Automatic Cash Payment System Using ESP32

Mohammad Ardhan Fawwaz[1*], Noorman Rinanto[2], Muhammad Khoirul Hasin[3],
Adianto[4],Putri Nur Rahayu[5]
[1,2,3] Automation Engineering Study Program
[4] Marine Electrical Engineering Study program
[5] Business Management Study Program
[1,2,3,4,5] Shipbuilding Institute of Polytechnic Surabaya, Indonesia
[1]mardhan@student.ppns.ac.id, [2]noorman.rinanto@ppns.ac.id, [3]khoirul.hasin@ppns.ac.id
[4]adianto@ppns.ac.id, [5]putri.nur@ppns.ac.id

Abstract -This research evaluates an automatic cash payment system using an ESP32 microcontroller integrated with a bill acceptor, a Rp1000 coin acceptor, and a coin hopper. The system is designed to accept banknote and coin payments and dispense change automatically. The system was fully tested for 40 transactions. Results showed a success rate of 87.5% with the most problems being the incorrect reading of banknotes and coin positioning. The system shows high potential for use in vending machines or self-service platforms

Keywords: *ESP32, cash payment system, bill acceptor, coin acceptor, coin*

I. INTRODUCTION

Vending machines are standalone devices used to distribute products to users after money is inserted. Initially, these machines were only used to sell food, drinks, and other small items. However, along with the development of technology, vending machines are now also used to sell products such as cameras, cell phones, and various other electronic items. Vending machines are commonly found in places with high mobility such as airports, train stations, and tourist attractions. They are especially popular in developed countries such as Japan, the United States, and the United Kingdom. Conventionally, cash is the standard method of payment at vending machines[1]

In the 1880s, the first coin-operated vending machine was introduced in London, England, by Percival Everitt in 1883. These machines were widely found in train stations and post offices to facilitate the purchase of envelopes, postcards and note paper. In Indonesia, vending machines are not very common. Therefore, the real-time implementation of such machines has the potential to provide great benefits to society. The advantages of these machines include efficiency, no need for human supervision, ease of maintenance, portability, and ease of use[1].

One of the important elements in vending machines is the automatic cash payment system, which is also a major component in other self-service such as vending machines. The ESP32 microcontroller is the main choice for the development of this system because it has completed digital communication features, is power efficient, and is able to be integrated with various additional devices.

Efficiency in cash payment transactions is a major challenge in sectors with high transaction volumes, such as retail stores, restaurants, and public transportation. Manual processes that involve counting cash and giving change are often time-consuming, prone to human error, and have the potential to decrease customer satisfaction. In addition, there are few conventional stores that provide candy returns[2], therefore, there is a need for an automated payment system that can overcome these challenges quickly, accurately, and reliably.

This research aims to develop and evaluate an ESP32 microcontroller-based automatic cash payment system for vending machines. The system is designed to accept rupiah banknotes, recognize the nominal value of the banknotes, ensure their authenticity, calculate the transaction amount, and provide change in the form of Rp1,000 coins. By integrating three main components, namely bill acceptor, coin acceptor, and coin hopper, the system is expected to improve efficiency and accuracy in cash transactions.

On the other hand, other studies have designed a device that uses color sensors to recognize the value of banknotes. However, the weakness of this device is that it only uses black and white color sensors[2], therefore the bill acceptor in this system is also equipped with the ability to detect counterfeit bills based on physical parameters such as magnetism, ultraviolet light, and infrared. Although this feature is not the main focus, it provides an additional layer of security to ensure reliable transactions.

This research is expected to make a real contribution to the development of vending machines in Indonesia, especially by utilizing technology to improve the quality of self-service and meet the needs of an increasingly dynamic society.

