Logo

Description automatically generated

CECS 347 Spring 2022 Project # 2

An Autonomous Wall Follower Robot Car

By

Abhishek Jasti, Anand Jasti

March 6, 2022

Design an autonomous wall follower robot car with GPIO, edge-triggered interrupts, hardware PWM, power supply circuit, IR sensors, ADC, DC motors, and motor drivers using TM4C123G Launchpad Microcontroller.

**CECS 347 Project 2 Report**

**Introduction**

Building an autonomous robot car than can navigate through any prebuilt tracks with walls on both sides. Using two IR sensor to detect left and right distances to the walls. This detection will allow us to determine the amount of power given to each wheel to keep the robot car on the track. We are using basic hardware components in our embedded systems, while including GPIO, hardware timer, interrupt, analog to digital conversion (ADC), and hardware PWM. Our goal was to be able to use GPIO to interface basic input and output devices, switches, LEDs, motors, sensors, and input/output drivers.

**Operation**

In this beginning we implemented hardware PWM to drive the two DC motors at different speeds. We used two IR sensors on the front of the car to determine the direction of the robot car. To display direction and action taken we used LEDs:

* YELLOW LED:
  + Should display for 2 seconds before the robot car starts to move.
* PURPLE LED:
  + Should display when the robot car reaches end of the track
* RED LED:
  + Should display when the robot car is too close to a wall
* GREEN LED:
  + Should display when the robot car is closer to the left wall
* BLUE LED:
  + Should display when the robot car is closer to the right wall
* WHITE LED:
  + Should display when the robot car is changing in speed cycles

No LED should display when the car is in the middle of the track. For this project we also used the two onboard switch buttons.

SWITCH 1:

* Toggle car on/off and toggle reverse/forward direction.
* Turn wheels off.
* Show yellow LED for 2 seconds.
* Start moving from one end of the track and navigate by itself toward the other end of the track.
* Stop at the end of the track, and show purple LED.

SWITCH 2:

* Toggle speed setting mode on/off.
* Check potentiometer value to set robot speed.

**Link to Demonstration videos:**

**Forward in the middle:** [**https://drive.google.com/file/d/1fAIQaFzEt25InrQ1jtdM3Aa2dBDHU0qu/view?usp=sharing**](https://drive.google.com/file/d/1fAIQaFzEt25InrQ1jtdM3Aa2dBDHU0qu/view?usp=sharing)

**Forward 20cm to the left wall:** [**https://drive.google.com/file/d/10gVMWSqpyYfpY3bGPPP2jZvi5TY2i3Tj/view?usp=sharing**](https://drive.google.com/file/d/10gVMWSqpyYfpY3bGPPP2jZvi5TY2i3Tj/view?usp=sharing)

**Forward 20cm to the right wall:** [**https://drive.google.com/file/d/1-pqo6zZy\_n0lxN7lO8MPlM4ZZM\_nMYVI/view?usp=sharing**](https://drive.google.com/file/d/1-pqo6zZy_n0lxN7lO8MPlM4ZZM_nMYVI/view?usp=sharing)

**Reverse in the middle:** [**https://drive.google.com/file/d/1NH1eZ6dgRBY0rsilYsrmZOckWh4X2Fyd/view?usp=sharing**](https://drive.google.com/file/d/1NH1eZ6dgRBY0rsilYsrmZOckWh4X2Fyd/view?usp=sharing)

**Reverse 20cm to the left wall:** [**https://drive.google.com/file/d/1HjY5L2w5W1tQeFkzIl9BgcMAF3Wiy7SO/view?usp=sharing**](https://drive.google.com/file/d/1HjY5L2w5W1tQeFkzIl9BgcMAF3Wiy7SO/view?usp=sharing)

**Reverse 20cm to the right wall:** [**https://drive.google.com/file/d/1mbyDITCCr6dYre1cbwVV89UjX4r2NHHF/view?usp=sharing**](https://drive.google.com/file/d/1mbyDITCCr6dYre1cbwVV89UjX4r2NHHF/view?usp=sharing)

**Live Demo:** [**https://drive.google.com/file/d/1vTv4ONJz2DTVVRbfhmeUwLhzmh2mgO7M/view?usp=sharing**](https://drive.google.com/file/d/1vTv4ONJz2DTVVRbfhmeUwLhzmh2mgO7M/view?usp=sharing)

**Theory**

This project uses ARM Cortex TM4C123GH6PM Microcontroller, more specifically we used two of the six General-Purpose I/O ports (PF, PA). In port A, we used two pins for direction (PA4, PA5) and two pins for PWM signal (PA6, PA7). In port F we used the onboard buttons pin 0 and pin 4 for toggling on/off the car and toggling on/off the speed mode (more specifically defined in Operation and Hardware design). In this project we used basic hardware components, GPIO pins, edge-triggered interrupts, hardware PWM, power supply circuits, DC motors, motor drivers, IR sensors, and ADC to implement this autonomous wall follower robot car. We used hardware PWM on pins PA6 and PA7 to generate a 1KHz frequency to control the speed of the robot car. We also used GPIO pins PA4 and PA5 to control the direction (forward and backward) of the robot car. We used the two IR sensors to guide our robot car. We also used the perineometer to set the speed of the car (more specifically defined in Software design).

