Safe-power Raspberry Pi UPS datasheet

TG Tronx 30/06/2019

Technical data	
max current while not charging Lipo cell	2 mA
max current charging Lipo cell	250 mA
Temperature range	$0\,^{\circ}$C - $50\,^{\circ}$C
rechargeable Battery	Lipo 3.7V
time to shutdown after power failure	10s

Table 1: max ratings

Dimensions

The UPS comes in HAT form, specified by https://github.com/raspberrypi/hats but does not have the eeprom.

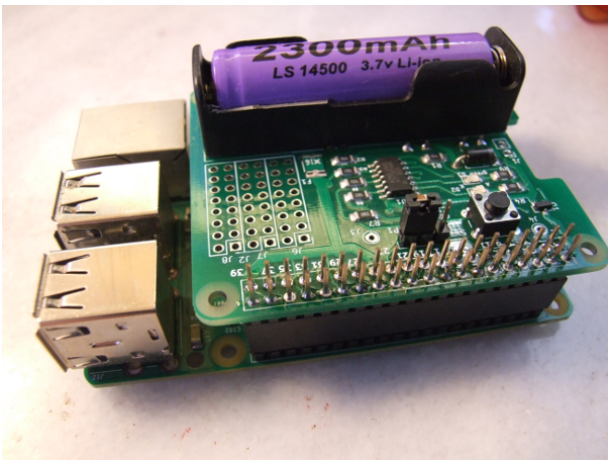


Figure 1: safe-power installed on Raspberry 2

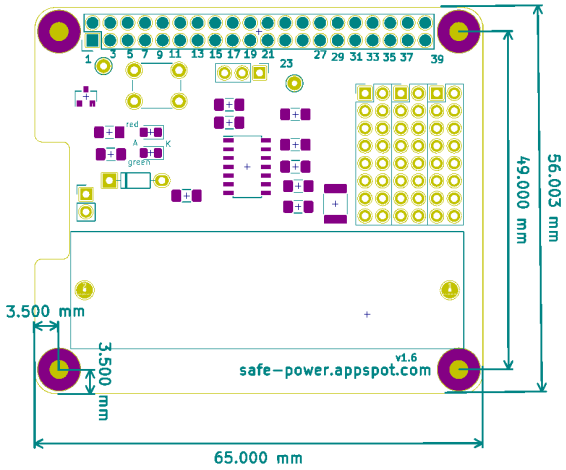


Figure 2: shield specification

onboard Lipo charging circuit

The TP4056 is a complete constant-current/constant-voltage linear charger for single cell lithium-ion batteries.
ABSOLUTE MAXIMUM RATINGS
· Preset 4.2V Charge Voltage
·Input Supply Voltage(V CC):-0.3V 8V
·BAT Short-Circuit Duration:Continuous
·BAT Pin Current:200mA
TP4056 Other features include current monitor, under voltage lockout, automatic recharge and two status LED to indicate charge termination and the presence of an input voltage.

LED codes:

Steady green – power has been applied, Raspberry boots
Blinking green 2 seconds – normal operation power ok
Blinking red fast – power failure detected
Steady red – shutdown initiated (manual or after power failure)
Blinking red 2 seconds – power failure, Raspberry is shutdown
Blinking red and green 2 seconds – system in shutdown After manual shutdown by button
Blinking red and green alternating 5 times – Safe-power Microcontroller boots

shutdown operation

Save this script in /bin and execute it via crontab at boot time.

