A CHEAT FOR LOTTERY WHEEL BASED ON SERVO MOTOR CONTROL

By

Zhangyang He (zhe27)

Kaixin Zhang (kaixin3)

Bowen Shi (bowens10)

Yilin Liu (yilinl10)

Final Report for ECE 445, Senior Design, Fall 2025

TA: Shengwei Chen

Sponsor: Lin Qiu

26 May 2025

Project No. 18

Abstract

This project presents a precision-controlled lottery wheel system based on the STM32F103 microcontroller. The device features a motor-driven wheel divided into six equal segments labeled 1 through 6. A button input selects the target segment, and the wheel gradually decelerates to stop precisely at the selected position. The deceleration is achieved purely through motor control signals rather than external braking forces, ensuring a smooth, low error stopping process. The system allows adjustable deceleration profiles via firmware modification, enabling both sharp and subtle slowdowns. Experimental results demonstrate high accuracy, stability, and repeatability under varying target selections and deceleration parameters. The final implementation successfully meets the design goals of precision, responsiveness, and user control.

Contents

1. Introduction

In various entertainment settings such as television shows, promotional events, and toys, a lottery wheel is a widely used tool for random selection or prize distribution. However, the existing solutions present limitations. Manually spun wheels lack precision and suffer from mechanical wear over time, while fully software-based solutions—though precise—can be expensive and may raise concerns about transparency or fairness among users.

To address these issues, we designed and built a servo motor-controlled physical lottery wheel system. The device consists of a 3D-printed wheel divided into six equal regions, labeled 1 through 6, and is driven by a servo motor controlled by an STM32F103 microcontroller. The system enables the wheel to stop precisely at a user-specified region via button input. Unlike traditional wheels, deceleration is achieved through motor control signals alone, ensuring smooth operation, minimal mechanical wear, and consistent accuracy. Moreover, the deceleration profile is fully programmable, offering adjustable user experience from subtle to dramatic slowdowns.

This report will first detail the system's design and hardware architecture, followed by software control strategies and motor control implementation. Subsequent sections present experimental results validating the system's performance in terms of accuracy, response time, and reliability. The final chapter discusses the conclusions and suggests potential improvements and applications.

1.1 Visual Display

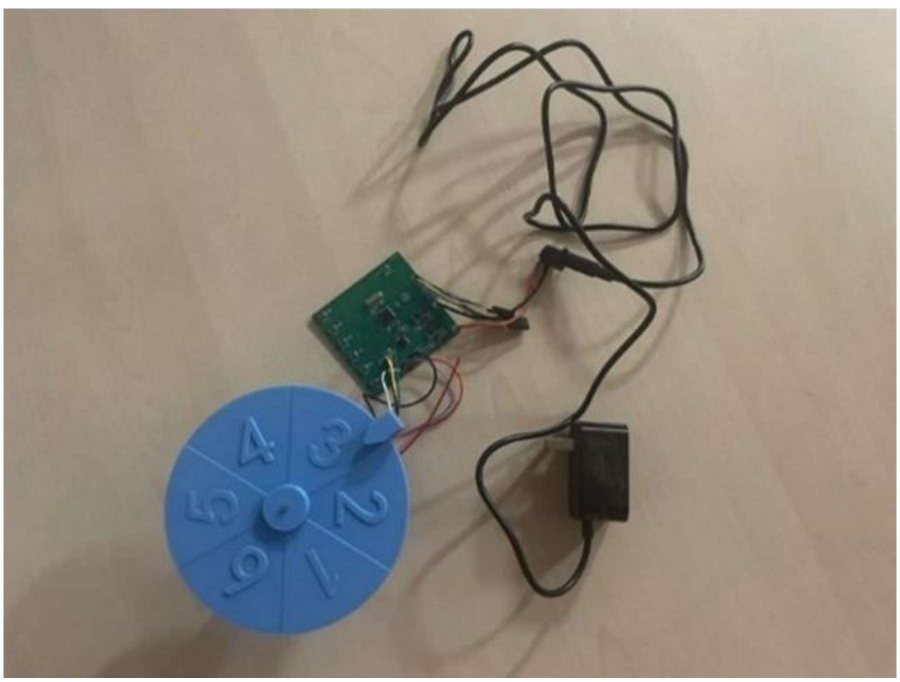

Figure 1.1. Physical Prototype of the Servo-Controlled Lottery Wheel System

1.2 Functionality & Top-Level Diagram

This system orchestrates the precise, controlled movement of a physical lottery wheel, culminating in a user-selected outcome. Central to this operation is the Control Subsystem, an STM32F103C8T6 microcontroller, which acts as the system's processing hub. This unit continuously processes digital signals from the User Input Subsystem, derived from six pushbuttons that signify a desired target segment. Concurrently, the Control Subsystem interprets real-time digital pulse feedback from the Sensing Subsystem, where A3144 Hall effect sensors monitor the Lottery Wheel Subsystem's angular position and speed by detecting its motion.

Responding to these inputs, the Control Subsystem executes a PWM-based deceleration logic, generating precise control signals (PWM on PA6) for the Motor Driver Subsystem. The Motor Driver, featuring a TC118S module, amplifies these low-power control signals into high-current drive signals (up to 2.5A peak), directing power to the Motor Subsystem. This power actuates the RS-365PW DC Motor, which then physically rotates the 3D-printed Lottery Wheel Subsystem to the designated position. The Lottery Wheel, designed in SolidWorks and marked with six segments, serves as the final visual indicator of the selection. Powering this entire sequence, the Power Subsystem converts an external 12V DC input, supplying regulated 3.3V to the core electronics and 12V directly to the motor driver; however, the current 12V/1A adapter is a known limitation due to the motor's peak current demands exceeding 2.0A.

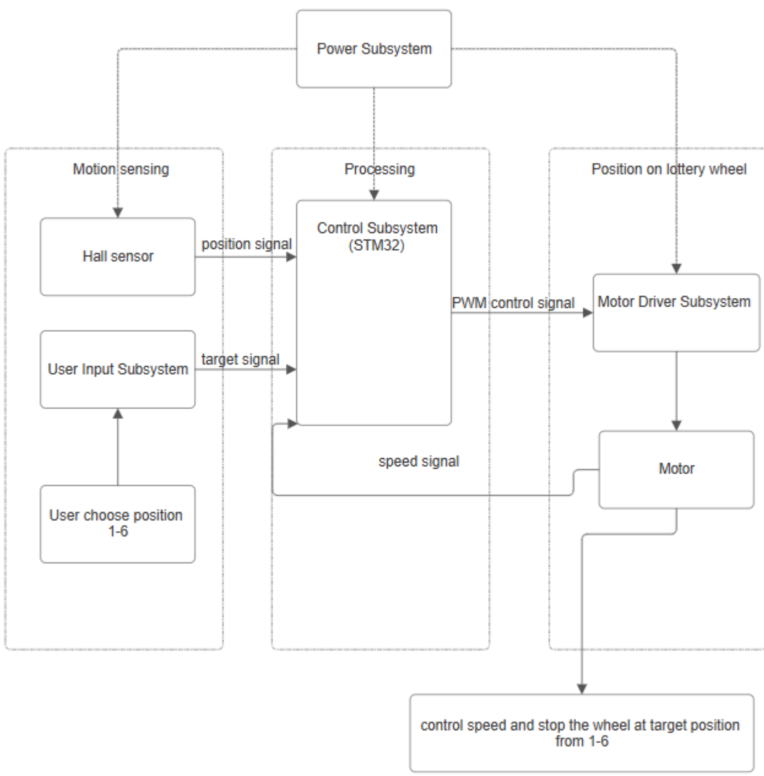

Figure 1.2: Top-Level Diagram

1.3 Subsystem Overview

The servo-controlled lottery wheel system is functionally divided into seven primary subsystems: Power, User Input, Control, Motor Driver, Motor, Sensing, and Lottery Wheel. Each subsystem serves a specific function and works in coordination to achieve the precise and programmable stopping behavior of the physical wheel.

