Digital Weighing Machine

26/04/2024

1 Team Members:

1. Kunal Thorawade - EE23BTECH11035
2. Sasa Mardi - EE23BTECH11222
3. Ganne Gopi Chandu - EE23BTECH11021
4. Manoj Kumar - EE23BTECH11211

2 Aim:

To build digital weighing machine that can measure weight upto 1kg precisely.

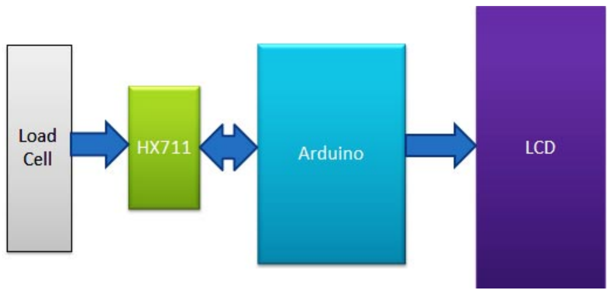
3 Introduction:

In this project we made a digital weighing scale with the help of arduino nano board and other modules that can measure weight upto 1kg. We also 3D printed components in this project.

4 Components Required:

1. 1X Arduino Nano Board
2. 1X HX711 Module
3. 1X 16X2 I2C LCD DIsplay
4. 1X 5kg Load Cell
5. 1X Bread Board
6. Jumper Wires

5 Circuit Block Diagram:



6 Components of Circuit:

6.1 5kg Load Cell:

The Load cell used in this project is bending beam load cell. It is a strain gauge load cell that converts the load acting on it into electrical signals. It cam can translate up to 5 kg of pressure (force) into an electrical signal. This straight bar load cell is made from an aluminum alloy and is capable of reading a capacity of 1KG of weight.It has Four lead wires which can be connected to HX711 A/D Pressure Sensor. It is easy to use with driving voltage 5-10V and produce the output voltage as per the force changes over it.

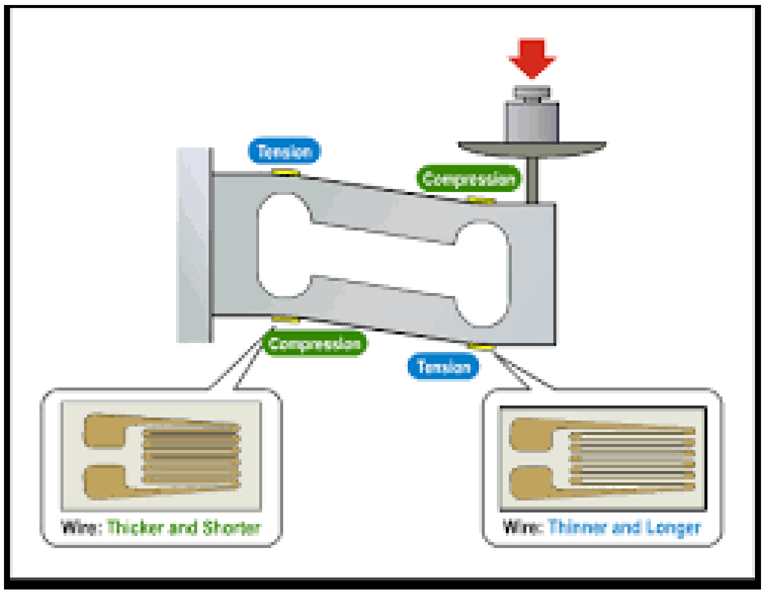


Figure 1: Load cell

6.1.1 Strain Gauge:

It is a transducer that converts mechanincal displacement into a change of resistance. It is use to measure strain of object. The resistance changes as per the formula $R = \frac{\rho \times L}{A}$.

