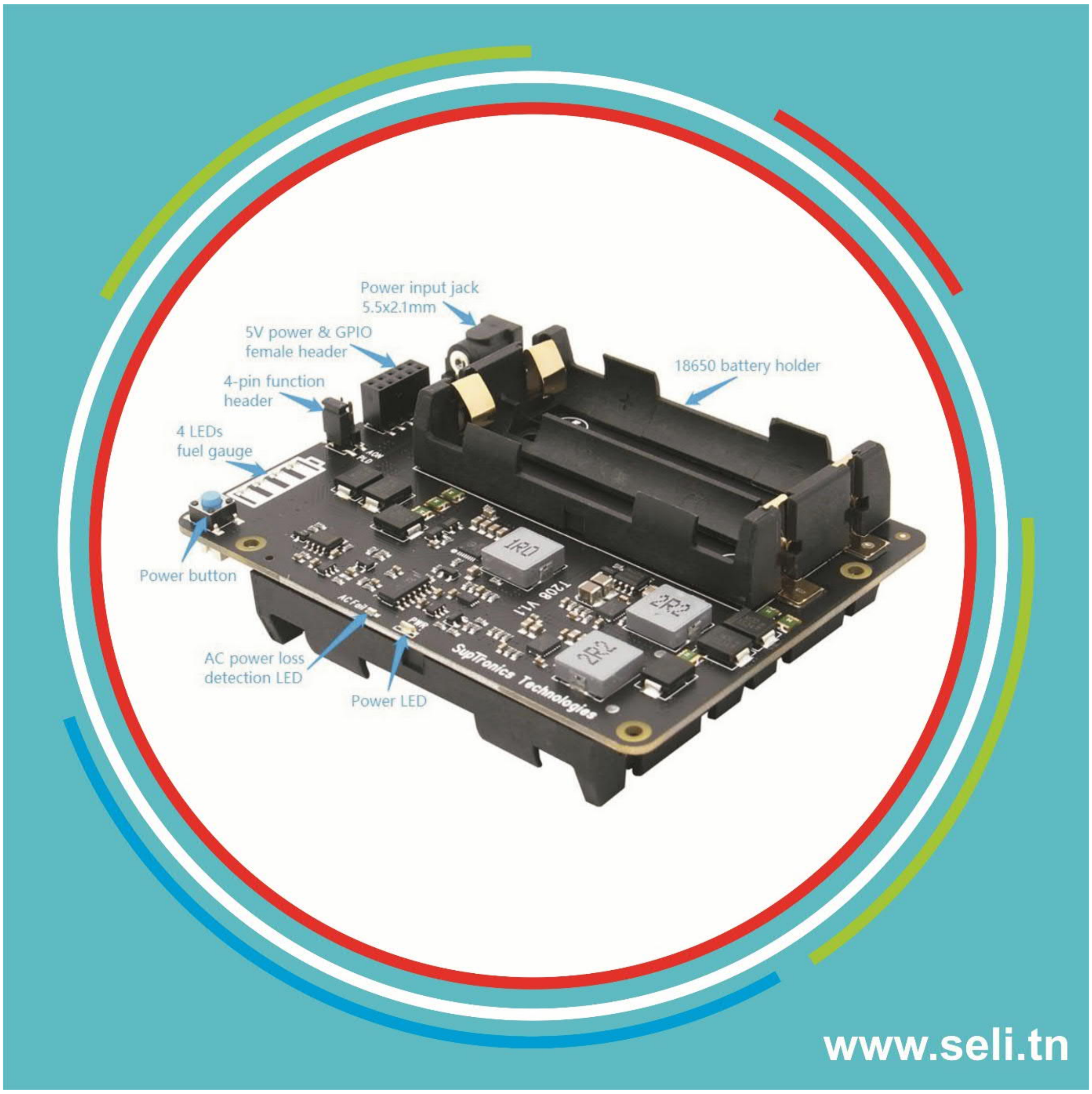
