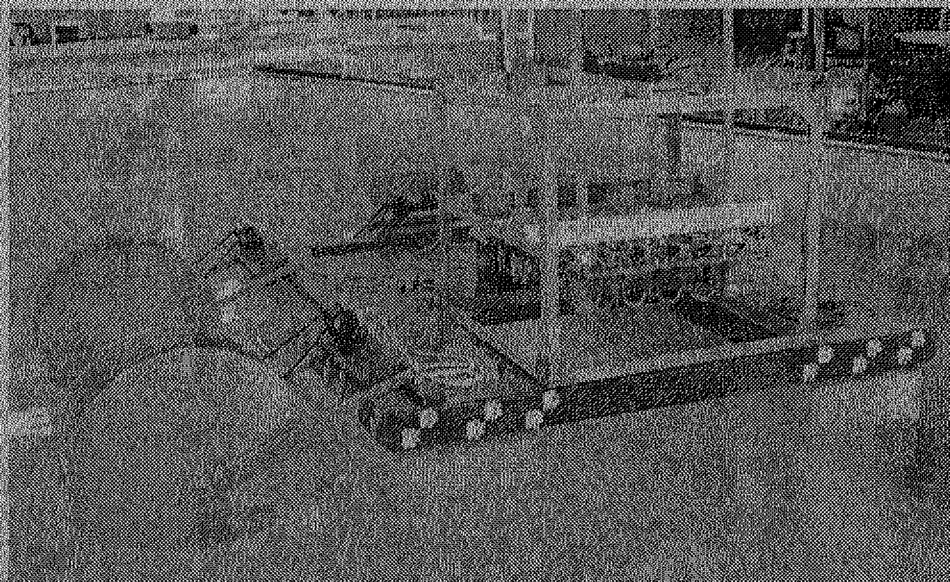
The Clean Cat



University of New Hampshire
Tech 797: Ocean Research Projects
Senior Design

2001-2002

Project Advisors:
David Fredriksson
Tom Ballestero
Nancy Kinner

Team Members:
Stanley Boduch
Matt Manning
Mike Mazzone

Date Submitted:
April 26, 2002

Sea Grant
New Hampshire

The Clean Cat

University of New Hampshire
Tech 797: Ocean Research Projects
April 26, 2002

Stanley Boduch
Matt Manning
Mike Mazzone

Advisors:
Prof. David Fredriksson
Prof. Tom Ballestero
Prof. Nancy Kinner

UNHMP-TR-SG-02-12

Acknowledgements

Team Clean Cat would like pause for a few minutes to thank all those who have helped make this project come to life:

Prof. David Fredriksson	– UNH ME/OE
Prof. Nancy Kinner	– UNH CIE
Prof. Thomas Ballestero	– UNH CIE
Prof. Igor Tsukrov	– UNH ME
Prof. Barry Fussel	– UNH ME
Adam Perkins	– UNH EE
Tracey Harvey	– UNH ME
Jim Irish	– Woods Hole Oceanographic Institute
The University of New Hampshire Ocean Engineering department	
Bison Gear and Engineer Corp.	– 3850 Ohio Ave. St. Charles, IL 60174 630-377-4327
MACY Industries	– Manchester, NH
Rabbit Semiconductor	– 2932 Spafford Street Davis, CA 95616 530-757-8400
Technology Systems	– 16610 East Laser Drive Suite 10 Fountain Hills, AZ 85268 480-837-5200
Mass Design	– 22 Tanguay Ave. Nashua, NH 03063 603-886-6460
Datel	– 11 Cabot Boulevard Mansfield, MA 02048-1151 508-339-3000
MTI Research	– 7 Littleton Road Westford, MA 01886 978-692-9898

This work is the result of research sponsored in part by the National Sea Grant College Program, NOAA, Department of Commerce, under grant #NA16RG1035 to the New Hampshire Sea Grant Program.

Table of Contents

List of Figures

List of Figures Cont.

List of Tables

Abstract

When coastal oil spills occur the petroleum will tend to move ashore if there is an incoming tide present. This tidal surge carries contaminate into surrounding estuaries and salt marshes, devastating the local ecosystem. The oil mousse, as it is called, clings to everything in its path including birds, fishes, mammals and all forms of plant-life, the further it is diluted the more surfaces adhesion occurs. The immediate marsh cleanup response starts with containing the affected area with booms to prevent further spread, then either wiping the affected location with absorbent materials, pressure washing, raking/bulldozing, or commonly a combination of all three. Current long-term methods involve either in-situ cutting and burning of the marsh grass, sediment removal and replanting, natural attenuation (biodegradation), or bioremediation. Bioremediation, by biostimulation or bioaugmentation, has recently emerged as one of the most effective and inexpensive methods of decontaminating groundwater and soils. Of specific importance to this project is biostimulation by the introduction of nitrate amendments to the marsh, which has been shown to increase the rate of hydrocarbon degradation in the sediments by as much as 50%. This technique introduces the need for an apparatus to distribute the stimulant throughout the marsh, as current methods are costly and inefficient, a system that will not damage the site or disturb the ecosystem would be ideal.

In this project a scale prototype shallow water injection delivery vessel was designed and constructed with a budget of less than $2000. To design a system of this nature many ideas were discussed on the most effective means of transporting an injection system in a marsh environment. Many propulsion, flotation and control system ideas were discussed and tested. The use of large paddle wheels with slow spinning motors became the primary objective for the design of the craft, which would provide the vessel with a high amount of torque at a low speed in order to 'crawl' through the marsh grass. Flotation was achieved using closed cell polyurethane foam for the needed vessel buoyancy of 500-800 lbs and maximum draft of no more than six inches. The vessel is controlled wirelessly through a single board computer and has the potential of being automated with the addition of a positioning system. This system has been designed as a feasibility model for future implementation of a full-scale prototype and possible production for commercial use.

Fore River Creek, Portland ME - October 2001

Introduction

As the use of fossil fuels, such as petroleum, continues to grow [1], the likelihood of oil spills become decreased due to the increase in stringent shipping safety standards and better hull containment engineering. However, coastal waters, estuaries, and marshes made fragile by extensive human impact become increasingly susceptible to catastrophic degradation by an oil spill. When such an event occurs response has to be swift, thorough, and considerate of the delicate surroundings.

Marine shorelines serve as homes to a variety of plant and wildlife. Many species of birds build their nests on sand, while others regularly wander the shoreline searching for food. Fish, such as salmon, swim near shorelines on their upriver migrations during spawning season, their offspring travel down through the same areas on their way to the sea the following year. Marine mammals, such as seals, otters and sea lions, come ashore to breed and bear their young. There are a host of other insect, microorganisms, and vegetation that create sustenance and shelter which allow these ecosystems to survive.

When coastal oil spills occur the petroleum will tend to move inshore usually by tidal currents but sometimes due to wind. This surge carries contaminate into surrounding estuaries and salt marshes, devastating the local ecosystem. Light oils tend to evaporate and degrade quickly and therefore, do not tend deposit themselves in large quantities on banks and shorelines. Heavier oils tend to form a thick oil-and-water mixture called mousse that clings to everything in its path including birds, fishes, mammals and all forms of plant-life along with the surrounding rocks and sand/sediment. The immediate marsh cleanup response starts with containing the affected area with booms to prevent further spread, then either wiping the affected location with absorbent materials, pressure washing, raking/bulldozing, or commonly a combination of all three [2]. Existing long-term methods involve either in-situ cutting and burning of the marsh grass, sediment removal and replanting, natural attenuation (biodegradation), or bioremediation. Bioremediation, by biostimulation or bioaugmentation, has recently emerged as one of the most effective and inexpensive methods of decontaminating groundwater and soils [3][4].

Of specific importance to this project is biostimulation by the introduction of nitrate amendments to the marsh, which has been shown to increase the rate of hydrocarbon degradation in the sediments by as much as 50%. Existing methods being studied for amendment introduction include; one, installation of horizontal wells, which were time consuming to install, damaged the sediments upon installation and removal, and provided only minimal marsh coverage; and two manually injecting the nutrients into the sediment using large syringes, which requires much labor. Therefore, due to the cost and inefficiency of these methods it would be beneficial to create an apparatus to distribute the amendment throughout the marsh, a system that will not damage the site or disturb the ecosystem would be ideal.

In this project a scale prototype shallow water injection delivery vessel was designed and constructed. "The Clean Cat", as its aptly named is the result of a two part feasibility project, one being the injection system nicknamed "NISS", and the other part being the actual shallow delivery craft.

The general design criteria for The Clean Cat were one maneuverability in the marsh, two minimize damage to the marsh grass, and three the vessel had to draw less than six inches of water at full load. The production budget of less than $2000 was also a limiting factor in the system design.

To design a system of this nature many ideas were discussed on the most effective means of transporting an injection system in a marsh environment. Many propulsion, flotation, and control system ideas were discussed and tested. The use of large paddle wheels with slow spinning motors became the primary objective for the design of the craft, which would provide the vessel with a high amount of torque at a low speed in order to crawl through the marsh grass. Flotation was achieved using closed cell polyurethane foam for the needed buoyancy of 500-800 lbs and maximum draft of no more than six inches. The Clean Cat came to life as the control system was added; it is controlled wirelessly through a single board computer and has the potential of being automated with the addition of a positioning system. This system, as stated, has been designed as a feasibility model for future implementation of a full-scale prototype and possible production for commercial use.

Design Overview

Since there are no designs like this in use today, all the systems had to be developed from a clean drawing board to obtain the desired operation and control of the craft in the marsh environment. The major design issues faced were to not damage the surroundings in any way and to not stir up bottom sediment of the marsh. The vessel velocity would not be an issue due to the fact that the injection cycle needs to occur every three feet and lasts for fifteen minutes (See NISS TECH 797 project report). The system was also to be controlled wirelessly, so the operator would not have to be onboard during the remediation process.

Paddle wheels, driven by slow turning gear motors, were chosen to propel the vessel through the marsh environment. This allows for a large torque so that the paddle wheels can 'grip' and push off the thick marsh grass. The benefit of using the slow moving paddle wheels is to allow the marsh grass to slide between the paddles-free of damage.

The flotation system is comprised of two closed cell polyurethane foam pontoons. The foam was chosen due to its effectiveness in providing buoyancy at an inexpensive price. The ideal flotation units would be made of lightweight aluminum shells or rigid plastic, these would provide the most resistance to collision damage in the marsh area.

The control system has been designed for remote user control of the craft. The controller allows the user to turn the paddle wheels on and off whenever needed and link to the injection system to start the injection process. This is a necessary first step in the automation design process because at some point an operator will have remotely override the automated system to bring it to shore or a docking station for maintenance and inspection purposes. The next step would be to fully automate the system using a proposed laser positioning system that will allow the vessel to carry out its injection process autonomously.

Propulsion System: Design, Testing, and its Operating Environment

The first phase in the design of the bioremediation vessel was to determine a way of moving the vessel through the marsh grass without causing damage. This is a difficult task because the grass is delicate and can easily be destroyed. The first thought was to use small trolling motors incorporating weed guards, or shrouds, to prevent the grass from becoming entangled in the propeller. This led to testing the concept, using a canoe and a small trolling motor in a similar environment. Unfortunately, a canoe paddle was used more than the trolling motor because the grass became entangled very easily choking the propeller. This led to the motor chewing the grass (See Figure 1). The idea of a shroud was analyzed but dismissed upon the realization that the amount of force needed to move the craft through the water was much more than the unshrouded propeller could provide.

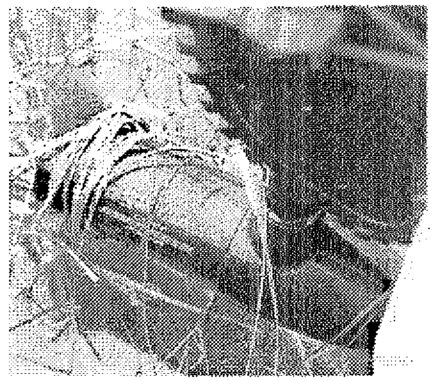


Figure 1: Trolling Motor testing, note the marsh grass getting caught around the propeller shaft.

The next concept was to use large paddle wheels on the sides of the craft to allow it to move slowly through the marsh and to prevent grass entanglement. One of these paddle wheel designs is shown in Figure 2.

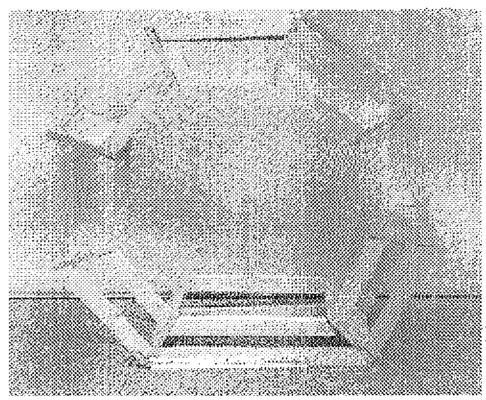


Figure 2: Wooden Paddle wheel Model

This design was simplified into what is shown in Figure 3. The basic design of this paddle wheel was to show that by using large width paddles, the wheel would be able to produce an effective amount of thrust while spinning very slowly. To test the paddle wheel concept, a prototype paddle wheel was mounted to two floating docks using some inexpensive lumber in a UNH test tank. This paddle wheel, consisted of two 36-inch diameter disks made entirely of steel, which made it heavy, and gave it a lot of inertia.

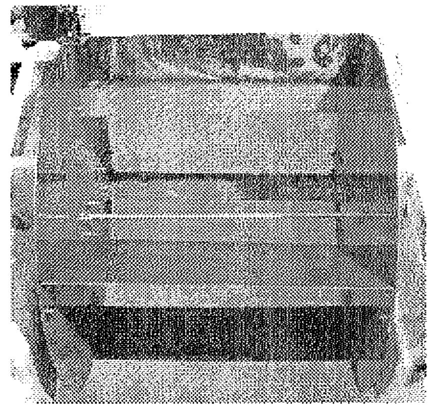


Figure 3: Test paddle wheel (steel design)

The system incorporated 4 flat paddles that were 28 inches wide with 4-inch flaps to bolt them to the disks. Two extra paddles were included to see how it would work with more propulsion area. A small AC gear motor with a 10:1 gear ratio was then attached to a pulley system that used a 3:1 gear ratio to increase system torque. As shown in Figure 4, the paddle wheel concept proved to work effectively.

Testing

To further test the paddle wheel concept, the system was dynamically modeled to investigate the response of the vessel during operation. This testing was done in conjunction with a project for a class on complex systems design and modeling. The system was modeled using two spinning masses, one for the motor and one for the paddle wheel. Some damping was taken into account for losses due to friction in the bushings and the flotation drag in the water. Figure 4 shows the test model setup.

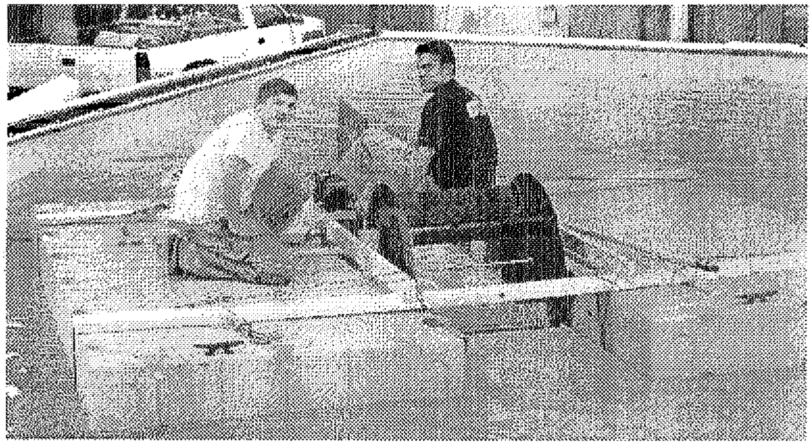


Figure 4: Testing apparatus for paddle wheel

The first test was to determine the maximum speed of the setup as it moved across the pool at different voltages. The data acquired was used to determine how fast the paddle wheels would need to rotate for final vessel design. A spring scale attached between the test-apparatus and the edge of the pool determined the thrust at various voltages. This setup produces a large amount of thrust at a very low speed, which is the primary goal. The thrust did however vary depending on where the paddle wheel was positioned in its cycle. It was speculated that the undercurrents formed by the stationary, yet rotating, paddle wheel form eddies and push the apparatus backward. The

Voltage (V)	Low Load (Lbs)	High Load
12	0	3
16.4	1.5	5
25	4.5	8.5
30	7	11
35	8	13

Table 1: Force response of spinning paddle wheel as voltage was varied

results show the low and high values during this cycle. The theory is that if the wheel rotation speeds are low enough then the design should allow the grass to slip between the paddles unharmed. The data that acquired by this test is shown in Table 1.

The next test performed determined how much drag there was on the test system (from the dock floats). To obtain the drag of the boat it was dragged across the pool using the movable XY table mounted on the tank. The table consists of two I-beams that move the distance of the pool on rollers. The velocity of the table can be varied with a handheld remote control. Using the tow system and a spring scale, the drag force was measured that resulted from pulling the system at a constant velocity across the pool. The data obtained for this test is shown in Table 2. The floats used for this test gave an over estimate of the pontoon area by double the amount. Using such large floats was one of the methods used to account for the additional affects of the marsh grass to help size the propulsion system.

Trial	Force (Lbs)
1	6
2	6
3	5.5
4	6

Table 2: Drag testing results when system was pulled at 1.08 ft/s

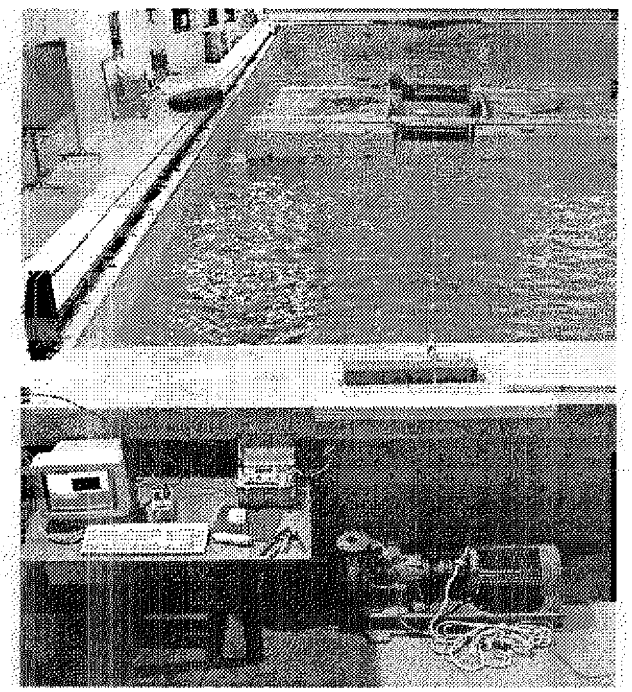


Figure 5: Load cell testing.

The final test conducted was using a load cell to determine the thrust being produced at a given voltage. This test setup is shown in Figure 5. The test showed gave an indication of how the thrust varied as the paddles cycled through the water and also backed up the initial spring scale thrust data with concrete data.

This testing resulted in some basic design parameters when considering motor size and output rotation speed. It was decided that a rotation speed of 20 RPM's would be sufficient for the system, with a motor torque greater than 100 ft-lbs.

Alternative Propulsion Ideas

Before deciding on the existing paddle wheel design, some other ideas were considered and could be implemented in future designs. The idea of using some kind of jet pump system was thought of and dismissed due to budget issues and complex designs. The jet pump would also move at high speeds, which is not necessary for this system, due to the fact that the boat only moves three feet at a time. Another design considered was to use the injection grid to 'pull' the vessel through the water. This design could be costly and complicated. This concept has potential, however, due to the fact that it would allow the craft to move while it is injecting, hence speed up the injection process by covering more distance in less time. Figure 6 shows the design in more detail.

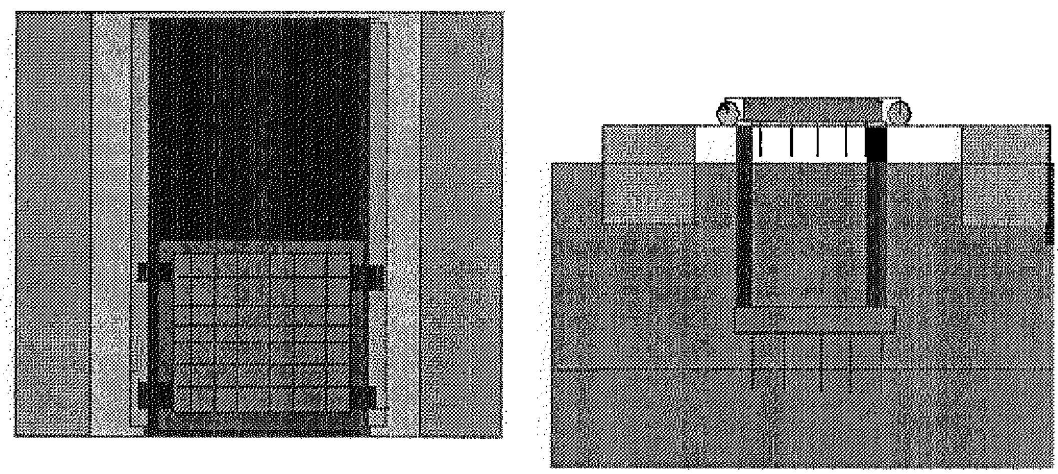


Figure 6: Alternative Propulsion System

The basic idea to make this work would be using a couple of small servo gear motors and some long rigid tubing to raise and lower the injection system. Keep in mind that this would all have to be tested to see if it would work, the idea was found to be intriguing. However, it was clear that at this stage of the design process that the paddle wheel system would work best.

With these ideas in mind the propulsion system had to be implemented based on cost and effectiveness. The paddle wheel was the optimum choice because it could be constructed for less than $400, and had the dynamic of speed and power the vessel would need to travel in the marsh environment.

The next task was to place them accordingly so that weight distribution and control capability were maximized. The first idea was to use two paddle wheels on the sides of the craft as shown in Figure 7. This was dismissed after realizing turning in thick grass might become an impossible.

