# Final Report

**William Patterson**

**Robot: Hexy and the Swarm**

**EEL 5666 – Intelligent Machines Design Laboratory**

**Instructors:**
Dr. A. Antonio Arroyo
Dr. Eric M. Schwartz

**Teaching Assistants:**
Mike Pridgen, Ryan Stevens,
Tim Martin, Thomas Vermeer,
Devin Hughes

Date – December 7, 2010

# Table of Contents

Abstract ....3
Executive Summary (Final Report only) ....4
Introduction ....5
Integrated System....5
Mobile Platform ....6
Actuation ....7
Sensors....7
    Bump Sensors....7
    IR Proximity Sensors....8
    CdS Cell....9
    Thermistor....10
    Battery Voltage Monitor....11
    38kHz IR Sensor....11
    Bluetooth Communication....12
    36kHz Beacon....13
Behaviors ....14
Experimental Layout and Results ....14
Conclusion ....15
Documentation ....16
    Bibliography....16
    Appendices ....16
    Main Code....16
    Serial Communication Source and Header....23

# Abstract

Swarm robotics is an approach to the coordination of multi-robot systems which consist of multiple simple physical robots. Traditional swarms are mesh networks, or networks in which no single robot is in control of the swarm. The goal of this project is to create more adaptive system in which a master node controls all swarm behavior and is able to change and create swarm behaviors on the fly.

To accomplish this goal multiple simple autonomous hexapod robots with various sensors will be built. Hexapods were chosen as the locomotion due to the simplicity that a three servo hexapod provides in both control and construction. The swarm communication will be done using the revision two bluetooth protocol. The robots will have class 2 bluetooth modules and be set as slave nodes. The master node will consist of a personal computer with a class 1 bluetooth dongle using the Linux Bluez bluetooth stack to interface with multiple nodes. Each robot is capable of accomplishing simple tasks on it's own. However when the master node is activated swarm behaviors can be accomplished. The swarm behaviors that will be performed are: an all halt of all swarm members on the location of a dark area by one member, and a group return to home base on location of a heat source.

The final result of the project was four fully autonomous robots capable of exploring, object avoidance, avoiding heat sources, and pausing in the dark. Also, a working master node that could direct the robots in simple swarm tasks.

## Executive Summary (Final Report only)

The goal of this project is to create a swarm and explore swarm behaviors. To do so a group of hexapods will be created. These hexapods are propelled by three servos using very unique synchronized gate. The hexapods power source is 'AAA' lithium ion battery. The robot is controlled by an Atmega328P on a seperate carrier board with integrated voltage regulator. The sensors consist of: two IR proximity sensors and bump whiskers for object avoidance, a CdS cell, and a thermistor for environmental data, a 36kHz receiver used for locating home base. Finally, for swarm communication a bluetooth module is integrated into the robot. A carrier board will be designed to mount all of the sensors, the actuators, and the microprocessor. The carrier board will then be professionally produced by an outside fabrication house.

Each robot will be controlled by a subsumptive architecture. The program is built around an array of structures. Each structure represents a behavior, the structure consists of the walking direction that the behavior suggests, the strength of the behavior decided by the behavior itself, and the priority based on the decisions of other behaviors, and finally a time count that tells how many steps the behavior has been dominant for. An arbitrator then decides which behavior is dominant by multiplying the strength by the priority of each behavior and selecting the highest rated behavior. This architecture allows for the system to be expanded with new behaviors very quickly and easily. Also, the architecture lends itself easily to swarm tasks since each behavior is very compartmentalized; this compartmentalization prevents any contention for control of the robots locomotion.

The swarm communicates through a point to multi-point bluetooth interface. A personal computer with a class 1 bluetooth dongle works as the master node. The rest of the swarm has class 2 bluetooth modules acting as slaves to the personal computer. The communication then works by first selecting a swarm behavior. Then the master asks all of the swarm members for what individual behavior they are performing. When the master finds a swarm member performing the desired individual behavior the master sends out a command to each of the swarm members changing there behaviors. The master is then capable of continuing this swarm behavior, canceling it, or beginning a new behavior.

