DiosWalker

By Michael Simpson
Updated on August 29, 2006

Many of you have seen my FaceWalker that was featured in the August Issue of Servo Magazine. While this project was a great success I wanted to port the IK routines over to the Dios so that a much more inexpensive version of the walker could be built. In this article we are going to take it down to the bare bones. As you read through this article please note that much of the content from the original FaceWalker article was used. It all applies here and saved me loads of time. This is kind of like the statement "Please return your shopping carts so that we may keep our prices low"

Part 1
Base Construction

There are two key components that make up the DiosWalker.

- The base, which consists of the LynxMotion EH3-R and a SSC-32 servo controller and support components such as batteries and connectors and Pocket PC mount.
- A DiosPro microcontroller that controls the walker movement and interfaces to a wireless PS2 Controller. Even if you want to build a fully autonomous robot walker I still recommend the PS2 controller. This will help you test and place the walker in specific modes. A DiosPro starter kit will be used here. This small carrier board only takes up .75" x 1.5" of space and gives us plenty of functionality.

The Base

No matter what kind of robot you build, you have to build a base that is strong enough to hold all the components and batteries. I calculated that I would need a base that would support almost 10 Lbs of total weight. For a walker this consideration is more critical than a track or wheel based robot since the legs have to support the weight and still be able to articulate.

While I was searching the web I found plenty of walker bases but none stood out like the LynxMotion EH3-R Hexapod base shown in Figure 2. The EH3-R is a special round version of their Hexapod. The main advantage of a round walker is that it is omni directional. It can move freely in any direction. The other advantages are that we can take care of the IK routines developed by to Laurent Gay. I have ported these routines to the Dios as a library. More on this later.

I decided to go with the EH3R and 18, HS645 servos. The HS645 servos are much more expensive but I wanted something that would hold up to some major experiments.
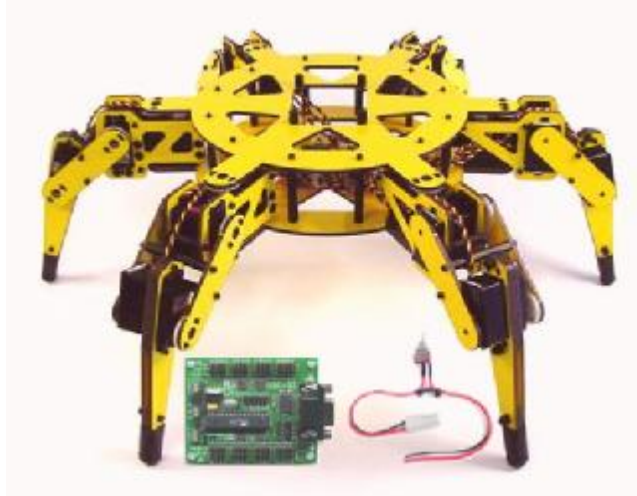
Figure 2

I started sourcing other components I would need for the base. I will list the source and location for all the items I used to create the DiosWalker at the end of this article.

Figure 3 shows the additional components I needed: 7.2v Stick Battery, 6-Cell AA Battery Holder, 9v clip, 6 AA Batteries, SPDT switch, Velcro strip. Most of these items can be obtained at your local department store or nearest Radio Shack.

Figure 3

You will also need some additional hardware shown in Figure 4. The hardware is used to attach a platform on which we will install our Pocket PC mount. You will need 22 #4 machine screws, 20 #4 lock washers, and 6 hex nuts. You will also need some 1-1/2" standoffs but because they are hard to find, I used 11, #4 M-F .5" and 11 #4 F-F 1" long to create what I needed.

Figure 4

The platform on which we will attach our Pocket PC mount will need to be at least 7" in diameter. You can also create a bit more of a bug shape as shown in Figure 5.
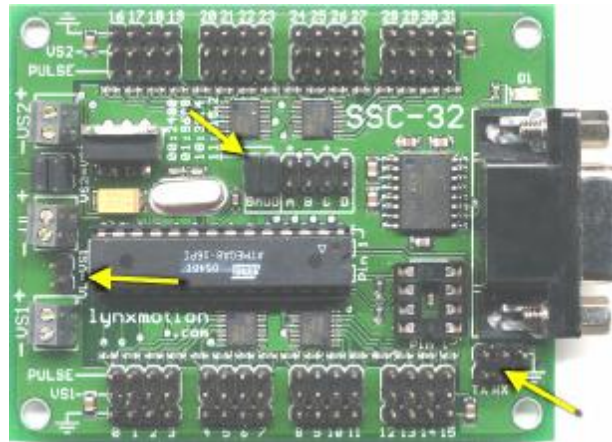

Figure 5

It is important to note that you can use any material you wish for this platform. However, if you are going to use a Pocket PC mount with a suction cup you will want as smooth as surface as possible. I used 1/8" clear Lexan. I then painted the underside black.

SSC-32 Prep

Before we start assembling the base the SSC-32 needs to be prepped. First set the jumpers as shown in Figure 6. You will need to remove the VL=VS1 jumper and the TX jumper. Place both the baud rate jumpers to set the speed to 115200.

Figure 6

The DiosWalker has two power sources. An RC Stick battery is used to power the servos and a set of 6 AA batteries are used to power the logic and various other components such as the wireless controller. You will want to be able to switch this power source on and off so a switch will have to be wired in series with the 8" 9v battery clip as shown in Figure 8. Use some heat shrink to insulate the connections on the switch. This also adds reinforcement as well.
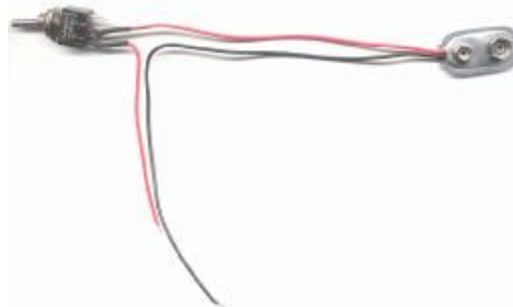
Figure 8

Once you create the logic power harness, connect to the VL header as shown in Figure 9.

Figure 9

At this point you can proceed with the EH3R base assembly.

Base Assembly

The EH3R base arrived after 4 days via UPS. Figures 10 and 11 show the EH3R Lexan and hardware. The 18 HS-645 Servos are packaged separately. You will need to download the assembly instructions from the Lynxmotion web site at:

Leg Assembly Instructions - http://www.lynxmotion.com/images/html/build38b.htm
Body Assembly Instructions - http://www.lynxmotion.com/images/html/build42b.htm

Figure 10　　　　　　　**Figure 11**

Before doing any assembly I recommend that you do the following.

- Read the information about Polycarbonate on the Lynxmotion website at: http://www.lynxmotion.com/images/html/infolexa.htm
- Remove and separate all pieces from the sheets. The individual pieces are laser cut from Lexan sheets. The pieces are still attached to the sheet and need to be separated. You could do this as you assemble but it will slow you down.
- Remove the protective plastic covering from the Lexan surfaces. If you wait till after assembly it is much harder to remove.
- Punch out all the small holes. There will be small Lexan pieces in each of these holes left over from the cutting process. Again, it is much easier to do this all at once.

The first thing you will need to do in the assembly process is to assemble the legs. Refer to the leg assembly instructions. You will need to assemble 3 right legs and 3 left legs as shown in Figure 12. The left legs are a mirror image of the right but I got a bit confused as I was building them.
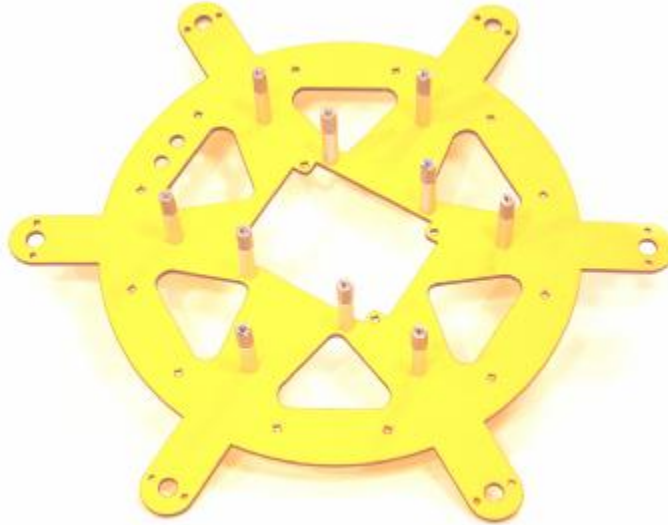
Figure 12

Figure 13 shows a partial assembly for the left legs. This should help.