@reboot /bin/safe-power.py &

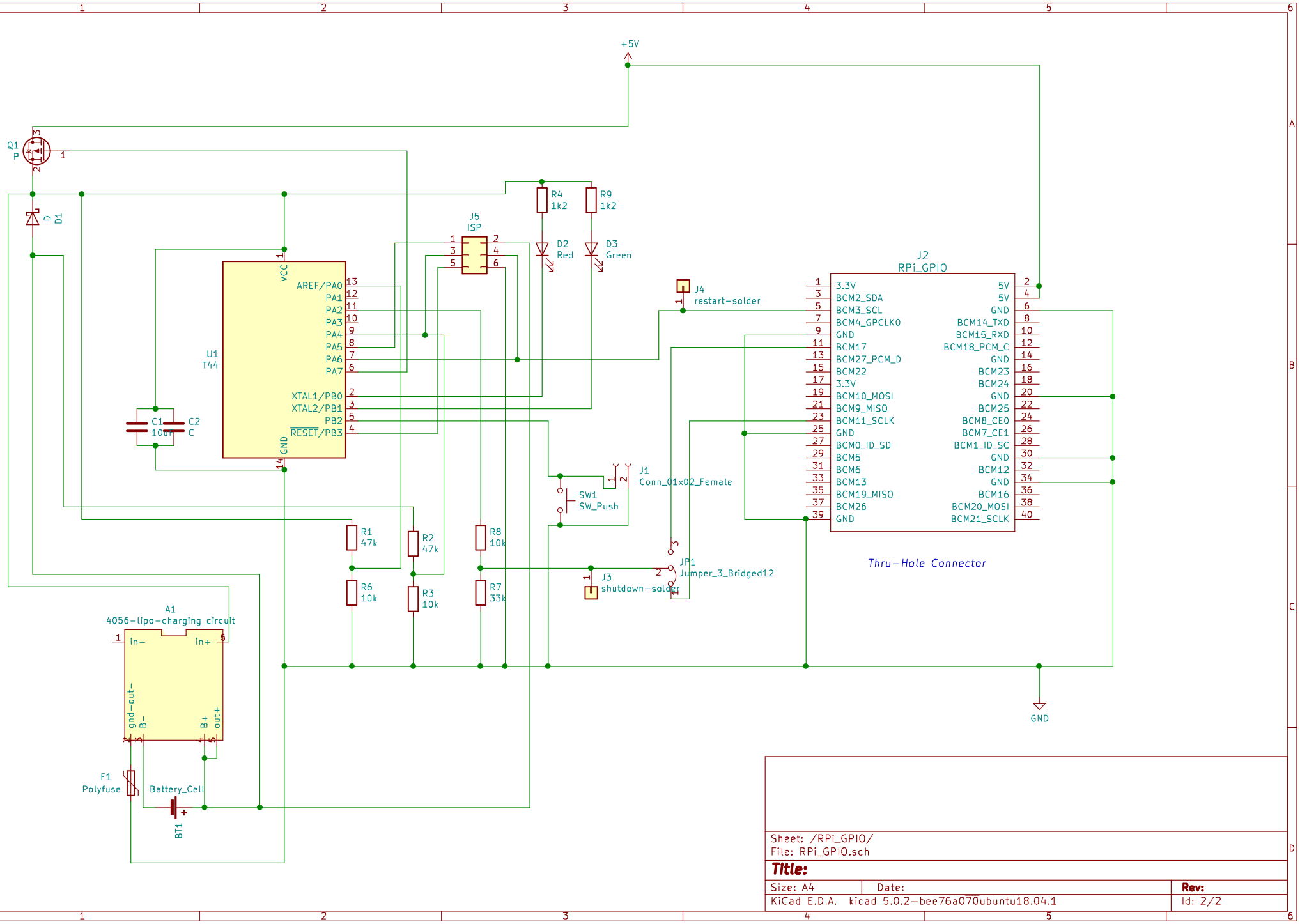
```
#!/usr/bin/env python
#script to shutdown the raspberry by safe-power raspberry UPS
#save this script as root under /bin/safe-power.py
#add this script AS LAST LINE of root's crontab in the following way
#  @reboot /bin/safe-power.py &
# important!! dont forget the "&"  in the end!!
#the script will be started in the background at reboot
#and safe power will be operational
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
import os
import time
# GPIO 11 = pin23 set up as input. It is pulled up to stop false signals
GPIO.setup(11, GPIO.IN, pull_up_down=GPIO.PUD_UP)
# now the program will do nothing until the shutdown signal on pin 23
# gets LOW.
#During this waiting time, your raspberry is not
#wasting resources by polling the pin

try:
    GPIO.wait_for_edge(11, GPIO.FALLING)

# warn all logged users of the shutdown event
    os.system("wall shutdown by UPS")
#now the system will shut down
    os.system("sudo poweroff")
#except if this script will be cancelled by the user explicitely
except KeyboardInterrupt:
    GPIO.cleanup()       # clean up GPIO on CTRL+C exit
    GPIO.cleanup()           # clean up GPIO on normal exit
```

safe–power.py

Further installation instructions on http://safe-power.appspot.com/setup


+5V
Q1
P
D1
D
U1
T44
VCC
GND
AREF/PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
XTAL1/PB0
XTAL2/PB1
PB2
RESET/PB3
C1
10u
C2
C
R1
47k
R2
47k
R3
10k
R4
1k2
R6
10k
R7
33k
R8
10k
R9
1k2
J5
ISP
D2
Red
D3
Green
J4
restart-solder
J1
Conn_01x02_Female
SW1
SW_Push
J3
shutdown-solder
JP1
Jumper_3_Bridged12
J2
RPi_GPIO
3.3V
BCM2_SDA
BCM3_SCL
BCM4_GPCLK0
GND
BCM17
BCM27_PCM_D
BCM22
3.3V
BCM10_MOSI
BCM9_MISO
BCM11_SCLK
GND
BCM0_ID_SD
BCM5
BCM6
BCM13
BCM19_MISO
BCM26
GND
5V
5V
GND
BCM14_TXD
BCM15_RXD
BCM18_PCM_C
GND
BCM23
BCM24
GND
BCM25
BCM8_CE0
BCM7_CE1
BCM1_ID_SC
GND
BCM12
GND
BCM16
BCM20_MOSI
BCM21_SCLK
Thru-Hole Connector
A1
4056-lipo-charging circuit
in-
in+
gnd-out-
B-
B+
out+
F1
Polyfuse
Battery_Cell
BT1
GND
Sheet: /RPi_GPIO/
File: RPi_GPIO.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad 5.0.2-bee76a070ubuntu18.04.1
Id: 2/2