International Research Journal of Education and Technology
Peer Reviewed Journal
ISSN 2581-7795

WEIGHBRIDGE USING STM32 MICROCONTROLLER

Vinith kumar S[1], Siddiq kesavan S[2], Sivanahul R[3] , Ms Subharathna N[4]

[1,2,3]UG Student, Department of Electronics and Communication Engineering,
[1]Assistant Professor, Department of Electronics and Communication Engineering,
[1,2,3,4] Bannari Amman Institute of Technology

---***---

Abstract

The design and development of the weighbridge using STM32 Microcontroller for measuring the weight of the vehicle using Load cell and STM32 Microcontroller offers a reliable and efficient solution for the weight measurement. Weighbridge plays a crucial role in the industry. In this project, an STM32 microcontroller is used to control the load cell and convert the analog weight measurement data into digital form. The weight measurement data is transmitted to a cloud server using an ESP8266 microcontroller.

The STM32 orchestrates the entire weighbridge system, leveraging its high-resolution analog-to-digital converters (ADCs) and precise timers to ensure the utmost accuracy in weight measurement. The load cell is used to measure the weight of the vehicle. The data from transfer the load cell to STM32 microcontroller using RS232 Serial Communication. The ESP8266 microcontroller is used to transmit the weight measurement data to the cloud.

The use of the STM32 microcontroller and load cell ensures accurate weight measurement, while the ESP8266 microcontroller and Blynk cloud platform provide a convenient way to store and analyze the data. Blynk allows users to log and visualize historical data from their IoT devices, making it easier to track trends and analyze data over time. Blynk serves as the central repository for weight data, offering cloud-based storage, advanced data analytics, and real-time accessibility, revolutionizing how weighbridge data is managed and utilized.

Keyword: STM32, RS232 Serial Communication, Load Cell, ESP8266 Microcontroller, Blynk, Weighbridge.

1. INTRODUCTION

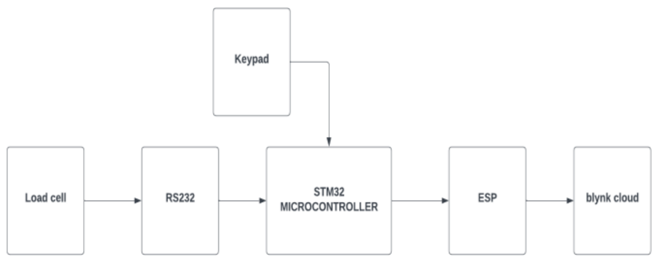
In the modern industrial landscape, precision in weight measurement is of paramount importance across various sectors, ranging from logistics to manufacturing. The success and efficiency of many critical processes hinge on the ability to accurately determine the weight of objects and materials. This project introduces an innovative weighbridge system that leverages advanced technologies to address the challenges associated with traditional weighing methods. At its core, this system utilizes the STM32 microcontroller, load cells, RS232 communication, and the ESP8266 controller to achieve precise weight measurements and efficient data management.

The main goal of this project is to create a dependable weight measurement system that overcomes the challenges of traditional manual weighing methods. To achieve this, we incorporate load cells into the weighbridge setup. These load cells will help us obtain precise and consistent weight measurements. Additionally, we're utilizing the STM32 microcontroller to handle data processing tasks. It will convert the analog weight measurements into digital data instantly, eliminating the chances of errors and inaccuracies that often occur with manual weighing processes.

In this project, we've integrated RS232 communication for gathering data and UART communication to facilitate the smooth transfer of information between the Load cell and the STM32 microcontroller. The ESP8266 controller comes with Wi-Fi capabilities, enabling it to establish a connection between the weighbridge and the Blynk cloud.

1.1 System Information

This innovative system combines the synergy of multiple cutting-edge components to deliver precise weight measurement, efficient data communication, and cloud-based data management. At its core, the system employs load cells strategically placed within the weigh bridge to capture weight data with exceptional accuracy. This data is seamlessly processed and managed by the STM32 microcontroller, which boasts high-resolution analog-to-digital converters (ADCs) and precise timers, ensuring that weight measurements are not only reliable but also consistently accurate.

