MCT-241: Embedded Systems II

Speed Control & Current & RPM Measurement of BLDC Motor using TM4C123GXL

Submitted By:

Muhammad Haider Ali

Muhammad Burhan

Abdul Hadi

Registration No:

2022-MC-45

2022-MC-65

2022-MC-20

Department of Mechatronics & Control Engineering, University of Engineering & Technology, Lahore

December 2, 2024

Objective:

The primary objectives of this project are:

1. To implement speed control for a Brushless DC (BLDC) motor using the TM4C123GXL microcontroller.
2. To measure and monitor the current drawn by the motor.
3. To calculate and display the RPM (Revolutions Per Minute) of the motor.
4. Use UART interfacing to display the graph values.

Materials Required:

1. TM4C123GXL (Tiva C LaunchPad – TM4C123GH6PM)
2. BLDC motor (A2212).
3. ESC (30A).
4. Hall-effect sensors (usually built into the BLDC motor for RPM measurement)
5. Current sensor (e.g., ACS712 or INA219)
6. 12V/24V DC power supply (based on motor specification)
7. 5V DC supply for the microcontroller (optional if USB powered)
8. Resistors, capacitors, diodes for circuit protection and signal conditioning
9. Pull-up resistors for Hall sensor signals
10. Potentiometer (for manual speed control input)
11. Oscilloscope (for debugging and waveform analysis)
12. Breadboard and jumper wires for connections
13. PC or laptop with programming IDE (Code Composer Studio or Keil)

Schematic Diagram:

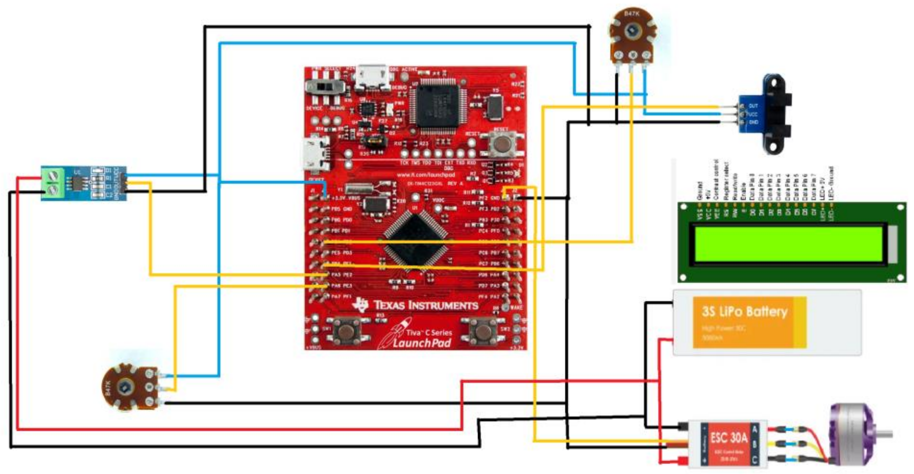

Fig1.1: Schematic Diagram

Circuit Implementation:

PWM Generation:

- Configure PWM pins on the TM4C123GXL for speed control.
- Adjust the duty cycle based on the desired speed.

Current Measurement:

- Read the analog signal from the current sensor using the ADC.
- Convert ADC readings to current values using the sensor's specifications.

RPM Calculation:

- Use Hall sensor feedback to count pulses over a specific time period.
- Calculate RPM using the formula:

$$RPM = \left(\frac{\text{Pulse Count} \times 60}{\text{Number of Pole Pairs} \times \text{Time Period (s)}} \right)$$

Speed Control:

- Implement a PID controller or simple feedback mechanism to adjust PWM duty cycle based on the desired RPM.

Display:

- Use UART or I2C to display RPM and current data on a serial monitor or an LCD.

