

# UPS-Lite 使用说明

for Raspberry Pi Zero W

-- by XiaoJ



# 1 Introduction

UPS-Lite is a specially Raspberry Pi Zero (hereinafter referred to as pi) designed UPS, a lithium polymer battery using the power supply is 1000mAh, the external power source insertion detecting support, support through charging discharge, both external plug when power, pi by an external power supply, external power supply when unplugged, pi transferred by a lithium battery. UPS-Lite by 10 pi plunger and connected to the board, all without the use of a pi usb data lines connected with UPS-Lite. In addition UPS-Lite also integrates professional fuel gauge chip MAX17040G, USB-to-UART serial chip CP2104, charge indicator.



parameter:

recharging current: **max 400mA @ 5V**

Output current: **max 1.3A@5V** ( In the case where the external power supply is not inserted, the battery only)

**max 2A @ 5V** ( In the case inserted into the external power source)

Power measurement: percentage of battery power output, an error  $\pm 2\%$ , the measurement errors of voltages  $\pm 3mV$

## 2. Install

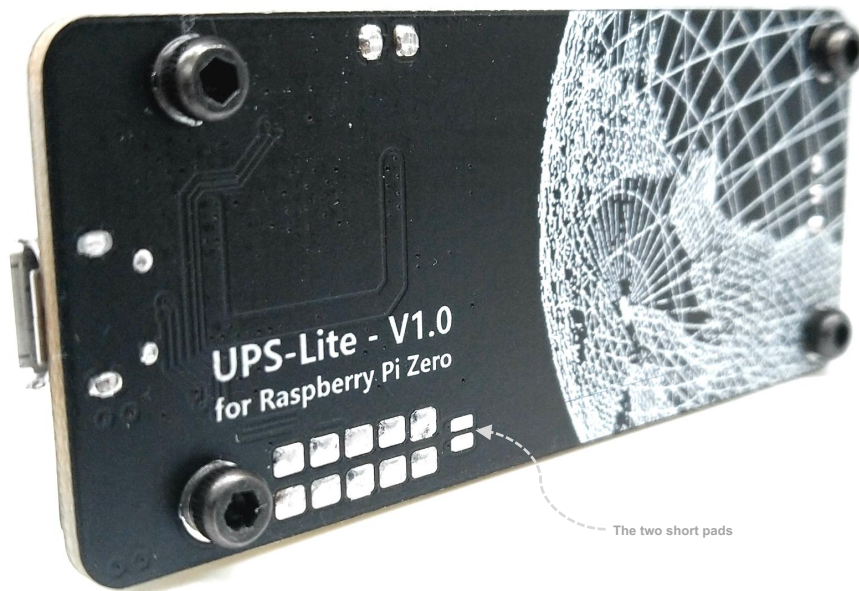
The four screw holes are aligned with the four screws into pi, supporting the lock nut can be. Note, pi need to weld the pin to use UPS-Lite. Since pi is connected to the UPS-Lite is by contact of the plunger and pin to achieve.



## 3. Use function

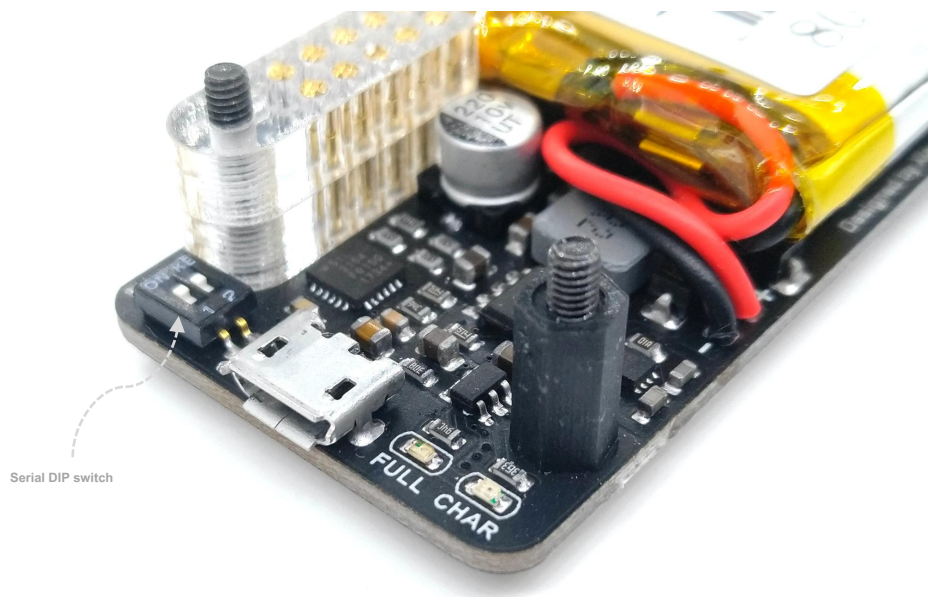
### 3.1 Charging power supply insertion detecting function