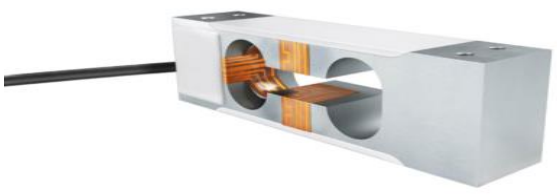


For data communication, the system leverages RS232 serial communication and the ESP module. RS232 serves as a robust means of data transfer within the weighbridge system, enabling efficient and reliable transmission of weight data. The ESP module, on the other hand, extends the system's connectivity to the cloud. It facilitates the integration with The use of the STM32 microcontroller and load cell ensures accurate weight measurement, while the ESP32 microcontroller and Blynk cloud platform provide a convenient way to store and analyze the data. The Blynk cloud platform offers various subscription plans that allow users to access their video clips with reduced-loading times from the cloud. Blynk serves as the central repository for weight data, offering cloud-based storage, advanced data analytics, and real-time accessibility, revolutionizing how weighbridge data is managed and utilized. ,a cloud-based platform that acts as the central repository for weight data. Blynk goes beyond mere data storage, offering advanced analytics and remote accessibility, making it a powerful tool for real-time monitoring and data-driven decision-making. In tandem, these components create a comprehensive system that elevates weighbridge operations to new levels of precision, efficiency, and data management.

2. OVERALL DESIGN OF WEIGHING SYSTEM

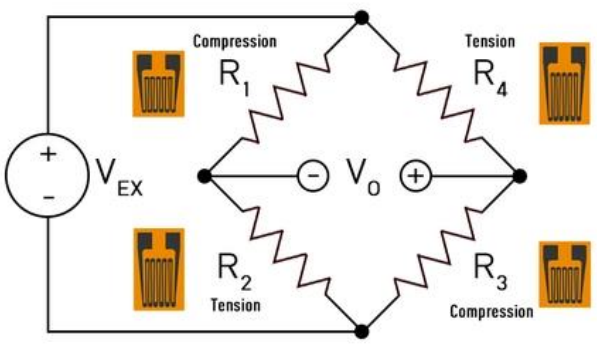
2.1 Load cell:

A load cell is a sensor or a transducer that transforms a load or force acting on it into an electronic signal, which can be a voltage change, current change, or frequency change.



2.2 Wheatstone Bridge:

A Wheatstone bridge is a configuration of four balanced resistors with a known excitation voltage applied as shown below:

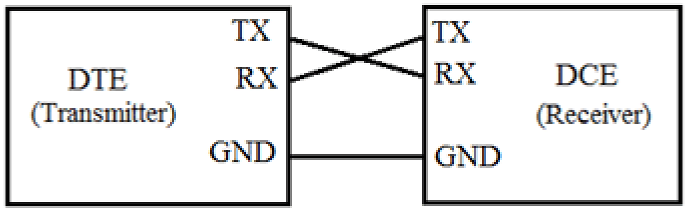


These load cells are composed of a rigid metal structure, also known as a "spring component," with strain gauges securely attached to it. When an external force is applied to the load cell, the spring component undergoes slight deformation, which it quickly recovers from due to its elastic characteristics.
As the shape of the spring component changes, so does the shape of the attached strain gauges. Consequently, the electrical resistance of the strain gauges either increases or decreases. By passing an electric current through the strain gauges, the change in resistance is reflected in the measured voltage output. Since this change in output is directly proportional to the applied force, the weight of the object can be determined based on the observed voltage change.
To ensure that the spring component deflects with minimal permanent deformation, it is essential to minimize the deflection. Relying solely on the small resistance change of a single strain gauge for calculations may result in imprecision and potential errors. To address this, and to achieve a high level of accuracy in load cell measurements, multiple strain gauges are utilized. These strain gauges are arranged in a Wheatstone bridge configuration, allowing for the determination of the overall resistance change across all four strain gauges using Ohm's law and an appropriate equation.

$$V_O = \frac{R_3}{R_3+R_4} - \frac{R_2}{R_1+R_2} \quad V_{EX}$$

2.3 RS-232:

RS232C “Recommended Standard 232C” is the recent version of Standard 25 pin whereas, RS232D which is of 22 pins. In new PC’s Male D-type which is of 9 pins.
In serial communication, RS232 is a standard protocol that is used to connect computers to peripheral devices for data exchange. In this case, it obtains the voltage for the path that will be used to exchange data. With a speed of 1.492 kbps, it is used for serial communications up to 50 feet. Data Transmission Equipment (DTE) and Data Communication Equipment (DCE) are connected to the RS232 according to EIA specifications.



