Namal Institute Mianwali

EE 340: Control Systems

ESL Project

Bluetooth Controlled Car with STM32F401CB and L298 Motor Driver

Student's Name	Muhammad Faizan Ikram
Roll No.	2018-UET-NML-Elect-27
Date Submission	May 30, 2021
Marks Obtained	
Instructor's signature	

Instructors: Dr. Hamza Zad

Lab Engineer: Engr. Awais Yaqoob

NAMAL INSTITUTE

Namal Institute Mianwali

Table of Contents

Introduction:

This report is about a Bluetooth Controlled Car Project which has been done with the help of STM32F401CB Launchpad and L298 Motor Driver. We have used a Bluetooth module which is used for transmit and receive data. This module helps us to remotely control our electronic devices by connecting it to a Bluetooth Module App. These modules are widely used in line follower robot, obstacle avoidance robot type of applications. In this project we have used it in a Car controlled project.

In this project we have controlled the direction of the Car. We have designed this project to control DC motors move 'Forward', 'Reverse', 'Right', 'Left' and 'Stop' We have used Keil uvision to write and upload code to STM32F401CB Launchpad to control the direction of the DC motors.

Problem Statement:

"To design a Bluetooth controlled car with **L298 Motor Driver** and **STM32F401CB** and **Bluetooth** modules."

Experimental Tools:

1. Four 9 Volt DC Motors
2. STM32F401CB Launchpad
3. L298 Motor Driver
4. DC Power Supply
5. Bluetooth-HC-05 Module

Environment (IDE):

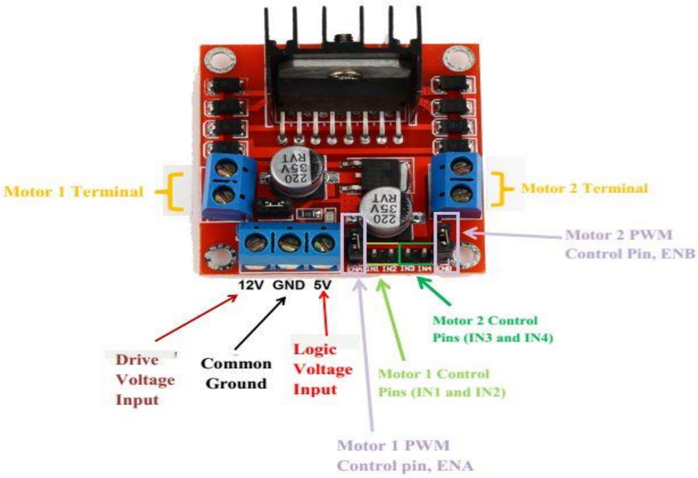
1. Keil uvision IDE
2. Proteus Software
3. CubeMX

Methodology:

Motor Driver Connection and Working:

To achieve our objective, first of all, we build our circuit by connecting four 9 volts DC Motors with **L298 Motor Driver** keeping connection as shown in figure [1] below**.** This L298 motor driver provides input signals from microcontrollers to the DC motors. This transmits signal using two H-Bridge circuits to drive two separate DC motors. On top of that each H-Bridge has an enable pin which is used to provide ON/OFF signal. We can also use Enable pins to provide PWM signal to each DC motor.

Namal Institute Mianwali



(Figure 1)

Note: This L298 Motor Driver has connection for only two DC motors. But we have connected four motors using paralleled technique. Two motors are parallel on one connection and other two motors are paralleled on second connection.

Microcontroller Connection and Working:

After that we have connected L298 Motor Driver with STM32F401CB Launchpad. For this connection, we have connected PA1, PA1. PA3 and PA4 with motor driver connection of IN1, IN2, IN3 and IN4 respectively. This transfer signals from Port-A pins to motor driver pins to turn on/off motors and drive them on particular direction depending upon signal 1/0.

Bluetooth Module Connection and Working:

At the end we connected Bluetooth module in such a way that the **Transmission Pin TX** of Bluetooth is connected to PA-10 which is **Receiver Pin RX of Microcontroller**. Similarly, Receiver Pin RX of Bluetooth is connected with Transmission Pin RX of Microcontroller.

This is because the data we send from Bluetooth is transmitted to microcontroller and then microcontroller receives this data to perform action. So, we connect pins in **transmit-receive** technology.

STM32F401CB Configuration and Working:

We have used STM32F401CB in our project for the simulation purpose. The connections [2] of STM32F401CB are shown in figure 2 below. We have used UART-1 in Async receiver mode @ 9600bps with interrupts enabled for communication of Bluetooth module. It provides transmit pin TX and receive pin RX to send and receive Bluetooth data respectively. UART data reception is in interrupt mode so whenever data is received, we check its value and decide to perform action on Port-A pins which are enabled as Output Pins. This operation continues and repeat after each interruption is received. This logic is explained in our code as well.

