Embedded Systems Project

Team 3

LKA-CI-SS

Members:

- Abdallah Mohamed Adel
  - T-09 49-14632 Abdallah.nada@student.guc.edu.eg
- Mohamed Kamal Gamil
  - T-16 46-7203 Mohamed.monir@student.guc.edu.eg
- Seif Ahmed Elberkawy
  - T-09 46-1953 Seif.elberkawy@student.guc.edu.eg
- Ahmad Abdallah
  - T-10 46-10950 Ahmed.mohamedhussin@student.guc.edu.eg

Introduction:

Our project is simply an implementation of an autonomous car with some extra features that are found in some of the cars that exist now such as: Mercedes, Tesla, etc…

These features are:

1. “L.K.A” Lane keeping assist which is the behavior of the car to keep moving in the same lane and whenever it starts drifting away or the street has a turn it can move to stay in the lane and not leave it.
2. “C.I” Control indicators
    a. A way to tell the passenger which gear the car is on right now through a seven-segment-display and he can control these gears through a joystick.
    b. Another feature is the adaptive headlights which is the way headlights adapt to the outer world’s light. In other words, the car’s headlights would be off in the morning and start to lighten up as the sunsets. It also depends on the street’s lights at night whether to give strong light in dark places or just medium one in streets that have light.
3. “S.S” Sound system. It is simply an mp3 system that gives the passenger the option to choose to play songs saved on an SD card or pause them through a touch screen. He can also choose to play the next song or the previous

one and the desired music will be played through an external speaker found in the car.

Our approach to implement such a project started from easy to complex to easy to complex again. This means that we started by implementing the gearbox with the Seven-Segment-Display and the joystick. Then when it worked perfectly we started to level-up with something harder such as displaying the buttons on the touchscreen which will allow the user to choose to play or pause a song. After that we integrated the touchscreen with the mp3 module, the external speaker and the SD card. When all these things were done we took a break with something easy which is the adaptive headlights. This one did not take time and was finished in less than an hour. When all these things were working we started to make the car move and started implementing the lane keep assists which was the hard part since we kept trying over and over to make the car turn right or left to keep following on the lane. Finally we did an “I2C” which is the bus to make two Arduino boards communicate together. Why did we do it? Well, because the touchscreen used all of the pins of an Arduino UNO board and the rest of the project used pins from an Arduino MEGA board, so we needed to make the touchscreen with the mp3 to work with the whole car not alone by themselves. And that’s it we had our autonomous car that has extra features all working together.

Components used:

1. Arduino Mega to support all pins needed and have the software code uploaded to it.
2. Arduino UNO to support the touchscreen since it needed too many pins and to have the software code of the touchscreen uploaded to it.
3. Seven Segment Display to show which gear the car is on “P, R, N, D”.
4. Joystick to choose which gear needed.
5. Light sensor to detect the light intensity in the surroundings.
6. LED to show how the adaptive headlights work with the light sensor.
7. Touchscreen that allows a passenger to choose to play or pause music or to play the next or previous song and also shows him which song is currently playing.
8. MP3 Module that plays or pauses the song based on the passenger chosen option from the touchscreen.
9. SD Card that contains songs that the MP3 plays.
10. External speaker where the audio of the songs come out from.
11. 4 DC Motors, one for each wheel of the car to give it power to move on a specified speed.
12. 2 H-Bridges, one for the two frontal DC Motors and one for the back ones, are used to connect the motors with the Arduino Mega board so they can be programmed to move forward or backward and to control their power generated to control the speed of the car.

13.2 Infrared sensors that detect colors (most noticeable are black and white). It should always detect black which is the lane color and once detected white or something not black it should return a signal for the motors to change their speed to turn the car back to the lane. One sensor on each side to detect whether to go left or right.

