iMoto High Current Motion Controller Software Development Guide

Introduction

iMoto High Current Motion Controller provide UART and I2C Bus interface for easy interface with host like PC or microcontrollers. This document provides details of the UART command set and I2C commands.

The UART command set consists of ASCII based AT commands. The advantage of using ASCII based commands is that user is able to used terminal emulators like Hypertermial to send command directly to the iMoto. Note that in order to support the terminal emulators, all commands sent to the iMoto are echo back immediately before any replies. The baud rate used by UART is 115200.

The I2C interface implements the I2C Slave. It supports both read and write. Note that the iMoto may stretch the clock if necessary during read transaction.

Motion Control AT Command Guide

The format of the AT command is as follows. Each AT command is terminated by “Enter” key or “0x0D” in hex.

Command Format

AT+	Command Word	=	Command parameters	0x0D

AT Command Summary (CR means Carriage or Press return key)

SN	AT Command	Description	
1	AT[CR]	Check if AT command interpreter is working	
2	AT+VEL=12.34,45.78[CR]	Change Speed	
3	AT+VEL?[CR]	Read Speed	
4	AT+VELn=12.34,45.78[CR]	Change Speed of motor n (n = 0, 1)	
5	AT+VELn?[CR]	Read Speed of motor n (n = 0, 1)	
6	AT+POS=12.34,45.78[CR]	Change Position	
7	AT+POS?[CR]	Read Position	
8	AT+POSn=12.34[CR]	Change Position of Motor n (n = 0, 1)	
9	AT+POSn?[CR]	Read Position of Motor n (n=0,1)	
10	AT+PWMn=0.2[CR]	Change PWM Output for motor n (n = 0, 1)	
11	AT+PWMn?[CR]	Read PWM Output of motor n (n = 0, 1)	
12	AT+VMAX=12.34,45.78[CR]	Change Max Velocity for position mode	
13	AT+VMAX?[CR]	Read Max Velocity for position mode	
14	AT+VMAXn=12.34[CR]	Change Max Velocity for Motor n (n=0,1)	
15	AT+VMAXn?[CR]	Read Max Velocity of Motor n (n=0,1)	
16	AT+ACC=12.34,45.78[CR]	Change Acceleration	
17	AT+ACC?[CR]	Read Acceleration	
18	AT+ACCn=12.34[CR]	Change Acceleration of Motor n (n=0,1)	
19	AT+ACCn?[CR]	Read Acceleration of Motor n (n=0,1)	
20	AT+PIDn=1.2,3.4,5.6[CR]	Change PID Parameter for motor n (n = 0,1)	
21	AT+PIDn?[CR]	Read PID Parameter for motor n (n=0,1)	
22	AT+WHEELn=1.2,3.4,2048.0[CR]	Change Wheel Parameters for motor n (n = 0,1)	
23	AT+WHEELn?	Read Wheel Parameters for motor n (n = 0,1)	
24	AT+ ESTOP[CR]	Emergency Stop	
25	AT+ ESTOPn[CR]	Emergency Stop for motor n	
26	AT+ RESET[CR]	Reset Motion Controller	
27	AT+ RESETn[CR]	Reset Motion Controller for motor n (n = 0,1)	
28	AT+MOTOROFF[CR]	Turn OFF Motor Power	
29	AT+MOTORON[CR]	Turn ON Motor Power	
30	AT+MOTORnOFF[CR] (n = 0, 1)	Turn OFF Motor n Power	
31	AT+MOTORnON[CR] (n = 0, 1)	Turn ON Motor n Power	
32	AT+RCMAXVI=1000.0, 1000.0[CR]	Change the MAX Velocity in RC-Independent Mode	
33	AT+RCMAXVI?[CR]	Read the MAX Velocity in RC-Independent Mode	
34	AT+RCMAXVC=1000.0, 1000.0[CR]	Change the MAX Velocity in RC-Coordinated Mode	
35	AT+RCMAXVC?[CR]	Read the MAX Velocity in RC-Coordinated Mode	
36	AT+RCDEADBAND=100[CR]	Change RC neutral band width in uSec	
37	AT+RCDEADBAND?	Read RC neutral band width in uSec	
38	AT+RCMAXPWMI=1.0,1.0	Change the MAX PWM duty cycle in RC-Independent Mode	
39	AT+RCMAXPWMI?	Read the MAX PWM duty cycle in RC-Independent Mode	
40	AT+RCMAXPWMC=1.0,1.0	Change the MAX PWM duty cycle in RC-Coordinated Mode	
41	AT+RCMAXPWMC?	Read the MAX PWM duty cycle in RC-Coordinated Mode	
42	AT+SAVEALL[CR]	Save all RC related information in the non-volatile memory	