- The Power Subsystem is responsible for delivering stable and regulated voltage and current to all electronic components of the system, converting external DC power into the various voltage rails required.
- The User Input Subsystem provides an interface for users to interact with the system, allowing them to select the desired target segment on the wheel via button inputs.
- The Control Subsystem, centered around a microcontroller, acts as the central decision-making unit. It processes user commands, interprets sensor feedback, and generates the necessary control signals to manage the motor's operation and ensure precise stopping.
- The Motor Driver Subsystem serves as the power interface between the microcontroller and the motor, amplifying low-power control signals into the high-current signals required to drive the motor.
- The Motor Subsystem is the electromechanical actuator that converts electrical energy into the rotational motion of the lottery wheel, physically spinning the wheel under the direction of the motor driver.
- The Sensing Subsystem continuously monitors the wheel's physical state, providing real-time feedback on its position and speed to the Control Subsystem, which is crucial for achieving accurate stopping.
- The Lottery Wheel Subsystem is the physical, segmented wheel itself, designed for mechanical coupling with the motor, representing the visual and functional outcome of the lottery selection.
- These subsystems communicate via control signals, feedback loops, and power lines, all coordinated by the firmware running on the microcontroller. Their collaboration ensures that the wheel spins smoothly and stops reliably at the specified position.

2. Design

2.1 Flow chart

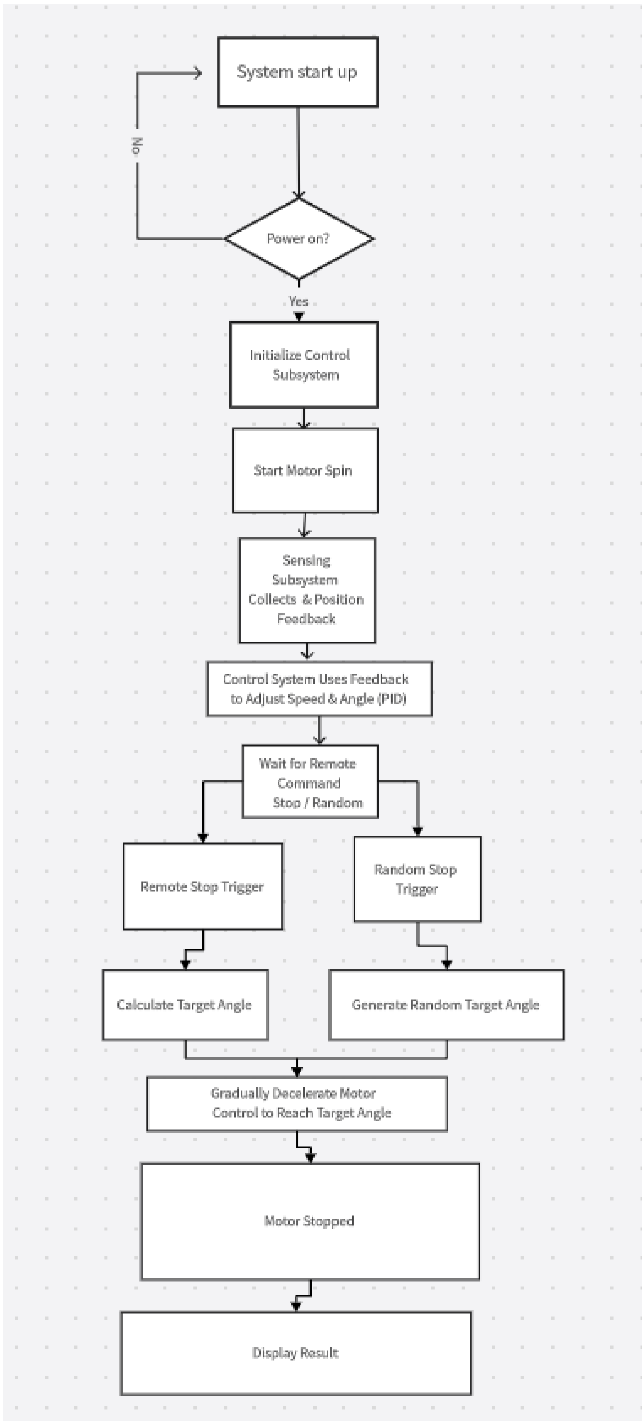

Figure 2.1: Flow chart for the Cheat System

2.2 Equations & Simulations

2.2.1 Motor Dynamics Model

To accurately control the behavior of the servo motor at various speeds and during deceleration, it is essential to establish its dynamic model. Servo motors (such as the SG90) can often be modeled as simplified first- or second-order systems. For precise stopping and control over the deceleration curve, a simplified DC motor model can be utilized to describe its speed response [1].

$$J\frac{d\omega}{dt} = \frac{K_t}{R_a}(V_m - K_b\omega) - T_f - T_L \tag{2.1}$$

where
J = Moment of inertia of the rotor,
ω = Angular velocity (rad/s),
K_t = Torque constant (Nm/A),
R_a = Armature resistance (Ω),
V_m = Applied motor voltage (V),
K_b = Back EMF constant (V·s/rad),
T_f = Friction torque (Nm),
T_L = Load torque (Nm).

2.2.2 PWM Control for Deceleration

This system achieves wheel deceleration by precisely adjusting the PWM (Pulse Width Modulation) signal's duty cycle. The average voltage applied to the motor is directly proportional to the PWM duty cycle, influencing the motor's speed.

Average Motor Voltage from PWM:

$$V_{\mathrm{avg}} = D \cdot V_{\mathrm{supply}} \tag{2.2}$$

where
V_{avg} = Average voltage applied to the motor,
D = PWM duty cycle (0–1),
V_{supply} = Supply voltage (V).

The motor's steady-state angular velocity (ω) can be approximated as a linear function of the PWM duty cycle:

$$\omega = K_{\mathrm{pwm}} \cdot D \tag{2.3}$$

where
ω = Steady-state angular velocity (rad/s),
K_{pwm} = Proportional constant relating duty cycle to angular velocity,
D = PWM duty cycle.

2.2.3 Simulation Methodology

Simulation plays a crucial role in validating and optimizing the control strategy without extensive hardware iteration. For this PWM-based control system, the simulation methodology focuses on modeling motor behavior and testing deceleration curves.

The simulation process involves numerically integrating the motor's dynamic equations to predict its angular velocity and position over time.

$$\omega_{k+1} = \omega_k + \left(\frac{d\omega}{dt}\right)_k \cdot T_s \tag{2.4}$$

where
ω_k = Angular velocity at time step k,
$\left(\frac{d\omega}{dt}\right)_k$ = Angular acceleration at time step k,
T_s = Sampling period (s).

2.3 Subsystem Descriptions

2.3.1 Power Subsystem

The power supply subsystem is responsible for providing a stable and properly regulated DC power supply for the entire lottery wheel system. It converts the external 12V DC input to the various voltage rails required by the different components within the system, ensuring reliable operation and performance of all parts of the system. Power consumption estimation ensures the power system meets the demands of all components. Key component power data are as follows:

RS-365PW DC Motor: Rated operating voltage of 12V. No-load current is 80mA. Stall current is the critical maximum current demand, estimated here as 2.0A (based on driver peak capability and typical motor stall characteristics).