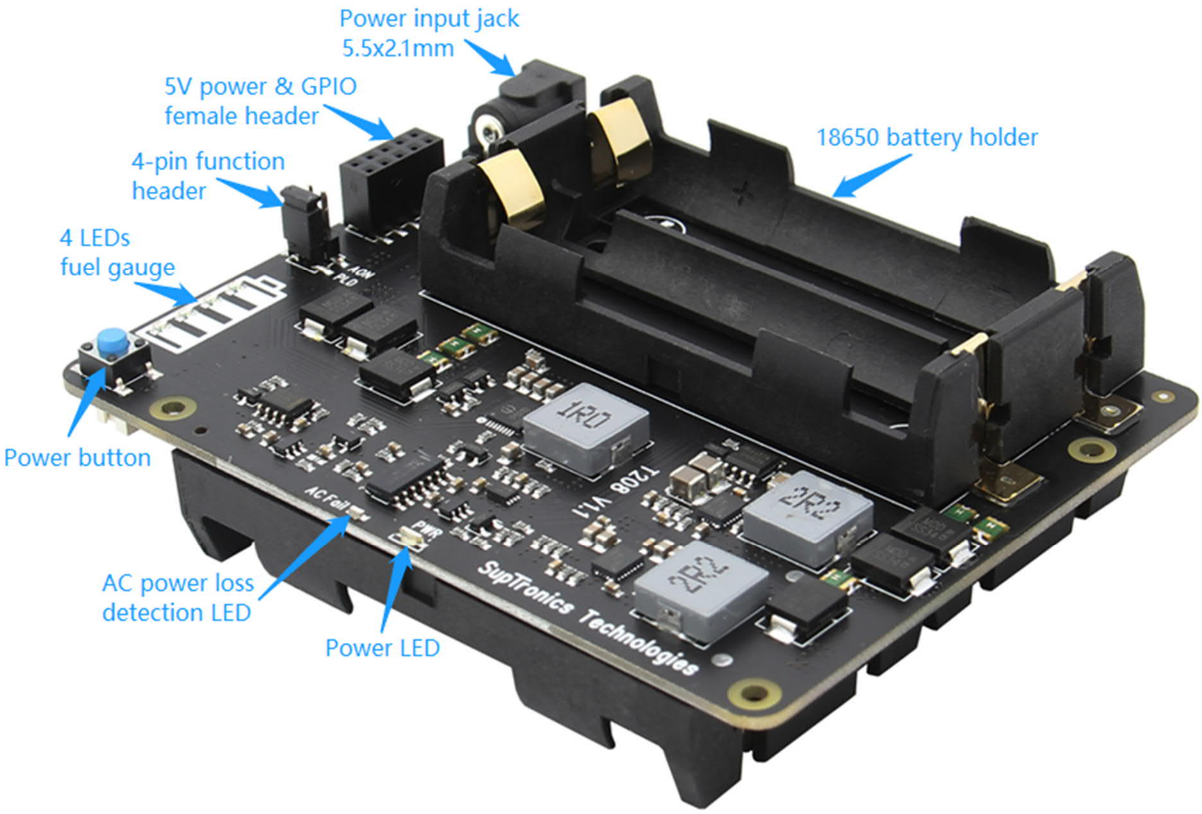
MOD503

