

Build a DoggyDish Water Monitor  
as seen in  
May 2007 of Nuts & Volts Magazine

Pick up an issue at  
<http://www.nutsvolts.com>



Last month I gave you details on how to communicate with various X10 computer interfaces. This month I had planned on showing you how to use each one of those interfaces with a microcontroller.

I want to deviate a bit and actually show you how to build a cool project using the CM17A FireCracker.

Over the last 10 years I have built many different automatic dog watering dishes. Some worked pretty well and others did not. Inevitably all dishes eventually failed and one conclusion I came to was that it was not practical to automatically fill the dish; as eventually all the systems failed and I ended up with a mess. What I wanted instead was a system that would let us know when the dish needed to be filled. I came up with the following project requirements:

- "No probes are to come in contact with the water.
- "No Floats or other items are to be placed inside the dish.
- "The system must be portable and able to be installed under the dish.

While playing with the Firecracker and the microcontroller interface the idea for this project came to me. The Firecracker (CM17A) is small and gets its power from the actual control leads. The interface is simple and requires no external components. All that was left was to come up with a way to detect the amount of water in the dish.

The easiest way would have been to sink a couple of leads into the water, but this is not safe for the animals no matter how low the current and voltages. Floats are a pain and get in the way. In the past I had used a special platform that measured the weight of the dish. While this system worked well, it was not portable and the mechanics were prone to failure.

I tried using sonar and IR sensors to detect the depth but they were a failure as well. Then I remembered an experiment I performed about a year ago using an IO IRQ as a counter. I set up a DiosPro microcontroller with one of its leads as an IRQ counter and created a routine that would increment a variable each time the IRQ fired.

During one of my experiments I found that if I toggled one of the other ports while I was touching the IRQ port it would detect the toggled port. My body was acting as a giant antenna for the very sensitive IRQ port and just about any electrical activity would cause the IRQ to fire.

To expand on the experiment I held the IRQ port low with a 100K resistor. This would reduce the sensitivity, but when I touched the IRQ port nothing happened. If I touched both the IRQ port and the toggling port the IRQ would fire with each toggle. I then connected large metal plates to both the IRQ port and the toggle port. I placed them very close without touching and the toggle port would cause the IRQ port to count. This is exactly what I was looking for so I placed the plates against a plastic jug and when filled with water the IRQ port would pick up the toggle port level changes. It's the results of this experiment that I am basing this project.

## Building Sensor Pads

It's time to create a couple of sensor pads that you can use for your own experiments and eventually use on your own doggy dish project. You will need some foil tape. This is the metal tape that you can purchase from the heating and air-conditioning section of your home center. One side is conductive and the other has a very strong adhesive. Take an 8" piece of foil tape and split it down the middle as shown in Figure 2.



Figure 2

Next, take a 12" piece of solid hookup wire and strip about 4" from the end. Lay the wire on one of the tape strips foil side up as shown in Figure 3.



Figure 3

Take the other tape strip and remove the adhesive backing and place it over the wire and original strip as shown in Figure 4. Use your thumb to smooth out the bubbles and to make sure the wire is firmly sandwiched between the two strips. You now have a Sensor pad. You will need two of these sensor pads.



Figure 4

For now attach the two sensor pads to a jar similar to the one shown in Figure 5. Use cellophane tape so they can be removed later.



Figure 5

## Controller Board Construction

Before we can proceed with the first test we need to build the controller board that will take readings and send out our X10 control codes. We will use a DiosPro 28-pin chip, a Dios Carrier1 board and a FireCracker X17A. I will provide a complete list of materials at the end of this article along with sources for the various components.

### Step 1 - Build the Dios Carrier board

Using the instructions that come with the Dios Carrier 1 board, assemble it. Don't assemble the two 12-pin headers. When complete it should look like Figure 6.

Install a 1-pin header into port 7. Install a 1-pin header into port 13. Install a 2-pin header into the two pads marked - and + near the upper right hand of the board shown in Figure 7. Install a 5-pin female header into the pads marked -, +, Rx, Tx, Atn in the lower left hand corner of the board. This header will be used to insert an EZRS232 for programming the Dios Pro.

Install a 100K resistor between IO port 7 and Vss as shown in Figure 8.

