



People's Democratic Republic of Algeria  
Ministry of Higher Education and Scientific Research  
University of Ghardaïa



Faculty of Sciences and Technology

Department of Automation and Electromechanical Engineering

Final Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

**MASTER**

**Field of Study:** Science and Technology

**Branch:** Automation

**Specialization:** Automation and Systems Engineering

**Title:**

**Design and Implementation of an Arduino  
Based PLC for an Educational Purpose**

مذكرة مؤسسة ناشئة وفق القرار الوزاري 1275.

**Presented by:**

Mohammed Abdellatif GUENDOUZ

Zakaria REZKI

**Publicly defended on:** 22/06/2026

**Board of Examiners:**

**Dr. Boukhari Hamed, (MCA), University of Ghardaïa – President**

**Dr. BEKKAR Belgacem, (MCA), University of Ghardaïa – Examiner**

**Dr. Rafik EULDJI, (MCB), University of Ghardaïa – Supervisor**

**Academic year: 2025 - 2026**

# ملخص

نحن في هذا العمل بصدد دراسة وتصميم لوحة إلكترونية تحاكي عمل وحدة تحكم منطقية قابلة للبرمجة معتمدة على شريحة "أردوينو نانو". و يتم تجربة هذا المشروع في بعض التطبيقات حتى نبرهن نجاحه وفعاليته في أداء مهامه. يتكون هذا المشروع من 4 أقسام رئيسية هي:

**الجزء الأول :** دراسة وحدات التحكم القابلة للبرمجة بشكل شامل (تعريفها، أنواعها، لغات البرمجة المستخدمة فيها، شكلها العام، تاريخها...)، ثم مقارنتها بأنواع أخرى من المتحكمات مثل الشرائح الإلكترونية، وأخيراً شرح عام للوحة الأردوينو ومقارنتها معاً والتعريف بأنواعها من ناحية الأداء والتكلفة.

**الجزء الثاني :** دراسة وشرح مفصل للتطبيقات المستخدمة في البرمجة والمحاكاة ورسم اللوحة الإلكترونية مع إعطاء مثال للتوضيح في كيفية ربط هاته البرامج ببعضها.

**الجزء الثالث :** دراسة وشرح تركيب المشروع بشكل مفصل بشكل عام ثم يتم شرحه جزءاً جزءاً، والتعريف بكل محتوياته المستخدمة، ثم شرح قائمة العناصر (أشباه النواقل، الأردوينو، المكثفات...) كل على حدة (آلية العمل، المداخل والمخارج...).

**الجزء الرابع :** تجربة مشروعنا في تطبيقات عملية ووضع بعض الأمثلة الأخرى لاختبار الأداء والتشغيل والتحكم السليم.

**الكلمات المفتاحية:** وحدة تحكم قابلة للبرمجة، لوحة دائرة مطبوعة، وحدة تحكم تعليمية، محرر لغة السلم، برمجة، دائرة إلكترونية، أردوينو نانو، تصميم دارات كهربائية، بروتينوس.

# Abstract

In this work, we are studying and designing an electronic circuit board that simulates the operation of a Programmable Logic Controller (PLC) based on the "Arduino Nano" chip. This project has been tested in several applications to demonstrate its success and effectiveness in performing its tasks. The project consists of four main sections:

**Part One:** A comprehensive study of PLCs (definition, types, programming languages, general architecture, history, etc.), followed by a comparison with electronic

microcontrôleurs. Enfin, il fournit une explication générale de la carte Arduino, comparant différents types en termes de performance et de coût.

**Part Two:** Une étude détaillée et une explication des applications utilisées en programmation, simulation, et conception électronique PCB, ainsi qu'un exemple illustratif démontrant comment ces outils sont interconnectés.

**Part Three:** Une étude complète et une explication de l'assemblage du projet, commençant par un aperçu global, suivi d'une décomposition étape par étape. Cela inclut l'identification de tous les composants utilisés et l'explication séparée de chaque élément (semi-conducteurs, Arduino, condensateurs, etc.), couvrant leurs principes de fonctionnement, entrées, et sorties.

**Part Four:** Tester notre projet dans des applications pratiques et fournir des exemples supplémentaires pour évaluer les performances, le fonctionnement, et la fonctionnalité de contrôle appropriée.

**Keywords:** Contrôleur Logique Programmable (PLC), Carte à Circuit Imprimé (PCB), Contrôleur Didactique, Générateur de Ladder, Programmation, Circuit Électronique, Arduino Nano, Conception de Circuit Électrique, Proteus.

# Résumé

Dans ce travail, nous étudions et concevons une carte électronique simulant le fonctionnement d'un Automate Programmable Industriel (API) basé sur la puce "Arduino Nano". Ce projet a été testé dans plusieurs applications afin de démontrer son succès et son efficacité dans l'exécution de ses tâches.

Le projet se compose de quatre sections principales :

**Première Partie :** Une étude complète des API (définition, types, langages de programmation, architecture générale, histoire, etc.), suivie d'une comparaison avec d'autres types de contrôleurs tels que les microcontrôleurs électroniques. Enfin, elle fournit une

explication générale de la carte Arduino, en comparant les différents types en termes de performance et de coût.

**Deuxième Partie :** Une étude détaillée et une explication des applications utilisées pour la programmation, la simulation et la conception de circuits imprimés (PCB), accompagnées d'un exemple illustratif démontrant comment ces outils sont interconnectés.

**Troisième Partie :** Une étude complète et une explication de l'assemblage du projet, commençant par une vue d'ensemble globale, suivie d'une décomposition étape par étape. Cela inclut l'identification de tous les composants utilisés et fournit une explication distincte pour chaque élément (semi-conducteurs, Arduino, condensateurs, etc.), couvrant leurs principes de fonctionnement, leurs entrées et leurs sorties.

**Quatrième Partie :** Test de notre projet dans des applications pratiques et fourniture d'exemples supplémentaires pour évaluer la performance, le fonctionnement et la fonctionnalité de contrôle appropriée.

**Les Mots Clé :** Automate Programmable Industriel (API), Carte de circuit imprimé (PCB), Contrôleur didactique, Éditeur de langage Ladder, Programmation, Circuit électronique, Arduino Nano, Conception de circuits électriques, Proteus.

# Dedication

First, we praise Allah, Glory be to Him, the Almighty, for His success and grace that enabled us to perform this work.

To our parents and families, who were our support throughout our journey.

To our honorable teachers, who nurtured us with their knowledge and experience.

To everyone who supported us in our academic journey, to our colleagues and friends.

May Allah reward you with the best reward, and thank you for your support.

أولاً، نحمد الله سبحانه وتعالى ونشكره على توفيقه وفضله الذي أعاننا على إنجاز هذا العمل.

إلى والدينا وعائلتنا، الذين كانوا سنداً لنا طوال رحلتنا.

إلى أساتذتنا الأفاضل، الذين غمرونا بعلمهم وخبرتهم.

إلى كل من ساندنا في مسيرتنا الأكاديمية، إلى زملائنا وأصدقائنا.

جزاكم الله عنا خير الجزاء، وشكراً لكم على دعمكم.

# Acknowledgments

First and foremost, we wish to thank Almighty God for granting us the strength, patience, will, courage, and health to complete this work.

We express our deepest gratitude to our parents and families, who supported and encouraged us throughout our long years of study.

We would like to express our utmost respect and gratitude to our supervisor, **Dr. Rafik EULDJI**, for:

- Accepting the responsibility of supervising us.
- Believing in our capabilities.
- Providing his invaluable advice and encouragement.
- And, above all, for his immense patience.

We also wish to thank the members of the jury.

Finally, our thanks go out to all the members of our graduating class and everyone who assisted us, whether morally or materially.

نحمد ونشكر الله عز وجل الذي منحنا القوة، والصبر، والإرادة، والشجاعة، والصحة لإتمام هذا العمل  
كما نعرب عن عميق امتناننا لوالدينا وعائلاتنا، الذين قدموا لنا كل الدعم والتشجيع طوال سنوات دراستنا الطويلة.

ونود أن نعبر عن أسمى آيات الاحترام والتقدير للمشرف الفاضل، الدكتور رفيق علجي، على:

- قبول مسؤولية الإشراف علينا.
- إيمانه بقدراتنا.
- توجيهاته القيمة وتشجيعه المستمر. وقبل كل شيء، على صبره الرحب اللامحدود.

كما نتوجه بالشكر الجزيل إلى أعضاء لجنة المناقشة الأفاضل.

وختاماً، يتجدد شكرنا لجميع زملائنا في دفعة التخرج، ولكل من قدم لنا يد العون والمساعدة، سواء كان ذلك مادياً أو معنوياً.

# List of tables

|                                                                               |        |
|-------------------------------------------------------------------------------|--------|
| Table 1.1: Comparison between the characteristics of the Arduino boards. .... | - 12 - |
| Table 1.2: Comparison between the PLC and Arduino. ....                       | - 13 - |
|                                                                               |        |
| Table 2.1: Description of the Instructions. ....                              | - 26 - |
| Table 2.2: Arithmetic Operations, Comparisons, ADC and PWM. ....              | - 26 - |
| Table 2.3: The Edit List Tools.....                                           | - 27 - |
| Table 2.4: Arduino IDE Tools Explanation. ....                                | - 29 - |
|                                                                               |        |
| Table 3.1: CNY17-3 Optocoupler Characteristics. ....                          | - 41 - |
| Table 3.2: The Amplifier Characteristics.....                                 | - 44 - |
| Table 3.3: The IRFZ48N MOSFET Characteristics. ....                           | - 47 - |
| Table 3.4: MOC3021 Opto-Triac Characteristics. ....                           | - 51 - |
| Table 3.5: BTA06-600 Triac Characteristics. ....                              | - 52 - |

# List of figures

|                                                                                       |        |
|---------------------------------------------------------------------------------------|--------|
| Figure 1.1: Programmable Logic Controller (PLC).....                                  | - 3 -  |
| Figure 1.2: The functioning of Programmable Logic Controller PLC.....                 | - 4 -  |
| Figure 1.3: The architecture of PLC. ....                                             | - 4 -  |
| Figure 1.4: Compact PLC picture. ....                                                 | - 6 -  |
| Figure 1.5: Modular PLC Picture and Diagram.....                                      | - 7 -  |
| Figure 1.6: The history and competition between the companies (1968 - 1995). ....     | - 8 -  |
| Figure 1.7: The Detailed explanation of Arduino. ....                                 | - 10 - |
| Figure 1.8: Arduino Types. ....                                                       | - 11 - |
|                                                                                       |        |
| Figure 2.1: Ladder Maker Interface. ....                                              | - 16 - |
| Figure 2.2: Ladder Maker Tool Description. ....                                       | - 17 - |
| Figure 2.3: The File Menu.....                                                        | - 18 - |
| Figure 2.4: The Edit Menu. ....                                                       | - 18 - |
| Figure 2.5: The Hardware Menu.....                                                    | - 19 - |
| Figure 2.6: The Help Menu. ....                                                       | - 19 - |
| Figure 2.7: The Physical Pins and the Equivalent in Ladder Maker of Arduino Uno. .... | - 21 - |
| Figure 2.8: The Physical Pins and the Equivalent in Ladder Maker of Arduino Nano....  | - 22 - |
| Figure 2.9: The Equivalent Pins in Ladder Maker. ....                                 | - 23 - |
| Figure 2.10: The Physical Arduino Mega Pins. ....                                     | - 24 - |
| Figure 2.11: Elements tools. ....                                                     | - 24 - |
| Figure 2.12: Arduino IDE Tools Explanation. ....                                      | - 29 - |
| Figure 2.13: The Example of the 2 Push Buttons Controlling Digital Inputs. ....       | - 31 - |
| Figure 2.14: The Ladder program. ....                                                 | - 32 - |
| Figure 2.15: The C++ Program.....                                                     | - 33 - |
| Figure 2.16: The Overlapping Orders Between C++ and Ladder Diagram. ....              | - 33 - |
| Figure 2.17: The Link That Needs to be Transferred to Proteus. ....                   | - 34 - |
| Figure 2.18: The Link in Proteus. ....                                                | - 35 - |
| Figure 2.19: Simulation Mode (Then Press the Start and Stop Button).....              | - 35 - |
|                                                                                       |        |
| Figure 3.1: The Electronic Circuit Board.....                                         | - 38 - |
| Figure 3.2: The Proteus ISIS Circuit.....                                             | - 39 - |

|                                                          |        |
|----------------------------------------------------------|--------|
| Figure 3.3: Digital Inputs Diagram (Proteus ISIS).       | - 40 - |
| Figure 3.4: CNY17-X Optocoupler.                         | - 41 - |
| Figure 3.5: The Analog Inputs.                           | - 43 - |
| Figure 3.6: The Operational Amplifier LM324N.            | - 44 - |
| Figure 3.7: PWM Outputs.                                 | - 46 - |
| Figure 3.8: IRFZ48N MOSFET.                              | - 47 - |
| Figure 3.9: The Digital Outputs.                         | - 48 - |
| Figure 3.10: 220VAC Analog Outputs.                      | - 50 - |
| Figure 3.11: MOC3021 Opto-Triac.                         | - 50 - |
| Figure 3.12: BTA06 Triac Picture and Pinout.             | - 51 - |
| Figure 3.13: Arduino Nano Pinout and the Power Section.  | - 53 - |
| Figure 3.14: Arduino Nano.                               | - 54 - |
| Figure 3.15: The Terminal Blocks of the Power Section.   | - 55 - |
| Figure 3.16: ARES Diagram.                               | - 56 - |
| Figure 3.17: The PCB Design.                             | - 56 - |
| Figure 3.18: The Design of the Didactic Controller.      | - 57 - |
| Figure 3.19: Rotary Potentiometer.                       | - 58 - |
| Figure 3.20: ABB 24VDC Contactor.                        | - 59 - |
| Figure 3.21: Banana Jacks.                               | - 59 - |
| Figure 3.22: Pushbuttons.                                | - 60 - |
| Figure 3.23: LED Pilot Lights.                           | - 60 - |
| Figure 3.24: ATX Power Supply.                           | - 61 - |
| Figure 3.25: The Didactic Controller.                    | - 61 - |
| Figure 3.26: The Security Steps.                         | - 62 - |
|                                                          |        |
| Figure 4.1: Start/Stop Latching Program.                 | - 64 - |
| Figure 4.2: The Test of the Start/Stop Latching Program. | - 65 - |
| Figure 4.3: Analog to Digital Converter Program.         | - 66 - |
| Figure 4.4: The Test of The ADC Converter Program.       | - 66 - |