The circuit operates by controlling a BLDC motor using pulse-width modulation (PWM) signals generated by the microcontroller. The GPIO pins used for PWM output are configured to control the motor’s speed and direction. In the code, we configure the appropriate GPIO pins (such as GPIO pins 18, 19, 20, 21 for PWM output, depending on the microcontroller) for motor control. These pins are set to output PWM signals with a frequency typically ranging from 20 kHz to 50 kHz, depending on the motor's requirements. The microcontroller's peripherals are configured to generate PWM signals, and the motor's speed is controlled by adjusting the duty cycle of these signals. The code ensures that the PWM signals are synchronized and that the rotor is driven in the correct sequence, allowing for efficient motor operation. We also ensure that any necessary PLL (Phase-Locked Loop) settings are configured to maintain a stable clock frequency for the PWM signals. The software handles motor initialization, control, and feedback from sensors, enabling precise control of the motor's speed and position based on the user input or predefined settings.

<u>Component Descriptions:</u>

- **TM4C123GXL (TivaC LaunchPad – TM4C123GH6PM):**

An ARM Cortex-M4-based microcontroller board ideal for motor control.

Offers onboard peripherals like GPIO, PWM, and UART for system interfacing.

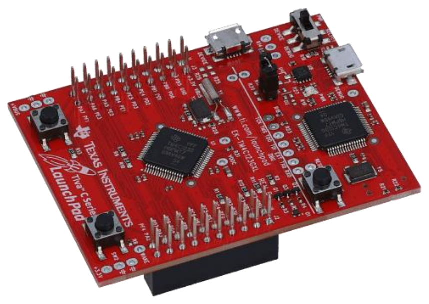

Fig1.2: TM4C123GXL (TivaC LaunchPad – TM4C123GH6PM)

- **BLDC Motor (A2212):**

A high-efficiency brushless DC motor used in drones and RC vehicles.

Delivers high torque and RPM with low power consumption.

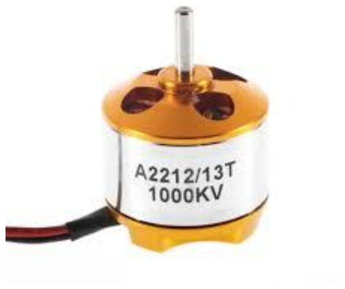

Fig1.3: A2212 BLDC Motor 1400KV

- **ESC (30A):**

Controls speed and direction of the BLDC motor via PWM signals.

Handles currents up to 30A for reliable motor operation.

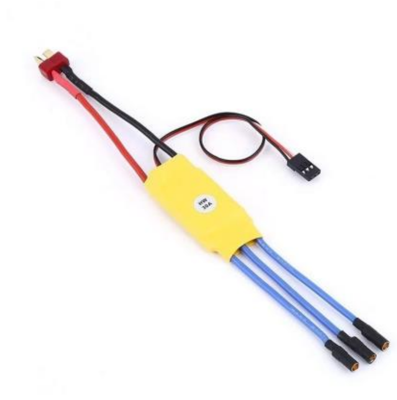

Fig1.4: 30A ESC

- **Hall-Effect Sensors:**

Built into the BLDC motor for rotor position and RPM measurement.

Essential for closed-loop motor speed and position control.

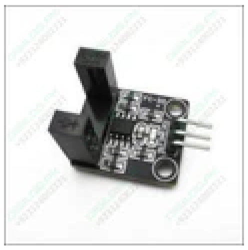

Fig1.4: Fc-33 10mm Arduino Rpm Sensor

- **Current Sensor (ACS712/INA219):**

Measures current flow to monitor power usage and detect faults.

Provides analog or digital outputs for microcontroller interfacing.

Fig1.5: 30A ACS712 Current Sensor

- **12V/24V DC Power Supply:**

Supplies the necessary voltage and current for motor operation.

Voltage selection depends on the motor's rating (12V or 24V).

Fig1.6: 12V 7A Power Supply

- **5V DC Supply (optional):**

Powers the microcontroller if USB is not used.

Ensures stable operation of the board and peripherals.

Fig1.7: 5V 2A Power Supply

- **Resistors, Capacitors, Diodes:**

Protect circuits from voltage spikes and filter noise in signals.

