Obstacle Avoiding Rover

1st Adi Šoše
Facultaty of Information Technologies
"Džemal Bijedić" University
Mostar, Bosnia and Herzegovina
Mentor: doc. dr. Elmir Babović
adi.sose@edu.fit.ba

***Abstract*—This paper covers the process of building a four wheeled rover using the STM32F103C8T6 microcontroller. The rover can be controlled by an Android app.**

***Index Terms*—STM32CubeMX, proteus, embedded, rover, encoder, DC motor, motor driver, UART, ultrasonic**

I. Introduction

The main purpose of this project is to build a rover which has the following functionalities. Speed of the rover can be controller by turning a knob on its top. Direction can be directly controlled by user via bluetooth. When a robot gets close to an obstacle (17cm) it blocks the movement in the direction of the obstacle until it is either cleared or the rover is far away enough again. The paper will not be covering the physical assembly of the non-electronic parts.

II. Hardware components

Significant features and specifications of hardware used in this project will be noted in this section.

A. Microcontroller

The microcontroller chosen for this project is STM32F103C8T6. It's core is an ARM Cortex-M3 32-bit processor with a RISC instruction set. While the maximum oprating frequency of this chip is 72MHz it is also paired with 128KB of flash memory and up to 20KB of SRAM. [1]. This board is made by ST-Electronics[1] and among the enthusiasts recognized under the name "Blue pill".

TABLE I
MICROCONTROLLER DIMENSIONS

Height	Width
53mm	23mm

M3 processors made by ARM[2] provide a cheap architecture which has low power comsumption levels which makes it a good choice for this type of projects.

The board itself has 48 pins with 37 of them being General-purpose I/O (PA0-PA15, PB0-PB15, PC13-PC15) and 15 of them supporting pulse width modulation (PA0-PA3, PA6-PA10, PB0-PB1, PB6-PB9). All digital pins have interrupt capability. The pins are separated into 3 ports, A, B and C.

[1]https://www.st.com/
[2]https://www.arm.com/

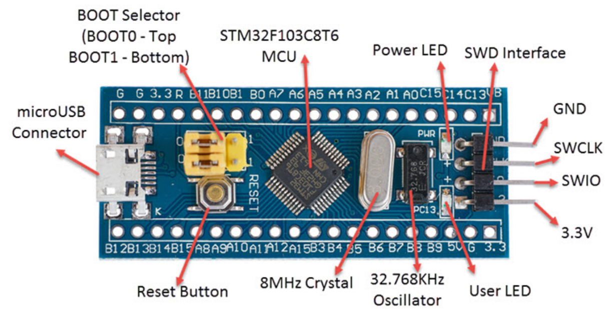

Fig. 1. STM32F103C8T6

B. Rotary encoder

Rotary encoder is a device used for sensing rotation as well as the direction of the rotation. Even though there are many types of encoders that subject will not be covered in this paper. The rotary encoder used in this project is an absolute incremental encoder KY-040. The way absolute incremental encoders work is by having the controller sense two output signals (Clock and Deadtime pins). When the handle of the encoder is both of the signals are set to high voltage. The order in which the signals are triggered determines direction of the rotation. To sum this up, once the controller recieves interrupt on the first pin,direction of rotation can be determined by checking the state of the of the second pin.

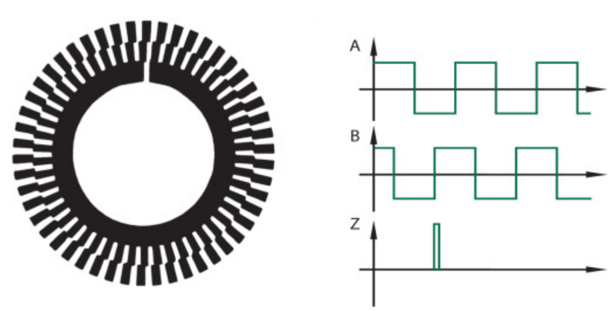

Fig. 2. Absolute incremental encoder

Tha role of an encoder in this project is to tweak the speed of rovers wheels by turning it clockwise or counterclockwise.

C. DC Motor

DC motors are consisted of to key components: an armature (rotating part) and a stator (stationary part). By turning the electric into magnetic energy, this type of motor achieves rotation by turning its magneticly polarized shaft. The direction in which the shaft turns is determined by the direction of current flowing trough the motor.