# List of abbreviations

**A:** Analog.

**AC:** Alternating Current.

**ADC:** Analog to Digital Converter.

**AI:** Artificial Intelligence.

**AVR:** Automatic Voltage Regulator.

**CLK:** Clock.

**COM:** Common.

**CPU:** Central Processing Unit.

**CTU:** Count Up.

**D:** Digital.

**DC:** Direct Current.

**DIP:** Dual In-line Package.

**EMI:** Electromagnetic Interference Environment.

**ERP:** Enterprise Resource Planning.

**FBD:** Function Block Diagram.

**GM:** General Motors.

**GPS:** Global Positioning System.

**GSM:** Global System for Mobile Communications.

**HMI:** Human Machine Interface.

**I/O:** Inputs and Outputs.

**IC:** Integrated Circuit.

**IDE:** Integrated Development

**IEC:** International Electrotechnical

**IL:** Instruction List.

**IN:** Input.

**IoT:** Internet of Things.

**ISR:** Interrupt Service Routine.

**IT:** Information Technology.

**LD:** Ladder Diagram.

**LDR:** Light Dependent Resistor.

**LED:** Light-Emitting Diode.

**MOSFET:** Metal-Oxide-Semiconductor Field-Effect Transistor.

**MQTT:** Message Queuing Telemetry Transport.

**NC:** Normally Close.

**NO:** Normally Open.

**OOP:** Object-Oriented Programming.

**OT:** Operational Technology.

**OUT:** Output.

**PC:** Personal Computer.

**PCB:** Printed Circuit Board.

**PID:** Proportional Integral Derivative.

**PLC:** Programmable Logic Controller.

**PWM:** Pulse Width Modulation.

**SCADA:** Supervisory Control and Data Acquisition.

**SFC:** Sequential Function Chart.

**SME:** Small and Medium Enterprises.

**SSR:** solid-state relays.

**ST:** Structured Text.

**TOF:** Timer Off-Delay.

**TON:** On-Delay Timers.

**USB:** Universal Serial Bus.

# Table of Content

|                            |     |
|----------------------------|-----|
| ..... ملخص                 |     |
| Abstract .....             |     |
| Résumé .....               |     |
| Dedication.....            |     |
| Acknowledgments.....       |     |
| List of tables .....       | I   |
| List of figures.....       | II  |
| List of abbreviations..... | IV  |
| Table of Content.....      | VII |

## Chapter 01: Overview About Programmable Logic Controller

|                                                           |        |
|-----------------------------------------------------------|--------|
| I.1. Introduction: .....                                  | - 2 -  |
| I.2. Definition: .....                                    | - 2 -  |
| I.3. Function of a PLC:.....                              | - 3 -  |
| I.4. The architecture of the PLCs: .....                  | - 4 -  |
| I.5. The types of PLCs:.....                              | - 5 -  |
| I.5.1. Compact PLC (Fixed PLC): .....                     | - 6 -  |
| I.5.2. Modular PLC: .....                                 | - 6 -  |
| I.6. Historical background: .....                         | - 7 -  |
| I.7. Arduino Versus PLC :.....                            | - 9 -  |
| I.7.1. What is Arduino? .....                             | - 9 -  |
| I.7.2. The Function of the Arduino Board: .....           | - 10 - |
| I.7.3. The types and the characteristics of Arduino:..... | - 10 - |
| I.7.4. The Comparison between the Arduino boards:.....    | - 11 - |
| I.7.5. The Similarities: .....                            | - 12 - |
| I.7.6. Difference between Them:.....                      | - 12 - |
| I.8. Motivation: .....                                    | - 13 - |
| I.9. Conclusion:.....                                     | - 14 - |

## Chapter 02: Software Tutorial

|                           |        |
|---------------------------|--------|
| II.1. Introduction: ..... | - 16 - |
| II.2. Ladder Maker: ..... | - 16 - |
| II.2.1. Definition: ..... | - 16 - |

|                                                                     |        |
|---------------------------------------------------------------------|--------|
| II.2.2: Tool Descriptions:.....                                     | - 17 - |
| II.3: Arduino IDE Tutorial:.....                                    | - 28 - |
| II.3.1: Definition: .....                                           | - 28 - |
| II.3.2: Arduino IDE Tools:.....                                     | - 29 - |
| II.4. A Practical Example Illustrating How They Work Together:..... | - 30 - |
| II.4.1. The Explanation of the Example: .....                       | - 30 - |
| II.4.2. Ladder Maker Part: .....                                    | - 31 - |
| II.4.3. Arduino IDE Part: .....                                     | - 33 - |
| II.4.4. Proteus ISIS Part:.....                                     | - 34 - |
| II.5. Conclusion: .....                                             | - 35 - |

### **Chapter 03: Hardware Design and Architecture**

|                                                    |        |
|----------------------------------------------------|--------|
| III.1. Introduction:.....                          | - 37 - |
| III.2. Overview About the Electronic Board: .....  | - 37 - |
| III.2.1. General Explanation of The Board: .....   | - 37 - |
| III.3. The Digital Inputs:.....                    | - 39 - |
| III.3.1. The Installation of the Block: .....      | - 39 - |
| III.3.2. Explain of the Components used: .....     | - 41 - |
| III.4. Analog Inputs: .....                        | - 42 - |
| III.4.1. The Installation of the Block: .....      | - 42 - |
| III.4.2. Explain of the Components used: .....     | - 43 - |
| III.5. PWM (Pulse Width Modulation) Outputs: ..... | - 45 - |
| III.5.1. The Installation of the Block: .....      | - 45 - |
| III.5.2. Explain of the Components used: .....     | - 46 - |
| III.6. The Digital Outputs: .....                  | - 47 - |
| III.6.1. The Installation of the Block: .....      | - 47 - |
| III.7. The (220VAC) Analog Outputs:.....           | - 49 - |
| III.7.1. The Installation of the Block: .....      | - 49 - |
| III.7.2. Explain of the Components used: .....     | - 50 - |
| III.8. Arduino Nano Part:.....                     | - 52 - |
| III.8.1. The Installation of the Block: .....      | - 52 - |
| III.8.2. Explain of the Components used: .....     | - 54 - |
| III.9. The Power Terminal Blocks:.....             | - 54 - |
| III.10. The PCB Part (ARES): .....                 | - 55 - |
| III.11. The Didactic Controller: .....             | - 57 - |
| III.11.1. Explanation of the Model:.....           | - 57 - |

|                                                |        |
|------------------------------------------------|--------|
| III.11.2. Explanation of the Components: ..... | - 57 - |
| III.12. The Security Steps:.....               | - 62 - |
| III.13. Conclusion: .....                      | - 62 - |

## **Chapter 04: The Experimental Results**

|                                                                |        |
|----------------------------------------------------------------|--------|
| IV.1. Introduction: .....                                      | - 64 - |
| IV.2. Programming Phase and Didactic Controller Testing: ..... | - 64 - |
| IV.2.1. Digital inputs Example:.....                           | - 64 - |
| IV.2.2. Analog to Digital Converter Program: .....             | - 65 - |
| IV.3. Conclusion: .....                                        | - 66 - |
| General Conclusion:.....                                       |        |
| References .....                                               |        |
| Appendix .....                                                 |        |

# **GENERAL INTRODUCTION**

In the industrial field, where competition is intense, organizations face the necessity of improving the quality and quantity of their products while reducing prices and costs. To achieve this, they must possess flexible and efficient production lines while minimizing human intervention. It is from this context that the concept of industrial automation emerged.

To carry out this task, the industrial Programmable Logic Controller (PLC) was introduced. PLCs first appeared in the United States around 1969 in automobile assembly lines, then emerged in France in 1971, subsequently becoming a global industrial revolution.

A Programmable Logic Controller is a programmable electronic machine capable of driving an industrial system using a simple and easy programming language, with precise response times in real-time.

Our work involves creating an educational tool that serves as a control system using an Arduino Nano board. We are referring here to a Programmable Logic Controller (PLC) designed based on programmed logic. This controller will be capable of executing most required applications simply by modifying the software while maintaining the same hardware architecture. Furthermore, the controller we are developing will generally meet the same requirements expected from classical controllers, particularly the ability to be programmed using standard programming languages via the "Ladder Maker" application.

This project can be used as an emulator to a Programmable Logic Controller (PLC). It has been tested through several programs to verify its readiness and efficiency in performing tasks.

Our work is organized into four chapters as follows:

### ***Chapter One:***

This chapter presents a general overview of industrial programmable logic controllers, including their history, fields of application, types, and programming languages. It also highlights the differences between PLCs and other control systems.

### ***Chapter Two:***

This chapter is an explanation of the software part used to design the printed circuit board (PCB) and the simulation using "Proteus" is explained, along with the implementation of programming using "Ladder Maker", we also use the "Arduino IDE" to be able to test the codes we create in the Ladder language in Proteus.

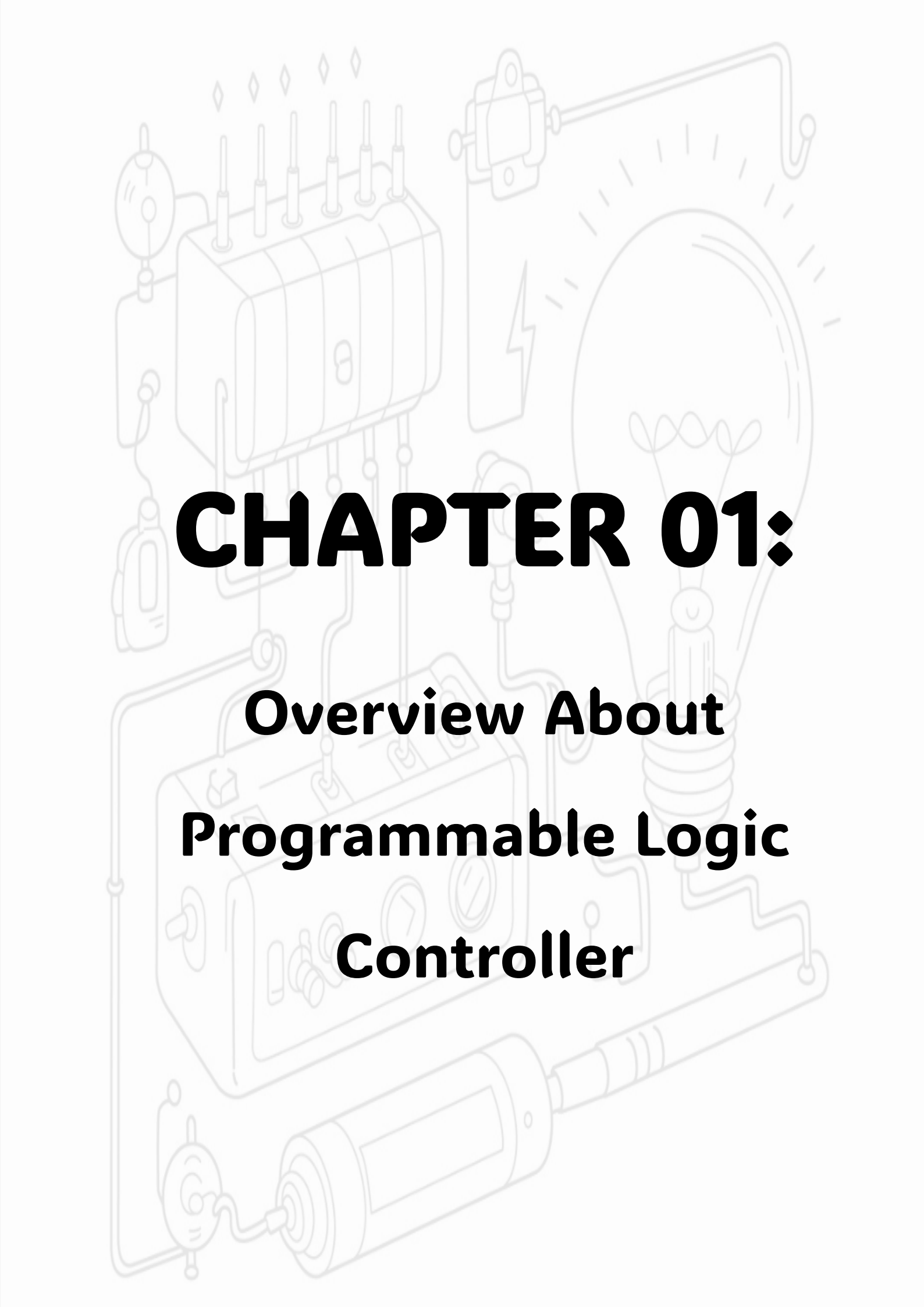
### ***Chapter Three:***

This chapter is dedicated to the design of the hardware part of the board. It is divided into input modules, output modules, and a processing unit. A detailed description of all the components and tools used in the project is provided.

### ***Chapter Four:***

This chapter focuses on the use of the electronic board after programming it in various projects to ensure its effectiveness in performing the assigned tasks. It also includes a discussion and analysis of the obtained results.

And finally, the work concludes with a general conclusion and future perspectives.



# **CHAPTER 01:**

## **Overview About Programmable Logic Controller**

## **I.1. Introduction:**

In this chapter, we will present a brief overview of the PLC, its operational mechanism, and its history. This will be followed by a comparison between the PLC and Arduino, highlighting the points of similarity and difference regarding characteristics, features, and other technical aspects.

## **I.2. Definition:**

A Programmable Logic Controller (PLC) is an industrial digital computer engineered to automate manufacturing processes and control systems. It was originally developed to replace traditional hardwired relay logic with more flexible, software-based control.

Built to endure extreme conditions such as high temperatures, dust, vibration, and electrical interference PLCs serve as the backbone of modern industrial automation. A standard PLC architecture consists of five core elements:

- **Central Processing Unit (CPU):** Known as the "brain" of the system, it executes the control logic and processes data.
- **Input Modules:** These units receive incoming signals and data from field sensors and devices.
- **Output Modules:** These components transmit command signals to actuators, including motors, solenoids, and valves.
- **Power Supply:** Ensures a stable and regulated voltage is provided to all internal components.
- **Communication Ports:** Facilitate seamless data exchange between the PLC and other controllers, Human-Machine Interfaces (HMIs), and supervisory management systems.