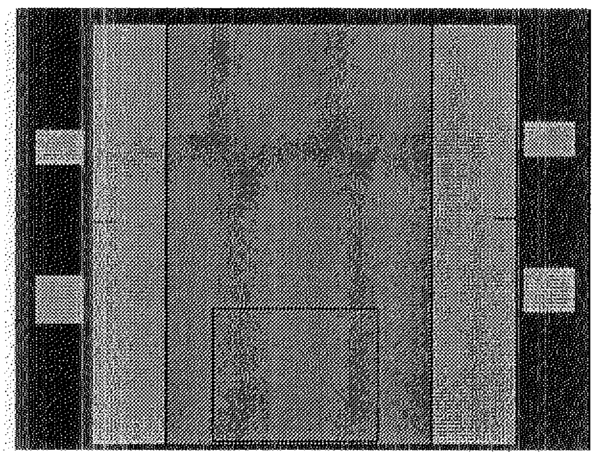


Figure 7: Side paddle wheel design concept

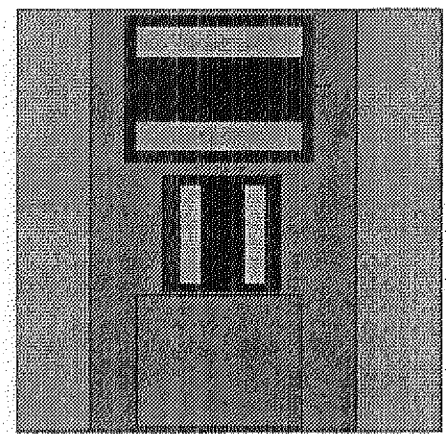


Figure 8: Center paddle wheel design concept

The next placement idea was to use one paddle wheel in the middle to go side ways and one towards one end to drive forward and reverse. This idea was also dismissed because of the additional drag of trying to push the large pontoons sideways through the marsh grass with such a small paddle wheel in the center. This idea is shown in Figure 8 and as is clearly visible there is very little room present on the deck for control devices, tanks, batteries, and other instrumentation.

Then came the idea of using one main driving wheel that would be suspended from the rear of the craft. For side-to-side motion, the craft would utilize two smaller paddle wheels located on the sides of the craft. Now there are two different possibilities for placement of the two side wheels; the first possibility is having one on either side at opposite corners as shown in Figure 9. This idea was dismissed because the team believes that the craft will still have minimal maneuverability. This led to the idea of using the two paddle wheels on one side of the craft, as well as a major driving wheel on the stern of the craft. This has been determined to give the craft optimum maneuverability within the marsh environment, and allow us to run a simply plotted square grid of waypoints once in the marsh. Once the placement of the paddle wheels was determined, the issue of what to turn them with became the next major concern.

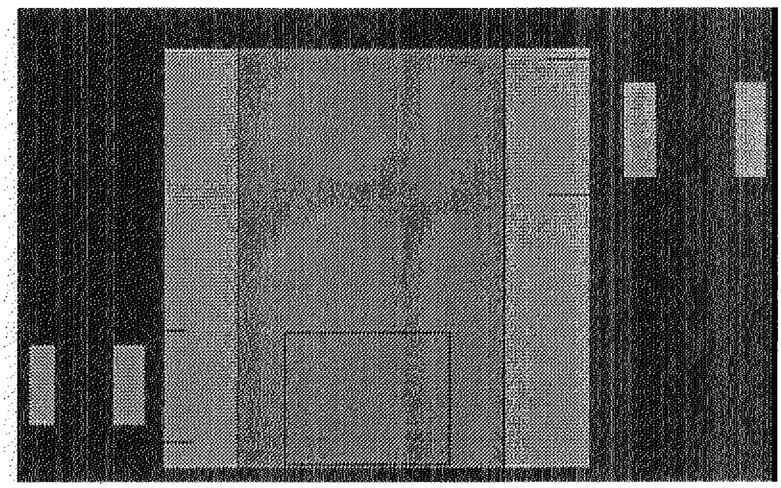


Figure 9: Side paddle wheel design concept

Final Paddle Wheel Design

The final paddle wheels used in the vessel are made out of 20-gauge aluminum for both weight and durability. All the wheels have the same 36-inch diameter disks for the sides and the larger stern wheel has 24-inch long paddles with two-inch flaps at the ends to mount them to the disks. The smaller side units have 14-inch long paddles with two-inch flaps at the end for mounting. The larger paddles are 6 inches wide while the smaller ones are 4 inches wide.

The paddle wheels were assembled using zinc nuts, bolts, and washers so they could be dismantled or changed very easily during prototyping. After testing the prototype system the wheels should be welded for added strength. Threaded rod was used as an axle for the paddle wheels, which made it easy to mount the gears. The threaded rod was also essential to the bushings that were used on the arms. Figures 10 and 11 show the paddle wheel assemblies and size variations.

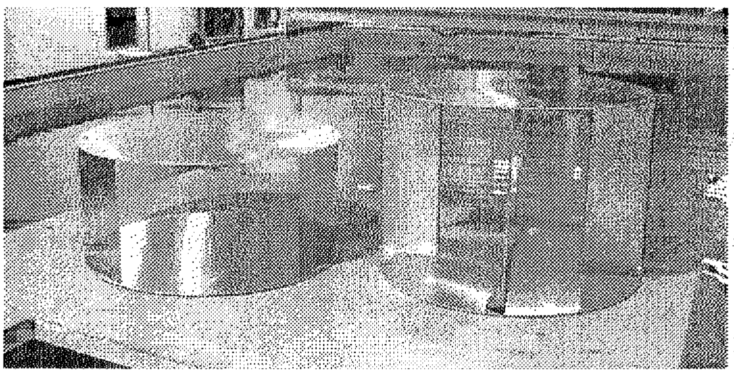


Figure 10: Paddle Wheel Sizes

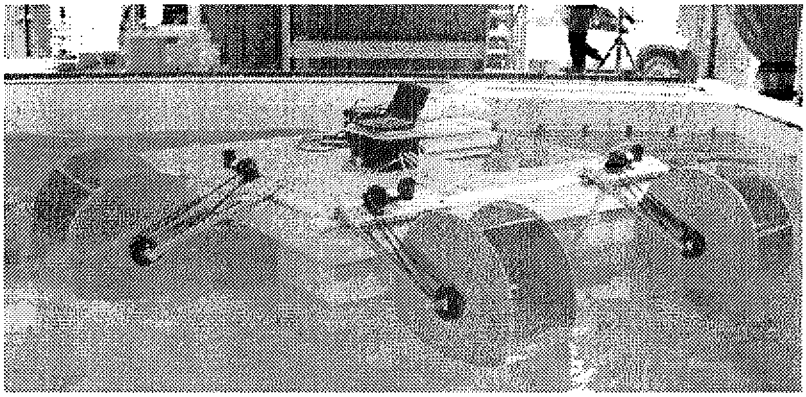


Figure 11: Final Paddle Wheel Setup (Note the arms holding the wheels away from the vessel can pivot upward)

Structural Analysis of Paddle Wheels

To ensure that the paddle wheel design would be able to withstand the propulsion forces, the structure was modeled using an FEA program for worst-case scenarios. The worst-case loading on the paddle will occur when it strikes a rock, or other hard object, and rotates itself over the object causing the paddle wheel assembly to lift on the two pivot arms and cause all of the paddle weight to rest on the center of one paddle. MENTAT, a finite elemental analysis program, was used to determine stresses and deflections, which will allow analysis of the structural integrity under these conditions. The analysis was done for two different loading cases and two end support scenarios: one, simply supported with a point load and then again with a distributed load; and two, built-in with a point and distributed load as well. The paddle was modeled using plate/shell elements to account for the large ratio of surface area to material thickness.

The first type of supporting that was used to model the paddles was a plate with built-in ends. This means that both ends of the paddle will have zero deflection and zero slope at the built-in ends. This analysis gave an underestimate of the actual deflection for both cases because the paddles are bolted to the disks and therefore can have some slope at each of the supporting ends as this is clearly visible when loaded. Figure 12 shows the stresses and the deflections for a point load with built in ends.

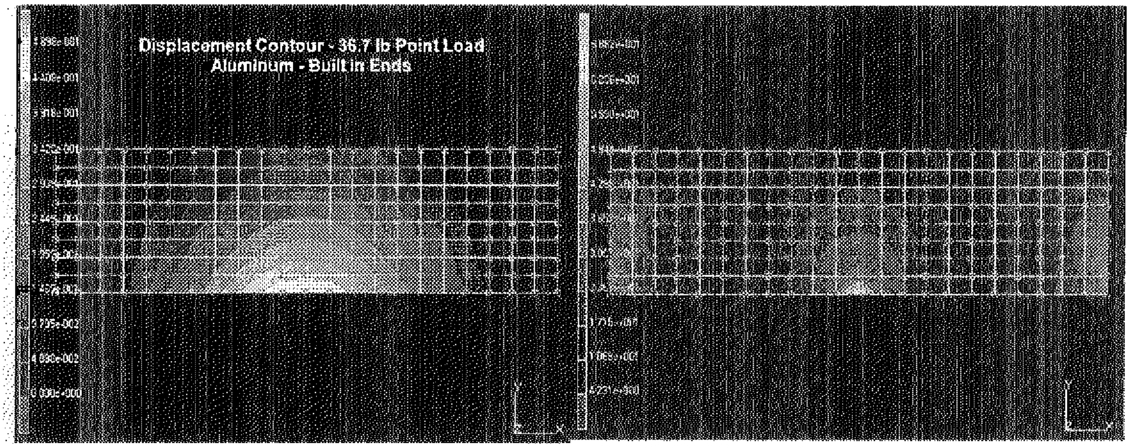


Figure 12: Displacement contours (left) and stress contours (right) for the Aluminum Plate model using built-in ends for support.

The second type of support that was used to model the paddle wheel was a simply supported case. This was expected to overestimate the deflections and stresses present in the paddle, as provide in Table 3. Also provided on Table 3 are the values of all the stresses and deflections for both supporting cases and materials.

Load Case	Load (lb)	Max Def (in)
Steel Plate Point Load – Simple Support	100	1.38
Steel Plate Point Load – Built In	100	0.44
Steel Plate Distributed Load - Built In	100	0.4
Al Plate Point Load – Simple Support	36.7	1.52
Al Plate Point Load – Built In	36.7	0.489
Al Plate Distributed Load - Built In	36.7	0.441

Table 3: FEA results for all load/support cases

MENTAT allows for easy changing of the material properties within the model, this led to looking at alternative materials to aluminum such as plastics. Using a plastic paddle wheel was looked into and decided against because in order to get the deflection down to a safe amount the paddles needed to be substantially thick and would weigh more than the aluminum model. The other disadvantage to using plastic would be the sun degradation that might occur when out in the marsh, it could alter the physical characteristics enough to cause failure.

The results of the testing indicate that the aluminum would be the ideal choice for the application and it would easily support the worst case loading. The thought of using steel had come to mind but was dismissed because of weight issues. The wheels could be made lighter by reducing some of the material in the large disks by punching holes in them without much structural degradation.

Motor Selection

Since the propulsion motors are a primary component of The Clean Cat, some special considerations have to be addressed including costs and the affect of the harsh environment. Electric motors designed for use in a marine environment are usually expensive. Due to budget constraints the motors currently implemented are not ideal for use in the marsh, but work well for small-scale prototyping and demonstration. The same type of motors can be purchased as marine grade units, but at a much greater cost.

The next major issue to be addressed was the output shaft speed of the motor. The vessel is designed for very short travel intervals and with the large paddle wheels. Therefore, the motors would have to have a slow output shaft speed (ideally 64 RPM). This allows them to be geared down to drive the paddle wheels at approximately 20 RPM. This is still a little fast for the application and a larger gear ratio from the motor to the paddle wheel or motor speed control may have to be used in the future to slow it down.

The chosen motors were ordered from Bison Gear & Engineering Corp. The company was able to provide 24 VDC motors with 1/8 HP at a discounted educational price of $175.00 apiece. The motors use an internal gearbox with a reduction of 28:1 to provide 126 lb-in of output shaft torque. As previously discussed, the motors were chosen for their high torque ratings at low speeds to turn the paddles through very thick marsh grass (See Figure 13 for close-up view of motor specifications).

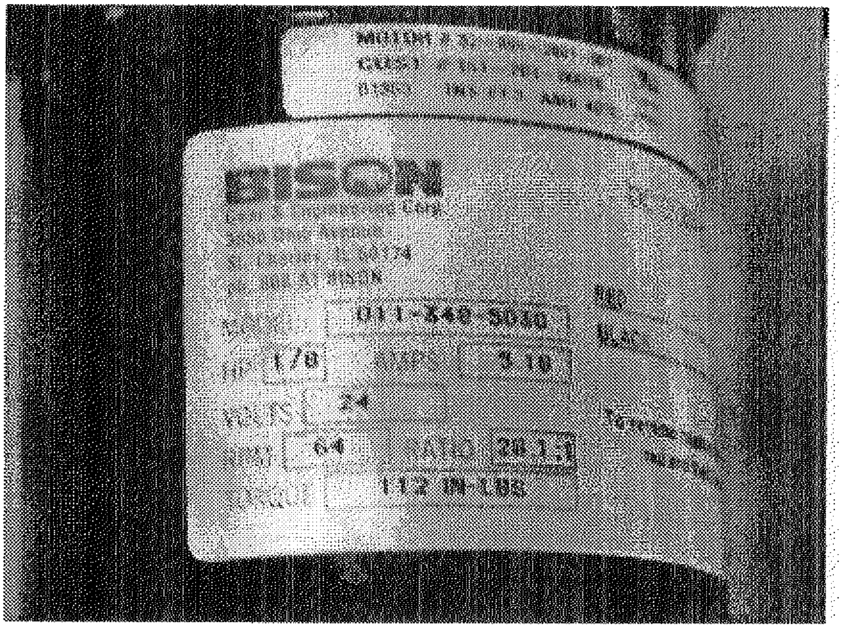


Figure 13: Close-up view of motor specifications

Flotation: Design and Analysis

The vessel's buoyancy depends on the pontoon's flotation capabilities to offset the weight of the assembly. The vessel's target design weight was 800-pounds, this value was achieved using the estimated weight of the injection system, propulsion system, power sources, and any nutrient storage tanks. The propulsion system's weight derives itself mostly from the paddle wheels, which are the means of moving through the marsh. Additional weight is added with the three drive motors that turn the paddle wheels, along with the required mounting hardware.

The pontoons must be able to withstand water penetration and possible collision with foreign objects such as rocks and driftwood. To remain afloat, the integrity of the pontoons must not be compromised in such instances. Closed-cell polyurethane foam, the type used in the construction of the pontoons, is semi-impenetrable to water, so over time the foam will absorb a small quantity of water. To avoid this, a hard membrane should be applied to the pontoon. The pontoons must support the weight of the boat with minimal draft, survive in the marsh environment, and cost less than $300. The marsh environment can be a harsh one for a small vessel such as this one. Much debris is scattered throughout the marsh, including driftwood and other foreign objects such as a safe, which had been discarded in the bottom of the marsh. Numerous designs of the pontoon have been considered, as well as multiple shapes to achieve a draft of 6 inches or less and good hydrodynamic characteristic.

Another option considered would be to replace the initial fiberglass design with sheet metal. The sheet metal will give the watertight seal that the fiberglass did as well as some added protection. The same foam core design will be used and a coating of 24 gage aluminum. For lengthy use, a non-corrosive metal capable of surviving in the salt water for long periods of time would be needed. Since cost was also a factor in the construction of the pontoons, closed-cell buoyancy billets, used for constructing docks were used. The billets were sixty dollars per 7-inch x 20-inch x 9-foot section. If the hard coating is applied the seams will have to be sealed in a durable manner that is impenetrable to the water. Two possibilities are an epoxy or, depending on the material used, a small weld on the exterior coating.

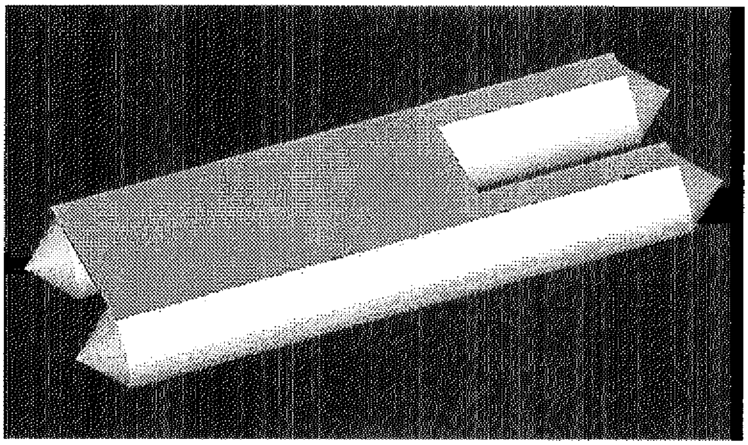


Figure 14: Preliminary design utilized shaped 9-foot buoyancy billets

Many types of pontoon shapes have been considered (i.e. cylindrical, trapezoidal, square, oval, and rectangular). The weight that the boat is able to hold is proportional to the weight of the volume of water displaced by the pontoon. So to achieve the shallow draft required, the design must have a broad base. The initial design consideration was to use cylindrical pontoons such as those on many recreational pontoon boats. This design, shown in Figure 14, has improved hydrodynamic properties as opposed to less sleek designs. More importantly, a cylindrical pontoon has a large draft. This is because the shallowest draft is achieved when a pontoon with a large bottom surface area is used.

The ideal shape turned out to be rectangular, which happens to be the shape of the stock foam when ordered from the supplier.

Positioning of the pontoons is dependant on the number of pontoons it will take to float the system. Calculations were performed to determine the amount of buoyancy needed at a given draft. These results are shown in Table 4, while possible positioning configurations are shown on Figure 15a and 15b. The objective is to maximize the bottom surface area of the pontoon, because the foam out of the water is not contributing to the buoyancy of a vessel, because it is the actual submerged foam which is displacing the water that gives a boat flotation.

Table 4: Buoyant force calculated to determine the desired draft for the specified load

	FOUR FOAM BILLETS		SIX FOAM BILLETS	
Draft	Volume of Foam	Buoyant Force	Volume of Foam	Buoyant Force
(in)	(in^3)	(lbf)	(in^3)	(lbf)
3	9072	335.66	13608	503.50
4	12096	447.55	18144	671.33
5	15120	559.44	22680	839.16
6	18144	671.33	27216	1006.99
7	21168	783.22	31752	1174.82
8	24192	895.10	36288	1342.66
9	27216	1006.99	40824	1510.49
10	30240	1118.88	45360	1678.32

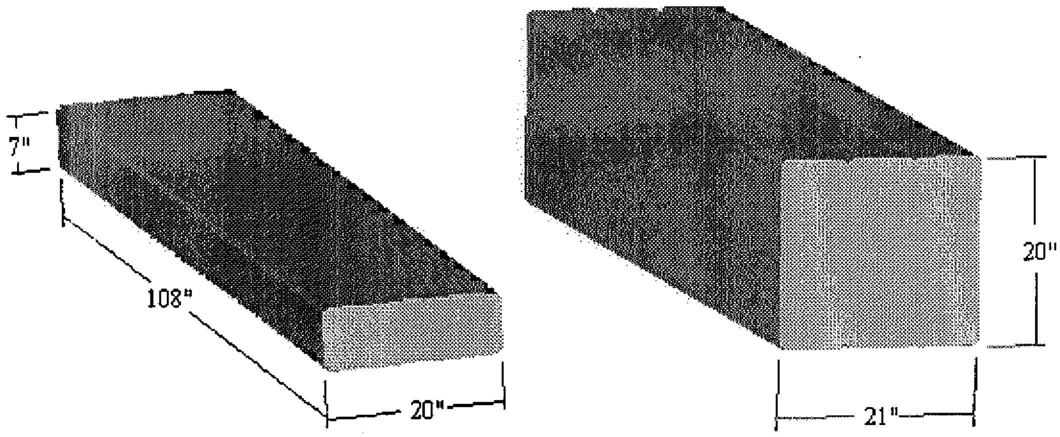


Figure 15: (a) Single foam billet showing major dimensions, (b) final pontoon size, three billets per side. The closed-cell foam has a density of 2 lb/ft³.

The boat was first assembled with two pontoons, made up of two buoyancy billets each. When propulsion testing was performed this proved to be insufficient flotation, because the desired draft was exceeded. This was concluded after weight was added to see how it would perform. The next step was to add a third buoyancy billet to each side. Adding the third billet on each side increased the flotation by 335 pounds, resulting in a six-inch draft (Table 4).

Once the flotation requirements were finalized, the construction of the vessel began using lumber, foam, and hardware. The deck size considered placement of all of the supplies and apparatus that were to be placed onboard. The resulting deck is 10-inch x 8-inch, 3/8-inch plywood with 2-inch x 6-inch and 2-inch x 4-inch supports. The deck design is shown in Figure 16 and the assembled deck is shown in Figure 17, notice the 3-foot x 3-foot cutout in the stern for the injection system.

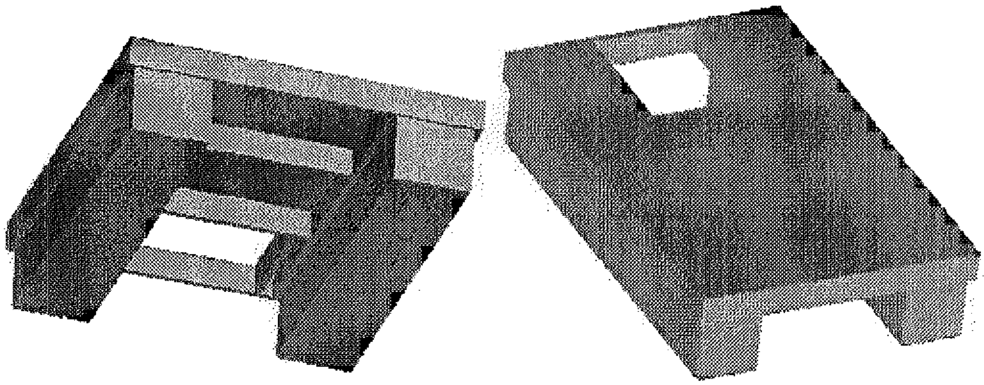


Figure 16: Schematic of the boat deck and pontoon vessel

The intent of this vessel is to test the low draft pontoon concept with the paddle wheel and control system prior to building an expensive prototype. If the concept testing shows promise future design iterations should considering changing the following aspects of the design. Most of the hardware is galvanized, which is rust resistant, but will corrode in the high humidity environment. Ideally, all hardware should be stainless steel. The lumber used for the deck and the frame would be replaced with either weather resistant wood, such as pressure treated lumber. The other option being replacement with either aluminum or another metal that would hold its integrity in the salt marsh environment. These changes would significantly increase the cost of the vessel, but would provide the needed protection from the elements.

