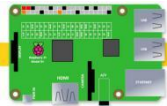

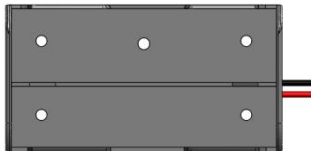



Lesson 13 Battery Level Detection and Alarm

13.1 Overview

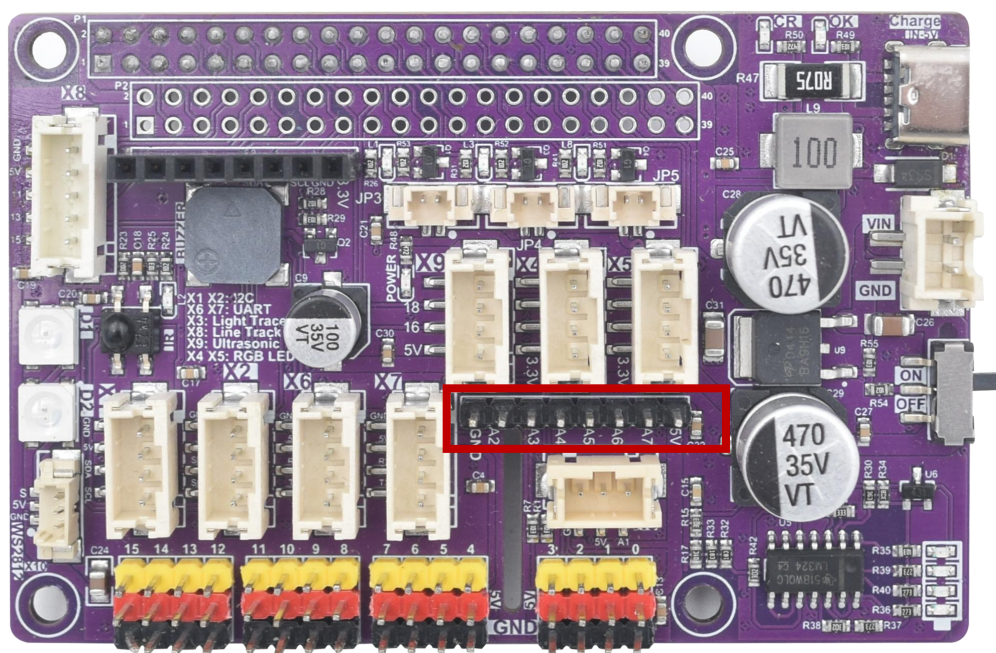
In this lesson, we will learn how to detect the battery level of a power source connected to an Adeept Robot HAT V3.2 using a Raspberry Pi. We'll also set up an alarm system that notifies us when the battery level drops below a certain threshold.

13.2 Required Components

Components	Quantity	Picture
Raspberry Pi	1	
Adeept Robot HAT V3.2	1	
18650 Battery Holder	1	
18650 Battery	2	

