Version 1.2

CMPE 443 PRINCIPLES OF EMBEDDED SYSTEMS DESIGN

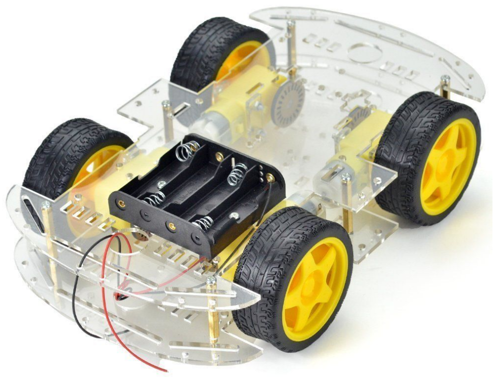
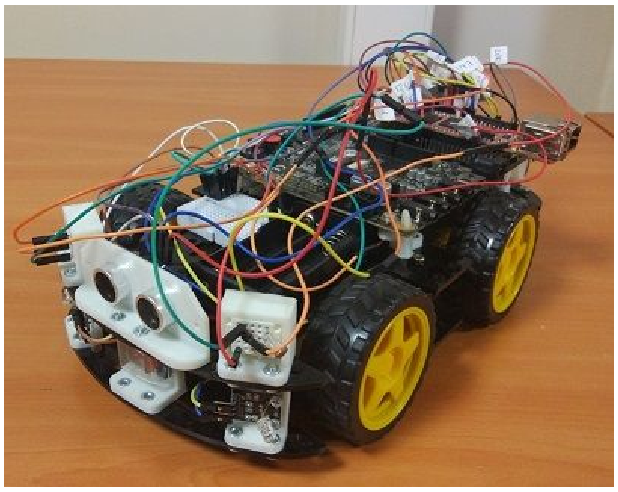
Final Project

“Wall Follower Car”

Deadline: 4th of January, 2020

1) Problem Description

In this final project, you will build a toy car that will follow along a wall. For the car, you will use 1 Motor Controller, 2 LDRs, 4 LEDs, 1 potentiometer and 1 Ultrasonic sensor. The car will have two separate operating modes. One of them is Test mode which will be used to test the car using serial communication. The other one is Autonomous mode which we will discuss its details below.

2) Requirements

The demo will consist of two scenarios with different requirements. The device, when initialized, must start at TEST MODE. There are some requirements which apply to both of these scenarios. Other mode specific requirements will be grouped accordingly below.

GLOBAL requirements	**Points(42)**
When the device stops, all LEDs shall be turned off.	2
While the device is moving forward, only the LEDs at the front shall be turned on.	2
While the device is moving backward, only the LEDs at the back shall be turned on.	2
While the device is turning left (counter-clockwise direction), the LEDs at the left shall blink twice a second. Other LEDs must stay turned off.	3
While the device is turning right (clockwise direction), the LEDs at the right shall blink twice a second. Other LEDs must stay turned off.	3
Whenever the ASCII text “STATUS” followed by a carriage return and a line feed is sent to the device via serial communication, the device should return a status information. (Check out the *Status Information* section below)	10
An external potentiometer must be used to control the velocity of the car. The potentiometer must allow us to set the speed from 0% to 100%, where 0% being literally no movement and 100% being the maximum speed.	5
Points for code:	**15**
The code is a mess. Nobody would want to read/reuse this code. It has no structure and no thought process was put into the work. Miraculously though, it works.	0
The code is structured fine, but it is still hard to read. It has some inefficiencies algorithmically and some standards we would use for embedded systems were not followed (ie. using int instead of int32_t, using global variable for an immediate and constant value instead of a #define macro).	5
The code is well structured, but has some errors here and there. The code would perform well only for the given task. The code is definitely not reusable for other tasks.	10
The code is well structured and every component does whatever it is supposed to do. It promises high readability and reusability.	15

TEST MODE requirements	**Points(38)**
When entered, the device should send the ASCII text “TESTING” followed by a carriage return and a line feed via serial communication.	5
Whenever the ASCII text “LEFT” followed by a carriage return and a line feed is sent to the device via serial communication, the device must turn counter-clockwise 90 degrees in its place and then stop.	4
Whenever the ASCII text “RIGHT” followed by a carriage return and a line feed is sent to the device via serial communication, the device must turn clockwise 90 degrees in its place and then stop.	4
Whenever the ASCII text “FORWARD” followed by a carriage return and a line feed is sent to the device via serial communication, the device should start moving forward.	4
Whenever the ASCII text “BACK” followed by a carriage return and a line feed is sent to the device via serial communication, the device should start moving backward.	4
Whenever the ASCII text “STOP” followed by a carriage return and a line feed is sent to the device via serial communication, the device should stop moving, unless it is performing a 90 degree turn.	4
Whenever a command word followed by a carriage return and a line feed is sent, the device should echo the sent command back via the serial communication followed by a carriage return and a line feed.	5
Whenever LDRs detect a light source (a light source (more than **300 lumens**) which is much brighter than the room light), the car must stop. When the light source is gone, the device must continue doing whatever it was doing before.	3
Whenever the ASCII text “AUTO” followed by a carriage return and a line feed is sent to the device via serial communication, the device should switch to AUTONOMOUS MODE. The requirements for the AUTONOMOUS MODE is below.	2

To test the AUTONOMOUS MODE, we will put 1 LED strip at the end of the road and we will put 1 wall path to the left side of the car. You car should follow the wall via Ultrasonic sensor and when it reached the LED stript, it should stop.