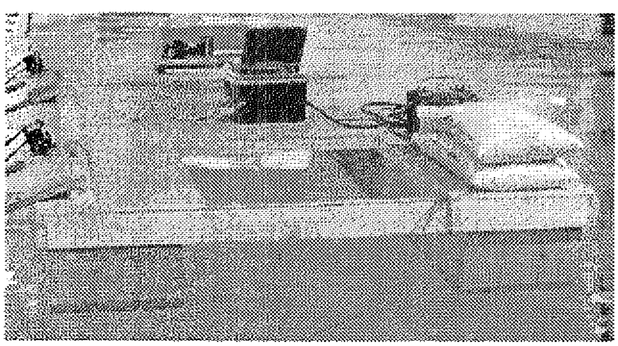


Figure 17: Assembled deck

To streamline the boat, allowing it to push marsh grass aside, nose cones are placed on the ends of each pontoon. The cones are made of scrap foam, as used for the construction of the pontoons. A simple wedge design provides acceptable performance and handling. Prior to applying the end cones, it is believed that much of the water drag was being created as a result of the blunt ends of the pontoons, these ends would have made it impossible to slide through the grass. The design concept used for the cones is shown on Figure 18.

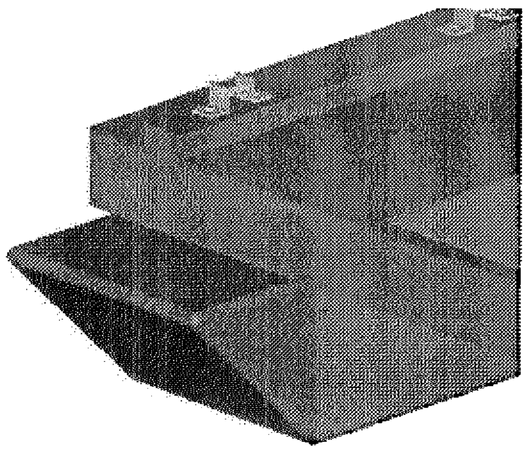


Figure 18: Nose cone

Another feature of the boat that is disrupting the maneuverability is the drag from the side paddle wheels. It was initially thought that by means of a winch system, the side paddle wheels could be able to be raised and lowered depending on the motion of the boat. Due to budget constraints, however this concept was not utilized in the presented design, but is recommended for future iterations of the project. Shown below in Figure 19 is a system that would make it possible to raise and lower the side paddles with ease. It works by moving the paddle wheel subsystem about its' axis of rotation or its' pivot, which is later shown in Figure 21b. This subsystem could be added to the design at any time, as it has no effect on the current vessel.

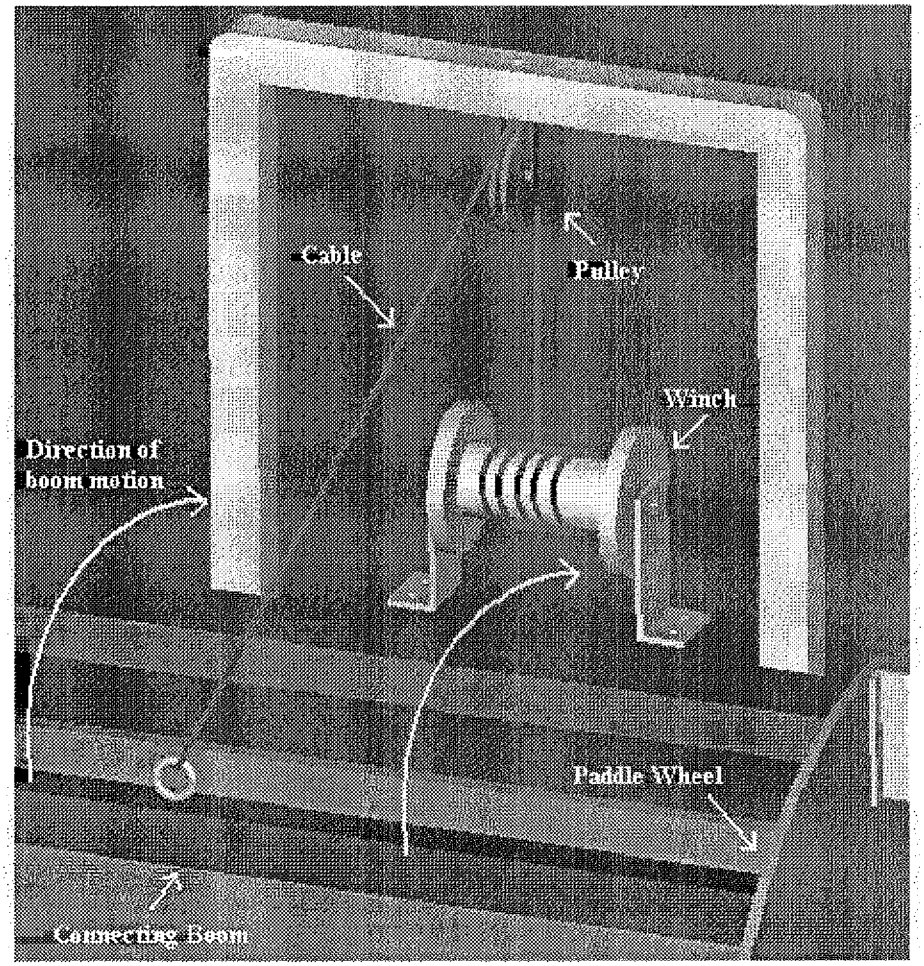


Figure 19: Winch system design recommendation for future implementation

Each paddle is equipped with two, three-foot arms, which allows the paddles to be raised. The arms are constructed out of Unistrut, which consist of U-shaped 2 inch x 2 inch steel tubing shown in Figure 20. The Unistrut is connected to the axle of the paddle wheel using sleeve bearings, and also to the onboard joint. Bearings are necessary to ease the rotation of the paddle wheel and to allow for the paddle wheels to be raised and lowered. The bearings are made out of steel conduit and cut to size for each of the joints.

The two rotating joints are shown in Figure 21(a) and the sleeve bearing is shown in 21(b).

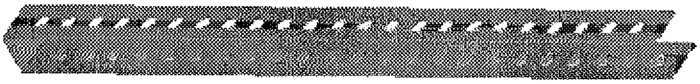
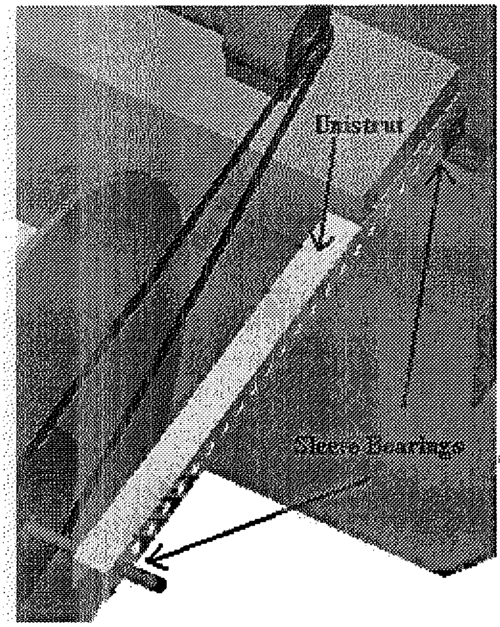
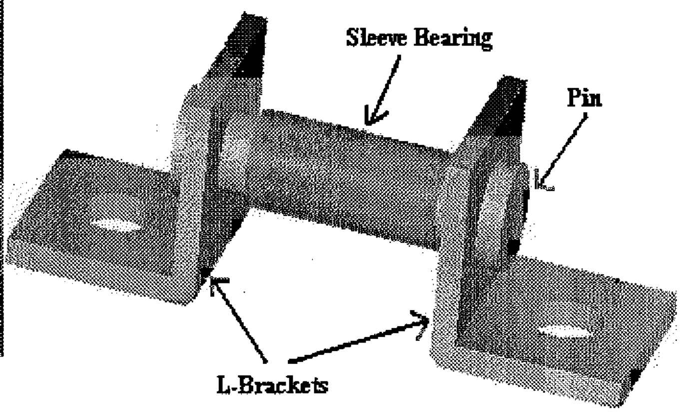


Figure 20: Unistrut swing arms, 3 feet in length





(a) *(b)*

Figure 21: (a) Swing arm joints, (b) paddle wheel pivot point

Control System

Overview:

To properly design a complete control system the designer must first understand how every aspect of the machine works and what the end user or client wants the automated machine to accomplish. To start designing the control system for 'The Clean Cat' biovessel this initial step had to be completed.

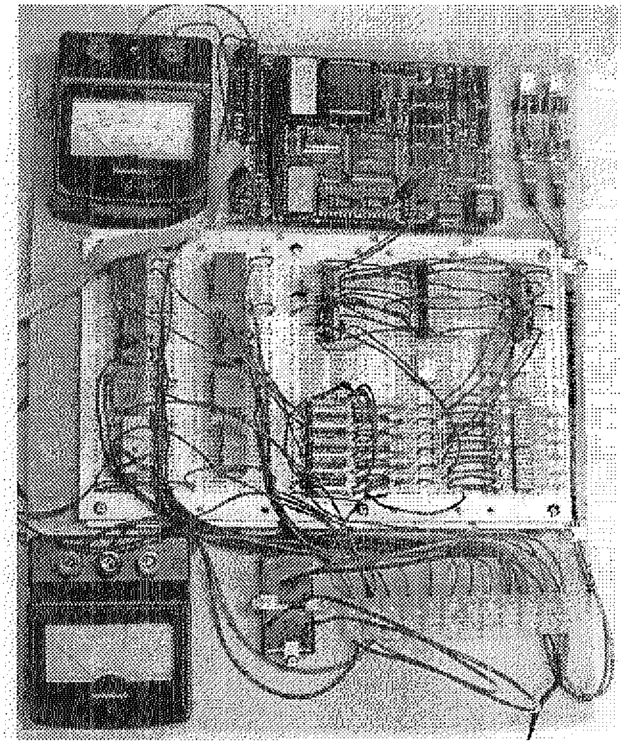


Figure 22: Complete single board computer control system without wireless interface.

First, a meeting was setup with Professors Nancy Kinner and Tom Ballestero to get insight into what the vessel was to be designed to do and bounce initial ideas back and forth about how to jump into designing such a system. In the initial proposal it was suggested to build a fully automated vessel that will roam about the marsh during the three hours surrounding high tide and perform its injection process. This proposal brought about some serious questions as to what was meant by 'fully automated.' While discussing this question it was suggested that this design be aimed towards being an autonomous vehicle requiring no human interaction to carry out its duty. With less than two-thousand dollars worth of project funds available this task was redefined as a concept test and much of the 'production' would have to be made theoretically.

Secondly a trip to Portland, ME was taken to acquire knowledge about how the geographic and physical dimensioning of a typical salt marsh. This was critical in the sense that it erased all of the preconceived notions of salt marshes (i.e. being 'clam-flat-like' with not much bottom irregularity and not having many obstructions) and gave a strong foundation in the idea of the complexity of designing an automated vessel. This complexity is not only attributed to the fact that the vessel will be moving around an open water environment that could possibly harbor strong currents and wind to blow it off course but also the sheer size of the marsh threw a large variable into the design equation.

After seeing the marsh first hand, the problem became apparent yet the solution was more distorted. Initially whenever an engineer is given a proposal usually he or she has an initial 'gut' feeling as to how to tackle the problem, as the project is investigated further this solution has to be catered to any unforeseen complications that arise. During this project, this step was avoided in that the solution developed has always been open ended and constructed such that things could be changed with little effort if need be. This is important when building a prototype of any apparatus since the engineer should not follow one path (brute force approach) and in the end have no way to change any failing aspect of the system. In the following pages this open-ended design process is detailed and will hopefully provide great reference to anyone trying to develop the system further in the future along with giving anyone unfamiliar with the system detailed information into the development insight.

Positioning:

The first part of creating any autonomous machine is trying to develop the part of the control system that tells the machine where it is in relation to where it is supposed to be and what direction it is facing. To find a system that would allow the boat to maneuver around the marsh without umbilical control led to a few options.

The first option explored involved using the global positioning system (GPS) to tell the boat its actual coordinates that would allow the boat to 'decide' which direction it has to move. The GPS, consists of a network of satellites orbiting the earth, constantly emitting a precisely timed code, which a receiver moving around the earth's surface can detect [6]. The receiver, in turn, takes the code (from at least two satellites) to calculate its position by trilateration techniques based on time of flight for the codes to reach it. Standard GPS tends to have accuracies around 15 meters [6], this is due to the constantly changing position of the satellites and the refraction of the spread spectrum code by the atmosphere [7]. To increase this accuracy a scheme was devised to place two GPS units in the area of interest, one stationary and one mobile. The stationary unit will give indication that it is moving due to the inherent error and this error can be calculated by simple subtraction. The mobile unit moving in the nearby area will also detect this error but the clever observer standing at the stationary unit can relay the error to the mobile unit and the mobile observer can subtract the error to give itself a more precise sense of location. This method of obtaining position is incorporated into 'differential' global positioning systems of DGPS', though instead of having an immobile unit the correction signals are sent out either by local Coast Guard stations or if the unit is being used inland several private companies offer their own correction signals on a subscription basis.

Theoretically this should give close to zero tolerances, but the GPS receivers often times can't measure distance changes less than three meters (especially the less expensive ones that this project's budget would allow, high precision, less than one meter accuracy, units are available from $30K to $50K), they tend to 'jump' three meters at a time when moving [6].

The UNH Ocean Engineering department had one such unit sitting in the electronics lab, after several days of trying to piece it back together from years of non-use a simple stationary test[1] was performed to obtain that its accuracy was approximately three meters. It was concluded from this result that the system would be inaccurate for positioning the vessel for several reasons. The most important reason being that if the vessel was to be positioned in one place and the DGPS received a signal that it was to be three meters away from its current location when in reality it was in the correct place, the injection would not occur where it was supposed to be. Also in terms of creating an algorithm to control the placement of the vessel the error for compensation for such a placement would have to allow for this 'slack' or 'play' or else the system would continuously try to compensate for the error and reposition itself. Therefore, the error would be multiplied to maintain a steady-state operating point. Three meters is a large error alone since the injection system being installed on the boat is one meter square, the placement of the one-meter square three meters from where it is supposed to be isn't unacceptable but when introducing the algorithmic error the value can easily become unwieldy.

[1] The stationary test consisted of having the unit sit immobile and reading the output of the unit to see the magnitude of variation. The results varied by three meters.

The other methods of positioning looked at were all in a sense small GPS systems. The first one was using a radar system mounted on the vessel and have several reflectors mounted on poles throughout the marsh. This is very similar to the method fishermen employ to keep track of where their nets are in the water as they are being trawled. Inexpensive reflectors are available at most marine stockyards and radar systems can be obtained from various manufacturers (i.e. Raytheon, Garmin etc.). The disadvantage of using this method is that the unit is emitting radiation and sitting as close to the water as it is to be placed it is unclear if this will have any impacting effects on the already fragile ecosystem surrounding the vessel.

The positioning method that appears most promising uses light instead of radiation to find position. The vessel will hold a unit that has a rotating, flat, eye-safe laser beam. Two beacons will be set up in the marsh to reflect this rotating beam and allow the unit to calculate its position (Figure 23). The system to be implemented on the vessel varies from this caption in that the PC used for position control will be part of the mobile laser STROAB unit, and the stationary reflectors or NOADS will be linked wirelessly [8]. MTI Research from Westford, MA has been a pioneer in the design of laser positioning systems that have a great deal of accuracy [6]. The system has sub-centimeter accuracy and position refresh rates up to 40 Hz. The

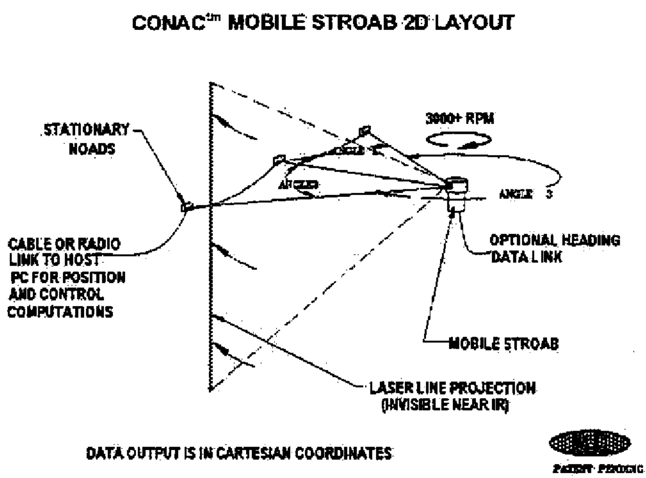


Figure 23: MTIR CONAC laser positioning system proposed for marsh use. The mobile STROAB will be mounted to the vessel and the stationary NOADS will communicate wirelessly (i.e. cabling not necessary).

other major advantage of using this system over DGPS or RADAR is that the laser guidance also tells the heading of the vessel whereas if DGPS was implemented an external electronic compass or gyroscope would be necessary for the heading detection. MTIR produces the CONAC system (Computerized Opto-electronic Navigation and Control) that has been used extensively throughout the world by companies and organizations such as Lockheed Martin, US DOT, Walt Disney Imagineering Group, and Boeing Aerospace. Ed MacLeod, President of MTIR came to UNH to give a presentation after learning about the interest this control project had in obtaining one of their systems. The presentation was very informative and he has been a constant source for detailed information as to how their system would integrate into the control system being designed for the vessel. MTIR submitted a quote of $16,480 for a laser positioning system that would adequately handle the marsh environment. This exceeds the funds available to this project by far, so it was decided to delay purchase of the system until the vessel and propulsion system has been thoroughly tested.

Automation – Computer Control:

The key ingredient that makes the system automated is its controller. There are thousands of different control systems available that can do a large variety of different tasks. Almost all systems have to be modified or added onto in order for it to work for a specific application.

Initially, before receiving the written proposal for the project it was thought that a simple programmable logic controller or PLC (Figure 24) could take care of the control processing aspects of the system. This was based on the understanding that a platform was going to float in the marsh and at a certain tide cycle start the injection process. This is a simple task for a PLC to undertake. After learning that the vessel was to be propelled around the marsh and be autonomous the PLC concept began to fade. PLCs are great for repeated tasks, motor control, mathematics, but lack a lot of processing power in terms of being able to accept RS-232 (serial) data, as would come from the CONAC laser guidance system, and analyze it for control purposes [9]. The other downfall to using a PLC is that they tend to consume more power than other processing units available on the market. Since, it has not been determined how the craft's deep cell batteries will be replenished (whether by solar power, a small gas generator or some other means) it is important to keep power consumption to a minimum. It would be ideal to power the boat with solar power, therefore creating a more environmentally friendly craft, but without having the final power requirements of the actual motors and pumps used to drive the system this is impossible to suggest. The amount of energy consumption, though, can be kept to a minimum.

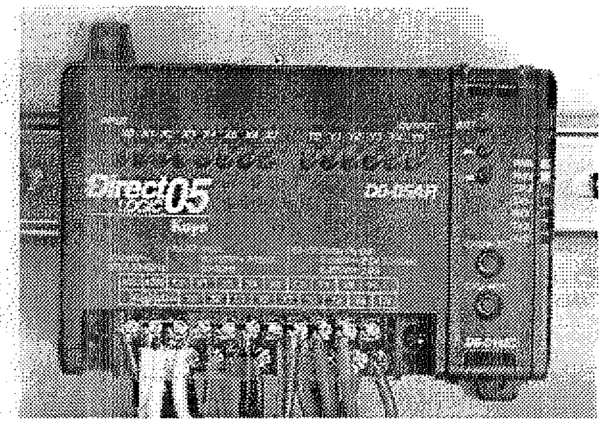


Figure 24: Automation Direct's Direct Logic 05 PLC, provides ease of programmability, yet does not have enough ability to interface with other onboard systems or allow for wireless control.

After obtaining the written project proposal and learning that the vessel was to be automated the first idea that came to mind was creating a wireless link between a base controlling station and the vessel's controller. This would allow an operator to override the automated control system on the vessel in order to drive the boat to where it can easily be worked on. Several single board computer (SBC) manufacturer's on the market advertise boards with ethernet ports on them, this port would allow the board to be connected into a wireless network. After reviewing quite a few of these board's specifications it was decided to purchase an Intel 386EX SBC from Technologic Systems in Fountain Hills, AZ. The TS-2800 SBC (Figure 25) includes a DOS-ROM based operating system with full TCP/IP support, 2 PC/AT RS232 serial ports, 8 Mbytes of RAM, 1 Mbyte of flash, and 24 I/O (input/output) points along with a host of other features.

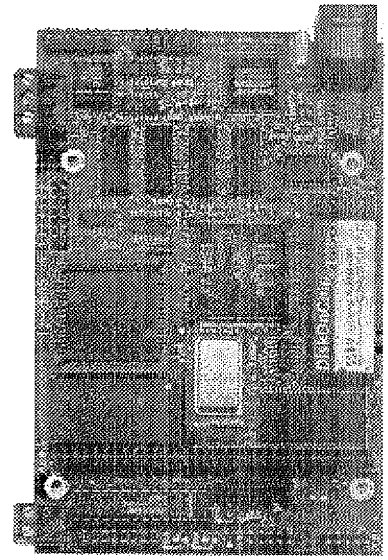


Figure 25: Technologic Systems TS-2800, Intel 386 EX based SBC, used to control The Clean Cat wirelessly.

One other very important feature that was looked for while finding the controller was a battery backed real time clock. This feature is critical since the boat will have to perform its operation during the high tide cycle, therefore the controller will need to be programmed with a function to determine this time. The TS-2800 includes the master of all time clocks: a Dallas Semiconductor DS12887. This is a completely self-contained module that includes the Motorola 146818 compatible clock chip, the lithium battery, and 114 bytes of battery-backed CMOS RAM. This clock is guaranteed to maintain time for a minimum of ten years in the absence of power. Even though the vessel's control system will never be in the absence of power for more than ten months (over the winter), it is nice to know it has the capabilities to be stored for a few years with time retention.

The price on the TS-2800 SBC was originally $299, but was discounted to $228.50 since one of the DIO (digital input/output) points failed during the factory test.

There was a verbal agreement made with Technologic Systems that if the board did not function when implemented into the control system they would exchange it for another at the standard price.