A synchronous data receiver & transmitter (UART) connects to an RS232 port to transfer data between the printer and the computer. Microcontrollers are incapable of handling such voltage levels, so connectors are connected to RS232 signals. DB-9 connectors are serial port connectors and come in two types: male connectors (DTE) and female connectors (DCE).
Working
RS232 operates using bidirectional communication for data exchange. Two devices, namely Data Transmission Equipment (DTE) and Data Communication Equipment (DCE), are interconnected. These devices are equipped with pins such as TXD, RXD, and RTS & CTS. The DTE source initiates a request through RTS to transmit the data, while the DCE source clears the path for data reception by activating CTS. Once the path is cleared, a signal is sent to the RTS of the DTE source to proceed with data transmission. The bits are then transmitted from the DTE to the DCE. Similarly, the DCE source can generate a request through RTS, and the CTS of the DTE source clears the path for data reception, followed by a signal to initiate

data transmission. This entire process facilitates data transmission.

2.4 STM32 Microcontroller:

The STM32 is a family of microcontrollers developed by STMicroelectronics. Microcontrollers are compact integrated circuits that combine a processor (CPU), memory, and various peripherals into a single chip. STM32 microcontrollers support various communication interfaces such as USB, CAN (Controller Area Network), Ethernet, and more, making them suitable for a wide range of connectivity needs. STMicroelectronics provides a comprehensive development ecosystem for STM32 microcontrollers, including development boards, software development kits (SDKs), and integrated development environments (IDEs) like STM32CubeIDE. These tools simplify the process of programming, debugging, and testing STM32-based projects.

Specification:

1. Microcontroller with ARM® 32 bit Cortex™-M0 CPU, frequency up to 48MHz in 48 pin LQFP package
2. These cores are 32-bit RISC (Reduced Instruction Set Computer) processors known for their efficiency and performance.
3. CRC calculation unit, 64Kb flash, 8KB SRAM
4. 2.4V to 3.6V voltage range, power-on/power down reset (POR/PDR)
5. 4 to 32MHz crystal oscillator, internal 40KHz RC oscillator
6. 39 GPIO, 5-channel DMA controller
7. One 12 channels, 12 bit synchronized ADC
8. 1 (16bit) advanced control, 5 (16bit) general purpose and 1 (16bit) basic timer
9. Calendar RTC with alarm and periodic wakeup from stop/standby
10. SPI, I2C and USART communication interfaces
11. Serial wire debug (SWD)

2.5 Blynk IoT Platform

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, visualize it. It is a complete software ecosystem with Smartphone as the Central and Major monitoring as well as a controller component.
The Software system was designed for a wide variety of IoT Hardware including Arduino Platform. The Blynk platform enables any embedded system and DIY Project to have IoT features very practical and efficient to set up.
Blynk provides a cloud-based infrastructure that facilitates communication between IoT devices and the mobile app. This cloud platform acts as an intermediary, allowing access to devices and data. Users don't need to worry about setting up complex networking or server configurations.
Blynk supports a wide range of popular hardware platforms, microcontrollers, and development boards, including Arduino, Raspberry Pi, ESP8266, ESP32, and many others. This versatility makes it suitable for various IoT projects.
Blynk provides secure authentication methods, ensuring that only authorized users can control or access IoT devices. User-generated token or API keys are used to authenticate devices with the Blynk cloud.
Blynk allows users to log and visualize historical data from their IoT devices, making it easier to track trends and analyze data over time.

2.6 ESP32:

The ESP8266 is a versatile and popular microcontroller module designed by Espressif Systems. It is widely used in Internet of Things (IoT) and embedded systems projects due to its powerful features, low cost, and extensive community support. The ESP8266 builds upon the success of its predecessor, the ESP8266, and offers even more capabilities.

Here are some key aspects and features of the ESP8266:

1. Microcontroller and Processor: The ESP8266 is built around a dual-core Xtensa LX6 microcontroller, which includes two processor cores. This dual-core architecture allows the ESP8266 to handle multiple tasks simultaneously, making it suitable for more complex applications.

2. Wireless Connectivity: One of the standout features of the ESP8266 is its built-in support for various wireless communication protocols, including Wi-Fi and Bluetooth. It supports both Wi-Fi 802.11 b/g/n and Bluetooth 4.2 (BLE), making it suitable for IoT applications that require wireless connectivity.