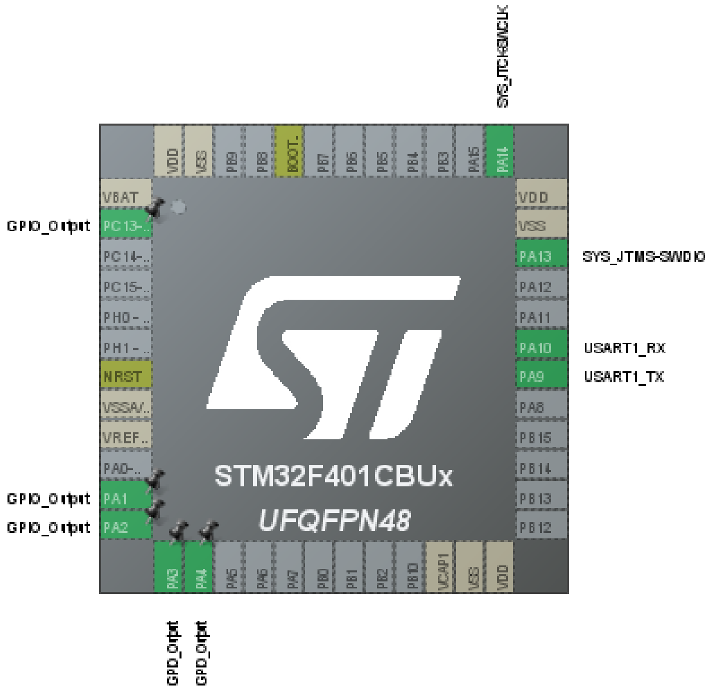
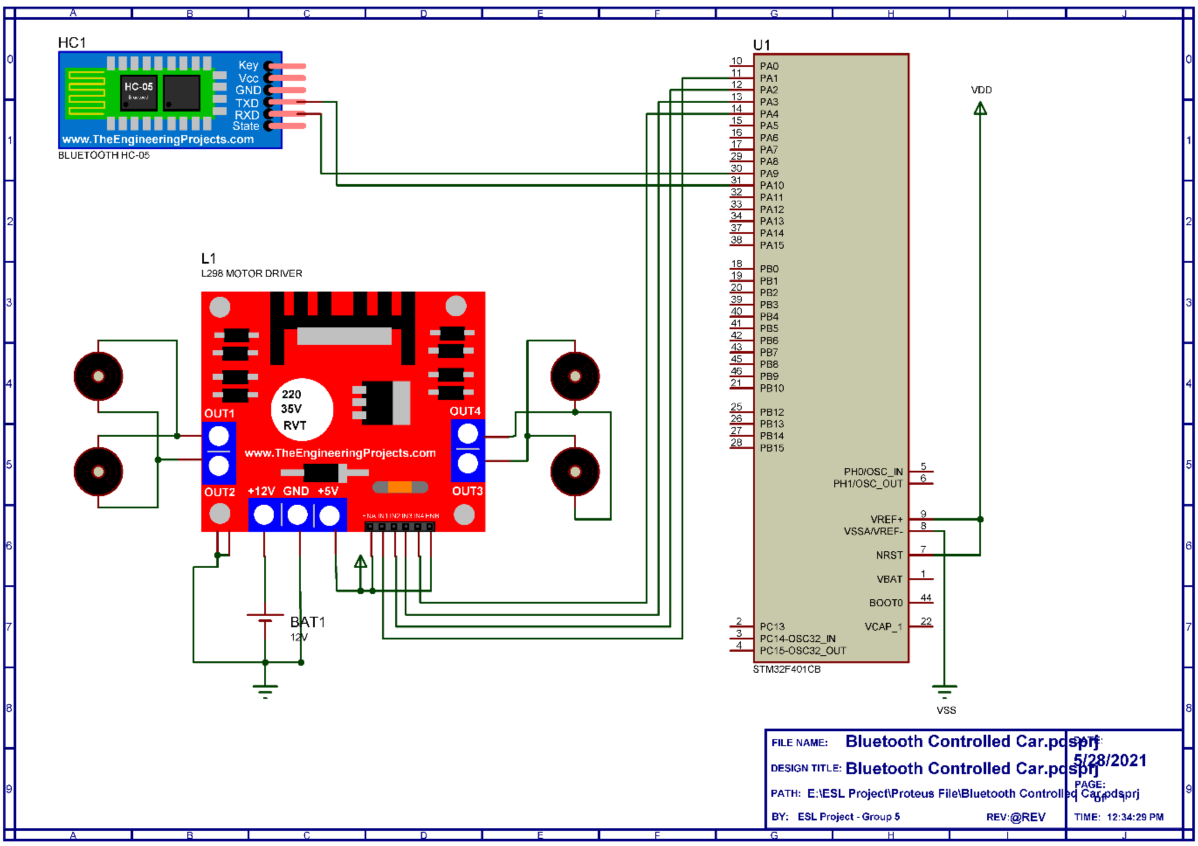