Change Speed

AT+VEL=12.34,45.78[CR]		
Description	• Set the speed of left motor and right motor in mm/sec •	
Reply	OK if successful, ERROR if unsuccessful	

Read Speed

AT+VEL?[CR]		
Description	• Read the speed of motor0 and motor1 • The motor speed is in floating point value	
Reply	+VEL=12.34,56.78 OK	

Change Speed of motor n (n = 0, 1)

AT+VELn=12.34,45.78[CR]		
Description	• Set the speed of motor n in mm/sec	
Reply	OK if successful, ERROR if unsuccessful	

Read Speed of motor n (n = 0, 1)

AT+VELn?[CR]		
Description	• Read the speed of motorn • The motor speed is in floating point value	
Reply	+VELn=12.34,56.78 OK	

Change Position

AT+POS=12.34,45.78[CR]		
Description	• Set the relative position of motor0 and motor1 with respect to the current position	
Reply	OK if successful, ERROR if unsuccessful	

Read Position

AT+POS?[CR]		
Description	• Read the current absolute position of the two motors in mm	
Reply	+POS=12.34,56.78 OK	

Change Position of Motor n (n = 0, 1)

AT+POSn=12.34[CR]		
Description	• Set the relative position of motor n with respect to the current position	
Reply	OK if successful, ERROR if unsuccessful	

Read Position of Motor n (n=0,1)

AT+POSn?[CR]		
Description	• Read the current position of the motor n in mm	
Reply	POSn=12.34 OK	

Change PWM Output for motor n (n = 0, 1)

AT+PWMn=0.2[CR]		
Description	• Set the duty cycle (1.0 for 100%) of the PWM output to the motor n •	
Reply	OK if successful, ERROR if unsuccessful	

Read PWM Output of motor n (n = 0, 1)

AT+PWMn?[CR]		
Description	• Read the current PWM output duty cycle of motor n • Return value is from -1.0 to +1.0 (-100% to +100%)	
Reply	+PWMn=0.112 OK	

Change Max Velocity for position mode

AT+VMAX=12.34,45.78[CR]		
Description	• Set the maximum absolution velocity limit in mm/sec for position mode.	
Reply	OK if successful, ERROR if unsuccessful	

Read Max Velocity for position mode

AT+VMAX?[CR]		
Description	• Read the maximum absolution velocity limit in mm/sec for position mode.	
Reply	+VMAX=12.34,56.78 OK	

Change Max Velocity for Motor n (n=0,1)

AT+VMAXn=12.34[CR]		
Description	• Set the maximum absolution velocity limit of motor n in mm/sec for position mode.	
Reply	OK if successful, ERROR if unsuccessful	

Read Max Velocity of Motor n (n=0,1)

AT+VMAXn?[CR]		
Description	• Read the maximum absolution velocity limit of motor n in mm/sec for position mode.	
Reply	+VAMXn=12.34 OK	

Change Acceleration

AT+ACC=12.34,45.78[CR]		
Description	• Set the Acceleration limit of motor 0 and motor 1 in mm/sec*sec	
Reply	OK if successful, ERROR if unsuccessful	

Read Acceleration

AT+ACC?[CR]		
Description	• Read the Acceleration limit of the two motors	
Reply	+ACC=12.34,56.78 OK	

Change Acceleration of Motor n (n=0,1)

AT+ACCn=12.34[CR]		
Description	• Set the Acceleration limit of motor n in mm/sec*sec	
Reply	OK if successful, ERROR if unsuccessful	

Read Acceleration of Motor n (n=0,1)

AT+ACCn?[CR]		
Description	• Read the Acceleration limit of the two motors	
Reply	+ACCn=12.34 OK	

Change PID Parameter for motor n (n = 0,1)