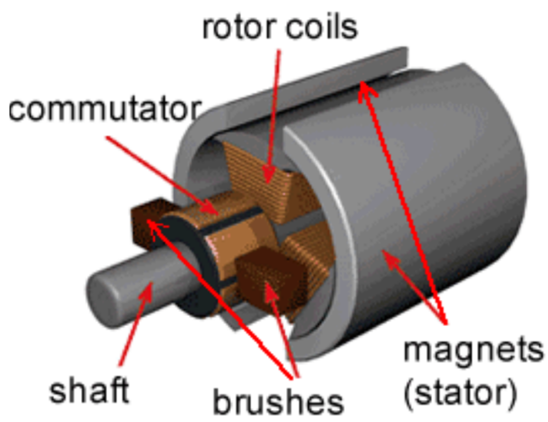

Fig. 3. DC Motor internals

This project requires four DC motors, one for each wheel.

D. Motor driver

Changing the current flow in a DC motor to achieve rotation using a microcontroller is not a trivial task. This problem lead to development of an electric circuit which changes the polairity of applied voltage called the H-bridge. H-bridges consist of four switches which act as a device that can change the flow of current going thorough a motor.

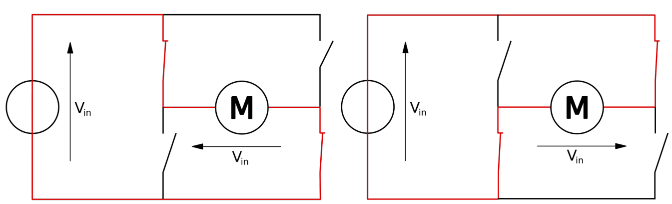

Fig. 4. H-bridge states

A more complex component which resolves this problem is a motor driver. These components, such as L298N which was used in this project, have the ability to control when and in which direction the motor rotates. This type of motor driver is popular among hobbyists mainly due to its low price. L298N is a dual-channel driver meaning that it can run two motors simultaneously . The driver operates under 5V with the maximum power consumption of 20W[3]. There are three main pins per motor with them being two input pins and an enbale pin. The purpose of enable pin is to connect a PWM[4] to it in order to control the speed of a motor. Remaining two pins are used to determine the direction of rotation. [2]

TABLE II
INPUT PIN COMBINATIONS

Input 1	Input 2	Direction
HIGH	HIGH	None
HIGH	LOW	Direction 1
LOW	HIGH	Directino 2
LOW	LOW	None

Since this project requires four DC motors there will be two L298Ns runnig two motors each.

E. Bluetooth module

In order to control the robot there needs to be a way of communication. Since there is already an android app designed to control a robot of different purpose[5] there is no need to look further for a solution. The solution for communication between devices is bluetooth. Most popular bluetooth module for these kind of projects is HC-05 and its newer version HC-06. HC-05 was the module used in this project due to its ease of use compared to HC-06 which is known to have some bugs. This module uses PSK[6] to communicate with other devices while UART[7] protocol is implemented for the component to communicate with the rest of the circuit [3]. UART protocol utilizes one data channel which makes it slower compared to its counterparts I^2C[8] and SPI[9] which is compensated by its lower resource usage. The HC-05 has two pins that support UART, recieve and transmit pin which are used for the circuit to send and recieve data outside of it through serial communication.

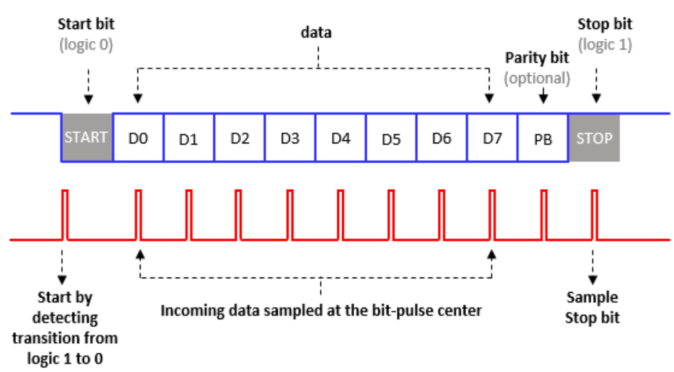

Fig. 5. UART Data Package

The speed of this communication is 9600 bits per second.

