1 2 9 0

UNIVERSIDADE D
COIMBRA

Samuel Marques Salgado

DESENVOLVIMENTO E CONTROLO EM MALHA FECHADA DE UMA BARREIRA ANTIFOGO ATIVA REFRIGERADA A ÁGUA

VOLUME 1

Dissertação no âmbito do Mestrado em Engenharia Mecânica na Especialidade de Energia e Ambiente orientada pelos Professores Doutores Carlos Xavier Pais Viegas e Miguel Rosa Oliveira Panão, apresentada à Faculdade de Ciências e Tecnologia/Departamento de Engenharia Mecânica.

Julho de 2020

1290 FACULDADE DE CIÊNCIAS E TECNOLOGIA UNIVERSIDADE D COIMBRA

Development and closed-loop control of an active water-cooled fire-proof barrier

Submitted in Partial Fulfilment of the Requirements for the Degree of Master in Mechanical Engineering in the specialty of Energy and Environment

Desenvolvimento e controlo em malha fechada de uma barreira antifogo ativa e refrigerada a água

Author

Samuel Marques Salgado

Advisors

Carlos Xavier Pais Viegas

Miguel Rosa Oliveira Panão

Jury

President	Professor Doutor **Domingos Xavier Filomeno Carlos Viegas** Professor Catedrático da Universidade de Coimbra
Advisor	Professor Doutor **Carlos Xavier Pais Viegas** Professor Auxiliar da Universidade de Coimbra
Vowels	Professor Doutor **Joaquim Norberto Cardoso Pires da Silva** Professor Associado da Universidade de Coimbra Professor Doutor **Jorge Rafael Nogueira Raposo** Professor Auxiliar da Universidade de Coimbra

Institutional Collaboration

Associação para o Desenvolvimento da Aerodinâmica Industrial

Centro de Estudos sobre Incêndios Florestais

Coimbra, July, 2020

One day, you might look up and see me playing the game at 50. Don't laugh.
Never say never, because limits, like fears, are often just an illusion.
Michael Jordan, Basketball Hall of Fame induction speech, 2009

Dedicated do my family and friends.

Acknowledgments

Concluding this work was a difficult task, and it required persistence and an extra effort given by those who surround me. Although it is a somewhat ungrateful task to choose the names of those who helped me, I would like to mention some to whom I cannot fail to pay my public thanks.

My first thanks go to the ADAI institution, for the courage and effort made in the acquisition of the equipment needed for this work. Also, I would like to thank those who work there for the friendliness and sympathy during my time.

Succeeding, I would like to thank Prof. Dr. Miguel Rosa Oliveira Panão and especially to Prof. Dr. Carlos Xavier Pais for their permanent readiness, flexibility, and all the suggestions given for the improvement of this work. A huge thanks to them.

To my colleague with whom I shared the theme of this thesis, I thank you for the help over the last year. Besides, I must mention my friends and housemates from the last four years, for all the moments lived and for the permanent companionship. Without you, this journey would not have been so pleased. Finally, to my remaining friends for the support and friendship all the time, a truthful thanks.

Finally, I couldn't help but thank my whole family, including Carolina, for the patience, encouragement, and all the leisure moments provided in the last years. Here are my sincere thanks to everyone.

Abstract

The disasters that occurred during the summer due to forest fires, and their consequent devastation, with more incidence of occurrence in recent years, are due to climate change, as well as insufficient knowledge about the patterns of fire behavior in any circumstances.

In order to develop knowledge on this topic, numerous research works have been carried out in DEM. One of these resulted in the prototype of a fire-proof barrier to be installed around a structure, protecting it from the flames advance. To this barrier was added a *flat* spray to cool the fabric composed of fiberglass and covered by an aluminum tissue (highly reflective material, essential to reflect heat). The tests with the water-cooled prototype pointed to the ability to stop the fire advancing.

The argument addressed in this dissertation consists of implementing an automatic operating system to manage a valve as a function of the fabric surface temperature, optimizing the water consumption. Thus, the methodology explored consists of programming the valve that controls the production of the incident spray in the barrier, correctly programming the components involved, and verifying the feasibility of this solution through experimental tests.

In the experimental tests, carried out with real fire, we tried to verify the temperature distribution in the barrier, compare the state of the fabric after the end of combustion, and evaluate water consumption. From the tests, it was concluded that the autonomy generated by the water consumption control represents a saving of water between 35 to 50%, increasing the action time of the system if the supply comes from a tank with a limited amount of water available.

Keywords: Forest fires, Fire-proof Barrier, Valve, Automatic sprinkler system, Temperature, Controller.

Resumo

Os desastres ocorridos durante o Verão devido aos fogos florestais, e a sua consequente devastação, com mais incidência de ocorrência nos últimos anos, devem-se às alterações climáticas, assim como ao insuficiente conhecimento sobre os padrões de comportamento do fogo em qualquer circunstância.

Com o intuito de desenvolver conhecimento sobre esta temática, vários trabalhos de investigação têm sido realizados no DEM. Um desses resultou no protótipo de uma barreira anti-fogo, a ser instalada à volta de uma estrutura, protegendo-a do avanço das chamas. A esta barreira foi instalado um *flat* spray para arrefecer a tela composta por fibra de vidro e coberta por uma película de alumínio (material altamente refletivo, essencial para refletir o calor). Os testes realizados com o protótipo arrefecido a água apontaram para a capacidade de travar o avanço do fogo.

O argumento abordado nesta dissertação consiste em implementar um sistema de funcionamento automático de atuação de uma válvula em função da temperatura na superfície da tela, otimizando o consumo de água incidente na barreira. Assim, a metodologia explorada consiste na programação da válvula que controla a produção do spray incidente na barreira, programar devidamente os componentes envolvidos e verificar a viabilidade desta solução através de testes experimentais.

Nos testes experimentais, realizados com fogo, procurou-se verificar a distribuição de temperatura na barreira, comparar o estado da tela após o término da combustão e avaliar o consumo de água. Dos ensaios conclui-se que a autonomia gerada pelo controlo do consumo de água representa uma poupança da água entre 35 a 50%, aumentando o tempo de ação do sistema, caso o fornecimento provenha de um depósito com uma quantidade de água disponível limitada.

Palavras-chave: Incêndios florestais, Barreira antifogo, Válvula, Sistema de supressão automático, Temperatura, Controlador.

Contents

LIST OF FIGURES

LIST OF TABLES

ACRONYMS/ABBREVIATIONS

ADAI – Associação para o Desenvolvimento da Aerodinâmica Industrial

ARM – Aggressive Response Map

CRM – Conservative Response Map

DEM – Departamento de Engenharia Mecânica

FF – Forest Fires

GUI – Graphical User Interface

NC – Normally Closed

NO – Normally Open

WSS – Wildfire Suppression System

1. INTRODUCTION

1.1. Motivation

In 2017, Portugal suffered two catastrophic forest fires (FF) in only four months, on June 17th, and later on October 15th. In these two fires, more than 500 companies saw their infrastructures consumed by flames, and about 1,300 homes met the same end. The fires claimed more than 120 victims, making 2017 the worst year in the country's recorded history in wildfire fatalities (Viegas et al., 2019). In July 2018, in locations near Athens (Greece), the population experienced the deadliest FF in the history of that country, where 102 people died, and more than 1,500 first-homes were destroyed (Konstantinidis, 2019). In the same year, in California (USA), a massive fire consumed more than 6,500 homes, killing 97 people and becoming the deadliest fire in the history of that state (Fuller, and Nicas, 2018), where every year such catastrophes become systematic. Recently, between September 2019 and January 2020, wildfires in Australia consumed more than 11 million hectares of forest and rural areas, having killed at least 33 people (Lawrie, 2020). For comparison purposes, England has roughly 13 million hectares, which means the total burned area in Australia corresponds to 84 % of England's territory. In all these tragic events, there were extremely favorable conditions for the fast spread of fire, with high temperatures and wind speeds and shallow humidity values.

In Australia, the causes of the large FF were related to the abnormally hot and dry climate felt during the year of 2019, the hottest and driest year since 1910 (Lopes et al., 2020). Furthermore, there were also records of strong gusts of wind and dry thunderstorms. All these factors, combined with criminal action – only in the state of New South Wales, 24 people were arrested for arson (Lopes et al., 2020) – led to what the authorities and media called "the perfect storm" and enabled the proper conditions for the ignition and fast spreading of a fire front.

FF phenomenon, combined with climate change, raises the levels of anthropogenic destruction of natural environments with severe societal implications. Although the number of fires in Australia was not the highest ever (Lopes et al., 2020), the burned area in 2019 was the largest since 1970 – the year when the New South Wales (NSW) office of environment and heritage began to make this registration. Figure 1.1 depicts the record of burned areas in the last fifty years (Morton, 2019).

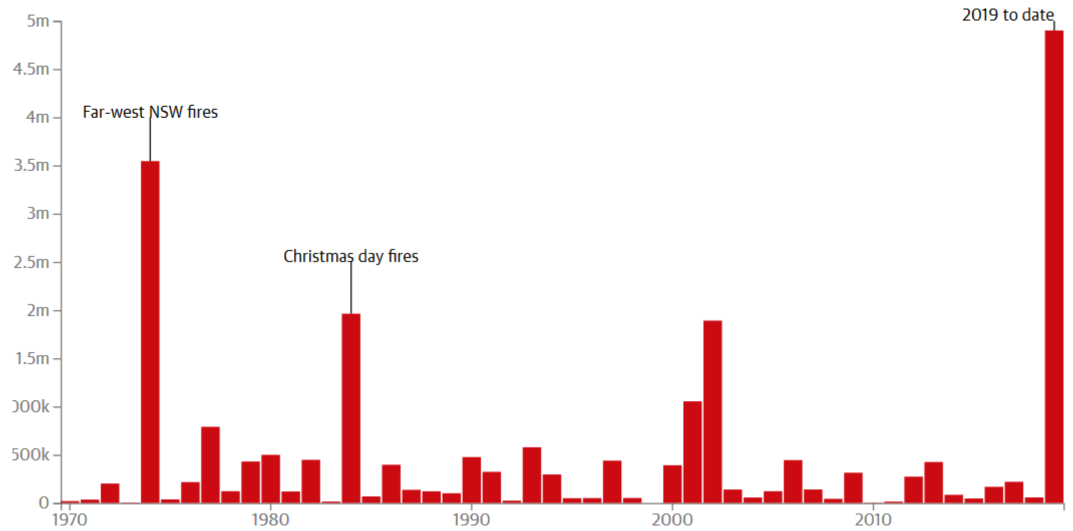


Figure 1.1 – Total area burned in hectares per fire season in Australia, NSW office of environment and heritage

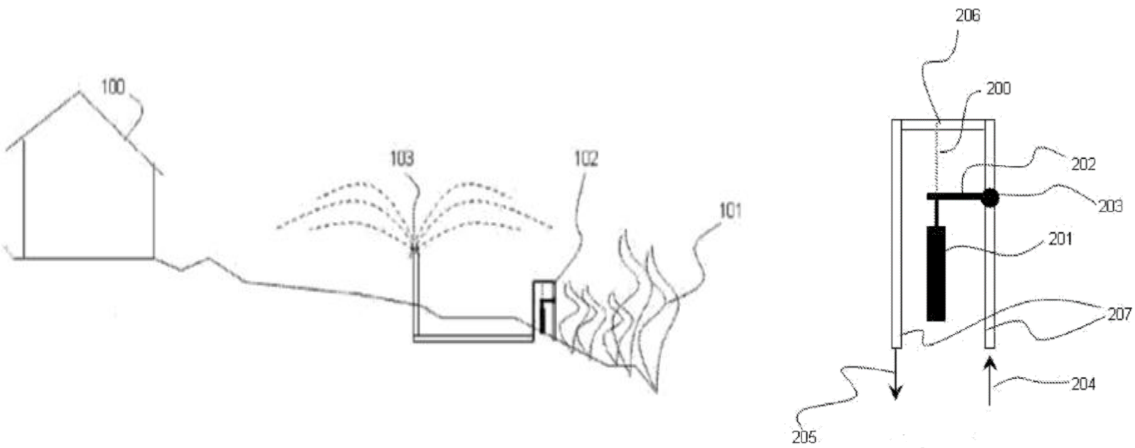
In addition to the loss of 33 human lives and the vast burned area, almost 3,000 homes were devastated due to direct fire action (Readfearn, 2020). This destruction was mainly due to flames intensity and the lack of means to help the population in defense of their material goods. Weather conditions are increasingly adverse each year, and preventive measures are required to avoid the destruction caused by this phenom. Some of the critical factors already validated for these occasions are the following: land management and forest planning; investment in human and material resources or; investment in solutions that allow citizens to defend their properties. An example of the latter is the fire-proof barrier discussed in this work or fire blanket's applied to houses with " "*endurance under severe heat-exposure and high-wind conditions*" " (Takahashi, 2019).

Based on the examples above, and taking into account the trend towards the occurrence of longer hot seasons and higher temperatures – " "*the annual number of days with a heat index above 37 ºC will double when compared to the end of the 20th century*" " (Dahl et al., 2019) – setting the conditions for an increase of the magnitude of fires. The work presented in this thesis aims to create a solution that could help to minimize the damage from fires and protect the population and their belongings.

1.2. State-of-the-art

The process of designing a closed-loop control system for the active water cooling system of a fire-proof barrier begins by reviewing similar solutions and understand their operating principles. Namely, the methods for fire protection using water sprinklers.

Charles Yount proposed an outdoor system, shown in Figure 1.2, that uses water or retarding agent for fire suppression (Yount, 2012). Fire suppression occurs when the flames contact the valve control system (shown in detail in Figure 1.2, a)), containing a “hair trigger”[1] (200) that weakens or fails when exposed to high temperatures. The failure of the “hair trigger” causes the falling of weights (201), opening the hydraulic valve (203) used to control the water flow that enters (204) and exits (205) through the duct (207). From the moment the valve opens, the retarding agent flows to the sprinkler system (103), strategically placed between the valve and the protected structure (100).



(a) (b)

Figure 1.2 – (a) Basic scheme of Yount’s mechanism; (b) Hair trigger mechanism in detail

This system has the advantage of being low cost, robust and reliable (since the “hair trigger” is only activated when it becomes in contact with temperatures in the order of flame temperature). Thus, the occurrence of false alarms is unlikely. However, there are some disadvantages to this system, namely the absence of any form of water flow management. This system trigger design only allows a binary response, which is entirely closed or fully open. Another design limitation is the trigger response delay. If the fire spread

[1] A “hair trigger” is a weak link, made of fibrous material designed to fail when exposed to high temperatures, characteristic of wildfires, typically in the order of 550ºC

rate is very high, the fire front can reach the property before the “hair trigger” heats at a temperature high enough to activate. Furthermore, it is a system that requires periodic maintenance, in particular, the replacement of the “hair trigger” mechanism after usage.

From this work, the idea of developing a solution whose aesthetic impact is as small as possible is kept. Aspect is significant, not only because of the aesthetic issue itself but also since the more unnoticed the system goes, the better, even for vandalism-related issues. Like in this work, we will also try to create a solution as little expense as possible, knowing, however, that the equipment to be employed will have to be further technologically evolved, thus not allowing to equalize the costs of that system.

In 2015, Weber, (2015) patented the WSS (Wildfire Suppression System) that uses a set of wildfire sensors to identify a fire threat at up to 8 km. These sensors employ thermal and visual imaging to identify fire spots. WSS is also designed to inform authorities (firefighters and police) about the presence of fire and includes the possibility of having its own electric generation unit. In this way, it is possible to activate the WSS in case of a power grid failure.

Figure 1.3 illustrates how the WWS system works. When the wildfire sensors (1) detect the presence of fire – given by the presence of flames (15), smoke (17), abnormally high temperatures, or climatic conditions (16) – they send a signal via wireless to the computer (3) where it is processed. Then, in case of detecting a potential fire threat, the WWS deploys various protection systems, notifies firefighters and police authorities, and initiates the hydraulic pump (7). When the fire-retardant pressure stabilizes, it pumps the coolant to the sprinklers (8) to humidify the initially delineated zones, protecting the building (12) and the surrounding landscape (13 and 14).

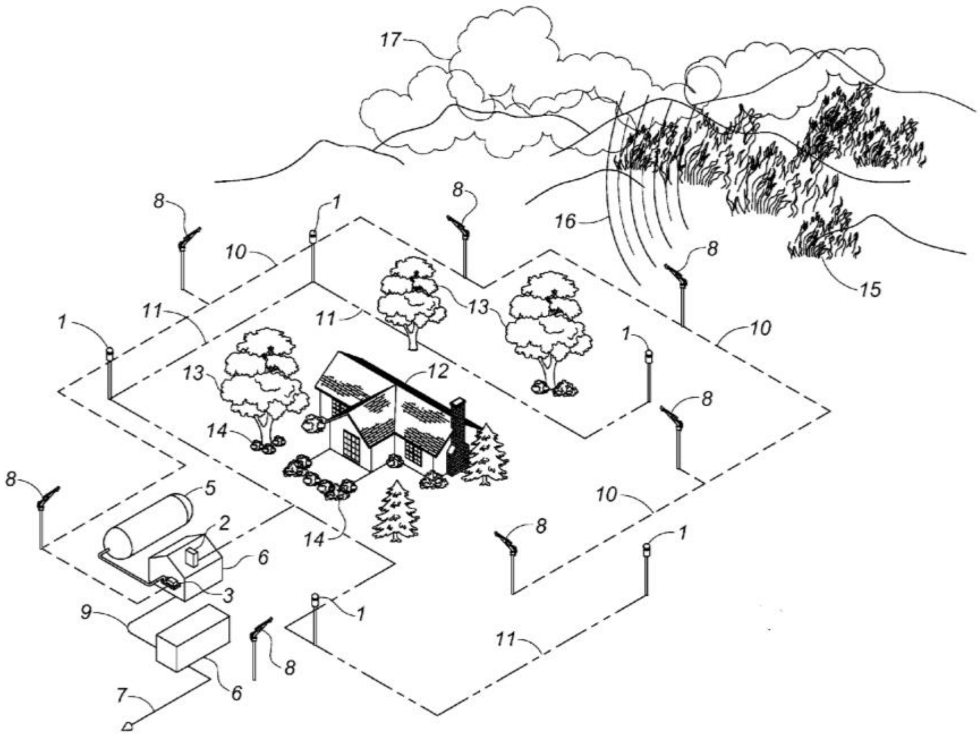


Figure 1.3 – Possible format for positioning the components of the WSS

Analyzing this system, one should highlight the broad range of solutions it presents, such as it's an automatic mechanism, which is adapted to overcome possible faults of the electricity grid, and might notify the authorities automatically. On the other hand, the fact that the sprinklers act manually (unable to change the position depending on the signals received by the sensors) implies the need for human supervision. Also, the sensitivity of sensors that may be programmed to detect fire signals at considerable distances requires some caution as it puts at risk the possibility of incurring false alarms and cause damage to the water-saving.

Regarding this work, the idea of introducing a generator to deal with possible power failures is preserved. However, this possibility is subject to future work. Besides, when we compare this system to the intended to develop, we conclude that we desire a solution with a different kind of layout.

In 2017, Smith, et al., (2017) proposed the kit AWSS (Automatic Wildfire Suppression System), which consists of a set of temperature sensors (thermocouples or thermistors), a water distribution system, a solenoid valve, and a controller capable of processing data that comes through Wi-Fi® or Bluetooth®. According to Figure 1.4, when sensors (340) detect a temperature value above the safety limit, this information is sent to the controller (305), which activates the alarm, alerts the authorities, and instructs the opening of the solenoid valve (330). The water in the reservoir (325), pumped through the distribution system (310), reaches the sprinkler system (335) and acts to suppress the fire. This set of operations makes up the AWSS kit (300).

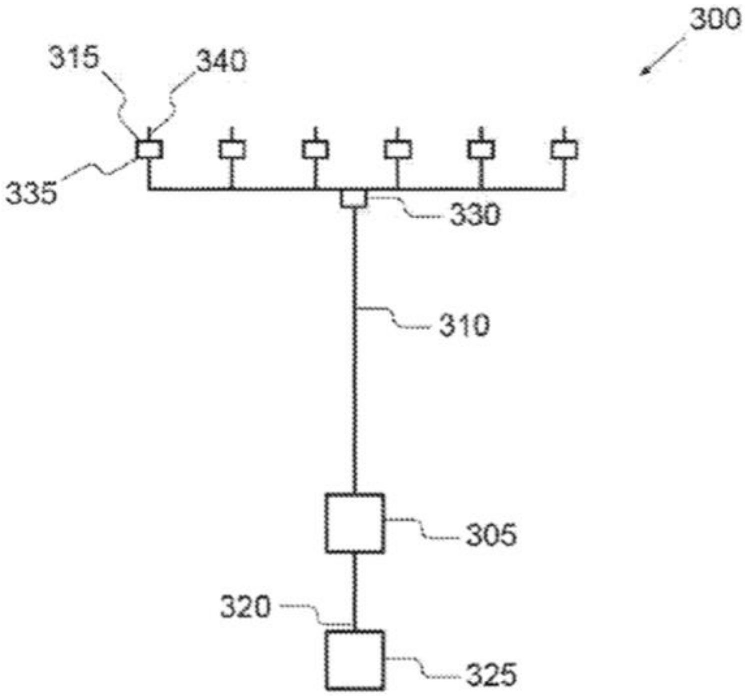


Figure 1.4 – AWSS proposed scheme

The kit AWSS stands out by its ability to be easily assembled or disassembled. It is a simple system that needs little equipment to fulfill its function. Lastly, it also does not require any human intervention throughout the process. On the other hand, this kit confinement to pre a defined perimeter may turn out to be too short to prevent the spread of fire objectively. Besides, to optimize the performance of this system, a solution should be explored to mitigate water waste.

As for AWSS invention, its simplicity and the low number of equipment to make automatic control is relevant. The solution under development in this dissertation is similar to the AWSS. It aims to ensure the efficient cooling of the fabric AWSS, where the range is limited to a reduced area.

Finally, none of the products available in the market fits our standards. The product closer to the desired outcome consists of a mechanism involving sprinklers installed on the roof to humidify the surrounding environment. The actuation of these sprinklers depends on the feedback signal provided by a temperature sensor. The system would help fight the flames in small/medium-sized fires but would be inefficient in fires where the effect of radiation is decisive. Therefore, one may conclude that, here, there is an unexplored market niche.

1.3. Starting point

The work in this thesis complements previous research in the development of a fire-proof barrier (see Batista, (2018), Costa, (2018), and Albino, (2019)).

According to these works, the most efficient solution to protect materials with a fire-proof barrier against flames was the following: use a barrier where fabric consists of a material bonding an aluminum foil with a fiberglass tissue, disposed along the perimeter surrounding the area intended for protection – Figure 1.5. Also, for every 1.2m (length of the barrier prototype, Figure 1.6), a flat spray issued from a sprinkler incorporated in a pipe system above the barrier increases the humidity level in the vicinity, since it is essential to avoid the fabric´s degradation (Viegas et al., 2020). The viability of the fire-proof barrier was successfully verified both in laboratory and field tests, with the details provided in the three studies mentioned at the beginning of this section.