Figure 13

With the legs built we can move on to the body. Again, you will need to refer to the body instructions mentioned previously. Before you start with the body, connect the 1/2" standoff to the 1" standoff to create a 1.5" standoff and attach it to the top side off the top panel as shown in Figure 14.

Figure 14

Six of them are attached to the middle ring of holes. The remaining 4 connect to the four holes near the center square, not the corner holes, which are used to mount the servo controller. Notice that the top panel has a slight cup. These standoffs connect to the convex side of the cup. Later we will attach our platform to these standoffs.

When you get to step 3 of the body instructions, **Do Not** install the VL=VS1 jumper. Just connect the RC battery harness as shown in Figure 15.
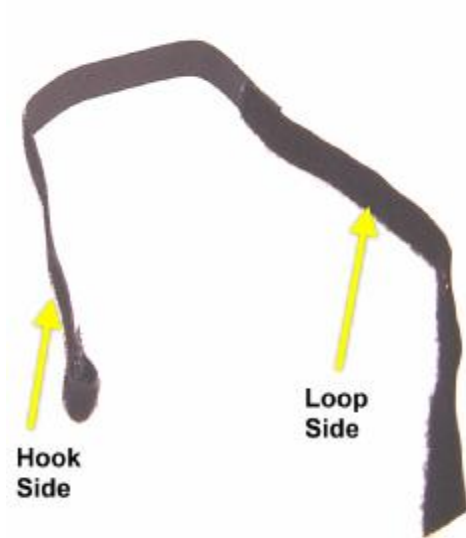

Figure 15

In Step 4, install the Logic Power switch into the first hole as shown in Figure 16.

Figure 16

Once you have completed the construction of the body there needs to be a few finishing touches made. First you will need to create a battery compartment for the stick battery. This is done by cutting a 9" section of hook and loop (Velcro). You will need to attach the hook portion to the loop portion with some hot glue as shown in Figure 17.

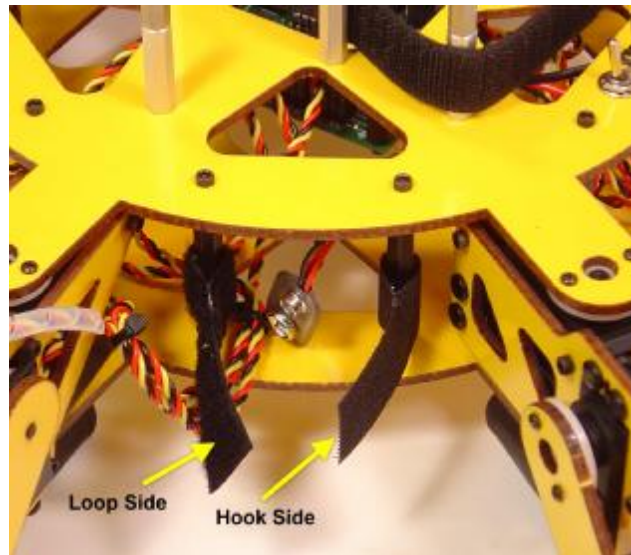**Figure 17**   **Figure 18**

Once the two pieces are connected, add a small loop to the end of the Hook side. Note that the Loop section is the piece that feels like soft carpet.

The key here is to attach the loop that you made around the first standoff then weave the Velcro around the other posts as shown in Figure 18. This creates a nice padded battery compartment. You should dry fit the pieces of hook and loop before you actually hot

glue them in order to get the placement correct. You may find it easier to add the small loop first.

Next you need to create a small compartment to hold the 6-cell logic battery. This is done by connecting a piece of Velcro to the posts to the left of the switches as shown in Figure 19. Wrap a hook piece around one post and secure it with hot glue. Then wrap a loop piece around the other post and secure. Dry fit first so that you can get the orientation of the hook and loop correct. Next, route the 9v clip into this space as well. Now all you need to do is connect the battery clip to the 6-cell pack and slip it into the small space. Connect the two pieces of Velcro to hold the pack in place.

Figure 19

The last step is to connect the platform to the completed body. As a minimum, the platform needs to be 7" in diameter. You can make the shape whatever you wish. I created a segmented shape for mine as shown in Figure 20. If you are using clear Lexan the attachment is very simple. Place the platform on top of the standoffs and mark the 6 outermost posts.   Don't create holes for the inner 4 standoffs as they may get in the way of your Pocket PC mount. Once marked, drill 1/8" holes and attach with #4 machine screws. **Do Not** paint the platform yet as we will be tooling it a bit more in parts 2 and 3 of this article.
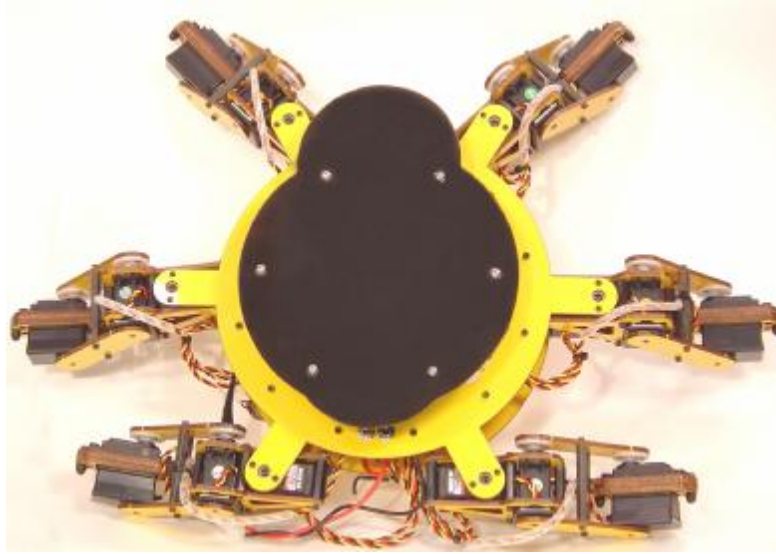
Figure 20

While we don't have our Pocket PC or PS2 controller connected there are a few things we can do to test our DiosWalker. I have included a two test programs that will allow you to connect your Desktop or Laptop to the walker with a serial cable. The first program is called **SSC32Test**. This program, shown in Figure 21 lets you set any servo connected to the SSC32 to any position.   The second program is called **DiosWalkerPC**. This program shown in Figure 22 is a desktop program that will allow you to put your walker into motion. The speed of the walker has been fixed at a slow pace as the program is meant for testing your walker only.
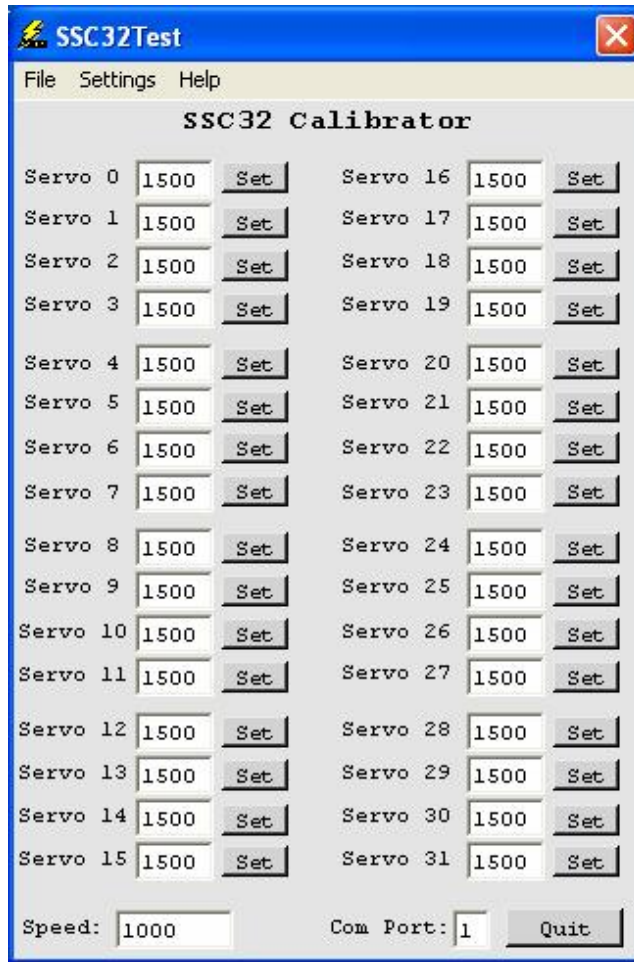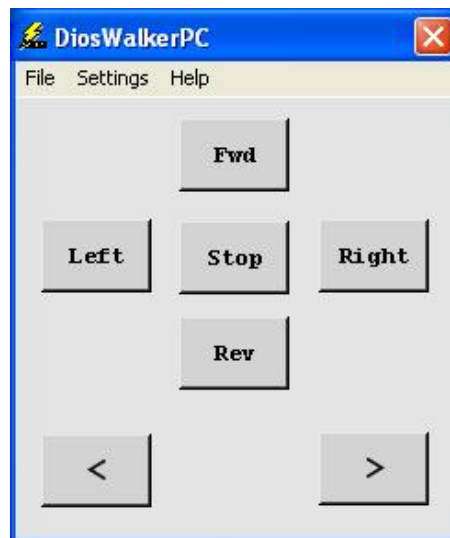
Figure 21