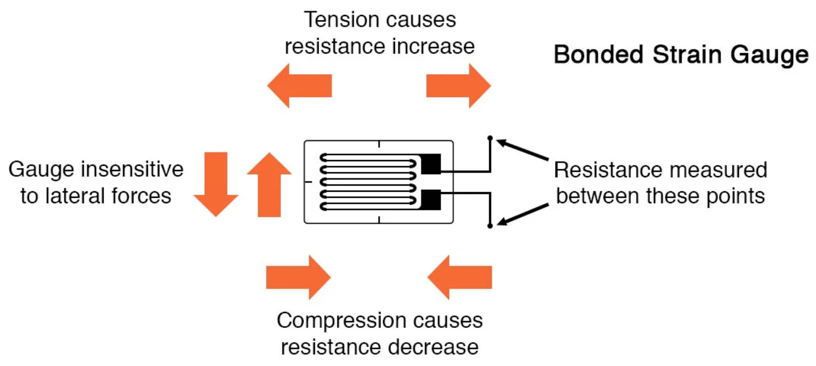

Tension causes resistance increase
Bonded Strain Gauge
Gauge insensitive to lateral forces
Resistance measured between these points
Compression causes resistance decrease

Figure 2: Strain Gauge

6.1.2 Circuit Diagram:

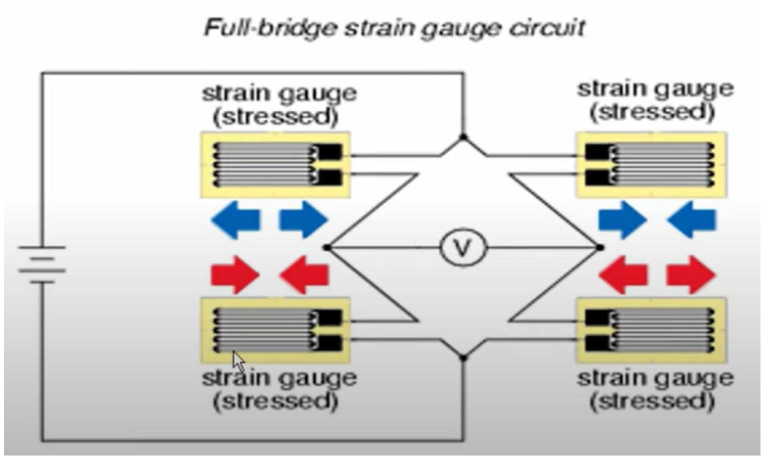


Figure 3: Circuit Diagram of load cell

The force is calculated by measuring V, when there is no load on the cell then resistance of all four gauges is same and therefore $V = 0$.

6.1.3 Structure:

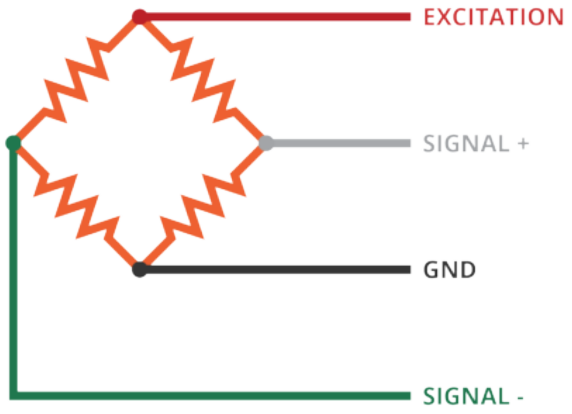


Figure 4: Structure of load cell

Red and Black wires is for to give input voltage across and white and green wires are used to measure output voltage across load cell.

6.2 HX711 Module:

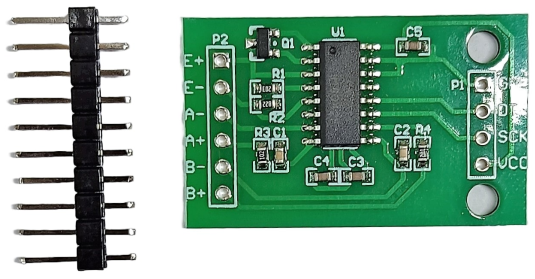


Figure 5: HX711 Module

6.2.1 Load Cell to HX711 Module Connections

1. **Red Wire (Excitation+)**: Carries excitation voltage to power strain gauges.
2. **Black Wire (Excitation-)**: Completes excitation voltage circuit and ensures proper grounding.

