# Automation of an Optical Measurement Workbench using LabVIEW

*André Ricardo Oliveira Cardoso*

**Dissertação de Mestrado**

Supervisor at FEUP: Prof. Joaquim Gabriel Magalhães Mendes

Supervisor in the company: Eng. Carlos Oliveira


**Mestrado Integrado em Engenharia Mecânica**

**Especialização em Automação**

2020-09-21

## Abstract

The correct optical evaluation of digital instrument clusters present in modern cars is part of the quality validation in the car's industry. In this context, the displays are subject to a wide range of no contact measurements to assure the full verification of key points.

This dissertation describes the automation of a manual workbench designed for digital instrument cluster tests. The initial machine was composed by three linear axes and two angular rotations, all of them actuated manually. Thus, the movement of each axis was performed one at a time, therefore taking a lot of time to adjust the desired position and presenting low accuracy and repeatability. In this dissertation the hand cranks of the three linear axes and of one rotational axis were replaced by motors and drivers; it was also installed nine optical limit switches and one emergency button. The machine user interface was developed in LabVIEW 2017, which is connected to the workbench through an Arduino Mega 2560 board. The communication between the LabVIEW and the Arduino was based on the library LINX.

The project took place over several phases. First, the collection of information of similar workbenches and their positioning specifications and feature requirements. Secondly, the selection of the motors and hardware necessary for the automation of the setup. Then, a few mechanical parts were designed to assemble the motors and other parts to the workbench. Afterwards, an emergency stop button was installed.

The LabVIEW software enables the command of each axes through three different actuation modes: “Automatic mode”, “Manual mode”, and by recorded positions. To evaluate the accomplishment of the objectives and the correct behavior of the system, a set of tests was performed: electrical shut down during the movement, position repeatability (standard deviation less than 0.007 mm) and position resolution (0.02 mm).

The project was successfully concluded; it is now possible to automatically command the position of the axes in less than 30 seconds. The user interface was evaluated by the machine operators and their suggestions incorporated in the final version to make it friendly and easy to use.

## Automatização de uma Bancada de Medições Óticas utilizando LabVIEW

## Resumo

A correta avaliação ótica dos painéis de instrumentos digitais presentes nos automóveis modernos assume um papel de grande relevância na validação de qualidade da indústria automóvel. Nesse sentido, os visores são sujeitos a uma vasta gama de medições sem contacto de forma a assegurar a verificação completa de caraterísticas importantes.

Esta dissertação descreve a automatização de uma bancada manual concebida para testes de painéis de instrumentos digitais. A máquina inicialmente era composta por três eixos lineares e dois eixos angulares, todos eles acionados manualmente. Assim, o movimento da bancada era realizado eixo a eixo, demorando muito tempo a ajustar a posição desejada e apresentando baixa precisão e repetibilidade. Nesta dissertação, as manivelas dos três eixos lineares e de um eixo de rotação foram substituídas por motores e controladores; foram também instalados nove sensores óticos de posição e um botão de emergência. A interface do utilizador da máquina foi desenvolvida em LabVIEW 2017, que está conectada à bancada através de uma placa Arduíno Mega 2560. A comunicação entre o LabVIEW e o Arduíno foi baseada na biblioteca LINX.

O projeto desenvolveu-se ao longo de várias fases. Primeiro, na recolha de informação de bancadas semelhantes e das suas especificações de posicionamento. Em segundo, no dimensionamento dos motores e do hardware necessários para a automatização do movimento. Depois, algumas peças mecânicas foram desenhadas e concebidas para fixar os motores e os sensores à bancada. Posteriormente, foi instalado um botão de emergência.

O software LabVIEW permite o comando de cada eixo através de três modos de acionamento diferentes: "Modo automático", "Modo manual", e por posições previamente registadas. Para avaliar o cumprimento dos objetivos e o comportamento correto do sistema, foi realizado um conjunto de testes: corte de energia durante o movimento, repetibilidade (desvio padrão inferior a 0,007 mm) e resolução (0,02 mm).

O projeto foi concluído com sucesso; é agora possível comandar automaticamente a posição dos eixos em menos de 30 segundos. A interface do utilizador foi avaliada pelos operadores da máquina e as suas sugestões incorporadas na versão final, tornando-a assim mais apelativa e fácil de utilizar.

## Acknowledgments

First of all, I would like to thank Professor Joaquim Gabriel Mendes for his encouragement and support as the supervisor of this project. I sincerely thank him for his dedication and sharing of knowledge and skills throughout this dissertation.

Secondly, to Engineer Carlos Oliveira, for all his availability and all the follow-up during these months. As a mentor in the company, he has always been accessible and did everything to make sure the project ever had what it needed.

Special thanks also to Engineer Rui Barros, for his sharing of experience and the opportunity to develop this project. To Engineer Manuel Sarmento, for always being involved and helping in a pleasant way to any kind of issues. To all the other ECM3 members, also a message of gratitude for the way they hosted me inside the team, making me feel welcomed and comfortable during all my internship.

I'm also thankful to my friends, who have accompanied me in this personal and academic journey over the past few years. Last but not least, a big thank you to my family, especially my parents and sisters, for the inspiration they transmit me and for always giving me the best.

# Contents

# List of Figures

# List of Tables

# Acronyms

| | |
|---|---|
| CHUD | Car Head-up Display |
| DFF | Deutsches Flachdisplay Forum |
| ICDM | International Committee for Display Metrology |
| IDE | Integrated Development Environment |
| LabVIEW | Laboratory Virtual Instrument Engineering Workbench |
| LIFA | LabVIEW interface for Arduino |
| LMD | Light Measurement Device |
| VI | Virtual Instrument |

# 1 Introduction

This chapter aims to illustrate the context in which this work was carried out, the project at Bosch Car Multimedia Portugal, S.A., where the internship took place, and the objectives proposed to the dissertation project. The methodology, project execution, and dissertation organization are also discussed in the course of this section.

## 1.1 Project Motivation

Nowadays, the displays are widely used in the automotive industry. The rising number of digital instrument clusters, and other types of screens like mirror displays and car head-up displays (CHUD's) are just some examples where it is important to warranty the safety. It is where light measurement devices like photometers, spectrometers, and colourimeters take much importance. These tools perform a wide variety of light measurement tests, which ensures that all the monitors present in the market and installed in cars are in perfect conditions.

This work took place at Bosch Car Multimedia Portugal, S.A., in the team ECM3, where it was possible to see up close the development of CHUD's and digital instrument clusters to the automotive area. With the purpose of making the light measurement tests quicker, the goal of the internship was to make the automation of the manual testing workbench from the Optical Laboratory, using an integration communication in LabVIEW.

The project, besides being stimulating, it was challenging, mostly because it gave me the possibility of work not only in the mechanical area but also in electronics and software. The opportunity to automate something important and be inside the automotive industry, truly motivated me.

## 1.2 BOSCH Group

Started with a workshop in Stuttgart in 1886, Robert Bosch took the first steps towards what would become today a multinational company. Following an ideology of achieving top quality and reliability of its products, BOSCH Group ensures the creation of technology that is "Invented for Life", as its slogan describes [1].

Being a global company placed in 126 locations worldwide, BOSCH Group employs approximately 400 thousand associates and its operations are divided into four business sectors [1]:

1. Mobility Solutions;

2. Industrial Technology;
3. Consumer Goods;
4. Energy and Building Technology.

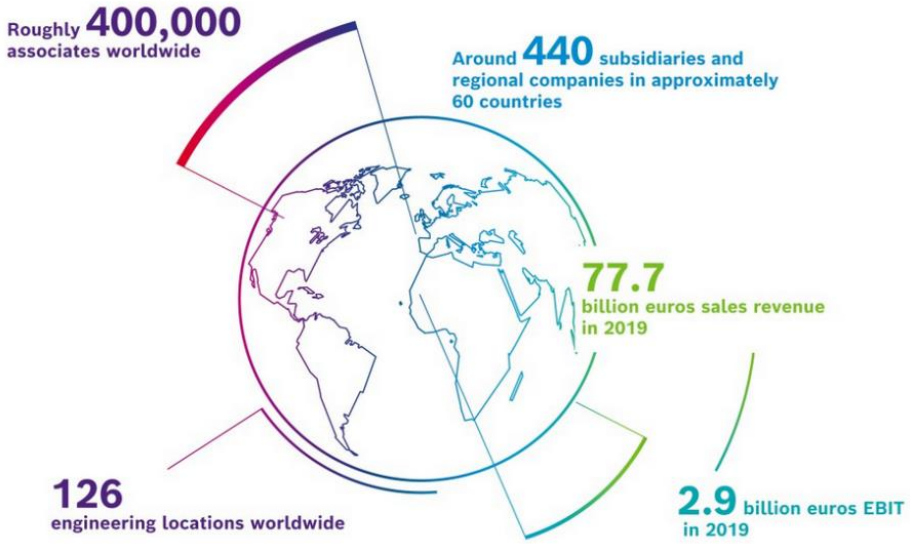



Figure 1.1 - Bosch numbers at a glance [2]

In 2019, the company sales revenue was 77.7 billion euros, being 60% from the mobility solutions sector, 23% from consumer goods sector, 10% from industrial technology and 7 % from energy and building technology [1].

## 1.3 BOSCH Car Multimedia Portugal, S.A.

In Portugal, the company employs 6360 collaborators spread by two headquarters in Lisbon and three main plants, BOSCH Termotecnologia, in Aveiro, BOSCH Security Systems, in Ovar, and BOSCH Car Multimedia, in Braga [1].

BOSCH Car Multimedia, created in 1990, is the largest and the leading plant of the Car Multimedia division in the BOSCH Group, being a reference in the global market of the car's electronics. It dedicates to the development and production of infotainment systems, instrumentation, and safety sensors for the automotive industry [1].

One of the recent investments was the creation of a development centre, shown in Figure 1.2 which has significant relevance in the Car Multimedia division.


Figure 1.2 - Bosch Development Centre

## 1.4 Optical Lab

Inside the Bosch Car Multimedia Portugal, S.A., and with the coordination of the ECM3 team, there is an Optical Lab started in 2004, which main goal is the validation of the instrument digital clusters produced by the Braga plant. Nowadays, to facilitate the optical tests, two testing workbenches are used: a robot-testing workbench and a manual testing workbench, as shown in the optical lab layout (Figure 1.3).

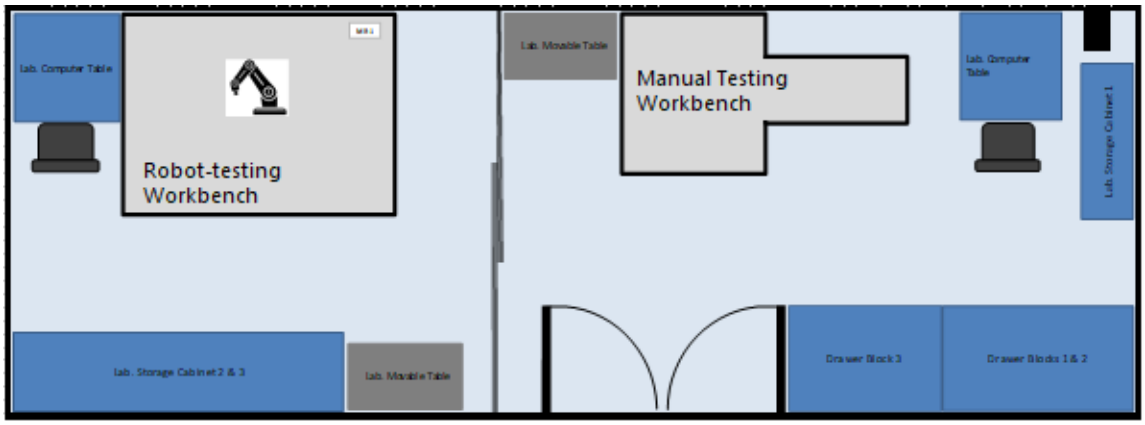



Figure 1.3 - Optical Lab layout

The robot-testing workbench includes a Staubli TX60 arm robot, which fixes the light measurement devices with its claw and moves them to the correct position and inclination angle to execute the tests. The manual testing workbench is also used for testing, however, the movement of this workbench is done by manual rotating three cranks, one for each of the three positioning axes, and by manual turning two angles. When the performance of both optical testing workbenches is compared, it is easy to notice that testing in the manual workbench takes longer, shows lower repeatability and reliability.

The current project focuses on this manual optical test workbench, shown in Figure 1.4, and in the improvement of its capabilities.

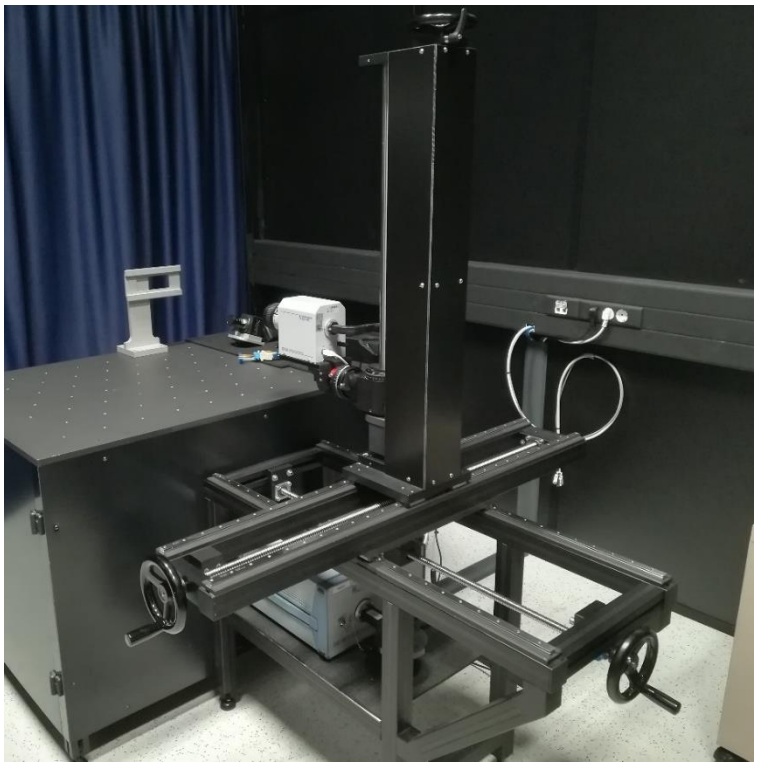


Figure 1.4 - Manual testing workbench in the beginning of the project

## 1.5 Project Goals

The project's main objective is to automate the manual testing workbench of the Bosch Optical Lab, including a command interface in LabVIEW, in all its three linear axes, to reduce the positioning time of the optical devices between the different light measurement tests and increase the consistency and reliability of the process. This objective requires the design, development and construction of a motion transmission system, based on the existing manual testing workbench, and the control of a group of motors to automate all the three linear axes of the workbench.

With an average of 700 mm of travel on each linear axis, the system goal is that the workbench could move from one point to the other within a maximum of 30 seconds, transporting a light measurement device which can weigh up to 8 kg. Given the needs of the optical test workbench, in terms of reliability it is proposed that the system has both repeatability and accuracy error of less than 0.5 mm, as well as a resolution of no more than 0.1 mm. Since the optical testing machine is located within the Bosch Development Centre, the mechanism should also not make any noise that could disturb the work of the engineers located around the optical laboratory.

## 1.6 Methodology

Before addressing any case scenario, it was necessary to understand the context where this testing workbench is integrated and its purposes. Knowing the light measurement devices they must carry, as well as, the tests and measurements it should perform, it is essential not only to be integrated and aware of the project background, but also to better understand the characteristics that the workbench must have. The second step is to realize what are the forces and torques applied in the system so that it is possible to choose the best fit motors to use as well as to design a transmission system for the workbench axes. After that, the support parts designed in the previous step can be developed. With the use of a stepper motor and a limit switch sensor from the optical lab, also the programming in LabVIEW of one workbench axis can be performed. With the arrival of the orders, the next step is to integrate all motors into a single LabVIEW program. With this program, the aim is to control the position on 3-axis of the optical workbench. After the LabVIEW program correct execution, the next level is the assembly of all motors, transmission systems, and supporting parts on the workbench. Some calibration tests must be performed so that the ranges of the motors are correctly defined, and the limit sensor of each axis is resetting the position counting to zero. Last, but not least, the workbench was validated.

Based on all the previous stages of the project's implementation, a flowchart was realized and can be observed in Figure 1.5.

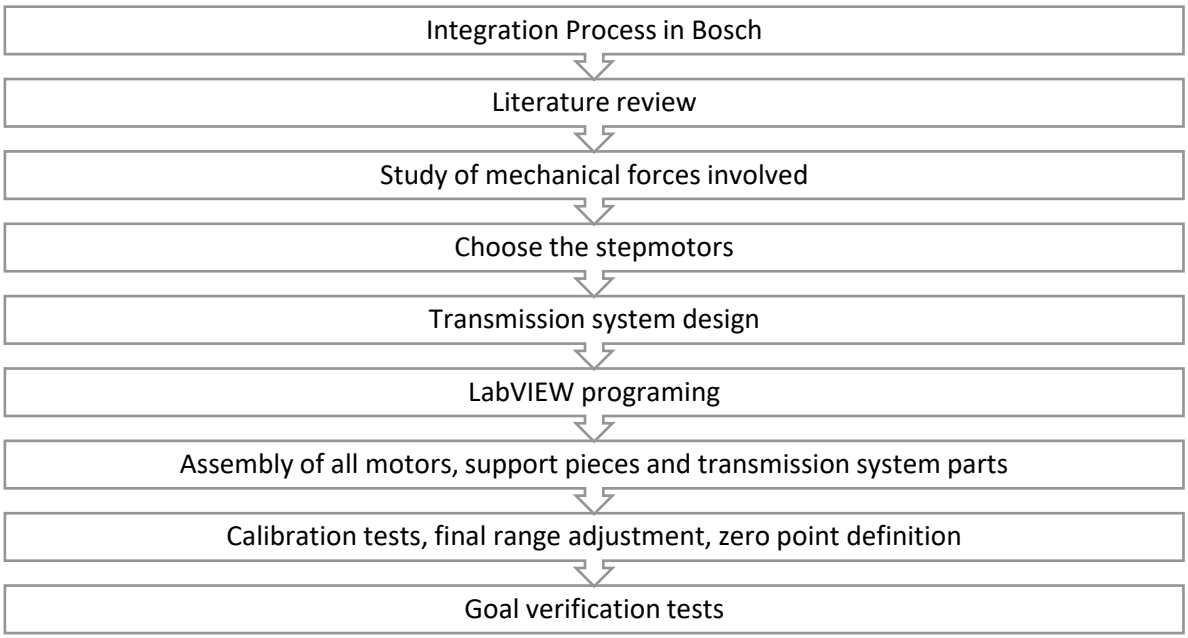



Figure 1.5 - Flowchart of the developed project

## 1.7 Dissertation Organization

The remaining chapters of the dissertation are organised as follows. In **chapter 2**, it will be realised a bibliographical review about the digital instrument clusters, the optical tests performed on them and the optical measurement workbenches. Afterwards, it will be presented the Bosch manual workbench and it will be described its main characteristics. In **chapter 3**, the solution proposal to achieve the initial objectives will be illustrated. Then, in **chapter 4**, the process of implementing the proposed solution found in the previous chapter will be described. Later, **chapter 5** describes the software design as well as explains the command interface of the mechanism. A set of validation tests are after performed, and their results are written in **chapter 6**.

Lastly, **chapter 7** addresses the conclusions of the work and proposals for future work.

# 2 Literature Review

The current chapter presents the literature review of three related concepts: the digital instrument clusters, a trend in the last years in the automobile industry [2], the optical measurements, needed for fully characterize the properties of automotive displays, and the testing workbenches, which manoeuvre the light sensors and perform optical measurements on digital instrument clusters.

The first section discusses the increasing utilization of those automobile clusters. There it will be presented the history of them, the reasons why industry adopts this tendency and how they are produced.

The second section presents the optical measurements required before the market introduction of some automotive displays. It will also explain the two different types of light measurement devices as well as what some of them measure.

The last section deals with the testing workbenches, which are machines necessary for carrying out some of those optical measurements, and it will be presented some solutions already implemented in the market.

## 2.1 Digital Instrument Clusters

Since the introduction of digital clock displays decades ago, the increasing of electronic and graphical panels in all cars is a constant [3]. The digital instrument clusters, replaced the traditional and electromechanical instrument panels in the modern cars [2][4]. Also called electronic instrument clusters, digital dashes, panels or simply cluster (Figure 2.1), they are a set of instrumentation displayed with a digital readout, which can include since the basic options like the speedometer, temperature, and fuel consumption gauge, to the more sophisticated ones, like navigation, multimedia access, and different safety functions to the new class of driver assistance systems [5].

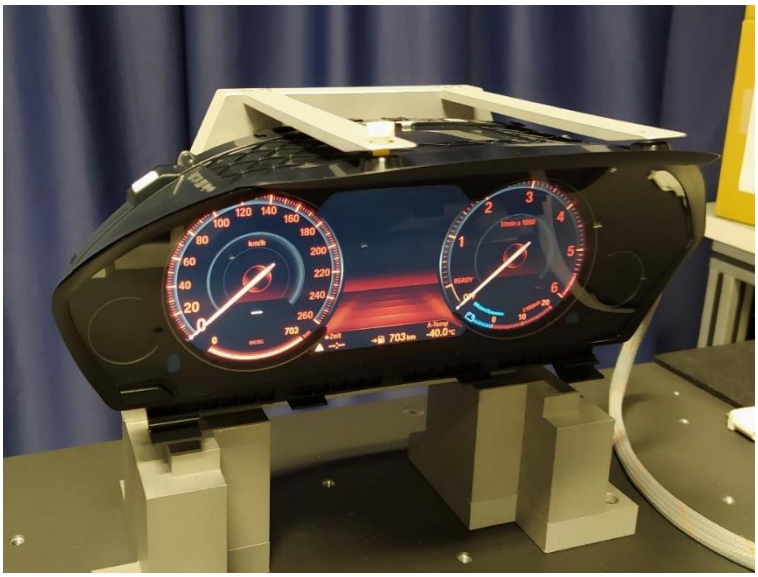


Figure 2.1 - BMW digital instrument cluster

### *2.1.1 History*

In the first cars, the automotive instrumentation was only a speedometer and some warning lamps. After, around the 1950s, it was implanted in the instrumentation of the car some single gauges like radiator temperature, tachometer indicator, and fuel gauge. At the beginning of the following decade, single instrumentation units were superseded by instrument clusters, a set of instrumentation units placed in one just housing (Figure 2.2: 1) [5]. The instrumentation in the cars stayed, with that approach, less costly, more reliable, and easier to manufacture and to install in the vehicle [6].

Two different ways of development appeared later, the digital instruments (Figure 2.2: 2) and the analogue instruments (Figure 2.2: 3). Clusters with digital instruments were mostly used in the USA and Japan, mainly in high-capacity vehicles, and have not been as successful in Europe. In contrast, analogue instruments remained the mainstream of information representation in the European instrument cluster industry.

Over time, the instrument clusters increased in size and were added additional areas (Figure 2.2: 4). Some graphic-capable displays appeared in those approaches and with them, new functions relevant to the driver and vehicle. Service intervals, check functions for the vehicle's operating condition or vehicle diagnosis, and some symbols, like arrows, to give information about route guidance was implemented. The appearance of the displays in the clusters, near the driver's first field of view, gave the advantage that information could be read more quickly without the driver having to take his eyes off the road for an extended period.

Consequently, the next step was the translation of the analogue instrument clusters, just with some little displays, to complete graphic displays (Figure 2.2: 5), where the mechanical gauges were entirely superseded by the graphical ones [5].

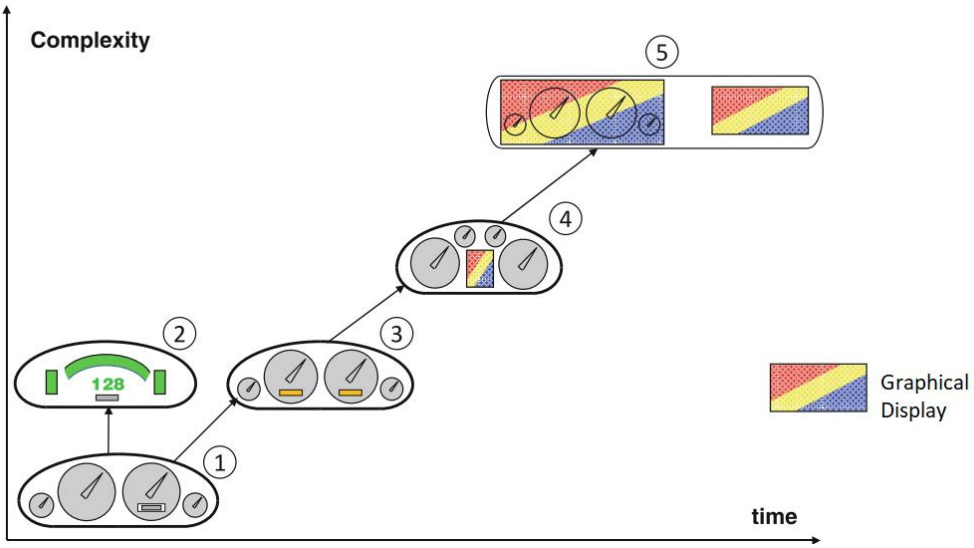



Figure 2.2 - Evolution of the digital instrument clusters [5]

### *2.1.2 Drawbacks and Advantages*

At first sight, manufacturers might have thought there were drawbacks in replacing traditional clusters with digital clusters. Large displays were necessary, and the requirement of high-cost hardware and software solutions were essential to support high-level application environment and appealing graphics [2].

Nevertheless, some reasons led manufacturers to upgrade to digital clusters. The first one was the reusability because the same cluster can now be inserted in different vehicle models, just by switching to different graphics. As the distinctive appearance of the instrument cluster is realized by software, it helps to lower the cost of instrumentation in the long term [5]. Scalability was another motive. The digital clusters have the possibility of adding new functionalities, just modifying the software and not changing nothing with the hardware, which gives it an excellent vantage [2]. The simplicity was an additional significant reason because the translation to digital instrument clusters makes it possible to present a wide variety of information to the driver that is adapted dynamically to suit the current driving situation (Figure 2.3). Displaying only the information that the driver currently requires, digital clusters can, therefore, ensure a simple screen, which makes the information very easy to read in all situations. It is possible, with that, to ensure less distraction and reduce the accident risk [2][7]. Another reason was the attractiveness because the digital instrument cluster can enhance the appeal of a vehicle with eye-catching graphics and features [2].

Together with the above factors, the development of faster and less expensive computers and the increasing demand for supplying the driver with more information that helps to drive safer were other reasons that accelerated the transition to digital instrument clusters [8]. They are also easier to install, configure, and maintain [5], and they offered now improved precision than the traditional speedometers.


Figure 2.3 - Dynamic adaptation of the digital instrument cluster [7]

Regarding the usefulness, the digital instrument cluster also has many benefits. About the design, the customization of the display is now available, where colours, wallpapers or gauge layouts can be adjusted, and different displays can appear to different modes of drive operation (performance, of-road, fuel-saving, … ). Additionally, the digital clusters can also inform drivers about the road conditions, showing visible speed limits, perceptible road ice indicators or surrounding vehicles. Other advanced features as navigation, backup camera, nearest gas stations, multimedia, can also be included.

However, there is still room for improvement: the achievement of high quality of automotive displays in terms of optical performance and quality like lifetime, the search of the right combination of design-cost-performance and the reduction of development time [2][3].

### *2.1.3 Production and Manufacturing*

After the specifications, the development and production of digital instrument clusters are usually based in four key points: graphical design, hardware, firmware and housing.

As digital instrument clusters are a set of virtual instruments, the graphic design is essential and typically where its production starts [2]. The representation should be unambiguous in all the cases, being easy to interpret the cluster with minimum effort. Correct placing of proper indicators, its sizing and colour-coding of the dashboard also have a lot of importance. As most of them are interactive, a functional language of interface is necessary. It should help the driver to find the objective quickly, like setting the clocks or resetting the trip counter. Digital instrument clusters can have more than one design, for example by switching between night or day mode, or depending on the driving operation mode (performance, of-road, fuel-saving...).

The firmware development is where the virtual dashboard programming took place. It is responsible for imagining what is visible in the digital cluster and for having a user interface with dialogues, controls and logic. Communication protocols should be implemented for interaction with other on-board vehicle systems. As digital instrument clusters are real-time devices, a real-time operating system (RTOS) is used.

The housing, as a container of electronics components, has some requirements for precision, rigidity and internal structure, which dictate the interior design. As well, there are requirements for external housing features, like dimensions and connector positions. After combining those requirements with aesthetic and protective function, the external design of the cluster is made, and the material is selected. At the end of this stage, a prototype is usually performed.

Before being introduced on the market, and since automotive displays require significantly more evaluation effort than standard displays [3], new products have to pass some rigorous optical verification tests.

## 2.2 Optical Measurements

Optical displays, even used in other different devices than digital instrument clusters, have to pass several measurement requirements in order to avoid eye discomfort, mental load and visual fatigue [4]. The optical measurements fully characterize the properties of the display and are very important to access performance for product development, manufacture and intended quality control [9].

Many optical measurements of the digital instrument clusters are necessary, being the most required luminance, colour, spatial uniformity, angular distribution, reflectance, and temporal characteristics. These provide necessary information regarding the quality, ease of use and readability of viewing under different conditions of use [9]. A more accurate inspection may also require measurement of black mura, where is measured the number of missing pixels [10].

The optical measurements of the displays started as handmade tests, where operators had the responsibility to compare the display under test to a reference. Nowadays, controlled optical instruments perform these optical inspections, improving the quality and the consistently deliver of displays in the long term. Based on optical science, these optical instrumented tests also reduce inspection time and increase test reliability [10][11].

Nevertheless, automotive displays have additional requirements in terms of optical performance and longevity in severe environments, needing significantly more evaluation effort and advanced procedures. They require an improved lifetime, because the utilization of cars without the visualization of speed is forbidden and the display replacement is expensive [3]. Depending on the display manufacturer and the client requirements, some examples of additional optical measurements that may be required are ghosting, flicker, gamma and weber fraction.

It is also essential to be aware that a considerable number of external factors can influence the measured parameters. Depending on the chosen standard, specific measurement precautions, different display setup conditions, such as ambient and darkroom conditions, as well as different ways of calibrating the optical instrument, should be taken in consideration

[12]. However, the manufacturer of the display often defines more specifications, dedicated equipment and test procedures [13].

Since most modern displays are integrated into systems, such as digital instrument clusters, by different companies from those that have produced the displays [12], it is essential to have a common understanding on minimum requirements and relevant optical parameters as well as the unification of the measurement and evaluation methods in the entire supply chain (from manufactures to integrators and car manufactures). With that purpose, some automotive-related companies within the German Flat Panel (DFF) had set a dedicated requirement's specification. The aim is to have a standardization, which takes to a reduction of effort along the automotive display supply chain [3].

