General Info

The weight scale subsystem consists of four load cells, load cell cases, and an HX711 analog-to-digital converter which interfaces with the Raspberry Pi Pico. The load cells are placed at the corners of a rectangular board and connect to the HX711 through four wires. The HX711 takes the input voltages and converts them into digital weight data, which is sent directly to the Pico. It also relies on the Pico for power, ground, and a clock signal. Through our code file, an accurate weight can be sent to the data collection subsystem when requested.

Parts

HX711 Chip

We are using an HX711 chip as an ADC to convert pressure on load cells to digital signals so that the Raspberry Pi can read weight through GPIOs.

The HX711 can operate between 2.5 and 5 V with less than 1.5 mA of current. More info can be found via this link: https://www.sparkfun.com/products/13879

In terms of implementation, solder the load cells as in Figure 1 and connect the red wires from the load cell to the corresponding tag (E+, E-, A-, A+). The order is arbitrary as long as the relative position is right.

There are five pins that are designed to connect with a Raspberry Pi: VCC, VDD, GND, DAT (data), and SCK (clock). VCC is connected to the 3V3 output voltage pin from the Pico, and GND is connected to the ground. DAT and SCK are connected to GPIOs on the Pico, and the specifics are described in the following sections.

Load Cells (Weight Sensors)

We implemented four weight sensors (load cells) at the four corners of the weight scale. The connection with the HX711 chip is depicted in the following figure:

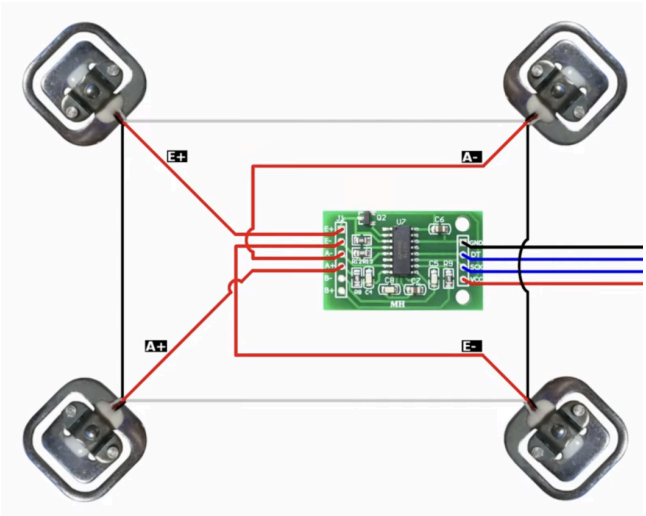

Fig. 1 Connections of Load Cells HX711 Chip

3D-Printed Case for Load Cells:

We also designed our 3D-printed case to support our load cells and to fix it on a position on the board. We used a PLA filament to print our case, which is a solid, light, and waterproof material that protects the wires from water and natural tearing. The STL files are uploaded in Box.

Raspberry Pi Pico:

Finally, we calculate and receive every data gathered in a Raspberry Pi Pico. Our device doesn't require an I2C protocol on the Pico, so it is fine to connect the Data and Clock signal to any GPIO pins. Here is a piece of example code to retrieve weight

```
def GetWeight(a = -4.482101*10**(-5), b = 381.3089):
    weight = readCount()
    if weight == 8388608:
        print("Error reading HX711")
        return -1
    weight = a * weight + b
    result = round(weight, 3)
    return result


def readCount():

    i = 0
    Count = 0
    SCK = Pin(15, Pin.OUT)
```

```
    DT = Pin(14, Pin.OUT)
    DT.high()
    DT.low()
    DT = Pin(14, Pin.IN)

    while DT.value() == 1:
        i = 0

    for i in range(24):
        SCK.high()
        Count = Count << 1
        SCK.low()

        if DT.value() == 1:
            Count += 1

    SCK.high()

    Count = Count ^ 0x800000
    SCK.low()
    return Count
```

There are two pieces of code, where GetWeight() returns the weight in kilograms of the current measurement, and readCount() returns the digital signal from the chip.

GetWeight(a, b):

GetWeight() first retrieves the raw digital signal from HX711 and then converts it into kilograms or checks if the HX711 is working properly. The inputs a and b are used for linear scaling and offsetting from raw data to kilograms, and they might be subject to change due to the different sensitivity of load cells.

readCount():

readCount reads a 24-bit long signal from the DT and SCK pin. Make sure to change the pin number to the correct pins for your own implementation.

Assembling Instructions

Load Cell Cases

Fit the load cell into the bottom piece of the load cell case such that it lays flat, with the wires fitted into the center slot. Then line up the top piece with the bottom piece and place it on top. Repeat for the other three load cells.

If they are not already attached to the board, arrange the load cell cases at the corners of the board such that they form a rectangle.

Connections

Connect the four wires from the load cells (E+, E-, A+, A-) to the corresponding pins on the HX711. From the HX711, connect VCC, ground, data, and clock to the appropriate pins on the Pico.

Notes and Links

References

- MicroPython:
  - https://docs.micropython.org/en/latest/
    - https://docs.micropython.org/en/latest/library/machine.SPI.html
    - https://docs.micropython.org/en/latest/library/machine.Pin.html
  - https://github.com/micropython/micropython/wiki
  - https://forum.micropython.org/
- GitHub link for weigh scale code, tutorials, and settings for Thonny:
  - https://github.com/arthinox/WaggleNet-stuff

Troubleshooting

Drifted data:

The weight is measured by change in resistance of the load cells, and the deformation takes time. As a result, the measured weight will drift up or down for a few hours and stabilize to a certain value. When an object with the weight similar to a bee hive, the drift is usually within 1%.

Outlier Samples:

If we list all samples, there might be some outliers, abnormally large or small numbers, in the final list. We think this might be caused by the instability of load cells and weather conditions. A way to solve this is to apply a median filter to get rid of the outliers.

Incorrect Signals:

During our processing of making this device, we have encountered several times the weight scale reading constant values no matter how much weight is on it. This problem is likely because of accidental damaging the HX711 chip while soldering. In most cases, using a new chip can solve this problem.