As stated earlier the TS-2800 runs a DOS-ROM based operating system, this is not identical to Microsoft DOS, but very similar. All of the external components within the vessel that the computer controls are run by the four 8-bit I/O ports. These ports are turned on and off by writing a combination of 1's and 0's to memory locations on the board that correspond to the locations of the IO port [10]. This can be accomplished manually by a user connected directly to the board through serial console redirection typing assembly code into the board (very tedious, but it was done to figure out how the board worked!) or by writing the code into files and compiling it for execution through the wireless interface (still the user has to invoke the compiled code). Programming the board can be done in a variety of languages such as BASIC, FORTRAN, C, C++, Assembly Language or many others. To write the programs to turn on and off the DIO ports the programs were written in Assembly Language, compiled to DOS COM files (using Borland's Turbo Assembler V2.01), then invoked by either DOS batch files or C program files compiled to DOS executables (EXEs). Sometimes the Assembly Language files were invoked by a combination of C programs and batch files to simplify the operation (not the programming!). A simple example of an assembly language file that will turn on the rear paddle wheel motor in the forward direction is as follows:

```
%TITLE 'OOOI'
; Created by Stanley Boduch 3/11/02
; This program turns on P3.0
    IDEAL
    MODEL TINY
    CODESEG
    ORG 100h
Start:
```

```
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110001b   ; change the register so LB is 0001
        out     dx, al          ; write the register back out to 0F872

    Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program
        END Start
```

This program, though it looks confusing, performs a very simple function. Where the comment says 'change the register so LB is 0001, this is the important statement to recognize in the program (everything else is necessary for the program to function, but this line is where all the action takes place). Essentially the program opens Port 3 (one of the 8 bit IO ports) and turns on pin P3.0 (this is Pin 1 on Port 3). This program could have just as easily turned on P3.1 (Pin 2) by writing a 0010, or pins P3.0, P3.2, and P3.3 by writing a 1101 to the port. Once the program is compiled to a DOS COM file all you have to do to is type the name of the file (ex. name.com) and the program is executed. This same operation can be accomplished from within a C program, but is a bit more complex in coding. The complete code, including the program to invoke the COM files can be found in Appendix B.

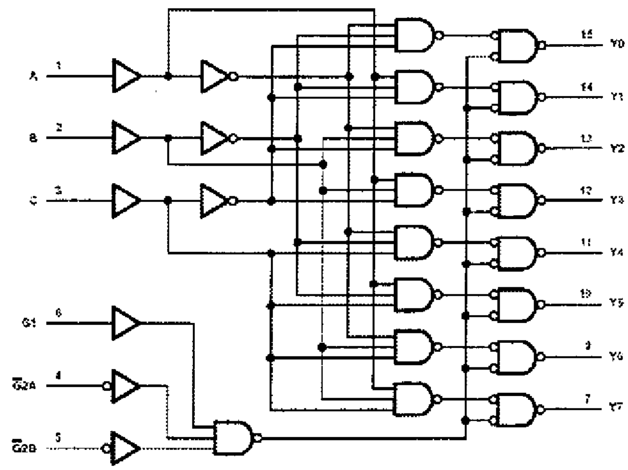
At this point it is important to point out what the 1's and 0's mean. When a 1 is written out to a port pin the pin is turned on, when a 0 is written it is turned off. This being 'turned on' and 'turned off' means different things depending on how that port pin is configured. All of the port pins used for the control of the vessels auxiliary equipment can be configured either as high impedance inputs, open-drain outputs or complementary outputs. For this system the output pins are configured as complementary outputs. The advantage of configuring them as complementary outputs rather than open-drain outputs is that it eliminates the worry of sending too much current into the pins, risking damage to the board. Each pin allows a current of 25 mA to be sourced from the pin, so the pins can be overloaded even when configured as complementary outputs but the risk of high-level board damage is much less. As complimentary outputs, the pins are configured such that when turned on they provide +5 volts and when off, the voltage is turned off. The 1's and 0's come from the binary numbering system the following list of numbers is an example of how to count from zero to seven in binary along with the decimal equivalent.

$(0000)_{binary} = (0)_{decimal}$	$(0100)_{binary} = (4)_{decimal}$
$(0001)_{binary} = (1)_{decimal}$	$(0101)_{binary} = (5)_{decimal}$
$(0010)_{binary} = (2)_{decimal}$	$(0110)_{binary} = (6)_{decimal}$
$(0011)_{binary} = (3)_{decimal}$	$(0111)_{binary} = (7)_{decimal}$

This list of numbers was very important during system development since the method of controlling each of the motors on and off was created based on the simple method of counting. First, the external peripherals connected to the TS-2800 have to be described before this method can be discussed.

Each paddle wheel motor is driven by two control lines; one line turns on the power to the motor, the second line reverses the motor. All three drive motors, running both in forward and reverse, are controlled by three pins of the TS-2800 (Port 3 – 3.0, 3.1, 3.2). To take a set of binary numbers and turn them into six different control lines (one line for forward, one for reverse for three motors) the three port pins must first be passed through a decoder (Texas Instruments part number SN74HCT138). Essentially, a decoder consists of NANDS and NEGATES which accept a certain number of data line inputs (in this case three, but two input decoders are common) and converts it to a decimal number on the output side [12], a schematic and truth table detail the innards of the decoder and the output for any given input is shown on the next page.

The data lines are labeled A-B-C, as the lines are toggled from low to high (0 to 1) using the binary counting method, the outputs toggle with the corresponding decimal value. The only problem is that the outputs are in the opposite state than what is useful for controlling a single motor line since if a decimal 5 is written to Port 3 all of the output pins except for pin five will be powered. Therefore, the output from the decoder has to be put through an inverter to change the pin status from either a high to low or low to high (Texas Instruments part number CD74HCT04E). After putting the data lines through the inverter it is possible to read the lines as decimal equivalents of the binary number entered on lines A-B-C of the decoder.



FUNCTION TABLE

INPUTS						OUTPUTS							
ENABLE			SELECT										
G1	$\overline{G2A}$	$\overline{G2B}$	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	L	H	H	H	H	H	L	H	H	H	H
H	L	L	H	L	L	H	H	H	H	L	H	H	H
H	L	L	H	L	H	H	H	H	H	H	L	H	H
H	L	L	H	H	L	H	H	H	H	H	H	L	H
H	L	L	H	H	H	H	H	H	H	H	H	H	L

Figure 26: Decoder schematic and function table, alternatively a latching decoder could have been implemented but would have required additional port control pins.

At this point there is another block stopping the controller from driving the motor system: only one number at a time can be outputted from the decoder. This might not sound like a big dilemma, but it doesn't allow the computer to drive more than one motor at a time or even allow any of the motors to be driven in reverse. A latch must be introduced to store the status of the control lines while new data is entered on Port 3. To do this latches were constructed out of Motorola MC14001BCP quad-two-input NOR

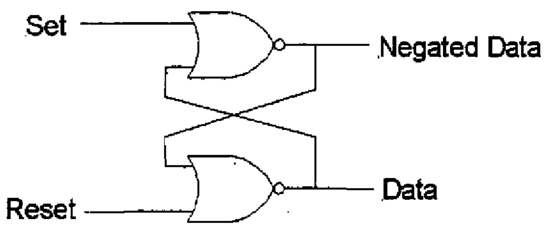


Figure 27: Simple method of creating a custom data latch out of two NOR gates.

integrated circuits (ICs). The schematics in Figure 27 and 28 detail how this was accomplished along with the method implemented to wire the IC [12]. When the 'set' line goes high the data line turns from low to high and remain high until the 'reset' line goes high and the data resets to zero. Each motor requires one MC14001BCP for it to be driven both forward and reverse, so there are three of these ICs on the control system. The resets are activated by writing a 0111 to Port 3 after this passes through the decoder and inverter it corresponds to control line 7. Control line seven is tied to all of the MC14001BCP 'resets' within the controller. When the 0111 is written to the system all motors are stopped and the off status of the latch ICs is set. When any motor is written to (by writing any number from one to six) that motor is turned on the motor remains on until the a 0111 is written, therefore more than one motor can be turned on by writing the corresponding number to it. The following figure (Figure 29) is a schematic detailing the wiring method used, from Port 3 of the TS-2800 through to the outputs of the latches. See the complete control system schematic in the Appendix on page A-2.

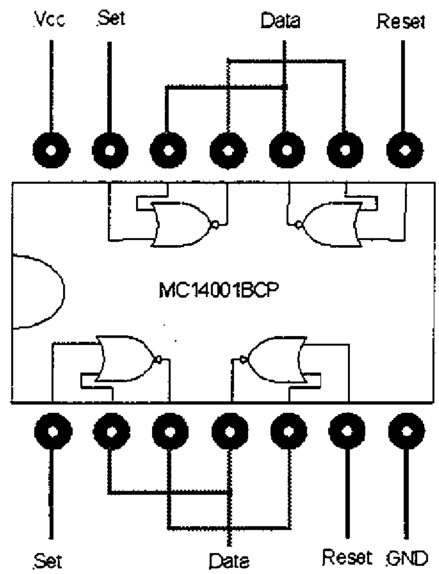


Figure 28: Wiring schematic used to create the data latches implemented in the control system.

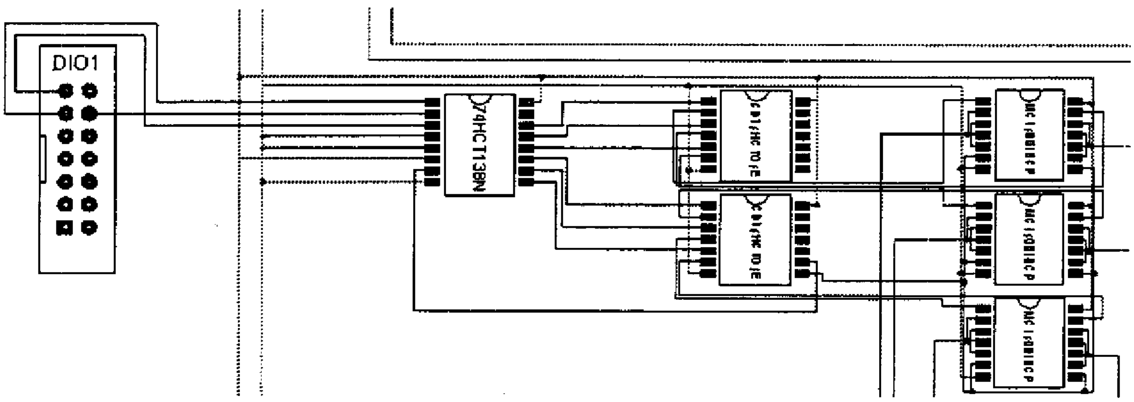


Figure 29:Section of the full control system electrical schematics detailing the DIO port, decoder, inverters, and custom data latches.

Recall how it was previously stated that none of the motors will be able to run in reverse if only one number at a time is written to Port 3 (i.e. without the latch). This is due to the way the motors are wired into the control system. Each motor is controlled by a set of four relays, two 5 V reed relays which are driven by the SBC and two larger relays to handle the actual motor loads. The load relays are controlled by the reed relays, this was designed this way for three reasons. The first was to limit the amount of current being drawn out of the IO port on the TS-2800, reed relays draw only 20 mA whereas the larger load relays draw at least 30 mA. The second was that the reed relays can be found with a 5 V coil. Due to budgetary reasons whatever components could be obtained or borrowed were used to build the control system. The following schematic shows how the two motor load relays they reverse the DC polarity in order to change the motor direction and turn on and off the motor [13].

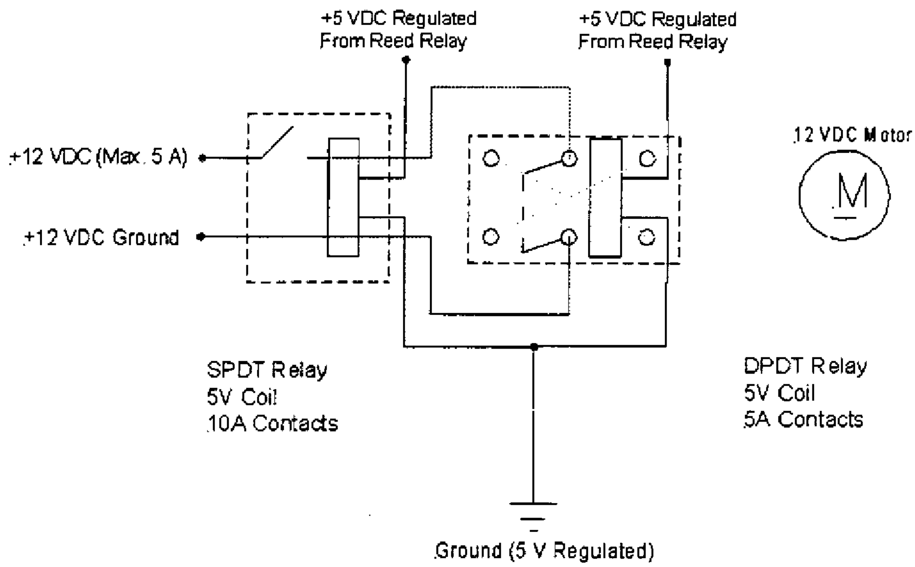


Figure 30: Schematic detailing motor control relays (Note how the DPDT contacts are crossed over in order to reverse polarity.).

When the single pole double throw (SPDT) relay coil is energized, the motor turns on, going in the forward direction, when the double pole double throw (DPDT) relay turns on the polarity is reversed and the motor reverses. Note, however, that the SPDT relay needs to remain on while the DPDT relay is running to allow the motor to run in reverse. This is where the latch comes into play again. First the number corresponding to the SPDT for the particular motor is written to then the DPDT. When executed from within an assembly language program this occurs almost instantaneously, it is impossible to detect the two relays being turned on separately [14].

From the ICs the output is put through an external circuit in order to increase the amperage to drive the reed relays and the indicator light emitting diode (LED). This circuit consists of an NPN transistor, a diode, and a few resistors. This is a common circuit used for driving loads with transistor-transistor logic (TTL)/microprocessor electronics. Figure 31 is a schematic of the output interface between the latch circuit and the reed relay/indicator light [12].

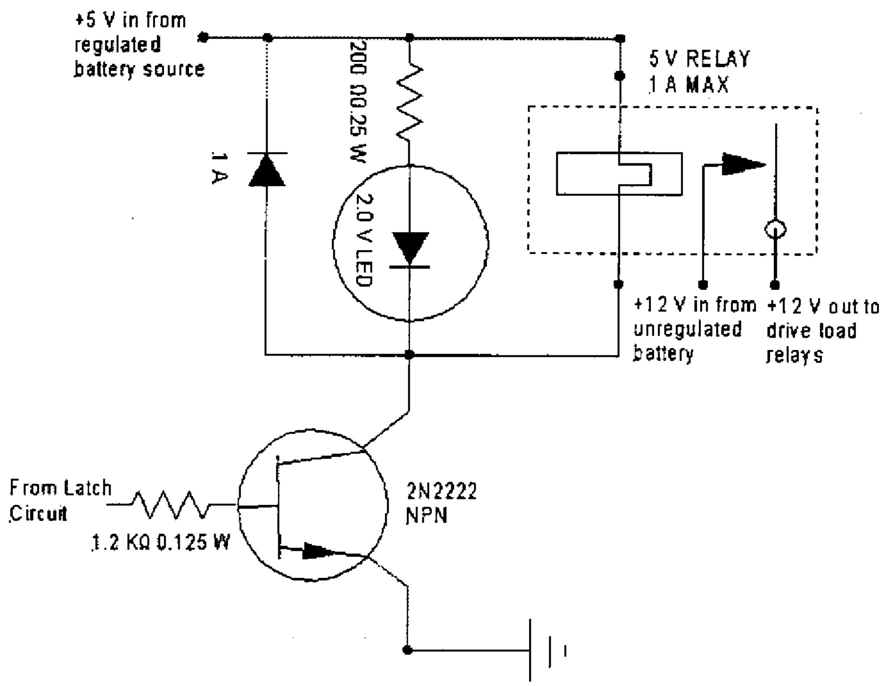


Figure 31:Schematic of the output interface between the latch circuit and the reed relay/indicator light.

The NPN transistor acts like a valve, it takes very little effort to turn on a valve and let a lot of fluid move through a pipe. The same holds true for the transistor it requires very little current to turn on the NPN transistor, yet the 2N2222 allows quite a bit of current to flow through when on (800mA DC). The 1.2 KΩ resistor on the base of the NPN is there to limit current (by using Kirchoff's voltage law I=V/R (5/1200), it is shown that I=0.0042 A). The 2.0 V LED is the indicator light mentioned before, this was key in developing the system due to the fact that it didn't need to be hooked up to the drive motors all the time to know which control line was turned on. Also, it is very difficult to tell when a relay is on just by glancing and the lights made debugging the system visually much easier.

The main purpose of designing the control system with many switches was to limit the power requirements. Another alternative to using coil relays would have been to use solid-state relays. The downfall to using solid-state relays is that there is always current leakage due to the circuitry always being live.

One other alternative to using the relays that would require very little power consumption would have been to use H-bridges and pulse width modulation. This method would allow all the motor's speed to be controlled by the computer whereas right now the motors are either on or off depending if the relays are. One crude way to vary the speed right now would be to throw a resister on the motor load line to drop the voltage going out to the motors. This is inefficient due to the power lost from heat generation in the voltage reducing resistor, but does provide a simple way to vary motor speed while prototyping.

As stated the H-bridge system would allow full motor speed control using the TS-2800 IO port. This does require higher level programming than the simple relay system, but is much more powerful in that if the vessel gets caught in some current, right now the controller cannot vary the speed to compensate. A simple H-bridge controller schematic is shown here (Figure 32).

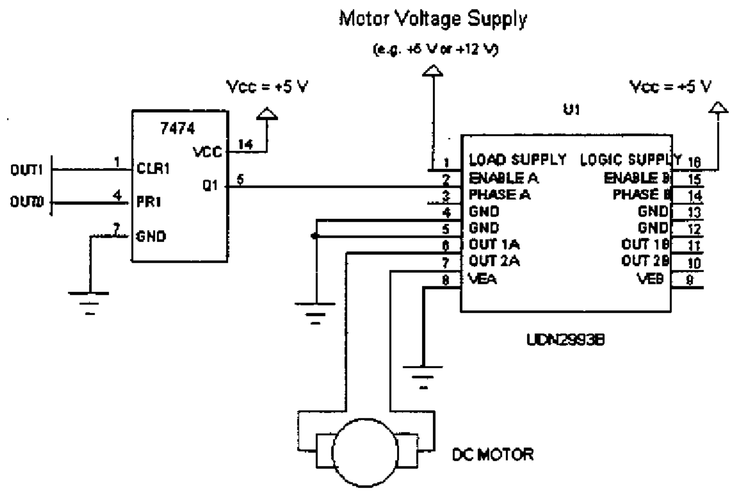


Figure 32: Schematic of one proposed method of motor speed control using an H-bridg and pulse width modulation.

The principle behind H-bridge speed control is pulse width modulation. Essentially the TS-2800 port(s) would pulse on and off at a constant frequency, but to vary the speed the time that the pulse remains on is varied. To increase the speed the 'on time' should be increased, this amount of 'on time' is referred to as duty cycle where 0% duty cycle is used when the motor isn't supposed to run and 100% duty cycle is used when the motor runs at full speed [13]. Figure 33 is an oscilloscope capture of an 80% duty cycle pulse (Note, how the line is on ~80% of the time).

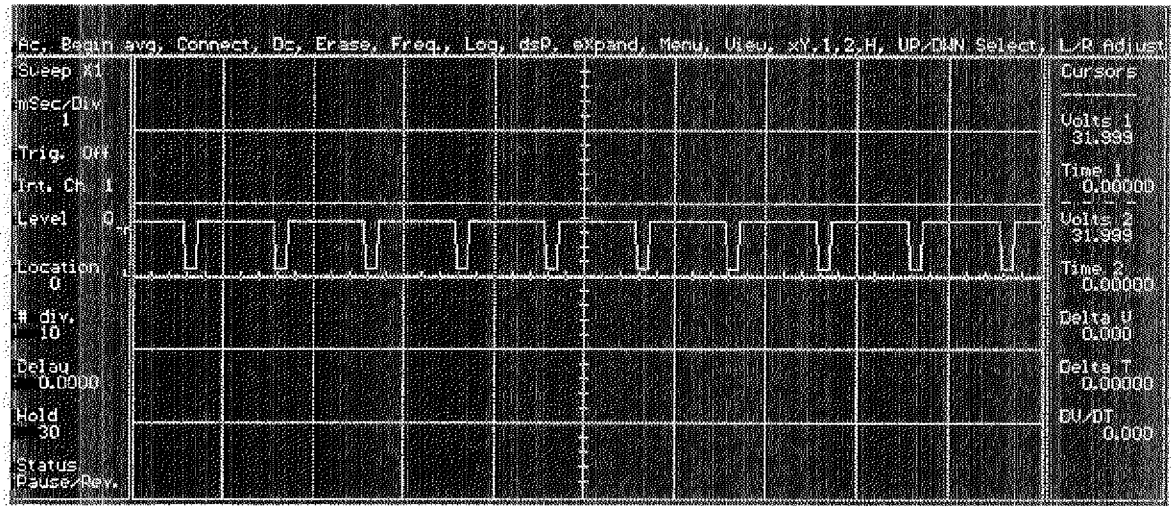


Figure 33: Oscilloscope capture of a pulse width modulation control line running at 80% duty cycle.

The reason that this system was not implemented on the vessel was due to budgetary constraints. It is suggested however that in the future, pulse width modulation speed and direction control should be incorporated into the vessel. The current electronic design has been created with that in mind, every component up to and including the latches should be reused for the speed control version.

Wireless Control:

Probably the key to making this prototype system useful for testing is its wireless control capabilities. It was previously stated that the TS-2800 contains an ethernet port, and this would be connected to a wireless network for remote control purposes. Along with remote controlling the vessel this link is very useful for data acquisition from shore.

The TS-2800 has full analog to digital capabilities, therefore any analog instrument (salinity/pH meters, depth gauges, etc.) could be connected to the SBC for either analysis or storage and the wireless link would allow someone on (or anywhere else on the planet if the system is connected to the internet) shore to monitor and gain real-time data. A wireless network was established on another SBC that was being tested for controlling the vessel (Rabbit Semiconductor RM-2200, Figure 34). This board was not used due to its lack of IO ports, but did provide a good basic knowledge of connecting wireless equipment to SBC's. A demonstration was constructed to show how four light bulbs (60 W 120 VAC) could be toggled on and off by a web server programmed on the Rabbit board through a wireless network (very impressive to watch!).