Figure 22

Part 2
Adding the Wireless Controller

One of the problems you have with a walker of this nature is that you need a great deal of control points to really push the robot to its limits. I originally attempted to use an RC radio to control the DiosWalker, but this approach presented multiple problems:

1. Ground based radios with more than 2 channels are very expensive.
2. They are slow to interface as you must poll each channel individually.
3. The transmitters are not very compact.
4. The transmitter's battery requirements can be greater than the device you are trying to control.

While I was researching other options I came across the PS2 controller. I have never owned a PS2 so it never occurred to me that this would be a viable option.

The PS2 controller has 2 full position analog joy sticks, and 14 additional buttons all within reach of your fingers while you are operating the joysticks.

You can pick up a wired controller for as little as $5.95 and a wireless for $9.95. Figure 23 shows a very popular wireless model called the Predator by Pelican.

Figure 23

The Predator runs on 2 AA batteries which will last several hours. The wireless controllers operate on the 2.4Ghz band. You lock a transmitter to the receiver so that you can operate multiple transmitters at the same time. I have a WiFi wireless network and Bluetooth and never had any problems using the Predator. While these wireless controllers are rated at 30 feet, I have used mine at ranges well over 100 feet outdoors.

Note that not all controllers will work with this interface. I have found that the **INTEC Mini does not work**.

The receiver module shown in Figure 24 is small and operates on 5v so it is well suited for use with microcontrollers.
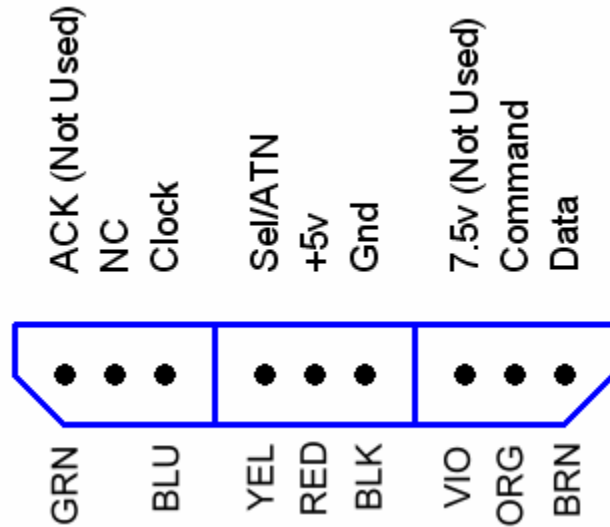

Figure 24

Figure 25 shows a pin diagram of a PS2 controller. The only connections needed are the Gnd, +5 for power, and the Data, Command, ATN, and Clock for communications.
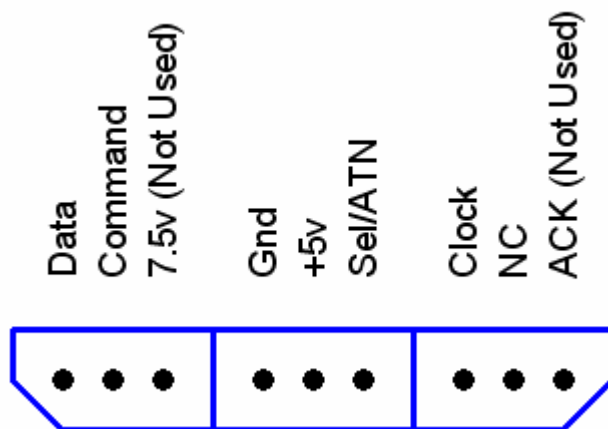
Figure 25

You may connect directly to these pins if you like. I prefer using a PS2 extension cable that has been cut in half. The extension cable will cost you $5-$10 and will give you a cool way of connecting various controllers to your application. The pins on the extension cable are reversed for obvious reasons as shown in Figure 26.

ACK (Not Used)
NC
Clock

Sel/ATN
+5v
Gnd

7.5v (Not Used)
Command
Data

GRN
BLU
YEL
RED
BLK
VIO
ORG
BRN

Figure 26

The PS2 interface uses a SPI interface. There are 4 signal pins as well as Gnd and power needed for this interface. The SEL and CMD signals are used to place the controller into the correct mode. The Data signal lead is used to send or receive data depending upon the command. The Clock signal is used to clock the bits in or out.

To interface to a PS2 controller you need to use a microcontroller. For the DiosWalker, we will be using a DiosPro microcontroller. I am not going to go into the protocol details as they are outside the scope of this article. I created two DiosPro functions called PS2Init and PS2Read. These functions will take care of handling the various timing requirements of different controller types as well as the protocol needed for the actual hardware interface.

Update August 17 2006

I have added the colors of the extension cable leads. Please note that these colors only apply if you use a **Datel Extension Cable #PS22710D**. These are very popular and can be purchased from Best Buy for under $10. If you are using a different cable you will need to use a meter to check the colors against the pins.

Part 3
Dios Construction

Before we get started with the DiosPro Controller assembly, let's take a quick look at Schematic 1. A couple of LED's were added for status. The remaining connections are connected to the SSC32 board's RS232 interface and the SSC32's regulated side of the logic power.
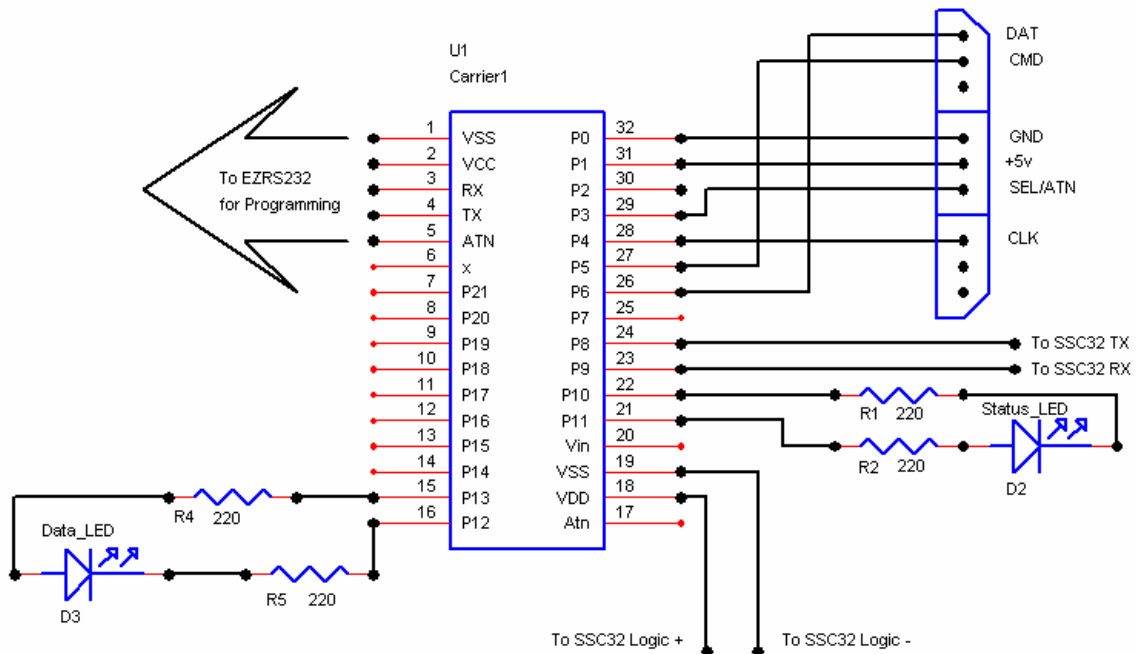
Schematic 1

Carrier Assembly

Let's start by assembling the Dios Carrier 1 board. The kit comes with an assembly manual which we will follow until you get to Step 7. In Step 7 install the headers on the top of the board as shown in Figure 27.

