Simulation Steering Wheel Controller driven by STM32

Zahar Veresnyuk
Faculty of Applied Sciences
Ukrainian Catholic University
Lviv, Ukraine
veresniuk.pn@ucu.edu.ua

Bobryk Vladyslav
Faculty of Applied Sciences
Ukrainian Catholic University
Lviv, Ukraine
bobryk.pn@ucu.edu.ua

Arsen Botsko
Faculty of Applied Sciences
Ukrainian Catholic University
Lviv, Ukraine
botsko.pn@ucu.edu.ua

Yulian-Volodymyr Zaiats
Faculty of Applied Sciences
Ukrainian Catholic University
Lviv, Ukraine
zaiats.pn@ucu.edu.ua

Ivan Sen
Faculty of Applied Sciences
Ukrainian Catholic University
Lviv, Ukraine
sen.pn@ucu.edu.ua

Danylo Bykov
Faculty of Applied Sciences
Ukrainian Catholic University
Lviv, Ukraine
bykov.pn@ucu.edu.ua

Ivan Shevchuk
Faculty of Applied Sciences
Ukrainian Catholic University
Lviv, Ukraine
shevchuk.pn@ucu.edu.ua

***Abstract*—This document describes general concepts, implementation path, and troubleshooting steps during the implementation of the Simulation Steering Wheel Controller with the STMicroelectronics 32-bit microcontroller family (STM32) as the central processing unit.**

***Index Terms*—steering wheel controller, STM32, USB HID, rotary encoder,pulley, load cell, button matrix, force feedback**

I. Introduction

This project aims to design and implement a steering wheel and pedal assembly for vehicle simulation using an `STM32` microcontroller. The system aims to emulate a commercial racing wheel setup compatible with driving simulators via the USB Human Interface Device (HID) protocol. The device should provide an intuitive and realistic driving interface including steering, pedals, wheel paddles & control buttons, and Force Feedback (FFB) with gearbox as extra features.

II. Implementation overview

The project implementation is partitioned into several stages, some of which can be done simultaneously:

1) Mechanical subsystem design:

- **Steering wheel** frame with integrated paddles for gear shifting.
- **Steering shaft** using pulleys+belts, with a rotary encoder [5] as a rotation angle reader
- **Pedal base** with three pedals (clutch, brake, acceleration) equipped with linear potentiometers with a geared axis.
- **Spring-based** resistance mechanism for pedals.
- Correctly selected **bearings** for the wheel shaft.
- Mechanical wheel borders using a movable toothed bar

2) `STM32` Programming and Sensor Input Processing:

- **Analog input** processing of the wheel and pedal positions (filtering/deadband)
- **Digital input** reading from the paddles and the wheel front buttons.
- Manual startup **calibration** for analog sensors.

3) USB interface for PC communication:

- **vJoy** feeder implemented on C++ to pass data from axis/buttons into the HID driver
- **STM32-PC** communication using UART protocol via a COM port

*4) FFB implementation **(extra)**:*

- **RS775 Direct Current (DC) motor** integration with the wheel shaft pulled by a pulley-belt pair.
- **BTS7960** H-Bridge driver for high-current motor control.
- **Current/torque** control implementation via PWM.
- **HID Output Reports** for two-way communication.
- Automatic startup **calibration**.

*5) Mechanical gear-shifter module implementation **(extra)**:*

- H-pattern shifter design with mode switching.
- Programming of a 3x2 microswitch matrix for gear selection.

III. Hardware Implementation

The central processing unit chosen for this implementation is the STM32F411E-DISCO development board.

A. Steering wheel positioning sensor

For commercial-grade rotation angle precision, a **rotary encoder** [5] is used directly to provide positioning accuracy.

The encoder is interfaced with the STM32F411 utilizing the hardware Timer Encoder Mode. This configuration allows for the counting of quadrature pulses directly in hardware, unburdening the CPU and ensuring zero-latency tracking of the wheel angle even during rapid rotation.

