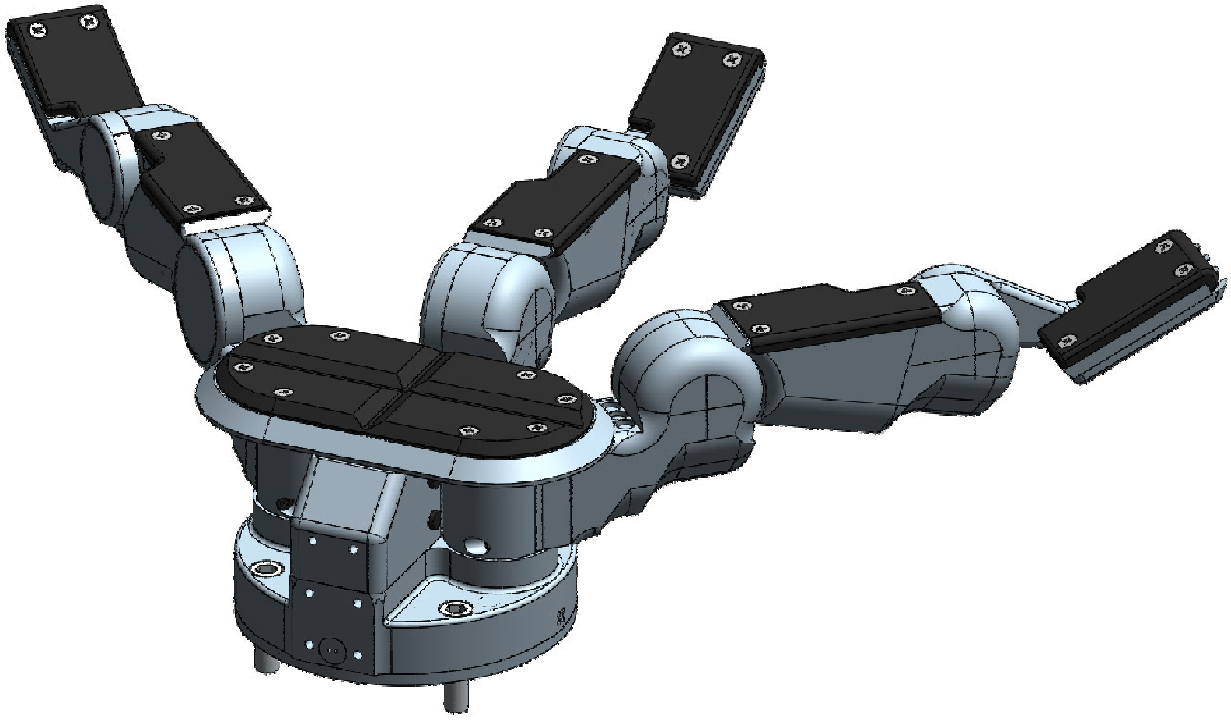
Wraptor™

BH8-610 Series User Manual



Barrett Technology Inc.

Document: D2000
Version: AB.00

Table of Contents

1 System Description

1.1 What's new in the 610-Series Wraptor™

We upgraded the motor feedback sensor from an optical sensor to a more robust magnetic one.

1.2 Introduction

Thank you for choosing the Wraptor™. The Wraptor™ is designed to overcome the inflexibility of conventional industrial grippers with DSP-enabled dexterity while maintaining durability, compactness, and ease of use. The Wraptor™ is a multi-fingered grasper with the dexterity to secure target objects of different sizes, shapes, and orientations. Rather than rely on gripper friction from pinching or permanent gripper-jaw-shape customization, the Wraptor™ gently envelops targets, securely locking its joints until commanded to release the target.

System integration with any robotic arm is fast and simple. Even with its low, 7-kg, weight and compact form, it is totally self-contained. The Wraptor™ uses industry-standard Ethernet communications, which is the common denominator of network communications, for guaranteed universal compatibility. Eight (8) on-board DSPs combined with Barrett's open Grasper Control Language (GCL) endow the Wraptor™ with millisecond responsiveness.

The compactness and low weight of the Wraptor™ assures that the enhanced dexterity does not compromise arm payload. Its low mass and short base-to-grasp-center distance minimize joint loading on the host robot and reduce extraneous arm movements during object reorientation. The custom control-electronics package is contained entirely within the Wraptor™, reducing electrical wiring to a single cable carrying only Ethernet communications and DC power.

We hope that you enjoy the versatility and functionality of the Wraptor™. Please never hesitate to give feedback to Barrett's engineers and to ask Barrett's engineers for advice. You may contact Barrett's engineers at <service@barrett.com>, or US+617-252-9000, or at <http://www.barrett.com/robot/>.

