Ministry of Education, Culture and Research of the Republic of Moldova

Technical University of Moldova

Faculty of Computers, Informatics and Microelectronics

Department of Software and Automatic Engineering

REPORT

Laboratory Work №3
at Embeded Systems

Topic: Actuators

Done by: **FAF-191 | Basso Nicolae**

Teacher: **Andrei Bragarenco | Master of Engineering | Senior Lecturer**

Chișinău 2022

Topic: Actuators

Objective: To gain skills on controlling a motor using the L298 driver and to control an electric lamp through a relay.

Domain:

An **actuator** is a component of a machine that is responsible for moving and controlling a mechanism or system, for example by opening a valve. In simple terms, it is a "mover".

An actuator requires a control signal and a source of energy. The control signal is relatively low energy and may be electric voltage or current, pneumatic or hydraulic pressure, or even human power. Its main energy source may be an electric current, hydraulic fluid pressure, or pneumatic pressure. When it receives a control signal, an actuator responds by converting the source's energy into mechanical motion.

Component description:

74HC595 - The 74HC595 consists of an 8−bit shift register and an 8−bit D−type latch with three−state parallel outputs. The shift register accepts serial data and provides a serial output. The shift register also provides parallel data to the 8−bit latch. The shift register and latch have independent clock inputs. This device also has an asynchronous reset for the shift register.

L298 MOTOR DRIVER - The L298 is an integrated monolithic circuit in a 15- lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

Motor - A **DC motor** is any of a class of rotary electrical **motors** that converts direct current electrical energy into mechanical energy. The most common types rely on the forces produced by magnetic fields.

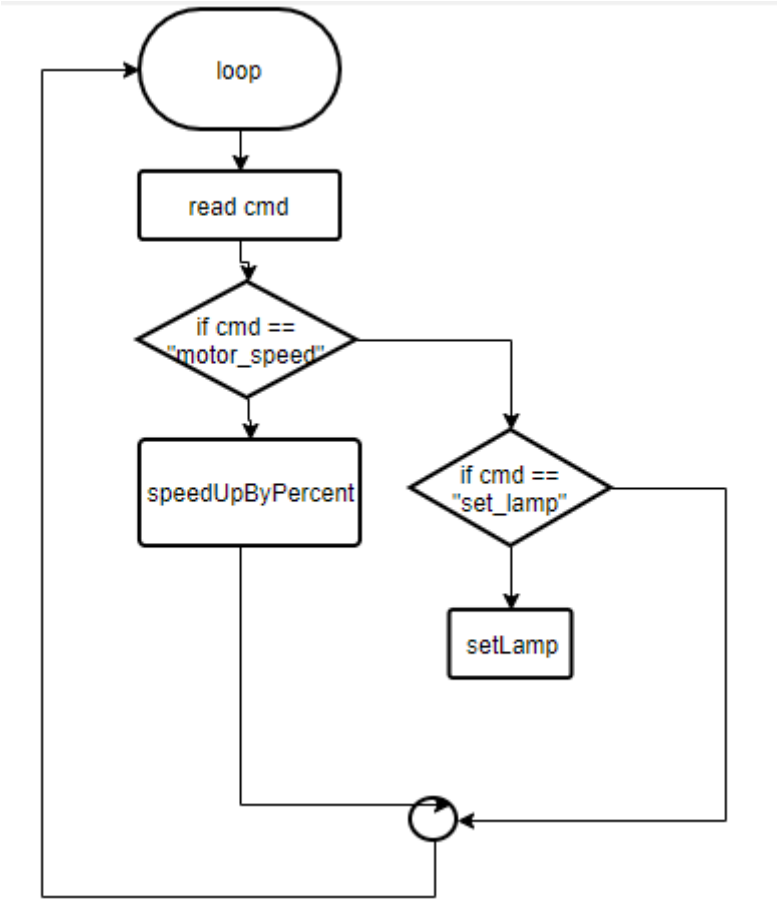
Electric lamp - An **electric lamp** is a conventional light emitting component used in different circuits, mainly for lighting and indicating purposes. The construction of lamp is quite simple, it has one filament surrounding which, a transparent glass made spherical cover is provided. The filament of the lamp is mainly made of tungsten as it has high melting point temperature. A lamp emits light energy as the thin small tungsten filament of lamp glows without being melted, while current flows through it.

Arduino Uno - The **Arduino Uno** is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc.[2][3] The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.[1] The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.[4] It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. It is also similar to the Arduino Nano and Leonardo.[5][6] The hardware reference design is distributed

under a Creative Commons Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available.

Implementation:

https://drive.google.com/open?id=170NcWg_85zL62u17zcBaALDLsCSufYJ_



Conclusions: Working on this laboratory work I have managed control a motor using a L298N driver and also to control a lamp using a relay. In order to move the motor forward and backward by percent and knowing that the value for the speed can be between 0 and 255 it is needed to calculate the speed when a percent is givedn based on this range: the formula is speed = (percentage * 255) / 100.