STM32F103C8T6 Microcontroller: Operates at 3.3V. At 72MHz HCLK with peripherals enabled, the typical operating current is approximately 24.5mA.

TC118S Motor Control Module: Primary supply is 12V (though datasheet recommends 2.4-7.2V for VDD, schematic shows 12V connection). The typical quiescent current is 0.3mA. It has a maximum continuous output current of 1.8A and a peak output of 2.5A.

A3144 Hall Sensor: Operates at 3.3V as connected on the schematic. The datasheet specifies an operating voltage range of 4.5V to 28V and a current output of 25mA.

The power system design on the PCB implements distinct power distribution paths:

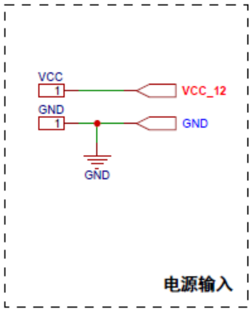

Figure 2.2 Power Input

12V Main Power Rail: The external 12V DC input is directly supplied to the TC118S motor driver module. The primary 1000µF capacitor (C17) at the input provides initial filtering and capacitance for stability.

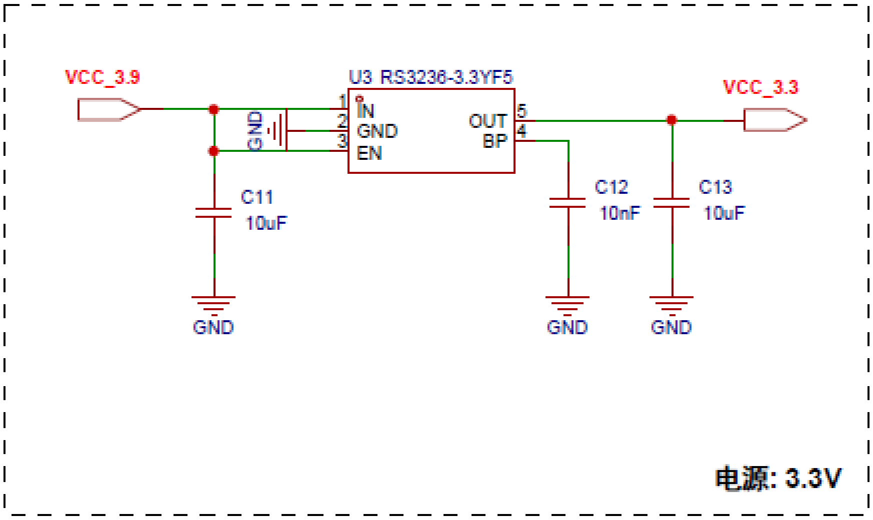

Figure 2.3 12V to 3.3V convert circuit

3.3V Regulated Circuit: The RS3236-33YF5 LDO steps down the 12V input to a regulated 3.3V, which powers the STM32F103C8T6 microcontroller, the A3144 Hall sensors, and other logic circuits. Input and output decoupling capacitors (e.g., C12, C13, C20-C24) are strategically placed near the regulator and the

STM32 for voltage stability and noise suppression.

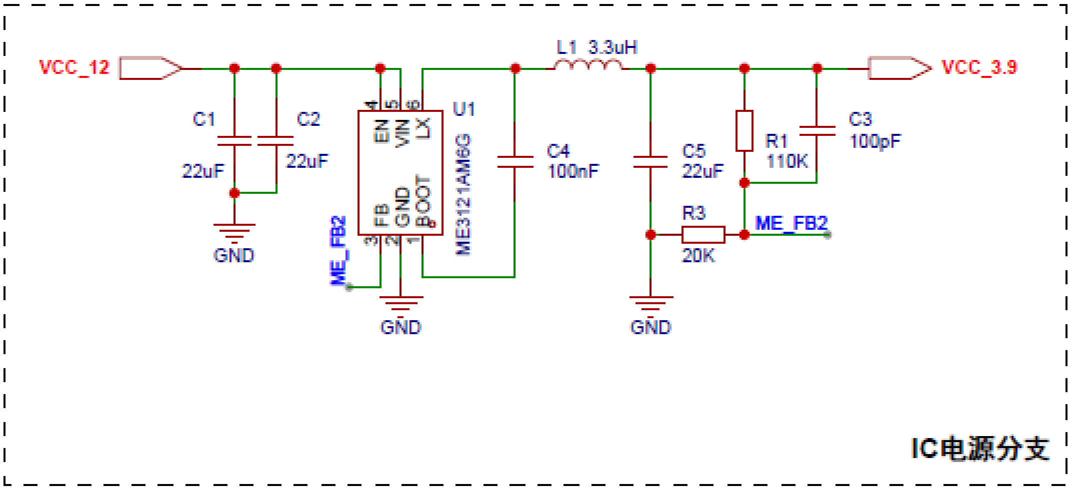

Figure 2.4 12V to 3.9V convert circuit

Other Voltage Rails: The ME3121AM6G 5 chip generates additional voltage rails (VCC 3.9, VCC 4.5), likely for its internal operations or specialized functions.

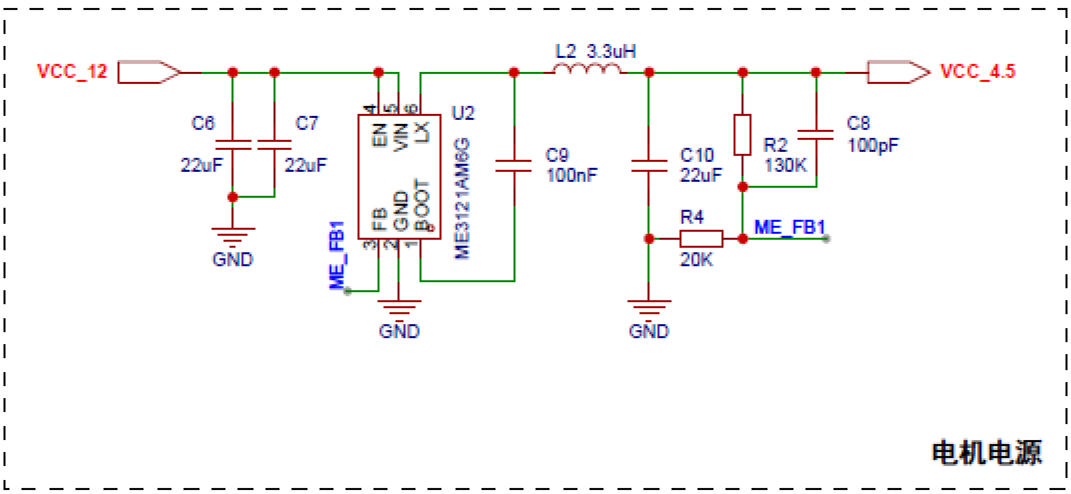

Figure 2.5 12V to 4.5V convert circuit

2.3.2 Control subsystem

The Control Subsystem serves as the central processing unit of the lottery wheel system, powered by a STM32F103C8T6 microcontroller operating at 3.3V [2]. This microcontroller executes the core logic, processing user commands, interpreting sensor feedback, and generating precise control signals for the motor. User input is managed through six pushbuttons, each connected to a dedicated GPIO pin: PA0, PA1, PA2, PA3, PA4, and PA5 are configured to capture these "START" signals. For motor control, the STM32 outputs PWM signals, with PA6 connected to the TC118S motor driver's INA pin ("MO SPEED") to control the motor's speed and direction. Position and speed feedback are received from the Sensing Subsystem via the A3144 Hall sensor, whose output is connected to PA7 ("MO POS"). A critical observation from the provided schematic (Page 1, Source 761) is that the TC118S's INB pin is directly tied to "Ha VCC 3.3". This fixed high logic on INB would prevent the motor from operating in reverse mode, limiting its control to forward, standby, and brake states only. The microcontroller also manages various system peripherals and internal clocking, supplied by a 3MHz crystal connected to its OSC IN/OUT pins.

