Sprints

Description:

Suppose you have a four-diving wheel car, you are required to design the system so that the car avoid any object in front.

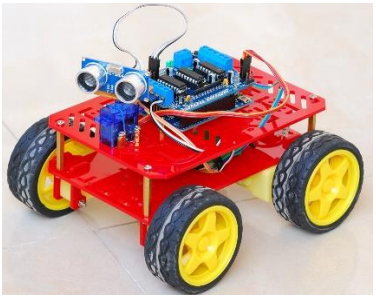

Figure 1: Object Detection Robot

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 change default direction of rotation (**PBUTTON0**)
        3. Keypad button 1 to start
        4. Keypad button 2 to stop
        5. **One Ultrasonic** sensor connected as follows
            1. **Vcc to 5V in the Board**
            2. **GND to the ground In the Board**
            3. **Trig to PB3**
            4. **Echo to PB2**
        6. **Ultrasonic** sensor
        7. **LCD**
    2. **System Requirements**:
        1. The car **starts initially** from **0 speed**
        2. The default rotation direction is to the **right**
        3. Press **PB2** to start or stop the robot
        4. **After Pressing Start**:
            1. The LCD will display a centered message in line 1 **“Set Def. Rot.”**
            2. The LCD will display the selected option in line 2 **“Right”**
            3. The robot will **wait** for **5 seconds** to choose between **Right and Left**

1. When **PB1** is pressed **once**, the default rotation will be **Left** and the **LCD line 2 will be updated**
2. When **PB1** is pressed **again**, the default rotation will be **Right** and the **LCD line 2 will be updated**
3. *For each press the default rotation will changed and the LCD line 2 is updated*
4. **After the 5 seconds the default value of rotation is set**

4. The robot will move **after 2 seconds** from setting the default direction of rotation.

5. **For No obstacles or object is far than 70 centimeters:**
    1. The robot will move forward with 30% speed for 5 seconds
    2. After 5 seconds it will move with 50% speed as long as there was no object or objects are located at more than 70 centimeters distance
    3. The LCD will display the speed and moving direction in line 1: **"Speed:00% Dir: F/B/R/S"**, **F**: forward, **B**: Backwards, **R**: Rotating, and **S**: Stopped
    4. The LCD will display Object distance in line 2 **"Dist.: 000 Cm"**
6. **For Obstacles located between 30 and 70 centimeters**
    1. The robot will decrease its speed to 30%
    2. LCD data is updated
7. **For Obstacles located between 20 and 30 centimeters**
    1. The robot will stop and rotates 90 degrees to right/left according to the chosen configuration
    2. The LCD data is updated
8. **For Obstacles located less than 20 centimeters**
    1. The robot will **stop**, move **backwards** with **30% speed** until distance is **greater than 20 and less than 30**
    2. The LCD data is updated
    3. **Then preform point 8**
9. **Obstacles surrounding the robot (Bonus)**
    1. If the robot **rotated for 360 degrees without finding any distance greater than 20 it will stop**
    2. LCD data will be updated.
    3. The robot will frequently (each 3 seconds) check if any of the obstacles was removed or not and move in the direction of the furthest object

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, ICU** will be software implemented
2. Create a PDF file with the name **Obstacle Avoidance Robot V1.0 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

    2. Pre-compiling configurations for each module
    3. Linking configurations for each module

4. **Preparing development environment**
    1. Create layer's 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 **lower case**
        2. All driver folders names should be in **lower case**
        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. Function's 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 fulfil 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 **simulator** (**Mandatory**)
    5. Execute the test cases on **hardware** (**Optional**)

Sprints

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 did he delivered through the backlog and what test strategy did 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 simulator and/or the hardware
    4. Any limitation or failed test cases should be communicated in the video