



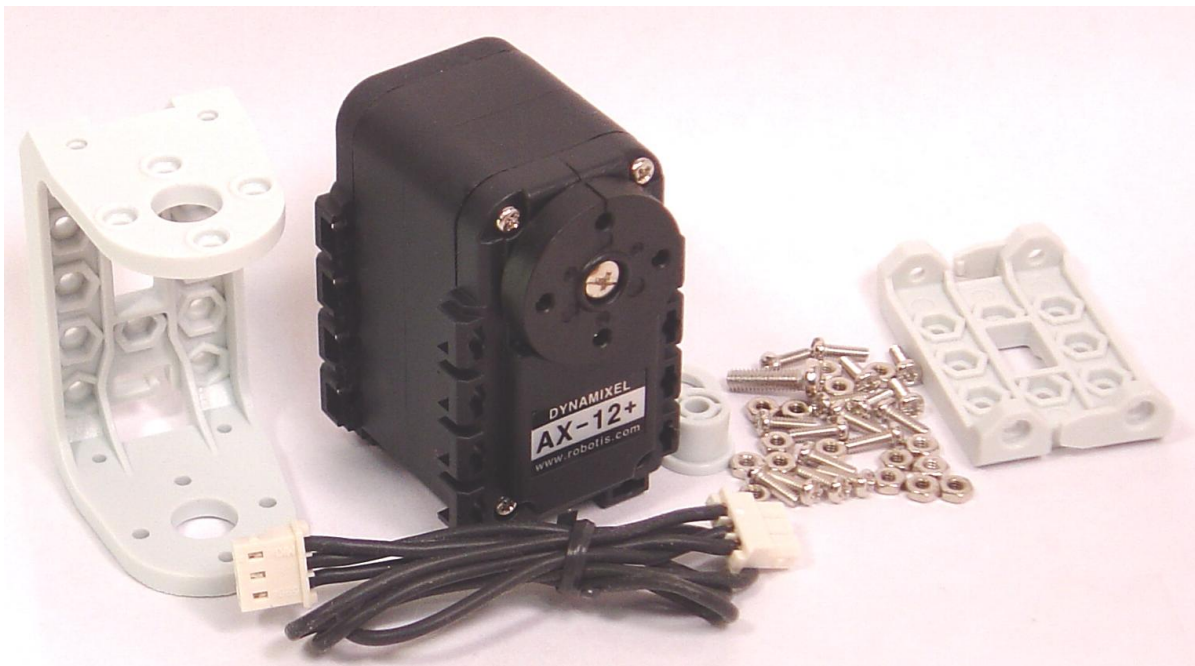
---

Understanding the AX-12  
as seen in  
March 2007 of Servo Magazine

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

While researching my next RobotFest project I came across the AX-12. It takes something special to get me fired up but this little gem does the trick. There are two ways to get you hands on an AX-12. You can purchase them individually or as part of the Bioloid kit. Both can be purchased from CrustCrawler at [www.crustcrawler.com](http://www.crustcrawler.com).

When purchased separately the AX-12 Servo Kit comes with 2 frame sets and a good deal of hardware as well as a nice 8" cable as shown in below.



Let's take a closer look at the AX-12 and how to use it.

## AX-12 Features

- 1,000,000 bps communications Speed
- Full feedback on Position, Speed, Load, Voltage, Temperature
- Can be set to full rotation mode (gear motor mode)
- Full 300 degree movement in 1024 increments
- Full control over speed in 1024 increments
- Full control over max torque in 1024 increments
- Built-in LED that can be used as a status indicator
- Automatic shutdown based on voltage, load or temperature
- Single cable network connections
- You can control 100's of AX-12 with only 2 data ports
- Synchronized servo movements
- Servo movement range can be set by user

The feedback provided by the AX-12 enables you to constantly monitor the device. You can look at its current position as well as the load being placed on the device. You can monitor current voltage and temperature as well. Another cool feature is that you can turn off the motor on the AX-12 and monitor the current position of the device. This makes it possible to pose more complex robots which will aid in their programming. I will use this feature in future articles when I build more complex robots.

Before moving on I need to mention that the AX-12 is not a toy and I do not recommend its use in robots where small children are likely to come in contact with the arms and legs of your bot. The reason is that the AX-12 is extremely powerful and with a holding torque of 16Kg they can pinch or even break small fingers.

One of the more important features is the communication speed. I have dealt with servo controllers in the past that used 9600 bps or even 115200 bps. The problem is that when you start issuing many commands to many servos you start to get delays. This also takes processing time away from your microcontroller. At 1,000,000 bps you can send several commands to many servos and get near instant results.

## AX-12 Physical Characteristics

The AX-12 is roughly the size of a standard servo without the mounting tabs as shown in Figure 1. In lieu of the typical servo mounting tabs there are 20 small mounting tabs. These tabs are designed so that small #2M nuts can be inserted into each one. This allows you to attach the AX-12 using #2M machine screws.

The AX-12 also has a special bearing located opposite the main drive. This allows you to attach various frame sets like the one shown in Figure 3. This makes assembling robot joints very easy.