AUTONOMOUS MODE requirements	**Points(35)**
When switched to AUTONOMOUS MODE, the device should send the ASCII text “AUTONOMOUS” followed by a carriage return and a line feed via serial communication.	5
When switched to AUTONOMOUS MODE, the device must wait until the ASCII text “START” followed by a carriage return and a line feed is sent through serial communication.	5
While moving and next to a wall, the device should move along the wall, within 15 to 35 centimeters from the wall. The exact measurements may vary on the demo day, but what we want with this requirement is that we want the car to stay at a reasonable distance from the wall. It is worth to mention that the wall will never have a curve that will be impossible to follow along for the configuration of your cars. (For example there will be no sharp 90 degree turns)	15
Once the device faces the LED strip at the end of the road while moving, it must stop and send the ASCII text “FINISH” followed by a carriage return and a line feed through serial communication.	1
Whenever the ASCII text “TEST” followed by a carriage return and a line feed is sent to the device via serial communication, the device should switch to TEST MODE. The requirements for the TEST MODE is above.	2

For the AUTONOMOUS MODE, we allow groups to bring their own batteries. In the lab, we use Samsung ICR18650-26JM. If you are to bring your own battery/battery pack for the demo, the battery must be an 18650 type battery and the capacity must not exceed 2600 mAh.

We thought that some people might not be able to finish all the requirements, especially the Bluetooth ones. Although almost everything is done via bluetooth, we thought we should provide a way for people who have not completely finished the communication, but was able to get the car moving. Below are the conditions about incomplete communication requirements.

Communication status	**Points**
Neither bluetooth, nor UART communication was not implemented. The device does move but has no interaction.	-40
Neither bluetooth, nor UART communication was not implemented. The device does move but has limited interaction. (ie. it does stop when it sees a light source)	-30

Neither bluetooth, nor UART communication was implemented. Though, the device interacts with a numerical keypad (will be provided by us if requested).	-20
Bluetooth was not implemented, but the device receives commands via UART. Some commands are not working.	-15
Bluetooth was not implemented, but the device receives commands via UART and all commands work fine/perfectly.	-10
Bluetooth was implemented, but some commands are not working. (Considered qualified for the tournament as long as the race commands work fine)	-5
Bluetooth was implemented, and all commands work fine/perfectly. (Considered qualified for the tournament)	0

3) Status Information

Status information is a JSON object. It must be sent by the device whenever the STATUS command is sent to the device.
CLARIFICATION: The status information must be sent after the command string STATUS is echoed back through the serial communication.
Below is the structure of the status information object.

- distance: <integer> (in cm)
  - The distance from the wall, as measured by the Ultrasonic Sensor.
- light_level_left: <integer> (between 0-1023)
  - The light level as measured by the LDR on the left.
- light_level_right: <integer> (between 0-1023)
  - The light level as measured by the LDR on the right.
- op_mode: <string>(AUTO|TEST)
  - The current operating mode of the device.

Here is an example status information object:

```
{"distance":5,"light_level_left":150,"light_level_right":200,"op_
mode":"AUTO"}
```

After sending the status information object, send a carriage return followed by a line feed through the serial communication. Do not use this character sequence inside the status information string. Note that JSON standards use double quotes (ASCII=34=0x22), no single quotes, so be careful while preparing your JSON string.

4) Serial Communication Example

Here is an example serial communication sequence that could happen between the car and some controller.

```
    Car  > TESTING\r\n
    Ctrl > STATUS\r\n
    Car  >
STATUS\r\n{"distance":10,"light_level_left":150,"light_level_right":2
00,"op_mode":"TEST"}\r\n
    Ctrl > LEFT\r\n
    Car  > LEFT\r\n
    Ctrl > FORWARD\r\n
    Car  > FORWARD\r\n
    Ctrl > AUTO\r\n
    Car  > AUTO\r\nAUTONOMOUS\r\n
    Ctrl > START\r\n
    Car  > START\r\n
    Car  > FINISH\r\n
    Ctrl > STATUS\r\n
    Car  >
STATUS\r\n{"distance":20,"light_level_left":300,"light_level_right":4
00,"op_mode":"AUTO"}\r\n
    Ctrl > TEST\r\n
    Car  > TEST\r\nTESTING\r\n
```

5) Implementation

You will have access to HWLAB to implement your projects on certain times that will be announced later. For times you don't have access to the lab, you may want to implement your high level logic in a test driven manner. Here's an example of how you may want to achieve this: https://github.com/karacasoft/example-unit-testing-with-c

It is highly suggested to have a plan with backups. For example, you might want to get the bluetooth communication working first as it is used in every stage of the project. But when it starts taking too much time, you may want to fallback to UART communication and start working on motors for the car to move, because a moving car with UART communication gives you much higher points than a car with bluetooth communication that doesn't move.

