

# *HOBBY SERVO*

## *FUNDAMENTALS*

*BY: DARREN SAWICZ*

## INTRODUCTION

Hobby servos are a popular and inexpensive method of motion control. They provide an off-the-shelf solution for most of the R/C and robotic hobbyist's needs. Hobby servos eliminate the need to custom design a control system for each application. Without hobby servos (hereafter referred to only as servos) you would have to:

- design a control system
- analyze the transient response
- fine tune the feedback loop
- determine the proper gear ratio for the desired speed or efficiency
- choose a motor
- build the mechanical sections
- build the amplifier and motor driver
- try to make it fit inside what ever you're trying to control!

Although servos are very common, technical information is hard to come by. A search of the Internet will bring up bits and pieces but it is hard to locate a single source for all the information. In this document I will attempt to provide all the information you would need to hack a servo or make your own.

## SERVO BASICS

Throughout this document I will be using the Tower Hobbies TS-53 servo to describe how servos work. Other servos may be a different size or have metal gears or bearings but the basic concept of operation is the same.

Since the early 1990's servos have used a de-facto standard pulse width modulation technique to control the position of the output shaft. The

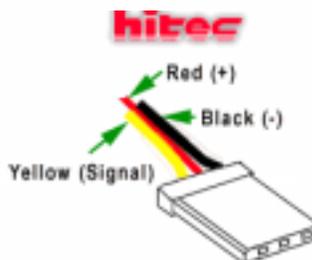
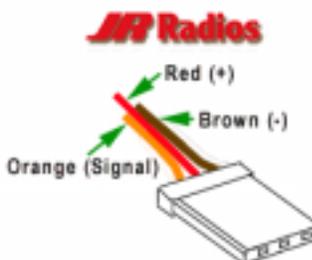
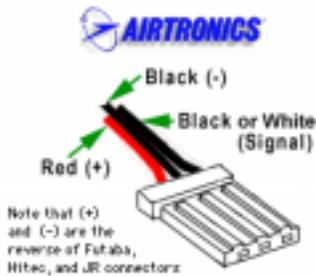
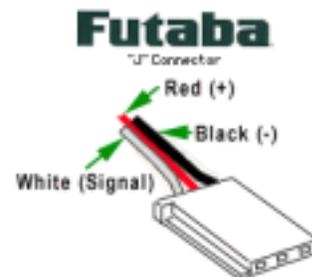
pulse is fed to the servo via a control line. The control line does not supply power to the motor directly it is an input to a control chip inside the servo and as such it does not have to supply much current to the servo. Consequently if you are designing a servo controller you may use just about any technology (CMOS, TTL, Discreet Components) to drive the control line.

A separate power wire supplies the power to

the servo. The ground for power is also used as the ground for the control line.

Although there are only three wires,

manufacturers haven't standardized the pinouts. Most connectors are wired the same but there are



exceptions like the Airtronics connector where the power and ground wire is reversed. Connecting a servo that is wired incorrectly can blow your servo or fry the receiver or both.

Most servos can be rewired simply by lifting the plastic tab holding the pin into the connector, pulling the pin out the back, and reinserting the pins in the proper place.

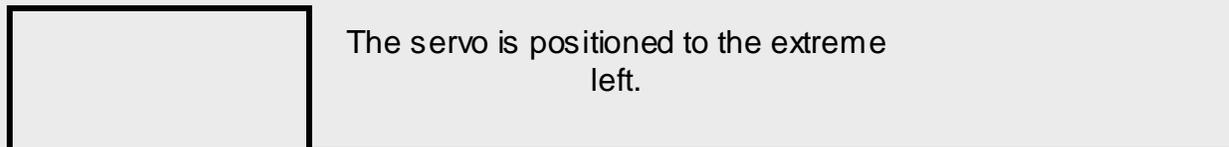
The power wire carries the majority of the current to the motor. The current varies from almost nothing (9.6mA for the TS-53) when the servo isn't receiving any control signals, to a maximum current when the servo is fully loaded (600mA for

the TS-53).

The receiver regulates the voltage on the power wire when it is used with R/C equipment. The servos do not get the full voltage of the battery pack. Applying more than 5 Volts to the servos can damage them.