**Hardware design**

Diagram

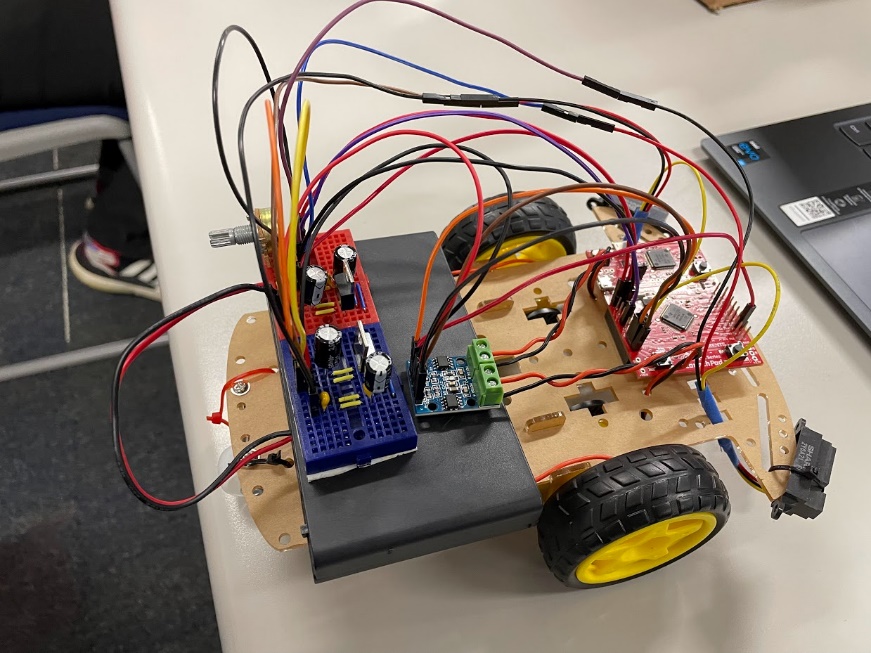
Description automatically generated **Schematic:**

**Outputs: Inputs:**

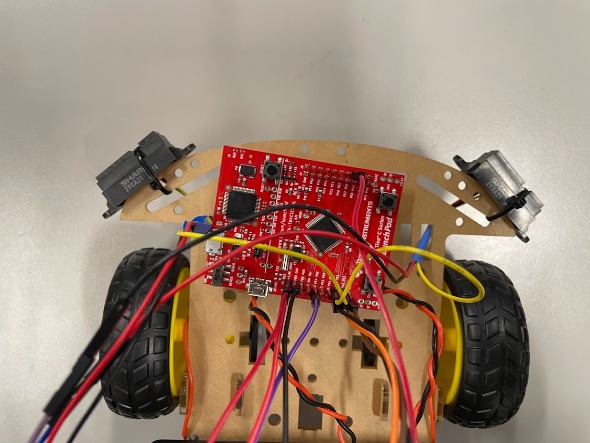
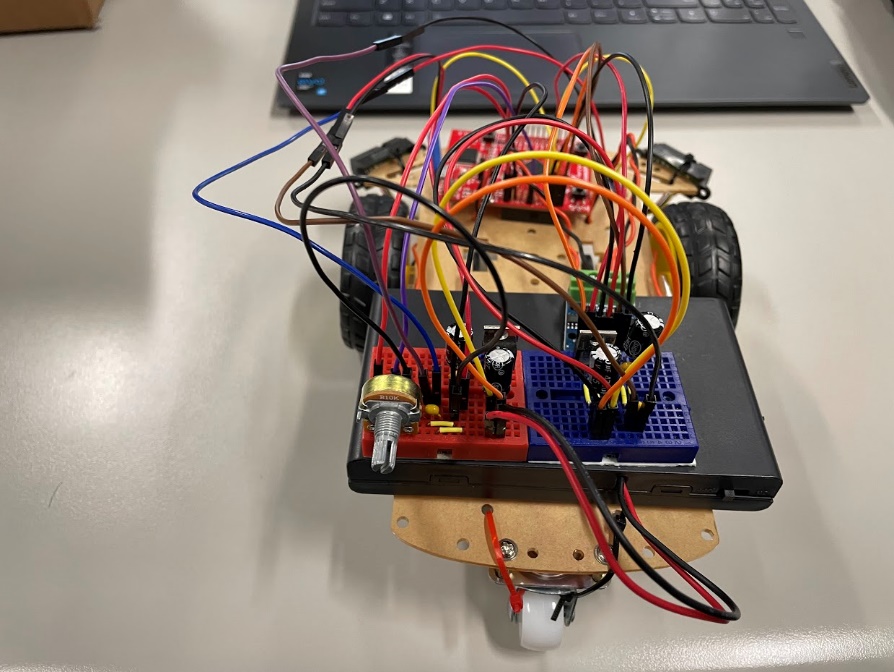
|  |  |
| --- | --- |
| Red LED | PF1 |
| Blue LED | PF2 |
| Green LED | PF3 |
| Direction signal for Left motor (BI-B pin 3 on L9110) | PA4 |
| Direction signal for Right motor (AI-B pin 5 on L9110) | PA5 |
| Speed signal for Left motor  (B-IA pin 4 on L9110) | PA6 |
| Speed signal for Right motor  (A-IA pin 6 on L9110) | PA7 |

