mightyZAP API Manual for Python

- Force Control version actuator (model name starts with 12Lf) users can use all the following parameters.
- The parameter marked **[F/C version only]** applies only to the Force Control version actuator (model name starts with 12Lf) and does not apply to the Position control version actuator.
- Position Control version actuator (model name starts with L12 or D12) user can use all parameters except **[F/C version only]** parameters.
- For users of Raspberry Pi HAT IR-ST02, please use "PythonLibMightyZap_Rasp.py" API. (Use GPIO pin -> RS485/TTL: GPIO23, ttyAMA0 / PWM pin: GPIO18)

Start – COM port connection

OpenMightyZap(portname,BaudRate)

Function : Port connection for linear actuator control.

Parameter :

- portname : Put Com Port number like a"COM6"
- BaudRate : Assign baudrate among 9600,19200,57600 and115200bps

Termination

CloseMightyZap()

Function : Close(terminate) port connection which controlled Linear actuator.

Parameter : N/A

Restart

Restart(bID)

Function : Restart servo actuator system

(Restart system in case of shutdown due to Overload or input Voltage Error, etc)

Parameter :

- bID : Actuator ID (1~253), Broad CAST ID(254), Stand-alone ID (0)
  (herein after, explanation about actuator ID to be omitted)

Moving to Goal Position

GoalPosition(bID, position)

Function : Move stroke position to the desired Goal position

Parameter :

- bID : Actuator ID
- position : Desired position (0~4095)

Moving Speed Adjustment [F/C Version Only]

GoalSpeed(bID, speed)

Function : Adjust moving speed of actuator

Parameter :

-bID : Actuator ID

-speed : Set moving Speed (0~1023)

(0 : Max speed, 1~1023 Speed adjustment (1:low, 1023:high))

· Goal Current Adjustment [F/C Version Only]

GoalCurrent(bID, current)

Function : Adjust max current of actuator to decrease the force or relieve stress of motor

Parameter :

-bID : Actuator ID

-speed : Set moving Speed (0~1023)

(0 : Max speed, 1~1023 Speed adjustment (1:low, 1023:high)

※ Speed will be decreased when current value is decreased.

※ The value represents current[mA] and has +/-15% tolerance.

※ High Goal current setting may affect to the lifespan of servo actuator. Please refer to the data sheet for proper use.

· Read/Feedback Present Position

PresentPosition(bID)

Function : Read / feedback present stroke position

Parameter :

-bID : Actuator ID

- return : feedback present stroke position (Return '-1' value in case of Timeout)

• Acceleration Control [F/C Version Only]

Acceleration(bID, accel)

Function: Control acceleration rate of servo actuator

Parameter :

- bID : Actuator ID

- accel: Acceleration rate value of actuator (1~255)

The higher the value, the faster the acceleration.

• Deceleration Control [F/C Version Only]]

Deceleration(bID, decel)

Function: Control deceleration rate of servo actuator

Parameter :

- bID : Actuator ID

- decel: Deceleration rate value of actuator (1~255)

The larger the value, the larger the deceleration rate and the slower the deceleration time.

※If it is set too large, overshoot and minute vibration may occur when reaching the final position.

※ If overshoot occurs while changing the basic settings of the servo motor such as Speed and GoalCurrent, you can reduce the deceleration rate.

· Force On / Off

ForceEnable(bID, enable)

Function : Activate/Inactivate Linear actuator Force (Force ON/OFF)

Parameter :

-bID : Actuator ID

-enable : 0 - Force Off / 1 - Force On

※ Communication still alive even if actuator is under Force Off status.

※ Automatically Force ON when new goal position command is made.

What is the Force Off?

The actuator lineup having rated load 35N or more, have a characteristic to keep the position due to the mechanical design characteristics without motor power.
Therefore, if the servo actuator in the facility needs to maintain a certain position for considerable time, you can extend the life of the actuator by shutting off the motor power with the Force Off command. In this case, the communication is still maintained, and only the motor is powered off. When the position movement command is given again, the force is automatically turned on and the next command is executed.