This new approach, created additionally where basic procedures refer to the International Committee for Display Metrology (ICDM) and other display standards, enabled the standardization of automotive display performance values like luminance, brightness, gamma, contrast ratio, colour coordinates, response time, viewing angles and lifetime. New measurement specifications were created and some optimized optical tests were proposed [3]. The importance of standardized methods and evaluations are undeniable in order to guarantee comparability of measurement results within the complete supply chain [14].

Nevertheless, the standardization process of DFF is not finished, but an on-going task with some new challenges like free form, curved, and flexible displays [3]. As the displays are going through a significant evolution, it is predictable that this standardization process will also remain continually developing [10].

### *2.2.1 Light Measurement Devices*

In order to measure with quality the optical performances of the digital instrument cluster displays, light measurement devices (LMD) are indispensable. They are responsible for capturing and analysing image characteristics and delivering physical quantities that inform the optical performance of the displays [15].

Generally, LMD can be divided into two categories, the spot light measurement devices (spot-LMD) and the imaging light measurement devices (imaging LMD).

The spot light measurement devices measure the integrated values of optical quantities, like chromaticity and luminance, within a well-defined area. Some examples are the colourimeters and the spectroradiometers [14]. The light from that defined area element is guided, via optical lenses in most of the cases, to a detector element, and its electrical output is proportional to the parameter intensity of the incident light to be measured. The area of measurement, when it is used a positive lens, is defined by the light-sensitive area of the detector element, as showed in Figure 2.4. In this configuration, the aperture angle, $\alpha$, of the receiver (the combination of detector element and optical system) is defined by the lens aperture and the work distance. When directional variations are to be evaluated, this aperture angle, $\alpha$, should not exceed $5^{o}$ [16]. Another significant angle that should also be well distinguished is the measurement field angle, $\beta$, which is often specified in the datasheets of LMD's. It is usually fixed, therefore the measurement field can be adjusted by changing the working distance [17].

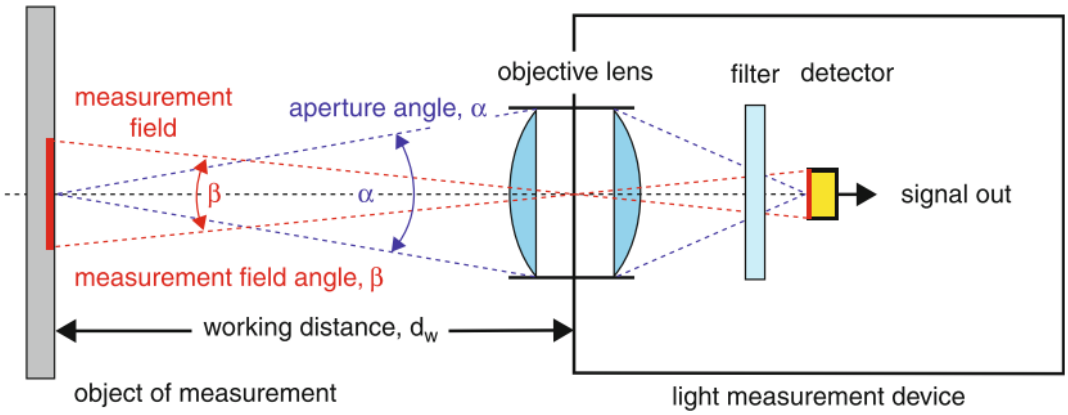



Figure 2.4 - Basic concept of a light measurement device [17].

On the other hand, the imaging light measurement devices determine the optical parameters with spatial resolution on an extended measurement field, like consumer photo cameras [14]. They are composed by a regular array of detector elements (Figure 2.5) that each one provides one output signal, which is proportional to the parameter "intensity" of the incident light coming from the array of elementary measurement fields on the digital instrument cluster under test. Like in consumer photo cameras, each detector element corresponds to one area element on the object of measurement [17].

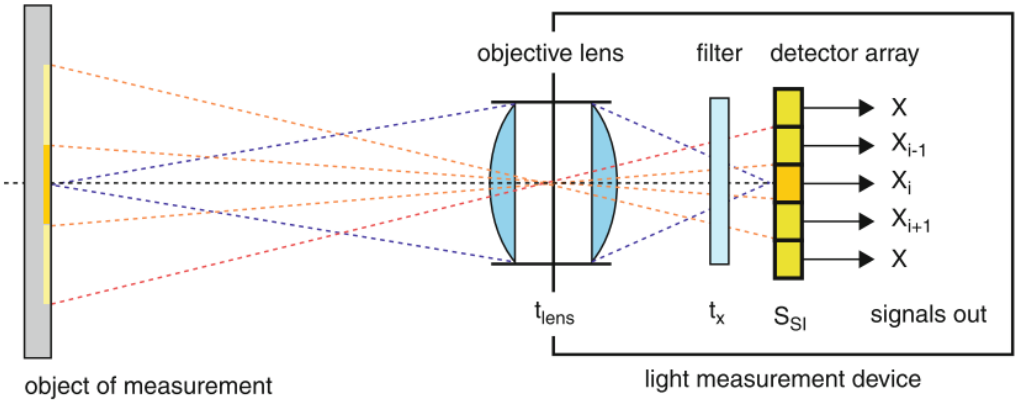



Figure 2.5 - Basic concept of an imaging LMD [17]

Figure 2.6 shows the differences in the captured light between the spot LMD and imaging LMD. At left (spot LMD) it is possible to see the subpixels of the device under test as projected on the photo detector as well as the grey region, which are not sensitive to light. In this type of LMD's, the output of the element detector is proportional to the sum of the illuminance over the field of measurement. In the other and, in imaging LMD's (at right) the subpixels of the device under test are projected onto the array of detector elements (in this case, the 5x6 white grid). As a result, the detector array provides 5 x 6 output signals, each of them proportional to the illuminance incident on the corresponding detector area [17].

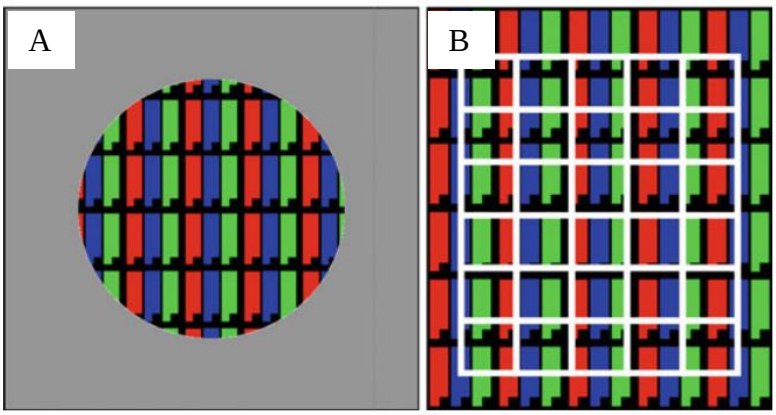



Figure 2.6 - Comparison of spot LMD (A) and imaging LMD (B) in terms of captured light [17]

As there is a wide range of optical parameters to be measured in digital instrument clusters, such as luminance, colour, angle of view and response time, there is also a large variety of measurement procedures and devices [18].

By first deciding on the appropriate quantity to be measured and the required measurement geometry, the next step is to determine whether measurements should be made spectrally or as photometric value and which LMD is the most appropriate to use. One example is the "brightness", which is usually specified in terms of its average luminance. In this case, a luminance meter that can average over a large area of the display would usually make the measurements. However, if it is necessary to measure the spectral characteristics of optical radiation falling on the screen from other light sources in the environment, it would be more appropriate to use a spectroradiometer [19].

Additionally, for any measurement, it is essential not only to select the most appropriate measuring equipment but also to ensure that it is correctly calibrated as well as to consider all potential sources of measurement error or uncertainty [19].

Depending on which test or which optical parameter is necessary to measure, different LMD's are used. Broadband detectors, such as photometers, luminance meters, and colourimeters, perform most of the measurements.

The photometers can measure most of the photometric measurements. The device under test is exposed to light, and the photometer measures its intensity converting light into an electric current by a photosensitive element [19].

Another light measurement instrument is the luminance meters which usually incorporate an imaging system to focus the area being measured onto the detector. They are generally designed to allow the area being measured to be viewed and identified through an eyepiece and some of them also have an additional "close-up" lens that can be affixed to enable minimal areas to be measured [19]. They are single element detectors and measure general basis photometric parameters like the amount of emitted or reflected light from a surface. Typically, they are lightweight, compact and portable and can also be connected to a PC which can store the measurement data and control the device.

Measurements of the optical proprieties of displays may also be realized using spectroradiometers if spectral values are required. They provide measures of spectral radiance or irradiance, from which photometric and colourimetric quantities can be measured. They are

light measurement instruments used to measure properties of light as a function of its portion of the electromagnetic spectrum, typically its wavelength, frequency, or energy. The most commons spectroradiometers are the scanning monochromator spectroradiometers [19].

Additional optical instruments to perform colour measurements are the array spectrometers and the colourimeters. Instead of being constituted by a grating that moves along of the measurement like in scanning monochromators (Figure 2.7: A), in array spectrometers, the incoming light is split by a fixed grating and captured by an optical line detector (Figure 2.7: B). This characteristic makes the array spectrometer a faster but not so accurate measurement. On the other hand, the advantages of colourimeters are that they are inexpensive instruments and perform a fast operation. They measure light input with three sensors with matched filters (Figure 2.7: C) so that the resulting spectral sensitivity corresponds for each one to a colour-matching function. The outputs are amplified, the tristimulus values are achieved and with the help of computing power is performed the transformation to colour spaces [18].

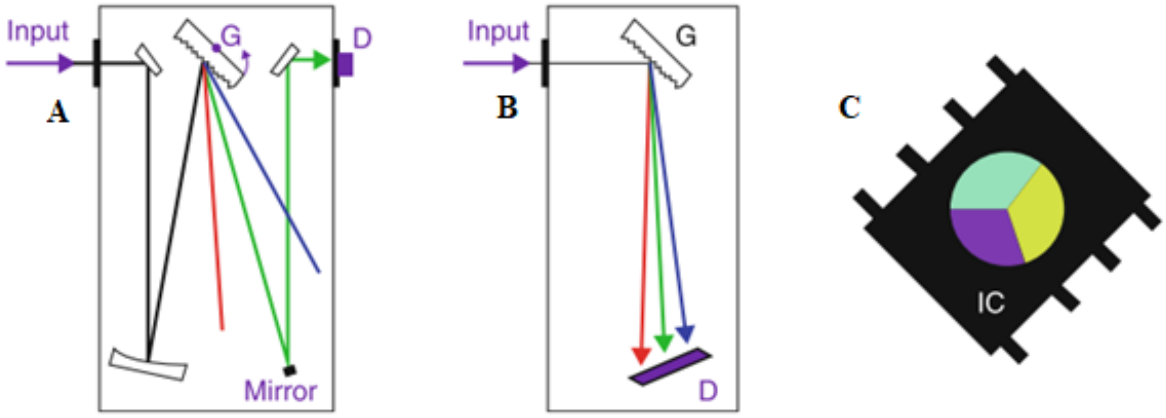



Figure 2.7 - Basic principle of scanning monochromators (A), array spectrometers (B) and colorimeters (C) [18]

## 2.3 Optical Testing Workbenches

In order to make the optical verifications and respect the specifications of the tests, light measurement devices have some requirements in terms of position and orientation angle to the cluster display that is pretended to measure. Between different optical tests, it can be necessary to modify the position of the light measurement device in three coordinate axes and three orientation angles.

It is with that purpose that appeared the optical testing workbenches, also called measurement workbenches. They act as holders for light measurement devices and allow them to move in all the axes. They give better repeatability and improved accuracy of the final position of the light measurement device they support. These testing workbenches can also be automated and synchronized with the measurement instrument, reducing the time between the different optical tests, which improves productivity.

From the beginning of the 20th century, some modest testing workbenches had already been published. Based on some photometer workbenches, Sutton [20] proposed some developments and designed an improved photometer workbench, which incorporates all the needed features. This machine created to measure lamps was rigid, worked through a movement

of the lamp about the photometer and had a graduated ruler and a pointer along the trolley (Figure 2.8). Hence, it was possible to visualize the distance between the photometer and the lamp.

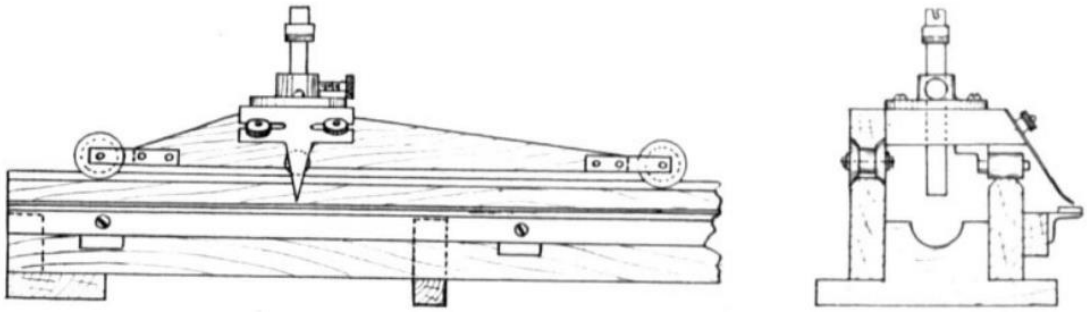


Figure 2.8 - Lamp trolley of the photometer workbench [20]

More recently and with the technological evolution, the testing workbenches are associated with powerful robots that support light measurement devices. They ensure excellent manoeuvrability to the optical instruments, they give to the optical verifications another level of quickness and productivity, and they reduce position errors thanks to their ability to be programmed.

One example is the robotic display measurement system (Figure 2.9) from Gamma Scientifics, which combines a 6-axis robot and high-performance spectroradiometers for fast and accurate display measurements [15]. Keeping the display stationary during the measuring process and making available a broad set of measurement capabilities, the measurement system ensures an improved accuracy of the light measurement device position, good angular repeatability, about ± 0.01 degrees, and an attractive angle resolution, 0.002 degrees [21].

Figure 2.9 - Robotic display measurement system from Gamma Scientifics [22]

Another good example is the robot-testing workbench of the Optical lab in the Bosch Car Multimedia, S.A. (Figure 2.10). Also with 6-axis freedom and with the help of an industrial robot Staubli TX60, this workbench can support the array spectrometer CAS 140CT, the fast broadband spectroradiometer JETI Specbos 1211, the luminance measuring camera LMK 5, the imaging colourimeter LumiCam 1300 as well as the 5-megapixel measurement camera Prosilica GT 2450. Thanks to this, the robotic workbench can perform a wide range of optical

measurements, only a few examples being the measurements of luminance, radiance, illuminance, colour distribution, black mura, uniformity, ghosting and flicker. It has the manoeuvrability to hit a wide range of positions and required orientation angles. Being inside a cabin and surrounded by protective barriers, the robot-testing workbench is able to realize light measurements verifications with high speed, rigidity and about ±0.02 mm of repeatability [23].

Figure 2.10 - Robot-testing workbench of the Optical lab

## 2.4 Bosch Manual Workbench

This section describes the Optical Lab manual workbench, which will be the object of automation in this dissertation.

Like the workbenches talked in the previous section, this manual workbench supports a light measuring device (Figure 2.11), and it is essential to increase the checking capacity of the Optical Lab. Having the possibility to support different measurement instruments, like the array spectrometer CAS 140CT, the fast broadband spectroradiometer JETI Specbos 1211, the luminance measuring camera LMK 5 and the imaging colourimeter LumiCam 1300, this workbench allows a significant number of light measurements: brightness, ISO contrast, luminance, luminance distribution, radiance, illuminance, illuminance distribution, irradiance, colour, colour distribution and luminous intensity distribution are just some of the light measurements available.

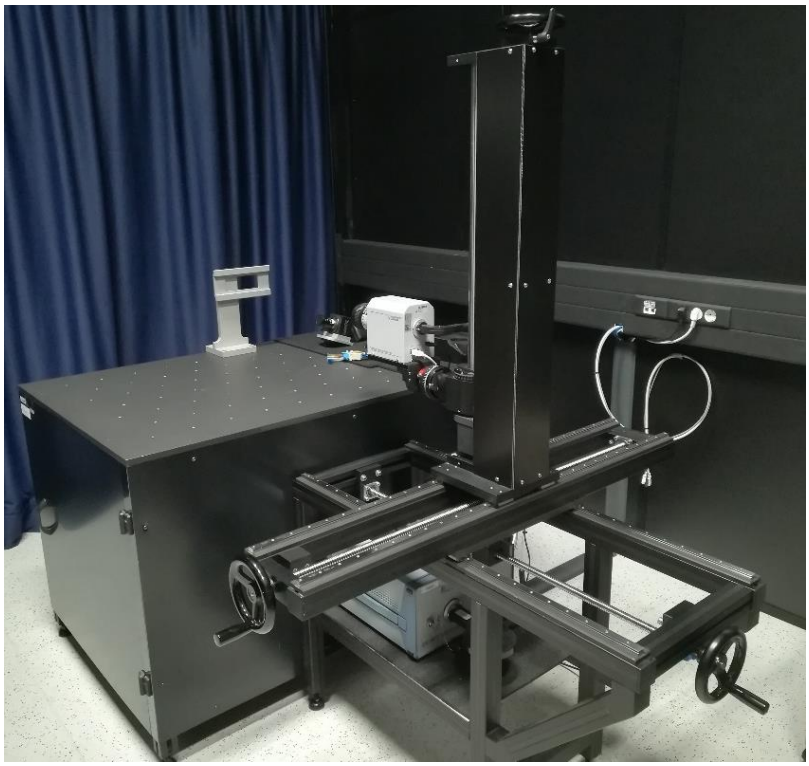


Figure 2.11 - Manual testing workbench with a LumiCan 1300

The workbench is composed of three trolleys, which are commanded by each respective hand crank, allowing the optical instrument motion in three axes of spatial position. As the vertical trolley, which moves the light measuring device in the vertical axe, is affected by the gravity forces, it has a hand-operated spring break, holding the position (Figure 2.12).

Figure 2.12 - Hand operated spring break with vertical hand crank

In order to ensure more flexible tests and angular position requirements, the support allows the rotation in two axes, giving a total of five degrees of freedom to the instrument. As with linear motion, angular motion is ensured manually by the workbench engineer operator, rotating the pretended axe. A graduated angular scale helps the operator and ensure some accuracy to the angular movement (Figure 2.13).

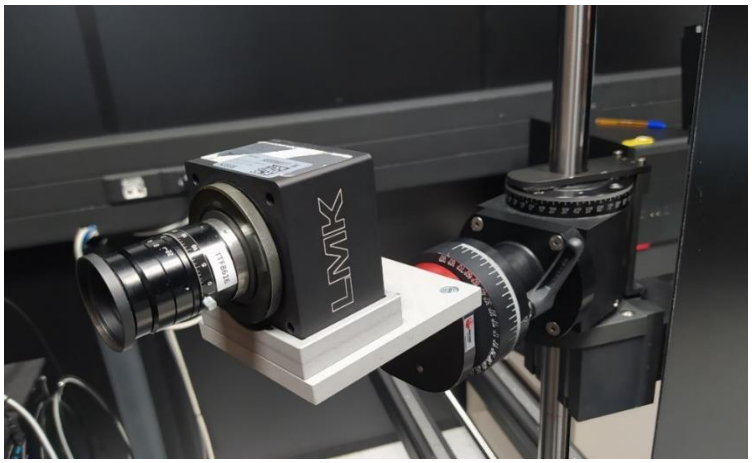


Figure 2.13 - Graduated angular scales with angular axes

Despite increasing the volume of tests carried out by the optical lab, this workbench has some disadvantages compared to the robotic workbench also presented in the lab and mentioned in the previous chapter. As there is no graduated ruler along with the linear trolleys, the reliability is weak, and repeatability and accuracy of the movement depend on the operator. The velocity of moving from one position to another is also low. As the ball screw pitch is only five millimetres, a movement between two points can make necessary to turn the crank by hand a high number of times. It adversely affects the velocity of the trolleys and the physical fatigue of the technician.

This project aims to automate this workbench in order to decrease the positioning time and increase the consistency of the process and its reliability. It will include the design, development and construction of a motion transmission system, the implementation of a group of motors and the development of an interface in LabVIEW. The interface should command the automatic movement of all the three linear axes, which are the most used. Although the specific project goals have already been described in Chapter 1, Table 1 define them in a resumed way.

Table 1 - Specifications of the project

| | |
|---|---|
| Main Purpose | Automate the workbench |
| Axes to be commanded | 3 |
| Maximum time travel | 30 seconds |
| Maximum weight to support | 8 kg |
| Repeatability | ≤ 0.5 mm |
| Accuracy | ≤ 0.5 mm |
| Resolution | ≤ 0.1 mm |
| Noise limitation | Keep the operation noise lower as to not disturb nearby labs |

## 2.5 Concluding Remarks

Throughout this chapter, it was essentially studied the digital instrument clusters present in modern cars, the optical tests applied to them and the optical measurement workbenches where they occur.

Besides demonstrating the technological evolution of the instrument panels over the last century, the advantages of inserting digital clusters in modern cars were explained. Better reusability, scalability, simplicity and attractiveness are just some of the advantages that led to the replacement.

The main optical tests usually carried out were mentioned, as well as the complexity of standardising the measurements carried out along the supply chain. The differences between spot and imaging LMD's were mentioned, and the operation of some measuring instruments was explained.

The optical measurement workbenches on the market were analysed. It was noticed that some of the most advanced ones are performed by robotic arms, with an excellent level of positioning reliability. Finally, the Bosch manual workbench was described.

# 3 Proposed Solution

Since the workbench and its requirements are defined, this chapter explains the entire proposed solution. To this end, the entire linear movement of the workbench is described, from the choice of actuators that will move the trolleys to the definition of the position sensors that will continuously read system information.

Given this, all arguments for the hardware choice are explained as well as the most significant calculations, whether by analytical calculations, spreadsheets, or specific simulation programs.

In the first part, the chapter describes how the linear workbench motion command will be performed, which speed profiles will be used, and which motors and controllers will be chosen. In a final part of the chapter, the hardware chosen to limit this linear movement will be described, from limit sensors to emergency buttons that will quickly and safely disable the system.

## 3.1 Linear Motion

The first aspect covered was the movement of the linear trolleys of the manual workbench. They give to the workbench the possibility of motion in three axes spatial space, and they use to be guided by three manual hand cranks.

Taking into account the application requirements, and to positioning the workbench trolleys with good speed and satisfying reliability, one possible solution considered was the use of stepper motors in an open ring control mode. The stepper motors are devices that can convert electrical impulses in a mechanical motion through fixed increments. They can achieve high precision in open-loop control. Due to its digital nature, they are easy to connect with the computer and be commanded by LabVIEW, and, if well-sized, they do not accumulate errors and thus become excellence open loop positioning devices. They have comparatively high torques at low velocities, which often avoids the need to introduce gearbox-type transmission elements.

Other possibility could be the use of stepper motors in a closed-loop or the use of servomotors. The first case is a relatively low-cost solution, where the implementation of one position sensor as an absolute encoder gives a feedback position to the system. Alternatively, the use of servomotors offers even more precision to the system. They operate with high efficiency, and its output can achieve a particular angle, position, and velocity that a conventional motor does not reach.

Although servomotors and closed-loop stepper motors would offer a more precise positioning, the solution chosen was an open-loop stepper motor. They are the lower-cost solution, and also its accuracy is already sufficient for the purpose of the workbench: the movement of a measuring device. Besides, the company had already one free stepper motor, 103H5208-5240 of Sanyo Denki that gave the possibility of experimentation before order the others stepper motors.

However, the use of stepper motors also has certain drawbacks. Sometimes some drivers can make them prone to vibration and over-extension. In specific applications, these resonance effects can lead the motor to skip steps and went out of sync at specific speeds. With the use of micro-step drivers, it is possible to overcome many of the problems associated with stepper motors, such as coarse performance at low velocity and make the motor run a lot more smoothly without vibrations and over-extensions.

When used in an open loop in positioning systems, the stepper motor may fail or lose position if the load torque exceeds the available torque. Therefore, as a principle of sizing, it is appropriate that at maximum speed, the stepper motor has at least 50% additional torque as a safety margin. As important as selecting the proper motor based on the torque/speed curve is also important to maintain an appropriate relationship between the inertia of the load and the motor inertia lower than 10 times. Inadequate inertial load/motor ratios can lead to too long accommodation times or even affect the stability of the system.

### *3.1.1 Velocity Profile*

In order to realize the entire transmission system sizing, it is essential to define the velocity profile of the linear axis so that it is possible to account the efforts arising from the acceleration of the trolley and the load it carries.

In a linear trolley, an important criterion is the travel time: moving the load between the two positions in the shortest time possible. To avoid too high acceleration, a trapezoidal velocity profile was selected.

By stipulating a maximum cycle time of 30 seconds between the workbench extremities, a simple distribution of cycle time over the three sectors A, B and C as shown in Figure 3.1, and knowing the maximum distance travel of the trolleys, it is possible to calculate the maximum speeds and accelerations necessaries.

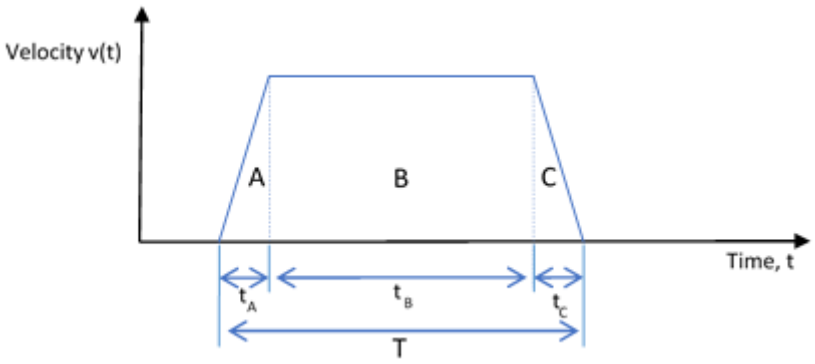



Figure 3.1 - Velocity profile sectors (A - 10%T; B - 80%T; C - 10%T)

Since the total distance travelled by the trolley ($d_T$) is equal to the sum of the areas of the three regions A, B and C, the expressions for calculating maximum velocities ($v_{max}$) and maximum accelerations ($\alpha_{max}$) are as follows:

$$d_T = \left(\frac{1}{2} \cdot t_A \cdot v_{max}\right) + \left(t_B \cdot v_{max}\right) + \left(\frac{1}{2} \cdot t_C \cdot v_{max}\right) \quad 3.1$$

Where:

$d_T$= total distance (m)

$t_A$= acceleration time (s)

$t_B$= constant speed time (s)

$t_C$= deceleration time (s)

$v_{max}$= maximum velocity (m/s)

Developing Equation 3.1, it obtains the following:

$$v_{max} = \frac{d_T}{\frac{1}{2} t_A + t_B + \frac{1}{2} t_C} \quad 3.2$$

$$a_{max} = \frac{v_{max}}{t_A} \quad 3.3$$

By knowing the total distance of each of the three trolleys, the maximum velocities and accelerations are expressed in Table 2.

Table 2 - Required trolleys maximum velocities and accelerations

| | $d_T$ (m) | $t_A$ (s) | $t_B$ (s) | $t_C$ (s) | $v_{max}$ (m/s) | $\alpha_{max}$ (m/s$^2$) |
|---|---|---|---|---|---|---|
| X-axis | 0.75 | 3 | 24 | 3 | 0.02778 | 0.00926 |
| Y-axis | 0.80 | 3 | 24 | 3 | 0.02962 | 0.00988 |
| Z-axis | 0.60 | 3 | 24 | 3 | 0.02222 | 0.00741 |

In order to obtain the velocity, it is necessary to have in account the transmission of the system. Each trolley movement has a ball screw, which translates rotational motion, coming from the motor or the hand crank, into linear motion. With this mechanical linear actuator, high thrust loads can be withstood with minimal internal friction. The three ball screws installed in the workbench, which one coupled with the three spatial axes, has 5 mm of lead. Assuming motors of 200 steps per revolution, and 1:1 transmission between the shaft of the ball screw and the engine, the maximum velocities are in Table 3, expressed in revolutions per minute (rpm), radians per second (rad/s) and steps per second (steps/s). The maximum accelerations are also registered.

Table 3 - Required motors maximum velocities and accelerations

| | n (rpm) | ω (rad/s) | ω steps/s) | α (rad/s$^2$) | α (steps/s$^2$) |
|---|---|---|---|---|---|
| X-axis | 333.3 | 34.9 | 1111 | 11.64 | 371 |
| Y-axis | 355.6 | 37.2 | 1185 | 12.41 | 395 |
| Z-axis | 266.7 | 27.9 | 889 | 9.31 | 296 |

### *3.1.2 Transmission*

Now that the velocity profile is known, the next step is to define the transmission between the motors and the ball screw shafts, and consequently, the maximum torque necessary to move the axis. Once the stepper motors have high torques at low velocities, it is not needed the utilization of reduction-type transmission elements, and it is acceptable to start assuming a 1:1 ratio between the motor and the ball screw shaft.

In order not to increase the dimensions of the workbench, the use of the motor directly coupled to the screw shaft was avoided. Instead, on the horizontal axes, it was decided to place the motor under the ball screw and perform the rotation transmission through two gears, as shown in Figure 3.2: A. It allowed the optical workbench not to increase its size considerably, and also allowed good circulation within the optical laboratory. As there was no such problem with the vertical shaft, it was decided to place the motor directly coupled to the ball screw shaft (Figure 3.2: B). For that, it was used a flexible shaft coupling, which corrects eventual misalignments, and it allows the reduction of vibration and noise.

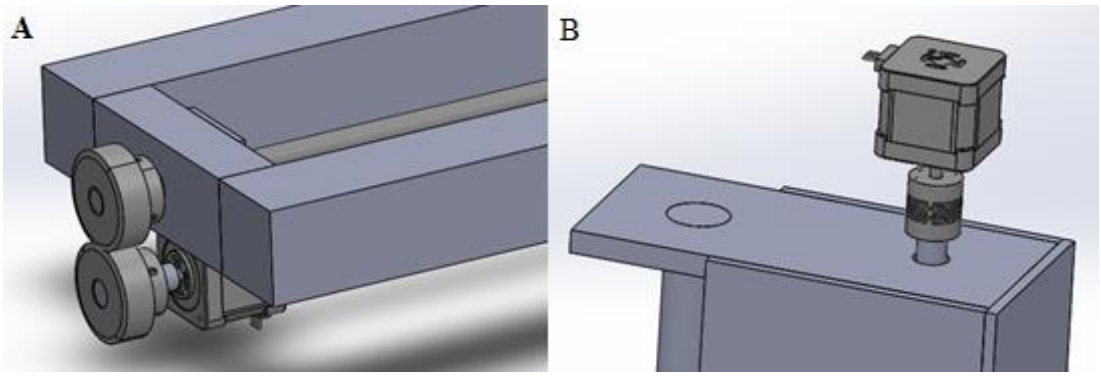



Figure 3.2 - Torque transmission in the horizontal axis (A) and in the vertical axis (B)

An essential factor for the gear' selection was the noise that this mechanism could make and, because of this, nylon and helical gears were preferred. Even if the nylon gears are not as durable as those made of metal, they are not only corrosion-free but also lighter, less expensive, and not prone to noise when under vibration. Helical gears also have advantage, due to the gradual way in which their teeth connect, and therefore more resistant to wear. However, they promote an axial force on the axis that must be taken into consideration. The gears size was chosen in a way that the motor could be placed under and close to the shaft. The one which meets these requirements was the NEGTM1.5-20-12 [24].