3. Memory: The ESP8266 typically comes with a good amount of Flash memory for program storage and RAM for data storage. The exact memory configuration can vary depending on the specific module.

4. GPIO Pins: ESP8266 modules come equipped with a substantial number of GPIO (General-Purpose Input/Output) pins, which can be used for interfacing with various sensors, displays, and other peripheral devices. These pins are highly configurable and support a variety of digital and analog input/output modes.

5. Peripherals: The ESP8266 includes a rich set of peripherals such as UART, SPI, I2C, PWM controllers, ADCs, DACs, and more. These peripherals enable easy interfacing with external devices and sensors.

6. Built-in Antenna Options: ESP8266 modules are available with different antenna options, including PCB trace antennas and external connectors, to suit different use cases and range requirements.

7. Low Power Modes: The ESP8266 offers various low-power modes, allowing it to conserve energy when not actively processing data. This makes it suitable for battery-powered and energy-efficient IoT applications.
8. Security Features: The ESP8266 includes hardware-based security features such as Secure Boot, Flash Encryption, and cryptographic accelerator to protect against unauthorized access and secure communication.

9. Development Environment: Espressif provides an official development framework called ESP-IDF (Espressif IoT Development Framework) for programming the ESP8266. Additionally, the Arduino IDE supports ESP8266, making it accessible to a wide range of developers.

10. Community and Libraries: The ESP8266 has a thriving and active community of developers and enthusiasts who contribute to its ecosystem. There are numerous libraries, tutorials, and projects available online, making it easier for users to get started with their own IoT projects.

11. Customizable and Modular: The ESP8266 is available in various module formats and configurations, allowing developers to choose the one that best fits their project's requirements.

2.7Applications Of The System

1. Weigh bridges are extensively used in transportation and logistics to determine the weight of trucks, trailers, and their cargo. This helps ensure compliance with weight limits, prevents overloading, and promotes road safety.

2. In agriculture, weigh bridges are employed for weighing harvested crops, livestock, and farm equipment. Farmers use this data for inventory management, crop yield calculations, and livestock management.
3. Weigh bridges play a vital role in the mining and quarrying industries to measure the weight of extracted minerals, rocks, and vehicles carrying them. Accurate weight data is essential for efficient resource management.

4. Waste disposal and recycling centers use weigh bridges to weigh incoming and outgoing waste loads. This data helps with billing, waste disposal fee calculation, and monitoring recycling efforts.

5. In the forestry industry, weigh bridges are employed to measure the weight of logs and timber products. This information assists in timber inventory management and transportation planning.

3. PROPOSED WORK AND MODULES:

With the introduction of the Internet of Things (IoT), a new era of connected technology and intelligent systems that can improve and simplify various aspects of our daily lives has begun. The "Weigh Bridge with STM32" project marks a significant advancement in precision weight measurement and data management. It seamlessly integrates load cells, the STM32 microcontroller, RS232 serial communication, and cloud connectivity, reshaping the landscape of weighing and data handling.

3.1 METHODOLOGY:

3.1.1 Hardware architecture:

1. Collect the necessary components, which are STM32 Microcontroller , Load cell, RS232 serial communication, jumper wire and usb cable to power the stm32 Microcontroller.

2. To power the STM32 microcontroller, a stable and regulated power supply voltage is 12 Volt (DC).

3. Connect the other end of the load cell cable into 230V power supply.

4. Connect the output wires of the load cell to an RS232 serial communication and another end of the RS232 is connected to STM32 microcontroller.

5. By identifying the pins on STM32 and Connecting the pins with ESP32, connect the other end of the ESP32 into the computer.

6. By connecting the Tx pin to PC1 pin and Rx pin to PC0 pin in the STM32 board.

7. Set up my development environment with the STM32cubeIDE.

8. Install the necessary library for the STM32 Microcontroller

3.1.2 Software architecture:

1. Create the new Project file in STM32 CubeIDE.

2. Download The required libraries and select the STM32 board name and give the project name.

3. configure the required pins based on project requirements in Hardware Abstraction Layer .

4. Selecting the pins and assigning the port configuration.

5.In the connectivity USART1 & USART2 are configured to receive the input from RS232 serial communication.

