TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology

IDK40LT

Ilja Kudrjavtsev 140169IAPB

AFTERMARKET ELECTRONIC DIFFERENTIAL LOCK BASED ON EXISTING ABS

Bachelor's thesis

Supervisor: Martin Rebane
MSc
Lecturer

Tallinn 2017

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

IDK40LT

Ilja Kudrjavtsev 140169IAPB

JÄRELTURU ELEKTRILINE DIFERENTSIAALILUKK AUTO ABS PÕHJAL

Bakalaureusetöö

Juhendaja: Martin Rebane
MSc
Lektor

Tallinn 2017

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Ilja Kudrjavtsev

17.05.2017

Abstract

The purpose of this thesis is to create an electronic traction control system, which will work as an electronic differential lock built on top of an existing in-vehicle ABS (Anti-lock Braking System) and controlled using an Arduino single-board micro controller.

The main problem of the project is that most off-road cars on the market have an open differential, which works on the following principle: “The wheel offering the least resistance to grip will spin, that automatically negates the drive to the wheel on the opposite side of that axis, and power will always go to the wheel offering the least resistance” [1]. At the same time there are close to no differential locks available, both original and aftermarket. Some manufacturers of aftermarket parts offer physical differential locks for some models, however, they are expensive and hard to install for the common user.

The plan is to splice in an Arduino controller into the ABS sensor circuit to get information about the wheel status. The Arduino will read the input of both sensors on an axis (right and left wheel ABS sensor) and compare them to determine which wheel has lost contact with the ground or is spinning. Based on this information, the controller will then use a pump system and several electronic “Normally Open” (Solenoid that is open in normal state) solenoids to block needed wheel with the car brakes.

The result of the project should be a working workbench prototype, which allows to be integrated into a real car’s system.

This thesis is written in English and is 29 pages long, including 5 chapters, 10 figures and 1 table.

Annotatsioon

Antud töö eesmärk on luua veojõukontrolli analoog, mis töötab nagu elektriline diferentsiaalilukk ja kasutab auto ABS seadmeid ja Arduino single-board mikrokontrollerit.

Põhiprobleemideks on, et enamik masturitest omavad tavalisi avatud diferentsiaale ja neil puuduvad diferentsiaalilukud või isegi võimalus osta see järelturult. Mõned järelturutootjad pakkuvad füüsilisi diferentsiaalilukke, aga need on tavaliselt kallid, ja tavalise kasutajale on neid raske installeerida.

Käesoleva töö idee on ABS anduri ahelasse lisada Arduino mikrokontroller, et kätte saada informatsiooni ratta olekust. Arduino loeb parameetrid mõlemast rattast samal teljel, võrdleb neid omavahel, et teada saada, milline ratas pöörleb vabalt ja milline ratas teljel on blokeeritud. Selle info põhjal otsustakse, mida teha klappidega ja pumbaga ABS plokis, et blokeerida vabalt keerav rattas.

Testimine on plaanitud lauapealses prototüübina.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 29 leheküljel, 5 peatükki, 10 joonist, 1 tabelit.

List of abbreviations and terms

ABS	*Anti-lock Braking System* An automobile safety system that allows the wheels on a motor vehicle to maintain tractive contact with the road surface according to driver inputs while braking, preventing the wheels from locking up (ceasing rotation) and avoiding uncontrolled skidding
Differential	*Differential* A gear train with three shafts that has the property that the angular velocity of one shaft is the average of the angular velocities of the others, or a fixed multiple of that average
ESP	*Electronic Stability Control/Program* A computerized technology that improves a vehicle's stability by detecting and reducing loss of traction (*skidding*)
ECU	*Electronic Control Unit* A generic term for any embedded system that controls one or more of the electrical system or subsystems in a transport vehicle.
HPP	*High-pressure Pump* The pump in the ABS is used to restore the pressure to the hydraulic brakes after the valves have released it
LED	*Light Emitting Diode*
LSD	*Limited Slip Differential* A type of differential that allows its two output shafts to rotate at different speeds but limits the maximum difference between the two shafts
TC	*Traction Control* Is typically (but not necessarily) a secondary function of the electronic stability control (ESC) on production motor vehicles, designed to prevent loss of traction of driven road wheels. TCS is activated when throttle input and engine torque are mismatched to road surface conditions