### *3.1.3 Stepper Motors Selection*

The maximum torque is necessary during the acceleration phase of the inertial load with all its adjacent weight, to accelerate the gears and to accelerate the rotor.

$$T_{\max required} = T_{b.s.\ acc} + T_{g.\ acc} + T_{m.\ acc} \tag{3.4}$$

$$= T_{b.s.\ acc} + J_g \cdot \alpha_{motor} + J_{rotor} \cdot \alpha_{motor}$$

Where:

$T_{\max required}$ is the maximum torque required (N·m)

$T_{b.s.\ acc}$ is the torque required for the ball screw acceleration (N·m)

$T_{g.\ acc}$ is the torque required for the gears acceleration (N·m)

$T_{m.\ acc}$ is the torque required for the motor acceleration (N·m)

$J_g$ is the gears inertia (kg·m$^2$)

$\alpha_{motor}$ is the angular acceleration of the motor (rad/s$^2$)

$J_{rotor}$ is the rotor inertia (kg·m$^2$)

As there are no gears in the vertical axis, $T_{g.\ acc}$ will be equal to zero in the choice of that motor.

In this first part of the motor selection, the inertia of the rotor ($J_{rotor}$) is unknown since the stepper motor has not yet been chosen. Consequently, this value will be ignored in the first phase. It will be verified afterwards.

Starting by the total torque necessary to accelerate the ball screw ($T_{b.s.\ acc}$), it is the junction of the amount of torque at a constant speed ($T_c$) with the torque due to acceleration ($T_{acc}$). In turn, $T_c$ is calculated by the sum of three components: the torque to drive the load ($T_d$), the torque due to preload ($T_p$), and the torque due to the friction of support bearings and seals ($T_f$). On the other hand, the torque due to acceleration ($T_{acc}$) is the multiplication result of the system inertia (J) and the angular acceleration of the ball screw (α).

$$T_{b.s.\ acc} = T_c + T_{acc} \tag{3.5}$$

$$T_{b.s.\ acc} = T_d + T_p + T_f + (J_s + J_l) \cdot \alpha$$

Where:

$T_c$ is the torque at constant speed (N·m)

$T_{acc}$ is the torque due to acceleration (N·m)

$T_d$ is the torque to drive the load (N·m)

$T_p$ is the torque due to preload (N·m)

$T_f$ is the torque due to the friction of support bearings and seals (N·m)

$J_s$ is the screw shaft inertia (kg·m$^2$)

$J_l$ is the load inertia (kg·m$^2$)

$\alpha$ is the angular acceleration (rad/s$^2$)

Consequently, the torque to drive the load ($T_d$) is calculated by the following expression:

$$T_d = \frac{F_a \cdot P}{2 \cdot \pi \cdot \eta} \quad 3.6$$

Where:

$F_a$ is the total axial force (N)

$P$ is the lead (m)

$\eta$ is the efficiency of the ball screw

In the case of the ball screws which move the horizontal workbench trolleys, the maximum value of the total axial force ($F_{a\text{-horizontal b.s.}}$) is obtained in the moments of acceleration of the ball screws. At these moments, $F_{a\text{-horizontal b.s.}}$ is composed by the friction forces due to the linear load movement ($F_f$) and the necessary forces to its linear acceleration ($F_{acc}$). Also, it has been defined that the ball screw must eventually carry an extra horizontal load ($F_{extra}$) of up to 20 N.

$$F_{a\,-horizontal\ b.s.} = F_f + F_{acc} + F_{extra} \quad 3.7$$

Where:

$F_{a\,-horizontal\ b.s.}$ is the total axial force of the horizontal ball screws (N)

$F_f$ is the friction forces due to the linear movement of the load (N)

$F_{acc}$ is the necessary forces to the linear acceleration of the load (N)

$F_{extra}$ is the force due to the extra load (N)

It was estimated a friction coefficient of 0.003 in the bearings. The total weight of the system is 20 kg, 6 kg of each x- and y-axis trolley, and 8 kg for the light measuring devices and its support (z-axis trolley).

In the case of the vertical ball screw (z-axis), another force that should be included in the equation 3.7 is the one relative to the weight of the light measurement device and its support ($F_w$). In this case, the moment of maximum axial force is in the acceleration of an upward movement. The friction force is insignificant in this situation.

$$F_{a-vertical\ movement} = F_{acc} + F_{extra} + F_W \quad 3.8$$

Where:

$F_{a-\ vertical\ movement}$ is the total axial force in the vertical movement (N)

$F_W$ is the force relative to the weight of the measurement device and its support (N)

In Table 4 is demonstrated the total axial force ($F_a$) for each of the ball screws. It was taking into consideration the angular accelerations previous calculated in table 2.

Table 4 - Calculation of the total axial forces of the ball screws ($F_a$)

| | Calculation | $F_a$ (N) |
|---|---|---|
| X axis | $F_a = 20 \cdot 9.81 \cdot 0.003 + 20 \cdot 0.00926 + 20$ | 20.77 |
| Y-axis | $F_a = 14 \cdot 9.81 \cdot 0.003 + 14 \cdot 0.00988 + 20$ | 20.55 |
| Z-axis | $F_a = 8 \cdot 0.007407 + 20 + 8 \cdot 9.81$ | 98.56 |

Since it was not possible to have access to the reference of the ball screw, the value of the efficiency (η) from the manufacturer was not obtained. However, based on the relatively low average load on the ball screw and the moderately low-speed rates that the system requires, which in the worst case is 356 rpm, a conservative value of 0.7 was given to the efficiency of the ball screw.

Knowing that the lead of the screw is 5 mm, having in account the previous calculations and following Equation 3.6, the torque to drive the load ($T_d$) is calculated in Table 5.

Table 5 - Torque to drive the load ($T_d$)

| | $T_d$ (N·m) |
|---|---|
| X-axis | 0.024 |
| Y-axis | 0.023 |
| Z-axis | 0.112 |

The torques due to preload ($T_d$) and due to friction of support bearings and seals ($T_f$) are values usually given by the manufactures. However, as it was not possible to obtain the reference of the ball screw, $T_d$ and $T_f$ were estimated at 0.04 Nm, a typical number having in consideration similar ball screws. The maximum torque at a constant speed ($T_c$) is calculated by adding all these torques, and it is expressed in Table 6.

Table 6 - Torque at constant speed ($T_c$)

| | $T_c$ (N·m) |
|---|---|
| X-axis | 0.064 |
| Y-axis | 0.063 |
| Z-axis | 0.152 |

In order to know the total torque required for the ball screw acceleration ($T_{ball\ screw\ acceleration}$), the screw shaft inertia ($J_s$) and the load inertia ($J_l$) must also be calculated.

Once again that it was not possible to know the exact reference of the ball screw, the screw shaft inertia ($J_s$) was calculated based on the cylinder inertia [25]. In terms of a more approximated calculation, it was used an intermedium radius of the screw shaft.

$$J_s = \frac{1}{2} \cdot m_s \cdot r_m^2 \qquad 3.8$$

$$= \frac{\rho \cdot \pi \cdot r_m^4 \cdot l}{2}$$

Where:

$J_s$ is the screw shaft inertia (kg·m$^2$)

$m_s$ is the mass of the screw shaft (kg)

$r_m$is the intermedium radius of the screw shaft (m)

$\rho$ is the density of the steel (kg/m$^3$) = 7860 kg/m$^3$ [26]

$l$ is the length of the screw (m)

On the other hand, the load inertia ($J_l$) is calculated based on equation 3.9.

$$J_l = m \cdot \left(\frac{P}{2 \cdot \pi}\right)^2 \quad [27] \qquad 3.9$$

Where:

$J_l$ is the load inertia (kg·m$^2$)

$m$ is the mass of the transported load (kg)

$P$ is the lead of the screw shaft (m)

Determined the different screw shaft lengths of the axes, and knowing that the intermedium radius is 7 mm and the lead is 5 mm, the screw shaft inertia ($J_s$) and the load inertia ($J_l$) is

expressed in Table 7. Following equation 3.5, and consecutively the angular accelerations of Table 3, the total torque required for the ball screw acceleration ($T_{b.s.\ acc}$) is also calculated. It is visible that the torque due to the acceleration of the ball screw ($T_{acc}$) does not have much influence in $T_{b.s.\ acc}$.

Table 7 - System inertia (J) and total torque required for the ball screw acceleration ($T_{b.s.\ acc}$)

| | l (m) | $J_s$ (kg·m$^2$) | $J_l$ (kg·m$^2$) | $J_s + J_l$ (kg·m$^2$) | $T_{acc}$ (N·m) | $T_{b.s.\ acc}$ (N·m) |
|---|---|---|---|---|---|---|
| X-axis | 1 | 2.964E-05 | 1.267E-05 | 4.231E-05 | 4.923E-04 | 0.064 |
| Y-axis | 1 | 2.964E-05 | 8.866E-06 | 4.793E-05 | 4.779E-04 | 0.064 |
| Z-axis | 0.8 | 2.372E-05 | 5.066E-06 | 3.632E-05 | 2.679E-04 | 0.152 |

Despite having already $T_{b.s.\ acc}$, it must be remembered that this was obtained on some assumptions. The efficiency of the ball screw, as well as the preload torque and the torque due to friction of support bearings and seals, were estimated and may not truly represent the actual values. For this reason, a measurement of the torque required to move the ball screw was carried out using the analogue torque meters of the metrology laboratory. The results obtained are in Table 8.

Table 8 - Measured torque required for the ball screw acceleration ($T_{b.s.acc\ measured}$)

| | $T_{b.s.acc\ measured}$ (N·m) |
|---|---|
| X-axis | 0.06 |
| Y-axis | 0.06 |
| Z-axis | 0.12 |

Since the measurement of the torque was realized by an analogue torque meter, moved by hand, the values of Table 8 also not genuinely represent the torque necessary to move the ball screws at the maximum velocities (Table 3). Nevertheless, this demonstrates that the values calculated in Table 7 are acceptable values of the torque required to move the ball screw at the desired maximum speeds. In a conservative way, it was used the torques of Table 7 for the motor selection.

The next step was the calculation of the second parameter of equation . Since the gear inertia ($J_g$) is insignificant in most of the cases, the supplier did not provide it. However, it was possible to have a quick approximation of it form the CAD file. Using the student version of the Solidworks program and as the material of the gears it was known, the inertia was calculated, and the value obtained was 6.50E-06 kg·m$^2$.

Assuming a transmission relation of 1:1 and knowing the angular acceleration of the motor (Table 3), the torque necessary to the gears acceleration ($T_{g.\ acc}$) is expressed in Table 9. Compared with the torque necessary to accelerate the Ball Screw, $T_{g.\ acc}$ is negligible.

Table 9 - Torque required to the gears acceleration ($T_{g.\ acc}$)

| | $T_{g\ acc}$ (N·m) |
|---|---|
| X-axis | 1.30E-04 |
| Y-axis | 1.61E-04 |

The motor torque, besides exceed the torque required for the ball screw and gears acceleration, it must also ensure the acceleration of its rotor. Additionally, 50% more torque must be provided as a safety margin since this system will be implemented in an open loop.

On the other hand, to achieve high performance, the system inertia (J), which is the sum of the screw shaft inertia, the load inertia and the gears inertia, should not exceed ten times the rotor inertia. Inappropriate inertial load/rotor combinations can lead to too long accommodation times or even affect the stability of the system. Thus, taking into account the inertias calculated in Table 7 and adding the gear inertias, the motor must have inertia higher than 5.53E-06 kg·m$^2$.

By consulting the Sanyo Denki catalogue of 42 mm bipolar stepper motors (Appendix A), it is possible to see that unless the first three motors, all motors already exceed the minimum inertia required. As the Optical lab had already one of these motors, the 103H5208-5240, joining with a stepper motor driver BSD 02.V from R.T.A., it was verified if this motor coupled with this driver could be used in one of the axes. The motor has an inertia of 0.056E-04 kg·m$^2$, holding torque of 0.39 N·m and its characteristic torque/velocity diagram is shown in Figure 3.3.

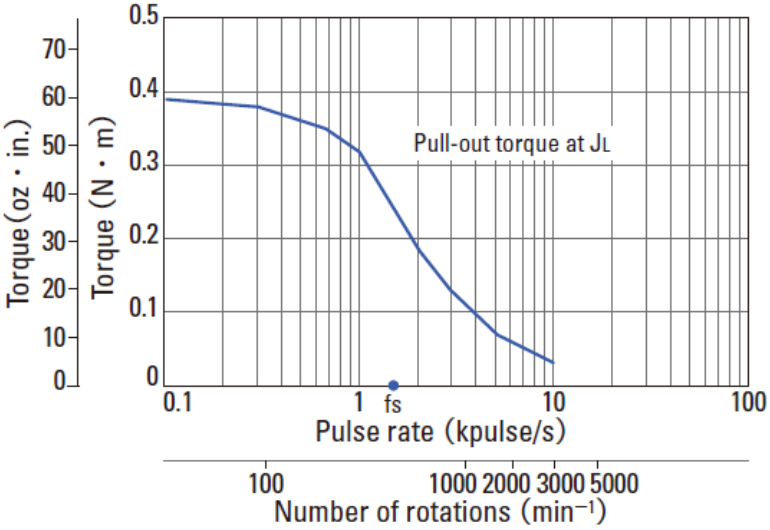



Figure 3.3 - Characteristic torque/velocity diagram of 103H5208-5240 [28]

Taking into account the inertia of this motor and multiplying it with the angular acceleration of each respective axis, according to equation , the maximum torque required by the motor ($T_{max\ required}$) is obtained. Adding more 50% as a safety margin, it results in the values of Table 10.

Table 10 - Maximum torque required ($T_{max\ required}$) and maximum torque required with 50% of safety margin ($T_{max\ required+50\%}$)

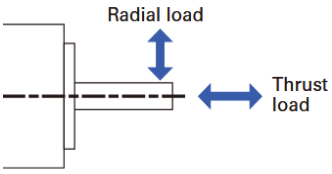
| | $T_{max\ required}$ (N·m) | $T_{max\ required+50\%}$ (N·m) |
|---|---|---|
| X-axis | 0.064 | 0.096 |
| Y-axis | 0.064 | 0.096 |
| Z-axis | 0.152 | 0.229 |

To this motor be acceptable for insertion into the workbench axis, it must withstand at least the $T_{max\ required+50\%}$ at the maximum speeds, calculated previously in Table 3. Thus, the motor should support a torque of 0.096 N·m at a rate of 1185 steps/s, and torque of 0.229 at a speed of 889 steps/s. It is visible in Figure 3.3 that all these two points mentioned are under the pull-out torque curve, which means that the 103H5208-5240 motor of Sanyo Denki can be used on any of the three axes. It also shows that it is possible to, eventually, rise the velocities of the trolleys, reducing the previously stipulated 30 seconds cycle time between the extremities.

In order to choose the two other motors, it was verified whether the other motors with the same inertia value, 103H208-5040 and 103H5208-5140, could withstand the speed/torque combinations mentioned above. As they could not, the two motors chosen were also the 103H5208-5240, because it was the one with lower inertia compatible that could support the trolley's movement.

#### *3.1.3.1 Forces on the motor shaft*

In the vertical axis, the forces on the motor shaft are minimum, since there is a flexible shaft coupling which minimizes the misalignments. However, in the horizontal axes is necessary to have in consideration the forces originated from the contact between the helical gear teeth, which can be divided into three components. Just one is responsible for the torque provided in the transmission shafts, which is the tangential component ($F_t$). The two other are the radial component ($F_r$) and the axial component ($F_a$), forces which it is necessary to have into account when choosing the engine. The first one because it causes bending in the motor shafts and the second one because it causes axial stress. Sanyo Denki [28] gives the maximum values for these two forces in Figure 3.4.




| Flange size | Model number | Distance from end of shaft : mm (in) | | | | Thrust load N (lbs) |
|---|---|---|---|---|---|---|
| | | 0 | 5 | 10 | 15 | |
| | | Radial load : N (lbs) | | | | |
| **14 mm sq. (0.55 in sq.)** | SH2141 | 10 (2.25) | 11 (2.47) | 13 (2.92) | - | 0.7 (0.16) |
| **28 mm sq.(1.10 in sq.)** | SH228 □ | 42 (9) | 48 (10) | 56 (12) | 66 (14) | 3 (0.67) |
| **35 mm sq.(1.38 in sq.)** | SH353 □ | 40 (8) | 50 (11) | 67 (15) | 98 (22) | 10 (2.25) |
| **42 mm sq.(1.65 in sq.)** | 103H52 □□<br>SH142 □ | 22 (4) | 26 (5) | 33 (7) | 46 (10) | 10 (2.25) |
| **50 mm sq.(1.97 in sq.)** | 103H670 □ | 71 (15) | 87 (19) | 115 (25) | 167 (37) | 15 (3.37) |
| **56 mm sq.(2.20 in sq.)** | 103H712 □ | 52 (11) | 65 (14) | 85 (19) | 123 (27) | 15 (3.37) |
| | 103H7128 | 85 (19) | 105 (23) | 138 (31) | 200 (44) | 15 (3.37) |
| **60 mm sq.(2.36 in sq.)** | 103H782 □ | 70 (15) | 87 (19) | 114 (25) | 165 (37) | 20 (4.50) |
| | SH160 □ | | | | | 15 (3.37) |
| **86 mm sq.(3.39 in sq.)** | SM286 □<br>SH286 □ | 167 (37) | 193 (43) | 229 (51) | 280 (62) | 60 (13.488) |
| **86 mm sq.(3.39 in sq.)** | 103H822 □ | 191 (43) | 234 (53) | 301 (68) | 421 (95) | 60 (13.488) |
| **ϕ106 mm (ϕ4.17 in)** | 103H8922 □ | 321 (72) | 356 (79) | 401 (90) | 457 (101) | 100 (22.48) |

Figure 3.4 - Allowable Radial / Thrust Load [28]

Knowing the torque in the gears, and the gear radius, which is 21.215 mm [24], it is possible to calculate the tangential force using the torque definition. Thus, the maximum tangential radius is the quotient between the maximum torque required with 50% of safety, which is 0.096 N·m in the horizontal axes (Table 10), and the gear radius, resulting in 4.52 N.

Using this maximum value of the tangential force ($F_t$), and the pressure angle of the gears (20º) [24], it is possible to calculate the radial force ($F_r$) by the equation 3.10 [29].

$$F_r = F_t \cdot \tan(\emptyset) \tag{3.10}$$

Where:

$F_r$ is the radial force (N)

$F_t$ is the tangential force (N)

$\emptyset$ is the pressure angle (º)

On the other hand, the axial force ($F_a$) depends on the helix angle, which in the case of the chosen gears is 45º, and can be calculated by equation 3.11 [29].

$$F_a = F_t \cdot \tan(\varphi) \tag{3.11}$$

Where:

$F_a$ is the axial force (N)

$\varphi$ is the helix angle (º)

The radial and the axial forces provided by the previous equations are 1.64 N and 4.52 N, respectively. Since the efforts caused by the helical gears are less than the motor is capable of supporting, it is possible to confirm that no extra bearings will be required to protect the motor.

### *3.1.4 Stepper Motors Drivers*

The BSD 0.2V is a current driver, with adaptive micro-stepping from 400 to 3200 steps per revolution, which offers smooth movement, low noise and vibration control. It has an operating voltage range of 24 to 48 $V_{DC}$ and an adaptive phase current from 0.7 to 2.2 A, which can be adjustable using the driver dip-switches [30]. As RTA [31] ensures coupling with motors with drive current between 0.7 and 2.2A, Voltage from 24 to 48 $V_{DC}$ and inductance 1 to 12 mH, the stepper 103H5208-5240 is compatible since all its parameters are between the specifications. The driver datasheet is shown in Appendix A.

The possibility to control the movements of the linear trolleys by micro-steps offers unique advantages to the system. The micro-step drivers can divide each full step of the motor into smaller steps, obtaining smoothness in the motor rotation, especially at lower velocities. Besides, it results in lower vibrations and noise in the system. The resolution is also improved with this type of drivers which, however, require higher control frequency.

The BSD 02.V manufacture ensures that every motor connected to this driver provide the same torque/velocity curve at every resolution, with an error of 10%. Since the maximum torques required with 50% safety margin (Table 10) are below the entire torque/velocity curve even if there is a 10 % reduction in torque, it is shown that this driver is compatible with this application. For this reason, it was also ordered two more BSD 02.V drivers to command the steppers.

## 3.2 Limit Switches

Once the linear movement of the trolleys has been planned, and in order to ensure the security that they do not go beyond the workbench physical limits, it was necessary to introduce position detectors in its extremities. They protect not only the trolley itself, but also the optical measuring instrument they carry from a possible shock.

Another vital function of the limit switches is to serve as reference positions in the homing routine so that the system always knows the absolute position of the trolleys in relation to the entire optical workbench. Regarding this function, it was thought to use two position detectors, one at the “zero” point reference and the other in the middle of the axle, so that the homing routine would be shorter and would not force the trolley to move over a great distance. Together with the other two safety detectors at the extremities, a total of 3 position detectors were planned to be used on each workbench axis, having the detector at the “zero” coordinate both the safety and reference position functions.

Essentially for the function in the homing routine, position detectors are required with a high repeatability and a quick response. The ones ordered and which satisfied the previous requirements were the PMK-25 micro photoelectric detectors, from Panasonic (Figure 3.5). They have a compact size, a short response time of 20 μs, and low repeatability of 0.1 mm [32], being an economical detector.

Figure 3.5 - PM-K25 photoelectric detector

The maximum velocity of the trolleys is 0.029 m/s, and a response time of 20 µs corresponds to an extra advance of 2.37E-09 mm, which is insignificant compared with its repeatability.

In terms of out of bounds movements of the trolley, the possible 0.1 mm extra displacement does not have great importance in the system. However, when the trolleys are finding its position reference, one displacement of 0.1 mm will affect all future positions of that axis in the same value.

## 3.3 Security Concerns

Once the entire movement of the linear axes and its command has been planned, from the selection of the motors to the position detectors, it was time to think about ways to ensure the safety of the entire application.

Firstly, all access to the gears must be adequately covered in a carefree manner, because a simple distraction or a simple thread of clothing that gets stuck to the gear can result in injury.

An emergency button must also be configured in the application so that all system movements are stopped suddenly if necessary. It must be in an accessible location close to the workbench operator, and whenever pressed, it disconnects the motors. It ensures that in case of software failures, material assets can be safeguarded.

The emergency button, the protective box used to place all stepper drivers, and one of the chain cable carriers used in the development of this optical workbench automation project were those shown in Figure 3.6.

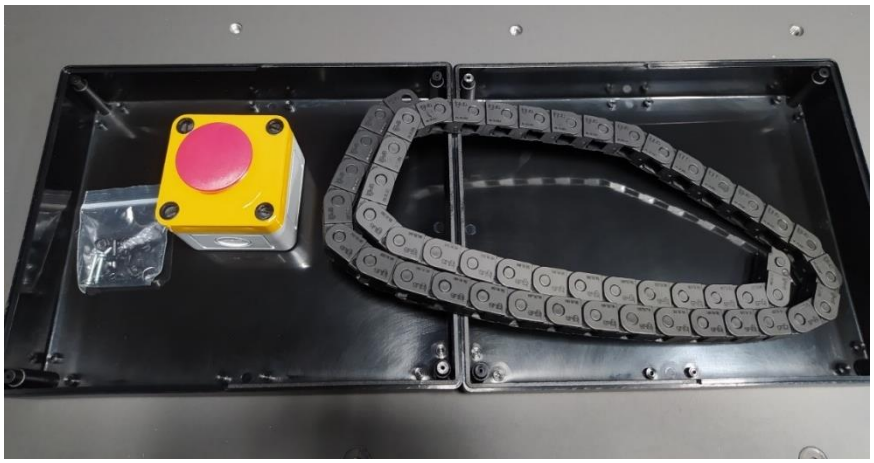


Figure 3.6 - Emergency button, chain cable carrier and protective box (opened)

## 3.4 Concluding Remarks

This chapter essentially served to explain all the components of the proposed solution to be implemented on the optical measurement bench.

The use of stepper motors in open-loop control was chosen to move the trolleys since these were the most economical option, and they already allow good positioning characteristics.

Afterwards, the speed profile was studied, and the stepper motors were chosen according to the characteristics of each axle. Then it was decided to use three position detectors on each axis. They not only have the function of preventing the trolley from going beyond its limits but also intend to serve as a reference position in the homing routines of the final automation.

Lastly, the safety measures to be applied on the workbench were explained.

# 4 Development and Assembly

Taking into account the solution proposal and the chosen material in the previous chapter, this one shows the project development and illustrates the materials assembly in the optical workbench. Extra parts needed to fix the components were designed, being its CAD files exposed in this chapter.

It will start by describing the stepper drivers configuration in order to realise the motor operations, and by explaining the assembly of all the transmission system. After, the parts to fix the photoelectric detectors are revealed, as well as the tasks performed to increase the system safety. At the end, it will be narrated the transmission and movement of a rotational axis of the workbench. It was possible to automate also that axis in this project since there was extra time for its realisation.

## 4.1 Stepper Drivers Configuration

The driver has four dip-switches that can adjust the resolution and the nominal phase current sent to the motor windings, which can be 0.7, 1.1, 1.6 or 2.2 amper. The 103H5208-5240 motor only accepts up to 1 ampere of current in its windings, so the dip switches have been set to 0.7 amper.

Table 11 shows the maximum velocities defined in the previous chapter, as well as the frequencies necessary to run the motor at the different micro-step resolutions.

Table 11 - Frequencies necessary to run the motor at the maximum stipulated velocity, in function of the driver resolution

| | | steps per revolution | | | |
|---|---|---|---|---|---|
| | max. velocity (m/s) | 400 | 800 | 1600 | 3200 |
| X-axis | 0.0278 | 2222 Hz | 4444 Hz | 8889 Hz | 17778 Hz |
| Y-axis | 0.0296 | 2370 Hz | 4740 Hz | 9481 Hz | 18963 Hz |
| Z-axis | 0.0222 | 1778 Hz | 3556 Hz | 7111 Hz | 14222 Hz |

The generation of the driver control signals (step, direction, start/stop) was performed using LabVIEW and an Arduino Mega 2560 board, since it was available in the lab and it is a low cost solution. Through LIFA (LabVIEW interface for Arduino) or LINX, a library from Digilent, the interaction with the Arduino is possible by the download of a pre-built firmware in the board, programmed to be continuously waiting to LabVIEW orders.

After some tests it was possible to verify that the maximum frequency sent to the driver was around the 100 Hz, a low value compared with the frequencies of Table 11.

However, in the Arduino IDE, it was possible to modify the firmware of the board and realize pre-configured functions which, every time the LabVIEW sends a codified command, the Arduino board realize its respective function without the interference of the LabVIEW. Thus, the maximum frequency generated is only limited by the capacity of the Arduino board, and not by the communication LabVIEW/Arduino.

To perform the firmware modification in the Arduino IDE, it was used the AccelStepper library, which allows an easy configuration of stepper motors and, therefore, smooth accelerations in its command. However, the accelerations use square roots in its calculation, which limits the frequency of pulse signals supplied to the stepper driver, and, for devices as Arduino Mega 2560 that has a clock frequency of 16 MHz, it reduces the maximum capacity to 4000 steps per second [33]. Consequently, and having in consideration the required frequencies of Table 11, it was preferable to use the driver with a resolution of 400 steps per revolution in order to provide the expected acceleration.

Figure 4.1 shows the stepper driver already configured, with all its electrical connections and the dip-switches in the chosen positions.


Figure 4.1 - BSD 02.V stepper driver

After the stepper driver configuration, basic programs in LabVIEW could already perform simple commands in the motor. For that, it was required a boolean signal to activate the driver (start/stop), another one to indicate the rotation direction (direction) and a digital signal pulse with the desired frequency (step), which is directly proportional to the motor output speed.

## 4.2 Motors and Transmission Assembly

As mentioned in the previous chapter, in order not to increase the workbench dimensions, it was requested that the motors of the horizontal shafts were not directly coupled to the shaft, but underneath it, transmitting the torque through two gears. On the contrary, as the optical laboratory has no space problems concerning height, the vertical axis motor was directly coupled to its ball screw axis.

Through a CAD program, support parts have been designed for fixing the three motors, which are shown in Figure 4.2. In order to produce them quickly and economically, the drawings were converted into ".stl" files and then the parts were printed in ABS (acrylonitrile butadiene styrene) using a 3D printing machine available.

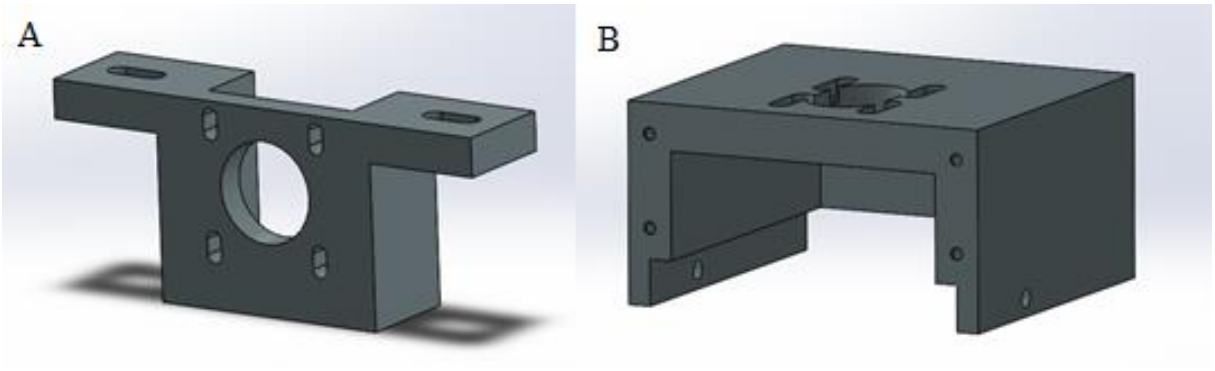



Figure 4.2 - Cad image of motors support parts in the horizontal axes (A) and in the vertical axis (B)

In the vertical axis of the optical test workbench, the coupling between the motor and the ball screw shaft was realized through a flexible shaft coupling, where each shaft was fixed to it using two M4 hex socket set screws. After all assembly, the motor and the flexible shaft coupling mounted on the vertical axis of the workbench is shown in Figure 4.3.


