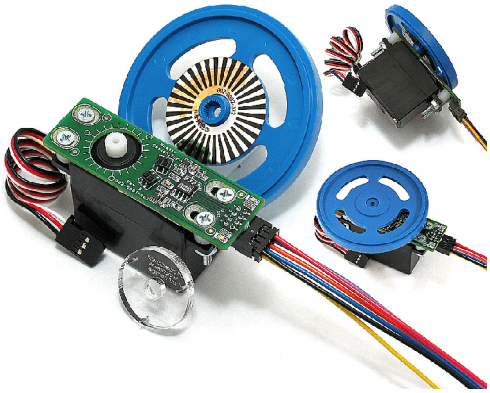
NUBOTICS

WW-01 WheelWatcher

Product Manual

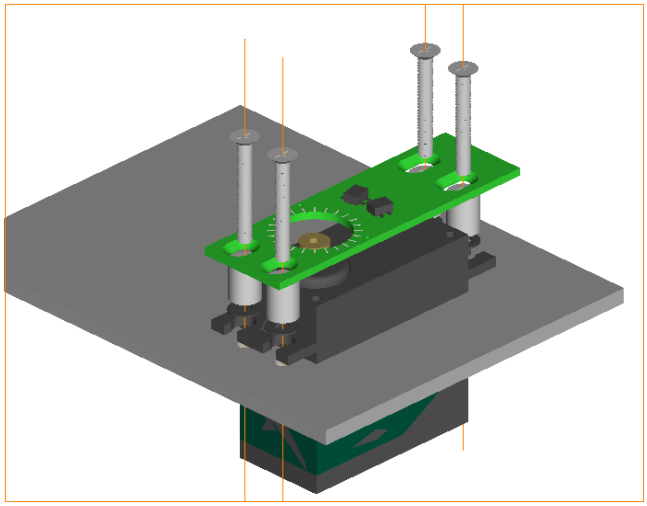


Incremental quadrature encoder system for standard RC servos

1.4 6/29/2005

WheelWatcher Features

- easy installation
- simple interface
- preprinted 32 stripe self-adhesive reflective codewheel
- hardware quadrature decode with 4x multiply for 128 clocks per rotation
- raw quadrature also provided
- compatible with standard-size servos from Hitec, Futaba, and GWS
- designed for use with standard injection-molded robot wheels – any color
- code examples for common robot controllers available



Description

The WW-01 WheelWatcher incremental encoder system enables robot builders to quickly add closed-loop control to their robots. Both standard ChA/ChB raw quadrature outputs (90° phase shift between channels), as well as decoded Clock and Direction signals are provided. The clock signal produces a 25µs pulse at each transition of ChA or ChB, providing a 4x increase in resolution compared to the number of stripes – 128 clocks per servo rotation – while the direction signal indicates the decoded direction of rotation, making it very easy to add to any microcontroller.

WW-01 Parts List

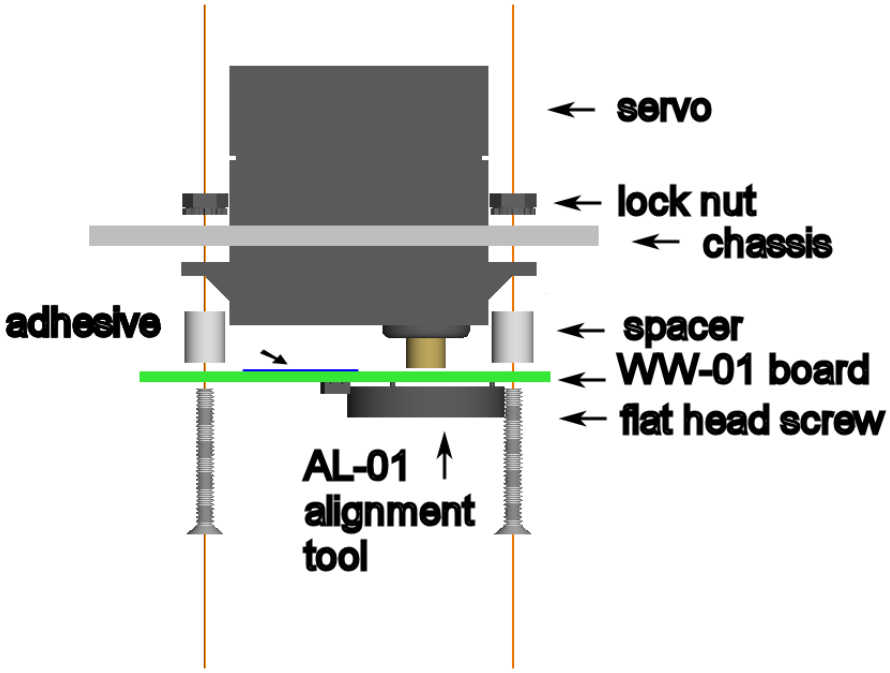
1. adhesive-backed printed circuit board, preassembled
2. alignment tool
3. self-adhesive codewheel
4. 6" four lead color-coded cable
5. two extra wires provided for use with raw-quadrature output
6. mounting hardware