Figure 27

Once you have the carrier built, install the DiosPro chip and attach a small piece of foam tape to the bottom of the board. Attach the board to the EH3-R base as shown in Figure 28.

Figure 28

Center the carrier against the standoff. For orientation, notice the switches in the upper left side of Figure 28. Notice also that the notch in the chip is facing down.

Power the Carrier

In order to supply power to the Dios we need to build the small 2-conductor connector shown in Figure 29. Start by cutting two 5" wires. Since this is going to be a power connector, you need to make each conductor a different color. In this case I used red and

black. Next, cut two 2-pin sections from the 36-pin female header. Cut 4 pieces of 1/16"
heat shrink 1/2" long.

Figure 29

Start assembling the connector by soldering a wire to each of the pins on one of the 2-pin
headers. Slip a heat shrink section onto the wire and slide it over the solder joint and
heat. Now slip one of the two remaining heat shrink sections over each wire. Then
solder the wires to the remaining 2-pin header. Slide the heat shrink down over the joints
and apply heat.

Connect one end of the connector to the +- header pins on the SSC32 shown in Figure 30.
This header is located next to the Baud Jumpers. Connect the other end of the connector
to the +- header pins on the Dios Carrier 1 as shown in Figure 31.
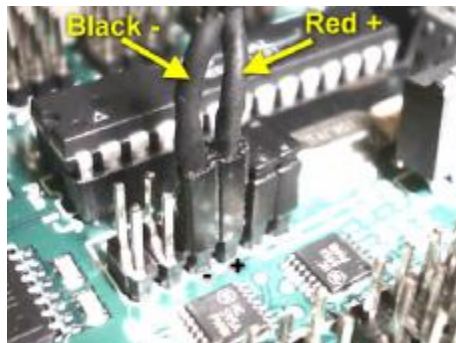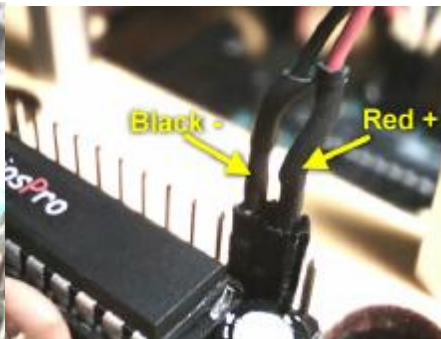
Figure 30   **Figure 31**

Status Indictors

In order to create the status indicators you will need the components shown in Figure 32.
- Red LED
- Green LED
- 4, 220 Ohm Resistors
- 4, Female Plugs
- 2, 3" Pieces of 1/8" heat shrink

Figure 32

Take the red LED and clip the leads so that they are 3/8" long. Clip resister leads to about 1/4" as shown in Figure 33. Solder one end of the resistors into the open end of a female plug as shown. Solder the other end to the led as shown in Figure 34.

Figure 33   **Figure 34**

Cut just enough of the 1/8" heat shrink to cover the complete lead and slip it over the complete plug/resistor assembly and heat as shown in Figure 35. Repeat the same procedures for the green LED.

Figure 35

Insert the green status LED onto the Port 10 and Port 11 header pins as shown in Figure 36. Make sure the flat side of the LED is facing left.   Insert the red data LED onto the Port 13 and Port 14 header pins as shown in Figure 37. Make sure the flat side of the LED is facing right.
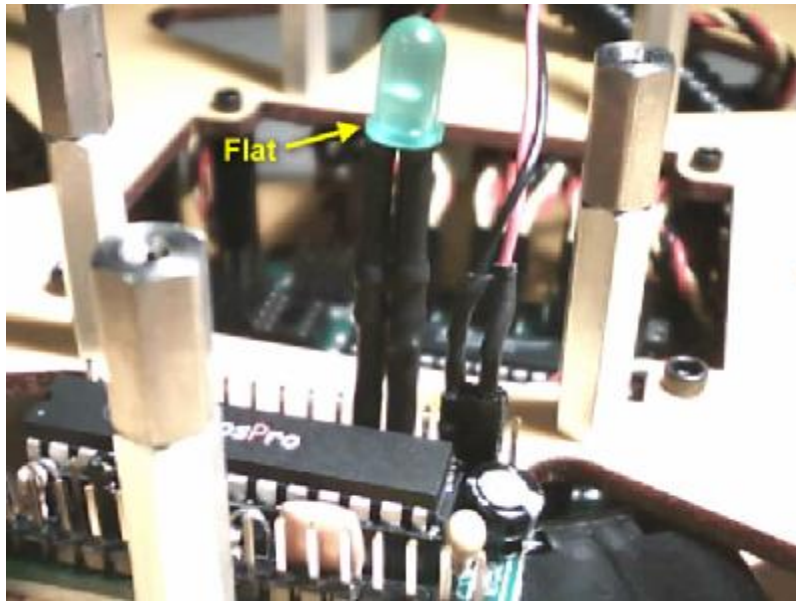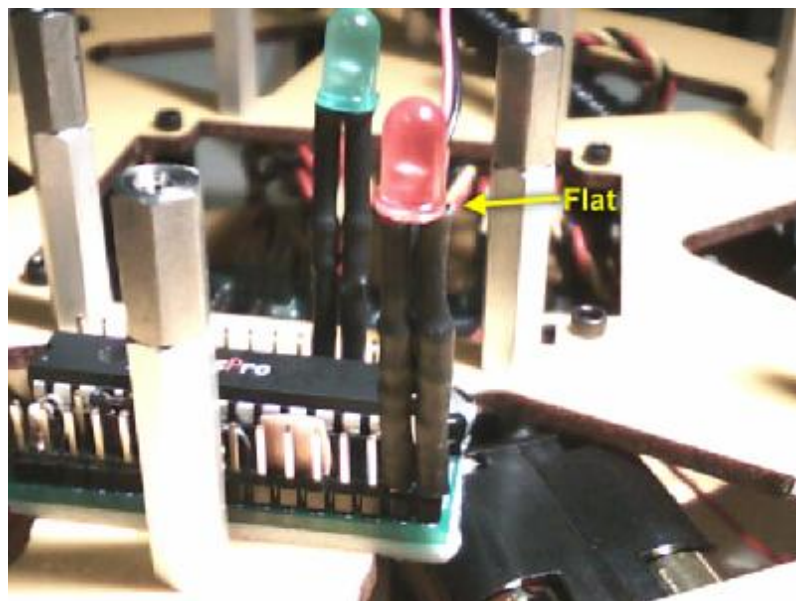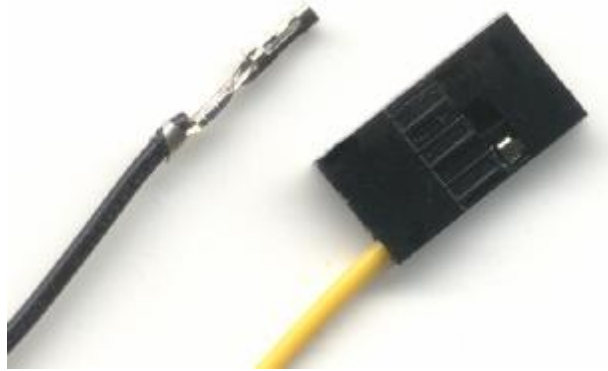
Figure 36

Figure 37

LED Test

The Dios comes programmed with a test program that toggles all the IO ports. If everything is connected properly and we provide power to the SSC32, the status LED's should begin to flash.


UART Connection

The DiosPro chip has a built-in UART and we need to connect this to the SSC32. This is done with the small yellow and black connector that comes with the SSC32. On the 3-pin end you need to remove the black wire from the connector housing. This is done with a razor knife or a very small screwdriver by slipping it under the small flap and gently prying up. The wire should pull out as shown in Figure 38.