Used in signal conditioning and power regulation circuits.

Fig1.8: Resistor, Capacitor, Diode

- **Pull-Up Resistors:**

Ensures stable high signals from Hall-effect sensors.

Prevents floating input pins on the microcontroller.

- **Potentiometer:**

Acts as a manual input device for adjustable speed control.

Provides an analog voltage output to the microcontroller.

Fig1.9: 10K, 50K Potentiometer

- **Oscilloscope:**

Analyzes electrical waveforms for debugging and signal validation.

Helps visualize PWM signals and sensor outputs.

Fig2.0: Oscilloscope

- **Breadboard and Jumper Wires:**

Facilitate prototyping and temporary circuit assembly.

Allow quick and flexible connections between components.

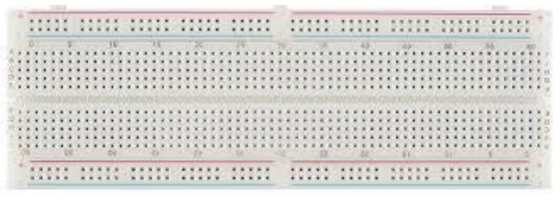

Fig2.1: Breadboard

Fig2.2: Jumper Wires (Female, Male)

<u>**Component Pinout:**</u>

Peripheral	**Pin**	**Port**	**Description**
LCD (4-bit Mode)	PB4-PB7	Port B	LCD data/control pins (4-bit mode).
LCD RS	PB4	Port B	Register Select.
LCD RW	PB5	Port B	Read/Write control.
LCD EN	PB6	Port B	Enable signal.
UART0	PA0, PA1	Port A	PA0: UART0 Rx, PA1: UART0 Tx.
PWM Output	PF2	Port F	PWM signal for M1PWM6.
ADC Input (Pot.)	PE3	Port E	ADC input for potentiometer (AIN0).
ADC Input (Current Sensor)	PE2	Port E	ADC input for current sensor (AIN1).
External Interrupt	PE0	Port E	External interrupt input for RPM.

<u>Microcontroller Pins:</u>

Pin	**Port**	**Functionality**	**Usage**
PA0	UART0 Rx	UART Receive	Virtual COM Port for receiving data from the PC
PA1	UART0 Tx	UART Transmit	Virtual COM Port for sending data to the PC
PB4	GPIOB Pin 4	RS (LCD Register Select)	LCD Control Line RS
PB5	GPIOB Pin 5	RW (LCD Read/Write)	LCD Control Line RW
PB6	GPIOB Pin 6	EN (LCD Enable)	LCD Control Line Enable (EN)
PB7-PB4	GPIOB Pin 4-7	Data lines for LCD	4-bit Data Bus for LCD
PE0	GPIOB Pin 0	External Interrupt Input	RPM Sensor Interrupt
PE2	GPIOB Pin 2	Analog Input (AIN1)	Current Sensor Input
PE3	GPIOB Pin 3	Analog Input (AIN0)	Potentiometer Input
PF2	GPIOB Pin 2	M1PWM6	PWM Output for Motor

Steps to configure/setup the graph displaying functionality:

Install Required Libraries:

Ensure you have the necessary Python libraries installed:

```
pip install pyserial matplotlib
```

Serial Port Setup:

The code initializes a serial connection to the device:

```
ser = serial.Serial('COM7', 9600, timeout=1)
```

- Replace 'COM7' with the appropriate port for your device.
- 9600 is the baud rate; ensure it matches your device's settings.
- The timeout parameter ensures the program doesn't hang if no data is received.

Global Variables and Data Buffers:

- Global variables are used to store the parsed rpm, current, and duty values.
- Circular buffers (deque) are used for efficient data storage and real-time plotting:

```
time_data = deque(maxlen=100)

rpm_data = deque(maxlen=100)

current_data = deque(maxlen=100)
```

- maxlen=100 ensures only the latest 100 data points are stored.