Installation

Proper spacing between the photodetectors and codewheel, as well as alignment of the photodetectors to the codewheel, is essential to proper operation.

Proper spacing depends on the board being held tightly against the top of the servo using adhesive backing as well as mounting hardware, and also depends on the use of standard injection molded wheels with standard size servos. See the *Nonstandard Mountings* section for information about using the WW-01 with other attachments.

Accurate alignment of the photodetectors to the codewheel depends on the use of the provided alignment tool to center the board's axle hole with the servo output shaft, and accurate placement of the sticker on the back of the wheel. It also depends on reliable maintenance of that alignment by means of the adhesive backing and mounting hardware.



Installation Step by Step

PLEASE SEE TABLE 1 FOR SUGGESTED SPACERS. An animation of the installation procedure is available on the Nubotics website at http://www.nubotics.com/products/ww01/anim.

1. if the servo is already mounted on the robot, remove it[1]
2. remove the servo manufacturer's label on the top side of the servo, as the mounting adhesive on the back of the WW-01 will work better when joining the board to the servo case directly, rather than to their label
3. place the servo on a solid surface (such as a table) with the output spline (the output shaft) facing up
4. line up the pins on the alignment tool with the matching holes on the board on the component (top) side
5. squeeze the board and tool together until they touch and the tool and board are parallel
6. remove the paper backing from the board's mounting adhesive
7. lower the board and tool over the servo output spline, with the spline going into the hole in the center of the alignment tool, making sure the board is lined up with the sides of the servo case
8. press the board the rest of the way down onto the top of the servo; as you do so, it will slide down the pins of the tool
9. press down on the board with your finger tips to ensure proper adhesion (never use a sharp tool for pressing, as this may damage the board)
10. there are two different lengths of spacers provided to accommodate variation between various servo models; pick the one size that fits your servos (see Table 1); be sure to use all of the same length spacers; select a spacer length that results in the board being held tightly against the servo once the screws are tightened, without a gap under the adhesive; do not tighten the screws so much that the board bends – just enough to keep the board in alignment
11. leave the alignment tool engaged while you mount the servo and WW-01 to the chassis, using the provided spacers, flat-head 4-40 machine screws[2], and 4-40 k-lock nuts
12. wire up the cable to your microcontroller; black is ground, red is Vcc (regulated +5v is recommended), and the rest are ChA, ChB, Dir, and Clk; see the Connector Pinout for assembly and Interfacing Examples for details
13. always remove the alignment tool before powering on the robot
14. place the codewheel sticker on the back of the wheel, being sure to center it with the hub; the back of the wheel is the side into which the servo motor shaft is inserted; the raised hub fits inside the sticker's center hole

Notes:

[1] if your servos are mounted to the chassis with servo tape or double stick foam tape, you might be able to leave the servos mounted

[2] You may substitute your own screws if you need shorter or longer ones; just be sure to use flat-head, 100° head-angle 4-40 screws to ensure the tops of the screws near the connector do not rub on the rim of the wheel.

