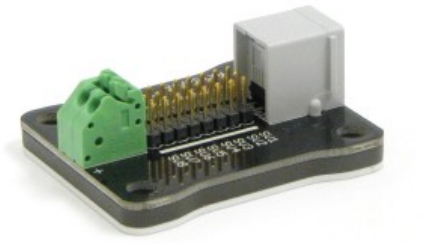
What is NXTServo-v3

NXTServo-v3 is an 8 channel Servo controller module. It allows you to control speed and position of up to 8 RC servo motors (or micro RC servos) using I2C commands sent from NXT or EV3.



Previous models of the NXTServo had an LED located on the side of the device. However, the newer model of NXTServo, has had the LED removed.

Supported Servos

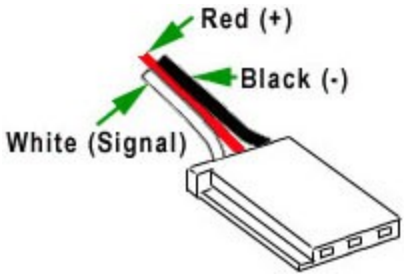
Any RC servo motor with "1500 µS neutral" specifications.

The common brands available for this spec are: Hitec, Futaba. You could use 90° or 180° or continuous rotation servos.

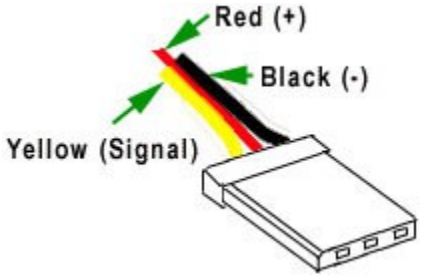
Supported connectors: Futaba-J and Hitec-S compatible plugs.

Pictured as follows:

Futuba-J



Hitec-S



Power Supply

NXTServo module requires two different power sources. The power for logic section is 5 volts, and is taken from NXT or EV3, whereas, power for motors is taken from an external source. This power depends on capacity and ratings of the servos you will be using and the load they will be carrying.

⚠ **NOTE**

NXTServo is rated to handle maximum of 16 volts. However, most common RC servo motors are rated for much lower, and always ensure that you are not exceeding the rated voltage of your RC servo motor.

⚠ **WARNING**

While connecting to NXT or EV3, do not connect to Motor Port, doing so may damage the NXTServo circuit. (Always connect to Sensor Port).

As you attach more servos to NXTServo, more capacity batteries will be needed. Use following table as a general guideline in choosing external power source.

Function	Recommended battery
Upto 8 mini/micro servos, any load	6V - 4AA batteries
Upto 2 Standard RC Servos, moderate load	6V - 4AA batteries
Upto 6 mini/micro RC servos, moderate load	6V - 4AA batteries
Upto 4 standard RC servos, heavy load	7.2V - RC Lithium Polymer battery pack
8 RC servos, heavy load	7.5 V - RC NiMH battery pack

⚠ **WARNING**

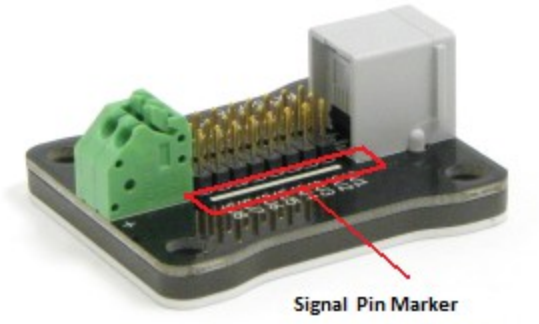
While connecting the battery ensure to connect positive and negative terminals to correct connectors. NXTServo-v3 is designed such that reversed polarity will not operate your RC servos until polarity is corrected. However, reversed polarity for extended period may cause fire, damage NXTServo-v3 and/or your servo motors.

It is recommended to disconnect (or power-off) the external power from NXTServo while NXT is turned off. (While NXT is turned off, the connected servos will draw small amount of current from the external power source.)

Connections

Connect NXTServo to NXT using standard NXT cable supplied with Mindstorms kit or Flexi-cable from mindsensors.com.

Servo motors are connected directly on the NXTservo, with it's signal pin pointing towards the thick white line adjacent to the connector.



Connect the external battery to NXTServo while ensuring the correct polarity (as marked near the External Power Connector).

How to identify Signal Pin for common RC servos

Table below lists color coding for various servos available in market:

Wire colors (in following order)	Signal Pin
White/Red/Black	White
Yellow/Red/Black	Yellow
Orange/Red/Brown	Orange

The center pins is always power, and the remaining pin is always ground.