**FIGURE 1: SERVO CONTROL TIMING**

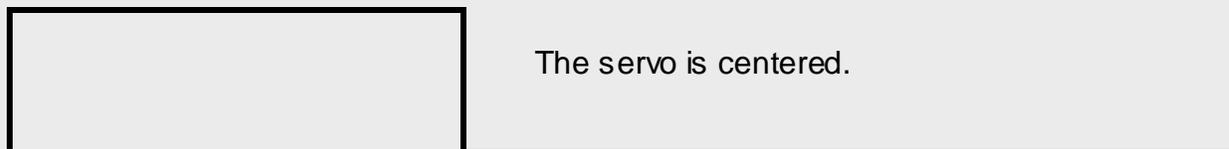
1ms (Minimum Required Time)



2ms



1.5ms



## ELECTRICAL SIGNALS

At first glance the electrical signals used to control servos seem overly complex. When you consider that one signal line is presenting the servo with digital direction and proportion information you start to see the elegance of the solution.

This elegance comes at a price for the poor hobbyist with little or no background in electronics. You can't just hook it up to a battery and watch it go (unless you modify the servo to do just that). There are however circuits which allow you to control a servo without using a receiver. Lets look at the signal we need to generate.

The voltage on the control line should be 0 Volts for a low signal (logic 0) and 5 Volts for a high signal (logic 1). The voltage to the control line should be applied through a 10k resistor to limit the current in case something fries.

In **FIGURE 1** you can see the timing relationship between the pulse width and servo position.

The servo's control logic needs to see at least a one millisecond pulse before the actual pulse width modulated (PWM) signal.

If you were trying to implement the timing from a microcontroller you could use a register to hold a 0 to 100% PWM signal and tack it on to the end of a 1ms-delay routine

The control part of the signal is broken down into the 1ms minimum time, the 1ms PWM signal, and a roughly 40ms delay. This delay is not as critical as the other parts of the timing signal. It is essentially the dead time between control signals. If you repeat the control signals too quickly (i.e. 10ms delay) the servo will buzz and jitter. If you repeat the control signals too slow (i.e. 70ms delay) the servo will shut off between signals and its position will not remain constant.

In the case of R/C equipment this dead time is used by the transmitter to transmit the other channels. If the channels were placed back-to-back there would be enough room for approximately 20 channels before you would have to transmit the first channel again.

The receiver demultiplexes the received pulses and puts one pulse on each output. The rising edge of each pulse is the start of a new channel.

It is important to note that the "channels" of a two or four, etc. channel transmitter, are not separated by frequency like radio or TV stations (Frequency Division Multiplexed), they are pulses on the same radio channel separated only by time (Time Division Multiplexed). The only way you can change the radio channel you are on is to replace the crystal.

If you are designing a multi-channel controller you will soon discover that because the PWM portion of the signal is variable you will have a

variable length delay time. For example:

You design an eight-channel controller. The fixed time component is  $1\text{ms} \times 8\text{ channels} = 8\text{ms}$ . The variable PWM signal can vary between  $0\text{ms} \times 8\text{ channels} = 0\text{ms}$  when they are all fully counter clockwise, to  $1\text{ms} \times 8\text{ channels} = 8\text{ms}$  when they are all fully clockwise. This adds up to between  $8\text{ms}$  and  $16\text{ms}$  depending on the positions of the servos.

If you were trying to have a constant repetition rate of  $40\text{ms}$  you would need to change the dead time accordingly. With a microcontroller you could add up all the constant time plus all the variable time and subtract it from  $40\text{ms}$  to calculate the dead time. This takes time and uses up the microcontroller's resources needlessly. The dead time is not written in stone. It does not have to be the same each time.

**NOTE:** If you are custom building a wireless transmitter the  $\approx 40\text{ms}$  repeat time is required to comply with FCC and DOC bandwidth limitations. The sidebands generated by the FM modulation cannot extend into adjacent channels.

## SERVO CIRCUITS

Servos have their own proprietary circuitry built inside the servo case. This circuitry consists of a pulse width comparator, which compares the incoming signal from the receiver with a one-shot timer whose period depends on the resistance of a potentiometer connected to the servo's drive shaft. This feedback is what provides the stability for the control circuitry. The difference between the control signal and the feedback signal is the error signal. This error signal is used to control a flip-flop that toggles the direction the current flows through the motor. The outputs of the flip-flop drive an H-Bridge circuit that handles the high current going through the motor.