Serial Data Reading:

The read_serial_data function continuously reads and parses data from the serial port:

```
if "RPM:" in line and "DUTY :" in line and "Current :" in line: ...
```

- The expected data format is: RPM: <value>, DUTY : <value>, Current :<value>.
- It normalizes spaces around colons to ensure consistent parsing.

If the data format is unexpected or an error occurs during parsing, the issue is logged:

```
print(f"Error parsing data: {e}. Line received: {line}")
```

Updating Graph Data:

The update_graph_data function appends new data points to the deque buffers:

```
time_data.append(current_time) rpm_data.append(rpm) current_data.append(current)
```

- current_time calculates elapsed time since the program started.

Real-Time Plotting:

The update_plots function refreshes the plots:

- ax1 displays RPM vs. Time.
- ax2 displays Current vs. Time.

Styling includes:

- Titles, axis labels, and legends with appropriate colors.
- A fixed y-axis range (0-10,000 for RPM and 0-6 for Current).
- A black background with white text and grid lines.

Multi-Threading

- Serial data reading runs on a separate thread:

```
serial_thread = threading.Thread(target=read_serial_data, daemon=True)
serial_thread.start()
```

- This prevents the main thread (handling the GUI) from being blocked.

Main Loop:

- The main loop continuously updates the plots

```
while True: update_plots()
```

- plt.pause(0.1) introduces a small delay to ensure smooth updates.

Steps to run and use the project:

To run and use the project described in the code above, follow these steps:

1. Set up the Development Environment

- Ensure you have the necessary software and hardware for working with the **TM4C123** microcontroller (Tiva C series). This includes:

    - **TI Tiva C Series LaunchPad** or similar development board
    - **Code Composer Studio (CCS)** or **Keil uVision** for development
    - A **JTAG or SWD programmer** to upload code to the board (if not using a built-in USB programmer)
    - **TivaWare** software package from Texas Instruments for libraries and drivers (optional but recommended)

2. Project Setup

- Create a new project in **Code Composer Studio** or your preferred IDE for the **TM4C123GH6PM** microcontroller.
- Import or copy the provided code into your project.
- Ensure you are linking to the correct startup files and device libraries for the TM4C123.

3. Configure the Clock

- The system clock is configured using the PLL_Init() function, which sets the clock to 16 MHz using PLL (Phase-Locked Loop). Make sure this setup is correct in your project to avoid clock-related issues.

4. Peripheral Setup

- **GPIO Initialization**:
    - The project uses various GPIO pins for LCD communication, UART communication, and PWM output (e.g., PF2 for PWM signal). Make sure these pins are correctly configured and that the appropriate ports are enabled.
- **PWM Setup**:
    - The function PF2_as_M1PWM6_Init() sets up the PWM signal output on **PF2** for controlling motors or other PWM-driven components.
    - The function PWM_Module1_Channel6_Init() configures the PWM parameters, such as frequency and duty cycle.
- **ADC Setup**:
    - ADC_Enable() initializes the ADC to read the potentiometer and current sensor connected to **PE3** and **PE2**.
- **Interrupts**:
    - **PE0** is used for external interrupts, presumably for counting pulses related to RPM. The interrupt handler is set up in the function GPIOE_Handler(), which increments the RPM count when a falling edge is detected.

5. LCD Setup

- **LCD Communication**:
  - The LCD is connected to **PORTB (PB4 - PB7)**. The initialization and data writing functions (LCD4bits_Init(), LCD_Write4bits(), LCD_WriteString()) ensure that the LCD is correctly initialized and updated.
  - The display shows RPM and current values calculated from ADC readings.

6. UART Setup

- The UART0 module is initialized to communicate over the serial interface using **PA0 (Rx) and PA1 (Tx)**. This allows the system to send data like RPM, duty cycle, and current values over UART.
- Make sure that you use a **USB-to-serial adapter** if you want to monitor UART output on a PC.