PLCs were invented in the late 1960s to replace complex, hard-wired relay logic systems in automotive manufacturing. A standard PLC consists of a Central Processing Unit (CPU), power supply, and input/output (I/O) modules.

- **Hardware and Environment:** PLCs are built to survive harsh industrial environments. They are highly resistant to extreme temperatures, electrical noise

(EMI/RFI), moisture, and heavy vibration. They typically operate on industrial voltage standards, most commonly **24V DC or 120/240V AC** for inputs and outputs.

- **Programming:** PLCs are programmed using standardized languages defined by the **IEC 61131-3** standard. The most common is Ladder Logic (LD), which resembles electrical relay schematics, making it intuitive for electricians and plant engineers. Other languages include Structured Text (ST), Function Block Diagram (FBD), and Sequential Function Chart (SFC).
- **Execution:** PLCs operate on a strict, continuous scanning cycle: they read all inputs, execute the logic program, update all outputs, and run internal diagnostics. This ensures deterministic, real-time control.

The figure below provides a simplified block diagram illustrating the flow: sensors feeding into input modules, the CPU at the core, output modules driving actuators:



Figure 1.1: Programmable Logic Controller (PLC).

### **I.3. Function of a PLC:**

The functioning of a PLC is characterized by a continuous, repetitive process known as the Scan Cycle. This cycle ensures that the controller is always reacting to real-time changes on the factory floor.

1. **Input Scan:** The PLC examines the status of every input device (such as push buttons, limit switches, or temperature sensors) and stores this data in its internal memory, often called the "input image table."

2. Program Execution: The CPU processes the user-defined logic typically written in Ladder Logic or Structured Text. It uses the data collected during the input scan to determine what actions need to take place according to the programmed instructions.
3. Output Scan: Based on the results of the program execution, the PLC updates the "output image table." It then sends these signals to the physical output devices, such as starting a motor or opening a solenoid valve.
4. Housekeeping/Communication: Finally, the PLC performs internal diagnostics and manages communication tasks (like updating a human-machine interface or talking to other controllers) before restarting the entire cycle.

The speed of this cycle is measured in milliseconds, allowing the PLC to maintain near-instantaneous control over industrial machinery.

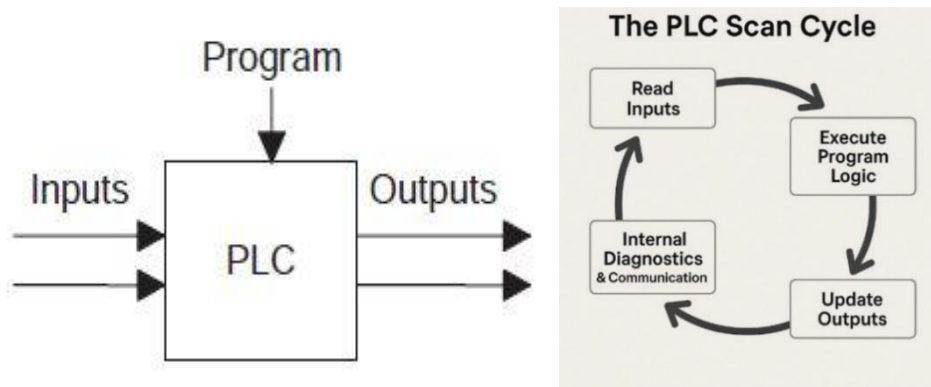


Figure 1.2: The functioning of Programmable Logic Controller PLC.

## **I.4. The architecture of the PLCs:**

Typically, a PLC system has the basic functional components of processor unit, memory, power supply unit, input/output interface section, communications interface, and the programming device. Figure 1.3 shows the basic arrangement. The constituent elements are:

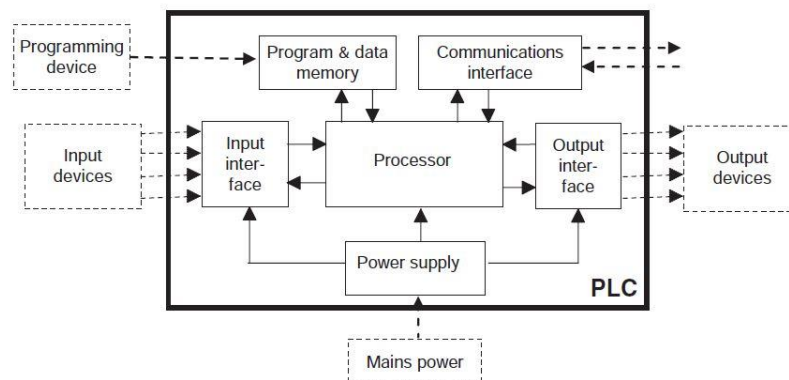


Figure 1.3: The architecture of PLC.

- The processor unit or central processing unit (CPU) is the unit containing the microprocessor. This unit interprets the input signals and carries out the control actions according to the program stored in its memory, communicating the decisions as action signals to the outputs.
- The power supply unit is needed to convert the mains AC voltage to the low DC voltage necessary for the processor and the circuits in the input and output interface modules.
- The programming device is used to enter the required program into the memory of the processor. The program is developed in the device and then transferred to the memory unit of the PLC.
- The memory unit is where the program containing the control actions to be exercised by the microprocessor is stored and where the data is stored from the input for processing and for the output.
- The input and output sections are where the processor receives information from external devices and communicates information to external devices. The inputs might thus be from switches, the automatic drill, or other sensors such as photoelectric cells, temperature sensors, flow sensors, or the like. The outputs might be to motor starter coils, solenoid valves, or similar things. Input and output devices can be classified as giving signals that are discrete, digital or analog. Devices giving discrete or digital signals are ones where the signals are either off or on. Thus, a switch is a device giving a discrete signal, either no voltage or a voltage. Digital devices can be considered essentially as discrete devices that give a sequence of on/off signals. Analog devices give signals of which the size is proportional to the size of the variable being monitored. For example, a temperature sensor may give a voltage proportional to the temperature.
- The communications interface is used to receive and transmit data on communication networks from or to other remote PLCs. It is concerned with such actions as device verification, data acquisition, synchronization between user applications, and connection management.

## **I.5. The types of PLCs:**

In industrial automation, Programmable Logic Controllers (PLCs) are generally categorized into two primary hardware architectures based on their physical construction and scalability: Compact and Modular.

### **I.5.1. Compact PLC (Fixed PLC):**

A Compact PLC is a self-contained unit where all the essential components the Central Processing Unit (CPU), Power Supply, and Input/Output (I/O) terminals are housed within a single, integrated enclosure.

**Characteristics:** It has a fixed number of I/O points determined by the manufacturer. While some modern compact PLCs allow for limited expansion via small plug-in modules, their primary capacity is "fixed."

**Best For:** Small-scale applications, standalone machines, and simple control tasks where the I/O requirements are unlikely to change.

**Advantages:** Lower cost, smaller physical footprint (space-saving), and simpler installation.



Figure 1.4: Compact PLC picture.

### **I.5.2. Modular PLC:**

A Modular PLC consists of a "rack," "backplane," or "bus" system that allows various separate hardware modules to be plugged in to build a customized system.

**Characteristics:** The CPU, Power Supply, and I/O are all separate modules. You can mix and match different types of modules (Digital Input, Analog Output, High-Speed Counters, Communication modules) according to the specific needs of the process.

**Advantages:** High scalability, easier troubleshooting (you can replace a single faulty module rather than the whole unit), and the ability to handle thousands of I/O points.

**Best For:** Complex industrial processes, large factory assembly lines, and systems that may require future expansion or specialized sensor integration.

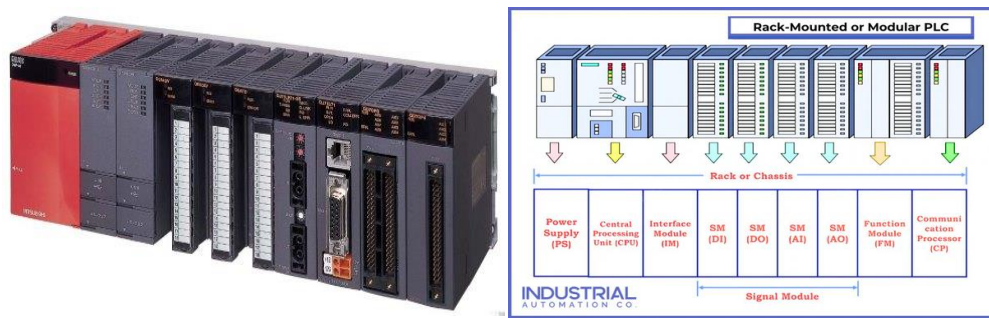


Figure 1.5: Modular PLC Picture and Diagram.

## I.6. Historical background:

The evolution of the Programmable Logic Controller (PLC) is a masterclass in industrial resilience, beginning as a pragmatic solution to a massive engineering headache. In 1968, GM was fed up with the "relay hell" of electromechanical systems vast, unreliable cabinets of hardwired components that required weeks of downtime to reconfigure for model year changes. Bedford Associates (later Modicon) answered this with the Modicon 084 in 1969. This was a "ruggedized" computer designed to survive the EMI, heat, and vibration of a factory floor. While other tech giants like Allen-Bradley leaned into the term "PLC" to make the technology feel less like a fragile PC and more like a dedicated industrial tool.

Despite early skepticism from plant floor veterans who didn't trust "invisible" software logic, the PLC's sheer uptime and the shift to PC-based programming environments eventually made it an industrial staple. The 80s and 90s saw the emergence of proprietary networking and the quest for interoperability. However, the 2000s marked a massive paradigm shift. We saw the rise of Industrial Ethernet and Programmable Automation Controllers (PACs), which effectively demolished the wall between IT (Information Technology) and OT (Operational Technology). This convergence allowed real-time data to flow from a single sensor on a conveyor belt all the way up to a corporate ERP system.

By the 2010s, the hardware had shrunk in footprint but exploded in capability. Controllers evolved into IoT-ready edge devices, boasting native cloud connectivity and hardened cybersecurity protocols. We moved beyond simple cyclic execution to controllers that could handle MQTT protocols and web servers internally. This wasn't just about turning a motor on and off anymore; it was about the PLC acting as a secure gateway for big data.

Today, we are looking at sixth-generation smart controllers that define the industry 4.0 landscape. These units utilize multi-core processing and AI-driven analytics to perform predictive maintenance and autonomous optimization. They have transitioned from being simple "black boxes" of logic into high-performance, intelligent backbones. In modern smart factories, these controllers manage everything from high-speed motion control to complex data synthesis, ensuring that global production lines are both resilient and fully transparent.

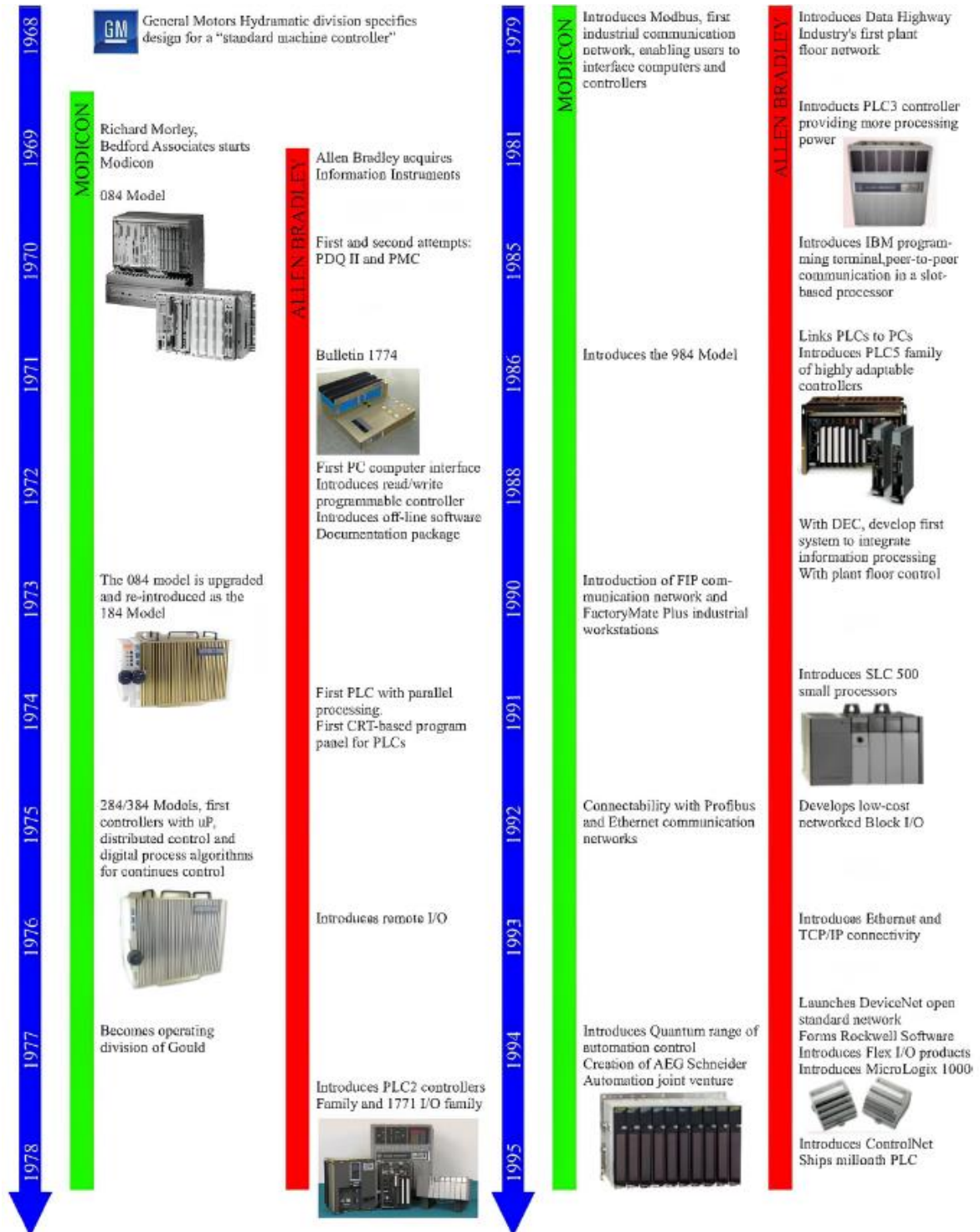


Figure 1.6: The history and competition between the companies (1968 - 1995).

