var.py

The file called “var.py” is quite simple, it contains all the required variables and registers to configure the LMP91000 chip (hereinafter, LMP). It also opens the I2C (used to send commands to LMP) and SPI (to fetch data from the ADC of the LMP EVM) communications.

Some comments:

SPI Config, in the RPi 3, you may need to use Spi 1 (1,2). We will look into this when you receive the RPi and we will be trying to make it work.

```
spi = spidev.SpiDev()
spi.open(0,0)
spi.mode = 1
spi.max_speed_hz=1000000
```

Open I2C channel:

```
bus = smbus.SMBus(1)
address = 0x48
```

settings.py

This file contains some useful functions that will be used in the main program.

Some clarifications:

`write()` function sends a certain value of a register to the LMP.

`readacd()` function obtain the raw value from the ADC of the LMP EVM. Note that the first 3 bits are discarded, according to the ADC datasheet (`bin_r = bin_r[2:18]`)

`Status()` merely prints the status of the LMP.

cvgit.py

This is the main routine, which should be executed from the terminal as “python cvgit.py”, to initialize the GUI. The library used to plot the GUI is Tkinter. There are several elements comprising the GUI, such as Entry, Label, ProgressBar, Grid, among other. Recommended to look at the Tkinter library user manual.

Once initialized the GUI, the cvgit.py file executes the plotting are, whereby the current response will be plotted. The list of numbers called data can be deleted, it was merely an example to plot in the plotting area without using the device.

The smooth function can be used to reduce the noise level. It is up to you to use it or not.

When you push the start button, you call the main function of the program routine, called startCV(). GPIOs 11 and 13 are used with some leds, this part can be removed.

```
TIA = TIA_dicc["{}".format(variable_TIA.get())]
```

This code fetches the option selected from the GUI, and tells you the value to be used in your analysis.

Then, the code `DATA = sweep(TIA,OPMODE)` starts the potential sweep, which varies over time the potential applied onto the electrochemical cell, then the results are stored, and some functions are used to save the graph or the data in some csv files.

The sweep() function initialize the config of the LMP as well as apply a given technique. Note that, for instance, in the fixed voltage, you only have to call the step() function using as a parameter the corresponding REFCN register (see LMP datasheet). The plotting area is updated by calling the update() function.

The lines related with time library are aimed at providing exactly 1 second between samples.

The main file to be called to start running the programme (or GUI) is CVGIT. This one, in turn, includes both settings.py and var.py. It can be said that settings.py is high level and provides you with some specific functions. Instead, var.py is a low-level file, in which variables etc are contained.

.desktop files are aimed at providing you with an executable file in the desktop, you can remove it and simply run from the terminal of Raspbian. Note finally that there is also available a brief user manual.