Figure 38

Slip the wire back into the connector housing center position as shown in Figure 39.
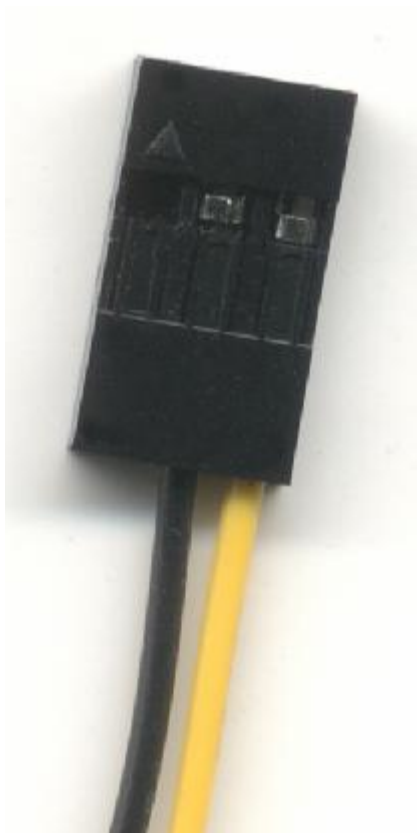
Figure 39

Now take the modified end of the connector and plug it into the SSC32 as shown in Figure 40. The connector plugs into the 3-pin header closest to the DB9 connector. Notice that the yellow wire is on the inside and the black wire is connected to the center pin.

Figure 40

Rout the wires through the base as shown in Figure 41.
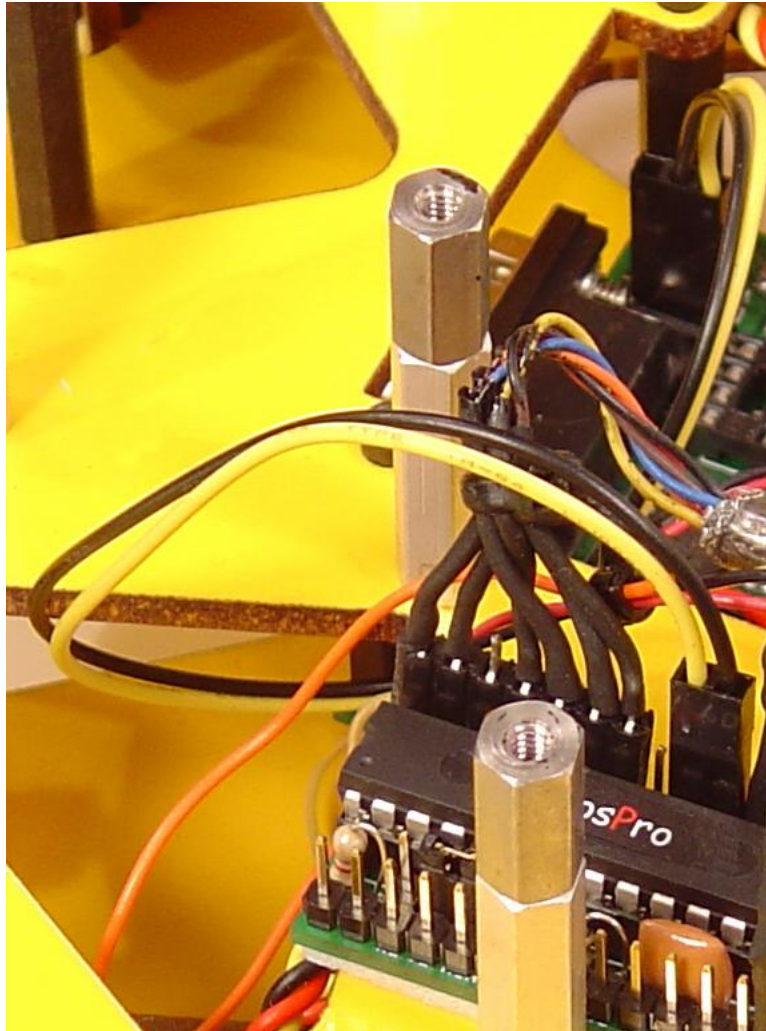
Figure 41

Plug the 2-pin end into the header next to the green LED as shown in Figure 42. The black lead should be the one closest to the green LED.

Figure 42

Part 4
PS2 Connection

We have come a long way, but it is time for our final connection. For this you will need a PS2 controller extension cable, some heat shrink, and a 7-pin female header.

- Cut the end of the extension cable that mates to your PS2 controller. You want about 12" of cable. Don't confuse this with the end that plugs into the PS2.
- Strip about 2" of the plastic insulator back revealing the small colored wires.
- Trim about 1/8" off each of these wires.
- Take a multi-meter and using Figure 5 determine which wire is mated to each pin.
- Using Figure 43 connect the various wires to the header. Slip a 1" piece of 1/6" heat shrink over each wire before you solder it in place. Once a wire is soldered, pull the heat shrink over the connection and heat.
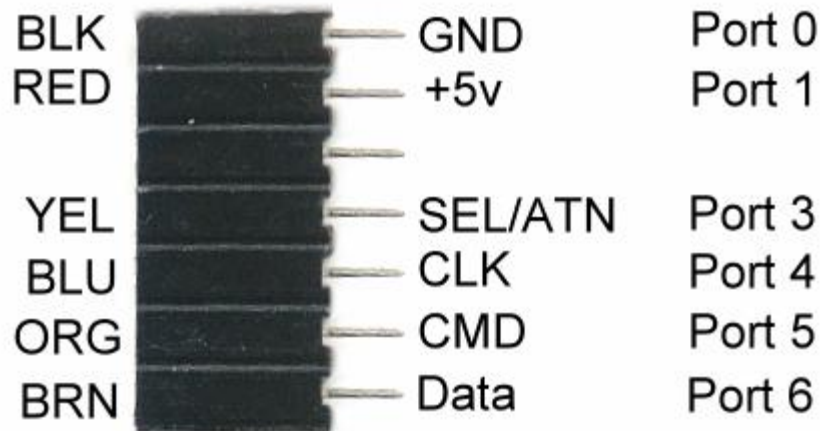

| BLK | GND | Port 0 |
RED	+5v	Port 1
YEL	SEL/ATN	Port 3
BLU	CLK	Port 4
ORG	CMD	Port 5
BRN	Data	Port 6

Figure 43

When all the wires have been connected you should end up with the cable assembly shown in Figure 44. I secured the ends with a small tie wrap. Now you can connect the female header to the header pins on the Dios carrier as shown in Figure 29. Notice that the first two pins are the Gnd and +5v. These mate up with Ports 0 and 1 on the carrier. Port 0 is indicated with the yellow arrow in Figure 45.

Update August 17, 2006

Again I have added the colors for the **Datel Extension Cable #PS22710D.** I have also added the Port numbers that are used to interface to the controller.

Figure 44
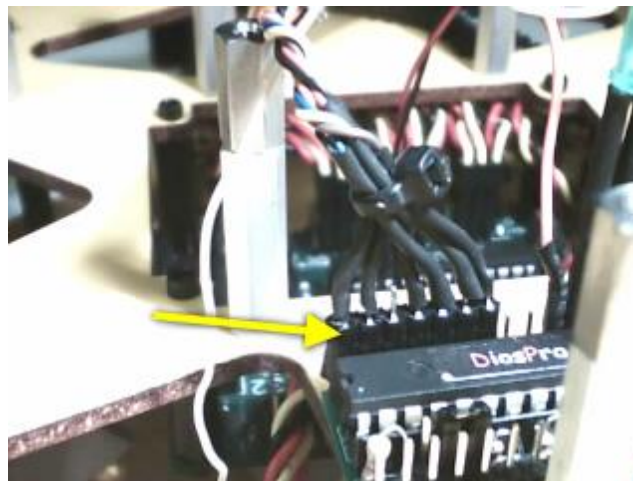
Figure 45

It's time to run a little test but first we need to setup the Dios carrier so that we can program the DiosPro chip. In order to do this we need to create a small adapter by cutting two 5-pin sections from the 36-pin female header and solder them together as shown in Figure 46.