## **I.7. Arduino Versus PLC :**

### **I.7.1. What is Arduino?**

Arduino is an open-source electronics ecosystem designed to simplify the interaction between software and the physical world. It serves as a programmable interface that allows users to create 'smart' environments by processing input signals and executing physical outputs. More than just a tool, it is a cross-platform framework that democratizes embedded computing, enabling rapid prototyping for anyone from hobbyists to professional developers through a simplified coding environment. The board itself comprises several vital components:

- **Microcontroller:** The "brain" those processes instructions.
- **Digital Pins:** Used to sense logic states (on/off) or provide logic outputs.
- **Analog Pins:** Used to read varying voltage levels (like temperature or sound volume).
- **USB Connector:** Used to power the board and upload code from a computer.
- **Power Jack:** Used to power the board from a wall outlet or battery.

Arduino Introduced in 2005 as a tool for students without a background in electronics and programming, Arduino has become the global standard for rapid prototyping.

**Hardware and Environment:** Standard Arduino boards (like the Arduino Uno) are built around Atmel AVR microcontrollers. They feature a set of digital and analog I/O pins that can interface to various expansion boards ("shields") and other circuits. They operate at low-voltage logic levels, typically **5V or 3.3V DC**. They are essentially bare-board electronics and offer little to no protection against industrial electrical noise, high voltages, or physical extremes.

**Programming:** Arduinos are programmed via the Arduino IDE using a simplified version of C/C++. The environment abstracts many complex microcontroller configurations, allowing users to simply write a `setup()` function (runs once) and a `loop()` function (runs continuously).

**Ecosystem:** The greatest strength of Arduino is its massive, global open-source community. There is an endless supply of pre-written code libraries for almost any sensor, motor, or communication module in existence.

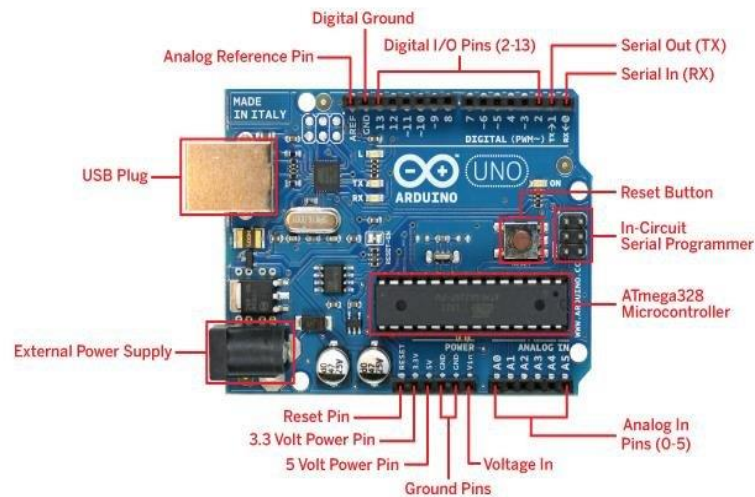


Figure 1.7: The Detailed explanation of Arduino.

### **I.7.2. The Function of the Arduino Board:**

The flexibility of the Arduino board is enormous so that one can do anything they imagine. This board can be connected very easily to different modules such as obstacle sensors, presence detectors, fire sensors, GSM Modules GPS modules, etc. The main function of the Arduino board is to control electronics through reading inputs & changing it into outputs because this board works like a tool. This board is also used to make different electronics projects in the field of electronics, electrical, robotics, etc.

### **I.7.3. The types and the characteristics of Arduino:**

Arduino boards serve as the versatile "brains" of the maker world, functioning as accessible tools that read inputs from sensors and translate them into physical outputs like motor movement or light. While the Arduino Uno remains the industry standard for beginners due to its simplicity, the ecosystem offers a diverse lineup tailored to specific needs.

For space-constrained projects, the Nano and Micro provide compact power, while the Mega and Due cater to complex builds requiring a high volume of I/O pins and faster 32-bit processing. Creative applications are supported by specialized boards like the wearable, washable LilyPad, the gamepad-style Esplora, and the wheeled Arduino Robot. Other variants, such as the Ethernet, Bluetooth, and Zero, focus on connectivity and IoT performance. Ultimately, the primary differences between these boards lie in their memory capacity, operating voltage (typically 3.3V or 5V), and pin count, allowing users to choose the perfect hardware for anything from simple automation to advanced robotics.

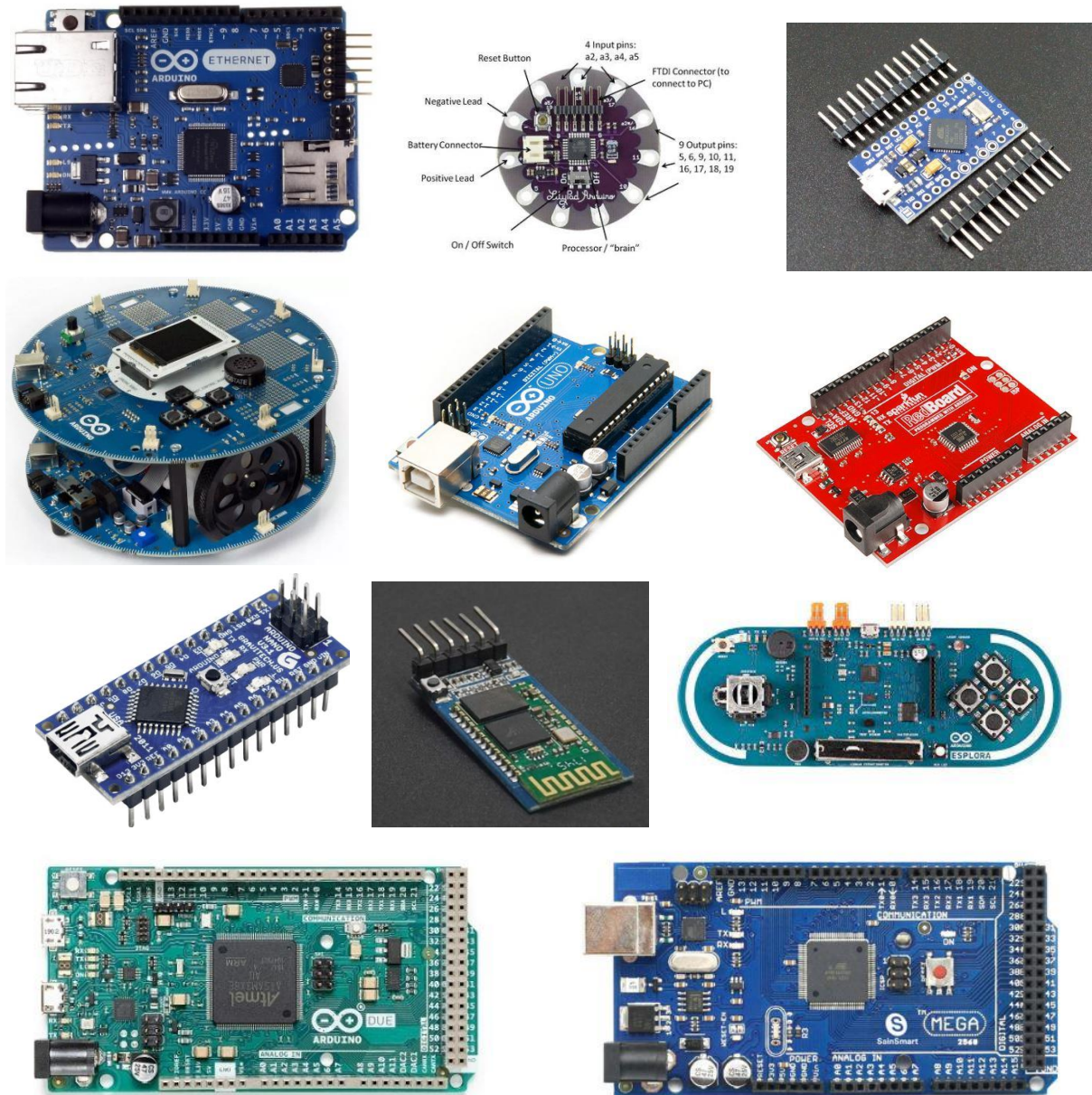


Figure 1.8: Arduino Types.

#### **I.7.4. The Comparison between the Arduino boards:**

While many of these boards share identical operating voltages and clock speeds, they primarily differ in their available memory capacities and the total number of input/output pins.

Table 1.1: Comparison between the characteristics of the Arduino boards.

| Arduino Board    | (V) of system | CLK Speed | Digital I/O | Analog inputs | Memory                   |
|------------------|---------------|-----------|-------------|---------------|--------------------------|
| Arduino Uno      | 5V            | 16MHz     | 14          | 6             | 2KB SRAM, 32KB flash     |
| RedBoard         | 5V            | 16MHz     | 14          | 6             | 2KB SRAM, 32KB flash     |
| Arduino Ethernet | 5V            | 16MHz     | 14          | 6             | 2KB SRAM, 32KB Flash     |
| Arduino Due      | 3.3V          | 84MHz     | 54          | 12            | 96KB SRAM, 512KB flash   |
| LilyPad Arduino  | 3.3V          | 8MHz      | 9           | 4             | 2.5 KB SRAM, 32 KB Flash |
| Arduino Mega     | 5V            | 16MHz     | 54          | 16            | 8KB SRAM, 256KB flash    |

### **I.7.5. The Similarities:**

Despite their differences, PLCs and Arduinos share a common architecture: both are microprocessor-based systems designed for I/O control, reading sensors to trigger physical outputs. They both operate via cyclic execution, running code in a continuous loop, and remain fully programmable, allowing logic to be updated as system needs evolve.

### **I.7.6. Difference between Them:**

PLCs and Arduino represent the divide between industrial-grade reliability and open-source flexibility. A PLC is a ruggedized, purpose-built computer designed to handle the harsh electrical noise, heat, and vibration of a factory floor. While it offers high-speed, deterministic processing for mission-critical tasks, it requires higher investment and specialized programming expertise.

In contrast, Arduino is an affordable, open-source microcontroller board ideal for prototyping and small-scale projects. It offers immense programming freedom and ease of use but lacks the native hardware protection and safety certifications required for heavy industrial use. Essentially, the PLC is built for the high-stakes reliability of a production line, while Arduino excels in versatile, low-cost applications.

Table 1.2: Comparison between the PLC and Arduino.

| Feature                        | PLC                                                                           | Arduino                                                                          |
|--------------------------------|-------------------------------------------------------------------------------|----------------------------------------------------------------------------------|
| <b>Primary Use Case</b>        | Industrial automation, critical infrastructure.                               | Prototyping, education, hobbyist projects, IoT.                                  |
| <b>Robustness</b>              | Industrial-grade; shielded against EMI, dust, and vibration.                  | Bare board; vulnerable to environmental factors and electrical noise.            |
| <b>Operating Voltage (I/O)</b> | 24V DC, 120V/240V AC (Standard).                                              | 5V DC or 3.3V DC (Requires external relays for high power).                      |
| <b>Programming Language</b>    | Ladder Logic, Structured Text, FBD (IEC 61131-3).                             | C/C++ (via Arduino IDE).                                                         |
| <b>Cost</b>                    | Hundreds to tens of thousands of dollars.                                     | \$10 to \$50.                                                                    |
| <b>Expandability</b>           | Snap-on I/O modules, communication racks (easily scales to thousands of I/O). | "Shields" stacked on top, or breadboard wiring (limited I/O scaling).            |
| <b>Lifespan/Support</b>        | Supported for decades; built to run 24/7 for years without failure.           | Rapid iteration; not designed for decades of continuous mission-critical uptime. |

Ultimately, the choice between a PLC and an Arduino hinge on the specific demands of the project specifically its requirements for precision, reliability, and scalability balanced against the available budget and technical proficiency.

## **I.8. Motivation:**

In this work, we aim to design an electronic circuit. The question arises whether to utilize a PLC or an Arduino for this project.

PLC Analysis: While a PLC provides high-performance industrial logic, it is characterized by high cost and a complex management interface.

Arduino Analysis: We have determined that an Arduino is the more suitable choice for this specific project. It is cost-effective and performs the required tasks with the same level of efficiency as a PLC.