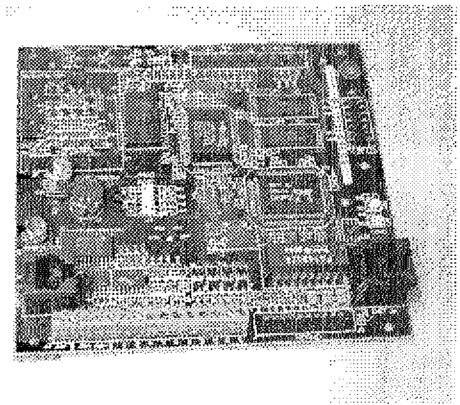


Figure 34: Rabbit RM-2200 TCP-IP SBC used to demonstrate wireless ethernet control of four light bulbs.

The wireless network interface used was built by Breezenet and is currently being used on the UNH OE tow tank for wireless data acquisition through a National Instruments ethernet field IO module. The system (Figure 36) consisted of a wireless access point and a 'PC card'. The access point looks like a standard access point, but the PC card was not the standard card that fits inside one of the slots in a computer's motherboard. Instead, it was a separate unit that had a standard RJ-45 ethernet jack, this allows any computer with a standard ethernet card, like the TS-2800's Crystal Lan, to be directly wired with an eight conductor category five cable to the 'PC card'.

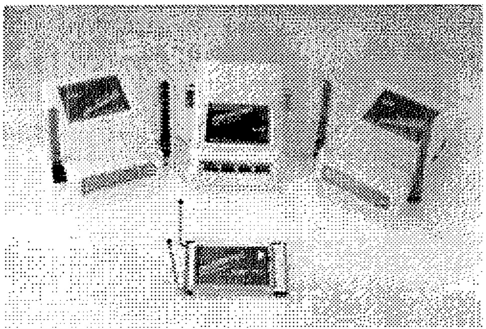


Figure 36: Breezenet wireless networking equipment used in demonstration.

The wireless network is currently not the method being employed for remote control during the prototyping stage. Instead, a spread spectrum serial interface links one of the serial ports on the TS-2800 (COM2) to a serial port on a controlling computer. The link is an Ewave Super Screamer 900 MHz Multi-Protocol Wireless Link (Figure 35). This system was used primarily because of the DOS ROM operating system that the TS-2800 runs on is incompatible with many ethernet applications (common TELNET in particularly). The TS-2800 can be upgraded to a Linux based operating system which will allow for true multi-thread processing (more than one program can run at a time) which is key for having a TELNET host application (for remotely overriding the control system) running and an HTML status page server run at the same time. The TELNET application is the most application for remotely controlling a computer, it can be configured to allow full reprogramming of the system while it is running and it works on the same level as the operating system instead of being masked like an HTML server does. After putting the system through testing and building a full-scale prototype it is suggested that the TS-2800 be upgraded to Linux and all the control software be rewritten to work for the Linux operating system.

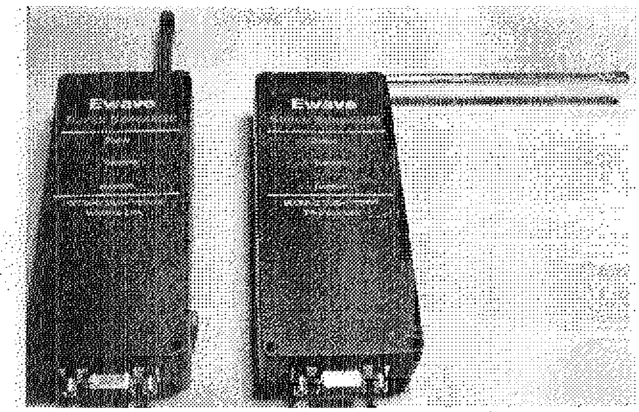


Figure 35: Ewave spread spectrum serial interface implemented for wireless manual control of The Clean Cat

Injection Connection:

The vessel control system does currently control the injection system by cold booting the injection system's PLC. There is a power connector on the vessel control system, which is turned on when the injection system is supposed to run and turned off at a programmed amount of time when the injection process is complete.

The PLC only requires 0.25 A of 12 VDC power. It gets this power from the vessels isolated 12 V deep cell battery when a relay on the vessel control system is activated. The relay is wired the same way all of the motor control relays were wired (using the same 2N2222 transistor/LED/resistor configuration). The only differences in wiring are one the DIO header pin that controls the injection process does not pass through the decoder/latch system (it provides a direct control line), and two the relay being used to turn the power on and off to the injection system PLC is a 5 V DPDT with 0.5 A contacts. This system connection can be seen in the full control system schematic on page A-2.

User Interface:

Software has been created for the TS-2800 to allow a remote user interface to be established through the spread spectrum serial link. This software is constantly being developed as the system is prototyped and therefore any description of the user interface given here may vary from that which is on the vessel.

Currently, when the control system is turned on the TS-2800 boots into the vessel control program. The user establishes a Windows HyperTerminal session corresponding with the COM port the spread spectrum serial link is connected to and is prompted with a logon screen from the TS-2800 computer (Figure 37).

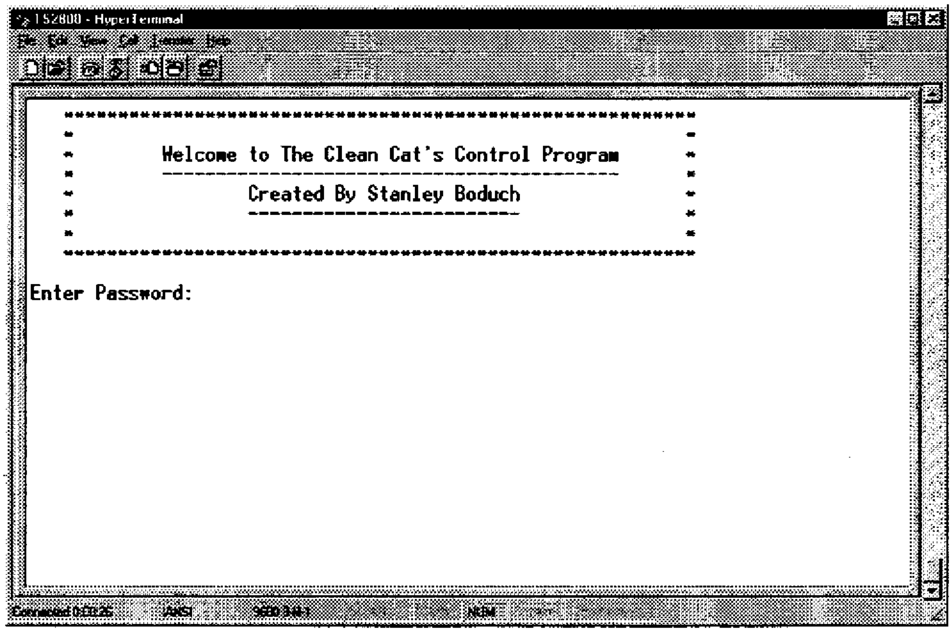


Figure 37: User Interface – Remote Logon Screen

After entering the correct password the system starts the actual vessel control program (Figure 38). From here the user can start entering commands to control the motors on the vessel by typing the number sequences on the command list followed by the [ENTER] key.

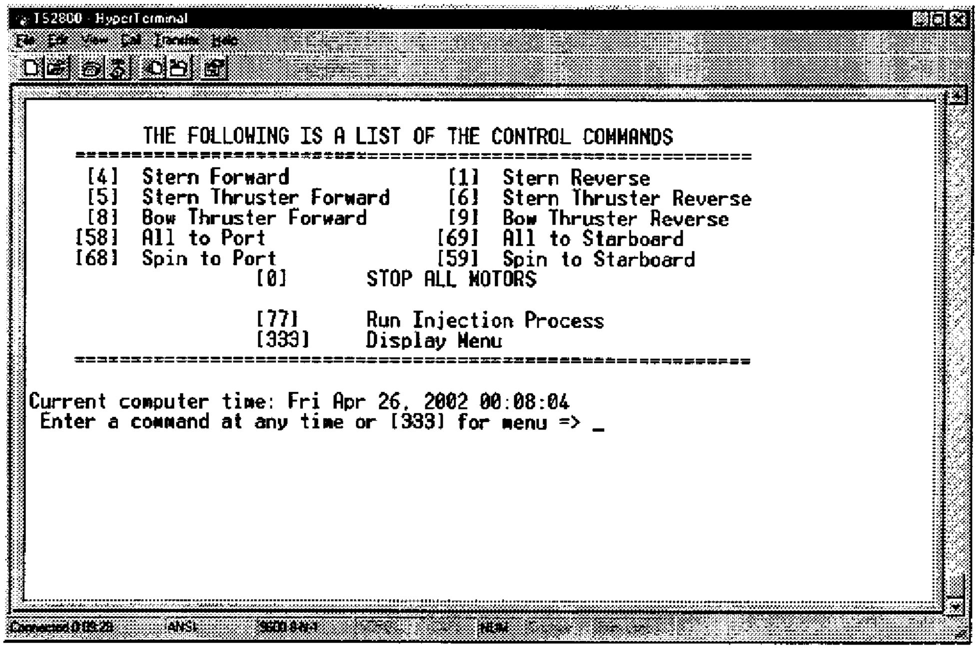


Figure 38: User Interface – Remote Vessel Control Menu

The commands were designed such that they correspond to the layout of a numeric keypad superimposed on the vessel's paddle wheel configuration; the stern center wheel is controlled using 1&4 (1 is forward, 4 is reverse), the bow thruster is controlled using 8&9, and the stern thruster is controlled using 5&6. This layout makes control like using a joystick and is more intuitive that picking commands using some other method (ex. SF would be stern forward, SR would be stern reverse), this can be confusing when you are controlling the vessel from anyplace other than sitting on top of it.

The user interface also allows for control of the injection process. The injection system currently is controlled by a PLC and the vessel control system cold boots it into a control program. The TS-2800 cannot vary any of the PLC parameters, such as wait time after injection for amendments to settle, etc. It does however provide a 'monitor' to show the status of the PLC process. This is not a true monitor in the sense that the TS-2800 receives no data from the PLC, but it has the same time structure programmed into its software, therefore simulates what the PLC is currently doing in real-time. A screen shot of the injection process control screen is shown in Figure 39.

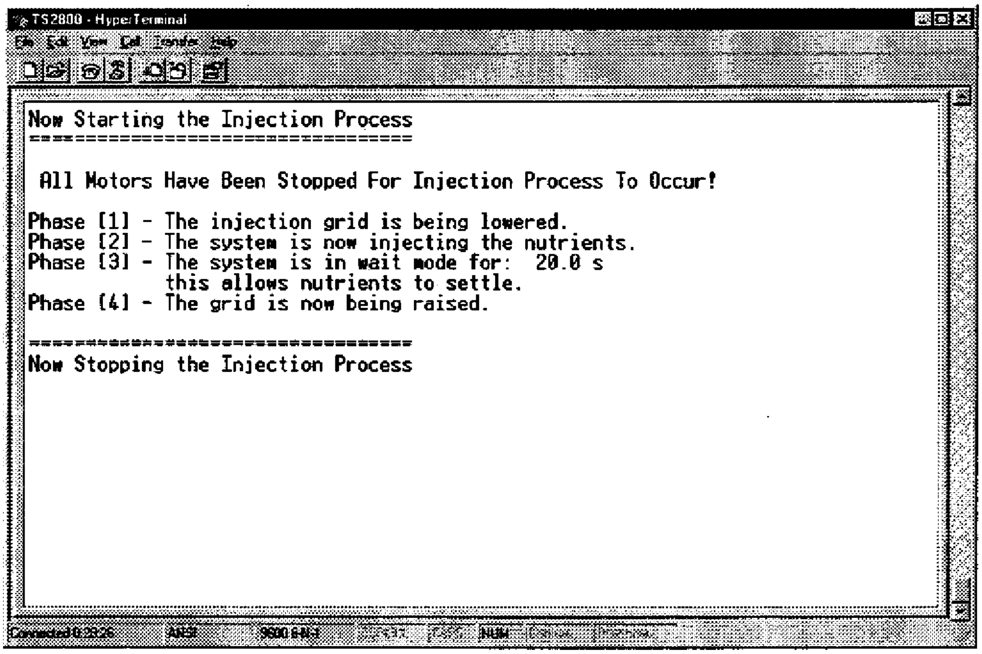


Figure 39: User Interface – Injection System Control

As stated the user interface software is constantly under development, these screen shots may not look the same as what is currently in use on the vessel. There are other advanced options available within the interface software which allow the user to exit the control program to the TS-2800 DOS system prompt and work on the system level. This allows the remote user to program any aspect of the TS-2800 and upload and download the programs without the need of connecting and disconnecting cables creating a true 'on the fly' development platform. This is critical during prototyping when parameters need to be constantly changed.

The software listing for the control program is included in Appendix B starting on page B-2.

Power Supply:

One integral part of the control system that has not been discussed yet has been the power supply or distribution to the different components in the system. All of the delicate electronics (TS-2800, LEDs, ICs, etc.) are isolated from the drive motor battery. This was done by including two 12 V marine batteries on board, one for the electronics, one for the heavy motors. There are a few reasons why this practice normally occurs. The most important being that when the heavy drive motors are on, specifically at startup, when they tend to draw a lot of current if the sensitive electronics were connected to the same source there can be enough of a voltage drop to turn off the SBC or the relays. This same phenomena occurs during the summer in city power grids when it is a hot day and everyone is running air conditioners, the line voltage drops significantly and can cause adverse affects on electrical equipment (breakers tripping, computers shutting off, etc.). So this is the primary reason to isolate the source. Another reason is the opposite of this, when the motors turn off there can be voltage spikes that can permanently damage sensitive electronics (this is usually why computers are plugged into surge suppressors in industrial buildings).

The TS-2800, ICs, and reed relays all run on 5 VDC isolated power. To provide this power a DC-DC converter was obtained from Datel Inc. in Mansfield, MA. These converters are extremely efficient (guaranteed at least 87%), and provide constant 5 VDC over an input voltage range of 10-18 VDC. This is useful since when 12 V marine batteries are charged they can sometimes produce up to 13.8 V and when almost drained the voltage drops to ~10.5 V. The TS-2800 could not function with a fluctuating voltage and this converter keeps the supply steady even as the condition of the battery changes.

This steady voltage production is primarily accomplished by the power-switching regulator contained within the converter. The simplified schematic (Figure 40) details more about how the switcher is controlled. This converter would take a lengthy discussion to detail how the system, for further reading into simple switching regulators to get the basic idea of how they function consult [13]. The converter can produce a maximum of 15 watts of output power to the electronics control system. The TS-2800 requires 800 mA at full power (4 W @ 5 VDC) The full spec sheets on the Datel DC-DC converter can be found from [15]. This is a very useful converter for any power sensitive system being designed to run off batteries and should be kept in mind for future system design. Datel also produces a wide range of converters with a variety of input and output characteristics, the spec sheets detail many of these.

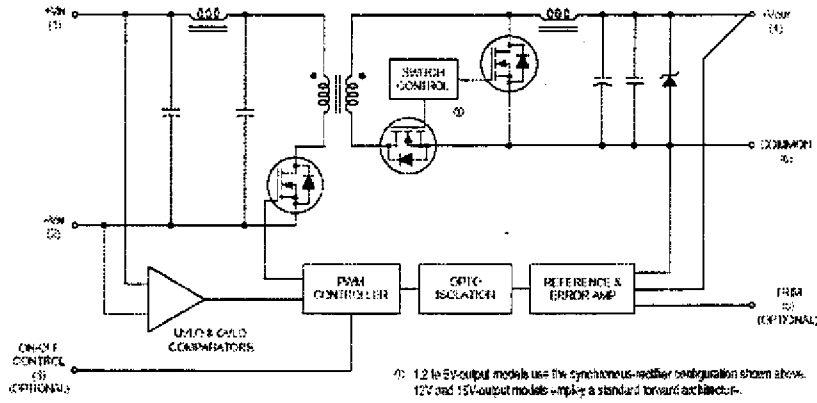


Figure 40: Simplified Schematic of Datel DC-DC converter (12 VDC to 5 VDC) used to generate control system power from 12 V deep cells.

Budget

This project was allotted less than $2000 for design, construction and testing of the entire craft. The exact dollar amount allotted was never stated and therefore was assumed to be $2000. Table 1 shows the allotted amounts for each component and the actual amount spent. The project ultimately went over budget due to the expense of some of the major components, and due to the fact that the original proposed amount was cut back to $1500. There were several donations and a few discounts on things like the motors and control components, which helped offset the budget overshoot.

Table 5: Project Spendings

Date	Company	Product or Service	Transaction Amount
12/3/01	Home Depot	Wood for test paddle wheel setup	$59.84
2/6/02	Technologic Systems	SBC for wireless control	$228.50
2/7/02	Northeast Building Material	Pontoon floats	$186.00
2/11/02	Warren's Hardware	Adhesive and fasteners for floats	$7.47
2/15/02	Bison Motors	Paddle wheel Motors	$560.00
2/15/02	MACY Industries	Sheet metal for paddle wheels	$300.44
2/16/02	Home Depot	Wood for boat deck	$60.65
3/5/02	Home Depot	Hardware for paddles	$42.27
3/5/02	Rockingham Electric	Unistrut for paddle arms	$22.85
3/9/02	Home Depot	Hardware for paddles	$25.26
3/11/02	Ace Hardware	Hardware for paddles	$4.78
3/11/02	Northeast Building Material	More pontoon floats	$93.00
3/12/02	Ace Hardware	Hardware for paddles	$9.45
4/13/02	**Total**		**$1,600.51**

Conclusion

The Clean Cat has been created to test a concept low draft injection delivery vessel prior to full-scale prototype design. Operation in marsh areas for any duration is not recommended due to the composition, specification and grade of the parts used. The Clean Cat does move as was proposed without the winch system on it, and with additional funds one could be placed onboard. The fundamental systems of the vessel were designed with upscale production in mind. Each system was also designed and fabricated in an open-ended fashion so that assembly and disassembly could be done for any changes necessary in design. The control system has also been designed for interaction with the injection system so full remote control of the injection process is possible. Future plans for the vessel include fully automating the operation and construction for long durations in the marine environment.

References

[1] United States Department of Energy-Energy Information Administration, Annual Energy Review, DOE/EIA-0384(2000), http://www.eia.doe.gov/emeu/aer/pdf/pages/sec5.pdf, 2001.

[2] United States Environmental Protection Agency-Office of Emergency and Remedial Response, Understanding Oil Spills and Oil Spill Response, PB2000-963401, 1999.

[3] International Tanker Owners Pollution Federation Limited, Clean-Up Techniques-Alternative Techniques for Spill Response, http://www.itopf.com.

[4] Fredriksson, David. CICEET FY 2002 Proposal Application: Injection Vessel to Enhance the Bioremediation of Oil-Contaminated Salt Marshes. University of New Hampshire, Durham, NH, 2002.

[5] Hildebrand, D., Natural and Terminal Electron Acceptor-Enhanced In-Situ Bioremediation of Oil Contaminated Salt Marsh Sediments. Master's Degree Thesis submitted in partial requirement for the Civil Engineering degree program. University of New Hampshire, Durham, NH, 2001.

[6] Borenstein, Everett, Feng. Where am I? Sensors and Methods for Mobile Robot Positioning, University of Michigan, 1996.

[7] Everett, H.R. Sensors for Mobil Robots – Theory and Application, MA: A K Peters, Ltd. 1995.

[8] MTI Research Website Homepage, www.mtir.com.

[9] Parr E.A., Programmable Controllers an Engineer's Guide, Second Edition, Massachusetts: Newnes, 1999.

[10] Driscoll, F.F., Coughlin R.F., Villanucci R.S., Data Acquisition and Process Control with the M68HC11 Microcontroller, Second Edition, New Jersey: Prentice-Hall, Inc., 2000.

[11] Tabak, Daniel, Advanced Microprocessors, New York: McGraw-Hill, Inc., 1995.

[12] Cogdell J.R., Foundations of Electrical Engineering, Second Edition, New Jersey: Prentice Hall, Inc., 1996.

[13] McComb Gordon, The Robot Builder's Bonanza, Second Edition, New York: McGraw-Hill, Inc., 2001.

[14] Lunt Karl, Build Your Own Robot!, Massachusetts: A K Peters, Ltd., 2000.

[15] Datel DC-DC Converter Specifications Sheet. http://www.datel.com/data/power/uwr15wa-series.pdf

[16] Anderson, Bjedov, Scarbrough, Essential C An Introduction for Scientists & Engineers, Florida: 1995.

[17] Hanly J.R, Koffman E.B, C Program Design for Engineers, Second Edition, Massachusetts: Addison Wesley Longman, Inc., 2001.

[18] Jamsa, K, Klander, L, Jamsa's C/C++ Programmer's Bible, NV: Jamsa Press, 1998.

[19] Norton Peter, Peter Norton's Complete Guide to DOS 6.22, Indiàna: Sams Publishing, 1994.

[20] Avallone E. A. and T. Baumeister III, Mark's Standard Handbook for Mechanical Engineers, Tenth Edition, New York: McGraw-Hill, Inc., 1996.