The group behaviors that the swarm is capable of is a swarm search for the dark. Then once any dark is found the entire swarm freezes. This behavior mimics that of insects, and demonstrates very effectively group communication. The next group behavior is a group search for heat, if a swarm member locates a heat source all of the swarm members run back to base. Other, group behaviors that are controlled completely by the master node is: a group pause where all members pause for a short period before continuing with there previous behavior, and a forced charge where certain members of the swarm are commanded to return to base and begin charging.

# Introduction

Swarm robotics is one of the newest and trendy research areas today, this is due to the fact that only recently swarm robotics has become possible. Swarm robotics involves building many, usually identical platforms that work together through decentralized collective behavior, using self-organized systems to perform a task. These robots are usually not the most technically advanced that robotics has to offer, however the interest of swarms comes from the emergent behaviors that surface when many simple robots work together being governed by a few simple rules. The reason why that swarm research is relatively new, is because in the past it was not economically feasible. Previously microprocessors were expensive, but within the past two decades the cost and size of microprocessor's has plummeted and their power has increased vastly. Now, small simple robots may be created to explore the intricacies of creating a swarm and observing it's behavior.

The goal of my robotic swarm is to explore the idea of randomly searching large areas, for an event or object. Imagine a large multi-floored laboratory that is involved in the research of toxic/explosive substances. It would be extremely expensive to use people to monitor all of the floors and rooms for safety, and time consuming and inefficient if you had just one extremely intelligent robot to check all the rooms for dangers. However, if a few small robots per floor, equipped with a few sensors, a heat sensor for fire, a chemical sensor for dangerous leaks, a pyroelectric sensor for intruders, were allowed to wonder randomly about the building the time it takes for the problem/danger to be found should dramatically decrease.

For this experiment I will make at least four small hexapod robots that can sense their surrounding environment and communicate with one another to perform cooperative tasks, such as searching an area for a heat source, or a burnt out light, and hopefully other tasks that will be discovered throughout the research.

The rest of this paper will explain the physical components of the robot, how their intelligence/behaviors were programmed, and how the individual robots organize and behave with one another.

# Integrated System

The Hexy hardware system is designed to be a very simple system. The proximity sensors have simple digital inputs, the environmental sensors give relatively noise free outputs, motor control is simple PWM controlled.

The software is designed to be very close to the hardware which allows for unprecedented control and flexibility of the robot. The system is based on a subsumption architecture. In the architecture an infinite loop is run where the sensors are checked, the behaviors then use the sensor data to give a suggestion on direction to walk, finally the arbitrator looks at all the behaviors decides which one is most important, and then controls the servos. The servos run asynchronously to the behavior loop allowing smooth movement. The speed of behavioral loop runs at such a high speed that the behaviors appear to be running in parallel.

In Illustration 1 the software structure is demonstrated in a block diagram. The behaviors listed in the block diagram will be further explained in the behavior section.