II. METHODS

Microcontroller ESP 32

ESP32 microcontroller is an integrated SoC (System on Chip) microcontroller equipped with WiFi 802.11 b/g/n, Bluetooth version 4.2, and various peripherals, ESP32 is a fairly complete chip, there is a processor, storage and access to GPIO (General Purpose Input Output). ESP32 can be used for a replacement circuit on Arduino, ESP32 has the ability to support connecting to WI-FI directly[3]

Coin Acceptor

Figure 1 Coin Acceptor

Figure 1 is the shape of the coin acceptor used in this research; the coin acceptor device receives coins through the coin insertion slot. The sensor inside the coin receiver device will recognize and validate the coin based on the diameter of the coin. When the inserted coin is validated, it will send a high signal of 5-12 volts, otherwise, the coin will be returned[4].

Coin Hopper

Figure 2 shows the shape of the coin hopper used in this research, the coin hopper is an electronic device that can remove coins with a maximum dimension of 26mm, with a rotating motor drive, this device is also equipped with sending a high signal of 5volt when the coin comes out.

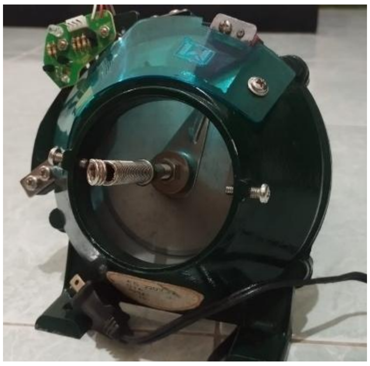

Figure 2.Coin Hopper

Bill Acceptor

Figure 3 is the type or type of bill acceptor used, Bill Acceptor is a money reader machine that is able to detect counterfeit money, the machine will reject if it is considered counterfeit / damaged. if reading the original nominal money, the machine will send a signal in the form of the nominal value of the currency. This machine has quite complete features, because it can distinguish the nominal banknotes and this machine is also equipped with a Reverse feature that is a system that will try to tidy up the damaged / worn money that is stuck inside to be issued again automatically.

Figure 3. Bill Acceptor

LCD 16X2

LCD (Liquid Crystal Display) screens are electronic components that are often used to display information in the form of text or numbers in various applications, such as household devices and automatic control systems. One common type is the 16x2 LCD, which can display up to 16 characters in two lines at once. This LCD has two main registers: a command register to set up work functions, such as cursor position or display mode, and a data register to store the ASCII code of the characters to be displayed. This capability makes it ideal for displaying real-time data in microcontroller-based systems such as the ESP32, with the advantages of power efficiency, ease of integration, and flexibility of use in various environmental conditions[5]

Relay

Relay is an electromechanical device that functions as an automatic switch, operated using a relatively low level of electrical voltage. This component can be used to control higher voltage circuits or larger currents, so it is often utilized in electronic and automation systems. The working principle of a relay is based on an electromagnet, which is the core of its mechanism[6]. This electromagnet consists of a coil of wire that produces a magnetic field when electrified. The magnetic field attracts a lever or mechanical contact inside the relay, which then opens or closes the circuit as needed. With its ability to isolate and control circuits, relays have become an important component in various applications, such as automatic control systems, electrical protection, and high load control devices.

Arduino Ide

Arduino IDE (Integrated Development Environment) is software used to develop integrated programming logic on various types of Arduino-based hardware. With the Arduino IDE, users can write code, compile it into binary code, and upload it directly to the microcontroller's memory. This environment is designed to simplify the process of developing microcontroller-based devices, especially for novice users. The programming language used in the Arduino IDE is the C language, which allows designing logic to manage inputs and outputs on hardware with great flexibility [7]. The existence of the Arduino IDE makes it an efficient platform for creating prototypes and applications that utilize microcontroller-based systems.

Electronic Circuits

Figure 4 is a block diagram of the electronic circuit used, ESP32 acts as a control center that reads pulse signals from bill acceptors and coin acceptors. Each unit generates digital pulses that represent the value of money received. These pulses enter the digital input pins on the ESP32 which are then processed to calculate the total payment.

