Open your mind. LUT.
Lappeenranta University of Technology

EXERCISE WORK 1

Group 11
Jan Librowski (student number 0549289)
Pavel Talchikov (student number 0549367)

Bachelor's thesis
LUT School of Energy Systems
Electrical Engineering

20 March 2019

1. INTRODUCTION

The aim of the work is to create a state machine model of car using Arduino kit with all kinds of details except motion sensor and the keypad module.

2. IMPLEMENTATION

2.1 Planning

It was decided to use Arduino MEGA instead of UNO because it has more pins for connection. The plan is to require minimum criteria and to add an extra feature – headlight turns on when the car starts working and stop signal turns on when the car brakes. Moreover, emergency lights can also be turned on.

Our work requires the following elements(except Arduino MEGA and breadboard with wires):

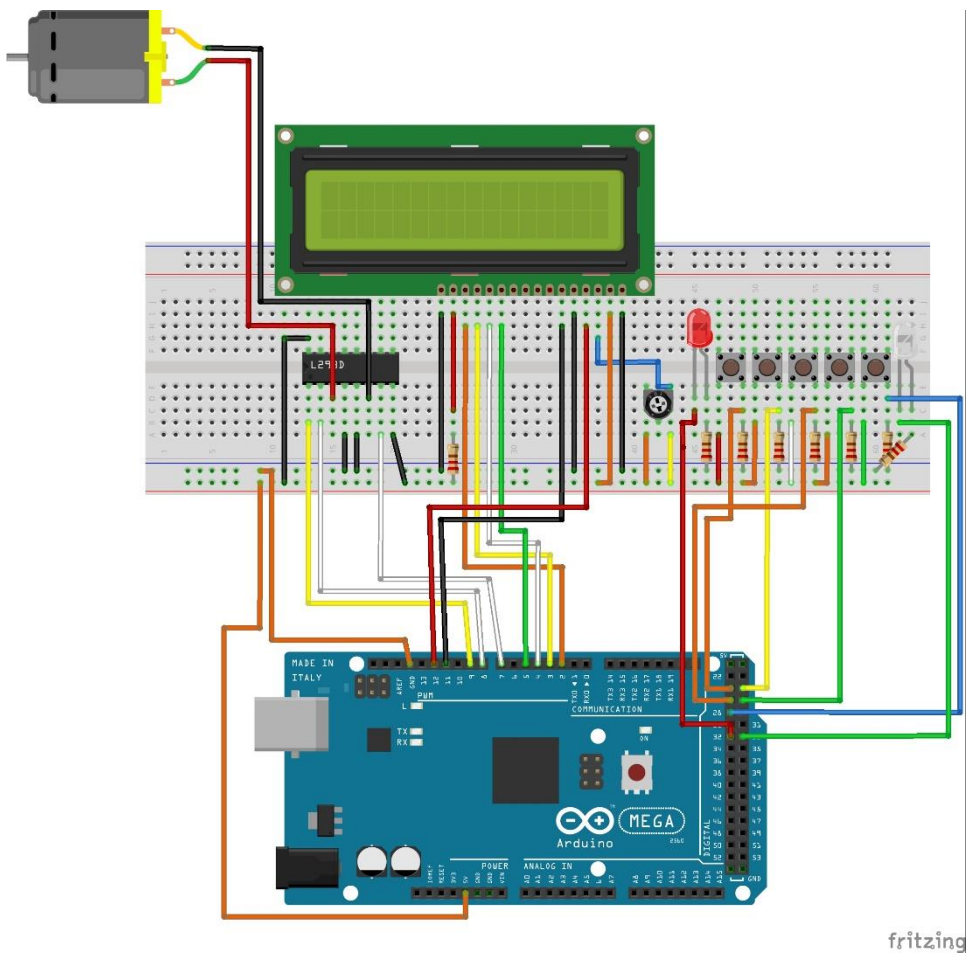
- LCD display
  Is used for displaying information for the user
- Motor
  Is used for indicating the speed of the car movement
- L293D
  Is used for controlling the motor
- Potentiometer (10 kΩ)
  Is used for the correct work of LCD display
- 5 × button(small)
  Are used for controlling the car: turning on/off, accelerating, changing the direction of driving, braking and turning on/off emergency lights
- 2 × LED (white and red)
  White LED is used as a headlight, red is used as a stop signal
- 8 × 220 kΩ resistor
  Are used for the correct work of buttons and LED lights
- 1 × 10 kΩ resistor
  Is used for the correct work of LCD display

In addition, a power supply module can be used to provide the breadboard with power.

2.2 Assembling the circuit

At this step the circuit was assembled and tested with a simple program checking correct input/output. Figure 1 illustrates the circuit schema completed in Fritzing program.
If a power supply module is used, it can be installed on the left side of the breadboard. This way, two orange wires which supply breadboard with power from Arduino can be removed.