Table of contents

List of figures

List of tables

1 Introduction

1.1 Background and problem

A large amount of off-road vehicles usually only have an open differential installed. It allows the outer wheel to rotate faster than the inner wheel during a turn. In addition to this, no variety of Limited Slip Differentials (LSD - is a type of differential that allows its two output shafts to rotate at different speeds but limits the maximum difference between the two shafts [2]) or aftermarket differential locks (device that physically locks open differential to be 100% locked to rotate 2 wheels with same velocity) are available. Even if there are offers, they are usually expensive and sold in other countries. Usually they are made for more common cars, such as a Nissan Patrol or a Mitsubishi Pajero. Estonia has a larger variety of car models available, such as an Opel Frontera, a Nissan Terrano, a Hyundai Terracan or even a Land Rover (which has a proprietary traction control using the brake system as a base). Due to this, the problem is actually relevant. Also, as mentioned, physical locks are expensive, so even on a Pajero or a Patrol, a TC based system will be cheaper.

Some manufacturers of aftermarket parts offer physical differential locks. They are difficult to install and it's usually quite simple to break them if they are not used correctly. The proposed system is much easier to install and considerably less expensive.

All common cars are equipped with an Anti-lock braking system. It includes four sensors (one for each wheel) and a separate unit with electronically-controlled valves. The system processes the readings from each sensor and notifies the Electronic Control Unit (ECU) which of the wheels are blocked. The ECU then closes the necessary valves to prevent these wheels from being blocked by the brakes. However, cars that are equipped only with an ABS system and not a Traction Control System (TCS) [2] are equipped only with the valves and not a High-Pressure Pump (HPP) [3].

A possible solution is a system which uses the ABS sensors to read their signals and basically work as a somewhat modified ABS system. Meaning that it would block free wheels, allowing the wheels that are blocked with mud or other terrain to spin.

This project is useful for people who use off-road cars, whose hobby is off-road riding or for aftermarket parts manufacturers and dealers.

The project is written and the practice job is done at home and a personal garage.

Prototype tests were made on using a workbench prototype model in the Spring of 2017.

1.2 The task

This project's goals:

1. To create a hardware solution of the project.
2. To create a software solution for the micro-controller.

Requirements for the device:

1. Ability to switch the device off to use the car on highways. Due to the ability of an open differential to spin both wheels on the axis at different rate during a turn, different output of sensors during highway drive may cause faults and blocking of wheels at speed.
2. Ability to control the ABS unit's valves.
3. Ability to control the ABS unit's HPP.

1.3 Methods

To achieve the goals a variety of hardware technologies were explored, which enable the fulfillment of all the requirements:

1. Operation and mechanism of different ABS/ESP and TC units.
2. Operation and mechanism of different single-board micro controllers.

An appropriate single-board micro-controller was chosen to replicate the Traction Control Electronic Control Unit (TC ECU) and an appropriate ABS/ESP (ESP is abbreviation of Traction Control) unit was also chosen. It was taken from a donor car.

A hardware solution was designed and created with the necessary software to drive it.

The final solution was tested under special conditions on a workbench, where all of the inputs and outputs were simulated, while still technically allowing easy integration in the future.

The work is done based on Internet theoretical information, tutorials, information given by manufacturers and knowledge acquired during studies at the Tallinn University of Technology.

1.4 Overview of work

The work is divided into chapters according to the objectives of the project - in the first chapter the hardware solution is discussed. And the second part is dedicated to discussing the software that controls the hardware.

2 Hardware solution

Main idea of this work is based on peculiar properties of the open differential. “Differential allows the outer drive wheel to spin faster than inner drive wheel during a turn. This is necessary when the vehicle turns, making the wheel that is traveling around the outside of the turning curve roll farther and faster than the other.” [4]. “The wheel offering the least resistance to grip will spin, that automatically negates the drive to the wheel on the opposite side of that axis, and power will always go to the wheel offering the least resistance” [1].

This chapter’s task is to explain and develop a hardware solution for this project considering the following requirements:

1. Can be used immediately. As soon as the vehicle’s on-board electronic circuit is enabled and delivers 12 volts of voltage, all hardware should be in working condition.

2. Can operate in 12V electric circuit (Arduino’s maximum output is 5V, that’s means other hardware components are needed to operate in 12V circuit, for example relay).