14. Black Tape which is the lane that the car moves on.

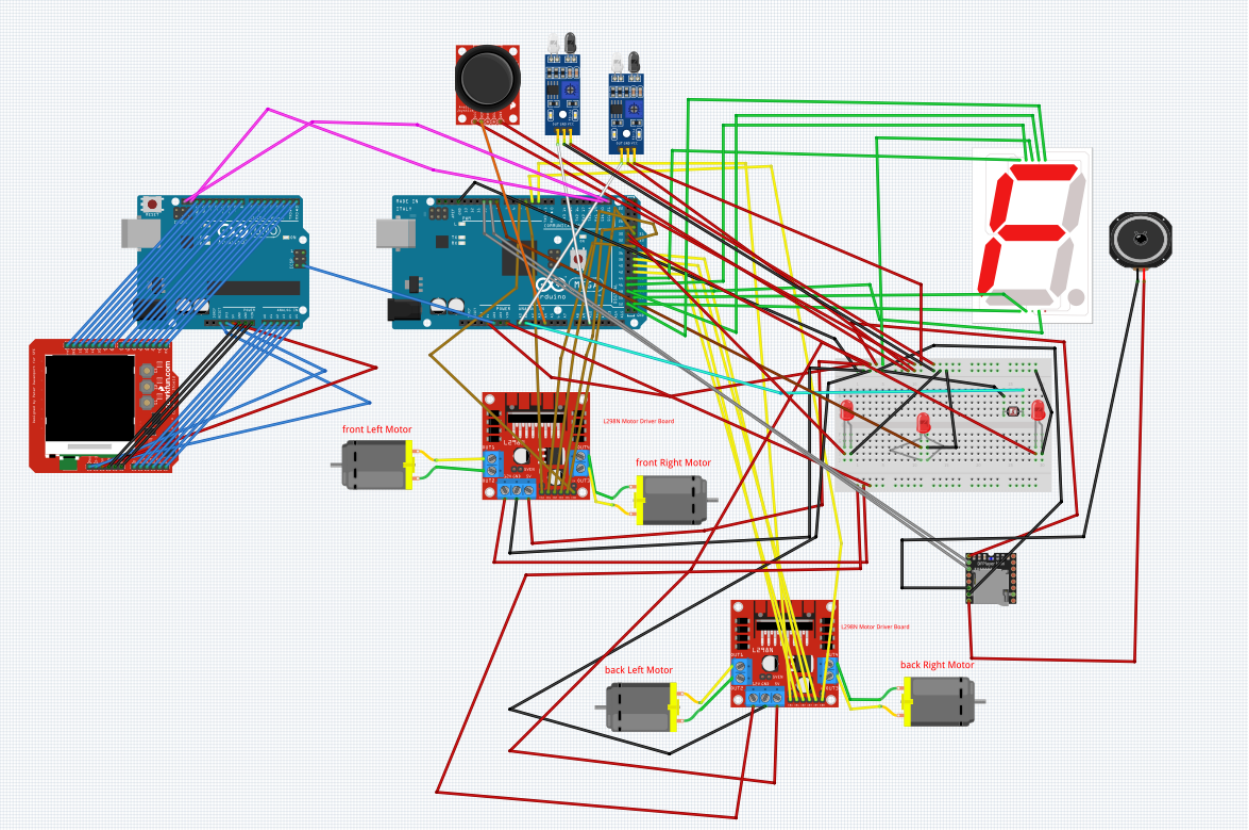
15. Male to male wires, female to male wires and female to female wires.

16. Breadboard to place some sensors and wires on it.

17. 12-volt adapter to power the boards and give needed power to the motors.

18. Double-sided-tape to stick the components on the body of the car.

Hardware design (Fritzing):



Libraries used:

- Arduino_FreeRTOS, to enable freeRTOS implementation.
- DFRobotDFPlayerMini, to play mp3 files.
- SoftwareSerial and Wire, to enable communication between the two Arduino boards.
- TouchScreen, to use the touchscreen.
- SD, to read data from an SD card.
- Adafruit_GFX and MCUFRIEND_kbv, to draw on the touchscreen.

Inputs and Outputs:

This part is not hard at all. You just want the input to be something that will read a specific thing and the output is something that you are waiting to see or detect.

A detailed explanation can be as follows:

- The sensors such as the infrared and light sensors took an input which is the voltage needed to power them and the ground. Then they have an output for what they detect which is connected as an input to the thing that a human can understand.