Figure 2

Circuit Diagram:



Code:

The main while loop code is given below:

```
while (1)
{
  /* USER CODE END WHILE */
  if(RX_BUFFER[0] == 'F')      // Move Forward
    {
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 1);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 1);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, 0);
    }
    else if(RX_BUFFER[0] == 'B')     // Move Backward
    {
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 1);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, 1);
    }
    else if(RX_BUFFER[0] == 'L')     // Turn Left
    {
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 1);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 0);
        HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, 1);
        HAL_Delay(1000);   // 1 sec delay
        RX_BUFFER[0] = 'S';
    }

     else if(RX_BUFFER[0] == 'R')     // Turn Right
     {
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 0);
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 1);
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 1);
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, 0);
         HAL_Delay(1000);   // 1 sec delay
         RX_BUFFER[0] = 'S';
     }
     else if(RX_BUFFER[0] == 'S')     // Stop
     {
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, 0);
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, 0);
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, 0);
         HAL_GPIO_WritePin(GPIOA, GPIO_PIN_4, 0);
     }
   /* USER CODE BEGIN 3 */
 }
```

Code Working:

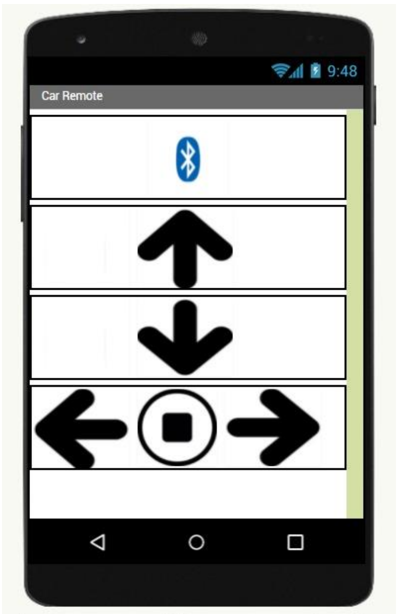
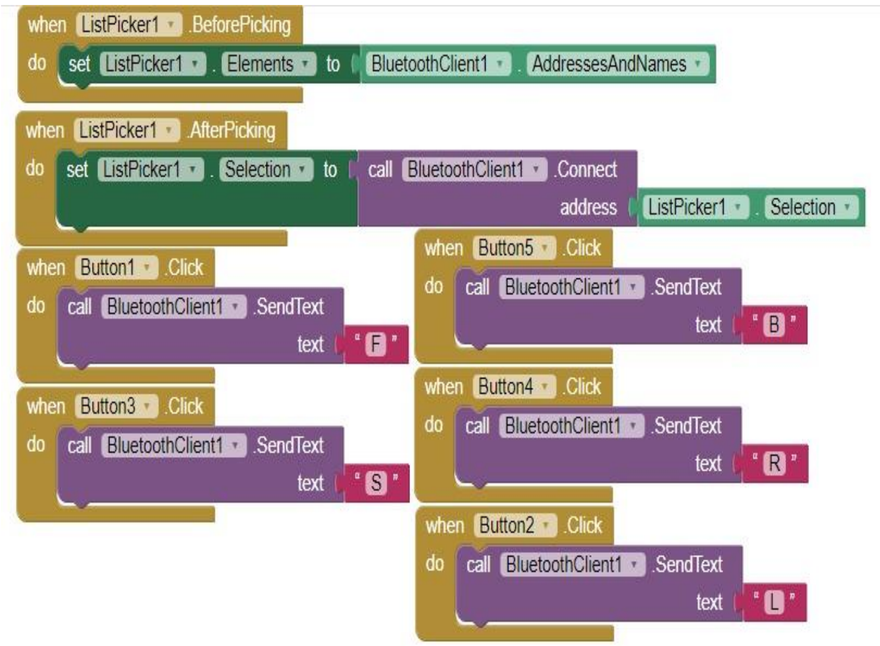
This code first sets the environment by configuring communication for the Bluetooth module and enabling GPIO output pins as well. We have defined private variables RX_BUFFER which acts in interruption mode and whenever data is received in this variable, it performs operation as shown above.