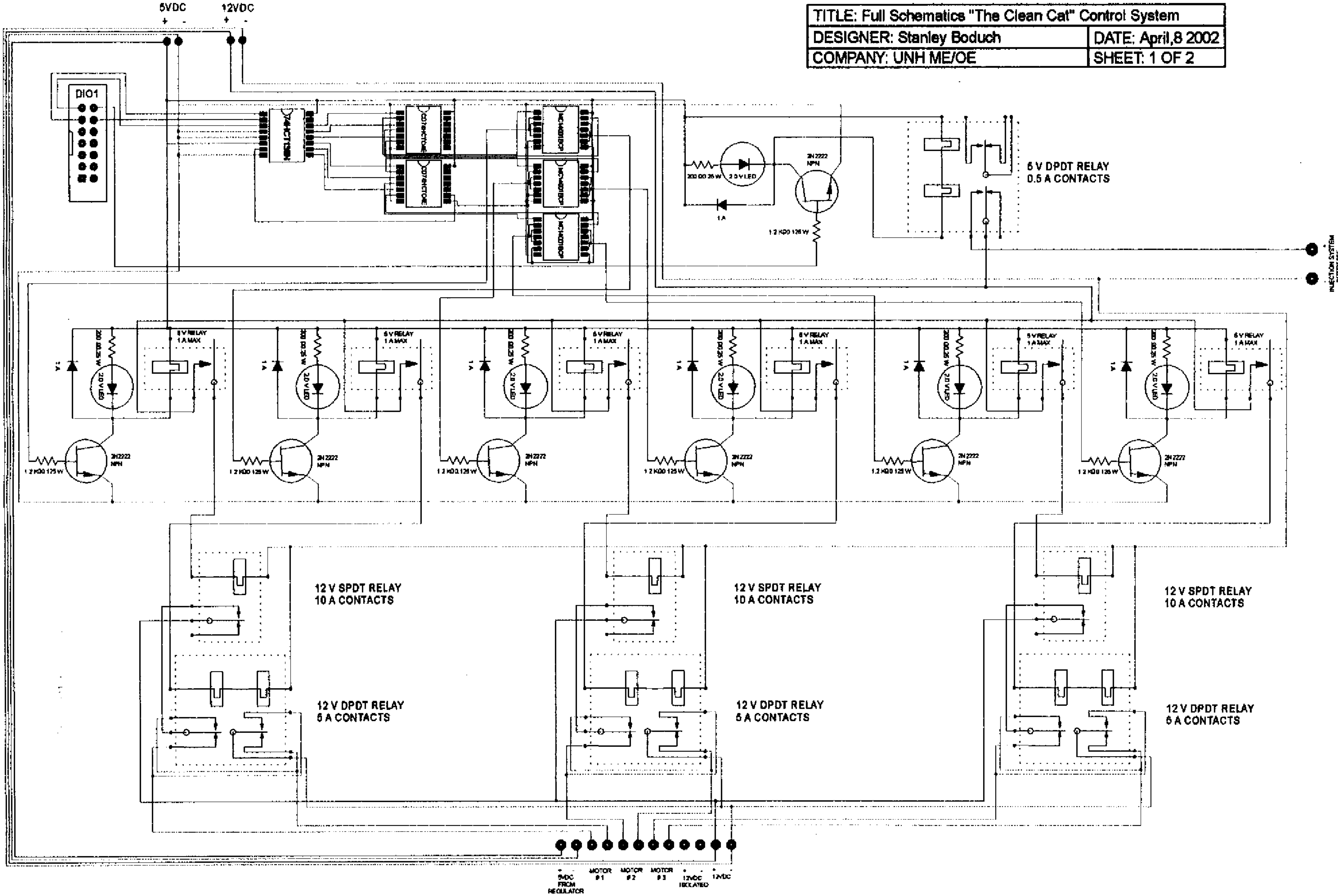
[21] Doeblin E. O., Measurement Systems Application and Design, Fourth Edition, Massachusetts: McGraw-Hill, Inc., 1990.

[22] Ogata K., System Dynamics, Third Edition, New Jersey: Prentice-Hall, Inc., 1998.

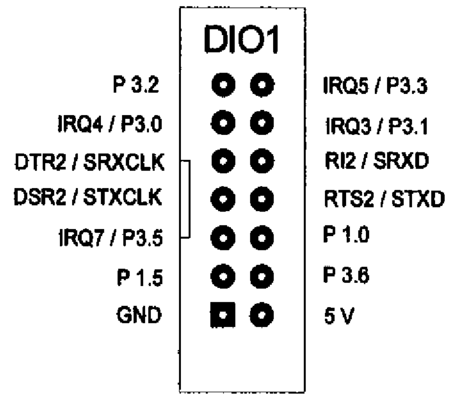
Appendix A. – Control System Electrical Schematics

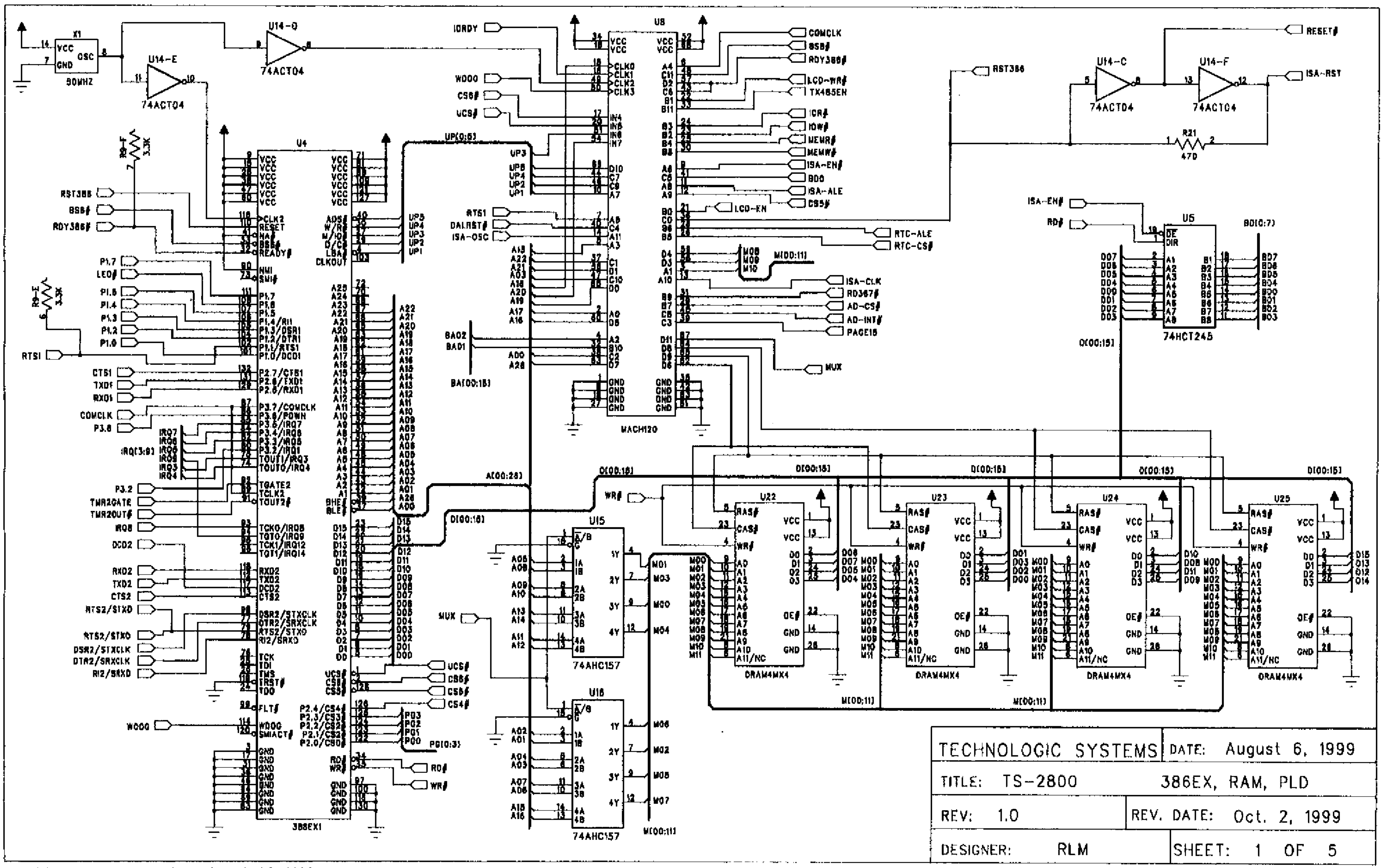
Table of Schematics

TITLE: Full Schematics "The Clean Cat" Control System	
DESIGNER: Stanley Boduch	DATE: April,8 2002
COMPANY: UNH ME/OE	SHEET: 1 OF 2

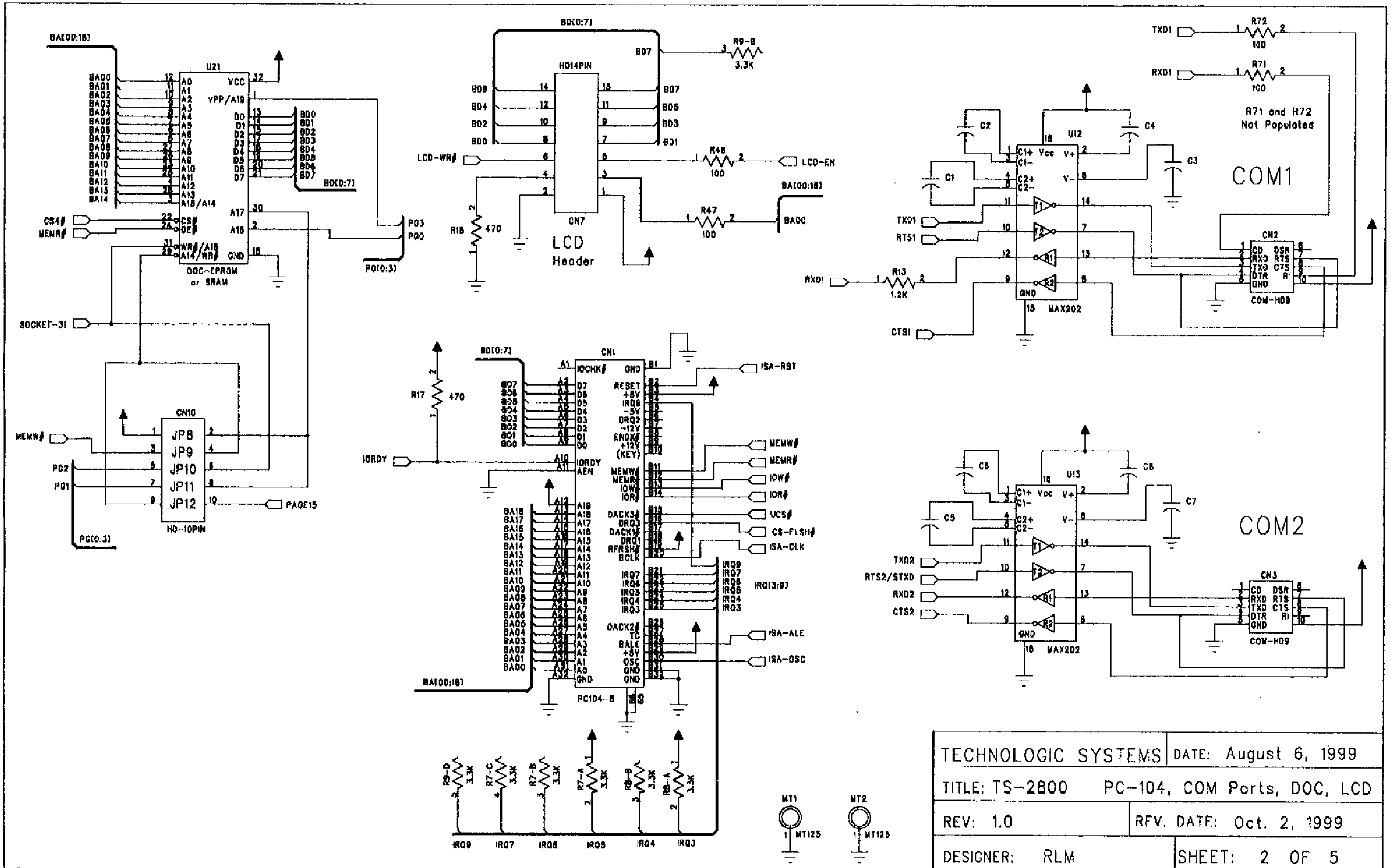


TITLE: Pinout for DIO1 header on the TS-2800 SBC	
DESIGNER: Stanley Boduch	DATE: April,8 2002
COMPANY: UNH ME/OE	SHEET: 2 OF 2

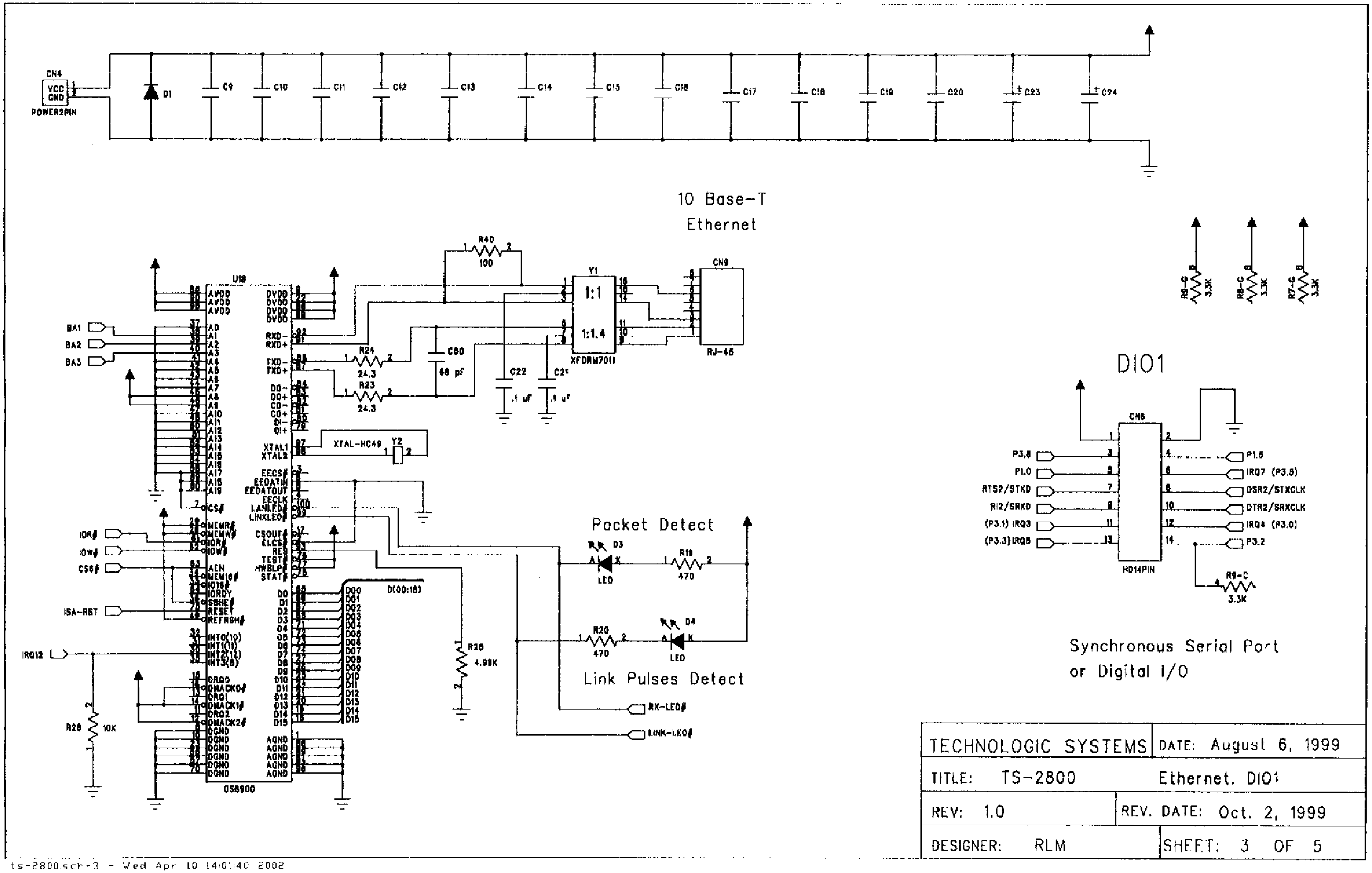


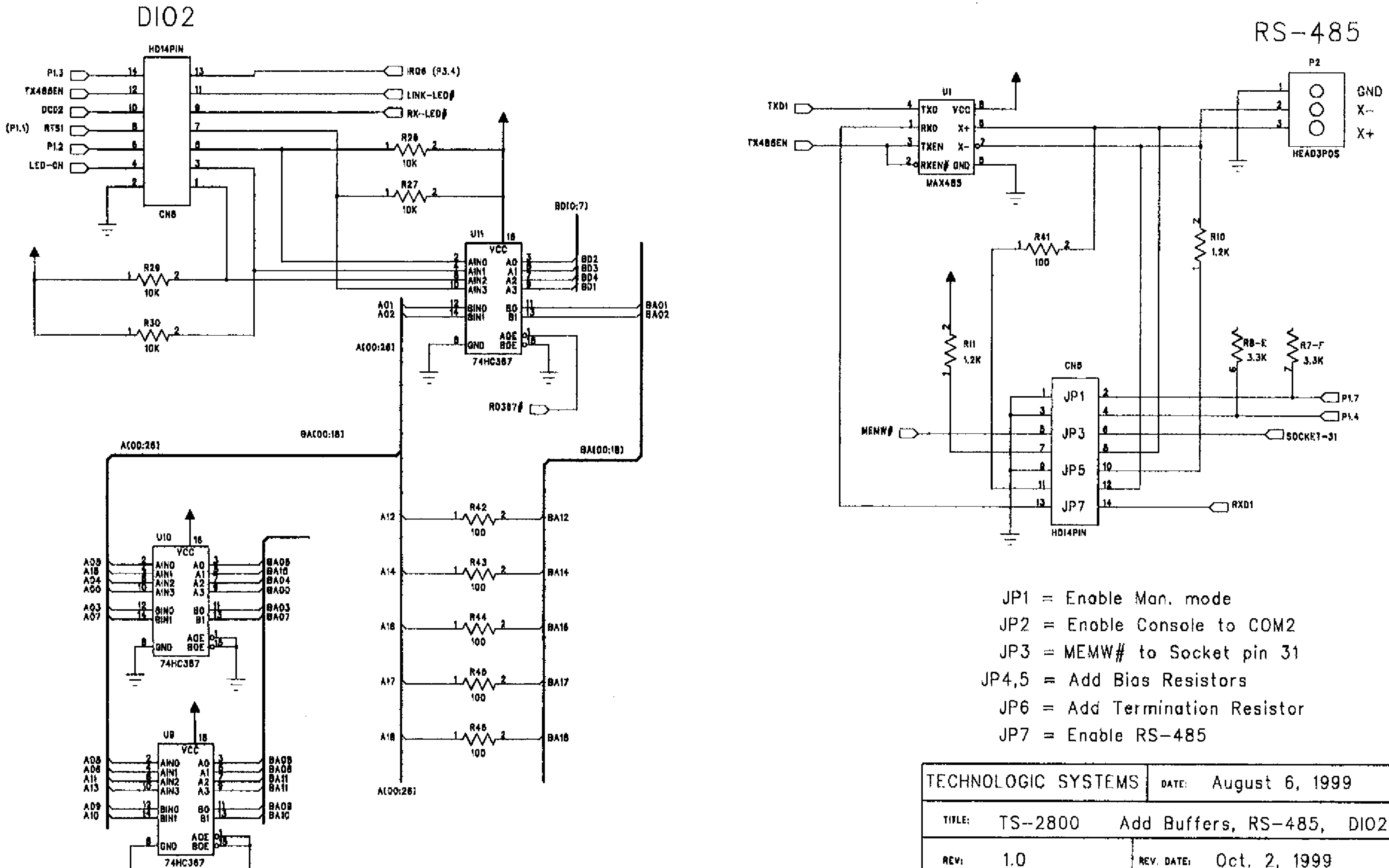


ts-2800.sch-1 - Wed Apr 10 14:01:39 2002



ts-2800.sch-2 - Wed Apr 10 14:01:40 2002

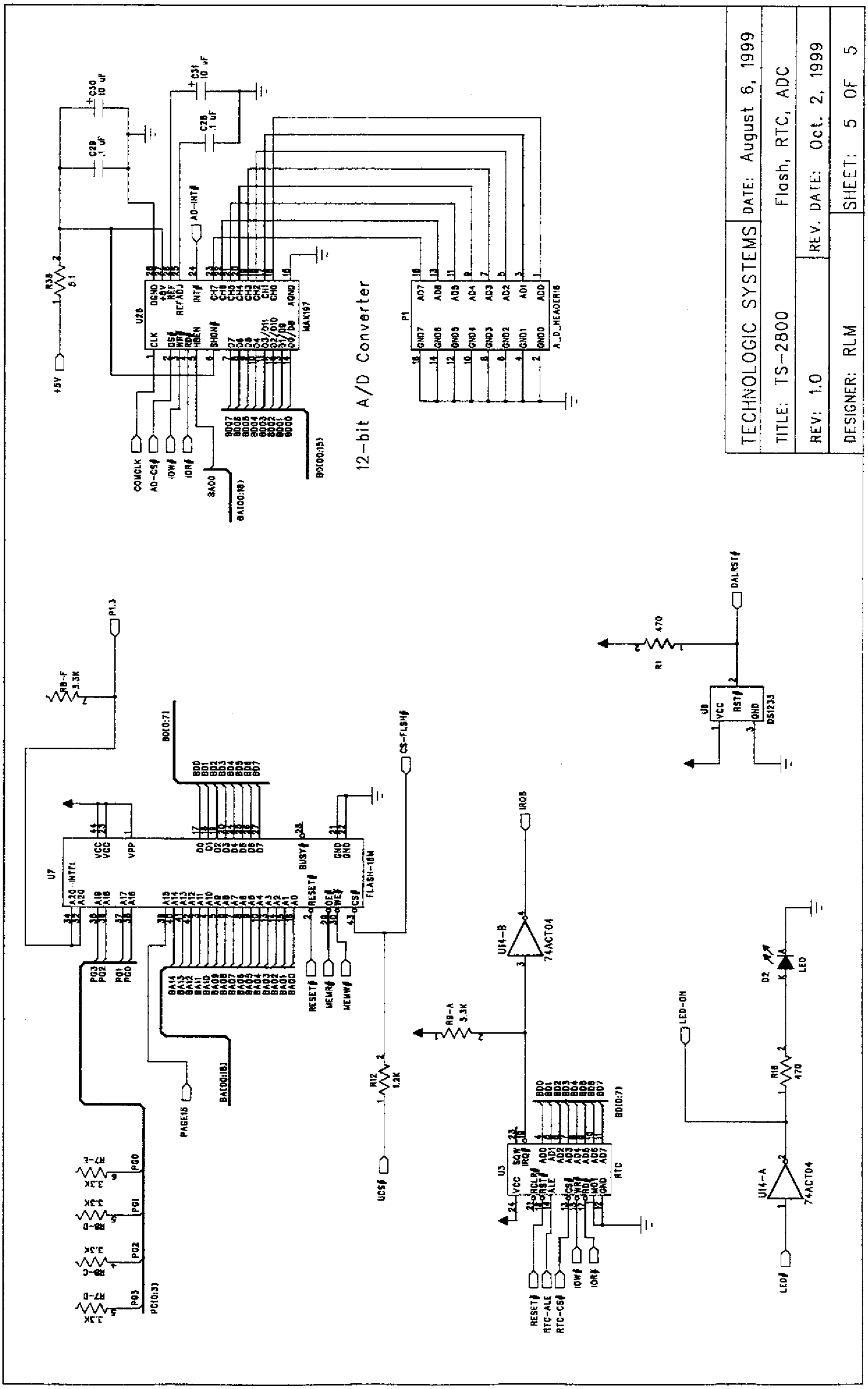

10 Base-T
Ethernet
DIO1
Packet Detect
Link Pulses Detect
Synchronous Serial Port
or Digital I/O
TECHNOLOGIC SYSTEMS
DATE: August 6, 1999
TITLE: TS-2800
Ethernet, DIO1
REV: 1.0
REV. DATE: Oct. 2, 1999
DESIGNER: RLM
SHEET: 3 OF 5
ts-2800.sch-3 - Wed Apr 10 14:01:40 2002