3. Shows operation status to user via a series of LEDs.

4. Allows switching the device off.

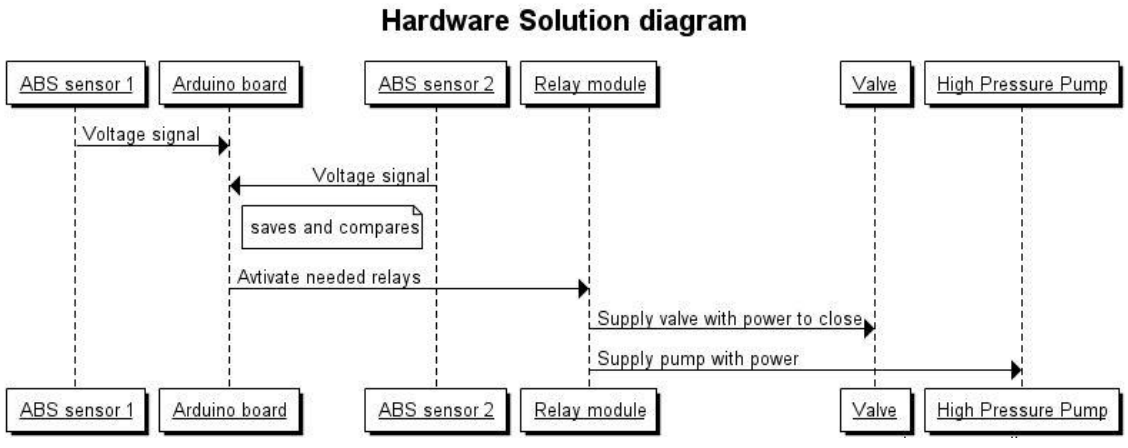

Figure 1. Sequence diagram of hardware solution. Thesis author’s diagram.

2.1 ABS/TC unit

ABS/TC unit requirements:

1. Should have four Normally Open valve (NO - normally open valve, which under normal condition is open, and is closed when a current is applied to it) because vehicles usually have four independent brake lines for each wheel, and the system should be able to control all of them.

2. Should have a High-Pressure Pump (HPP) to simulate the driver pressing the brake pedal to block wheels when needed valve is closed. Basically is used to drive the brake system.

3. Should be connectable to electrical circuit, to be powered and controlled.

A simple ABS unit was chosen. In addition, it also has an HPP. Theoretically there is no difference which unit to choose, but it should fulfill all the requirements. Chosen device was taken from a donor car, which was a BMW 7 series E23. This specific unit was chosen due to the donor car being already available. In addition to this, it already came with an HPP. The unit is a BOSCH part with the part number 0265201001, shown in Figure 1. The unit's price usually varies from 100€ to 200€ depending on the seller and the item's condition.

There are four valves in the ABS unit. One for each brake line in the car. The pump in the ABS is used to restore the pressure to the hydraulic brakes after the valves have released it. In this project, the pump is used to simulate the driver pressing the brake pedal, when needed valve is closed to block free wheel.

Figure 2. ABS/ESP(TC) Unit BOSCH

2.2 Micro-controller

Requirements for the micro-controller:

1. Should be easily programmable.

2. Should have the ability to be connected to an electrical circuit.

3. Should be user-friendly to add new features or fix bugs.

During selection of a micro-controller, the choice was between two single-board micro-controllers: IskraJS and Arduino.

IskraJS is a single-board micro-controller, an analog of the Open Source based Arduino, made by a Russian manufacturer by the name of Amperka. Its official price is 1690₽, or around 24€. IskraJS is almost the same micro-controller as an Arduino. The biggest difference is the programming language being used. IskraJS uses JavaScript, whereas an Arduino uses a subset of the C++ programming language.

Arduino is an Open Source single-board micro-controller which has Input and Output pins, allowing the user to create almost any hardware/software solution imagineable. Arduino uses the Wiring language for programming, which is a C++ based language with some changes. The program is usually written and compiled on a separate computer in a special Arduino IDE. After compilation, code is uploaded to the micro-controller via a USB cable.

Arduino Leonardo was chosen for this project due to its smaller price when compared to the base model Uno (price difference is 21€ vs. 26€), existence of a micro-USB port (easy to use due to owning a lot of micro-USB) and experience and advice of the Arduino community.

Arduino was only chosen due to availability of this micro-controller in different shops in Estonia (such as Oomipood), whereas IskraJS is available only in Russia and had to be delivered to Estonia.