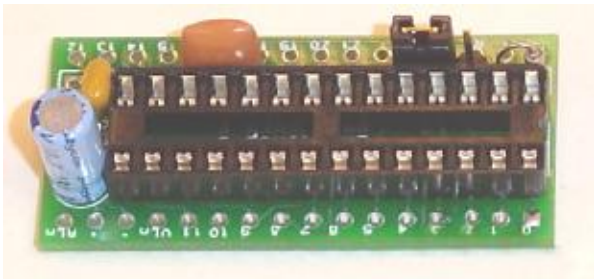


Figure 6

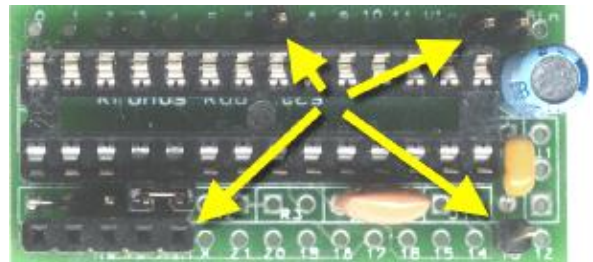


Figure 7

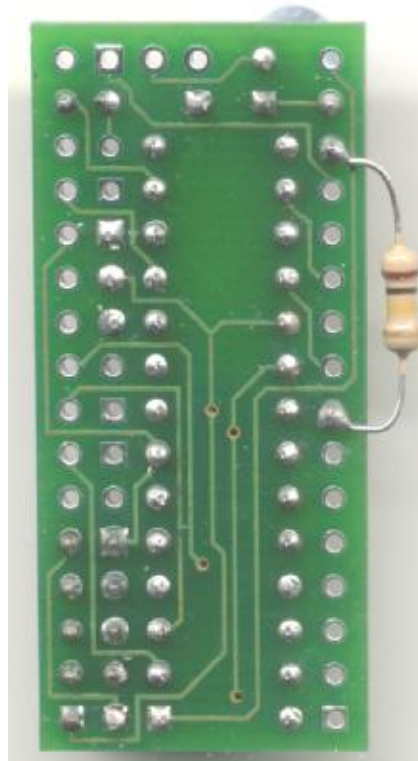


Figure 8

## Step 2 - Attach the DB-9 connector

Break off a 2-pin header from one of the headers that came with the Dios Carrier 1 and solder it to the pins 4 and 5 as shown in Figure 9A. Then break off a single pin and solder it to pin 7 as shown in Figure 9B.

Attach the DB-9 connector to the Dios Carrier 1 as shown in Figure 10. The header pins 4 and 5 are soldered on the top of the board to IO ports 2 and 3. The header pin 7 is soldered on the bottom of the board to IO port 0. This connector will eventually connect to the FireCracker.

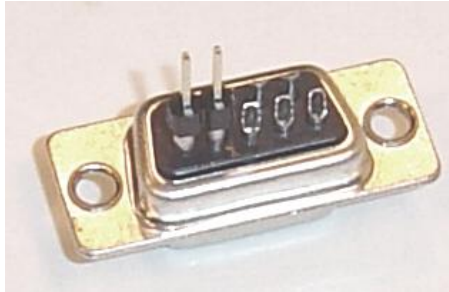


Figure 9a



Figure 9b

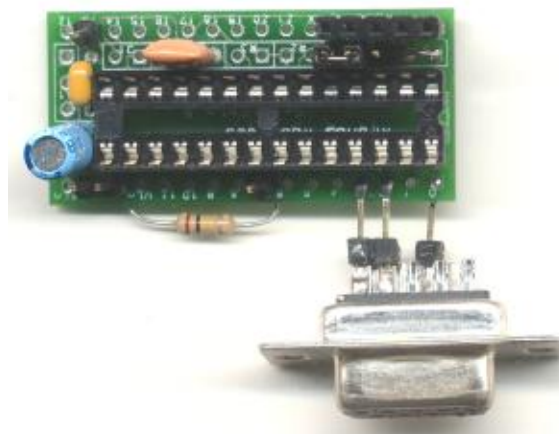


Figure 10

## Step 3 - Final Prep

Move the red wire on the battery connector so that it is located next to the back wire as shown in Figure 11. To do this you need to pry up the small tab covering the pin. Keeping the orientation of the receptacle the same, push it into the slot next to the black wire. It should snap into place.