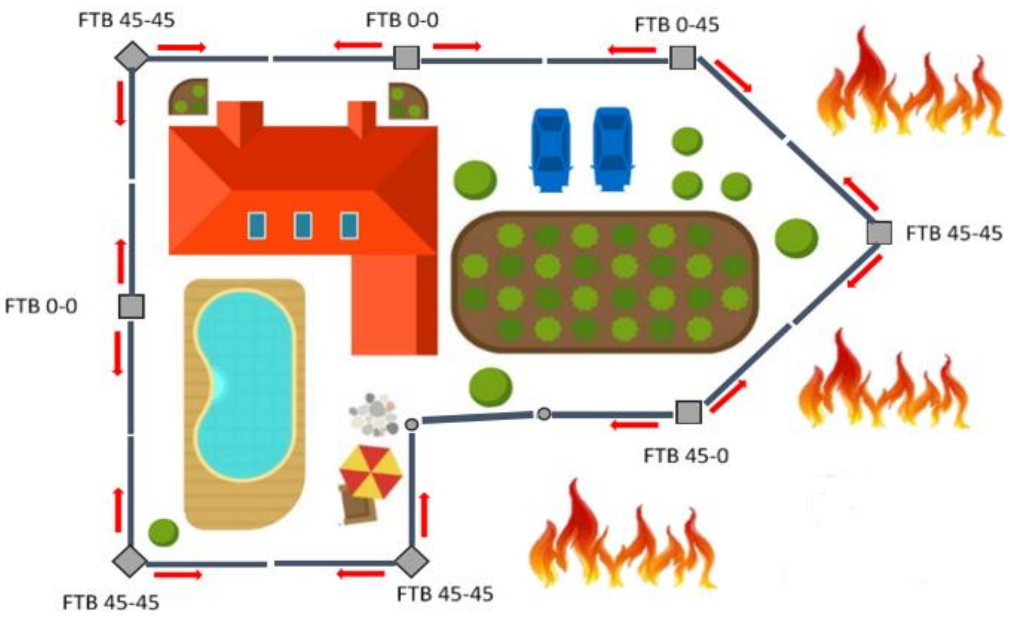
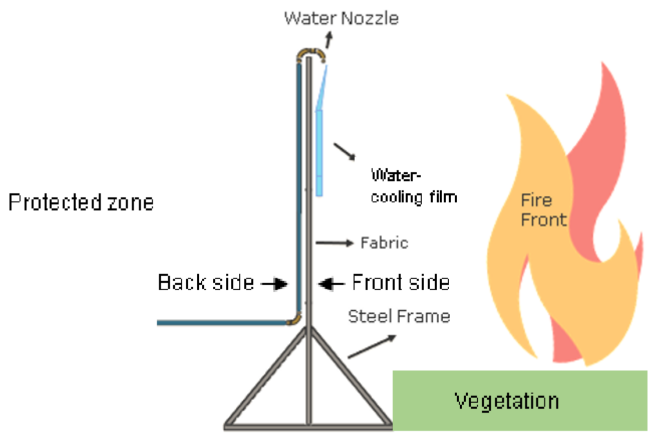


Figure 1.5 – Example of fire-proof barrier layout along with a dwelling, (Costa, 2018)



(a) (b)

Figure 1.6 – (a) Barrier scheme (b) Barrier prototype in action, (Viegas, et al., 2019)

The prototype developed at this point in the research did not fully meet the desired outcome. Thus, this dissertation elaborates on the optimization of the barrier by adding automatic control over the water distribution system. The management of the water flow system will depend on the temperature value monitoring the heat emanating from a fire front to avoid dry areas and, consequently, damage the fabric (Meredith et al., 2013).

1.4. **Objectives and methodology**

The main goal of the work presented in this dissertation is to develop a closed-loop automatic water sprinkler system, which guarantees the cooling of a fire-proof barrier while optimizing water usage.

The specific objectives defined to reach this goal are the following:

1. Development and implementation of hardware control instruments enabling an active control over the water cooling system, including mechanisms for actuation in closed-loop of a proportional solenoid valve.
2. Development and implementation of the control software.
3. Test and validation of the integrated control system.

The methodology used was the following: from previous works, we knew that a considerable amount of water would be saved if the water flow could be managed according to the temperature measured on the fabric. Consequently, a valve (flow regulation function, has 8 positions), and four thermocouples were adopted as central elements of this system. The next process went through an individual study on the valve – and its controller, RE4 – to understand its operating conditions and, from there, adjust with the introduction of new components to complement the solution (Arduino®, thermocouple amplifiers, power supply, and relay module).

This work was carried out simultaneously with another master's dissertation work, focused on modeling the phenomena of heat transmission in the water-cooled barrier system. The FireProtect project partially funded this research work – Systems for the Protection of People and Critical Elements Exposed to FF, Ref. CENTRO-01-0246-FEDER-000015, through of the Portugal 2020 program – European Fund for Regional Development.

1.5. **Outline**

This dissertation is organized as follows:

1. The first chapter deals with the introduction and motivation to this research, the objectives set, and the methodology employed. A study of similar state-of-the-art systems is presented, which lay the foundations for this research work.
2. The second chapter develops the approach followed for the control of the water-cooled active system. Also, it depicts the general scheme for the closed-loop control, and a detailed analysis of the equipment used.
3. The third chapter explains the details of the management software developed, specifying the methods employed. It also provides notions about the function of each command available to the user.
4. The fourth chapter presents the results and discussion of the laboratory experiments performed to test the system with and without active control of the water flow under real fire conditions.
5. Finally, the last chapter presents a summary of the work, critical considerations, and restrictions of this work.

2. ACTIVE WATER COOLING HARDWARE SYSTEM

2.1. Introduction

This chapter details the hardware installed and used in the fire barrier to achieving the active and closed-loop control of the water cooling system. The new setup aims to allow the remote control and management of the water flow in the cooling system, in manual or fully autonomous mode, according to the temperature on the fabric surface.

Figure 2.1 depicts the simplified scheme of the proposed water flow control with some of the equipment to be included such as, thermocouples, proportional solenoid valve, and the control hardware – responsible for acting on the valve position according to the temperature values that are obtained by the thermocouples. This new hardware will be analyzed in detail in the next subchapters.

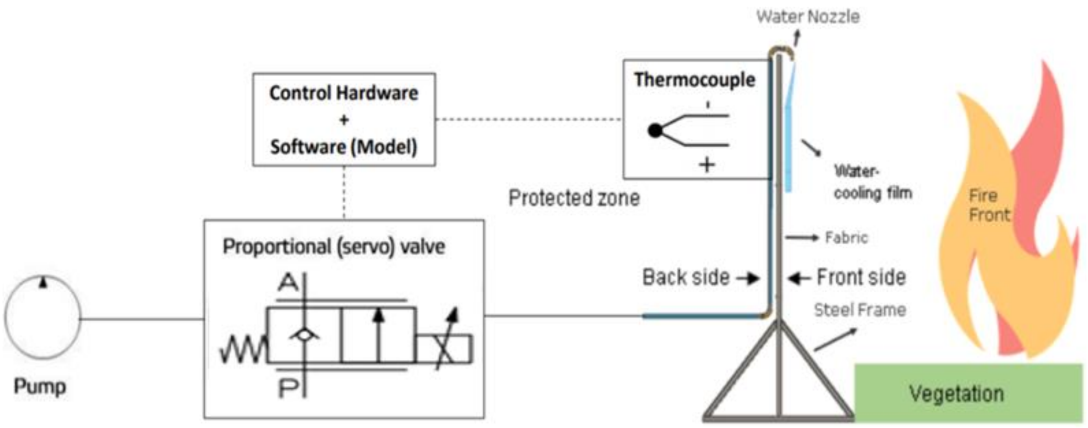


Figure 2.1 – Barrier with water distribution control system, (Viegas, et al., 2019)

2.2. Proposed scheme for closed-loop control

In Figure 2.2, the wiring scheme for the closed-loop control proposed is shown. The four red LEDs indicate the load circuit state on the relays.

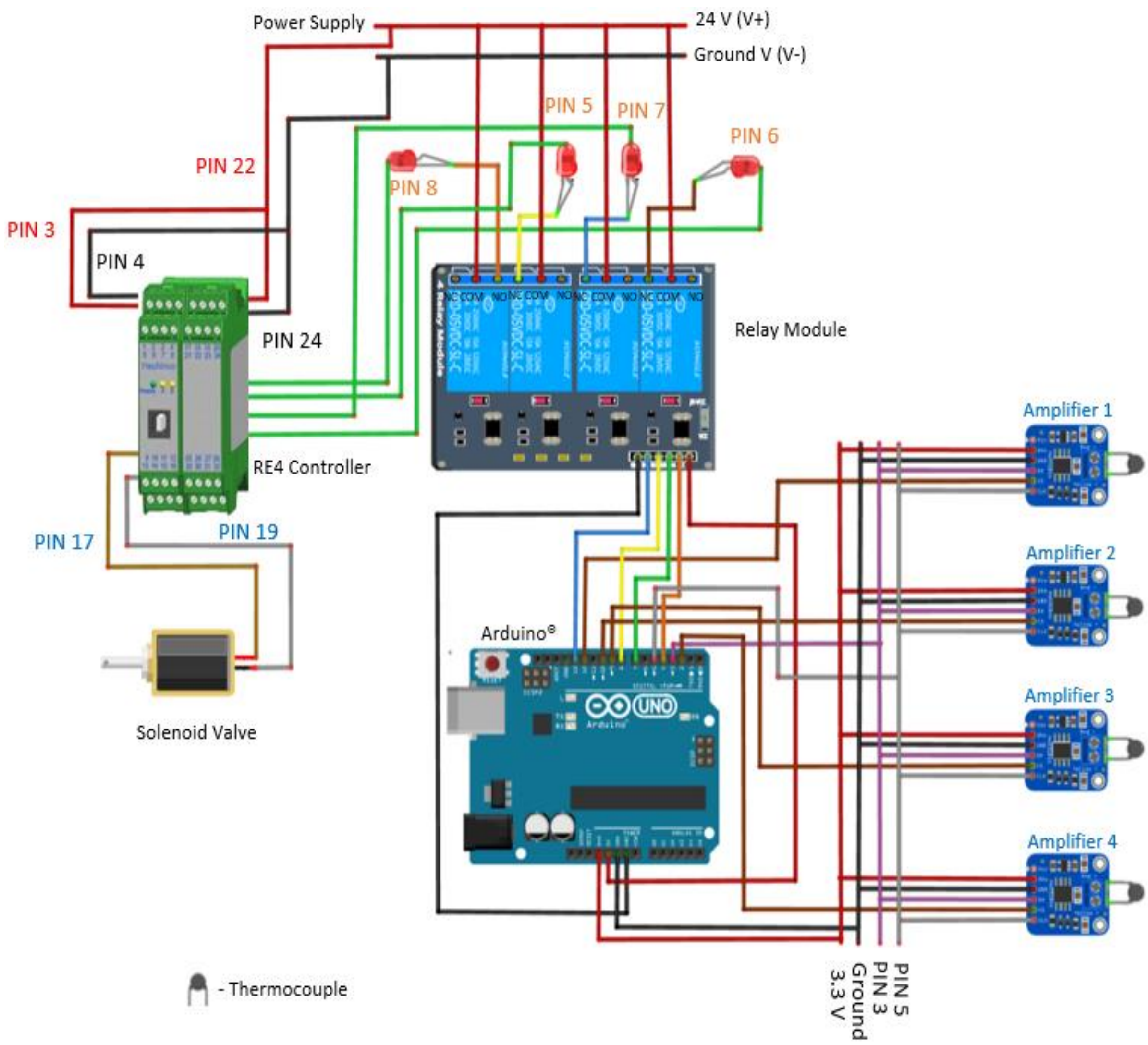


Figure 2.2 – Wiring scheme for proposed closed-loop control

The control of the valve position as a function of temperature will be achieved as follows: in the fire-proof barrier, the fabric surface temperature value will serve as a trigger to activate the water-cooling system. Consequently, four thermocouples were installed on the fabric – 1.2m length per 1m height – to obtain a reliable measure of its temperature to decrease the chance of failure or false alarms (false positives/negatives). Since the voltage signal generated by the type-K thermocouples is typically in the order of millivolts, it is difficult to distinguish a noise signal from a temperature signal. To overcome that, each thermocouple is connected to a corresponding thermocouple amplifier.

The RE4 unit, responsible for commanding the solenoid valve, possesses a software where it is possible to control and supervise in real-time the valve parameters. This equipment can be operated using external 24 VDC input signals and power feed. In order to source this power, a 100W power supply was used. However, this unit cannot be easily programmed or remote-controlled. The goal of this work was to develop a user platform (Server), to consent the user to command and supervise the system operation. The solution found was to use an Arduino® board as hardware control.

Consequently, the temperature values obtained by the thermocouples are sent to the Arduino® and are used as inputs for the valve position control program. In this program, several maps can be used, which relate the maximum temperature of any of the four thermocouples with a predefined valve position. The basic principle is, the higher the temperature, the more the water flow required to cool the fabric, thus the more the valve is opened.

Since the Arduino® produces a digital signal with a maximum voltage of 5 VDC and the valve controller (RE4) requires four digital signals of 24 VDC as inputs, a module with four relays was used. In this way, the RE4 controller reads the received digital signals, translating this information into an instruction for the valve solenoid allowing the valve movement to the desired position.

2.3. Detailed equipment overview

In this subsection, the working principle of each hardware component introduced in the fire barrier cooling system for the active and autonomous control of the water flow is explained.

2.3.1. 2/2-way proportional seat valve

The hydraulic valve used was a 2/2-way proportional seat valve, model number 6244270, manufactured by Hauhinco. This valve acts through a proportional solenoid[2] incorporated into its structure, which, in turn, is commanded via the Hauhinco control unit "RE4 Controller". This unit sets the degree of opening of the valve by controlling the current sent to the solenoid.

Referring to Figure 2.3, the valve operation method is as follows: through the controller, a digital signal is sent to the proportional solenoid (1). At that moment, when the current flows through the solenoid, an electromagnetic force is induced and acts through the

[2] A solenoid is a kind of coil, and it is used in hydraulic closed-loop systems in order to have high control precision of the position of the valve.

lever (2). Due to this, the tappet (3) will move the ball (4) and press it out of the valve seat (5). In this way, the communication channel between the inbound route (P) with the exit route (A) is open. Finally, the covers (6) have the function to support the flange seals and limit the maximum admissible volume flow.

Depending on the arrangement of the ball (4) and valve seat (5), the valve will have its standard position as Normally Closed (NC) or Normally Open (NO). In this case, this valve has the NC configuration, meaning that it remains fully closed when no signal is being given to the solenoid.

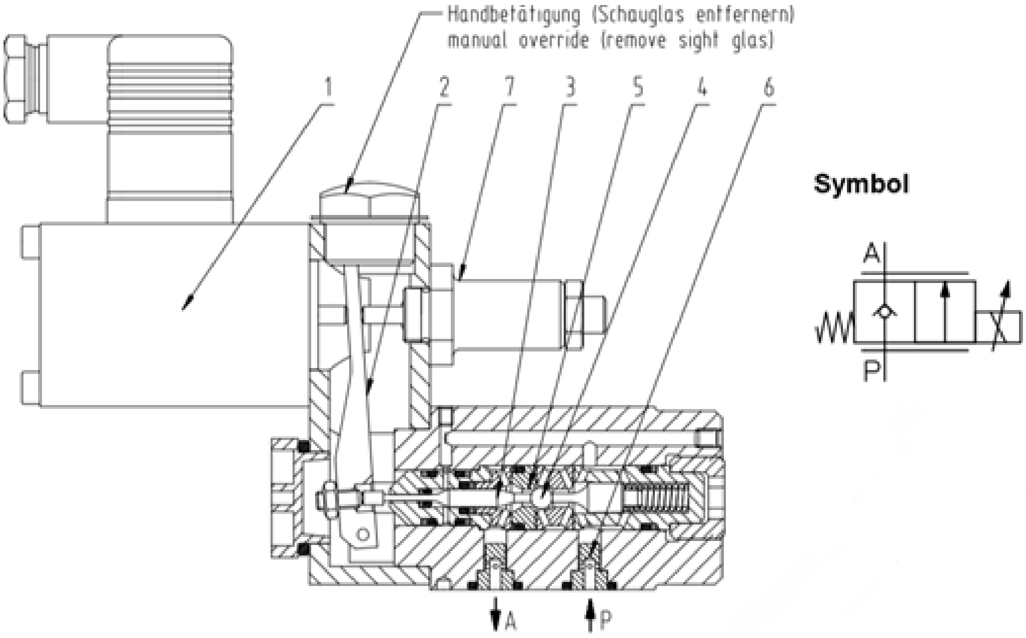


Figure 2.3 – Valve operating scheme

This valve supports DN 10 (17.2 mm, according to ISO Nominal Pipe Sizes) inlet (P) and outlet (A), at a maximum pressure of 320 bar and with a maximum flow rate of 40 l/min. Since the fluid used will be water, it is recommended that its temperature is within 5 to 50°C. Concerning power requirements, this valve has a power consumption of 21 W and requires 24 V of direct current (24 VDC) for its operation. Further specifications can be found in the valve technical datasheet included in ANNEX A.

2.3.2. RE4 controller

The RE4 controller, explicitly designed for Hauhinco, produces the control electrical signal for the valve solenoid, to act on valve position or pressure control. It comes with its own software – HAUHINCO 3.6 – that permits the user to change the valve control variables. Through the software, it is possible to save up to 8 different digital current signals corresponding to 8 different valve positions, between fully closed and fully open, inclusive. It has a current output parameterizable up to 2.6 A and can manage up to 2 different valves.

The RE4 controller must be connected to a minimum 60W power supply, providing 24 VDC. Once programmed and powered, the controller will operate according to Figure 2.4: the RE4 unit (1) receives from the user the message about what will be the next position (2). At that moment, the controller processes this information and sends a specified current to the solenoid, corresponding to the new position (3). The connection between the valve and the controller will be made through two wires.

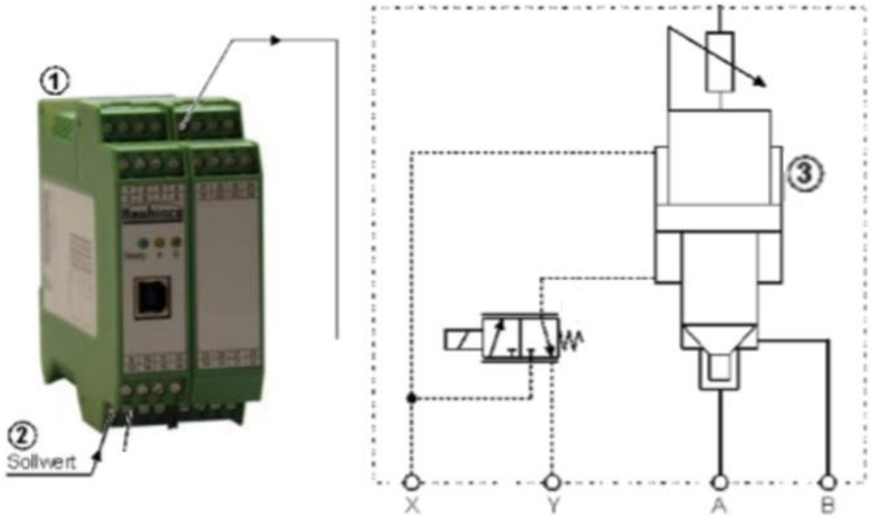


Figure 2.4 – Schematics of the RE4 controller, Hauhinco manual

This controller has a wide range of pins that permit, for example, working with analog signals, changing mode of operation, or even control position or pressure of two separated valves. In this application, four pins will be used to power the RE4 unit – Pins 3, 4, 22 and 24 – two others will connect to the valve solenoid – Pins 17 and 19 – and finally, four pins that will work as the controller inputs – Pins 5, 6, 7, and 8. Table 2.1 summarizes the pin layout for the RE4 unit.

Table 2.1 – RE4 controller input/output pins

	Pin connection	Description
Supply	3	Power supply
	22	Power supply power stage
	4	GND connection supply
	24	GND connection power stage
Solenoid outputs	17 and 19	PWM output solenoid A
Digital inputs	8	Enable input
	5, 6 and 7	Digital inputs for binary code

The digital input pins 5, 6, and 7 will be used to receive the binary signal input for the 8 (2^3) valve positions. In this case, it will be the Arduino® board responsible for generating the enabled (1) or disabled (0) signals through these three pins. The corresponding binary code for each valve position is present in Table 2.2. The order contained there will be respected, which means that the demand value S:0 will correspond to the position fully closed in the valve, gradually opening until value S:7, where the valve will be fully open. Is should be referred that the digital signal received by pin 8 will be the enabled signal (1) and will serve as a trigger for the RE4 controller to activate pins 5, 6, and 7 as inputs.

Table 2.2 – Information on the connections in the relays

			Pin		
		Position	5 (= S1)	6 (= S2)	7 (= S4)
Demand value	S:0 - 0%	Closed	0	0	0
	S:1 - 14,3%	1^{st} position	1	0	0
	S:2 - 28,6%	2^{nd} position	0	1	0
	S:3 - 42,9%	3^{rd} position	1	1	0
	S:4 - 57,2%	4^{th} position	0	0	1
	S:5 - 71,5%	5^{th} position	1	0	1
	S:6 - 85,8%	6^{th} position	0	1	1
	S:7 - 100%	Fully open	1	1	1

In Figure 2.5, the operation diagram of this controller is schematized. In addition to referring the pins that will be connected, several functions are also mentioned that will be used in the controller program, detailed in the next chapter. From left to right, the four 24 VDC digital signals will reach the controller´s input pins 5, 6, 7, and 8 via the relay module connected to Arduino® board. After the RE4 unit receives the signals, a current (ia) corresponding to a specific position is generated and supplied to the solenoid – through pins 17 and 19 – to adjust the valve position. Finally, the power stage of the controller is connected in parallel to the power supply by pins 22 and 24, which also supplies the internal controller through pins 3 and 4.

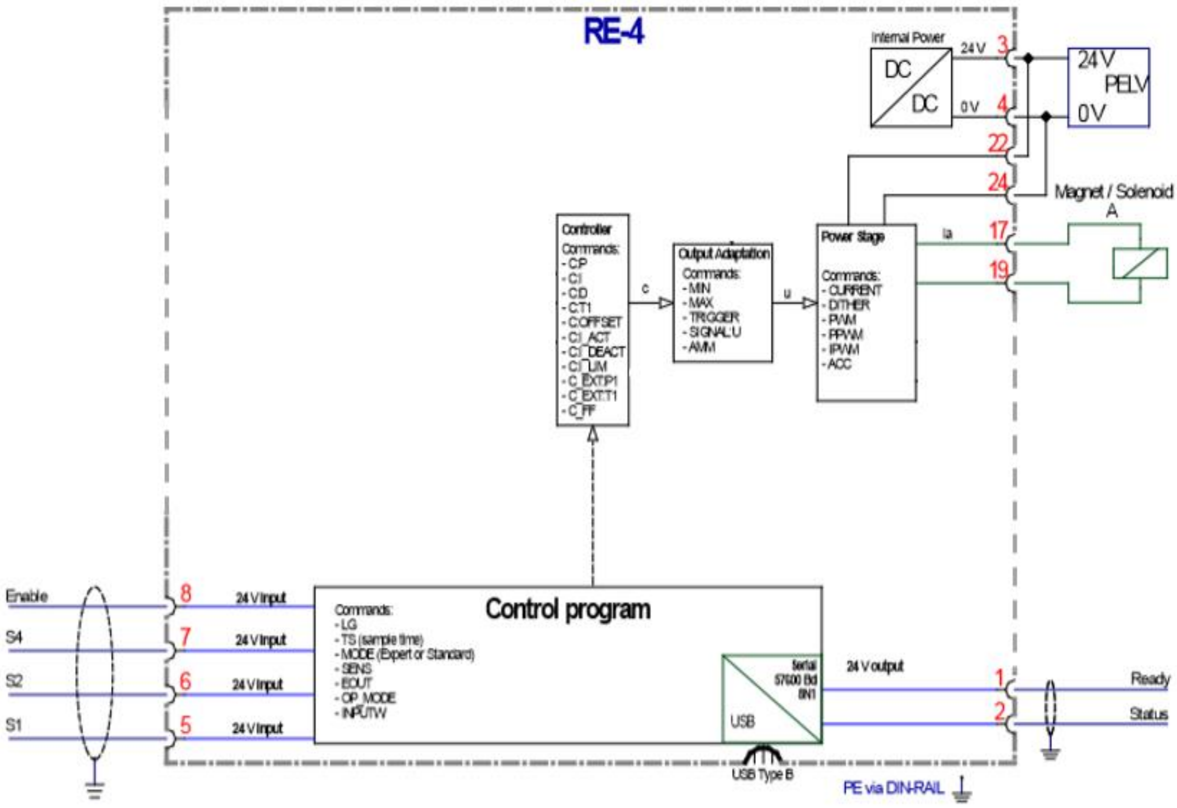


Figure 2.5 – Operation diagram of the RE4 controller, Hauhinco manual

For further information, the RE4 unit datasheet is included in ANNEX A.

2.3.3. Arduino® Uno Rev 3 board

The Uno Rev 3 board, depicted in Figure 2.6, is a low cost and robust solution found for the system´s closed-loop control, manufactured by Arduino®. This board has an Atmega328P microprocessor with 32 KB of memory, a 16 MHz clock speed, 14 digital pins, including 6 PWM outputs, and 6 analog inputs. This board works with a supply voltage between 7 and 12V and can transmit output signals with a maximum voltage of 5 VDC.

Figure 2.6 – Arduino® Uno Rev 3 board

This microcontroller is also able to supply other electronic devices through its 3.3V and 5V pins. These two pins will supply the thermocouple amplifiers and the relay module, respectively.

The communication between the Arduino® board and the computer is achieved through USB cable. The Arduino® Uno Rev 3 board complete specification sheet is available in ANNEX A.