Figure 4.3 - Motor and transmission assembled in the vertical axis.

Regarding the horizontal axis, it was necessary to design and machine a shaft that was able to transmit the torque between the motor output, with 5 mm of diameter, and the gear, with 12 mm of shaft bore diameter. After the assembly of it with the gears, the final product is visualized in Figure 4.4.

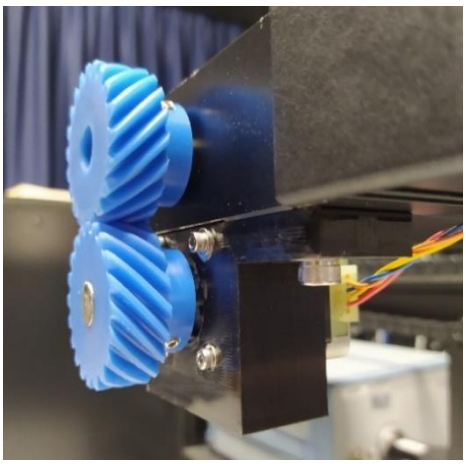


Figure 4.4 - Motor and transmission assembled in the horizontal axes

After all the linear workbench axes were already assembled, it was measured the possible clearance in the change of direction of the gears rotation movement (backlash). In order to measure it, since it would have a contribution to the accuracy and repeatability in the workbench automation, an analogue dial indicator was used against the horizontal trolleys of the workbench. By commanding the motor to be stopped and by manually rotating the ball screw shaft, it was possible to check the linear backlash in the trolleys resulting from the backlash of the gears. The measured values are visible in Table 12. As in the vertical axe the motor is directly coupled with the ball screw shaft and the weight keeps the components under contact, it was not found any backlash in the trolley movement.

Table 12 - Linear backlash resulting from the gears backlash

| | Linear Backlash (mm) |
|---|---|
| X-axis | 0.02 |
| Y-axis | 0.02 |
| Z-axis | 0.00 |

## 4.3 Photoelectric Detectors Fixing

As mentioned in the previous chapter, three detectors were introduced in each linear axis of the optical workbench, one at the beginning, one in the middle and one at the end. To fix them, polymeric parts were designed in a CAD program and later were printed in the 3D printer machine. The choice of material once again took into consideration the company's stock of polymers for 3D printing, and the material chosen was a transparent PolyJet photopolymer. In the same way, it was produced another piece that it was fixed to the trolley in order to block the

sensor's beam whenever the parts pass through. The CAD drawings of some of these parts are visible in Figure 4.5.

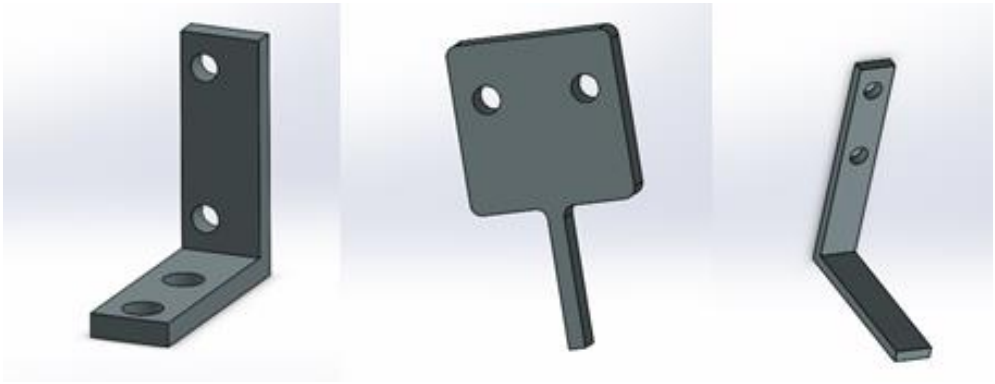


Figure 4.5 - 3D part to support the detectors

After mounting the photoelectric detectors in the limit positions of each workbench axis, the third detector was also introduced in an intermediate position of each trolley's travel (Figure 4.6), with the function of decrease the time the system needs to recognise its trolleys positions. However, the distance between them and the initial detector must be precisely known to minimise position errors as much as possible, regardless of the detector to which the trolley has travelled to recognise its absolute position.

Figure 4.6 - Intermediate detector and its support parts of one of the horizontal axes

Since it was already possible to command the motors, they were used to measure the average number of motor steps between the two detectors (Table 13). Once the motor was command with 400 steps per revolution, and each rotation of the ball screw moves the trolley 5 mm, it was possible to know the distance, also shown in Table 13.

Table 13 - Motor steps recorded between the two photoelectric detectors on each axis

| | X-axis | Y-axis | Z-axis |
|---|---|---|---|
| Average steps | 28525 | 34072 | 26276 |
| Distance (mm) | 356.56 | 425.90 | 328.45 |

## 4.4 Protection and Safety

Once the system was minimally operational, with the motors, transmission elements and photoelectric detectors adequately assembled, some tasks were carried out in order to increase the safety of the workbench.

The electrical cables have been placed wherever possible inside of the workbench and chain cable carriers were installed to protect the electrical cables in motion, as is shown in Figure 4.7.

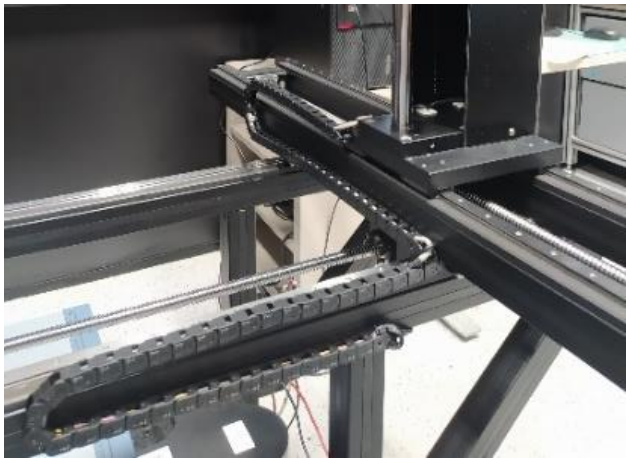


Figure 4.7 - Chain cable carriers assembled in the workbench

In order to protect all stepper drivers, the Arduino board, and all their electrical connections, a protective box was used (Figure 4.8).

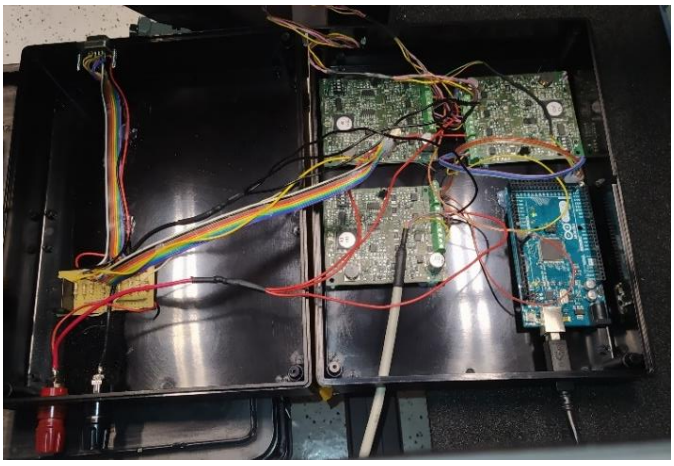


Figure 4.8 - Box opened with all its electrical connections

Regarding the protection against the gears, parts to involve them have been designed through a CAD program.

Last but not least, the emergency button was placed in one extremity of the workbench, near the computer from where the optical engineer will be commanding it. In addition to cutting the power supplied to the system, it is responsible for cutting the power to the coil of a relay, which will also shut down properly the software. In Figure 4.9, it is possible to see the emergency button, one of the protection parts against the gears as well as the protective box located under the workbench.


Figure 4.9 - Emergence button, protection part against gears and the protective box (below)

## 4.5 Rotation Axis

Since there was an extra time after the development and command of the three linear axes, it was added a rotational axis, the angle relative to the vertical axis (Figure 4.10).

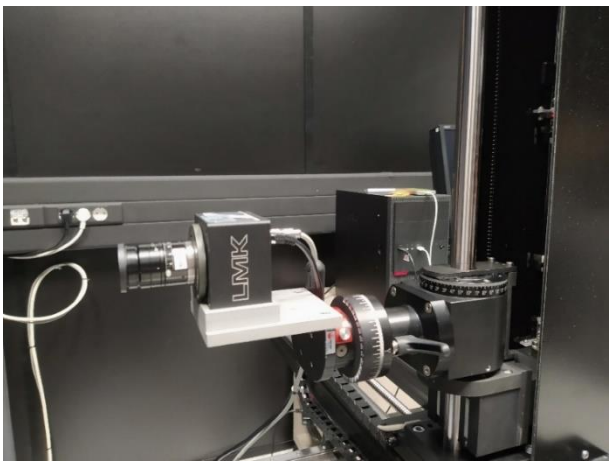



Figure 4.10 - Rotational axis of the optical workbench

For this purpose, a motor located in the optical laboratory, which had been previously bought to perform this same function, was used. It is the DDR25/M servomotor of the Thorlabs, based on a brushless DC rotary motor, and controlled by one KDC 101 controller, also purchased previously to the task. The motor has a torque capacity of up to 1.8 Nm, and it was verified that it could control the rotational axis without the use of any other sensor, as the motor always recognizes its origin in 360º. For this reason and the fact that the LMD support only rotates a maximum of 180º because of its constructive solution (Figure 4.10), a 1:2 gear transmission at the motor output was set. It not only takes advantage of the maximum ranges of both the LMD support and the motor but also increases torque capacity at the expense of rotation speed.

Therefore, the gears were established based on GEABM1.0-50-10-A-13 and GEABM1.0-100-10-A-26. The servomotor support has also been developed and designed in CAD software in order to meet the mechanical requirements of the rotary axis, being the parts shown in Figure 4.11.

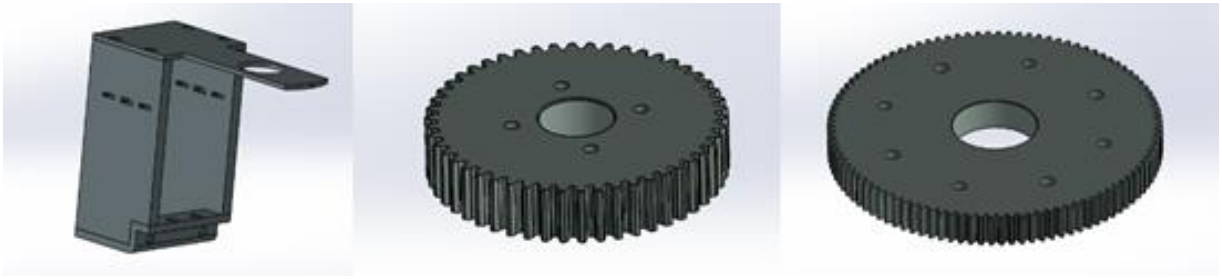


Figure 4.11 - CAD file of motor support and gears of rotary axis

Due to time constraints, both the gears and the motor support were designed to be printed on the company's 3D printers. However, technical problems in the last weeks made it impossible to obtain the parts, leaving the motor unassembled in the rotation axis.

In any case, the motor has already been properly configured for the task, to be as simple as possible to implement it on the axle. Given the software explained in the next chapter, it already contemplates the control of this servomotor, which will perform the movement of the rotation axis.

## 4.6 Concluding Remarks

This chapter explains the implementation of the solution proposed in the previous chapter. It was explained all the tasks performed on each workbench axis, from the production of motor support parts, their implementation, and the assembly of the gears. A maximum backlash of 0.02 mm was measured, which allowed knowing in advance part of the error that could occur in the trolleys positioning.

The plastic parts for fixing and actuating the detectors were also shown in this chapter. Once they were fixed, the exact distances between the intermediate and the initial detectors of each axis were measured by the motor steps number taken during those courses.

The workbench final appearance is shown in Figure 4.12.

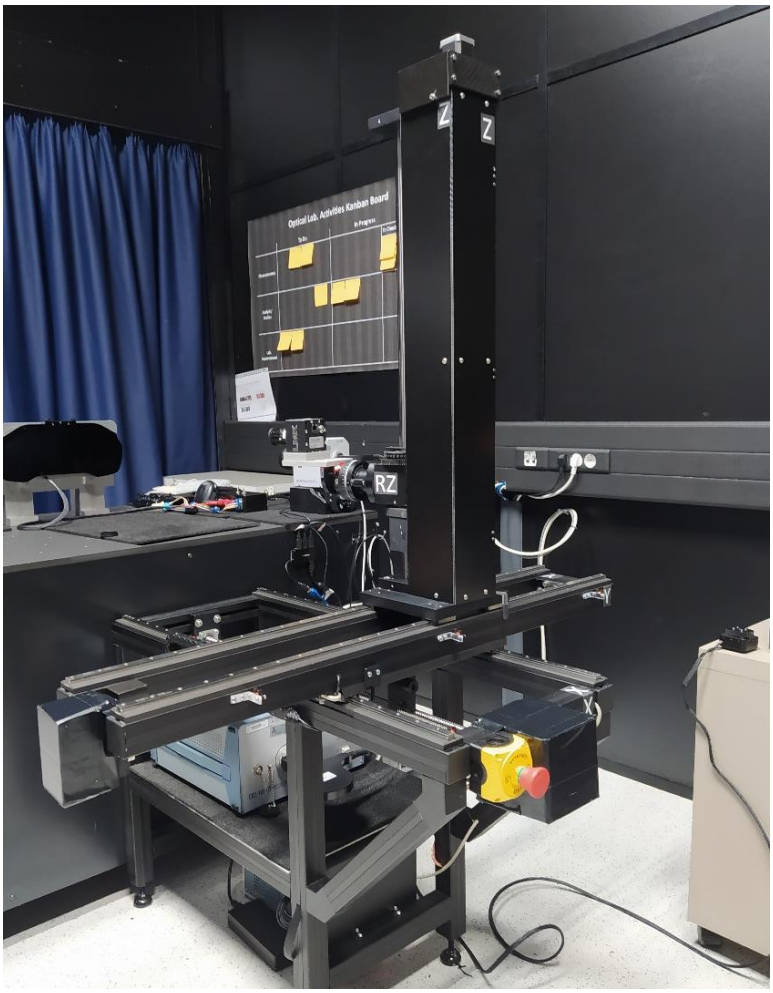



Figure 4.12 - Workbench final appearance

Besides, a system for rotating the vertical axis of the bench was also described in this chapter. To control this axle, the Thorlabs DDR25/M servomotor, which had previously been purchased for the same purpose, was considered.

Figure 4.13 shows a scheme with all the hardware used to automate the four axles of the workbench.

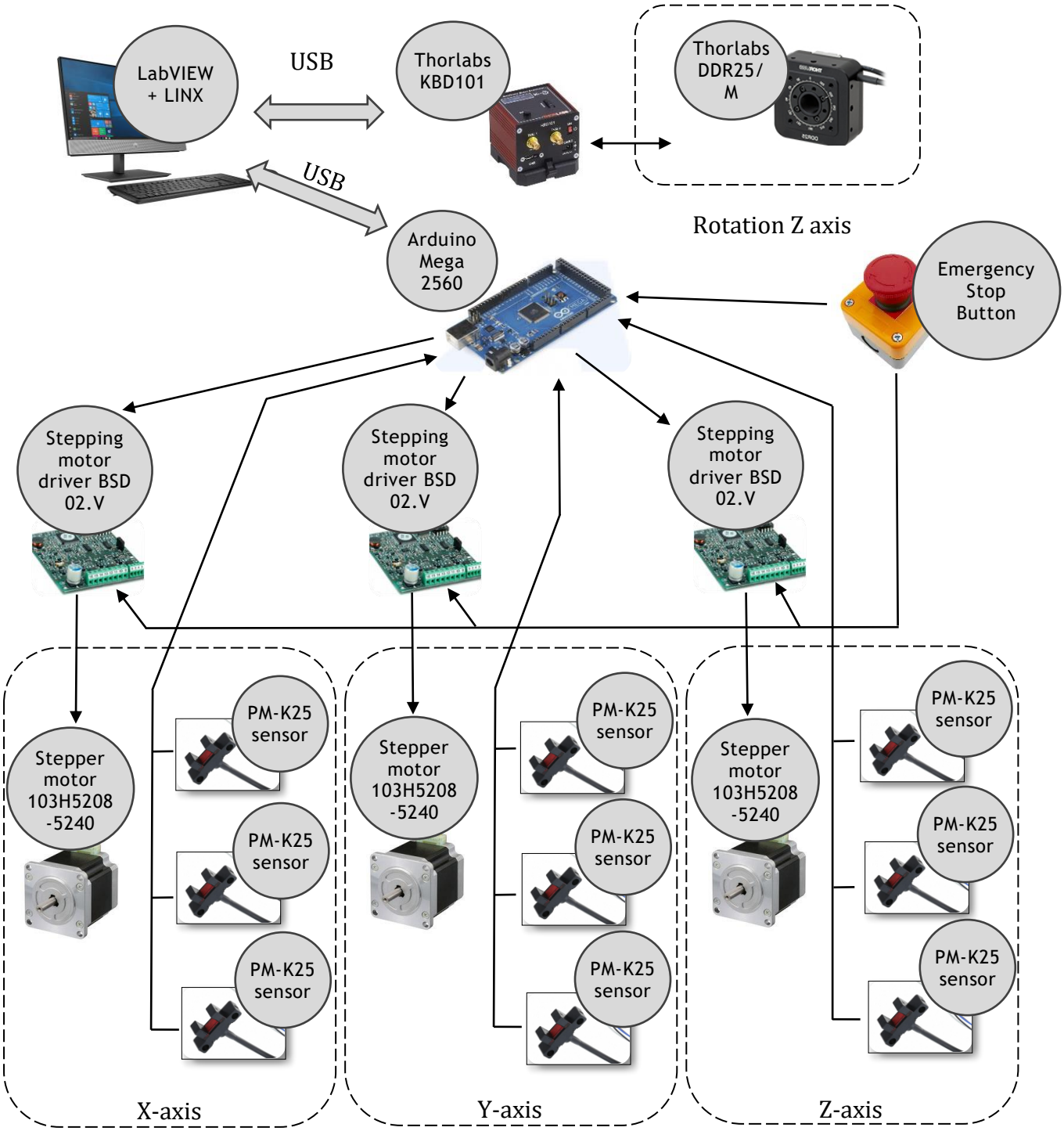



Figure 4.13 - Hardware installed in the workbench

# 5 Software Application

This chapter demonstrates the solution realized in LabVIEW, which allows the command of the three linear axes of the workbench as well as one rotary axis. The general architecture of the final software solution will be explained, as well as the communications between the various sub-programs. The electronic connections will then be displayed, and a general explanation of the algorithm behind the program will be realized. At the end of the chapter, the main interface of the program and its features will be demonstrated.

## 5.1 Overall Architecture

As required by the system prerequisites, the workbench motion command application was performed on the LabVIEW software, with plans to integrate with other LabVIEW applications of the company in the future.

However, an Arduino board was used for communication between the virtual interface in LabVIEW and the system inputs and outputs, such as the photoelectric detectors and the stepper motors. The communication between the LabVIEW and the Arduino was realized by the LINX (Figure 5.1).

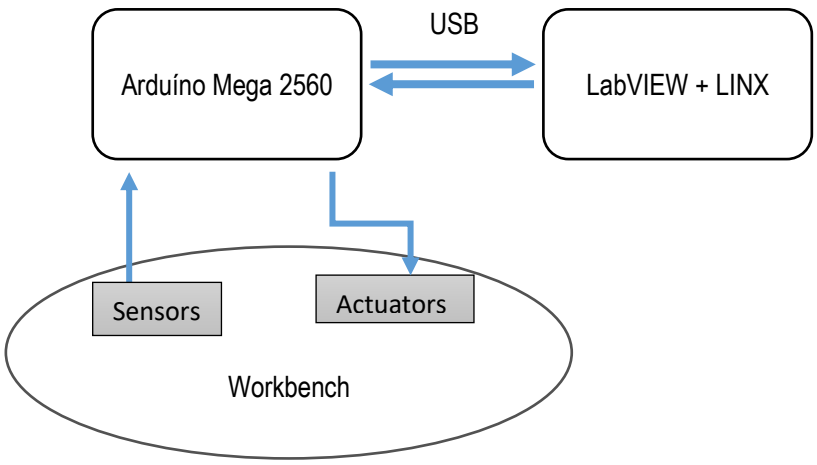



Figure 5.1 - Relation between LabVIEW, Arduino and the system

### 5.1.1 Arduino

The Arduino board is open-source hardware, equipped with a microcontroller, and with sets of digital and analogue inputs/outputs pins, invented to interact with its environment via sensors and actuators. It can be connected to various expansion boards (shields), and with other electronic circuits. The Arduino board uses serial communication ports to connect to computers and allow loading of programs. The programming can be done using the C and C++ languages in the integrated development environment (IDE), which is also open source. In this particular case, the board used was the Arduino Mega 2560, mainly due to its immediate availability in the optical laboratory.

### 5.1.2 LabVIEW

The LabVIEW (Laboratory Virtual Instrument Engineering Workbench) is a systems engineering software for applications that require test, measurement, and control with rapid access to hardware and data insights. It was developed by National Instruments, and it uses a visual programming language, named "G". It is used to perform functions as data acquisition, instrument control, test automation, analysis and signal processing.

Concerning the command of the workbench, it was developed some virtual instruments (VI's) to perform the tasks of reading detector signals, command the four motors, read and write in a ".csv" file and define the positions in every instant of the four axes of the workbench. The main VI is the user machine interface, which command the four axes by coordinates, by selecting one position of the pre-defined list or by manual jogging each of the four stepper motors. Through the same VI, important positions that will need to be performed more often in future optical measurements can be saved in the pre-stipulated list to be remembered and quickly chosen in the next time.

### 5.1.3 LINX

The LINX project by Digilent is an open-source project which allows the development of embedded applications based on chipKIT or Arduino, using LabVIEW. It includes hardware-agnostic applications program interfaces (API's) for accessing peripherals as digital I/O, analogue I/O or PWM (pulses with modulation), and it has VI's for some of the most common embedded systems. The LINX make it easy to visualize the operational data, to debug the code, and to create advanced embedded applications, by loading firmware in the embedded applications boards.

In the current project, besides the VI's for writing and reading analogue and digital signals, custom commands were used, which allows the system control both via the visual programming language of the LabVIEW and via C/C++, the language used in Arduino's IDE, which offers greater versatility.

## 5.2 Electric Connections

In order to enable the command of the optical measuring workbench by the software, it was necessary to connect the hardware devices to the Arduino Mega. Thanks to LINX, which allowed communication between the main VI and the inputs and outputs of Arduino Mega, the program can either read the position detectors consecutively, to ensure that the trolleys do not go beyond their limit positions, or operate the motors whenever it is required.

The electric connections are shown schematically in Figure 5.2. Since the Arduino board does not provide the control of the rotary axis motor, it is not included in the schematic diagram. This motor is controlled by a KDC 101 controller, which in turn is controlled by LabVIEW via a USB port.

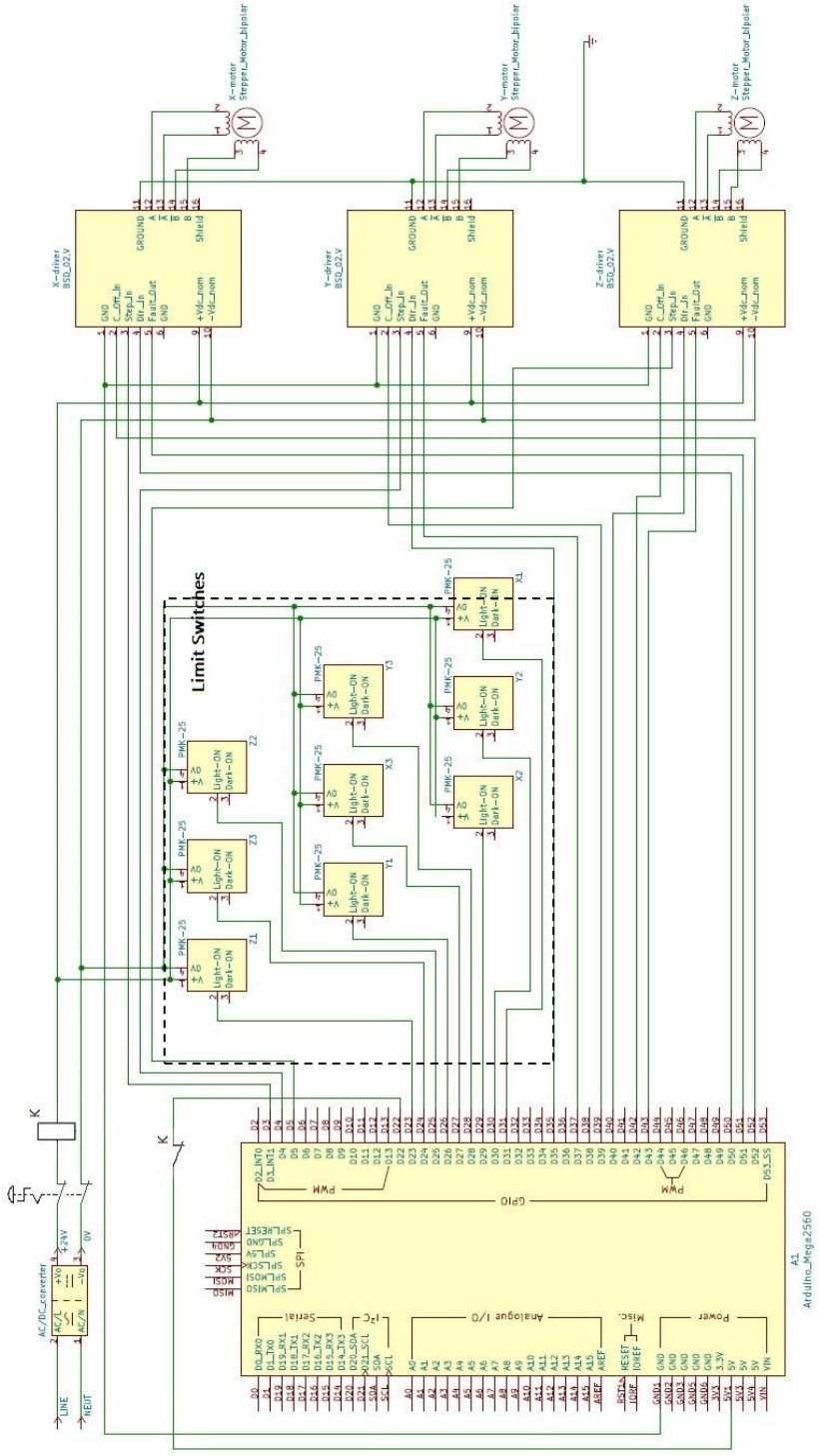



Figure 5.2 - Electric connections to Arduino board

## 5.3 State Machine Programming

The state, the event and the action are the three fundamental concepts of state machines. The state is the status or position of the system. It can have various actions associated with it, or, as an example, can just be waiting for user input. Each state can lead to one or multiple states and can also end the process flow. The events are incidences with meaning for the program. Can be just a simple input as a button click or a time trigger, and associated with specific states, they can be responsible for the changing between them. On the other hand, the system will perform in each state a set of actions associated with it. They can be, among many other things, movements of a specific engine, the performance of certain calculations or the introduction of a message to the user.

In the course of this section, more details on the programming logic will be covered, such as the operating modes to move the workbench, the logical states of the system and the respective state diagram.

### 5.3.1 Actuation Modes

To move the trolleys of the workbench, the user can proceed in three different ways: by relative moving, by inputting absolute coordinates or by reading a cluster list of positions.

The first way is the most basic and consists of moving each of the four motors individually. It is denominated “Manual Mode”, and it is only necessary to click on the directional arrows corresponding to each motor to move it forwards or backwards at a pre-selected distance (Figure 5.3). By default, the distance is always 10 mm, but the user can adjust this value.

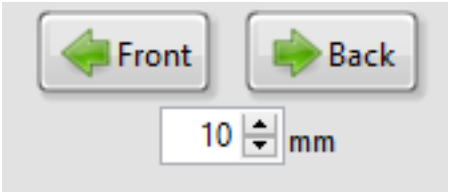



Figure 5.3 - Interface part where is possible to move one axis in the “Manual Mode”

The second way to command the movement of the workbench is through the “Automatic Mode”. The user must enter the coordinates for each of the four axes and click the “Go” button. Then all the trolleys will move to the desired position.

However, for repeated digital instrument clusters and the same type of optical tests, the measurement positions are the same. One way to avoid the constant input of the four desired coordinates is to store important positions in a list which the user has access. So, he only needs to select the name of the cluster he pretends to measure and the light measurement device, which is fixed in the workbench, will move to the correct position.

### 5.3.2 States

The main VI of this position control project consists of 13 different states, which are associated with different functions. The program starts in the "Initialize" state, where the connection to the Arduino is established, and then proceeds to the "Home" state. There, all the motors start rotating in the negative direction until they find one of the photoelectric position detectors, which makes the system recognize the true position of each axis at that exact moment. Then, the next state will be the "Read" state, where the system reads the saved workbench positions from a ".csv" file, arranges them in a table to be visible in the program and fills a "combo box" with the names of the digital instrument clusters for which these positions have been saved. The next is the "Wait for event", the central state of the whole program, where the system will be constantly waiting for an input from the user. The list of all states can be visible in Table 14.