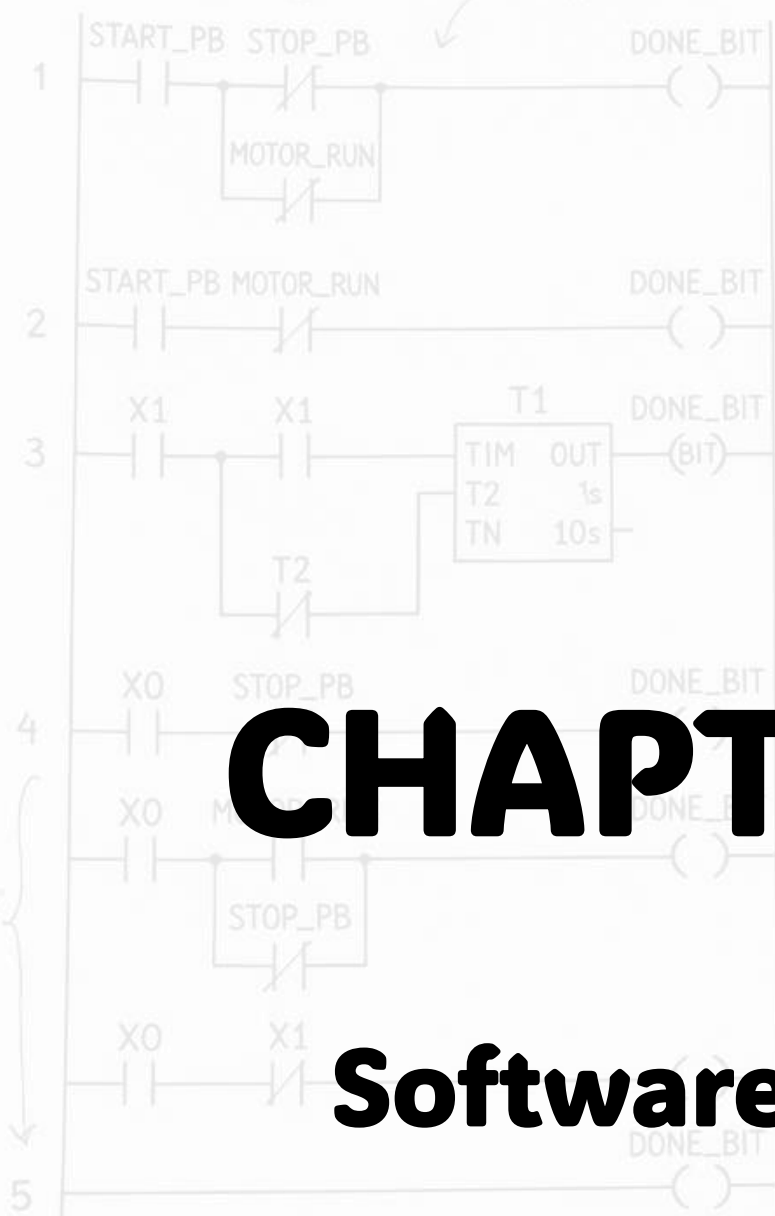
Therefore, by designing a specialized circuit for the project, we achieve a system that is simple, efficient, and robust. This approach offers a modern to using traditional PLCs in smaller-scale projects.

I believe that if this project is engineered correctly utilizing high-performance microchips and integrated circuits (ICs) while maintaining cost-efficiency it will serve as an effective substitute for Programmable Logic Controllers (PLCs) in specialized applications, particularly for Small and Medium Enterprises (SMEs) involving low-budget projects. Furthermore, the resulting system will be significantly more compact and exhibit superior aesthetic design parameters compared to traditional industrial frameworks.

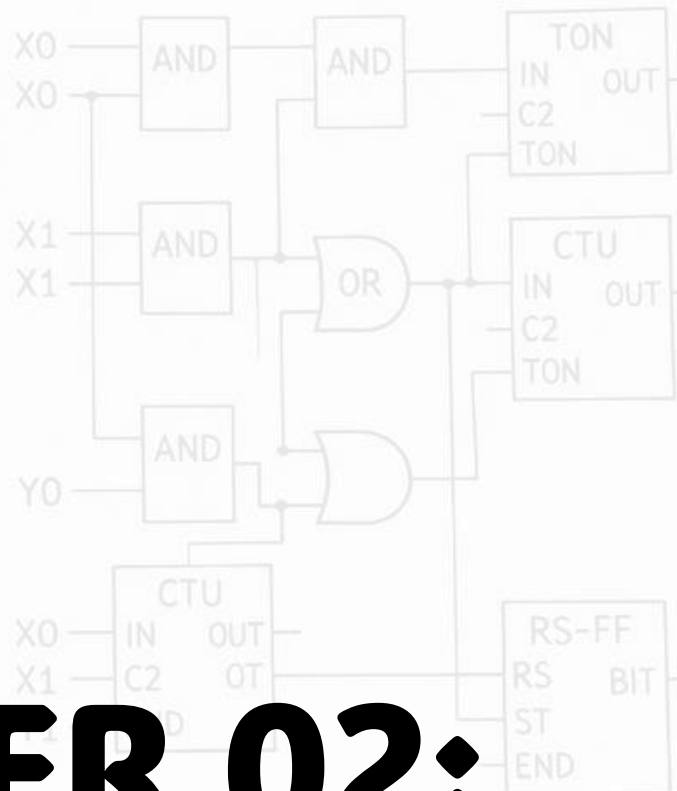
## **I.9. Conclusion:**

Through our study of this chapter, the difference between the PLC and Arduino has become clear to us, enabling us to determine which is most suitable for use in our project, as well as understanding its operational mechanism and creating a logic that resembles it.

## Ladder Logic



## FBD



# CHAPTER 02:

## Software Tutorial

### ST

→ IF Start\_Signal THEN  
Pump\_Running := TRUE;  
END\_IF;  
IF Start\_Signal THEN  
Pump\_Running := TRUE;  
END\_IF;

### IL

→ LD X0;  
AND X1;  
OUT Y0;  
OUT Y0;

### C++ code

```
#include <iostream>
int main()
{
    int var i = 1;
    while (i = 0; + i) {
        car i = 0;
        int i = 0;
        cout << "Value:" << i << endl;
    }
    return 0;
}
```

## II.1. Introduction:

In this chapter, we will address the software programming aspect of our project. Since we are working on implementing an electronic simulation board for a PLC, the best language to program it with is Ladder, which is the most prominent language used in the industrial field. We will explore this throughout this chapter.

## II.2. Ladder Maker:

### II.2.1. Definition:

Ladder Maker is a specialized, visual programming environment designed to transform an Arduino microcontroller into a functional Programmable Logic Controller (PLC) by utilizing Ladder Logic (LD), a graphical language modeled after relay-logic hardware schematics. Instead of writing lines of C++ code, users construct "rungs" on a virtual ladder where they drag and drop industrial symbols such as Normally Open (NO) and Normally Closed (NC) contacts, Output Coils, Counters, and Instruction Blocks like On-Delay Timers (TON). The software operates by mapping these virtual components to the physical General-Purpose Input/Output pins of the Arduino, then compiling the visual diagram into a low-level machine file (typically **.hex** or **.cpp**) that the hardware can execute. This detailed approach allows for the implementation of complex industrial automation tasks such as motor interlocking, sequential valve control, making it highly effective for users who prioritize electrical flow and Boolean logic over traditional software syntax.

When you turn on Ladder Maker, you will face 2 windows; A window for programming and work in it, and a window that shows you every step you take.

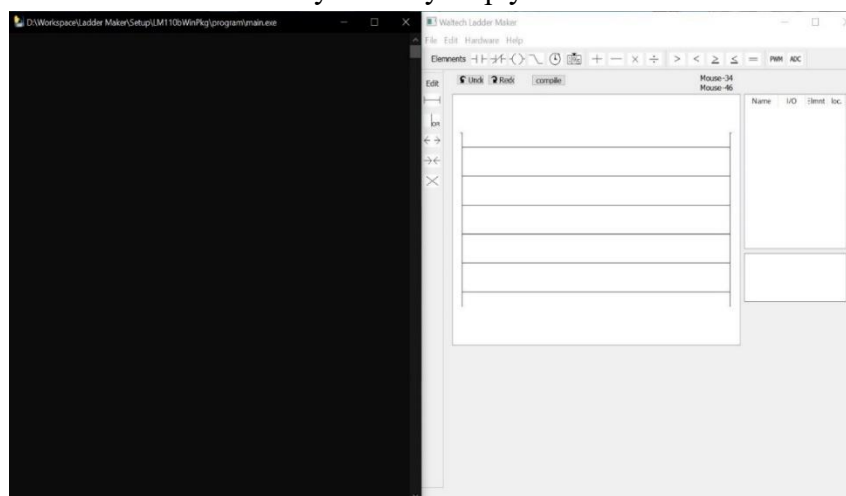


Figure 2.1: Ladder Maker Interface.

## II.2.2: Tool Descriptions:

The Ladder Maker consists of **9 main tools** you come across in the interface of the program, their details are in the image below, with the explanation attached below:

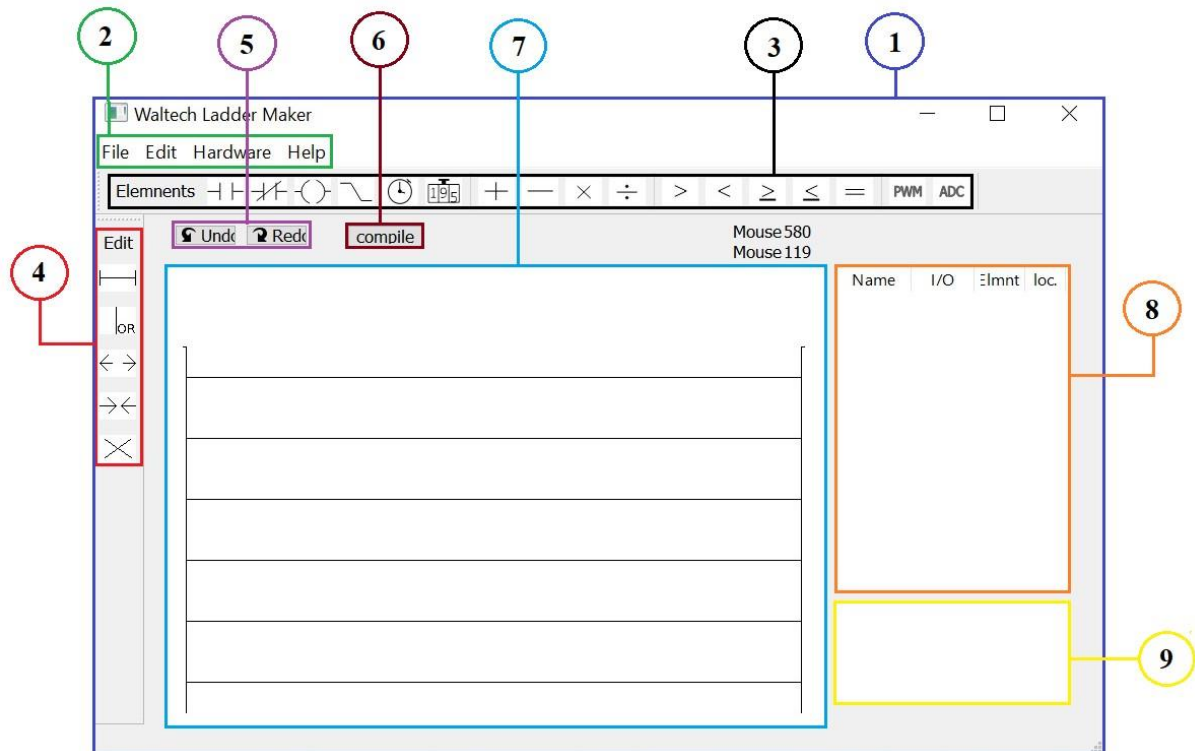


Figure 2.2: Ladder Maker Tool Description.

### II.2.2.1. Program Window:

The main container for the software. It provides the standard Windows controls (minimize, maximize, close) and acts as the frame for the entire IDE (Integrated Development Environment).

Window Control Icons Located at the top right of the Program Window.

- Minimize/Maximize: Standard Windows tools to manage your screen space, especially useful if you are running the Waltech Ladder Maker alongside a circuit simulator or a datasheet.
- Close (X): Exits the program. Usually, the software will trigger a pop-up asking if you want to save your changes to the ladder diagram before closing.

### II.2.2.2. Menu Bar:

Located at the top, it contains drop-down menus: File (for saving, opening, and new projects), Edit (for basic workspace adjustments), Hardware (to select your specific PLC model and communication ports), and Help (for documentation).

### The File Menu:

This is the most fundamental tool for project management.

- New/Open: Used to start a fresh ladder diagram or load an existing (.ld) file from your computer.
- Save/Save As: Critical for backing up your work. In industrial programming, it is common practice to save multiple versions (e.g., Project\_V1, Project\_V2) before making major changes to a live machine's logic.

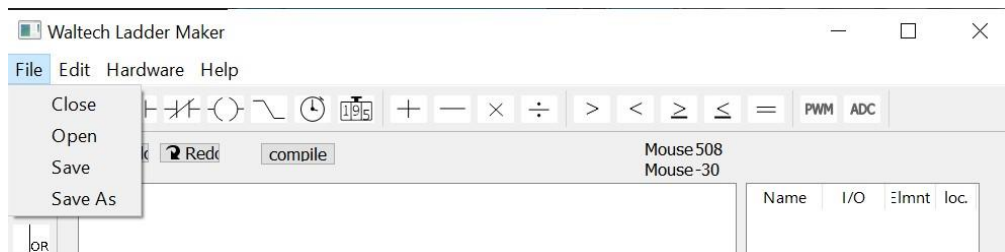


Figure 2.3: The File Menu.

### The Edit Menu:

There are just two orders in this list:

- Undo: If you made a mistake and you want to go one step back.
- Redo: If you delete something and you want to return back to it.

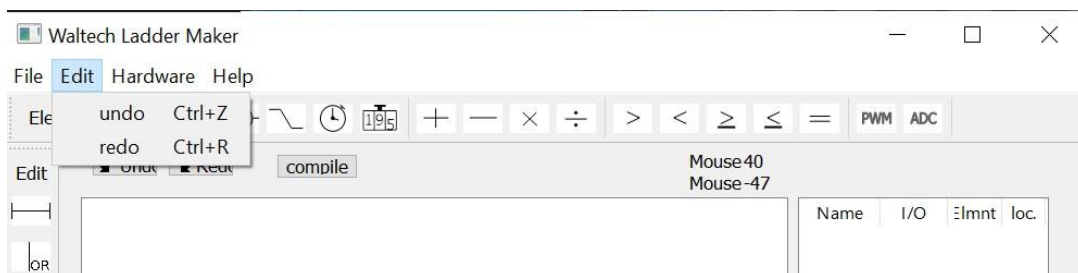


Figure 2.4: The Edit Menu.

### The Hardware Menu:

This is perhaps the most important menu for an engineer. It acts as the bridge between your computer and the physical controller.

**Controller Selection:** You use this to tell the software exactly which chip or PLC board you are using (e.g., an Arduino-based PLC or a specific Waltech model).

**Port Settings (COM Port):** This tool allows you to select the USB or Serial port where your hardware is plugged in. Without setting this correctly, the Compile & Download tool will fail to find the device.