2.3 Other hardware components

Out of the box the Arduino is not able to make connections that use voltages higher than 5V. Due the vehicle's electrical system operating at 12V, a relay module is needed. Relay modules are available on ebay for about 8€.

A relay is an electrically operated switch. Relays are usually used where it is necessary to control a circuit by a separate low-power signal, as it is in this project. The relay is connected to 12V+, output and 5V control wires. The control wire is coming from the Arduino's digital output and being controlled via its software. 12V+ and output are connected to the vehicle's battery and ABS valve or pump depending on software and digital output pin number.

To prototype the circuit a breadboard is also needed. A breadboard with 840 pins (64x171mm) is available in Oomipood for 11€ and a pack of breadboard wires (65 wires) is available in Oomipood for 6.60€. A breadboard allows to prototype electrical circuits easily and to modify them fast in case of a mistake.

In addition, 5 Light Emitting Diodes (LEDs) are needed, to show the current operational status of each valve, operational status of the HPP and the status of the entire system. If one of the valves is closed, a light turns on, signaling the operation of the valve. The fifth LED is used to signal operation of the HPP. To connect an LED to the Arduino board a 220Ω (ohms) resistor is needed, which has to be placed before LED [5] (Figure 2).

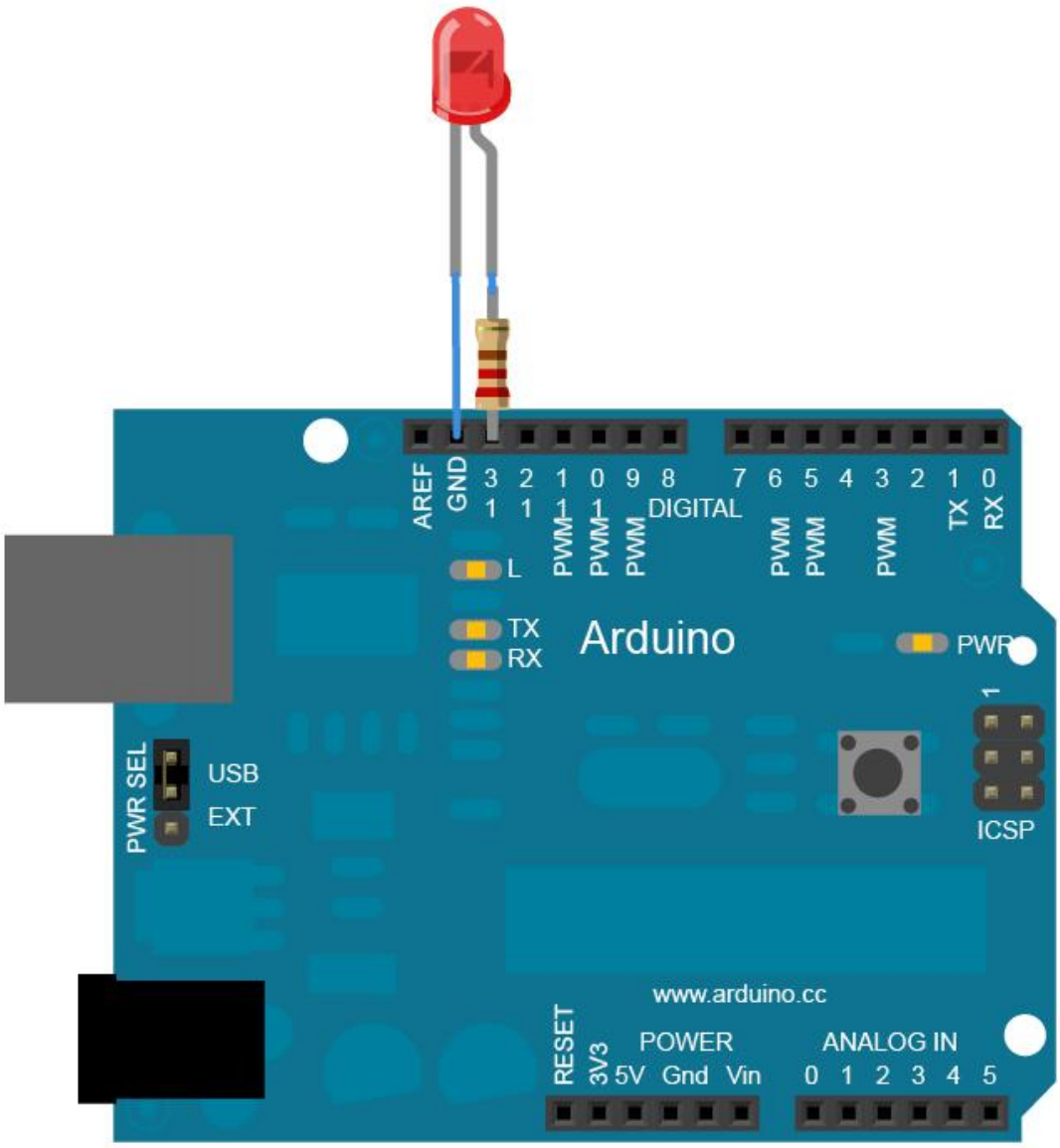

Figure 3. Scheme of connection of LED to Arduino Board. www.arduino.cc/en/tutorial/Blink

3 Micro-controller software

This chapter's task is to explain and develop a software solution for this project considering the following requirements:

1. The software should be able to read, convert and compare input signals from the ABS sensors. (There are two type of ABS sensors Active and Passive, Active produces AC voltage, Passive produces DC voltage as output. The Arduino is able to read voltage as an analog input, which can then be converted to a valid value [6])

2. Should know when, and how to operate the valves and the HPP.

3. Should show operational information via LED.

4. Code should be clean and editable to change variables depending on user's needs.

All Arduino code projects need file with an .ino extension, which the Arduino IDE uses. In this file there are two core functions "setup()" and "loop()" (Figure 3). This IDE is capable of compiling, controlling and uploading the program code to an Arduino board over a USB cable. "The Arduino IDE supports the languages C and C++ using special rules of code structuring" [7] [8].

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Figure 4. "setup()" and "loop()" functions in Arduino IDE.