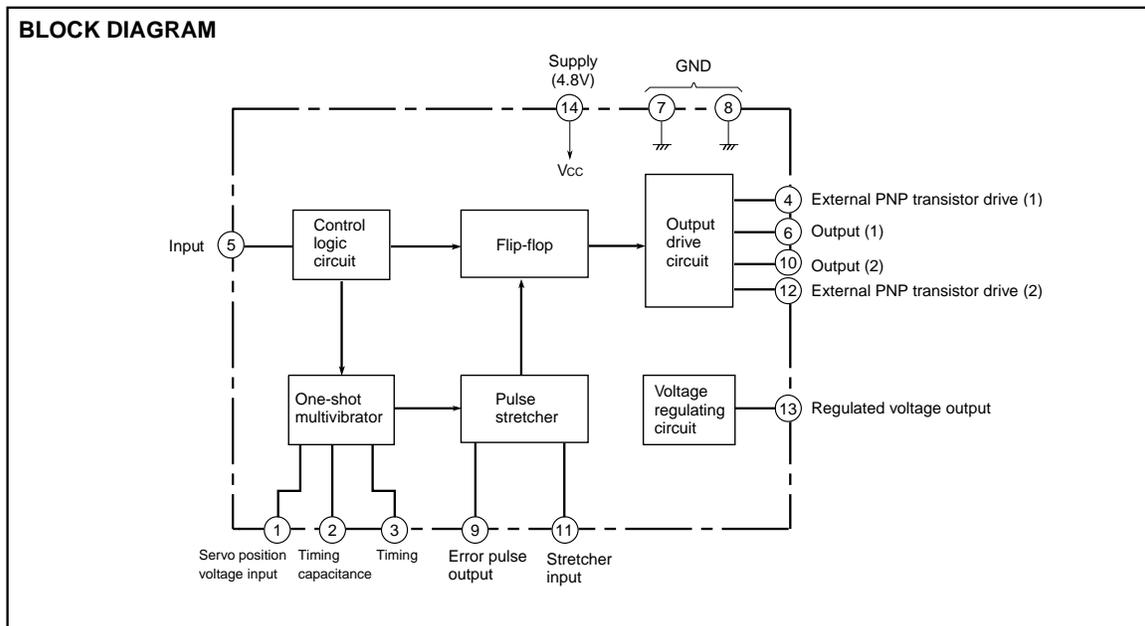
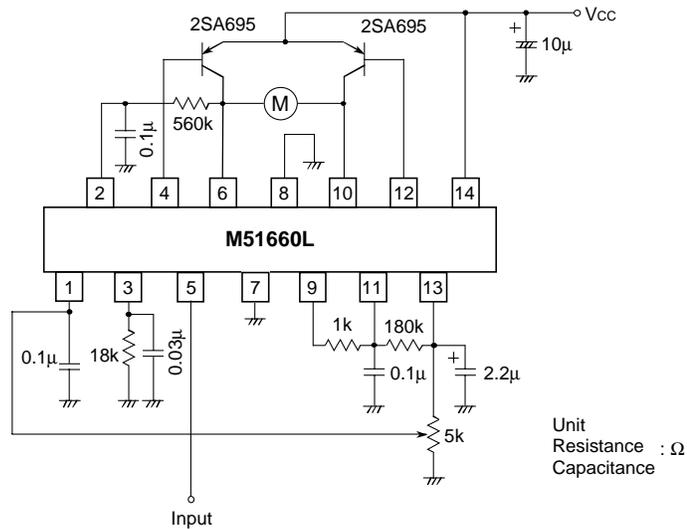
If you were going to try to build a circuit to take a PWM signal and convert it to a motor position, you'd probably end up with a fairly large circuit. A microcontroller would do the job but if you can't program you're back to square one.

Thankfully there is a chip available which does the work for you. The M51660L servo motor control chip from Mitsubishi contains all the electronics needed to decode the signal and control a motor.