Figure 46

With the small adapter we can plug an EZRS232 board into the Dios carrier as shown in Figure 47. This will allow us to upload the firmware to the chip. You can download the free Dios compiler software from the KronosRobotics web.

Figure 47

At this point you can program the DiosPro chip with the DiosWalker.txt firmware. Make sure the IK.lib file is located in the same directory as the DiosWalker.txt file.

PS2 Test

Connect the EZRS232 Driver as shown in Figure 47. Connect your 9-Pin cable to the PC and program the DiosPro with the program called **DWPS2Test.txt**. Once loaded, the DiosPro will display 6 numbers. The bit fields of the first two numbers represent the different buttons on the PS2 controller. The remaining 4 numbers are the analog positions of the two joysticks.

Part 5
RC Radio Option
Added Aug 19, 2006

Parts 2-4 were centered on a PS2 controller interface.   While this is a very inexpensive and quite capable interface there are times when you may want to use a traditional radio.


Figure 5-1

Some things to keep in mind when using an RC radio. There are ground and air based radios and it is illegal to use an air radio with a ground based system and visa versa. The radio pictured in Figure 5-1 is a 72Mhz Helicopter radio so by rights it should not be used with the DiosWalker. However I have modified my walker so that it can fly so I am now legal ☺

RC radios will give you a longer range. This helicopter radio also gives you quite a few more abilities. The one I like is that the channel 3 is free and not spring loaded so I can adjust the height of my walker on the fly and just leave the control in place for that height.

There are also a few more knobs that could be used to adjust the speed and number of steps.

The actual interface to the RC radio is covered by a 2nd DiosPro microcontroller so this makes hookup very easy.   If you take a look at Schematic 2 you will see that we have removed the status LED's as well as the connection to the PS2 controller and added a serial link to the 2nd Dios Carrier from pins 10 and 11.
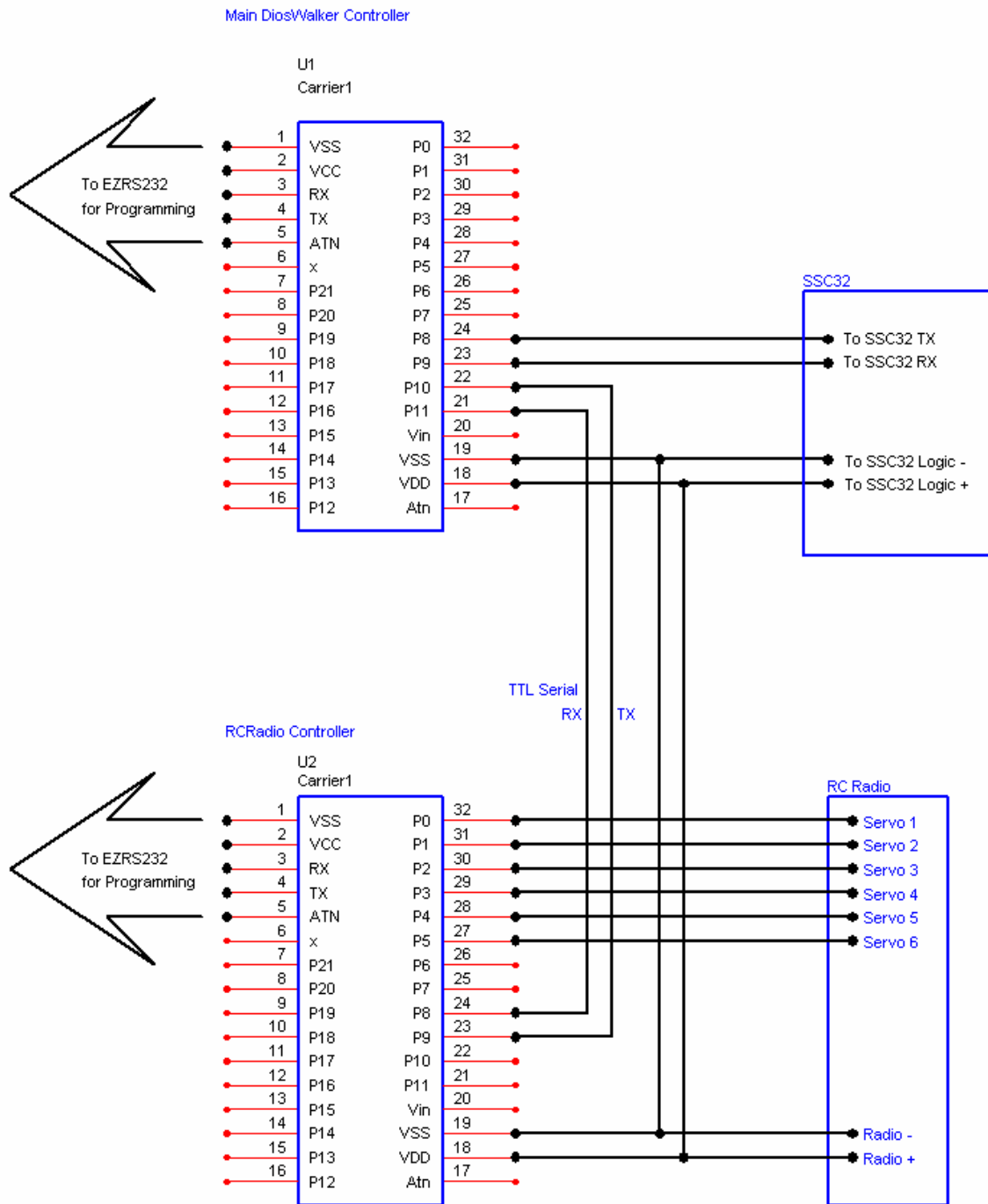

Schematic 2

I have included the code for the RC radio interface in the source files. The actual interface will be discussed in a future article in my "Control Your World" series. You will need to use the modified DiosWalkerRC.txt program as the main file. It will include the needed IK library.

I deviated from the schematic and used a DiosWorkboard deluxe shown in Figures 5-2 and 5-3. The breadboard area and built-in programming interface will make it much easier to experiment with various sensors when I add some automation to the DiosWalker later.
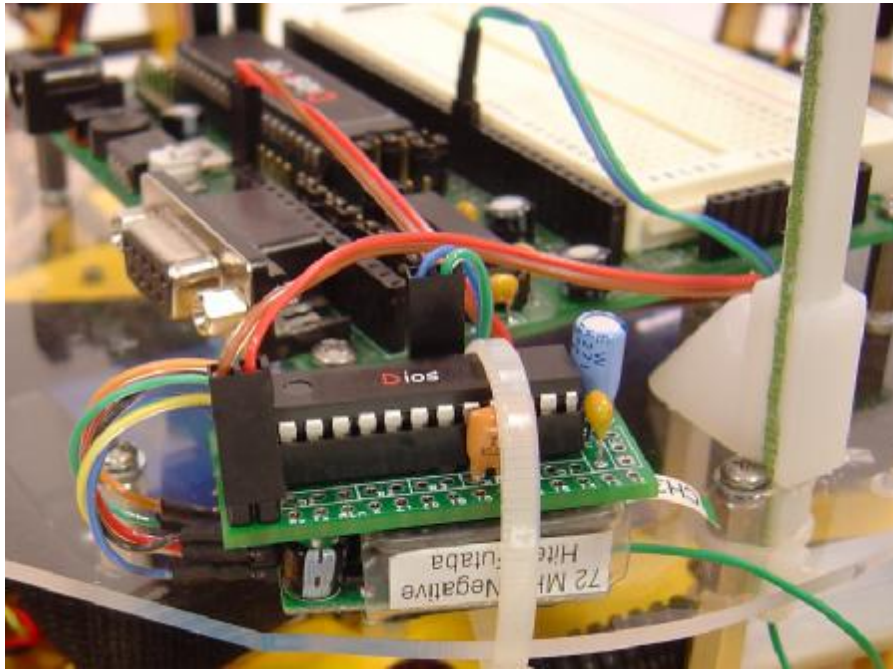
Figure 5-2