Good and Bad Board Mounting Examples

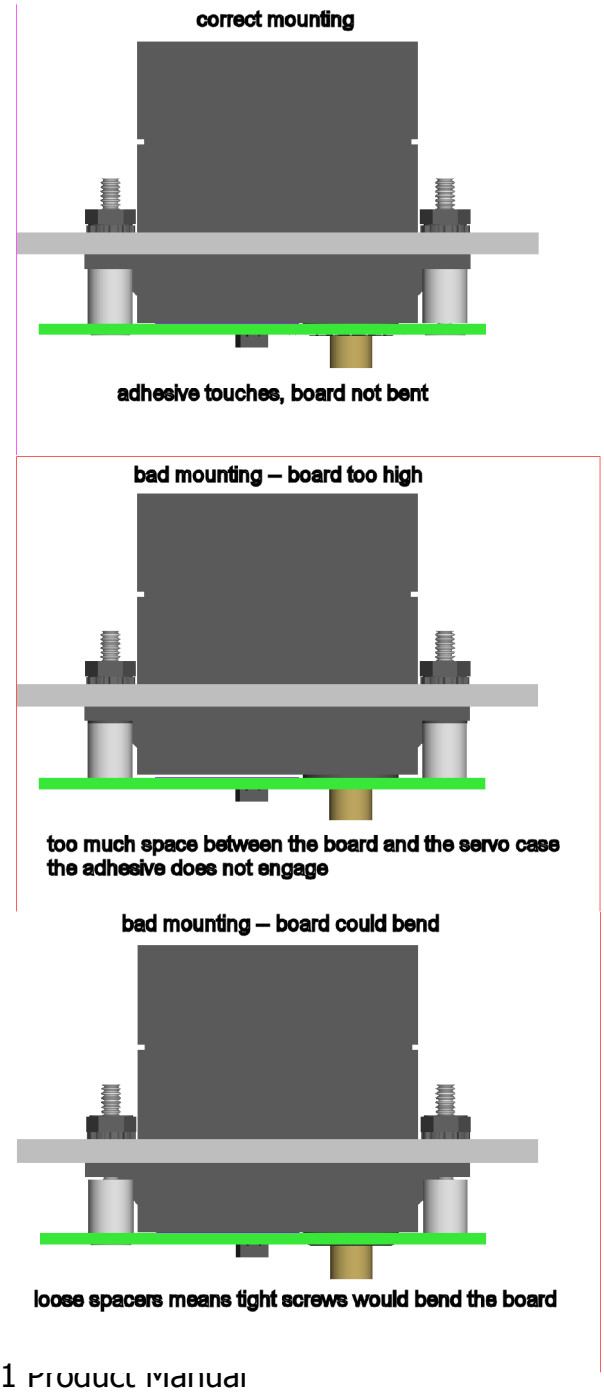


Table 1: Spacers For Various Servos

This table is for kits with Rev C boards only; see the Rev B manual for older kits.

Manufacturer	Model	Short Spacer (.278")	Tall Spacer (.312")
Futaba	S3001	✓	
	S3003	✓	
	S3004	✓	
	S9001	✓	
GWS	S03N		✓
	S03T		✓
	S03TXF		✓
	S06	✓	
Hitec	HS300	✓	
	HS322HD*	✓	
	HS425BB	✓	
	HS475HB	✓	
	HS605BB	✓	
Hobbico	CS-65	✓	
	CS-67	✓	

Note: if your specific servo is not listed, try a spacer from a similar model by the same manufacturer, or see step 10 of the installation instructions for how to determine the correct spacer.

*The Hitec HS322HD may need thin washers (~.014" thick, .25" diameter, #4 hole) for a tight fit without bending the board. If you need the washers, contact Nubotics support at info@nubotics.com.

Connector Pinout – Rev C Boards

Available July 2005 and later

Decoded Quadrature Grouping:

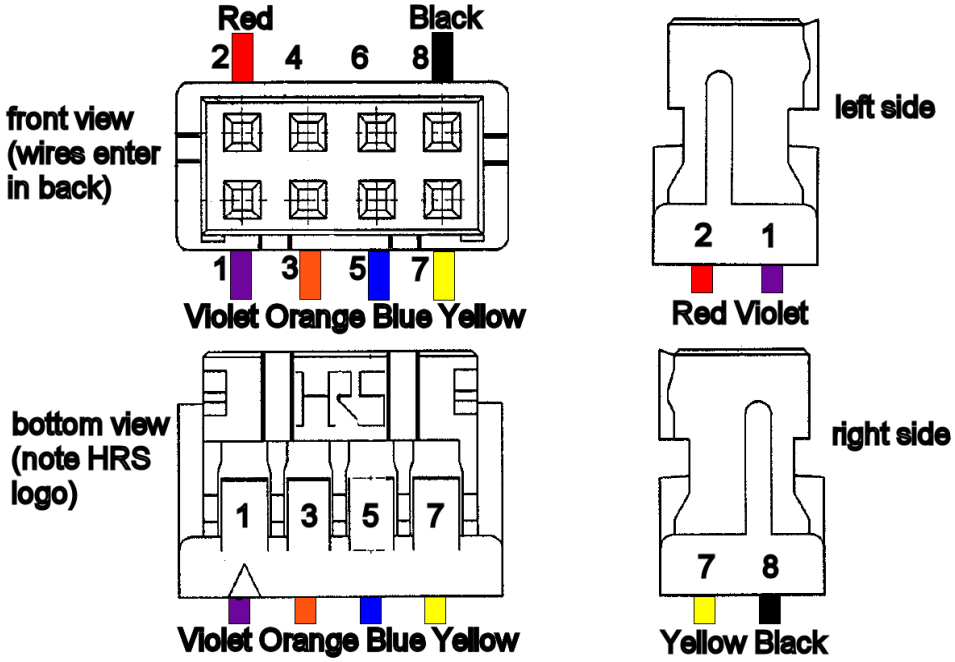
1. Clk: violet
2. Vcc: red
3. Dir: orange
4. Gnd (connects to pin 8)

These are the left four pins in the diagram.

Standard Quadrature Grouping:

5. ChB: blue
6. Vcc (connects to pin 2)
7. ChA: yellow
8. Gnd: black

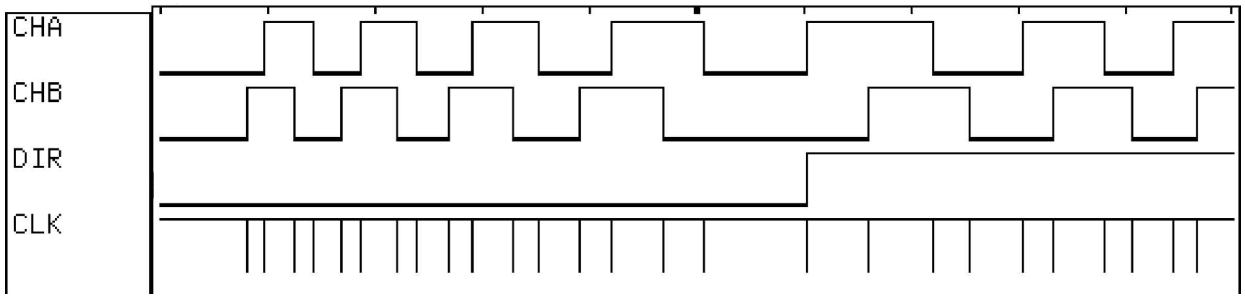
These are the right four pins in the diagram.



NOTE: the triangle mark indicates pin 1 on both the header and the connector. The ealier Rev B boards have a pin numbering on the silkscreen which does not match the pin numbering stamped on the Hirose connector, but are otherwise cable-compatible (a cable that works with a Rev B board will also work with a Rev C board). Rev C boards have an updated silkscreen that changes the numbering of the pins so that pin 1 on the board corresponds with the triangle mark and pin numbering on the Hirose 8 pin connector and header, in order to eliminate confusion. This pin numbering scheme is now the same as the WheelWatcher WW-02 Incremental Encoder for Gearhead Motors.

Timing Diagram

The following diagram illustrates the behavior of the ChA, ChB, Dir, and Clk signals as the wheel slows down and changes direction, then speeds back up.



