Lab 1 Code and discussions

Objective

Introduction to the controller unit and set up a Python Development Environment for writing programs that would be used to control the audio amplifier

Installed Packages

Python

notebook

ipywidgets

pyserial - enables python programs to communicate with devices via serial ports

numpy - data processing and analysis

plotly - visualization

Identifying Serial Ports

In [ ]:
```
import serial
import serial.tools.list_ports

ports = serial.tools.list_ports.comports() #contains devices connected to all se
```

Open communication channel with the controller

In [ ]:
```
VID = 61525 #vendor ID of the device
PID = 38912 #product ID of the device

for p in ports:
    if p.vid == VID and p.pid == PID:
        try:
            device=serial.Serial(p.device)
        except serial.SerialException: #raised if the device is not available
            print('Reconnect the controller unit')
    if device is None:
        raise Exception('No suitable device detected.') #if no matching device f
```

In [ ]:
```
print(device) #check the connection information
```

```
Serial<id=0x2152d8e38e0, open=True>(port='COM3', baudrate=9600, bytesize=8, parit
y='N', stopbits=1, timeout=None, xonxoff=False, rtscts=False, dsrdtr=False)
```

Simple microPython Commands using write() function

In [ ]:
```
device.write(bytes('pyb.LED(1).toggle()\r','utf-8')) #toggle the status of the D
```

Out[ ]: 20

bytes() - converts the string into bytes using the UTF-8 encoding

*string must end with '\r' which marks the end of the command string and needed for proper registration

write() - passes it to the controller unit

Adjusting DC supply voltage

5 Volts

In [ ]:
```
device.write(bytes('volt=5\r','utf-8'))
device.write(bytes('from pyb import Pin\r','utf-8'))
device.write(bytes('from machine import SPI\r','utf-8'))
device.write(bytes('spi = SPI(sck=Pin(\'PB13\',Pin.OUT), mosi=Pin(\'PB15\', Pin.
device.write(bytes('dz=Pin(\'PB12\', Pin.OUT)\r','utf-8'))
device.write(bytes('y=312-1020/volt\r','utf-8'))
device.write(bytes('dz.value(0)\r','utf-8'))
device.write(bytes('spi.write(b\'\\x11\')\r','utf-8'))
device.write(bytes('spi.write(bytes((int(y),)))\r','utf-8'))
device.write(bytes('dz.value(1)\r','utf-8'))
```

Out[ ]: 12

12 Volts

In [ ]:
```
device.write(bytes('volt=12\r','utf-8'))
device.write(bytes('from pyb import Pin\r','utf-8'))
device.write(bytes('from machine import SPI\r','utf-8'))
device.write(bytes('spi = SPI(sck=Pin(\'PB13\',Pin.OUT), mosi=Pin(\'PB15\', Pin.
device.write(bytes('dz=Pin(\'PB12\', Pin.OUT)\r','utf-8'))
device.write(bytes('y=312-1020/volt\r','utf-8'))
device.write(bytes('dz.value(0)\r','utf-8'))
device.write(bytes('spi.write(b\'\\x11\')\r','utf-8'))
device.write(bytes('spi.write(bytes((int(y),)))\r','utf-8'))
device.write(bytes('dz.value(1)\r','utf-8'))
```

Out[ ]: 12

Closing Connection

In [ ]:
```
device.close() # close connection
```

Open Ended Questions - What is SPI

SPI

Serial Peripheral Interface

Communcation protocal for synchronous *serial communication* for microcontroller

sck, mosi, miso, SPI(), PIN(), Pin.OUT, Pin.IN

sck - serial clock, generated by the master

mosi - Master to Slave, Master-Out-Salve-In, serial data transmitted from Master to Slave

miso - Slave to Master, Master-In-Slave-Out, the serial data transmitted by a Slave and sent to the Master

SPI() - Initialise the SPI Bus with parameters

PIN() - Access the pin peripheral (GPIO pin) associated with the given id. Additional arguments are used to initialise the pin.

Pin.OUT - Sets that pin as an Output

Pin.IN - Sets that pin as an input