B. Wheel shaft connection and Force Feedback

To transfer motion and provide FFB, a ”pulley+belt” system connects the steering shaft to the motor.

The force-feedback is provided by an ”RS775” brushed DC motor [9]. This motor was selected for its high torque-to-weight ratio and suitability for voltage-controlled torque simulation.

Driving the motor is a ”BTS7960” high-power H-Bridge driver [10]. The STM32F411 generates two complementary Pulse Width Modulation (PWM) signals to control the driver:

- **Duty Cycle:** Controls the magnitude of the torque (Force).
- **Logic State:** Controls the direction (Clockwise/Counter-Clockwise).

C. Pedal pressing measurements

The typical pedal angular travel is $\approx 15°–30°$.

A linear **potentiometer** [1] is used for each pedal (Clutch, Brake, Throttle). To utilize the full range of the potentiometer, a custom 3D-printed spur gear pair is installed. This gear ratio amplifies the physical pedal travel, rotating the potentiometer shaft significantly more than the pedal arm.

D. Gearbox Logic

The H-pattern shifter is implemented using a matrix of microswitches. A specific ”Mode Button” allows the user to toggle the logical behavior of the shifter between two distinct modes:

- **5+R Mode:** Standard 5-speed layout with Reverse.
- **6-Speed Mode:** Configuring the physical Reverse slot to act as a forward 6th gear for modern vehicle simulation.

IV. Implementation progress and steps

A. Design considerations

A ready-to-use steering wheel design [7] was taken and printed out with the calculation of the shaft diameter and places for the paddles/buttons.
It was decided to carve a metal shaft with space for two bearings with a diameter of 42 mm external and 20 mm internal with a distance of 120 mm between each other. The base, paddles, shaft support columns and wheel limiting mechanism have been designed from the scratch using CAD software and printed on a 3D-printer. The complete wiring connection and schematic layout of the developed system are presented in Fig.(Wiring connection layout) 1.

B. Functional components

The rotary encoder [8] has been chosen as the main wheel angle rotation reader. Linear potentiometers have been chosen for 3 pedals using plastic gears to rotate the resistor. 8 + 2 buttons have been used for wheel buttons and paddles.

C. Firmware Implementation

The embedded software is developed in C using the STM32CubeIDE environment. The system operates on a **super-loop** architecture, ensuring deterministic processing of sensor data and motor control signals.

1) UART Communication Interface:

Unlike standard HID devices, this controller utilizes a **Universal Asynchronous Receiver-Transmitter (UART)** interface for data exchange with the host PC.

- **Configuration:** The communication is established at a baud rate of **115200 bps**
- **Data Packet:** The MCU transmits a formatted data packet containing the steering angle, pedal values, and button states.
- **PC Integration:** A custom user-space driver (Feeder) on the PC listens to the COM port, parses the incoming serial stream, and injects the input data into the vJoy virtual device driver.

Acronyms

DC	Direct Current
FFB	Force Feedback
HID	Human Interface Device
PWM	Pulse Width Modulation
STM32	STMicroelectronics 32-bit microcontroller family

References

[1] Wikipedia, ‘Potentiometer Title”, Resource, 2025
[2] Wikipedia, “Load Cell”, Resource, 2025
[3] Github, “vJoy repository”, Resource, 2025
[4] GitHub, “FreeJoy repository”, Resource, 2025
[5] SV “Altera”, “Enkodery — datchyky liniynykh i kruhovykh peremishchen”', Resource, 2025
[6] STMicroelectronics, “Integrated Development Environment for STM32”, Resource, 2025
[7] Steering Wheel DIY, Resource, 2025
[8] Rotary Encoder, Resource, 2025
[9] Mabuchi Motor, “RS-775VC Motor Specifications”, 2025.
[10] Infineon Technologies, “BTS7960 High Current Half Bridge Datasheet”, 2025.