Figure 1. The



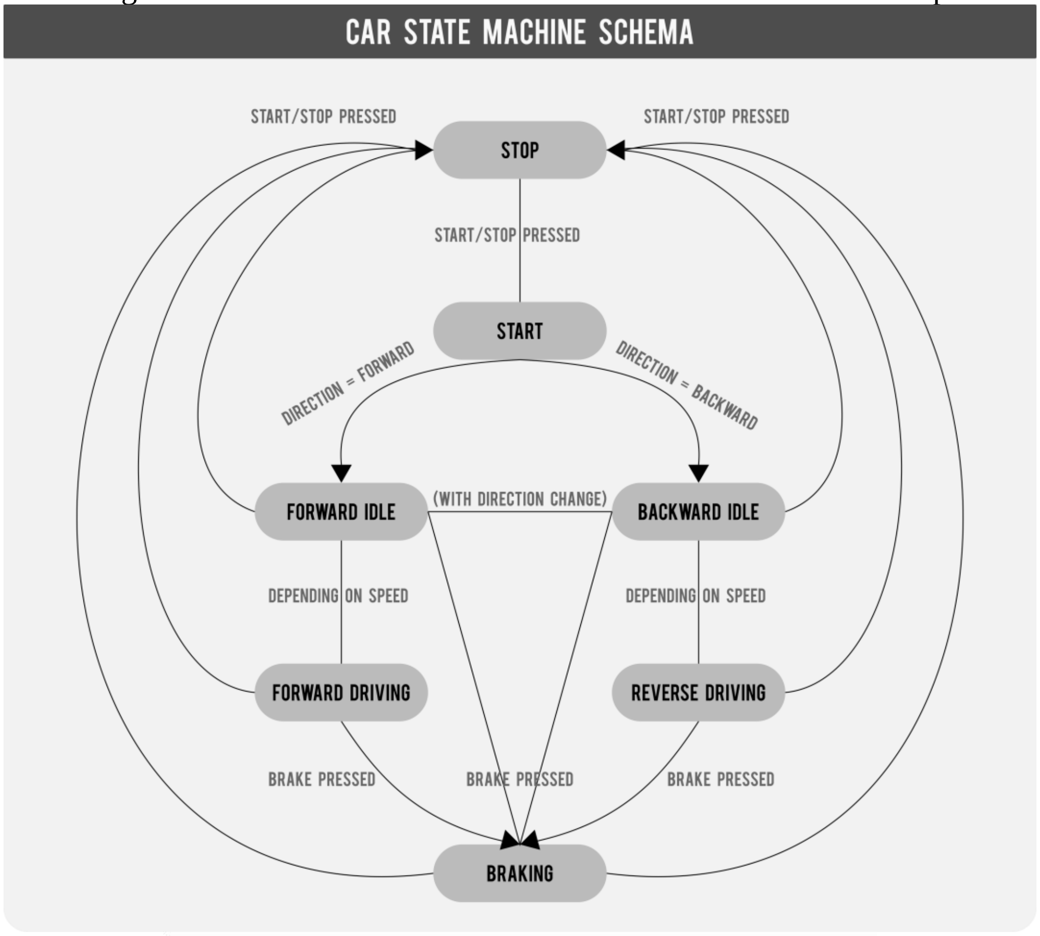
schema of the circuit

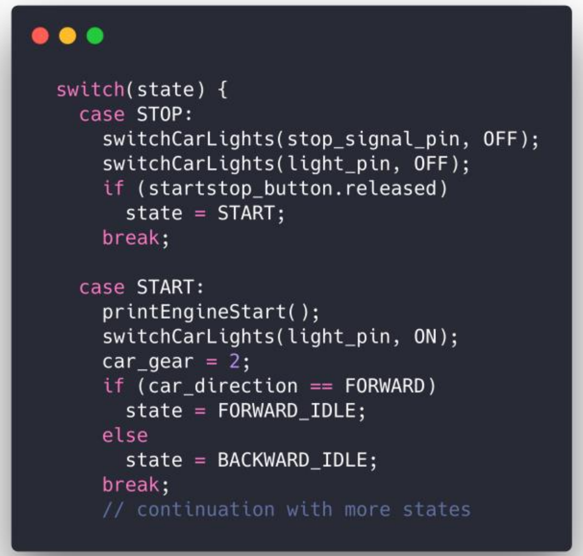
Completed circuit was tested, all flaws were corrected, and the implementation of the following step started.

2.3 Programming

State Machine Model

For maintaining car state Deterministic Finite Automata. The schema of it is depicted below:



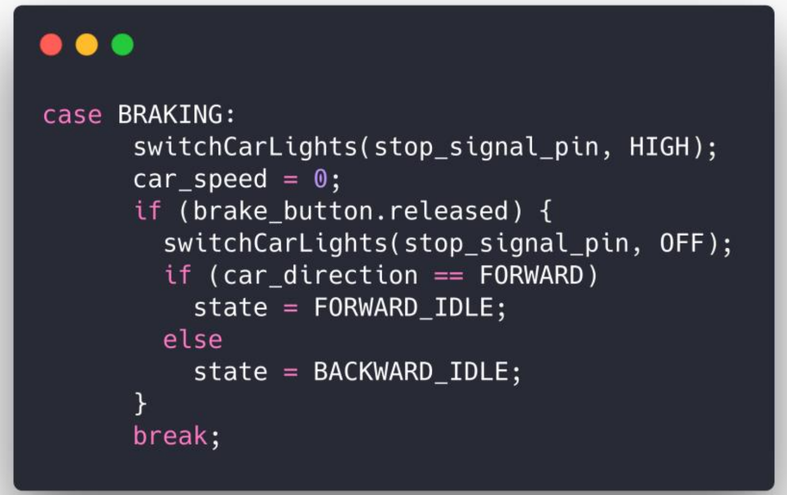


```
switch(state) {
  case STOP:
    switchCarLights(stop_signal_pin, OFF);
    switchCarLights(light_pin, OFF);
    if (startstop_button.released)
      state = START;
    break;

  case START:
    printEngineStart();
    switchCarLights(light_pin, ON);
    car_gear = 2;
    if (car_direction == FORWARD)
      state = FORWARD_IDLE;
    else
      state = BACKWARD_IDLE;
    break;
    // continuation with more states
```

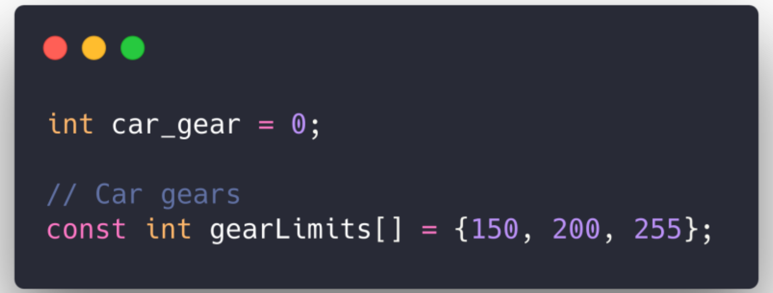
Additional features

- **Signal lights while braking**



```
case BRAKING:
      switchCarLights(stop_signal_pin, HIGH);
      car_speed = 0;
      if (brake_button.released) {
        switchCarLights(stop_signal_pin, OFF);
        if (car_direction == FORWARD)
          state = FORWARD_IDLE;
        else
          state = BACKWARD_IDLE;
      }
      break;
```

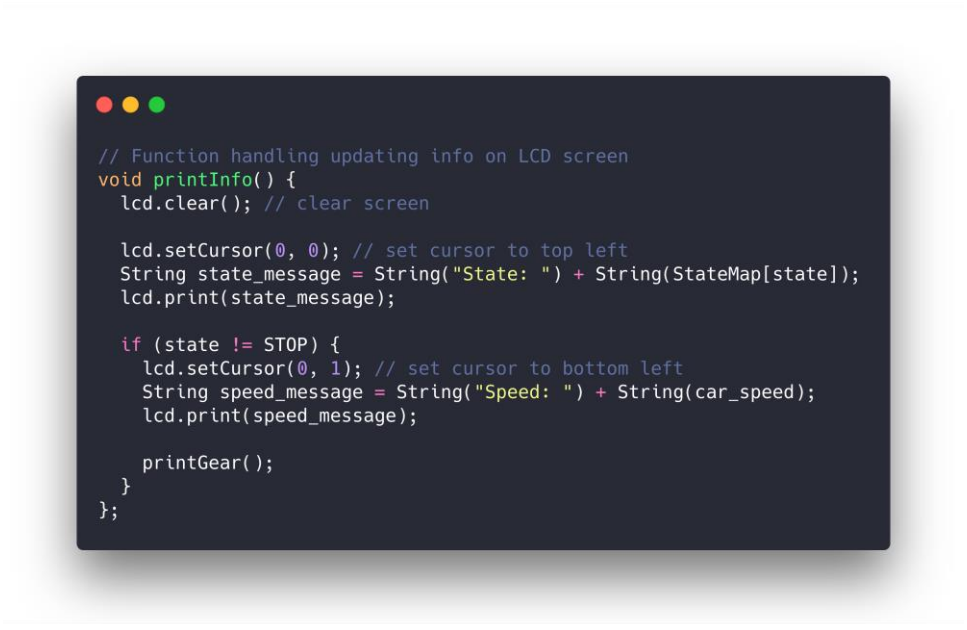
- **Car gears (or cruise control)**



```
int car_gear = 0;

// Car gears
const int gearLimits[] = {150, 200, 255};
```

- **LCD panel displaying car information (state, speed, current gear)**



```
// Function handling updating info on LCD screen
void printInfo() {
  lcd.clear(); // clear screen

  lcd.setCursor(0, 0); // set cursor to top left
  String state_message = String("State: ") + String(StateMap[state]);
  lcd.print(state_message);

  if (state != STOP) {
    lcd.setCursor(0, 1); // set cursor to bottom left
    String speed_message = String("Speed: ") + String(car_speed);
    lcd.print(speed_message);

    printGear();
  }
};
```

3. CONCLUSIONS

The aim of the work is done – the circuit and program work as planned and user can experience a car simulation state machine. Some additional features are implemented and usage of classes and functions was really helpful.