds every 8 hours. The pump system with Arduino controller was fitted back onto the water reservoir, the reservoir filled, and the output of the pump fed into a second reservoir. The Arduino-driven pump system was then plugged in and was allowed to run for one week.

The second possible failure point was tested by powering the main logic board and connecting the pump output of the microcontroller to an Arduino Uno. The Arduino was loaded with a sketch that continuously tracked the state of the pump output pin and reported the durations between each time the pump output was enabled or disabled.

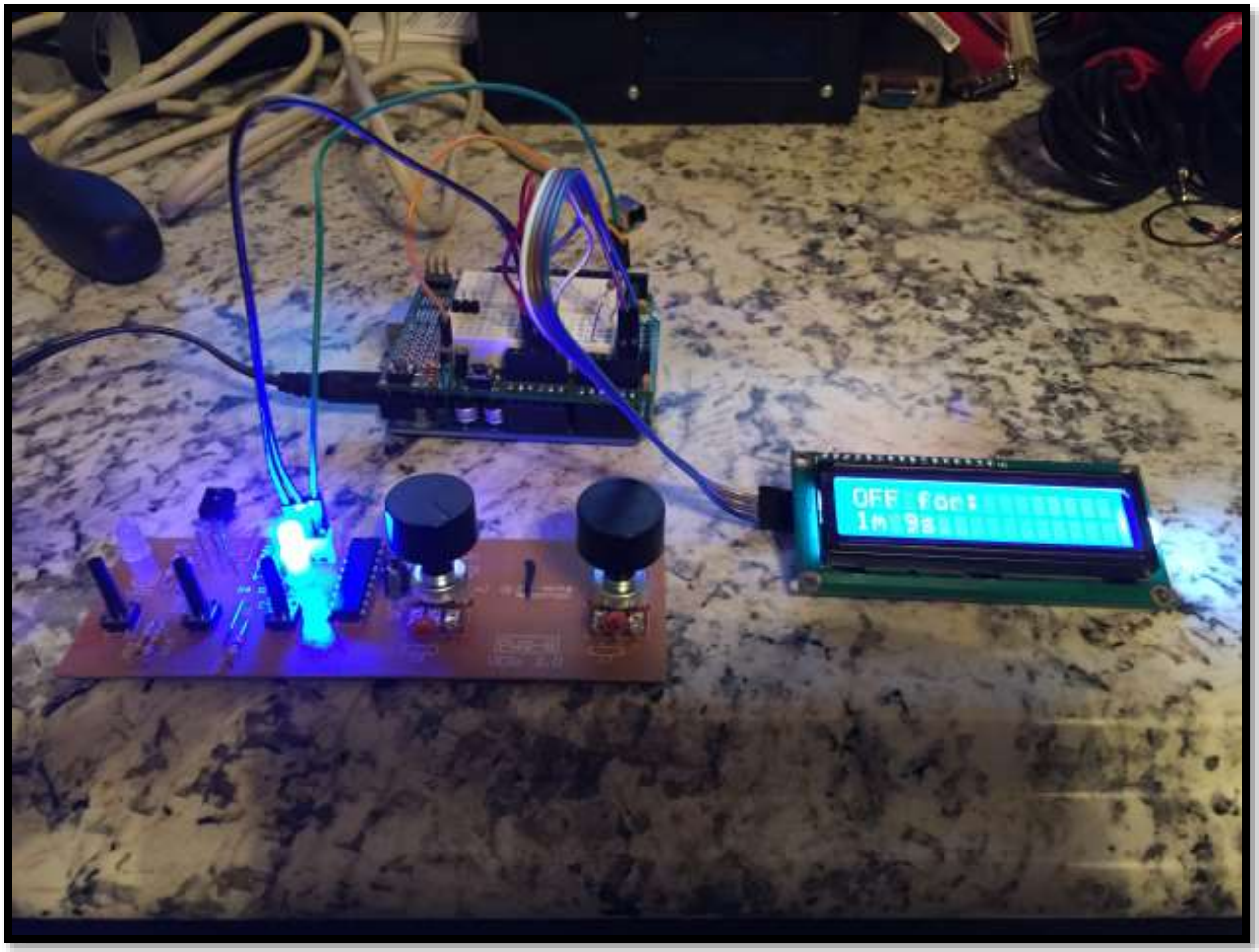

Figure 3: RS400 Main Logic Board Testing Setup

Frequency Flaw Analysis Results

The Arduino-driven pump system ran for one week, and never experienced any anomalies, and the pump was never observed to be in a “stuck” state. The reservoir on the Monsoon system was depleted at a constant rate.