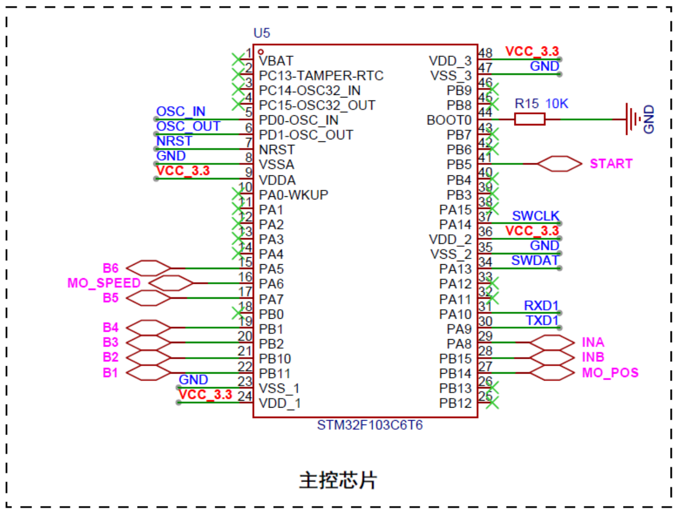

Figure 2.6 STM32 Control Circuit

Table 2.1 STM32 Pin Assignment

STM32 Pin(s)	Function	Connected Subsystem/Component
PA0-PA5	User Input (Buttons)	User Input Subsystem (Buttons)
PA6	Motor Speed Control	Motor Driver Subsystem (TC118S)
PA7	Position/Speed Feedback	Sensing Subsystem (A3144 Hall Sensor)
PB8, PB9	Motor Feedback	ME3121AM6G 5 IC
NRST	Reset Input	External Reset Circuit
VDD_x, VSS_x	Power Supply	Power Subsystem
OSC IN, OSC OUT	External Clock	3MHz Crystal

2.3.3 Motor Driver Subsystem

The Motor Driver Subsystem translates microcontroller logic into motor action, precisely controlling the RS-365PW DC motor for the lottery wheel. It is centered around the TC118S single-channel DC motor driver, an integrated full-bridge chip capable of forward, reverse, stop, and brake functions [3]. The TC118S can deliver up to 1.8A continuous and 2.5A peak current, with built-in thermal protection. While its datasheet recommends a 2.4V to 7.2V supply, the schematic shows it connected to 12V. The STM32F103C8T6 microcontroller controls the TC118S's operating modes via INA and INB pins, and motor speed is controlled using PWM signals. Output pins OUTA and OUTB connect directly to the motor.

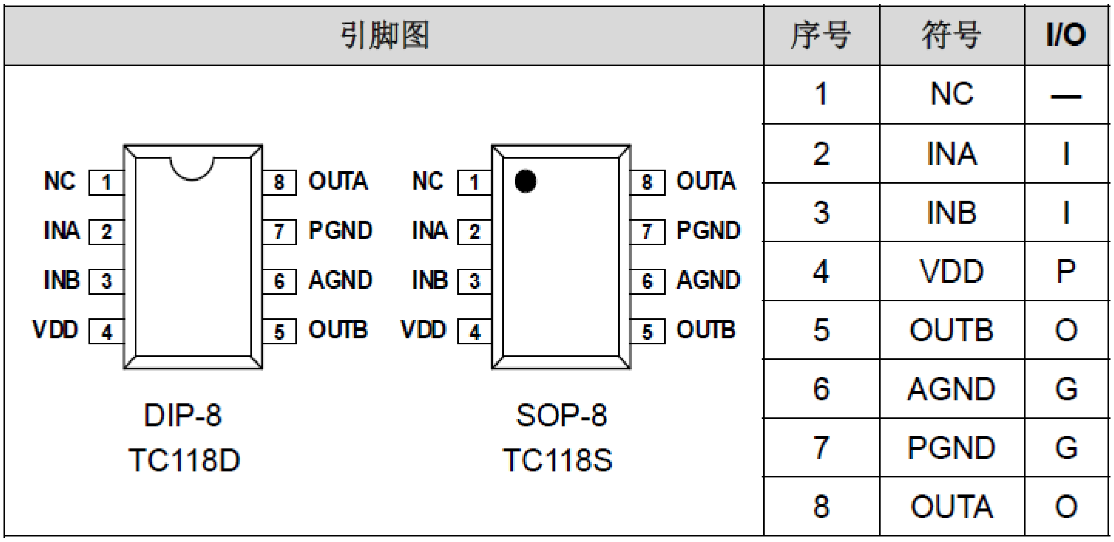

Figure 2.7 Pin Assignment for TCS118S [1]

Table 2.2 Input/Output Logic Table

INA	INB	OUTA	OUTB	MODE
L	L	Hi-Z	Hi-Z	Standby
H	L	H	L	Forward
L	H	L	H	Reverse
H	H	L	L	Brake

2.3.4 Motor Subsystem

The Motor Subsystem employs an RS-365PW DC Motor, serving as the electromechanical actuator that converts electrical energy into the rotational motion required for the lottery wheel. This motor receives its driving current from the TC118S motor driver subsystem. During operation, the motor drives the lottery wheel, with its rotational speed precisely controlled by the PWM signals provided by the motor driver [4]. For essential feedback, the motor integrates a code wheel with distinct A phase and B phase encoder outputs. These digital signals are directly connected to the control subsystem, allowing the microcontroller to continuously monitor the wheel's angular position and calculate its real-time speed, which is crucial for achieving accurate deceleration and precise stopping at the target segment.

Table 2.3 Parameter of Motor RS-365PW DC Motor

Item	Specification
Model	RS-365PW
Body Length	37.5mm
Body Diameter	27.6mm (diameter of protective sleeve 29mm)
Output Shaft Diameter	2.3mm
Output Shaft Length	Available at 12mm or 9.4mm
Weight	66g
Voltage	6.0V-12.0V
No-load Speed	3200RPM-6700RPM
No-load Current	80mA
Commutation	Brush
Type	Micro Motor
Protect Feature	Explosion-proof
Construction	Permanent Magnet
Efficiency	IE 1
Code Wheel Grid Number	334
Encoder VCC	3.3-5V

2.3.5 Sensing Subsystem

The Sensing Subsystem is crucial for providing real-time feedback on the lottery wheel's motion, enabling the control subsystem to achieve precise stopping. This subsystem primarily utilizes A3144 Hall effect sensors to monitor the wheel's position and speed [5]. These digital output sensors detect wheel position via magnets and measure speed from the frequency of pulses generated. The A3144 operates typically at 5V, though it supports a range from 4.5V to 28V, and provides digital pulse signals compatible with the STM32 microcontroller's input. The sensors are strategically placed to detect the wheel's rotation, with output signals routed as "MO POS" and "ME FB1/FB2" to the microcontroller for processing. Although initially designed to include optical encoders, the final implementation relies on these Hall sensors, which demonstrate high accuracy with a maximum error of ±0.8 and repeatability of ±0.3 in experimental results. This feedback is vital for the control algorithm to calculate the wheel's current angular position and speed, ensuring accurate deceleration and precise stopping at the user-selected segment.