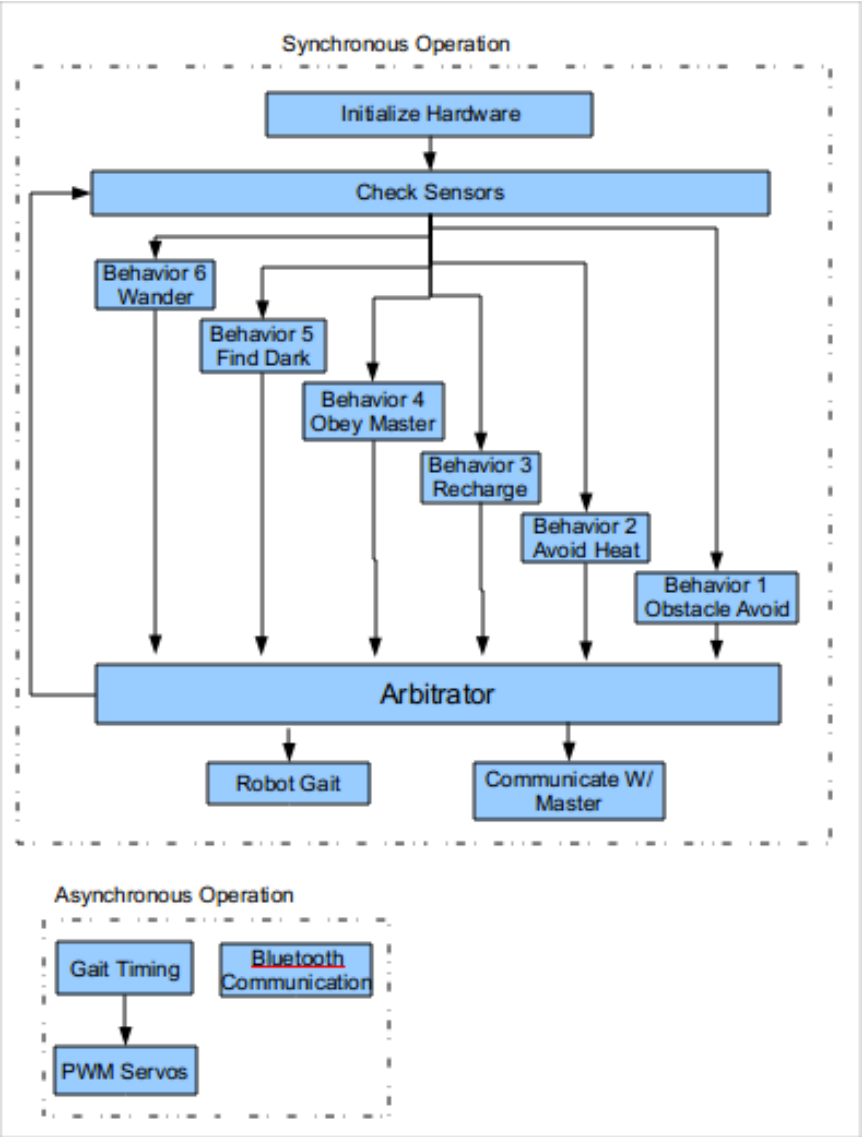



*Illustration 1: Subsumption architecture block diagram.*

## Mobile Platform

The robotic platform will be comprised of a very compact body that consists of the main components:

- A single AAA lithium ion battery to power the robot.
- Three ultra micro servos controlling three pairs of legs for locomotion.
- A microprocessor board for control functions.
- Peripheral sensors to interact with the environment and the swarm.
- Carrier board for the sensors and microprocessor to mount on and then connect to the motors.

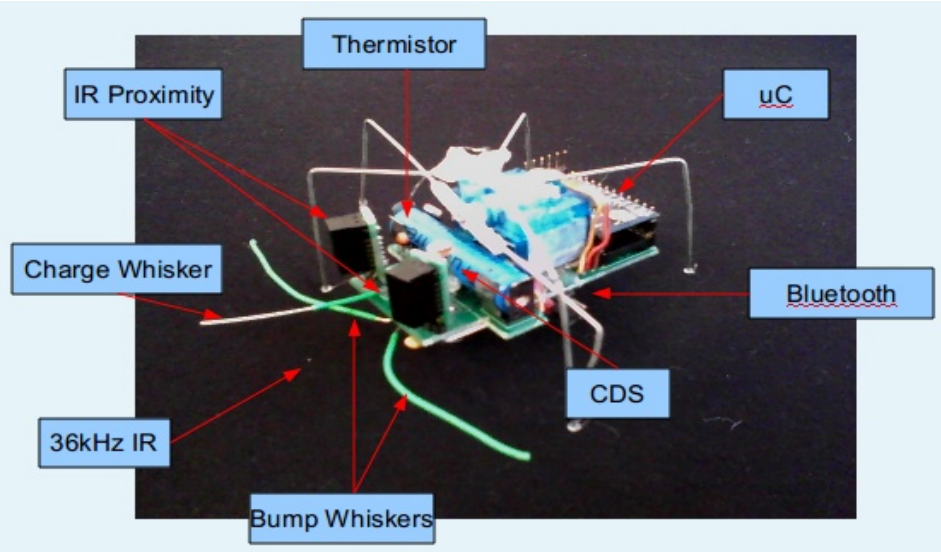


*Illustration 2: Picture of finished hexapod.*

## Actuation

The robotic swarm will be built using very limited and simple actuation. This is done to keep costs down, and because the main goal of the swarm is not complex movement or extreme physical tasks. The propulsion system of the swarm will consist of three ultra micro servos. These servos will each actuate a pair of legs that will provide a simplified insect like form of locomotion. The specifications for the servo are as follows:

- Weight: 3.7g / 0.13oz ( servo only)
- Operating Speed (4.8V no load): 0.12 sec/60 degrees
- Torque  at 4.8V: 0.7 kg-cm / 9.71 oz-in
- Torque at 6.0V: 0.8 kg-cm / 11.1 oz-in
- Operating Voltage: 4.8-6.0 V
- Size: 20.0 x 8.5 x 19.8 mm

As you can see the voltage is rated only for 4.8V at it's lowest. However, through experimentation the servo has shown that it still runs quite robustly even down at 3.6V. This is very important due to the fact that I will be running my entire system off of a 3.7V lithium ion battery. Also, servos are required to accept TTL level control signals so a 3.3V PWM signal is not a problem.

The servos will be controlled using three of the six waveform generators provided on the mega328P. It will use the two wave form generators on the 16 bit Timer 1, and one of the waveform generators on Timer 0. The prescaler of Timer 1 will be set to 8. With the clock running at eight megahertz, after the prescaler the Timer 1 will be running at 1Mhz. Timer 2s prescaler will be set to 1024 resulting in a clock of 7812hz. The servos require a 50Hz wave so  Timer 1 will be set  to 10,000 in phase and frequency correct mode, and Timer 0 will be set to 156 in Fast PWM mode.

# Sensors

## *Bump Sensors*

Bump sensors, or whiskers, will be used on the front of the robots for the purposes of charging and object detection. Although, the IR proximity sensors that will be used on the robot will be adequate for most object avoidance, there is a chance that a extremely thin or short object may not be seen. The whiskers then provide nearly a failsafe method of preventing the robot from getting stuck on an obstacle, if the IR sensors fail to see the obstacle.

The whiskers that will be used will be created from stiff wire and pin headers. It was found that traditional buttons increased size and cost, and were not effective at the low forces that the small robot will produce.

The wires used for whiskers will form a sort of antennae off the front of the robot, as seen in Illustration 3, which is very fitting since it will mimic an insect.

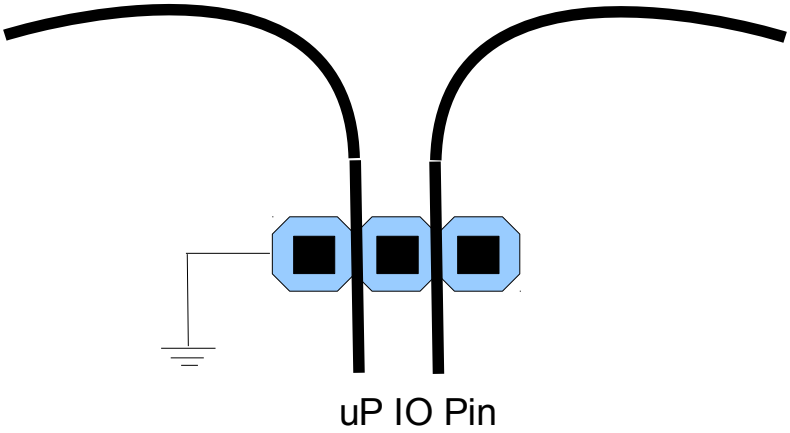



*Illustration 3: Whisker Design.*

The wires will be connected to one of the microprocessors digital IO pins. This pin will be internally drawn high using the internal pull up resistor. The stiff non sheathed wire will then lay in between two header pins, that will be connected to ground. Thus if the antennae is bumped at all it will cause the antennae to be connected with ground causing the microprocessors IO pin to be pulled low. At that point the microprocessor would acknowledge the fact that an object had been hit and act to free itself from the obstacle.

## *IR Proximity Sensors*

Two IR proximity sensors will be used on the robot to do majority of obstacle avoidance. The exact sensors I will be using are the Sharp GP2Y0D810Z0F. These sensors were chosen because of there extremely small form factor, and they provide very good performance. The sensors will be used on a carrier board by Pololu, these boards provide all of the necessary external components for proper operation (ie, resistors and capacitors), and minimizes the fourteen pin part to three 0.1 inch pins. This makes mounting very simple and effective.