3. **Green Wire (Signal+)**: Transmits positive differential signal from strain gauges.
4. **White Wire (Signal-)**: Carries negative differential signal for weight measurement.

- The HX711 module is a precision 24-bit analog-to-digital converter (ADC) specifically designed for weighing scales and industrial control applications.
- In an Arduino Nano-based weighing machine, the HX711 module plays a crucial role in converting the analog weight measurement from a load cell into a digital value that the Arduino Nano can process and display.

Here's how it works:

- The load cell generates an output signal that typically falls within the range of millivolts, which is too low for direct processing.
- To effectively utilize this signal, it requires amplification to elevate it to a level suitable for further processing.
- This is where the HX711 amplifier sensor comes into play.
- The HX711 amplifier sensor is specifically designed to address this requirement, featuring an integrated HX711 chip renowned for its precision analog-to-digital conversion capabilities, offering a high level of accuracy with 24-bit resolution.

In summary, the HX711 module in an Arduino Nano-based weighing machine serves as a vital interface between the load cell and the Arduino Nano, converting analog weight measurements into digital values for accurate and precise weight readings. It provides features like gain and offset adjustment for calibration, ensuring reliable performance and accuracy in weight measurement applications.

6.2.2 HX711 Module to Arduino Nano Connections

1. **VCC (5V)**: provide power to the HX711 module.
2. **GND (GND)**:to establish a common ground reference.
3. **DT (D4)**: to transfer data from the HX711 module to the Arduino Nano.
4. **SCK (D5)**:clock to synchronize data transmission between the HX711 module and the Arduino Nano.

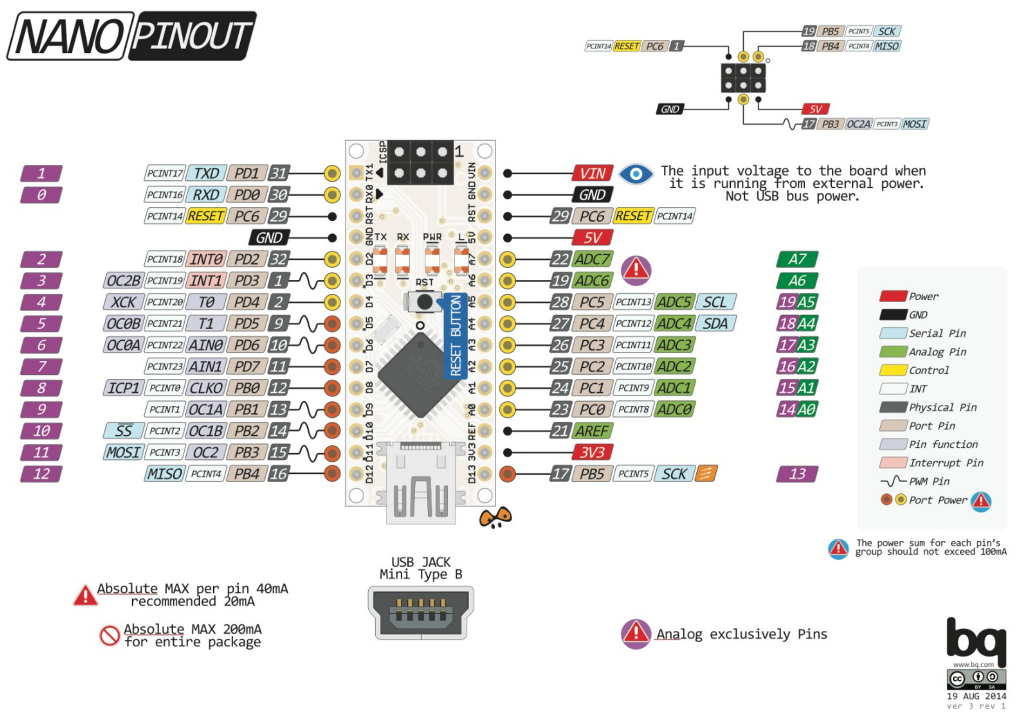
6.3 Arduino Nano:

The Arduino Nano is an open-source breadboard-friendly microcontroller board based on the Microchip ATmega328P microcontroller (MCU)
It offers the same connectivity and specs of the Arduino Uno board in a smaller form factor.

6.3.1 Technical Specifications:

Microcontroller: Microchip ATmega328P
Operating voltage: 5 volts
Input voltage: 5 to 20 volts
Digital I/O pins: 14 (6 optional PWM outputs)
Analog input pins: 8
DC per I/O pin: 40 mA
DC for 3.3 V pin: 50 mA
Flash memory: 32 KB, of which 2 KB is used by bootloader
SRAM: 2 KB
EEPROM: 1 KB
Clock speed: 16 MHz
Length: 45 mm
Width: 18 mm
Mass: 7 g
USB: Mini-USB Type-B
ICSP Header: Yes
DC Power Jack: No

6.3.2 BLOCK FIGURE



6.3.3 Automatic (software) reset:

Rather than requiring a physical press of the reset button before an upload, the Arduino Nano is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the FT232RL is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip.

6.3.4 Code:

```
#include <LiquidCrystal_I2C.h>
#include <HX711_ADC.h>
#if defined(ESP8266) || defined(ESP32) || defined(AVR)
#include <EEPROM.h>
#endif

// HX711 pin connections
const int HX711_dout = 4; // HX711 DOUT pin > Arduino pin
const int HX711_sck = 5;  // HX711 SCK pin > Arduino pin

// Create an instance of the HX711_ADC class
HX711_ADC LoadCell(HX711_dout, HX711_sck);

// Create an instance of the LiquidCrystal_I2C class
LiquidCrystal_I2C lcd(0x27, 16, 2); // Set the LCD I2C address

// EEPROM address for storing calibration value
const int calVal_eepromAddress = 0;

// Variable for storing the time to assist with timing print outputs
unsigned long t = 0;

void setup() {
  Serial.begin(57600);
  Serial.println(”\nStarting...”);

  lcd.init();                      // Initialize the LCD
  lcd.backlight();                 // Turn on the backlight

  float calibrationValue = 500.0;  // Calibration value for 0-1000 grams

#if defined(ESP8266) || defined(ESP32) || defined(AVR)
  if (EEPROM.read(calVal_eepromAddress) != 255) {
    EEPROM.get(calVal_eepromAddress, calibrationValue);
  }
#endif

  LoadCell.begin();
  LoadCell.start(1000, true);

  if (LoadCell.getTareTimeoutFlag()) {
    Serial.println(”Timeout, check MCU>HX711 wiring and pin designations”);
    lcd.print(”Check wiring!”);
  } else {
    LoadCell.setCalFactor(calibrationValue); // Apply the retrieved or default calibration value
    Serial.println("Startup is complete");
    lcd.print("Ready");
  }

  lcd.setCursor(0, 1);
  lcd.print("Cal: ");
  lcd.print(LoadCell.getCalFactor());
}

void loop() {
  static boolean newDataReady = false;

  if (LoadCell.update()) {
    newDataReady = true;
  }

  if (newDataReady && millis() > t + 500) { // 500ms update rate
    float weight = LoadCell.getData();
    Serial.print("Load_cell output val: ");
    Serial.println(weight);

    // Adjust weight to ensure it's within 0 to 1000 grams
    if (weight < 0) {
      weight = 0;
    } else if (weight > 1000) {
      weight = 1000;
    }

    lcd.setCursor(0, 0);
    lcd.print("Weight: ");
    lcd.print(weight, 0); // Display weight without decimal places
    lcd.print(" g  ");

    newDataReady = false;
    t = millis();
  }

  if (Serial.available() > 0) {
    char inByte = Serial.read();
    if (inByte == 't') {
      LoadCell.tareNoDelay();
      lcd.setCursor(0, 1);
      lcd.print("Taring...      ");
    }
  }

  if (LoadCell.getTareStatus()) {
    lcd.setCursor(0, 1);
    lcd.print("Tare complete ");
  }
}
```