Snap off a single 1-pin section from the 36-pin female header and solder the end of the wire connected to the Sensor Pad to this female header. For more strength slip a piece of 1/8" heat shrink over the pin and shrink as shown in Figure 12.

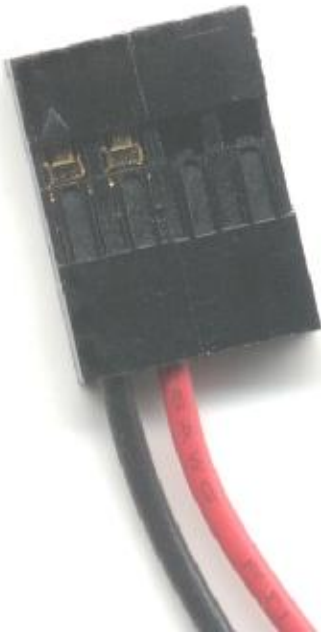


Figure 11

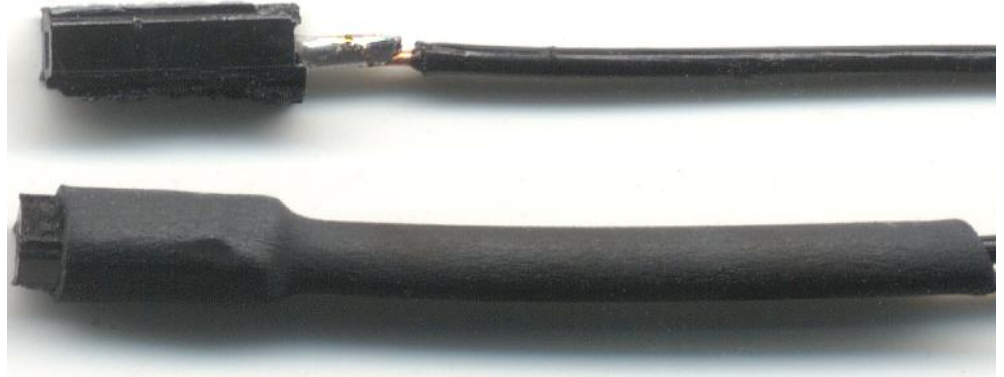


Figure 12

#### Step 4 - Sensor Pad Test

If you have not done so you need to assemble and test the EZRS232 board according to the accompanying instructions, then download and install the free Dios compiler.

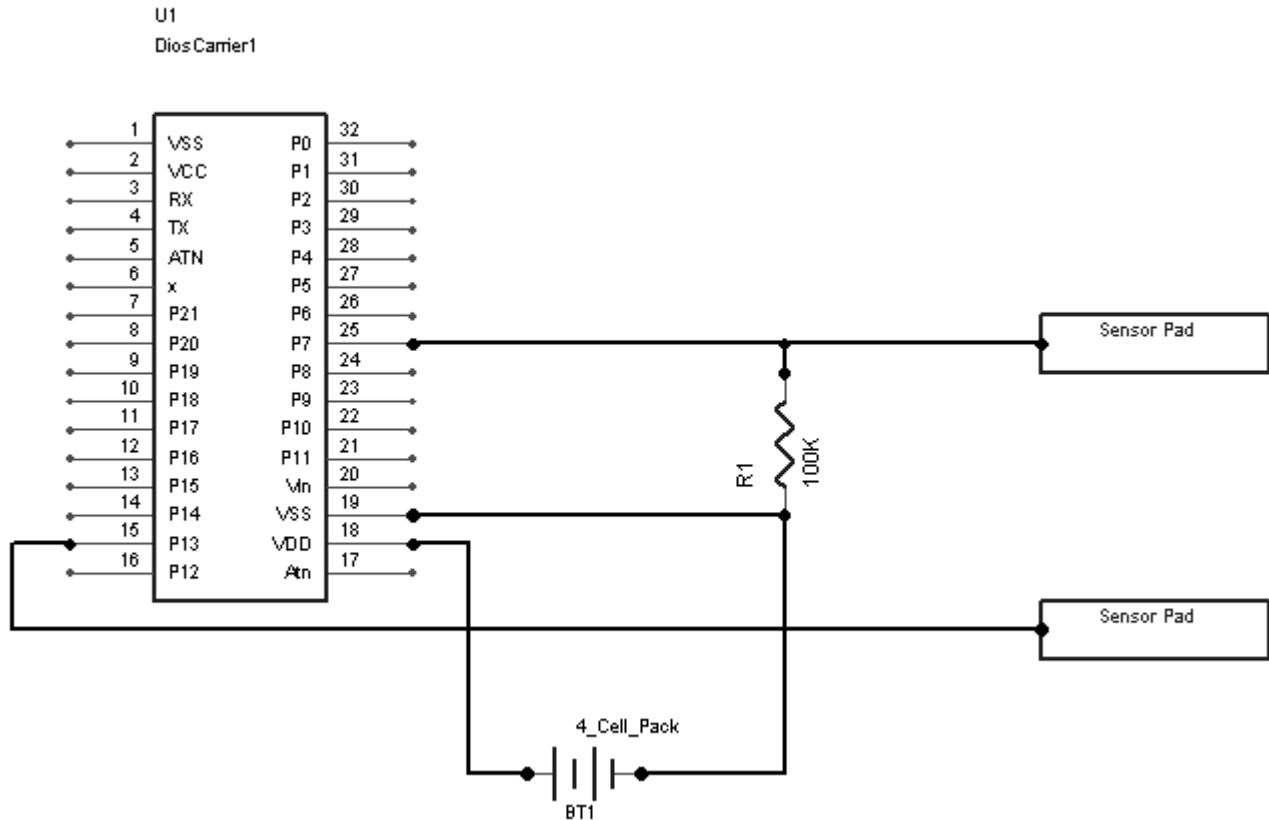
Insert the DiosPro chip into the carrier, then plug the EZRS232 driver into the 5-pin female connector on the carrier and connect the battery connector as shown in Figure 13. Make sure the orientation of the black and red wires are correct. If they are not, you will damage the DiosPro chip.