Mounted on the drive shaft is a small hub. This hub has 4 mounting holes with #2M nuts installed. This makes it easy to attach frame sets or other hardware to the hub. You can even attach wheels to the hub. In Figure 4 you can see how I attached the hub to a large 6" foam wheel. Why would you attach a wheel to a servo? The AX-12 is not a servo but a very powerful and efficient gear motor with a tiny microcontroller installed. This microcontroller let's you configure the AX-12 like a servo or like a gear motor in a fully rotational mode.

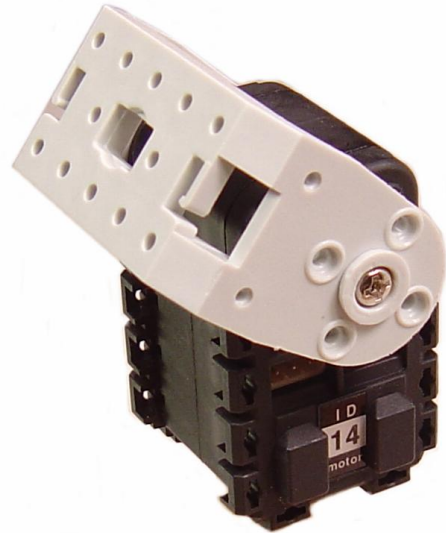


Figure 3



Figure 4

I found the easiest way to mount the AX-12 was to take the included OF-12S frame set and enlarge a few of the holes with a 3/32" drill bit as shown in Figure 5. This will allow you to easily self tap the piece using standard #6 machine screws.

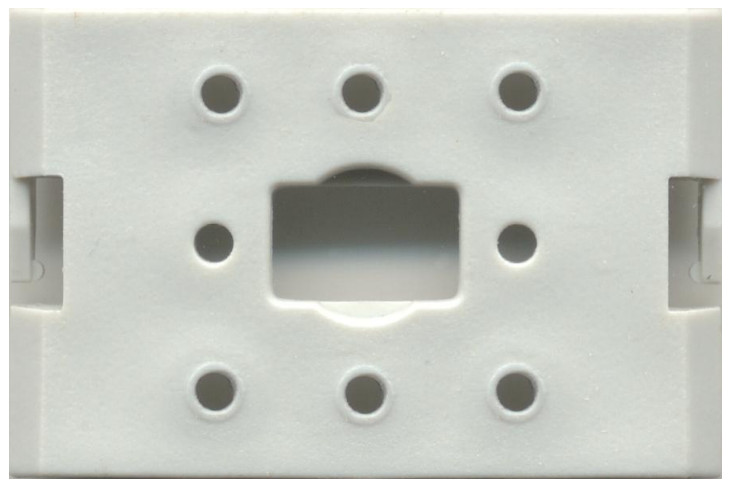


Figure 5

## AX-12 Electrical Characteristics

The AX-12 requires a 7V-10V power source to operate. At 7V the speed is .269 seconds for a 60 degree rotation. At 10V this improves to .195 seconds. You have full control of the speed and it has a maximum operating current of 900mA.

Unlike standard servos all AX-12's are connected on a 3-wire bus. This allows you to daisy chain the cables as shown in Figure 6.

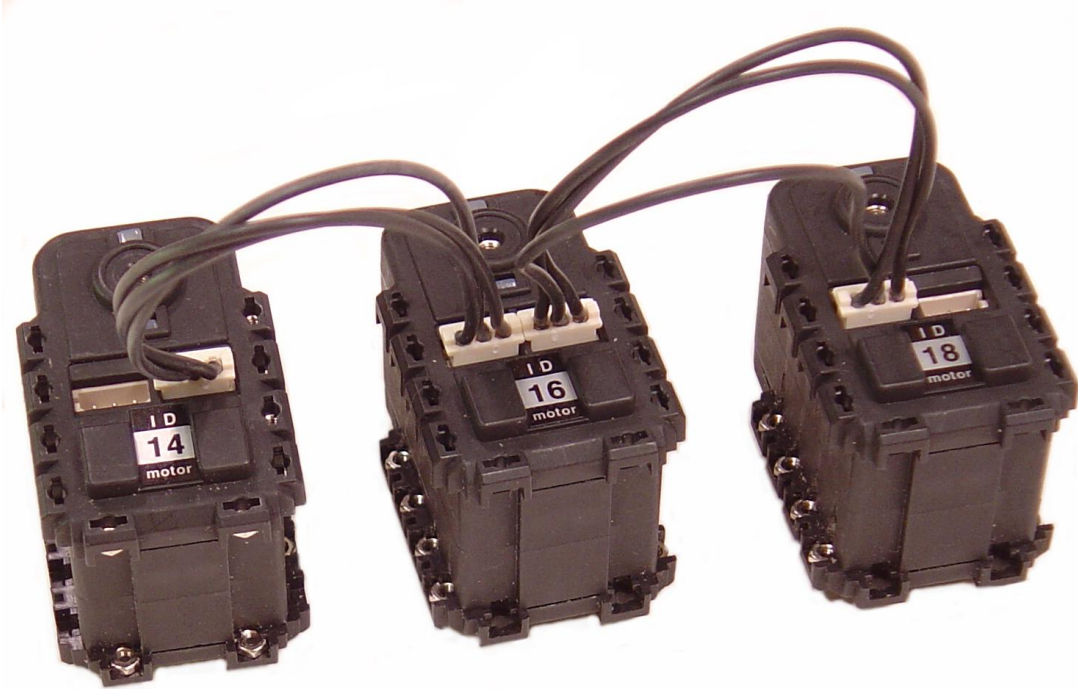


Figure 6

A small 3-wire cable is used to connect the servos to the controller. The pin spacing on the connector is .1" so you can use a standard servo header or make your own as shown in Figure 7. You will only need the header on the cable that is connected to the microcontroller.