Table 14 - Enumeration of each program state

| State Name | Description |
|---|---|
| Initialize | Initial state, where the connection with the Arduino is established by LINX. The following state is always the "Home" state. |
| Home | Can starts after the "Initialize" state, or after a user input in the "Home" button. The positions of the 4 axes of the workbench are found, as trolleys are taken to reference positions. If there are no errors, the next state is always the "Read Positions" state. |
| Read | The state always initializes after the "Home", "Delete" and "Add" states. The positions previously saved in a ".csv" file are read and a combo box is filled with the names of the clusters for which these positions have been saved. The next state is always the "Wait for Events" state. |
| Wait for Events | It is the central state of the program, which continually waits for user inputs. Depending on the input received, the system is redirected to the corresponding state. After the system has completed all its functions in the other states, it is to this state that the system returns typically. |
| Go | It always starts after the user click in the "Go" button. It converts the distance introduced in the variable X-coordinate, Y-coordinate and Z-coordinate into steps, and using a custom command, make all the motors go to the desired positions. Simultaneously, it directly controls the motor of the rotational axis to guide the LMD support to the desired angle. After, it returns to the "Wait for Event" state. |
| Combo box | This status starts as soon as the user selects a digital instrument cluster in the combo box. This state reads in the ".csv" file the coordinates necessary to measure that cluster and enters them into the X-coordinate, Y-coordinate, Z-coordinate and Rotation Z variables. After, it returns to the "Wait for Event" state. |

| | |
|---|---|
| Add | This state is started after clicking on “Add” button and records the current position of the 4 axes in the “.csv” file using string manipulation. To do this, it asks the user to insert the name of the cluster he measures in that specific position. After saving the position to the “.csv” file, the next state is always the “Read Positions” state. |
| Delete | After clicking the “Delete” button, the status starts. A sub VI appears and displays a list of the clusters associated with the positions currently saved in the “.csv” file. After the user selection of one, the position is deleted from the file, and then the system goes to the “Read Positions” state. |
| Single X | This state starts every time the click on “front” or “back” button is done and allows the forward or backward relative movement of the x-axis trolley respectively. The next state is always the “Wait for Event” state. |
| Single Y | This state starts every time the click on “left” or “right” button is done and allows the leftward or rightward relative movement of the y-axis trolley respectively. The next state is always the “Wait for Event” state. |
| Single Z | This state starts every time the click on “up” or “down” button is done and allows the upward or downward relative movement of the z-axis trolley respectively. The next state is always the “Wait for Event” state. |
| Single RZ | This state starts every time the click on “Clockwise (CW)” or “Counter-clockwise (CCW)” button is done and allows the clockwise or counter-clockwise relative movement of the z-angle of the LMD support respectively. The next state is always the “Wait for Event” state. |
| Exit | This state starts whenever there are errors in states that have as function the movement of the trolleys and can also be initialized by user input in “Exit button”. It is responsible for correctly terminating communications with Arduino, with the management of the occurred errors, and for assigning a “True” constant to the while loop’s conditional element, causing the programme closure. |

In order to understand better the relation between the different states, and in which conditions they change, Figure 5.4 shows the state diagram. The variables of the main VI, including the variables used in the communication between the LabVIEW software and the Arduino IDE, are enumerated in Appendix B to facilitate the visualization of the state diagram.

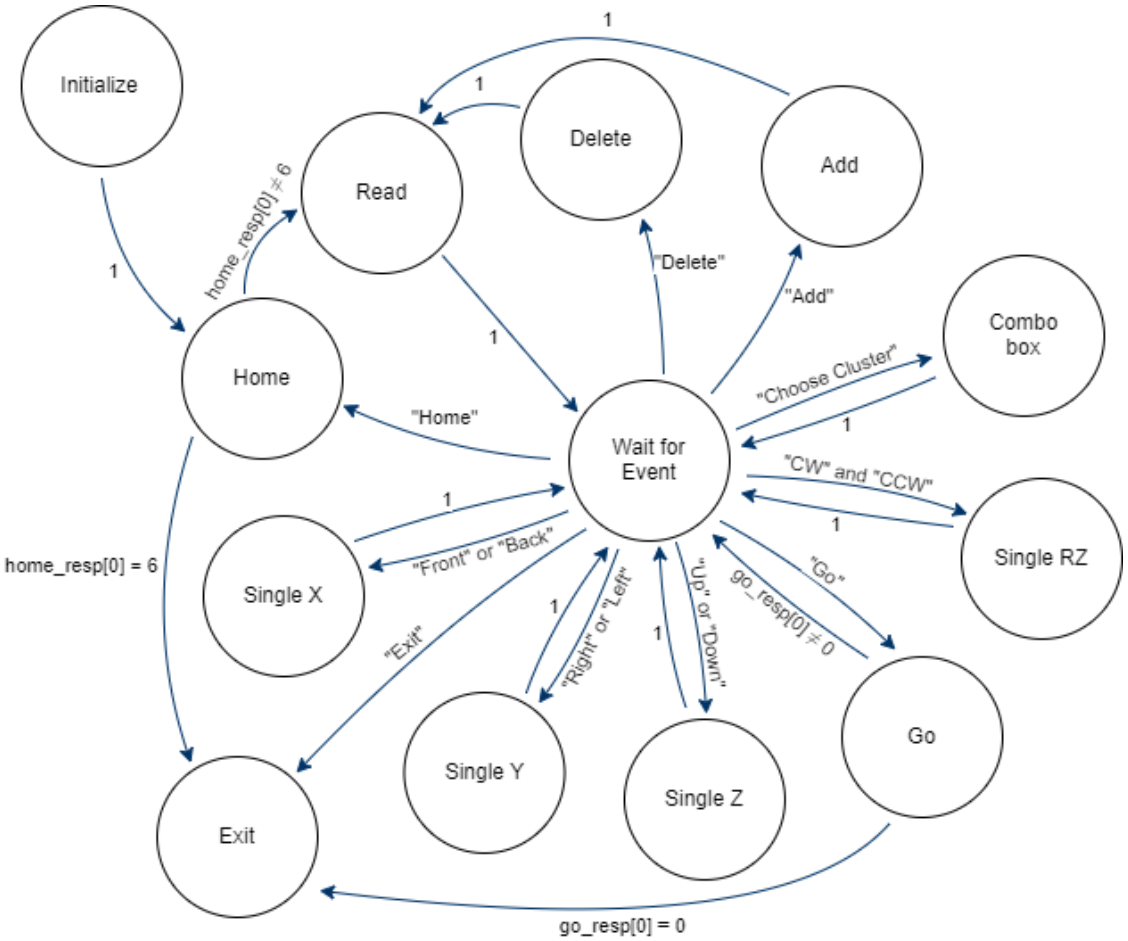



Figure 5.4 - State Diagram

Parallel to this entire status machine, the program is continually reading the signals from the position detectors installed in each of the linear axes and demonstrating them to the user in the form of virtual LEDs. These are located next to a graduated bar showing the current position of the trolley along each axis, and the LEDs only become visible if the signal from the detector is positive. Figure 5.5 is part of the programme interface and shows one LED visible, which means that the corresponding trolley is acting the respective photoelectric detector.

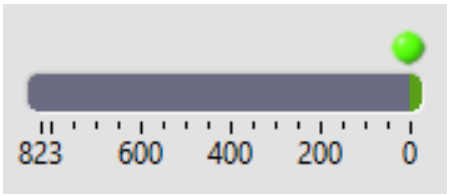



Figure 5.5 - Interface part where is visible that one of the trolleys is actuating its limit switch

### 5.3.3 LabVIEW and Arduino Code

Since are known the application states and the way they are related to each other, this section will show some of the programming code used in both LabVIEW and Arduino software.

In LabVIEW, the main VI consists of two while loops running in parallel, one containing the state machine, responsible for coordinating the program along with the states, and the other continually measuring the value provided by each of the photoelectric detectors.

The state machine is performed in LabVIEW by a case structure inside a while loop, wired by an enumerate constant, a particular type in the software where it is possible to create a list of string labels which will be associated with integer values. This constant, which needs to be called initially, is responsible at all times for telling the following system status and, with the help of the shift registers, can pass this information around in the while loop during the entire program execution. The enumerate constant makes the programming easier to the developer, once he is looking to words describing the next state, instead of being looking to a code number.

In the state machine created, each case of the case structure is associated with one of the program states (Table 14). The states that have no interference with the workbench movement ("Initialization", "Read", "Save", among others) are totally commanded through the LabVIEW code that is associated to that case. The states responsible for the motion ("Home", "Go" or "SingleX", among others) use a custom command that will communicate with the Arduino board and where the functions created in its firmware will be responsible for the workbench movement.

Since it is impossible to demonstrate the code associated to all the states of the application, due to its extension, the "Go" state code is available in Figure 5.6 since it has a set of features that serve as an example of what passes in the code related to the other states.

On the left side of the state machine while loop, it is possible to see the initial procedures that the program performs at the beginning of the application. Start by indicating in the "info_string" variable, which corresponds to an information text field located in the bottom left corner of the interface, that the system is initializing. Then it makes the connection with Arduino's board through the LINX library, being for this only necessary to establish the serial port and the Arduino's baud rate override. A memory control function is then used to create a reference for the data that will be transferred and accessed by the two while loops in a serialized way. At the same time, an enumerate constant provides information to the while loop state machine that it should be started in the "Initialize" case. Still, on the left side of the while loop, it is possible to see a LabVIEW sub VI that also runs at the beginning of the program. It is specifically responsible for the connection and the initial settings of the rotation z-axis servomotor. Throughout the program, the use of sub-VI's is recurrent to organize and simplify the viewing and understanding of the code.

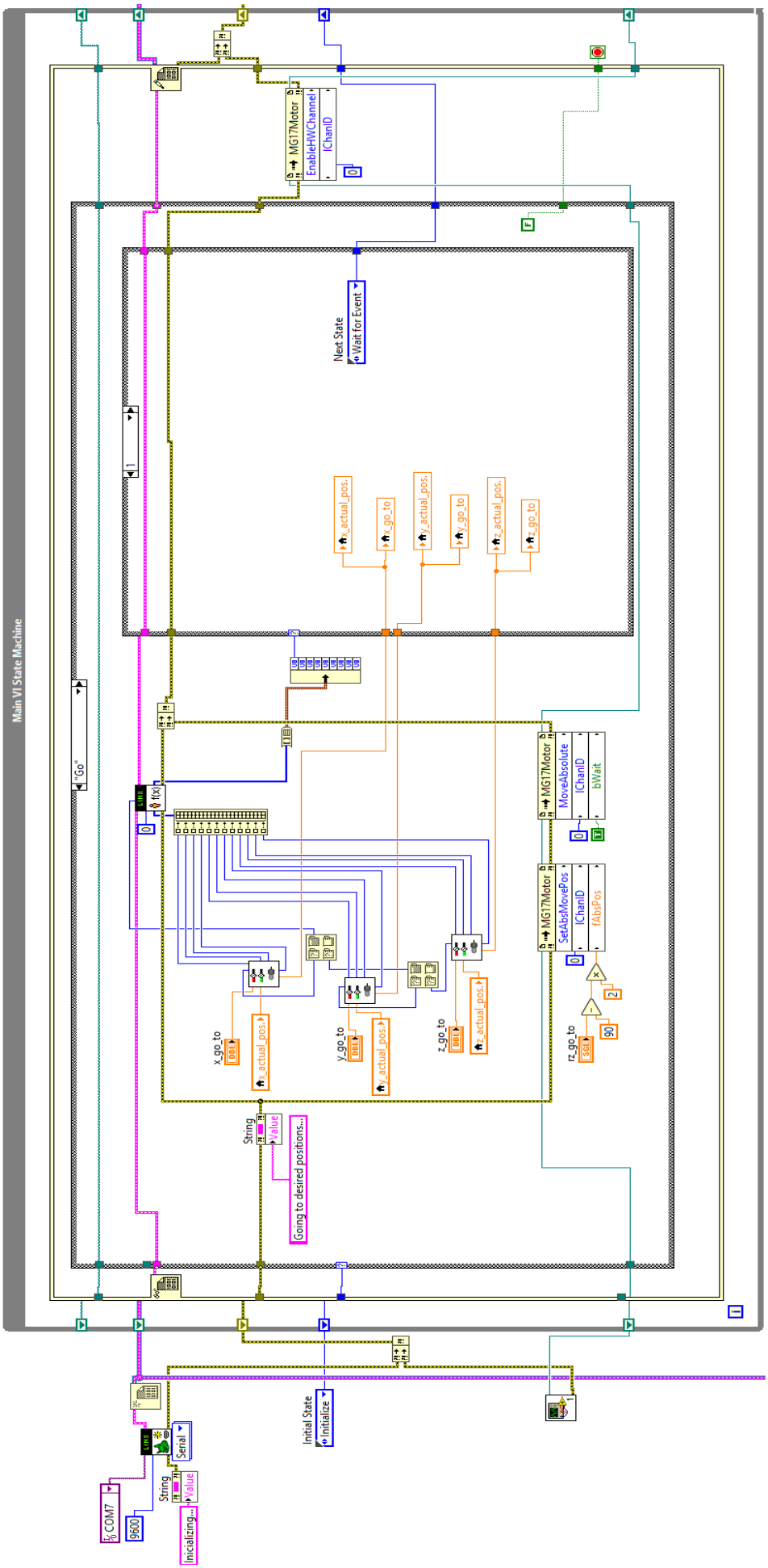



Figure 5.6 - LabVIEW code of “Go” state

Meanwhile, within the state machine while loop, it is possible to visualize the case structure that shows the code associated with the “Go” state. This case also starts by providing information in the variable “info_string”. After, from the actual and the desired positions of each trolley, the necessary motor steps are calculated in sub-VI's. This information is sent to the custom command 0, which will make the Arduino board perform the personalized function number 0 of its firmware, that is called “Gofunction”. This function will convert the motor steps in a real motion of the motors and, if any problem occurs, will send a byte to the LabVIEW containing the number 1, which means the programme can progress to the “Wait for Event” state.

The bottom of Figure 5.6 also shows the servo motor control from the desired position of its axis. It is executed entirely through the LabVIEW, and in the “Go” state only two property nodes are used: one to define the target position and the other to perform the motor movement.

In order to also demonstrate the code performed in the Arduino firmware, the “Gofunction” is provided in Figure 5.7. This function defines the maximum speeds and accelerations for each motor and, from the inputs received from LabVIEW, it defines the necessary steps that each motor must perform, storing them in the variables “go1”, “go2” and “go3”. From there the motors are commanded to their respective positions and, unless none of the limit detectors is activated or the system power fails, the Arduino will send the output “1”. In case of power failure, the Arduino board will communicate an output “2”, and in case of limit detectors actuation, an output “1”, which will trigger different reactions in the program. It should also be noted the signal debounce made during the “Go function”. It was necessary to introduce it to control the oscillations of the photoelectric detectors.

The LabVIEW code corresponding to all other program states as well as the different functions performed in the Arduino firmware can be seen in Appendix C.

```
int Gofunction(unsigned char numInputBytes, unsigned char* input, unsigned char* numResponseBytes, unsigned char*
go_resp)
{
  stepper1.enableOutputs();
  stepper2.enableOutputs();
  stepper3.enableOutputs();

  stepper1.setMaxSpeed(2223);
  stepper2.setMaxSpeed(2371);
  stepper3.setMaxSpeed(1778);

  stepper1.setAcceleration(3000);
  stepper2.setAcceleration(3000);
  stepper3.setAcceleration(3000);

  go1 = ((long)input[0]*1000)+((long)input[1]*10)+((long)input[2]);
  go2 = ((long)input[4]*1000)+((long)input[5]*10)+((long)input[6]);
  go3 = ((long)input[8]*1000)+((long)input[9]*10)+((long)input[10]);
```

```
  if (input[3]==1)
    {
        go1 = -go1;
    }
  if (input[7]==1)
    {
        go2 = -go2;
    }
  if (input[11]==0)
    {
        go3 = -go3;
    }

  stepper1.setCurrentPosition(0);
  stepper2.setCurrentPosition(0);
  stepper3.setCurrentPosition(0);
  bool securitySensor;

 if ((digitalRead(31)==LOW) && (digitalRead(27)==LOW) && (digitalRead(28)==LOW) &&
(digitalRead(26)==LOW) && (digitalRead(23)==LOW) && (digitalRead(24)==LOW))
      {
        stepper1.moveTo(go1);
        stepper2.moveTo(go2);
        stepper3.moveTo(go3);

        securitySensor = (digitalRead(31)==LOW) && (digitalRead(27)==LOW) && (digitalRead(28)==LOW) &&
        (digitalRead(26)==LOW) && (digitalRead(23)==LOW) && (digitalRead(24)==LOW);

        while( securitySensor && (digitalRead(22)==HIGH) && ((stepper1.currentPosition() != go1) ||
        (stepper2.currentPosition() != go2) || (stepper3.currentPosition() != go3)))
              {
                stepper1.run();
                stepper2.run();
                stepper3.run();

                securitySensor = (digitalRead(31)==LOW) && (digitalRead(27)==LOW) && (digitalRead(28)==LOW)
               && (digitalRead(26)==LOW) && (digitalRead(23)==LOW) && (digitalRead(24)==LOW);

                if (false == securitySensor) // debounce for digitalRead if any sensor has changed its value
                      {
                        delay(30);
                        securitySensor = (digitalRead(31)==LOW) && (digitalRead(27)==LOW) &&
                      (digitalRead(28)==LOW) && (digitalRead(26)==LOW) && (digitalRead(23)==LOW) &&
                      (digitalRead(24)==LOW);
                      }
              }
    }
  *numResponseBytes = 1;
  if ((stepper1.currentPosition() == go1) && (stepper2.currentPosition() == go2) && (stepper3.currentPosition() ==
go3))
    {
      go_resp[0]=1;
    }
  else if (securitySensor==false)
    {
      go_resp[0]=0;
    }
  else
    {
      go_resp[0]=2;
    }
  stepper1.disableOutputs();
  stepper2.disableOutputs();
  stepper3.disableOutputs();
  return 0;
 }
```

Figure 5.7 - Arduino code of “Gofunction”

## 5.4 User Interface

During the development of the program interface, and helped by the feedback from the lab collaborators, a large set of advancements were made in an attempt to simplify the interface as much as possible, leaving it increasingly friendly and visually agreeable (Figure 5.8).

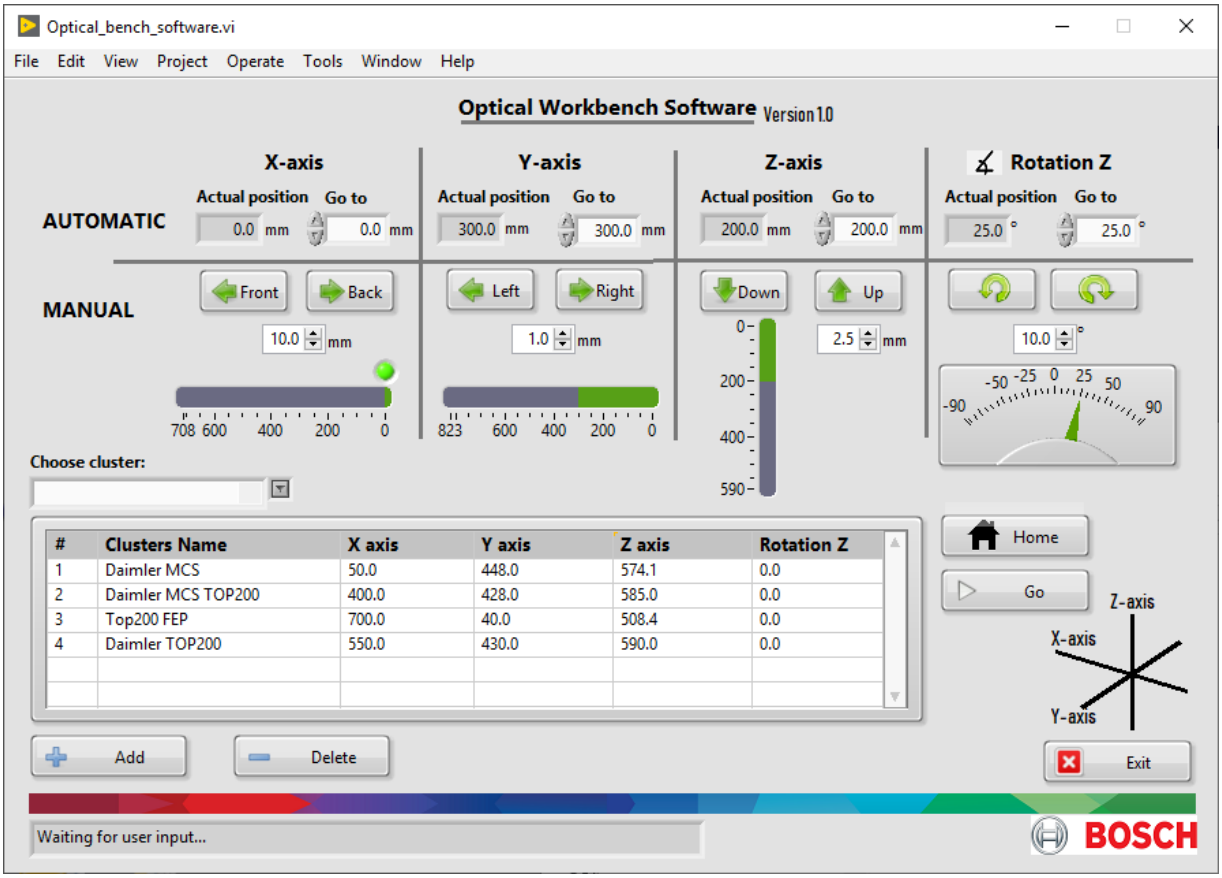



Figure 5.8 - User Interface of Optical Workbench software

At the top of the interface, there are two easily distinguishable modes, the automatic and the manual. In the first one, it is possible to visualize the actual position of each axis, and the user can select the desired absolute coordinates in the textbox “Go to”. After that, he can simply select the “Go” button, and each axis will move in accordance with the desired coordinates selected. On the other hand, the manual mode allows the individual movement of each axis in a relative way. Only by selecting one of the direction buttons, the trolley associated will move at the pre-selected distance. Although by default this distance is 10 mm, it can be adjusted at any time. The direction of the axes is identified in relation to the user who is in the computer of the optical laboratory, in order to be quite distinguishable to associate the real direction of each trolley movement.

At the same time, the interface provides constant information of the current position of each trolley using animated sliders, being the horizontal axes demonstrated by horizontal slide bar, the vertical axis by a vertical slide bar and the rotational axis by an indicator gauge. If at any time the trolley is in one of the extremities of its axis or actuating one of the photoelectric detectors, a small led will appear next to the slide bar to show this situation. It can be seen in Figure 5.8 that the x-axis trolley is in the zero coordinate.

Another actuation way for the workbench is by using the saved positions of the table. Whenever the user wishes to use the same workbench position in the future, he can select the "Add" button. A pop-up will appear and will ask to enter the name of the digital instrument cluster, and, after filling it without a repeated name or an empty string, a new row will appear in the table relative to this new saved position. From that moment, that position will always be quickly available for any user of the software, just by selecting it in the combo box "Choose cluster" and then pressing the "Go" button. To the right of the "Add" button is the "Delete" button, which allows the elimination of one of the table positions, being only necessary to select the unwanted position in the new pop-up that will appear.

Although the program always initializes with the homing function, which allows the software to recognize the exact position of each of the workbench axes, at any time the user can also force it to happen again, just by selecting the "Home" button.

In the lower-left corner of the interface, there is a small text box that provides useful information to the user, as indications about the current application state.

Lastly, in the lower-right corner of the screen, the "Exit" button properly closes the application.

## 5.5 Concluding Remarks

This chapter presents the main features of the software developed to control the four axes (three linear and one angular) of the optical measurement workbench.

By analysing the architecture of the software in a general way, it is possible to see that it is the LINX library which allows the communication between LabVIEW and the Arduino board. Custom commands were made to improve this communication.

Regarding the programming logic, it was developed, based on the state machine technique and consists of 13 different states. The diagram which relates all of them has been made available.

The code block and the final interface of the application are lastly shown, where it is possible to see the three possible modes of the workbench actuation.

# 6 Results and Performance Tests

From the project design and implementation, explained in the course of the previous chapters, this one examines its outcome as well as find practical ways to verify if the requirements have been achieved.

Regarding the overall objective, the implementation of a transmission system to automate the optical test workbench, it has been accomplished. In addition to the automation and consequent implementation of transmission mechanisms in the three linear axes, a fourth angular axis, which turns the z-axis, has also been developed. All these axes are now controllable through a software, majority programmed in LabVIEW, which communicates with the optical test workbench through an Arduino board.

Even taking into account that the main objective has been achieved, throughout this chapter a set of tests will be described to demonstrate how the whole system works, the reliability of the process as a positioning mechanism and the achievement of the specific requirements in terms of speed and noise.

## 6.1 Software Functionalities and System Features

This section will describe a set of functional tests of the application where a group of commands is provided to the system and its results are evaluated. At each command, or set of commands, provided are described both the response originated on the optical test workbench and the response of the interface itself to that input.

**Command 1**: Initialize the programme.

**Response**: The user interface shows “Initializing...” in the information text box. After a moment, the message that appears is “Homing...” accompanied by a movement of the workbench trolleys until the respective limit switch is actuated. When all the movements cease, the “actual position” variables of all the axes are actualized to the real values. Consequently, the gauge indicator and all the slide bars are also actualized. As the trolleys are in reference positions, a virtual LED over the slide bars appears to indicate what detector is actuated in each axe. The software also shows the information “Waiting for user input...” in the text box as seen in Figure 5.8.

**Command 2**: Initialize the programme with the emergency button pressed.

**Response**: The user interface shows “Initializing...” in the information text box, but quickly the program shows a pop-up, Figure 6.1, and after it is turned off.

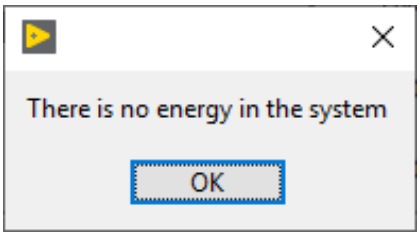



Figure 6.1 - Pop up result of command 2 and 3

**Command 3:** Initialize the programme with the power supply turned off.

**Response:** The same result of command 2 is obtained.

**Command 4:** Enter absolute coordinates in all axes and press the “Go” button.

**Response:** “Going to desired positions...” appears in the informative text box. Meanwhile, the axes are moving to the respective positions. When the movement is over, both the “actual position” variables and the slide bars or indicator gauge of each axis are updated. The software starts informing again “Waiting for user input...” in the information text box.

**Command 5:** Enter one absolute coordinate that overpass the axe limit and then press the “Go” button.

**Response:** The interface does not accept values above the capabilities of each axis. By inputting large absolute coordinates, these are automatically transformed into the maximum value to which that axis can move. If the “Go” button is then selected, the entire Go process is typically performed with this axis moving to its final extremity.

**Command 6:** Simulate a poor workbench calibration by manually moving one of the trolleys forward. Then enter its maximum coordinate and select the “Go” button.

**Response:** The axle, although poorly calibrated, moves to its maximum position, as in the “Go” process. Once it reaches its limit position, the detector will actuate, which makes to stop the entire workbench movement. In the user interface a pop-up window appears explaining the situation and the user can choose to either perform the “Homing” in order to calibrate the workbench again or exit to abandon the program (Figure 6.2).

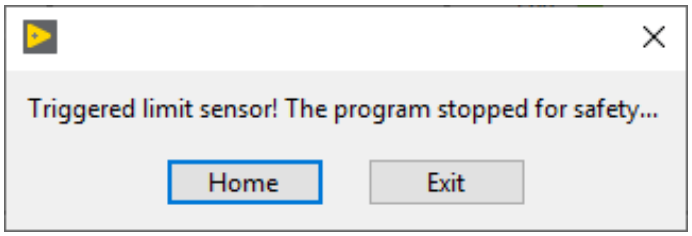



Figure 6.2 - Pop up result of command 6

**Command 7:** Pre-select a distance in the Manual mode and press one of the direction buttons relative to one of the axles

**Response:** The axis moves in the selected direction in the preselected distance. During the movement the interface informs "Going to desired positions..." and at the end of the movement it informs again "Waiting for user input...". It is also confirmed that both the "actual position" variables and the graphical position indicators are updated to this new axis position.

**Command 8:** Having an axis close to one of its limit positions, preselect a large distance and select the direction to make it move towards its nearest limit position.

**Response:** The "Manual" process is normally performed. However, as the program knows that the preselected distance is greater than the trolley can move, it only goes to its limit position. In that case, a pop-up window will appear to explain it (Figure 6.3).

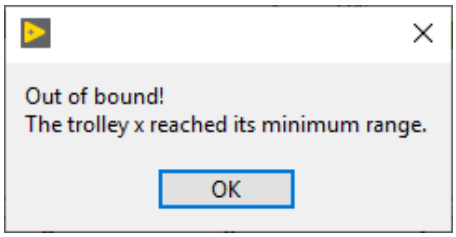



Figure 6.3 - Pop up result of command 8

**Command 9:** Simulate a poor workbench calibration by manually moving one of the trolleys backwards to near the limit position. Then preselect a distance and press the direction backwards.

**Response:** The "Manual" process is normally performed until the limit sensor actuates. Then a pop-up window appears (Figure 6.2) in the interface explaining that one of the limit detectors was triggered. The user can choose to either perform the "Homing" in order to calibrate the workbench again or exit and abandon the program.

**Command 10:** Press the "Home" button

**Response:** The message "Homing..." appears in the text box, each axle starts moving simultaneously backwards and only ends until one of the detectors on that same axle is actuated. When all the movements cease, the "actual position" variables of all the axes are actualized to the real values, as well as all the graphical positions indicators. As the trolleys are in reference

positions, a virtual LED over the slide bars appears to indicate what detector is actuated in each axe. The software starts informing “Waiting for user input...”.

**Command 11:** Press the “Home” button when one of the axes is already actuating one of the detectors.

**Response:** The axle moves forward until the detector stops acting. Then, the normal Homing process is automatically performed.

**Command 12:** Press the emergency button when the workbench is in motion

**Response:** The workbench stops immediately, and the computer application also goes down, showing the pop-up “There system power was cut” (Figure 6.4).

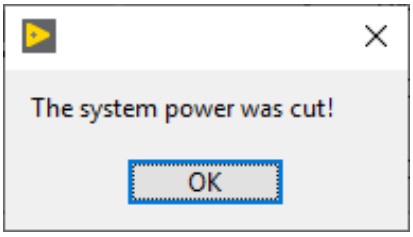



Figure 6.4 - Pop up result of command 12 and 13

**Command 13:** Turn off the power supply when the workbench is in motion, simulating a power failure.

**Response:** The same result of command 12 is obtained.

**Command 14:** Press “Add” button.

**Response:** A pop-up window appears asking the name of the cluster pretended to measure in that position (Figure 6.5). After input a not repeated name, a new position in table appears with the given cluster name and the actual position of all the workbench axes.

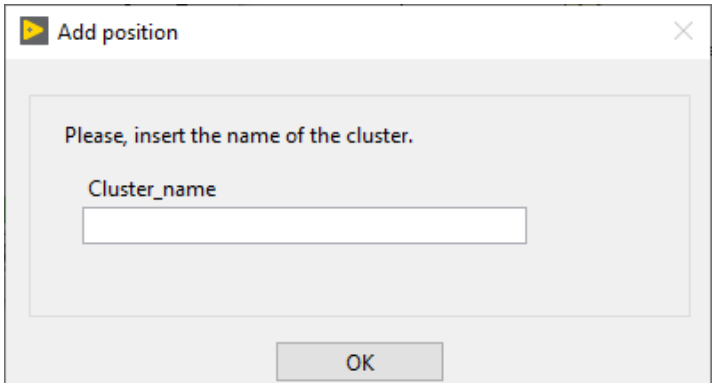



Figure 6.5 - Pop up result of command 14

**Command 15:** Press “Add” button and input a repeated position name.

**Response:** After inputting a repeated position name, a new pop-up indicates in the software appear that the position name already exists and to introduce another name (Figure 6.6). The user can select “Ok” to introduce another name, or “Cancel” to do not save any position.