1.3 About the Wraptor™

The Wraptor™ has three fingers identified as F1, F2 and F3. Two of the fingers, F1 & F2, rotate synchronously and symmetrically about the base joint in a spreading action. The "spread" motion around the palm allows "on-the-fly" grasp reconfiguration to adapt to varying target object sizes, shapes, and orientations. Aside from the spread motion, each of the three fingers on the Wraptor™ feature two joints driven by separate DC brushless servo motors. When the inner joint contacts an object, the outer joint continues to close around it, making a secure grasp. Using the fingers together allows the Wraptor™ to "grasp" a wide variety of objects securely. The multi-jointed fingers, combined with the spread function, make object grasping nearly target-independent.

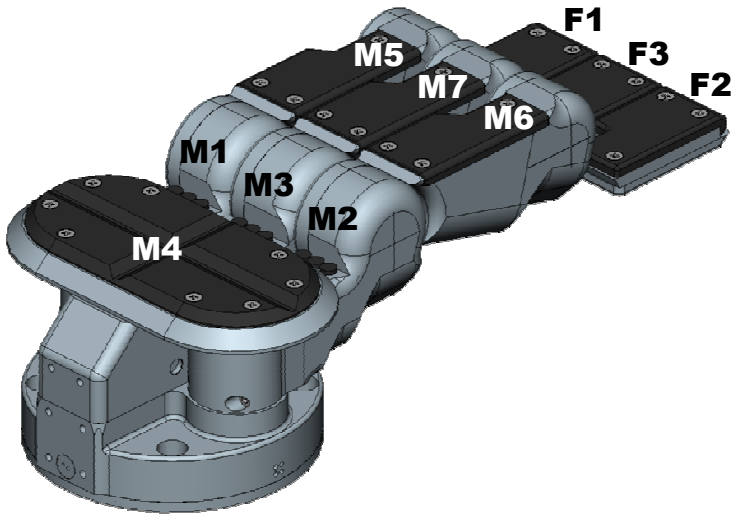


Figure 1: Isometric view of Wraptor showing Finger and Motor numbers

1.4 Technical Specifications

1.4.1 Overview

Dimensions:	132h x 658w x 192d (mm)
Mass:	6.9kg
Voltage requirements:	48V±2%
Current requirements:	5A Min, 15A Typ, 50A Max
Load limits:	50kg / finger
Operating temperature limits:	0 to 70 degrees C (internal)
Storage temperature limits:	-25 to 95 degrees C (non-condensing)
Environmentally sealed:	IP-65