2.3.4. Relay module

A relay is an electrically operated switch constituted by a coil. When an electrical current crosses the coil, it generates, through its electromagnetic field, a force capable of changing its state from open to closed or vice versa, thus opening or closing the load circuit. These devices are typically controlled by a low voltage signal on the control circuit (low voltage side), much lower than the voltage of the load circuit (high voltage side). In this case, the voltage of the control circuit and load circuit will be 5 VDC and 24 VDC, respectively.

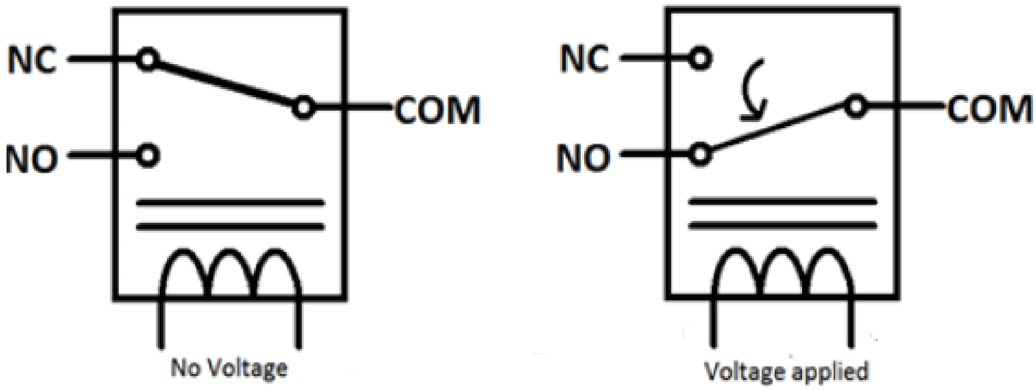


Figure 2.7 – Representative scheme of the relay operation

The relay module used, model SRD-05VDC-SL-C, manufactured by SONGLE ®, requires six input pins connected to the control circuit: VCC and Ground to power up the module and four digital signals coming from the Arduino®, connected according to Table 2.3. On the load circuit, each relay has three sockets for two configurations: common (COM), normally closed (NC) – in this configuration the relay will be closed by default, meaning the current is flowing unless a signal is sent to the relay in order to open – and normally open (NO), precisely the opposite of what was described to NC configuration. One of these wires is always connected to COM socket, and the other must be connected to one of the two remaining sockets (NC or NO). Only the signal coming from pin 13 of Arduino® will have the NO setting since RE4 controller pin 8 must always be enabled so that the other 3 (therefore pins 5, 6, and 7) are recognized as inputs.

Table 2.3 – Information on the connections in the relays

<table>
<tr><th></th><th>Arduino®</th><th>Relays (low voltage)</th><th>Relays (high voltage)</th><th>Controller RE4</th></tr>
<tr><td rowspan="6">P
I
N
S</td><td>Ground</td><td>Ground</td><td>-</td><td>-</td></tr>
<tr><td>5 V</td><td>VCC</td><td>-</td><td>-</td></tr>
<tr><td>PIN 13</td><td>IN 1</td><td>NO and COM</td><td>PIN 8</td></tr>
<tr><td>PIN 8</td><td>IN 2</td><td>NC and COM</td><td>PIN 5</td></tr>
<tr><td>PIN 7</td><td>IN 3</td><td>NC and COM</td><td>PIN 7</td></tr>
<tr><td>PIN 4</td><td>IN 4</td><td>NC and COM</td><td>PIN 6</td></tr>
</table>

Please note that the COM socket will be connected to the power supply, providing 24 VDC (V+). The other socket (NC or NO) will connect to the respective RE4 controller pin. Figure 2.8 presents the wiring scheme between the Arduino® and the relays – the four red LEDs indicate the state of each relay load circuit.

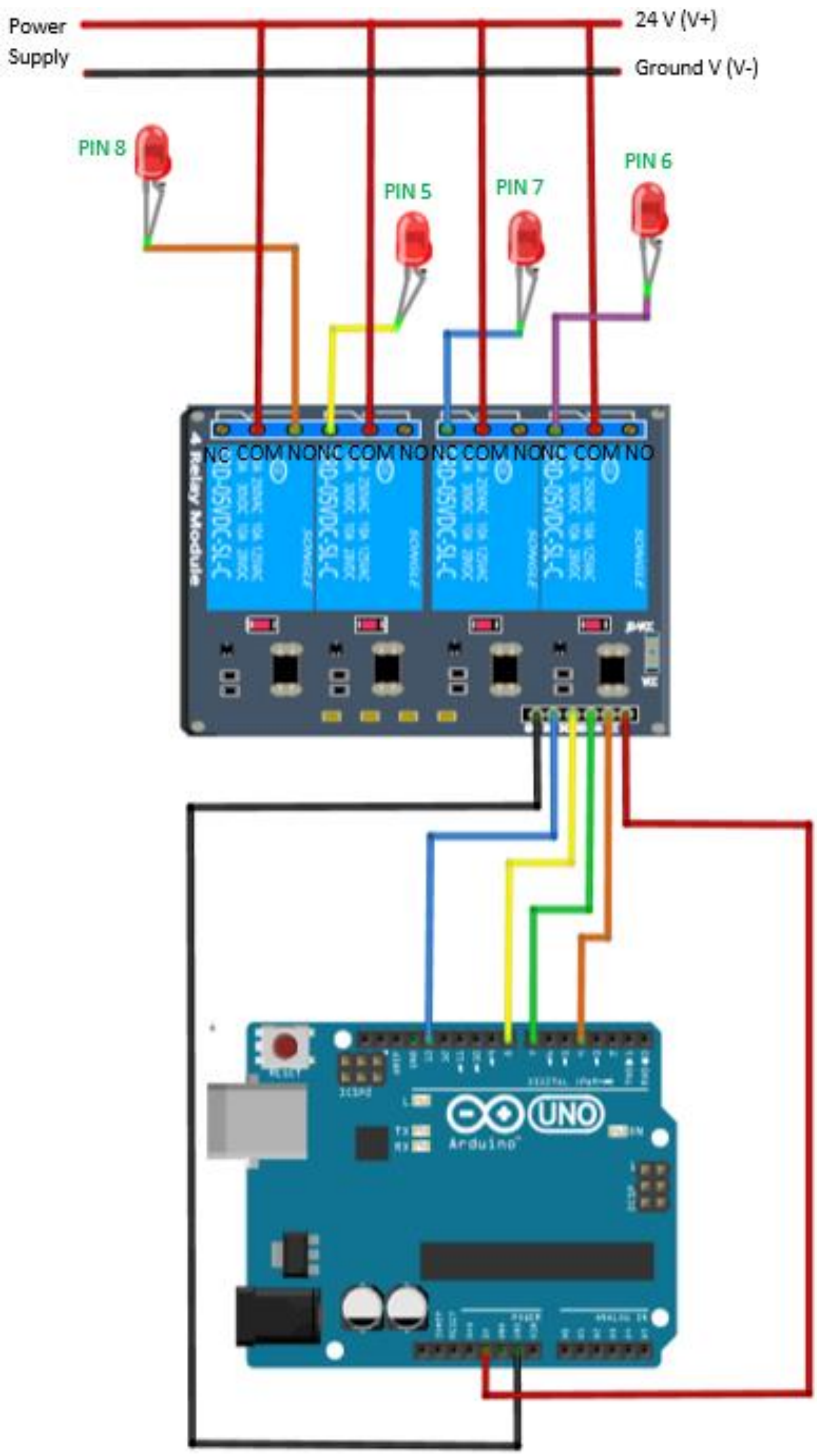


Figure 2.8 – Scheme of the connection between Arduino® and controller via relays

The relay module complete specifications sheet is available in ANNEX A.

2.3.5. Power supply

A power supply includes a series of parts such as transformer, rectifier, filter, and a voltage regulator that allows it to convert alternating current (AC), available in electrical conventional household power sockets, in direct current (DC) necessary to power the RE4 unit and COM socket of the relays. For the proposed setup, a power supply unit of 100W, with two voltage outputs of 24 VDC @ 4.5 A, model RS-100-24 manufactured by Mean Well, was employed. This device is capable of supplying power both to the RE4 controller (60W) and valve solenoid. This unit is depicted in Figure 2.9.

Figure 2.9 – Power supply used, showing its connections

Notice that a battery can also be used as a backup power supply in case of power grid failure. However, this redundancy was not implemented in this work. The power supply unit complete specifications sheet is available in ANNEX A.

2.3.6. Thermocouples

Four K-type thermocouples (Nickel-Chromium/Nickel-aluminum) were used to measure the temperature on the fabric surface.

A thermocouple consists of two wires made from different metals that are welded together, creating a junction where the temperature is measured. When this junction experiences a change in temperature, a voltage is created, usually following a linear relation, as depicted in Figure 2.10. Arduino® libraries then process this voltage signal, and its translated to a temperature value. This technology is typically selected because of its low cost, wide temperature ranges, and durable nature. These thermocouples possess a range grid between -200 to 1250 ºC, and an accuracy of 1.5 ºC or 0.4%, whichever is greater.

As for K-type thermocouples, the higher the temperature in contact, the higher the value of the generated voltage, and consequently, the more accurate the thermal measurement will be. Still, about this graph, thermocouples type N and J for presenting a good compromise between the maximum limit of supported temperatures and generated voltage value could also be selected.

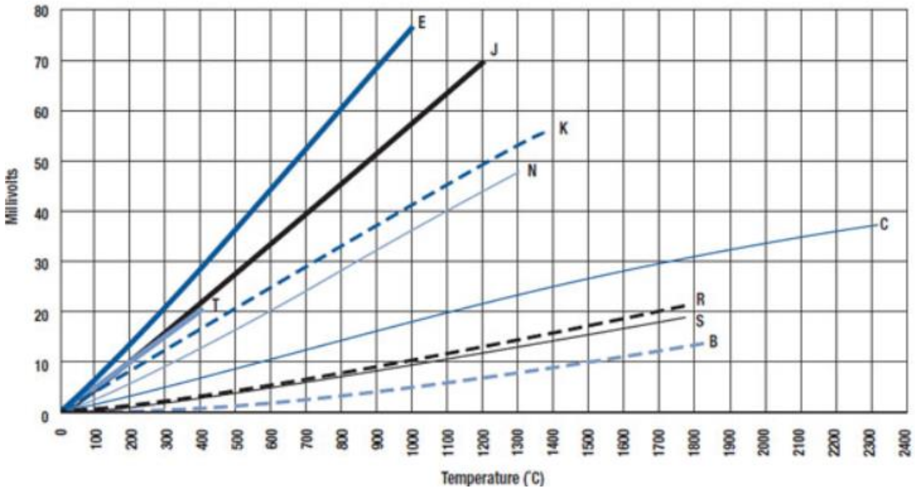


Figure 2.10 – Response of the different types of thermocouples

2.3.7. Thermocouple amplifiers

In this work, four thermocouple amplifiers, model MAX31855 from Adafruit, depicted in Figure 2.11, were employed. Each amplifier is directly connected both to the thermocouple and the Arduino®.

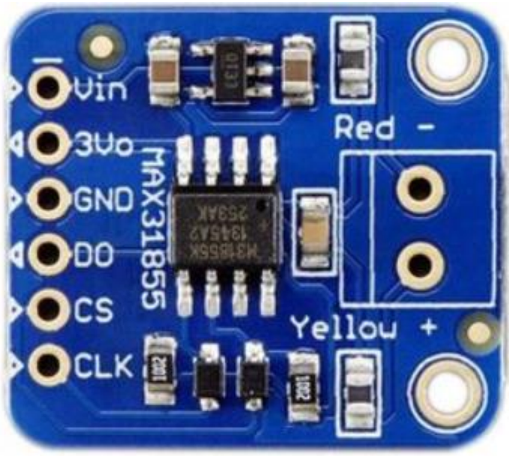


Figure 2.11 – The thermocouple amplifier MAX 31855 used

The thermocouple negative (usually red) and the positive (usually yellow) wires are linked to the two connections on the right of the board. On the left side, there are six pins, five of which are connected to Arduino®, according to Table 2.4.

Table 2.4 – Information on the connections in the amplifiers

	Arduino®	1 - MAX 31855	2 - MAX 31855	3 - MAX 31855	4 - MAX 31855
PINS	Ground	Ground	Ground	Ground	Ground
	3,3 V	3Vo	3Vo	3Vo	3Vo
	PIN 3	DO	DO	DO	DO
	PIN 5	CLK	CLK	CLK	CLK
	PIN 12	CS	-	-	-
	PIN 10	-	CS	-	-
	PIN 9	-	-	CS	-
	PIN 2	-	-	-	CS

Regarding the temperature measuring process, this will be done with a 10ms interval between consecutive measurements (acquisition rate of 100Hz). It is a short time interval, but in this way, it allows the DO pin (Data Out, responsible for carrying the information from the thermocouple amplifier to Arduino®) and CLK pin (Clock, an input to the amplifier that once activated, instructs to present the data) to be shared among all four amplifiers. This configuration makes it impossible to obtain all four thermocouple readings simultaneously. Finally, the CS pin (chip select) serves to identify each amplifier, allowing to select the one that is currently being requested for data. Therefore, if read instruction is given to the thermocouple amplifier 1, CS pin (pin 12 of Arduino®, according to Table 2.4) will be activated together with CLK (pin 5). From this conjugation, the DO pin (pin 3) will carry the amplifier 1 data to Arduino®.

Finally, this device has the possibility of being powered by 5V, when linked to Vin, or 3.3V if connected to 3Vo. In this application, the power supply used was 3.3V. The wiring diagram for the Arduino® board and the four amplifiers is shown in Figure 2.12.

The amplifier board complete specifications sheet is available in ANNEX A.

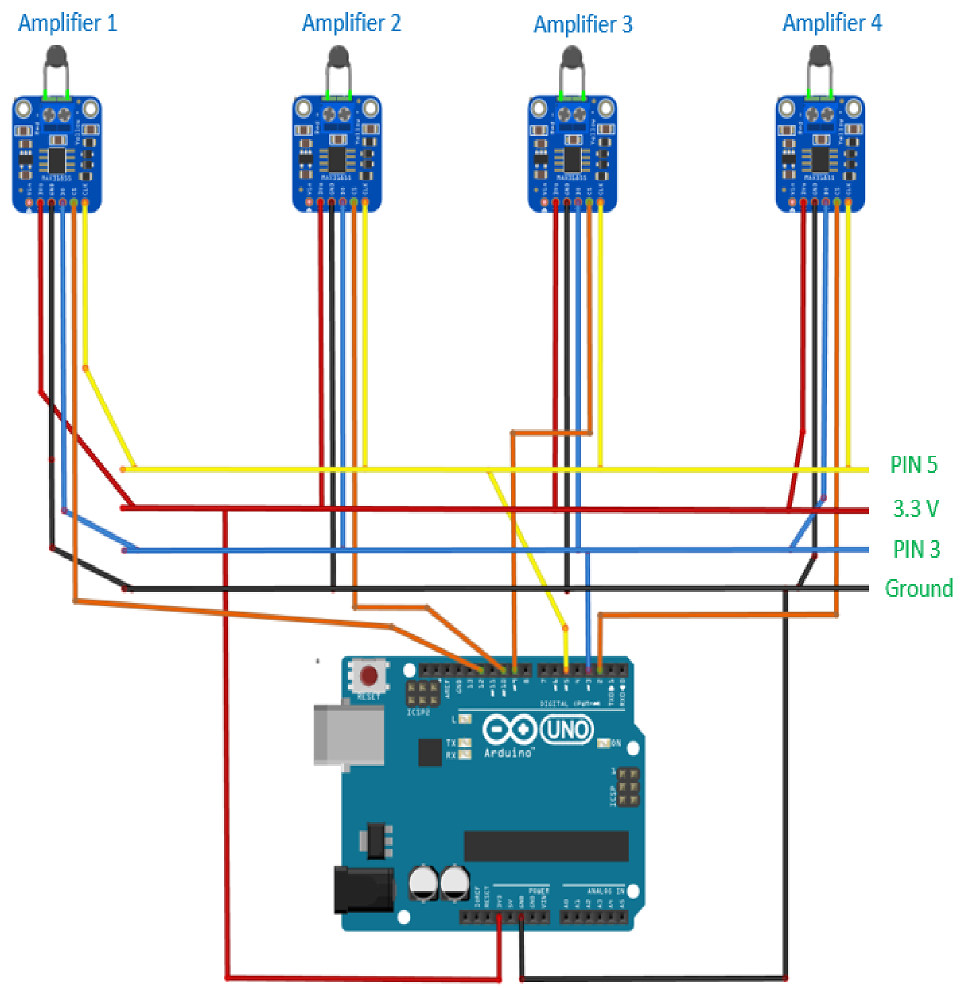


Figure 2.12 – Detail of the wiring scheme between the Arduino® and the four thermocouple amplifiers.

2.4. **Discussion**

The biggest challenge during the accomplishment of the system control hardware was to understand how the information is exchanged between the valve solenoid and the RE4 controller. Besides, we took too much time to understand the connection between the relay model with the other devices. After this, the assembly process went smoothly and without significant difficulties.

As for negative points, the level of complexity of the diagram is above what was expected, which is something not desirable for a final solution since it becomes more prone to failure. Some improvements to the current configuration might be:

- a 24VDC Power Line Communication (PLC) instead of the Arduino®, avoiding the need for a relay module.
- Replace the thermocouples by wireless temperature sensors, increasing the portability of the solution.

3. ACTIVE WATER COOLING MANAGEMENT SOFTWARE

The previous chapter focused on the hardware used for closed-loop control. This chapter focuses on the software that commands the hardware and permits the user to control or supervise the system. One of the objectives set for this dissertation was to create an automatic management software system, with a simple and intuitive interface so any user could easily interact with it and control its most important parameters and functions. This software will control the system according to the conditions set in each software running on both controllers, Arduino®, and RE4 unit. For smooth operation, it is essential to have perfect synchronization between these three programs, Figure 3.1.

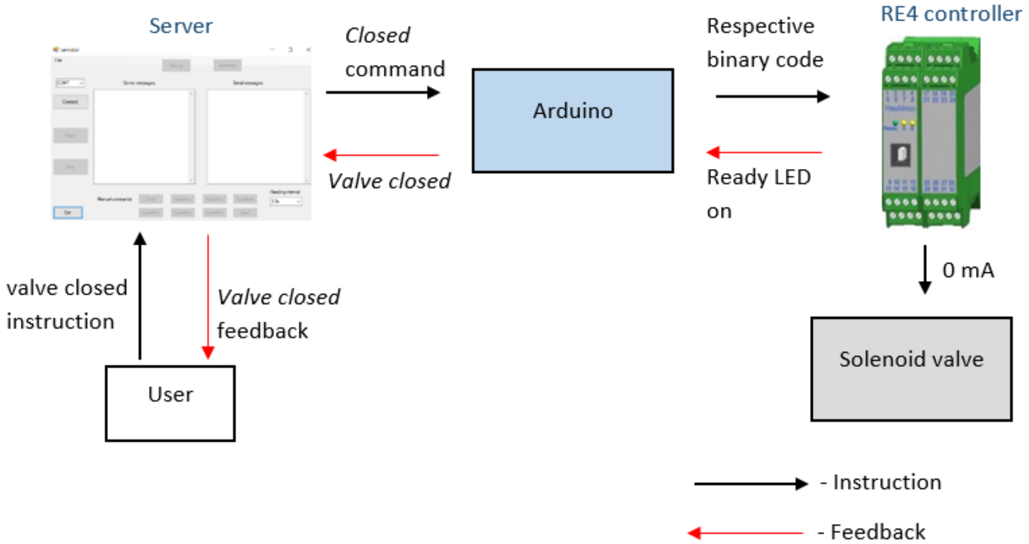


Figure 3.1 – Software´s relation scheme

3.1. RE4 Controller software

The Hauhinco RE4 controller, responsible for commanding the valve, has its proprietary software – HAUHINCO 3.6. This software is not open access and does not allow the inclusion of advanced programming routines. Its purpose is to allow the user to change or chose several functioning parameters of the valve or supervise its status in real-time. In Figure 3.2, the interface of this program is depicted.

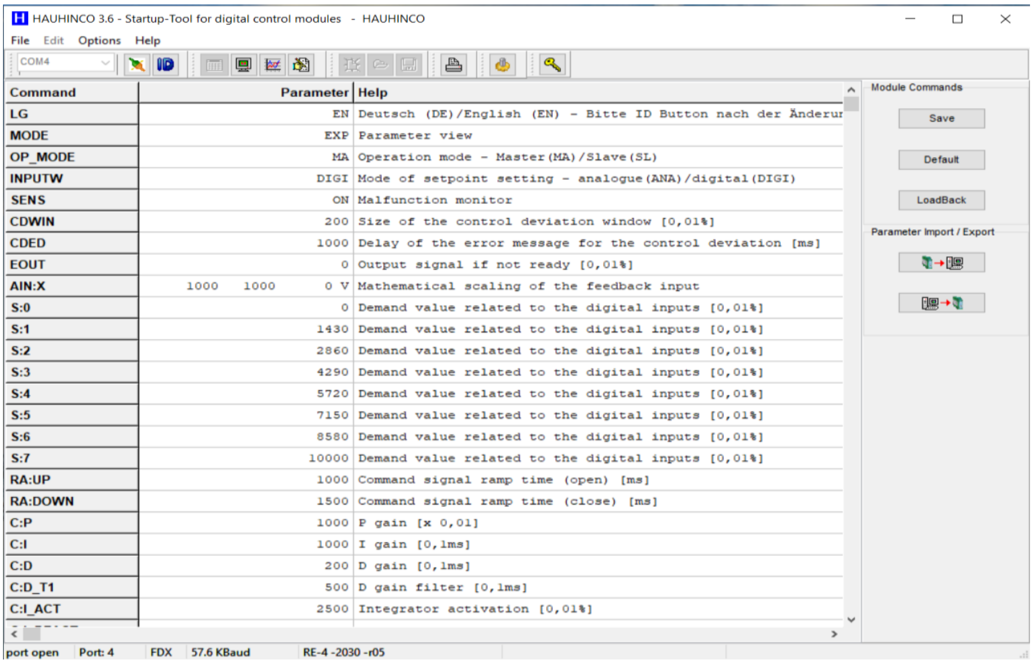


Figure 3.2 – Hauhinco software interface

By default, the software is in standard mode. The expert mode was chosen to gain more control over the system variables. After that, master mode was adopted – in this mode, demand value and actual signal are read, and the current value is calculated depending on the previously parameterization – in detriment of the slave – external equipment drives the valve via analog pins 9 and 10 of the controller, through specified settings. Still, in the basic parameters, the input chosen was a digital signal, consequently allowing pre-defined binary code to be processed.

The RE4 unit can be programmed to manage up to two valves, controlling the pressure or position of each valve. It is through the output signal adaption parameters that the information is transmitted to the software that it will have the function of managing only the position of one valve.

The output *Current* command is part of the power stage parameters. This command has been assigned with 2.5 A. This value is only sent to the solenoid when the instruction for valve total opening is received. Otherwise, a fraction of this number is sent, corresponding to the desired opening level, following a linear response depicted in Figure 3.3.

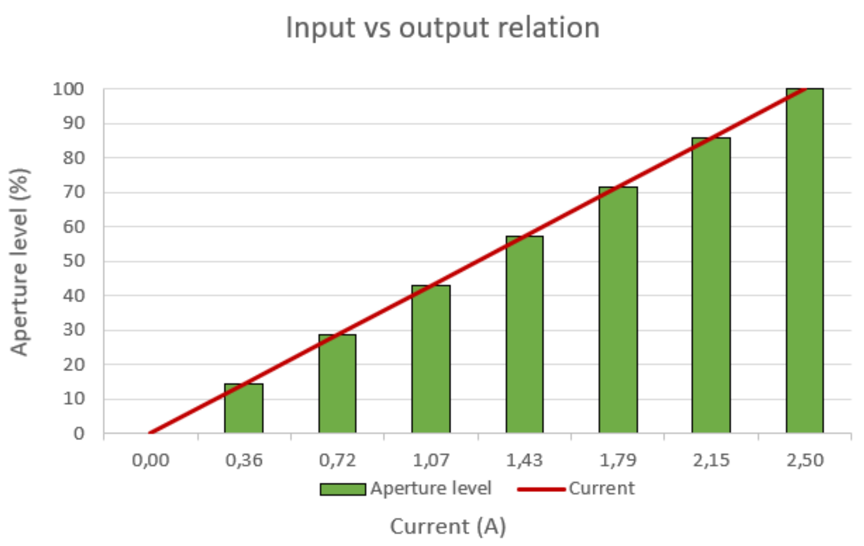


Figure 3.3 – Response curve of the valve solenoid according to the input signal current

The valve aperture levels for each input instruction have been previously defined in the input signal parameters. These levels are directly related to the output current value this way: fully closed position, corresponding to "demand value S:0" instruction, has been assigned the value of 0%, meaning that is no current required to the solenoid in order to achieve this valve position. This effect happens because the valve is normally closed type. The following seven positions (from 1 to 7), represent successive increases of 14.3% on the valve aperture, up to the instruction "demand value S:7" where the opening is total and the current supplied is maximum (2.5 A).