6) Race Tournament

Any group that have completed all the requirements will be allowed to enter the tournament. The race will take place at HWLAB.

There will be two identical tracks that have a wall on the right of it and marks on the left of it. The cars are expected to move along the wall as done in the AUTONOMOUS MODE. At the end of the tracks, there will be an LED strip that marks the end of the road.

The race is split into runs, The cars will be put on the markers at the beginning of the tracks and the timer will start automatically as soon as the bluetooth commands are sent to devices. The first car that reaches the end of the track while following the rules and sends the finish command through bluetooth will be the winner of that run. For each run, the tracks will be switched. The format of the tournament will be determined during the demo day, depending on the number of groups that are eligible to enter the tournament.

The rules of a run are as follows:

- The car must not get too far away from the wall on the right, the markers on the left will be used to determine if the car is outside the boundaries. The car that goes outside the boundaries will be disqualified for that run. If both cars are disqualified, the one that has gone furthest along the wall will be the winner of that run. Measurements are done from the start marker to the back of the car. The decision will be finalized by the referee and no objections will be accepted.
- The car must not interfere with the other car before and during the race, either by physical means or by interfering with the bluetooth communication or by any other means. When detected, the group will be disqualified for the race and the remaining runs will not be played.
- The car must stop whenever it sees the LED strip at the end of the road. It may stay 1-2 centimeters behind the LED strip, but must not try to go over the LED strip. Doing this will result in a penalty of 30 seconds added to the race time. Groups may object to this decision (blaming the lighting conditions etc.) and if the objection is accepted by the referee, the environment will be fixed as the group requests and the run is repeated. This can happen only once in a race. Further objections may result in disqualification.

Awards of the race tournament:

➢ Winner: Each member will receive +10 pts to her/his cumulative average.

- ➢ 1st Runner-up: Each member will receive +5 pts to her/his cumulative average.
- ➢ 2nd Runner-up: Each member will receive +3 pts to her/his cumulative average.

  Important Notes about the Awards:

  1. Awards are designed for only one team per rank.

  2. If there are two or more teams at a rank, the award will be divided by the number of teams. Example 1: Consider the case that teams A and B appear at the first place, team C appears at the second place. Then award for A and B will be round(10/2)=5. This award will shift A and B to the second place, while leaving the first place empty, pushing C to the third place. C's award will be 3.

  Example 2: Consider the case that teams A appears at the first place, teams B and C appear at the second place. Then award for A will be 10. C and D will get round(5/2)=3 pts.

  Example 3: There are 3 teams at the first place. Then round(10/3)=3 will be the award of each team. So they will be shifted to third place.

  Example 4: There are 4 teams that cannot beat each other. Then no awards will be granted.

  3. All of the team members have to be present during the races so that they will get their awards.

7) Group Report

Your report must include these sections:

- Title
- Summary (some information about the problem and the solution provided)
- Block diagram(s)
- A flowchart of the system
- Pin connection tables
- Circuit Schematic(s) (if custom circuits were used)
- Short explanation of how your system was intended to be used, think about third parties that might want to try to use your code
- Detailed description of functional parts of your project. You are encouraged to provide all the technical details of your system here. You may want to use pseudo-codes, sequence diagrams, etc.
- Expense List
- References

Your report has to be genuine. You will deliver the hard copy of the report with PDF and Tex files.

The report will be graded over a 100 and will have a multiplicative effect on the final grade. For example for some group who have the grade ***X*** for the implementation part, and grade ***Y*** for the report, final grade of this group will be calculated as follows:

Grade = ((***X***/115) * 100) * (***Y***/100)

This basically means that the report is pretty important, so be mindful of that while building your projects.

Some Important Notes

- The total cost of the components that you buy for this experiment should not exceed the 100 TL.
- We will not allow coding during Demo. You will be given ten minutes to prepare the car for the demo. Hence, we suggest you to finish all kinds of settings before the demo day.
- Do not steal someone else's work. Provide your sources in your report if you are to use someone else's code and verify that the code is licensed in a way that you can use in your project.
- In case of a tie, both groups will be considered that they got the lower rank instead of the higher rank, (for example, there will not be any two groups that get the first place prize. But there can be two groups that both get the second place prize. It is the same for a tie between three groups, they all will be considered third place.)
- Total points reported in this document is 115. Your project grade will be scaled down to a maximum of 100 points. For example, a group who acquired 115 points will be graded 100, whereas a group with 23 points will get 20 points from the project out of a 100 points.

Version History

Version: 1.2:

- Added the potentiometer requirement, which was forgotten to be mentioned.
- Added the demo day.
- Updated the grading formulas to reflect the new total grade after adding the potentiometer requirement.

Version: 1.1:

- Added grading criteria for report
- Added some clarifications to the "follow along the wall" requirement.

Version: 1.0:

- Initial Release