6.3.5 Process for uploading code into Arduino:

The code compile it in the Arduino IDE to check for any errors. Then, connect your Arduino Nano to your computer via USB and upload the code to the board. The Arduino IDE handles the compilation and uploading process automatically.

6.3.6 Testing and Debugging:

After uploading the code, test your project to ensure that it functions as expected. Use serial output, LEDs, or other debugging techniques to verify that your code is running correctly and responding to inputs as intended.

6.3.7 Explanation of pins

1. D6 pin:
   The D6 pin on an Arduino Nano is the seventh pin down on the right side of the board, when viewed from the component side, with the USB port at the top. It is mapped to Port F pin 4 on the ATmega4809. The D6 pin is a digital IO pin and is used to access the AIN0 and AIN1

2. D5:
   Digital pin D5 refers to one of the GPIO (General Purpose Input/Output) pins on the microcontroller. You can use it for both digital input and output operations.

3. D4:
   D4 is pin 7 on the Arduino Nano V2.3. It is a digital IO pin and is reprogrammed to accept hardware interrupts.

4. A4 and A5:
   Pins A4 and A5 are used by the microcontroller to communicate with some onboard peripherals (I2C), such as the crypto chip (required for Cloud connectivity). Avoid reading from these pins if your project relies on I2C communication. The input voltage range is lower than the board's 3.3 V operating voltage.

5. 5v pin:
   The 5V pin on the Arduino Nano is a regulated 5V output that can power sensors, displays, Bluetooth modules, and other components that require 5V and a total load of less than 500mA.

6. GND: GND is ground

6.3.8 Connnections:

The 5v,GND,A4 and A5 pins are connected to Lcd display, 5v and D6 are connected to resistor

6.4 Resistor:

The resistor helps limit the current flowing into the input pin, ensuring that it stays within safe operating limits, even if there are fluctuations or spikes in the voltage. This helps protect the Arduino from potential damage and ensures reliable operation of the weighing machine project.

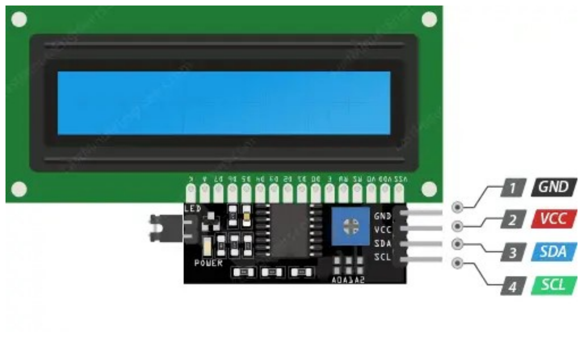


Figure 6: LCD

6.5 LCD (Liquid Crystal Display) with I2C module

LCD (Liquid Crystal Display) screens are widely used in various electronic devices for displaying information. A 16x2 LCD display refers to a display with 16 columns and 2 rows of characters. Integrating an I2C (Inter-Integrated Circuit) module with the LCD display enhances its usability and simplifies the interface with microcontrollers.