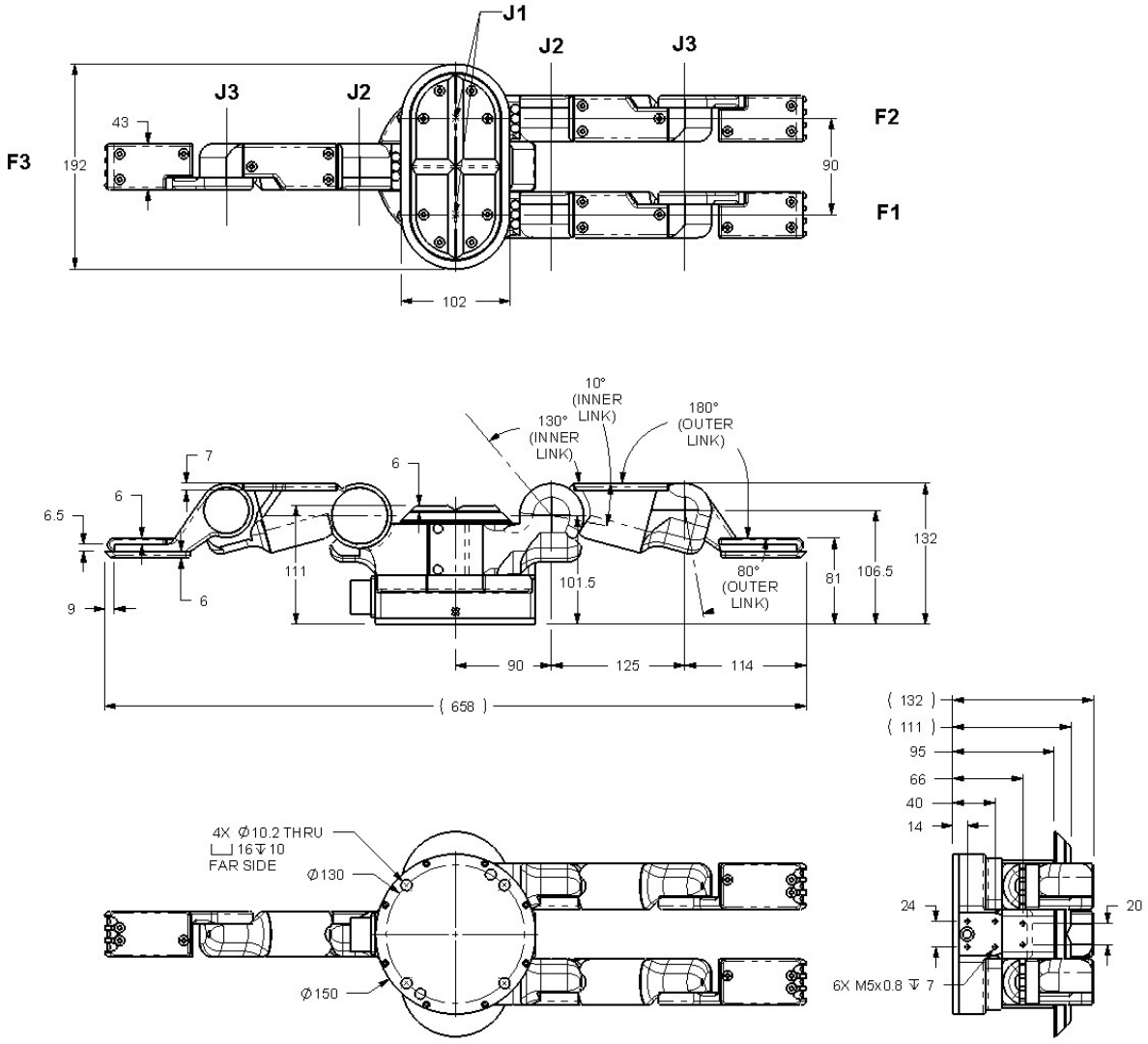


Figure 2: Mechanical schematic of Wraptor

Table 1 - Joint Ranges

		Inner Link	Outer Link	Spread
GCL Counts	**Min**	-5,500	-35,000	0
	Max	72,000	79,000	37,000
Degrees	**Min**	-10	-80	0
	Max	130	180	180

1.4.2 Conversion ratios

Joint rotation is measured in encoder counts at the motor shaft. To convert between the motor angle and the joint angle, use the following table:

Table 2 – Conversion Ratios

Ratio	Value
GCL Encoder Count : Motor Revolution	4096:1
Motor Revolution : Inner Link Revolution	49:1
Motor Revolution : Outer Link Revolution	39:1
Motor Revolution : Spread Revolution	18.28:1

Example Calculation- Determine the GCL encoder count value needed to move an inner link by 90º:

$$90^{\circ}\left(\frac{1\,link\ \mathrm{Re}\,v}{360^{\circ}}\right)\left(\frac{49\,motor\ \mathrm{Re}\,v}{1\,link\ \mathrm{Re}\,v}\right)\left(\frac{4096\,EncoderCounts}{1\,motor\ \mathrm{Re}\,v}\right)=50176\ GCL\ Counts$$

1.4.3 Brushless Motors

The Wraptor™ utilizes advanced DC brushless servo motors. Because the motors have no brushes, and thus less inherent friction, they achieve a better torque/mass ratio than typical brushed servos. There is also no need to replace worn brushes after the motors have been in service over a period of time. The following table shows Wraptor™ motor properties.

Table 3 - Wraptor™ Motor Properties

Parameter	Value
Number of Phases	3
Number of Magnetic Poles	6
Rotor Magnet Material	Highest-Grade Neodymium Rare-Earth
Commutation	Sinusoidal Space-Vector Brushless PWM
Peak Torque	0.65 N-m
Steady State Torque	0.10 N-m
Torque Constant	0.065 N-m/amp
Motor Constant	0.042 N-m-$W^{1/2}$
Position Feedback for Spread motor	3 Digital Hall Sensors X 6 Motor Magnet Poles: 18 states per motor revolution that are interpolated into 4,096 positions/motor rev. so true spread joint resolution = 360/(18*18.28) = 1.1 degrees.
Position Feedback for inner and outer link motors	Absolute Magnetic Encoder Sensor on Motor Shaft: Absolute 4,096 positions/motor rev. via magnetic encoder, so inner and outer link resolutions are 0.0018 & 0.0023 degrees respectively.

1.5 Theory of Operation

The host PC connects and communicates to the Wraptor™ using Ethernet. A device server inside the Wraptor™ converts command packets between Ethernet and RS-232 serial (at 9600 bps) to communicate with the primary controller in the Wraptor™ . This controller compiles the serial packet into a valid CAN packet and places it on the internal CAN (Controller Area Network) bus to be delivered to the motor controllers (Pucks).

When the primary controller receives a CAN packet from a Puck which is addressed to the host PC, it sends the data to the device server via RS-232 serial. The device server then routes the data back to the host PC via Ethernet.

1.6 Control Software

The BH8-610 Series System control software consists of:
1. Firmware (*.tek software), and
2. Example programs.

Included with the software in electronic form are:
1. BH8-610 Series User Manual (this manual)

The Wraptor™ has firmware that resides on the control electronics inside the palm. Depending on the configuration purchased, you can control the Wraptor™ in one of three ways:
1. RS-232 Serial Communication: Simple, ASCII-based commands are sent over the serial port and interpreted by the Wraptor™.
2. Ethernet Communication: Simple, ASCII-based commands are sent over an Ethernet network and interpreted by the Wraptor™.
3. CAN Communication: High-speed, pre-compiled commands are sent directly to the motors of the Wraptor™, eliminating the need for an interpreter and allowing absolute control of each motor.