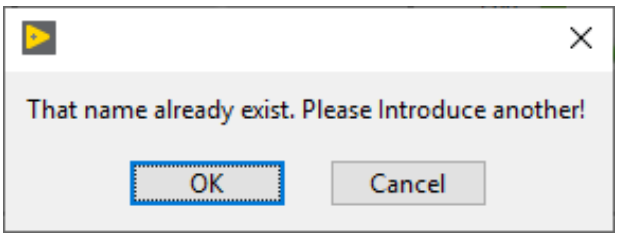



Figure 6.6 - Pop up result of command 15

**Command 16:** Press the “Add” button and input an empty string in the position name.

**Response:** A new pop-up appears saying to “please insert something” (Figure 6.7). After the “Ok” from the user, the interface asks a name for the position again.

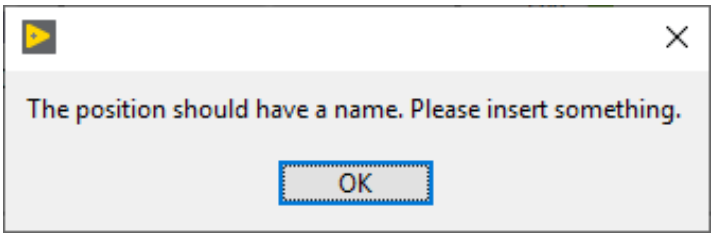



Figure 6.7 - Pop up result of command 16

**Command 17:** Select one cluster of the combo box “Choose cluster”.

**Response:** The coordinates of that position appear automatically in the variables “Go to” of the respective axis, and then, after press in the “Go” button, the Go process is normally performed.

**Command 18:** Select the button “Delete”.

**Response:** A pop-up window appears (Figure 6.8), asking to choose one of the clusters whose position is saved in the table. After choosing that position is eliminated

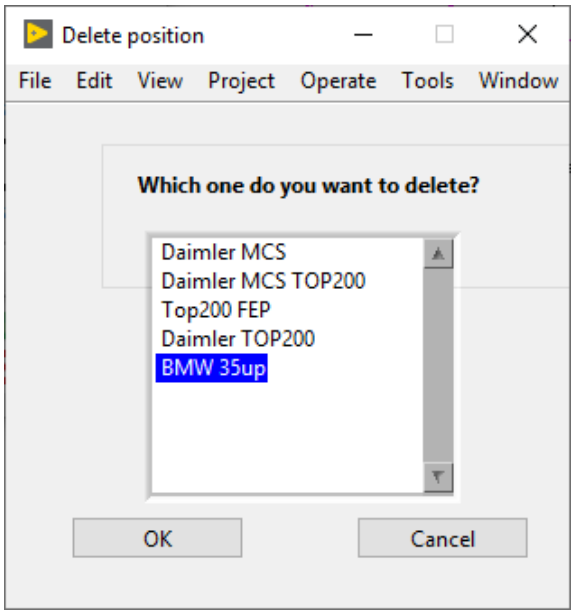



Figure 6.8 - Pop up result of command 18

## 6.2 Positioning Time

Since the functionalities of the program are explained and the workbench is corresponding to what is requested, its performance in relation to its positioning speed will now be evaluated in this section.

One of the initial requirements of the workbench was to perform an automatic positioning of the light measurement device under 30 seconds, which is a fast time considering the positioning time before this project. To find out if this requirement has been met, it was measured the time each trolley takes to travel between the extremities of its axis. The times for each axle have been recorded and are visible in Table 15. Even if the servo motor is not properly coupled to the workbench, the time it takes to perform the movement that causes the rotation of the entire range of the rotation angle z has also been measured.

Table 15 - Travel time of each axis

| | Travel time (seconds) |
|---|---|
| X-axis | 27.9 |
| Y-axis | 31.2 |
| Z-axis | 29.3 |
| Rotation Z | 15.2 |

Although y-axis slightly exceeds 30 seconds, it was considered among the optical lab engineers very acceptable time values because it is also often necessary to rearrange the settings on the programs which will perform the optical measurements on the digital instrument cluster. By first commanding the movement of the measuring equipment to the desired location, the user can take advantage of this positioning time to realize the configuration in the equipment

program, not necessarily have to wait for the workbench movement. As before this project, the workbench only could move manually, the user could not take advantage of this time to perform other tasks simultaneously.

The workbench is therefore considered to meet its requirements in terms of positioning time.

## 6.3 Noise

Since the optical laboratory is located inside the Bosch development centre, far from the production area, it was necessary to pay attention to the noise of the machine to make sure that the regular work of the engineers is not disturbed.

In order to be able to evaluate it with certain reliability, it was used the "Decibel meter" android application to determinate the average noise inside the optical laboratory, with and without the optical workbench in motion. It was measured hourly over a typical working day, and it was calculated that the optical laboratory has an average noise of 53.3 decibels. Realizing the same measurement in identical situations but with the workbench in motion, the average noise measured was 56.6 decibels.

However, in order to truly calculate the noise that the machine can cause among the engineers in the development centre, it was decided to perform the same experiment but this time outside the optical laboratory, near its entrance, as this is where the machine could be more audible. There, it was an average noise of 51.1 decibels over a typical working day, without the optical workbench in motion. Recording the noise values at identical times of the day, but this time with the workbench working, the average value recorded was 51.4 decibels.

With the workbench in operation, it is registered a minimal increase in the noise. The result was expected since the workbench is almost inaudible outside of the optical laboratory. Considering that the measurements were realized in the area of the development centre where the noise could be most audible, and that the measuring device used is not the most reliable, this small difference in the values obtained is considered negligible and that it does not disturb the normal working of the development centre engineers.

## 6.4 Positioning Performance

Since all workbench features are working correctly as well as the noise and the travel times are within the requirements, this section will expose the tests performed in order to evaluate the positioning performance of the workbench trolleys. It was mostly concerned with its repeatability, resolution and accuracy.

Since the motor of the z-axis rotation was not assembled in its proper place to move the angular axis, only the results for the linear axes are shown throughout this section.

### *6.4.1 Repeatability*

One of the workbench automation objectives was to obtain repeatability, which is the ability to generate similar results for multiple preparations of the same sample [34], of less than 0.5 mm on all the axes.

In an attempt to evaluate if the requirement is met, a gauge digital dial indicator with a resolution of one-hundredth of a millimetre was used. The test consisted of fixing it at the end of each axis and evaluating its unidirectional repeatability, using the followed process for all the linear axis:

1. Shift to coordinate 0 of the axis.
2. Move the trolley to a fixed point near the other end of the axis, in order to act the gauge digital dial indicator. Record the value displayed on the instrument.
3. Repeat points 1 and 2 until a certain number of samples are collected.

The number of samples collected in order to calculate the repeatability of positioning was ten because it is an average number of different positions that the workbench has over a typical working day.

Performing the method explained, the mean and standard deviation of the records obtained on the X, Y and Z axes were those presented in Table 16.

Table 16 - Repeatability test (going back to 0 by coordinates)

| | X-axis | Y-axis | Z-axis |
|---|---|---|---|
| Mean (mm) | 0.001 | 0.000 | 0.001 |
| Standard deviation (mm) | 0.003 | 0.000 | 0.003 |

The results obtained seemed acceptable since the various samples performed on each axis have only minimal differences between them, a much better value than the 0.5 mm intended. However, it was decided to carry out the same test with a different particularity: in point 1, instead of moving to the 0 coordinate, it was used the help of the “homing” function, where the workbench recognizes its reference position (coordinate 0) through the photoelectric detector. Using this method, the values recorded over the ten samples, present in Table 17, were not as satisfactory as the previous ones.

Table 17 - Repeatability test (going back to 0 by homing)

| | X-axis | Y-axis | Z-axis |
|---|---|---|---|
| Mean (mm) | 0.003 | 0.006 | 0.002 |
| Standard deviation (mm) | 0.008 | 0.11 | 0.007 |

The worst repeatability recorded in Table 17 is originated from the “homing” function, suspecting that the definition of the 0 axis coordinate is associated with significant repeatability errors. It was also possible to experimentally verify that the differences found depended on the speed at which the trolley retreated during the “homing” function. At higher speeds, a higher

value was recorded in the digital gauge, which meant that the trolley stopped a little beyond the photoelectric detector. In order to avoid these positioning variations when defining the 0 coordinate, it was decided to change the software by assigning a lower speed of 1300 pulses per second in the homing function. This way, the repeatability test through the "homing" function was performed again. The results are shown in Table 18.

Table 18 - Repeatability test (going back to 0 by homing), after slowing down the "homing" velocity

| | X-axis | Y-axis | Z-axis |
|---|---|---|---|
| Mean (mm) | 0.000 | 0.000 | 0.002 |
| Standard deviation (mm) | 0.004 | 0.007 | 0.006 |

It can be seen that with this speed decrease during the homing movement, the repeatability values obtained in Table 18 are more satisfactory compared with the values of Table 17. Contemplating both the repeatability tests with the 0 coordinate movements performed by inputting coordinates (Table 16) and through the "homing" function (Table 18), it can be concluded that the workbench repeatability met its requirement.

### *6.4.2 Resolution*

Once the workbench's repeatability has been calculated, the same instrument was used to verify the mechanical resolution of each axis.

The procedure was to command the trolleys successively for smaller distances, evaluate in the measuring instrument if these movements were recorded and obtain the minimum possible increment experimentally to command in each axis. Through this procedure, it was found that each axis begins to show less constant behaviour when increments of one-hundredth of a millimetre are added. Values higher than this are always confirmed on the digital gauge display.

Consequently, as 2 hundredths of a millimetre are the smallest value to which the various axes respond correctly, this is the resolution of the linear workbench axes, which completely meets the requirements.

However, despite having this resolution, the final software programmatically prevents the introduction of such small values, as it only supports values with one decimal place as it is enough for the purpose.

Summarizing, although it is possible to make smaller increments, the resolution was limited in the software to 0.1 mm.

### *6.4.3 Accuracy*

Regarding the positioning accuracy, and taking into account the motors work in open-loop control without any feedback, there is no encoders that could accurately measure the distance of the trolley's movement. As no sensor was available to perform this measurement, the travel distance was not confirmed.

# 7 Final Considerations

## 7.1 Conclusions

In the automotive industry, all screen manufacturing, such as digital instrument clusters or infotainment screens, have to pass rigorous performance tests. Characteristics as luminance, colour, spatial uniformity, and angular distribution are evaluated, which allow detecting any failures that would jeopardise the display quality or the driver's difficulty in viewing the screen. These tests are performed on workbenches, which allow the movement of the Light Measurement Devices (LMD's) in relation to the display to be tested.

The workbench presents in the development centre of Bosch Car Multimedia has support to LMD's, but all its movements were performed manually. In addition to the slow movement of the measuring instrument, the positioning in each axis was not very reliable. In this context, the end goal of this project was to develop a command system for that manual optical workbench in the LabVIEW software. This goal presupposed the increase of the movement velocity, consequently reducing the time between the different optical tests, and the raise of the LMD's positioning reliability, which in turn also improves the quality of the measurements.

In the first stage of the work, extensive research was carried out on the commercial workbenches that perform the same type of support for optical instruments. While this research was taking place, there was a need to know a little better about the optical tests that were intended to be performed and their operating conditions, requirements and positioning specifications.

In a second phase, a previous study of the workbench was carried out, all its characteristics were verified, and the possible ways of controlling its linear axes were considered. Among the options, the choice prevailed in the use of stepper motors. They are easy to implement, even in open-loop control they provide a better positioning than the required in the measurement tests specifications and are the most economical option.

After studying the speed and torque that the motors should provide, it was verified that the bipolar stepper motor available in the optical laboratory was compatible with what was required for one of the workbench axles. The other two motors were ordered as well as the micro-step drivers that would command them. This type of drivers was chosen because it allows a smoother control of the movement, eliminating vibrations and consecutively the noise, a relevant requirement. On the other hand, they require digital pulses at a higher frequency so that the motor performs at the same speed.

Furthermore, one hardware was needed to materialise the LabVIEW code into electrical impulses which in turn are going to command the micro-step drivers. Since the Arduino Mega 2560 was already available in the optical laboratory, this was the chosen option. For the complete control of the workbench movement, position limit switches and an emergency stop button were ordered. Regarding the transmission between the motors shaft and the ball screws shaft, it was preferred to use gears on the horizontal axes not to increase the workbench dimensions and, consequently, not reduce the space available inside the optical laboratory.

After, it was realized the electrical connections between these devices and the installation of the limit detectors, motors and the emergency stop button. Protective chain cable carriers have also been installed on the workbench trolleys to conduct the electrical cables safely.

In addition to implementing and developing a transmission system for the linear axes, another transmission system was also designed to allow the angular movement around the vertical axis. For this task, it was used a previously bought motor, and it was designed its fixation to the workbench and the movement transmission to the LMD support. Unfortunately, it was not possible to obtain part of these pieces in time, which resulted in an incomplete assembly of this axle. However, the motor was left adequately installed and controlled by the software. As a consequence, the axis can be easily assembled as soon as the missing parts are produced.

Regarding the command programme for the four axes of the workbench, this was carried out in the LabVIEW software, programmed by implemented state machine. It is through the LINX library that the program communicates with the Arduino board, which in turn emits and receives the electrical signals from the optical workbench. Due to the firmware change and later download on the board, it was possible the use of custom commands. They solved the problem of slow communication between Arduino and LabVIEW since with the custom commands it was possible to perform the workbench movements through functions that were already downloaded on Arduino's board and only needed to be called.

Thus, the library "AccelStepper" was used in Arduino's IDE for programming these movements. This library facilitates the correct programming of the desired movement accelerations, but with the cost of some calculation power, which limits the maximum frequency of Arduino's digital output signal to 4000 Hz. At this frequency and for the desired velocities of the axes, it was necessary to adjust the micro-step driver to 400 steps per revolution, not taking advantage of the device's full capacity.

Regarding the programme, apart from allowing position command through absolute coordinates, it also has a "manual mode" which allows a relative movement of each axis individually. Besides, it allows the registration of positions in a list, which not only increases the speed of typing in the programme but also makes it possible to standardise the positions of each test. The interface of the program has been realised and progressively developed based on the feedback received by the optical laboratory engineers, in order to make it increasingly intuitive and with an appealing design.

Currently, the programme is already implemented in the optical laboratory and is already operating the workbench axes. It has been verified that all the functionalities for which it has been designed are working as well as responding safely to possible power failure and emergency stop button actuation.

The automation of this workbench has given a significant increase in positioning velocity. Whereas previously the user would have had to turn the crank of each linear axis a few dozen times, which was a high time-consuming process, now it only needs to enter the desired coordinates into the interface. The axes will move into the chosen position in a maximum time of 31.2 seconds, time which can be used by the user to prepare the next optical measurement. It means that the user may not even have to wait for the trolley's movement, which, considerably increases the optical measurements that the workbench can perform during a working day.

It was decided to program the movements with variable acceleration, which resulted in much smoother starts and stops, and the noise caused during the movement of the trolleys was not considered excessive.

Regarding the positioning reliability, a set of tests were carried out to evaluate characteristics such as repeatability, resolution, and accuracy. The first two characteristics proved to be quite acceptable concerning what was required. Concerning the accuracy, it was not possible to truly assess it, as an instrument that could measure it along the axes was not available. However, given the excellent repeatability and resolution values, the good theoretical capability of stepper motors and the accuracy tests which have proven good values albeit only at shorter travel distances, it is expected that this positioning parameter is also within the initially required.

Summarising, this automation project and all the improvements associated with it have enabled this workbench to achieve a higher performance, which in a way are not much lower than those demonstrated by the robotic workbenches. Although the levels of positioning reliability are still lower as well as speed, this workbench, taking advantage of resources already available and with one very intuitive software, allows performing almost the same types of measurements since it meets sufficient positioning requirements. Thus, this optical measurement workbench not only improves the quality of its measurements but also enables a higher volume of tests to be performed in the optical laboratory, which is an added value for the laboratory itself, the ECM3 team and the Bosch Car Multimedia development centre.

## 7.2 Future Work

Once the workbench automation has been completed, all the conditions are met for a more detailed analysis of passive improvement situations and the integration of the machine into superior systems of the automation.

First of all, the small tasks left unfinished in this project should be completed appropriately. By this, it is meant the final assembly in the rotary axis, where the transmission gears and the motor support part can be fixed as soon as possible. Indications have been left so that this task runs effortless. On the other hand, the movement accuracy along the workbench axes must be fully characterized to prove indeed that this machine meets all the requirements of the measurement tests. Both a draw-wire encoder and a laser distance sensor are devices that would allow the evaluation of this parameter in a fast and intuitive way.

Secondly, despite being less used than the others, the last degree of freedom of the workbench, which allows the rotation on the y-axis, could also be automated and incorporated into the software.

In order to take advantage of the increased capacity of the micro-step drivers, the Arduino board could be replaced by another board that can generate electrical pulses at much higher frequencies. It would allow the use of the micro-step driver up to resolutions of 3200 steps per revolution. Smoother movement, less vibration and a more pleasant workbench noise would be the practical result of this change.

Lastly, with the prospect of integrating the system in an Industry 4.0 environment, communication between the workbench positioning LabVIEW program and the measurement software could be envisaged. Only one specific measurement command would be required and, with access to a database, the workbench would automatically know where its axes should move. The report could also be automatic performed, recording the axis positions and the results of the executed measure.

# 8. References

[1] Robert Bosch GmbH, Invented for life, Bosch Global, 2020. [Online]. Available: https: //www.bosch.com/ [Accessed Mar. 20, 2020].

[2] J. Pešić, K. Omerović, I. Nikolić, and M. Z. Bjelica, "Automotive cluster graphics: current approaches and possibilities," IEEE 6th International Conference on Consumer Electronics, vol. 2016-Octob, pp. 12-14, Oct 2016. [Online]. doi: 10.1109/ICCE-Berlin.2016.7684705 [Accessed Apr 20, 2020].

[3] K. Blankenbach, "Advanced automotive display measurements: selected challenges and solutions," Journal of the Society for Information Display, vol. 26, no. 9, pp. 517-525, Sep 2018. [Online]. doi: 10.1002/jsid.625 [Accessed Apr 24, 2020].

[4] F. Leccese, G. Salvadori, and M. Rocca, "Visual ergonomics of video-display-terminal workstations: field measurements of luminance for various display settings," Displays, vol. 42, no. 10.2016, pp. 9-18, Apr 2016. [Online]. doi: 10.1016/j.displa.2016.02.001 [Accessed Apr 24, 2020]

[5] P. M. Knoll, "Automotive Displays," in Handbook of Visual Display Technology, J. Chen, W. Cranton, and M. Fihn, Ed. Berlim: Springer, 2016, pp. 231-251.

[6] P. M. Knoll, "The use of displays in automotive applications," Journal of the Society for Information Display, vol. 5, no. 3, pp. 165-172, Sept 1997. [Online]. doi: 10.1889/ 1.1985149. [Accessed Apr 27, 2020]

[7] Robert Bosch GmbH, Digital Instrument Clusters, 2020. [Online]. Available: https://www.bosch-mobility-solutions.com/en/products-and-services/commercial-vehicles/interior-and-body-systems/freely-programmable-instrument-clusters/. [Accessed Apr. 20, 2020].

[8] P. M. Knoll and R. Vollmer, "Car information, communication and entertainment system-advances and new developments," Society of Automotive Engineering Techical Papers, no. 41 2, p. 8, Mar1994. [Online]. doi: 10.4271/941040 [Accessed Apr 24, 2020]

[9] T. Goodman, "Overview of the Photometric Characterization of Visual Displays," in Handbook of Visual Display Technology, J. Chen, W. Cranton, and M. Fihn, Ed. Berlim: Springer, 2016, pp. 337-350.

[10] P. Notermans and N. Cohen, “Next-generation metrology facilitates next-generation displays,” Information Display, vol. 32, no. 6, pp. 24-28, Nov 2016. [Online]. doi: 10.1002/j.2637-496x.2016.tb00950.x [Accessed Apr 24, 2020]

[11] T. Fiske, “Display metrology: the ‘informed conscience’ of the industry,” Information Display, vol. 35, no. 5, pp. 28-31, Sep 2019. [Online]. doi: 10.1002/msid.1062 [Accessed Apr 24, 2020]

[12] K. Blankenbach, “Introduction to Display Metrology,” in Handbook of Visual Display Technology, J. Chen, W. Cranton, and M. Fihn, Ed. Berlim: Springer, 2016, pp. 3073-3085.

[13] K. Blankenbach, “Standards and Test Patterns,” in Handbook of Visual Display Technology, J. Chen, W. Cranton, and M. Fihn, Ed. Berlim: Springer, 2016, pp. 3265-3281.

[14] M. Wolf, M. E. Becker, and J. Neumeier, “Metrological challenges of curved displays,” Society for Information Display Symposium Digest of Technical Papers, vol. 48, no. 1, pp. 463-466, May 2017. [Online]. doi: 10.1002/sdtp.11650 [Accessed Apr 24, 2020]

[15] T. Fiske, “Image Quality and Metrology,” Information Display, vol. 31, no. 5, pp. 12-15, 2015. [Online]. doi: 10.1002/j.2637-496x.2015.tb00840.x [Accessed Apr 24, 2020]

[16] IEC Liquid Crystal and Solid-state Display Devices, “IEC 61747,” 1998.

[17] M. E. Becker, “Imaging Light Measurement Devices,” in Handbook of Visual Display Technology, J. Chen, W. Cranton, and M. Fihn, Ed. Berlim: Springer, 2016, pp. 3285-3312.

[18] K. Blankenbach, “Measurement Devices,” in Handbook of Visual Display Technology, J. Chen, W. Cranton, and M. Fihn, Ed. Berlim: Springer, 2016, pp. 3313-3334.

[19] T. Goodman, “Measurement Instrumentation and Calibration Standards,” in Handbook of Visual Display Technology, J. Chen, W. Cranton, and M. Fihn, Ed. Berlim: Springer, 2016, pp. 323-335.

[20] J. T. Walsh, “The photometer bench,” Journal of Scientific Instruments, vol. 5, no. 7, pp. 231-232, 1928. [Online]. doi: 10.1088/0950-7671/5/7/111 [Accessed Apr 24, 2020]

[21] Gamma Scientific, GS-1290-DMS-RBT Robotic Goniometric Display Measurement System Data sheet. [Pdf]. Available: https://www.gamma-sci.com/sites/default/files/ data-sheet/gs-1290-dms-rbt_rev_7.18.1.pdf. [Accessed May. 11, 2020]

[22] Gamma Scientific, GS-1290-DMS-RBT Robotic Goniometric Display Measurement System, 2020. [Online]. Available: https://www.gamma-sci.com/products/gs-1290-dms-rbt-robotic-goniometric-display-measurement-system. [Accessed May. 11, 2020].

[23] Stäubli International AG., 6 axis industrial robots TX2-60, 2020. [Online]. Available: https://www.staubli.com/en/robotics/product-range/industrial-robots/6-axis-robots/tx2-60/. [Accessed May. 11, 2020]].

[24] MISUMI Corporation, Helical Gears/Pressure Angle 20Deg./Helix Angle 45Deg. [Online].

Available: https://uk.misumi-ec.com/vona2/detail/110302003770/?PNSearch=NEGTM1.5-20-12-R&HissuCode=NEGTM1.5-20-12-R&searchFlow=suggest2products&Keyword=NEGTM1.5-20-12-R. [Accessed Apr. 13, 2020].

[25] R. A. Serway and J. W. Jewett, Physics for scientists and engineers with modern physics, Ninth edit., vol. 34, no. 07. Boston: Mary Finch, Charlie Hartford, 2016.

[26] Engineers Edge, Densities of Metals and Elements Table. [Online]. Available: https://www.engineersedge.com/materials/densities_of_metals_and_elements_table_13976.htm. [Accessed Apr. 8, 2020].

[27] D. Collins, How to calculate motor drive torque for ball screws, 2016. [Online]. Available: https://www.linearmotiontips.com/calculate-motor-drive-torque-ball-screws/. [Accessed Apr. 20, 2020].

[28] Sanmotion-Sanyo Denki Co. Ltd., "2-Phase Stepping Systems, F2," pp. 62, 2013.

[29] A. Junior, "Engrenagens Helicoidais," 2007. [Online]. Available: http://www.fem.unicamp.br/~lafer/em718/arquivos/Engrenagens_Helicoidais.pdf

[30] R.T.A. Group, "Stepping Motor Drives Catalogue," 2012. [Online]. Available: https://www.rta.it/uploads/BSD_eng%203.pdf

[31] R.T.A. Group, "BSD Series Stepping Motor Drives,".[Online]. Available: https://docs.rs-online.com/ecbb/0900766b81585e9f.pdf

[32] L. Panasonic Industrial Devices Sunx Co., "Micro Photoelectric Sensor," Japan, 2015. [Online]. Available: https://mediap.industry.panasonic.eu/assets/download-files/import/ds_pm254565_de.pdf

[33] Doxygen, AccelStepper Class Reference, 2018. [Online]. Available: https://www.airspayce.com/mikem/arduino/AccelStepper/classAccelStepper.html [Accessed Jul. 20, 2020].

[34] A. S. Lister, " Validation of HPLC Methods in Pharmaceutical Analysis," in Handbook of Pharmaceutical Analysis by HPLC, S. Ahuja, M. Dong, Ed. Academic Press, 2005, pp 191-217.

# APPENDIX A: Technical Datasheets

## A.1 Sanyo Denki 2-Phase Stepper Motors [28]

### Stepping Motors

# 42 mm sq. (1.65 inch sq.)

1.8° /step RoHS

Bipolar winding, Lead wire type

Unipolar winding, Connector type ▶p. 61

**Customizing**

Hollow | Shaft modification | Decelerator | Encoder | Brake

Varies depending on the model number and quantity. Contact us for details.

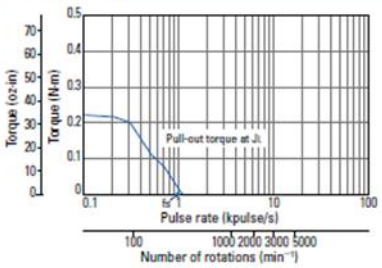
**Bipolar winding, Lead wire type**

| Model number | | Holding torque at 2-phase energization | Rated current | Wiring resistance | Winding inductance | Rotor inertia | Mass (Weight) | Motor length (L) |
|---|---|---|---|---|---|---|---|---|
| Single shaft | Dual shaft | [N·m (oz·in) min.] | A/phase | Ω/phase | mH/phase | [×10⁻⁴kg·m² (oz·in²)] | [kg (lbs)] | mm (in) |
| 103H5205-5040 | 103H5205-5010 | 0.23 (32.57) | 0.25 | 54 | 78 | 0.036 (0.20) | 0.23 (0.51) | 33 (1.25) |
| 103H5205-5140 | 103H5205-5110 | 0.25 (35.40) | 0.5 | 13.4 | 23.4 | 0.036 (0.20) | 0.23 (0.51) | 33 (1.25) |
| 103H5205-5240 | 103H5205-5210 | 0.265 (37.53) | 1 | 3.4 | 6.5 | 0.036 (0.20) | 0.23 (0.51) | 33 (1.25) |
| 103H5208-5040 | 103H5208-5010 | 0.35 (49.56) | 0.25 | 66 | 116 | 0.056 (0.31) | 0.29 (0.64) | 39 (1.54) |
| 103H5208-5140 | 103H5208-5110 | 0.38 (53.81) | 0.5 | 16.5 | 34 | 0.056 (0.31) | 0.29 (0.64) | 39 (1.54) |
| 103H5208-5240 | 103H5208-5210 | 0.39 (55.23) | 1 | 4.1 | 9.5 | 0.056 (0.31) | 0.29 (0.64) | 39 (1.54) |
| 103H5209-5040 | 103H5209-5010 | 0.38 (53.81) | 0.25 | 71.4 | 133 | 0.062 (0.34) | 0.31 (0.68) | 41 (1.61) |
| 103H5209-5140 | 103H5209-5110 | 0.41 (58.06) | 0.5 | 18.2 | 39 | 0.062 (0.34) | 0.31 (0.68) | 41 (1.61) |
| 103H5209-5240 | 103H5209-5210 | 0.425 (60.18) | 1 | 4.4 | 11 | 0.062 (0.34) | 0.31 (0.68) | 41 (1.61) |
| 103H5210-5040 | 103H5210-5010 | 0.465 (65.85) | 0.25 | 80 | 123.3 | 0.074 (0.40) | 0.37 (0.82) | 48 (1.89) |
| 103H5210-5140 | 103H5210-5110 | 0.49 (69.39) | 0.5 | 20 | 35 | 0.074 (0.40) | 0.37 (0.82) | 48 (1.89) |
| 103H5210-5240 | 103H5210-5210 | 0.51 (72.22) | 1 | 4.8 | 9.5 | 0.074 (0.40) | 0.37 (0.82) | 48 (1.89) |

### Characteristics diagram

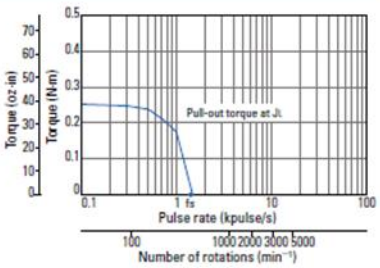
**103H5205-5040**
**103H5205-5010**

Constant current circuit
Source voltage: 24 VDC
Operating current:
0.25 A/phase, 2-phase energization (full-step)
$J_L$=[0.94 × 10⁻⁴kg·m² (5.14 oz·in²) use the rubber coupling]
fs: Maximum self-start frequency when not loaded



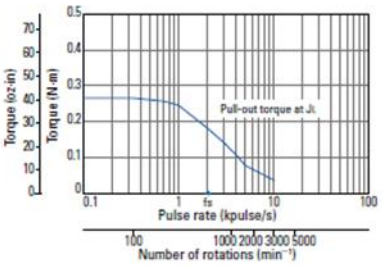

**103H5205-5140**
**103H5205-5110**

Constant current circuit
Source voltage: 24 VDC
Operating current:
0.5 A/phase, 2-phase energization (full-step)
$J_L$=[0.94 × 10⁻⁴kg·m² (5.14 oz·in²) use the rubber coupling]
fs: Maximum self-start frequency when not loaded



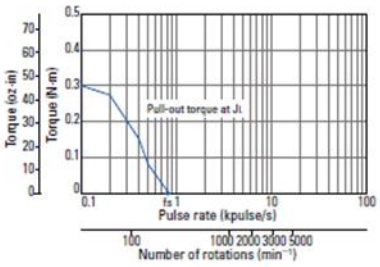

**103H5205-5240**
**103H5205-5210**

Constant current circuit
Source voltage: 24 VDC
Operating current:
1 A/phase, 2-phase energization (full-step)
$J_L$=[0.94 × 10⁻⁴kg·m² (5.14 oz·in²) use the rubber coupling]
fs: Maximum self-start frequency when not loaded



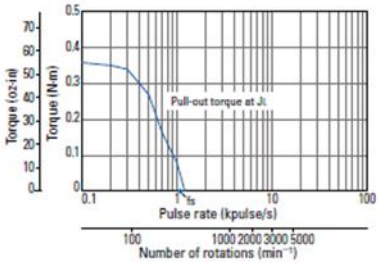

**103H5208-5040**
**103H5208-5010**

Constant current circuit
Source voltage: 24 VDC
Operating current:
0.25 A/phase, 2-phase energization (full-step)
$J_L$=[0.94 × 10⁻⁴kg·m² (5.14 oz·in²) use the rubber coupling]
fs: Maximum self-start frequency when not loaded



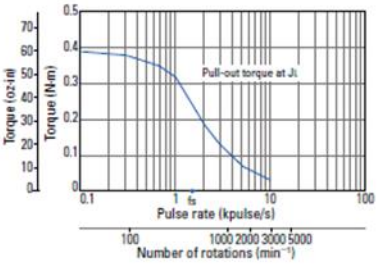

**103H5208-5140**
**103H5208-5110**

Constant current circuit
Source voltage: 24 VDC
Operating current:
0.5 A/phase, 2-phase energization (full-step)
$J_L$=[0.94 × 10⁻⁴kg·m² (5.14 oz·in²) use the rubber coupling]
fs: Maximum self-start frequency when not loaded




**103H5208-5240**
**103H5208-5210**

Constant current circuit
Source voltage: 24 VDC
Operating current:
1 A/phase, 2-phase energization (full-step)
$J_L$=[0.94 × 10⁻⁴kg·m² (5.14 oz·in²) use the rubber coupling]
fs: Maximum self-start frequency when not loaded



# A.2 BSD 02.V Stepper Motor Driver [31]

## 1. GENERAL CHARACTERISTICS

| | | BSD 02 - BSD 02.V |
|---|---|---|
| $V_{DC}$ with stabilized supply (+/- 5%) | (V) | from 22 to 50* |
| $V_{DC}$ with unstabilized supply (+/- 20%) | (V) | from 24 to 45 |
| $I_{NP}$ min | (A) | 0.7 |
| $I_{NP}$ max | (A) | 2.2 |
| Dimensions | (mm) | 78 × 68 × 21 |
| Operating temperature | | from + 5°C to + 45°C (see point 7.3) |

Table 1

**Terms definition in Table 1**

- **$V_{DC}$** Nominal value of DC voltage supply (range) at which the drive can operate.
- **$I_{NP}$** Nominal phase current (peak value) which flows in each motor winding, measurable with motor turning at low speed (see Table 2). Automatic current reduction at motor standstill is 50% of value set.
- **$I_{NP}$ min and max** Minimum and maximum value of nominal phase current setting using DIP-SWITCH.