Recommended power to the UPS-Lite and the above charging 5V2A power adapter. Because when the lithium battery is low, the external power supply adapter pi not give further desirable to provide part of the current supply lithium battery, the charging process, the red indicator light indicating that the battery is charging. When lithium battery is fully charged, the green indicator light, red light is off. Also UPS-Lite insertion detecting function with a power adapter, the insertion of the power pi io4 (BCM number) detects the high level, when pulled low, enabling the weld shorting function requires two back of the UPS disc, as shown below in detail.



### 3.2 Serial function

The DIP switches are all appropriated "ON", PC chip CP2104 tops of the serial port driver, usb pi plug wire connected to the PC, open software corresponding serial PuTTY as the PC, and then open the UPS-Lite switch to the power output pi power pi of the case can communicate through the serial port.



### 3.3 Battery measurement function

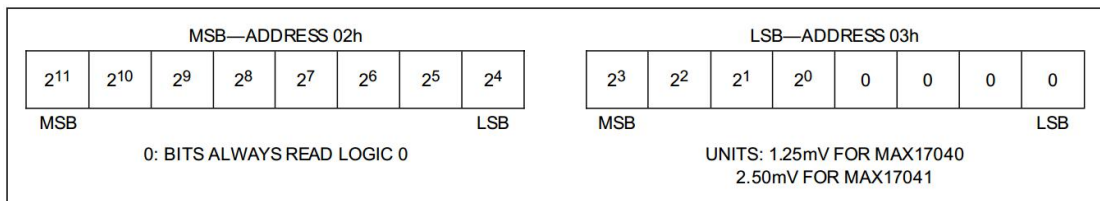
Create a new name with the nano editor for UPS\_Lite.py the script, the code is as follows (see Appendix detailed code). This script uses Python's library of MAX17040G smbus be i2c read. MAX17040G device address 0x36, is a 12bit register VCELL ADC battery voltage measurement value, address 02h-03h, ADC precision measurement unit is 1.25mV. 16bit register SOC is a percentage of battery capacity reading, address 04h-05h, SOC high 8bit units of 1% of the battery capacity, low 8bit units of 1/256%, the percentage of battery capacity to provide readings of decimal places. They know to save a script under UPS\_Lite.py directory (as described below to save / home pi // directory), and then run the program with Python, every two seconds can be seen that the program will output the current value of the battery voltage and the percentage of the battery capacity. In addition, as calculated MAX17040G battery capacity, the battery capacity when the reading of 1%, and the reading of the battery voltage is about 3.5V. Lithium is generally due to voltage 3.5V, corresponding to the battery capacity has been

When very low, excessive discharge may damage the battery, the user subsequent preparation of low battery automatic shutdown procedure, the battery capacity is recommended automatically closed pi 1%, if it continues to run, when the battery voltage drops to 3.0V, UPS-Lite It will automatically stop the power supply. Specific steps below

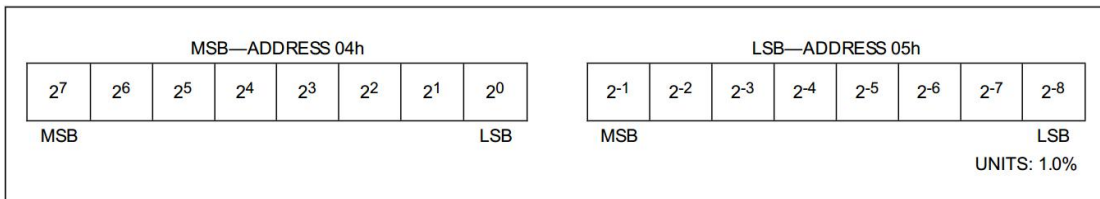
MAX17040G register address and Features

ADDRESS (HEX)	REGISTER	DESCRIPTION	READ/ WRITE	DEFAULT (HEX)
02h–03h	VCELL	Reports 12-bit A/D measurement of battery voltage.	R	—
04h–05h	SOC	Reports 16-bit SOC result calculated by ModelGauge algorithm.	R	—
06h–07h	MODE	Sends special commands to the IC.	W	—
08h–09h	VERSION	Returns IC version.	R	—
0Ch–0Dh	RCOMP	Battery compensation. Adjusts IC performance based on application conditions.	R/W	9700h
FEh–FFh	COMMAND	Sends special commands to the IC.	W	—