6. After complete the configuration process started to write a code.

7. Get the channel ID and Write API key from the Blynk channel settings. Set up the STM32 board.

8. Write the code to send data to Blynk. The code will need to include the Blynk library and the API key.

9. Upload the code to the STM32 board.

4. RESULTS & DISCUSSION:

The main findings and their consequences after successfully constructing the Weighbridge Using STM 32 Microcontroller.

4.1. System Input validation:

To measure the weight of the component by using the load cell is a major part of the project and an important process of the weighbridge measurement. The data is transferred to the STM32 by using the RS232.

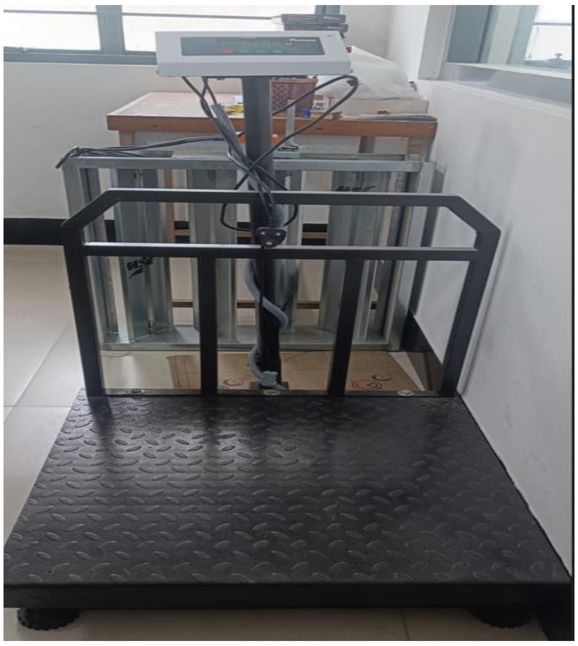


Figure 4.1. Load Cell

4.1.1 Data Transmission & processing:

This RS232 serial communication protocol is used to transmit the data from load cell to STM32, do some calculation and convert the data into digital to analog process.

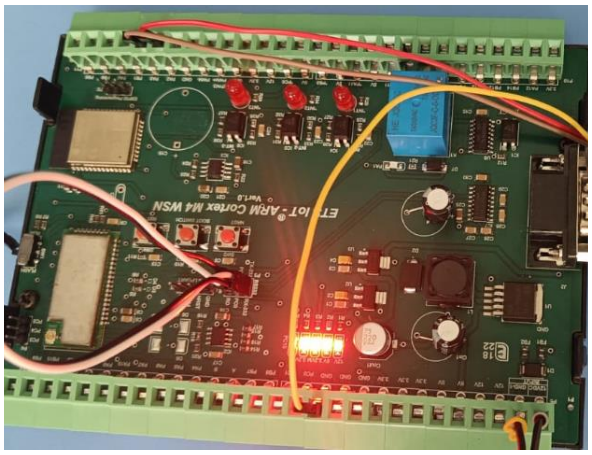
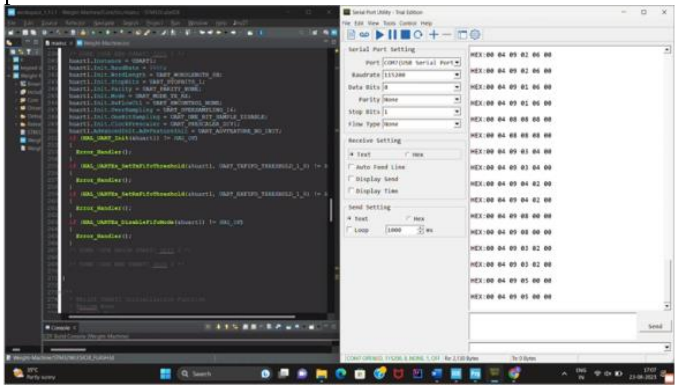


