Automatic Clownfish Hatchery

By: Laura Quesada

ELECTRICAL ENGINEERING DEPARTMENT

California Polytechnic State University

San Luis Obispo

2011
# Table of Contents

## Acknowledgments

## I. Introduction

## II. Background

### Requirements

- **A. Main Tank**
- **B. Larvae Kreisel Tank**
- **C. Phytoplankton System**
- **D. Rotifer System**
- **E. Pump and Light Management**
- **F. Optional Upgrades**

## III. Design

- **A. Automatic Clownfish Hatchery System Design**
- **B. Tank Design**
- **C. Sensors**
- **D. Communication**
- **E. Circuit design**
- **F. PCB design**

## IV. Construction

- **A. Tank Construction**
- **B. PCB Construction**

## V. Testing
**List of Tables and Figures**

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Clownfish Lifecycle (&quot;Fish Life Cycle.&quot;)</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Nannochloroposis Phytoplankton (&quot;Reefs.org&quot;)</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>Rotifers Under 100x Magnification (&quot;Reed Mariculture - Instant Rotifers.&quot;)</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>Example of a Kreisel Tank (&quot;Monterey Bay Aquarium&quot;)</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>Top Level System Design</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>Microcontroller Connections</td>
<td>16</td>
</tr>
<tr>
<td>7</td>
<td>Geometric Kreisel Design</td>
<td>20</td>
</tr>
<tr>
<td>8</td>
<td>Phytoplankton Tank Design</td>
<td>21</td>
</tr>
<tr>
<td>9</td>
<td>Example of a Dual Voltage Regulation Circuit</td>
<td>24</td>
</tr>
<tr>
<td>10</td>
<td>Example of a Single Inductive Load Driver Circuit</td>
<td>25</td>
</tr>
<tr>
<td>11</td>
<td>Example of a Single Supply Inverting Amplifier</td>
<td>26</td>
</tr>
<tr>
<td>12</td>
<td>Second Order Butterworth Active Filter</td>
<td>26</td>
</tr>
<tr>
<td>13</td>
<td>Example of a Relay Circuit</td>
<td>27</td>
</tr>
<tr>
<td>14</td>
<td>Laser Cutter Cutting Acrylic</td>
<td>28</td>
</tr>
<tr>
<td>15</td>
<td>Kreisel Tank</td>
<td>29</td>
</tr>
<tr>
<td>16</td>
<td>Laser Cutter Template for the Rotifer Tank</td>
<td>29</td>
</tr>
<tr>
<td>17</td>
<td>Rotifer Tank</td>
<td>30</td>
</tr>
<tr>
<td>18</td>
<td>Phytoplankton Drilling (left) and System (right)</td>
<td>30</td>
</tr>
<tr>
<td>19</td>
<td>Removing Toner from PCB’s</td>
<td>32</td>
</tr>
<tr>
<td>20</td>
<td>Tinning of PCB’s</td>
<td>32</td>
</tr>
<tr>
<td>21</td>
<td>Example Hemocytometer Slide and Grid</td>
<td>35</td>
</tr>
<tr>
<td>22</td>
<td>Green Led Density vs Green LED Reading</td>
<td>36</td>
</tr>
</tbody>
</table>
Figure 23: Density of Rotifers Using the Red Reading from the Color Light Sensor

Figure 24: Density of Phytoplankton Using the Green Reading from the Color Light Sensor

Figure 25: Micro Diaphragm Pumps

Figure 26: Flow rate of the 12VDC Micro Diaphragm Pumps

Figure 27: Completed Circuit System
Acknowledgments

I would like to thank the following people for their help and support through the process of my senior project. To John Oliver, my senior project advisor, who was there for me and helped keep me on track. To Jaime Carmo, who played a major role in the tank manufacturing process. To my parents, Marco and Colleen Quesada, who helped me financially and emotionally. To Joseph Rye, the love of my life, who supports me in everything I do and was an integral part of manufacturing my printed circuit boards. Without them my senior project would not have been successful.
I. Introduction

Marine fish are a passion of mine and I have always wanted to breed them, but unlike freshwater fish they are much more difficult and time consuming. Marine larvae are too small to accept prepared foods when first hatched. The most common food given to the larvae before metamorphosis are rotifers, microscopic zooplankton that must be grown and are fed by phytoplankton. This phytoplankton must also be cultured and grown to maturity while being kept at a constant density in order to keep the rotifers alive.

For my senior project I created an automatic clownfish hatchery. I chose clownfish because they are demersal spawning fish, easily make pairs and spawn regularly in a reef aquarium. Demersal spawning fish live and lay eggs near the bottom of the ocean. Clownfish live in coral reefs and lay eggs in a defined area where they can be guarded until hatching occurs. They produce adhesive eggs that attach to a solid surface such as a rock near their host anemone or coral which makes the eggs easy to transfer to a rearing tank. This tank will be designed to automatically raise the larvae through metamorphosis and simulate the environment found in the ocean.

To make the hatchery automatic, I used microcontrollers, environmental sensors, and micro pumps. The design is made up of three different systems, the main tank where the larvae will be held, the rotifer culture system, and the phytoplankton culture system. During this project I designed, built, and tested these systems in order to create an automatic hatchery that should raise clownfish through metamorphosis.

Most marine aquarium fishes available to the consumer are wild-captured. Over collecting in areas that are not regulated is stresses coral reefs. Breeding fish in a home environment has a potential to alleviate some of this stress, not to mention tank bred fish are hardier and easier for a home aquarist to take care of.
II. Background

Over 90% of freshwater ornamental fish and invertebrates are commercially raised and very few are taken from the wild. On the other hand a very high percentage of marine ornamental fish and invertebrates come from the ocean and few are tank raised or tank bred (Lass 1). Clownish are one of the types of marine fish that can be bred in captivity and are a good starting point to learn how to breed.

There are three types of clownfish available to the public. Wild clownfish, tank raised clownfish and tank bred clownfish. Wild clownfish are taken from the ocean reefs. They historically have a higher than average mortality rate after transport due to the stress of capture (Wittenrich). These fish are always in demand so they are constantly collected but are not always sustainably. In some countries fish are taken in such great numbers that they disturb the natural balance, collectors sometimes use explosives to stun the fish before capture, and there are some divers that use cyanide poison to stun fish before they are collected. This cyanide poison kills 90% of fish that live in the reef as well as any coral it touches. The fish affected by this poison often last long enough to make it to market and be sold before they die (“Coral Reefs - Ocean World”). These unsustainable practices for collecting wild fish are killing the oceans reefs.

Tank raised clownfish are not bred in captivity, they are caught as settlement stage larva and raised to market size (Wittenrich). The problem with this type of aquaculture is that it is unknown what type of fish, invert, or other marine animal is caught. Many unwanted marine animals are raised in this type of operation. This is not only wasteful, but it also takes settlement stage larva out of the ocean.

Tank bred clownfish are clownfish that have been laid, hatched, and raised in an aquarium. They tend to be easier to acclimate to a home aquarium, have fewer parasites,
get fewer diseases, and they are sustainable. Clownfish have been tank bred for over 40 years. They are indistinguishable in behavior and morphology from clownfish in nature unless bad breeding procedures are followed and the fish are inbred to the point of mutation. Sometimes these mutations are exploited in the coloring or pattern on the fish but mutations can also cause deformity and other unwanted issues such as poor immune systems or gills that don’t adsorb oxygen as well as they should.

There are a couple of different ways clownfish are bred in captivity. Clownfish are bred both by hobbyists and large aquaculture farms. On any scale, breeding clownfish is an enormous task. This includes starting and maintaining cultures of phytoplankton and zooplankton. The phytoplankton must be fed to the zooplankton and the zooplankton must be fed to the clownfish often enough so both zooplankton and clownfish larvae never go hungry. Some hobbyists only breed a single spawn at a time while fish farms breed and raise huge numbers of clownfish every day and have facilities dedicated to the task.

Whether setting up in an empty room or on a giant facility every breeder needs four key elements. The first element is a mated pair of clownfish that regularly lay eggs. Next is the broodstock tank where the breeding clownfish will live, a larval rearing tank, and an area for live food culture and enrichment.

Clownfish begin life as egg then hatch into larvae. These larvae are usually around 4mm in length and look more like tadpoles than actual fish. In the next couple of weeks the clownfish will change drastically including the body structure, internal organs, and coloring (Wittenrich). These changes can be seen in figure 1 on the next page.
When the eggs arrive they can be taken out of the broodstock tank and placed in the larvae rearing tank hours before hatch or they can be allowed to hatch in the broodstock tank and carefully scooped out and released into the larvae tank. Once in the larvae tank, the young clownfish need to be fed.

After hatching clownfish need to eat constantly due to their rudimentary stomachs. In the wild they float out into the open ocean and feed on plankton until they reach metamorphosis. To re-create this artificially phytoplankton and zooplankton must be provided at all times to the larvae clownfish. The tank needs to be stocked with 3-15 zooplankton and 2 million to 5 million phytoplankton cells per ml.

Hobbyists usually feed the larvae tank anywhere from 5 to 12 times per day to maintain proper densities while the pond technique as described above is often used for commercial farm.
One popular technique to raise larvae fish is the greenwater method. In this method phytoplankton is used to turn the rearing tank a light green. This phytoplankton acts as the food for zooplankton such as rotifers when they are introduced to the tank. It also acts as a contrast between prey organisms and the background and helps to maintain water quality by consuming dissolved wastes in the larvae tank (Wittenrich).

For commercial scale fish farms the greenwater technique is also used but instead of supplying the tank with phytoplankton they use outdoor concrete ponds and wait for an algae bloom then add zooplankton. Larvae fish are then released into the pond and prey on any available organisms. This gives little control over the larva growth process.

Phytoplankton are tiny, free floating, photosynthetic organisms about 2-12 microns in diameter. These phytoplankton are used as food source for zooplankton. They also help keep the water in the main tank clean by absorbing dissolved wastes. The usual phytoplankton used for raising clownfish is nannochloroposis for its nutritional value (Hoff). An example of various densities of nannochloroposis plankton can be seen below in figure 2 with the densest culture on the left.

![Figure 2: Nannochloroposis Phytoplankton ("Reefs.org")](image)
Zooplankton are microscopic multicellular invertebrates that range from 150 to 500 microns in length. These zooplankton are the next step in the food chain and are fed directly to the larvae clownfish (Hoff). Rotifers are the usual zooplankton used in raising clownfish and can be seen below in figure 3.

![Figure 3: Rotifers Under 100x Magnification ("Reed Mariculture - Instant Rotifers.")](image)

One way people keep larvae fish safe is the use of a Kreisel tank. It was originally designed to keep pelagic jellyfish alive in captivity yet can be used to keep larval fish and prey organisms confined by the water flow in the middle of the tank reducing the potential for contact with the tank walls and the surface tension of the water (Wittenrich). Figure 4 below shows an example of a Kreisel tank used for jellyfish.