Programming Techniques

EV3:

To use capabilities of the sensor, please download EV3 blocks available at following URL:

http://www.mindsensors.com/index.php?controller=attachment&id_attachment=236

Installation instructions for EV3 block are available at:

http://www.mindsensors.com/content/13-how-to-install-blocks-in-ev3

Download EV3 sample program from following URL and modify it to suit your needs.

http://www.mindsensors.com/index.php?controller=attachment&id_attachment=237

NXT-G:

To use capabilities of the sensor, please download NXT-G blocks available at following URL:

http://www.mindsensors.com/index.php?controller=attachment&id_attachment=40

Installation instructions for NXT-G block are available at:

http://www.mindsensors.com/content/21-nxt-g-blocks-how-to-install-blocks

Download NXT-G sample program from following URL and modify it to suit your needs.

http://www.mindsensors.com/index.php?controller=attachment&id_attachment=41

⚠ **NOTE:** While using with NXT-G, ensure to use latest firmware on your NXT.

RobotC:

Download the library file and sample programs available at following URL:

https://github.com/botbench/robotcdriversuite

You may modify the sample programs to suite your needs.

NXC:

Download the library file and sample programs available at following URL:

http://www.mindsensors.com/index.php?controller=attachment&id_attachment=38

You should include the library file (NXTServo-lib.nxc) in your program with #include directive, and use the API's provided by the library.

Alternately, you may modify the sample programs to suite your needs.

⚠ **NOTE:** While using with NXC ensure to use latest firmware on your NXT.

I2C register Summary

General Registers

Reg	Read	Write
0x41	Battery voltage (for EV3 compaatible firmware, Battery Voltage is relocated to 0x62).	Control Register to write commands (see section on commands)

Servo Position Registers

Reg		
0x42	Servo 1 position: low byte time in µS	Servo 1 position: low byte time in µS
0x43	Servo 1 position: hi byte time in µS	Servo 1 position: hi byte time in µS
0x44	Servo 2 position: low byte time in µS	Servo 2 position: low byte time in µS

Reg		
0x45	Servo 2 position: hi byte time in µS	Servo 2 position: hi byte time in µS
0x46	Servo 3 position: low byte time in µS	Servo 3 position: low byte time in µS
0x47	Servo 3 position: hi byte time in µS	Servo 3 position: hi byte time in µS
0x48	Servo 4 position: low byte time in µS	Servo 4 position: low byte time in µS
0x49	Servo 4 position: hi byte time in µS	Servo 4 position: hi byte time in µS
0x4A	Servo 5 position: low byte time in µS	Servo 5 position: low byte time in µS
0x4B	Servo 5 position: hi byte time in µS	Servo 5 position: hi byte time in µS
0x4C	Servo 6 position: low byte time in µS	Servo 6 position: low byte time in µS
0x4D	Servo 6 position: hi byte time in µS	Servo 6 position: hi byte time in µS
0x4E	Servo 7 position: low byte time in µS	Servo 7 position: low byte time in µS
0x4F	Servo 7 position: hi byte time in µS	Servo 7 position: hi byte time in µS
0x50	Servo 8 position: low byte time in µS	Servo 8 position: low byte time in µS
0x51	Servo 8 position: hi byte time in µS	Servo 8 position: hi byte time in µS

Speed Registers

Reg	Read	Write
0x52	Speed: Servo 1	Speed: Servo 1
0x53	Speed: Servo 2	Speed: Servo 2
0x54	Speed: Servo 3	Speed: Servo 3
0x55	Speed: Servo 4	Speed: Servo 4
0x56	Speed: Servo 5	Speed: Servo 5
0x57	Speed: Servo 6	Speed: Servo 6

Reg	Read	Write
0x58	Speed: Servo 7	Speed: Servo 7
0x59	Speed: Servo 8	Speed: Servo 8

Quick Registers

Reg	Read	Write
0x5A	NA	Position Servo 1
0x5B	NA	Position Servo 2
0x5C	NA	Position Servo 3
0x5D	NA	Position Servo 4
0x5E	NA	Position Servo 5
0x5F	NA	Position Servo 6
0x60	NA	Position Servo 7
0x61	NA	Position Servo 8

Register Descriptions

Servo Position Register

The position (low byte/high byte) is a 16-bit number, which directly sets the output pulse width in uS. Setting the position to 1500 (1500uS or 1.5mS) will typically set servos to their center position. The range of pulse widths that are normally supported are from 500uS (0.5mS) to 2500uS (2.5mS). Take care though, as it is easy to make the servo run into internal stops, if you give it pulse widths at the upper or lower extremes.

The registers can also be read back. The position will be the current position of the servo during a speed-controlled movement, so you can track its progress towards the requested position. Setting the servo position to 0 will deactivate the servo output (rendering servo in floating condition).

Servo Speed Register

The speed register controls the speed at which the servo moves to its new position. The servo pulses are automatically refreshed every 24mS. On power up the Speed registers are set to the EEPROM stored value stored using STORE command.

If the Speed register is zero (0x00) then the servo is simply set to the requested position at highest permissible speed.