JP1 = Enable Man. mode
JP2 = Enable Console to COM2
JP3 = MEMW# to Socket pin 31
JP4,5 = Add Bias Resistors
JP6 = Add Termination Resistor
JP7 = Enable RS-485

TECHNOLOGIC SYSTEMS		DATE:	August 6, 1999
TITLE:	TS-2800 Add Buffers, RS-485, DIO2		
REV:	1.0	REV. DATE:	Oct. 2, 1999
DESIGNER:	RLM	SHEET:	4 of 5

ts-2800.sch-4 - Wed Apr 10 14:01:40 2002


12-bit A/D Converter
TECHNOLOGIC SYSTEMS
DATE: August 6, 1999
TITLE: TS-2800
Flash, RTC, ADC
REV: 1.0
REV. DATE: Oct. 2, 1999
DESIGNER: RLM
SHEET: 5 OF 5
ts-2800.sch-5 - Wed Apr 10 14:01:41 2002

Appendix B. – Control System Program Listings

Table of Program Listings

The Clean Cat Main Control Program Listing:
(Note: This program is constantly being developed as the system is prototyped and the current code being used may not have the same appearance. This code, however, is guaranteed to function and has been tested on the system. It was compiled using Sybase Watcom C/C++ V 11) [16][17][18]

```
/*************************************************************************

   The Clean Cat Control Program        Created By Stanley Boduch
                                                   UNH ME/OE
   Last Updated: 4/23/02

*************************************************************************/

#include <stdio.h>
#include <string.h>
#include <conio.h>
#include <time.h>

main(){
   int n;
   int j;
   int flag;
   int ex=1;
   float downtime=5.0;
   float uptime=5.0;
   float injecttime=2.5;
   float waittime=20.0;
   float totaltime=downtime+uptime+injecttime+waittime+3.5;
   float eltime;

   char buffer[128];
   char password[10];

   struct tm *datetime;
   time_t current_time;
   time_t start_time;
   system(" cls ");

   printf("   **********************************************************\n");
   printf("   *                                                        *\n");
   printf("   *        Welcome to The Clean Cat's Control Program      *\n");
   printf("   *        ------------------------------------------      *\n");
   printf("   *                Created By Stanley Boduch               *\n");
   printf("   *                -------------------------               *\n");
   printf("   *                                                        *\n");
   printf("   **********************************************************\n\n");
   while(ex==1){
      printf("Enter Password: ");
      j=0;
      scanf("%c", &password[j]);
      while(password[j]!='\n'){
         j=j+1;
         flag=scanf("%c", &password[j]);
      }
      password[j]='\0';
      if (strcmp(password, "oil"))
        printf("\nPassword Incorrect\n");
      else{
        printf("\nPassword OK\n");
        time(&start_time);
        do{time(&current_time);}while(difftime(current_time, start_time)<1.5);
        ex=2;
      }
   }
      system(" cls ");
      printf("\n          THE FOLLOWING IS A LIST OF THE CONTROL COMMANDS\n");
      printf("     ============================================================\n");
```

```c
    printf("      [4]  Stern Forward               [1]  Stern Reverse\n");
    printf("      [5]  Stern Thruster Forward      [6]  Stern Thruster Reverse\n");
    printf("      [8]  Bow Thruster Forward        [9]  Bow Thruster Reverse\n");
    printf("     [58]  All to Port               . [69]  All to Starboard\n");
    printf("     [68]  Spin to Port               [59]  Spin to Starboard\n");
    printf("                    [0]       STOP ALL MOTORS\n\n");
    printf("                    [77]      Run Injection Process\n");
    printf("                    [333]     Display Menu\n");
    printf("     ============================================================\n\n");

    time(&current_time);
    datetime=localtime(&current_time);
    strftime(buffer, sizeof(buffer), "%x %X", datetime);
    printf("Current computer time: %s\n", buffer);

 ex=1;
 while(ex==1){
    printf(" Enter a command at any time or [333] for menu => ");
    scanf("%d", &n);
    if(n==0){
       system(" 0 ");
       printf("[0] All Motors Stopped\n");
    }

    else if(n==4){
       system(" 4 ");
       printf("[4] Stern Motor Forward\n");
    }

    else if(n==1){
       system(" 1 ");
       printf("[1] Stern Motor Reversed\n");
    }

    else if(n==5){
       system(" 5 ");
       printf("[5] Stern Thruster Forward\n");
    }

    else if(n==6){
       system(" 6 ");
       printf("[6] Stern Thruster Reversed\n");
    }

    else if(n==8){
       system(" 8 ");
       printf("[8] Bow Thruster Forward\n");
    }

    else if(n==9){
       system(" 9 ");
       printf("[9] Bow Thruster Reversed\n");
    }

    else if(n==58||n==85){
       system(" 58 ");
       printf("[58] All to Port\n");
    }

    else if(n==69||n==96){
       system(" 69 ");
       printf("[69] All to Starboard\n");
    }

    else if(n==68||n==86){
       system(" 68 ");
       printf("[68] Spin to Port\n");
    }

    else if(n==59||n==95){
       system(" 59 ");
```

```
   printf("[59] Spin to Starboard\n");
}

else if(n==333){
   system(" cls ");
   printf("\n          THE FOLLOWING IS A LIST OF THE CONTROL COMMANDS\n");
   printf("    =============================================================\n");
   printf("     [4]  Stern Forward              [1]  Stern Reverse\n");
   printf("     [5]  Stern Thruster Forward     [6]  Stern Thruster Reverse\n");
   printf("     [8]  Bow Thruster Forward       [9]  Bow Thruster Reverse\n");
   printf("    [58]  All to Port               [69]  All to Starboard\n");
   printf("    [68]  Spin to Port              [59]  Spin to Starboard\n");
   printf("                  [0]        STOP ALL MOTORS\n\n");
   printf("                  [77]       Run Injection Process\n");
   printf("                  [333]      Display Menu \n");
   printf("    =============================================================\n");
   time(&current_time);
   datetime=localtime(&current_time);
   strftime(buffer, sizeof(buffer), "%x %X", datetime);
   printf("Current computer time: %s\n", buffer);
}

else if(n==77){
   system(" cls ");
   printf("Now Starting the Injection Process\n");
   printf("==================================\n\n");
   system(" 0 ");
   printf(" All Motors Have Been Stopped For Injection Process To Occur!\n\n");

   time(&start_time);
   do{time(&current_time);}while(difftime(current_time, start_time)<1.5);
   system(" 77 ");

   time(&start_time);
   eltime=downtime;
   printf("Phase [1] - The injection grid is being lowered.\n");
   do{time(&current_time);}while(difftime(current_time, start_time)<eltime);

   eltime=eltime+injecttime;
   printf("Phase [2] - The system is now injecting the nutrients.\n");
   do{time(&current_time);}while(difftime(current_time, start_time)<eltime);

   eltime=eltime+waittime;
   printf("Phase [3] - The system is in wait mode for: %5.1f s \n", waittime);
   printf("            this allows nutrients to settle.\n");
   do{time(&current_time);}while(difftime(current_time, start_time)<eltime);

   eltime=eltime+uptime;
   printf("Phase [4] - The grid is now being raised.\n\n");
   do{time(&current_time);}while(difftime(current_time, start_time)<eltime);

   printf("==================================\n");
   printf("Now Stopping the Injection Process\n");
   do{time(&current_time);}while(difftime(current_time, start_time)<totaltime);
   system(" 0 ");
   system(" cls ");
   printf("Injection Process Complete! - Resuming Vessel Control Program\n\n");
}

else if(n==123){
   system(" cls ");
   printf("\n       THE FOLLOWING IS A LIST OF ADVANCED CONTROL COMMANDS\n");
   printf("    =============================================================\n");
   printf("                  [0]        STOP ALL MOTORS\n");
   printf("                  [333]      Display Basic Control Menu\n");
   printf("                  [888]      EXIT TO DOS\n");
   printf("    =============================================================\n\n");
   time(&current_time);
   printf("The current date and time: %s\n", ctime(&current_time));
}
```

```
        else if(n==888){ex=2;}

        else{printf("\n Invalid Choice\n\n");
        }
    }
}
```

The Clean Cat Assembly Code Listings:

(Note: These programs are constantly being developed as the system is prototyped. These code listings are guaranteed to function and have been tested on the system. They were compiled to DOS *.COM files using Borland's Turbo Assembler V 2.01)

Assembly Code Listing 1: PORTCON.ASM

```
%TITLE 'PORT CONFIG'

; Created by Stanley Boduch 4/11/02
; This program initializes the ports on the
; Intel 386 EX board for control purposes.

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F824h      ; load 0F824 (P3CFG) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F824

        mov     dx, 0F874h      ; load 0F874 (P3DIR) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, ax          ; write the register back out to 0F824

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, ax          ; write the register back out to 0F824

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110000b   ; change the register so LB is 0000
        out     dx, ax          ; write the register back out to 0F824

Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 2: 0.ASM

```
%TITLE '0'

; Created by Stanley Boduch 4/11/02
; This program turns off all output points
; (stops the motors and injection process)

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 3: 4.ASM

```
%TITLE '4'

; Created by Stanley Boduch 4/11/02
; This program turns on the stern motor in the forward direction

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110001b   ; change the register so LB is 0001
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 4: 1.ASM

```
%TITLE '1'

; Created by Stanley Boduch 4/11/02
; This program turns on the stern motor in the reverse direction

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110001b   ; change the register so LB is 0001
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110010b   ; change the register so LB is 0010
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 5: 5.ASM

```
%TITLE '5'

; Created by Stanley Boduch 4/11/02
; This program turns on the stern thruster in the forward direction

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110011b   ; change the register so LB is 0011
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872


Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 6: 6.ASM

```
%TITLE '6'

; Created by Stanley Boduch 4/11/02
; This program turns on the stern thruster in the reverse direction

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110011b   ; change the register so LB is 0011
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110100b   ; change the register so LB is 0100
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 7: 8.ASM

```
%TITLE '8'

; Created by Stanley Boduch 4/11/02
; This program turns on the bow thruster in the forward direction

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110101b   ; change the register so LB is 0101
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872


Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 8: 9.ASM

```
%TITLE '9'

; Created by Stanley Boduch 4/11/02
; This program turns on the bow thruster in the reverse direction

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110101b   ; change the register so LB is 0101
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110110b   ; change the register so LB is 0110
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 9: 58.ASM

```
%TITLE '58'

; Created by Stanley Boduch 4/11/02
; This program turns on both the bow and stern
; thrusters in the forward direction

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110011b   ; change the register so LB is 0011
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110101b   ; change the register so LB is 0101
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872


Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 10: 69.ASM

```
%TITLE '69'

; Created by Stanley Boduch 4/11/02
; This program turns on both the bow and
; stern thrusters in the reverse direction

     IDEAL

     MODEL TINY

     CODESEG
     ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110101b   ; change the register so LB is 0011
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110110b   ; change the register so LB is 0100
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110011b   ; change the register so LB is 0011
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110100b   ; change the register so LB is 0100
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

     END Start
```

Assembly Code Listing 11: 68.ASM

```
%TITLE '68'

; Created by Stanley Boduch 4/11/02
; This program turns on the stern thruster in the reverse direction
; and the bow thruster in the forward direction.

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110101b   ; change the register so LB is 0101
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110011b   ; change the register so LB is 0011
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110100b   ; change the register so LB is 0100
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 12: 59.ASM

```
%TITLE '59'

; Created by Stanley Boduch 4/11/02
; This program turns on the stern thruster in the forward direction
; and the bow thruster in the reverse direction.

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110011b   ; change the register so LB is 0011
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110101b   ; change the register so LB is 0011
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110110b   ; change the register so LB is 0100
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872


Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

Assembly Code Listing 13: 77.ASM

```
%TITLE '77'

; Created by Stanley Boduch 4/11/02
; This program turns on the injection system controller

    IDEAL

    MODEL TINY

    CODESEG
    ORG 100h
Start:
        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11110111b   ; change the register so LB is 0111
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        or      al, 11111000b   ; change the register so LB is 1000
        out     dx, al          ; write the register back out to 0F872

        mov     dx, 0F872h      ; load 0F872 (P3LTC) into dx
        in      al, dx          ; read the register
        and     al, 11110000b   ; change the register so LB is 0000
        out     dx, al          ; write the register back out to 0F872


Exit:
        mov     ax, 04C00h      ; DOS Function
        int     21h             ; End this program

    END Start
```

The Clean Cat CONFIG.SYS and AUTOEXEC.BAT file necessary to boot into the control program [19]:

System Configuration File Listing: CONFIG.SYS

```
rem to setup RAM disk uncomment the following line
rem device=\dos\vdisk.sys /kbtouse=1024
rem using a RAM disk may cause performance lags on all disk accesses
```

Automatic Startup Execution Batch File Listing: AUTOEXEC.BAT

