INTRODUCTION

The ClearCore Command Protocol (CCCP) is an example project for use on the Teknic ClearCore I/O and motion controller, with optional integration to Teknic ClearPath-SD integrated servo motors and additional I/O points. Commands are passed to the CCCP running on ClearCore, where they are parsed and their commanded functionality realized.

Important Note: The ClearCore Command Protocol (CCCP) example program must be downloaded onto a ClearCore before use. ClearCore is not pre-loaded with any program.

Natively, the CCCP implements the following functionality:

- Controlling ClearPath-SD motors, including
  - Enabling and disabling
  - Commanding absolute or relative positional moves
  - Commanding velocity moves
- Querying motor information: position, velocity, and status
- Modifying parameters for motors, including
  - Setting acceleration and velocity limits on movement commands
  - Clearing motor alerts
  - Zeroing number space (useful for absolute positional moves)
- Reading and writing to digital and analog I/O ports
- Sending specific feedback and error messages, configurable as either simple numerical feedback or verbose, human-readable messages

The CCCP by default accepts input and sends output via USB connection to a terminal, such as the serial monitor in the Arduino IDE, the data visualizer in Microchip Studio, or another terminal application on a PC. It can also be configured to accept commands from other streams - such as ClearCore's COM ports, ethernet port, or XBee connection - and sources - such as typed user input, input from a text file, or control from another device sending text commands to ClearCore. Other ClearCore examples demonstrate communication via these alternate connection options.

The CCCP is designed to be highly customizable, easily modified and adapted to fit a variety of application needs. This guide highlights features and functionality of the CCCP without modification, and offers advice on how to expand its functionality.

Additional Resources

ClearCore Software Documentation:
https://teknic-inc.github.io/ClearCore-library/
ClearCore Manual:
https://www.teknic.com/files/downloads/clearcore_user_manual.pdf
ClearCore System Diagram and Connection Diagrams:
https://www.teknic.com/files/downloads/clearcore_user_manual.pdf
ClearPath Manual (DC Power):
https://www.teknic.com/files/downloads/clearpath_user_manual.pdf
ClearPath Manual (AC Power):
https://www.teknic.com/files/downloads/ac_clearpath-mc-sd_manual.pdf

Commands

General notes on command formatting and parsing

- Commands are formatted according to the following:
  - The first character of a command is a single letter. The CCCP is not case sensitive.
  - There is no space between a command letter and a required motor/connector number parameter.
  - There is not a space between a required motor/connector number parameter and a second letter parameter, e.g. query and limit commands.
  - There is a space between a required motor/connector number parameter (or a second letter parameter) and a second numerical parameter, e.g. velocity and limit commands.
- Acceptable motor numbers include 0, 1, 2, and 3. Assuming all motors have been correctly configured based on the header in the source code, all motor commands are valid on any motor.
- Acceptable connector numbers include 0, 1, 2, …, 11, and 12. To read or write values from a specific connector, the Mode of the connector must match the command. Connectors cannot be configured for each Mode. See INPUT AND OUTPUT, CONNECTOR CONFIGURATION below for more information.
- The standard library function atoi() (ASCII to integer) is used to parse numerical input:
  - atoi() accepts an optional sign character (- or +) followed by numerical characters.
  - atoi() ignores any leading whitespace
  - atoi() stops parsing at the first nonnumerical character. This includes a decimal point (.), meaning that atoi() will truncate any fractional/decimal value and return only the integer portion of a numerical parameter.
  - If no parameter or an invalid parameter is passed to atoi(), the function will return a value of zero. The CCCP is not designed to handle this behavior differently than if the command included a parameter of zero. Be sure to pass required parameters when necessary to avoid unexpected behavior.

EXISTING COMMANDS AND EXPLANATIONS

The following commands are available natively on the ClearCore Command Protocol.

Command	Functionality	Example (explanation)	Notes
e#	Enable motor	e0 (enable motor0)	The CCCP waits on HLFB to assert before accepting any subsequent commands. This allows a motor to complete any automatic homing procedure without being interrupted.
d#	Disable motor	d1 (disable motor1)	
m# distance	Command a positional move	m2 1000 (If ABSOLUTE_MOVE is set to 1, move motor2 to position 1000 steps. If ABSOLUTE_MOVE is set to 0, move motor2 1000 steps in the positive direction.)	The CCCP can be configured to command either absolute or relative positional commands by changing the value of the ABSOLUTE_MOVE #define compiler constant. A value of 1 configures the CCCP to command absolute moves; a value of 0 configures relative moves.
v# velocity	Command a velocity move	v3 10000 (command motor3 to move at 10000 steps/s)	
q#<p/v/s>	Query motor position, velocity, or status	q0s (query motor0's status)	Only one value (position, velocity, or status) can be queried in a single command. To query multiple values, utilize multiple commands. For position/velocity queries, the CCCP returns the real-time position/velocity that is being commanded by ClearCore. For status queries, the fields within the motor status register are returned.
l#<v/a> limit	Limit motor velocity or acceleration	l1v 100 (limit motor1's velocity to 100 steps/s)	Note that velocity limits only apply to positional moves. (Velocity moves will not be limited by the velocity limit set by this command.) Acceleration limits apply to both positional and velocity commands. If a limit is changed during a move, the limit will take effect on the next commanded move; no change will take place on the current move. The CCCP restricts the range of possible acceleration and velocity limits. These bounds can be viewed in the source code.
c#	Clear motor alerts	c2 (clear alerts on motor2)	ClearCore has built-in alerts that will cancel and disallow motion if they occur. This function will clear alerts and allow further motion. Ensure the original cause of the alert has been remedied before calling. If ClearPath motor shutdowns have occurred, this command will clear the shutdown by cycling the enable signal to the motor.