Connect the carrier to your PC by using a 9-pin serial cable. All new DiosPro chips contain a small test program that will display a test message on the debug terminal.



Figure 13

Now connect each of the sensor pad female headers to the single pin male headers on IO ports 7 and 13 that we installed earlier. Schematic 1 shows the complete hookup for this experiment.



Schematic 1

From the downloads program DDishP1.txt into the DiosPro. For The Dios Carrier 1 you will need to use an EZRS232 driver. The DiosWorkboards have the RS232 driver built in so experimentation is much easier.

Once you run the program it should start reporting 0 in the debug window. Start to fill the jar or container and as the water reaches the top Sensor Pad the reading should change to 50.

The program is simple enough. The startirqasm is a simple assembly routine that fires each time there is a change from low to high on port 7. The routine increments the global variable dishvarb. The DishTest routine is called each time you want to test the water level. What it does is toggle the port 13 100 times. This is 50 low to high transitions and 50 high to low transitions. Upon entry to the routine I set the global variable dishvarb to 0 so we get a fresh count with each call.

Play with the location of the sensor pads as well as the 100K resistor. I have found that you may also place the two sensors across from each other on the container.

### Step 5 - Doggy Dish Assembly

Dish selection is important. You need to select a dish that has some sort of chamber on the under side. This chamber will be used to house the DiosPro, Fire Cracker and Battery Pack. The bowl shown in Figure 14 is a very common bowl available at most pet and department stores and is available in many sizes and colors. You may also make your own bowl by attaching two bowls as shown in Figure 15. The best way to attach the two is with some industrial double stick foam tape or hot glue or both.



Figure 14



Figure 15

Remove the sensor pads from the jar and using cellophane tape attach the two pads near the bottom of the doggy dish bowl across from each other as shown in Figure 16.

Set the bowl right side up and connect the sensor pads to the DiosPro and power it up by plugging in the battery connector. Slowly fill the dish with water until the readout starts to display 50. Using a small cup remove a small amounts of water until the display reads 0 once again, this is the point that the dish will alert you when it needs more water.



Figure 16

If you aren't happy with the level reposition the sensor pads until you are.

### Step 6 - Test the FireCracker Interface

Connect the FireCracker to the DB9 connector that you attached to the carrier 1 board back in Step 2 as shown in Figure 17. Load up the program DDishP2.txt included in the downloads and program it into the DiosPro using the EZRS232.



Figure 17

The program is set to turn device K3 on and off depending on the water level. You can change this by modifying the constants used in the call to the SendX17Adata routines. Connect the sensor pads if they are not already connected and add water to the bowl. The K3 device should turn off as long as the water level is ok. It will turn on when the water level drops below your sensors. The program tests the dish every 10 seconds so it's plenty fast enough to test your dish.