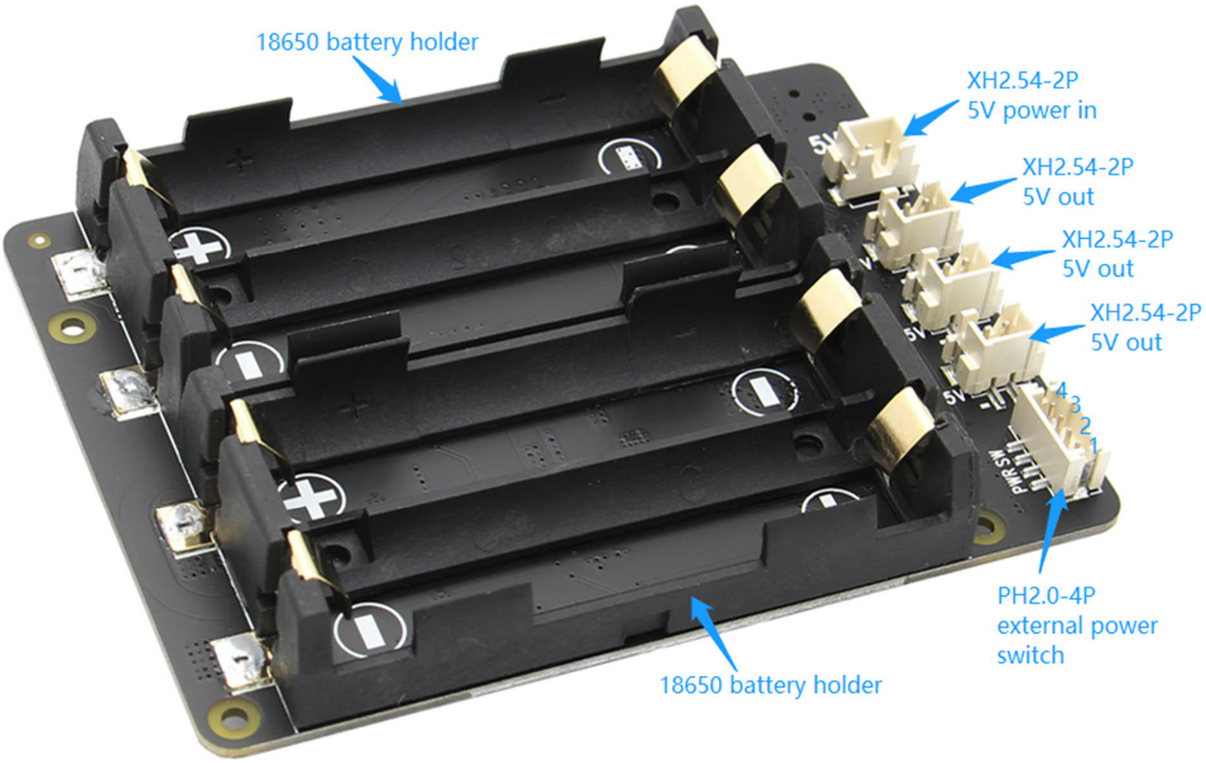
MODULE UPS POUR JETSON NANO ALIMENTATION SANS INTERRUPTION DFR0865



Introduction

Designed for Jetson Nano Developer Kit, this advanced uninterruptible Power supply UPS module provides up to 5.1V 8000mA backup power for Jetson Nano board. With intelligent power management, the module supports multiple functions like power loss/adapter fault detection, automatic startup, complete shutdown from software, advanced power path management, I2C bus communication and onboard LED indicators for monitoring battery capacity. It is powered by six 18650 batteries.





Specification

- Power input: 5Vdc±5%, ≥4A
- UPS output: 5.1V±5% maximum 8A
- UPS charging current: 3A (DC jack)
- Terminal battery voltage: 4.24V
- Charging threshold: 4.1V

Features

- Continuous operation for up to 10~30 hours (depending on the battery type and the amount of battery used)
- Provides a maximum of 5.1V 8000mA backup power
- Can work with 1/2/3/4/5/6 18650 lithium-ion batteries
- Support 3A fast battery charging
- Integrated battery protection circuit
- Integrated over-current protection and over-voltage protection
- Integrate Maxim's fuel gauge system (read battery voltage and percentage through i2c)
- Intelligent automatic charging and discharging
- On-board 4 green LEDs indicate battery charging and discharging levels of 25%, 50%, 75% and 100%
- The on-board blue LED indicator shows the power on/off status
- The on-board button can control the power on/off (short press the button-turn on, press and hold the button for at least 3 seconds-turn off)
- Power loss or power adapter fault detection (LED and GPIO indication)
- On-board XH2.54 power output connector for powering other devices
- Mounting holes and standoffs provide mechanical stability of Jetson Nano and T208 components

Safety Instructions and Warnings

- Lithium polymer and lithium-ion batteries are very unstable. When charging or using improperly, it may cause fire, personal injury and property damage.
- When charging and discharging the battery pack, do not misconnect polarities. Always check the polarity of the battery connector carefully to ensure "+" to "+" and "-" to "-".
- Do not mix or use old and new batteries, or use other brands of batteries.
- Lithium batteries have a limited cycle life. Please replace the old batteries with new ones when they reach the end of their service life or after two years of use (whichever comes first).
- Keep UPS away from children.

Function and Assembly

- Relationship between light display and power
- Connect with Jetson Nano by 10Pin