z#	Set the zero position for motor# to the current commanded position	z3 (define the zero position for motor3 as the current position)	This command is used to zero ClearCore's position reference by defining the current commanded position as the zero position. (No motion is commanded to the motor.)
i#	Read input	i9 (read the input on connector9 (A-9))	Read the input on connector. Digital input pins will return 0 or 1. Analog input pins will a return value on the range [0, 4095] corresponding to an input on the range [0,10]V. See INPUT AND OUTPUT, CONNECTOR CONFIGURATION below for more information.
o# outputVal	Write output	o5 1 (write to connector5 (IO-5) a value of 1; this turns on connector IO-5 if configured as a digital output.)	Write output on connector. Digital output pins accept a parameter of 0 or 1. Analog output pins accept a parameter on the range [409,2047] corresponding to an output on the range [4,20]mA. Ensure these values accurately command your desired output current, and make adjustments as necessary. See INPUT AND OUTPUT, CONNECTOR CONFIGURATION below for more information.
f fdbkType	Change feedback type	f 1 (change the feedback type to verbose)	Passing a 0 sets feedback to numerical; passing a 1 sets feedback to verbose.
h		h (display the help message)	

Adding new commands to the ClearCore Command Protocol

The CCCP is designed to be expanded easily by adding new commands and functionality. To add a new command (defined by a letter currently unassigned to a command), simply add a new case to the switch(input[0]) statement. A few things to keep in mind:

- Most of the existing CCCP commands include some level of error checking. What could cause your command to malfunction or command incorrect or unexpected behavior? Be sure to check these conditions as you implement a new command.
- If your command has a numerical parameter, it can be parsed by the atoi() command, which will return an integer. Pass the address of the first character where the numerical parameter (and any optional whitespace and an optional sign character (+ or -)) can appear. See other cases for examples of this implementation.
- It will likely be convenient to update the help message to include your new command. As necessary, also consider adding a new feedback or error message if your command introduces a unique failure case. See FEEDBACK below for information on how to add a new feedback message.
- Be sure to include a break; in your new case.

Feedback

The ClearCore Command Protocol reports feedback, either as verbose feedback messages (enabled by default) or as simple numerical feedback, for a variety of errors and general behavior. The choice between sending verbose versus numerical feedback is held in the verboseFeedback boolean (0=numerical, 1=verbose), which is accessed by SendFeedback() and can be modified by using the "f" command.

This feature is designed to be adaptable as functionality grows in user applications. To define and implement a new feedback message, the following steps should be taken:

- #define a new number for the feedback.
- Initialize a new message (implemented as a char*).
- Update the FeedbackMessages array to include a new entry with the new number and message.
- Increment the length of the FeedbackMessages array.

Input and Output, Connector Configuration

ClearCore has 13 configurable points of I/O. A table summarizing the acceptable connector modes for each ClearCore connector can be found in the ClearCore manual and is reproduced below for convenience. While each connector may have the capability to operate in more than one Mode (e.g. INPUT_DIGITAL, OUTPUT_ANALOG), a connector is configured to operate in one specific Mode at a time. The function of each connector is determined by its preconfigured Mode. The CCCP sets the operational Mode for each connector by default, but these Modes can be reconfigured according to the table in the manual. The input and output commands verify that the requested connector's configured Mode is acceptable for the attempted command.

Label	Digital Input	Digital Output[1]	0-10V Analog Input	4-20 mA Output[2]	Servos or Steppers	Speaker Tones	DC Motor Drive
IO-0	yes	yes		yes			
IO-1	yes	yes					
IO-2	yes	yes					
IO-3	yes	yes					
IO-4	yes	yes				yes	yes
IO-5	yes	yes				yes	yes
DI-6	yes						
DI-7	yes						
DI-8	yes						
A-9	yes		yes				
A-10	yes		yes				
A-11	yes		yes				
A-12	yes		yes				
M-0					yes[3]		
M-1					yes[3]		
M-2					yes[3]		
M-3					yes[3]		
CCIO-8[4]	yes	yes					

Note 1: All digital outputs are PWM capable (except for those on the CCIO-8 expansion board).

Note 2: This output can also provide 0-20mA, which is less commonly used.

Note 3: Each motor connector has 3 digital outputs (step, dir., enable) and 1 digital input.

Note 4: There are 8 of these I/O points on the CCIO-8 expansion module.

Absolute and Relative Moves

This example can be configured to command either relative positional moves or absolute positional moves. This functionality is controlled by the #define compiler constant ABSOLUTE_MOVE, which is set for absolute moves by default. To configure the CCCP to command absolute moves, ABSOLUTE_MOVE should have a value of 1. To command relative moves, ABSOLUTE_MOVE should have a value of 0. This change must be made when the project is uploaded to ClearCore; there is no native functionality to modify this setting during operation.