![Figure 4: Example of a Kreisel Tank ("Monterey Bay Aquarium")](image)
Requirements

A. Main Tank
The main tank is where the clownfish larvae will be held. In order for the larvae to survive they need to have a clean environment with adequate food, lighting, and water chemistry.

1. Capacity
   - The main tank will have a 14 gallon capacity. This is enough water to maintain good water quality while allowing the tank to be stocked with an adequate density of prey organisms.

2. Materials of Construction
   - The tank will be built out of glass, silicone, and plastic (store bought). All of these materials are safe for marine fish and easily obtained.

3. Stocking
   - The main tank will be stocked with 3 -15 rotifers and 2 million - 5 million cells of phytoplankton per mL of water. These densities of plankton will allow for an adequate number of prey organisms for the larvae with high nutritional value.

4. Lighting
   - The lighting will be power compact 50/50 6500k/420. This is a standard type of light for marine aquariums.
   - Lighting will be on 8 hours out of the day.
5. **Pumps**
   - Micro water pumps will have a flow 10 mL/min - 400 mL/min into the tank. This will be slow enough to allow for a controlled amount of liquid whether it is plankton or chemicals to be introduced to the tank.
   - Micro water pumps will be either a diaphragm or peristaltic design. These two pump designs will not damage the plankton.
   - There must be a circulation pump in order to evenly distribute the plankton and chemicals.
   - Air pump must have enough flow to produce air bubbles through an air stone at the bottom of the tank.

6. **Filter**
   - The main tank will have chemical, biological, and mechanical filtration in order to maintain the water quality at optimal levels.
   - Chemical filtration will be with the use of the media Chemi-Pure.
   - Biological filtration will be through 14 lbs of live rock and the phytoplankton in the water.
   - Mechanical filtration will be through the use of a protein skimmer.

7. **Water Chemistry and Temp**
   - The Ph will be kept within a range of 7.9 and 8.3. This is the optimal range for marine aquariums.
   - A constant slow water change will be used for water quality purposes.
8. Sensors
- Sensor(s) will be used to provide an approximate density of both rotifers and phytoplankton.
- A pH sensor will be used to measure the pH in the main tank.
- These sensors will relay to the microcontroller.

B. Larvae Kreisel Tank
The Kreisel tank will hang over the side of the main tank and the larvae will be held here until they reach metamorphosis.

1. Capacity
- The Kreisel will have a 1-2 gallon capacity. This is large enough to hold the larvae clownfish until metamorphosis.

2. Materials and Construction
- The tank will be built out of acrylic or PVC plastic. Both of these materials are safe for marine fish and easily manufactured.
- The back and sides of the tank will be constructed out of black or opaque material and the front will be constructed out of clear or transparent material. The larvae need very little light levels and see prey organisms better against a dark background.
- It will be of a Kreisel design. This design will keep the larvae safe from getting sucked into filters, stuck in corners, or trapped in the surface tension of the water.
3. Stocking  
- The tank will have the same water as the main tank circulating through so it will have the same density of rotifers and phytoplankton.

4. Lighting  
- Lighting will be minimal, only ambient light from the top and front will be allowed.

5. Pumps  
- A small sump pump will be used to circulate the water around the Kreisel.  
- The flow will be adjustable by a valve.

6. Water Chemistry  
- The tank will have the same water as the main tank circulating through so it will have the same water chemistry.

7. Filtration  
- Filtration will take place in the main tank.

C. Phytoplankton System  
The phytoplankton system will continuously grow live phytoplankton until it is fully developed and able to be fed to zooplankton (rotifers) or used for the greenwater method in the main tank.

1. Capacity  
- The phytoplankton system will have 4+ liters of capacity. This gives a large enough volume of water for stocking the main tank and feeding the rotifers.
2. **Materials of Construction**
   - There will be 4+ containers for the phytoplankton.
   - These are the stages; each one will increase in phytoplankton density till the culture is fully developed.
   - The material will be acrylic.
   - The design of each container will be cylindrical so the most amount of light will reach the plankton.

3. **Stocking**
   - 20 - 40 million cells of phytoplankton per mL for a fully developed phytoplankton culture.
   - Each stage will increase in density till this number is reached.

4. **Lighting**
   - 2ft Compact Florescent tube around 6000k will be used for optimal phytoplankton growth.
   - Lighting will be on 16 hours per day.

5. **Filter**
   - No filtration is needed for the phytoplankton system. The plankton will move through the system in about 7 days so the water quality will stay up.
D. Rotifer System

The rotifer system will continuously grow a fully developed culture of live rotifers to feed to larvae clownfish in the main tank.

1. Capacity
   - The rotifer system will have 4+ liters of capacity which is enough volume to be able to stock the main tank.

2. Materials of Construction
   - There will be 1+ containers for the rotifers.
     - If multiple containers are used it will be due to redundancy, just in case one of the system fails.
     - The material will be acrylic plastic.
     - The design will be an upside down pyramid so all the dead rotifers as well as any waste will fall to the bottom where it can be easily sucked out by a pump during water changes.

3. Stocking
   - There will be 200-500 rotifers per ml of water for a fully developed rotifer culture with 2 million to 5 million phytoplankton cells for the rotifers to feed on.

4. Lighting
   - Lighting will be provided by ambient light.

5. Filter
   - No filtration is needed for the phytoplankton system.
   - Water changes will be made every four days for water quality purposes.
E. Pump and Light Management

The pumps will be managed by microcontrollers using sensor inputs in order to add or subtract water from the different tank systems as well as turn on and off the lights and air pumps.

- Microcontroller will turn on and off the pumps depending on information obtained from the sensors.
- Microcontroller will turn on and off the lights and air pumps.

F. Optional Upgrades

These are some upgrades to the system that will improve upon the previous requirements.

- LCD screen integration.
- Two operations available for user control.
  - One operation is for cultures when no larvae are present in order to keep cultures alive and growing.
  - Another operation is for larvae feeding.
- Temperature sensor will be installed, displaying a warning on the LCD screen if water temperature drops below 77 degrees or above 81 degrees.
- Baby Brine shrimp container will be added for better nutrition.
III. Design

There are three main systems in the Automatic Clownfish Hatchery: the phytoplankton system, the rotifer system, and the main system. These systems are connected through a series of pumps controlled by a microcontroller network using the data collected from several sensors and a real time clock. Figure 5 below shows a top level system design.

A. Automatic Clownfish Hatchery System Design

Figure 5: Top Level System Design
The color light sensor is used to sense the density of both rotifers and phytoplankton in the main tank. The green LED sensor is used to sense the density of phytoplankton in the rotifer tank. Both these sensors, with the real time clock control the pumps in the system to keep the tanks stocked with the correct density of plankton, perform water changes, and keep the plankton alive.

Figure 6: Microcontroller Connections
As seen in figure 6 on the previous page, four different microcontrollers were used in the Automatic Clownfish Hatchery. Four microcontrollers were used due to the modular design of the device. These devices were chosen due to familiarity, cost, availability, on hand supply, and the correct I/O distribution. Memory was also a factor for choosing the main board. Atmel microcontrollers were exclusively used in the design. This is mostly due to familiarity with the chips yet PIC, NXP, or other microcontrollers with the correct pins and memory could have been used.

Two ATtiny2313’s was chosen for the motor driver boards because they are very inexpensive chips and these boards do not need much memory space. These chips were already on hand and convenient, yet any chip with 11 I/O, a UART channel, and SPI could have been used.

The ATmega48A was used in the ADC board because of its available ADC channels. These chips although slightly more expensive than the ATtiny’s are also inexpensive. An I²C ADC chip could have been used instead of the ATmega48A for the ADC board and would have made for a better design. Even though it would be more expensive there would be no processing or programming overhead and the chip could have been easily added to the I²C of the main board. The requirements for this board were 6 external ADC channels, a UART channel, and SPI. Other microcontrollers that met these requirements could have been used.

The main board was originally designed with an ATmega8A yet there was not enough memory on the chip to run all the programs. This chip was replaced with a pin compatible ATmega328P. These two chips are exactly the same but the 328P has more memory. Another alternative would be to use the 168A since it is the next step up in memory from the 8A, but this chip was not available. The main board needed a chip with I²C, a UART channel, SPI, and at least 9k memory.

If the project were designed to be all on one board, only two chips would be needed. Depending on the programming consolidation the whole thing could be
designed with two ATmega8A’s. Both chips would be needed for the availability of different types of pins (ADC, TWI,...).

B. Tank Design

This project has three main tank systems: the main tank, the rotifer system, and the phytoplankton system. Each of these tank systems needs to be designed in order to provide the best environment for its inhabitants.

1. Main Tank

There are many different types of tanks that can be used to raise larvae clownfish yet some are better than others. A standard glass aquarium is not the best candidate due to the poor water flow in the corners where the larval fish can get stuck, the open top where the larvae can get trapped in the surface tension, and the reflective walls which will confuse the larvae and keep them from searching for food.

Many aquarists use black plastic tubs with rounded corners. This alleviates the problems with the corners and reflection but not the problem with water surface tension. In order to keep the larvae safe from all these problems a tank design such as a Kreisel would be needed.

Another thing to consider when building a larvae tank is the size. If the tank is small it will be hard to maintain the water quality yet easy to maintain the correct density of plankton. A larger tank will have better water quality, but the amount of plankton for correct densities might be so great that it is unattainable.
A 10 gallon tank is probably the smallest tank that can be used while keeping a proper water quality.

For this project a 14 gallon standard rectangular glass tank was used for the main tank, yet this is not were the larvae are held due to reasons stated above. Hanging over the main tank is a Kreisel, this tank will keep the larvae safe, and the 14 gallon will give enough water volume to keep the water quality up. The 14 gallon tank also allows for external filtration such as a protein skimmer and live rock. The design could have been done with a 10-14 gallon Kreisel, yet without the filtration water quality would have decreased.

2. Kreisel

The Kreisel tank is a design that was developed in Germany. Ideally it is a circular tank with no corners yet without a way to bend acrylic accurately a heptagon tank was design was created. This design can be seen in figure 7 on the next page. For this design to work, an inlet of water needs to come in the upper left corner of the tank. This water flow needs to be uniform across the depth of the tank. To do this a spray bar was used. The water was directed at the bottom left where it then hits the angled corners around the tank and ends up back at the inlet.

The outlet is cut out of the side of the tank. The larvae will not get sucked into this outlet due to the greater size than the inlet, and the pressure of the spray bar flowing over the outlet.
3. Phytoplankton

The phytoplankton system design was created to keep a continuous culture. For years 2 liter plastic bottles were used to culture plankton. After about 7 days the contents of the fully cultured bottle are split and new cultures are started. This design could have been implemented, yet it would have involved more pumps, would have had more components, and would be hard to run continuously.