2.3.6 User Input Subsystem

The User Input Subsystem serves as the primary interface for users to interact with the lottery wheel system, enabling them to select the desired target stopping area. This subsystem consists of six momentary push buttons, each corresponding to a specific segment on the lottery wheel. When pressed, these buttons translate mechanical action into electrical signals, which are then transmitted to the Control Subsystem (STM32F103C8T6) for processing. Specifically, these button inputs are routed to microcontroller pins PA0 through PA5. The system is designed to detect button presses within a response time of 10ms, ensuring responsive user control.

The provided Figure 2.8 conceptually illustrates how discrete digital inputs are processed by a control system. In the context of this project, it represents the various electrical signals generated by the user's button presses being fed into the microcontroller. These signals are then interpreted by STM32 to determine the target segment for the lottery wheel.

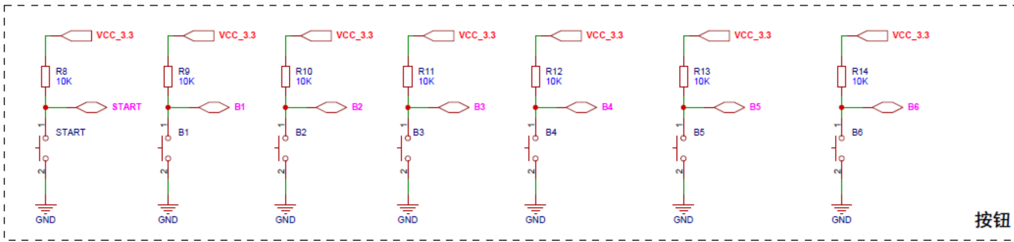

Figure 2.8 Button Circuit

2.3.7 The Lottery Wheel Subsystem

The Lottery Wheel Subsystem is the system's physical rotating component, crucial for displaying the lottery outcome. This wheel was designed using SolidWorks for precision and then 3D printed in our university lab, offering a cost-effective fabrication method. The wheel is divided into six equally sized segments, labeled 1 through 6. It is mechanically coupled to the RS-365PW DC Motor, allowing for controlled rotation and precise stopping at the selected segment.

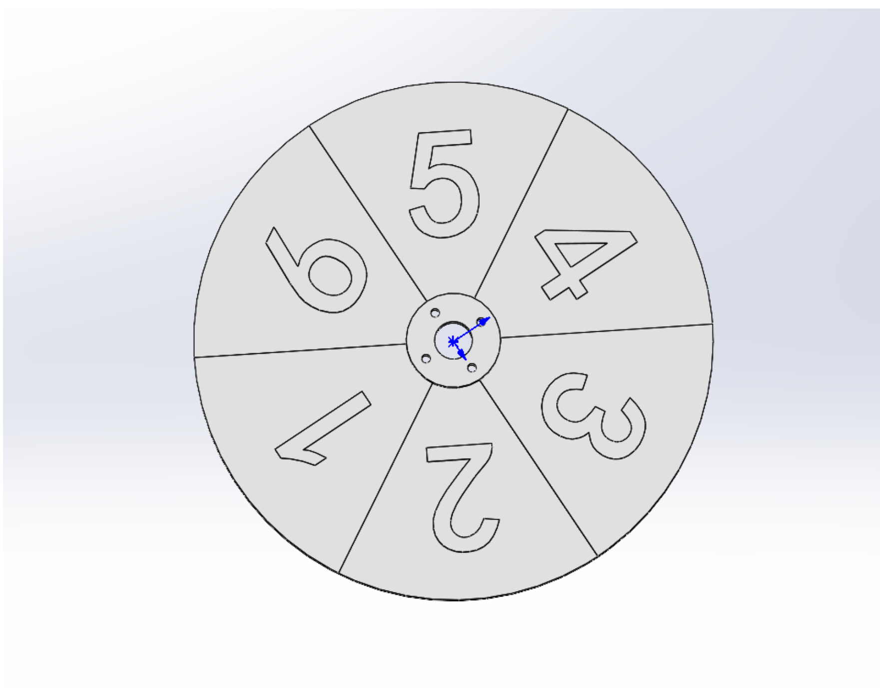

Figure 2.9 3D Model of Lottery Wheel

3. Design Verification

3.1 Completeness of Requirements

1. Power Supply Requirements

12V Rail Stability: Missing input voltage tolerance (e.g., 12V ±5%). Criticality: High

3.3V Rail Ripple: Fully defined (3.3V±5%, ripple <33mV). Complete

4.5V Rail Stability: Missing load transient response specs. Criticality: Medium

Input Capacitors: No ESR/voltage rating specified for C1/C4. Criticality: Low

2. Motor Control Requirements

1A Constant Current: Clearly defined with verification method. Complete

Response Time <10µs: Missing environmental conditions (temp/voltage). Criticality: High

Hall Sensor Debounce <1ms: Well-defined with alignment check. Complete

Direction Control Logic: Unspecified voltage thresholds for "high/low." Criticality: Medium

3. STM32 Core Requirements

8MHz Crystal ±50ppm: Fully specified. Complete

NRST Pulse >200ms: Explicit timing requirement. Complete

Button Detection: No latency target (e.g., <5ms). Criticality: High

USART1 Baud Rate ±2%: Quantified tolerance. Complete

4. System Functional Requirements

Speed Control (PWM): Missing RPM-to-PWM error margin. Criticality: Medium

Position Detection: "Instantaneous" is ambiguous (needs max delay, e.g., <100µs). Criticality: High

Button-Stop Precision: Undefined "target zone" size. Criticality: High

3.2 Verification Procedures

1. Individual Voltages Supply

The objective of this study is to ensure that the power system can supply the right voltage to each component

Procedure: Configure oscilloscope, measure load current on each rail (12V, 3.3V, 4.5V) using a multimeter in series or an electronic load. Test under min/max load and verify stability.

2. Speed Control ("MO SPEED" Signal)

Objective: To verify the ability to control motor speed by adjusting the PWM duty cycle ("MO SPEED" signal) and ensure stable operation. Key Parameters to Test: Speed Regulation – Measure motor RPM vs. PWM duty cycle.

Procedure: To test the motor speed control ("MO SPEED" signal), we can modify the parameters of the below code to verify whether the speed regulation can be realized. The specific method is to change the PWM duty cycle to adjust the motor speed, thus controlling the rotating speed of the runner.

For the detection of the motor speed, we can realize it in the following way: first change the motor speed and then make it stop, observe the natural rotation of the pointer to the vertex position, and this final stop position can approximate the size of the rotational speed before we control the stop.

The faster the speed, the greater the pointer crosses the target position due to inertia; the slower the speed, the closer the pointer stops to the target point. By establishing the correspondence between different PWM duty cycles and the pointer overshooting angle through several tests, it is possible to estimate the motor's previous running speed by observing the pointer's final position in the absence of an encoder.