To activate the coin hopper, a relay is used which is controlled via one of the ESP32 output pins. When the system detects that change needs to be given, the ESP32 sends a logic high signal to activate the coin hopper for the required number of pulses. The system is powered by 12V for the bill acceptor and coin hopper, and 5V for the ESP32 through a step-down module.

The system is also equipped with a 16x2 I2C LCD that serves to display information on the amount of money that has been received, payment status, and a description if the payment is successful or change is required. The I2C LCD is connected to the ESP32 via I2C communication (SDA and SCL), and works on 5V voltage.

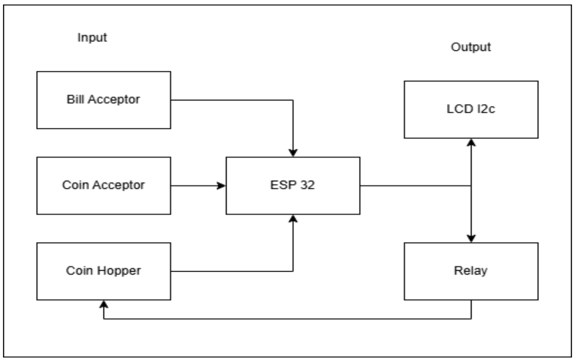

Figure 4.Circuit Block Diagram

Logic design of Payment System

We can see the flowchart of the system workflow in Figure 5, This automatic cash payment system works with the principle of reading pulses from the money receiver and issuing coin change through microcontroller control. When a user inserts paper money through the bill acceptor or Rp1000 coin through the coin acceptor, each device sends a pulse to the ESP32 which is then interpreted as a certain nominal value. The total amount of money inserted is calculated accumulatively by the ESP32.

If the amount of money received is greater than the service price, the system will automatically calculate the change and activate the coin hopper to dispense coins according to the change value. During this process, information on the amount of money received, the amount of change, as well as the payment status will be displayed in real-time on the I2C LCD screen connected to the ESP32 via I2C communication. With this display, the user can know the transaction status directly. The entire system logic was developed using the Arduino C/C++ programming language and uploaded to the ESP32 through the Arduino IDE platform.

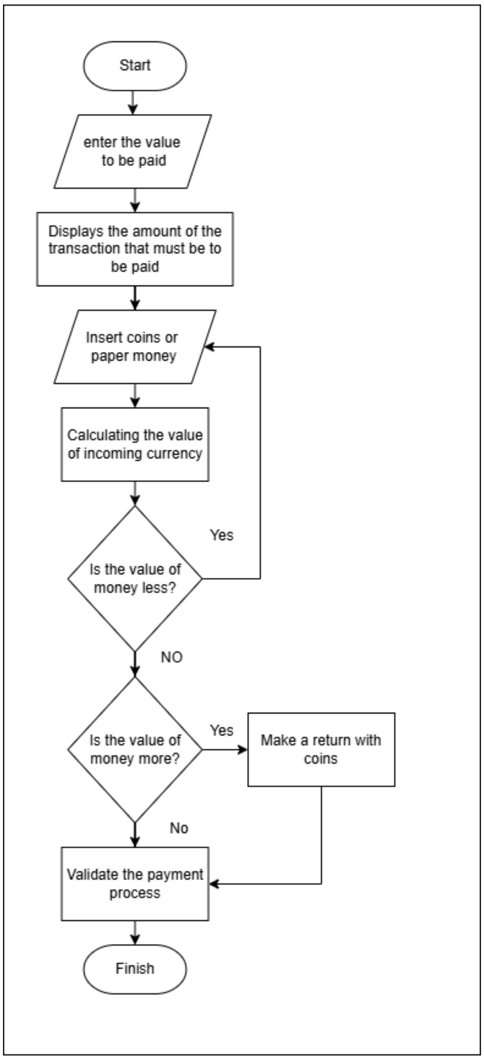

Figure 5.System workflow flow chart

Testing Method

System testing was conducted to evaluate the performance of each component and the overall automated cash payment system that has been designed. 40 transactions were conducted with different scenarios involving a combination of banknotes, coins, and a mixture of both. The main objective of the testing was to assess the success rate of the system in reading the nominal amount of the incoming money, calculating the total amount of payment, as well as determining and issuing the appropriate amount of change.