The final design was a series of cylindrical containers that were attached by tubes about 1/2 inch down from the top of the container seen in figure 8 on the next page. The cylindrical design has been proven to be the best shape for culturing phytoplankton. Each container decreases in height by 1 inch so that gravity can move the phytoplankton across the containers. The first and last
containers have overflows in case there is not enough phytoplankton pulled out. Any excess phytoplankton can be fed to coral in the fish tanks.

![Figure 8: Phytoplankton Tank Design](image)

4. Rotifers

For the rotifer design two upside down pyramid tanks were used. This tank was originally designed for the larvae, yet was found to be inadequate for the larvae but effective for the rotifers. The design was implemented due to the fact that any dead organisms or debris will fall into the bottom where they can be easily extracted using a pump therefore keeping the water quality up.

The drawbacks to this design are that the volume of the tank is decreased by 1/3 and it is very difficult to construct. This design would have been near impossible to create on a table saw or other manual power tool, yet the use of a CNC laser made the design feasible. The laser templates for this tank can be seen in figure 16 page 29.

An overflow tube was put into the design for two reasons. It acted as an overflow just in case the pumps or programs malfunction, but it also gave a place
for float switches to be mounted. These two float switches were used to keep the tank topped off and to perform water changes.

C. Sensors

Using two green LED’s with a 565nm wavelength it is possible to get a rough estimate of the phytoplankton. One LED is used as a regular LED while the other is used as a photodiode. This works because the diode has been doped for a specific wavelength and if used as a photodiode it will be sensitive to the same wavelength. Since green LED’s were used, the phytoplankton both absorbs the light through the chlorophyll and blocks the light. The denser the phytoplankton is the smaller the voltage the phototransistor outputs. The green LED’s were chosen over the IR sensor due to the increased sensitivity to density change within the phytoplankton.

The color light sensor Avago ADJD-S371-Q999 can detect even the smallest changes in visible color. This color light sensor is sensitive to varying degrees of white light and is split up into quadrants with red, green, and blue filters as well as clear (this quadrant has no color filter). By changing the integration time the sensitivity of each quadrant can be adjusted. For this application only green and red values were used. This covers the green phytoplankton as well as the brown zooplankton (rotifers) used.

D. Communication

Universal Serial Asynchronous Receive and Transmit (USART) was used to communicate between the master controller and the slave controllers. Due to the slow speed, it minimizes signal losses and maintains signal integrity without any special design considerations.
Only one USART channel exists on the Atmega328P therefore a USART hub was created in order to communicate between the three slave devices. The slave devices were given addresses so the master board could select which device to communicate with.

Two wire interface (I²C) was used for both the real time clock and the color light sensor. I²C is the native protocol for both these devices and because of this both devices were connected through the two lines (SDA and SCL). This can be seen in figure 6 on page 16.

E. Circuit design

1. Linear Regulator Circuits

To provide the required voltages, each board contains two of the following linear voltage regulators: LM317 (adjustable), LM1117-5, and LM1117-3.3. The voltage supplied to each board is 12V generated from a switching power supply with maximum current rating of 5 amps. An example of a linear regulator circuit can be seen in figure 9 on the next page.

The LM317 linear voltage regulator was chosen because it is an industry standard inexpensive adjustable voltage regulator. The output filter capacitor on the LM317 is a tantalum type capacitor for its low equivalent series resistance (ESR). This is because the LM317 has stability issues with high ESR filter capacitors.

The LM1117 linear voltage regulators were chosen based on cost. These regulators do not require the more expensive tantalum capacitors and were used on the circuits with standard digital voltages.
On the motor driver boards, the LM317’s are bypassed allowing the pumps to operate at 12V (rated voltage). Originally the LM317 was used to adjust the voltage up 22V for a 24V supply. This was because the original pumps were rated at 24V. The LM1117-5 is used to supply the 5V required by the microcontroller.

The ADC board has a LM317 adjusted to 7V for the opamps as well as a LM1117-5 for the microcontroller.

The master board has an LM1117-5 for the microcontroller and a LM1117-3.3 for the color light sensor and the real time clock.

Figure 9: Example of a Dual Voltage Regulation Circuit

2. Inductive Load Driver Circuits

The Inductive load driver circuit can be found the motor driver boards. Each inductive load driving circuit consists of one n-channel mosfet, a schottky diode, and an LED indicator. There are six of these circuits on each motor driver board and each one corresponds to a single pump. A single inductive load driver can be seen in figure 10 on the next page.
When the microcontroller provides 5V to the gate of the mosfet, the mosfet allows current to pass from VCC to ground through the pump motor effectively turning the pump on. When the 5V signal is removed, the pump motor generates a back EMF which is dissipated through the schottky diode. The LED indicates when the motor is on for an easy visual conformation even if the pump is not attached.

3. Single Supply Inverting Amplifier

In order to supply a 0-5V sensor input to the ADC, a single supply inverting amplifier centered at 2.5V was implemented. An example of this can be seen in figure 11 on the next page. This type of amplifier is used on the ADC board.
4. **Active Filter**

A second order Butterworth active filter seen in figure 12 was simulated and implemented yet was not tested before implementation. For reasons still unknown the filter did not work properly and was physically cut out of the PCB.

5. **Relay Circuit**

The relay circuit seen in figure 13 on the next page is used to interface between the microcontroller and 120V devices. The microcontroller supplies 5V to the biasing resistor of the BJT which induces a current through the coil of the relay, this toggles the state of the normally open (NO) contact allowing current to flow from hot to neutral through the 120V device. The LED turns off when current is flowing through the desired device.
F. PCB design

All PCB’s were designed using EagleCAD lite. EagleCAD lite is a free PCB layout program which is limited to 100mm by 160mm. For this reason the design was split into many boards which consisted of one master board, several slave boards, and a few additions. These boards include: ATmega328P master board, master daughter board (extra 22 inputs), two ATtiny2313 motor driver boards, motor driver daughter board (extra outputs), motor driver current adapter, ATmega48A ADC board, UART board, relay board, LED driver board, and the power supply board. Some of these boards could be consolidated and with more powerful layout software the design could have been consolidated to a single board utilizing two ATmega 328P’s.
IV. Construction

A. Tank Construction

The Kreisel, rotifer tank, and phytoplankton tank were all constructed out of acrylic and put together using acrylic glue as well as silicon. These tanks were either cut with a table saw, horizontal bandsaw, or CNC laser. Any holes were cut with a drill press or CNC laser. Figure 14 below shows the CNC laser cutter cutting out small acrylic parts.

![Figure 14: Laser Cutter Cutting Acrylic](image)

The Kreisel was built out of both black and clear acrylic. The black acrylic made up the back, sides, and top of the Kreisel while the clear acrylic made up the front. This can be seen in figure 15 on the next page. The black acrylic was purposefully textured to get rid of any reflective properties so the babies would not get stuck looking at their own reflections. Each piece was cut on a table saw and put together using acrylic glue. A square was cut out of the left panel to act as an output for the water. A hole was drilled into the upper left hand corner of the back panel for an input of the water. For the input an acrylic tube was drilled with many holes to create a spray bar and was inserted into the hole in the back panel and secured in by gluing around the hole and to the front clear panel. This creates the current around the Kreisel so the babies will be corralled to the middle of the tank.
The Rotifer tank was built out of clear acrylic. All the pieces were cut using the CNC laser and put together using acrylic glue with silicone on top to waterproof it. One 3/8 inch hole was cut in each upside down triangle tank about 3/4 of the way down from the top of the tank for the overflow. The overflow was created out of 3/8 inch diameter acrylic tubing. This tubing was bent using a heat gun in order to achieve a vertical orientation at the top of the tank. Two mounting bars were created and glued to the overflow to hold the float switches. The completed rotifer tank can be seen in figure 17 on the next page. The rotifer tank template for the laser cutter can be seen below in figure 16.
The Phytoplankton tank was made of 5 chambers. These chambers were created out of 3 inch diameter acrylic tubing cut using a horizontal bandsaw. Each chamber stepped down in height by 1 inches and a hole was cut into both sides of all three middle chambers and on one side of the two outside chambers that correspond to the next chamber so a gravity feed could be used. The caps were cut out of acrylic sheet using the CNC laser. Plastic tubes were added through the caps to allow air to flow into the chambers. The phytoplankton system can be seen in figure 18 below.
B. PCB Construction

The toner transfer method was used in the manufacturing of all the PCB’s. These steps include:

1. Printing of artwork

   The artwork is printed onto glossy photo paper using a toner based laser printer. If the board is double sided alignment markings must be present on the artwork.

2. Clean copper clad FR-4

   Using a clean green scouring pad and hot water thoroughly clean the surface of the copper.

3. Heat transfer artwork to copper

   Place the artwork on top of the newly cleaned copper board. Using heat and pressure go over every square inch of the photo paper several times with a hot iron. Once completed allow ample time for the board to come to room temperature.

4. Paper removal

   To remove the paper use hot soapy water and a toothbrush while being careful not to disturb the toner artwork.

5. Etching

   After all the paper is removed rinse and dry the board. Using nitril gloves, pour ferric chloride into a non-reactive container and immerse the prepared board. Fresh ferric chloride should take 30min to 1hour to fully etch a board. Monitor to make sure the copper is being removed properly.

6. Drilling

   Using a drill press and proper sized drill bits, drill all the necessary holes into the board.
7. Toner Removal

Remove toner using a fresh pair of nitril gloves and acetone. Figure 19 below shows a circuit board with half the toner removed.

![Figure 19: Removing Toner from PCB’s](image)

8. Tinning (optional)

Pour a small amount of liquid tin into a non-reactive container and immerse the clean etched board. Leave the board in the solution for 3 to 5 min for adequate plating. Figure 20 below shows the tinning process.

![Figure 20: Tinning of PCB’s](image)

9. Trimming

Trim the board to size. To do this you can use a dremel or table saw.

10. Soldering

Solder all components onto the board.
Many of the chemicals and materials used in this process are highly toxic so it is important to take proper safety precautions including but not limited to proper eye, respiratory, and skin protection. It is also essential to do all processes in a well-ventilated room with adequate lighting. Make sure to read all chemical labels and know how to clean up any spills as well as how to dispose the chemicals properly.
V. Testing

A. Communication

To test the USART and I\textsuperscript{2}C libraries master and slave communication programs were written. Using the ET-avr ATmega128 development board as a master to communicate with an ATmega48A running a slave program and an LCD various sanarios were run. These sanarios tested the various functions of the serial libraries and the results of each test were displayed LCD for visual conformation.

B. Circuits

The filters, amplifiers, and motor driver circuits were first simulated using LTspice. Proper operations and specifications were verified yet when implemented the amplifier circuits did not operate as expected. Finished PCB were tested using a multimeter for resistance and continuity checks.

C. Sensors

The green LED sensor was calibrated and tested in a half scale tank. Five readings were taken from pure saltwater to a fully developed phytoplankton culture. After each reading was taken a sample of water was obtained and the cells were counted through the use of a microscope and a hemocytometer. The results can be seen in figure 22 on page 34.