**(*) NOTE:** Never use any power supply voltage higher than allowed one.

## 2. LOGIC INPUT AND OUTPUT SIGNALS (AM3 Connector, see Fig. 1, 2, 3, 4 and chap. 7.2)

**2- CURRENT OFF INPUT:** When this signal is HIGH, the drive is active. When it is LOW, the drive is inhibited, thus motor current (and so holding torque) is turned to zero.

**3- STEP INPUT:** Step is performed on HIGH to LOW transition of this signal. Suggested duty-cycle: 50%. Max. frequency: 60 KHz with square wave signal supplied from a logic output at 5 volt. With duty cycle different from 50%, STEP signal half period has to be longer than 8 µsec.

**4- DIRECTION INPUT:** With this signal HIGH motor rotation direction is opposite to the one obtained with input LOW. This signal has to be valid at least 100 µs before STEP signal and has to stay in this state for at least 100 µs after last step sent to the drive.

**5- DRIVER FAULT OUTPUT:** When drive is normally working, this output is SHORTED to GND; when drive is in no-working state, the output is OPEN. The drive automatically goes in no-working state when some protection is active and automatically recovers when the protection resets.

**1 and 6- INTERNAL GND:** The terminals are internally connected between each others and to terminals 10 (the power supply common), 11 and 16. They can be used to connect the shield of logic signal cable (this is mandatory or useful depending on type of control system).

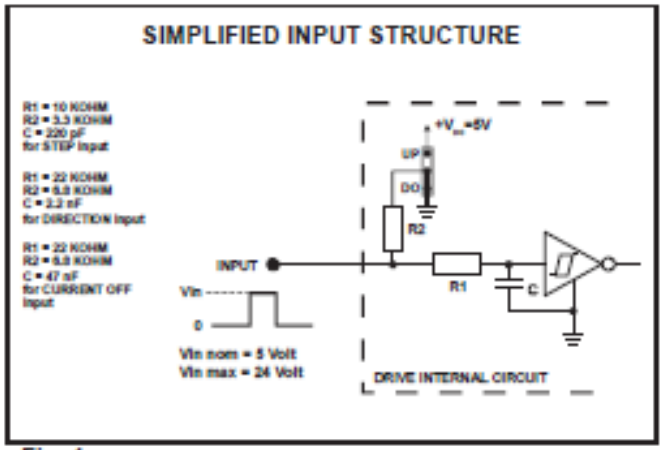



Fig. 1 a

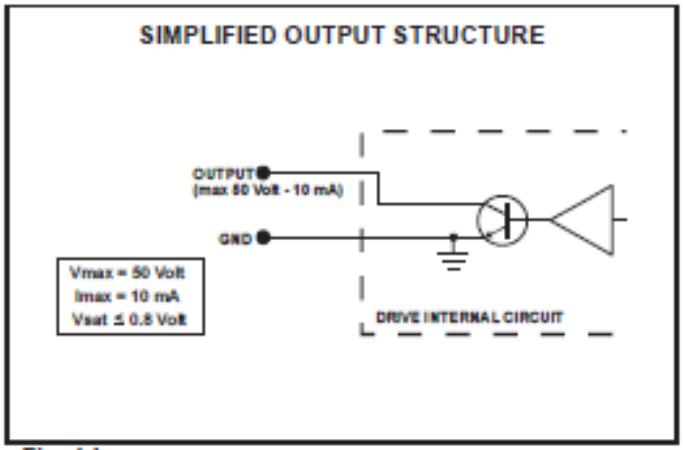



Fig. 1 b

### 3. POWER INPUTS AND OUTPUTS (AM1, AM2 Connectors; see Fig. 2 and 3)

**9** - Power supply positive pole. (+ $V_{DC\ nom}$).
**10** - Power supply negative pole. (- $V_{DC\ nom}$).
**11** - GROUND; connect to Protective Earth terminal (PE).
**12** - Motor winding terminal **A**.
**13** - Motor winding terminal **A-**.
**14** - Motor winding terminal B-.
**15** - Motor winding terminal B.
**16** - Connection point for shield of motor shielded cable.

### 4. MAIN SETTINGS (Dip switch and jumpers).

Main setting must be done by the user at the installation or replacement time, according to motor type and working condition. Wrong setting could damage drive or motor and cause a not correct work of the motor.

| | NOMINAL CURRENT $I_{NP}$ (A) | | |
|---|---|---|---|
| | Dip 1 | Dip 2 | BSD 02 - BSD 02.V |
| ► | ON | ON | 0.7 |
| | ON | OFF | 1.1 |
| | OFF | ON | 1.6 |
| | OFF | OFF | 2.2 |

Table 2

| | LOGIC INPUTS OPERATION MODE | | |
|---|---|---|---|
| | JUMPERS SETTING | | |
| ► | UP | ON | PULL-UP |
| | DOWN | OFF | |
| | UP | OFF | PULL-DOWN |
| | DOWN | ON | |

Table 3

| | RISOLUZIONE | | |
|---|---|---|---|
| | Dip 3 | Dip 4 | PASSI PER GIRO |
| | | | BSD 02 - BSD 02.V |
| | ON | ON | 3.200 |
| ► | ON | OFF | 1.600 |
| | OFF | ON | 800 |
| | OFF | OFF | 400 |

Tabella 4

► = *Default settings*.

**NOTE:** It is suggested to use maximum allowed resolution to reduce acoustic noise and vibrations according with the specific application.

### 5. LED DRIVE STATUS:

**LED HV green:** **ON** = supply voltage higher than minimum allowed value.
**OFF** = supply voltage lower than minimum allowed value.
**BLINKING** = Short circuit or wrong connection at motor output.

### 6. MOTOR – DRIVE COUPLING

BSD series drives can be used with the following Sanyo Denki stepping motors or similar. In case of need to use motors with different features

| MOTOR CODE | CURRENT SETTING (A) | MOTOR CODE | CURRENT SETTING (A) |
|---|---|---|---|
| 103H7823-0740 | 2.2 | 103-H5208-0483 | 1.1 |
| 103-H7126-0740 | 2.2 | 103-H5210-4240 | 1.1 |
| 103-H7123-0740 | 2.2 | 103-547-52500 | 0.7 |
| 103-H7123-5040 | 2.2 | | |

Table 5

In the typical application conditions, for each setting shown in Table 5, the maximum allowed motion duty-cycle is as follows:
- 100% with 24 VDC
- 50% with 48 VDC

If other motors are used ensure drive current is between 0.7 and 2.2A, Voltage is 24 to 48VDC, inductance 1 to 12mH

# APPENDIX B: Software Variables

Table 19 - Main VI variables

| Variables | Type | Description |
|---|---|---|
| x_go_to | double precision | Indicates the desired coordinate of the x-axis |
| y_go_to | double precision | Indicates the desired coordinate of the y-axis |
| z_go_to | double precision | Indicates the desired coordinate of the z-axis |
| rz_go_to | double precision | Indicates the desired coordinate of the rotational Z axis |
| x_actual_pos. | double precision | Indicates the current x-axis coordinate |
| y_actual_pos. | double precision | Indicates the current y-axis coordinate |
| z_actual_pos. | double precision | Indicates the current y-axis coordinate |
| rz_actual_pos | double precision | Indicates the current y-axis coordinate |
| home | boolean | It corresponds to the click on the “Home” button. When it is “True”, the workbench starts the “Home” process. |
| front | boolean | It corresponds to the click on the “Front” button. When it is “True”, the trolley x moves forwards in the pre-selected distance. |
| back | boolean | It corresponds to the click on the “Back” button. When it is “True”, the trolley x moves backwards in the pre-selected distance. |
| left | boolean | It corresponds to the click on the “Left” button. When it is “True”, the trolley y moves leftwards in the pre-selected distance. |
| right | boolean | It corresponds to the click on the “Right” button. When it is “True”, the trolley y moves rightwards in the pre-selected distance. |
| down | boolean | It corresponds to the click on the “Down” button. When it is “True”, the trolley z moves downwards in the pre-selected distance. |
| up | boolean | It corresponds to the click on the “Up” button. When it is “True”, the trolley z moves upwards in the pre-selected distance. |
| CCW | boolean | It corresponds to the click on the “CCW” button. When it is “True”, the LMD support rotates counterclockwise in the pre-selected angle. |
| CW | boolean | It corresponds to the click on the “CW” button. When it is “True”, the LMD |

| | | |
|---|---|---|
| | | support rotates clockwise in the pre-selected angle. |
| go | boolean | It corresponds to the click on the “Go” button. When it is “True”, each axis moves to the desired coordinates. |
| add | boolean | It corresponds to the click on the “Add” button. When it is “True”, The “Add” state starts. |
| delete | boolean | It corresponds to the click on the “Delete” button. When it is “True”, The “Delete” state starts. |
| choose_cluster | array of strings | Indicates all digital clusters for which its test positions are recorded. |
| exit | boolean | It corresponds to the click on the “Exit” button. When it is “True”, The “Exit” state starts and the program closes. |
| go_resp | array of unsigned bytes | It is the response from the Arduino “goFunction”. |
| home_resp | array of unsigned bytes | It is the response from the Arduino “homeFunction”. |
| singleX_resp | array of unsigned bytes | It is the response from the Arduino “singleX” function. |
| singleY_resp | array of unsigned bytes | It is the response from the Arduino “singleY” function. |
| singleZ_resp | array of unsigned bytes | It is the response from the Arduino “singleY” function. |
| x1 | boolean | Indicates the status of the x-axis initial photoelectric detector. |
| x2 | boolean | Indicates the status of the x-axis intermediate photoelectric sensor |
| x3 | boolean | Indicates the status of the x-axis final photoelectric detector. |
| y1 | boolean | Indicates the status of the y-axis initial photoelectric detector. |
| y2 | boolean | Indicates the status of the y-axis intermediate photoelectric detector. |
| y3 | boolean | Indicates the status of the y-axis final photoelectric detector. |
| z1 | boolean | Indicates the status of the z-axis initial photoelectric detector. |
| z2 | boolean | Indicates the status of the z-axis intermediate photoelectric detector. |
| z3 | boolean | Indicates the status of the z-axis final photoelectric detector. |
| info_string | string | Indicates important information to the user |
| Next state | Enumerate constant | Indicates the next state of the program |

# APPENDIX C: Software Block Codes

In this section, the software code will be exposed. First, the LabVIEW code that commands the whole system, and second, the functions implemented in the Arduino firmware in order to perform the custom commands.

The code presented will be a complement to the code demonstrated in section 6.3.3. As for LabVIEW , the code for each status of the program will be displayed, with the exception of the status “Single Y” and “Single Z”, as they are similar to the status “SingleX”. Due to the code complexity, the presentation of all the cases of the “case structures” within it is not guaranteed.