The program code was written, controlled and compiled in the Arduino IDE's editor. The program was tested on a breadboard using LEDs and Arduino IDE's console.

To use the Arduino IDE's console, code should have a line: "Serial.begin(9600)" (Figure 4.). This command opens serial port at speed of 9600 bits per second. Line "Serial.begin(9600)" should be written in "setup()" method or "setupInput()" separate method for clean code, as it is made in this project. [9]

This way we allow Arduino board to send messages to IDE's console over USB cable. This function call can then be removed in the final product to reduce the amount of resources consumed.

```
void setupInput() {
  Serial.begin(9600);
```

Figure 5. Serial.begin line in Arduino IDE.

```
void setupInput() {
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }
}

void setup() {
  setupInput();
```

Figure 6. Serial.begin in separate method. While makes program do with serial connection nothing if there is no serial port to connect.

3.1 User interface

3.1.1 Signal LED

The LEDs on the breadboard should be able to show the current operational status of each valve and the HPP. To make an LED light up, “pinMode(pinNumber, OUTPUT)” line is needed [10]. The “pinMode” function sets up the pin where the LED is located. “OUTPUT” constant defines that this pin is used as an output pin. If the LED should light up, the function “digitalWrite(pinNumber, HIGH)” is called, where “pinNumber” is number of pin where LED is and “HIGH” is a constant which defines a high voltage level (HIGH - 5V, LOW - 0V … Analog of ON/OFF) [11].

The prototype was made under special conditions on a workbench, but the program should work with a real (in-vehicle) prototype as well. To makes testing easier, instead of relays LEDs are connected to the micro-controller outputs. In the program there is a separate method to close the valves, but it is explained later.

3.2 Input scan

Requirements for input scan method:

1. Should be able to read and save voltages of each wheel.
2. Should be able to compare and know, which wheel spins faster and which wheel is being blocked.
3. Should call other methods with scanned voltages.

For the system to work, it needs input from the ABS sensors. The Arduino is able to read analog DC voltage as an input, but it reads it as number in the range of 0-1023. To convert it into an actual voltage reading in the range of 0-5V it is needed to convert it using the following formula: “voltage = input_voltage * (5.0/1023.0);”, where input_voltage is a number in range of 0-1023 we get via the input, 5.0 is range to measure voltage and Arduino’s board voltage and 1023.0 maximum measured number