If the Speed register is set to something other than zero, then that value is added to the current position every 24mS until the target position is reached. e.g. If you wish to move from 1000 to 2000 and the Speed register

is set to 10, then it will take 2.4 seconds to reach the set position. The formula for the time it will take to make the move is: ((Target position-Start position)/Speed Reg)*24mS.

Control Register:

Control register allows you to change the I2C address of the device as well store and reset the start up conditions. By default, factory shipped NXTServo-v3 module has all registers set to 0x00. That is, all the servos are disabled on power up. However you can select the desired start up condition by storing it in the internal EEPROM.

Supported Commands:

CMD	Description
In	Store the initial speed and position properties of the servo motor 'n'. Current speed and position values of the nth servo is read from the servo speed register and servo position register and written to permanent memory.
S	Reset servo properties to factory default. (Initial Position of servos to 1500, and speed to 0)
H *	Halt Macro (This command re-initializes the macro environment)
R *	Resume macro Execution ((This command resumes macro where it was paused last, using the same environment)
Gx *	Go to EEPROM position x (This command re-initializes the macro environment)
EM *	Edit Macro
P *	Pause Macro (This command will pause the macro, and save the environment for subsequent resumption)

* Refer to NXTServo-Macro-Guide for further information about writing and running macros.

Quick Registers:

Quick register set allows quick change to the servo position (at a reduced resolution).

Although quick registers are not directly readable, the servo position can be read back using Servo Position Registers.

Quick registers are 1/10th the resolution of Servo Position Registers. The active value range of Quick registers is 50 to 250, where neutral position is at 150.

Software Revision Number:

Register 00 contains the software release number.

Current Release Number is 1.0 (0x0a).

Battery Voltage Register:

Voltage register contains value between 0 and 255 (with linear graduations), where value of 127 corresponds to 4700mV. Maximum voltage that can be reported is 9400 mV (corresponds to register value 255).

The register location is updated every 24mS, whether it is read or not.

I2C Address:

Factory shipped NXTServo module has address of 0xb0 on the I2C bus. This address can be changed to any thing above 0xa0. Control Register allows you to change the I2C address of the device.

Command Sequence to change the I2C address is:

```
0xa0 0xaa 0xa5 (new I2C address).
```

Note: Send these commands with no break/read operation in between. This new address is effective immediately. Please note down your address carefully for future reference.

Instructions for changing the I2C address can be found at:

NXT and EV3

http://www.mindsensors.com/blog/how-to/change-i2c-device-address.

PiStorms

http://www.mindsensors.com/blog/how-to/change-i2c-device-address-with-pistorms

NOTE

Any servo has physical limits to move from one position to the other. In general, an average RC servo takes about 120 milli-seconds for a 60 degree rotation. You may see jittery movement if your program is changing servo positions faster than physical limits of the servo.

Power On Position

When powered ON, the initial position of servo is set to neutral (i.e. 1500) and speed is set to 0.

Using Continuous Rotation Servos with NXTServo

Continuous rotation servo needs to be calibrated for neutral position (also known as stop position). Use following guide to understand and calibrate a CR servo:

http://www.mindsensors.com/pdfs/NXTServo-with-Continuous-Rotation-Servos.pdf

Following utility programs can be used in the calibration process:

http://www.mindsensors.com/index.php?controller=attachment&id_attachment=238

Current Consumption

NXTServo will draw about **3mA** current from NXT or EV3 for its internal circuit.

The current consumption of servos driven by NXTServo is not included in this value.

Macros on NXTServo

For advanced operations, you can write macros for repetitive tasks and store them onto NXTServo. NXTServo can then run those commands independently from the NXT brick.

For example, if you are making a hexapod, where the 6 servos are moving in unison with each other, the movements of these servos can be coded in a macro and executed on NXTServo.

Please refer to NXTServo Macro Guide for further information.

http://mindsensors.com/pdfs/NXTServo-Macro-Guide.pdf

How to change default neutral position stored in NXTServo

When the NXTServo is powered on, it sets the servos to their neutral positions stored in NXTServo's memory. The factory default value of neutral position is 1500 µS. It is possible to change these values so that when powered on, your servos will be set to your desired values.

Download the NXTServo EV3 Neutral Utility from following location:

http://www.mindsensors.com/index.php?controller=attachment&id_attachment=238

Open the program in the EV3 programming environment and download it to your EV3 or NXT brick.

Attach NXTServo to your EV3 or NXT on port 1, attach power to NXTServo.

Attach a Servo to NXTServo.

Run program Set Neutral.

Use the arrleft and reight buttons on the eV3 or NXT to select the NXTServo pin number to which you have attached the servo motor and push the center button.

Using the arrow keys on your EV3 or NXT, change the position of the servo motor to that which you desire to become the new neutral position and push the center button.

The NXTServo will now move the servo motor to that position every time it is powered.