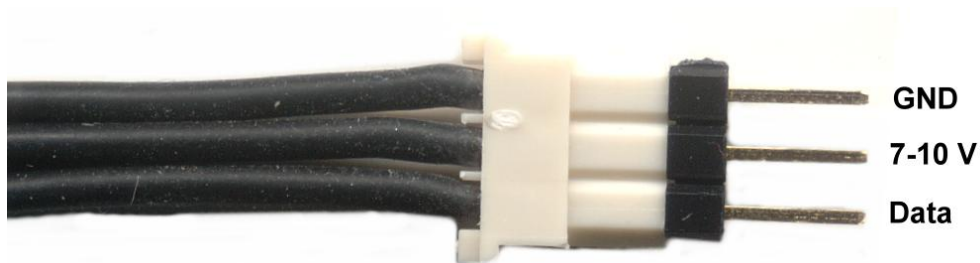


Figure 7

## AX-12 Communications Protocol

The AX-12 protocol is quite simple. In order to communicate you need to be able to communicate at 1,000,000 bps and since the protocol is half duplex you need to be able to tie your TX and RX leads together and have the ability to take your TX off line as needed. The speed alone will eliminate most microcontrollers. With the PC software that is included with the Bioloid kits you can change the baud rate but I don't recommend it. The slower the communications speed the more unresponsive the AX-12 becomes.

Let's look closer at the actual protocol. Don't worry if you don't understand all the bits and bytes as I have provided both Dios and Zeus libraries that take care of all the lower level communications.

ByteName	Description
1Start	Always 255
2Start	Always 355
3ID	The IF of AX-12 you wish to talk to
4Length	The number of parameters + 2
5Instruction	The action for the AX-12 to perform
6Data 1-N	Optional Data in addition to instruction
7Checksum	= sum of ID + Length + Instruction + Parm1 + ParmN ...

### Start

The first two bytes are always 255. They indicate the start of an incoming packet.

### ID

Only 0-254 may be used. A value of 254 indicates a broadcast to all devices on the network.

### Length

If a single instruction is used with no data then the length value will be 2. If one data parameter is used then the length value will be 3.

### Instruction

Valid instructions are as follows

1	Ping
2	Read Data
3	Write Data
4	Reg Write
5	Action
6	Reset
7	Sync Write

The Reg Write instruction is used in conjunction with the Action command to help synchronize certain commands. You send a Reg Write command to all the AX-12's, then issue an Action instruction. This tells the AX-12 to issue the stored command.

### Data1-N

This is the optional data parameter(s). Some instructions like Reset and Ping do not have data parameters. Others have 1 or more data parameters.

### Check Sum

All bytes except for the start bytes are added together. Only the first 8 bits are used. Once all bytes are added you must invert the bits.

Most of the instructions you will send to the AX-12 involve reading or writing one of the registers. For instance to set the Goal Position register you would write to register 30. All the registers, as well as packet examples, are listed in the AX-12 manual and can be downloaded from the CrustCrawler website at:  
<http://www.crustcrawler.com/products/bioloid/docs/AX-12.pdf>

When a packet is sent to the AX-12 it will respond with a packet of its own. This packet is much simpler and will contain the data you may have requested, as well as an error code.

ByteName	Description
1Start	Always 255
2Start	Always 355
3ID	The ID of AX-12 you wish to talk to
4Length	The number of parameters + 2
5Error	Reports the Error status of the last command
6Data 1-N	Optional Data in addition to instruction
7Checksum=	sum of ID + Length + Instruction + Parm1 + ParmN ...

### Start

The first two bytes are always 255. They indicate the start of an incoming packet.

### ID

Will contain the ID of the device indicated by the packet just received.

### Length

If a single instruction is used with no data then the length value will be 2.

### Error

A single byte indicating the status of the last command packet. Each bit has a particular meaning

Bit	Name	Description
0	Input Voltage Error	If set to 1, indicates the operating voltage is out of range. The high and low range is set by registers 12 and 13.
1	Angle Limit Error	If set to 1, indicates the goal position sent is out of range. The limit range is set by registers 6-9.
2	Temperature Error	If set to 1, the internal temperature of the AX-12 has exceeded the limit set in register 11.
3	Range Error	Set to 1, if the parameter data is out of range for the specific instruction in the last packet.
4	Checksum Error	Set to 1, if the checksum received in the last packet is incorrect.
5	Overload Error	Set to 1, if the load on the servo has exceeded that which is set by registers 14 and 15.
6	Instruction Error	Set to 1, if an invalid instruction has been received or an Action command has been received without sending a Reg Write Command.
7	Not Used	

## Data1-N

This is the optional data parameter(s). It will be sent if you have requested data from the AX-12.