### Step 7 - Final assembly

By this point we have tested the control system, the sensors, and the water levels and are happy with their operation. It's much harder to program the DiosPro once it has been installed under the bowl.

Dry fit the controller and battery holder first by holding them in place. Make sure you can reach the board with the sensor pad wires as shown in Figure 18. Don't install the batteries and controller too close to any of the sensor pads or they may interfere with the readings. Position the Carrier 1 board so that you can still install the EZRS232 when needed.



Figure 18

Once you are happy with the location of the controller, place some double stick foam tape on the FireCracker and stick it in place. You will also want to attach some foam tape to the Carrier 1 board as well. The board sits higher than the FireCracker so you will need 2-3 layers of the foam tape for the added height. Attach the 4-cell battery holder next. You may also want to use some tape to hold the sensor pad wires in place. Make sure you don't use the metal tape for this.

Load up program DDishP3.txt and program it into the DiosPro. This program takes advantage of a few of the hardware features built into the DiosPro. The DiosPro has the ability to be put to sleep and will draw only about 40us of power from the battery in this mode. To wake up the DiosPro we use another feature called the watchdog timer. When turned on it will wake up the DiosPro after it has been asleep for 2 seconds. If you look at the code you will see that I make 130 calls to the sleep command. This causes the DiosPro to stay in a low power mode for 5 minutes. Once it wakes up, it does a dish test then turns the X10 device on or off depending on the reading.

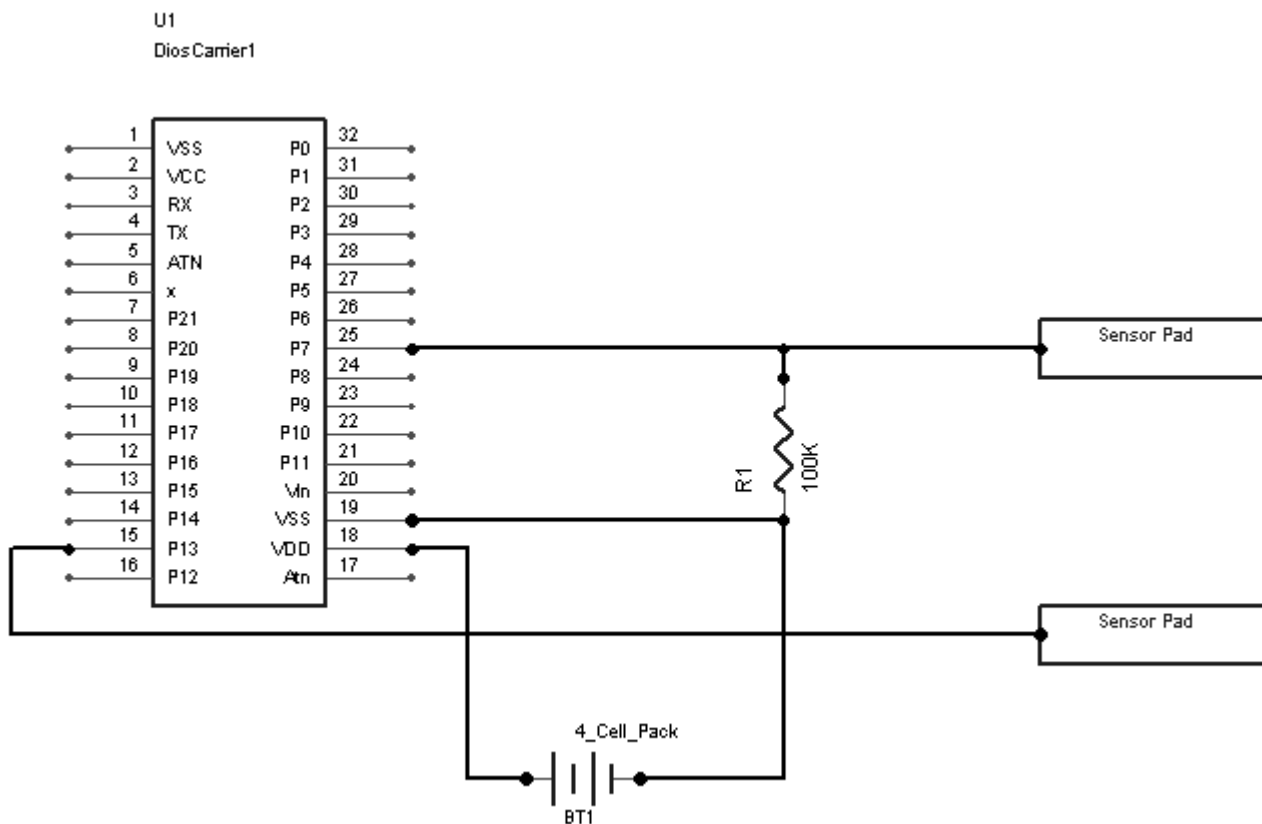
We are providing the power to the FireCracker, so just before we put the chip to sleep the power is removed so that the FireCracker itself does not drain the battery.