Figure 4.2. STM32 Microcontroller

This is the image that the output is verified by using the serial port software


Serial Port Utility - Trial Edition
File Edit View Tools Control Help
Send

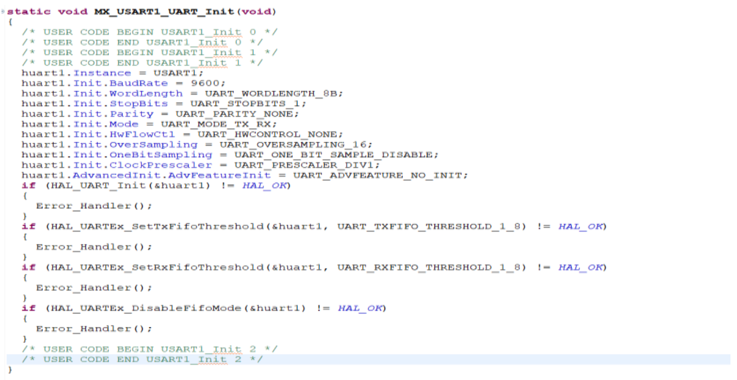
```
Serial Port Setting
Port       COM7(USB Serial Port)
Baudrate   115200
Data Bits  8
Parity     None
Stop Bits  1
Flow Type  None
Receive Setting
(•) Text   ( ) Hex
[ ] Auto Feed Line
[ ] Display Send
[ ] Display Time
Send Setting
(•) Text   ( ) Hex
[ ] Loop   1000 ms
```

```
HEX:00 04 09 02 06 00
HEX:00 04 09 02 06 00
HEX:00 04 09 01 06 00
HEX:00 04 09 01 06 00
HEX:00 04 08 08 08 00
HEX:00 04 08 08 08 00
HEX:00 04 09 03 04 00
HEX:00 04 09 03 04 00
HEX:00 04 09 04 02 00
HEX:00 04 09 04 02 00
HEX:00 04 09 08 00 00
HEX:00 04 09 08 00 00
HEX:00 04 09 03 02 00
HEX:00 04 09 03 02 00
HEX:00 04 09 05 00 00
HEX:00 04 09 05 00 00
```

Figure 4.3. Serial Output Verification by Utility Port

4.1.2. Conversion Process

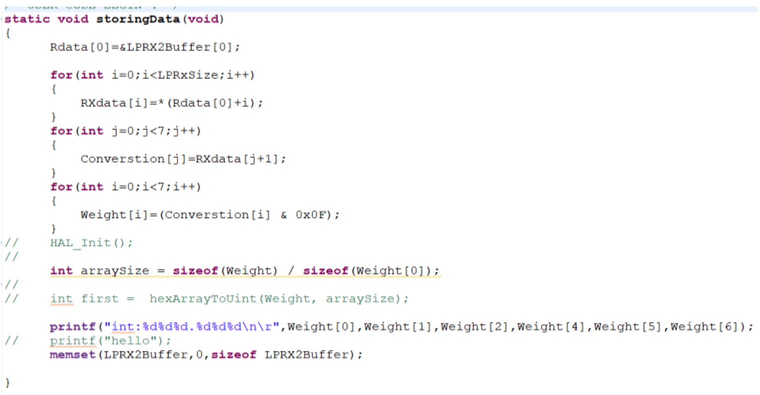
The data received from the load cell by using the RS232 is the process.



```
static void MX_USART1_UART_Init(void)
{
  /* USER CODE BEGIN USART1_Init 0 */
  /* USER CODE END USART1_Init 0 */
  /* USER CODE BEGIN USART1_Init 1 */
  /* USER CODE END USART1_Init 1 */
  huart1.Instance = USART1;
  huart1.Init.BaudRate = 9600;
  huart1.Init.WordLength = UART_WORDLENGTH_8B;
  huart1.Init.StopBits = UART_STOPBITS_1;
  huart1.Init.Parity = UART_PARITY_NONE;
  huart1.Init.Mode = UART_MODE_TX_RX;
  huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
  huart1.Init.OverSampling = UART_OVERSAMPLING_16;
  huart1.Init.OneBitSampling = UART_ONE_BIT_SAMPLE_DISABLE;
  huart1.Init.ClockPrescaler = UART_PRESCALER_DIV1;
  huart1.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
  if (HAL_UART_Init(&huart1) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_UARTEx_SetTxFifoThreshold(&huart1, UART_TXFIFO_THRESHOLD_1_8) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_UARTEx_SetRxFifoThreshold(&huart1, UART_RXFIFO_THRESHOLD_1_8) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_UARTEx_DisableFifoMode(&huart1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN USART1_Init 2 */
  /* USER CODE END USART1_Init 2 */
}
```