Main function of our Program is to make running DC Motors and change their direction when data is received. Since there are **two motors** (*Actual Four*) so we have set that if data received is **‘F’** it sets direction to **1010** which drives both motors in forward direction. Similarly, when data received is **‘B’** the direction sets to **0101** which drives both motors in backward direction and same for other directions as well (i.e.; **Left and Right**). We have introduced a short delay after Left and Right direction and then motors stopped because direction is achieved. We have also added some additional code [3] for Bluetooth configuration.

This remains continue and upon each interrupt, it repeats same instructions. Thus, we achieve our desired operations by having the above code uploaded on STM32F401CB Launchpad.

MIT Application:

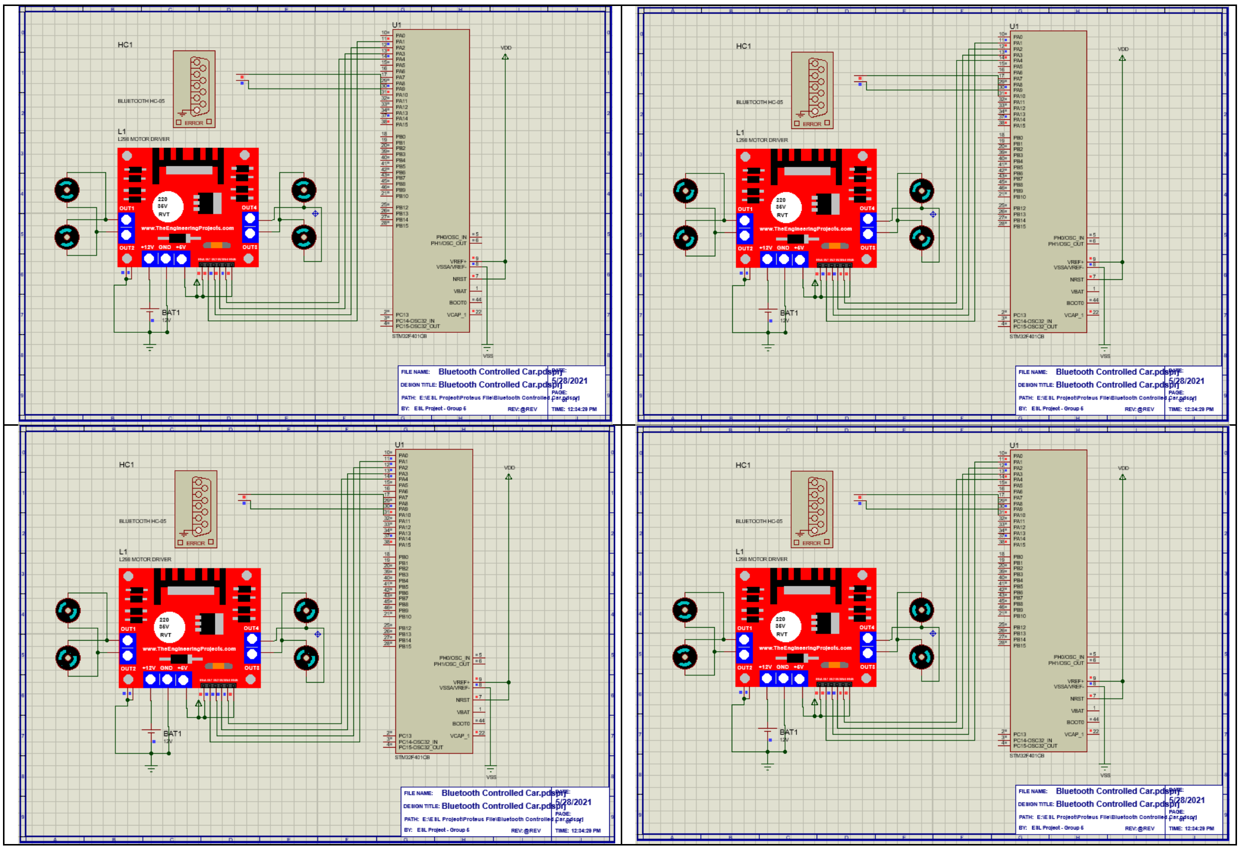
We have used MIT App Inventor to design application for communication of Bluetooth module. The app is simply designed by drag and drop and designing code on backend.

