Micronix Python Programming Guide

Introduction

This documentation describes the implementations of the Python programming file to connect and interface with the MMC Controllers

Library : MMC_PyLibrary.py

Features

- Serial Port Communication with MMC Devices
  - Connect to / Disconnect from MMC Devices via COM Port
  - Send command to / Receive data from MMC Devices.

- Ethernet Communication with MMC Device
  - Connect to / Disconnect from MMC Devices via Server Address and Port
  - Send command to / Receive data from MMC Devices.

Supported MMC Devices

MMC-10 , MMC-100 , MMC-103 , MMC-110 , MMC-200, MMC-Ethernet

Required Python Packages

Built based on Version : Python 3.9.13

1. pyserial

   Can install in the python directory by:

   - pip install pyserial

Current List of Functions

- openComPortMMC(string port)
- connectEthMMC(string serverName, string serverPort)
- pollValues(string axis, string command)
- writeCommandToMMC(string axis, string command)
- closeConnectionMMC()

Setting Up Python Environment

Start: Please include ‘MMC_PyLibrary.py’ in your python environment directory.

To get Started :

In your example code or main python file, import the library file as :

```
import MMC_PyLibrary as pyMMC
```

Basic Example :

```
import MMC_PyLibrary as pyMMC

# For serial Connection On ComPort : COM6, Uncomment below
#pyMMC.openComPortMMC("COM6")

# For Ethernet Connection for default Port and Server Address
pyMMC.ConnectEthMMC("192.168.0.20", "5000")


# For Querying response from the MMC Controller
# Using the format of Axis and Command
# Please refer to the MMC Manual for the Command List
pyMMC.pollValues("1", "ver")

pyMMC.pollValues("2", "ver")


# Allows to write Command without waiting for response
pyMMC.writeCommandToMMC("2", "enc50")

# Close connection will close depending on serial or ethernet
pyMMC.closeConnectionMMC()
```

Terminal Output:

```
MMC-ETHERNET V2.1

MMC-200 V1.4.23
```

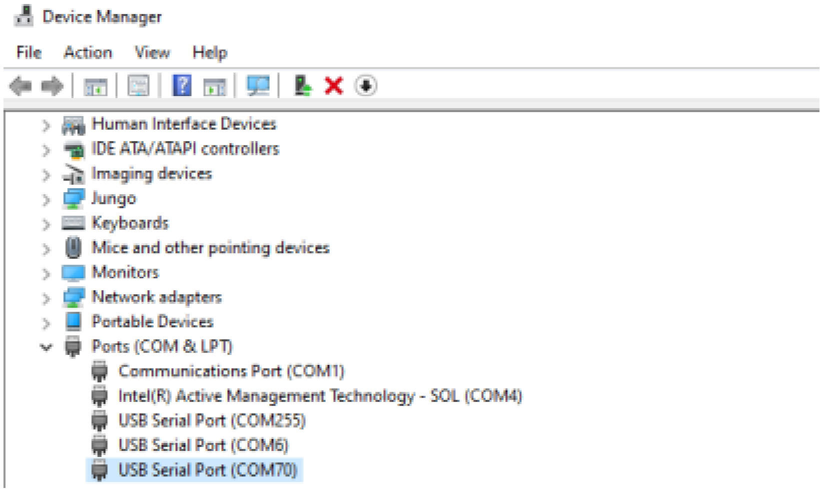
Library Functions

- openComPortMMC (string port)

Function :

Connects to the MMC Device Virtual COM Port through the USB Connection
This function requires user to know the name of the COM port of the MMC Device.

COM port name can be found in Device Manager as “USB Serial Port (COMx)”



Return Value :

bool output (True or False)

- True is returned when COM port is successfully connected.

- False is returned when COM port fails to connect.

  - Reason for not connecting properly is usually because COM is already accessed

Example :

```
import MMC_PyLibrary as pyMMC

connect = pyMMC.openComPortMMC("COM6")

print(connect)
```

Library Functions

- connectEthMMC (string serverName, string serverPort)

Function :

Connects through the Ethernet using TCP Protocol
Server Address and Port as defined by the settings of the MMC Ethernet Module

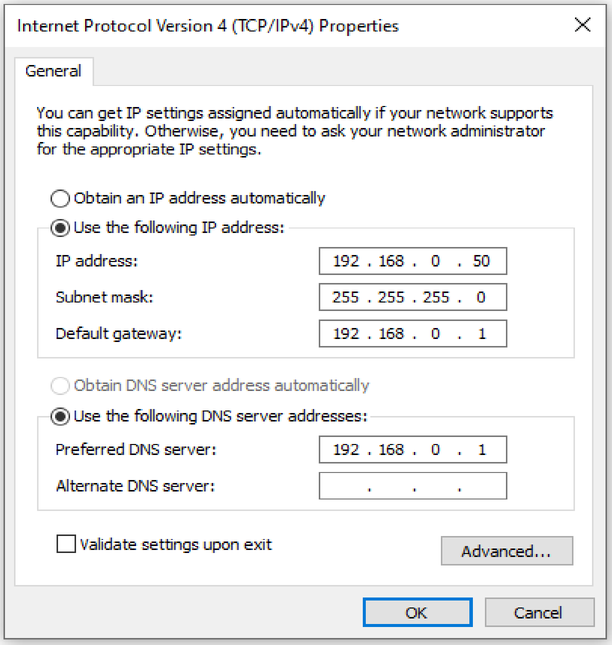
Return Value :

bool output (True or False)

- True is returned when Ethernet is successfully connected.

- False is returned when Ethernet fails to connect.

Make sure the Ethernet Property Settings Internet Protocol Version 4 (TCP/IPv4) are:



Internet Protocol Version 4 (TCP/IPv4) Properties

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically
● Use the following IP address:

IP address: 192 . 168 . 0 . 50
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 1

○ Obtain DNS server address automatically
● Use the following DNS server addresses:

Preferred DNS server: 192 . 168 . 0 . 1
Alternate DNS server: . . .

☐ Validate settings upon exit

Example :

```
import MMC_PyLibrary as pyMMC

connect = pyMMC.connectEthMMC("192.168.0.20", "5000")

print(connect)
```

Library Functions

- pollValues (string axis, string command)

Function :

Allows communication with the controller to query commands
Gives a response based on the command supported by the controller

Note: Please refer to MMC Manual for complete list of command with response

Return Value :

String Output Value

If Successful, Response value for the query
If failed, returns
- False, If no connection done yet
- Return Issue if connection done but failed to query

Note: This function works with both serial and ethernet connection and depending on the connection done it selects the mode of communication, if both are connected then the first preference is given through serial connection (USB ComPort).

Example :

```
poll1 = pyMMC.pollValues("1", "ver")

print(poll1)

poll2 = pyMMC.pollValues("2", "ver")

print (poll2)
```

Output :

```
MMC-ETHERNET V2.1

MMC-200 V1.4.23
```

Library Functions

- writeCommandToMMC (string axis, string command)

Function :

Function allows to send command which do not have a response

Return Value:

No return Value

Example

```
pyMMC.writeCommandToMMC("1", "enc50")

pyMMC.writeCommandToMMC("2", "enc50.20")
```

Library Functions

- closeConnectionMMC ( )

Function :

Close the connection for both serial and/or ethernet

Return Value:

bool value (True or False)
- True, if closed successfully
- False, if to close connection failed

Example :

```
closed = closeConnectionMMC( )

print(closed)
```