The microscope, hemocytometer, and other lab equipment were checked out through the biology department at Cal Poly San Luis Obispo. The Operations Supervisor Nancy Reid was incremental in the process of both obtaining the equipment and providing assistance.
A hemocytometer, seen in figure 21 above, was originally used to count human blood cells but can also be used to accurately count microalgae cells. It is a thick glass slide with a mirrored surface with etched grids defining a known volume. A thick cover slip is placed over the surface of the grids and the sample is pulled into the grids by capillary attraction.

To use the hemocytometer a drop of two of the sample on the sample introduction point (figure 21) until the grid is covered. There are a defined number of large and small squares on grid of the hemocytometer and each one of these corresponds to an exact volume. Depending on the density of the culture and the size of the phytoplankton, the whole grid, a number of squares, or single squares can be used to count the number of phytoplankton cells. Once the cells are counted the count is then multiplied to get an approximate cell density. The multiplication factor changes depending on the type of hemocytometer used and the volume each square represents.

For calibration the Fuchs-Rosenthal hemocytometer was used. A microalgae density stick and a hemocytometer cell calculator were used to verify the values.
The color light sensor was calibrated and tested in the same way yet before taking the readings the sensor integration was adjusted for calibration using food coloring in the same specific density as the plankton culture. For the red integration 10 drops of red 40 were used in 10ml of water. The output was set to 100 by changing the integration time to 750. For the green integration 3 drops of green food coloring (blue 1 and yellow 5 mix) and 5 extra drops of yellow 5 were added to 10ml of water. The output was set to 100 by changing the integration time to 1600. Clear water was used as a set point and the sensor outputs 189 red and 470 green. The results for the color light sensor can be seen below in figures 23 and 24.
Figure 23: Density of Rotifers Using the Red Reading from the Color Light Sensor

Figure 24: Density of Phytoplankton Using the Green Reading from the Color Light Sensor
D. Pumps

Micro diaphragm pumps seen above in figure 25 were tested by supplying the pump with 12V DC and changing the duty cycle. The results can be seen in figure 26 below.

![Figure 25: Micro Diaphragm Pumps](image)

![Figure 26: Flow rate of the 12VDC Micro Diaphragm Pumps](image)
E. Complete System

The finished project was tested visually through the LED’s on each board. By changing the time and day each operation could be visually verified by the status of all pumps and relays. The completed circuit system can be seen below in figure 27.

Figure 27: Completed Circuit System
VI. Conclusions and Recommendations

The Automatic Clownfish Hatchery was never fully tested due to the inability to obtain clownfish eggs. The addition of clownfish larvae would have given a more accurate test and would allow for fine tuning of parameters.

Even without larvae, there are several changes that could be made to optimize the design. These include, but are not limited to: more research and testing, redesign of amplifier/filter circuits, combining all circuit boards onto one, and modifications to the manufacturing of the tanks.

In order to achieve greater accuracy with the sensors it would be beneficial to find the ideal frequency of light to determine rotifer density without being influenced by the presence of phytoplankton. It would also be beneficial to use two color light sensors since the color light sensor has better accuracy than the green led sensor. One sensor would be on the main tank and the other would be on the rotifer tank. These sensors are more expensive, but give a better picture of what organisms are in the water.

The filter circuits designed for the ADC board did not end up working and were cut out of the circuit. These filters should be re-designed and implemented for better accuracy of the LED sensors.

For ease of manufacturing and building, the combination of all the PCB’s into one main board would be beneficial. Using a different software than EagleCAD light, or the full version of EagleCAD, the entire design could be implemented on one board using two ATmega8A (or equivalent) microcontrollers.

Although the CNC laser was fast and efficient, it left the acrylic with rounded with slightly slanted edges. This made it hard to glue the acrylic together due to the acrylic glue needing a flat textured surface to make a good bond. The use of a CNC router or milling machine would be a better option to cut the material.

Overall the project was a success; it filled the main tank with proper densities of phytoplankton and greenwater as well as keeping both cultures alive and growing.
VII. Bibliography


