# Digital Weighing Device Development



## Watch our video by clicking the link below

https://drive.google.com/drive/u/0/folders/1udP51ES5tZ4oDZnTauOK7WFT08RLYecf

## Arduino Code

```
#include <Keypad.h>
#include <LiquidCrystal.h>
#include <HX711.h>
// #include <HX711_ADC.h>
#include<EEPROM.h>

const int rs = 7, en = 6, d4 = 5, d5 = 4, d6 = 3 , d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

char keys[4][4] = {
  {'1', '2', '3', '+'},
  {'4', '5', '6', '-'},
  {'7', '8', '9', '/'},
  {'C', '0', '#', '*'}
};
```

```
byte rowPins[4] = {8, 9, 10, 11};  // Connect to the row pinouts
byte colPins[4] = {12, 13, A0, A1}; // Use analog pins for flexi
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, 4, 4)


// Pins for HX711
// const int HX711_dout = 1; // mcu > HX711 dout pin
// const int HX711_sck = 0; // mcu > HX711 sck pin
uint8_t dataPin = 1;
uint8_t clockPin = 0;

// HX711 constructor
// HX711_ADC LoadCell(HX711_dout, HX711_sck);
HX711 scale;

const int calVal_eepromAdress = 0;

unsigned char  Veg_Price[8] = {15, 5, 25, 23, 20, 10, 7, 9};
unsigned char Fruit_Price[8] = {30, 35, 45, 60, 50, 30, 45, 40};

float Weight=0;
float tPrice = 0;
float Price=0;
unsigned char operation=0;
unsigned char PRICES[10] = {0};
float mass[10]={0};
String pro_Name[10];
unsigned char arr_op[10]={0};
unsigned char INDEX=0;
float READING=0;
char key =NO_KEY; // Read the key
unsigned long long int Delay=0xffffffffffffffU;

float Calc_weight()
{
```

```
  float weight;
   if (scale.is_ready()) {
  weight = scale.get_units();
  weight*=2.381;
   }
  return weight;
}


char pressedKey() {
  return keypad.getKey();
}

unsigned char remove_op(unsigned char OP_NUM)
{
  if(arr_op[OP_NUM]==0)
  {
    return 0;
  }
  lcd.clear();
  lcd.print(pro_Name[OP_NUM]);
  lcd.setCursor(0,1);
  lcd.print(mass[OP_NUM]);
  lcd.setCursor(10,1);
  lcd.print("grams");
  delay(1000);
  tPrice-=PRICES[OP_NUM];
  lcd.clear();
  lcd.print("Tprice after remove");
  lcd.setCursor(0,1);
  lcd.print(tPrice);
  delay(2000);
  arr_op[OP_NUM]=0;
  INDEX--;

  return 1;
```

```
}
void ASK_NEXT()
{
   while(1)
  {
    lcd.clear();
    lcd.print("1-add products");
    lcd.setCursor(0,1);
    lcd.print("2-remove product");
    delay(2000);
    lcd.clear();
    lcd.print("3-To pay");
    delay(2000);
    lcd.clear();
    lcd.print("enter a choice");
    Delay=0xffffffffffffffU;
    do
    {
      operation =  pressedKey();
    }while(  (operation ==NO_KEY)&&(Delay--) );
    if( operation=='1')
    {
      operation = '+';
      break;
    }
    else if(operation=='2' )
    {
      operation='-';
      break;
    }
    else if(operation=='3' )
    {
      operation='e';
      break;
    }
    else if((operation>'3')&&(operation<'1'))
```

```
    {
      lcd.clear();
      lcd.print("un valid option");
      delay(2000);
    }
    else
    {
      lcd.clear();
      lcd.print("select an option");
      delay(2000);
    }
  }
  delay(2000);
  if(operation=='+')
  {
    while(1)