Figure 5-2 shows the micro receiver I used. It is piggybacked to a DiosCarrier 1 board and together they weigh less than 20g. Only two 4 wires are needed to connect this assembly to the main DiosPro controller. These are the 5v Power and the Tx and RX leads.
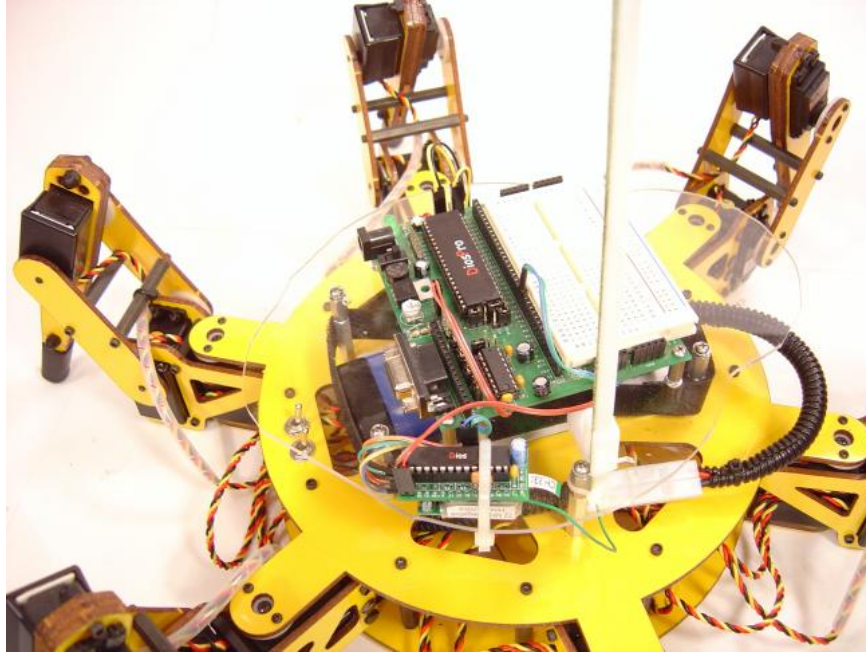
Figure 5-3

I attached a monster antenna tube to raise the radio antenna.   The batteries run out before the range does.

You could also use a 2 channel RC car radio with some creative programming. A 2 channel radio would be a perfect companion for an autonomous walker. You would use the radio only to put the walker back on track to override some function.

Part 6
Finishing up the Platform

We are going to add a upper platform to the DiosWalker. At first the only purpose of this platform is to hide the battery pack and to provide a place to mount the PS2 receiver.

If you have not done so already you need to cut out your upper platform. This platform needs to be at least 7" in diameter but can assume any shape you wish to add a different effect to the DiosWalker. I am going to assume you are using clear Lexan for your platform but just about any smooth material will work. Most home centers sell 1/8" Lexan sheets and an 8" x 10" sheet will cost you under $5.

The Lexan sheet will come with a protective plastic covering. Leave this in place while you cut out the platform. The covering makes a nice surface on which to draw your pattern and provides lubrication while you cut the sheet. If you have some scrap Lexan or plastic material without this covering you will need to add some masking tape. As you cut the plastic, the covering shreds and provides a lubricant that keeps your scroll saw or band saw from melting the plastic. This will yield you a much better edge.

Once the platform shape has been cut out you will need to mark out and drill the holes as outlined back in Part 1 of this series. We also need to create the holes for the two LEDs in our PS2 controller interface. This is done by inserting 6 #4 machine screws into the standoffs as shown in Figure 48.
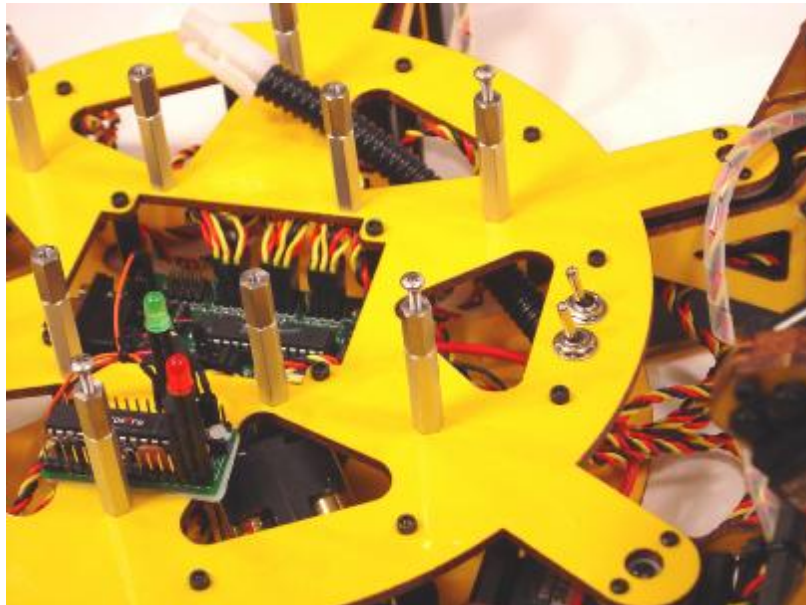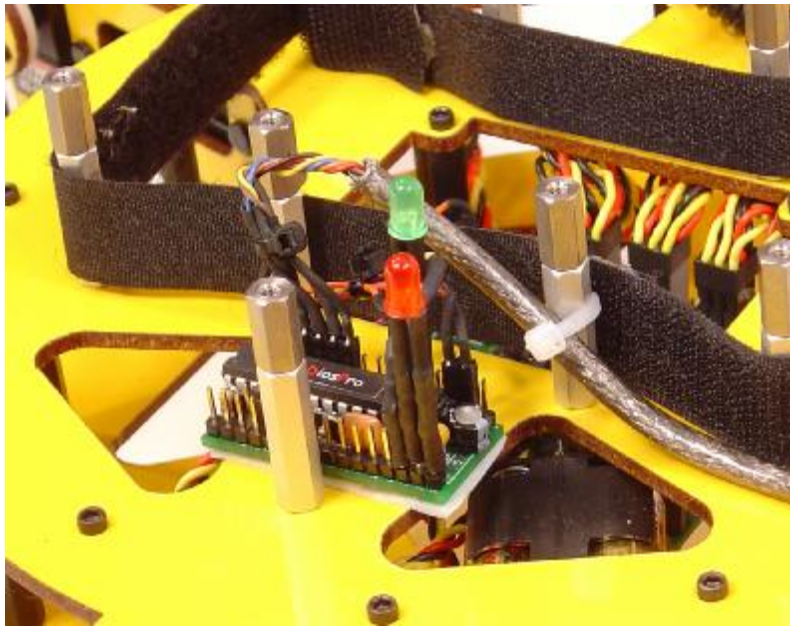

Figure 48
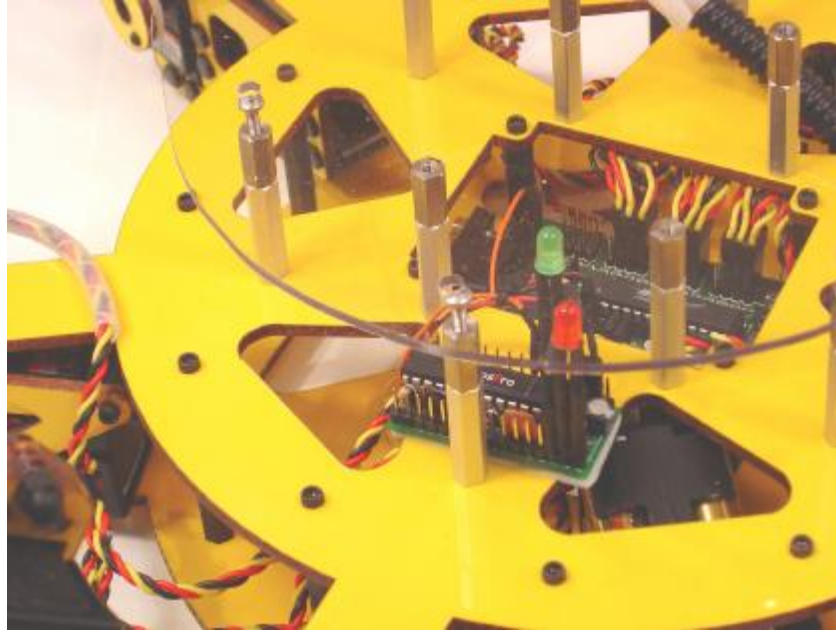
Figure 49

You then place the platform on top of these screws matching up the holes as shown in
Figure 49. At his point you can place a small mark on the Lexan over each LED. For
standard size LEDs a 7/32" drill bit is perfect but a 1/4" will do if you don't have that
size. After this is done you can paint the underside of the platform. Use a paint that was
designed for Lexan. You can get his from your local hobby shop as it is what they use to
paint RC car bodies.