All you supply are two PNP transistors for the upper half of the H-bridge and a handful of resistors and capacitors. A complete circuit is given on the second last page of their data sheet:

### APPLICATION EXAMPLE

#### Servo motor control circuit for radio-controlled



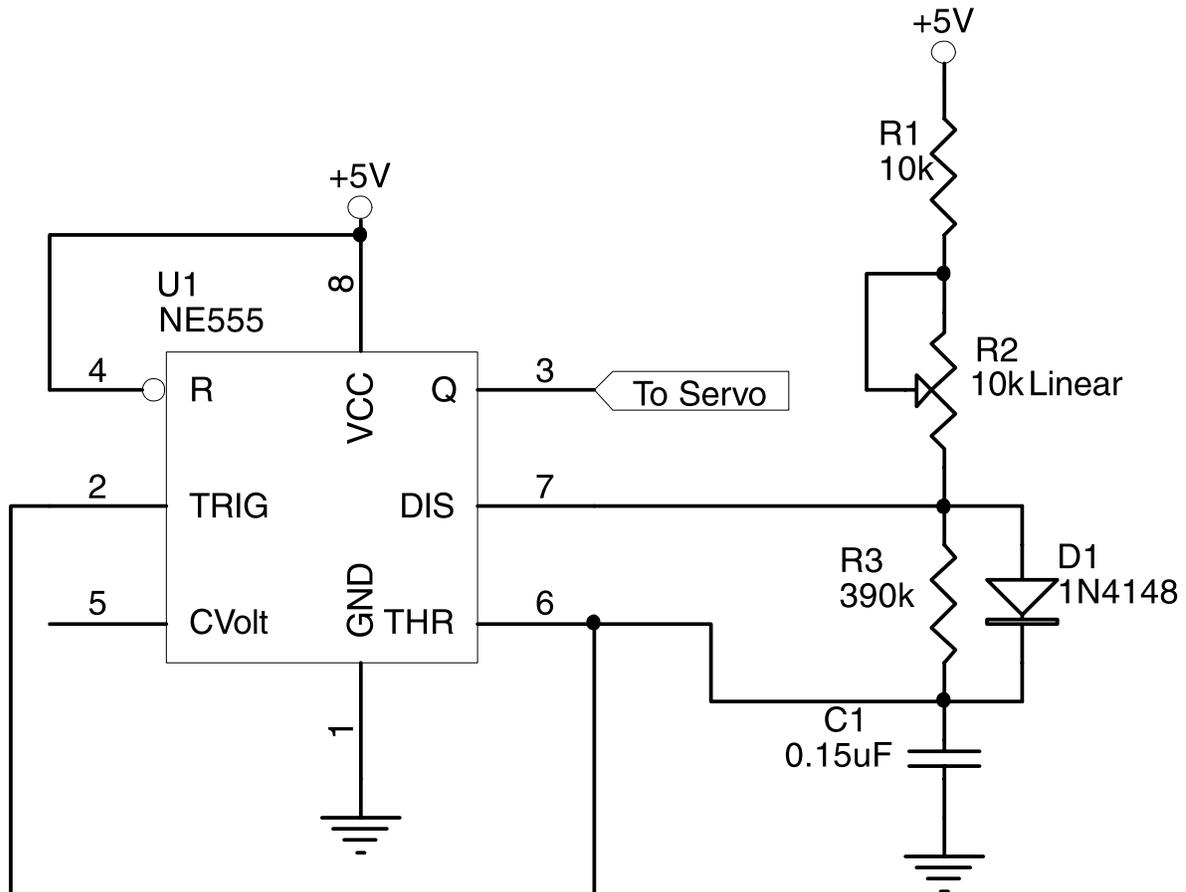
If you don't have a radio transmitter and receiver handy you won't have any way of generating the control signal for the servo. A simple circuit using a 555 timer chip can generate the needed signal.

$$0.693(10k + 0)0.15e^{-6} = 1.039mS$$

The time the signal is low is equal to:

$$0.693(390k)0.15e^{-6} = 40.54mS$$

The diode is there to bypass resistor



The equations for the 555 timer are simple and easy to use. They are as follows:

$$T_{HIGH} = 0.693(R_1 + R_2)C$$

$$T_{LOW} = 0.693(R_3)C$$

Since  $R_2$  is variable the time the signal is high will vary from:

$$0.693(10k + 10k)0.15e^{-6} = 2.079mS$$

to

$R_3$  during the  $T_{HIGH}$  charging phase.

The timing values are close enough to work with just about any servo. If you want exact timings you can replace resistor  $R_1$  with a trimmer potentiometer.

## SERVO REVERSERS

Servos occasionally have to be mounted in such a way that they respond opposite to the intended direction. You push the control stick left and the car turns right. You cannot simply reverse the leads to make the motor turn the other way. What is needed in this case is the inverse of the applied signal.