The Monsoon main logic board, however, had radically different results. The logic board was set, using the potentiometers, with a cycle of 8 hours and a duration of 16 seconds, meaning that once every 8 hours, the pump would be active for 16 seconds. As potentiometers are far from exact, the actual cycle setting that the microcontroller was attempting to set was likely 7 hours, 10 minutes, and 40 seconds (this cycle value appeared multiple times in the log). The main logic board was sampled for 45.47 hours, and over that course, 818 samples were recorded. See below for tables detailing the results.

OFF Time Setting	**8**	**hours**
Correct OFF Time Total	45.44	hours
Measured OFF Time Total	42.88	hours
Measured OFF Time Mean	175.5	seconds
Measured OFF Time Median	175.5	seconds
Number of Times OFF is ZERO	349	
Number of Times OFF is Less than 60 Seconds	365	
Number of Times OFF is Less than 8 Hours	405	

Table 1: RS400 Main Logic Board "OFF Time" Frequency Analysis

ON Time Setting	**16**	**seconds**
Correct ON Time Total	91	seconds
Measured ON Time Total	9327	seconds
Measured ON Time Mean	59	seconds
Measured ON Time Median	59	seconds
Number of Times ON is ZERO	0	
Number of Times ON is More than 16 Seconds	126	
Number of Times ON is More than 30 Seconds	90	

Table 2: RS400 Main Logic Board "ON Time" Frequency Analysis

The critical items in the data are the "Correct ON Time Total" and the "Measured ON Time Total." These values show the total number of seconds that the mister should have been active during the 45-hour period, and the number of seconds that the mister actually was active during the 45-hour period. As the data shows, the mister should have been active for a total of 91 seconds, and it was actually active for 9327 seconds (2 hours, 35 minutes and 27 seconds). This means the mister ran 102.5 times more than it should have.

Frequency Flaw Root Cause Analysis

The testing data clearly demonstrated that the main logic board in the Monsoon RS400 has one or more serious flaws. Since the Monsoon's logic board was not connected to the pump, pump motor noise can be ruled out as a cause for the flaw. The only places left where these flaws could occur would be within the software of the microcontroller or within the potentiometers themselves. Debugging the software in the microcontroller is virtually impossible without having the source code for the software, so the only components that can be tested are the potentiometers. In order to accurately test the potentiometers, the PIC microcontroller was removed from the board, and an Arduino Uno was connected to the board in its place. See below for the pin configuration of the microcontroller.

Pin	Function
1	Vcc (5v)
2	GND
3	GND
4	10kOhm Resistor to GND
5	GND
6	N/C
7	Pump/Relay Output
8	LED2
9	LED1
10	K1 (Btn)
11	K2 (Btn)
12	K3 (Btn)
13	IR Input
14	Vcc (5v)
15	Crystal Oscillator
16	Crystal Oscillator
17	W2 (Pot)
18	W1 (Pot)
J1 From Component Side:	
Pin	Function
1	GND
2	Vcc (5v)
3	Relay Output

Table 3: RS400 Microcontroller Pinout

The Arduino Uno was loaded with a sketch that would read the potentiometer voltages using two analog inputs and display the resulting voltages. The resulting data showed that the potentiometers readily supply inconsistent voltages, many times even supplying a voltage of 0 when set otherwise.

Potentiometers, like many other analog information sources, are prone to spurious readings (2). When sampling analog systems, it's important that the software perform smoothing in order to help filter out bad or spurious readings from the device. It's apparent that the PIC microcontroller is only reading each potentiometer once. If a single reading produces an incorrect value, it could (and apparently does) cause the mister to turn on at an incorrect time.

Frequency Flaw Resolution

In order to properly read the analog potentiometers, multiple values should be read and stored. Following a number of analog readings, an average should be taken across all samples. Since these potentiometers produce spurious readings fairly frequently, 20-50 readings should be taken and averaged (the number of readings can depend on how quickly the microcontroller can read an analog input). To further refine the results, the software could scan back through the list of readings and remove any readings that are significantly different from the average, then re-average the remaining values.

Digital Upgrade

Since the PIC in the Monsoon RS400 cannot easily be identified, it's difficult to attempt replacing the firmware that's loaded on the microcontroller. In order to salvage this particular unit, I wanted to replace the PIC with an Arduino. Unfortunately, the pin configuration of an ATmega 328 does not match the pin configuration of the PIC. Therefore, I removed the PIC and replaced it with two female header sockets, onto which I plugged a small board containing an Arduino Nano and a few components. I also cut the original PCB in half, removing the potentiometers and crystal oscillator in order to make room for a 16x2 character LCD.

