Description:

Suppose you have a four-diving wheel robot, you are required to design the system so that

the car moves in a rectangular shape.

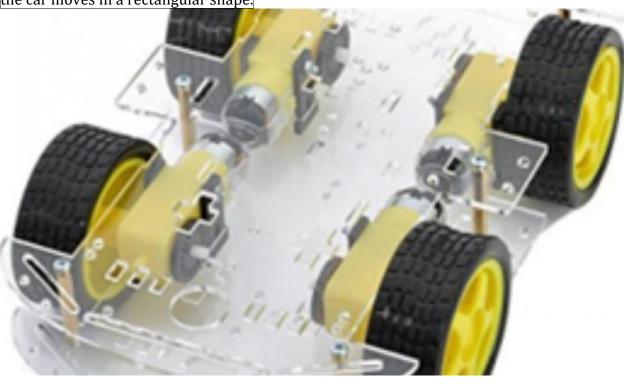


Figure 1: 4WD Car

Detailed Requirements

1. Create a backlog for the team

- 1. Create an Excel sheet named **Team Backlog** that contains the below columns
 - 1. Task Name
 - 2. Assignee
 - 3. Task Status
 - 4. Expected time to finish
 - 5. Actual time to finish

2. Read System Requirement Specifications

- 1. Car Components:
 - 1. **Four** motors (**M1**, **M2**, **M3**, **M4**)
 - 2. **One** button to start **(PB1)**
 - 3. **One** button for stop **(PB2)**
 - 4. Four LEDs (LED1, LED2, LED3, LED4)
- 2. System Requirements:

- 1. The car **starts initially** from **0 speed**
- 2. When **PB1** is **pressed**, the car will **move forward after 1 second**
- 3. The car will move forward to **create the longest side of the rectangle for 3** seconds with 50% of its maximum speed
- 4. After finishing the first longest side the car will **stop for 0.5 seconds**, **rotate 90 degrees to the right**, and **stop for 0.5 second**
- 5. The car will move to **create the short side** of the rectangle at **30% of its speed for 2 seconds**
- 6. After finishing the shortest side, the car will stop for **0.5 seconds**, **rotate 90 degrees to the right**, and **stop for 0.5 second**
- 7. Steps **3 to 6** will be **repeated infinitely** until you press the **stop button (PB2)**
- 8. **PB2** acts as a **sudden break**, and it has the highest priority
- 9. **LEDs Operations**
 - 1. **LED1:** On means moving forward on the long side
 - 2. **LED2**: On means moving forward on the short side
 - 3. **LED3**: On means stop
 - 4. **LED4:** On means Rotating

3. Prepare your design

- 1. Please note that any functionality based on timers should be separated in a separate module, and all timers should be operating in **Normal mode**
- 2. Create a PDF file with the name **Moving Car Design**
- 3. The design document should contain the below fields
 - 1. Cover Page
 - 2. Table of content
 - 3. Project Introduction
 - 4. High-Level Design
 - 1. Layered architecture
 - 2. Modules Descriptions
 - 3. Drivers' documentation
 - 5. Low-Level Design
 - 1. Provide the flowchart for each function in each module

4. Preparing development environment

- 1. Create layers' folders
 - 1. Create a folder for each layer
 - 2. All folders should be in **upper case**
 - 3. Ex: **MCAL**, **HAL**, **APP**, ... etc
- 2. Create diver's folders and files
 - 1. Create a folder for each driver
 - 1. Each folder contains **only one** .c file and **at least one** .h file
 - 2. All files' names should be in lowercase
 - 2. All driver folders' names should be in **lowercase**
 - 3. Ex: **dio**, **timer**, **pwm**, ... etc.
- 3. Add header file guard
 - 1. All header files must include the header file guard

5. Drivers' implementation and code convention

- 1. All drivers provided in the design document should be implemented
- 2. All drivers should be tested against different test cases
- 3. Functions' descriptions should be included
- 4. Don't use magic numbers, use Macros or Enums instead
- 5. Follow a proper indentation in your code
- 6. Use a meaningful name for your variables
- 7. Follow the below naming for the functions
 - 1. MODULENAME functionName
- 8. Follow this convention for naming variables
 - 1. typeIndicator_scopeIndicator_variableName
 - 2. typeIndicators (u8, u16, u32, i8, i16, st (struct), en (enum), arr (array), .. etc)
 - 3. scopeIndicators (g (global), gs (global static), a (argument))

6. Implement and integrate the main application

1. Implement the main application that fulfills the system requirements

7. **Test your application**

- 1. Create an Excel sheet named **Test Protocol**
- 2. The sheet should contain the below columns
 - 1. Test Case ID
 - 2. Test Case Description
 - 3. Test Case steps
 - 4. Expected Result
 - 5. Actual Result
 - 6. Pass/Fail
- 3. Fill in the sheet with the test cases you will execute
- 4. Execute the test cases on the **simulator (Mandatory)**
- 5. Execute the test cases on **hardware (Optional)**

Delivery

- 1. Deliver the Team Backlog sheet
- 2. Deliver the Design Document
- 3. Deliver all project files and folders including the .hex file
- 4. All code conventions must be followed
- 5. English Video recording presenting all of your work as a team
 - 1. The video should be 15 minutes maximum
 - 2. Each team member should present himself and discuss his role and what he delivered through the backlog and what test strategy he/she made to test his/her work
 - 3. Application testing should be presented by the team coordinator starting from the Test protocol sheet to the simulator and/or the hardware
 - 4. Any limitations or failed test cases should be communicated in the video