*MLA formatting by BibMe.org.*
VIII. Appendix A: Schematics

A. Master Board

B. Master Daughter Board
C. Motor Driver Board
D. Motor Driver Daughter Board

E. Motor Driver Current Adapter Board
F. ADC Board

G. USART Board
H. Power Supply Adapter

I. Relay Board
J. LED Driver Board
## IX. Appendix B: Parts List, Cost, and Time Schedule Allocation

### A. Parts List and Cost

<table>
<thead>
<tr>
<th>Part Description</th>
<th>Distributor</th>
<th>Part Number</th>
<th>Quantity</th>
<th>Unit Price</th>
<th>Extended Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCU AVR 8K FLASH 16MHZ 32-TQFP</td>
<td>Digikey</td>
<td>ATMega8A-AU-ND</td>
<td>1</td>
<td>3.2100</td>
<td>3.21</td>
</tr>
<tr>
<td>DIODE SCHOTTKY 1A 20V POWERMITE1</td>
<td>Digikey</td>
<td>UPS5817E3CT-ND</td>
<td>1</td>
<td>0.2150</td>
<td>0.22</td>
</tr>
<tr>
<td>CONNECTOR FEMALE 16POS.1&quot; TIN</td>
<td>Digikey</td>
<td>SSF-16SH-0503-020</td>
<td>3</td>
<td>1.1200</td>
<td>3.36</td>
</tr>
<tr>
<td>LED 5MM 570NM GREEN WTR CLR</td>
<td>Digikey</td>
<td>RMCF0805J1K00CT-ND</td>
<td>100</td>
<td>0.0032</td>
<td>3.20</td>
</tr>
<tr>
<td>PCB COPPER CLAD 3 X 4.5&quot; 1 SIDE</td>
<td>Digikey</td>
<td>PC5-ND</td>
<td>2</td>
<td>3.5300</td>
<td>7.06</td>
</tr>
<tr>
<td>PCB COPPER CLAD 3X4.5&quot; 2 SIDE</td>
<td>Digikey</td>
<td>PC74-ND</td>
<td>2</td>
<td>4.8400</td>
<td>9.68</td>
</tr>
<tr>
<td>IC RTC SERIAL 2WIRE LP 8-DIP</td>
<td>Digikey</td>
<td>DS1337+ND</td>
<td>1</td>
<td>2.7300</td>
<td>2.73</td>
</tr>
<tr>
<td>CRYSTAL 32.768 KHZ 6PF CYL</td>
<td>Digikey</td>
<td>300-8301-ND</td>
<td>1</td>
<td>0.2800</td>
<td>0.28</td>
</tr>
<tr>
<td>IC REG LDO 1.0A 3.3V SOT-223</td>
<td>Digikey</td>
<td>AP1117E33GDICT-ND</td>
<td>1</td>
<td>0.5600</td>
<td>0.56</td>
</tr>
<tr>
<td>IC REG LDO 1.0A 5.0V SOT-223</td>
<td>Digikey</td>
<td>AP1117E50GDICT-ND</td>
<td>4</td>
<td>0.5600</td>
<td>2.24</td>
</tr>
<tr>
<td>IC MCU AVR 4K FLASH 28PDIP</td>
<td>Digikey</td>
<td>ATMega48A-PU-ND</td>
<td>2</td>
<td>2.5800</td>
<td>5.16</td>
</tr>
<tr>
<td>CONNECTOR BNC R/A 50 OHM PCB</td>
<td>Jameco</td>
<td>A97555-ND</td>
<td>1</td>
<td>2.3000</td>
<td>2.30</td>
</tr>
<tr>
<td>PCB, BLANK, SGL, DBL SIDED, PHENOLIC,</td>
<td>Jameco</td>
<td>161926</td>
<td>1</td>
<td>9.9500</td>
<td>9.95</td>
</tr>
<tr>
<td>PCB, BLANK, SGL, DBL SIDED, PHENOLIC,</td>
<td>Jameco</td>
<td>169325</td>
<td>1</td>
<td>19.9500</td>
<td>19.95</td>
</tr>
<tr>
<td>SMD, 1206, GRN, 569NM, CLR,</td>
<td>Jameco</td>
<td>697741</td>
<td>1</td>
<td>0.1000</td>
<td>1.00</td>
</tr>
<tr>
<td>Cap, Ceramic, SMT, 0805, 1uf,</td>
<td>Jameco</td>
<td>185891</td>
<td>100</td>
<td>0.0100</td>
<td>1.00</td>
</tr>
<tr>
<td>SMD LED, 0805 SupR THIN, YLLW</td>
<td>Jameco</td>
<td>207536</td>
<td>1</td>
<td>0.0700</td>
<td>0.70</td>
</tr>
<tr>
<td>CONN, TERMINAL BLOCK, 2 POSITION</td>
<td>Jameco</td>
<td>2120647</td>
<td>40</td>
<td>0.2500</td>
<td>10.00</td>
</tr>
<tr>
<td>SOCKET, IC, 20 PIN, 390261-6,</td>
<td>Jameco</td>
<td>526221</td>
<td>2</td>
<td>0.1600</td>
<td>0.32</td>
</tr>
<tr>
<td>IC, MCU, ATTINY2313V-10P, 8-BIT</td>
<td>Jameco</td>
<td>680939</td>
<td>2</td>
<td>2.2500</td>
<td>4.50</td>
</tr>
<tr>
<td>CONN, TERMINAL BLOCK, 2 POSITION</td>
<td>Jameco</td>
<td>2120647</td>
<td>40</td>
<td>0.2500</td>
<td>10.00</td>
</tr>
<tr>
<td>HEADERS, ST MALE, 1R, 40PIN</td>
<td>Jameco</td>
<td>152726</td>
<td>1</td>
<td>0.5500</td>
<td>0.55</td>
</tr>
<tr>
<td>CONNECTOR, 1&quot;, 1R, 4PIN HSG</td>
<td>Jameco</td>
<td>152734</td>
<td>1</td>
<td>0.4500</td>
<td>0.45</td>
</tr>
<tr>
<td>@CONNECTOR, PIN, FEM, SQ, POST TIN</td>
<td>Jameco</td>
<td>181673</td>
<td>4</td>
<td>0.0400</td>
<td>0.16</td>
</tr>
<tr>
<td>BATTERY HOLDER FOR CR2032,</td>
<td>Jameco</td>
<td>355434</td>
<td>1</td>
<td>0.7900</td>
<td>0.79</td>
</tr>
<tr>
<td>@RESISTOR, SMT, 1206, 10K OHM,</td>
<td>Jameco</td>
<td>2078848</td>
<td>100</td>
<td>0.0080</td>
<td>0.80</td>
</tr>
<tr>
<td>RELAY, SPD, 6VDC, 10A</td>
<td>Jameco</td>
<td>227737</td>
<td>3</td>
<td>1.4900</td>
<td>4.47</td>
</tr>
<tr>
<td>Ferric Chloride</td>
<td>Mouser</td>
<td>590-415-4L</td>
<td>1</td>
<td>28.9500</td>
<td>28.95</td>
</tr>
<tr>
<td>Liquid Tin</td>
<td>Mouser</td>
<td>590-421-500ML</td>
<td>1</td>
<td>33.3800</td>
<td>33.38</td>
</tr>
<tr>
<td>10-pin female header</td>
<td>Mouser</td>
<td>517-974-02-10</td>
<td>5</td>
<td>1.4700</td>
<td>7.35</td>
</tr>
<tr>
<td>Tantalum Capacitors - Solid SMD 35volts 1uF 20%</td>
<td>Mouser</td>
<td>80-845196H6105M109</td>
<td>10</td>
<td>0.2200</td>
<td>2.20</td>
</tr>
<tr>
<td>Schottky (Diodes &amp; Rectifiers) 1mA 40 Volt</td>
<td>Mouser</td>
<td>625-LL101C</td>
<td>12</td>
<td>0.0500</td>
<td>0.60</td>
</tr>
<tr>
<td>MOSFET Power TAPE13 PWR-MOS</td>
<td>Mouser</td>
<td>771-BUK98150-55A/T3</td>
<td>12</td>
<td>0.2100</td>
<td>2.52</td>
</tr>
<tr>
<td>Trimmer Resistors - SMD 5Kohms Sealed 3mm Sngl turn</td>
<td>Mouser</td>
<td>81-PVG3AS020C01R00</td>
<td>4</td>
<td>0.8300</td>
<td>3.32</td>
</tr>
<tr>
<td>Thick Film Resistors - SMD 1/4watts 240ohms 5%</td>
<td>Mouser</td>
<td>660-RK7382BTTD241J</td>
<td>10</td>
<td>0.0500</td>
<td>0.50</td>
</tr>
<tr>
<td>Multilayer Ceramic Capacitors (MLCC) - SMD/SMT 50volts 0.1uf 25U 20%</td>
<td>Mouser</td>
<td>80-C1206C104M5U7210</td>
<td>10</td>
<td>0.0300</td>
<td>0.30</td>
</tr>
<tr>
<td>Linear Regulators - Standard 3 Term Adj. Pos.</td>
<td>Mouser</td>
<td>595-LM317DCYR</td>
<td>4</td>
<td>0.7900</td>
<td>3.16</td>
</tr>
<tr>
<td>Thick Film Resistors - SMD 1/8WATT 100OHS</td>
<td>Mouser</td>
<td>290-110-RC</td>
<td>4</td>
<td>0.0500</td>
<td>0.20</td>
</tr>
<tr>
<td>Thick Film Resistors - SMD 1/8WATT 8.25KOHMS</td>
<td>Mouser</td>
<td>290-8.25K-RC</td>
<td>4</td>
<td>0.0500</td>
<td>0.20</td>
</tr>
<tr>
<td>Thick Film Resistors - SMD 1/8WATT</td>
<td>Mouser</td>
<td>290-31.6K-RC</td>
<td>4</td>
<td>0.0500</td>
<td>0.20</td>
</tr>
<tr>
<td>Description</td>
<td>Brand</td>
<td>Part Number</td>
<td>Quantity</td>
<td>Price</td>
<td></td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------</td>
<td>-------------</td>
<td>----------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>Trimmer Resistors - SMD 3MM 20KOHMS 25% 0.1 WATT OPEN FRM</td>
<td>Mouser</td>
<td>652-TC33X-2-203E</td>
<td>2</td>
<td>0.18</td>
<td></td>
</tr>
<tr>
<td>Op Amps Dual Gen Purp JFET</td>
<td>Mouser</td>
<td>511-TL082ACD</td>
<td>4</td>
<td>0.44</td>
<td></td>
</tr>
<tr>
<td>Thick Film Resistors - SMD 1/8WATT 1.5KOHMS</td>
<td>Mouser</td>
<td>290-1.5K-RC</td>
<td>10</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>Thick Film Resistors - SMD 56K 5%</td>
<td>Mouser</td>
<td>652-CR1206JW-563ELF</td>
<td>4</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>Trimmer Resistors - SMD 3MM 100KOHMS 25% 0.1 WATT OPEN FRM</td>
<td>Mouser</td>
<td>652-TC33X-2-104E</td>
<td>3</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>Thick Film Resistors - SMD 10K 1%</td>
<td>Mouser</td>
<td>652-CR1206FX-1002ELF</td>
<td>50</td>
<td>0.03</td>
<td></td>
</tr>
<tr>
<td>Thick Film Resistors - SMD 4.7KOHM 1/4 WATT 5%</td>
<td>Mouser</td>
<td>652-CR1206JW-472GLF</td>
<td>10</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>Op Amps Single Gen Purp JFET</td>
<td>Mouser</td>
<td>511-TL081CD</td>
<td>1</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>Trimmer Resistors - SMD 10K OHM 3MM ST CERM</td>
<td>Mouser</td>
<td>858-35WR10KLFTR</td>
<td>2</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td>Zener Diodes Diode Zener Single 5.6V 1% 250mW 3-Pin</td>
<td>Mouser</td>
<td>771-BZX84-A5V6,215</td>
<td>1</td>
<td>0.19</td>
<td></td>
</tr>
<tr>
<td>Phone Line Cord (50ft., white)</td>
<td>GE</td>
<td>76530</td>
<td>1</td>
<td>7.88</td>
<td></td>
</tr>
<tr>
<td>Modular Line Cord Plugs 10 Pk</td>
<td>GE</td>
<td>26133</td>
<td>1</td>
<td>3.64</td>
<td></td>
</tr>
<tr>
<td>Clear Acrylic Sheet 18x24x.093</td>
<td>Optix</td>
<td>MC-05</td>
<td>6</td>
<td>7.98</td>
<td></td>
</tr>
<tr>
<td>Vinyl Tubing  5/8&quot;OD x 1/2&quot;ID x 20'</td>
<td>BT50</td>
<td>1</td>
<td>2.49</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vinyl Tubing  1/2&quot;OD x 3/8&quot;ID x 10'</td>
<td>D328</td>
<td>3</td>
<td>2.76</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4&quot; Barbed Tees</td>
<td>H828</td>
<td>1</td>
<td>4.79</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/4&quot; In Line Shut Off Valve</td>
<td>D33A</td>
<td>3</td>
<td>1.98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1/2&quot; x 10' PVC40 PE Pipe</td>
<td>US Plastics</td>
<td>44449</td>
<td>2</td>
<td>9.85</td>
<td></td>
</tr>
<tr>
<td>Acrylic Sheet 12x24x1/8</td>
<td>US Plastics</td>
<td>44111</td>
<td>1</td>
<td>1.21</td>
<td></td>
</tr>
<tr>
<td>3/4&quot; x 12&quot; Clear Acrylic Piano Hinge</td>
<td>US Plastics</td>
<td>44037</td>
<td>1</td>
<td>21.78</td>
<td></td>
</tr>
<tr>
<td>3/8&quot; OD x 2 3/4&quot; x 6' ID Acrylic Tubing</td>
<td>US Plastics</td>
<td>44019</td>
<td>1</td>
<td>1.08</td>
<td></td>
</tr>
<tr>
<td>MAXI CULTURE KIT™, FOR MICROALGAE AND ROTIFERS</td>
<td>Florida Aqua Farms</td>
<td>GA-MACK</td>
<td>1</td>
<td>103.5</td>
<td></td>
</tr>
<tr>
<td>Aquarium Starter Kit 15 Gallon</td>
<td>Top Fin</td>
<td>HY-10</td>
<td>10</td>
<td>11.28</td>
<td></td>
</tr>
<tr>
<td>12VDC 3W Micopump 100ml/min</td>
<td>CXDF</td>
<td>HY-10</td>
<td>10</td>
<td>112.8</td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td></td>
<td><strong>615.24</strong></td>
<td></td>
</tr>
</tbody>
</table>
### B. Time Schedule Allocation Gantt Chart

#### Automatic Clearwell Hatchery

<table>
<thead>
<tr>
<th>Task Description</th>
<th>Duration</th>
<th>Jan</th>
<th>Feb</th>
<th>Mar</th>
<th>Apr</th>
<th>May</th>
<th>Jun</th>
<th>Jul</th>
<th>Aug</th>
<th>Sep</th>
<th>Oct</th>
<th>Nov</th>
<th>Dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering Specification</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Review of Plans</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Surveys</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Site 1 (Primary Tank)</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Site 2 (Secondary)</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Site 3 (Shack)</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Civic Utility Services</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Plumbing</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Electrical</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>39</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Automatic Clearwell Hatchery

<table>
<thead>
<tr>
<th>Task Description</th>
<th>Duration</th>
<th>Jan</th>
<th>Feb</th>
<th>Mar</th>
<th>Apr</th>
<th>May</th>
<th>Jun</th>
<th>Jul</th>
<th>Aug</th>
<th>Sep</th>
<th>Oct</th>
<th>Nov</th>
<th>Dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering Specification</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Review of Plans</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Surveys</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Site 1 (Primary Tank)</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Site 2 (Secondary)</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Site 3 (Shack)</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Civic Utility Services</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Plumbing</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Electrical</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>39</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
X. Appendix C: Printed Circuit Board Artwork

A. Master Board

B. Master Daughter Board
C. Motor Driver Board

D. Motor Driver Daughter Board

E. Motor Driver Current Adapter Board
F. ADC Board

G. USART Board

H. Power Supply Adapter
I. Relay Board

J. LED Driver Board
XI. Appendix D: Basic Program Listing

A. ACH_Master_Controller.c

The following code contains the Interrupt Service Routine for capturing data sent from the ADC controller as well as the function call to the code responsible for all of the supervisory logic of the ACH.

```c
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <math.h>
#include <stdlib.h>
#include <MonzireAVR/Serial/twi.h>
#include <MonzireAVR/Displays/lcd4.h>

#include "ADC.h"
#include "operators.h"
#include "uart.h"
#include "waterchange.h"
#include "plankton.h"
#include "babies.h"
#include "motor_control_com.h"
#include "ds1337.h"

#define MASTER_ADDRESS   0x0A
#define SENSOR_CONTROLLER  0x03

#define PLANKTON   0
#define BABIES    1
#define WATERCHANGE   2

#define MASTER_INPUT_0   0
#define MASTER_INPUT_1   1
#define MASTER_INPUT_2   2
#define MASTER_INPUT_3   3
#define MASTER_INPUT_4   6
#define MASTER_INPUT_5   7

#define RED_INT    750
#define GREEN_INT    1600

volatile unsigned char usart_selected = 0;
```

// This variable will be 1 if this device is selected on the usart.
volatile unsigned char usart_selected = 0;

// This variable denotes the data sequence of the current USART transmission.
unsigned char usart_sequence = 0;

int main(void)
{
    //Initialize local variables
    unsigned char mode = 1;