## Check Sum

All bytes except for the start bytes are added together. Only the first 8 bits are used. Once all bytes are added you must invert the bits.

In summary you send a command packet and get back a status packet. If you requested data it will be contained in the status packet as data parameters. At 1,000,000 bps this all takes place very quickly.

## Microcontroller Interface

Another strength of the AX-12 is that you don't need a servo controller. It is a simple task to connect a microcontroller to the AX-12. However a few things need to be kept in mind when selecting a microcontroller to use. First, to take advantage of the high speed interface of the AX-12 you need a microcontroller that can communicate at 1,000,000 bps. You also need the ability to place the transmit lead in a high impedance state when receiving data from the AX-12.

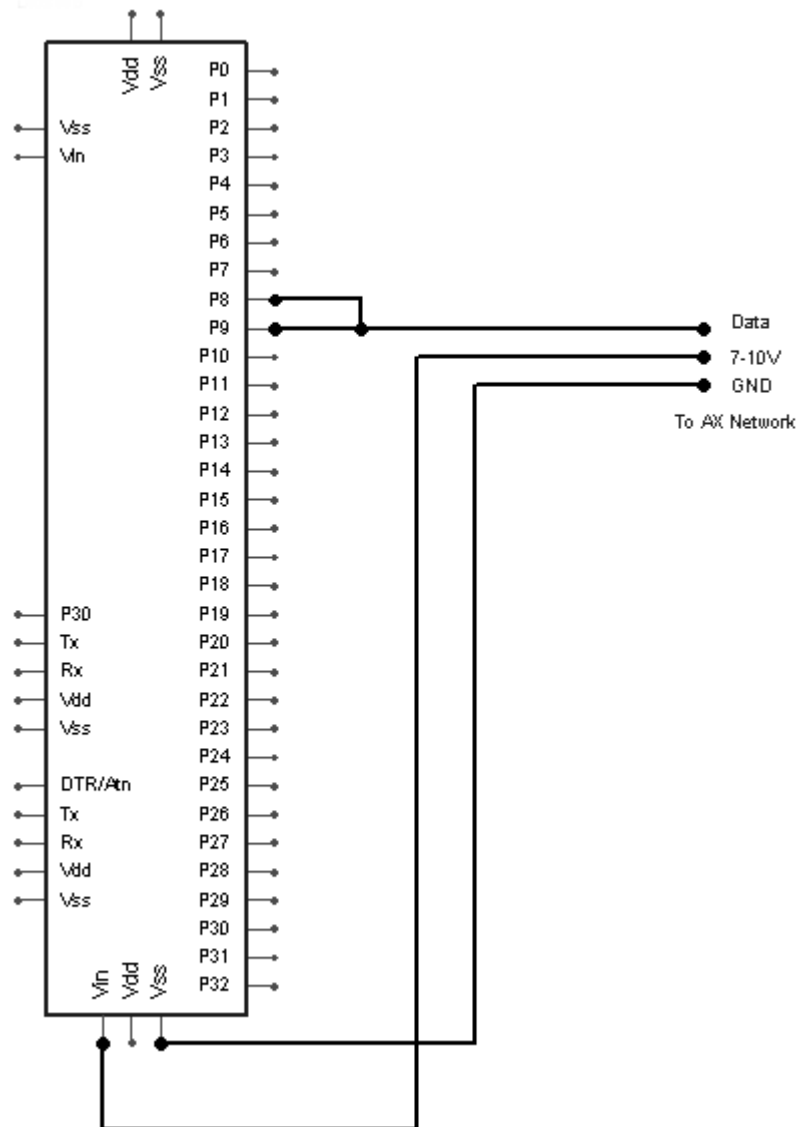
The DiosPro microcontroller is a perfect match. With just a few lines of code you can communicate with the AX-12.

Schematic 1 is a simplified schematic of the DiosPro Workboard. It shows how simple it is to connect an AX-12 to a DiosPro chip or board. You simply need to connect ports 8 and 9 of the DiosPro together. These are then connected to the data leads on the AX-12.

### Important

The Dios Workboard can handle a wide range of input voltages, but if you are using it to power an AX-12 device you should use a power source in the range of 8-12v.

Each AX-12 has a built-in LED. Program 12 shows how easy it is to blink that LED. You simply need to change the device constant to reflect the actual device ID of the AX-12 you are communicating with.



Schematic 1

I have written a complete library that is now included with the free Dios compiler.

## AX-12 Bot Example

Many of you who have seen my robotic articles in the past know that I like to use a simple IR interface to test my robot creations. The commands for doing this are built-in to the DiosPro so a single call will allow you to use a universal remote to issue commands to even the most complex bot.

As I mentioned earlier the AX-12 can be operated in full rotational mode. This means they can be used as very powerful gear motors with full speed control. Unlike a modified servo, the AX-12 gives you a full range of control over the speed.

To place an AX-12 in full rotational mode, simply issue the following commands:

```
AXwriteword(AXnum,AXCCW_Angle_Limit,0)  
AXwriteword(AXnum,AXCW_Angle_Limit,0)
```

Now, to place the motor in motion issue the command:

```
AXwriteword(IAXnum,AXMoving_Speed,speed)
```

Speed can be any value of 0 to 1023. If you set bit-10 the motor will rotate in the opposite direction. A simple way to do this is by adding 1024 to the speed.