The GP2Y0D810Z0F is a very different IR proximity detector then other detectors in the past. It

provides only a binary digital signal out. This is both a pro and a con. The good part about this simple on or off behavior is that it makes interfacing very simple because there is no analog to digital conversion that is required by most IR sensors. However, due to it only giving two values the sensor can give no real information about the distance of an object other then whether or not the object is within its ten centimeter range. I considered this to be an acceptable part though because the robots do not have wheels, and because of this they can stop and turn on a dime. Also, the ten centimeter range is just slightly longer then the entire robot, and will keep the robot a safe distance away from any objects it may get tangled up in.

The performance of this device is spectacular and does not seem to be influenced by outside environmental factors such as sunlight, and object color. This is due to the fact that the GP2Y0D810Z0F identifies distance using a triangulation method rather than reflectance intensity like past IR sensors.

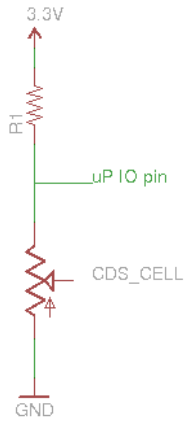
It should be noted that a capacitor may be needed to be placed between the power and ground of the sensor to provide the chip with a clean enough power supply to not cause erratic behavior. This was the case while testing the chip on a breadboard using a 5V wall wart.

Further technical information about the GP2Y0D810Z0F may be found on the data sheet in the appendices.

Upon testing the sensors in the final product it was found that the distance for sensing is slightly below the specified distance of 10cm. On average it sensed objects within 7cm, I believe this is due to the fact that the sensor is run on 3.7V instead of the tested 5V.

## *CdS Cell*

CdS cells, otherwise known as light detectors, are one of the two environmental interaction sensors that will be on the robot. A CdS cell is a semiconductor component that changes it's resistance as light intensity varies.  It will be set up as a simple voltage divider with R1 equal to 2kΩ and input into an analog input of the microprocessor as seen in Drawing 1.




*Drawing 1: CDS cell voltage divider schematic.*

To use the cells you must know the ambient light intensity of the room, or else you would not be able to tell when there was a change in light intensity from the normal. To do this you must have a thresh hold

set in the microprocessor every time the robots are turned on. This threshold is set and from then on all values will be compared to that threshold value to see if any negligible light change has occurred.

The CDS cell was tested in various lighting conditions to make sure the sensing was robust. The data from the testing can be seen below in table 1.

| **Light Type** | **ADC Value** |
|---|---|
| Sunlight | 252 |
| Sunlight covered by hand | 214 |
| Fluorescent | 244 |
| Fluorescent covered by hand | 215 |
| Incandescent | 241 |
| Incandescent covered by hand | 212 |

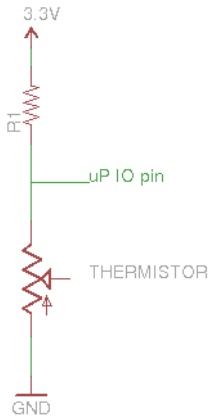
*Table 1: CDS cell ADC outputs in various lighting conditions.*

As can be seen in table 1 the outputs are relatively close. So, all that is necessary to make the CDS cell consistently is for the CDS cell to take a threshold value when the robot turns on, and then assume that the nominal lighting will not change.

As the final robot was finished it was found that the bluetooth module put a large amount of noise on the analog signals. To overcome this noise a four point averaging filter is run on the sensor to provide a stable noise free output.

## *Thermistor*

Thermistor used for temperature readings, are one of the two environmental interaction sensors that will be on the robot. A thermistor is a semiconductor component that changes it's resistance as the temperature varies. It will be set up as a simple voltage divider and input into an analog input of the microprocessor as seen in Drawing 2. In the voltage divider R1 is set as 10kΩ.




*Drawing 2: Thermistor voltage divider schematic.*

To use the thermistor you must know the ambient room temperature, or else you would not be able to tell when there was a change in temperature from the normal. To do this you must have a thresh hold set in the microprocessor every time the robots are turned on. This threshold is set and from then on all values will be compared to that threshold value to see if any negligible temperature changes have

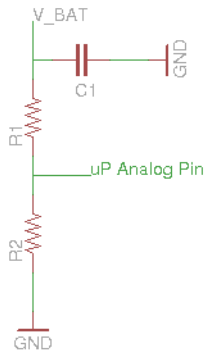
occurred.