AT+PIDn=1.2,3.4,5.6[CR]		
Description	• Set the Kp, Ki, Kd for motor n •	
Reply	OK if successful, ERROR if unsuccessful	

Read PID Parameter for motor n (n=0,1)

AT+PIDn?[CR]		
Description	• Read the Kp, Ki, Kd for motor n •	
Reply	+PIDn=1.1,2.2,3.3 OK	

Change Wheel Parameters for motor n (n = 0,1)

AT+WHEELn=1.2,3.4,2048.0[CR]		
Description	• Set the Gear Ratio, Wheel Diameter, CPR for motor n •	
Reply	OK if successful, ERROR if unsuccessful	

Read Wheel Parameters for motor n (n = 0,1)

AT+WHEELn?		
Description	• Read the Gear Ratio, Wheel Diameter, CPR for motor n	
Reply	+WHEELn=1.2,3.4,2048.0 OK	

Emergency Stop

AT+ ESTOP[CR]		
Description	• Stop the 2 motors immediately at best effort.	
Reply	OK if successful, ERROR if unsuccessful	

Emergency Stop for motor n

AT+ ESTOPn[CR]		
Description	• Stop the 2 motors immediately at best effort.	
Reply	OK if successful, ERROR if unsuccessful	

Reset Motion Controller

AT+ RESET[CR]		
Description	• Reset the motion controller.	
Reply	OK if successful, ERROR if unsuccessful	

Reset Motion Controller for motor n (n = 0,1)

AT+ RESETn[CR]		
Description	• Reset the motion controller for motor n.	
Reply	OK if successful, ERROR if unsuccessful	

Turn OFF Motor Power

AT+MOTOROFF[CR]		
Description	• Turn OFF the power for the 2 motors. The 2 motors are in free-running state.	
Reply	OK if successful, ERROR if unsuccessful	

Turn ON Motor Power

AT+MOTORON[CR]		
Description	• Turn ON the power for the 2 motors. The 2 motors are in velocity mode.	
Reply	OK if successful, ERROR if unsuccessful	

Turn OFF Motor n Power

AT+MOTORnOFF[CR] (n = 0, 1)		
Description	• Turn OFF the power for the motor n. The motor is in free-running state.	
Reply	OK if successful, ERROR if unsuccessful	

Turn ON Motor n Power

AT+MOTORnON[CR] (n = 0, 1)		
Description	• Turn ON the power for the motor n. The motor is in velocity mode.	
Reply	OK if successful, ERROR if unsuccessful	

Set MAX Velocity in RC-Independent Mode

AT+RCMAXVI=1000.0, 1000.0[CR]		
Description	• Change the MAX Velocity in RC-Independent Mode in mm/sec. The MAX velocity is the velocity when the RC PWM pulse width is at 2.0msec or 1.0msec	
Reply	OK if successful, ERROR if unsuccessful	

Read Max Velocity for RC-Independent Mode

AT+RCMAXVI?[CR]		
Description	• Read the MAX velocity limit in mm/sec for RC-Independent mode.	
Reply	+ RCMAXVI =1000.0,1000.0 OK	

Set MAX Velocity in RC-Coordinated Mode

AT+RCMAXVC=1000.0, 1000.0[CR]		
Description	• Change the MAX Velocity in RC-Coordinated Mode in mm/sec. The MAX velocity is the velocity when the RC PWM pulse width is at 2.0msec or 1.0msec	
Reply	OK if successful, ERROR if unsuccessful	

Read Max Velocity for RC-Coordinated Mode

AT+RCMAXVC?[CR]		
Description	• Read the MAX velocity limit in mm/sec for RC-Coordinated mode.	
Reply	+ RCMAXVC =1000.0,1000.0 OK	

Set RC neutral band width

AT+RCDEADBAND=400[CR]		
Description	• Change the neutral band width tolerance for RC PWM input in uSec. The neutral band width is pulse width around 1.5ms to be considered neutral.	
Reply	OK if successful, ERROR if unsuccessful	

Read RC neutral band width

AT+RCDEADBAND?[CR]		
Description	• Read the neutral band width tolerance for RC PWM input in uSec	
Reply	+ RCDEADBAND=400.0 OK	