The firmware on the Arduino precisely controls the water pump, and allows the user to change the following settings using the three momentary-contact push buttons: cycle time, duration time, LED brightness, LCD backlight brightness, and LCD contrast. All mister functions continue to operate as originally designed, including the IR receiver. (3)

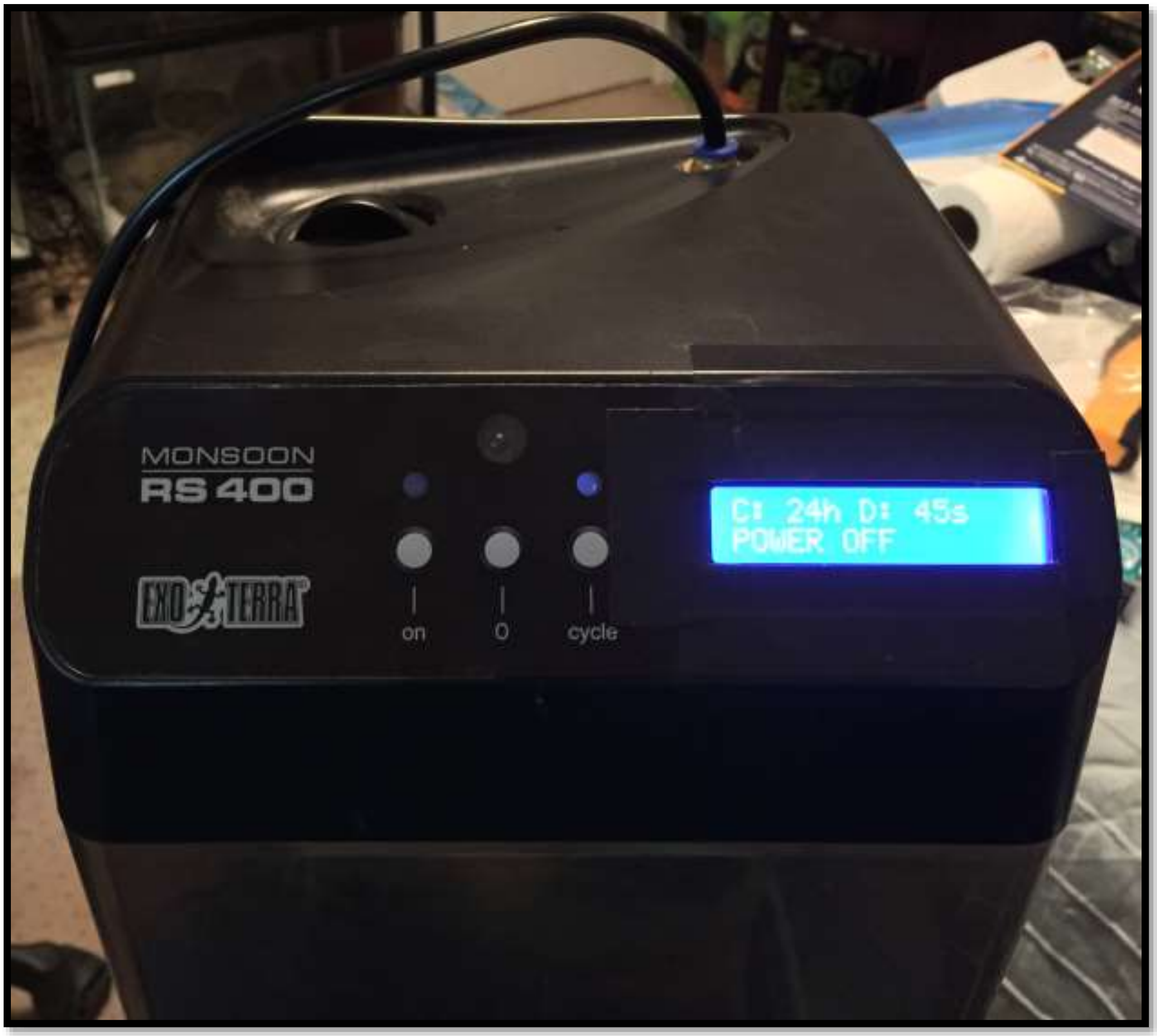

Figure 4: RS400 Digital Upgrade

Works Cited

1. **Exo Terra : Monsoon RS400 / High-pressure Misting System. [Online] [Cited: 12 2, 2014.] http://www.exo-terra.com/en/products/monsoon_rs400.php.**

2. Zettlex. *Why do engineers dislike potentiometers?* [Online] [Cited: 12 22, 2014.] http://www.zettlex.com/articles/engineers-dislike-potentiometers/.

3. Rising, Scott. MonsoonRS400Controller Arduino Sketch. *GitHub.* [Online] 12 22, 2014. [Cited: 12 22, 2014.] https://github.com/Xerxes3rd/MonsoonRS400Controller.