    {
      Delay=65535U;
      lcd.clear();
      lcd.print("1-Vegatables.");
      lcd.setCursor(0,1);
      lcd.print("2-Fruits.");
      delay(2000);
      lcd.clear();
      lcd.print("enter a choice :");
      do
      {
        key =  pressedKey();
      }while( (key ==NO_KEY) );
      if(key != NO_KEY )
      {
        break;
      }
      lcd.clear();
      lcd.print("un valid option");
      delay(2000);
```

```
    }
    logic_market(key);
  }
  else if(operation=='-')
  {
     while(1)
    {
      Delay=65535U;
      lcd.clear();
      lcd.print("enter operation");
      lcd.setCursor(0, 1);
      lcd.print("num");
      do
      {
        key =  pressedKey();
        if(key!=NO_KEY&&arr_op[key-'0'-1]==0)
        {
          lcd.clear();
          lcd.print("wrong op num");
          delay(2000);
          key=NO_KEY;
        }
      }while( (key ==NO_KEY));
      if(key != NO_KEY )
      {
        remove_op((key-'0'-1));
        ASK_NEXT();
      }
      lcd.clear();
      lcd.print("un valid option");
      delay(2000);
    }
  }
  else if(operation=='e')
  {
    lcd.clear();
```

```
    lcd.print(tPrice);
    lcd.setCursor(6,0);
    lcd.print("EGP");
    delay(2000);
    lcd.clear();
    lcd.print("have a blessed day");
    delay(2000);
    key='e';
  }
  else
  {
    /nothing/
    ASK_NEXT();
  }
}
void logic_market(char TYPE)
{
  if(TYPE == '1')
  {
    key=NO_KEY;
      while(key ==NO_KEY)
      {
        lcd.clear();
        lcd.print("1-Cucumber");
        lcd.setCursor(10, 0); // Column 10, Row 0
        lcd.print("2-Corn");
        lcd.setCursor(0, 1); // Column 0, Row 1
        lcd.print("3-Peas");
        lcd.setCursor(7, 1);
        lcd.print("4-Tomato");

        delay(2000);

        lcd.clear();

        lcd.print("5-Potato");
```

```
        lcd.setCursor(7,0);
        lcd.print("6-Pumpkin");
        lcd.setCursor(0,1);
        lcd.print("7-Onions");
        lcd.setCursor(8,1);
        lcd.print("8-Carrot");

        delay(2000);
        lcd.clear();
        lcd.print("Enter a choice");
        Delay=0xffffffffffffffU;
         while((Delay--)&& (key ==NO_KEY) )
        {
          key =  pressedKey();
        }
        if((key>'8'||key<'1')&&key!=NO_KEY)
        {
          lcd.clear();
          lcd.print("un valid option");
          key=NO_KEY;
          delay(2000);
        }
     }
     Price = Veg_Price[key-'0'-1];

    switch (key)
    {
      case '1':
      pro_Name[INDEX]= "Cucumber";
      break;
      case '2':
      pro_Name[INDEX]= "Corn";
      break;
      case '3':
      pro_Name[INDEX]= "Peas";
      break;
```

```
      case '4':
      pro_Name[INDEX]= "Tomato";
      break;
      case '5':
      pro_Name[INDEX]= "Potato";
      break;
      case '6':
      pro_Name[INDEX]= "Pumpkin";
      break;
      case '7':
      pro_Name[INDEX]= "Onions";
      break;
      case '8':
      pro_Name[INDEX]= "Carrot";
      break;
    }
  }
  else if (TYPE == '2')
  {
    key=NO_KEY;