Set MAX PWM output in RC-Independent Mode

AT+RCMAXPWMI=1.0, 1.0[CR]		
Description	• Change the MAX PWM output duty cycle in RC-Independent Mode in percentage. The MAX PWM is the PWM duty cycle when the RC PWM pulse width is at 2.0msec or 1.0msec	
Reply	OK if successful, ERROR if unsuccessful	

Read MAX PWM output in RC-Independent Mode

AT+RCMAXPWMI?[CR]		
Description	• Read the MAX PWM output duty cycle in RC-Independent Mode in percentage.	
Reply	+ RCMAXPWMI =1.0,1.0 OK	

Set MAX PWM output in RC-Coordinated Mode

AT+RCMAXPWMC=1.0, 1.0[CR]		
Description	• Change the MAX PWM output duty cycle in RC-Coordinated Mode in percentage. The MAX PWM is the PWM duty cycle when the RC PWM pulse width is at 2.0msec or 1.0msec	
Reply	OK if successful, ERROR if unsuccessful	

Read MAX PWM output in RC-Coordinated Mode

AT+RCMAXPWMC?[CR]		
Description	• Read the MAX PWM output duty cycle in RC-Coordinated Mode in percentage.	
Reply	+ RCMAXPWMC=1.0,1.0 OK	

Save all the RC related parameter in the EEPROM

AT+SAVEALL[CR]		
Description	• Save all RC related information in EEPROM	
Reply	OK if successful, ERROR if unsuccessful	

Motion Control I2C Command Guide

I2C Write Command Sequence

S	I2C Address	A	Register	A	Length	A	Data0	A	Data1	A	….	Data n-1	A	P

S	Start Condition
I2C Address	I2C address of the slave, with R/W bit = 0
A	Acknowledge from slave
Register	Register address to written. The register address for motion control is **0x20**
Length	Length of the data to be written to slave. **Note:** The maximum length is 127. The MSB of the length bye is 0 for writing data. The MSB of the length byte is 1 for reading data.
Data0 – Data n-1	The data bytes (assume the length is n)
P	Stop condition

I2C Read Command Sequence

S	I2C Address	A	Register	A	Length	A	RS	Data0	Am	Data1	Am	….	Data n-1	NAm	P

S	Start Condition
I2C Address	I2C address of the slave, with R/W bit = 0
A	Acknowledge from slave
Register	Register address to written. The register address for motion control is **0x20**
Length	Length of the data to be read from slave. **Note:** The maximum length is 127. The MSB of the length bye is 0 for writing data. The MSB of the length byte is 1 for reading data.
RSAm	Re-start condition
Data0 – Data n-1	The data bytes (assume the length is n)
Am	Acknowledge from Master
NAm	No Acknowledge from Master
P	Stop condition

Motion Control Command Data Format

The data packet to write to the motion controller and read from motion controller should follow the following format.

CMD_TYPE	Parameter Bytes	0x00

CMD_TYPE	1 byte to indicate the command to be used
Data Bytes	The parameters for this command. The length varies for different commands
0x00	Indicates the end of the command packet

When writing to the motion controller, the data packet is formed and the length is calculated. The length and data packet is then sent in the I2C write command sequence to send to motion controller.

When reading from the motion controller, the expected length of data is sent in the I2C read sequence. Once the data packet is received, the data can be interpreted using the command format.

The command type and parameters are shown in the table below.

Motion Control Command to Change Motor State

To change the motor state, the corresponding command packet is to be written to the motion controller using the Write Command Sequence.

Change Velocity (mm/sec) of Motor 0 and Motor 1

CMD_TYPE	0x00					
I2C Write Packet	0x00	Velocity 0	Velocity 1	0x00		
Data Length (byte)	10					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Change Position (mm) of Motor 0 and Motor 1

CMD_TYPE	0x01					
I2C Write Packet	0x01	Position 0	Position1	0x00		
Data Length (byte)	10					
Note:	Position data is 4 byte each, IEEE-754 floating point format					

Change PWM duty cycle of driver of Motor 0 and Motor 1

CMD_TYPE	0x08					
I2C Write Packet	0x08	PWM 0	PWM 1	0x00		
Data Length (byte)	10					
Note:	PWM data is 4 byte each, IEEE-754 floating point format					