1.7 C-Function Library

The Wraptor™ C-Function Library is a helpful tool for programming the Wraptor™ using the C language on IBM-compatible PC's without having to manage the issues of communication and timing of multi-axis motion control. The library contains easy-to-use functions that streamline the development of custom motion control routines by the end user. All of the functions are available when the library is linked to the program.

The C-Function Library is written in C and compiled for Windows 95 or higher. The library uses a multithreaded mechanism for communications, allowing commands to be issued and responses received simultaneously. The library also manages all input and output buffers and makes it easy to manipulate the Wraptor™ from a custom control application.

1.8 Control Software/Firmware Upgrades

Barrett Technology makes software and firmware upgrades periodically. Upgrades are available for purchase or free of charge for customers of Barrett's subscription service. Refer to Barrett's enclosed Warranty and Subscription Service Policy for more information.

2 Safety and Cautions

PLEASE READ THIS SECTION IN ITS ENTIRETY BEFORE USING YOUR WRAPTOR™.

Following these safety instructions will help prevent user injury and equipment damage.

2.1 Human Safety

- The Wraptor™ has been designed to apply human-scale maximum active forces of several kilograms even though the Wraptor™ is designed to lift 50 kg once the fingers have secured their grasp. HOWEVER, the safety of the Wraptor™ is limited by the system safety, including the arm, which may be capable of tons of force. If the arm that transports the Wraptor™ is able to move, then the end user must be aware that the Wraptor™ is capable of (and will) transmit the arm's end forces directly to people and delicate equipment. If an articulated robotic arm is rated for 50 kg, for example, it is likely to be able to produce many tons of force in many leveraged configurations or when it is moving fast upon an impact.
- Beware of situations when the Wraptor™ is stationary and secured on the end of a high-payload robot, but there is other active moving equipment nearby. The Wraptor™ generally will not accommodate and reduce the force of any collision, if it occurs.
- Do not place any part of your body or delicate objects within the grasp of the Wraptor™ without first verifying control of the unit and confirming appropriate force levels.
- Beware that the sharp finger-nail like claws attached nominally to the outer links can cause pain and possibly cut bare human skin depending on details of the grasp.

2.2 Wraptor Safety

- Do not connect or disconnect any electrical cables while the Power Supply is turned on. Failure to follow this instruction could impart irreparable damage to the onboard electronics.
- Ensure that input power to the Positive- and Negative-voltage terminals are never reversed.
- Never allow power to be connected to the communication pins at the input connector of the Wraptor
- Do not exceed the load limit of the fingers, 50kg per finger. Consider all loading situations including accelerated loads, cantilever loads from long objects, robot collisions, active loads, etc. (See also, next paragraph.)
- Similar to the concerns about Human safety when the Wraptor™ is mounted on a robotic arm, one must recognize that, for example, a 50-kg lift robotic arm is easily capable of producing several tons (!!!) of force even when moving slowly (in many configurations with high mechanical leverage) or when moving quickly just before a collision (regardless of configuration). Since forces cannot be seen by eye, any contact with rigid surfaces or even with the base or other links of the robotic arm can easily exceed tons of force without operator awareness. It is advisable to be very cautious when running the full system to minimize these collisions or to ensure that a force/torque sensor or collision-break-away mechanism is installed between the tool-plate of the robot and the base of the Wraptor™.
- Do not allow the Wraptor™ to be exposed to corrosive liquids that may cause damage to the body or cable of the unit.
- The operating temperature of the Wraptor™ is monitored continuously and automatically so that no Puck can approach OTEMP, which is set nominally to 82 C. If the temperature reaches within 16C of OTEMP, then the max command torque (motor current) will be temporarily limited to prevent damage. Under normal conditions, the Wraptor™ operates between 35 and 60C. The Wraptor™ was designed with non-backdrivable finger joints to take advantage of the motors' peak operating performance in short bursts. The spread, however, is backdrivable to aid in target-independent grasping and requires constant motor current to actively hold position. Idling the spread motor (thus activating the spread brake), when possible, will help keep the temperature lower.
- Do not immerse the Wraptor™ in liquid.
- Do not expose the Wraptor™ to corrosive liquids.
- Do not expose the Wraptor™ to mud or slurries.
- Ensure that the electrical Wraptor™ cable and connector cannot be damaged by collision, cutting, pulling, or twisting.

3 Initial Setup and Walk-through

1) Mount the Wraptor™ securely to a robot arm or test stand.
2) Supply the Wraptor™ with 48V using the included power supply.
3) Plug the Wraptor™ into an Ethernet network.
4) Launch the "DS Manager" utility, select "DS Settings".
5) Enter a free IP address on your Ethernet subnet. Save your changes and exit the DS Manager.
6) Launch the TeraTerm application.
7) Type "HI" to perform a "Hand Initialize" on the Wraptor™.
    a. Note: All commands must be followed by pressing <Enter>.
    b. Note: All commands are case-insensitive and all white space is optional.
8) Get the initial temperature of the inner joint motors: Type "123 GET TEMP"
    a. Note: "123 GET TEMP" is read as "Motors **1, 2,** and **3, Get** your **TEMP**erature"
    b. Note: The Wraptor™ should return with its inner joint temperatures, in degrees C.
9) Move the fingers to position 20000: "123 M 20000"
    a. Note: "123 M 20000" is read as "Motors **1**, **2**, and **3**, **M**ove to position **20000**"
10) Move the spread (motor 4) to its mid-position: Type "4 M 18500"
11) Close finger 1: Type "1 C"
12) Open the whole hand: Type "O" (that is the letter 'O')
13) Get the final temperature of the inner joint motors: Type "123 GET TEMP"
14) Save the present positions for shutting down: Type "SAVE"
15) Remove power from the Wraptor™.