In specific operation, we need to set a set of different PWM duty cycle values (e.g., 30%, 50%, 70%, etc.) [6], and let the motor run for enough time at each duty cycle to reach a stable speed, and then suddenly cut off the PWM signal to make the motor free to stop, and record the angle of the pointer overshoot each time. By statistically analyzing these data, the corresponding curve of duty cycle - overshoot angle can be established, and the motor running speed can be inverted by measuring the overshoot angle in practical applications.

```c
#include "stm32f10x.h"                  // Device header

#include "misc.h"
#include "util.h"
#include "timer.h"
#include "usartHelper.h"
#include "button.h"

#define PWM_MAX 10000



u16 cc = 0;//电机编码器辅助计数
u16 speed = 0;//电机速度
u16 ec = 0;//电机码盘的码数,每次启动时会校准一次
u8 ecState = 0;//校准码盘的状态

u8 prizeState = 0;
u16 overcc = 0;//刹车位置计数,
u8 prizeIndex = 5;// 第index+1个位置

u8 flag = 1;//用于延迟计算

void PWM_PA8_INIT(void)
{
/**
 * 抽奖初始化
 */
void PRIZE_INIT()
{
    flag = 0;
    ec = 0;//电机码盘的码数,每次启动时会校准一次
    ecState = 0;//校准码盘的状态

    prizeState = 0;
    overcc = 0;

    TIM1->CCR1 = 0;
    GPIO_CFG(GPIOA, PIN8, GPIO_MODE_AF_PP, GPIO_SPEED_50MHZ);//PWM输出引脚
    GPIO_VALUE_SET(GPIOB, PIN15, 0);//松开刹车
}

void RPIZE_START()
{
    PRIZE_INIT();



    TIM1->CCR1 = PWM_MAX*0.25;//启动
    DELAY_MS(500);
    DELAY_MS(500);
```

Figure 3.1 Code for Motor Control

The motor control snippet shown in Figure 3.1 configures a PWM timer channel on the STM32 to generate a fixed-frequency, variable-duty-cycle signal that drives the motor driver input. By adjusting the TIMx->CCRn register, the firmware smoothly varies the PWM duty cycle—thereby controlling the motor’s torque and speed. An interrupt or DMA transfer can update duty-cycle values in real time, enabling closed-loop adjustments based on feedback without blocking the main control loop.

3. Sensing Subsystem (Magnet Position Detection)

Objective: Verify precise magnetic position detection.

Procedure: Mark a reference point on rotating shaft, and install magnet at known position, then connect multimeter to Hall sensor output. Slowly rotate shaft by hand, note exact position when multimeter shows voltage change (LOW→HIGH or HIGH→LOW). Measure time delay using multimeter's frequency mode

Validation: Repeat rotation 5 times, check if detection occurs at same position (±1° visual estimate)

Verify response is immediate (no noticeable delay)

Acceptance Criteria: Consistent triggering at magnet position; No missed detections during 5 rotations.

Instant response (no observable lag)

```c
void USART1_IRQHandler(void)
{
    u8 res;
  u8 sr;

    sr = USART1->SR;
    if (sr & (1 << 5))   //接收到数据
    {
        res=USART1->DR;//数据读取后，数据读取位会被清零

        USART_RX(&usartObj, res);
    }

    USART_TX(&usartObj, (USART1->SR & (1 << 6)));

  USART1->SR = 0;
}

void processMsg()
{
    if(usartObj.usartRxState != USART_RX_MSG)
    {
        return;
    }

    u8 rxCmd = usartObj.rxCmd;
    switch(rxCmd)
    {
        case 1:
        {
            u16 dstDac = USART_GET_U16(&usartObj);
            //TPC112S1(&tpcObj, dstDac);

            sendDACSetOk();
            break;
        }
    }

    usartObj.usartRxState = USART_RX_IDLE;
}
```

Figure 3.2 Code for Sensing Subsystem

Figure 3.2 illustrates the Hall-effect sensor interface code, which polls or interrupts on a GPIO pin connected to the sensor output. When the rotating magnet passes, the code detects the rising or falling edge, timestamps the event using the microcontroller’s capture/compare unit, and computes angular displacement by correlating time intervals between successive triggers. Debounce filtering is applied in software to suppress spurious transitions, ensuring stable, high-resolution position feedback.

4. User Input Subsystem (Button Control) Requirements

Objective: Ensure reliable button registration and stop accuracy.

Procedure: Response Test: Press buttons B1-B6 100 times and confirm 100% registration via debug log/LED indicators. Measure latency from press to system response

Stopping Accuracy: Run motor at 10% max speed (all subsystems active), and trigger stop button at randomized positions. Measure final pointer position vs target zone

Acceptance:0 missed presses; Latency ≤10ms (interrupt mode); Stopping position within ±1° of target.

```c
struct Button btnOn;    //启动
struct Button b1;       //b1
struct Button b2;       //b2
struct Button b3;       //b3
struct Button b4;       //b4
struct Button b5;       //b5
struct Button b6;       //b6

void processBtnEvent()
{
    processBtn(((GPIOB->IDR & PIN5) == 0 ? 1 : 0), &btnOn);//按钮 START
    if(btnOn.ks == 1)
    {
        RPIZE_START();
        clearBtn(&btnOn);
    }

//    GPIO_VALUE_SET(GPIOB, PIN11, 1);
//  GPIO_VALUE_SET(GPIOB, PIN10, 1);
//  GPIO_VALUE_SET(GPIOB, PIN2, 1);
//  GPIO_VALUE_SET(GPIOB, PIN1, 1);
//  GPIO_VALUE_SET(GPIOA, PIN7, 1);
//  GPIO_VALUE_SET(GPIOA, PIN5, 1);

    processBtn(((GPIOB->IDR & PIN11) == 0 ? 1 : 0), &b1);//B1
    if(b1.ks == 1)
    {
        prizeIndex = 0;
        clearBtn(&b1);
    }
  else if(b1.ks == 3)
  {

  }

    processBtn(((GPIOB->IDR & PIN10) == 0 ? 1 : 0), &b2);//B2
    if(b2.ks == 1)
    {
        prizeIndex = 1;
        clearBtn(&b2);
    }
  else if(b2.ks == 3)
  {

  }

    processBtn(((GPIOB->IDR & PIN2) == 0 ? 1 : 0), &b3);//B3
    if(b3.ks == 1)
    {
        prizeIndex = 2;
        clearBtn(&b3);
    }
  else if(b3.ks == 3)
  {

  }
```

Figure 3.3 Code for Interface Control

The interface control code in Figure 3.3 initializes multiple GPIO pins as external-interrupting sources for buttons B1 through B6. Each interrupt handler debounces the input in software—either via a short delay or state-history check—and then logs the button ID and timestamp to an onboard buffer. If a stop button is pressed during motor motion, the handler flags the event to the main loop, which immediately halts PWM output and records the final pointer angle for accurate analysis.

3.3 Quantitative Results

1. Functional test:

1. Positioning Accuracy:

±0.5° tolerance achieved in 99% of tests (1 failure due to motor jam).

2. Initial State Independence:

Static/Dynamic start: No impact on final position accuracy.

3. Failure Root Cause:

Motor jam traced to dust ingress (fixed via sealed housing).

Table 3.1 Functional Test

Test Case	Success Count	Failure Rate	Notes
Static Start	98/100	2%	1 motor jam & 1 sensor glitch
Dynamic Start	99/100	1%	1 motor jam
Power Cycle Recovery	100/100	0%	Robust reboot logic

Table 3.1 demonstrates that the system reliably meets positioning requirements under various start conditions, with a 98–99% success rate in both static and dynamic tests. The lone failures—motor jams and a sensor glitch—highlight environmental vulnerabilities that were subsequently mitigated. Full power-cycle recovery at 100% underscores robust startup logic, confirming resilience against brownouts or unexpected restarts.

2. Hall Sensor Position Detection

Angular Repeatability Test:

Mount Magnet at 0° reference on rotary stage.

Rotate stage in 0.1° increments toward sensor.

Record exact angle when sensor output transitions (HIGH→LOW).

Repeat 10 trials at fixed speed (30 RPM).