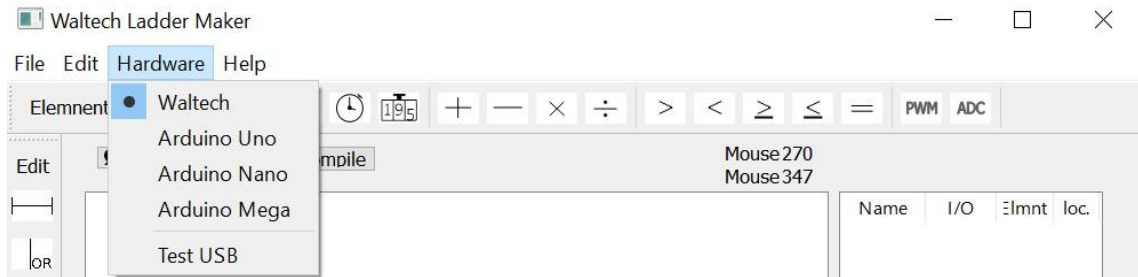


Figure 2.5: The Hardware Menu.

### The Help Menu:

This menu is your primary reference guide.

- **WhatsThis:** Provides context-sensitive help for specific buttons within the software.
- **USB:** Offers instructions on how to install drivers so your computer can talk to the Arduino boards.
- **Arduino Uno/Nano/Mega IO:** These are the most critical links. Clicking these opens the diagrams you uploaded, which act as a "Map" for your wiring.

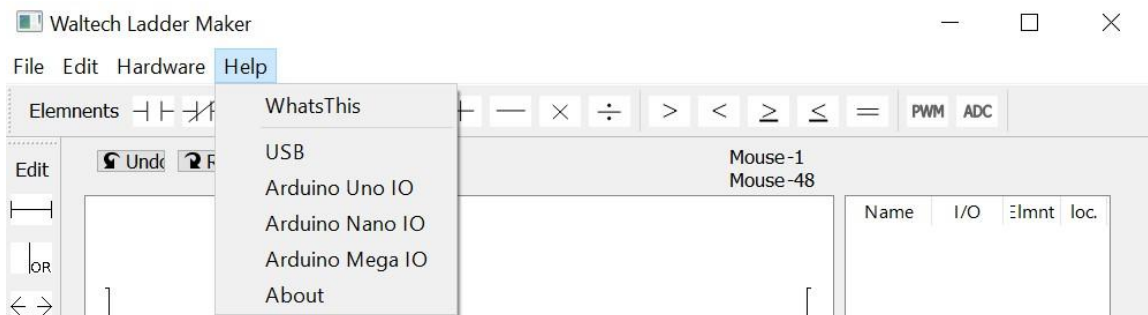


Figure 2.6: The Help Menu.

The software uses a simplified naming convention (**in**, **out**, **ADC**, **PWM**) to make programming easier. These diagrams tell you which physical pin on the Arduino corresponds to those names in your I/O Information Table.

### ❖ *Arduino Uno equivalent pins in Ladder Maker:*

The Arduino Uno in the Ladder Maker has a different arrangement than the physical entries for the Arduino, so you should read the “Help” file to find out these entries.

- ***ADC (Analog-to-Digital Converter):***

The ADC pins are responsible for bridge the gap between the physical world and digital logic. While a digital pin only understands "On" or "Off," an ADC pin can measure a range of voltage.

**How it works:** It converts a continuous voltage signal (usually 0V to 5V) into a multi-level digital number.

**Typical Use:** These are used for sensors that provide a variable signal, such as a temperature sensor, a potentiometer, or a light-dependent resistor (LDR).

- ***IN (Digital Inputs):***

The IN category refers to standard digital inputs. These are the most basic way for the controller to receive information about its environment.

**How it works:** They look for a specific voltage threshold. If the voltage is high (usually near 5V), the controller reads it as a logic **1** (TRUE). If it is low (near 0V), it reads as a logic **0** (FALSE).

**Typical Use:** These are perfect for binary devices like momentary push buttons (Start/Stop), limit switches, or proximity sensors that tell you if an object is present or absent.

- ***PWM (Pulse Width Modulation):***

PWM is a clever way to simulate an "analog" output using digital signals. Since the Arduino cannot actually output 2.5V (it can only do 0V or 5V), it uses PWM to fake it.

**How it works:** The pin switches between 0V and 5V thousands of times per second. By changing the "Duty Cycle" (the percentage of time the signal is ON vs. OFF), you can control the average power delivered to a device.

**Typical Use:** This is essential for controlling the speed of a DC motor, the brightness of an LED, or the position of a servo motor.

- ***OUT (Digital Outputs):***

The OUT category covers standard digital outputs. These are the "muscles" of your project that carry out the decisions made by your code.

**How it works:** When the program sets an output to HIGH, the pin provides 5V. When set to LOW, it connects to ground (0V).

**Typical Use:** Digital outputs are used to turn things on or off completely, such as triggering a relay to switch a high-power AC load, sounding a buzzer, or lighting up a status indicator.

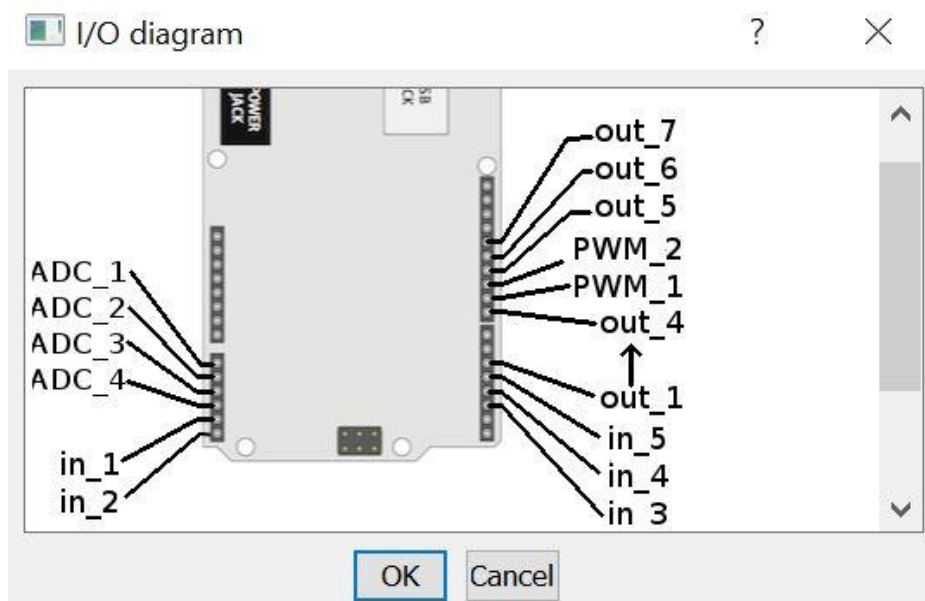
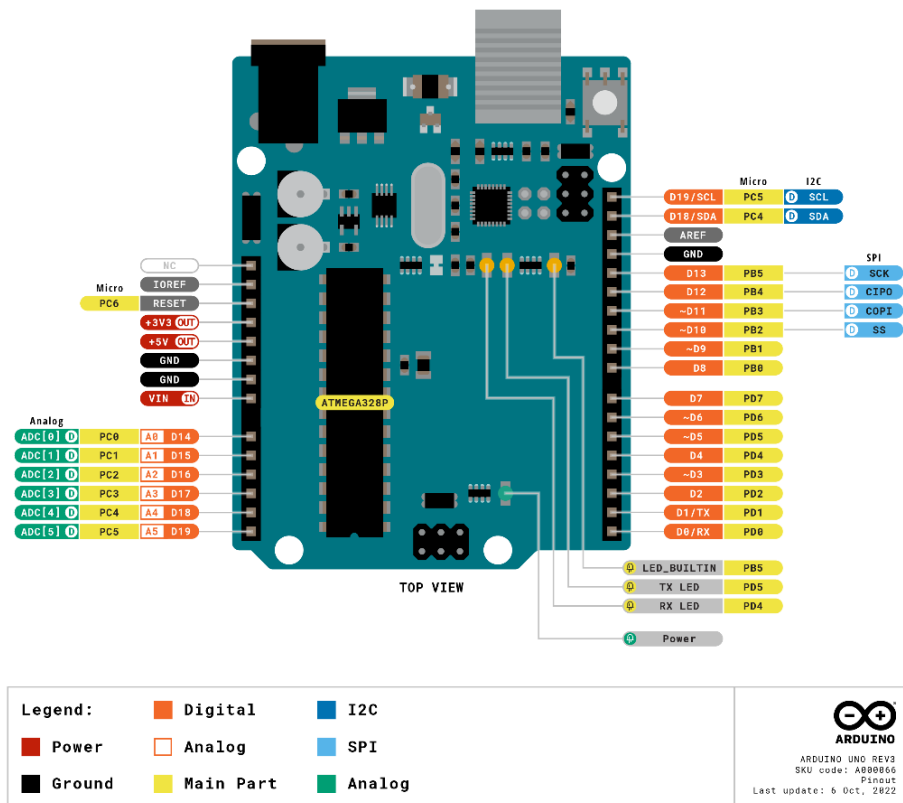


Figure 2.7: The Physical Pins and the Equivalent in Ladder Maker of Arduino Uno.

❖ *The I/O Diagram of the Arduino Uno:*

Analog Inputs (ADC): Pins A0, A1, A2, and A3.

Digital Inputs (IN): Pins A4, A5, 2, 3, and 4.

Digital Outputs (OUT): Pins 5, 6, 7, 8, 11, 12, and 13.

PWM Outputs: Pins 9 and 10.

❖ **Arduino Nano equivalent pins in Ladder Maker:**

Digital Inputs (IN): Pins A4, A5, D2, D3, and D4.

Digital Outputs (OUT): Pins D5, D6, D7, D8, D11, D12, and D13.

Analog Inputs (ADC): Pins A0, A1, A2, and A3.

PWM Outputs: Pins D9 and D10.

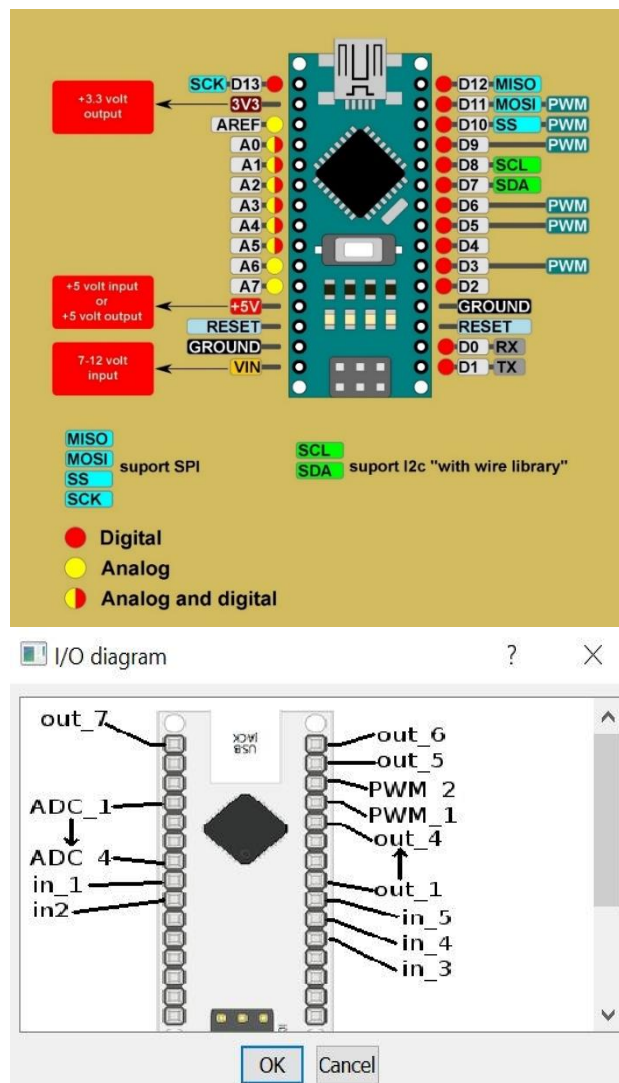


Figure 2.8: The Physical Pins and the Equivalent in Ladder Maker of Arduino Nano.

**Practical Example:** If you want a physical emergency stop button wired to the physical pin A1 on an Arduino Nano, you must look at the ArdNanoHelp.JPG map. You will see that physical pin A1 is labeled as ADC\_2. In your software, you would then set your input contact to address ADC\_2.

❖ *Arduino Mega equivalent pins in Ladder Maker:*

Analog Inputs (ADC): Pins A0 to A7.

Digital Inputs (IN): Pins A8 to A15, 20 to 33, 21, and 20.

Digital Outputs (OUT): Pins 14 to 19 and 34 to 49.

PWM Outputs: Pins 2, 3, 4, 5, 7, 11, 12, and 13.

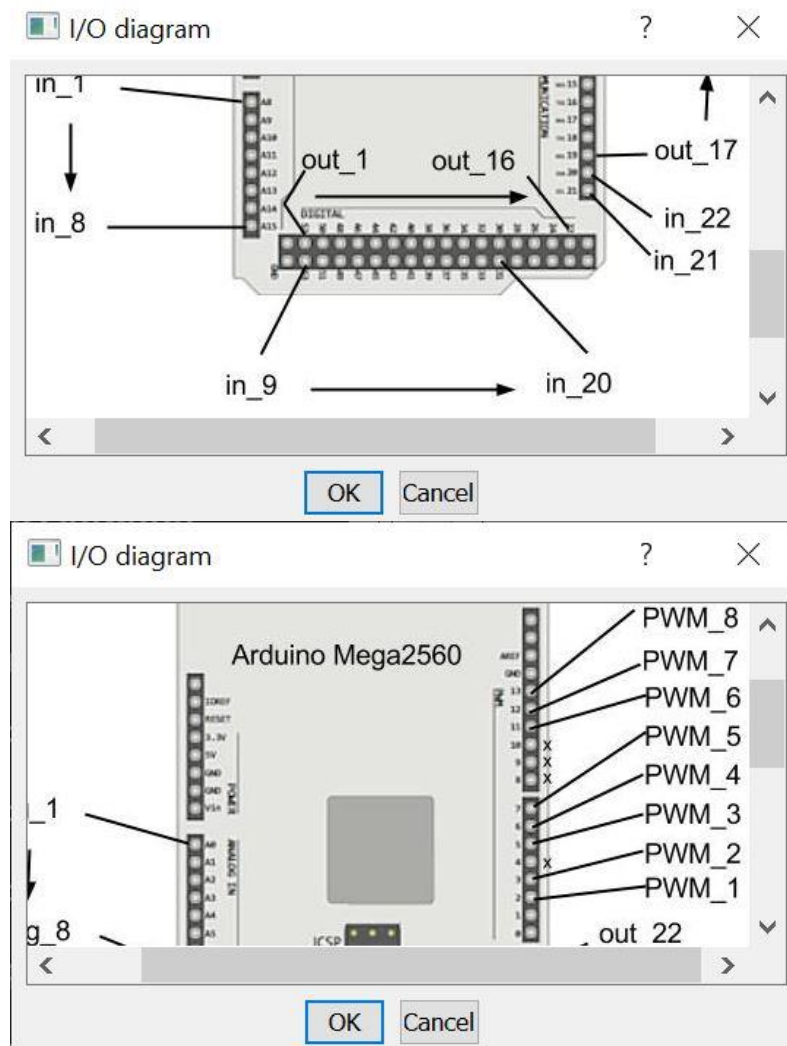


Figure 2.9: The Equivalent Pins in Ladder Maker.