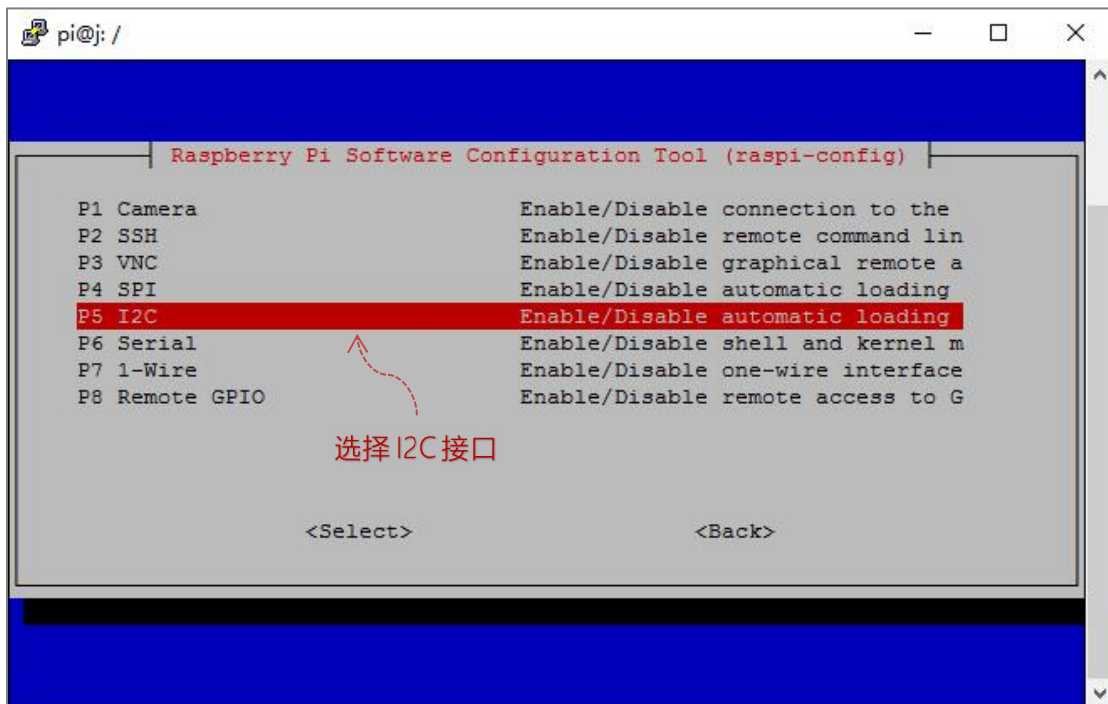
VCELL register



SOC register

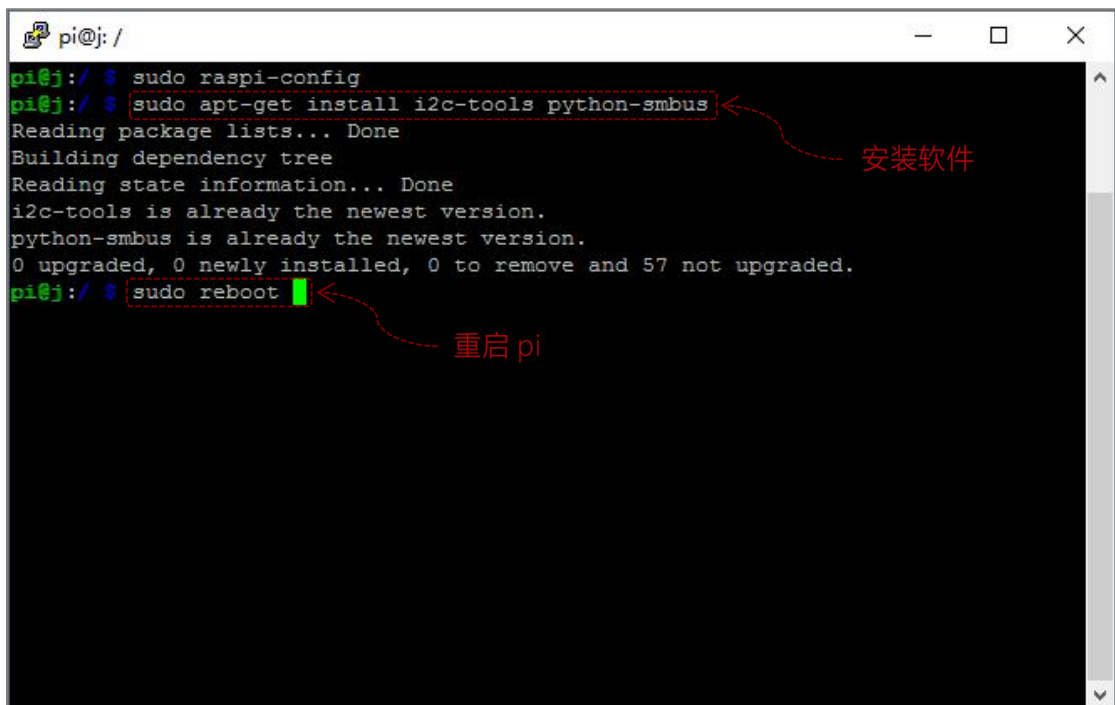


a. Open pi configuration tool raspi-config, enable I2C interface





b. Install i2c-tools and python-smbus, after the installation is complete restart the pi





c. Run `sudo i2cdetect -l` to view the current pi which uses i2c bus.

```
pi@j: /  
pi@j: / $ sudo i2cdetect -l  
i2c-1 i2c bcm2835 I2C adapter I2C adapter  
pi@j: / $
```

运行 i2cdetect -l 查看 i2c 总线

确定系统 i2c 总线为 1

d. Run `view sudo i2cdetect -y 1` currently mounted on the pi i2c bus devices.

```
pi@j: /  
pi@j: / $ sudo i2cdetect -l  
i2c-1 i2c bcm2835 I2C adapter I2C adapter  
pi@j: / $ sudo i2cdetect -y 1  
 0 1 2 3 4 5 6 7 8 9 a b c d e f  
00: -- -- -- -- -- -- -- -- -- -- -- --  
10: -- -- -- -- -- -- -- -- -- -- -- --  
20: -- -- -- -- -- -- -- -- -- -- -- --  
30: -- -- -- -- 36 -- -- -- -- -- -- --  
40: -- -- -- -- -- -- -- -- -- -- -- --  
50: -- -- -- -- -- -- -- -- -- -- -- --  
60: -- -- -- -- -- 68 -- -- -- -- -- --  
70: -- -- -- -- -- -- -- -- -- -- -- --  
pi@j: / $
```