During the testing process, each transaction was observed and recorded manually to identify any system errors, either from the pulse reading, the processing logic process on the microcontroller, or errors in the coin hopper's dispensing of change. The results of these tests were then analyzed to determine the level of accuracy and reliability of the system, as well as to identify technical problems that arose during the operational process. With this approach, data is obtained that represents the performance of the system as a whole in real conditions, which is the basis for evaluating the effectiveness of the electronic circuit design and system control software.

III. RESULT AND DISCUSSION

Coin Acceptor Testing

The coin acceptor test was conducted partially to determine the device's ability to detect and distinguish metal objects that enter the coin receiver channel. The test was conducted on two types of metal, namely real coins of Rp1,000 and brass metal that has the same diameter and dimensions as the coin. Each metal was tested 10 times to determine the accuracy of pulse detection by the ESP32 microcontroller.

To detect pulses from the coin acceptor, a simple program is used in the Arduino IDE that reads the digital status on GPIO pin 13. When HIGH logic is detected on the pin, the ESP32 prints the number "1000" on the serial monitor as a sign that one coin is detected. The following is the Arduino code used for testing:

```
const int coin = 13;
void setup() {
  Serial.begin(9600);
  pinMode(coin, INPUT);
}
void loop() {
  int nilai = digitalRead(coin);
  if (nilai == HIGH) {
    Serial.println("1000");
    delay(200);
  }
}
```

Table 1.Coin Acceptor Testing Results

No	Coin Type	Test Amount	Escaped	Not Passed
1	Rp1.000	10	10	0
2	Brass metal	10	0	10
	Total	20	10	10

Table 1 is the test result of the coin acceptor, the test results show that the coin acceptor can detect and accept Rp1,000 coins with 100% accuracy, and is able to reject 100% of brass metals that resemble the physical shape of real coins. This proves that the sensor has the ability to be selective to the type of metal received, and can be relied upon in preventing the use of counterfeit objects during the transaction process.

Coin Hopper Testing

Coin hopper testing is done to ensure that the device can dispense coins according to the amount commanded via serial communication. This system uses an ESP32 microcontroller connected to a relay to activate the coin hopper motor, as well as a pulse sensor to count the number of coins that have come out. Each time a coin is successfully ejected, a pulse is received through the digital pin and the number of coins is recorded.

Testing is done by entering a target number (e.g. 1-9) into the Arduino IDE serial monitor. The number will be a reference for the ESP32 to activate the relay and move the coin hopper until the number of high signals read from the input pin reaches the intended number. Once the target is reached, the relay is automatically turned off to stop coin dispensing. Here is the arduino code idea for testing:

```
#define hopper 26
#define RELAY_PIN 27
int hopperPressCount = 0;
bool relayState = false;
unsigned long lastInputTime = 0;
const unsigned long timeout = 10000;
int targetNumber = 0;
void setup() {
 pinMode(hopper, INPUT);
 pinMode(RELAY_PIN, OUTPUT);
 digitalWrite(RELAY_PIN, LOW);
 Serial.begin(9600);
}
void loop() {
 if (Serial.available() > 0) {
  String input = Serial.readStringUntil('\n');
  input.trim();
  targetNumber = input.toInt();
  if (targetNumber != 0) {
   relayState = true;
   digitalWrite(RELAY_PIN, HIGH);
   lastInputTime = millis();
  }
 }
 if (relayState && (millis() - lastInputTime > timeout)) {
  relayState = false;
  digitalWrite(RELAY_PIN, LOW);
 }
 int hopperState = digitalRead(hopper);
 if (hopperState == HIGH) {
  hopperPressCount++;
  delay(100);
  if (hopperPressCount == targetNumber) {
   relayState = false;
   digitalWrite(RELAY_PIN, LOW);
   hopperPressCount = 0;
  }
 }
}
```