[6] . “scanForInput” method reads the sensors' values, converts the readings using the “convertToVoltage” method and saves the inputs as local variables. Then it prints out valuable debugging information for user values (voltages) to console and compares voltages to tell which OUTPUT (valve) to close. First “if” block checks that difference between inputs of right and left wheel are greater than the “TRIGGER_DIFFERENCE” constant, which is needed to avoid small fluctuations in readings. Second and third “if” block checks if some valve should be closed due to difference of inputs (Figure 7.). Method “shouldClose” checks that one voltage is “BASE_VOLTAGE”, which is voltage of wheel with zero speed and second wheel’s speed is not “BASE_VOLTAGE”. This method allows program to check that one wheel is blocked, and other is not. Once the condition is satisfied, the required valve is closed.

```
void scanForInput(){
  double frontLeft = convertToVoltage(analogRead(WHEEL_SENSORS[0]));
  double frontRight = convertToVoltage(analogRead(WHEEL_SENSORS[1]));
  double rearLeft = convertToVoltage(analogRead(WHEEL_SENSORS[2]));
  double rearRight = convertToVoltage(analogRead(WHEEL_SENSORS[3]));
  String values = "FL: ";
  values += frontLeft;
  values += ", FR: ";
  values += frontRight;
  Serial.println(values);
  if(abs(frontLeft - frontRight) > TRIGGER_DIFFERENCE) {
    if(shouldClose(frontLeft, frontRight)) {
      closeValve(FRONT_LEFT_OUTPUT);
    } else if (shouldClose(frontRight, frontLeft)) {
      closeValve(FRONT_RIGHT_OUTPUT);
    }
  }
}
```

Figure 7. “scanForInput” method in program code.

3.3 Operate with valves

Requirements for the function controlling the valves:

1. Should close valve for a short amount of time, set with a constant called "HOLD_DURATION". This allows the system to give the vehicle that short burst of traction that it needs and then release to let second wheel to spin.

2. Should activate the HPP right after the valve is closed, to activate brakes, which will block the free wheel, and let the second wheel, blocked with terrain/mud, to spin.

3. Should release the valve and switch the HPP off after the duration specified in the constant named "HOLD_DURATION".

4. Should not let the system begin reacting to the readings right after releasing the valve to avoid the system constantly blocking the same wheel, not letting other one to spin.

Previously, it was shown that the program compared the speeds of the wheels in the "scanForInput" method and called the "closeValve" method. Method gets pin name as an input argument.

First off, the method prints out the valve's pin number, which it received as input. Then it closes valve with "digitalWrite" command and activates the HPP (over relay also with "digitalWrite". On workbench prototype the relay is replaced with an LED) to push the brakes, which block the freely spinning wheel and allow the blocked with terrain/mud wheel to spin due to the aforementioned properties of an open differential. As soon as the braking force exerted on the manually stopped wheel is greater than the one being exerted on the wheel that was simply stuck, the stuck wheel begins to spin, while the stopped wheel is indeed stopped.

The manually blocked wheel is being blocked for short duration of time, to imitate differential lock and then it is released. At the same time the HPP is also deactivated. This will allow both wheels to spin in turns for short duration of time with the same speed. Program writes information about releasing a valve to the console, and then makes a "post-hold" delay, to avoid blocking the wheel immediately after releasing it

due to reading the input signal before it really had a chance to change. This allows second wheel to spin for this delayed time.

```
void closeValve(int pin) {
  String message = "Closing valve: ";
  message += pin;
  Serial.println(message);
  digitalWrite(pin, HIGH);
  digitalWrite(PUMP, HIGH);
  delay(HOLD_DURATION);
  digitalWrite(pin, LOW);
  digitalWrite(PUMP, LOW);
  String releaseMessage = "Releasing valve: ";
  releaseMessage += pin;
  Serial.println(releaseMessage);
  delay(POST_HOLD_DURATION);
}
```

Figure 8. “closeValve” method in program code.

Input signal		Output signal		
Right wheel	**Left wheel**	**Right valve**	**Left valve**	**HPP**
0	0	0	0	0
1	0	1	0	1
0	1	0	1	1
1	1	0	0	0

Table 1. Operation of system. 0 - spinning/open/deactivated, 1 - blocked/closed/activated.

4 Prototype

4.1 Prototype version 1.0

Requirements for prototype:

1. Prototype should fulfill all the requirements of hardware solution.

Prototype version 1.0 is a prototype, built on a breadboard. Instead of relays LEDs are used to show the operation of valves and the HPP. An input signal is generated with a DC generator. Due to using LEDs instead of valves and the HPP, there was no need to connect relays to allow operation in a 12V circuit. However, the breadboard prototype allows to connect relays with/before LEDs to control valves and the HPP with Arduino's code and show operational status to the user. To fulfill requirement of ability to switch system off, on/off switch is integrated. This allows to cut off Arduino's power supply.