6.5.1 Features of 16x2 LCD Display:

1. **Dimensions:** 16 characters wide and 2 rows
2. **Character Set:** Typically supports ASCII characters
3. **Backlight:** May include an LED backlight for visibility in low light conditions
4. **Contrast Adjustment:** Allows adjusting the contrast of displayed characters
5. **Interface:** Traditional displays require multiple GPIO (General-purpose input/output) pins for interfacing with microcontrollers

6.5.2 Working Principle:

The 16x2 LCD display operates based on the principle of liquid crystal modulation. Each character position consists of a matrix of pixels, and by controlling the voltage across each pixel, characters can be displayed. The I2C module simplifies the interface by converting the parallel data signals into serial data, reducing the number of required GPIO pins on the microcontroller.

6.5.3 Integration with I2C Module:

The I2C moduletypically includes a small circuit board with an I2C serial interface and a PCF8574 I/O expander chip. This module connects to the LCD display and allows communication with the microcontroller through the I2C protocol. By using the I2C protocol, only two wires (SDA and SCL) are required to interface with the microcontroller, simplifying the wiring and conserving GPIO pins.

6.5.4 Connections: with Arduino

1. GND (Ground):

- This connection is used to provide a common reference voltage for all the components in the circuit.
- connect the GND pin of the LCD module to any GND pin on the Arduino board. This ensures that the ground potentials of both devices are equalized.

2 VCC (Power):

- VCC is the power supply voltage for the LCD module.
- Connect the VCC pin of the LCD module to the 5V pin on the Arduino board. This provides the necessary power for the LCD module to operate.

3. SDA (Serial Data):

- SDA is the serial data line used for bidirectional communication between the Arduino and the LCD module.
- Connect the SDA pin of the LCD module to the SDA pin on the Arduino.

4. SCL (Serial Clock):

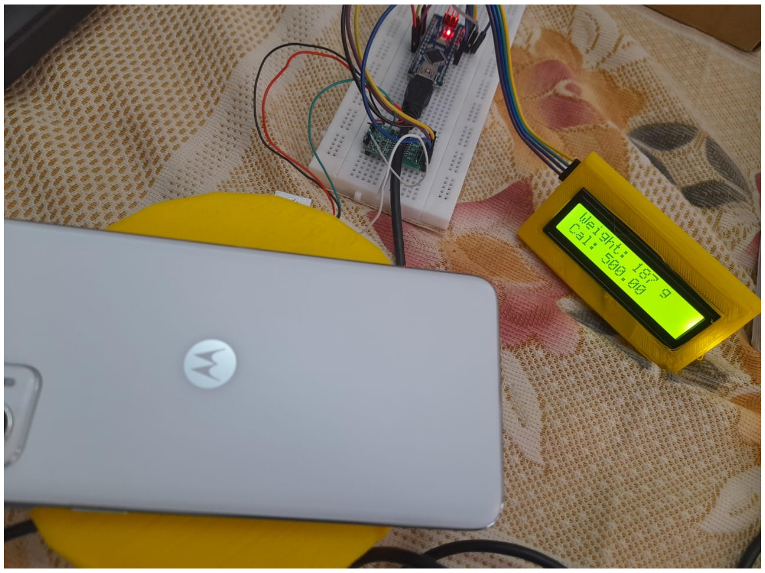
- SCL is the serial clock line used to synchronize data transmission between the Arduino and the LCD module.
- Connect the SCL pin of the LCD module to the SCL pin on the Arduino. On most Arduino board

6.5.5 Advantages:

1. Fewer Wires: With the I2C module, we only need two wires instead of a bunch, making it much simpler to connect.
2. Saves Pins: Devices, like microcontrollers, have limited pins. The I2C module helps save these pins for other uses.
3. Ease of Use: Simplified interface makes it easier to integrate the LCD display into projects without extensive wiring.

7 Observation Table:

S No. Error (g)	Components	Actual Weight (g)	Measured Weight (g)
1 3	Kangaroo stapler 10	100	97
2 4	Samsung S23	168	164
3 0	Moto G74	187	187
4 5	Samsung A34	200	195



8 Conclusion:

The project was successful and various weights were measured using weighing machine. There were some errors in measurments due to temperature dependent properties of load cell and orientation of object whose weight is measured.