7. Main Program Logic

- **ADC Conversion**:
  - The main loop starts ADC conversions on PE3 and PE2 to read the potentiometer (for duty cycle control) and current sensor.
- **PWM Control**:
  - The duty cycle for the PWM is calculated based on the potentiometer input. The PWM signal on PF2 is adjusted accordingly.
- **RPM Display**:
  - The current RPM is incremented by the interrupt handler on PE0, and the RPM value is displayed on the LCD.
- **UART Output**:
  - The calculated RPM, duty cycle, and current values are sent via UART for logging or debugging.

8. Compile and Upload

- Compile the project and resolve any errors or warnings.
- Load the binary onto the TM4C123 using the debugger or programming tool.

9. Testing

- **Verify PWM Output**: Use an oscilloscope or multimeter to measure the PWM signal on **PF2** to ensure it is functioning correctly.
- **Verify LCD Output**: The RPM and current values should be displayed on the LCD.

- **Verify UART Output**: Use a serial terminal (like PuTTY or Tera Term) to read the UART output from the TM4C123. You should see the RPM, duty cycle, and current being printed at regular intervals.
- **Verify Interrupts**: Ensure that the RPM counter is incrementing correctly based on the external interrupt on **PE0**.

10. Adjustments

- You can fine-tune the code based on your actual hardware setup (e.g., sensor calibration, PWM frequency, etc.).
- If you plan to use the UART output on a different interface, you may need to adjust the UART settings (baud rate, etc.) or modify the code accordingly.

By following these steps, you should be able to run and use the project for controlling PWM signals based on a potentiometer, reading current values from a sensor, and displaying/communicating data via LCD and UART.

Conclusion:

In this project, we successfully developed a motor control system that integrates various components to monitor and control motor speed and current consumption. The application utilizes a Tiva C microcontroller and several hardware peripherals, such as an LCD display, potentiometer, current sensor, and an external interrupt for RPM measurement. By leveraging PWM control, we were able to adjust the motor speed based on user input from the potentiometer, while real-time monitoring of motor parameters such as RPM and current consumption was displayed on the LCD and transmitted via UART.

This system offers several key advantages:

- **Real-time Control and Monitoring:** The combination of PWM control and real-time sensor feedback provides accurate control over the motor speed and a clear picture of its performance through real-time data displays on the LCD.
- **Ease of Use:** The use of a potentiometer for speed adjustment offers a simple and intuitive user interface for controlling the motor, making it ideal for applications that require manual control of motor parameters.
- **Efficiency:** The system efficiently integrates ADC, PWM, and interrupt handling to perform real-time operations with minimal computational overhead, ensuring smooth motor operation and fast response times.

Potential Improvements

While the system demonstrates the core functionality of motor control and monitoring, there are several areas for improvement and enhancement:

- **Enhanced Communication Protocols:** The current UART communication provides basic data transmission, but upgrading to more advanced communication protocols, such as SPI or I2C, could allow for faster data exchange, especially in systems with multiple sensors or controllers.
- **User Interface:** The current system only provides basic feedback on the LCD. Implementing a more sophisticated graphical user interface (GUI) could provide a clearer representation of motor performance, such as speed trends or current usage over time.
- **Safety Features:** Adding safety mechanisms such as overcurrent protection, motor stall detection, or thermal monitoring would improve the system's robustness and ensure safe operation in real-world applications.
- **Wireless Control:** Incorporating wireless communication (e.g., Bluetooth or Wi-Fi) would allow for remote monitoring and control of the motor, making it more versatile in applications like robotics or automation.
- **Advanced Motor Control Algorithms:** Implementing more sophisticated motor control algorithms, such as PID (Proportional-Integral-Derivative) control, could result in smoother motor operation, especially in applications that require precise speed regulation.

Overall, this project provides a solid foundation for motor control and monitoring systems, with several opportunities for future enhancement to increase functionality, safety, and usability.