    while(key ==NO_KEY)
    {
        lcd.clear();
        lcd.print("1-Apple");
        lcd.setCursor(9, 0); // Column 10, Row 0
        lcd.print("2-Peach");
        lcd.setCursor(0, 1); // Column 0, Row 1
        lcd.print("3-Banana");
        lcd.setCursor(9, 1);
        lcd.print("4-Mango");

        delay(2000);

        lcd.clear();

        lcd.print("5-Kewi");
```

```
        lcd.setCursor(9,0);
        lcd.print("6-Guava");
        lcd.setCursor(0,1);
        lcd.print("7-Grape");
        lcd.setCursor(8,1);
        lcd.print("8-Orange");

        delay(2000);
        lcd.clear();
        lcd.print("Enter a choice");
        Delay=0xffffffffffffffU;
        while((Delay--)&& (key ==NO_KEY) )
        {
          key =  pressedKey();
        }
        if((key>'8'||key<'1')&&key!=NO_KEY)
        {
          lcd.clear();
          lcd.print("un valid option");
          key=NO_KEY;
          delay(2000);
        }
    }
    Price = Fruit_Price[key-'0'-1];
    switch (key)
    {
      case '1':
      pro_Name[INDEX]= "Apple";
      break;
      case '2':
      pro_Name[INDEX]= "Peach";
      break;
      case '3':
      pro_Name[INDEX]= "Banana";
      break;
      case '4':
```

```
      pro_Name[INDEX]= "Mango";
      break;
      case '5':
      pro_Name[INDEX]= "Kewi";
      break;
      case '6':
      pro_Name[INDEX]= "Guava";
      break;
      case '7':
      pro_Name[INDEX]= "Grape";
      break;
      case '8':
      pro_Name[INDEX]= "Orange";
      break;
    }

  }
  else
  {
    /nothing/
  }
  /*
      sensor reading

  */
  Weight= Calc_weight();

  delay(2000);
  lcd.clear();
  lcd.print("Price of ");
  lcd.print(pro_Name[INDEX]);
  lcd.setCursor(0,1);
  lcd.print(Price);
  delay(2000);

  Price = (Price*Weight)/1000U;
```

```
  tPrice += Price;
  PRICES[INDEX] = Price;
  mass[INDEX]=Weight;
  arr_op[INDEX]=1;
  INDEX++;

  lcd.clear();
  lcd.print("Weight in grams: ");
  lcd.setCursor(0,1);
  lcd.print(Weight);
  delay(2000);
  lcd.clear();
  lcd.print("Price : ");
  lcd.setCursor(0,1);
  lcd.print(Price);
  lcd.print(" EGP");
  delay(2000);
  lcd.clear();
  lcd.print("Total Price : ");
  lcd.setCursor(0,1);
  lcd.print(tPrice);
  lcd.print(" EGP");
  delay(2000);

  ASK_NEXT();
}

void setup() {
  lcd.begin(16, 2);

  scale.begin(dataPin, clockPin);

  // float calibrationFactor =  2280.0; // Replace with your cal
  // scale.set_scale(calibrationFactor);
  // scale.tare(); // Tare the scale to zero
```

```
}

void loop()
{

  // Delay=65535U;
  while(1)
  {
    Delay=65535U;
    lcd.clear();
    lcd.print("1-Vegatables.");
    lcd.setCursor(0,1);
    lcd.print("2-fruits.");
    delay(2000);
    lcd.clear();
    lcd.print("enter a choice :");
    do
    {
      key =  pressedKey();
    }while( (key == NO_KEY) );
    if(key == '1'||key=='2' )
    {
      break;
    }
    lcd.clear();
    lcd.print("un valid option");
    delay(2000);
  }


  logic_market(key);

  if(key=='e')
  {
    return;
  }
```

```
}
```