运行 i2cdetect -l 查看 i2c 总线

地址 0x36 为 MAX17040G

e. the new script by UPS\_Lite.py nano editor (see appendix detailed code)

```
pi@j: ~
GNU nano 2.7.4 File: UPS_Lite.py
#!/usr/bin/env python
import struct
import smbus
import sys
import time

def readVoltage(bus):
    """This function returns as float the voltage from the Raspi UPS Hat via I2C"""
    address = 0x36
    read = bus.read_word_data(address, 2)
    swapped = struct.unpack("<H", struct.pack(">H", read))[0]
    voltage = swapped * 1.25 /1000/16
    return voltage

def readCapacity(bus):
    """This function returns as a float the remaining capacity of the battery via I2C"""
    address = 0x36
    read = bus.read_word_data(address, 4)
    swapped = struct.unpack("<H", struct.pack(">H", read))[0]
    capacity = swapped/256
    return capacity

bus = smbus.SMBus(1) # 0 = /dev/i2c-0 (port I2C0), 1 = /dev/i2c-1 (port I2C1)

while True:

    print "+++++"
    print "Voltage:%5.2fV" % readVoltage(bus)

    print "Battery:%5i%%" % readCapacity(bus)

    if readCapacity(bus) == 100:

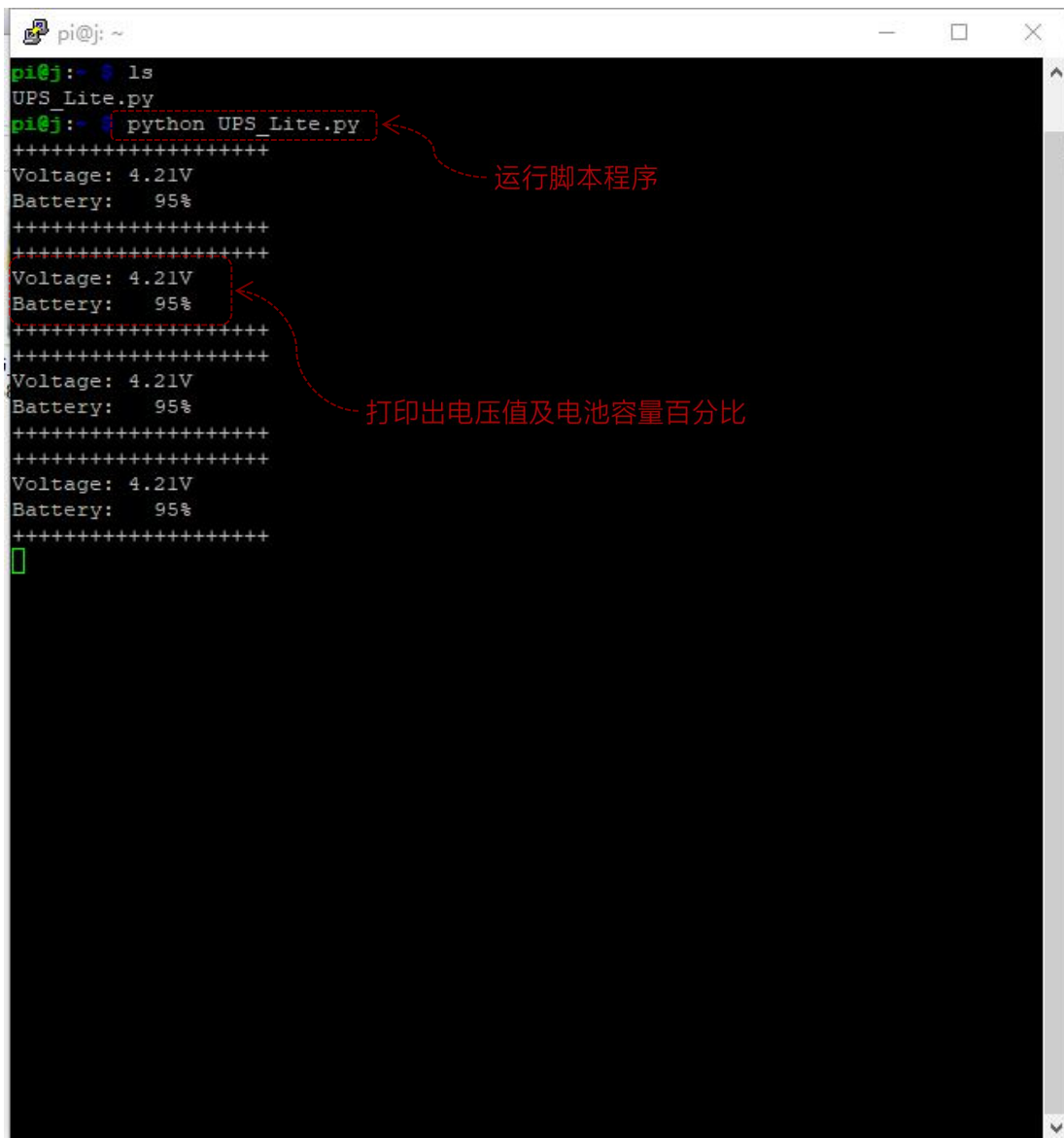
        print "Battery FULL"

    if readCapacity(bus) < 20:

        print "Battery LOW"
    print "+++++"
    time.sleep(2)

[ Unbound key: M-^A ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Linter ^_ Go To Line
```

f. Run the script in Python



```
pi@j: ~  
pi@j:~$ ls  
UPS_Lite.py  
pi@j:~$ python UPS_Lite.py  
+++++  
Voltage: 4.21V  
Battery: 95%  
+++++  
+++++  
Voltage: 4.21V  
Battery: 95%  
+++++  
+++++  
Voltage: 4.21V  
Battery: 95%  
+++++  
+++++  
Voltage: 4.21V  
Battery: 95%  
+++++  
[
```