The thermistor variations are minimal since the temperature difference that I will be a able to provide with a blow dryer is small. So, on startup a threshold value will need to be set. Then when the voltage measurement from the ADC goes above the threshold the robot will be said to have found a heat source/fire.

As the final robot was finished it was found that the bluetooth module put a large amount of noise on the analog signals. To overcome this noise a four point averaging filter is run on the sensor to provide a stable noise free output.

## *Battery Voltage Monitor*

The battery voltage monitor is a simple internal sensor to provide information on battery power that is left. It is just a simple voltage divider connected to an analog input of the microcontroller . The interesting part of this sensor is using the the lithium ion discharge graphs to accurately tell the robot when it needs to go and find a charging station, and still have power to get there. The schematic can be seen below in Drawing 3.




*Drawing 3: Battery voltage monitor schematic.*

The resistor R1 will be selected as 50kΩ to minimize current draw. The capacitor C1 is present to help stabilize the voltage read at the A/D pin because the servos will cause a large amount of noise on the power plane. The value of the capacitor C1 will be 1µF. To find the value of R2, equation 1 below will be used :

$$R_2 = \frac{R_1}{(V_{Bat}/V_{Logic}) - 1} \quad (1)$$

$V_{bat}$ is the maximum value of the battery voltage. For the lithium ion batteries the maximum battery voltage 4.2V. $V_{logic}$ is the voltage value that will make the D/A converter read out 100%, this value will be 3.3V. This results in $R_2$ being equal to 165kΩ, and since that is not a common resistor value a 160kΩ resistor will be used. This will cause the max output voltage to be 3.27V, which is an acceptable value.

This analog sensor was found to be surprisingly noise free even though it was connected to an unregulated power source. Still an averaging filter was added to the sensor to reassure robust operation, at very little CPU cost.

### *38kHz IR Sensor*

This sensor will be used to find home base, or possibly other robots. The exact device is the TSOP34336 from Vishay electronics. It is a very simple 3 pin device. Provide it with power and ground, and it will output a digital signal.

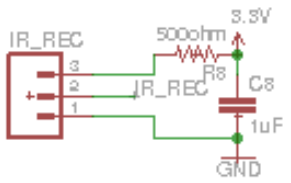



*Illustration 4: 36kHz Schematic.*

The signal is low when it does not find a 36kHz signal and high when it does. This signal will be connected to one of the digital IO pins of the microprocessor. Then it will be able to find it's charging station by locating the 36kHz IR beacon that will be placed on it, and discover direction based off of the charging station.

This sensor provides very robust output. It can sense a strong 100mA IR LED from about one meter away if the sensor is pointed very accurately at the emitter. For the particular application a lower power emitter will be used lowering the range to about one 30cm.

### *Bluetooth Communication*

The blue tooth communication module will be the interface through which the swarm will communicate. The module that will be used is the Roving Networks RN-42. It is a class 2 bluetooth module appropriate for up to ten meters communication. It will communicate solely with a host computer, however to send inter robot messages the host computer can relay the message to the correct robot.

Connection with the RN-42 is extremely simple requiring only a receive and transmit line from the microprocessor, and the data flow control lines are linked to each other. The RN-42 has a default configuration of 1 start bit, 8 data bits, and 1 stop bit, and transmit rate of 115.2kbps. The reset line will also be connected to the microprocessor as a debug tool. The factory reset pin and 9600 baud rate are broken out if they're necessary. To provide more accurate communication the 9600 baud rate will be used.