## Construction

I used a Dios Workboard Deluxe to make experimenting easier. It was a simple matter to wire the AX-12 network cable and Vishay IR to the breadboard shown in Figure 9.

The base for the AXbot is made from a piece of 1/8" x 5" x 7" wood or Plexiglas. You can also use compressed PVC. I shaped the front of my AXBot as shown in Figure 8. Feel free to make as many modifications as you need.

The four outside standoffs are marked by placing the workboard on top of the base and marking the 4 holes. Drill 1/8" holes into these 4 holes.

Mark the holes for the caster or roller ball for the front support again as shown in Figure 9. This support should be as far forward as you can place it.

```
DiosPro  
func main()  
  
    const device 18  
    AXinit()  
  
Loop:  
    AXwritebyte(device,25,1)  
    pause 200  
    AXwritebyte(device,25,0)  
    pause 200  
    goto Loop  
  
endfunc  
  
include \lib\AX.lib
```

Program 1

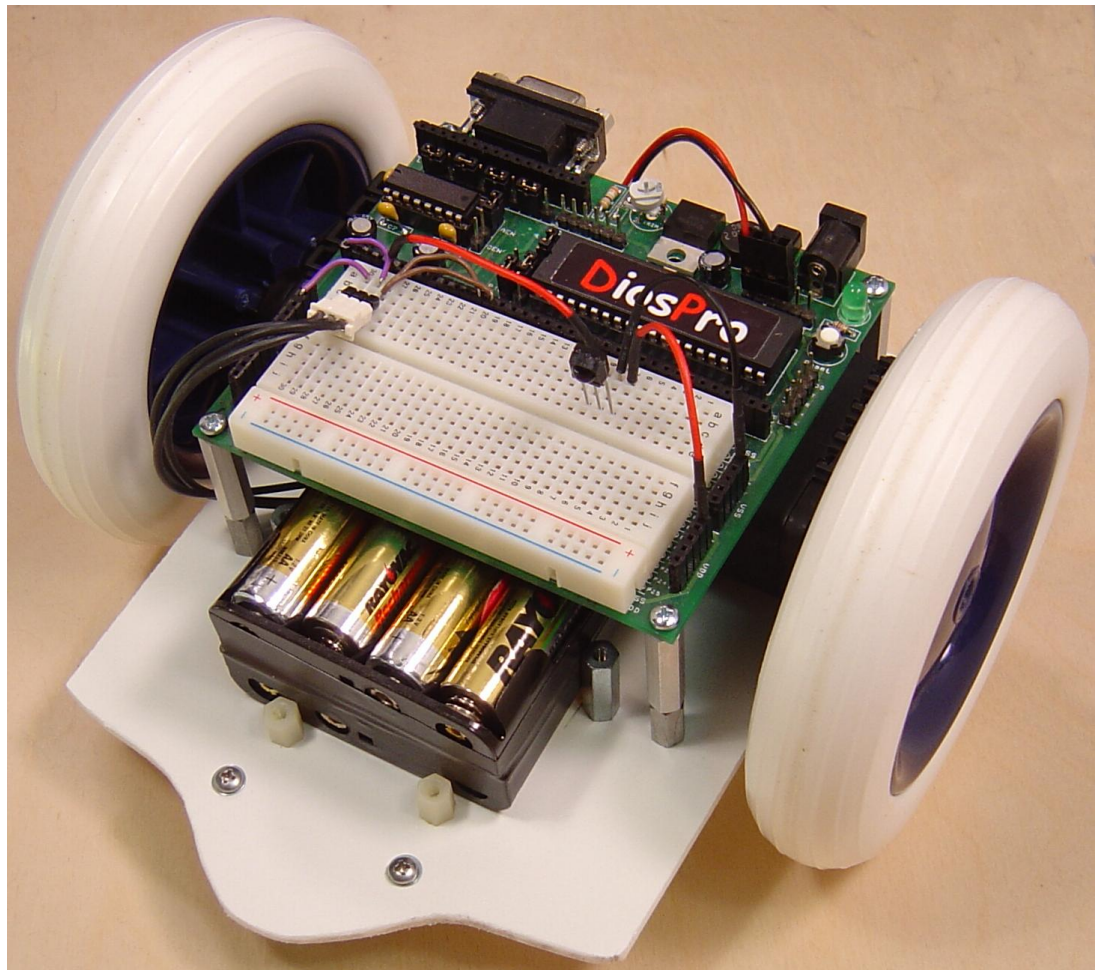


Figure 8

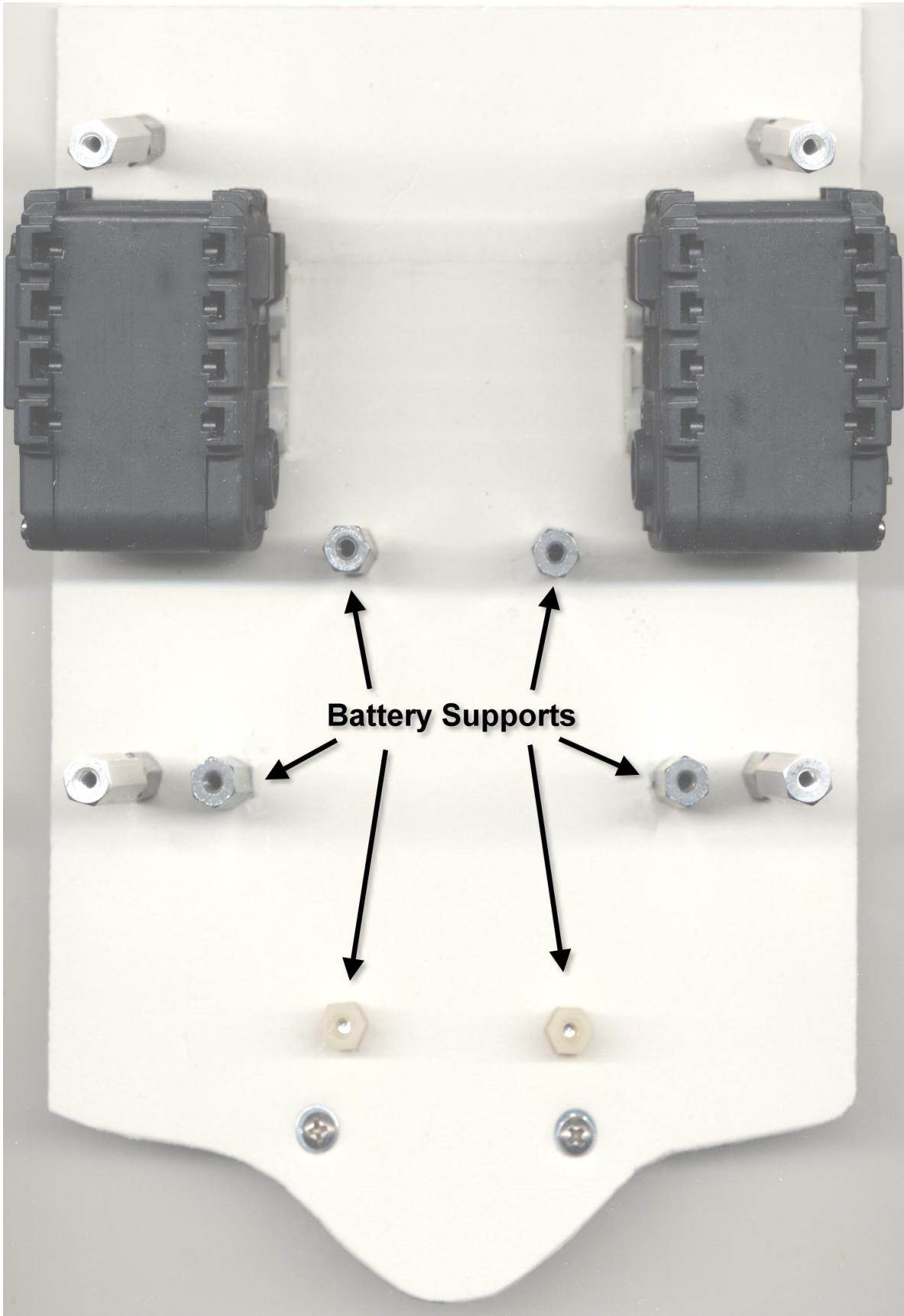


Figure 9

Use the 12S mount that comes with the AX-12 to mark 4 holes so that you can mount each AX-12 in the position shown. Placement is not critical but the wheels should form a triangle with the front castor or roller ball. The AX-12 should be roughly located as shown in Figures 9 and 10.

Once the holes are marked you will need to drill 1/16" holes so that you can mount the 12S. Once attached to the base you can mount the AX-12 to the 12S mount.

You will need to make a 9.6v battery pack for your AXbot. I used two 4-Cell AA battery holders connected in series as shown in Figure 11. The battery holders come with foam tape on the back. I removed this and used black tape to hold the two pieces together. If you already have a pack it may just be a simple matter of creating an adapter for the connector. You may also use a 7.2v pack but the bot won't be as fast and it won't have as much run time.

You need to add some standoffs to hold the battery pack in place. To mark the location of these standoffs just place your pack on the base. Make sure it is placed a bit forward so that the weight of the packs rests near the front of the wheel/roller triangle. This will keep your bot from flipping when moving forward. When in place, mark the locations for the battery supports as shown in Figure 9. I used 3/4" standoffs for the rear and sides and 1/4" for the front. This will allow you to slide the pack in the front of the AXbot.