Change Acceleration Limit (mm/sec/sec) of Motor 0 and Motor 1

CMD_TYPE	0x02					
I2C Write Packet	0x02	Acceleration 0	Acceleration 1	0x00		
Data Length (byte)	10					
Note:	Acceleration data is 4 byte each, IEEE-754 floating point format					

Change Velocity Limit (mm/sec) for position mode of Motor 0 and Motor 1

CMD_TYPE	0x03					
I2C Write Packet	0x03	Velocity 0	Velocity 1	0x00		
Data Length (byte)	10					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Change PID parameters for Motor0

CMD_TYPE	0x04					
I2C Write Packet	0x04	Kp	Ki	Kd	0x00	
Data Length (byte)	14					
Note:	PID data is 4 byte each, IEEE-754 floating point format					

Change PID parameters for Motor1

CMD_TYPE	0x06					
I2C Write Packet	0x06	Kp	Ki	Kd	0x00	
Data Length (byte)	14					
Note:	PID data is 4 byte each, IEEE-754 floating point format					

Change Wheel parameters for Motor0

CMD_TYPE	0x05					
I2C Write Packet	0x05	Gear Ratio	Wheel Diameter	CPR	0x00	
Data Length (byte)	14					
Note:	All wheel data is 4 byte each, IEEE-754 floating point format					

Change Wheel parameters for Motor1

CMD_TYPE	0x07					
I2C Write Packet	0x07	Gear Ratio	Wheel Diameter	CPR	0x00	
Data Length (byte)	14					
Note:	All wheel data is 4 byte each, IEEE-754 floating point format					

Change Velocity (mm/sec) of Motor 0

CMD_TYPE	0x10					
I2C Write Packet	0x10	Velocity 0	0x00			
Data Length (byte)	6					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Change Velocity (mm/sec) of Motor 1

CMD_TYPE	0x20					
I2C Write Packet	0x20	Velocity 1	0x00			
Data Length (byte)	6					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Change Position (mm) of Motor 0

CMD_TYPE	0x11					
I2C Write Packet	0x11	Position 0	0x00			
Data Length (byte)	6					
Note:	Position data is 4 byte each, IEEE-754 floating point format					

Change Position (mm) of Motor 1

CMD_TYPE	0x21					
I2C Write Packet	0x21	Position 1	0x00			
Data Length (byte)	6					
Note:	Position data is 4 byte each, IEEE-754 floating point format					

Change PWM duty cycle of driver of Motor 0

CMD_TYPE	0x14					
I2C Write Packet	0x14	PWM 0	0x00			
Data Length (byte)	6					
Note:	PWM data is 4 byte each, IEEE-754 floating point format					

Change PWM duty cycle of driver of Motor 1

CMD_TYPE	0x24					
I2C Write Packet	0x24	PWM 1	0x00			
Data Length (byte)	6					
Note:	PWM data is 4 byte each, IEEE-754 floating point format					

Change Acceleration Limit (mm/sec/sec) of Motor 0

CMD_TYPE	0x12					
I2C Write Packet	0x12	Acceleration 0	0x00			
Data Length (byte)	6					
Note:	Acceleration data is 4 byte each, IEEE-754 floating point format					

Change Acceleration Limit (mm/sec/sec) of Motor 1

CMD_TYPE	0x22					
I2C Write Packet	0x22	Acceleration 1	0x00			
Data Length (byte)	6					
Note:	Acceleration data is 4 byte each, IEEE-754 floating point format					

Change Velocity Limit (mm/sec) for position mode of Motor 0

CMD_TYPE	0x13					
I2C Write Packet	0x13	Max Velocity 0	0x00			
Data Length (byte)	6					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Change Velocity Limit (mm/sec) for position mode of Motor 1

CMD_TYPE	0x13					
I2C Write Packet	0x13	Max Velocity 1	0x00			
Data Length (byte)	6					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Emergency Stop for Motor 0 and Motor 1

CMD_TYPE	0x70					
I2C Write Packet	0x70	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Emergency Stop for Motor 0

CMD_TYPE	0x71					
I2C Write Packet	0x71	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Emergency Stop for Motor 1

CMD_TYPE	0x72					
I2C Write Packet	0x72	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Perform Reset for Motor 0 and Motor 1