## Code Explanation:

**1. Code Overview:**

- This Arduino code implements a basic electronic weighing scale system for a market.
- It uses a keypad for user input, an LCD display for user interaction and displaying information, and a load cell sensor (HX711) to measure the weight of produce.
- The system allows users to select between vegetables and fruits, choose specific items, and calculate the total price based on weight and predefined prices.

**2. Key Components:**

- **Keypad:**
  - Used for user input (selecting product categories, individual items, and performing actions).
- **LCD Display:**
  - Provides user interface for displaying messages, instructions, product selections, weight, and price information.
- **Load Cell (HX711):**
  - Measures the weight of the produce placed on the scale.
- **EEPROM:**
  - Used to store calibration values for the load cell.

**3. Core Functionality:**

- **Initialization:**
  - In `setup()`, the LCD, keypad, and load cell are initialized.
  - The load cell is calibrated using a stored calibration value from EEPROM.

- **User Interaction:**
  - The code guides the user through a series of steps:
    - Select between "Vegetables" or "Fruits".
    - Choose a specific item from the available options.
    - The system then reads the weight from the load cell.
    - Calculates the price based on weight and the predefined price of the selected item.
    - Displays the weight, price per unit, and total price on the LCD.
- **Price Calculation:**
  - The code maintains a running total of the overall price.
  - It allows users to add or remove items from the "cart".
- **Final Output:**
  - At the end, the system displays the final total price on the LCD.

**4. Important Code Sections:**

- `Calc_weight()`:
  - Reads multiple weight readings from the load cell to improve accuracy and filter out noise.
- `pressedKey()`:
  - Reads and returns the key pressed on the keypad.
- `logic_market()`:
  - Handles the core logic for selecting products, reading weight, and calculating prices.
- `ASK_NEXT()`:
  - Provides options to the user: add products, remove products, or proceed to payment.
- `remove_op(unsigned char OP_NUM)`

**Purpose:**

- This function is designed to remove a previously added product from the current order.

**Functionality:**

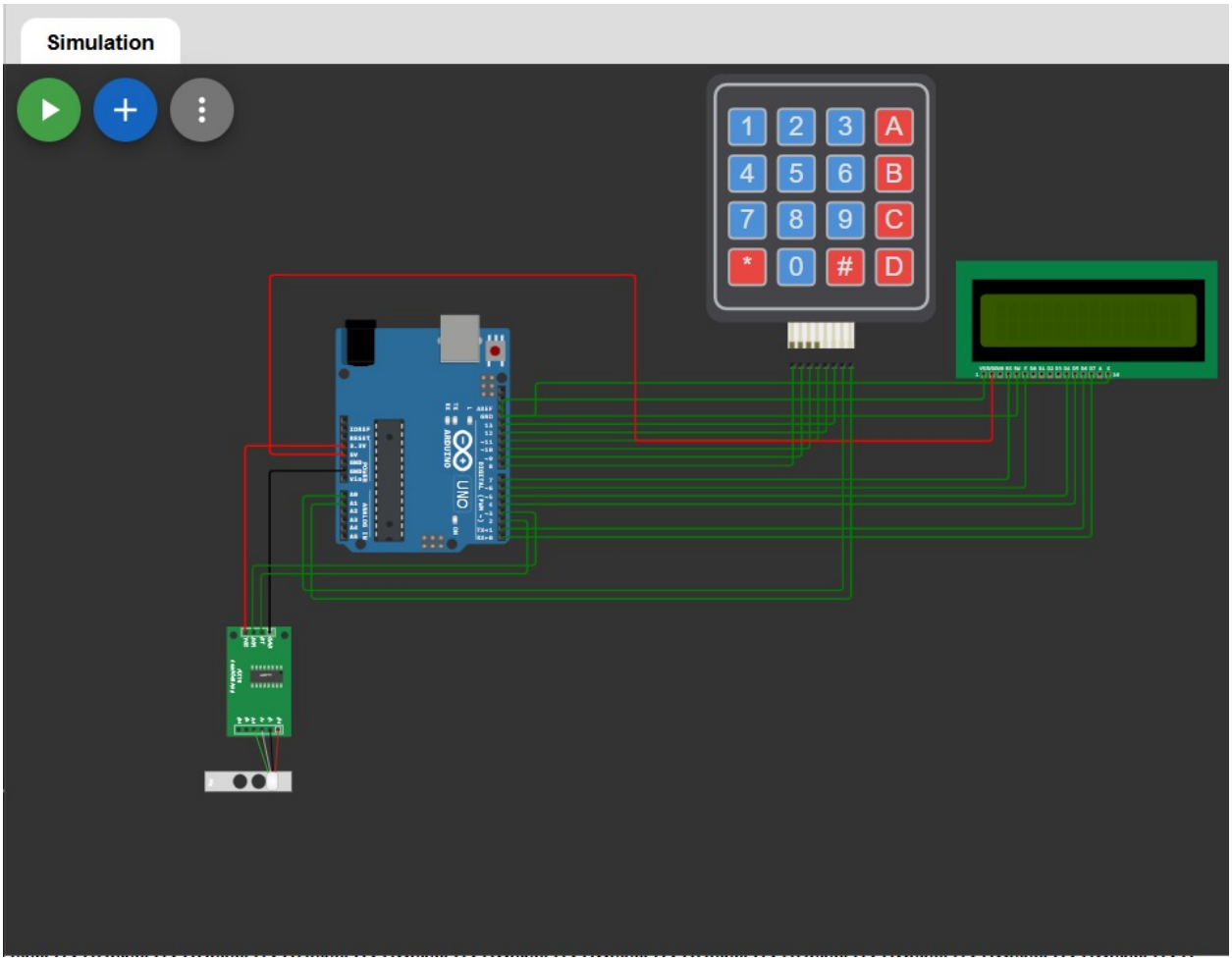
- It takes an `OP_NUM` (operation number) as input, which presumably represents the index of the product to be removed.
- It checks if the product at the specified index is actually present in the order (by checking the `arr_op` array).
- If the product exists:
  - It displays the name and weight of the product being removed on the LCD.
  - It updates the `tPrice` (total price) by subtracting the price of the removed product.
  - It marks the product as removed in the `arr_op` array.
  - It decrements the `INDEX` variable, which likely keeps track of the number of products in the order.