Figure 4.4. RS232 Code Snippet

Then the data received and calibrated by checking process we print the converted data to the serial port.



```
static void storingData(void)
{
        Rdata[0]=&LPRX2Buffer[0];

        for(int i=0;i<LPRxSize;i++)
        {
            RXdata[i]=*(Rdata[0]+i);
        }
        for(int j=0;j<7;j++)
        {
            Converstion[j]=RXdata[j+1];
        }
        for(int i=0;i<7;i++)
        {
            Weight[i]=(Converstion[i] & 0x0F);
        }
//      HAL_Init();
//
        int arraySize = sizeof(Weight) / sizeof(Weight[0]);
//
//      int first =  hexArrayToUint(Weight, arraySize);

        printf("int:%d%d%d.%d%d%d\n\r",Weight[0],Weight[1],Weight[2],Weight[4],Weight[5],Weight[6]);
//      printf("hello");
        memset(LPRX2Buffer,0,sizeof LPRX2Buffer);

}
```

Figure 4.5. Code Snippet

4.2. Cloud:

The calibrated data is sent to the Blynk cloud by using the ESP8266 Microcontroller. In this process we use the Arduino IDE to Write a code for an ESP8266 connection image.

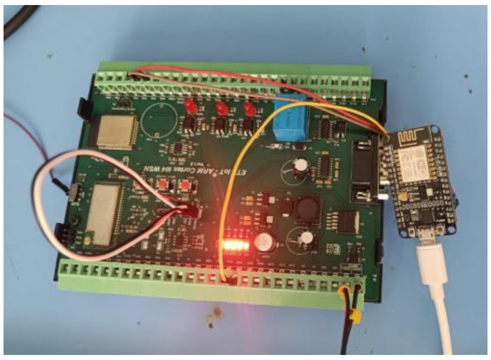
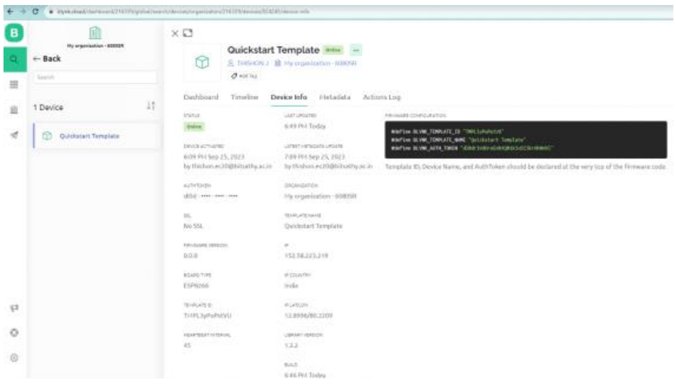


Figure 4.6. STM32 Microcontroller and ESP8266 Interface

4.2.1.Cloud data receiving image:



STATUS
Online

LAST UPDATED
6:49 PM Today

FIRMWARE CONFIGURATION

```
#define BLYNK_TEMPLATE_ID "TMPL3yPoPstVU"
#define BLYNK_TEMPLATE_NAME "Quickstart Template"
#define BLYNK_AUTH_TOKEN "[illegible]"
```

Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code.

DEVICE ACTIVATED
6:09 PM Sep 25, 2023
by thishon.ec20@bitsathy.ac.in

LATEST METADATA UPDATE
7:09 PM Sep 25, 2023
by thishon.ec20@bitsathy.ac.in

AUTH TOKEN
dB0d · •••• · •••• · ••••

ORGANIZATION
My organization - 6080SR

SSL
No SSL

TEMPLATE NAME
Quickstart Template

FIRMWARE VERSION
0.0.0

IP
152.58.223.219

BOARD TYPE
ESP8266

IP COUNTRY
India

TEMPLATE ID
TMPL3yPoPstVU

IP LAT/LON
12.8996/80.2209

HEARTBEAT INTERVAL
45

LIBRARY VERSION
1.3.2

BUILD
6:46 PM Today

Figure 4.7. Blynk IOT Device Configuration

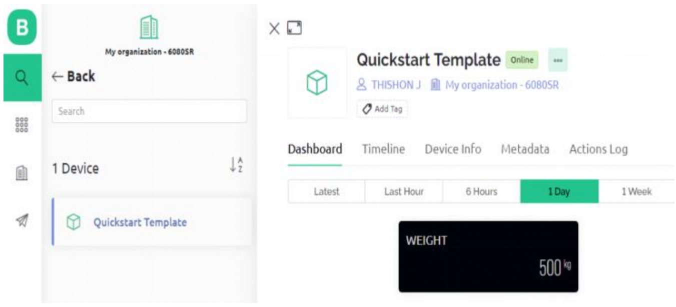


Figure 4.8. Blynk IOT Dashboard

4.3. Discussion

The results obtained from the implementation of the weighbridge system using the STM32 microcontroller, load cells, RS232 communication, and the ESP8266 controller have demonstrated significant achievements in precision, efficiency, and data management.