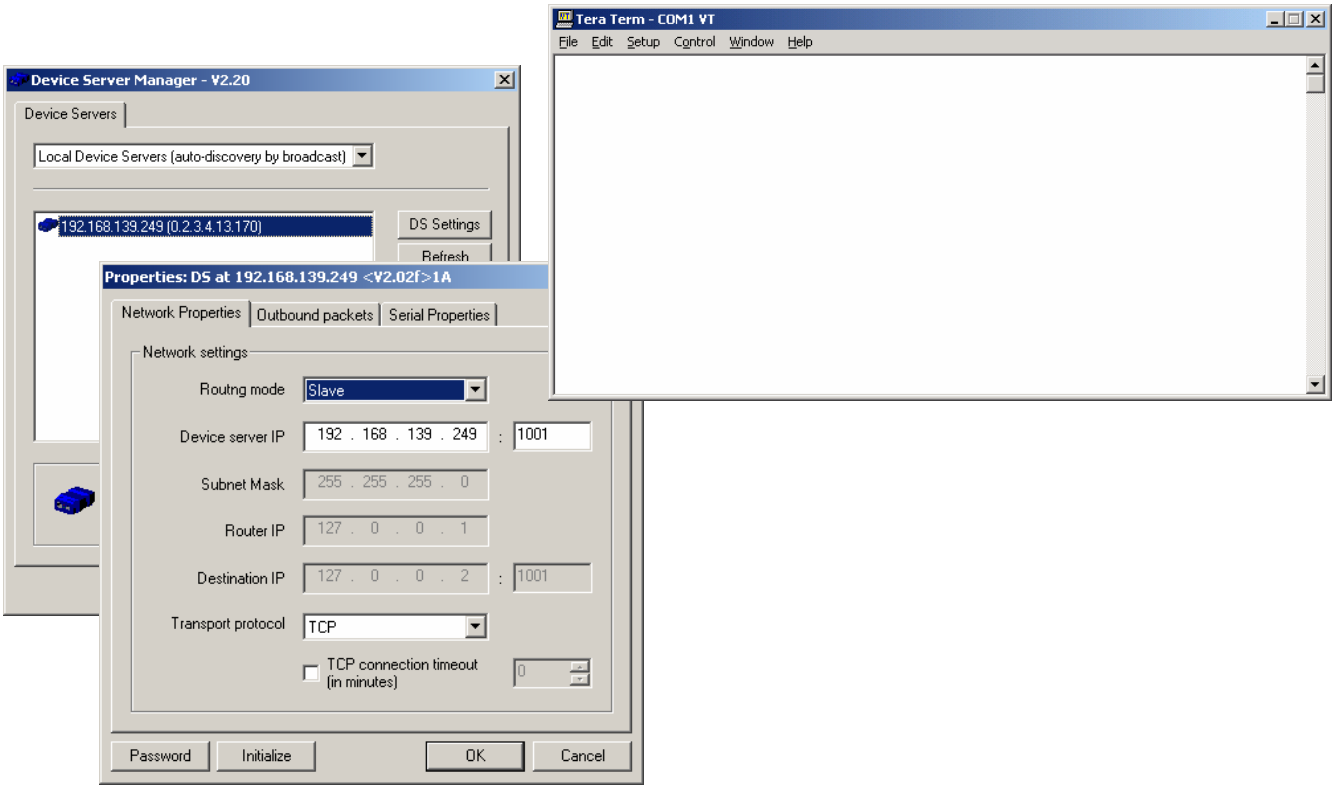


Figure 3: Screenshots of computer windows

4 Supervisory Control

Supervisory mode leverages the primary controller onboard the Wraptor™. This processor interprets incoming Supervisory Commands and then applies control signals across the set of seven (7) motion-control microprocessors. Supervisory mode allows you to command individual or multiple motors to close, open, and move to specific positions; it also provides for setting the various configuration parameters and reporting positions and torques.

At the simplest level, Supervisory mode allows you to type and receive ASCII text characters on a terminal (using any type of computer hardware or operating system, such as UNIX, Macintosh, PalmPilot, and proprietary robot controllers, etc.). To automate grasping applications, you can write programs, scripts, or macros that send and receive these text characters through an Ethernet connection (e.g. the optional Wraptor™ C-Function Library).