运行脚本程序

打印出电压值及电池容量百分比

## appendix:

### UPS\_Lite.py script program code:

```
#!/usr/bin/env python

import struct
import smbus
import sys
import time

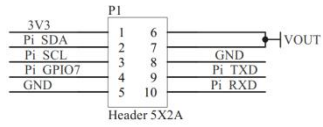
def readVoltage ( bus ):
    """This function returns as float Hat via the voltage from the Raspi UPS the provided SMBus object"""
    address = 0x36
    read = bus.read_word_data ( address , 2 )
    swapped = struct.unpack ( "<H" , struct.pack ( ">H" , read )) [ 0 ]
    voltage = swapped * 1.25 / 1000 / 16
    return voltage

def readCapacity ( bus ):
    """This function returns as a float the remaining capacity of the battery connected to the Raspi UPS
    Hat via the provided SMBus object """
    address = 0x36
    read = bus.read_word_data ( address , 4 )
    swapped = struct.unpack ( "<H" , struct.pack ( ">H" , read )) [ 0 ]
    capacity = swapped / 256
    return capacity

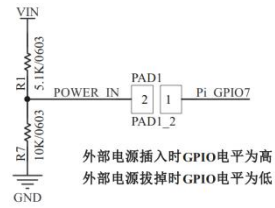
bus = smbus.SMBus ( 1 ) # 0 = /dev/i2c-0 (port I2C0), 1 = /dev/i2c-1 (port I2C1)

while True:
    print "+++++"
    print "Voltage:% 5.2fV" % readVoltage ( bus )
    print "Battery:% 5i %" % readCapacity ( bus )
    if readCapacity ( bus ) == 100 :
        print "Battery FULL"
    if readCapacity ( bus ) < 20 :
        print "Battery LOW"
    print "+++++"
    time.sleep ( 2 )
```

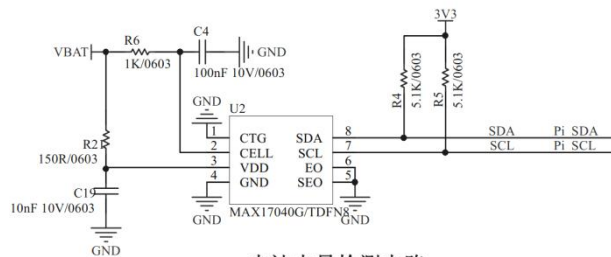
Schematic Reference Part:



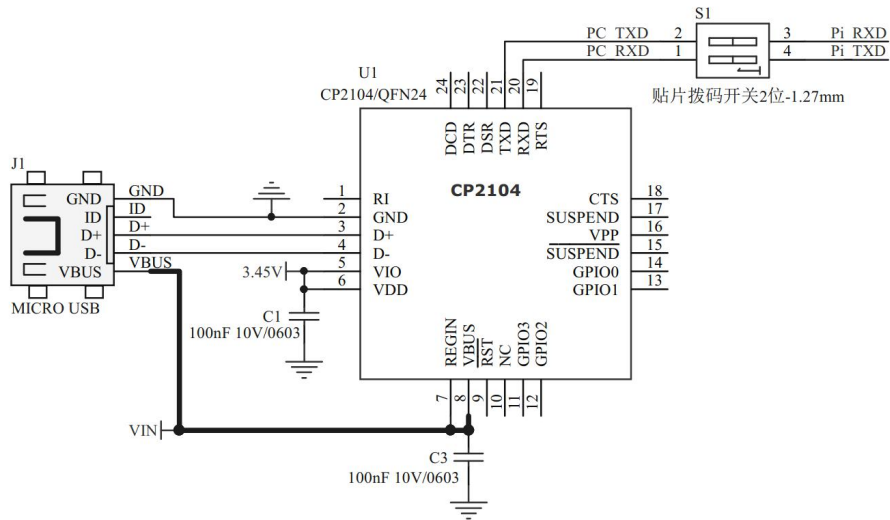
树莓派顶针接口



外部电源插入检测



电池电量检测电路



USB转UART

**References:**

**Raspberry Pi study notes - I2C Tools Study Notes - CSDN Blog**

<http://blog.csdn.net/xukai871105/article/details/15029843>

**MAX17040 Compact low cost 1S / 2S Fuel Gauge - Maxim Maxim**

<https://www.maximintegrated.com/cn/products/power/battery-management/MAX17040.html>

**CP2104 Driver Download USB to UART Bridge VCP Drivers | Silicon Labs**

<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

**contact me:**



Sweep the micro-channel

感謝