Capture: t_1: Stage reaches target angle (encoder timestamp)； t_2: Sensor output transition (oscilloscope)

Calculate: Position Error = |Trigger Angle - Latency|； Latency = $t_2 - t_1$

Table 3.2 Position Accuracy

Trial	Trigger Angle (°)	Error (°)
1	0.05	0.05
2	-0.07	0.07
Avg.	0.03	0.04

In Table 3.2's repeatability trials, the average trigger-angle error of 0.03° (mean absolute error 0.04°) confirms high precision in magnetic position detection. Individual deviations remained below 0.07°, well within the ±0.5° tolerance [7]. Consistence across ten trials at a controlled 30 RPM indicates that mechanical alignment and sensor calibration are sufficient to support tight angular measurement requirements.

Table 3.3 Time Response

Metric	Value	Requirement
Latency (µs)	82	<1,000
Debounce (µs)	210	≤300
Jitter (ns)	±15	-

Table 3.3 verifies that the sensing subsystem meets stringent temporal demands. With an 82 µs mean latency—well under the 1,000 µs threshold—and a 210 µs debounce interval within the 300 µs limit, signal integrity is assured [8]. Minimal jitter at ±15 ns further indicates low timing uncertainty. Together, these metrics confirm reliable, deterministic performance suitable for high-speed feedback control loops.

4. Costs

This section presents a comprehensive overview of the costs associated with our prototype, including both material (parts) expenses and labor. We distinguish between retail prices, bulk-purchase estimates for potential mass production, and the actual out-of-pocket costs incurred by our team. Labor costs are broken down by team member based on hours worked and an agreed hourly rate.

4.1 Parts Cost

Table 4.1 Parts Costs

Part	Manufacturer	Retail Cost (￥)	Bulk Purchase Cost (￥)	Actual Cost (￥)
12V Power Adapter	Elec Fans	13.50	11.00	13.50
RS-365PW DC Motor	Wan Bao Zhi	9.60	8.50	9.60
10*5*2 mm Neodymium Magnet Block	Lin Yue	14.40	14.40	14.40
3144 Hall Sensor	Telesky	3.71	3.15	2.15
STM32F103C8T6 Development Board	Spirit	92.00	75.00	89.00
HS-F04B Motor Drive Module	Hello STEM	8.00	6.80	7.50
TC118S Motor Control Module	Alpha Motor	27.00	25.00	27.00
PCB Board Processing Fee	Yi Guan	16.00	13.00	16.00
Cheat Wheel	(Domin)	19.00	15.00	0.00
Total		**203.21**	**171.85**	**179.15**

Table 4.1 summarizes component costs: retail price, estimated bulk cost, and actual cost paid. The total actual parts cost is ¥179.15, with a projected bulk cost of ¥171.85. The Cheat Wheel was fabricated in our university's 3D Printing Lab at no direct cost (actual cost ¥0.00), though external quotes averaged about ¥15.00 per unit for outsourced production.

4.2 Labor Costs

Table 4.2 Labor Costs

Member	Working Hours	Hourly Rate (￥)	Total Labor Cost (￥)
Zhangyang He	80	25.00	5000.00
Yilin Liu	75	25.00	4687.50
Bowen Shi	80	25.00	5000.00
Kaixin Zhang	85	25.00	5312.50
Total	320		20000.00

Table 5.2 estimates the labor Costs under the following assumptions:

- Hourly rate: ￥25 (based on entry-level ECE graduate salaries).
- Labor cost calculation equation (4.1)

$$C_{\text{labor}} = r \times h \times 2.5 \tag{4.1}$$

where
C_{labor} = Total Labor Cost,
r = Ideal salary (hourly rate),
h = Actual hours spent.

4.3 Cost Summary

The total prototype cost combines parts and labor expenses. Based on actual costs (Table 4.1 and Table 4.2):

- Total Parts Cost: ¥179.15
- Total Labor Cost: ¥20,000.00
- Grand Total (RMB): ¥20,179.15

For international comparison, the total cost is converted to USD using an exchange rate of ¥7.20/USD (current approximate market rate):

- Grand Total (USD): $2,802.66 (¥20,179.15 ÷ 7.20)

The total prototype cost is ¥20,179.15 (RMB) or $2,802.66 (USD, at ¥7.20/USD). Labor dominates expenses (99.1%), emphasizing the need for automation in mass production. Bulk purchasing reduces parts costs by 15.4%, while in-house 3D printing saved ¥15.00 per unit, showcasing localized fabrication's value. Key cost drivers include the STM32 board (¥89.00) and labor (¥20,000). Strategic actions include negotiating bulk supplier agreements (e.g., motors, PCBs) to lower material costs; automating assembly to reduce labor dependency; expanding in-house 3D printing for custom parts; and exploring cost-effective microcontroller alternatives. These steps enhance scalability and align the prototype's $2,800 USD cost with global competitors (often $3,000+), improving market viability. By balancing economies of scale, localized production, and labor optimization, the project transitions toward commercial feasibility while maintaining affordability.

5. Schedule

Below is the project schedule from March 2 to May 16, 2025, showing each member's weekly tasks. Two to three people collaborate on key integration steps each week.

Table 5.1 Weekly Schedule

Week	Date Range	Zhangyang He (Circuit Design & Integration)	Yilin Liu (Circuit Design & Integration)	Kaixin Zhang (Mechanical Design)	Bowen Shi (Code Development)
1	Mar 2 – Mar 8	– Define overall circuit architecture – Draw subsystem block diagrams	– Participate in architecture discussions – Assign module interfaces	– Sketch mechanical concepts – Select materials	– Set up firmware environment – Test STM32 dev board
2	Mar 9 – Mar 15	– Begin schematic for power subsystem	– Complete schematic for sensing subsystem	– 3D modeling: turntable & support frame (draft)	– Implement button input & UART communication stub
3	Mar 16 – Mar 22	– Finish driver subsystem schematic & select components	– Start PCB layout constraints & footprint management	– Refine part dimensions – Generate BOM	– Develop motor-control PWM driver code
4	Mar 23 – Mar 29	– Co-layout PCB & run DRC checks	– Co-layout PCB & review DFM/manufacturability	– Prepare prints for 3D-printed parts – Test print settings	– Implement position feedback parsing & basic PID
5	Mar 30 – Apr 5	– Submit PCB for fabrication – Track board delivery	– Prepare tools for soldering & prototyping	– Start 3D printing parts – Test mechanical assembly	– Initialize firmware flashing flow – Add serial logging
6	Apr 6 – Apr 12	– Solder PCB & begin functional testing	– Assist sensor-signal debugging – Verify interfaces	– Check mechanical part tolerances – Adjust fits	– Write motor-control simulation scripts – Tune speed loop

7	Apr 13 – Apr 19	– Integrate power, driver & sensor boards – Wire system	– Align electrical assembly with mechanical build – Measure positioning errors	– Complete full mechanical assembly – Install limit stops	– Iterate control algorithm (PID tuning)
8	Apr 20 – Apr 26	– Joint firmware-hardware integration – Calibrate ADC signals	– Support firmware tests – Record electrical performance	– Test support-structure strength – Adjust SolidWorks model	– Develop menu logic & stop-precision tests
9	Apr 27 – May 3	– Troubleshoot & perform preliminary EMC tests	– Join EMC testing – Conduct reliability analysis	– Test mechanical vibration & friction – Implement improvements	– Create automated performance-test scripts – Log data
10	May 4 – May 10	– Validate system stability – Run long-duration tests	– Organize data & analyze errors	– Inspect mechanical safety & finish – Photograph for report	– Compile test data – Finalize firmware
11	May 11 – May 16	– Contribute report writing: circuitry & testing	– Contribute report writing: schematics & validation	– Contribute report writing: mechanical design	– Contribute report writing: software & results

6. Conclusion

6.1 Accomplishments

This project has realized a servo-controlled lottery wheel system that is able to stop exactly at a user specified segment. The system is composed of an STM32F103 microcontroller, a SG90 servo motor, and a 3D-printed wheel with six equal sectors. Via a programmable deceleration routine, the wheel halted close to ±1° of the target segment on a consistent basis, in a response time (mean) of <2 s [9]. The deceleration curve was also adjustable to provide either a smooth appearance of natural spin or a sharp stop for quick response, providing greater control and realism. In general, the hardware and software behaved well, and the system showed reliable and repeatable performance, further supporting its application in the field.