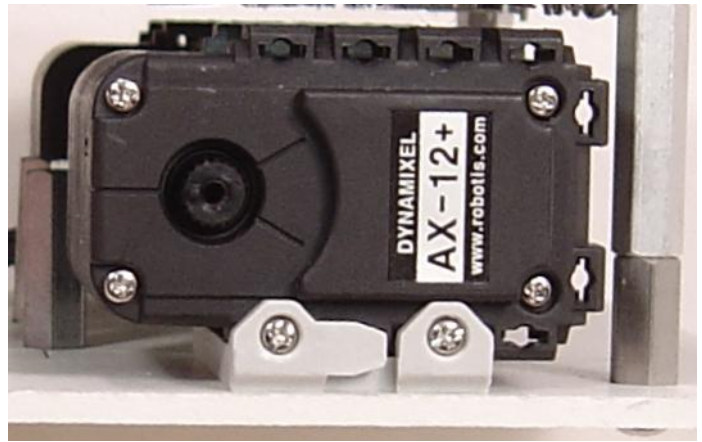


Figure 10



Figure 11

The Dios Workboard Deluxe has a header for supplying power. This is convenient if you don't want to use the built-in coax connector. By connecting your battery to the COAX and AC2 as shown in Figure 12 you connect to the bridge rectifier and switch. This allows you to use any polarity on the connection and the ability to switch the power on and off via the switch.

**Important - Make sure the connection is as described. If you connect your battery to the wrong pins you will damage the DiosPro chip. Depending on the connector that you use, you may have to move one of the wires on the connector.**

To connect the AX-12 network to the DiosPro. You need to use a 3-pin header as mentioned earlier. Plug the other end of the header into the breadboard, then wire data and power leads as shown in Figure 13. You will also use the breadboard to wire the Vishay IR module.