CMD_TYPE	0x73					
I2C Write Packet	0x73	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Perform Reset for Motor 0

CMD_TYPE	0x74					
I2C Write Packet	0x74	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Perform Reset for Motor 1

CMD_TYPE	0x75					
I2C Write Packet	0x75	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Turn OFF power of Motor 0 and Motor 1

CMD_TYPE	0x77					
I2C Write Packet	0x77	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Turn ON power of Motor 0 and Motor 1

CMD_TYPE	0x78					
I2C Write Packet	0x78	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Turn OFF power for Motor 0

CMD_TYPE	0x79					
I2C Write Packet	0x79	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Turn ON power for Motor 0

CMD_TYPE	0x7A					
I2C Write Packet	0x7A	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Turn OFF power for Motor 1

CMD_TYPE	0x7B					
I2C Write Packet	0x7B	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Turn ON power for Motor 1

CMD_TYPE	0x7C					
I2C Write Packet	0x7C	0x00				
Data Length (byte)	2					
Note:	No parameter byte needed					

Motion Control Command to Read Motor State

To read the motor state, the corresponding command packet is to be written to the motion controller using the Write Command Sequence first, then the Read Command Sequence is performed to read the data back.

Read Velocity (mm/sec) of Motor 0 and Motor 1

CMD_TYPE	0x80					
I2C Write Packet	0x80	0x00				
Data Length (byte)	2					
Read Data Packet	0x80	Velocity 0	Velocity1	0x00		
Data Length (byte)	10					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Read Position (mm) of Motor 0 and Motor 1

CMD_TYPE	0x81					
I2C Write Packet	0x81	0x00				
Data Length (byte)	2					
Read Data Packet	0x81	Position 0	Position 1	0x00		
Data Length (byte)	10					
Note:	Position data is 4 byte each, IEEE-754 floating point format					

Read PWM Duty Cycle of Motor 0 and Motor 1

CMD_TYPE	0x88					
I2C Write Packet	0x88	0x00				
Data Length (byte)	2					
Read Data Packet	0x88	PWM 0	PWM 1	0x00		
Data Length (byte)	10					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Read Acceleration Limit of Motor 0 and Motor 1

CMD_TYPE	0x82					
I2C Write Packet	0x82	0x00				
Data Length (byte)	2					
Read Data Packet	0x82	Acceleration 0	Acceleration 1	0x00		
Data Length (byte)	10					
Note:	Acceleration data is 4 byte each, IEEE-754 floating point format					

Read Velocity Limit (mm/sec) for position mode of Motor 0 and Motor 1

CMD_TYPE	0x83					
I2C Write Packet	0x83	0x00				
Data Length (byte)	2					
Read Data Packet	0x83	Velocity 0	Velocity 1	0x00		
Data Length (byte)	10					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Read PID parameters of Motor 0

CMD_TYPE	0x84					
I2C Write Packet	0x84	0x00				
Data Length (byte)	2					
Read Data Packet	0x84	Kp	Ki	Kd	0x00	
Data Length (byte)	14					
Note:	PID data is 4 byte each, IEEE-754 floating point format					

Read PID parameters of Motor 1

CMD_TYPE	0x86					
I2C Write Packet	0x86	0x00				
Data Length (byte)	2					
Read Data Packet	0x86	Kp	Ki	Kd	0x00	
Data Length (byte)	14					
Note:	PID data is 4 byte each, IEEE-754 floating point format					

Read Wheel parameters of Motor 0

CMD_TYPE	0x85					
I2C Write Packet	0x85	0x00				
Data Length (byte)	2					
Read Data Packet	0x85	Gear Ratio	Wheel Diameter	CPR	0x00	
Data Length (byte)	14					
Note:	Wheel data is 4 byte each, IEEE-754 floating point format					

Read Wheel parameters of Motor 1

CMD_TYPE	0x87					
I2C Write Packet	0x87	0x00				
Data Length (byte)	2					
Read Data Packet	0x85	Gear Ratio	Wheel Diameter	CPR	0x00	
Data Length (byte)	14					
Note:	Wheel data is 4 byte each, IEEE-754 floating point format					

Read Velocity (mm/sec) of Motor0

CMD_TYPE	0x90					
I2C Write Packet	0x90	0x00				
Data Length (byte)	2					
Read Data Packet	0x90	Velocity 0	0x00			
Data Length (byte)	6					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Read Velocity (mm/sec) of Motor1