4.1 Commands

When the Wraptor™ firmware is ready to process a command, it prints a prompt of "=> " to your host computer. A command can then be entered as a single line, terminated by a carriage return character (0x0d). Once the firmware receives the carriage return, it processes the line, executes the command, and then prints a new prompt. Once a command has been started, no configuration changes can be made until the command has completed.

Many of the commands take one or more parameters; space characters should separate these from the command and each other. The command syntax is:

<motorPrefixes><command> [<Arguments>]

Motor commands refer to one or more of the seven motors. By default, all seven motors are affected and the Wraptor™ will simulate the BarrettHand's functionality. To select fewer than seven motors, a motor prefix must be placed before the command (with no space between the prefix and the command).

The motor prefixes and the resulting motors selected are:

Table 4 – Motor Prefixes

Value	**Motor**
1	Finger 1 (motors 1 and 5)
2	Finger 2 (motors 2 and 6)
3	Finger 3 (motors 3 and 7)
4	Spread (motor 4)
5	Finger 1 outer link (motor 5)
6	Finger 2 outer link (motor 6)
7	Finger 3 outer link (motor 7)
G	Grasp (fingers 1, 2, and 3)
S	Spread (motor 4)
IL	Inner Links (motors 1, 2, and 3)
OL	Outer Links (motors, 5, 6, and 7)
<No Motor Specified>	All motors

Example:

"12SET DP 8000" sets the default position for fingers 1 and 2 to 8000 GCL counts

Supervisory mode commands are organized into the following categories:

1) Movement Commands
2) Motor Parameter Commands
3) Administrative Commands

4.1.1 Movement Commands

Movement commands are motor commands: they immediately affect one or more of the motors. Each can take motor prefixes.

Command: **C**
Name: Close
Purpose: Commands the selected motor(s) to move fingers in close direction with a velocity ramp-down at target limit. If selected, each finger moves towards the palm, and the spread motor moves so that fingers F1 and F2 are adjacent to finger F3.
Arguments: (none)
Example: SC
Notes: The C command is similar to a MOVE command, with the value of the Close Target (CT) motor parameter as the destination. Move is trapezoidal: Uses ACCEL to ramp up the velocity to MV, then uses –ACCEL when approaching the target position. If an object is encountered before the trapezoidal velocity profile completes, the torque will ramp up to MT. See HOLD for more info.

Command: **HI**
Name: Hand Initialize
Purpose: Initializes the selected motor controller(s), preparing them for use by other movement commands.
Arguments: (none)
Example: HI
Notes: HI must be run before any other movement command. Generally it is run without a motor prefix, initializing all four motors; although, if desired, a subset of the motors can be specified. After an HI, all motors are in their home position; at 0 encoder counts.

Command: **HOME**
Name: Home
Purpose: Moves the selected motor(s) to position 0.
Arguments: (none)
Example: SGHOME
Notes:

Command: **IO**
Name: Incremental Open
Purpose: Opens the selected motor(s) DS counts.
Arguments: (none)
Example: 12IO
Notes:

Command: **IC**
Name: Incremental Close
Purpose: Closes the selected motor(s) DS counts.
Arguments: (none)
Example: GIC
Notes:

Command: **M**
Name: Move
Purpose: Moves the selected motor(s) to the given position. If no argument specified, then the motor(s) move(s) to the position given by parameter DP.
Arguments: Target position in GCL counts (optional)
Example: 13M 1000
Notes: Move is trapezoidal: Uses ACCEL to ramp up the velocity to MV, then uses –ACCEL when approaching the target position. If an object is encountered before the trapezoidal velocity profile completes, the torque will ramp up to MT. See HOLD for more info.

Command: **O**
Name: Open
Purpose: Commands the selected motor(s) to move fingers in open direction with a velocity ramp-down at target limits. If selected, F1, F2, and F3 open away from the palm, and the spread motor moves so that fingers F1 and F2 are opposite finger F3.
Arguments: (none)
Example: GO
Notes: The O command is similar to a MOVE command, with the value of the OT motor parameter as the destination. Move is trapezoidal: Uses ACCEL to ramp up the velocity to MV, then uses –ACCEL when approaching the target position. If an object is encountered before the trapezoidal velocity profile completes, the torque will ramp up to MT. See HOLD for more info.

Command: **T**
Name: Terminate power
Purpose: Turns of power to the selected motor(s).
Arguments: (none)
Example: ST
Notes:

Command: **TC**
Name: Torque-Controlled Close
Purpose: Sets the torque of selected motor(s) to MT.
Arguments: (none)
Example: STC
Notes: Does not check for hitting joint limits. Torque is applied until a new torque is applied or a different movement command is issued. Use caution when issued with a high MT- there is no velocity limit imposed during a torque move.

Command: **TO**
Name: Torque-Controlled Open
Purpose: Sets the torque of selected motor(s) to –MT.
Arguments: (none)
Example: STO
Notes: Does not check for hitting joint limits. Torque is applied until a new torque is applied or a different movement command is issued. Use caution when issued with a high MT- there is no velocity limit imposed during a torque move.