The system exhibited exceptional accuracy in weight measurements. The integration of load cells, known for their high precision, allowed for reliable and consistent weight data acquisition. The STM32 microcontroller's role in real-time data processing and conversion played a vital role in ensuring the accuracy of weight measurements.

The use of RS232 communication for data acquisition from the load cells and UART communication for data transfer to the ESP8266 controller ensured a seamless flow of information. The ESP8266's Wi-Fi capabilities enabled swift and reliable data transmission to the Blynk cloud, where it could be accessed and analysed in real-time. This efficiency is essential for optimizing weighbridge operations, reducing downtime, and improving decision-making processes.

In conclusion, the results of this project highlight the successful integration of advanced technologies to create a weighbridge system that excels in precision, efficiency, and data management. The combination of the STM32 microcontroller, load cells, RS232 communication, ESP8266 controller, and cloud-based data storage through Blynk has the potential to revolutionize weighbridge operations across various industries, enhancing accuracy, reducing operational costs, and enabling data-driven decision-making.

5. CONCLUSION:

The creation of a weighbridge with a STM 32 Microcontroller to read data from a Load cell sensor and processed by STM 32 Microcontroller has paved the way for a technological advancement that aims to close the accessibility and data precision gap in the field of IoT. By combining a precise tool, the Load cell sensor, RS232 serial communication, the inclusion of load cells as precision sensors is fundamental to achieving accurate and reliable weight measurements. The STM32 microcontroller, celebrated for its computational prowess, takes center stage as the project's central intelligence hub. With high-resolution analog-to-digital converters (ADCs) and robust communication capabilities, it orchestrates data acquisition with finesse, ensuring weight measurements of exceptional precision. Its seamless integration of RS232 serial communication provides a versatile means of real-time data transfer and external device connectivity. Cloud integration, facilitated through Blynk, introduces a dynamic and scalable platform for data storage and analysis. It centralizes weight measurements and transaction data, eliminating the need for complex local storage solutions

Reference:

[1] Yu Zhang , Qiang Zhao, Xuemeng Liang, Huanbo Qiao, "Design of On-board Weighing System Based on STM32", International Journal of Scientific Advances, Volume : 2 | Issues : 2 |Mar – Apr 2021.

[2] Weikeng Lin, Lei Huang, "Design of Port Unattended Weighbridge System Based on Internet of Things" Advances in Engineering Research , Volume 166.

[3] Mega Jaya, Suharjito, Emny Harna Yossy, "Mobile Application Design of Embedded Weighbridge System for Palm Oil Industry ", 2020 International Conference on Information Management and Technology (ICIMTech).

[4] Sichangi M. Sydney ,"Automation of Number Plate and Weight Scale Readings at a CaneFactory Weighbridge through Image and Character Recognition" ,International Journal of Computer Science and Software Engineering (IJCSSE), Volume 7, Issue 3, March 2018.

[5] STMicroelectronics, "ARM® Cortex®-M4 32b MCU+FPU, 225 DMIPS, up to 512kB Flash/128+4KB RAM, USB OTG HS/FS, 17 TIMs, 3 ADCs, 20 comm. interfaces", 2015.

[6]P. Pal, A. K. Singh, and S. D. Gaikwad,"Automated Weighbridge System with SMS Notification"2019 International Conference on Wireless Communications, Signal Processing and Networking.

[7]M. S. Alam, S. J. Rahman, and M. M. S. Islam”Load Cell Based Weighing System with Digital Weight Display”2013 International Conference on Electrical Information and Communication Technology.

[8]A. A. Sawant, M. R. Anuse, and M. A. Sankhe”Wireless Weight Measurement Using Bluetooth and IoT”2018 International Journal for Research in Applied Science & Engineering Technology.