For safety, it was defined that in the event of an error (for example, a deactivated enable input) also the output and "ready" LED are switched off, thus allowing the valve to remain in its actual position. This way, the water keeps flowing even if the control hardware is compromised.

Figure 3.4 depicts the HAUHINCO 3.6 software running and controlling the valve in real-time. The instant captured in this figure, the controller has just received information that the S2 pin has been activated (small red square box on the right), which according to the binary code, corresponds to demand value S:2, meaning the second aperture level of the valve (28.6%). Automatically, the controller acknowledges this information and produces an output current signal (IA, the green square box on the left) with a value of 702 mA that corresponds to the 28.18 % fraction (this discrepancy on the input and output values is due to signal transmission losses) of the 2.5 A assigned to the *Current* command. This set of actions, when performed without any error, activates the *Ready* LED (blue square box) and successfully commands the valve.

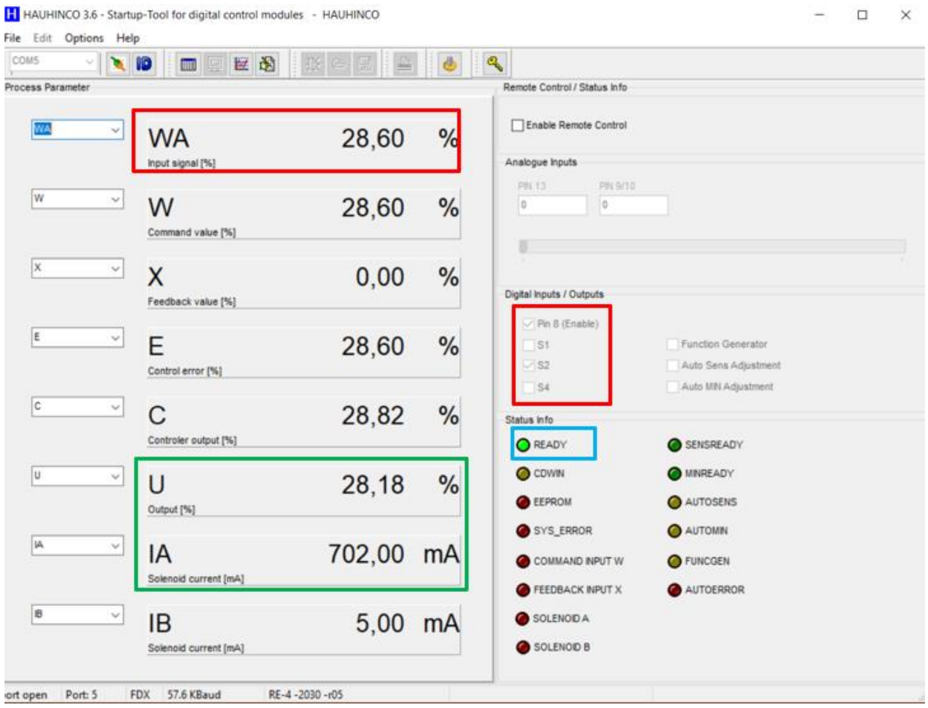


Figure 3.4 – Behavior of the program in operation

The adjustment between the input signals and the controller response is made through an integrated PID controller, which improves the transient response or relative stability by combining three components: proportional, integral, and differential. The response of the m(t) control signal is given by equation 3.1.

$$m(t) \ = \ kp * e(t) + kd * \frac{de(t)}{dt} + ki * \int_0^t e(t)\,dt \qquad 3.1)$$

The first term of the equation concerns the proportional component, where kp is a constant called sensitivity or gain, and e(t) the error. The second term refers to the differential component, in which kd is a differential constant. As for the last term, of the integral component, ki refers to an integral constant. In conclusion, the software makes it possible to change the values of these three constants, improving stability, decreasing overshoot, or removing the error in the transient regime of output current signal (Pires, 2019). Kp constant was assigned with 100 (only by assigning this number, the current value is given in a ratio of 1:1 with the input signal). Kd and Ki, both were assigned with default values, 2000, and 300, respectively.

3.2. Arduino® software

The Arduino® script program was built in Arduino® IDE software, and its purpose was to provide an automatic and autonomous means for the system to function with no user input. The intention of this script, running in the Arduino® controller, was to receive data from the thermocouples and determine the required water-cooling flow using a pre-loaded map or mathematical model[3].

The Arduino® script was designed for two different operating modes: fully automatic or manual. The *Automatic* mode uses a *WHILE* cycle to receive temperatures continuously from the thermocouples and determine the resulting aperture level of the valve. The cycle ends when the user gives the stop instruction (a *STOP* command was created for this purpose). In *Manual* mode, the program enters in another *WHILE* cycle where the temperature is received and shown in the graphical interface. Then the user can act based on this information and command the aperture of the valve through the buttons on the server interface. Figure 3.5 depicts a print screen of the Arduino® program, showing both operation modes.

[3] In this work, two valve response maps were used, relating the real time temperature readings with the valve aperture opening level. Further details are given in the next chapter. In future work, a mathematical model developed within another Master's degree dissertation of the student David Pereira, which determine the required water-cooling flow as a function of the fabric surface temperature, taking into account the heat transfer phenomena occurring in the fabric, will be implemented.

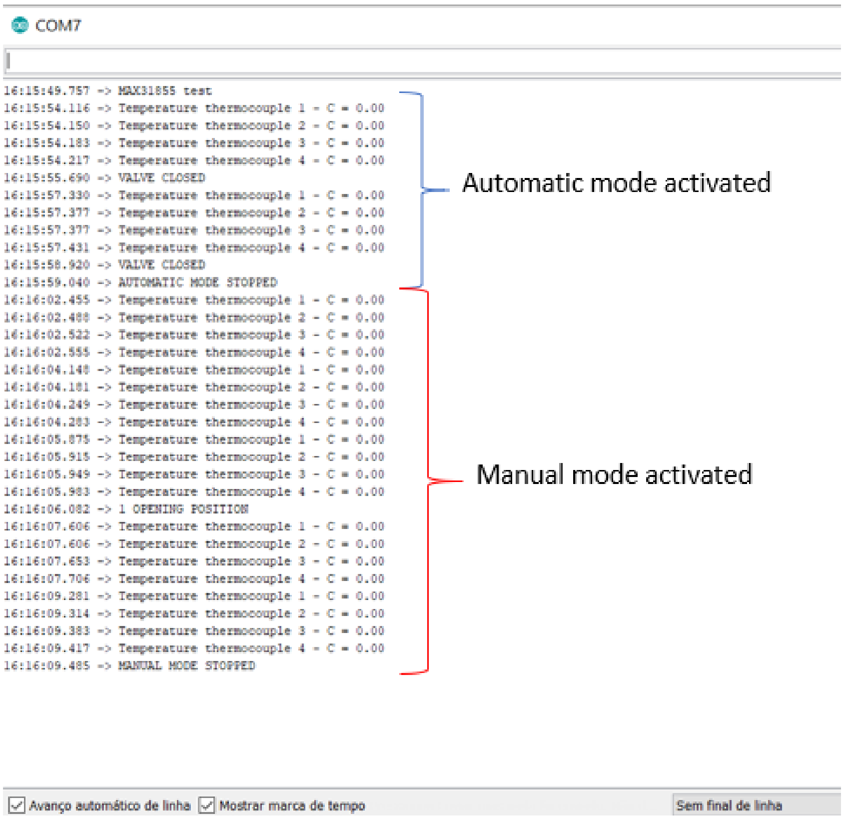


Figure 3.5 – Arduino® program in operation

The *Manual* mode acts as a safety bypass in case there is malfunctioning on the thermocouples or any other subsystem, which may lead to false positives or negatives. However, the *Automatic* mode should be preferred as it is optimized to be the most efficient way of managing water usage while maintaining fabric and barrier integrity. Besides, this system becomes more advantageous because it does not require human intervention during operation.

Furthermore, the program is also designed to overcome possible failures in temperature measurements: if a thermocouple stops returning any reading, the program will alert the user and assume regular operation with the remaining thermocouples. The complete software code can be consulted in APPENDIX A

3.3. **Server software**

The server program, created in C# programming language, acts as an interface between the Arduino® program and the user. Figure 3.6 shows the graphical user interface (GUI) built for this software.

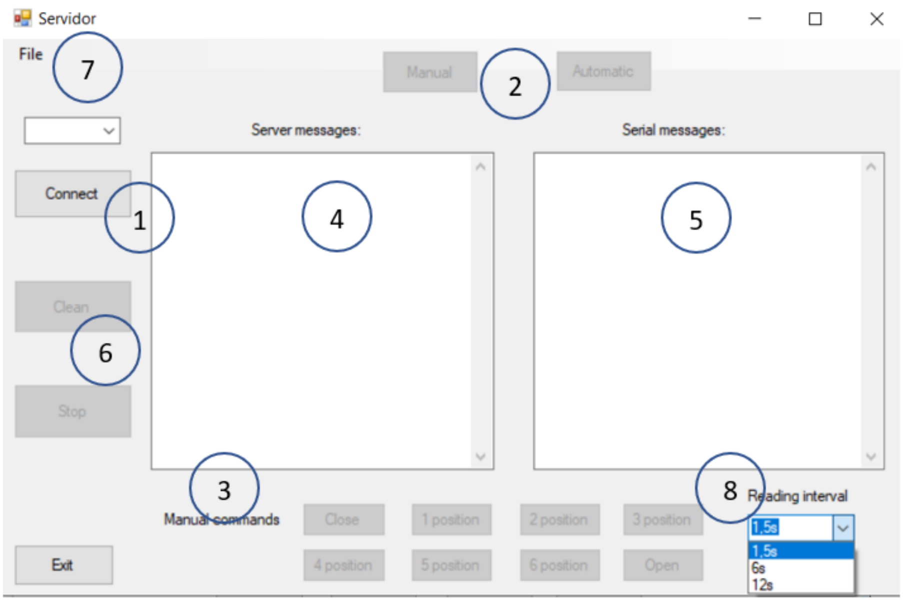


Figure 3.6 – Server user interface

This GUI works as follows: the *CONNECT* button, labeled 1, is programmed to initiate communication between the user and Arduino®. A timer was created so that, every two seconds (2s), runs a subroutine programmed to check if there are new COM-ports connected to the computer, and presenting them in the combo box. When the COM-port that corresponds to the Arduino® board is found and chosen, the communication starts with the *READING INTERVAL* (labeled 8) selected. As a result, the buttons *MANUAL* and *AUTOMATIC* for the operation mode (labeled 2), and the buttons *STOP* and *CLEAN* (labeled 6) became visible.

Under the *Automatic* operation mode, the program will run without any user input. In the *Manual* mode, the "Manual commands" will become visible and can be clicked by the user. These commands force the valve to assume one of the eight pre-defined positions, including fully opened or closed.

In order to avoid problems in the use of the interface, *MANUAL*, and *AUTOMATIC* buttons (labeled 2) will always be available to be selected. However, when chosen, they will automatically give the *Stop* instruction to the system in the first place. For safety, all instructions will only be sent to Arduino® if it is verified that the serial port is open, thus, avoiding any kind of communication errors.

Text boxes labeled with 4 and 5 serve to provide information to the user. The text box "Serial messages" (5) shows the temperatures acquired by the thermocouples in real-time. At the same time, it provides feedback on the system after each instruction. This event was accomplished by using a subroutine, associated with a timer that, at intervals chosen through *READING INTERVAL* (1.5s, 6s ou 12s), displays in this textbox the data received through the serial port. "Server messages" text box (4) records the operation messages and command history, which could be useful to keep track of the system operation.

The menu strip, labeled with 7, when clicked displays the three options that can be seen in Figure 3.7.

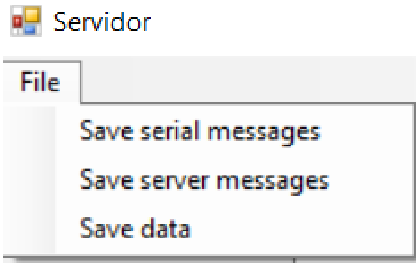


Figure 3.7 – Menu strip saving options

These options are designed to store information about the operation of the system and measured values in text files. The option labeled *SAVE SERIAL MESSAGES* will save all instructions sent from Arduino® presented in "serial messages" text box (5). The option labeled *SAVE SERVER MESSAGES* will save the commands that were sent by the user (contained in the "server messages" text box (4)). Lastly, the option labeled *SAVE DATA* saves to a text file only the temperature values obtained by the thermocouples, thus simplifying the data analysis.

Figure 3.8 shows the GUI during system operation.

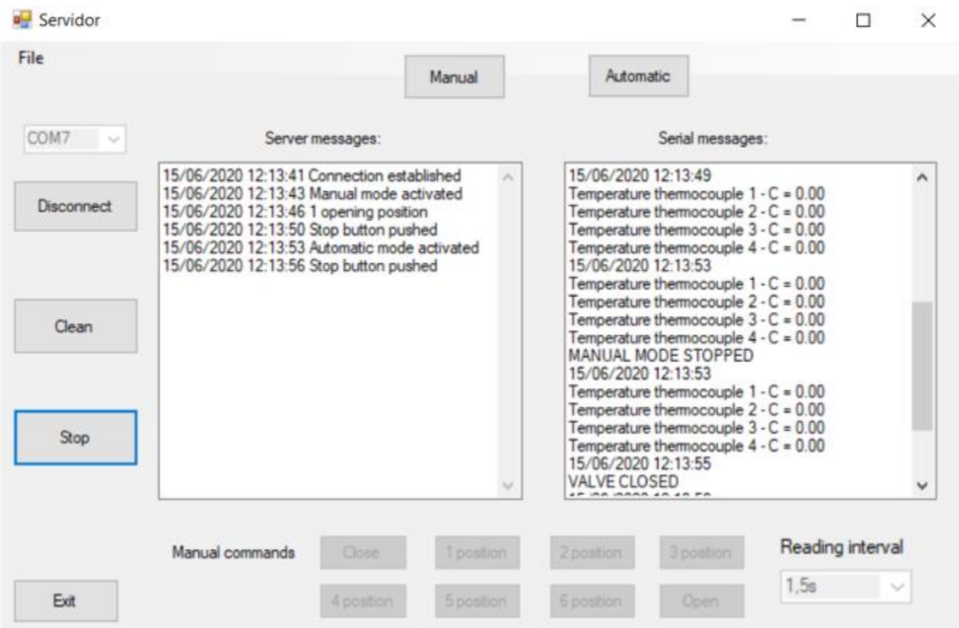


Figure 3.8 – Server program in operation

The complete code for the server program can be found in APPENDIX A.

3.4. Discussion

In this stage, the most considerable difficulty was in the lack of details about the function of the several commands provided by the RE4 unit. To overcome this problem, it was necessary to collect information directly with the valve manufacturer. As a result, the initially planned time for the implementation of this task has been largely exceeded. Moreover, due to the complexity of the system, there were also difficulties in the prevention of errors that may arise from its use.

As an upgrade, we suggest the implementation of control via Wi-Fi®, increasing the complexity of programming but improving the portability of the solution.

4. LABORATORY EXPERIMENTS: RESULTS AND DISCUSSION

4.1. Experimental method

The viability of the solution presented in the previous chapters considers laboratory experiments with real fire.

The experimental setup consisted of a barrier distanced 30cm from the fuel basket, containing 5 kg of shrubs (dry basis) with a moisture content of 13%. This distance ensures a fire intensity equivalent to real conditions, allowing the flame to contact the barrier (Viegas et al., 2020). The fuel basket was filled with 5 kg of shrubs (dry basis), with a moisture content of 13%. Two combustion tunnel ventilators generate an airflow of 2 m/s. Figure 4.1 schematically depicts the experimental setup,

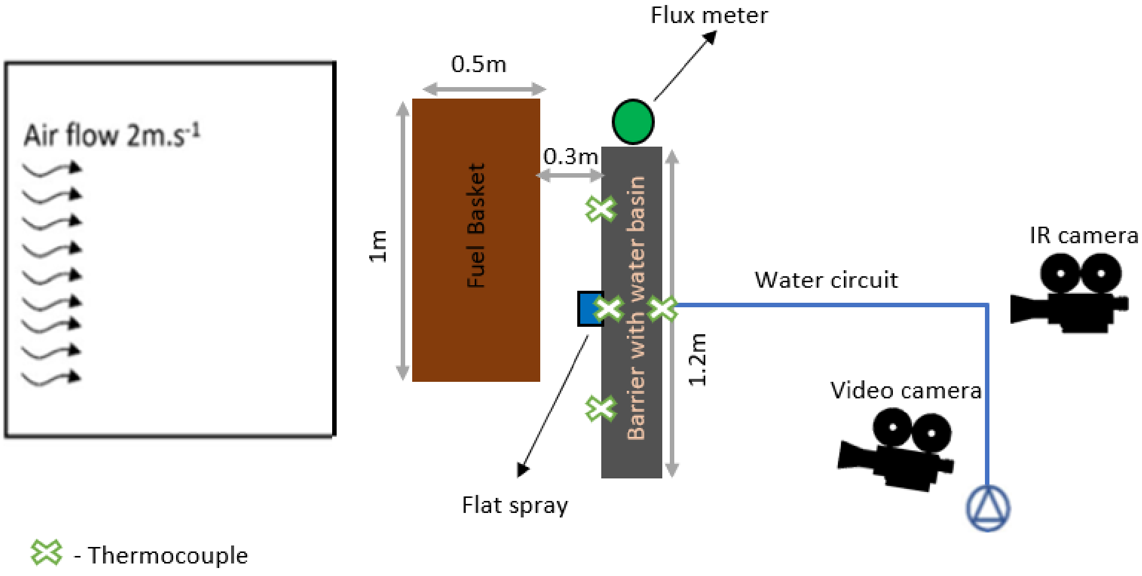
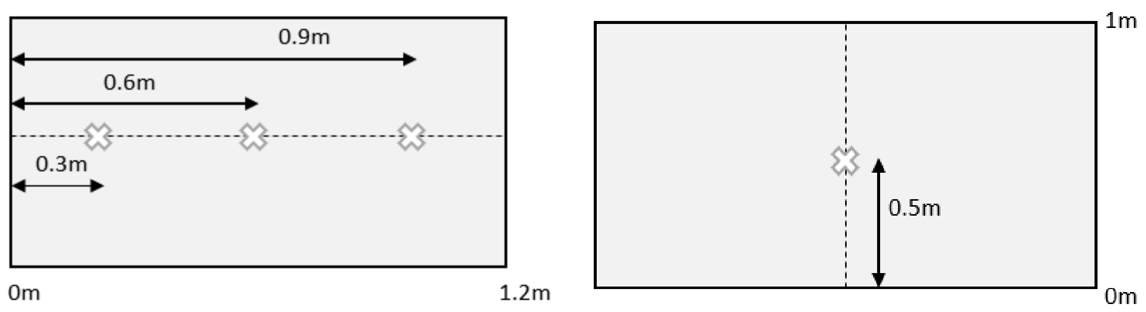


Figure 4.1 – Representation of the experimental assembly

In the fabric, the instrumentation of three thermocouples at the front side (middle-left, middle-center, and middle-right, see Figure 4.2 a) allows monitoring by the system of any change induced by the firefront. A thermocouple instrumented in the back (middle-center, see Figure 4.2 b), can serve as a reference for the IR camera. Finally, a thermocouple in the fuel basket connected to an acquisition board provides feedback on the fire propagation.



(a) (b)

Figure 4.2 – Thermocouples distribution in (a) front of the barrier (b) back of the barrier

Viegas et al., (2020) concluded that a water flow of 1.71 l/min per meter of barrier would be ideal to guarantee fabric integrity. Consequently, the flat spray received 2 l/min of water flow (1.71 l/min/m*1.2m) at a pressure of 2 bar through a water hose connected directly to the valve outlet. The valve inlet was connected to the hydraulic pump. The flux meter, an IR (FLIR Systems, ThermaCAM S Series) camera, and a video camera were also employed. The purpose of the simultaneous use of all these diagnostic equipment is to obtain the maximum information possible from the experiments. Finally, a container under the fabric allows storing the wasted water.

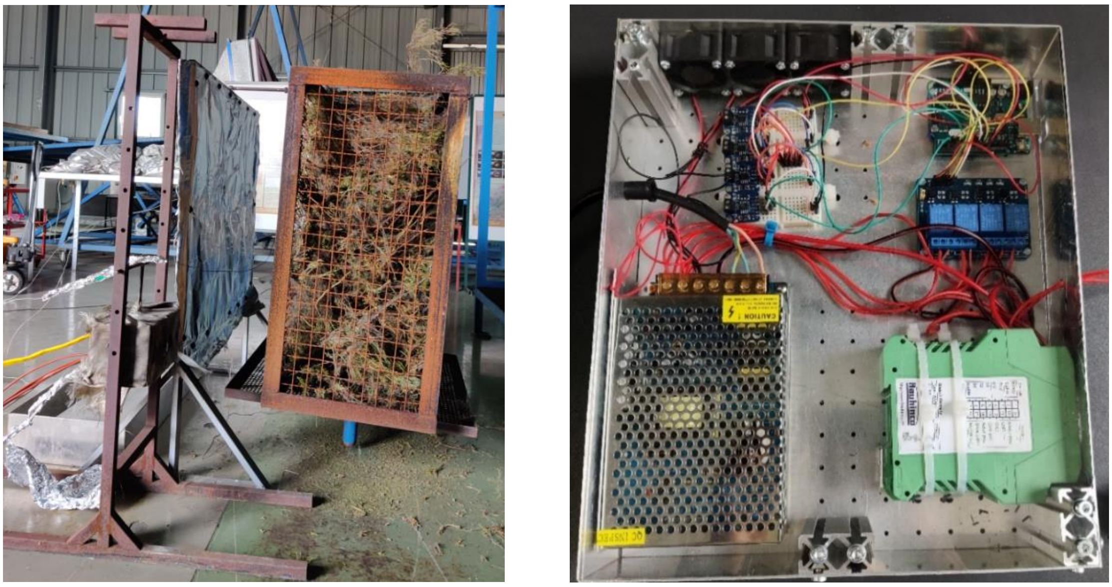
A time limit of 120 seconds was imposed for water cooling functioning and data acquisition. The data acquisition rate was 1/3 Hz (intervals of 3s, i.e., the first 1.5s for temperature acquisition and the remaining 1.5s used to instruct the valve).

All experiments were performed on the same day, under similar ambient conditions, with maximum temperature and relative humidity amplitude between tests of 2.6 ºC and 6%, respectively. Table 4.1 summarizes this information.

Table 4.1 – Relative humidity and room temperature in the experiments

Property	Test A	Test B	Test C
Relative Humidity	59%	55%	53%
Room Temperature	22,5	23,9	25,1

Figure 4.3 a) shows an overall view of the experimental setup prepared for the first test. The installation is the same for all experiments, except for the fabric that changed between trials. Figure 4.3 b) depicts in detail the valve control system.



(a) (b)

Figure 4.3 –a) Experimental setup prepared for ignition (b) Detailed view of the valve control system

Measurements include:

- The volume of water lost during the cooling of the fabric;
- The fabric condition at the end of the combustion process;
- The temperature of the basket to obtain a reference value of the fire temperature, assuming a similar magnitude and evolution in all tests performed;
- Temperature distribution on the fabric.

These measures were performed under three operating conditions of the active cooling system. The three experimental tests consist of:

A. The cooling of the fabric in manual mode with the valve fully open for 120s, which means the flat spray is fully open during the combustion period, immediately after a fire ignition.
B. The valve acted in automatic mode for 120s (according to the values presented in Table 4.1, Conservative Response Map, CRM).
C. The valve also worked in automatic mode for 120s, but with lower temperatures limits to open faster (Aggressive Response Map, ARM).

The values imposed for tests B and C are synthesized in Table 4.2. The experimental procedure employed is detailed in APPENDIX B.