Specifications (PRELIMINARY)

• Supply Voltage (Vcc)	+4.5v to +5.5v
• Supply Current (Icc)	30mA MAX (TBD)
• DC Output Voltage	0v to Vcc
• DC Output Current (pins 1, 3)	+/- 50mA
• DC Output Current (pins 5, 7)	+1.0mA (Vout 4.5v), -1.75mA (Vout 0.4v)
• Clock pulse width	25 µs
• Radial misalignment	TBD
• Tangential misalignment	TBD
• Angular misalignment	TBD
• Codewheel tilt	TBD
• Phase error	TBD
• detector top to codewheel	0.8mm (0.031") - 1.1mm (0.043") NOMINAL MIN 0.5mm (0.02") (TBD) MAX 2mm (0.08") (TBD)

Nonstandard Mountings

It is possible to use the WheelWatcher system with other commercial wheels, custom wheels, or even to give feedback on the rate of rotation or relative position of a leg or arm joint, by using one of the optional Codewheel Spacers (sold separately).

The CS-040-01 Universal Codewheel Spacer provides 36 laser drilled guide holes that line up with useful hole locations on many popular servo horns. Many commercial wheels use servo control horns and their existing holes for locating mounting screws; simply drill out the correct holes on the CS-040-01 to allow the screw heads to hide underneath the Codewheel Sticker. It is important that the screw heads recess below the top surface of the Codewheel Spacer so that the Codewheel Sticker is flat, not "lumpy."

The CS-040-02 Codewheel Spacer for Wheels provides large, predrilled holes designed specifically for use with wheels made by Rogue Robotics and Budget Robotics. See Application Note AN01 for details.

Interfacing Examples

Please visit www.nubotics.com to view and download example code for Atmel AVR Basic and C, Microchip PIC Basic and C, Kronos Robotics DIOS, Motorola 68hc11 Interactive C and Ridgesoft RoboJDE Java, Parallax Basic Stamp, and Savage Innovations OOPIC controllers.

Raw Quadrature: ChA / ChB Only

The ChA/ChB Raw Quadrature signals can be used with motor controllers that accept industrial-style incremental encoder signals, such as the Acroname BrainStem Moto, Savage Innovations OOPIC, or Solutions Cubed MiniPID.

ChA and ChB are 50% duty cycle, 90° out of phase signals, created by having two Omron EE-SY125 (or equivalent) photodetector packages spaced at a very specific angle with respect to each other, at a specific radius from the center of the axis of rotation, and expect to be used with a 32 stripe codewheel with a 50% silver/50% black radial stripe pattern.

Decoded Quadrature: Clk / Dir Only

The decoded outputs Clock and Dir are useful when interfacing the WheelWatcher to the Parallax BasicStamp II, Microchip Technologies' PIC midrange family, or other microcontrollers with hardware counter inputs or external interrupt pins. The clock line pulses low for 25µs upon each transition of either ChA or ChB. The direction line is high when the wheel rotates one way, and low when it rotates the opposite way.

For example, on a Basic Stamp II, the PulsIn command operating on a Clock signal results in a direct measurement of the period of rotation of a wheel. Use that value to calculate a new servo pulse period to maintain a desired velocity.

On a PIC 16F877, tie the Clock signals to T0CKI and T1CKI, then tie the Direction signals to B4 and B5, which can issue an interrupt on change. Whenever B4 or B5 change, read the timer value, add or subtract it from a running position value based on the last direction value (since the counts in the counter were from previous motion before the direction changed),

save the new direction value, then reset the hardware counter. Accurate, relatively high-bandwidth measurements of wheel velocity can be taken by using Timer2 to measure the time between pulses of the Clock signals. This allows you to tightly control wheel velocity and acceleration despite the low sample rate (up to 128 clock pulses per second for a 60 RPM wheel speed).

Hybrid Signaling: ChA / Dir

The Acroname Brainstem cannot detect the 25µs pulses, but has no problem interfacing with ChA or ChB and the decoded Dir signal. A TEA program can count the edges of ChA, resulting in 64 counts per rotation, and, based on the level of Dir, decide whether to increment or decrement a position counter. The pulse width timing capability might be used to directly measure wheel velocity by timing the width of ChA. See www.acroname.com for examples.

WheelWatcher Frequently Asked Questions

* Where can I buy it?
www.acroname.com

* Do I need to use the alignment tool?
Yes, it is highly recommended. If the board's sensors are not aligned well with the codewheel sticker, the timing of the various signals will be strongly affected, to the point that you may end up not getting 128 clock pulses per rotation. Further, the time between each clock pulse will not be consistent, but will instead vary from pulse to pulse, usually with a pattern that repeats every 4 clocks. Severe misalignment may result in ChA and ChB not always being in phase with each other, resulting in inconsistent counts.