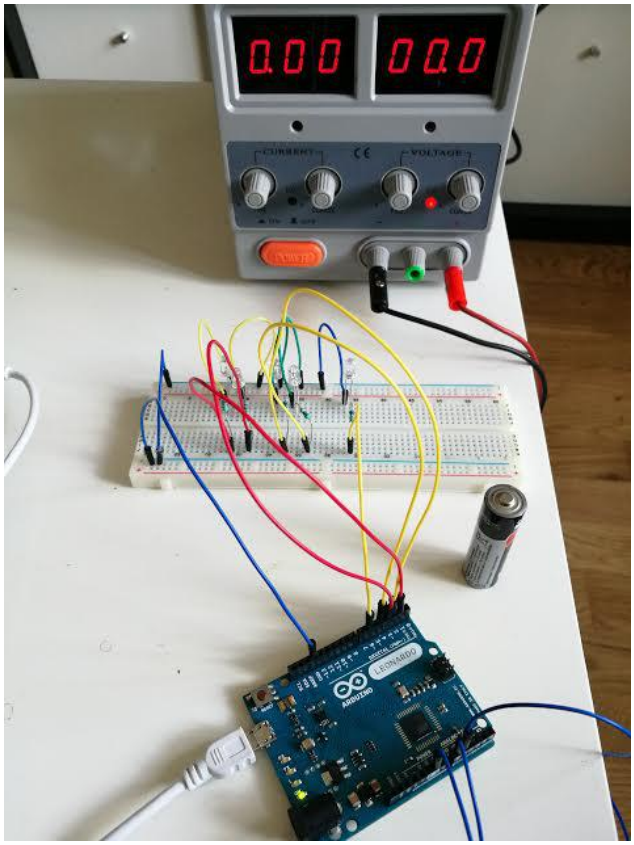

Figure 9. Prototype version 1.0

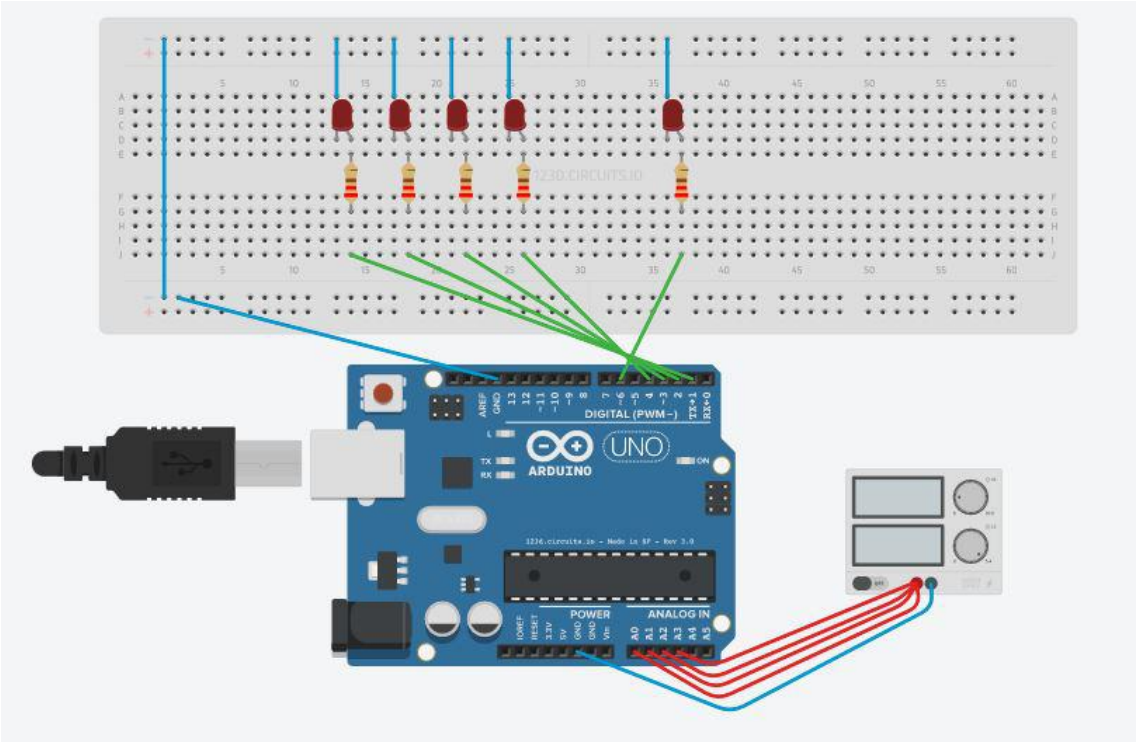

Figure 10. Prototype scheme. Created with circuits.io online editor [12]. Thesis author's scheme.1