Table 4.2 – Trigger temperatures for tests B and C (CRM and ARM)

	Positions							
	Fully Closed	1st opening	2nd opening	3th opening	4th opening	5th opening	6th opening	Fully Open
CRM	<75 ˚C	75 ≤˚C< 115	115 ≤˚C< 200	200 ≤˚C< 300	300 ≤˚C< 415	415 ≤˚C< 520	520 ≤˚C< 600	˚C≥ 600
ARM	<45 ˚C	45 ≤˚C< 74	74 ≤˚C< 105	105 ≤˚C< 150	150 ≤˚C< 190	190 ≤˚C< 240	240 ≤˚C< 300	˚C≥ 300

The values executed on the CRM map were imposed to obtain a response with a broader range of performance. For the ARM map, this was done in order to achieve a more aggressive response in the valve opening speed.

4.2. **Reference experimental case**

The results obtained in Test A will serve as a reference for comparing with Tests B and C using two approaches for the Response Map of the valve automation control. Figure 4.4 depicts all temperature values.

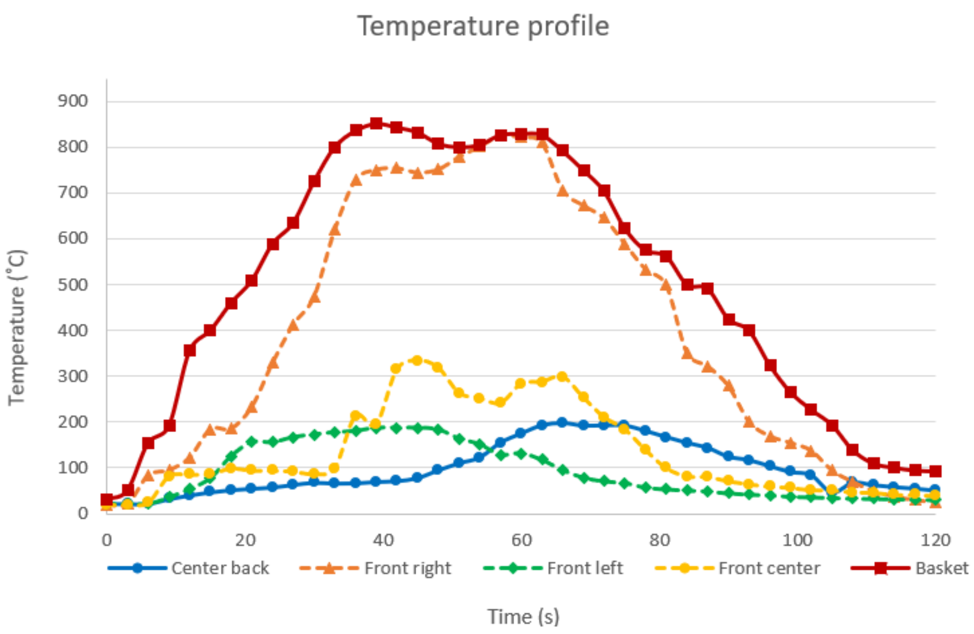


Figure 4.4 – Fabric temperature profile in the first test

The first observation is the precise relation between the evolution of the flame temperature provided by the thermocouple in the basket and the front right temperature of the barrier. The result is evident in the image retrieved from the IR camera (see Figure 4.5 at t= 66s), where it is clear how the flame's deviation to the right side leads to a significant increase in the barrier's temperature. The IR camera helps to uncover that the reason behind this trend was the intentional misalignment between the fuel basket and the barrier. So that, the flux meter – placed on the right side of the barrier, their results turned out to be inconclusive and, therefore, not used – could be under similar heat exposure conditions as the barrier.

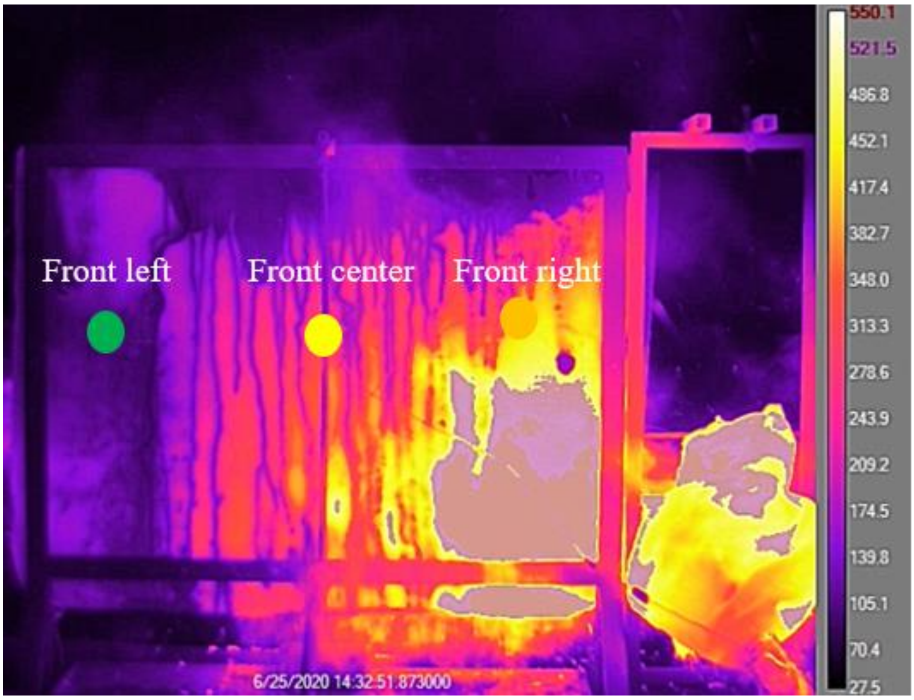


Figure 4.5 – Fabric temperature distribution at t=66s during the first experiment

In fact, as a result of this asymmetric interaction between the fire and the barrier's area, with a higher incidence of the flame on the right side, led to the permanent damage of the aluminum foil as shown in Figure 4.6.

Figure 4.6 – The fire barrier fabric after the first experiment

It is noteworthy how the barrier's temperature values at the center back invert with the front side, decreasing its value, while in contrast, the backside temperature is higher then the front, even if gradually reduces because the flame ceased its activity. From the IR image shown in Figure 4.7 at t = 75s, this temperature decrease is likely due to the water rivulets reaching the neighboring zones of the thermocouple placed in the front center.

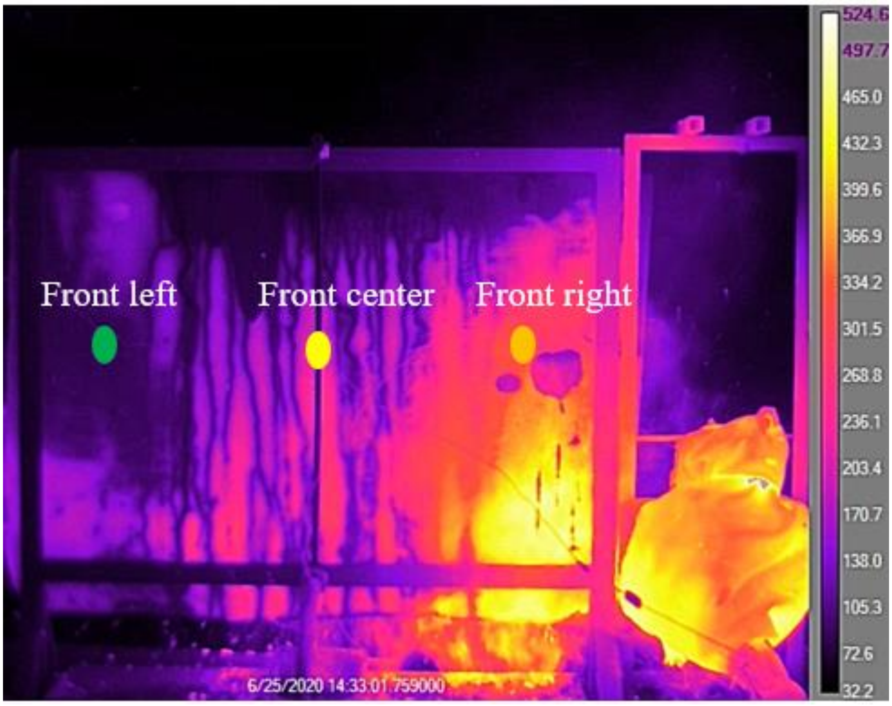


Figure 4.7 – Fabric temperature distribution at t=75s during the first experiment

The results above report the thermal behavior of the barrier for the valve fully open. The following section discusses the application of automated control, considering a conservative (CRM) and aggressive (ARM) response map of the water flow rate using the maximum temperature value on the front as feedback information.

4.3. **Experiments with automated control of water flow**

The experiments with the automated control of the water flow assume the barrier's front is the side directly interacting with the firefront. Therefore, only the information of the front thermocouples is considered. The feedback information provided to control the opening and closing of the valve corresponds to the highest temperature measured between the three instrumented thermocouples. The reasoning is the unstable position of the flame front, as observed in the reference case. Once the program processes the high-temperature value, it decides the next instruction.

In the second experiment (Test B), the valve operates in a conservative mode, and the actuation criteria summarized in Table 4.2. Figure 4.8 contains the temperature values and the valve actuation profile.

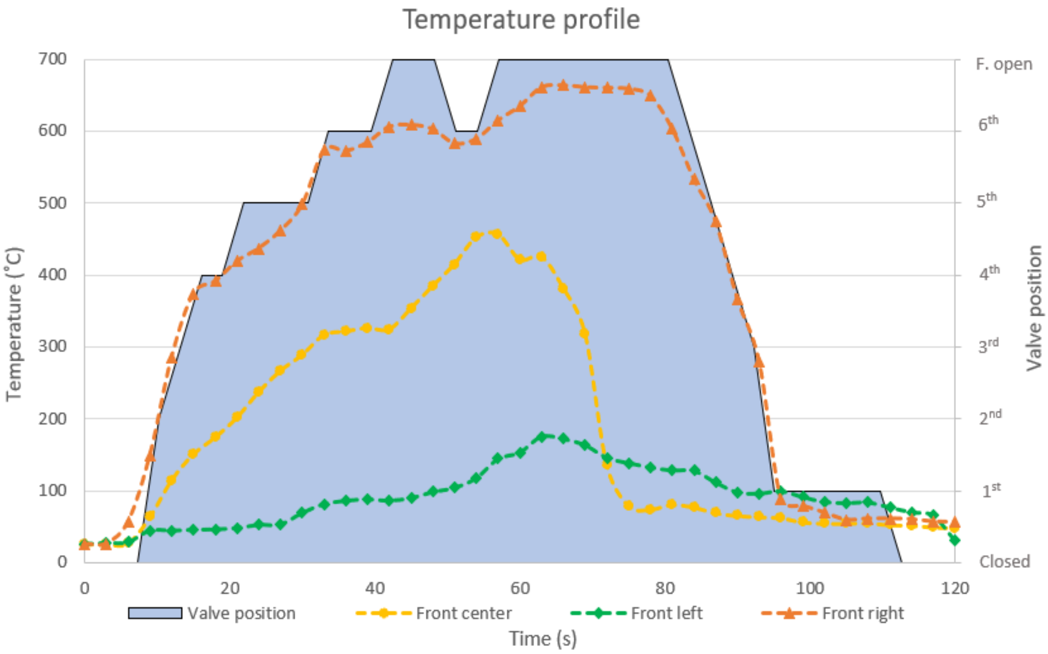


Figure 4.8 – Fabric temperature profile in the second test

Similarly to the reference case, the temperature values on the front right side of the fabric denote a higher interaction with the flame. Moreover, the IR image depicted in Figure 4.9 at t = 66s shows a higher water deposition focused on the front center, coherent with the sudden decrease of temperature in this region, as detected by the front center thermocouple.

Figure 4.9 – Fabric temperature distribution at t=66s during the second experiment

In terms of damage to the fabric due to the incidence of fire by specific zones, test B does not show the destruction of the aluminum foil. However, it is still inconclusive whether this outcome represents an improvement (see Figure 4.10).

Figure 4.10 – The fire barrier fabric after the second experiment

The third test is considered a more aggressive approach, especially in the temperature intervals responsible for triggering each position of the valve. Figure 4.11 shows the results for the temperature values and corresponding valve position. Clearly, the lower range of temperature values leads to the faster opening of the valve, resulting in more extended actuation periods when compared to Test B. Also, the higher temperature at the front center points to a different interaction pattern between the barrier and the flame.

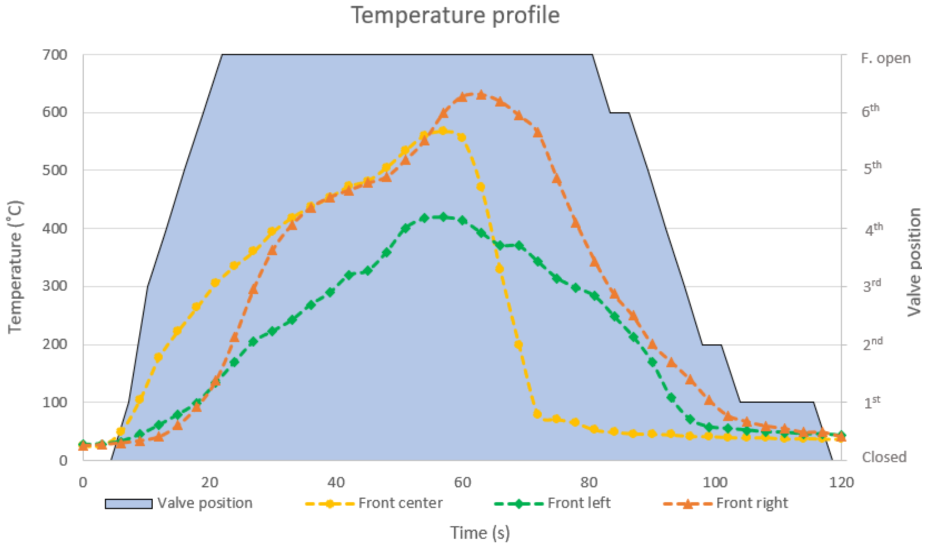


Figure 4.11 – Fabric temperature profile in the third test

This result might be due to the centralization between the fuel basket and the barrier in this test – the flux meter was removed. However, it is noteworthy that the local temperature values might not reproduce the actual interaction between the fabric and the flame, as shown in the IR image of Figure 4.12, especially in the lower part of the fabric that has no sensors.

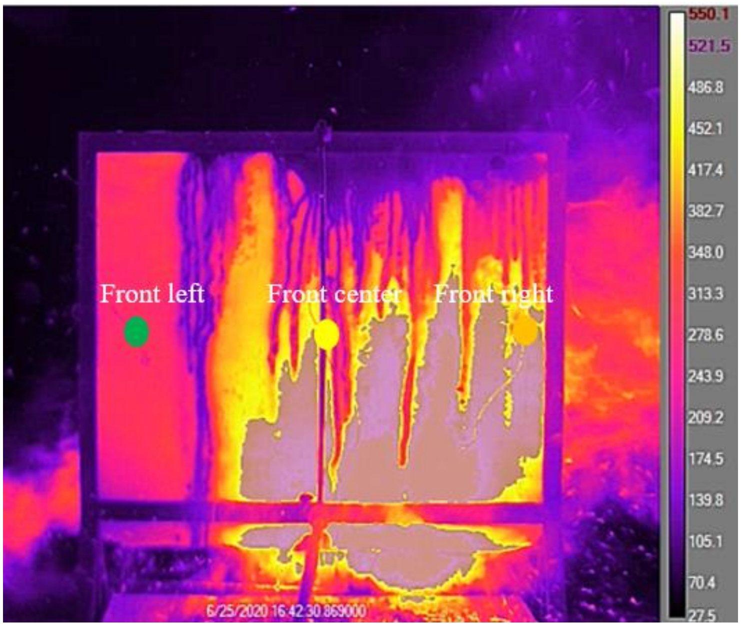


Figure 4.12 – Fabric temperature distribution at t=66s during the third experiment

Accordingly, the fabric damage shown in Figure 4.13 is higher compared to the results obtained in tests A and B. This result was not an expected outcome since more water was injected than in Test B (1.3L against 1L). The results evidence the importance of covering the fabric with more temperature points, especially in the bottom part, to increase the feedback information required to activate the water control system in time to protect the fabric.

Figure 4.13 –The fire barrier fabric after the first experiment

Table 4.3 summarizes the results of the amount of water collected in the three tests performed. Also, this table displays the maximum temperature recorded and the maximum average temperature in each test.

Table 4.3 – Complementary data from the experiments

Test	Max. Temp. (˚C)	Average Max. Temp. (˚C)	Amount of water collected (L)
A	825,3	352,6	2,0
B	663,5	420,4	1,0
C	631,3	532,5	1,3

Test two was the best-optimized in terms of water wasting (only 1L was lost). These values were in line with initial expectations, with the first experiment being the worse (2L lost) followed by the third (1.3L).

The following graphic, Figure 4.14, illustrates, for the three experiments, the relationship between the average temperature on the fabric surface and the valve position.

The average values were obtained from the four thermocouples arranged on the fabric (three in the case of the second and third tests).

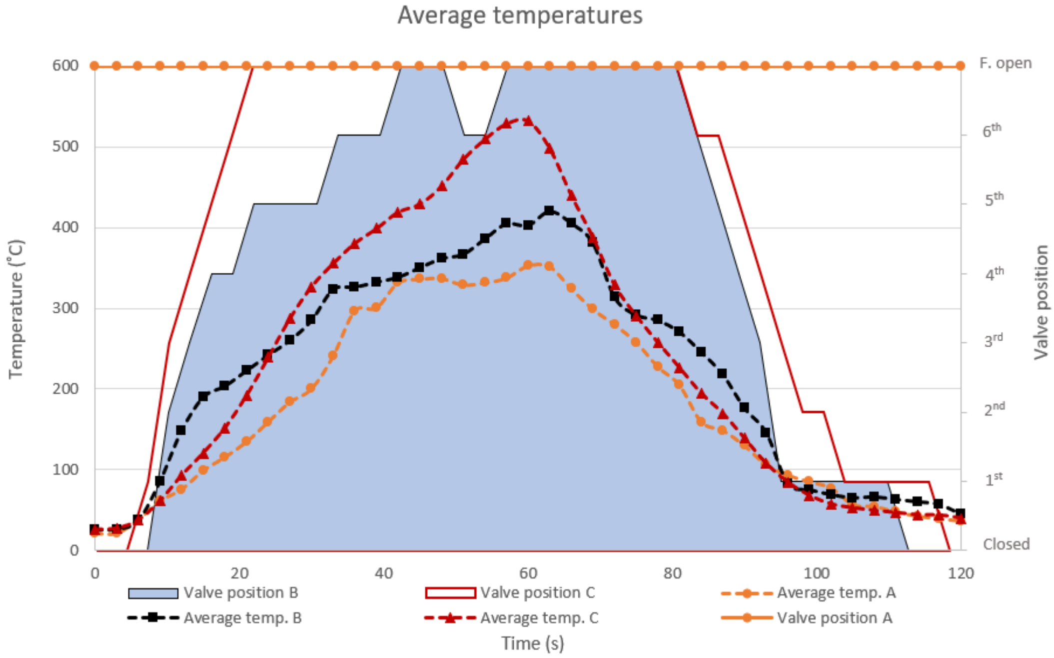


Figure 4.14 – Average temperature and valve position for each test

According to Figure 4.14, and despite the maximum registered temperature in all tests was in Test A, the average temperature in the fabric was the lowest. This result is in line with the expectations since the fabric was injected with a higher amount of water. Also, according to the expectations, the average fabric temperature in Test B was higher when compared to A. Consequently, one may conclude that a higher average temperature does not necessarily mean more damage to the fabric. In test C, this graphic depicts a period (between t=27s and t=72s) in which test C possesses a higher average. This result can be justified by the equitable temperature distribution by the fabric, increasing the value of the global temperature average.

Lastly, Figure 4.15 depicts the relationship between the maximum temperature difference among two consecutive points, divided by the time interval and the position of the valve. This graph aims to analyze valve response and evaluate the possibility of including valve control according to this parameter in the future.

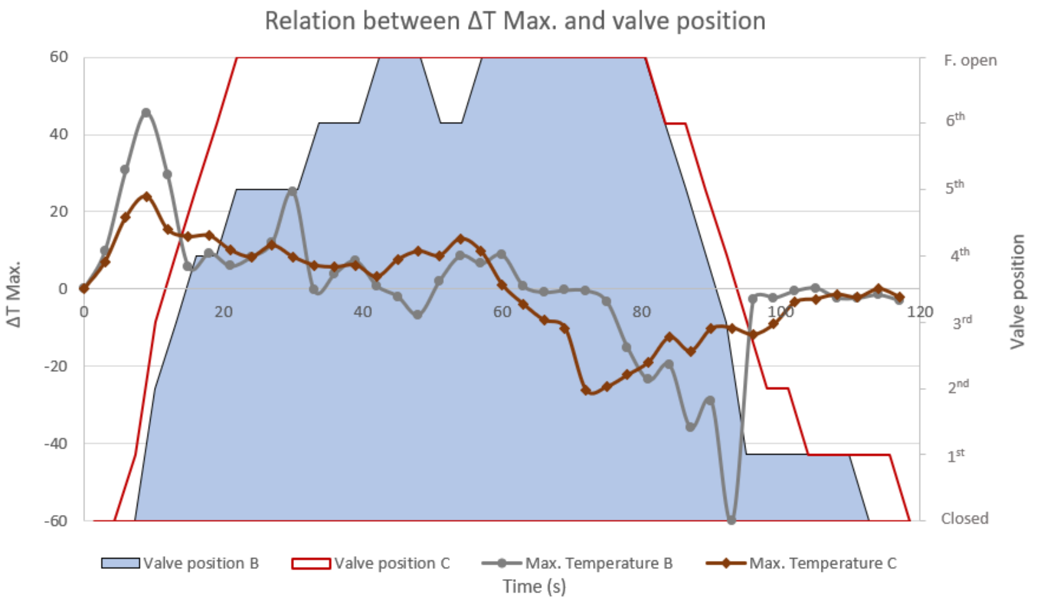


Figure 4.15 – Maximum temperature with valve position

The reason for the type of valve control (depending on temperature limits) was due to the similarity of operation with the mathematical model. For this reason, we opted for this control over others, such as valve control according to temperature derivative or definition of a target temperature and valve control according to the difference between the measured temperature and the target value – closed-loop control.

Although, in Figure 4.15, a brief analysis is made concerning the maximum temperature difference. Regarding these results, we verify that test C has a more stable behavior that might be justified by the lower variability of valve positions and the longer action time. On the other hand, test B presents a random behavior, possibly due to the variability of the valve position. Besides, among t = 84s and t = 96s, there is an abrupt decrease that can be explained by the existent delay between the instruction to the valve and the moment when this instruction is verified. These distinct behaviors indicate the necessity to include a more robust control for the valve. Adding the control option depending on this parameter seems a worthy complement.

To finalize, the difference between the moment when the valve received instruction, and the moment it was verified in the flat spray was about 7s. This difference is significant and could affect the results. Accordingly, it is recommended to use hoses with shorter length – we used hoses with 25m – to reduce this time.

4.4. Case study application

Considering the following case study, shown in Figure 4.16: a property, with an installed fire barrier on a 200m perimeter around it, and an approaching firefront, that may hit the property perimeter on any side and ultimately surround it. Let us also consider that the fire residence time in the vicinity of the property is 15 minutes.

Figure 4.16 – Example of a case study

Given that, the fabric needs 5130L (1.71 l/min per meter of barrier). According to the results of these 5130L, 2500L would be wasted if there was a system with no valve. With the valve incorporated, water losses would drop to about 1625L or 1250L. These values mean savings between 35 to 50%.

In the case of a massive fire spread, it is not as unlikely as this, the occurrence of power cuts, also affecting the water supply network. In such cases, the population often uses small domestic tanks and, with the help of generators and pumps, tries to humidify the surrounding area. If they have this solution, they can make more efficient combat, increasing the time action, and sustaining the spread of fire. This last parameter can mean the difference between the salvation or total loss of the assets

5. CONCLUSIONS

5.1. Achievements

The main goals set for this work, to implement and test the hardware and software for automatic water flow control for the fabric cooling system, as a function of the temperature on its surface, were fully achieved.

Despite the few experiments made, it was demonstrated the feasibility of the proposed solution in maintaining the fabric integrity while managing water usage – water savings between 35 to 50%. However, one can observe that fire is a very volatile phenomenon. Consequently, further experiments to corroborate and support the results and conclusions obtained in this work are recommended.

Even though the experimental results were satisfactory, further improvements on the control parameters can be made to improve the system response time. By using the real-time temperature value solely as a trigger, we are merely reacting to what is happening. Given the fact that there is also a delay in the system response due to the length of the hydraulic installation, plus the typical high-temperature rise rates of the barrier when exposed to the fire, a more robust triggering and control model should be employed. For this purpose, the real-time temperature value and its variation rate (first derivative) should be taken into account, allowing to predict the effect of the fire rather than reacting to it.