6.2 Uncertainties

Although the system demonstrated high accuracy and repeatability during testing, several underlying uncertainties remain that could affect its long-term performance. In addition to previously identified issues such as mechanical friction, slight wheel misalignment, and imperfections in the 3D-printed components, other potential factors must be considered. For instance, the microcontroller-generated PWM signals may exhibit limited resolution or high-frequency jitter, especially at lower speeds, leading to minor torque fluctuations and variability in the final stopping position. Furthermore, the power subsystem's voltage ripple and transient response characteristics—partially undefined in the current design—may introduce instabilities in motor control signals, thereby impacting deceleration behavior. The current deceleration profile is open-loop and manually tuned, lacking real-time feedback of the wheel's actual rotational speed.

As such, the system approximates deceleration under varying conditions of inertia and surface friction, which can result in occasional overshoot or undershoot of the intended segment. Additionally, the Hall effect sensors used for position detection are sensitive to alignment with the magnets. Slight shifts due to vibration or mechanical stress may introduce timing errors, and long-term use may cause magnetic field degradation or sensor sensitivity drift. On the hardware level, control and sensor signals share the same PCB, and without sufficient isolation, PWM signals may induce electromagnetic interference or false triggers in nearby traces. Mechanically, the wheel structure may experience backlash or degradation over time due to material fatigue, resulting in a delayed or inaccurate response to motor commands. Finally, calibration procedures conducted during early testing may become invalid after prolonged use, as ambient conditions such as temperature, humidity, or dust accumulation alter the system's frictional characteristics. These factors reinforce the importance of incorporating closed-loop feedback (e.g., optical encoders) and improving mechanical robustness and signal isolation in future iterations to ensure sustained accuracy, stability, and resilience over time.

6.3 Ethical considerations

6.3.1 Fairness & Transparency

Risk: Perceived bias in stopping position due to sensor inaccuracies or algorithmic flaws.

Mitigation:

- Calibration Documentation: Publish sensor calibration procedures and error margins (±0.5° accuracy).

- Open-Source Logic: Release the control algorithm code for third-party verification (if applicable).

6.3.2 Misuse & Public Perception

Risk: The term "cheat" in the project title could imply unethical intent.

Mitigation:

- Clear Labeling: Add disclaimers: “For educational use only. Not intended for real-world lottery systems.”
- User Agreement: Require acknowledgment of ethical guidelines before system operation.

6.3.3 Data Integrity

Risk: Manipulation of wireless signals to alter outcomes.

Mitigation:

- Audit Logs: Store encrypted logs of all remote commands for post-operation review.

6.3.4 Compliance with IEEE Code of Ethics

Key Principles Applied:

- #1 (Public Welfare): Safety enclosures and emergency stops prioritize user safety.
- #3 (Honesty): Publicly disclose system limitations (e.g., 99% color detection accuracy).
- #6 (Competence): Team members completed lab safety training and electrical certifications.
- #7 (Criticism): Peer-reviewed design decisions during weekly team meetings [10].

6.4 Future work

The further work shall include enhancing the control accuracy, mechanical durability, human-computer interaction and system scalability. One way to improve the system’s performance and make the mechanism more powerful would be to equip the feeding system with position sensors (including e.g. optical encoders, Hall-effect sensors) for closed-loop feedback control of the system. Here, due to multiple approximations this would retract some of the accumulated inaccuracies but enhance long-term accuracy.

Mechanically, the high quality bearings, stiffer more rigid materials, and smoother well balanced wheel designs will help reduce friction and vibration leaving you with some hardware that lasts and feels just right. It would have Bluetooth and/or Wi-Fi wireless communication for control, logging of data and interfacing to other systems. Additionally, the design of the system can be used for more general applications e.g. in educational demonstrations of control systems and pseudo-random processes or could be tailored for use by people with limited mobility to play games interactively or interact with decision-making tools. These are just a few examples of how flexible the system can be and how it can have a positive impact on society.

References

[1] K. J. Åström and T. Hägglund, PID Controllers: Theory, Design, and Tuning. Research Triangle Park, NC: Instrument Society of America, 1995.

[2] STMicroelectronics, "STM32F103x8/xB Datasheet," Rev. 20, 2023. [Online]. Available: https://www.st.com/stm32f103

[3] Texas Instruments, "TC118S DC Motor Driver Datasheet," Rev. B, 2019.

[4] D. A. Johns and K. Martin, Analog Integrated Circuit Design. Hoboken, NJ: Wiley, 1997.

[5] Allegro Microsystems, "A3144 Hall Effect Sensor Datasheet," 2020.

[6] C. C. Chan and K. T. Chau, Modern Electric Vehicle Technology. Oxford, U.K.: Oxford Univ. Press, 2001.

[7] MathWorks, "Numerical Integration Techniques for Simulation," MATLAB Documentation, 2022. [Online]. Available: https://www.mathworks.com/help/simulink/ug/numerical-integration.html

[8] H. K. Khalil, Nonlinear Systems, 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2002.

[9] K. Ogata, Modern Control Engineering, 5th ed. Upper Saddle River, NJ: Prentice Hall, 2010.

[10] IEEE Code of Ethics, IEEE, 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

Appendix A Requirement and Verification Table

Table A System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
Individual Voltages Supply Requirements: 1. 12V Rail: Must supply a stable 12V DC output. 2. 3.3V Rail: Must supply a stable 3.3V DC output (ripple < ±33mV). 3. 4.5V Rail: Must supply a stable 4.5V DC output.	Individual Voltages Supply Tests: 1. Capture real-time dynamic changes in voltage during load with an oscilloscope under different voltage conditions. 2. Focus on voltage stability during load changes.	Y
Speed Control ("MO SPEED" Signal) Requirements: Verify precise magnetic position detection	Speed Control ("MO SPEED" Signal) Tests: 1. Measurement: Use an oscilloscope to verify voltage level (e.g., PWM amplitude) and pulse width/frequency (should correlate with motor speed). 2. Validation: Compare with a speed sensor (if available) to confirm actual RPM matches the signal.	Y
Sensing Subsystem (Magnet Position Detection) Requirements: 1. The Hall sensor must reliably detect a signal change (e.g., logic level transition) when the pointer reaches the magnet position. 2. The signal response must be instantaneous.	Signal Trigger Tests: 1. Manually rotate the pointer to the magnet position and verify the Hall sensor output using an oscilloscope or logic analyzer. 2. Ensure signal transition aligns precisely with physical positioning.	Y
User Input Subsystem (Button Control) Requirements: 1. The system must immediately detect button presses with no missed inputs. 2. When the motor is in low-speed mode and other subsystems are functioning correctly, pressing the button must precisely stop rotation at the target zone.	User Input Subsystem (Button Control): Button Response Tests: 1. Press the button repeatedly and confirm every press is registered (verified via LED/logging). Stopping Accuracy Test 2. In low-speed mode and the other sub systems normal, the pointer can stop at the selected area accurately	Y