13.3 Principle Introduction

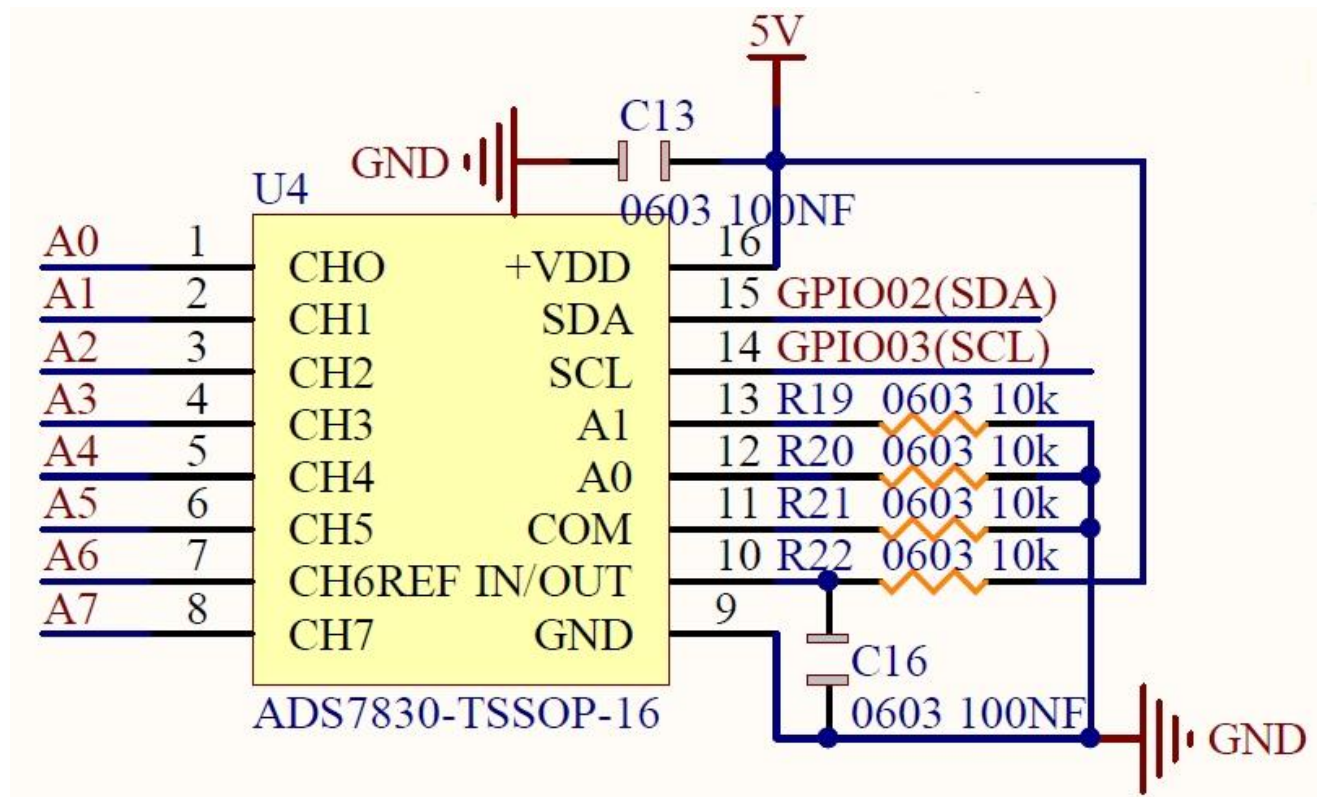
Adeept Robot HAT V3.2 expands 8 ADC pins from I2C pins through the ADS7830 chip. The ADC A1 pin is used for the Light Tracking interface, the A0 pin is used to detect battery power, and the other 6 pins are on the board, as shown below:

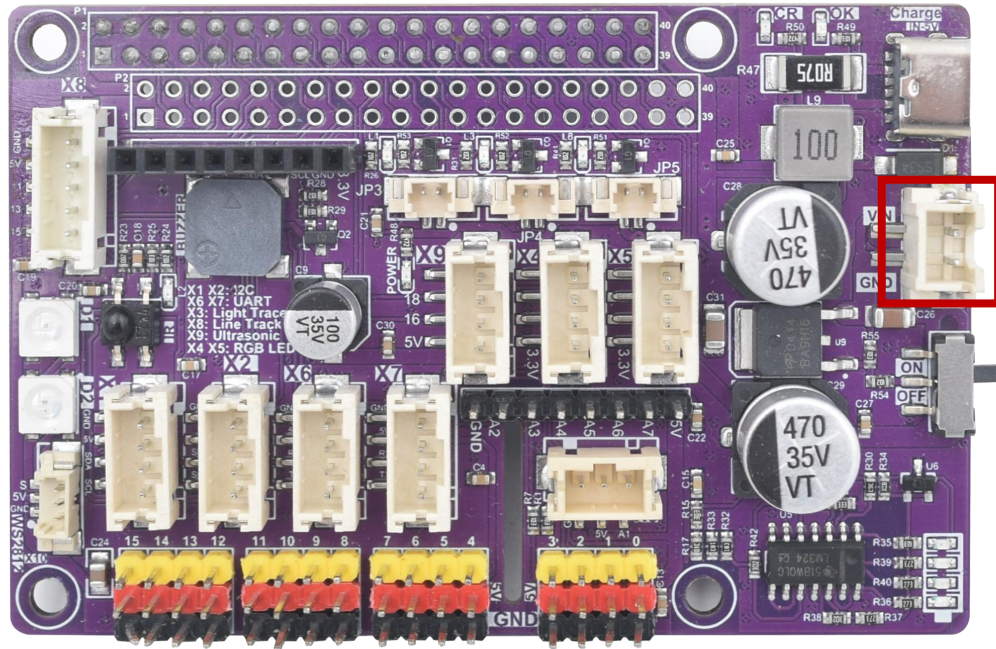


The address occupied by the ADS7830 chip on Adeept Robot HAT V3.2 is **0x48**

ADC is an electronic integrated circuit used to convert analog signals such as voltages to digital or binary form consisting of 1s and 0s. The range of our ADC on Raspberry Pi is 8 bits, that means the resolution is $2^8=256$, and it represents a range will be divided equally to 256 parts. The range of analog values corresponds to ADC values. So the more bits the ADC has, the denser the partition of analog will be and the greater the precision of the resulting conversion.

13.4 Wiring Diagram





13.5 Demonstration

1. **Remotely log:** Remotely log in to the Raspberry Pi terminal.
2. **Navigate to the Program Folder:** Enter the following command in the terminal and press **Enter** to access the folder where the program is located:

```
cd Adeept_RaspClaws-V3/Examples/07_Voltage/
```

```
pi@raspberrypi:~ $ cd Adeept_RaspClaws-V3/Examples/07_Voltage/
pi@raspberrypi:~/Adeept_RaspClaws-V3/Examples/07_Voltage $
```

3. **View Directory Contents:** Type "**ls**" in the terminal and press Enter. This will display all the files in the current directory, ensuring that the "**BatteryLevelMonitoring.py**" file is present:

```
ls
```

```
pi@raspberrypi:~/Adeept_RaspClaws-V3/Examples/07_Voltage $ ls
BatteryLevelMonitoring.py
```

4. **Run the Program:** Enter the command below and press **Enter** to start the **BatteryLevelMonitoring.py** program:

```
sudo python3 BatteryLevelMonitoring.py
```

```
pi@raspberrypi:~/Adeept_RaspClaws-V3/Examples/07_Voltage $ sudo python3 BatteryL
evelMonitoring.py
Current battery level: 63.73 %
Current battery level: 66.99 %
Current battery level: 66.99 %
Current battery level: 66.99 %
Current battery level: 66.99 %
```

5. **Observation and Termination:** After successfully running the program, you will see the console continuously output the percentage of the current battery level. When the battery level drops below 20%, a warning message will also be displayed. When you want to terminate a running program, you can press the **Ctrl+C** shortcut key on the keyboard.

13.6 Code

Complete code refer to [BatteryLevelMonitoring.py](#)

```
01  #!/usr/bin/env/python3
02  # File name   : BatteryLevelMonitoring.py
03  # Website    : www.Adeept.com
04  # Author     : Adeept
05  # Date      : 2025/04/9
06  import time
07  import board
08  import busio
09  from adafruit_bus_device.i2c_device import I2CDevice
10
11  i2c = busio.I2C(board.SCL, board.SDA)
12  # ADS7830 adress 0x48
13  device = I2CDevice(i2c, 0x48)
14
15  # Define constants
16  Vref = 8.4
17  WarningThreshold = 6.0
18  R15 = 3000
19  R17 = 1000
20  DivisionRatio = R17 / (R15 + R17)
21
22  #Define the ADC channel and command.
23  cmd = 0x84
```



```
24 channel = 0
25 control_byte = cmd | (((channel << 2 | channel >> 1) & 0x07) << 4)
26
27 if __name__ == "__main__":
28     buffer = [1]
29     while True:
30         device.write_then_readinto(bytes([control_byte]), buffer)
31         adcValue = buffer[0]
32         A0Voltage = (adcValue / 255) * 5
33         ActualBatteryVoltage = A0Voltage / DivisionRatio
34
35         BatteryPercentage = (ActualBatteryVoltage - WarningThreshold) / (Vref - WarningThreshold) * 100
36
37         print(f"Current battery level: {BatteryPercentage:.2f} %")
38
39         # Battery level warning judgment
40         if BatteryPercentage < 20:
41             print("Warning! The battery level is too low. Please charge in time!")
42         time.sleep(0.5)
```

Code explanation

First, initialize the I2C bus and the device, and define constants and communication control bytes.

In the main loop, send commands to the chip to read the ADC value. After conversion, calculate the actual battery voltage and the battery percentage, and then output the results. If the battery percentage is below 20%, a warning will be issued. The monitoring is performed every 0.5 seconds.