This system reveals some limitations, mainly in the temperature measuring process. The fact that four thermocouples have been used limits the portability of the system (since they need to be large enough to maintain a safe distance between the barrier and the system) and have proved to be few to obtain a credible sample of the temperature in the fabric. It is therefore suggested to use wireless temperature sensors in sufficient number to cover a larger mesh in the fabric.

Furthermore, one noticeable effect in the IR camera recordings captured during the experiments was the non-uniform water distribution in the fabric surface. An efficient method to uniformize the water deposition on the fabric should be formulated.

5.2. Future work

Several limitations were identified in the current system, and some improvement suggestions are presented in this subchapter, for future work. The suggestions are as follows:

1. Creation of a system that permits the recirculation of the wasted water in the cooling of the fabric.
2. Study on which the best retarding agent for fire suppression and the most effective way to depose it.
3. Complement this system with equipment that allows the installation to work in the event of power grid failure.
4. Modify thermocouples for intelligent wireless sensors with the ability to monitor additional parameters (besides temperature, humidity, wind speed, or presence of smoke).
5. Creation of a system capable of transmitting signals via Wi-Fi® or Bluetooth®, and complement it with the construction of a smartphone app.
6. Implementation of the heat transfer model and also make the automatic control of the pump.
7. Permit other ways for valve control: in addition to controlling based on the heat transfer model, execute a control depending on temperature derivative.

BIBLIOGRAPHY

Albino, A. (2019). *Thermal modelling and development of an active water-cooled fireproof barrier Modelação térmica e desenvolvimento de uma barreira antifogo ativa e refrigerada a água*. Universidade de Coimbra.

Batista, R. M. (2018). *Mechanisms for Active Protection of People and Infrastructures against Forest Fires*. Universidade de Coimbra.

Costa, M. C. (2018). *Structural design and analysis of an active forest fire propagation barrier*. Universidade de Coimbra.

Dahl, K., Licker, R., Abatzoglou, J. T., & Declet-Barreto, J. (2019). Increased frequency of and population exposure to extreme heat index days in the United States during the 21st century. *Environmental Research Communications*, *1*(7), 075002. https://doi.org/10.1088/2515-7620/ab27cf

Fuller, T. and Nicas, J. (2018). *Wildfire Becomes Deadliest in California History - The New York Times*. https://www.nytimes.com/2018/11/12/us/california-fires-camp-fire.html

Konstantinidis, A. (2019). *O antes e o depois do incêndio mais mortal da Grécia | Fotografia | PÚBLICO*. https://www.publico.pt/2019/07/23/p3/noticia/o-antes-e-o-depois-do-incendio-mais-mortal-da-grecia-1880935

Lawrie, E. (2020). *Australia fires: A visual guide to the bushfire crisis - BBC News*. https://www.bbc.com/news/world-australia-50951043

Lopes, F., Alves, J. and Silva, C. (2020). *Uma cicatriz a céu aberto: como os incêndios devastaram a Austrália | Alterações climáticas | PÚBLICO*. https://www.publico.pt/2020/01/21/infografia/cicatriz-ceu-aberto-incendios-devastaram-australia-408

Meredith, K., De Vries, J., Wang, Y., & Xin, Y. (2013). A comprehensive model for simulating the interaction of water with solid surfaces in fire suppression environments. *Proceedings of the Combustion Institute*, *34*(2), 2719–2726. https://doi.org/10.1016/j.proci.2012.06.094

Morton, A. (2019). *Yes, Australia has always had bushfires: but 2019 is like nothing we've seen before | Australia news | The Guardian*. https://www.theguardian.com/australia-news/2019/dec/25/factcheck-why-australias-monster-2019-bushfires-are-unprecedented

Pires, J. N. (2019). *Automação e Controlo Industrial - Indústria 4.0*. Lidel.

Readfearn, G. (2020). *Living in the climate emergency: Australia's new fire zone | Australia news | The Guardian*. https://www.theguardian.com/environment/ng-interactive/2020/feb/12/living-in-the-climate-emergency-australias-new-fire-zone

Smith, M., Hamm, T., Zajac, M., Chamreun, J. and Bollman, J. (2017). *Automated Wildfire Suppression System*. SMITH MICHAEL FRED. https://www.lens.org/images/patent/US/20170157441/A1/US_2017_0157441_A1.pdf

Takahashi, F. (2019). Whole-House Fire Blanket Protection From Wildland-Urban Interface Fires. *Frontiers in Mechanical Engineering*, *5*(October). https://doi.org/10.3389/fmech.2019.00060

Viegas, C., Batista, R., Albino, A., Coelho, M., Andrade, J., Alves, D. and Viegas, D. X. (2019). *ACTIVE BARRIER COMBINING FIRE-RESISTANT FIBERGLASS FABRIC AND WATER SPRINKLER SYSTEM FOR PROTECTION AGAINST FOREST FIRES*.

Viegas, C., Batista, R., Albino, A., Coelho, M., Andrade, J., Alves, D., & Viegas, D. X. (2020). Active Barrier Combining Fire-Resistant Fiberglass Fabric and Water Sprinkler System for Protection Against Forest Fires. *Fire Technology*. https://doi.org/10.1007/s10694-020-00991-1

Viegas, X., Almeida, M. F., Ribeiro, M., Almeida, M. A., Raposo, J., Viegas, M. T., Oliveira, R., Alves, D., Pinto, C., Rodrigues, A., Ribeiro, C., Lopes, S., Jorge, H., & Viegas, C. X. (2019). *Analysis of the Forest Fires Ocurred in October 15*. 260. https://www.portugal.gov.pt/download-ficheiros/ficheiro.aspx?v=c2da3d7e-dcdb-41cb-b6ae-f72123a1c47d

Weber, E. J. (2015). *Wildfire Suppression System*. WEBER JUNATA E OP - US 201314076620 A 20131111. https://www.lens.org/images/patent/US/20150129245/A1/US_2015_0129245_A1.pdf

Yount, C. J. (2012). *Remote Automated Wildfire Protection System*. YOUNT JOHN CHARLES OP - US 98106710 A 20101229. https://www.lens.org/images/patent/US/20120168185/A1/US_2012_0168185_A1.pdf

ANNEX A

As mentioned in the main text, this section will be used to provide detailed data concerning the equipment.

1. Hauhinco 2/2-way proportional seat valve:

1/1 | Hauhinco-2_2-way_proportional seat valve_DN10-PN320_PN500 | water hydraulics | 01/2016

Hauhinco

2/2-way proportional seat valve
Directly actuated

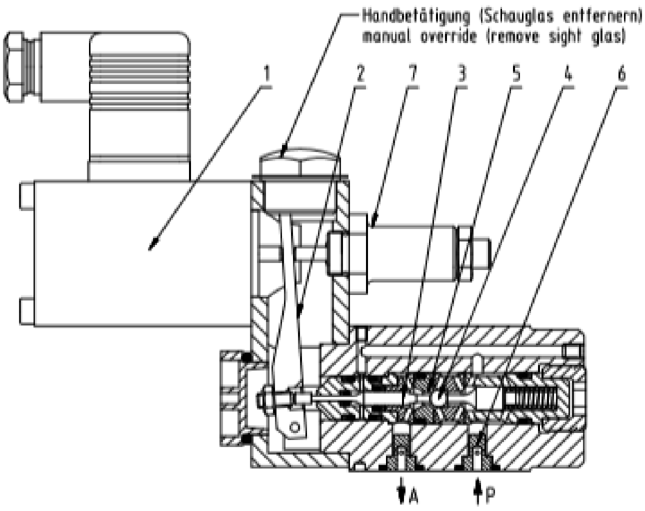
DN10
PN320, PN500

Features:
- Directly actuated directional seat valve for liquid media
- Valve actuation by means of proportional solenoid
- The proportional solenoid is activated via the Hauhinco automatic control unit "Controller RE4"
- The valve seat provides a leak-free seal
- All parts are made of corrosion-resistant materials, and they are easily replaceable
- Valve fastening structural plate form with a Hauhinco Connection diagram

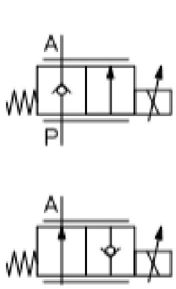
Function 2/2-way proportional seat valve:
The generated magnetic force (1) acts via the lever (2), the tappet (3) onto the ball (4) and presses it out of the valve seat (5). This is used to connect lines P and A. The covers (6) support the flange seals from the inside and limits the maximum admissible volume flow. A defined actuating force can be preset (factory setting) by means of the counterspring. The proportional solenoid is electrically activated by regulated current generated as the manipulated variable by the controlled electronic control unit, the Controller RE4. The controlled valve current specifies the degree of opening of the valve, thus, a hydraulic volume flow or pressure control can be performed.

Depending on the arrangement of the valve seat (5) and ball (4) the valve will have the basic position normally closed (NC) or normally open (NO).

Example 2/2-way valve:



Symbol



2/2 | Hauhinco-2_2-way_proportional seat valve_DN10-PN320_PN500 | water hydraulics | 01/2016

Hauhinco

2/2-way proportional seat valve DN10 | PN320, PN500

Technical data

measured with HFA medium 97/3%, at 20°C

general

Weight	see Order information
Installation position	any
Ambient temperature	-10 to 50°C (hydraulic fluids, heed standard requirements)
Material	
- Valve parts	Stainless steel, bronze except electromagnet
- Seals	NBR, PTFE

hydraulic

max. operating pressure of connector **P**	320 bar, 500 bar see Valve data
max. operating pressure of connector **A**	320 bar, 500 bar see Valve data
max. operating pressure of connector **T**	50 bar
max./min. control pressure of connector **Z**	see Order information
max. volume flow **P →A**	40 l/min
specified direction of flow	P→A
Pressure fluid	**Water, HFA**
- Medium - Temperature range	5 to 50°C
- Medium - Quality	see Hauhinco requirements on water and HFA media
- Cleanliness class, filter fineness	Class -/18/15, filter fineness 25μm
- Viscosity	0.6 to 100 mm²/s
Pressure fluid	**Mineral oil HLP**
- Medium - Temperature range	-10 to 50°C
- Medium - Quality	acc. to DIN 51524
- Cleanliness class, filter fineness	Class -/18/15, filter fineness 25μm
- Viscosity	0.6 to 100 mm²/s

Use of other pressure fluids on request.

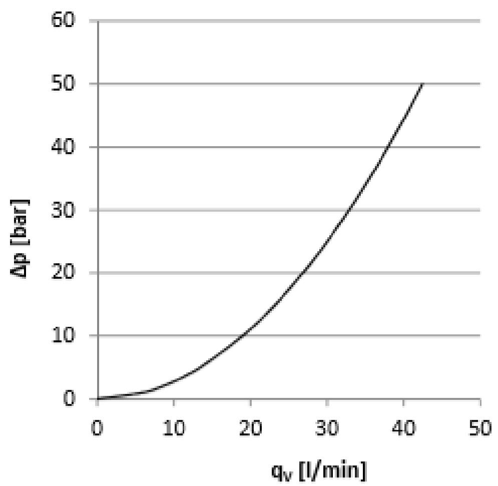
The covers (6) are designed with a viscosity of approx. 1.0 mm²/s; if a medium with a substantially different viscosity is used, the covers must be selected such that the maximum admissible volume flow is not exceeded.

electric

Voltage	24 VDC
Power consumption	21 W
Operating time	100% OT
Degree of protection acc. to EN60529	IP65

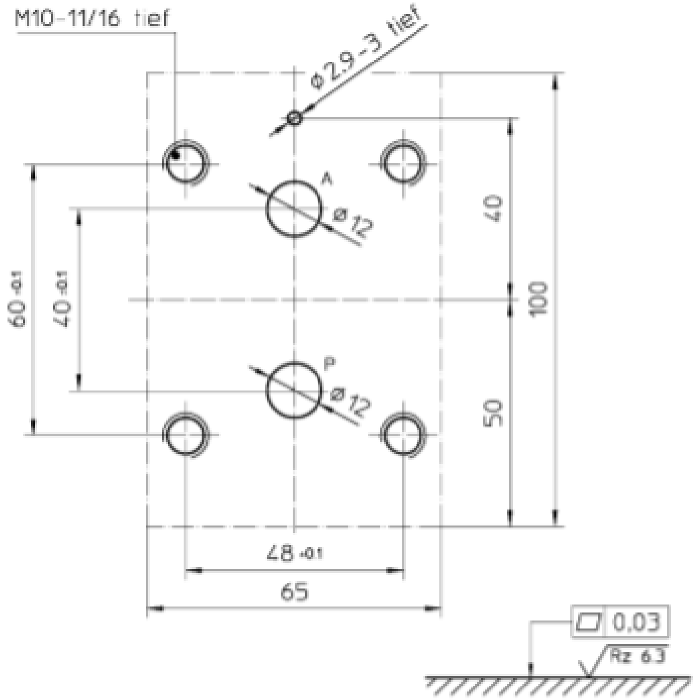
2/2-way proportional seat valve DN10 | PN320, PN500

$\Delta p - q_V$ characteristic curve



Connection diagram 2/2-way DN10

Drawing number 6559808



2. RE4 controller:

Characteristics Hauhinco

2. Characteristics

This electronic module has been developed for controlling closed loop controlled applications in hydraulic systems. This can be an axis position control as well as a pressure control. An integrated power stage and high dynamic control loops (1 ms for pressure control and 0.125 ms for the current loop control) offer a simple and powerful solution with optimal adaption to the relating valve.
Demand (W) and actual signal (X) are read in via scalable analogue signals (range 0… 10 V or 4… 20 mA). The demand value can also preset by eight programmed values and called via digital inputs. Ramp function and PID controller are usable universally. The output current is closed loop controlled and therefore independent from the power supply and the solenoid resistance. The power stage is monitored for cable breakdown, is short circuit proof and disables in case of an error. The advantage of the integrated power stage is founded in the integrated control behaviour without additional dead times. This allows higher dynamics and higher stability respectively.
Parameterisable are the controller parameters OFFSET, P, I, D, T1 in addition I_ACT, I_DEACT and I_LIM for the integrator controls. Furthermore implemented additional PT1 controller and feed forward value as well as additional integrator controls for activating, freezing and deactivating and the valve adaption with MIN, MAX, TRIGGER, DITHER, PWM and CURRENT.

Control functionality is selectable between two ball seat valves with two solenoids (two separate valves) and spool valve with one solenoid. For controlling several valves in parallel a master/slave mode is available (via independent in- and outputs). In slave mode the controller is not active and the control signal from the master device is given to the power stage via the MIN/MAX function.
The handling is simple and problem orientated, therefore a short training period is guaranteed.

Typical applications: Position and pressure controlling with Hauhinco control valves.

2.1 Features

- Compact housing
- Digital reproducable settings
- Free scalable analogue inputs for demand and actual value
- Alternative digital selectable and parameterisable demand values
- Universal PID controller
- Usable with control valves with one or two solenoids
- Free settings of Ramps, MIN and MAX, Dither (frequency and amplitude) and PWM frequency
- Automatic scaling of sensor and deadband compensation
- Function generator to monitoring control quality
- Master/Slave mode for parallel pilot valves
- Output current parameterisable up to 2.6.A
- Simple and application orientated parameter settings
- Fault diagnosis and extended function checking
- Adjustment via USB connection with own start-up software

3.2 Typical system structure

This minimal system consists of the following components

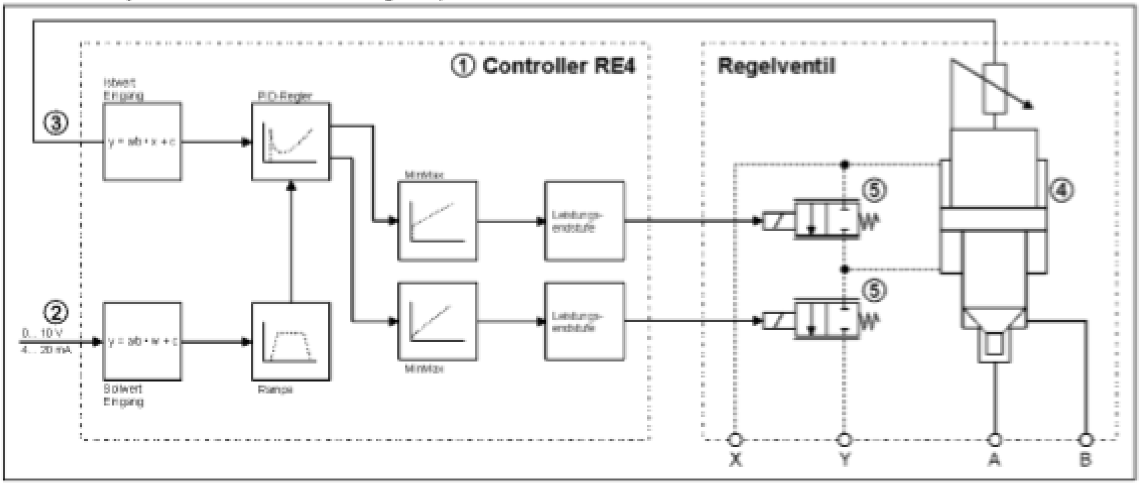


Fig. 2 Minimal system

1	Controller RE4	4	Main valve
2	Demand signal (e.g. interface to PLC)	5	Pilot valve
3	Actual signal (e.g. position indicator)		

3.3 Method of operation

This module is controlled by an analogue demand signal. An ENABLE signal (typical 24 V) activates the control function. When error free working it is reported via a READY output.
In case of a detected error the power stage gets deactivated and the error is reported by deactivating the READY output and a flashing READY LED.
The output current is closed loop controlled whereby a high accuracy and a good dynamic will be obtained.
With a modified/optimized PID controller the control behaviour can be adapted to the relative requirements. The control structure can be chosen between two ball seat valves with two solenoids and spool valve with one solenoid.
(→ Fig. 3 Control structure - Ball seat valve / Spool valve).
In special SLAVE mode the amplifier gets the control signal from the master module. The PID controller is not active, the control signal is given to the power stage via the valve adaption.

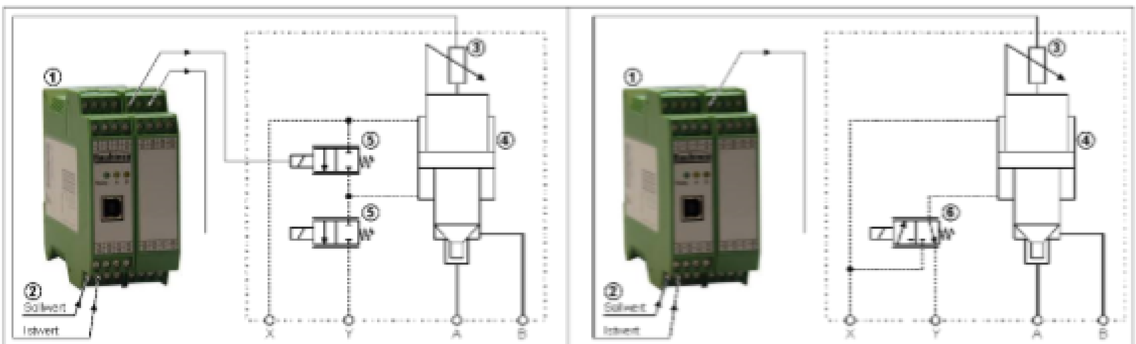


Fig. 3 Control structure - Ball seat valve / Spool valve

1	Controller RE4	3	Position indicator	5	Ball seat valve
2	Demand signal (e.g. interface to PLC)	4	Main valve	6	Spool valve

4. Technical description

4.1 Input and output signals

Connection	Supply
PIN 3 PIN 22	Power supply (see technical data) Power supply power stage
PIN 4 PIN 24	0 V (GND) connection supply 0 V (GND) connection power stage
Connection	**Analogue signals**
PIN 10 / 9	Slave demand value (W), range 0… 10 V (OP_MODE = SL) - **connect PIN 9 to GND**
PIN 13	Position demand value (W), range 0… 10 V or 4… 20 mA, scalable
PIN 14	Analogue position actual value (X), range 0… 10 V or 4… 20 mA, scalable
PIN 11	0 V (GND) connection for input signals
PIN 12	0 V (GND) connection for output signals
PIN 15	Control signal, output to slave valve, range 0… 5… 10 V (OP_MODE = MA)
PIN 16	Monitor output, source value selectable with SIGNAL:M, range 0… 10 V
Connection	**Solenoid outputs**
PIN 17 / 19	PWM output solenoid A
PIN 18 / 20	PWM output solenoid B
Connection	**Digital inputs and outputs**
PIN 8	**Enable input:** General enabling of the application. Erases error messages, activates the controller, the power stage and the READY signal.
PIN 5 / 6 / 7	**Digital inputs for recalling preset demand values:** PIN 5: S1 / PIN 6: S2 / PIN 7: S4 The parameterised demand values can be selected by binary coded combining of these inputs (INPUTW = DIGI).
PIN 1	**READY output:** **ON:** The module is enabled; there are no discernible errors. **OFF:** Enable or supply is missing or an error has been detected.
PIN 2	**STATUS output:** Display of the control deviation. Depending on CDWIN and CDED parameter the status output gets deactivated if the control deviation gets bigger than the set window. **ON:** The control deviation is lower than the set range. **OFF:** The control deviation is higher than the set range.

Tab. 2 Input and output signals

Hauhinco Technical description

4.6 Technical data

Supply voltage (U_b) Power requirement max. External protection	[VDC] [W] [A]	12… 30 (incl. ripple) 60 (depending on solenoid) 3 medium time lag
Digital inputs Input resistance	[V] [V] [kOhm]	OFF : < 2 ON : > 10 25
Digital outputs Maximum output current	[V] [V] [mA]	OFF: < 2 ON: max. U_b 50
Analogue inputs Signal resolution	[V] [mA] [%]	0… 10; min. 25 kOhm 4… 20; 240 Ohm 0,003 incl. oversampling
Analogue outputs Voltage Signal resolution	 [V] [mA] [%]	2 0… 10 10 (max. load) 0,006
PWM outputs Nominal current Frequency	 [mA] [Hz]	2 500… 2600; broken wire monitored and short circuit proof 61… 2604; adjustable in steps
Sample time PID controller Current controller	 [ms] [ms]	 1 0,125
Serial interface		USB in RS 232C Emulation (9600… 57600 Baud, 1 Stop bit, no parity, Echo Mode)
Housing		Snap-on module to EN 50022 PA 6.6 polyamide Flammability class V0 (UL94)
Protection class Temperature range Storage temperature Humidity	 [°C] [°C] [%]	IP20 -20… 60 -20… 70 < 95 (non-condensing)
Weight	[kg]	0,250
Connections		USB Type B 8 x 4 pole terminal blocks PE: via the DIN mounting rail
EMC		EN 61000-6-2: 8/2005 EN 61000-6-4: 6/2007 + A1:2011

Tab. 4 Technical data

3. Arduino® Uno board REV 3:

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Power

The Arduino Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- Vin. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- 5V.This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- GND. Ground pins.
- IOREF. This pin on the Arduino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

Input and Output

See the mapping between Arduino pins and ATmega328P ports. The mapping for the Atmega8, 168, and 328 is identical.