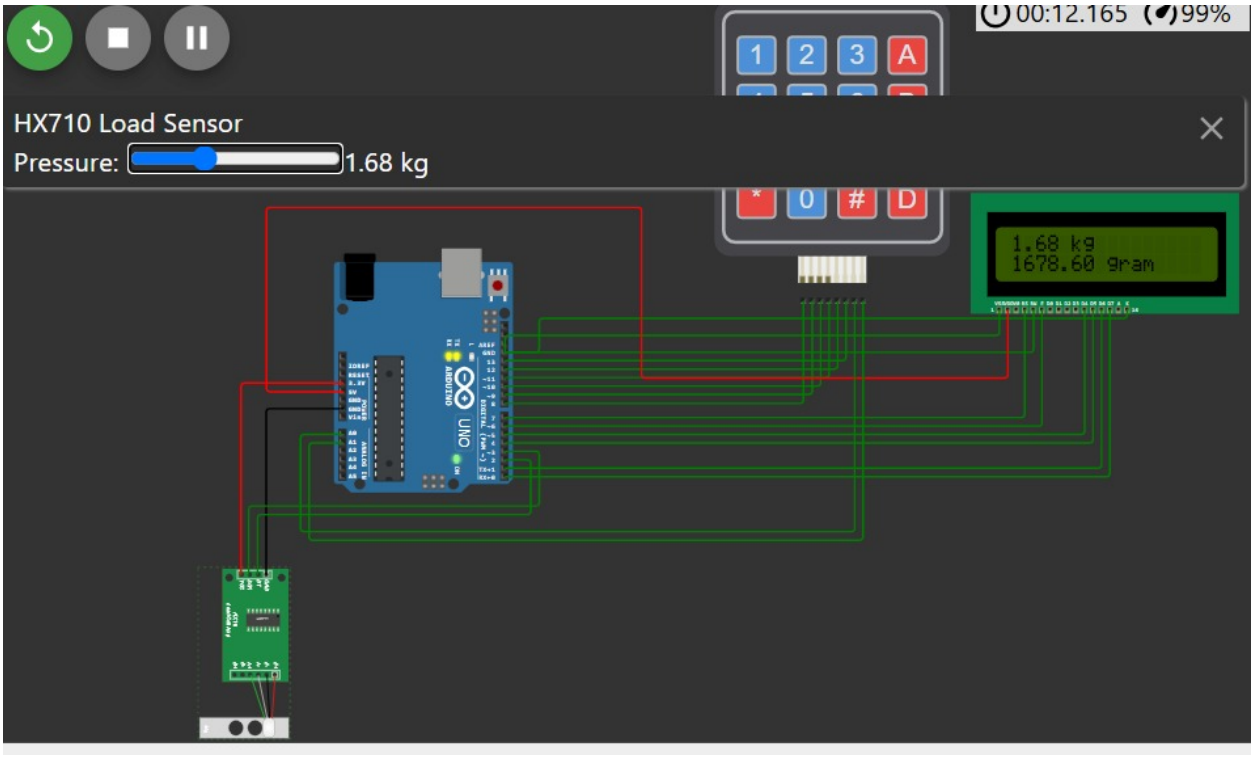
## Report and Project specifications:

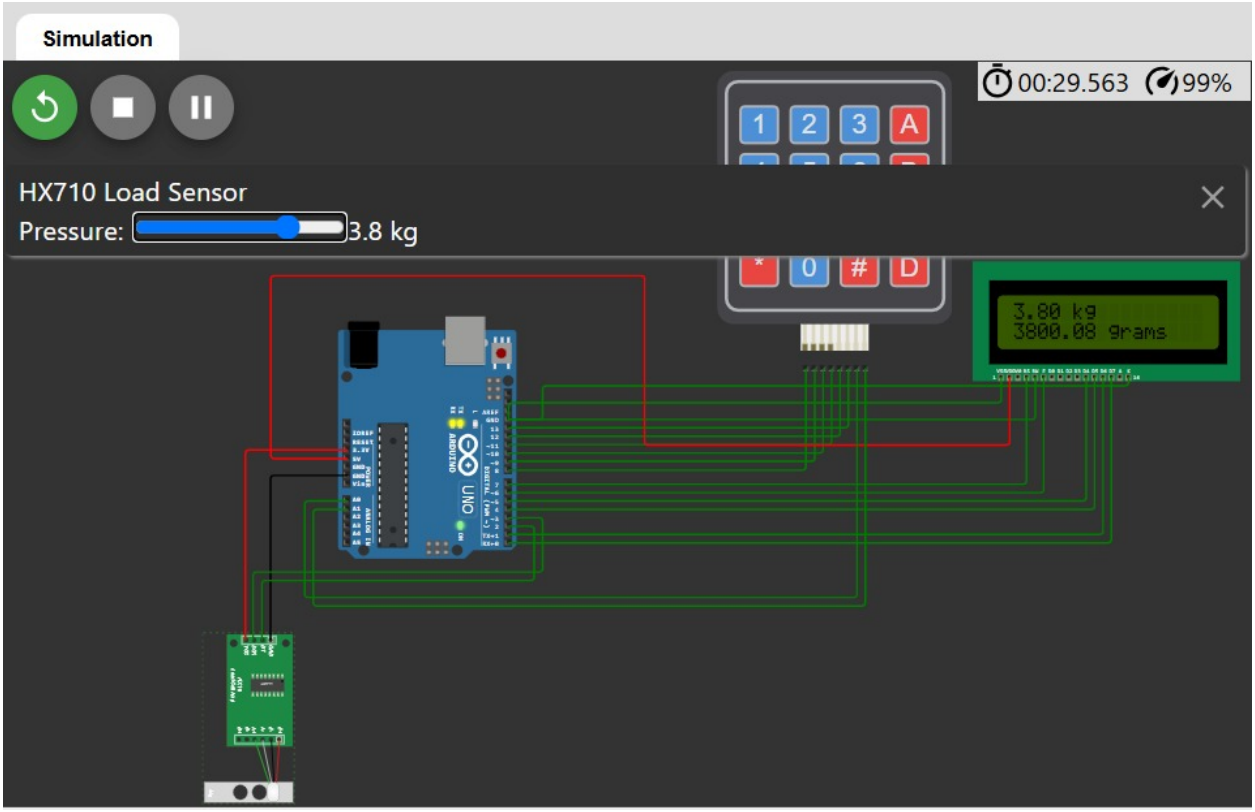
**https://drive.google.com/drive/folders/16Ll0aq3NMlEF-bu3Luwi1Nc7iAufVwlE?usp=sharing**