[3]when the temperature T = 75 °C
[4]PWM - Pulse Width Modulation

[5]https://www.gihtub.com/Adi-Sose/Fall-E
[6]PSK- Phase Shift Keying
[7]UART - Universal Asynchronous Receiver/Transmitter
[8]Inter-Integrated Circuit
[9]Serial Peripheral Interface

F. Ultrasonic Sensor

For the rover to be able to detect obstacles some kind of distance sensor is needed. Widely available ultrasonic sensor family is the HC-SR type. The way in which the HC-SR04, component used in the project, works is by sending a 10µs signal to its trigger pin. Then an ultrasonic wave is sent from the device and once it is returned the distance can be measured by the time it took for the signal to return which is equal to the time that the echo pin will be set to its high state. [4] Since the speed of sound is $340\frac{m}{s}$ a smiple equation can be created for calculating the distance.

$$v = 340\frac{m}{s} = 0.034\frac{cm}{\mu s} \tag{1}$$

$$t = \frac{s}{v} \tag{2}$$

$$s = t * \frac{0.034}{2} \tag{3}$$

Since the rover doesn't have a strict direction of movement there are two sensors of this type mounted on the front and the back of the robot.

III. The circuit

As mentioned in the text above the rover is consisted of six main components

TABLE III
COMPONENTS

Component	No. of pieces
STM32F103C8T6	1
KY-040	1
DC Motor	4
L298N	2
HC-05	1
HC-SR04	2

A. Power

Only components which are not connected to the micro-controller directly are the four DC motors. Power source for all the components is MCU's[10] 3.3V output. All the components share common grounding through the microcontroller. Secondary power source needed to run the motors is a 12V battery.

B. Scheme

The scheme was created using a free open-source CAD software for the design of electronics hardware[11]. As it hasn't been decided yet which type of battery will be used as a 12V power source for the motors to able to run propperly there is currently no real component in the figure 6. The components are connected using jumper wires with a 0.90mm diameter. There is also an unused pin on the KY-040 encoder which acts as a button that triggers once the leaver of the rotary encoder is pushed [5]. L298N doesn't utilize its 5V input/output pin as it is not needed for the project.

[10]MCU - Microcontroller Unit
[11]https://fritzing.org

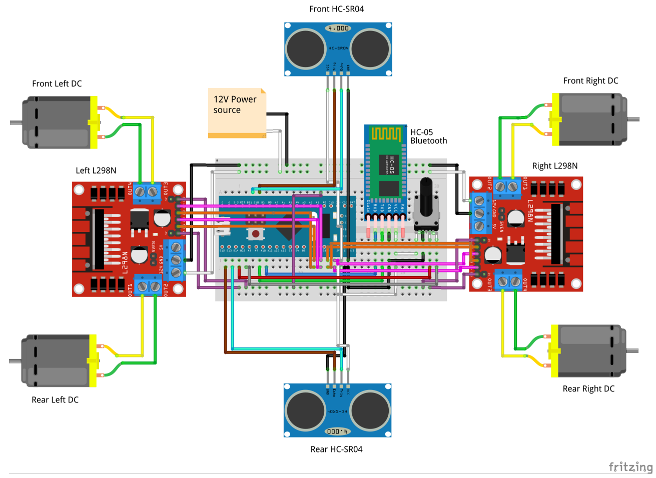

Fig. 6. Circuit Scheme

The full list of connections in the scheme on figure 6 can be seen in the table IV

TABLE IV
COMPONENT CONNECTIONS

Component	Pin	MCU pin
KY-040	CLK	B14/EXTI
KY-040	DT	B15
Left L298N	IN1	B6
Left L298N	IN2	B7
Left L298N	IN3	B6
Left L298N	IN4	B7
Left L298N	EN1	A1/PWM
Left L298N	EN2	A1/PWM
Right L298N	IN1	B8
Right L298N	IN2	B9
Right L298N	IN3	B8
Right L298N	IN4	B9
Right L298N	EN1	A1/PWM
Right L298N	EN2	A1/PWM
HC-05	RX	A9/TX
HC-05	TX	A10/RX
Front HC-SR04	TRIG	B25
Front HC-SR04	ECHO	B26
Rear HC-SR04	TRIG	B27
Rear HC-SR04	ECHO	B28

The fritzing fzz file can be found on the projects Github page[12].