- IR sensors have an input voltage and ground and output a signal connected to the Arduino then the Arduino will output to the motors whether to stay on the same speed or change it in order to turn right or left so the final output is that I will see whether the car will turn or continue moving.
- The motors obviously take an input from the H-Bridge depending on the output of the IR sensors and give an output speed/direction to the wheels.
- The light sensor takes an input voltage and ground to work and outputs to the Arduino what light intensity it reads.
- The LED takes an input from the Arduino as its voltage which depends on the output of the light sensor and takes a ground too to work.
- For the joystick it takes input voltage and ground and outputs a signal dictating it's direction to the Arduino.
- The Seven Segment Display has 7 inputs for the 7 bits and a ground since it is a cathode and depending on the signal of the joystick it outputs a letter from (P, R, N, D).
- The touchscreen is connected to all pins on the Arduino UNO most of them are input to allow the passenger to choose from it and outputs a signal -that is received by the Arduino Mega- with the chosen button to the MP3 module.
- The MP3 module takes input voltage and ground and the output from the touchscreen then outputs a signal to the speaker.
- The speaker has inputs voltage, ground and the output of the MP3 then outputs sound or no sound depending on what it received.

FreeRTOS:

We tried to keep the code atomic as much as possible by separating functions from each other and calling them correspondingly when needed.

This separation helped us to translate these functions into tasks immediately that run in a semi-parallel way. These tasks are:

1. Lane Keep Assistance, to govern the movement of the car and ensure that it is always keeping it's lane.
2. Adaptive Headlights, to control the headlights depending on the surrounding light intensity.
3. Gearbox, to control the gearbox using the joystick.

All of these tasks have the same priority, however, they have different periods. The LKA task is the most frequent one (with 10 ms period) as the car needs to keep polling for any updates from the IR sensors. Then comes the gearbox with the second highest frequency (250 ms period) to keep listening for any change in the joystick. Finally the adaptive headlights with the lowest frequency (1000 ms), as normally the current light source won't change a lot.

Regarding the touchscreen, all of its logic is present on the UNO board alone, hence it doesn't need to be scheduled. Whenever an input is given on the

touchscreen, the UNO sends a signal to the MEGA board, the MEGA board then receives the signal and accordingly calls the MP3 functionality (e.g: playing, pausing or playing the next song). That's why we didn't schedule the MP3 along with the other tasks on the MEGA as it is notified of any event happening thanks to the I2C connection.

Problems and Limitations faced:

We faced a main problem which was the power supply. The batteries would finish their charge very quickly and they did not provide the 12 volts needed for the H-bridges to work so the car was not moving as fast as needed. We overcame this problem by using a 12 volt adapter but that gave us a limitation that the car cannot traverse a long distance due to the length of the wire of the adapter. Another problem was the car's body. The components, nails and bolts were so small that sometimes they would fall since they were not intact due to not finding a suitable screwdriver to fix them. The last limitation was that the touchscreen size is too big that it took all the pins of the Arduino UNO, hence we could not add extra components such as a Bluetooth module to control the screen or the joystick for the gearbox from a distance.

Work distribution:

1. Seif and Mohamed: Worked together on the gearbox hardware and software.
2. Abdallah and Ahmad: Worked together on the software logic of the motors and connected the motors with the H-bridge.
3. Abdallah and Mohamed: Worked together on the touchscreen with the MP3 and speaker hardware and software.
4. Seif and Ahmad: Worked together on the software logic of the LKA and the hardware of the IR sensors.
5. Seif, Abdallah and Mohamed: Worked together on Adaptive headlight software logic and hardware.
6. We all worked together on the Fritzing diagram so that each one can be aware of the hardware connections that he did not work on.
7. We all worked together to implement the freeRTOS code to think together of the priorities and each one can have a glance of the software code he did not work on.

We would like to thank Dr. Hassan Soubra for this great project that literally made us apply what we learn in the course, and a huge thanks for Manar and Aya that always answered us and helped us finish this amazing project.