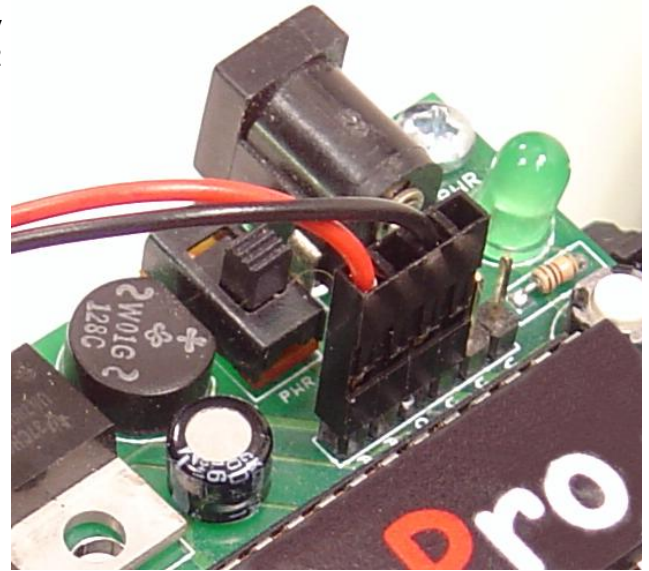


Figure 12

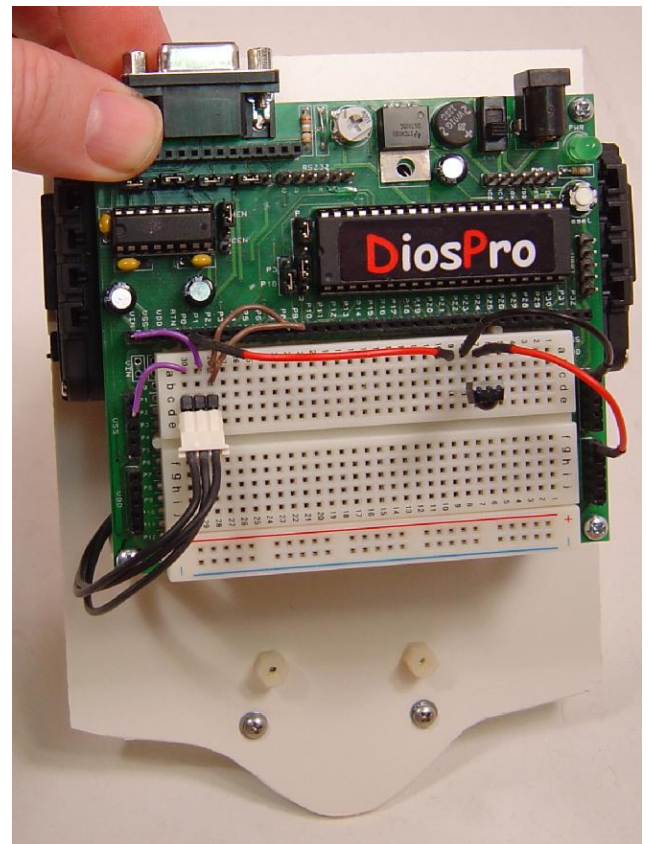
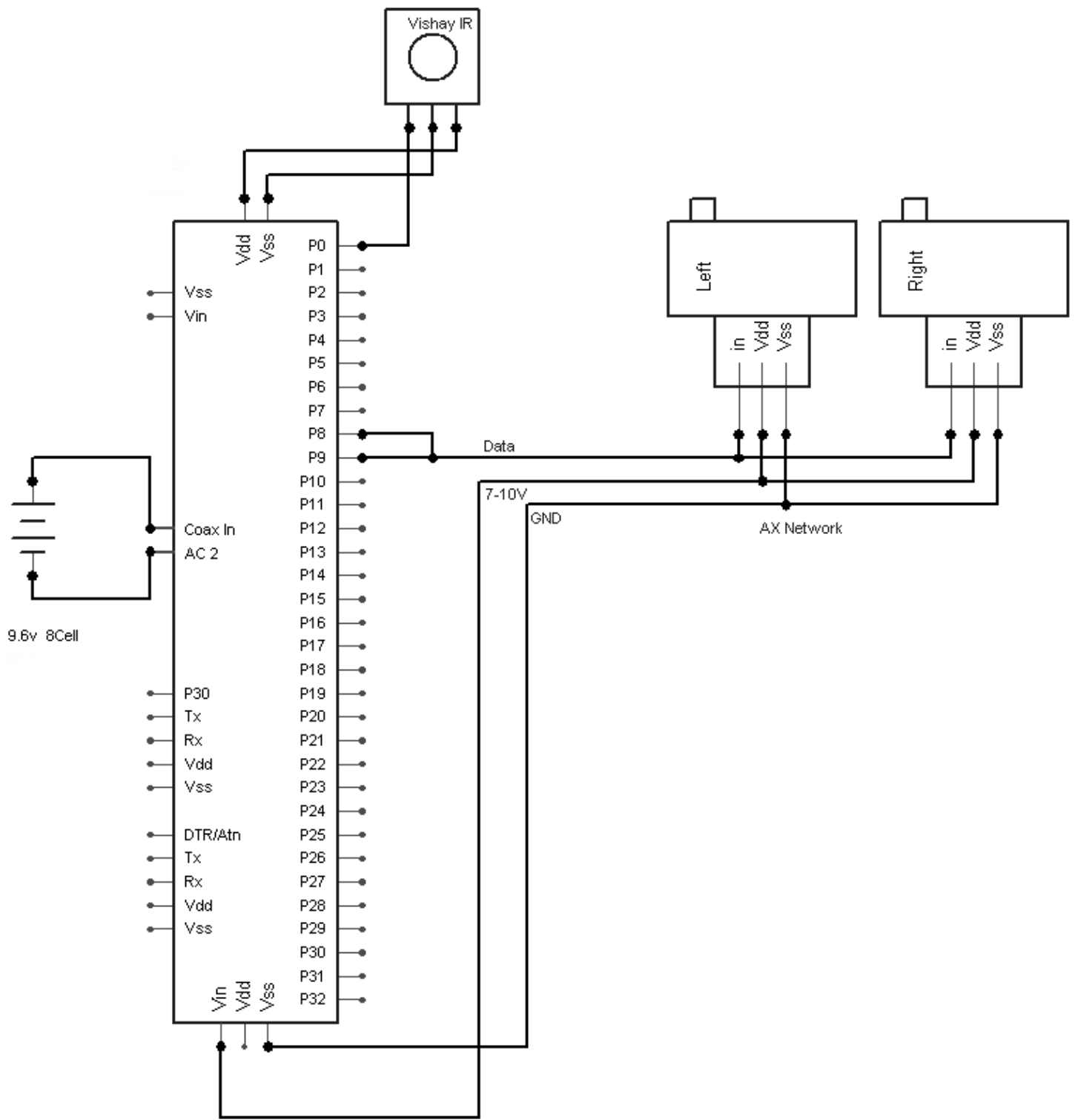


Figure 13

Schematic 2 shows the connection details of the AX-12 network and the IR module.



Schematic 2

## Testing the AXbot

I have included 2 programs to get you started in programming the AXbot. The first program called AX12IRbot.txt uses the channel and volume pads on an IR remote to move the AXbot. The remote can be any Sony compatible remote. If you are using a universal remote just set it to Sony TV or VCR mode. Make sure you set the lservo and rservo constants at the begging of the program to match the two AX-12's you are using.

The AX12IRBot2.txt program is a bit more advanced. This program adds ramp logic to smooth the speed transitions of the AXbot. Feel free to experiment with the ramp routines to fine tune your own AXbot.

## What's Next

We have barely taxed the power of the AX-12. Actuators like the AX-12 are the future of robotics. Next month I will show you step by step how to build a giant BioCrab using 18 AX-12's.

If you are into robotics, animatronics or anything that requires a high power servo I recommend picking up a couple AX-12's. The price of just under \$45 is pretty much what you would pay for an advanced servo with less holding power and no feedback.

All the example programs as well as the source are available for download at:  
<http://www.kronosrobotics.com/Projects/ax12a.shtml>

## Parts

Available from CrustCrawler - [www.crustcrawler.com](http://www.crustcrawler.com)

- 2, AX-12 Smart Servos

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

- 2, 4-Cell, AA battery holders. #16323
- DiosPro Chip #16148
- Dios Workboard Deluxe #16452
- Vishay IR module #16226
- 5/8" Roller ball #16311

## Misc

- 1, 5 x 7 x 1/8" thick Plexiglas, compressed PVC or Hobby Plywood. You can pick one of these up at most home or craft centers.
- 2, Wheels. These can be foam wheels that you can pickup from any hobby store. You can also use surplus wheels like those found on baby carriages.
- 4, 1-1/2" #4 standoffs
- 4, 3/4" #4 standoffs
- 2, 1/4" #4 standoffs
- 16, 3/8" #4 machine screws
- 2, #4 hex nuts.

I purchased all my standoffs and #4 hardware from Jamco Electronics at [www.jameco.com](http://www.jameco.com)