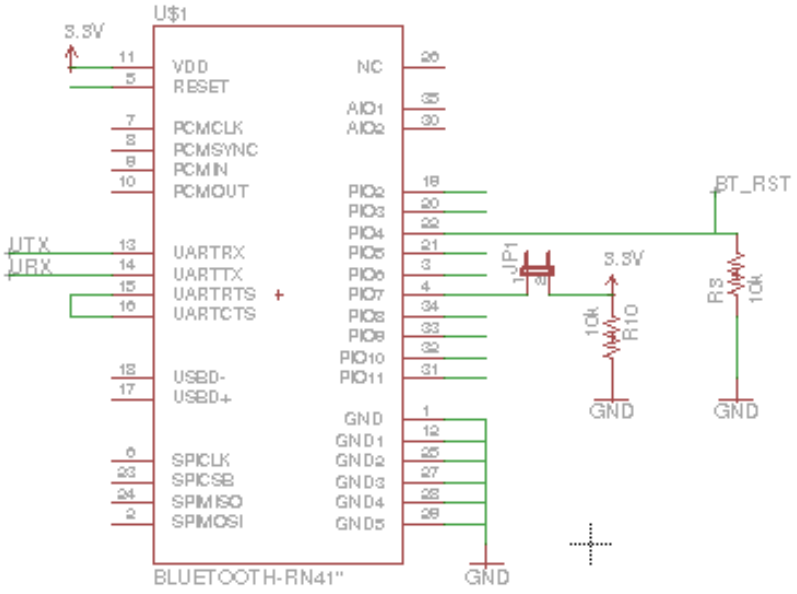



*Illustration 5: RN-42 Schematics.*

The master computer uses a class 1 dongle. It will use the open source rfcomm bluetooth framework to connect with all of the swarm modules. A custom framework programmed in C using rfcomm will be developed to relay messages between swarm members, and allow direct inputs from myself.

## *36kHz Beacon*

This is not necessarily an on-board sensor. However, it is a necessary part of the robot for the 36kHz sensor to work effectively. A 555 timer is used to create a 36kHz square wave then this is output to an IR led. The pulsed IR light can then be sensed by the IR sensor on the robot. The schematic for this circuit can be seen in illustration 6.

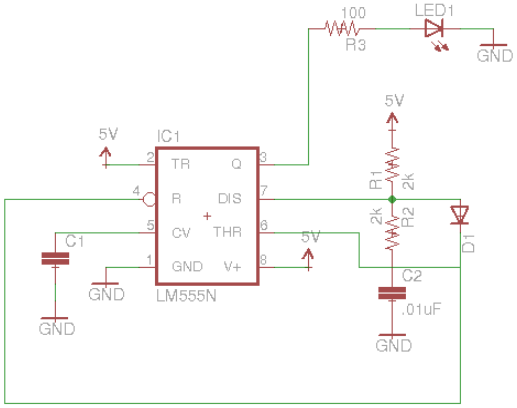



*Illustration 6: 555 timer 36kHz beacon.*

## Behaviors

Hexy's behaviors are programmed as a subsumptive architecture. That meaning that the more important behaviors subsume the less important behaviors. The behaviors that Hexy has in order of importance is:

1. Wander – the simplest behavior in which Hexy just walks randomly in a direction.
2. Find Dark – This behavior relies on the wander behavior to find the dark. However, once a dark area has been located Hexy will rest for a short period of time before wandering again.
3. Obey Master – This behavior allows for the master, and thus other members of the swarm to determine what Hexy will do. It may just control what behavior of Hexy takes control, or may take direct control of where the robot walks.
4. Recharge – In this behavior Hexy realizes that its battery power is running low, and thus will use the 36kHz receiver to search for home base. Once it has found home base it will run into the charging station with its charging antennae and charge until it is full.
5. Avoid Heat – Once Hexy has found heat it will turn around and avoid it assuming it's fire. Then Hexy will inform the rest of the swarm of the fire and then the swarm will act on the information together as a swarm.
6. Avoid Obstacles – The lowest level is just for Hexy to avoid running into objects. Hexy uses its two IR detectors, and whisker sensors to avoid objects.

The hardest obstacle to overcome while programming the subsumptive architecture was to make everything simple. By making everything interrupt driven for maximum speed I ran into issues where behaviors would fight for control over the motors. By realizing that even a very inefficient loop would still run much faster then 30Hz, a discreet time system was created and the system works very well.

## Experimental Layout and Results

For testing the swarm an arena was created, the layout can be seen below in illustration 7.

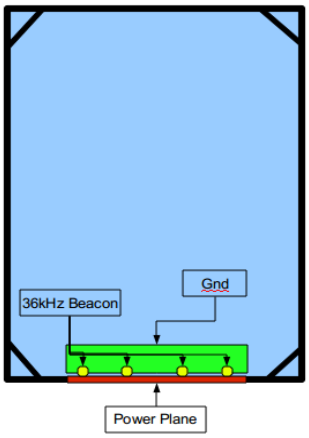



*Illustration 7: Arena.*

This provided a boundary so that obstacle avoidance can be demonstrated. Also, 36kHz IR beacons were placed along four spots along the south side of the arena. The side with the beacons also has a ground plane/plate and a power plane/plate that is used to recharge the swarm.