PIN MAPPING
ATmega328P

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(),digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

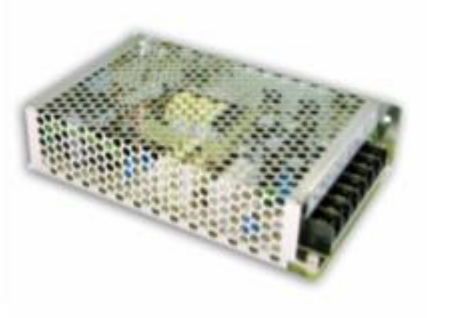
- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. There are a couple of other pins on the board:

- AREF. Reference voltage for the analog inputs. Used with analogReference().
- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

4. Power supply

MEAN WELL 100W Single Output Switching Power Supply **RS-100 series**



■ Features :

- Universal AC input / Full range
- Protections: Short circuit / Overload / Over voltage
- Cooling by free air convection
- LED indicator for power on
- 100% full load burn-in test
- All using 105℃ long life electrolytic capacitors
- Withstand 300VAC surge input for 5 second
- High operating temperature up to 70℃
- Withstand 5G vibration test
- High efficiency, long life and high reliability
- 3 years warranty

SPECIFICATION

MODEL		RS-100-3.3	RS-100-5	RS-100-12	RS-100-15	RS-100-24	RS-100-48
OUTPUT	DC VOLTAGE	3.3V	5V	12V	15V	24V	48V
	RATED CURRENT	20A	16A	8.5A	7A	4.5A	2.3A
	CURRENT RANGE	0 ~ 20A	0 ~ 16A	0 ~ 8.5A	0 ~ 7A	0 ~ 4.5A	0 ~ 2.3A
	RATED POWER	66W	80W	102W	105W	108W	110.4W
	RIPPLE & NOISE (max.) Note.2	80mVp-p	80mVp-p	120mVp-p	120mVp-p	120mVp-p	200mVp-p
	VOLTAGE ADJ. RANGE	3.2V ~ 3.5V	4.75 ~ 5.5V	11.4 ~ 13.2V	14.25 ~ 16.5V	22.8 ~ 26.4V	45.6 ~ 52.8V
	VOLTAGE TOLERANCE Note.3	±3.0%	±2.0%	±1.0%	±1.0%	±1.0%	±1.0%
	LINE REGULATION Note.4	±0.5%	±0.5%	±0.5%	±0.5%	±0.5%	±0.5%
	LOAD REGULATION Note.5	±2.0%	±1.0%	±0.5%	±0.5%	±0.5%	±0.5%
	SETUP, RISE TIME	500ms, 20ms/230VAC 1200ms, 30ms/115VAC at full load					
	HOLD UP TIME (Typ.)	95ms/230VAC 17ms/115VAC at full load					
INPUT	VOLTAGE RANGE	88 ~ 264VAC 125 ~ 373VDC (Withstand 300VAC surge for 5sec. Without damage)					
	FREQUENCY RANGE	47 ~ 63Hz					
	EFFICIENCY (Typ.)	74%	77%	81%	82%	84%	84%
	AC CURRENT (Typ.)	2.5A/115VAC 1.5A/230VAC					
	INRUSH CURRENT (Typ.)	COLD START 40A/230VAC					
	LEAKAGE CURRENT	<2mA / 240VAC					
PROTECTION	OVERLOAD	110 ~ 150% rated output power					
		Protection type : Hiccup mode, recovers automatically after fault condition is removed					
	OVER VOLTAGE	3.8 ~ 4.45V	5.75 ~ 6.75V	13.8 ~ 16.2V	17.25 ~ 20.25V	27.6 ~ 32.4V	55.2 ~ 64.8V
		Protection type : Hiccup mode, recovers automatically after fault condition is removed					
ENVIRONMENT	WORKING TEMP.	-25 ~ +70℃ (Refer to "Derating Curve")					
	WORKING HUMIDITY	20 ~ 90% RH non-condensing					
	STORAGE TEMP., HUMIDITY	-40 ~ +85℃, 10 ~ 95% RH					
	TEMP. COEFFICIENT	±0.03%/℃ (0 ~ 50℃)					
	VIBRATION	10 ~ 500Hz, 5G 10min./1cycle, period for 60min. each along X, Y, Z axes					
SAFETY & EMC (Note 6)	SAFETY STANDARDS	UL62368-1, TUV EN62368-1, EAC TP TC 004 approved					
	WITHSTAND VOLTAGE	I/P-O/P:3KVAC I/P-FG:2KVAC O/P-FG:0.5KVAC					
	ISOLATION RESISTANCE	I/P-O/P, I/P-FG, O/P-FG:100M Ohms / 500VDC / 25℃ / 70% RH					
	EMC EMISSION	Compliance to EN55032 (CISPR32) Class B, EN61000-3-2,-3, EAC TP TC 020					
	EMC IMMUNITY	Compliance to EN61000-4-2,3,4,5,6,8,11, EN61000-6-2 (EN50082-2), heavy industry level, criteria A, EAC TP TC 020					
OTHERS	MTBF	260.8Khrs min. MIL-HDBK-217F (25℃)					
	DIMENSION	159*97*38mm (L*W*H)					
	PACKING	0.6Kg; 24pcs/15.4Kg/0.7CUFT					

NOTE

1. All parameters NOT specially mentioned are measured at 230VAC input, rated load and 25℃ of ambient temperature.
2. Ripple & noise are measured at 20MHz of bandwidth by using a 12" twisted pair-wire terminated with a 0.1uf & 47uf parallel capacitor.
3. Tolerance : includes set up tolerance, line regulation and load regulation.
4. Line regulation is measured from low line to high line at rated load.
5. Load regulation is measured from 0% to 100% rated load.
6. The power supply is considered a component which will be installed into a final equipment. The final equipment must be re-confirmed that it still meets EMC directives. For guidance on how to perform these EMC tests, please refer to "EMI testing of component power supplies." (as available on http://www.meanwell.com)
7. Length of set up time is measured at cold first start. Turning ON/OFF the power supply very quickly may lead to increase of the set up time.
8. The ambient temperature derating of 3.5℃/1000m with fanless models and of 5℃/1000m with fan models for operating altitude higher than 2000m(6500ft).

5. Thermocouple amplifiers

19-5793; Rev 2; 2/12

MAX31855

Cold-Junction Compensated Thermocouple-to-Digital Converter

General Description

The MAX31855 performs cold-junction compensation and digitizes the signal from a K-, J-, N-, T-, S-, R-, or E-type thermocouple. The data is output in a signed 14-bit, SPI-compatible, read-only format. This converter resolves temperatures to 0.25°C, allows readings as high as +1800°C and as low as -270°C, and exhibits thermocouple accuracy of ±2°C for temperatures ranging from -200°C to +700°C for K-type thermocouples. For full range accuracies and other thermocouple types, see the *Thermal Characteristics* specifications.

Features

- **Cold-Junction Compensation**
- **14-Bit, 0.25°C Resolution**
- **Versions Available for K-, J-, N-, T-, S-, R-, and E-Type Thermocouples (see Table 1)**
- **Simple SPI-Compatible Interface (Read-Only)**
- **Detects Thermocouple Shorts to GND or VCC**
- **Detects Open Thermocouple**

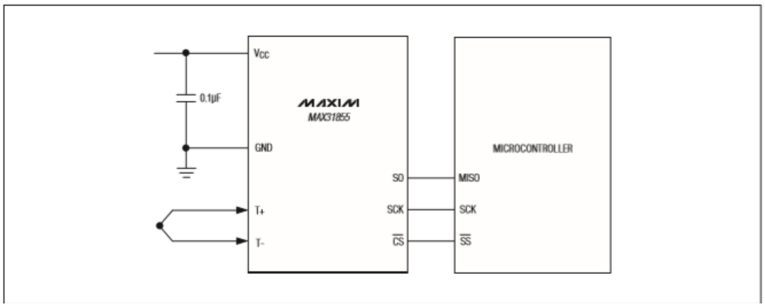
Ordering Information ***appears at end of data sheet.***

Applications

Industrial
Appliances
HVAC
Automotive

For related parts and recommended products to use with this part, refer to: www.maxim-ic.com/MAX31855.related

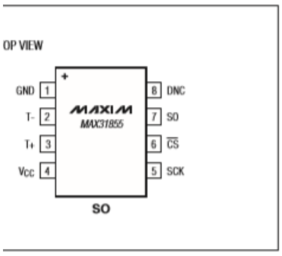
Typical Application Circuit



MAX31855

Cold-Junction Compensated Thermocouple-to-Digital Converter

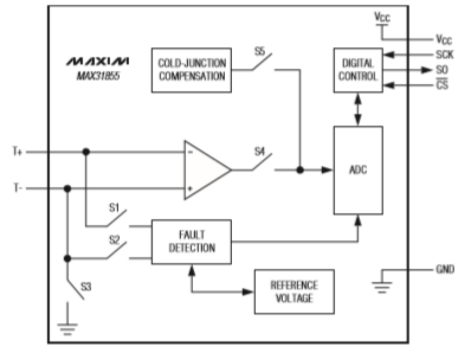
Pin Configuration



Pin Description

PIN	NAME	FUNCTION
1	GND	Ground
2	T-	Thermocouple Input. See Table 1. Do not connect to GND.
3	T+	Thermocouple Input. See Table 1.
4	V_{CC}	Power-Supply Voltage
5	SCK	Serial-Clock Input
6	$\overline{CS}$	Active-Low Chip Select. Set $\overline{CS}$ low to enable the serial interface.
7	SO	Serial-Data Output
8	DNC	Do Not Connect

Block Diagram



MAX31855

Cold-Junction Compensated Thermocouple-to-Digital Converter

Detailed Description

The MAX31855 is a sophisticated thermocouple-to-digital converter with a built-in 14-bit analog-to-digital converter (ADC). The device also contains cold-junction compensation sensing and correction, a digital controller, an SPI-compatible interface, and associated control logic. The device is designed to work in conjunction with an external microcontroller (µC) in thermostatic, process-control, or monitoring applications. The device is available in several versions, each optimized and trimmed for a specific thermocouple type (K, J, N, T, S, R, or E.). The thermocouple type is indicated in the suffix of the part number (e.g., MAX31855K). See the *Ordering Information* table for all options.

Temperature Conversion

The device includes signal-conditioning hardware to convert the thermocouple's signal into a voltage compatible with the input channels of the ADC. The T+ and T- inputs connect to internal circuitry that reduces the introduction of noise errors from the thermocouple wires.

Before converting the thermoelectric voltages into equivalent temperature values, it is necessary to compensate for the difference between the thermocouple cold-junction side (device ambient temperature) and a 0°C virtual reference. For a K-type thermocouple, the voltage changes by about 41µV/°C, which approximates the thermocouple characteristic with the following linear equation:

$$V_{OUT} = (41.276\mu V/°C) \times (T_R - T_{AMB})$$

where V_{OUT} is the thermocouple output voltage (µV), T_R is the temperature of the remote thermocouple junction (°C), and T_{AMB} is the temperature of the device (°C).

Other thermocouple types use a similar straight-line approximation but with different gain terms. Note that the MAX31855 assumes a linear relationship between temperature and voltage. Because all thermocouples exhibit some level of nonlinearity, apply appropriate correction to the device's output data.

Cold-Junction Compensation

The function of the thermocouple is to sense a difference in temperature between two ends of the thermocouple wires. The thermocouple's "hot" junction can be read across the operating temperature range (Table 1). The reference junction, or "cold" end (which should be at

Table 1. Thermocouple Wire Connections and Nominal Sensitivities

TYPE	T- WIRE	T+ WIRE	TEMP RANGE (°C)	SENSITIVITY (µV/°C)	COLD-JUNCTION SENSITIVITY (µV/°C) (0°C TO +70°C)
K	Alumel	Chromel	-270 to +1372	41.276 (0°C to +1000°C)	40.73
J	Constantan	Iron	-210 to +1200	57.953 (0°C to +750°C)	52.136
N	Nisil	Nicrosil	-270 to + 1300	36.256 (0°C to +1000°C)	27.171
S	Platinum	Platinum/Rhodium	+50 to +1768	9.587 (0°C to +1000°C)	6.181
T	Constantan	Copper	-270 to +400	52.18 (0°C to +400°C)	41.56
E	Constantan	Chromel	-270 to +1000	76.373 (0°C to +1000°C)	44.123
R	Platinum	Platinum/Rhodium	-50 to +1768	10.506 (0°C to +1000°C)	6.158

the same temperature as the board on which the device is mounted) can range from -55°C to +125°C. While the temperature at the cold end fluctuates, the device continues to accurately sense the temperature difference at the opposite end.

The device senses and corrects for the changes in the reference junction temperature with cold-junction compensation. It does this by first measuring its internal die temperature, which should be held at the same temperature as the reference junction. It then measures the voltage from the thermocouple's output at the reference junction and converts this to the noncompensated thermocouple temperature value. This value is then added to the device's die temperature to calculate the thermocouple's "hot junction" temperature. Note that the "hot junction" temperature can be lower than the cold junction (or reference junction) temperature.

Optimal performance from the device is achieved when the thermocouple cold junction and the device are at the same temperature. Avoid placing heat-generating devices or components near the MAX31855 because this could produce cold-junction-related errors.

Conversion Functions

During the conversion time, t_{CONV}, three functions are performed: the temperature conversion of the internal cold-junction temperature, the temperature conversion of the external thermocouple, and the detection of thermocouple faults.

When executing the temperature conversion for the internal cold-junction compensation circuit, the connection to signal from the external thermocouple is opened (switch S4) and the connection to the cold-junction compensation circuit is closed (switch S5). The internal T- reference to ground is still maintained (switch S3 is closed) and the connections to the fault-detection circuit are open (switches S1 and S2).

When executing the temperature conversion of the external thermocouple, the connections to the internal fault-detection circuit are opened (switches S1 and S2 in the *Block Diagram*) and the switch connecting the cold-junction compensation circuit is opened (switch S5). The internal ground reference connection (switch S3) and the connection to the ADC (switch S4) are closed. This allows the ADC to process the voltage detected across the T+ and T- terminals.

During fault detection, the connections from the external thermocouple and cold-junction compensation circuit to the ADC are opened (switches S4 and S5). The internal ground reference on T- is also opened (switch S3). The connections to the internal fault-detection circuit are closed (switch S1 and S2). The fault-detection circuit tests for shorted connections to V_{CC} or GND on the T+ and T- inputs, as well as looking for an open thermocouple condition. Bits D0, D1, and D2 of the output data are normally low. Bit D2 goes high to indicate a thermocouple short to V_{CC}, bit D1 goes high to indicate a thermocouple short to GND, and bit D0 goes high to indicate a thermocouple open circuit. If any of these conditions exists, bit D16 of the SO output data, which is normally low, also goes high to indicate that a fault has occurred.

Serial Interface

The *Typical Application Circuit* shows the device interfaced with a microcontroller. In this example, the device processes the reading from the thermocouple and transmits the data through a serial interface. Drive $\overline{CS}$ low and apply a clock signal at SCK to read the results at SO. Conversions are always being performed in the background. The fault and temperature data are only be updated when $\overline{CS}$ is high.

Drive $\overline{CS}$ low to output the first bit on the SO pin. A complete serial-interface read of the cold-junction compensated thermocouple temperature requires 14 clock cycles. Thirty-two clock cycles are required to read both the thermocouple and reference junction temperatures (Table 2 and Table 3.) The first bit, D31, is the thermocouple temperature sign bit, and is presented to the SO pin within t_{DV} of the falling edge of $\overline{CS}$. Bits D[30:18] contain the converted temperature in the order of MSB to LSB, and are presented to the SO pin within t_{D0} of the falling edge of SCK. Bit D16 is normally low and goes high when the thermocouple input is open or shorted to GND or V_{CC}. The reference junction temperature data begins with D15. $\overline{CS}$ can be taken high at any point while clocking out conversion data. If T+ and T- are unconnected, the thermocouple temperature sign bit (D31) is 0, and the remainder of the thermocouple temperature value (D[30:18]) is 1.

Figure 1 and Figure 2 show the serial-interface timing and order. Table 2 and Table 3 show the SO output bit weights and functions.

6. Relay module

SONGLE RELAY

松乐继电器 ® SONGLE RELAY	RELAY ISO9002	SRD

1. MAIN FEATURES

- Switching capacity available by 10A in spite of small size design for highdensity P.C. board mounting technique.
- UL,CUL,TUV recognized.
- Selection of plastic material for high temperature and better chemical solution performance.
- Sealed types available.
- Simple relay magnetic circuit to meet low cost of mass production.

2. APPLICATIONS

- Domestic appliance, office machine, audio, equipment, automobile, etc.
  (Remote control TV receiver, monitor display, audio equipment high rushing current use application.)

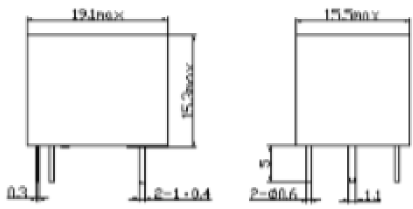
3. ORDERING INFORMATION

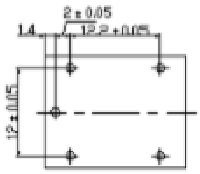
SRD	XX VDC	S	L	C
Model of relay	Nominal coil voltage	Structure	Coil sensitivity	Contact form
SRD	03、05、06、09、12、24、48VDC	S:Sealed type	L:0.36W	A:1 form A
				B:1 form B
		F:Flux free type	D:0.45W	C:1 form C

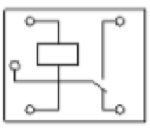
4. RATING

CCC	FILE NUMBER:CH0052885-2000	7A/240VDC
CCC	FILE NUMBER:CH0036746-99	10A/250VDC
UL /CUL	FILE NUMBER: E167996	10A/125VAC 28VDC
TUV	FILE NUMBER: R9933789	10A/240VAC 28VDC

5. DIMENSION(unit:mm) DRILLING(unit:mm) WIRING DIAGRAM







6. COIL DATA CHART (AT20ºC)

Coil Sensitivity	Coil Voltage Code	Nominal Voltage (VDC)	Nominal Current (mA)	Coil Resistance (Ω) ±10%	Power Consumption (W)	Pull-In Voltage (VDC)	Drop-Out Voltage (VDC)	Max-Allowable Voltage (VDC)
SRD (High Sensitivity)	03	03	120	25	abt. 0.36W	75%Max.	10% Min.	120%
	05	05	71.4	70				
	06	06	60	100				
	09	09	40	225				
	12	12	30	400				
	24	24	15	1600				
	48	48	7.5	6400				
SRD (Standard)	03	03	150	20	abt. 0.45W	75% Max.	10% Min.	110%
	05	05	89.3	55				
	06	06	75	80				
	09	09	50	180				
	12	12	37.5	320				
	24	24	18.7	1280				
	48	48	10	4500	abt. 0.51W			

7. CONTACT RATING

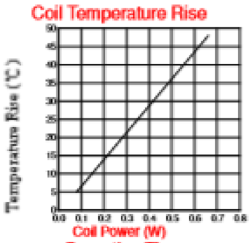
Item \ Type	SRD FORM C	SRD FORM A
Contact Capacity Resistive Load (cosΦ=1)	7A 28VDC 10A 125VAC 7A 240VAC	10A 28VDC 10A 240VAC
Inductive Load (cosΦ=0.4 L/R=7msec)	3A 120VAC 3A 28VDC	5A 120VAC 5A 28VDC
Max. Allowable Voltage	250VAC/110VDC	250VAC/110VDC
Max. Allowable Power Force	800VAC/240W	1200VA/300W
Contact Material	AgCdO	AgCdO

8. PERFORMANCE (at initial value)

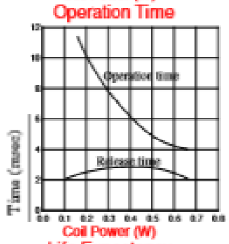
Item \ Type	SRD
Contact Resistance	100mΩ Max.
Operation Time	10msec Max.
Release Time	5msec Max.
Dielectric Strength Between coil & contact Between contacts	 1500VAC 50/60HZ (1 minute) 1000VAC 50/60HZ (1 minute)
Insulation Resistance	100 MΩ Min. (500VDC)
Max. ON/OFF Switching Mechanically Electrically	 300 operation/min 30 operation/min
Ambient Temperature	-25ºC to +70ºC
Operating Humidity	45 to 85% RH
Vibration Endurance Error Operation	 10 to 55Hz Double Amplitude 1.5mm 10 to 55Hz Double Amplitude 1.5mm
Shock Endurance Error Operation	 100G Min. 10G Min.
Life Expectancy Mechanically Electrically	 10^7 operations. Min. (no load) 10^5 operations. Min. (at rated coil voltage)
Weight	abt. 10grs.

9.REFERENCE DATA

Coil Temperature Rise

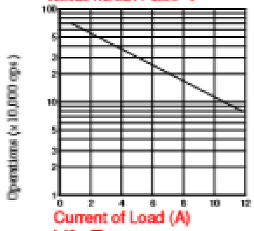


Operation Time

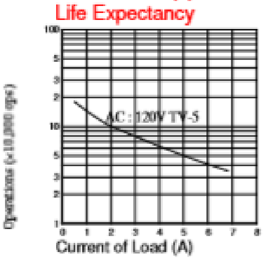


Life Expectancy

AC120V/DC24V cosΦ=1



Life Expectancy

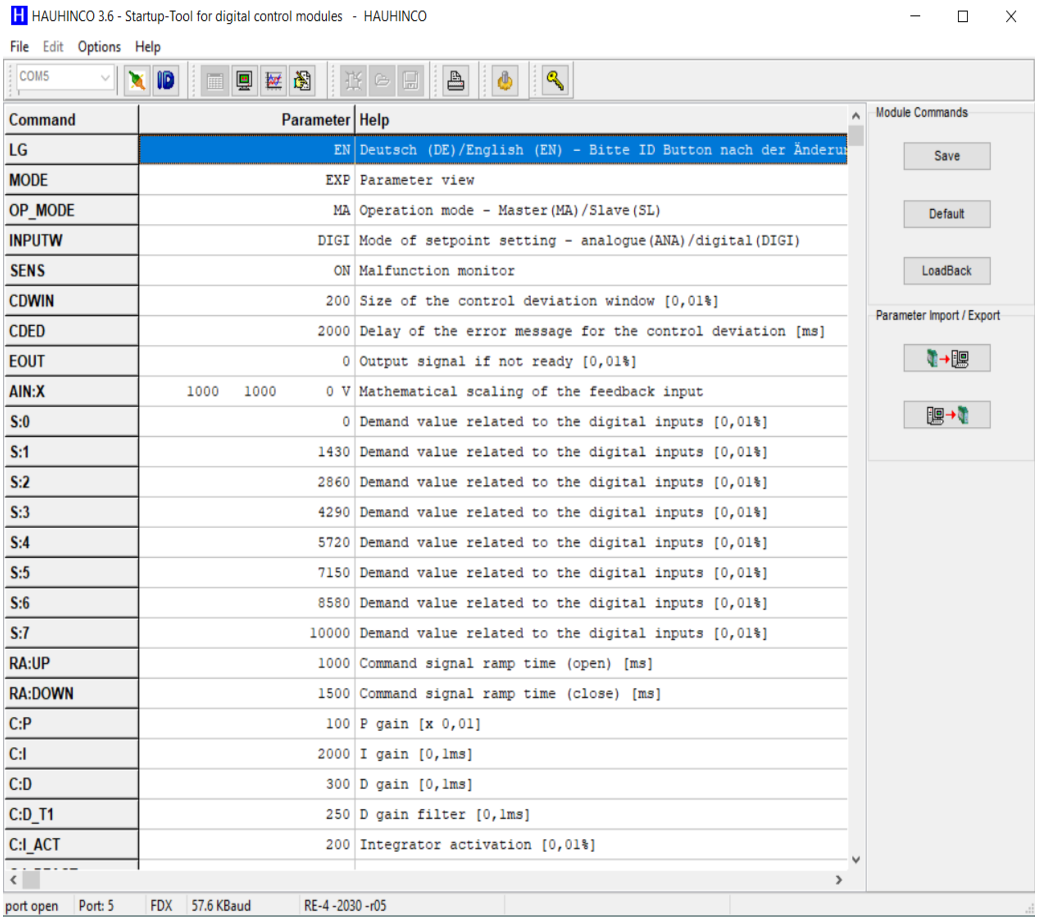


2

APPENDIX A

This space will be used to provide the code that was discussed in chapter 3 adequately.

a) RE4 Controller program



Command	Parameter	Help
LG	EN	Deutsch (DE)/English (EN) - Bitte ID Button nach der Änderu
MODE	EXP	Parameter view
OP_MODE	MA	Operation mode - Master(MA)/Slave(SL)
INPUTW	DIGI	Mode of setpoint setting - analogue(ANA)/digital(DIGI)
SENS	ON	Malfunction monitor
CDWIN	200	Size of the control deviation window [0,01%]
CDED	2000	Delay of the error message for the control deviation [ms]
EOUT	0	Output signal if not ready [0,01%]
AIN:X	1000 1000 0 V	Mathematical scaling of the feedback input
S:0	0	Demand value related to the digital inputs [0,01%]
S:1	1430	Demand value related to the digital inputs [0,01%]
S:2	2860	Demand value related to the digital inputs [0,01%]
S:3	4290	Demand value related to the digital inputs [0,01%]
S:4	5720	Demand value related to the digital inputs [0,01%]
S:5	7150	Demand value related to the digital inputs [0,01%]
S:6	8580	Demand value related to the digital inputs [0,01%]
S:7	10000	Demand value related to the digital inputs [0,01%]
RA:UP	1000	Command signal ramp time (open) [ms]
RA:DOWN	1500	Command signal ramp time (close) [ms]
C:P	100	P gain [x 0,01]
C:I	2000	I gain [0,1ms]
C:D	300	D gain [0,1ms]
C:D_T1	250	D gain filter [0,1ms]
C:I_ACT	200	Integrator activation [0,01%]