IV. Chip configuration and programming

The toolchain used for configuring and programming the chip consists of STM32CubeMX[13] and the free version of Keil µVision[14].

A. Configuration

STM32CubeMX software provides a simple, yet powerful interface for the configuration of MCUs. After selecting the board, user is able to set each pins purpose.

[12]https://www.github.com/Adi-Sose/Rover
[13]https://www.st.com/en/development-tools/stm32cubemx.html
[14]http://www2.keil.com/mdk5/uvision/

B. Clock

There are four options to be used for the measurement of the system tick: HSI[15], HSE[16], LSI[17], LSE[18]. Even though a HSI has a less accurate tick, it was used in this project due to its faster startup time compared to HSE. The frequency of the clock is set to 8Mhz on APB1 output which is used by both timers and peripheral clocks.

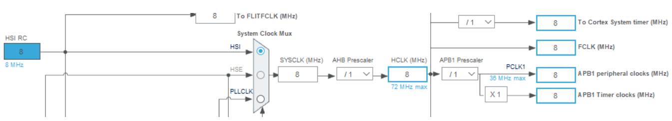

Fig. 7. Clock configuration

C. Timers

The chip offers four timers with four channels each which can be used for system interrupts and pulse width modulation. In this setup TIM1s first channel was used to produce a timed interrupt every second when the process of obstacle detection happens. The second utilized timer TIM2s second channel was used to generate a pwm on the pin A1.

D. Pinout

Fourteen pins were configured of which eleven are used as either GPIO input or output pins, two are used for UART communication and one is a pulse width modulation pin

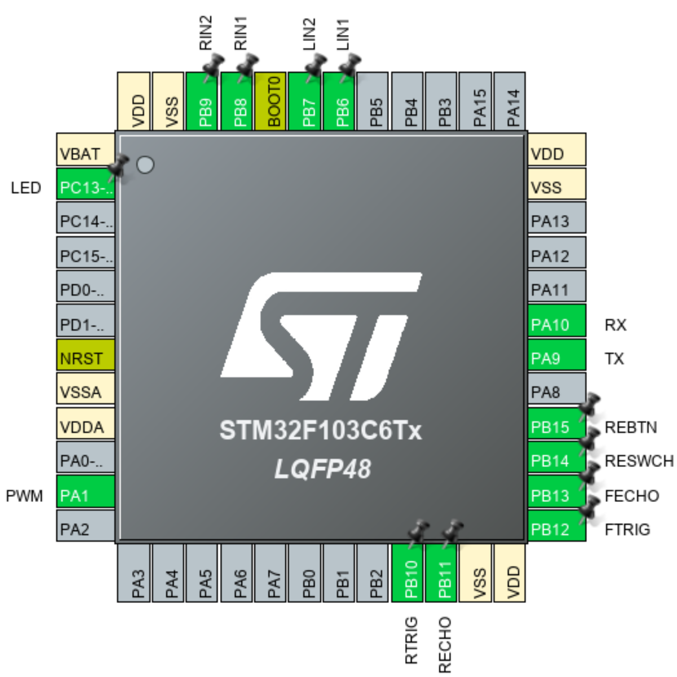

Fig. 8. Pinout

[15]HSI - High Speed Internal
[16]HSE - High Speed External
[17]LSI - Low Speed Internal
[18]LSE - Low Speed Exteral

E. Source code

The source code was written and compiled in Keil µVision. Embedded-C is the most popular choice for programming the HAL[19]. There was a small library of functions and structures created for handling the motors and the movement of the rover as well as setting up and acquiring data from the ultrasonic sensors. The full source code can be found on the projects GitHub page[20]. For the purpose of keeping this paper short the functions purposes will be explained with no code inserted.