The first step of testing is the power to each robot of the swarm is initiated. Then the swarms bluetooth communication is linked to the computer master node. Once, the bluetooth link is complete the swarm is activated by touching each of there front bump whiskers. Here the swarm begins in wander mode. In this mode the robot will explore the area avoiding running into any object. Once it locates a dark area the robot freezes and waits thirty seconds before beginning to wander again.

From this point swarm interactions may be initiated. The swarm behaviors that will be tested are a swarm dark reaction, or when one of the robots locates dark the entire swarm stops. Or, when one of them finds a heat source the rest of the swarm will run back to base. Finally, the charging interface will be demonstrated by remotely forcing a robot to believe it is low on charge then allowing it to find home base and connect to the charging plates.

# Conclusion

What I've accomplished during this project in a short summary is: Designed a surface mount professionally produced PCB that is used for integrating the sensors and actuators to a separate microprocessor board. Developed a complex subsumptive architecture for creating a CPU efficient and behavioral complex system. Integrated three servos into a working gate of hexapod propulsion. One of the servos was controlled using an 8 bit timer which is surprisingly difficult due to resolution issues. Finally a simple yet effective point to multi-point swarm communication was created using bluetooth and a PC master node.

One of the greatest limitations of this project was, each individual swarm member is unaware of the location of the other swarm members. Also, some of the sensors limit the ability of the robot due to some inconsistent output.

The bluetooth communication was extremely efficient. Also, the mobility of the robots exceeded expectations. The hexapod gates are extremely efficient and I am quite satisfied with them. However, if need be they could probably be improved even further.

Areas of improvement are the circuit board power traces could be routed more robustly. There were a few issues with the IR proximity sensors drawing too much power. Also, even though the IR proximity sensors and the 36kHz beacons frequencies were separated by at least 10kHz. Occasionally the beacons were able to activate the proximity sensors at close proximity. Also, the bump sensor whiskers need to be redesigned to integrate the charging contacts. In a nutshell another revision of the hardware would be necessary to work out majority of the major issues.

If a student were to follow up this project a goal would be to create a more robust bluetooth point to multi-point protocol could be developed by future students.

If I would restart this project I may have decided to use wheels instead of the hexapod. It would have simplified many aspects of the platform and movement. This includes the weight concern caused by the hexapod structure only being able to support a small amount of weight. Also, due to the gate of the hexapod being very integral to the behavioral design all of the behaviors had to be programmed around a discreet time base that was dictated by the gate of the robot. Had wheels been used less time would have been devoted to the hexapod propulsion and it would have allowed the robot to perform more complex swarm behaviors.

However, if this project were to continue the largest change would be a major overhaul of the circuit board. I would integrate the microprocessor onto the actual sensor board. This would not only reduce robot weight but allow for a cleaner more efficient board design. Also, I would have looked into the possibility of using an XBEE instead of bluetooth because of its advantages of ad-hoc multi point networks. This brings up the possibility of using a combination XBEE general microprocessor chip offered by AVR. This would allow more weight and power savings.

To expand upon this project I would like to integrate some sort of positional system. Either, a positional system based off the position of one robot to another. Or a positional system based off some sort of markers on the ground allowing for the robot to create a positional grid.

The benefit of running the swarm as a master and slave configuration is the ability to quickly change the behavior of the swarm without having to change any of the code on each robot. The swarm can be programmed to perform different behaviors very quickly assuming that the master only combines abilities that each member already has available. I believe that the approach that I have taken to swarm behavior is new, or at least to me. Each bot is completely autonomous not requiring a master node to perform individual tasks and behaviors. However, when the master node is present it allows the robots to be aware of the rest of the swarm and then larger tasks can be performed.

# Documentation

## *Bibliography*

Society of Robots. [Online]. Available:
http://www.societyofrobots.com/schematics_batterymonitor.shtml

## *Appendices*

## *Main Code*

See website. Http://plaza.ufl.edu/wmcpatterson