* What happens if the codewheel sticker is not centered on the wheel?
This will result in a wobbling pattern to the signals that repeats for each wheel rotation. It would be hard to correct for this in firmware, so go slowly and take care when mounting the sticker. This won't affect you much if you are only using the WheelWatcher for odometry (distance measurement), but will affect you if you are using it for velocity control.

* Do I need to use the adhesive on the back of the board? I might want to remove the WheelWatcher later.
The adhesive helps prevent the board from falling into misalignment if the mounting screws come loose. We chose a type of adhesive that sticks well to many types of plastic, but not so well that you cannot remove the board if you need to.

* What do the LEDs mean?
The RED LED (DL1) is connected to the ChA signal, so it pulses on and off 32 times per wheel rotation to show that the WheelWatcher is working.
The GREEN LED (DL2) is connected to the DIR signal, so it turns on when the wheel rotates clockwise (CW) and off when it rotates counter clock wise (CCW).

* Why am I not getting good counts from the encoders?
Check the behavior of the LEDs while your servos are turning. Do this either by turning the wheels slowly by hand (rough turning can strip the gears in the servos, so be gentle!) with the power on to the WW-01s but off to the servos, or by using your own test program. The green LEDs should be solid on or off, and should only change state when the direction of wheel rotation changes. The red LEDs should blink on and off, once per stripe -- 32 times on and 32 off for each wheel rotation.

If the green LED is flashing when it shouldn't be, or if the red LED is not pulsing 32 times per rotation, make sure your wheels are on tight. If the wheels are too high above the WW-01 PCBs, there won't be enough light reflected to the sensors on the boards. It is best to use the screws provided with your servos to hold the wheels in tight.

If you are still having problems, check the alignment of the PCB with the AL-02 alignment tool with the wheel temporarily removed. Also, check that the codewheel stickers are flat against the wheel, without bubbles under them, and that they are clean.

* How can I clean the codewheel stickers?
We recommend only gentle cleaning with a tissue or Q-Tip moistened with clean warm water. Do not use isopropyl (rubbing) alcohol or other cleaners, as they can cloud the silver areas and render them less

reflective. They could also damage the adhesive.

* What mounting hardware is provided?

We provide four 4-40 x 7/8" flat head machine screws, with a 100 degree head angle so that the screw heads will not rub on the wheel. We also provide four 4-40 k-lock nuts and four .25" diameter x .31" length #4 spacers. Kits containing the Rev B boards also included four .25" x .25" #4 spacers, four .014" thick washers, and four .030" thick washers. Kits containing the Rev C boards (released July 2005) do not include the .25" long spacers nor the washers. Instead, we have had custom spacers – .25" x .278" #4 -- manufactured that fit perfectly on non-GWS servos. This eliminates the need for the washers, greatly simplifying installation for most customers.

* Can I run the WheelWatcher from my +6v (or +9v, ...) battery pack?

No. The ICs on the board require Vcc to be between +4.5v and +5.5v. We recommend the use of a regulated +5v supply.

* My robot uses DC motors and hand made wheels. Can I use the WheelWatcher with it?

Maybe. You will need to mount the WW-01 such that it is aligned with your motor's shaft, and place the codewheel sticker on a flat surface (your wheel or something attached to the motor shaft) so that it is parallel to the surface of the WW-01 PCB. The distance between the top of the 2 sensors on the WW-01 and the codewheel sticker must be close to 1.1 mm (0.43") for it to function.

Beyond that, good luck. We can't support nonstandard motors, wheels, and mounting schemes -- the WW-01 is designed specifically for standard injection molded wheels and standard size RC servos. Let us know how it works for you, though.

* How can I tell how far my robot has travelled?

This is called odometry. Simply count up ChA and/or ChB rising, falling or both edges, or count CLK pulses, as the robot moves. See the next question for clues as to how to convert these counts to real world distance units. Note that wheel slippage or uneven terrain will result in inaccurate distance measurements, no matter what encoder system you use; this is the bane of dead reckoning..