```
path=a:\util;a:\dos;a:\ethernet;
epktisa 0x60
cd\programs
portcon
menu
```

Appendix C. Paddle Wheel System Dynamics Analysis

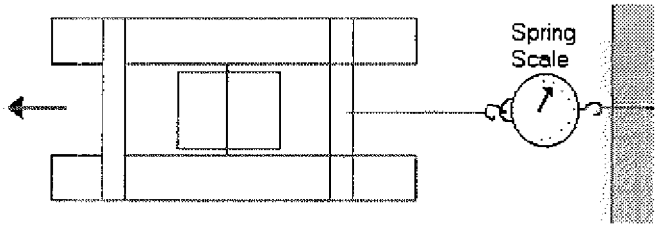
Experimental Overview

Experimental data was obtained to determine the acceleration, drag, and thrust of the spinning paddle wheel. The purpose of these experiments was to determine if the paddle wheel would meet the needed criteria of the bioremediation vessel (i.e. low speed, high torque).

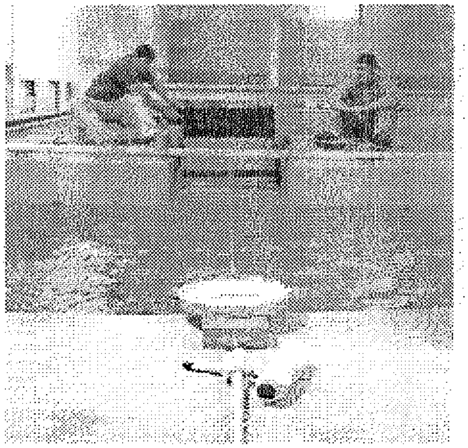
Physical Experiments Explained

To experimentally obtain the thrust, measurements were taken using a spring scale and a 100-pound load cell. To measure the thrust using the spring scale one side of the scale was fixed to the side of the pool and the other was tethered to the stern of the boat. This experimental setup closely resembles a mass-spring-damper system: the spring of the scale, the damping of the water, and the mass of the boat. Every time one of the paddles strikes the water there is an instantaneous increase in thrust and then directly following the thrust drops, this keeps repeating as the wheel turns. So an oscillation is created within the spring scale along with the stretching of the rope causing this oscillatory motion to occur. A better drive system, such as a chain drive as opposed to the belt may have decreased any slippage. For measurement purposes the tension of the belt was increased when taking measurements.

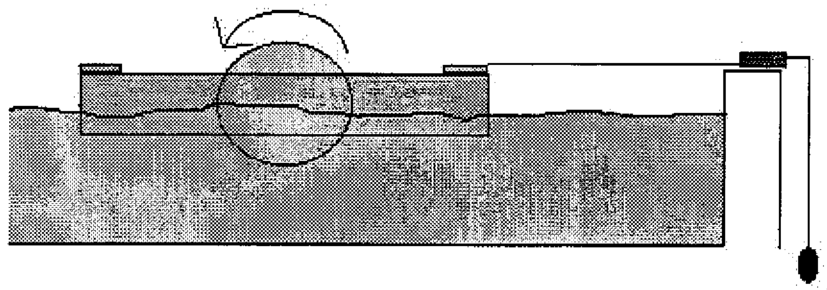
Figure 1- Overhead Illustration of Spring Scale Thrust Test



Picture of Spring Scale Thrust Test



Side View Spring Scale Thrust Test



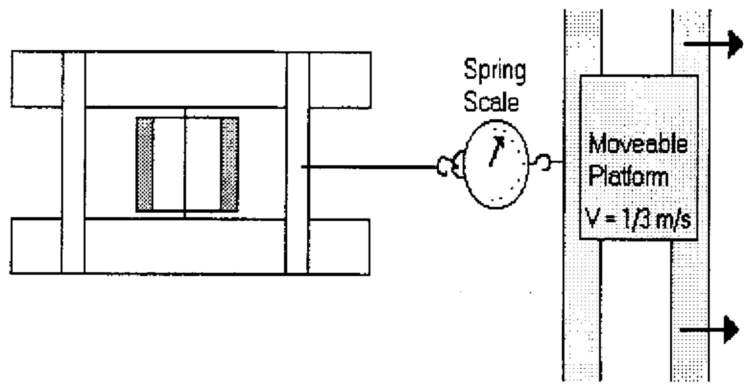
When using the load cell similar data was obtained. The setup was similar to that of the spring scale, but rather than having the fixed scale there was a force transducer in its place. Data was taken at 12.5, 18, 21, and 28 volts inputted into the motor. Data was taken at a rate of 10 Hz for 1000 samples. The data once again does not reach a steady-state value; it fluctuates between zero and a maximum value for each voltage measurement.

Table 1

Voltage (V)	Low Load (Lbs)	High Load (Lbs)
12	0	3
16.4	1.5	5
25	4.5	8.5
30	7	11
35	8	13

To obtain the drag of the boat it was dragged across the pool using the movable XY table mounted on the tank. The support consists of two I-beams that move the distance of the pool on rollers. The velocity of the table can be varied with a handheld remote control.

Figure 4 – Drag Illustration



From one of the beams a boom was dropped to the surface of the water. This way the force is completely horizontal to the boat and no force is pulling the boat upwards which would decrease the drag giving unrealistic results. The table moves at a known speed of 1/3 m/s and a spring scale was used to measure the force in pounds. For this experimental setup there is a large amount of drag due to the size of the pontoons, so this is a critical measurement.

Table 2

Trial	Force (Lbs)
1	6
2	6
3	5.5
4	6

The acceleration was found by running the boat from rest, a known distance and clocking the time. A steady velocity is never reached due to the short distance of the pool, so there is a constant acceleration for the 13 meters. The acceleration was found with one 12-volt battery, two batteries, and three batteries in series (Therefore producing 12.7, 25.5, and 38.11 VDC).

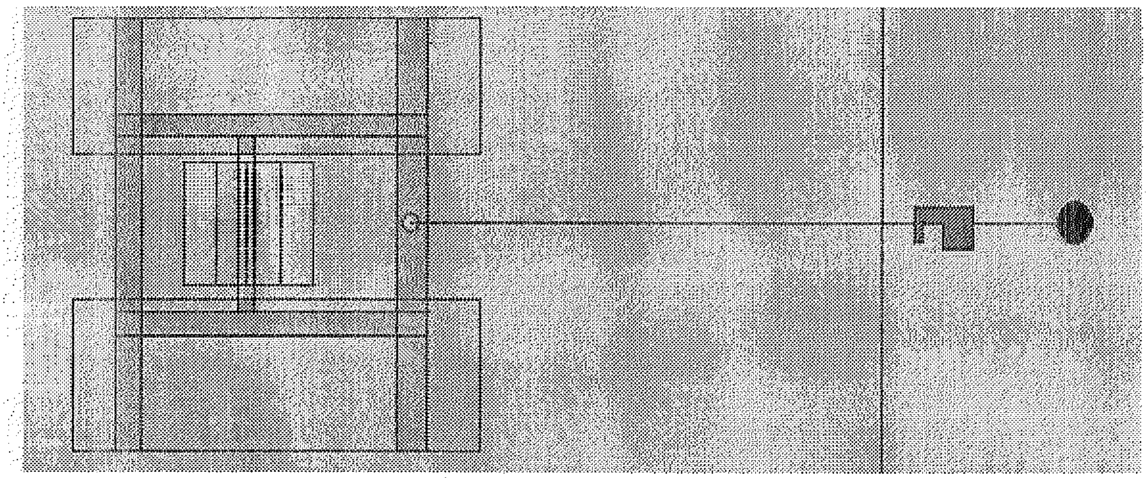
Table 3

Trial	Time (s)	Voltage (V)	Acceleration (m/s^2)
1	100	12.7	0.0026
2	105		0.0024
3	110		0.0021
1	48	25.5	0.0113
2	51		0.0100
3	51		0.0100
1	35	38.11	0.0212
2	36		0.0201
3	35		0.0212

Load Cell Data

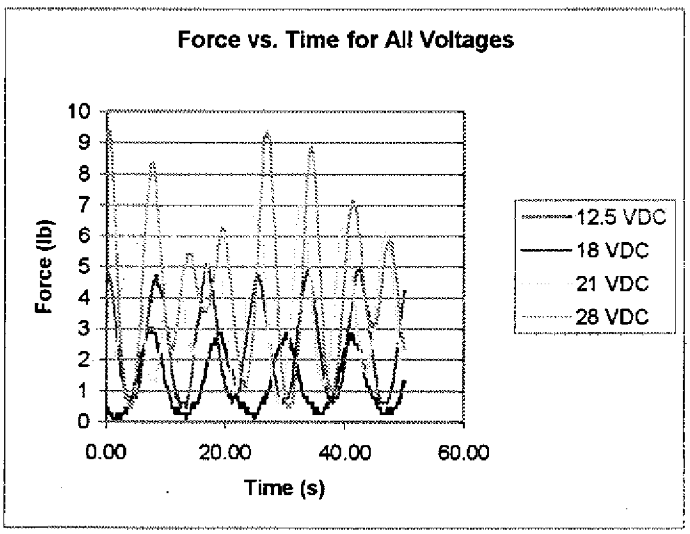
After using a spring scale to determine the amount of force the paddle wheel was producing, it was concluded that another method would have to be used in order to get a better understanding of what was actually happening when the paddle wheel was in motion. To do this a load cell was utilized to accurately predict the forces being generated by the paddle wheel. The load cell was attached to a thin string so that the self-weight of the string could be neglected. One end of the load cell was tied to an anchor and the other was tied to the vessel. Figure 4 illustrates this a little better.

Figure 5 – Load Cell Thrust Measurement



The load cell was then attached to a P3500 strain indicator to produce the necessary full bridge excitation voltages in the strain gages, the indicator was linked to a computer through an analog to digital converter, and data samples were taken at a rate of 10 Hz for 1000 samples. This process was repeated for motor input voltages of 12.5, 18, 21, and 28 volts. The data was used to generate a plot of the force vs. time to illustrate how the paddle wheel is producing a force, because this force is not always constant and actually represents more of a sinusoidal wave. Further analysis of the data was then done using an FFT method to find the harmonic frequencies of the paddle wheel and will be discussed in detail later. A plot of the force vs. time for the four input voltages is shown here.

Figure 6 – Force vs. Time



Fast Fourier Transforms (FFT's)

While obtaining the thrust data it became apparent that the propulsion system was applying a load periodically. This was both visually evident (the string connecting the load cell to the vessel would go from taught to relaxed with almost uncanny repeatability time-wise), and experimentally as the following plot of force vs. time displays.

Figure 9 – Plot of Force vs. Time for 12.5 VDC

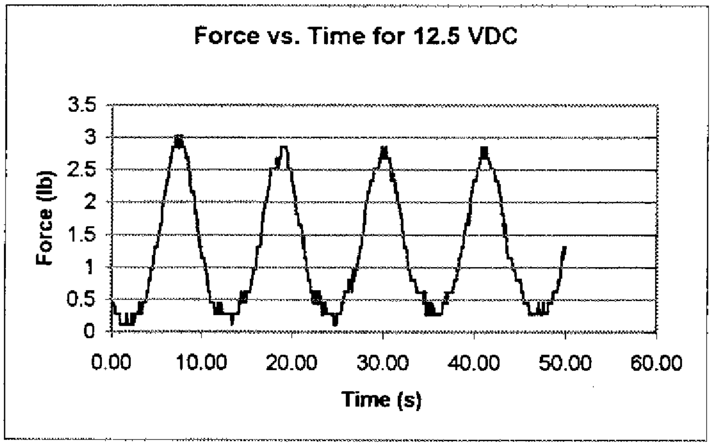
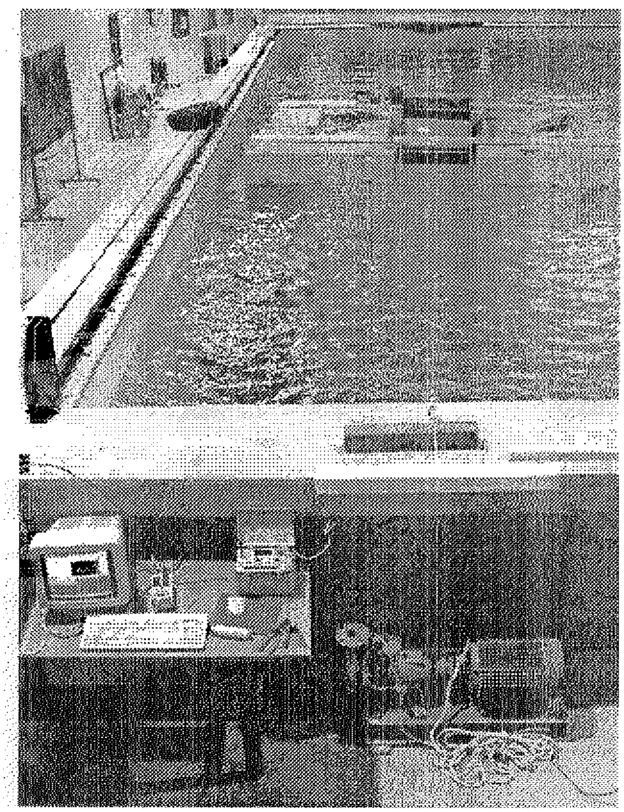


Figure 10 – Load Cell Setup



It was also apparent that at different voltage inputs there were different frequencies of oscillation and also some secondary frequencies involved. In order to find the harmonics of the system it was necessary to perform a Fourier analysis on the obtained data. Recall that the force data was obtained from a load cell at a rate of 10 Hz, fed through a strain indicator-which produced the necessary full bridge excitation voltage, then through an analog to digital converter (A to D) into the computer for data collection. The picture of the actual setup is shown to the right:

Both the strain indicator and the computer had to be put through a lengthy calibration process to determine what the actual readings meant in terms of force applied to the load cell.

The calibration was accomplished by hanging known weights on the load cell and converting these from microstrain on the P3500 strain indicator, and voltage on the computer. By applying several different weights to the system a series of calibration data was obtained, this data was analyzed using linear regression to obtain a function of the load in terms of voltage. This calibration data is shown here.

Table 4/5 – Load Cell Calibration

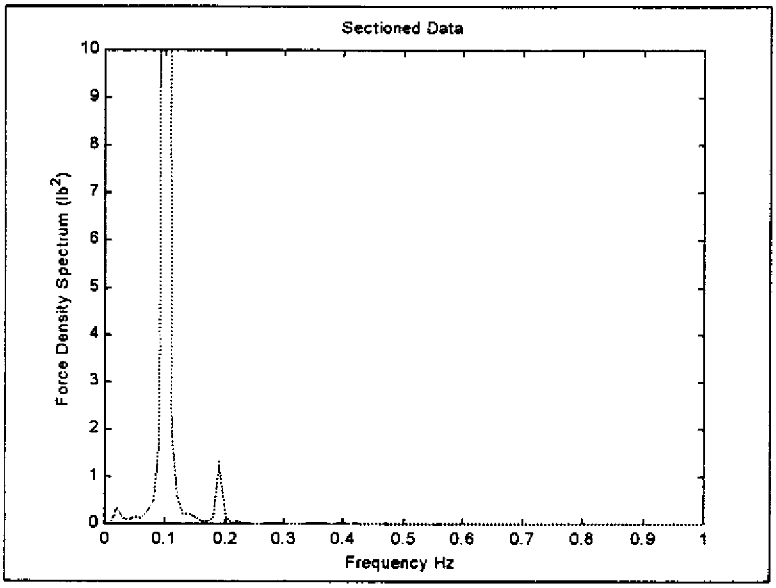
Load Cell Calibration Data					
Line Free Weight =	0.0146(V)	Adjusted			
Free Weight (Basket) =	0.0049(V)	Voltage	Mass (g)	Weight (N)	Weight (lb)
#1 Weighing =	0.0391(V)	0.0342(V)	537	5.268	1.184
#2 Weighing =	0.0635(V)	0.0586(V)	1070	10.497	2.360
#3 Weighing =	0.1052(V)	0.1003(V)	1613	15.824	3.557

Linear Regression Results from 'Load Cell Calibration Sheet'		
Function:	Force = (m)(voltage)+(b)	
where:	M	B
	35.12695	0.106098

After the calibration took place data was collected for four different voltage inputs (12.5, 18, 21, and 28 VDC). This data was converted from voltages to forces utilizing the previous calibration results. An example plot of the force vs. time data was shown on the previous page.

Taking these results it was deemed necessary to find the harmonics of the system for the various voltages. True, it would have been simple to look at the plots and eyeball the period length to get these frequencies, but looking over the range of data there appears to be secondary oscillations mixed in. To break the data down into its basic harmonics Fast Fourier Transforms were implemented, using Matlab to produce harmonic plots. The following is a Matlab plot of the force density data sectioned into frequency bins (after the FFT) for the 12.5 VDC case.

Figure 11 – FFT Results for the 12.5 VDC Case



Attached at the end of this study there is the other three FFT result plots. The following is a table of the harmonics with respect to the input voltages.

Table 6 – System Harmonics for the Various Voltages

Voltage (VDC)		12.5	18	21	28
Harmonics (Hz)	1	0.1	0.15	0.12	0.1
	2	0.2	0.25	0.15	0.16

Also included at the end of this study is the Matlab M files used in the FFT analysis. The file Guu_calc.m was an FFT analysis program created by Dave Fredriksson of the Ocean Engineering department for wave analysis but it hold true for any system displaying harmonics.

References used: [20][21][22]

System Specifications

Motor: Dayton permanent magnet DC motor
Model # 4Z2488
Motor Reference # 3331678-L92G
1/4 Hp
Torque = 9.10 in-lbs @ 1725 rpm
80 VDC, 3.2 A

Gear Box: Model # 42006C
Gear ratio = 10:1
Ref. # 200478CM891

Pulley/Belt Setup:
Pulley Ratio = 3.5:1

Physical Experiment Data

Testing Data

Test 1 Known voltage, measured load

Voltage (V)	Low Load (Lbs)	High Load (Lbs)
12	0	3
16.4	1.5	5
25	4.5	8.5
30	7	11
35	8	13

Test 3 Drag test (constant velocity, 0.33 m/s, pull)

Trial	Force (Lbs)
1	6
2	6
3	5.5
4	6

Test 2 Known voltage & distance (13 m), measured time

Trial	Time (s)	Voltage (V)
1	100	12.7
2	105	
3	110	
1	48	25.5
2	51	
3	51	
1	35	38.11
2	36	
3	35	

Load Cell Data

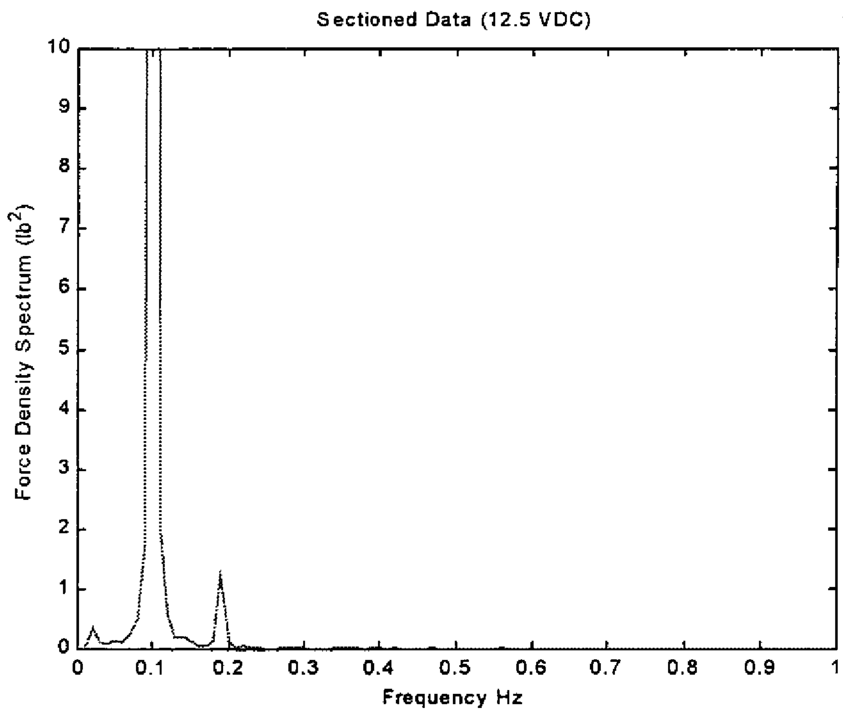
Load Cell Calibration Data					
Line Free Weight =	0.0146(V)	Adjusted			
Free Weight (Basket) =	0.0049(V)	Voltage	Mass (g)	Weight (N)	Weight (lb)
#1 Weighing =	0.0391(V)	0.0342(V)	537	5.268	1.184
#2 Weighing =	0.0635(V)	0.0586(V)	1070	10.497	2.360
#3 Weighing =	0.1052(V)	0.1003(V)	1613	15.824	3.557

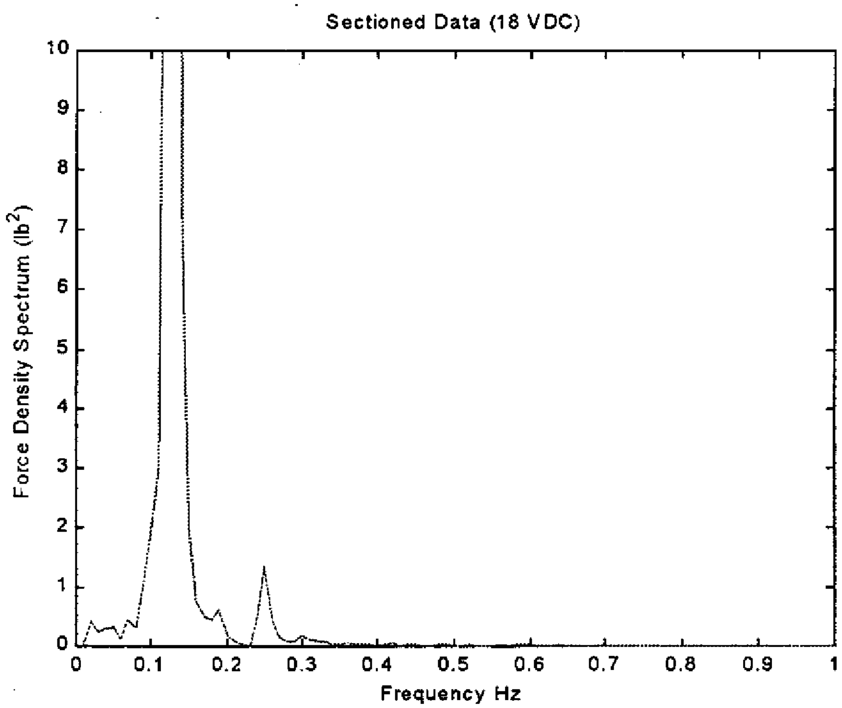
Linear Regression Results from 'Load Cell Calibration Sheet'		
Function:	Force = (m)(voltage)+(b)	
where:	m	b
	35.12695	0.106098

The following is a small Sample of Collected Load Cell Data with analyzed results:

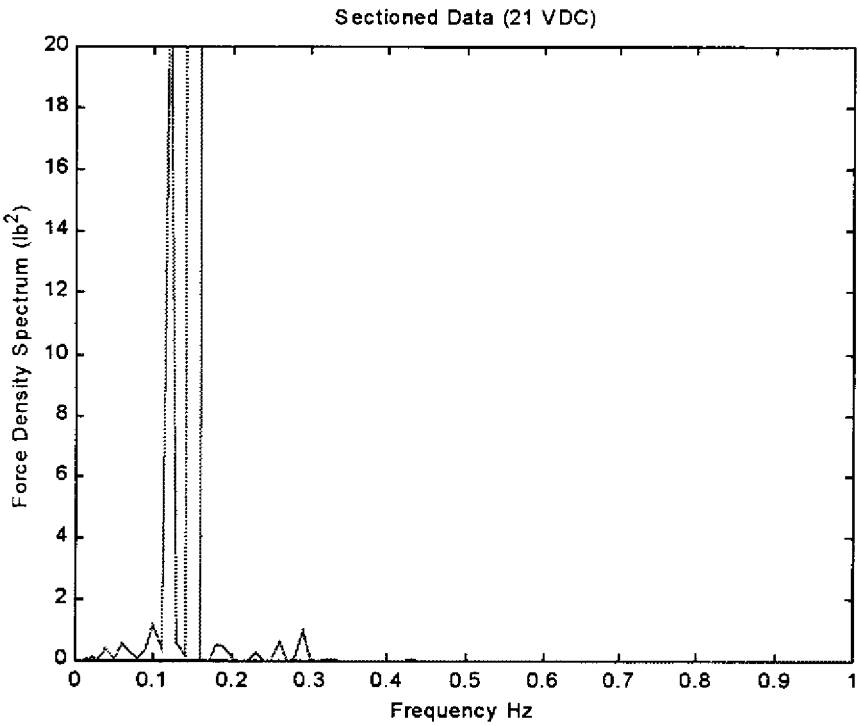
Motor Voltage:	12.5 VDC		18 VDC	
Time	Voltage	Load	Voltage	Load
(s)	(V)	(lb)	(V)	(lb)
0.00	0.0244	0.450342	0.1514	4.911466
0.10	0.0244	0.450342	0.1514	4.911466
0.20	0.0244	0.450342	0.1465	4.739343
0.30	0.0244	0.450342	0.1465	4.739343
0.40	0.0195	0.27822	0.1416	4.567221
0.50	0.0195	0.27822	0.1416	4.567221
0.60	0.0195	0.27822	0.1367	4.395099
0.70	0.0195	0.27822	0.1367	4.395099
	Min =	0.106098	Min =	0.450342
	Max =	3.025148	Max =	5.080075

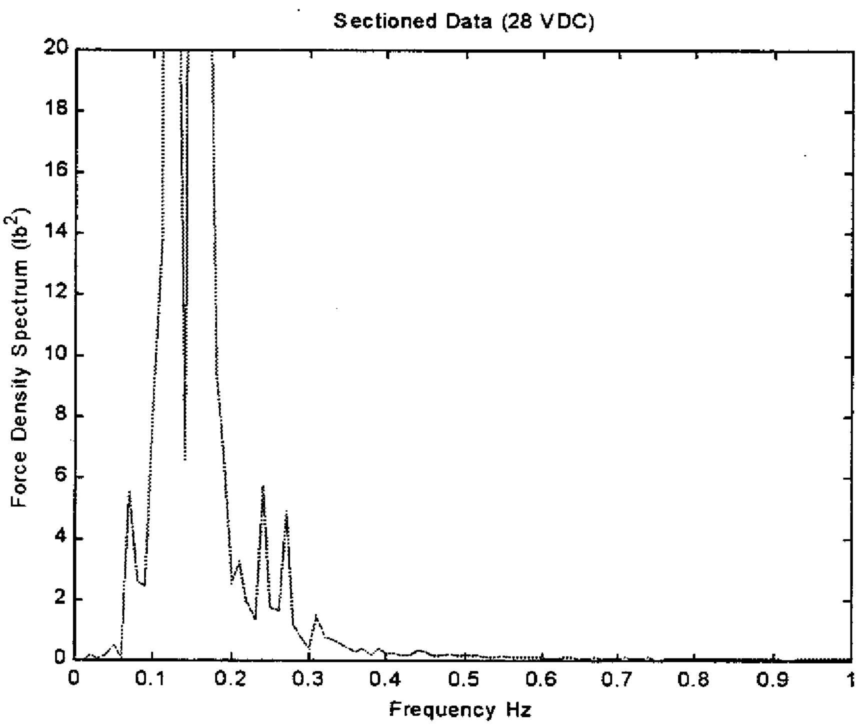
Obtained FFT Plots





Obtained FFT Plots (cont.)





fft125vdc.m – Runs an FFT analysis on the b125vdc.dat file

```
clear   % clears workspace
%NOTE:  a semi-colon suppresses output

load b125VDC.dat
rate = 10;           % sample rate was 10 Hz
num_sections = 10;  % the number of sections to use in the ensemble averaging


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Process the first section of pressure depth data

force = b125VDC;         % this sets pressure equal to ":" -all of, column 3 of
the matrix
force = force(1:1000)';

% Check of data: "Variance Check."
% if the calculations were done properly, the variance of the data should be
the same in both the time and the frequency domain

force_var = var(force)   % This returns the variance of the vector 'force'

% Calculation of the significant force within the data
H_sig_force = 4*(force_var)^.5    % the significant force yielded by this data
(force)

z_force = mean(force);   % this finds the mean force of the load cell -i.e.-
(force)

% Find the raw pressure spectrum using Dave Fredricksson's Guu_calc program
[Guu_force,f_force] = Guu_calc(force,rate);

% Section the pressure time series into 10 sections, find spectra and ensemble
average using dwf program 'section data'
[Guu_ensemble,f_ensemble] = section_data(force,rate,num_sections);


% Check of data: "Variance Check." To see if the calculations were done
properly --> the variance of the data should
%                  be the same in both the time and the frequency domain
% A trapezoidal integration program to calculate the area under the NRG spectra
and hence the variance in Freq. Domain

% Variance Check
var_check_force = 0;
area = 0;
for i = 1:(length(Guu_ensemble)-1)
    area = (f_ensemble(i+1) - f_ensemble(i)) * ((Guu_ensemble(i) +
Guu_ensemble(i+1))/2);
    var_check_force = var_check_force + area;
end

% Prints the frequency domain variance as calculated from the above integral
var_check_force

% Plot the raw and sectioned spectra on top of each other
figure(3);
    plot(f_force,Guu_force,'m')
    title('Sectioned Data (12.5 VDC)')
    xlabel('Frequency Hz')
    ylabel('Force Density Spectrum (lb^2)')
    axis([0,1,0,10])
```

Guu_calc.m – Dave Fredriksson's FFT analysis file

```
function [Guu,f] = Guu_calc(data,rate)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% This function calculates the fft
%% function [Guu,f] = Guu_calc(data,rate)


% Removes the linear trend or mean value.

t2 = detrend(data);
%n2 = length(data);
n2 = max(size(data));
%n2= 512;
%n2= 1024;
%n2 = 4096;
%n2 = 8192;
%n2=16384;

%w = blackman(n2);
%t2 = t2.*w;


% Take the FFT to identify discrete frequency components.

y2 = fft(t2,n2);

% The Power spectral density, a measurement of the energy at
% various frequencies.

Guu = 2*y2.* conj(y2)/(rate*n2);

% Formation of the frequency axis using "rate" Hz (sampling rate)

f = rate*(1:n2)/n2;
%f2= 1/f2;

b1 = round(length(Guu)/2);
b2 = round(length(f)/2);

Guu = Guu(1:b1);
f = f(1:b2);

%box = 50;
%j = 1;
%i = (1:box:450);
%for k = 1:450;
%   f(j) = mean(f(k:k+box));
%   Guu(j) = mean(Guu(k:k+box));
%   j = j + 1;
%     k = k + 1;
%end
%clear i
```