We can see the coin hopper test results in Table 2, the test results show that the system can issue the number of coins according to the command with an average success rate of 96.67%. A small percentage of failures occurred at low target values (1 or 3 coins), which may be caused by a delayed sensor response or a poorly detected high signal due to the coin output speed being too fast and resulting in coins being counted in the next process and resulting in less coins coming out in the next process. Overall, the coin hopper works reliably and responsively, and can be used as an automatic return system in cash payment applications.

Table 2.Coin Hopper Testing Results

No	Desired value	Test Amount	Success	Fail	Success Percentage
1	1	10	8	2	80%
2	2	10	10	0	100%
3	3	10	9	1	90%
4	4	10	10	0	100%
5	5	10	10	0	100%
6	6	10	10	0	100%
7	7	10	10	0	100%
8	8	10	10	0	100%
9	9	10	10	0	100%
	Total	90	35	3	average = 96,67%

Bill Acceptor Testing

The bill acceptor test aims to evaluate the ability of the device to read and distinguish the face value of Rupiah banknotes circulating in Indonesia, especially the latest versions. The test was conducted by inserting banknotes from Rp1,000 to Rp100,000 into the bill acceptor and observing the pulse output generated for each nominal. The ESP32 microcontroller is used to count the number of incoming pulses through interrupts and determine the value of the money based on a certain range of pulses.

To accurately detect pulses, the interrupt function on the ESP32 digital pins is used. Pulse is counted using countPulse() which will be triggered every time the pulse is rising. After the pulse stops for a certain time (300 ms), the system will count the total pulses and determine the amount of money based on the pulse classification that has been determined in the program. The following is the program code used to test the bill acceptor:

```
const int pulsePin = 35;
volatile unsigned long pulseCount = 0;
unsigned long lastPulseTime = 0;
const unsigned long timeout = 300;
bool counting = false;

void IRAM_ATTR countPulse() {
  pulseCount++;
  lastPulseTime = millis();
  counting = true;
}
void setup() {
  Serial.begin(9600);
  pinMode(pulsePin, INPUT);
  attachInterrupt(digitalPinToInterrupt(pulsePin), countPulse, RISING);
}
void loop() {
  if (counting && millis() - lastPulseTime > timeout) {
    noInterrupts();
    unsigned long totalPulses = pulseCount;
    pulseCount = 0;
    counting = false;
    interrupts();
    if (totalPulses >= 1 && totalPulses <= 2)
      Serial.println("Uang: Rp 1.000");
    else if (totalPulses >= 3 && totalPulses <= 5)
      Serial.println("Uang: Rp 2.000");
    else if (totalPulses >= 6 && totalPulses <= 10)
      Serial.println("Uang: Rp 5.000");
    else if (totalPulses >= 11 && totalPulses <= 19)
      Serial.println("Uang: Rp 10.000");
    else if (totalPulses >= 20 && totalPulses <= 49)
      Serial.println("Uang: Rp 20.000");
    else if (totalPulses >= 50 && totalPulses <= 90)
      Serial.println("Uang: Rp 50.000");
    else if (totalPulses >= 91)
      Serial.println("Uang: Rp 100.000");
    else
      Serial.println("Pulsa tidak valid!");
  }
}
```

Table 3.Bill Acceptor Testing Results

No	Currency value	Test Amount	Read	Illegible	Success Percentage
1	Rp1.000	10	10	0	100%
2	Rp2.000	10	8	2	80%
3	Rp5.000	10	10	0	100%
4	Rp10.000	10	10	0	100%
5	Rp20.000	10	10	0	100%
6	Rp50.000	10	10	0	100%
7	Rp100.000	10	10	0	100%
	Total	70	68	2	average = 97,14%

Table 3 shows the results of partial testing on the bill acceptor, showing that the system is able to recognize most banknotes with high accuracy. Two cases of failure occurred with the IDR 2,000 note, which may be due to the physical condition of the note or variations in the sensitivity of the sensor readings. The average reading success is 97.14%, indicating that the bill acceptor works very well in the designed system.