Annex:

Main.cpp

```
#include <Arduino.h>
#include "L298N.h"
#include "LAMP.h"
#include "LiquidCrystal.h"
#include "mystdio.h"

L298N motor(A1, 6, 5);
LAMP lamp(7);
LiquidCrystal lcd(10);
Mystdio mystdio;


void setup() {
  mystdio.open(StreamIO::SERIALIO);

  motor.setSpeed(200);
  motor.forward();

  lcd.begin(16, 2);
  printf("Started \r");
  lcd.print("Started");
}

void setMotorSpped() {
  lcd.clear();

  printf("speed (percent): ");
  int percent;
  scanf("%d", &percent);

  lcd.print("Motor speed: ");
  lcd.print(percent);
  delay(3000);
  lcd.clear();

  motor.speedUpByPercent(percent);
}

void setLamp() {
  printf("on/off?: ");
  int on;
  scanf("%d", &on);
  lcd.clear();
```

```
  if(on) {
    lamp.setOn();
    lcd.print("Lamp is on");
  } else {
    lamp.setOff();
    lcd.print("Lamp is off");
  }
  delay(3000);
  lcd.clear();
}

void loop() {
  String cmd = mystdio.readStr();
  if(cmd == "motor_speed:") {
    setMotorSpped();
  } else if(cmd == "set_lamp:") {
    setLamp();
  }
}
```

Mystdio.cpp

```
#include "mystdio.h"

String Mystdio::readStr() {
  char c;
  String str;
  do {
    scanf("%c", &c);
    if(c != 0 && c != '\r'){
      str.concat(c);
    }
  }while(c != '\r');

  return str;
}

void Mystdio::writeStr(String str) {
  for(int i = 0; i < str.length(); i++) {
    printf("%c", str[i]);
  }
}

void Mystdio::printDouble(double nmb) {
  int left = (int) nmb;
  int right = ((int)(nmb * 10)) % 10;

  printf("%d.", left);
  printf("%d", right);
}
```

```
static int Mystdio::putCharSerial(char c, FILE *stream)
{
    Serial.write(c) ;
    return 0 ;
}

static char Mystdio::getCharSerial(FILE *stream)
{
    char c;
    while(!Serial.available()) {}
    c = Serial.read();
    printf("%c", c);
    return c;
}

void Mystdio::open(StreamIO streamIO) {
  if(streamIO == SERIALIO) {
    Serial.begin(9600);
    f = fdevopen(Mystdio::putCharSerial, Mystdio::getCharSerial);
  }

  stdout = f;
  stdin = f;
}

Mystdio::Mystdio() {}
```

L298N.cpp

```
#include "L298N.h"

typedef void (*CallBackFunction) ();

L298N::L298N(uint8_t pinEnable, uint8_t pinIN1, uint8_t pinIN2){
  _pinEnable = pinEnable;
  _pinIN1 = pinIN1;
  _pinIN2 = pinIN2;
  _pwmVal = 100;
  _isMoving = false;
  _canMove = true;
  _lastMs = 0;

  pinMode(_pinEnable, OUTPUT);
  pinMode(_pinIN1, OUTPUT);
  pinMode(_pinIN2, OUTPUT);
}


void L298N::setSpeed(unsigned short pwmVal){
  _pwmVal = pwmVal;
}
```

```
unsigned short L298N::getSpeed(){
  return _pwmVal;
}

void L298N::forward(){
  digitalWrite(_pinIN1, HIGH);
  digitalWrite(_pinIN2, LOW);

  analogWrite(_pinEnable, _pwmVal);

  _isMoving = true;
}


void L298N::forwardFor(unsigned long delay, CallBackFunction callback){
  if ((_lastMs == 0) && _canMove) {
    _lastMs = millis();
    this->forward();
  }

  if (((millis() - _lastMs) > delay) && _canMove) {
    this->stop();
    _lastMs = 0;
    _canMove = false;

    callback();

  }
}

void L298N::forwardFor(unsigned long delay){
  this->forwardFor(delay, fakeCallback);
}


void L298N::backward(){
  digitalWrite(_pinIN1, LOW);
  digitalWrite(_pinIN2, HIGH);

  analogWrite(_pinEnable, _pwmVal);

  _isMoving = true;
}

void L298N::backwardFor(unsigned long delay, CallBackFunction callback){

  if ((_lastMs == 0) && _canMove) {
    _lastMs = millis();
    this->backward();
  }
```

```
  if (((millis() - _lastMs) > delay) && _canMove) {
    this->stop();
    _lastMs = 0;
    _canMove = false;

    callback();

  }
}

void L298N::backwardFor(unsigned long delay){
  this->backwardFor(delay, fakeCallback);
}

void L298N::run(uint8_t direction){
  switch(direction){
    case BACKWARD:
      this->backward();
    break;
    case FORWARD:
      this->forward();
    break;
  }
}

void L298N::stop(){
  digitalWrite(_pinIN1, LOW);
  digitalWrite(_pinIN2, LOW);

  analogWrite(_pinEnable, 255);

  _isMoving = false;
}

void L298N::reset(){
  _canMove = true;
}

boolean L298N::isMoving(){
  return _isMoving;
}

void L298N::fakeCallback(){

}

void L298N::speedUpByPercent(int8_t percent) {
  boolean to_forward = percent > 0;
  percent = abs(percent);
```

```
  int8_t speed = (percent * 255) / 100;
  stop();
  setSpeed(speed);
  if(to_forward) {
    forward();
  } else {
    backward();
  }
}
```

LAMP.cpp

```
#include <LAMP.h>

LAMP::LAMP() {
    this->pin = LED_BUILTIN;
    pinMode(this->pin, OUTPUT);
    this->on = false;
}

LAMP::LAMP(uint8_t pin) {
    this->pin = pin;
    pinMode(this->pin, OUTPUT);
    this->on = false;
}

boolean LAMP::isOn() {
    return this->on;
}

void LAMP::setOn() {
    digitalWrite(this->pin, HIGH);
    this->on = true;
}

void LAMP::setOff() {
    digitalWrite(this->pin, LOW);
    this->on = false;
}
```