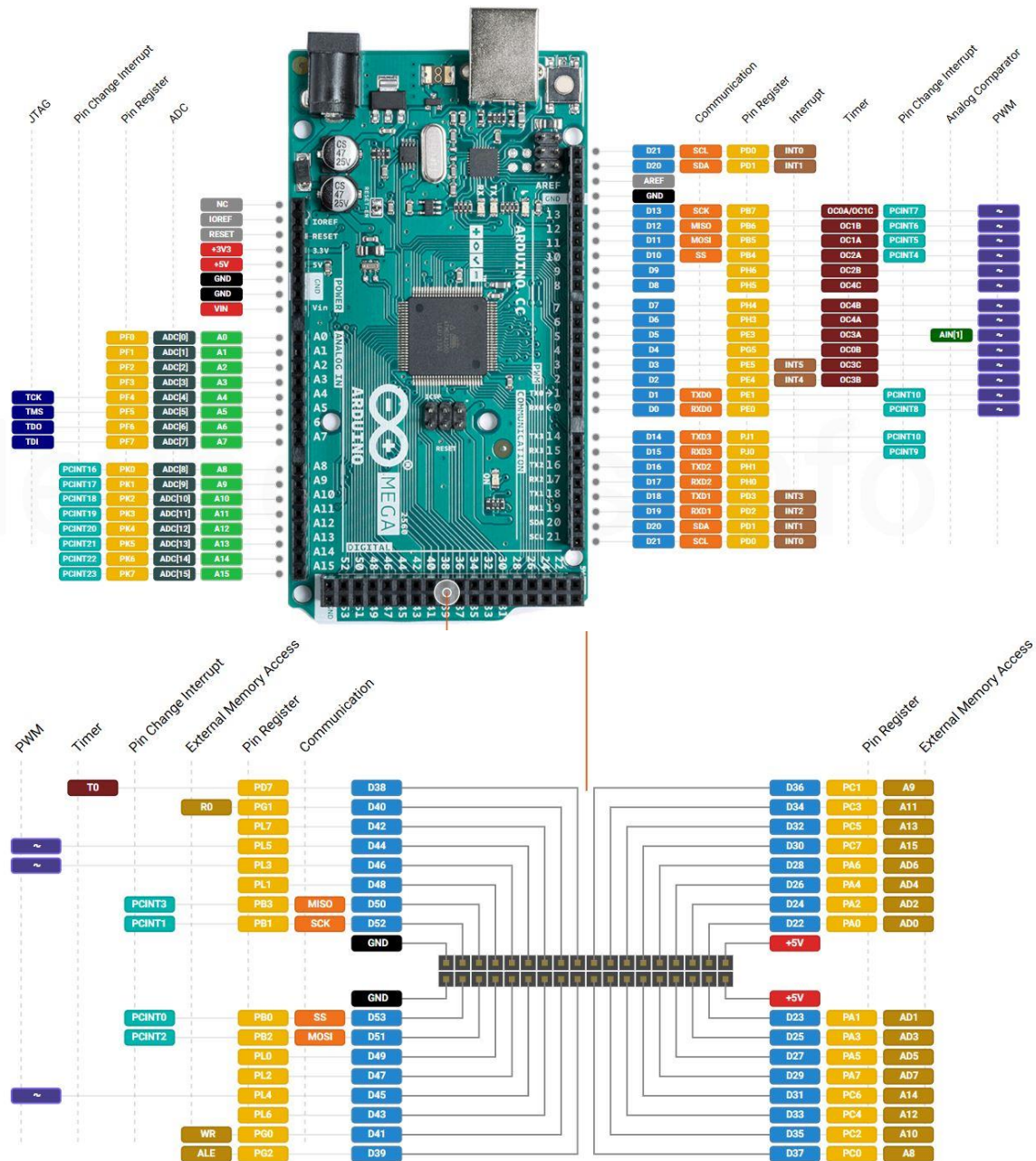


Figure 2.10: The Physical Arduino Mega Pins.

### II.2.2.3. Elements Bar:

This is your "toolbox" for logic design. It contains symbols for Normally Open and Normally Closed contacts, Output Coils, Timers, Counters, and mathematical operators.



Figure 2.11: Elements tools.

You drag these into the workspace to build your logic circuit.

### **Normally Open Contact (NO) [ -| - ]:**

This is the most basic input element, often representing a physical push-button or sensor. In its default state, it is "open," meaning it breaks the circuit and does not allow logic flow. It only becomes "true" (closed) when the associated physical input (like a Start button) is activated. In programming, this is used to say: "If this button is pressed, then continue".

### **Normally Closed Contact (NC) [ -|/- ]:**

This element works the opposite of the NO contact. In its default state, it is "closed," allowing logic to pass through. It only "opens" (breaks the circuit) when the associated input is activated. This is standard for Emergency Stop buttons or "Stop" commands because you want the circuit to remain active until someone intentionally breaks it for safety.

### **Output Coil [ -( )- ]:**

The coil represents the "load" or the device you want to turn on, such as a motor, a signaling lamp, or a solenoid valve. In the workspace, these are always placed on the far right of a rung. When the logic path to the left of the coil is complete (all contacts are closed), the coil "energizes," and the PLC sends a physical voltage to the corresponding output terminal.

### **Timers (TMR):**

Timers are used when you need a delay in your logic. There are two main types: On-Delay (TON), which waits for a set amount of time after a button is pressed before turning something on, and Off-Delay (TOF), which keeps an output running for a set time after a button is released. These are essential for processes like motor lubrication cycles or traffic lights.

### **Counters (CNT):**

Counters track how many times an event occurs. A Count-Up (CTU) element will increment its value every time a specific input (like a sensor on a conveyor belt) pulses. Once the counter reaches a "Preset" value (e.g., 10 items), it can trigger a coil to stop the machine or move to the next stage of production.

### **Mathematical & Comparison Operators (+, -, >, <):**

These tools allow the PLC to handle "Analog" data rather than just On/Off signals.

- Math: Used to calculate values, like adding two sensor readings together.

- Comparators: Used to trigger actions based on values. For example, "If the Temperature ( $>$ )  $180^{\circ}\text{C}$ , turn on the Cooling Fan."

### **PWM & ADC Tools:**

- ADC (Analog to Digital Converter): This tool reads raw voltage from a sensor (like a thermometer) and converts it into a number the program can understand.
- PWM (Pulse Width Modulation): This is used to control the speed of a motor or the brightness of a light by rapidly switching the output on and off at specific intervals.

So, the two tables below summarize for us that we have explained before:

**Table 2.1: Description of the Instructions.**

| Icon/Object                                                                         | Description                                     |
|-------------------------------------------------------------------------------------|-------------------------------------------------|
|    | Open Contact.                                   |
|    | Closed Contact.                                 |
|   | Coil.                                           |
|  | Negative Edge Detector (Falling Edge).          |
|  | BCD Timers (UINT) - Type: On Delay.             |
|  | Counters (UINT) - Type: Up Counter (Ascending). |

**Table 2.2: Arithmetic Operations, Comparisons, ADC and PWM.**

| Icon/Object                                                                         | Description                                                                 |
|-------------------------------------------------------------------------------------|-----------------------------------------------------------------------------|
|  | Addition Operation (INT).                                                   |
|  | Subtraction Operation (INT).                                                |
|  | Subtraction Operation (INT).                                                |
|  | Division Operation (INT).                                                   |
|  | Greater than (INT).                                                         |
|  | Less than (INT).                                                            |
|  | Greater or Equal than (INT).                                                |
|  | Less or Equal than (INT).                                                   |
|  | Equal to Comparison (INT).                                                  |
|  | PWM Output.                                                                 |
|  | ADC Input (Analog Input Configuration).<br>Quantization Level: 1024 Values. |

#### II.2.2.4. Editing Bar:

At the very top of this bar is the Edit button. This acts as a master switch. When active, it allows you to drag, drop, and click on the workspace to place elements. If you find you cannot move or delete a component, ensure this button is toggled on.

##### Vertical & Horizontal Rung Tools (The Lines):

The icons showing straight lines are used to create the electrical paths.

- Horizontal Lines: These extend the "rung" from the left power rail toward the output. You use these to bridge gaps between contacts.
- Vertical Lines: These are used to create Parallel Branches (OR logic).

##### The Arrow Tools (Navigation & Placement):

The Left (  ) and Right (  ) arrows are used to move the cursor or selected elements across the grid. This is helpful for precise alignment, ensuring that your contacts are placed exactly where they need to be to complete the circuit flow.

##### The "Delete" Tool (The 'X' Icon):

The red X at the bottom of the bar is the delete function. To use it, you typically select an element or a wire in the Workspace and then click the X. This is essential for cleaning up "broken" paths or removing old logic when you are troubleshooting a program.

##### The "Branch" Tool:

This tool allows you to start a new sub-path from an existing rung. In industrial automation, this is frequently used for Latching (Self-Holding) circuits. It allows the output to stay "ON" even after you release the start button by providing an alternative path for the current to flow.

Table 2.3: The Edit List Tools.

| Icon/Object                                                                         | Description                    |
|-------------------------------------------------------------------------------------|--------------------------------|
|  | Insert Segment.                |
|  | Insert parallel branch in OR.  |
|  | Increase space in segment.     |
|  | Reduce extra space in segment. |
|  | Delete element.                |

#### **II.2.2.5. Undo/Redo Functions:**

Standard quick-access buttons that allow you to step backward if you make a mistake or step forward if you accidentally undo a change.

#### **II.2.2.6. Compile Button:**

A critical tool that checks your ladder logic for errors. Once clicked, it translates your visual diagram into a binary format (machine code) that the PLC hardware can understand and execute.

#### **II.2.2.7. Workspace:**

The large central area with horizontal lines (rungs). This is where you physically "draw" your program by placing contacts on the left and coils on the right to create the flow of electricity/logic.

#### **II.2.2.8. I/O Information Table:**

A management panel that lists every Input, Output, and Memory bit used in your program. It helps you keep track of which physical pin on the PLC (like I0.1 or Q0.0) corresponds to which name (like "Button" or "Motor").

#### **II.2.2.9. Program Status & Information:**

The bottom panel provides feedback on the system's state. It displays if the PLC is connected, shows the progress of a download, and lists specific error messages if the "Compile" process fails.

### **II.3: Arduino IDE Tutorial:**

#### **II.3.1: Definition:**

Arduino IDE is an open-source Integrated Development Environment designed for programming Arduino microcontroller boards. It allows users to write, compile, and upload code known as "sketches" in a simplified version of C/C++ directly to the hardware via USB.

The IDE includes a text editor for coding, a compiler to check for errors, a message console for feedback, and tools like a serial monitor for debugging. It supports cross-platform use on Windows, macOS, and Linux, with easy board selection and library management for adding functionality like sensors or displays.

## II.3.2: Arduino IDE Tools:

The Arduino IDE has 4 basic windows for tools and controls, as shown in the image below:

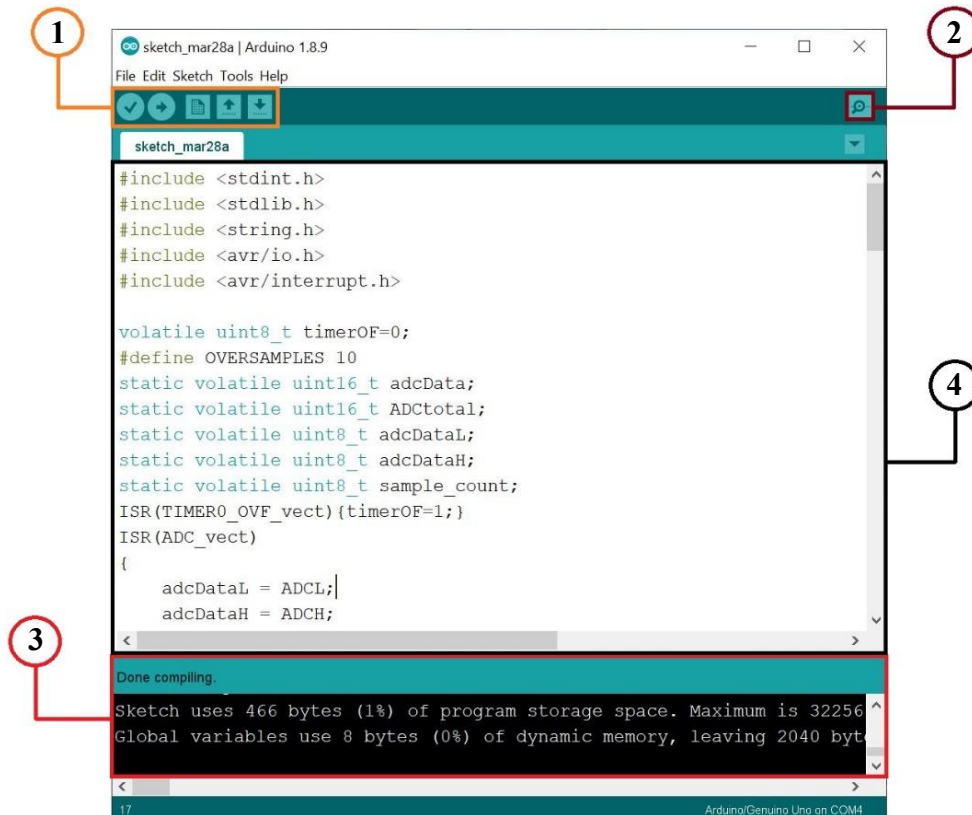


Figure 2.12: Arduino IDE Tools Explanation.