Whole System Testing

This test is carried out after all components, namely bill acceptor, coin acceptor, and coin hopper, have successfully functioned partially. The system was tested in an integrated condition as an ESP32-based automatic cash payment system by entering the transaction value on the arduino ide serial monitor. The user enters the payment amount through banknotes, coins, or a combination of both. If the payment amount exceeds the target, the system will automatically calculate the change and activate the coin hopper to dispense the required number of coins.

The program was developed using the Arduino IDE and involves the integration of logic to read pulses from the bill acceptor via interrupt, read digital input from the coin acceptor, calculate the total money in, compare with the target, and control the dispensing of change via relays to the coin hopper. The system also has an auto-return feature that only activates if the payment amount is greater than the target amount, as well as being able to handle combined coin and bill input scenarios. Below is the arduino program for the whole system:

```
#define HOPPER_PIN 26
#define RELAY_PIN 27
const int coinPin = 13;
unsigned long lastInputTime = 0;
const int pulsePin = 35;
volatile unsigned long pulseCount = 0;
unsigned long lastPulseTime = 0;
const unsigned long timeout = 300;
bool counting = false;
int target = 0;
bool pembayaranSelesai = false;
float totalUangMasuk=0;
bool relayState = false;
int HopperCount = 0;
const unsigned long hoptimeout = 10000;
 int kembalian =0;
 int pembagi=0;
 int gabungan=0;
```

```
 int total = 0;

void IRAM_ATTR countPulse() {
  pulseCount++;
  lastPulseTime = millis(); // Update waktu terakhir pulsa diterima
  counting = true;
}

void setup() {
  Serial.begin(9600);
  pinMode(pulsePin, INPUT);
  pinMode(HOPPER_PIN, INPUT);
  pinMode(RELAY_PIN, OUTPUT);
  pinMode(coinPin, INPUT);
  digitalWrite(RELAY_PIN, LOW);
  attachInterrupt(digitalPinToInterrupt(pulsePin), countPulse, RISING);

  Serial.println("Masukkan Nilai Uang");
}

void loop() {

  ///pembacaan serial monitor
    if (Serial.available() > 0 && target == 0) {
    String input = Serial.readStringUntil('\n');
    input.trim();
    target = input.toInt();

    if (target % 1000 != 0 || target <= 0) {
      Serial.println("Input tidak valid. Harus kelipatan 1000 dan > 0.");
      target = 0;
    } else {
      Serial.print("Target pembayaran: Rp ");
      Serial.println(target);
       }
      }


  if (counting){
    if (millis() - lastPulseTime > timeout) {
      noInterrupts();
      unsigned long totalPulses = pulseCount;
      pulseCount = 0;
      counting = false;
      interrupts();

      Serial.print("Pulsa selesai. Jumlah pulsa: ");
      Serial.println(totalPulses);
     float UangMasuk = konversiPulsaKeUang(totalPulses);

    Serial.println(UangMasuk);

    totalUangMasuk += UangMasuk ;
    Serial.println(totalUangMasuk);
    delay(300);
    }
  }
   if (totalUangMasuk == target && target > 0  ) {
    Serial.println("Pembayaran sukses.");

    delay(1000); // Tunda sejenak sebelum reset
    resetTransaksi(); // Reset transaksi
  }
    if (digitalRead(coinPin) == HIGH && !pembayaranSelesai && target >
0) {
    total += 1000;
    Serial.print("Koin diterima. Total: Rp ");
    Serial.println(total);
    delay(300);  // Debounce
  }
   if (total >= target && target > 0 && !pembayaranSelesai) {
    Serial.println("Pembayaran sukses.");
    pembayaranSelesai = true;
    delay(1000); // Tunda sejenak sebelum reset
    resetTransaksi(); // Reset transaksi
  }
  gabungan = totalUangMasuk + total;
    if (gabungan == target && target > 0  ) {
      Serial.println("Pembayaran sukses.");
      delay(1000); // Tunda sejenak sebelum reset
      resetTransaksi(); // Reset transaksi
    }
  if (totalUangMasuk > target && relayState == false) {
    kembalian = totalUangMasuk - target;
    pembagi = kembalian / 1000;
    Serial.print("Kembalian: Rp ");
    Serial.println(kembalian);

    relayState = true;
    HopperCount = 0; // Reset perhitungan hopper
    digitalWrite(RELAY_PIN, HIGH);
    lastInputTime = millis();
  }

  if (relayState) {
    if (millis() - lastInputTime > hoptimeout) {
      relayState = false;
      digitalWrite(RELAY_PIN, LOW);
      Serial.println("Relay dimatikan karena timeout.");
       delay(1000);
      resetTransaksi();
    }
      int hopperState = digitalRead(HOPPER_PIN);
    if (hopperState == HIGH) {
      HopperCount++;
      Serial.print("Koin keluar (hopper): ");
      Serial.println(HopperCount);
      delay(100); // debounce sederhana
      if (HopperCount >= pembagi) {
        relayState = false;
        digitalWrite(RELAY_PIN, LOW);
        Serial.println("Relay dimatikan: Jumlah kembalian terpenuhi.");
         delay(1000);
        resetTransaksi();
      }
    }
  }
  }

  int konversiPulsaKeUang(float pulsa) {
    // Langsung ubah ke nilai uang sesuai jumlah pulsa
    if (pulsa >= 1 && pulsa <= 2) return 1000;
    if (pulsa >= 3 && pulsa <= 5) return 2000;
    if (pulsa >= 6 && pulsa <= 10) return 5000;
    if (pulsa >= 11 && pulsa <= 19) return 10000;
    if (pulsa >= 20 && pulsa <= 49) return 20000;
    if (pulsa >= 50 && pulsa <= 90) return 50000;
    if (pulsa >= 91) return 100000;
    return 0;
  }
  void resetTransaksi() {
    target = 0;
    totalUangMasuk = 0;
    kembalian=0;
     total=0;
     pembayaranSelesai = false;
    Serial.println("\n=== Transaksi baru ===");
    Serial.println("Masukkan jumlah pembayaran (misalnya: 5000):");
  }
```