4.1.2 Motor Parameter Commands

Motor parameter commands act on the configuration parameters for one or more of the motors. See Section 4.2 for a complete list of motor parameters.

Command: **FSET**
Name: Finger Set
Purpose: Sets the given parameter to the given value for the selected motor(s)
Arguments: <parameterName> <parameterValue>
Example: SFSET DS 100
Notes: SET is a synonym for FSET.

Command: **FGET**
Name: Finger Get
Purpose: Gets and prints the given parameter's value for the selected motor(s). Each parameter prints one value for each selected motor separated by spaces.
Arguments: <parameterName>
Example: SFGET DS
Notes: GET is a synonym for FGET

Command: **FLOAD**
Name: Finger Load
Purpose: Loads the parameters of the selected motor(s) from non-volatile storage. This is done whenever the firmware starts up.
Arguments: (none)
Example: 3FLOAD
Notes: The non-volatile storage used does not depend on the super capacitor, so it retains its value even if the firmware is lost. LOAD is a synonym for FLOAD.

Command: **FSAVE**
Name: Finger Save
Purpose: Sets the parameters of the selected motor(s) to non-volatile storage.
Arguments: (none)
Example: 123FSAVE
Notes: SAVE is a synonym for FSAVE.

Command: **FDEF**
Name: Finger Default
Purpose: Sets the parameters of the selected motor(s) back to their factory default values.
Arguments: (none)
Example: SFDEF
Notes: Does not save the changed values to non-volatile storage. DEF is a synonym for FDEF.

4.1.3 Administrative Commands

Administrative commands implement various housekeeping functions.

Command: **RESET**
Name: Reset
Purpose: Resets the controller software. This is equivalent to doing a power cycle.
Arguments: (none)
Example: RESET
Notes: An HI is necessary after a RESET command.

Command: **VERS**
Name: Version
Purpose: Prints the firmware version.
Arguments: (none)
Example: VERS
Notes:

4.2 Parameters

Motor parameters change how a motor functions. Parameters are organized into the following categories:

1) Movement Parameters
2) Status Parameters
3) Advanced Parameters
4) Configuration Parameters

4.2.1 Movement Parameters

Movement parameters affect how a given motor moves.

Parameter:	**DP**
Name:	Default Position
Purpose:	Destination of M command if no argument specified
Values:	-2E9 to +2E9 (GCL counts)
Default:	Inner Link: 25000 Outer Link: 25000 Spread: 18500
Notes:	This parameter's true range of useful values is bounded by the joint limits of the axes (see Table 1- Joint limits).

Parameter:	**DS**
Name:	Default Step
Purpose:	Size of IC or IO command movement if no argument specified
Values:	-32768 to +32,767 (GCL counts)
Default:	Inner Link: 4096 Outer Link: 4096 Spread: 4096
Notes:	This parameter's true range of useful values is bounded by the joint limits of the axes (see Table 1- Joint limits). Setting DS to a negative number is allowed, but it is not recommended. Doing so would make the finger move in the open direction during an IC (Incremental Close) command, for example.

Parameter:	**MT**
Name:	Max Torque
Purpose:	Maximum torque to apply during any joint movement.
Values:	-32768 to +32,767 (mA, roughly)
Default:	6000
Notes:	It is useful to set this parameter before issuing a TC, TO, or M command.

Parameter:	**MV**
Name:	Max Velocity
Purpose:	Maximum velocity to spin the motor during a trapezoidal move (M) command.
Values:	-32768 to +32,767 (GCL counts / ms)
Default:	Inner Link: 100 Outer Link: 100 Spread: 20
Notes:	During a trapezoidal move, the joint will accelerate with ACCEL up to MV.

4.2.2 Status Parameters

Motor status parameters are read-only and give information about the state of a motor.

Parameter:	**P**
Name:	Position
Purpose:	The present position of the motor in GCL Counts.
Values:	See Table 1- Joint limits
Default:	N/A
Notes:	

Parameter:	**TEMP**
Name:	Temperature
Purpose:	The present temperature on the controller in degrees C.
Values:	0 to 125
Default:	N/A
Notes:	

Parameter:	**PTEMP**
Name:	Peak Temperature
Purpose:	The maximum temperature ever experienced by this controller
Values:	0 to 125
Default:	N/A
Notes:	This value is never reset; it is maintained through power failures and firmware downloads.

Parameter:	**SN**
Name:	Serial Number
Purpose:	The serial number of the controller.
Values:	N/A
Default:	N/A
Notes:	This value is never reset; it is maintained through power failures and firmware downloads.

4.2.3 Advanced Parameters

Users do not generally need these commands, but they might be useful.

Parameter:	**ACCEL**
Name:	Acceleration
Purpose:	Maximum acceleration and deceleration when moving from one position to another.
Values:	0 to 65,535
Default:	Inner Link: 100 Outer Link: 100 Spread: 10
Notes:	While the ACCEL parameter has a rather large range of values that it can accept, the motor can only follow a small subset of those values. In general, the useful range is from 0 to approx. 60. Above 60, the motors cannot provide enough torque to accelerate that quickly. The units for acceleration is defined as 256 * GCL Counts / ms / ms. So ACCEL = 10 yields 10/256 cts/ms/ms acceleration.