Unfortunately, due to the fact the author's vehicle did not have a fully working set of ABS sensors, it was not financially feasible to integrate the prototype into the vehicle. Because B-class (replica) sensors for the author's vehicle do not exist, the only possible solution was to buy a set of original sensors, each of which costs around 150-200€ per piece.

4.2 Measurements and tests

Test were made with a workbench breadboard prototype using a power adapter to generate input signals. A variety of input signals were tested, same as mentioned in Table 1 in chapter 3.3 (Operation with valves). Software and hardware solution fulfilled all of the requirements and worked correctly during the test period.

5 Summary

The purpose of this work was:

1. To create a hardware solution for the project.

2. To create a software solution for the micro-controller.

Results of the project:

1. Created a working hardware solution for the project.

2. Created and tested a software solution for the micro-controller.

Created solution is perfect for a common user, who has an off-road “project vehicle” with an ABS unit that has a HPP and working ABS sensors. Because code is user-friendly, it allows the user to connect all of the sensors to the correct inputs of the Arduino board.

Because the author had issues with the integration of the prototype into his own project vehicle, it was impossible to test the prototype on an actual vehicle, but prototype was successfully tested on a workbench, showing perfect results with all kind of input signals. Information provided by this failure is very necessary for future integration of prototype into the project vehicle.

To achieve success in the scope of this project, the author used his knowledge acquired during his studies at the Tallinn University of Technology, studying computer science.

This project was very useful for the author. The author learned how to use an Arduino micro-controller to change the physical state of different objects and how to control real world objects with lines of code. All of the experience gained by the author of the project has opened up a wider world of information technology solutions to him, many of which help to better the world.

Git repository

git@gitlab.com:Kudrjavtsev/AftermarketDiffLock.git

https://gitlab.com/Kudrjavtsev/AftermarketDiffLock

References

[1] “LandRover Traction control explained,” [Online]. Available: http://www.bhclrc.org.uk/info/DiffLocks_and_etc.htm. [Accessed February 2017].

[2] “Limited-slip Differential,” [Online]. Available: https://en.wikipedia.org/wiki/Limited-slip_differential. [Accessed March 2017].

[3] “Traction Control System,” [Online]. Available: https://en.wikipedia.org/wiki/Traction_control_system. [Accessed February 2017].

[4] “Anti-lock braking system,” [Online]. Available: https://en.wikipedia.org/wiki/Anti-lock_braking_system. [Accessed February 2017].

[5] “Differential (mechanical device),” [Online]. Available: https://en.wikipedia.org/wiki/Differential_(mechanical_device). [Accessed February 2017].

[6] “Arduino blink tutorial,” [Online]. Available: https://www.arduino.cc/en/tutorial/blink. [Accessed March 2017].

[7] “Arduino analog voltage measurement tutorial,” [Online]. Available: https://www.arduino.cc/en/Tutorial/ReadAnalogVoltage. [Accessed March 2017].

[8] “Arduino,” [Online]. Available: https://en.wikipedia.org/wiki/Arduino. [Accessed February 2017].

[9] “Arduino Software. Arduino IDE,” [Online]. Available: https://en.wikipedia.org/wiki/Arduino#Software. [Accessed March 2017].

[10] “Arduino Serial,” [Online]. Available: https://www.arduino.cc/en/serial/print. [Accessed March 2017].

[11] “Arduino Reference. Pin Mode,” [Online]. Available: https://www.arduino.cc/en/Reference/pinMode. [Accessed March 2017].

[12] “Arduino Reference. Digital Write,” [Online]. Available: https://www.arduino.cc/en/Reference/digitalWrite. [Accessed March 2017].

[13] “Autodesk Circuits online editor,” [Online]. Available: https://circuits.io/lab. [Accessed April 2017].