## C.1 LabVIEW

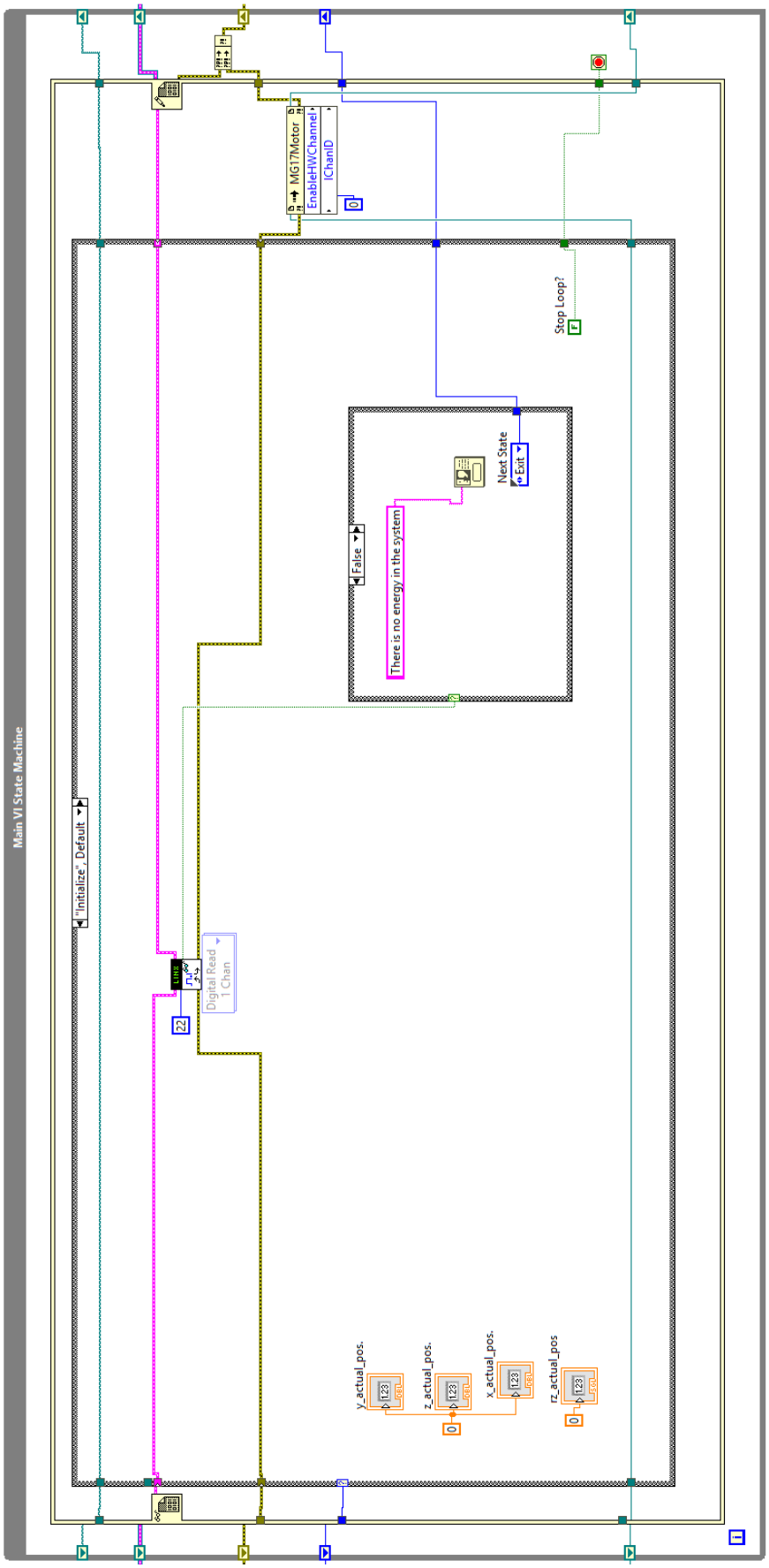



Figure 7.1 - Code of “Initialize” state

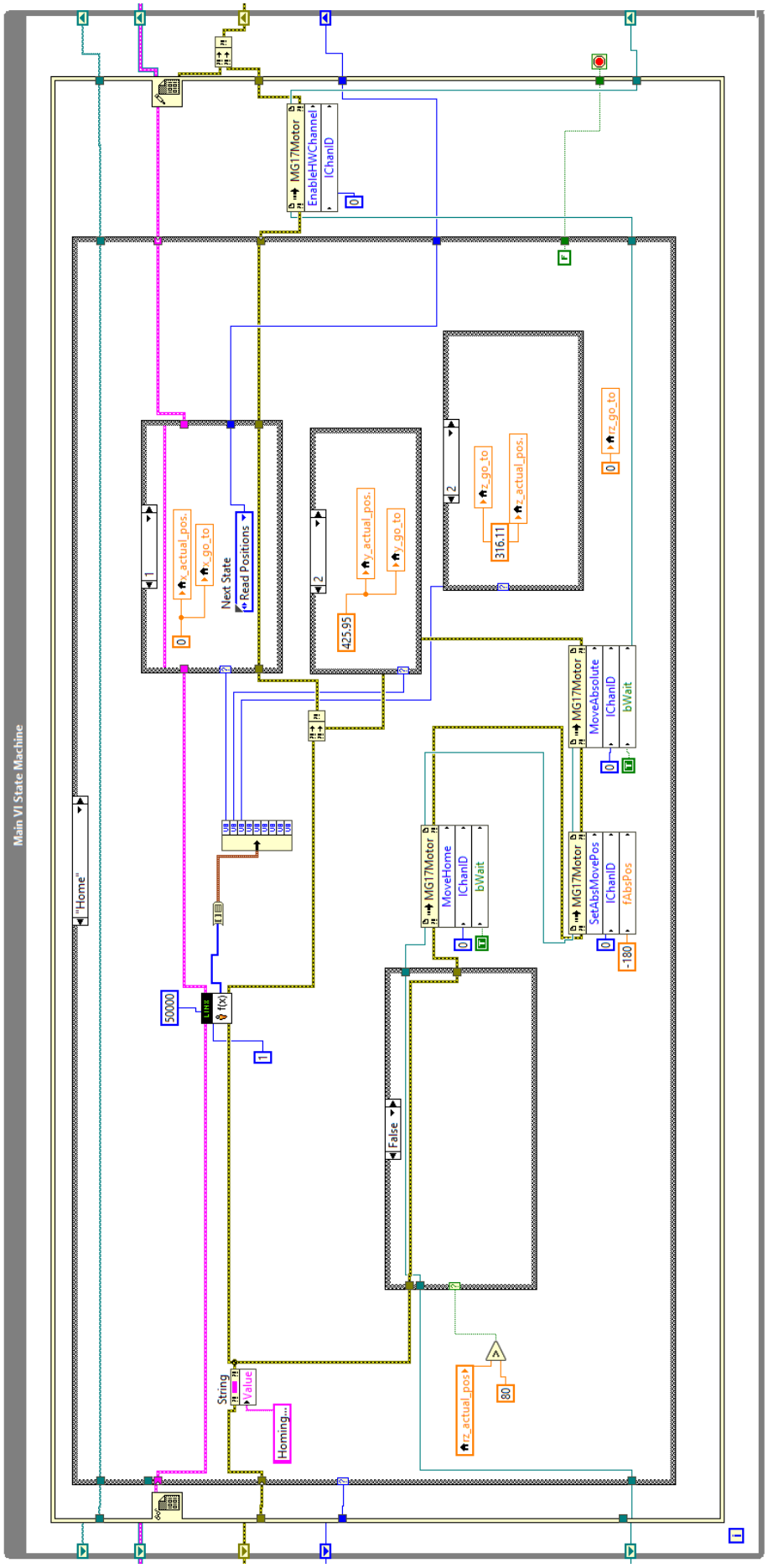



Figure 7.2 - Code of “Home” state

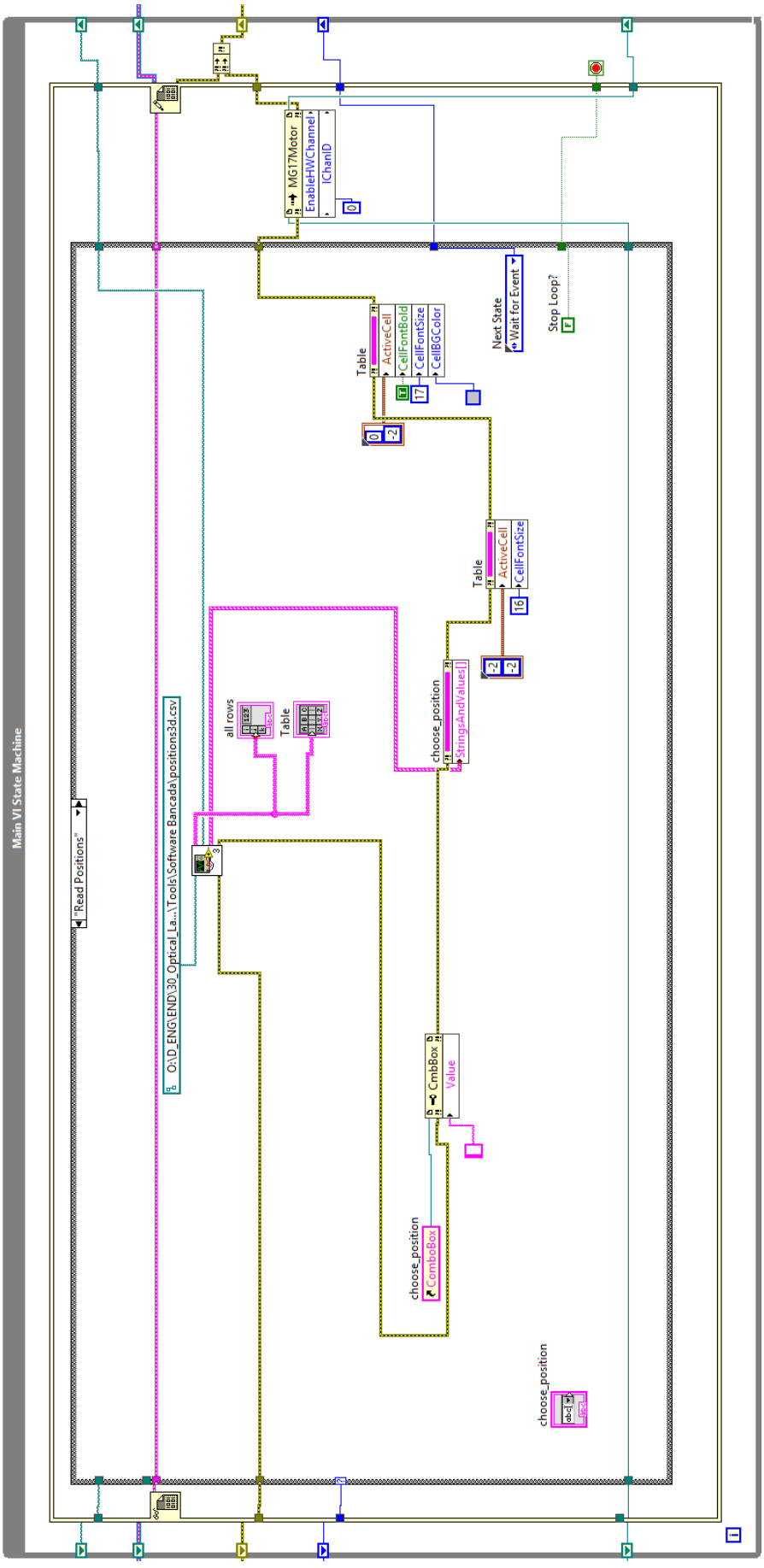



Figure 7.3 - Code of “Read” state

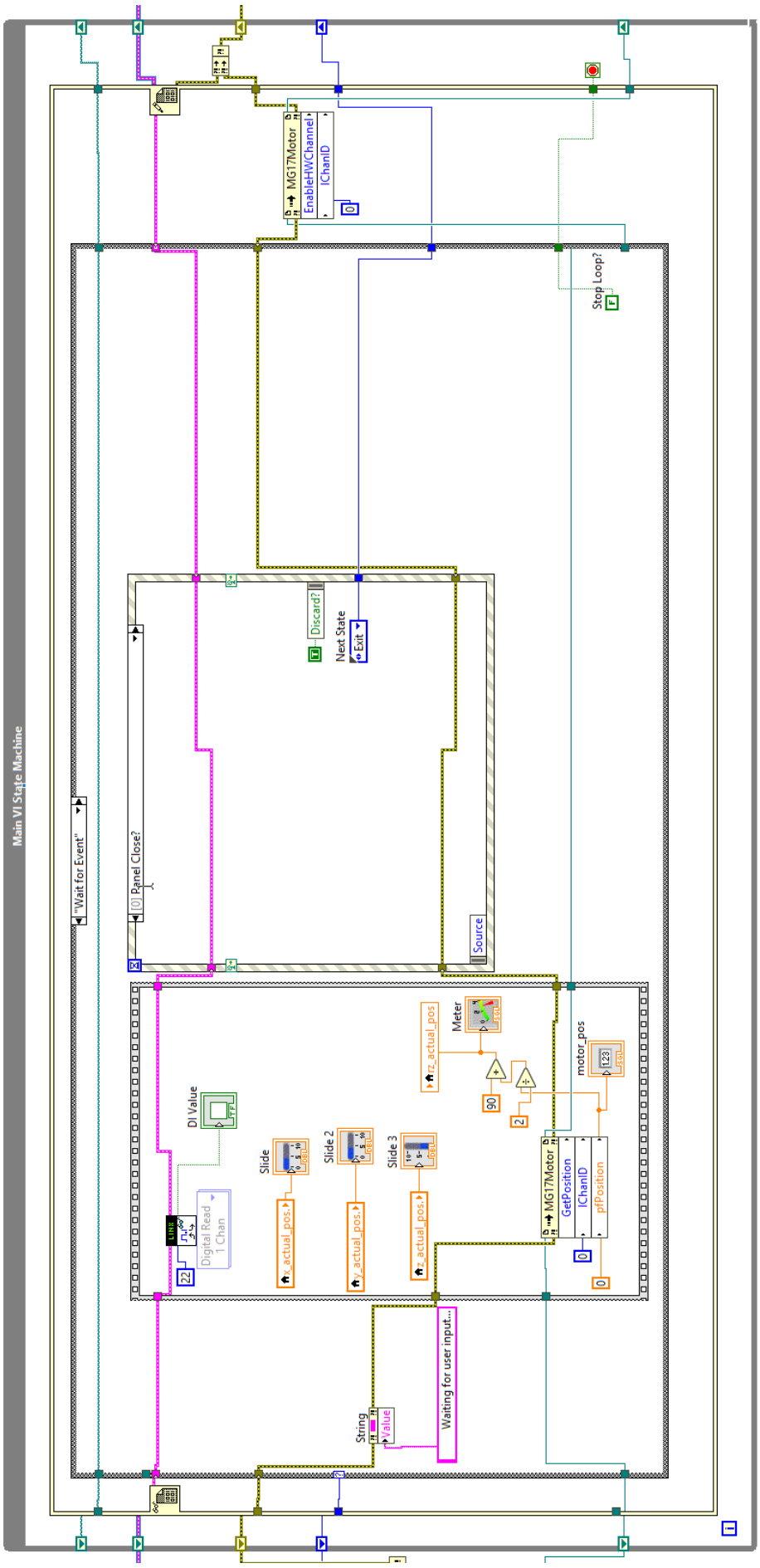



Figure 7.4 - Code of “Wait for Event” state

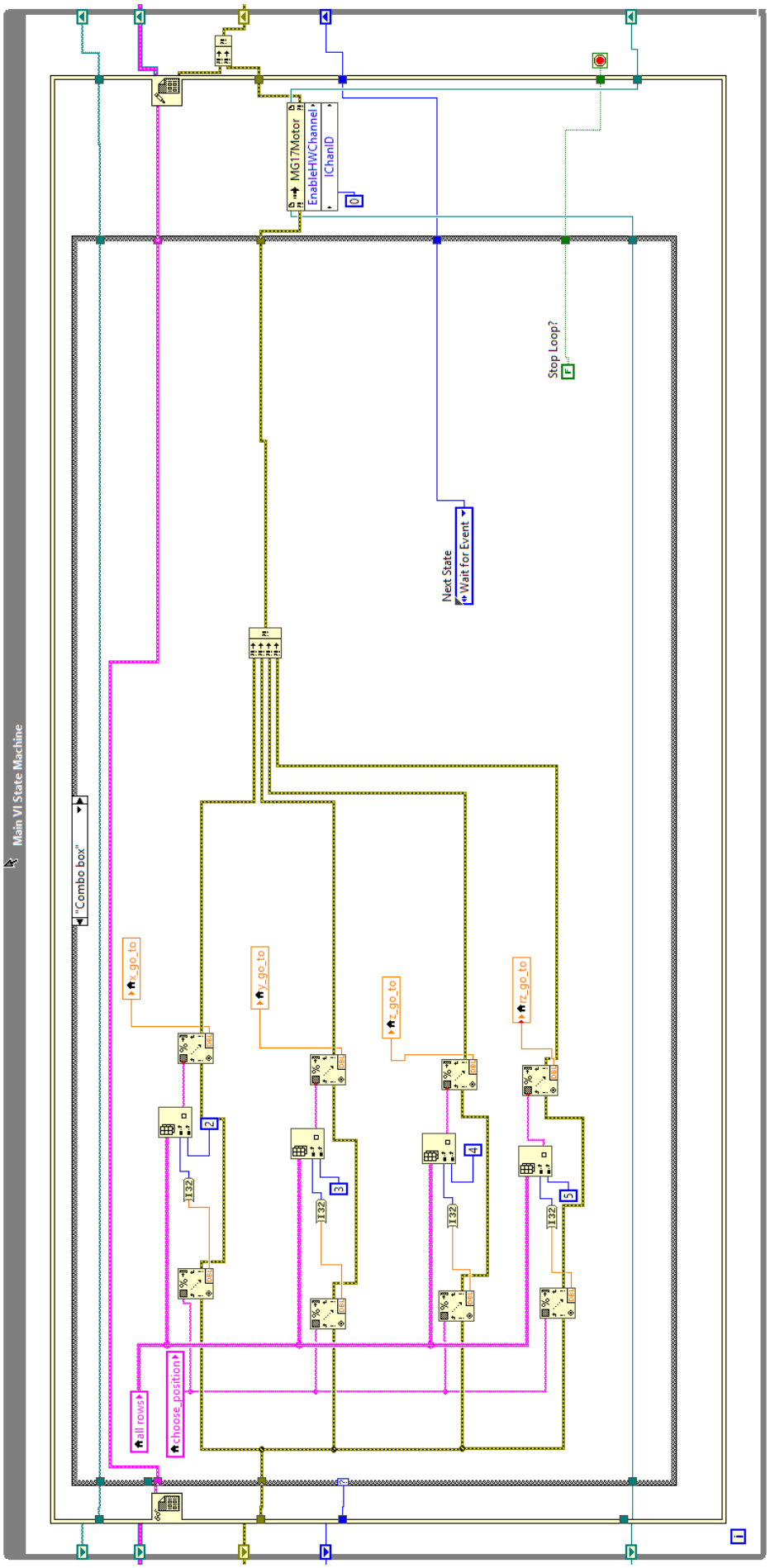



Figure 7.5 - Code of “Combo box” state

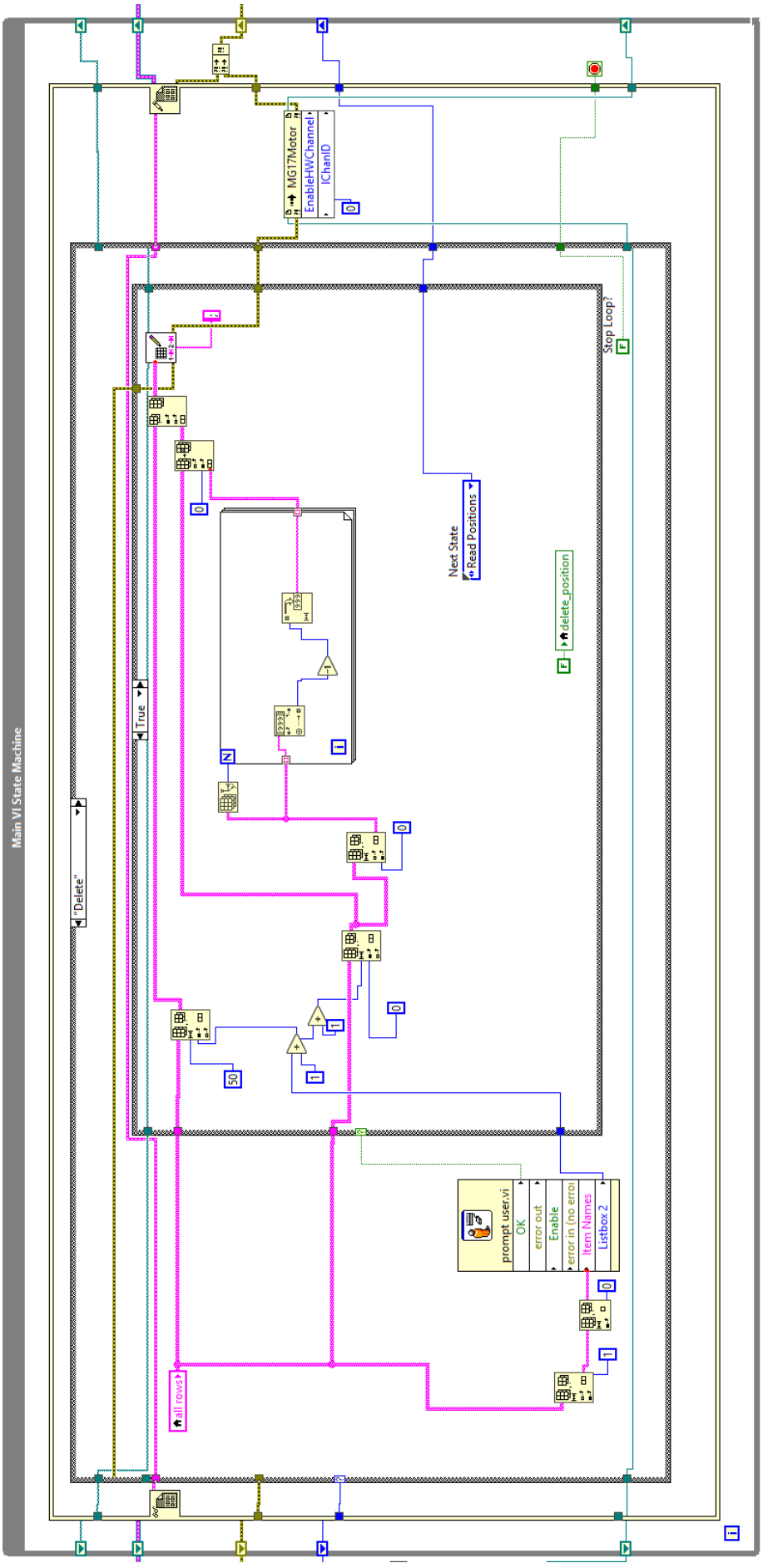



Figure 7.6 - Code of “Delete” state

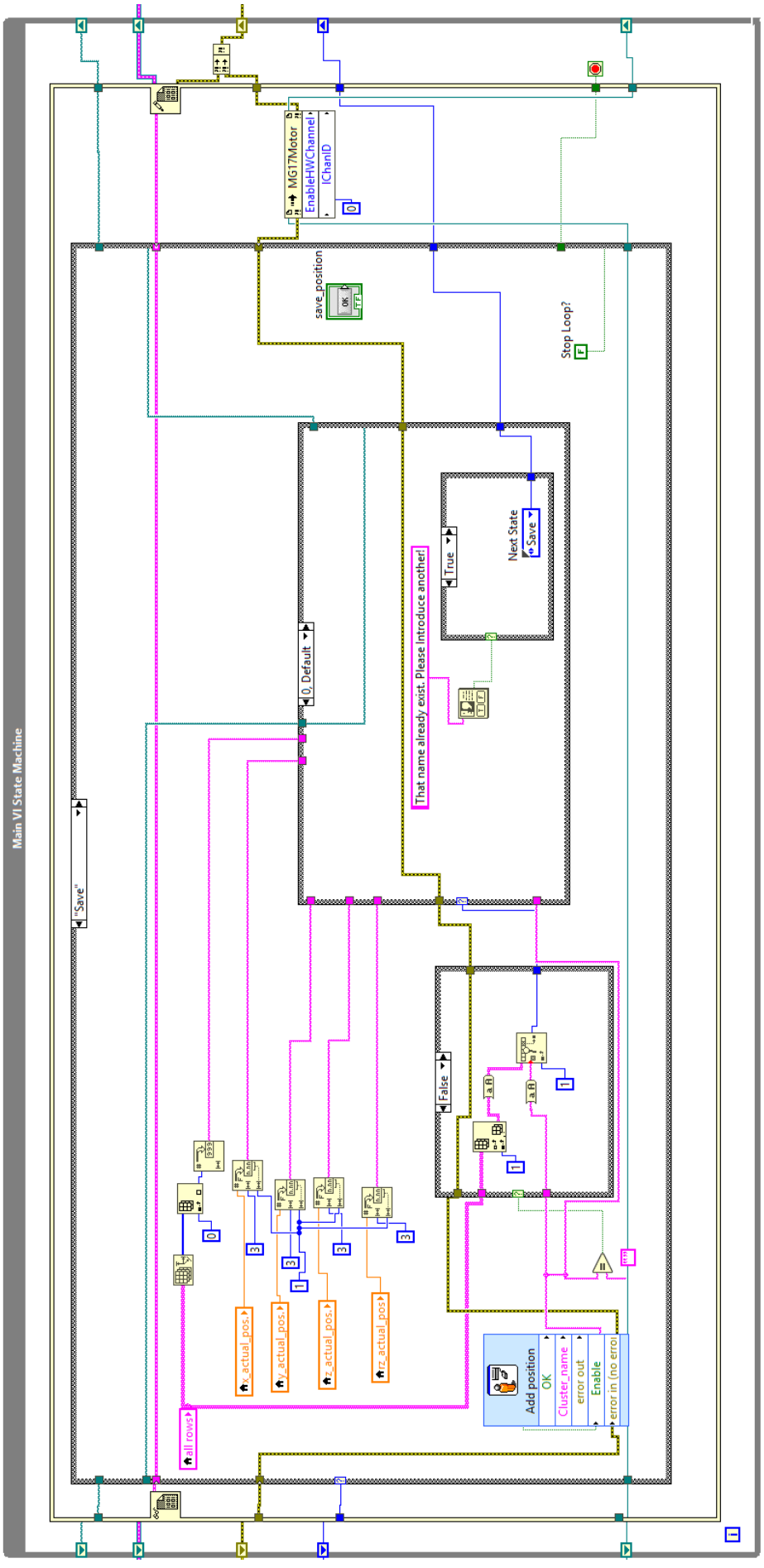



Figure 7.7 - Code of “Add” state

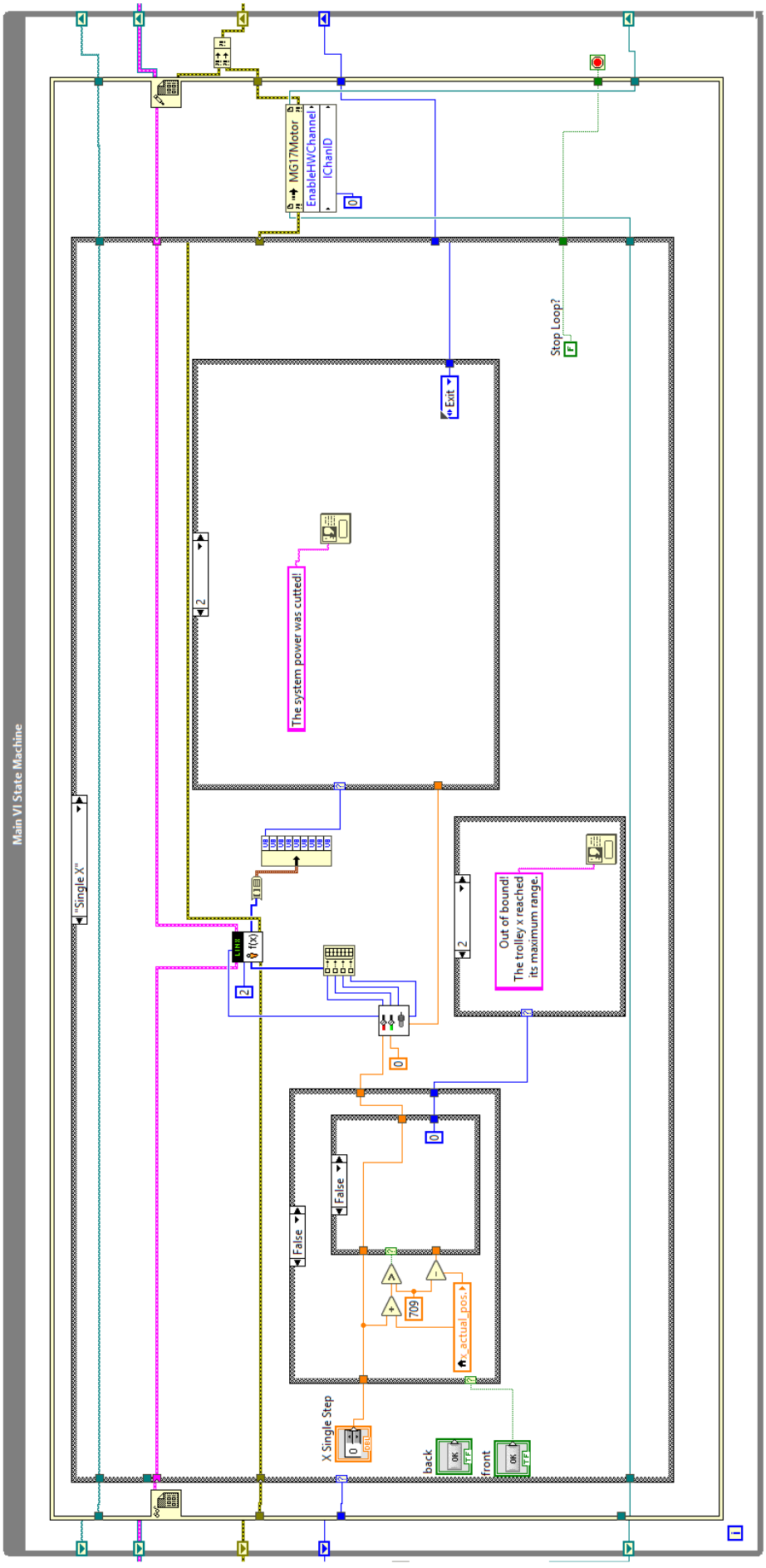



Figure 7.8 - Code of “Single X” state

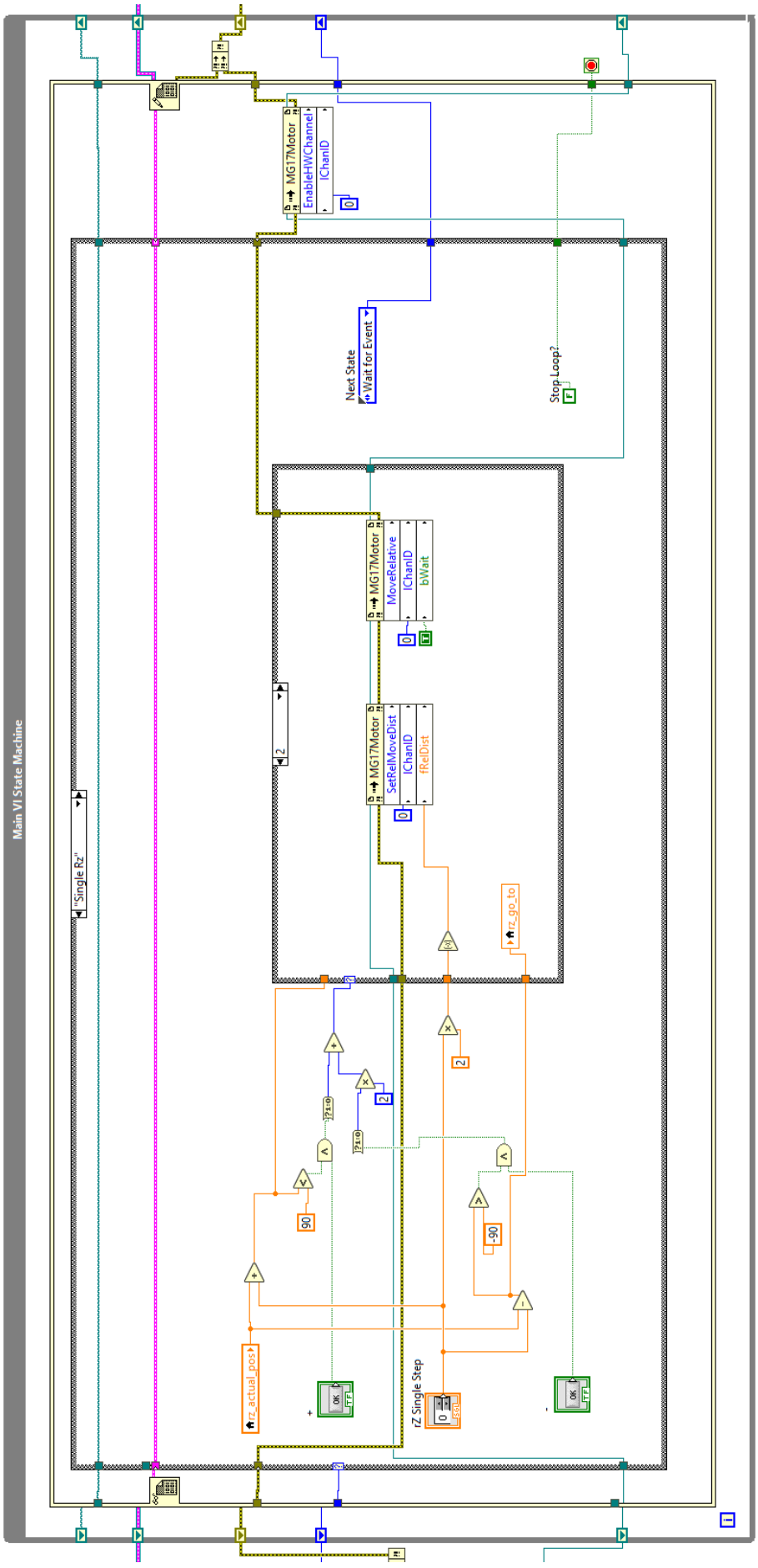



Figure 7.9 - Code of “Single RZ” state

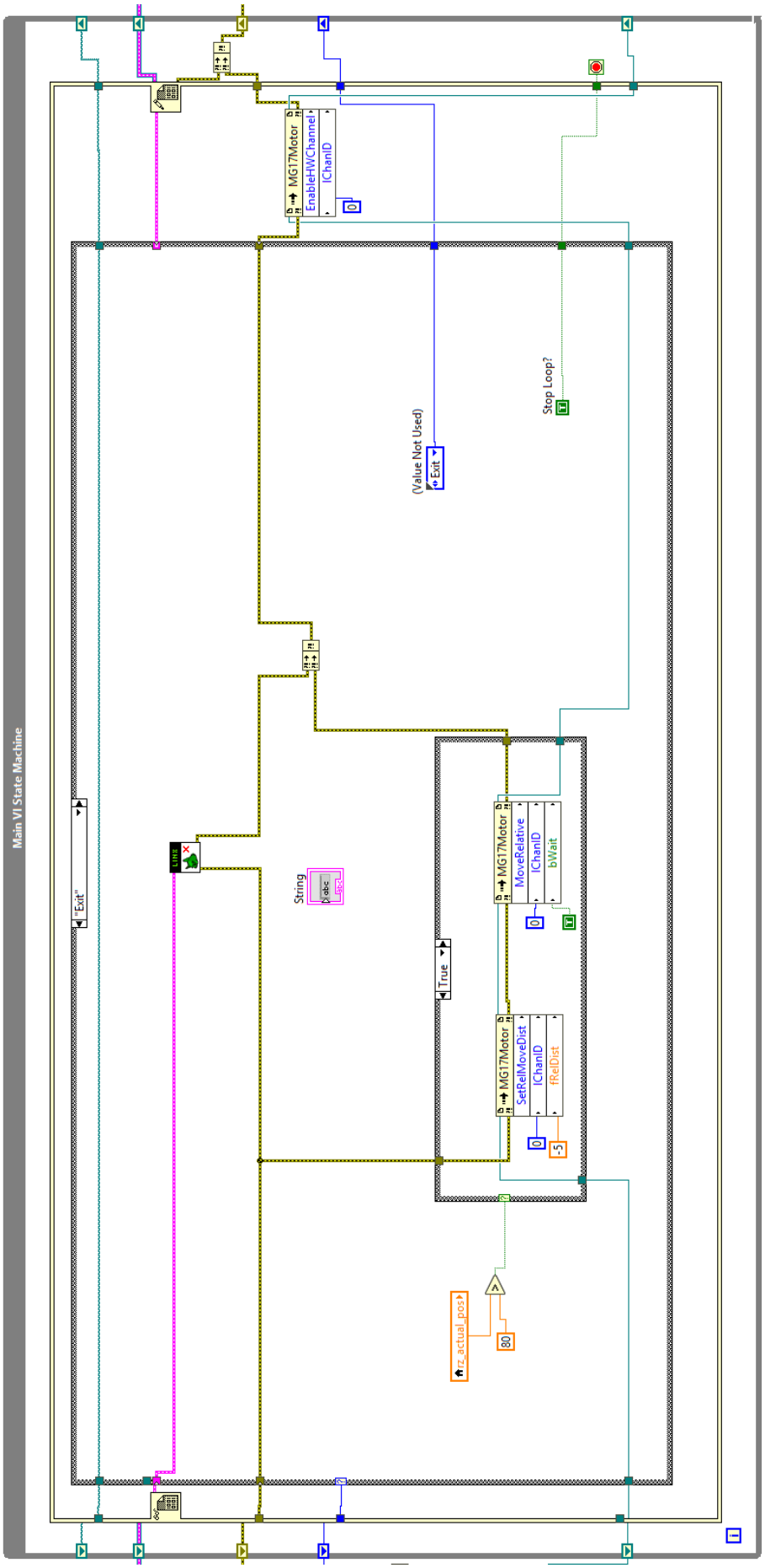



Figure 7.10 - Code of “Exit” state

## C.2 Arduino.

```
#include <SPI.h>
#include <Wire.h>
#include <EEPROM.h>
#include <Servo.h>

LinxArduinoMega2560* LinxDevice;
#include <AccelStepper.h>

#define dirPin1 52
#define stepPin1 5
#define dirPin2 50
#define stepPin2 3
#define dirPin3 48
#define stepPin3 2

#define motorInterfaceType 1
AccelStepper stepper1 = AccelStepper(motorInterfaceType, stepPin1, dirPin1);
AccelStepper stepper2 = AccelStepper(motorInterfaceType, stepPin2, dirPin2);
AccelStepper stepper3 = AccelStepper(motorInterfaceType, stepPin3, dirPin3);

int Gofunction();
int HomeFunction();
int singleX();
int singleY();
int singleZ();
long go1=0;
long go2=0;
long go3=0;

void setup()
{
   LinxDevice = new LinxArduinoMega2560();
   LinxSerialConnection.Start(LinxDevice, 0);
   LinxSerialConnection.AttachCustomCommand(0, Gofunction);
   LinxSerialConection.AttachCustomCommand(1, HomeFunction);
   LinxSerialConnection.AttachCustomCommand(2, singleX);
   LinxSerialConnection.AttachCustomCommand(4, singleY);
   LinxSerialConnection.AttachCustomCommand(6, singleZ);

   pinMode(31, INPUT);//zeroX
   pinMode(29, INPUT);//halfX
   pinMode(27, INPUT);//endX
   pinMode(26, INPUT);//zeroY
   pinMode(30, INPUT);//halfY
   pinMode(28, INPUT);//endY
   pinMode(23, INPUT);//zeroz
   pinMode(25, INPUT);//halfz
   pinMode(24, INPUT);//endz
   pinMode(22,INPUT);// relay contact

   stepper1.setEnablePin(53);
   stepper2.setEnablePin(51);
   stepper3.setEnablePin(49);

   stepper1.disableOutputs();
   stepper2.disableOutputs();
   stepper3.disableOutputs();
}
void loop()
{
   LinxSerialConnection.CheckForCommands();
}
```

Figure 7.11 - Initial Arduino code

```
int HomeFunction(unsigned char numInputBytes, unsigned char* input, unsigned char* numResponseBytes, unsigned
char* home_resp)
{
   stepper1.enableOutputs();
   stepper2.enableOutputs();
   stepper3.enableOutputs();

   stepper1.setMaxSpeed(1300);
   stepper1.setAcceleration(3000);
   stepper2.setMaxSpeed(1700);
   stepper2.setAcceleration(3000);
   stepper3.setMaxSpeed(1700);
   stepper3.setAcceleration(3000);

   stepper1.setCurrentPosition(0);
   stepper2.setCurrentPosition(0);
   stepper3.setCurrentPosition(0);

   if ((digitalRead(31)==HIGH)||(digitalRead(29)==HIGH))
     {
        stepper1.move(1000);
     }
   if ((digitalRead(26)==HIGH)||(digitalRead(30)==HIGH))
     {
        stepper2.move(1000);
     }
   if ((digitalRead(23)==HIGH)||(digitalRead(25)==HIGH))
     {
        stepper3.move(-1000);
      }
   while (stepper1.distanceToGo()!=0 || stepper2.distanceToGo()!=0 || stepper3.distanceToGo()!=0)
     {
        stepper1.run();
        stepper2.run();
        stepper3.run();
     }

   if ((((digitalRead(31)==LOW) && (digitalRead(29)==LOW)) || ((digitalRead(26)==LOW) &&
(digitalRead(30)==LOW)) || ((digitalRead(23)==LOW) && (digitalRead(25)==LOW))) &&
(digitalRead(22)==HIGH))
     {
        stepper1.moveTo(-999999999);
        stepper2.moveTo(-999999999);
        stepper3.moveTo(999999999);
        while((((digitalRead(31)==LOW) && (digitalRead(29)==LOW)) || ((digitalRead(26)==LOW) &&
(digitalRead(30)==LOW)) || ((digitalRead(23)==LOW) && (digitalRead(25)==LOW))) &&
(digitalRead(22)==HIGH))
        {
           if ((digitalRead(31)==LOW) && (digitalRead(29)==LOW))
             {
                stepper1.run();
             }
           if ((digitalRead(26)==LOW) && (digitalRead(30)==LOW))
              {
                 stepper2.run();
              }
           if ((digitalRead(23)==LOW) && (digitalRead(25)==LOW))
              {
                stepper3.run();
              }
        }
     }
```

```
  stepper1.setCurrentPosition(0);
  stepper2.setCurrentPosition(0);
  stepper3.setCurrentPosition(0);
  stepper1.setMaxSpeed(1000);
  stepper1.setAcceleration(1000);
  stepper2.setMaxSpeed(1000);
  stepper2.setAcceleration(1000);
  stepper3.setMaxSpeed(1000);
  stepper3.setAcceleration(1000);
  *numResponseBytes = 3;
  if (digitalRead(23)==HIGH)
    {
       home_resp[2] = 1;
       stepper3.move(-60);
    }
  else if (digitalRead(25)==HIGH)
    {
      home_resp[2] = 2;
      stepper3.move(-60);
    }
  else
    {
       home_resp[0]=6;
    }
  if (digitalRead(31)==HIGH)
    {
       home_resp[0] = 1;
       stepper1.move(60);
    }
  else if (digitalRead(29)==HIGH)
    {
       home_resp[0] = 2;
       stepper1.move(60);
    }
  else
     {
       home_resp[0] = 6;
     }
  if (digitalRead(26)==HIGH)
    {
       home_resp[1] = 1;
       stepper2.move(60);
    }
  else if (digitalRead(30)==HIGH)
    {
       home_resp[1] = 2;
       stepper2.move(60);
    }
  else
    {
       home_resp[0]=6;
    }
  while (stepper1.distanceToGo()!=0 || stepper2.distanceToGo()!=0 || stepper3.distanceToGo()!=0)
    {
       stepper1.run();
       stepper2.run();
       stepper3.run();
    }
  if ((digitalRead(31)==HIGH)|| (digitalRead(29)==HIGH) || (digitalRead(26)==HIGH)|| (digitalRead(30)==HIGH)||
(digitalRead(26)==HIGH) || (digitalRead(25)==HIGH))
    {
       home_resp[0]=7;
    }
  stepper1.disableOutputs();
  stepper2.disableOutputs();
  stepper3.disableOutputs();
  return 0;
 }
```

Figure 7.12 - Arduino Code: Home Function

```
int singleX(unsigned char numInputBytes, unsigned char* input, unsigned char* numResponseBytes, unsigned char*
upx_resp)
{
  stepper1.enableOutputs();
  stepper1.setMaxSpeed(2200);
  stepper1.setAcceleration(3000);
  go1 = ((long)input[0]*1000)+((long)input[1]*10)+((long)input[2]);

  if (input[3]==1)
    {
      go1 = -go1;
    }

  stepper1.setCurrentPosition(0);
  stepper1.move(go1);

  while (stepper1.distanceToGo()!=0 && digitalRead(31)==LOW && digitalRead(27)==LOW &&
digitalRead(22)==HIGH)
    {
      stepper1.run();
    }

  stepper1.setCurrentPosition(0);
  *numResponseBytes = 2;

   if (digitalRead(31)==HIGH || digitalRead(27)==HIGH)
    {
      upx_resp[0]=0;
    }
   else if (digitalRead(22)==LOW)
    {
      upx_resp[0]=2;
    }
   else
    {
      upx_resp[0]=1;
    }
  stepper1.disableOutputs();
  return 0;
}
```

Figure 7.13 - Arduino Code: SingleX function

```
int singleY(unsigned char numInputBytes, unsigned char* input, unsigned char* numResponseBytes, unsigned char*
upy_resp)
{
  stepper2.enableOutputs();
  stepper2.setMaxSpeed(2300);
  stepper2.setAcceleration(3000);
  go1 = ((long)input[0]*1000)+((long)input[1]*10)+((long)input[2]);

  if (input[3]==1)
    {
      go1 = -go1;
    }

  stepper2.setCurrentPosition(0);
  stepper2.move(go1);

  while (stepper2.distanceToGo()!=0 && digitalRead(26)==LOW && digitalRead(28)==LOW &&
digitalRead(22)==HIGH)
    {
      stepper2.run();
    }

  stepper2.setCurrentPosition(0);
  *numResponseBytes = 2;

  if (digitalRead(26)==HIGH || digitalRead(28)==HIGH)
    {
      upy_resp[0]=0;
    }
  else if (digitalRead(22)==LOW)
    {
      upy_resp[0]=2;
    }
  else
    {
      upy_resp[0]=1;
    }
  stepper2.disableOutputs();
  return 0;
}
```

Figure 7.14 - Arduino Code: SingleY function

```
int singleZ(unsigned char numInputBytes, unsigned char* input, unsigned char* numResponseBytes, unsigned char*
upz_resp)
{
  stepper3.enableOutputs();
  stepper3.setMaxSpeed(1700);
  stepper3.setAcceleration(3000);
  go1 = ((long)input[0]*1000)+((long)input[1]*10)+((long)input[2]);

  if (input[3]==0)
    {
      go1 = -go1;
    }

  stepper3.setCurrentPosition(0);
  stepper3.move(go1);

  while (stepper3.distanceToGo()!=0 && digitalRead(23)==LOW && digitalRead(24)==LOW &&
digitalRead(22)==HIGH)
    {
      stepper3.run();
    }

  stepper3.setCurrentPosition(0);
  *numResponseBytes = 2;

  if (digitalRead(23)==HIGH || digitalRead(24)==HIGH)
    {
      upz_resp[0]=0;
    }
  else if (digitalRead(22)==LOW)
    {
      upz_resp[0]=2;
    }
  else
    {
      upz_resp[0]=1;
    }
  stepper3.disableOutputs();
  return 0;
}
```

Figure 7.15 - Arduino Code: SingleZ function