App Interface – Home Screen	Backend Code:
	

We have used same logic in our app code as we used in our Keil uvision code. We send data **‘F’** for forward direction, **‘B’** for Backward direction, **‘L’** for Left direction, **‘R’** for right direction and **‘S’** to stop the motors as shown in above figure. When this data is sent to Bluetooth module the Bluetooth receives it and transmit this data to microcontroller which performs operation upon receiving data.

Results and Discussion:

Simulation have shown exactly same results as we discussed earlier. The snapshots of the simulations are given below:



The above pictures are practical demonstration of our **Bluetooth Controlled Car Project** using STM32F401CB and L298 Motor Driver with the help of Bluetooth HC-05 module. The both motors starts moving forward when the interuption occurs and data received is ‘F’. Similarly motors drive in reverse, right and left direction upon receiving data ‘B’, ‘R’ and ‘L’ respectively. The two other motors perform same operations as they are connected parallel to other two motors.

This remains continue and upon each interrupt, it repeats same instructions. Thus, we achieve our desired results and Car moves in the directions controlled by our MIT App.

The video demonstration of the simulation results can also be seen at this [YouTube Link]

Conclusion:

At the end of this project, we conclude that we can use CubeMX to configure a microcontroller and perform specific tasks for simulation in Proteus software. We have also seen that the Bluetooth configuration can also be done for remote communication. In this project, we used our previous knowledge and lab practices to implement and design a complete **Bluetooth Controlled Car System** using **STM32F401CB, Bluetooth HC-05** and **L298 Motor Driver.** We controlled the direction of the DC Motors for car driving using Keil uvision code with the help of STM32F401CB Launchpad and MIT App using Bluetooth module. We applied the concepts and knowledge of previous labs of using STM32F401CB and the programming required for this launchpad. The goal was to test and implement the lab learning and design a complete system.

References:

[1] *DC Motor Speed and Direction Control with TM4C123 and L298N Motor Driver*. [Online].
Available at: https://microcontrollerslab.com/

[2] *Interface STM32F1 dengan Bluetooth HC-05 sebagai data logger, CubeMX, Keil UVision* [Online].
Available at: https://youtu.be/XCoqIbDEhdg

[3] *STM32 HC-05 Bluetooth Interfacing HAL* [Online].
Available at: STM32 HC-05 Bluetooth Examples Master & Slave Pairing AT Commands (deepbluembedded.com)

Appendix-A:

Additional Code:

```
#define BUFFER_LEN  1

/* Private variables ---------------------------------------------------------*/

UART_HandleTypeDef huart1;

/*        USER CODE BEGIN    */

void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
   if(huart->Instance == huart1.Instance)
   {
          HAL_UART_Receive_IT(&huart1, RX_BUFFER, BUFFER_LEN);
   }
}
```

MIT App Helping Material:

https://gallery.appinventor.mit.edu/?galleryid=6328207850209280

Namal Institute Mianwali

NAMAL INSTITUTE

Assessment Rubric for Lab

Method of Evaluation: Lab report and instructor observation during Lab Session.

Outcomes Assessed:

CLO1: Ability to analyze and extract meaningful information from observed data (P).

CLO2: Ability to function as an effective team member (A).

CLO3: Ability to follow instructions and convey experiment results in an effective manner (A).

CLO5: Ability to use the techniques, skills, and modern engineering tools necessary to practice control engineering (P).

Performance	**Good (4-5)**	**Satisfactory (2-3)**	**Unsatisfactory (1)**	**Marks**
Task Completion (CLO5)	All tasks completed correctly	Most tasks completed correctly with some incomplete or incorrect	Most tasks either incomplete or incorrect	
Teamwork (CLO2)	Actively engages and cooperates with other group members in an effective manner	Cooperates with other group members in a reasonable manner	Distracts or discourages other group members from conducting the experiment	
Lab Report (CLO3)	Lab report has been filled in neatly with proper grammar and scientific terminology	Parts of the report not filled in neatly or with improper grammar	Report filled in illegible writing with improper pucntuations or a casual, non-scientific tone	
Discussion (CLO1)	Provides meaningful interpretation of results. Provides scientific reasoning and draws appropriate conclusions based on data.	Provides some interpretation of results. Includes some sort of conclusions.	Interpretation of results is not clearly explained. No conclusions are made based on the results.	
Validating results from instructor (CLO3)	Shows the obtained graphs/results at all points indicated in the manual.	Shows the graphs/results at some of the points indicated in the manual	Does not show the graphs/results at all.	
Difference of report from Lab Partner (CLO2)	The report is clearly filled in independently of the lab partner with same findings but with distinct text.	Most of the report is filled in independently. There is overlap with the lab partner in text of some answers	Significant overlap of report with lab partner. Clearly not filled in independently.	
			Total	