    //Set port status
    LCD_PORT_DDR = OUTPUT;  //Set LCD port (PortB) as output

    //Designate external pins on PortC
    DDRC = (0 << MASTER_INPUT_0) | (0 << MASTER_INPUT_1) | (0 << MASTER_INPUT_2) | (0 << MASTER_INPUT_3);  //as inputs

    //Designate external pins on PortD as inputs
    DDRD = (0 << MASTER_INPUT_4) | (0 << MASTER_INPUT_5);

    //System Initializations
    TWI_init();
    USART0_Init(MYUBRR);
    lcd4_init();

    //Set Color Light Sensor integration times
    ADJD_S371_WriteRegister(INT_RED_LO, (char)(RED_INT & 0x00FF));
    ADJD_S371_WriteRegister(INT_RED_HI, (char)((RED_INT & 0xFF00) >> 8));
    ADJD_S371_WriteRegister(INT_GREEN_LO, (char)(GREEN_INT & 0x00FF));
    ADJD_S371_WriteRegister(INT_GREEN_HI, (char)((GREEN_INT & 0xFF00) >> 8));

    while(1)
    {
        ACH(&mode);
    }
    return (0);
}

//USART Interrupt Routine
ISR(USART_RX_vect)
{
    cli(); //Keep interrupts from causing problems with data transfer
    unsigned char data;
    data = USART_Receive(); //Get data
    if(usart_adc_seq == 0) //First packet of data is ADC channel
    {
        usart_adc_seq = 1;
        adc_channel = data;
        exit_counter = 0;
    }
    else if(usart_adc_seq == 1) //Second data packet is adc value
    {
        usart_adc_seq = 2;
        //Data received needs to be within the expected limits
        if((data > ADC_limit_low[adc_channel]) && (data < ADC_limit_high[adc_channel]))
        {
            ADC_values[adc_channel] = data;
        }
        exit_counter = 0;
    }
}
B. Plankton.h

This file contains the function which handles the operations for the plankton mode of the Automatic Clownfish Hatchery. In this mode there are no larvae to feed in the main tank, therefore the program just runs the processes necessary for the continuous culture of the plankton.

```c
#ifndef __PLANKTON_H
#define __PLANKTON_H

#include <avr/io.h>
#include <util/delay.h>
#include <MonzireAVR/Serial/twi.h>
#include <MonzireAVR/Displays/lcd4.h>
#include <stdlib.h>
#include "motor_control_com.h"
#include "uart.h"
#include "ds1337.h"
#include "ADC.h"
#include "Time_Functions.h"
#include "babies.h"

#define MAIN_TANK_DAY_HOUR   10  //10:00 AM
#define MAIN_TANK_NIGHT_HOUR 18  //06:00 PM
#define PLANKTON_DAY_HOUR    7   //07:00 AM
#define PLANKTON_NIGHT_HOUR  23   //11:00 PM
#define PLANKTON_NEW_H2O    4    //Add water every 4 hours
#define ROTIFER_H2O_CHANGE_HOUR  9  //9:02 AM
#define ROTIFER_H2O_CHANGE_MINUTE  2

//Number of days between waterchanges
#define ROTIFER_PERIOD   4
#define ORGANISM_DENSITY 18    //Low density
```

//Signal the end of the data Rx
else if(data == 0xFF)
{
    if(exit_counter == 0)
    {
        exit_counter++;
    }
    else if(exit_counter == 1)
    {
        exit_counter = 0;
        uart_adc_seq = 0;
    }
}
sei();
/* EOF ACH_Master_Controller.c */
//Master Input Signals
#define ROTIFER_FLOAT1   0x01
#define ROTIFER_FLOAT1_PORT   PINC
#define ROTIFER_FLOAT2   0
#define ROTIFER_FLOAT2_PORT   PINC

//Switch mode
#define BABY_MODE    0x80
#define BABY_MODE_PORT   PIND

//Function Prototypes
void ACH(unsigned char* mode);

void ACH(unsigned char* mode)
{
    //Set LCD variables
    char LCD_Temp[4], LCD_ph[4];

    //Set RTC variables
    unsigned char Day, Hour, Minute, Second;
    unsigned char pm_flag;

    //Rotifer day of next water change
    unsigned char Rotifer_day = 5;
    unsigned char Rotifer_top_off = 0;

    //Plankton time of next water addition
    unsigned char Plankton_water = PLANKTON_DAY_HOUR;
    unsigned char Plankton_density_reading = 0;
    unsigned char Plankton_density_value = 0;

    //Flags
    unsigned char Relay1_on = 0; //Plankton Light
    unsigned char Relay3_on = 0; //Main Tank Light

lcd4_init();

    //Setup the Plankton mode display
    lcd4_ClearDisplay();
    lcd4_Command(HOME);

    //Display mode format
    lcd4_putstr("Phytoplankton   00:00");
    lcd4_pos_xy(0,2);
    //lcd4_putstr("Temp:    F   pH:    ");
    lcd4_putstr("GreenLED  Red  Green");
    //lcd4_pos_xy(7,2);
    //lcd4_putchar(DEGREES);
    Display_Time();

    //Disable opens the normally closed contact
    USART_motor_controller(MOTOR_CONTROLLER_2, RELAY_2, DISABLE, 100);

    sei(); //Enable interrupts

    //Main while loop of Plankton. Exit comes with menu choice or //timed interrupt.
    while(1)
    {
        //Get time
RTC_Get_Time(RTC_ADDRESS, 1, &Second, &Minute, &Hour);
pm_flag = Get_AMPM();

RTC_Get_Day(RTC_ADDRESS, &Day);

// 24 Hour mode
if(pm_flag == 1)
{
    Hour = Hour + 12;
}

// Display Time
Display_Time();

// Calculate Temp Value
Temp = (ADC_values[0] * 100) * V_STEP;

// Format Temp Value
itoa(Temp, LCD_Temp, 10);

// Display Temp
lcd4_pos_xy(5,2);
lcd4_putstr(LCD_Temp);

// Calculate pH value
pH = ( ( (ADC_values[5] - PH_OFFSET) * V_STEP ) * PH_STEP ) + 7;

// Format pH value
dtostrf(pH, 3, 1, LCD_ph);

// Display pH
lcd4_pos_xy(17,2);
lcd4_putstr(LCD_ph);

// Turn on Relay 1 (Plankton Light) if within the // appropriate time window
if((Hour >= PLANKTON_DAY_HOUR) && (Hour > PLANKTON_NIGHT_HOUR))
{
    USART_motor_controller(MOTOR_CONTROLLER_2, RELAY_1, DISABLE, 100);
    Relay1_on = 1;
}
else
{
    USART_motor_controller(MOTOR_CONTROLLER_2, RELAY_1, ENABLE, 100);
    Relay1_on = 0;
}

// Turn on Relay 3 (Main Tank Light) if within the // appropriate time window
if((Hour >= MAIN_TANK_DAY_HOUR) && (Hour < MAIN_TANK_NIGHT_HOUR))
{
    USART_motor_controller(MOTOR_CONTROLLER_2, RELAY_3, DISABLE, 100);
    Relay3_on = 1;
}
else
{
    USART_motor_controller(MOTOR_CONTROLLER_2, RELAY_3, ENABLE, 100);
Relay3_on = 0;

//Perform Rotifer Waterchange
if((Day == Rotifer_day) && (Hour == ROTIFER_H20_CHANGE_HOUR) && (Minute == ROTIFER_H20_CHANGE_MINUTE))
{
    //Perform Rotifer water change
    Rotifer_Waterchange(&Minute);

    //Calculate next waterchange day
    Rotifer_day = Day + ROTIFER_PERIOD;

    if(Rotifer_day > 7)
    {
        Rotifer_day = Rotifer_day - 7;
    }
}

//Add new water to phytoplankton every 4 hours.
if((Hour == Plankton_water) && (Minute < 2))
{
    USART_motor_controller(MOTOR_CONTROLLER_2, MOTOR_CHANNEL_1, ENABLE, 60);
}
else if((Hour == Plankton_water) && (Minute >=2))
{
    USART_motor_controller(MOTOR_CONTROLLER_2, MOTOR_CHANNEL_1, DISABLE, 60);

    Plankton_water = Plankton_water + PLANKTON_NEW_H20;

    if(Plankton_water > PLANKTON_NIGHT_HOUR)
    {
        Plankton_water = PLANKTON_DAY_HOUR;
    }
}

//Monitor Plankton density
itoa(ADC_values[2], LCD_Temp, 10);
lcd4_pos_xy(0,3);
lcd4_putstr(" ");
lcd4_pos_xy(3,3);
lcd4_putstr(LCD_Temp);
Plankton_density_value = ADC_values[2];

if( ((Hour >= PLANKTON_DAY_HOUR) && (Hour <= PLANKTON_NIGHT_HOUR)) &&
(Plankton_density_value > ORGANISM_DENSITY) && (Minute == Plankton_density_reading))
{
    Plankton_density_reading = Plankton_density_reading + 10;

    if(Plankton_density_reading > 59)
    {
        Plankton_density_reading = 0;
    }
USART_motor_controller(MOTOR_CONTROLLER_2, MOTOR_CHANNEL_3, ENABLE,  
100);
}

else if(Minute > Plankton_density_reading - 10)  
{  
    USART_motor_controller(MOTOR_CONTROLLER_2, MOTOR_CHANNEL_3, DISABLE,  
100);
}

//Rotifer water input pump
if((ROTIFER_FLOAT1_PORT & FLOAT1) != 1)  
{  
    USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_3, ENABLE,  
100);

    Rotifer_top_off = 1;
}
else if(Rotifer_top_off == 1)  
{  
    USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_3, DISABLE,  
100);

    Rotifer_top_off = 0;
}

//If baby mode switch is toggled high, exit phytoplankton //mode and enter baby mode
if(((BABY_MODE_PORT & BABY_MODE) == 0) && (*mode == 1))  
{  
    *mode = 2;

    lcd4_pos_xy(0,0);
    lcd4_putstr("    ");
    lcd4_pos_xy(0,0);
    lcd4_putstr("Baby mode");
}
else if(((BABY_MODE_PORT & BABY_MODE) == BABY_MODE) && (*mode == 2))  
{  
    *mode = 1;

    lcd4_pos_xy(0,0);
    lcd4_putstr("    ");
    lcd4_pos_xy(0,0);
    lcd4_putstr("Phytoplankton");
}

if(*mode == 2)  
{  
    Babies(&Day, &Hour, &Minute, &Second, MAIN_TANK_DAY_HOUR,  
MAIN_TANK_NIGHT_HOUR);
}