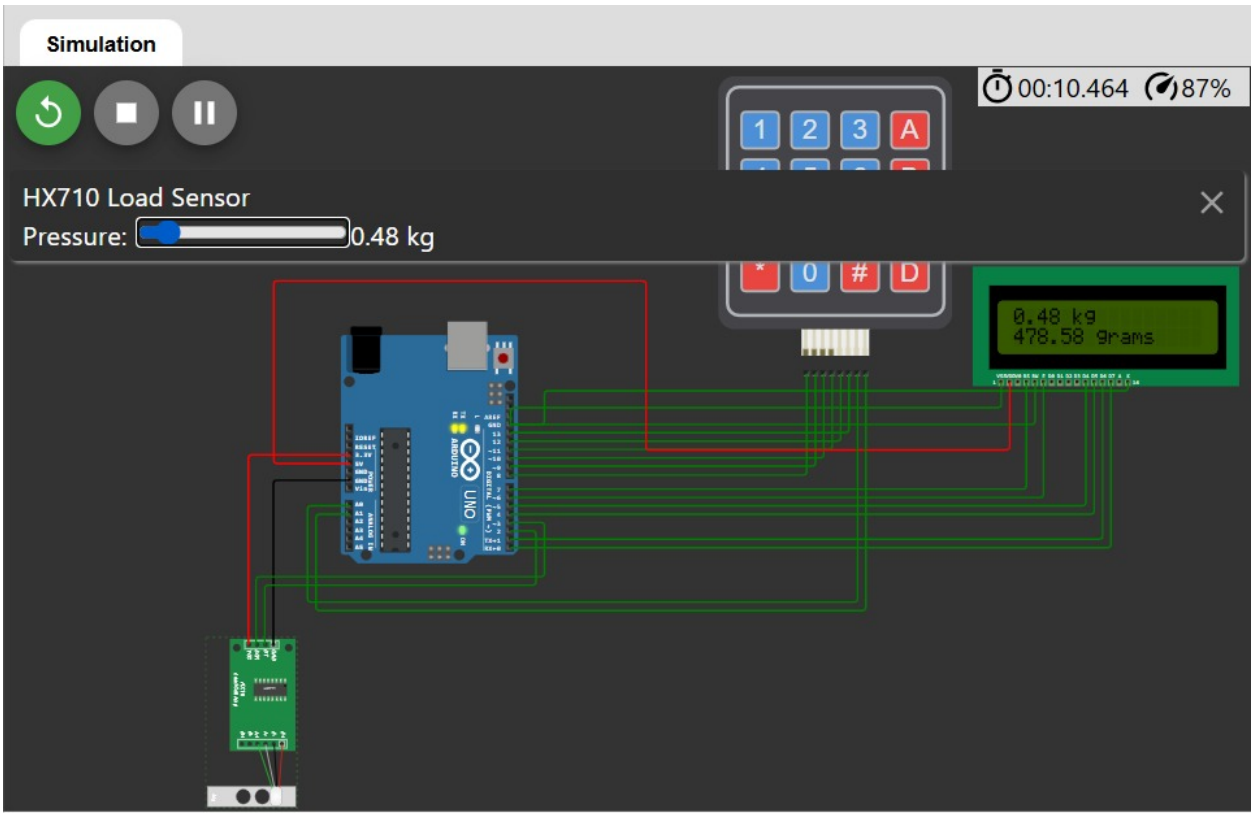
## Simulation

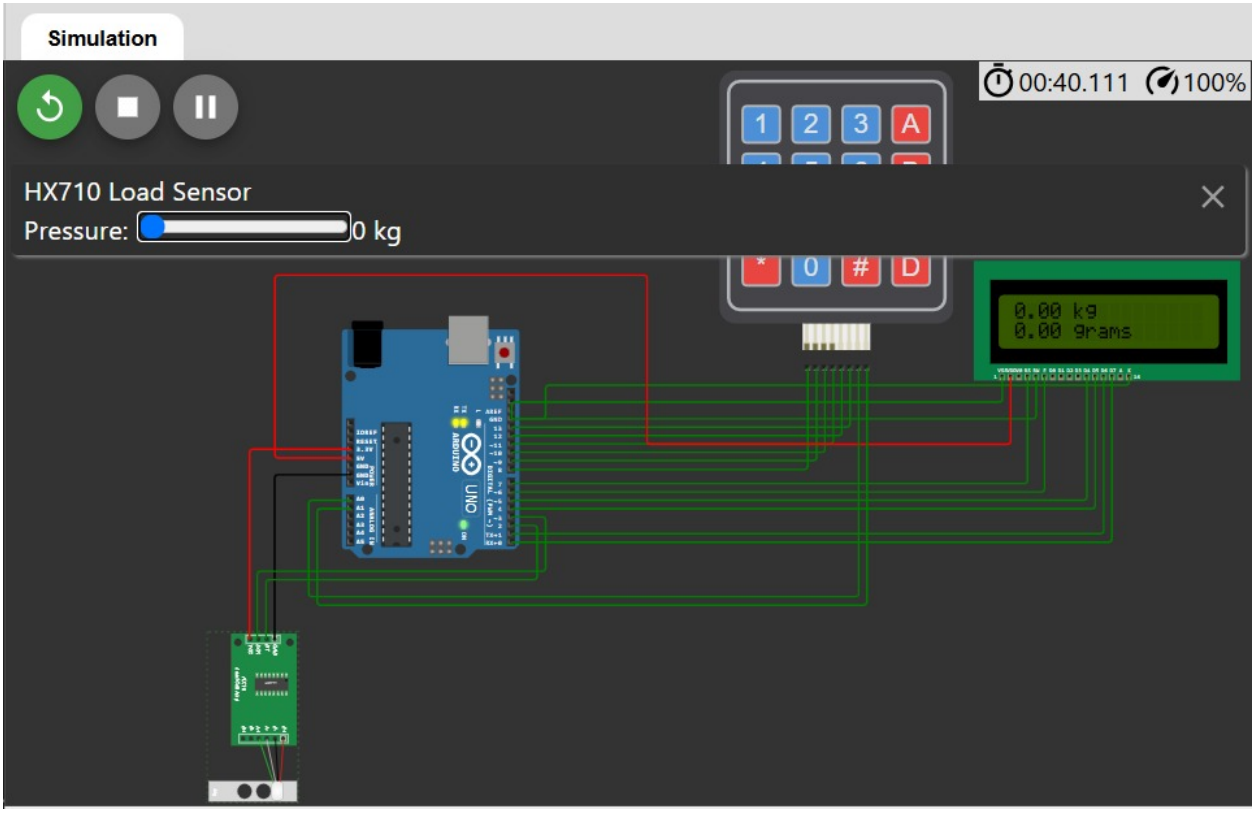
https://wokwi.com/projects/418369321627150337


Simulation
1 2 3 A
4 5 6 B
7 8 9 C
* 0 # D
ARDUINO
UNO


00:12.165
99%
1 2 3 A
HX710 Load Sensor
Pressure:
1.68 kg
* 0 # D
1.68 kg
1678.60 gram


Simulation
00:29.563
99%
1 2 3 A
HX710 Load Sensor
Pressure:
3.8 kg
* 0 # D
3.80 kg
3800.08 grams


Simulation
00:10.464
87%
HX710 Load Sensor
Pressure:
0.48 kg
1
2
3
A
*
0
#
D
0.48 kg
478.58 grams


Simulation
00:40.111
100%
HX710 Load Sensor
Pressure:
0 kg
1
2
3
A
*
0
#
D
0.00 kg
0.00 grams