Parameter:	**CT**
Name:	Close Target
Purpose:	This is the position gone to by a C (“Close”) command.
Values:	-2E9 to +2E9
Default:	Inner Link: 72000 Outer Link: 72000 Spread: 37000
Notes:	This parameter’s true range of useful values is bounded by the joint limits of the axes (see Table 1- Joint limits).

Parameter:	**HOLD**
Name:	Hold
Purpose:	If non-zero, then the motor is left energized after each motion command in order to hold the position constant.
Values:	0, 1
Default:	Inner Link: 0 Outer Link: 0 Spread: 1
Notes:	Since the fingers are not back-drivable, this is generally set to 1 only for the spread motor.

Parameter: **IOFF**
Name: Initialization Offset
Purpose: Number of GCL counts to move from the joint limit encountered during initialization (HI). After this move is complete, the joint's zero position is defined as the present position.
Values: -2E9 to +2E9
Default: Inner Link: 5575
Outer Link: -79872
Spread: 0
Notes: During an HI, the joint is driven into its joint limit at velocity IVEL, then moved away from the limit by IOFF GCL counts. The final position is defined as the joint's zero position. This parameter's true range of useful values is bounded by the joint limits of the axes (see Table 1- Joint limits).

Parameter: **IVEL**
Name: Initialization Velocity
Purpose: Velocity at which to drive the joint into its limit during initialization (HI).
Values: -32768 to +32,767 (GCL counts / ms)
Default: Inner Link: -20
Outer Link: 50
Spread: -10
Notes: During an HI, the joint is driven into its joint limit at velocity IVEL, then moved away from the limit by IOFF GCL counts. The final position is defined as the joint's zero position. This parameter's true range of useful values is bounded by the joint limits of the axes (see Table 1- Joint limits).

Parameter: **KD**
Name: Derivative Gain
Purpose: The difference between the previous position error and the present position error is multiplied by this gain to obtain the derivative component of the generated command torque.
Values: -32768 to +32,767
Default: Inner Link:8000
Outer Link: 8000
Spread: 0
Notes: Increasing this value reduces the instability caused by a high KP with a risk of instability due to overcompensation.

Parameter: **KI**
Name: Integral Gain
Purpose: The historical sum of the position errors are multiplied by this gain to obtain the integral component of the generated command torque.
Values: -32768 to +32,767
Default: Inner Link: 12
Outer Link: 12
Spread: 0
Notes: Increasing this value makes the actual position and command position align more closely with a risk of instability.

Parameter: **KP**
Name: Proportional Gain
Purpose: The error between the command position and the actual position is multiplied by this gain to obtain the proportional component of the generated command torque.
Values: -32768 to +32,767
Default: Inner Link: 2500
Outer Link: 2500
Spread: 1500
Notes: Increasing this value makes the joint control stiffer with a risk of instability.

Parameter:	**OT**
Name:	Open Target
Purpose:	This is the position gone to by an O (“Open”) command.
Values:	-2E9 to +2E9
Default:	0
Notes:	This parameter’s true range of useful values is bounded by the joint limits of the axes (see Table 1- Joint limits).

Parameter:	**TSTOP**
Name:	Time to Stop
Purpose:	Time in milliseconds before motor is considered stopped.
Values:	0 to 65,535
Default:	Inner Link: 1000 Outer Link: 1000 Spread: 1000
Notes:	WARNING: Please use caution when adjusting this parameter. Setting TSTOP higher than its default can result in the motors heating up very quickly under moderate to heavy usage.

4.2.4 Configuration Parameters

Global configuration parameters affect the hand as a whole.

Parameter:	**BAUD**
Name:	Baud rate
Purpose:	Controls the serial port baud rate.
Values:	300, 1200, 2400, 9600, 19200, 28800
Default:	9600
Notes:	If you change BAUD, you will need to change the baud rate at which the Tibbo Ethernet device server communicates with the primary Wraptor™ controller. Changes to BAUD are not saved between power cycles.

Parameter:	**OTEMP**
Name:	OverTemperature
Purpose:	When the controller temperature comes within 16 C of OTEMP, MT is automatically reduced for every degree rise in temperature. When the temperature >= OTEMP, MT is set to zero. As the controller cools, MT is automatically increased up to its preset value.
Values:	0 to 125
Default:	82
Notes:	Value is temperature in degrees C.

Parameter:	**TIE**
Name:	Tie the commands of one motor to another
Purpose:	Echo commands from one motor to another. Primarily to simplify control of the inner and outer links.
Values:	-32768 to +32,767 (GCL counts / ms)
Default:	Motor 1: 5, Motor 2: 6, Motor 3: 7, Motor 4: 0, Motor 5: 0, Motor 6: 0, Motor 7: 0
Notes:	“1 SET TIE 5” will make motor 1 echo to motor 5 any commands it receives. “1 SET TIE 0” will turn off the TIE.