CMD_TYPE	0xA0					
I2C Write Packet	0xA0	0x00				
Data Length (byte)	2					
Read Data Packet	0xA0	Velocity 0	0x00			
Data Length (byte)	6					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Read Position (mm) of Motor 0

CMD_TYPE	0x91					
I2C Write Packet	0x91	0x00				
Data Length (byte)	2					
Read Data Packet	0x91	Position 0	0x00			
Data Length (byte)	6					
Note:	Position data is 4 byte each, IEEE-754 floating point format					

Read Position (mm) of Motor 1

CMD_TYPE	0xA1					
I2C Write Packet	0xA1	0x00				
Data Length (byte)	2					
Read Data Packet	0xA1	Position 1	0x00			
Data Length (byte)	6					
Note:	Position data is 4 byte each, IEEE-754 floating point format					

Read Acceleration Limit (mm/sec/sec) of Motor 0

CMD_TYPE	0x92					
I2C Write Packet	0x92	0x00				
Data Length (byte)	2					
Read Data Packet	0x92	Acceleration 0	0x00			
Data Length (byte)	6					
Note:	Acceleration data is 4 byte each, IEEE-754 floating point format					

Read Acceleration Limit (mm/sec/sec) of Motor 1

CMD_TYPE	0xA2					
I2C Write Packet	0xA2	0x00				
Data Length (byte)	2					
Read Data Packet	0xA2	Acceleration 1	0x00			
Data Length (byte)	6					
Note:	Acceleration data is 4 byte each, IEEE-754 floating point format					

Read Velocity Limit (mm/sec) for position mode of Motor 0

CMD_TYPE	0x93					
I2C Write Packet	0x93	0x00				
Data Length (byte)	2					
Read Data Packet	0x93	Velocity 0	0x00			
Data Length (byte)	6					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Read Velocity Limit (mm/sec) for position mode of Motor 1

CMD_TYPE	0xA3					
I2C Write Packet	0xA3	0x00				
Data Length (byte)	2					
Read Data Packet	0xA3	Velocity 1	0x00			
Data Length (byte)	6					
Note:	Velocity data is 4 byte each, IEEE-754 floating point format					

Read PWM Duty Cycle of Motor 0

CMD_TYPE	0x94					
I2C Write Packet	0x94	0x00				
Data Length (byte)	2					
Read Data Packet	0x94	PWM 0	0x00			
Data Length (byte)	6					
Note:	PWM data is 4 byte each, IEEE-754 floating point format					

Read PWM Duty Cycle of Motor 1

CMD_TYPE	0xA4					
I2C Write Packet	0xA4	0x00				
Data Length (byte)	2					
Read Data Packet	0xA4	PWM 1	0x00			
Data Length (byte)	6					
Note:	PWM data is 4 byte each, IEEE-754 floating point format					

Example to change motor speed

iMoto Address: 0x70

To set: Motor 1 speed = 112.3 mm/sec, Motor 2 speed = -234.5 mm/sec

112.3 in hex is 0x42E0999A, -234.5 in hex is 0xC36A8000

Command Packet to sent to the iMoto

0x00	0x9A	0x99	0xE0	0x42	0x00	0x80	0x6A	0xC3	0x00

- Send start condition
- Send I2C address 0x70
- Send register address 0x20
- Send length 0x0A
- Send the command packet above
- Send Stop condition

Example to Read motor Position

iMoto Address: 0x70

Command Packet to sent to the iMoto

0x81	0x00

Read back data from iMoto is as follows (assume Motor0 = 112.3mm, Motor 1 is -234.5mm)

0x81	0x9A	0x99	0xE0	0x42	0x00	0x80	0x6A	0xC3	0x00

I2C Command Sequence

- Send start condition
- Send I2C address 0x70
- Send register address 0x20
- Send length 0x02
- Send the command packet above
- Send Stop condition

- Send start condition
- Send I2C address 0x71
- Send register address 0x20
- Send length 0x0A
- Send Restart Condition
- Read 10 byte of data (Acknowledge for the first 9 bytes, NO-ACK for the last byte)
- Send Stop condition