You cannot simply invert the signal or you will get a  $\approx 2\text{mS}$  off time with a  $\approx 40\text{mS}$  on time. Lets look at what we need to do:

$T_{\text{HIGH}} = 1\text{mS}$  the servo is fully CCW

$T_{\text{HIGH}} = 2\text{mS}$  the servo is fully CW

Therefore, you want a circuit which converts a  $1\text{mS}$  pulse into a  $2\text{mS}$  pulse and vice versa.

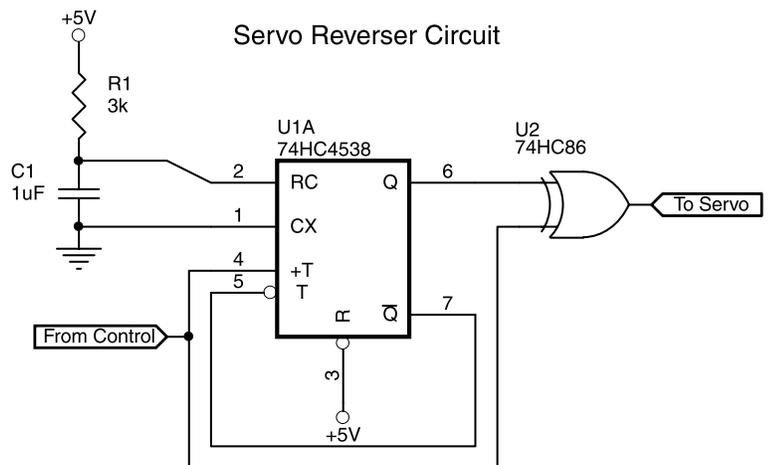
It might seem a little daunting but lets think of it in terms of subtraction. Three minus one leaves two. Three minus two leaves one. **JUST WHAT WE NEED!**

Now the problem is where do we get the three from, and how do we subtract times from pulses?

The answer is simpler than you think. A one-shot timer supplies the  $3\text{mS}$  pulse when it is triggered by the rising edge of the control signal. This lines up the rising edges so the "subtraction" has a reference point. The "subtraction" is even easier. Look at the truth table for an XOR Gate:

X	Y	OUTPUT
0	0	0
0	1	1
1	0	1
1	1	0

When the control signal (X) starts it goes high for at least  $1\text{mS}$ . The rising edge of X triggers the one-shot timer which then supplies the signal Y. Now notice on the chart that when X and Y are '1' the output is '0'.



Suppose X drops out at  $1.3\text{mS}$ . The control signal (X) goes low but the  $3\text{mS}$  one-shot signal (Y) stays high. Again looking at the chart, when X is '0' and Y is '1' the output is '1'. This continues until the one-shot times out at  $3\text{mS}$  at which time its output goes low until it is re-triggered by the next control signal. At the time Y goes low you can see that the output has been high for  $3\text{mS} - 1.3\text{mS} = 1.7\text{mS}$ . And thus the servo has reversed from a little to the left to a little to the right.



## **MODIFYING A SERVO**

There may come a time when it is advantageous to use the servo as a normal DC gearhead motor. Both the inner mechanical and electrical mechanisms need to be modified to allow for continuous rotation.

Start by removing the control horns attached to the output shaft. The gear cover won't come off if you don't. Next remove the four screws on the bottom of the servo case. They hold both the gear cover and bottom cover in place.

Carefully remove the gear cover. Pay attention to the placement of the gears. They can only go one way.

The gear you need to modify is the large black one in the picture above. It connects the output shaft and the feedback potentiometer to the rest of the drivetrain.

There is a mechanical stop on the gear that strikes another stop on the inside of the gear cover. This prevents the shaft from turning too far and damaging the potentiometer.

You need to remove this stop to allow for full rotation. It is easy to

remove using a pair of flush cutters, an X-acto knife, or a file.

## Remove This Tab



Removing the stop on the gear is only half the problem. There are still stops built into the potentiometer. You must now remove it as well.

The potentiometer is removed by desoldering it from the circuit board. You must then replace the pot with two resistors of equal value. Two 2.2K resistors should work well. The values aren't critical as long as they are equal.

One end of each resistor goes into the center (wiper) hole. The free end of each resistor then goes into the remaining hole on either side.

The resistors form a fixed voltage divider that replaces the variable voltage divider of the potentiometer. This tricks the servo into thinking it is centered all the time.