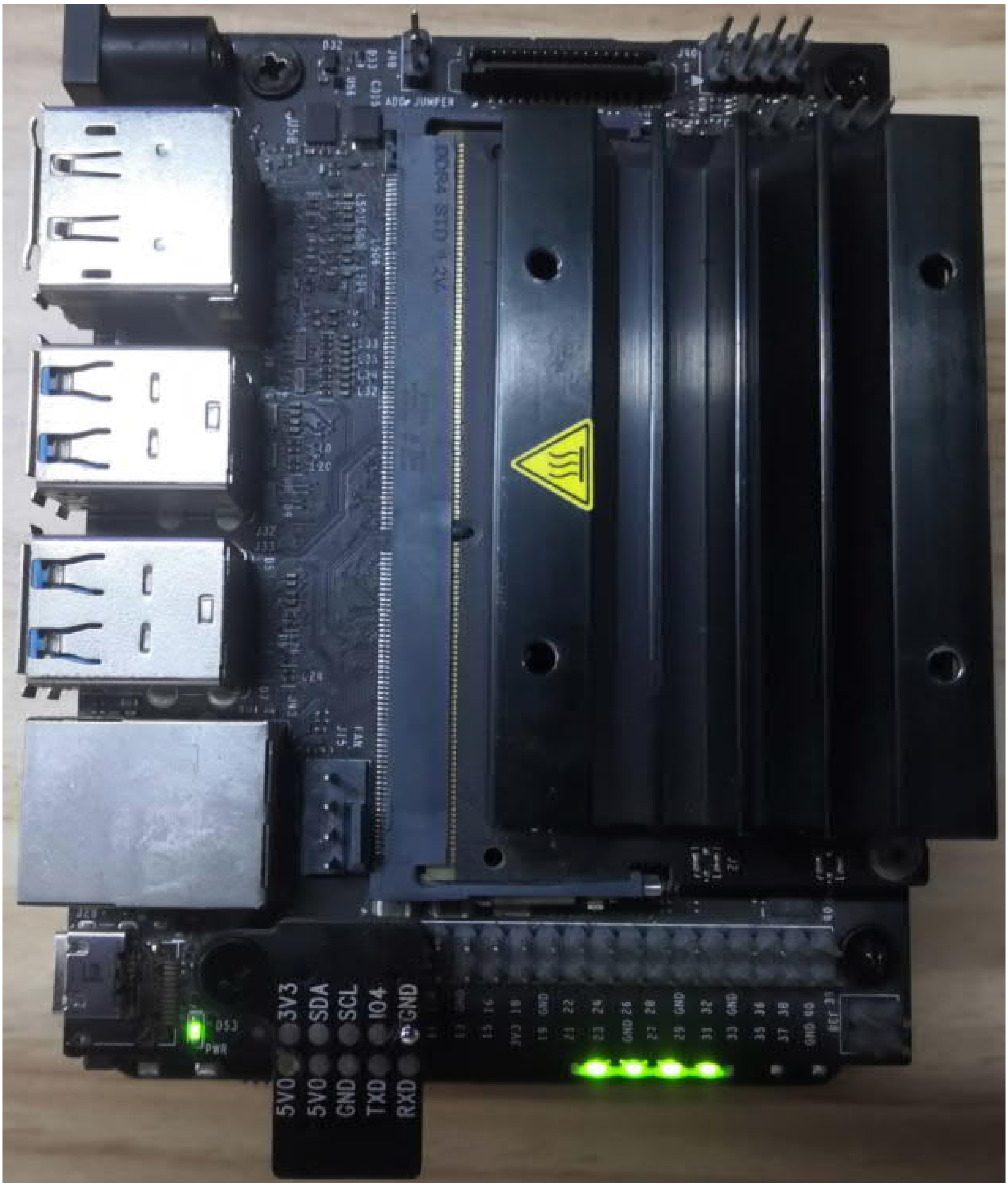
- On/off operation
- Battery power and voltage can be read by software

LEDs display and Electricity

Electricity	The number of LED lights on
0%~25%	1
26%~50%	2
51%~75%	3
76%~100%	4

Connect with Jetson Nano by 10Pin

Correctly install six 18650 batteries to the Jetson Nano UPS power board, and connect to Jetson Nano through the 10 Pin GPIO board. The connection diagram is as follows:



On/off operation

Long press for 3 seconds to turn off the UPS, short press once to turn on the UPS.

Battery power and voltage can be read by software

1. First, install the pip of Python3 on the Ubuntu system. Please run the following command as the root user or sudo user in the terminal:

```
sudo apt-get update
sudo apt-get install python-pip
```

2. Install Jetson GPIO library:

```
sudo pip install Jetson.GPIO
```

3. Install the I2C development library:

```
sudo apt-get install libi2c-dev i2c-tools
pip install smbus
```

4. Install Nano editor on Ubuntu:

```
sudo apt-get install nano
```

5. Sample code for reading battery voltage and battery percentage

```


#!/usr/bin/env python
import struct
import smbus
import sys
import time


def readVoltage(bus):

     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):

     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)
```

Create a new Bat.py file, and then run the command in the terminal:

```
sudo python Bat.py
```

Now you get the battery power percentage and battery voltage information.

FAQ