BOM (Bill of Materials):

Sr.	NAME	Description	Quantity	Price per piece	Total price	Contact info of vendor
01	BLDC	The **A2212 1400 kV BLDC motor** is a popular brushless motor commonly used in drones and other high-speed applications. With a 1400 kV rating, it spins at 1400 RPM per volt, providing a good balance of speed and power for medium-sized UAVs and other devices. The motor is known for its efficiency,	3	1200	3600	https://epro.pk epro rehman electronics center , 22 yasin street , hall road lahore +92 309 3335775

		reliability, and low maintenance, as it operates without brushes. It is typically paired with a 30A ESC (Electronic Speed Controller) and is designed for use with 3S or 4S LiPo batteries, offering smooth and precise control for high-performance applications.				
02	ESC 30A	A high efficiency 30A Electronic Speed Controller (ESC) used for regulating the speed, direction, and braking of BLDC motors. Designed with multiple settings to control various motor types, it ensures smooth operation and includes safety features to prevent damage.	3	1150	3450	https://epro.pk epro rehman electronics center , 22 yasin street , hall road lahore +92 309 3335775
03	P450 FLAME WHEEL ARM	A robust and lightweight Flamewheel Arm Frame, designed for drones and UAVs. Constructed from durable materials, it ensures stability and rigidity during flight, reducing vibrations for precise control. Suitable for mounting motors, ESCs, and other components.	1	800	800	https://epro.pk epro rehman electronics center , 22 yasin street , hall road lahore +92 309 3335775
04	POWER SUPPLY 12 V 6 A	A regulated power supply module delivering 12V at 6A. It provides consistent power to the microcontroller, sensors, and other electronics, ensuring stable and reliable	1	750	750	https://epro.pk epro rehman electronics center , 22 yasin street , hall road lahore

		operation of the overall system				+92 309 3335775
05	RPM SENSOR MODULE	An optocoupler-based RPM sensor module used to measure the rotational speed of the BLDC motor. It converts the motor's optical signals into digital signals, enabling accurate RPM measurements and data logging for analysis and display.	4	320	1280	https://epro.pk epro rehman electronics center , 22 yasin street , hall road lahore +92 309 3335775
06	CURRENT SENSOR ACS712	The ACS712 30A Current Sensor Module provides precise current measurements, ideal for monitoring the motor's power consumption. It outputs an analog signal proportional to the current flow, aiding in power management and safety monitoring of the system.	3	300	900	https://epro.pk +92 309 3335775
07	PROPELLER 8045	Durable and well-balanced propellers designed for quadcopter and drone use. These propellers provide efficient thrust, contributing to stable and responsive flight performance, essential for smooth control and maneuverability.	2	300	600	https://epro.pk +92 309 3335775
08	16x2 LCD	A character-based LCD display that shows 16 characters per line over 2 lines. Widely used for displaying real-time data such as RPM, current	4	240	960	https://epro.pk +92 309 3335775

		measurements, and status messages. It uses a 4-bit or 8-bit parallel interface for simple communication with microcontrollers.				
09	POTENTIOMETER	A variable resistor used to control the motor's speed by adjusting the voltage. It can fine-tune parameters in the system, offering precision control over the desired output range for the motor and other components.	5	60	300	https://epro.pk epro rehman electronics center , 22 yasin street , hall road lahore +92 309 3335775
10	FEMALE HEADER FOR PCB	To connect components that can not be placed on pcb with the pcb	4	30	120	The I.C. Shop 16 A Hall road 0322-4401074 WWW.THEICSHOP.PK
11	FECL3 SOLUTION	Ferric Chloride ($FeCl_3$) solution used for etching copper-clad PCBs during the circuit board manufacturing process. It reacts with copper to create precise and intricate circuit traces necessary for component connections.	1	200	200	https://epro.pk epro rehman electronics center , 22 yasin street , hall road lahore +92 309 3335775
12	Tapes(insulation & double sided)	For sticking components and wires	1	200	200	Gssc printing shop +92 323 4373141
13	Solder wire	For attachment of electronics with pcb	1	300	300	The I.C. Shop 16 A Hall Road 0322-4401074

						WWW.THEICSHOP.PK
14	BLDC Screws	To attach bldc with the drone arm	4	Already available		