- Motor set is a structure which holds the information about the two pins a motor uses.
  - Setup function initializes the object with the passed values.
  - Set direction function sets the pins in such a state that the motors run in chosen direction.
  - Stop function sets the pins states in such a combination that the motor stops moving.
- Motion controller is a project specific structure holding the information about two motorsets - the left and the right set of motors. It also provides information about current rotation or direction of the rover and allows for a blocking mechanism of movement direction.
  - Setup function initializes the object with the passed values.
  - Set direction sets both motor sets in a state which allows the rover to move in a chosen direction.
  - Set rotation sets both motor sets in a state which allows the rover to rotate in a chosen direction.
  - Stop function sends the stopping signal to both motorsets
  - Block direction disables the movement direction for the rover and acts as a locking mechanism to prevent movement in specified direction.
  - Unblock direction allows the rover to move in that direction once again.
- Ultrasonic structure holds the information about the TRIG and ECHO pin of a sensor
  - Setup function initializes the object with the passed values.
  - Get distance function sends a 10ms signal to the TRIG pin and waits for the ECHO pin to send its signal. After this step the program gets the time before and after the ECHO ping was set to its high state and performs a calculation to return the distance of the nearest object to the sensor.

While the rover is in its idle state, the controller is waiting for the input on its UART pin. Based on the signal that was passed by the Android phone with the app installed it determines in which direction the user wants the rover to move in. The movement stop once the user sends a stop signal or when the external interrupt which has the task to measure distance of the rover from obstales sends the blocking

[19]Hardware Abstraction Layer
[20]https://www.github.com/Adi-Sose/Rover

command to the motion controller. The purpose of encoder is to control the speed of the rover, by turning the knob an interrupt is triggered which then updates the duty cycle of pwm based on the direction of the rotation.

V. Simulation

To simulate everything before building the rover the tool Protesu 8[21] was used. This powerful tool allows the user to create any circuit and run a HEX file to controll any microcontroller. Scheme from the figure 6 was designed in the tool in order to test the configuration and programming of the board.

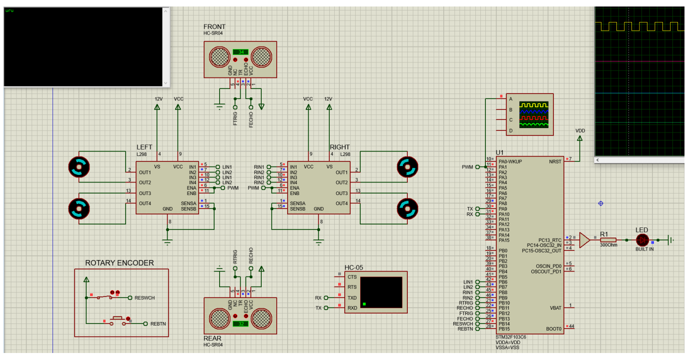

Fig. 9. Proteus simulation

This type of simulation also allows for instruments to be used such as the oscilloscope which is utilized to measuer PWM and a simulation of the bluetooth module using a terminal which supports UART serial communication. Once the Keil IDE compiles the code into HEX file it can be uploaded to the chip and the simulation can be started. As there is no rotary encoder one was simulated by using a switch and a button. Values of the ultrasnoic sensors can be altered in the simulation to simulate the movement of the rover. Motors are the only output from the system and the direction in which they move can be observed while the simulation is running.

VI. Future of the project

Here are a few improvements which are going to be implemented as iterations during further development of the robot.

- Use of a realtime operating system such as FreeRTOS[22] for a better control of tasks and their priorites.
- Implementation of HSE for a higher frequency and thus more precise measurements made by the controller
- Chosing a battery for the power supply
- Switching from the digital to analog input so that the robot can steer while moving in a direction.

[21] https://www.labcenter.com/
[22] https://www.freertos.org/

VII. Conclusion

This type of rover could prove itself quite useful since it has the ability of turning itself in one spot while maintaining the balance of a four wheeled vehicle. As a cheap solution the rover can be used in many applications and the low power consumption only adds up to its usability. Currently, the prototype hasn't been built for further testing but the logic and the configuration have been tested by using the Proteus tool.

References

[1] “STM32F103C8 Datasheet“ [Online] https://www.st.com/en/microcontrollers-microprocessors/stm32f103c8.html
[2] “L298N Datasheet“ [Online] https://www.sparkfun.com/datasheets/Robotics/L298_H_Bridge.pdf
[3] “HC-05 Datasheet“ [Online] https://components101.com/sites/default/files/component_datasheet/HC-05
[4] “HC-SR03 Datasheet“ [Online] https://www.futurel.bg/datasheets/4/SR03.pdf
[5] “KY-040 Datasheet“ [Online] https://www.handsontec.com/dataspecs/module/Rotary%20Encoder.pdf