PythonAPI　変更点
Python のソースコード(.py)から Windows DLL(.pyd)に変更しました。
抵抗測定が可能になりました。

変更になった関数(create,data,)
追加された関数(setup_unit,setup_cal,nums,start_impd,stop_impd,data_impd)

詳細はサンプルプログラムを参考にしてください。
(¥py37¥sample.py　¥py37sample_plot.py　¥py37¥sample.py)

・create 関数の変更

nDIC = polymini.create(nSampling, nChannel)
↓
hPM　= pp.create(nSampling, nChannel, nBuffer)

引数に受信バッファーの追加
nSampling : サンプリング周波数、
nChannel　: チャンネル数
nBuffer　: 受信バッファー

・data 関数の変更

nDATAs, nNums　= polymini.data(nDIC, nFrameNums, nTimeOut)
↓
nDATAs, nNums = pp.data(hPM, nFrameNums)

タイムアウトの削除
nDIC(hPM)　: Create で取得した変数
nFrameNums : 取得するフレーム数
nTimeOut　: タイムアウト

・setup_unit 関数の追加

単位ラベルの設定
pp.setup_unit(hPM, nChannel , nLabel)

hPM　: Create で取得した変数
nChannel　: チャンネル番号

nLabel ： 単位ラベル

・setup_cal 関数の追加

キャリブレーション用換算式の作成

pp.setup_cal(hPM,　nChannel , (0, -400),(4000, 400), nLabel)

hPM ：Create で取得した変数

nChannel ： チャンネル番号

nDataValue1 ： データ値 1

nVoltageValue1 ： 電圧値 1

nDataValue2 ： データ値 2

nVoltageValue2 ： 電圧値 2

nLabel ： 単位ラベル

例）pp.setup_cal(hPM,　9, (0, -400),(4000, 400), 'mV')

0mV のとき -400、 4000mV のとき 400 になるように換算式を作成する

・nums 関数の追加

取り込みデータ数の取得

nNums = pp.nums(hPM)

hPM ：Create で取得した変数

・start_impd 関数の追加

抵抗測定の開始

pp.start_impd(hPM)

hPM ：Create で取得した変数

・stop_impd 関数の追加

抵抗測定の中止

pp.stop_impd(hPM)

hPM ：Create で取得した変数

・data_impd 関数の追加

抵抗値の取得

nIMPD, nResult = pp.data_impd(hPM)

hPM ：Create で取得した変数