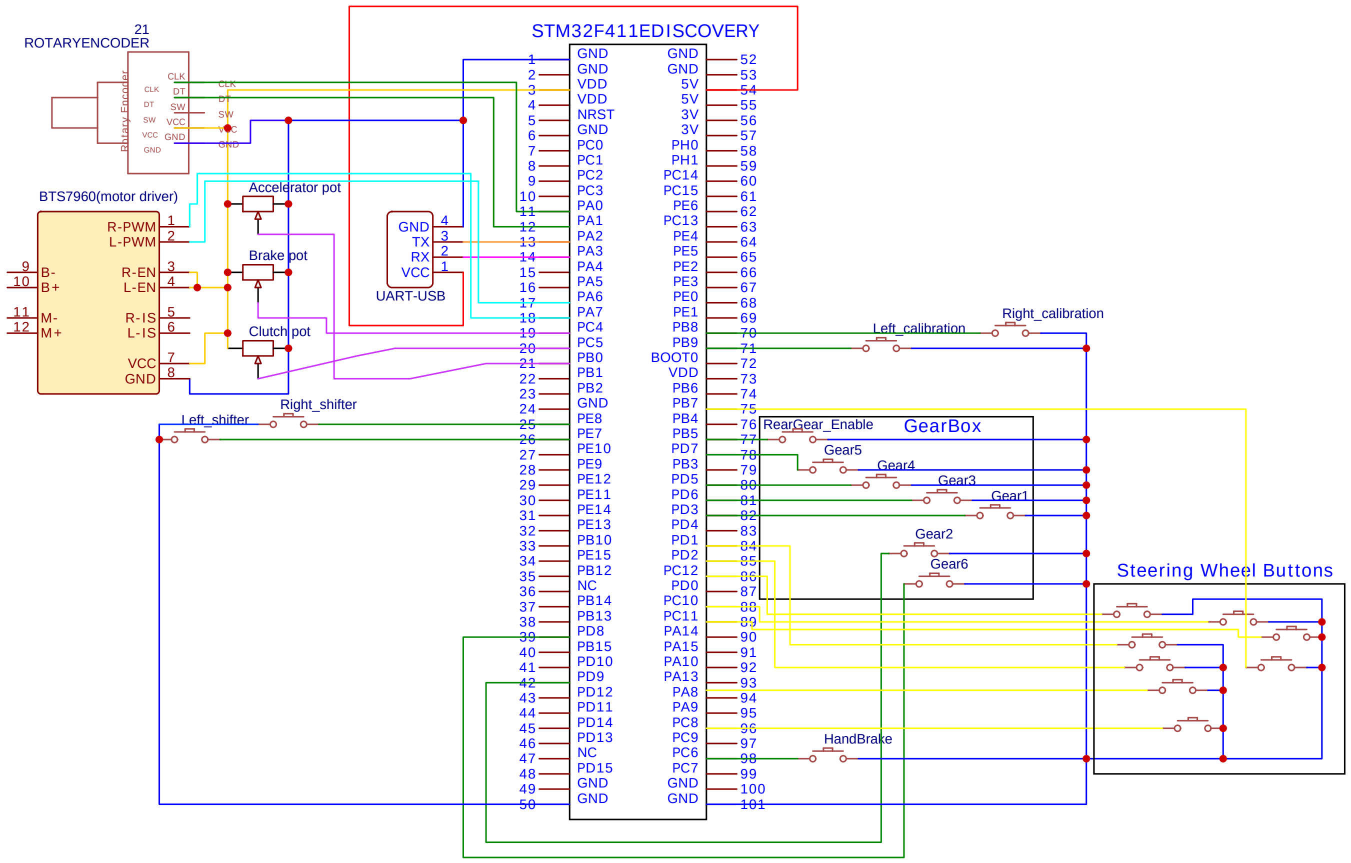

Fig. 1. Complete wiring diagram of the STM32-based Steering Wheel Controller