|  |  |
| --- | --- |
| Switch 1 | **PF4** |
| Switch 2 | **PF0** |
| Left Sharp IR Sensor | **PE2** |
| Right Sharp IR Sensor | **PE3** |
| Potentiometer for speed control | **PE4** |

The left and right motors are being driven by L9110. The pins connected to the L9110 from our launchpad are listed above. The Red, Green, and Blue LEDs are on-board LEDs provided by the launchpad.

A picture containing electronics

Description automatically generated **Pictures of Hardware System:**

****A picture containing electronics

Description automatically generated

**Software design**

**Text

Description automatically generated** **Software Source Code:**

In this bit of code above we are initializing Port A bit 6 and 7 for Hardware PWM, and we are also providing a function to change the duty cycle of PA6 and PA7. We are using these two pins to control the speed of the left and right DC motors using the L9110 motor driver.

Text

Description automatically generated In this code above we are initializing Port A pin 4 and 5 to be outputs. We are using these two pins to control the direction of the left and right DC motors using the L9110 motor driver.

Graphical user interface, text

Description automatically generated In this bit of code we are initializing SysTick timer for 1ms delay with interrupt enabled and priority 1. We are using the SysTick timer to sample ADC values. The three ADC values we are sampling are left sensor, right sensor, and the potentiometer.

Text, application

Description automatically generatedText

Description automatically generated In this code we are initializing Port F pins 1,2, and 3. We are using these three pins as outputs (which are built-in LEDs). In this project we are using the built-in LEDs to indicate car status. Here is the color information: yellow light: two seconds yellow light before robot start to move; purple light: when reaching end of the track, turn on purple light; red light: the car is too close to a wall and stopped before restart; green light: the car is closer to the left wall; blue light: the car is closer to the right wall; white light: the car is in speed setting mode. No light indicates the car is moving in the middle of the track.

Text

Description automatically generated In this code above we are initializing Port F pins 0 and 4 which are built-in push buttons to be inputs, using falling edge trigger interrupt. The push buttons are used as follows: sw1 will toggle on/off the car, sw2 will toggle speed setting mode on/off.

Text

Description automatically generated In this code we are providing a delay function to be used in Port F Handler for debouncing.

Text, letter

Description automatically generated

In the bit of code we are initializing ADC0 sequencer 2 Ain1(PE2), Ain0(PE3), Ain9(PE4) to be software trigger. We are using these three pins to sample ADC values from the left sensor, right sensor, and the potentiometer. We also provider a function to read all three ADC values and put it into an array to be used in the main function.

**Conclusion**

Implementing this project with the robot car was not as hard as we thought it would be. Building the robot car took the longest time in this project. Creating a power source using a small breadboard was a little tedious trying to fit everything on the breadboard but it was worth the amount of space we saved. We had the most trouble with soldering the wires to the motor, the soldering machine was not the best quality item we bought but we made it work. Fitting everything on the car was not the smartest idea, especially with a huge battery pack that takes eight AA batteries. We decided to use double sided tape to stick everything on the car instead of using zip ties or any other attaching item. Coding of this project for the robot car was not very simple. With the help of our previous labs using PWM and IR sensors helped us tremendously with coding this project for our robot car. Overall, this project took a long time. This project was very helpful to understand and to review the topics of GPIO, edge-triggered interrupts, hardware PWM, power supply circuits, DC Motors, IR sensors, ADC and motor drivers.