· Shutdown On / Off

SetShutDownEnable (bID, flag)

Function : Setting for Shutdown On/Off function for respective error

Parameter :

-bID : Actuator ID

- flag : Setting value (See mightyZAP user manual - Alarm shutdown)

· Read Shutdown status

GetErrorShutDownEnable(bID)

Function : Reading shutdown On/Off status for error

Parameter :

-bID :Actuator ID

- return : Read/feedback shutdown On/Off status for error. (Return '-1' value in case of Timeout)

· LED Error Indication

SetErrorLEDEnable(bID, flag)

Function : Setting for LED indications On/Off for the alarm of respective errors.

Parameter :

-bID :Actuator ID

- flag : Setting value (See mightyZAP user manual – Element description➔Error indication)

· Read LED Error Indication Status

GetErrorLEDEnable(bID,)

Function : Read/Feedback LED indications On/Off status for error

Parameter :

-bID :Actuator ID

- return : Feedback setting status (Return '-1' value in case of Timeout)

· Read Error

ReadError(bID)

Function : Read/feedback current error status

Parameter :

-bID :Actuator ID

- return : feedback current error status (Return '-1' value in case of Timeout)

· Write to Address

Write_Addr(bID, Addr,Size,Data)

Function : Writes data directly to the certain address of parameter(memory/parameter) in data map.

Parameter :

-bID : Actuator ID

- Addr : Address to write data
- Size : Data size to write
- Data : Input Data

(See mightyZAP user manual –Data Map)

· Read from Address

Read_Addr(bID, Addr, Size)

Function : READ data directly from the certain address of parameter(memory/parameter) in data map.

Parameter :

-bID : Actuator ID

- Addr : Address to read Data
- Size : Data Size to read
- return : Feedback read Data (Return '-1' value in case of Timeout)

(See mightyZAP user manual –Data Map)

• Save data to same address in multiple actuators

Sync_Write_data(Addr, data,size)

Function : Save data to same address in multiple actuators

Parameter :

- Addr : Parameter address to write data, See Factor#1 in the table below
- data : Input data matrix,

Enter factor values as a matrix after factor #2 in the table below

Length + ID1 + Datat#1, +Data#2 +ID2 + Data#1 + data#2 …..

- Size : Input Data Size

- Servo ID 1 Goal Position : 1023(0x03FF), Servo ID 2 Goal Position : 2047(0x07FF)

HEADER	ID	Size	Command	Factor #1	Factor #2
				addr	Length
0xFFFFFF	0xFE	0x0a	0x73	0x86	0x02

Factor #3	Factor #4	Factor #5	Factor #6	Factor #7	Factor #8	Checksum
1> ID	1> Data #1	1>Data #2	2>ID	2> Data #1	2> Data #2	
0x01	0xff	0x03	0x02	0xFF	0x07	0xF1

(For more detailed info, see mightyZAP user manual - Symmetric Store)

• Direct packet transmission

WritePacket(buffer, size)

Function : Use when the user directly constructs a packet and sends one completed packet to the servo actuator

Parameter :

- buffer : Packet buffer to be sent to the actuator
- size: Packet buffer size to be sent to the actuator

(See the mightyZAP user manual – Packet Description and Packet examples)

• Python Example : GoalPosition/ PresentPosition

```
importPythonLibMightyZap
importtime

MightyZap = PythonLibMightyZap
Actuator_ID = 0

MightyZap.OpenMightyZap('COM3',57600)

fori in range(0,2):
MightyZap.goalPosition(Actuator_ID,4095)
time.sleep(3)
print(MightyZap.presentPosition(0))
MightyZap.goalPosition(Actuator_ID,0)
time.sleep(3)
print(MightyZap.presentPosition(0))
```