When you supply a control signal  $>1.5\text{mS}$  it drives the motor CW, but because the feedback signal is telling the servo it is still centered the servo keeps trying to drive it

clockwise continuously. When the control signal is  $< 1.5\text{mS}$  the servo drives the motor CCW for the same reason.

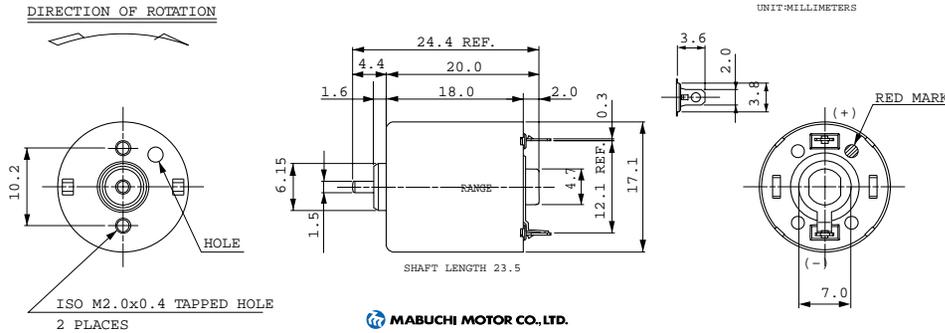
This solution is good if your servo is always moving. If you try to stop the servo you will probably notice that the servo creeps in one direction or the other, or it just vibrates back and forth. This is because you have removed the feedback loop. If the center signal produced by the voltage divider is different from the center signal generated in the transmitter the servo will move instead of stop.

The best way around this problem is to build your own controller with a microcontroller (or otherwise) and incorporate a deadband around 1.5mS. When the control signal is within this deadband ( $1.5\text{mS} \pm 5\mu\text{S}$  or so) the servo doesn't move.

If you are using an M51660 servo motor control chip you can increase the deadband by increasing the resistance on Pin 9.

If you want to build your own controller or just use the servo as a regular gearhead motor you can simply de-solder the motor from the circuit board, attach wires to the motor terminals, and put it back in the servo case.

# RF-020TH



MABUCHI MOTOR CO., LTD.

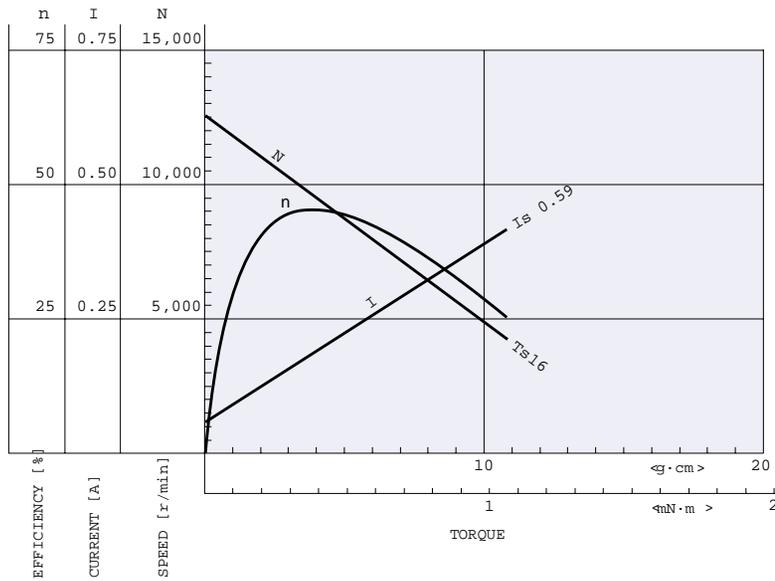
Usable male screw length 1.2 max. from motor mounting surface.

WEIGHT: 16g (APPROX)

MODEL	VOLTAGE		NO LOAD		AT MAXIMUM EFFICIENCY					STALL TORQUE		
	OPERATING RANGE	NOMINAL	SPEED	CURRENT	SPEED	CURRENT	TORQUE		OUTPUT	EFF	TORQUE	
			rpm	A	rpm	A	OZ-in	g-cm	W	%	OZ-in	g-cm
RF-020TH-10210	2.0 ~ 5.0	4.5V CONSTANT	12600	0.058	9580	0.184	0.055	3.93	0.386	46.6	0.228	16.4

## RF-020TH-10210

4.5V



**DATA SHEET FOR TS-53 SERVO'S ELECTRIC MOTOR**