C:I_DEACT	500	Integrator stop [0,01%]
C:I_LIM	2500	Integrator limitation [0,01%]
C_EXT:P	0	PT1: P gain [x 0,01]
C_EXT:T1	200	PT1: P gain filter [0,1ms]
C_EXT:FF	0	Feed forward [0,01%]
MIN:A	0	Deadband compensation [0,01%]
MIN:B	0	Deadband compensation [0,01%]
MAX:A	10000	Output scaling [0,01%]
MAX:B	10000	Output scaling [0,01%]
TRIGGER	200	Deadband compensation trigger point [0,01%]
STDBY	0	Standby output signal for pre magnetization [0,01%]
VA_OFFSET	1450	Offset value for zero position of the valve[0,01%]
SIGNAL:U	POS-15	Type of the controller output
SIGNAL:M	WA	Selection of the monitor signal (Analogue output)
CURRENT	2500	Rated solenoid current [mA]
FG:SGT	RECT	Type of signal for the signal generator - triangle(TRI) or r
FG:OFF	5000	Offset value for the signal generator [0,01%]
FG:AMP	5000	Amplitude (peak to peak) for the signal generator [0,01%]
FG:FRQ	500	Frequency for the signal generator [mHz]

Parameter Import / Export

b) Arduino® code

arduino_valve

```
#include <SPI.h> //This library allows you to communicat
#include "Adafruit_MAX31855.h"

int aux;
String comando;
String  tempoespera;
int tempo; float celsius1;
float celsius2; float celsius3;
float celsius4; float valormaximo1;
float valormaximo2; boolean valorrecebido;
float valormaximo;

#define MAXDO  3 //pwm
#define MAXCS1  12
#define MAXCS2  10
#define MAXCS3  9
#define MAXCS4  2
#define MAXCLK  5 //pwm

Adafruit_MAX31855 thermocouple1(MAXCLK, MAXCS1, MAXDO);
Adafruit_MAX31855 thermocouple2(MAXCLK, MAXCS2, MAXDO);
Adafruit_MAX31855 thermocouple3(MAXCLK, MAXCS3, MAXDO);
Adafruit_MAX31855 thermocouple4(MAXCLK, MAXCS4, MAXDO);

void setup()
{
  Serial.begin(9600);
  pinMode(8, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(13, OUTPUT);
  Serial.println("MAX31855 test");
  delay(1000);
}
void loop() {
  digitalWrite(13, LOW);
  valorrecebido = false;
  if (Serial.available() > 0)
  {
    Serial.setTimeout(100);
    tempoespera = Serial.readString();
    delay (100);
    tempo = tempoespera.toInt();
    valorrecebido = true;
    while (valorrecebido == true)
    {
      comando = Serial.readString(); //Reads characters
```

```
if (comando == "End")
{
  Serial.println("Waiting for reading interval");
  digitalWrite(13, HIGH);
  digitalWrite(8, LOW);
  digitalWrite(7, LOW);
  digitalWrite(4, LOW);
  valorrecebido= false;
  break;
}
if (comando == "Automatic")
{ aux = 0;
  while (aux == 0)
  {
    Serial.print("Temperature thermocouple 1 - ");
    celsius1 = thermocouple1.readCelsius();
    if (isnan(celsius1)) {
      celsius1 = 0;
      Serial.println("Something wrong with the thermocouple");
    }
    else {
      Serial.print("C = ");
      Serial.println(celsius1);
    }
    delay(10);

    Serial.print("Temperature thermocouple 2 - ");
    celsius2 = thermocouple2.readCelsius();
    if (isnan(celsius2)) {
      celsius2 = 1;
      Serial.println("Something wrong with the thermocouple");
    }
    else {
      Serial.print("C = ");
      Serial.println(celsius2);
    }
    delay(10);

    Serial.print("Temperature thermocouple 3 - ");
    celsius3 = thermocouple3.readCelsius();
    if (isnan(celsius3)) {
      celsius3 = 0;
      Serial.println("Something wrong with the thermocouple");
    }
    else {
      Serial.print("C = ");
      Serial.println(celsius3);
    }
    delay(10);
```

```
Serial.print("Temperature thermocouple 4 - ");
celsius4 = thermocouple4.readCelsius();
if (isnan(celsius4)) {
  celsius4 = 0.5;
  Serial.println("Something wrong with the thermocouple")
}
else {
  Serial.print("C = ");
  Serial.println(celsius4);
}
delay(tempo);

valormaximo1 = max (celsius1, celsius2);
delay(10);
valormaximo2 = max (celsius3, celsius4);
delay(10);
valormaximo = max (valormaximo1, valormaximo2);
delay (10);

if (valormaximo < 75) // CLOSED
{
  Serial.println("VALVE CLOSED");
  digitalWrite(8, LOW);
  digitalWrite(7, LOW);
  digitalWrite(4, LOW);//000
  delay (10);
}
if (valormaximo >= 75 && valormaximo < 115) // pos. 1
{
  Serial.println("1 OPENING POSITION");
  digitalWrite(8, HIGH);
  digitalWrite(7, LOW);
  digitalWrite(4, LOW);//100
  delay (10);
}
if (valormaximo >= 115 && valormaximo < 200) // pos. 2
{
  Serial.println("2 OPENING POSITION");
  digitalWrite(8, LOW);
  digitalWrite(7, LOW);
  digitalWrite(4, HIGH);//010
  delay (10);
}
if (valormaximo >= 200 && valormaximo < 300) // pos. 3
{
  Serial.println("3 OPENING POSITION");
  digitalWrite(8, HIGH);
  digitalWrite(7, LOW);
  digitalWrite(4, HIGH);//110
  delay (10);
}
if (valormaximo >= 300 && valormaximo < 415) // pos. 4)
{
  Serial.println("4 OPENING POSITION");
  digitalWrite(8, LOW);
  digitalWrite(7, HIGH);
  digitalWrite(4, LOW);//001
  delay (10);
}
if (valormaximo >= 415 && valormaximo < 520) // pos. 5)
{
  Serial.println("5 OPENING POSITION");
  digitalWrite(8, HIGH);
  digitalWrite(7, HIGH);
  digitalWrite(4, LOW);//101
  delay (10);
}
if (valormaximo >= 520 && valormaximo < 600) // pos. 6)
{
  Serial.println("6 OPENING POSITION");
  digitalWrite(8, LOW);
  digitalWrite(7, HIGH);
  digitalWrite(4, HIGH);//011
  delay (10);
}
if (valormaximo >= 600) //open
{
  Serial.println("VALVE TOTALLY OPEN");
  digitalWrite(8, HIGH);
  digitalWrite(7, HIGH);
  digitalWrite(4, HIGH);//111
  delay (10);
}
comando = Serial.readString();

if (comando == "Stop")
{
  aux = 1;
  Serial.println("AUTOMATIC MODE STOPPED");
}
delay (tempo);
  }
}

if (comando == "Manual")
{ aux = 1;
  while (aux == 1)
  {
```

```
Serial.print("Temperature thermocouple 1 - ");
celsius1 = thermocouple1.readCelsius();
if (isnan(celsius1)) {
  celsius1 = 0;
  Serial.println("Something wrong with the thermocoupl
}
else {
  Serial.print("C = ");
  Serial.println(celsius1);
}
delay(10);

Serial.print("Temperature thermocouple 2 - ");
celsius2 = thermocouple2.readCelsius();
if (isnan(celsius2)) {
  celsius2 = 0;
  Serial.println("Something wrong with the thermocoupl
}
else {
  Serial.print("C = ");
  Serial.println(celsius2);
}
delay (10);

Serial.print("Temperature thermocouple 3 - ");
celsius3 = thermocouple3.readCelsius();
if (isnan(celsius3)) {
  celsius3 = 0;
  Serial.println("Something wrong with the thermocoupl
}
else {
  Serial.print("C = ");
  Serial.println(celsius3);
}
delay (10);

Serial.print("Temperature thermocouple 4 - ");
celsius4 = thermocouple4.readCelsius();
if (isnan(celsius4)) {
  celsius4 = 0;
  Serial.println("Something wrong with the thermocoupl
}
else {
  Serial.print("C = ");
  Serial.println(celsius4);
}
delay (tempo);

comando = Serial.readString();
```

```
if (comando == "Close")
{
  Serial.println("VALVE CLOSED");
  digitalWrite(8, LOW);
  digitalWrite(7, LOW);
  digitalWrite(4, LOW);//000
  delay (30);
}
if (comando == "Open 1")
{
  Serial.println("1 OPENING POSITION"); // po:
  digitalWrite(8, HIGH);
  digitalWrite(7, LOW);
  digitalWrite(4, LOW);//100
  delay (30);
}
if (comando == "Open 2") // pos. 2
{
  Serial.println("2 OPENING POSITION");
  digitalWrite(8, LOW);
  digitalWrite(7, LOW);
  digitalWrite(4, HIGH);//010
  delay (30);
}
if (comando == "Open 3") // pos. 3
{
  Serial.println("3 OPENING POSITION");
  digitalWrite(8, HIGH);
  digitalWrite(7, LOW);
  digitalWrite(4, HIGH);//110
  delay (30);
}
if (comando == "Open 4") // pos. 4
{
  Serial.println("4 OPENING POSITION");
  digitalWrite(8, LOW);
  digitalWrite(7, HIGH);
  digitalWrite(4, LOW);//001
  delay (30);
}
if (comando == "Open 5") // pos. 5
{
  Serial.println("5 OPENING POSITION");
  digitalWrite(8, HIGH);
  digitalWrite(7, HIGH);
  digitalWrite(4, LOW);//101
  delay (30);
}
if (comando == "Open 6") // pos. 6
{
  Serial.println("6 OPENING POSITION");
  digitalWrite(8, LOW);
  digitalWrite(7, HIGH);
  digitalWrite(4, HIGH);//011
```

```
          if (comando == "Opened")
          {
            Serial.println("VALVE TOTALLY OPEN");
            digitalWrite(8, HIGH);
            digitalWrite(7, HIGH);
            digitalWrite(4, HIGH);//111
            delay (30);
          }
          if (comando == "Stop")
          {
            aux = 0;
            Serial.println("MANUAL MODE STOPPED");
          }
          delay (tempo);
        }
      }
    }
  }
}
```

c) Server code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using System.IO.Ports;
using System.Threading;

namespace ApptoArduino
{
    public partial class Form1 : Form
    {
        string auxstring; Boolean Stoppushed; int inicioprograma; Boolean connectpremido;

        public Form1()
        {
            InitializeComponent();
            TimerCOM.Enabled = true; Stoppushed = false; inicioprograma = 0;
            CMBintervalos.Items.Add("1,5s");
            CMBintervalos.Items.Add("6s");
            CMBintervalos.Items.Add("12s");CMBintervalos.SelectedIndex = 0;
        }
```

```
        private void atualizarportascom()
        {
            int contador;
            bool Nportasdif;

            contador = 0;
            Nportasdif = false;
            if (CmbCOMs.Items.Count == SerialPort.GetPortNames().Length)
            {
                foreach (string s in SerialPort.GetPortNames()) // foreac
                {
                    if (CmbCOMs.Items[contador++].Equals(s) == false) //V
                    {
                        Nportasdif = true;
                    }
                }
            }
            else
            {
                Nportasdif = true;
            }
            if (Nportasdif == false)
            {
                return; // The return statement terminates execution of t
            }
            CmbCOMs.Items.Clear();

            foreach (string s in SerialPort.GetPortNames())
            {
                CmbCOMs.Items.Add(s);
            }
            CmbCOMs.SelectedIndex = 0;
        }
```

```
        private void Timer_Tick(object sender, EventArgs e)
        {
            atualizarportascom();
        }

        private void BtnConnect_Click(object sender, EventArgs e)
        {
            if (serialPort.IsOpen == false)
            {
                try
                {serialPort.PortName = CmbCOMs.Items[CmbCOMs.SelectedIndex].ToString();
                    serialPort.Open(); }
                catch
                {
                    return;
                }
                if (serialPort.IsOpen == true)
                {
                    CMBintervalos.Enabled = false; BtnConnect.Text = "Disconnect";
                    connectpremido = true;
                    CmbCOMs.Enabled = false; btnclean.Enabled = true;
                    BtnManual.Enabled = true; BtnAuto.Enabled = true; BtnStop.Enabled = true;
                    TxtServidor.AppendText(DateTime.Now + " " + "Connection established");
                    TxtServidor.AppendText(Environment.NewLine);
                }
            }
            else
            {
                try
                {

                    CmbCOMs.Enabled = true;
                    BtnConnect.Text = "Connect"; connectpremido = false;
                    CMBintervalos.Enabled = true;
                    BtnManual.Enabled = false; BtnAuto.Enabled = false; BtnStop.Enabled = false; TimerLeituras.Stop();
                    TxtServidor.AppendText(DateTime.Now + " " + "Connection failed");
                    TxtServidor.AppendText(Environment.NewLine);
                }
                catch
                {
                    return;
                }
            }
             if (connectpremido == false )
             {
                serialPort.Write("End");
                serialPort.Close();
             }
```

```
                if (CMBintervalos.SelectedIndex == 0 & connectpremido == true)
                {
                    TimerLeituras.Interval = 1535;
                    serialPort.Write("1500");
                }
                else if (CMBintervalos.SelectedIndex == 1 & connectpremido == true)
                {
                    TimerLeituras.Interval = 6035;
                    serialPort.Write("6000");
                }
                else if (CMBintervalos.SelectedIndex == 2 & connectpremido == true)
                {
                    TimerLeituras.Interval = 12035;
                    serialPort.Write("12000");
                }

            }

        private void Form1_FormClosed(object sender, FormClosedEventArgs e)
            {
                if (serialPort.IsOpen == true)
                {
                    serialPort.Close();// proteção para que o programa não seja fecl
                }
            }

            private void BtnExit_Click(object sender, EventArgs e)
            {
                Close();
                serialPort.Close(); TimerLeituras.Stop(); TimerCOM.Stop();
             }
```

```
private void BtnManual_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true & Stoppushed == false & inicioprograma == 1 )
    {
        serialPort.Write("Stop");
        TimerLeituras.Stop();
        TxtServidor.AppendText(DateTime.Now + " " + "Stop button pushed");
        TxtServidor.AppendText(Environment.NewLine);
        Thread.Sleep(2*TimerLeituras.Interval);
        serialPort.Write("Manual");
        TimerLeituras.Start();
        TxtServidor.AppendText(DateTime.Now + " " + "Manual mode activated");
        TxtServidor.AppendText(Environment.NewLine);
        Stoppushed = false;
    }
    else if (serialPort.IsOpen == true)
    {
        serialPort.Write("Manual");
        TimerLeituras.Start();
        TxtServidor.AppendText(DateTime.Now + " " + "Manual mode activated");
        TxtServidor.AppendText(Environment.NewLine);
        Stoppushed = false; inicioprograma = 1;
    }
    BtnClose.Enabled = true; BtnOpen1.Enabled = true;
    BtnOpen.Enabled = true; BtnOpen2.Enabled = true;
    BtnOpen3.Enabled = true; BtnOpen4.Enabled = true;
    BtnOpen5.Enabled = true; BtnOpen6.Enabled = true;
}

private void BtnStop_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true)
    {
        serialPort.Write("Stop");
        TxtServidor.AppendText(DateTime.Now + " " + "Stop button pushed");
        TxtServidor.AppendText(Environment.NewLine);
        TimerLeituras.Stop();
        Stoppushed = true;
    }
    BtnClose.Enabled = false; BtnOpen1.Enabled = false;
    BtnOpen.Enabled = false; BtnOpen2.Enabled = false;
    BtnOpen3.Enabled = false; BtnOpen4.Enabled = false;
    BtnOpen5.Enabled = false; BtnOpen6.Enabled = false;
}
```

```
private void BtnAuto_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true & Stoppushed == false & inicioprograma == 1)
    {
        serialPort.Write("Stop");
        TimerLeituras.Stop();
        TxtServidor.AppendText(DateTime.Now + " " + "Stop button pushed");
        TxtServidor.AppendText(Environment.NewLine);
        Thread.Sleep(2 * TimerLeituras.Interval);
        serialPort.Write("Automatic");
        TimerLeituras.Start();
        TxtServidor.AppendText(DateTime.Now + " " + "Automatic mode activated");
        TxtServidor.AppendText(Environment.NewLine);
        Stoppushed = false; inicioprograma = 1;
    }
    else
    {
        serialPort.Write("Automatic");
        TimerLeituras.Start();
        TxtServidor.AppendText(DateTime.Now + " " + "Automatic mode activated");
        TxtServidor.AppendText(Environment.NewLine);
        Stoppushed = false; inicioprograma = 1;
    }
    BtnClose.Enabled = false; BtnOpen1.Enabled = false;
    BtnOpen.Enabled = false; BtnOpen2.Enabled = false;
    BtnOpen3.Enabled = false; BtnOpen4.Enabled = false;
    BtnOpen5.Enabled = false; BtnOpen6.Enabled = false;
}

private void TimerLeituras_Tick(object sender, EventArgs e)
{
if (serialPort.IsOpen == true)
    {
        auxstring = serialPort.ReadExisting(); //This method returns the contents
        TxtMensagens.AppendText(DateTime.Now + Environment.NewLine + auxstring);
    }
}
```

```
private void BtnClose_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true)
    {
        serialPort.Write("Close");
        TxtServidor.AppendText(DateTime.Now + " " + "Valve closed");
        TxtServidor.AppendText(Environment.NewLine);
    }
}

private void BtnOpen1_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true)
    {
    serialPort.Write("Open 1");
    TxtServidor.AppendText(DateTime.Now + " " + "1 opening position");
    TxtServidor.AppendText(Environment.NewLine);
    }
}

private void BtnOpen2_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true)
    {
    serialPort.Write("Open 2");
    TxtServidor.AppendText(DateTime.Now + " " + "2 opening position");
    TxtServidor.AppendText(Environment.NewLine);
    }
}

private void BtnOpen3_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true)
    {
    serialPort.Write("Open 3");
    TxtServidor.AppendText(DateTime.Now + " " + "3 opening position");
    TxtServidor.AppendText(Environment.NewLine);
    }
}
```

```
private void BtnOpen_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true)
    {
    serialPort.Write("Opened");
    TxtServidor.AppendText(DateTime.Now + " " + "Valve totally open");
    TxtServidor.AppendText(Environment.NewLine);
    }
}

private void BtnOpen4_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true)
    {
    serialPort.Write("Open 4");
    TxtServidor.AppendText(DateTime.Now + " " + "4 opening position");
    TxtServidor.AppendText(Environment.NewLine);
    }
}

private void BtnOpen5_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true)
    {
    serialPort.Write("Open 5");
    TxtServidor.AppendText(DateTime.Now + " " + "5 opening position");
    TxtServidor.AppendText(Environment.NewLine);
    }
}

private void BtnOpen6_Click(object sender, EventArgs e)
{
    if (serialPort.IsOpen == true)
    {
    serialPort.Write("Open 6");
    TxtServidor.AppendText(DateTime.Now + " " + "6 opening position");
    TxtServidor.AppendText(Environment.NewLine);
    }
}
```

```
private void saveDataToolStripMenuItem_Click(object sender, EventArgs e)
{
    SaveFileDialog gravar = new SaveFileDialog(); // Cria SaveFileDialog
    gravar.Filter = "Text files (*.txt)|*.txt"; // Mostra apenas os text f
    gravar.DefaultExt = "txt";
    DialogResult saved = gravar.ShowDialog();

    if (saved == DialogResult.OK)
    {
        StreamWriter Ficheiro = null; //Null keyword is a literal that rep
        try
        {
            Ficheiro = new StreamWriter(gravar.FileName); // Implements a
            Ficheiro.WriteLine(TxtMensagens.Text);
        }
        catch (IOException ex)
        {
            MessageBox.Show("IOException" + ex.Message); //IOException is
        }
        catch (Exception ex)
        {
            MessageBox.Show("Exception" + ex.Message); //Mostra mensagem c
        }
        finally
        {
            if (Ficheiro != null)
                Ficheiro.Close(); // Fecha a instancia StreamWriter, com a
        }
    }
}
```

```
private void saveServerMessagesToolStripMenuItem_Click(object sender, EventArgs e)
{
    SaveFileDialog gravar = new SaveFileDialog(); // Cria SaveFileDialog
    gravar.Filter = "Text files (*.txt)|*.txt"; // Mostra apenas os text files
    gravar.DefaultExt = "txt";
    DialogResult saved = gravar.ShowDialog();

    if (saved == DialogResult.OK)
    {
        StreamWriter Ficheiro = null;
        try
        {
            Ficheiro = new StreamWriter(gravar.FileName);
            Ficheiro.WriteLine(TxtServidor.Text);
        }
        catch (IOException ex)
        {
            MessageBox.Show("IOException" + ex.Message);
        }
        catch (Exception ex)
        {
            MessageBox.Show("Exception" + ex.Message);
        }
        finally
        {
            if (Ficheiro != null)
                Ficheiro.Close();
        }
    }
}
```

```
private void saveDataToolStripMenuItem1_Click(object sender, EventArgs e)
{
    SaveFileDialog gravar = new SaveFileDialog(); // Cria SaveFileDialog
    gravar.Filter = "Text files (*.txt)|*.txt"; // Mostra apenas os text files
    gravar.DefaultExt = "txt"; DialogResult saved = gravar.ShowDialog();

    if (saved == DialogResult.OK)
    {
        StreamWriter Ficheiro = null;
        try
        {
            TextReader ler = new System.IO.StringReader(TxtMensagens.Text);
            string[] frase = new string[TxtMensagens.Lines.Length];
            Ficheiro = new StreamWriter(gravar.FileName);
            for (int i =0; i< TxtMensagens.Lines.Length-1; i++)
            {
                frase[i] = ler.ReadLine();
                char[] letra = new char[frase[i].Length];
                int position = 0; int posiçãonum = 0; Boolean Temnumeros = false;
                foreach (char c in frase[i])
                {
                    letra[position] = c;
                    position++;
                    if ("=".Contains(c))
                    { posiçãonum = position;
                        Temnumeros = true;
                    }
                }
                if (Temnumeros == true)
                {
                    for (int x=posiçãonum; x< frase[i].Length; x++)
                    {
                    Ficheiro.Write(letra[x]);
                    }
                    Ficheiro.WriteLine();
                    Temnumeros = false;
                }
            }
        }
        catch (IOException ex)
        {
            MessageBox.Show("IOException" + ex.Message);
        }
        catch (Exception ex)
        {
            MessageBox.Show("Exception" + ex.Message);
        }
        finally
        {
            if (Ficheiro != null)
                Ficheiro.Close();
        }
    }
}
```

APPENDIX B

In this space will be presented the experimental procedure used in chapter 3 experiments:

Objective:

- Compare the volume of water wasted, the state of the fabric and the distribution of temperatures

1. TEST A

a) Mount the flat spray

b) Connect the hoses: pump-valve and valve-flat spray

c) Determine the temperature and relative humidity of the air

d) Fill the dry fuel basket and calculate its weight (to have a reference for the other tests). Introduce a thermocouple on the basket

e) Place IR and video camera in the rearview of the fabric

f) Install flux meter near the fabric

g) Weigh and then place the container where the excess water from the fabric may fall

h) Activate the wind tunnel to a constant speed of 2m/s

i) Impose a pressure of 2 bar on the pump and a flow rate of 1.71 l/min/m of barrier

j) Give the fully open instruction to the valve

k) Start data acquisition

l) Fuel ignition and measuring combustion time with a stopwatch (120s)

m) Shut down the wind tunnel

n) Turn off the pump

2. TEST B and TEST C

a) Mount the flat spray

b) Connect the hoses: pump-valve and valve-flat spray

c) Determine the temperature and relative humidity of the air

d) Fill the dry fuel basket and calculate its weight (to have a reference for the other tests). Introduce a thermocouple on the basket

e) Place IR and video camera in the rearview of the fabric

f) Install flux meter near the fabric

g) Weigh and then place the container where the excess water from the fabric may fall

h) Activate the wind tunnel to a constant speed of 2m/s

i) Impose a pressure of 2 bar on the pump and a flow rate of 1.71 l/min/m of barrier

j) Activate automatic valve mode

k) Start data acquisition

l) Fuel ignition and measuring combustion time with a stopwatch (120s)

m) Shut down the wind tunnel

n) Turn off the pump

Material:

1. Pump e DN 10 hoses;
2. Weighing machine;
3. Flux meter
4. IR and video camera;
5. Stopwatch
6. At least 3 fabrics;
7. Fuel;
8. Breadboard, amplifiers e jumpers
9. Thermocouples
10. Acquisition board