}//End while loop

#error
C. Babies.h

This file contains the function which handles the operations for the Babies mode of the Automatic Clownfish Hatchery. In this mode the main tank is stocked with the correct amount of rotifers and phytoplankton for the larvae clownfish.

```c
#ifndef __BABIES_H
#define __BABIES_H

#include <avr/io.h>
#include <stdlib.h>
#include <math.h>
#include <MonzireAVR/Serial/twi.h>
#include <MonzireAVR/Displays/lcd4.h>

#include "plankton.h"
#include "ADC.h"
#include "ADJD-S371.h"
#include "motor_control_com.h"
#include "Time_Functions.h"
#include "usart.h"
#include "ds1337.h"

#define RED_SET_POINT 220     //Color Light Sensor Red compare value for 10 Rotifer/ml
#define GREEN_SET_POINT 480    //Color Light Sensor  Green comp. value for 3M Phytoplankton/ml
#define CLS_SAMPLES 50         //Samples are taken and adjustments made
#define PH_LOWER_SET_POINT 8.0 //pH should not fall below this range
#define PH_UPPER_SET_POINT 8.3 //pH should not be above this point
#define PLANKTON_DENSITY_FREQ 5
#define ROTIFER_DENSITY_FREQ 5
#define PH_SENSE_FREQ   1     //Take samples every hour

//Global Variables
//Main Tank Phytoplankton
volatile unsigned char plankton_density = 0;
volatile unsigned char rotifer_density = 0;

//pH Control variables
//Repeat this every hour
volatile unsigned char pH_sense_time = 6;
//Stop the pump after 30 sec
volatile unsigned char pH_sense_stop = 30;

volatile unsigned int red_total = 0, green_total = 0;
volatile unsigned int red_average = 0, green_average = 0;
volatile unsigned char Cls_counter = 0;

//Function prototype
void Babies(unsigned char* Day, unsigned char* Hour, unsigned char* Minute, unsigned char* Second, unsigned char main_tank_day_hour, unsigned char main_tank_night_hour);

//Babies function definition.
```

62
void Babies(unsigned char* Day, unsigned char* Hour, unsigned char* Minute, unsigned char* Second, unsigned char main_tank_day_hour, unsigned char main_tank_night_hour)
{
    unsigned int Color_light_green, Color_light_red;
    char lcd_red[6], lcd_green[6];

    // Take color light sensor reading
    ADJD_S371_TakeSensorReading();

    // Get Red and Green color values
    Color_light_red = ADJD_S371_GetRegisterData(DATA_RED_LO) | (ADJD_S371_GetRegisterData(DATA_RED_HI) << 8);
    Color_light_green = ADJD_S371_GetRegisterData(DATA_GREEN_LO) | (ADJD_S371_GetRegisterData(DATA_GREEN_HI) << 8);

    red_total = Color_light_red + red_total;
    green_total = Color_light_green + green_total;

   Cls_counter ++;

    // If the correct number of samples have been taken, average
    // samples together and display results.
    if (Cls_counter > CLS_SAMPLES)
    {
        // Generate Green average
        red_average = red_total / (int)CLS_SAMPLES;
        red_total = 0; // Reset Red variable

        // Generate Green average
        green_average = green_total / (int)CLS_SAMPLES;
        green_total = 0; // Reset Green variable

        Cls_counter = 0;

        // Display Red and Green values
        itoa(red_average, lcd_red, 10);
        itoa(green_average, lcd_green, 10);
lcd4_pos_xy(10,3);
lcd4_putstr("          ");
lcd4_pos_xy(10,3);
lcd4_putstr(lcd_red);
lcd4_pos_xy(16, 3);
lcd4_putstr(lcd_green);
    }

    // Main Tank Phytoplankton Density
    if((*Hour >= main_tank_day_hour) && (*Hour <= main_tank_night_hour) && (green_average > GREEN_SET_POINT) && (*Minute == plankton_density))
    {
        USART_motor_controller(MOTOR_CONTROLLER_2, MOTOR_CHANNEL_2, ENABLE, 60);
    }

    else if((*Hour >= main_tank_day_hour) && (*Hour <= main_tank_night_hour) && (*Minute == plankton_density + 1))
    {
        USART_motor_controller(MOTOR_CONTROLLER_2, MOTOR_CHANNEL_2, DISABLE, 70);
        // If the next reading occurs past 59th minute of the hour
plankton_density = plankton_density + PLANKTON_DENSITY_FREQ;

if(plankton_density > 59)
{
    plankton_density = 0;
}

//Main Tank Rotifer Density
if(*Hour >= main_tank_day_hour) && (*Hour <= main_tank_night_hour) && (red_average > RED_SET_POINT) && (*Minute == rotifer_density))
{
    USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_2, ENABLE, 100);
}
else if(*Hour >= main_tank_day_hour) && (*Hour <= main_tank_night_hour) && (*Minute == rotifer_density + 1))
{
    USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_2, DISABLE, 100);
    rotifer_density = rotifer_density + ROTIFER_DENSITY_FREQ;
    //If the next reading occurs past 59th minute of the hour
    //then set the minute to 0.
    if(rotifer_density > 59)
    {
        rotifer_density = 0;
    }
}

//Compare pH value with pH set point and add chemical accordingly.
if(*Hour >= main_tank_day_hour) && (*Hour <= main_tank_night_hour) && ((pH > PH_UPPER_SET_POINT) || (pH < PH_LOWER_SET_POINT)) && (*Minute == pH_sense_time))
{
    USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_1, ENABLE, 60);
}
else if(*Hour >= main_tank_day_hour) && (*Hour <= main_tank_night_hour) && (*Second < pH_sense_stop))
{
    USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_1, DISABLE, 60);
    pH_sense_time = pH_sense_time + PH_SENSE_FREQ;
    //If the next time occurs outside of the sample window,
    //then set the time to the start of the day window.
    if(pH_sense_time > main_tank_night_hour)
    {
        pH_sense_time = main_tank_day_hour;
    }
}
#endif
D. Waterchange.h

This file contains the rotifer water change function which will perform all tasks associated with performing an automatic water change.

```c
#define __WATERCHANGE_H
#ifndef __WATERCHANGE_H
#define __WATERCHANGE_H

#include <avr/io.h>
#include <util/delay.h>
#include <MonzireAVR/Displays/lcd4.h>
#include <MonzireAVR/Serial/twi.h>
#include "motor_control_com.h"
#include "Time_Functions.h"
#include "ds1337.h"

#define FLOAT1 0x01
#define FLOAT1_PORT PINC
#define FLOAT2 0x02
#define FLOAT2_PORT PINC
#define FLOAT3 0x04
#define FLOAT3_PORT PINC
#define FLOAT4 0x08
#define FLOAT4_PORT PINC

//Function Prototypes
void Rotifer_Waterchange(unsigned char* Minute_start);

void Rotifer_Waterchange(unsigned char* Minute_start)
{
    unsigned char hour, minute = *Minute_start, second;

    //Turn off Rotifer air pump
    USART_motor_controller(MOTOR_CONTROLLER_2, RELAY_2, DISABLE, 100);
    USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_3, DISABLE, 100);

    I2C_WriteRegister(RTC_ADDRESS, MINUTES, 0x17);
    while(minute < *Minute_start + 15)
    {
        RTC_Get_Time(RTC_ADDRESS, 1, &second, &minute, &hour);
        Display_Time();
        _delay_ms(100);
    }

    USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_4, ENABLE, 100);

    while((FLOAT3_PORT & FLOAT3) != FLOAT3)
    {
        RTC_Get_Time(RTC_ADDRESS, 1, &second, &minute, &hour);
        Display_Time();
        _delay_ms(100);
    }

    USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_4, DISABLE, 100);
```

65
USART_motor_controller(MOTOR_CONTROLLER_1, MOTOR_CHANNEL_3, ENABLE, 100);

while((FLOAT1_PORT & FLOAT1) != FLOAT1)
{
    RTC_Get_Time(RTC_ADDRESS, 1, &second, &minute, &hour);
    Display_Time();
    _delay_ms(100);
}
#endif

E. ADC.h

This file is used in conjunction with ACH_Master_Controller.c. It contains information used to handle the received data from the ADC controller.

 ifndef __ADC_H
#define __ADC_H

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <math.h>

////////////////////////////////////////////
// ADC Constants
////////////////////////////////////////////
#define VREF  4.971  //Measured Reference Voltage
#define V_STEP  (VREF / 256)  //ADC voltage increment

////////////////////////////////////////////
// Sensor Constants
////////////////////////////////////////////
#define PH_STEP  -0.414
#define PH_OFFSET  128

////////////////////////////////////////////
// ADC Variables
////////////////////////////////////////////
volatile unsigned char ADC_values[6] = {0,0,0,0,0,0};
volatile unsigned char ADC_limit_high[6] = {200, 100, 100, 100, 100, 255};
volatile unsigned char ADC_limit_low[6] = { 0, 0, 0, 0, 0, 0};

////////////////////////////////////////////
// Sensor Variables
////////////////////////////////////////////
volatile char Temp;
volatile float pH;

////////////////////////////////////////////
// ADC USART Variables
////////////////////////////////////////////
volatile unsigned char exit_counter = 0;
volatile unsigned char usart_adc_seq = 0;
volatile unsigned char adc_channel = 0;

#endif //End ADC.h
F. ACH_ADC.h

The following code contains the functions and constants pertaining to ADC controller for the Automatic Clown fish Hatchery. The functions are used to take a series of averaged samples and forward the results to a waiting master controller.

```c
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include "usart.h"

// UART Constants
#define MYADDRESS 0x03
#define USART_EXIT 0xFF

// ADC Constants
#define NUM_SAMPLES 20
#define NUM_CHANNELS 6

//Global ADC Variables
volatile unsigned char sample_complete = 0;

ISR(ADC_vect)
{
    // This event causes the main program to continue
    // after the ADC sample is complete.
    sample_complete = 1;
}

void ADC_init(void)
{
    //Set Reference voltage to AREF
    //ADLAR: 1 - define 8-bit ADC mode
    ADMUX = (0<<REFS1) | (1<<REFS0) | (1<<ADLAR);

    //Enable ADC, ADC Interrupt, Set Prescaler to 128
    ADCSRA = (1<<ADEN) | (1<<ADIE) |
             (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0);

    //Disable the digital input on the analog channels
    DIDR0 = (1<<ADC0D) | (1<<ADC1D) | (1<<ADC2D) |
            (1<<ADC3D) | (1<<ADC4D) | (1<<ADC5D);
}

char ADC_Sample(unsigned char channel)
{
    unsigned char ADC_result = 0;
    sample_complete = 0;

    //Clear MUX values
    ADMUX = ADMUX & 0xF0;                  //This clears the last 4 values in ADMUX.
                                                                 //These are the MUX values
    //Set MUX values
    ADMUX = ADMUX | channel;

    //Start Conversion
    ADCSRA |= (1<<ADSC);
```

67
//Wait for conversion to complete
while (sample_complete == 0)

//Capture ADC result
ADC_result = ADCH;

//Format result for function output
//ADC_sample = (ADC_high << 8) | ADC_low;