Table 4. System Testing Results

No	Transaction Scheme	Test Amount	Success	Failed	Success Percentage
1	1 sheet IDR10,000	10	9	1	90%
2	3 coins of IDR 1,000	10	10	0	100%
3	1 sheet IDR5,000 + change	10	10	0	100%
4	Mixture of IDR 3,000 (IDR 2,000 note+1 coin)	10	6	4	60%
	Total	40	35		Average = 87,5%

Table 4 shows the results of the overall cash payment system testing, from a total of 40 transactions, the system successfully completed 35 transactions correctly and experienced 5 failures, resulting in an average success rate of 87.5%. The Rp1000 coin transaction scheme had the highest success rate of 100%, followed by the Rp5,000 payment and return scenario which also achieved 100% success. The scenario with a Rp10,000 note experienced one failure, when the note was read as a lower amount, possibly due to the poor physical condition of the note. Meanwhile, the mixed scenario (banknotes and coins) showed the lowest success rate of 60%, which was due to the misreading of a Rp2,000 note identified as Rp1,000 by the system.

Further analysis shows that the system performs very well in reading pulses from the coin acceptor and controlling the coin hopper, but still has limitations in the accuracy of identifying the value of banknotes under certain conditions. The coin hopper component performed smoothly in all tests that required a refund, with no cases of coins not coming out or jamming. In addition, the I2C LCD used to provide feedback to the user was able to display information correctly and in real-time, such as payment amount, transaction status, and change amount.