* How can I measure the velocity of the wheel?

Two methods are commonly employed. The easiest is to count the number of clock pulses N (or changes to ChA and/or ChB) over a certain interval of time T. Velocity V = N / T in terms of counts per unit time. If T is too small you sometimes won't get any counts, even if the wheel is still turning.

The standard injection molded wheel used with the WheelWatcher is 2.75" in diameter (for O-ring wheels). Pi * D gives a circumference of 8.64". If you are using the CLK signal, the counts per rotation C = 128. That gives C / (PI * D) = 128/8.64 = 14.8 clocks per inch of linear travel.

So, if you measured N using CLK and T in seconds, then V = (N / T) * pi * D / C in terms of inches per second.

The limitation with the first method is that you can only update your servo control pulse width or PWM value every T seconds. For a normal servo, the highest rotation speed is usually around 60 RPM, which is one rotation per second, or 128 CLK pulses per second; at slower speeds, you will get fewer pulses per second. Your update rate (or control loop bandwidth, in control theory language) will be slow.

Another method is to measure the time, using your microcontroller's hardware timers, between each CLK pulse (or ChA or ChB pulse), then take the inverse (1/T) to get counts per unit time. At 60RPM, you

could update your servo control pulse value or PWM value every 8 ms instead of every second, and have much better resolution too.

However, regardless of how well aligned the board is and well centered the sticker is, manufacturing error and alignment error will still result in some time variation from clock to clock, so we recommend that your firmware calculates a running average of the time between the 4 most recent pulses when using this method.

* My microcontroller is too slow to measure the CLK signals. What do I do?

For slow chips like the Basic Stamp, use the raw quadrature signals instead, such as ChA, ignoring ChB and CLK. If you need your program to know the direction of wheel rotation, you can easily read the DIR signals. While you will get less precision in positioning your robot, you will still get approximately 1/4" resolution.

* How can I control the velocity of the wheel?

This is an area of engineering known as control systems theory. There are many methods for using feedback to control velocity of a motor.

In the simplest method, one measures the actual velocity of the wheel, calculates an error signal by subtracting the actual velocity from the commanded or goal velocity, multiplies that by a constant, then feeds that value to the motor. Thus, if the wheel is spinning too slowly, the motor is told to speed up, and vice versa. What I just described is known as a P loop -- proportional control loop -- since the only error term is proportional to the difference in velocity.

One common technique that works better for varying terrain is called a PID loop, which stands for Proportional Integral Differential. Much has been written about this technique, and Google is your friend.

See the Example Code area on www.nubotics.com for examples of P and PI loops using the WheelWatcher for various robot platforms.

* Do the WheelWatcher stickers work with the Parallax BoEBot wheels?

Newer BoEBots use an injection molded wheel with a larger-than-normal hub. As a result, the codewheel sticker provided with the WheelWatcher does not fit. We are evaluating whether to produce a version of the sticker specifically for the BoEBot. Until that is available, you can replace the wheels with "Tigerbotics"-style wheels available from Acroname or other vendors.

* Are the two Vcc and Gnd pins internally wired together on the board? Can I use either Vcc pin and either Gnd pin?

Yes, pins 2 and 6 on the connector are Vcc, and are both connected on the board. Ground pins 4 and 8 are likewise wired to each other. We designed the pinout so that we could, in the future, offer Quadrature-Only or Decoded-Only versions of the board, by omitting or adding certain parts in the circuit during PCB assembly, and by soldering in a 4 pin connector in either the Standard Quadrature Grouping position (pins 5-8) or the Decoded Quadrature Grouping position (pins 1-4). So, feel free to use either pin 2 or 6 for Vcc, and pin 4 or 8 for Ground. NOTE: these pin numbers are for Rev C of the board.

Document History

Rev 1.3 - initial public release
Rev 1.31 - adds note clarifying pin 7 = triangle mark on connector; addition of FAQ
Rev 1.4 - updated to conform to Rev C circuit board (pin 1 = triangle mark), and custom spacer that eliminates washers

For more information visit: www.nubotics.com
Produced by Noetic Design, Inc., 25 NW 23rd PL, STE 6 PMB 181, Portland OR 97210