//Return function result
return(ADC_result);
}

/*********************/
//Main
/*********************/
int main(void)
{

//Define local variables
//Variable holding the sum of all samples for a given channel
unsigned int Data[6] = {0,0,0,0,0,0};
unassigned char ADC_Values[6] = {0,0,0,0,0,0}; //Averaged ADC values

unsigned char i; //First level loop variable
unsigned char j; //Second level loop variable

_delay_ms(100);

//Subsystem initializations
ADC_init(); //ADC
USART0_Init(MYUBRR); //USART

//Enable interrupts
sei();

//Main program loop (loop forever)
while(1)
{
    ADC_init(); //ADC
    USART0_Init(MYUBRR); //USART

    //Take ADC Samples
    for(i = 0; i < NUM_CHANNELS; i++)
    {
        for(j = 0; j < NUM_SAMPLES + 1; j++)
        {
            //Add current sample to previous total
            Data[i] = Data[i] + (int)ADC_Sample(i);
        }
    }

    //Average ADC Samples
    for(i = 0; i < NUM_CHANNELS; i++)
    {
        ADC_Values[i] = (char)(Data[i] / (int)NUM_SAMPLES);
        Data[i] = 0; //Reset summing variable
    }

    for(i = 0; i < NUM_CHANNELS; i++)
    {
        //Send Channel
        USART_Transmit(i);

        //Send adc values
        USART_Transmit(ADC_Values[i]);
    }
}
The following code is the main program for the motor controllers used in the Automatic Clownfish Hatchery. The code for motor controller #2 is identical to that of motor controller #1 with the exception of the external daughter card port.

```c
#include <avr/io.h>
#include <avr/interrupt.h>
#include "usart.h"
#include "operators.h"

//Define Delay parameters
#define F_CPU  8000000UL

//Auto Clownfish Hatchery Device Address
#define MYADDRESS 0x02

//Non-PWM motor frequency
#define PWM_TOP  10

//Timer Variables
#define TIMER0_TOP  0xFF
#define TIMER0_BOTTOM  0x00
#define TIMER1_TOP  0xC8
#define TIMER1_BOTTOM  0x00

/*****************************/
**   Channel Definitions   **
*****************************/

//Channel 1
#define CH1_PORT  PORTD
#define CH1_BIT  5

//Channel 2
#define CH2_PORT  PORTB
#define CH2_BIT  0

//Channel 3
#define CH3_PORT  PORTB
#define CH3_BIT  1

//Channel 4
#define CH4_PORT  PORTB
#define CH4_BIT  2

//Channel 5
#define CH5_PORT  PORTB
#define CH5_BIT  3
```
//Channel 6
#define CH6_PORT PORTB
#define CH6_BIT 4

//Channel 1
#define CH_EXT1_PORT PORTD
#define CH_EXT1_BIT 4

//Channel 2
#define CH_EXT2_PORT PORTD
#define CH_EXT2_BIT 3

//Channel 3
#define CH_EXT3_PORT PORTD
#define CH_EXT3_BIT 2

//Channel 4
#define CH_EXT4_PORT PORTA
#define CH_EXT4_BIT 1

//Channel 5
#define CH_EXT5_PORT PORTA
#define CH_EXT5_BIT 0

//Function Prototypes
void USART_Init(unsigned int ubrr);
void USART_Transmit(unsigned char data);
unsigned char USART_Receive(void);
void Timer0_init(void);
void Timer1_init(void);

//Global Variables

//USART Variables
volatile unsigned char usart_select = 0;
volatile unsigned char usart_data_sequence = 0;
volatile unsigned char usart_endoftx = 0;
volatile unsigned char channel = 0;
volatile unsigned char channel_enable = 1;
volatile unsigned char channel_DC = 0;

//Status Variables
volatile unsigned char CH1_Status = 0;
volatile unsigned char CH2_Status = 0;
volatile unsigned char CH3_Status = 0;

//Duty Cycle Variables
volatile unsigned char CH1_DC = 5;
volatile unsigned char CH2_DC = 5;
volatile unsigned char CH3_DC = 5;

int main(void)
{
    char Duty_cycle = 0;

    DDRA = (1<<CH_EXT4_BIT)|(1<<CH_EXT5_BIT);
    DDRB = (1<<CH2_BIT)|(1<<CH3_BIT)|(1<<CH4_BIT)|
    (1<<CH5_BIT)|(1<<CH6_BIT);
    DDRD = (1<<CH1_BIT)|(1<<CH_EXT1_BIT)|(1<<CH_EXT2_BIT)|(1<<CH_EXT3_BIT);

    PORTA = 0;
    PORTD = 0;
    PORTB = 0;
    Timer0_init();
}
Timer1_init();
USART_Init(MYUBRR);
sei();
while(1)
{
// Channel 1 waveform
if((CH1_Status == 1) & (CH1_DC >= Duty_cycle))
{
  SETBIT(CH1_PORT,CH1_BIT);
}
else
{
  CLEARBIT(CH1_PORT,CH1_BIT);
}

// Channel 2 waveform
if((CH2_Status == 1) & (CH2_DC >= Duty_cycle))
{
  SETBIT(CH2_PORT,CH2_BIT);
}
else
{
  CLEARBIT(CH2_PORT,CH2_BIT);
}

// Channel 3 waveform
if((CH3_Status == 1) & (CH3_DC >= Duty_cycle))
{
  SETBIT(CH3_PORT,CH3_BIT);
}
else
{
  CLEARBIT(CH3_PORT,CH3_BIT);
}

// Non-PWM waveform period control
if(Duty_cycle >= PWM_TOP)
{
  Duty_cycle = 0;
}
else
{
  Duty_cycle++;
}
}
return(0);
}
ISR(USART_RX_vect)
{
cli();
char data;
data = USART_Receive();
// Look for address bit
if(usart_select == 0)
{
  if(data == MYADDRESS)
  {
    usart_select = 1;
  }
/usrart_data_sequence should be 1 at this point
usrart_data_sequence++;
usrart_endoftx = 0;
}

} else if(data == 0xFF)
{
    usart_endoftx++;
    if(usart_endoftx == 2)
    {
        usart_data_sequence = 0;
        usart_select = 0;
        usart_endoftx = 0;
    }
}

//Perform operation if slave selected
else if(usart_select == 1)
{
    switch(usrart_data_sequence)
    {
        case 1: channel = data;
            usart_data_sequence++;
            usart_endoftx = 0;
            break;
        case 2: channel_enable = data;
            usart_data_sequence++;
            usart_endoftx = 0;
            break;
        case 3: channel_DC = data;
            switch(channel)
            {
                case 1: CH1_Status = channel_enable;
                    CH1_DC = (((unsigned int)channel_DC) * ((unsigned int)PWM_TOP))/100;
                    break;
                case 2: CH2_Status = channel_enable;
                    CH2_DC = (((unsigned int)channel_DC) * ((unsigned int)PWM_TOP))/100;
                    break;
                case 3: CH3_Status = channel_enable;
                    CH3_DC = (((unsigned int)channel_DC) * ((unsigned int)PWM_TOP))/100;
                    break;
                case 4: if(channel_enable == 0)
                    {
                        CLEARBIT(TCCR0A,COM0A1);
                    }
                    else
                    {
                        SETBIT(TCCR0A,COM0A1);
                        OCR0A = (((unsigned int)channel_DC) * ((unsigned int)TIMER0_TOP))/100;
                    }
                    break;
    }
case 5: if(channel_enable == 0)
{
    CLEARBIT(TCCR1A,COM1A1);
}
else
{
    SETBIT(TCCR1A,COM1A1);

    OCR1A = ((unsigned int)channel_DC) * ((unsigned int)TIMER1_TOP))/100;
}
break;

case 6: if(channel_enable == 0)
{
    CLEARBIT(TCCR1A,COM1B1);
}
else
{
    SETBIT(TCCR1A,COM1B1);

    OCR1B = ((unsigned int)channel_DC) * ((unsigned int)TIMER1_TOP))/100;
}
break;

case 7: if(channel_enable == 0)
{
    CLEARBIT(CH_EXT1_PORT,CH_EXT1_BIT);
}
else
{
    SETBIT(CH_EXT1_PORT,CH_EXT1_BIT);
}
break;

case 8: if(channel_enable == 0)
{
    CLEARBIT(CH_EXT2_PORT,CH_EXT2_BIT);
}
else
{
    SETBIT(CH_EXT2_PORT,CH_EXT2_BIT);
}
break;

case 9: if(channel_enable == 0)
{
    CLEARBIT(CH_EXT3_PORT,CH_EXT3_BIT);
}
else
{
    SETBIT(CH_EXT3_PORT,CH_EXT3_BIT);
}
break;

case 10: if(channel_enable == 0)
{
CLEARBIT(CH_EXT4_PORT, CH_EXT4_BIT);
}
else
{
SETBIT(CH_EXT4_PORT, CH_EXT4_BIT);
}
break;

case 11: if(channel_enable == 0)
{
CLEARBIT(CH_EXT5_PORT, CH_EXT5_BIT);
}
else
{
SETBIT(CH_EXT5_PORT, CH_EXT5_BIT);
}
break;

default: break;
}

usart_select = 0;
usart_data_sequence = 0;
usart_endoftx = 0;
break;

// If the code gets this far, there is an error. Reset USART //sequence
variables.
default: usart_select = 0;
usart_data_sequence = 0;
usart_endoftx = 0;
break;
}
sei();

// Channel 4 Waveform
void Timer0_init(void)
{
  // (COM0A1 = 1, COM0A0 = 0) To turn on Channel 4
  TCCR0A = (0<<COM0A1) | (0<<COM0A0) |
           (0<<COM0B1) | (0<<COM0B0) |
           (1<<WGM01) | (1<<WGM00);

  TCCR0B = (0<<FOC0A) | (0<<FOC0B) |
           (0<<WGM02) | (0<<CS02)  |
           (0<<CS01)  | (1<<CS00);

  // TIMSK = (0<<OCIE0B) | (1<<TOIE0) | (0<<OCIE0A);
  // TIFR = 0x00;
  OCR0A = 0x7F; //TIMER0_BOTTOM;
}

// Channel 5 and 6 Waveforms
void Timer1_init(void)
{
  // (COM1A1 = 1, COM1A0 = 0) To turn on Channel 5
  // (COM1B1 = 1, COM1B0 = 0) To turn on Channel 6
  TCCR1A = (0<<COM1A1) | (0<<COM1A0) |
           (0<<COM1B1) | (0<<COM1B0) |
           (0<<WGM11) | (0<<WGM10) ;
TCCR1B = (0<<ICNC1) | 
          (0<<ICES1) | 
          (1<<WGM13) | (0<<WGM12) | 
          (0<<CS12) | (0<<CS11) | (1<<CS10)

TCCR1C = (0<<FOC1A) | (0<<FOC1B)

OCR1A = TIMER1_BOTTOM;
OCR1B = TIMER1_BOTTOM;

ICR1 = TIMER1_TOP; //ICR1 controls the TOP value of the counter