Overall, the system performed well enough and reliably to be used in small-scale automated cash payment scenarios. However, there are still some technical aspects that need to be improved, especially in terms of improving the accuracy of banknote reading and physical guidance to ensure that the banknotes are in the correct position.

IV. CONCLUSION

This research successfully designed and implemented an automatic cash payment system based on ESP32 microcontroller that integrates several main components, namely bill acceptor, Rp1000 coin acceptor, coin hopper, and I2C LCD as information display media. The results of testing 40 transactions with various scenarios show that the system has an average success rate of 87.5%. The success reflects the system's ability to accurately read pulses from various types of inputs, process total payments, automatically issue change, and display real-time transaction status to users via LCD.

The system has a number of advantages, including a modular design, programming flexibility thanks to the use of ESP32, and fast response time to inputs. However, there were some weaknesses identified during testing, such as the sensitivity of the banknote reading that still needs to be improved, especially for certain denominations, and the dependence of the system's performance on the physical position of the banknote when inserted.

REFERENCES

[1] G. Mahajan, V. Phale, S. Mane, P. Ajit, and B. Patil, “Vending Machine with Cash and Cashless Payment Support,” 2020. [Online]. Available: www.irjet.net

[2] Firdaus Tarmidzi and R. A. Pratama, “Prosiding Seminar Nasional Teknologi dan Sistem Informasi 2023 Surabaya,” 2023.

[3] M. Nizam, H. Yuana, and Z. Wulansari, “ESP 32 Microcontroller As A Web-Based Door Monitoring Tool,” 2022.

[4] F. Prasetyo, “RANCANG BANGUN SISTEM PEMBAYARAN TUNAI PADA MESIN PENJUAL SWALAYAN (VENDING MACHINE),” Surabaya, 2019.

[5] P. Studi., “Arduino Uno R3-Based Monolithic Electronic Device Control System Prio Handoko,” 2017.

[6] A. N. Rohman, R. Hidayat, and F. R. Ramadhan, “Pemrograman Mesin Smart Bartender Menggunakan Software Arduino IDE Berbasis Microcontroller ATmega2560,” 2021.

[7] S. Haga, “IoT-Based Autonomous Pay-As-You-Go Payment System Development and Implementation,” International Journal of Distributed Sensor Networks, vol. 17, no. 9, pp. 1–14, 2021.

[8] M. Trianto, A. S. Suranta, and R. Sari, “Snack Vending Machine Using QR Code as Payment Method,” Ecotipe: Electronic Control, Telecommunication, Instrumentation, Power, and Electronics, vol. 2, no. 4, pp. 387–396, 2024.

[9] G. Tejaswai et al., “Development of an IoT-Based QR Code Access Control and Payment System using Arduino and ESP8266,” Journal of Science and Technology, vol. 9, no. 6, pp. 20–32, Jun. 2024.

[10] Rahman and S. Fadilla, “Biometric Based Payment System Using ESP32 Dev Kit V1 and Fingerprint Sensor,” World Journal of Advanced Research and Reviews, vol. 23, no. 1, pp. 1880–1892, 2024.

[11] F. P. Syahrani et al., “IoT-Enabled Smart Fence: Remote Security and Monitoring Using NodeMCU ESP32 and Blynk,” J-Hytel, vol. 3, no. 1, pp. 31–40, Jan. 2025.

[12] S. Kumar et al., “IoT Based Vending Machine with Cashless Payment,” International Research Journal of Engineering and Technology, vol. 6, no. 6, pp. 1856–1860, 2019.

[13] D. Lukman et al., “Bottle Refill Water Payment Using E-Card and ESP32 Microcontroller,” Journal of Electrical Engineering, Mechatronics and Computer Science, vol. 4, no. 1, pp. 63–72, 2023.