Earlier you added a connector to a PS2 extension cable. It is time to once again connect
that connector to the DiosPro. We need to take a small tie wrap and attach the cable to
the standoff shown in Figure 50.

Figure 50

This provides some strain relief to the connector that we made back in Part 2. Place a small 1" piece of double stick foam tape on the bottom of the PS2 connector and attach it to the platform as shown in Figure 51.
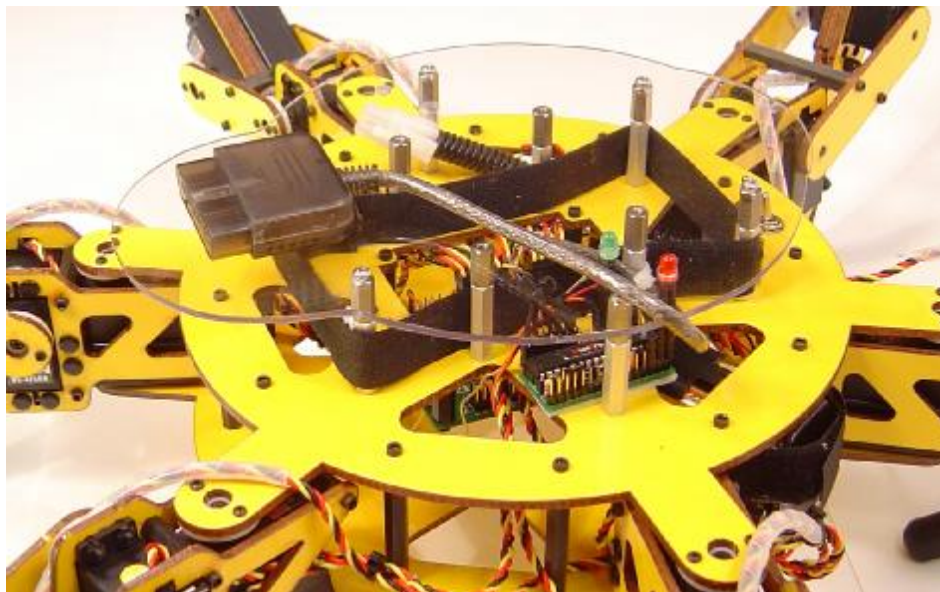
Figure 51

Part 7
Final Test

It's time to load the main program into the DiosPro. There are two files. The first is called **DiosWalker.txt**. This is the main DiosWalker firmware program. The second file is called **IK.lib**. This file gets loaded by the DiosWalker.txt file. Both files need to be located in the same directory.

Added August 29, 2006
Note that if you are using the RC radio the file is called DiosWalkerRC.txt

Attach the EZ232 driver to the carrier as you did earlier and turn on the logic power. Load the Dios Compiler and program the DiosWalker.txt into the DiosPro. Once complete turn off all power and remove the EZ232 adapter.

Power Up

I recommend elevating the DiosWalker the first time you test the program. You can do this by placing a 6" block of wood or some other object under the Walker base.

If you have not connected both the batteries do it now. Turn on both switches. If all goes well you will see the legs move up then slowly back down.

Ever so slightly move the right Joystick forward. You should see the legs start to animate. The further you push the joystick the more the legs will animate. Once you release the joy stick the animation should stop.

If the DiosWalker seems to be moving correctly it is time for a real walk test. Place the DiosWalker on the floor. Make sure you have lots of room and have fun. Here are a list of the button commands. Notice that there are buttons that have not been assigned.

Added August 29, 2006

PS2 Controller	RC Radio
Right Joy Stick	**Right Joy Stick**
Forward = Walk Forward Back = Walk Backward Left = Walk Sideways to the Left Right = Walk Sideways to the Right	Forward = Walk Forward Back = Walk Backward Left = Walk Sideways to the Left Right = Walk Sideways to the Right
Left Joy Stick	**Left Joy Stick**

Forward and Backward Not used Left = Rotate Counter Clockwise Right = Rotate Clockwise **Joy Pad** Forward = Fast Walk Mode Back = Slow Walk Mode Left or Right Normal Walk Mode **Other Buttons** Triangle = Normal Mode Square = Creep Mode Circle = Drunk Mode Cross = Tip Toe Mode	Forward = Adjust the Height Up Back= Adjust the Height Down Left = Rotate Counter Clockwise Right = Rotate Clockwise Use additional channels to control Speed and steps that are passed to the IK routine.

Going Further

The next step would be to add a sonar sensor and try for total automation of the DiosWalker. The current program only uses less than 20% of the available program space. So there is lots of room to add your own functions. We have made it a point to keep the variable usage down to a minimum and since most automation routines would be called outside the IK routines almost all of the local memory space is available for your program variables and there is still a considerable amount of global space left.

Something that I have considered is to use a Dios Workboard with two DiosPro chips. Once chip using the existing routines to control the walking and the other to take care of handling all the sensors and automation. You could literally plug the existing carrier into the prototype area of the workboard.

You may have noticed that we connected both the Transmit and Receive lines to the SSC32 board. While the IK routines only use the Transmit side to set the servos. There are a couple IO commands you can issue to the SSC32 board. There are unused servo positions that can be used as IO ports as well as a few AtoD ports that can be accessed. Refer to the SSC32 manual for a description of those commands. You wont be doing any high speed IO on these ports but they will work nicely as indicators or bump sensors.

Before moving on lets me say a few things about the scratchpad routines used. In order to conserve some of the global memory space I used some of the new commands available with version 43 of the DiosPro firmware. These commands allow you to copy local and global variables to and from the upper memory banks on the Dios and DiosPro chips. The Dios only has Bank5 available which is normally used for string variables but the Dios Pro has Banks 5-14 available. Banks 6-14 are yours to do with as you see fit. Each banks has 256 bytes of data available so you can place stored information there as well using the memread and memwrite commands.

Parts

LynxMotion www.lynxmotion.com
- Round base with 18 Servos #EH3R-KT
- HS-645 Upgrade #SUP-04
- Servo Controller #SSC-32
- Wiring Harness #WH-01

Jameco   www.jameco.com
- 22, #4 3/8" Machine Screw #40969CK

- 20, #4 Lock Washers #4106850CK
- 6, #4 Hex Nuts #40942CK
- 11, #4 M-F .5" Standoff #111755CK
- 11, #4 F-F 1" Standoff #139184CK
- SPDT Switch   #22832CK

Kronos Robotics www.kronosrobotics.com
- Dios Starter Kit 2   #16448
- 9-Pin Serial Cable   #16259
- 6-Cell Battery Holder   #16321
- 9v Battery Clip 8"   #16264
- 5, Heat shrink strips   #16287
- 36-Pin Female Header #16291
- 2, 3" Pieces of 1/16" heat shrink #16287
- 2, 3" Pieces of 1/8" heat shrink #16288
- Red LED   #16234
- Green LED   #16235
- 4, 220 Ohm Resistors   #16188

Dios Compiler (Free Download)
http://www.kronosrobotics.com/downloads/DiosSetup.exe

Other
- 7.2v 2000MaH – 3000 MaH Stick Battery.   Check Local Hobby shop, Radio Shack or LynxMotion.
- 6, 1.2v - 1.5v AA Batteries Rechargeable or Alkaline. Check department store.
- Velcro Strip – 3/4" x 1 Yard. Should be the non sticky type. Check fabric section of department store.
- 8"x10" Plexiglas or compressed PVC. Check local hardware or home center.
- Universal PDA Mount. Check Auto Store or Amazon.com
- Hookup Wire 22-26Ga. Different colors will be helpful.
- Double stick foam tape. Two 1" pieces should do it. Any department store or home center will carry this. A popular brand name is 3M.
- Pelican Predator Wireless PS2 Controller.
- Hookup Wire 22-26Ga. Different colors will be helpful. A spool of 24Ga telephone cable is perfect.

!! Important !!   Don't use a battery that supplies more than 7.2v or you will burn up the servos.   Even at 7.2v we are pushing the servos.

KRMicros www.krmicros.com
- ZeusPro   http://www.krmicros.com/Development/ZeusPro/ZeusPro.htm