Another hardware feature of the DiosPro is the ability to self monitor the supply battery. We make a call to this routine once each cycle and if the battery voltage drops below 4.24 volts it will toggle the device 5 times to warn us.

## Final Thoughts

I have been using the dish now for a couple of months and it has worked without failure. The alkaline batteries I have been using also show no signs of significant drain.

The complete schematic for the final assembly is shown in Schematic 2. Feel free to make modifications. There are several ports available for connecting things like LED's, beepers or other devices. You could also run a couple of additional sensor pads to test different water levels. For instance, use Port 14 to run a second toggle pad. Just make an additional call to the DishTest routine passing IO port 14.



Schematic 2

As for the water level warning indicator, just about any lamp will work. I use something a little different. I use one of those strobe flashers connected to an AC adapter as shown in Figure 19. I'm using the standard lamp module that came with the CM18 FireCracker kit. The strobe will flash whenever the dish needs more water.



Figure 19

The FireCracker - also known as the CM17A, is not an X10 device, nor does it communicate with the X10 protocol. It is a wireless transmitter that is designed to work with the TM751 Transceiver Module. While you can purchase the CM17A by itself, it won't do you any good unless you already have a TM751. This is why they created the CM18 FireCracker Kit. This kit contains the following modules:

- FireCracker Module (CM17A)
- Transceiver Module (TM751)
- Lamp Module (LM465)
- PalmPad Remote Control (HR12A)

Since the FireCracker is so easy to use with both a PC and microcontroller I recommend at least one CM18Kit. After that you may purchase additional CM17A units for as little as \$12 each for various projects.

As for projects using a CM17A and microcontroller, they are endless. I have already started on a sonar car parking aid for the garage.

All the example programs as well as the source are available for download at:  
<http://www.kronosrobotics.com/Projects/x10b.shtml>

## Parts

Available from Kronos Robotics at [www.kronosrobotics.com](http://www.kronosrobotics.com)

### DiosPro28

<http://www.kronosrobotics.com/xcart/customer/product.php?productid=16429>

### Dios Carrier 1

<http://www.kronosrobotics.com/xcart/customer/product.php?productid=16170>

### EZRS232 Driver

<http://www.kronosrobotics.com/xcart/customer/product.php?productid=16167>

**4-Cell Battery Holder**

<http://www.kronosrobotics.com/xcart/customer/product.php?productid=16323>

**36-pin female header**

<http://www.kronosrobotics.com/xcart/customer/product.php?productid=16291>

**9-pin DSub Plug**

<http://www.kronosrobotics.com/xcart/customer/product.php?productid=16254>

**100K Resistors**

<http://www.kronosrobotics.com/xcart/customer/product.php?productid=16195>

**HeatShrink**

<http://www.kronosrobotics.com/xcart/customer/product.php?productid=16288>

Available from [www.X10.Com](http://www.X10.Com)

**FireCracker**

(Note the FireCracker can be purchased cheaper from an eBay eStore.)

Available from All Electronics at [www.allelectronics.com](http://www.allelectronics.com)

**Strobe Light**

<http://www.allelectronics.com/cgi-bin/category/605/Strobes.html>

Available From Home Center

**Foil Tape**

Standard HVAC metal tape

**Double Stick Foam Tape**

Used to secure the Dios Carrier 1 and FireCracker to the inside of the bowl.

**Hookup Wire**

I used wire removed from standard telephone cable (Cat 3)

Available From Pet Store

**Water Dish**

I purchased the bowl I used from PetSmart. I have seen them at various department stores as well.