### II.3.2.1. Toolbar (Action Buttons):

This area contains the most frequently used shortcuts for managing your code. This table represent the icons:

Table 2.4: Arduino IDE Tools Explanation.

| Icons                                                                               | Description                                                                    |
|-------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|
|  | Compiles your code to check for syntax errors without sending it to the board. |
|  | Compiles the code and pushes it to your connected Arduino board.               |
|  | Creates a new blank sketch.                                                    |
|  | Opens an existing sketch from your computer.                                   |
|  | Saves the current sketch.                                                      |

### II.3.2.2. Serial Monitor:

The magnifying glass icon opens a separate window called the Serial Monitor. This is a diagnostic tool used to send and receive text data between your computer and the Arduino

board. It is essential for "debugging" because it allows you to see what the board is doing in real-time.

### **II.3.2.3. Console:**

This black area at the bottom provides feedback on the status of your actions.

**Status Bar:** The teal bar right above the black area tells you if the code is "Compiling," "Uploading," or "Done."

**Output Window:** The black text area shows detailed information, such as how much memory your program is using or specific error messages if your code fails to compile.

### **II.3.2.4. Text Editor:**

This is the main workspace where you write and edit your programming logic (the "sketch"). It features syntax highlighting, which color-codes different parts of the code (like `#include` or `volatile`) to make the structure easier to read and understand.

## **II.4. A Practical Example Illustrating How They Work Together:**

### **II.4.1. The Explanation of the Example:**

This industrial-grade interface uses galvanic isolation to safely connect 24V field signals (pushbuttons) to a 5V Arduino. Each 24V input channel uses a 6.8 k $\Omega$  current-limiting resistor and a CNY17-3 optocoupler for signal transfer, while a 10 k $\Omega$  shunt resistor and 1N4007 diode provide noise immunity and reverse polarity protection. When a button is pressed, the optocoupler's phototransistor pulls the Arduino input to ground, creating a Logic LOW state in the software.

To maintain signal stability, 10 k $\Omega$  pull-up resistors keep the Arduino pins at 5V (High) when the buttons are open. The physical wiring maps the Start button to A4 (IN1) and the Stop button to A5 (IN2), while a green LED on D13 (Out 7) serves as the logic indicator.

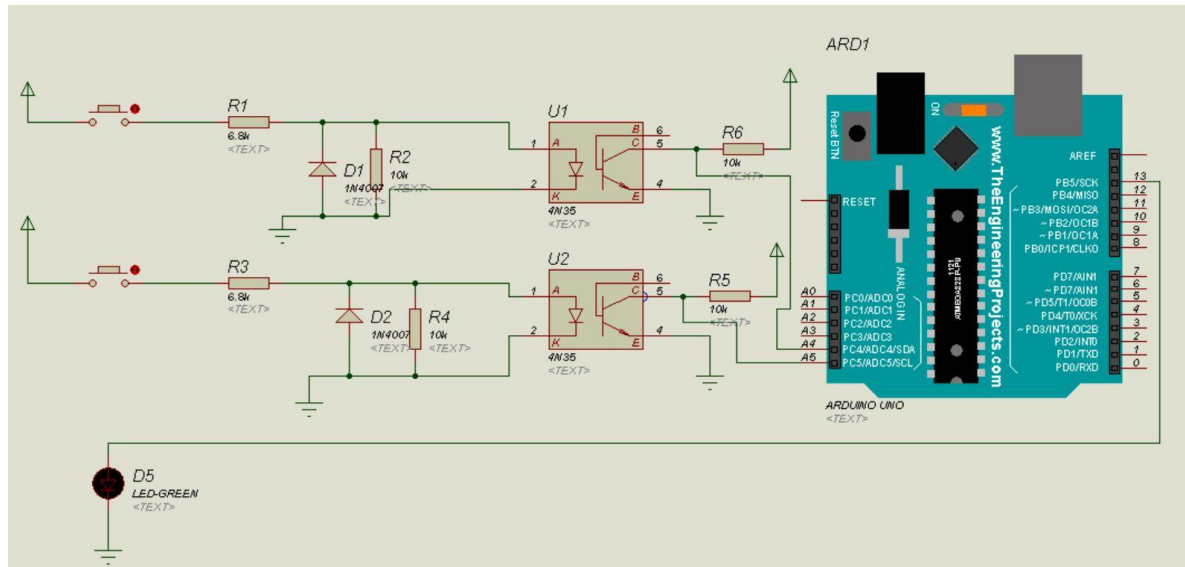


Figure 2.13: The Example of the 2 Push Buttons Controlling Digital Inputs.

## II.4.2. Ladder Maker Part:

This ladder diagram represents a sealed (latched) start–stop control circuit, a fundamental pattern in industrial automation. The rung consists of a normally closed (NC) stop contact (OFF) in series with a branch that includes a normally open (NO) start contact (ON) in parallel with an auxiliary NO contact of the output (LED), followed by the output coil. When the start button is pressed, the ON contact closes, allowing current to energize the LED coil. As soon as the output is energized, its auxiliary contact also closes, creating a parallel path that maintains current flow even after the start button is released—this is the seal-in (latching) mechanism.

To stop the system, the OFF button is pressed, which opens the NC contact and interrupts the circuit regardless of the latch state, immediately de-energizing the output. This design ensures that the system retains its ON state after a momentary start command but can always be safely stopped via the NC stop input. Functionally, it behaves as a bistable memory circuit, commonly used in applications such as motor starters and machine control, where reliable start/stop control with state retention is required.

The IN\_2 of OFF button in Ladder Maker means A5 in physical Arduino Uno Pins.

The IN\_1 of ON button in Ladder Maker means A4 in physical Arduino Uno Pins.

The OUT\_7 of LED output in Ladder Maker means D13 in physical Arduino Uno Pins.

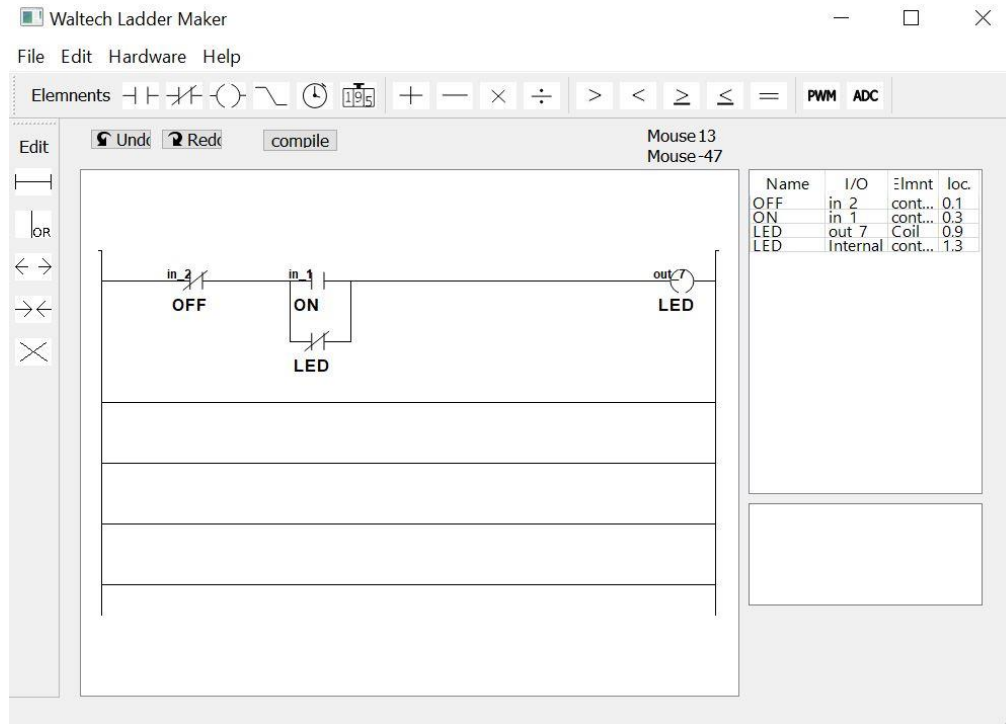


Figure 2.14: The Ladder program.

When you press the "Compile" button, it converts the program from Ladder Diagram to binary logic to facilitate reading it by the processor (Arduino), and some commands appear on the black screen, the most important of which is the program equivalent to the ladder program in C++. To be used in Arduino IDE.

❖ Why would we need a C++ program if a Ladder Diagram does the job?

We need this conversion if we want to test or simulate this program virtually before applying it in reality, for example, Proteus ISIS, the Ladder Maker application creates **(.hex)** shortcut so that you can use it in Proteus in the simulation.

This program is the equivalent of what we created in Ladder Maker, where it converts the drawn LADDER program into a C++ language program so that you can transfer it to the Arduino IDE and create a file that you can insert into Proteus to start the simulation. You cannot directly convert from Ladder Maker to Proteus.

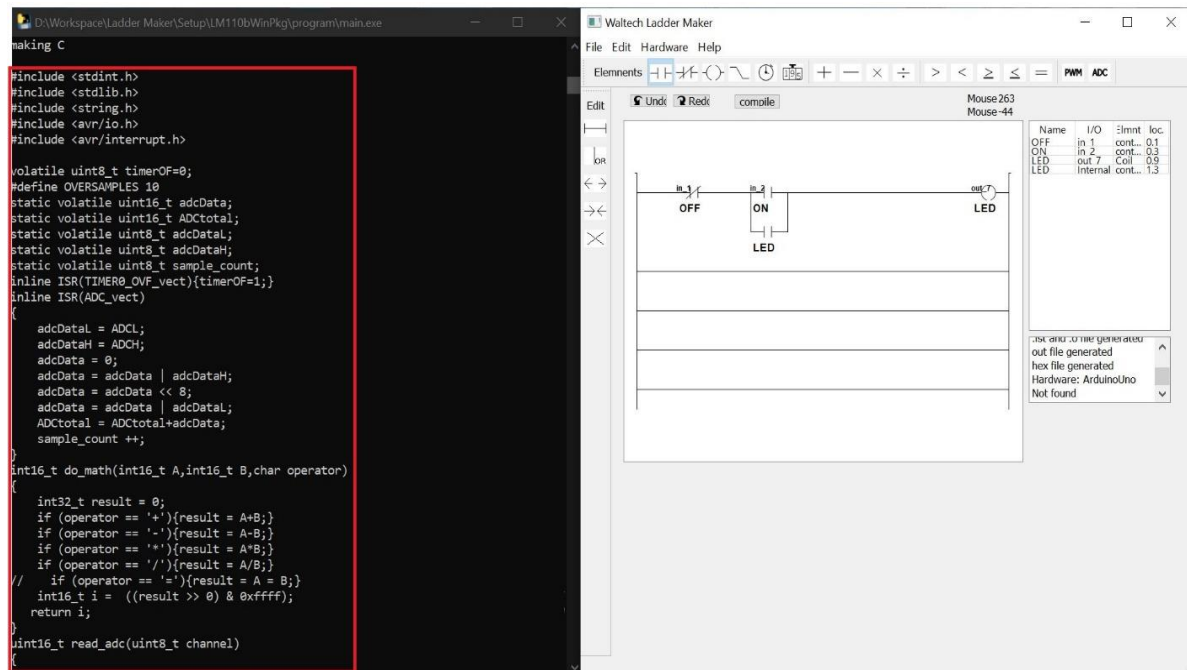
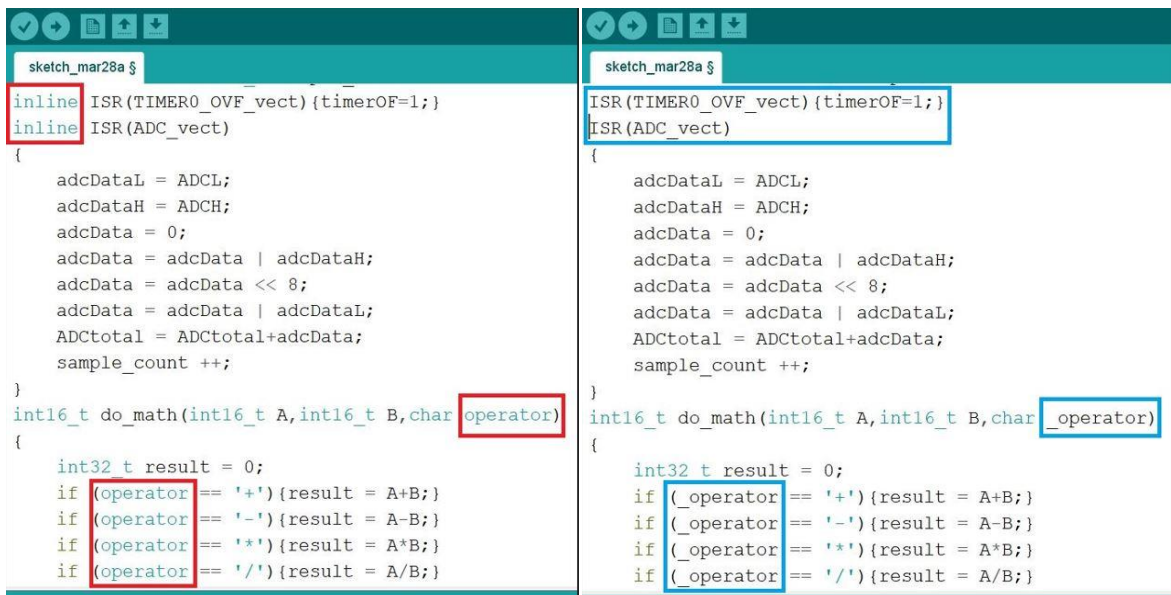


Figure 2.15: The C++ Program.

### II.4.3. Arduino IDE Part:

Now, transfer the program you copied from the console of Ladder Maker to the Arduino IDE, paste the program, and make some modifications to it as shown in the two images:



Before modifying.

After modifying.

Figure 2.16: The Overlapping Orders Between C++ and Ladder Diagram.

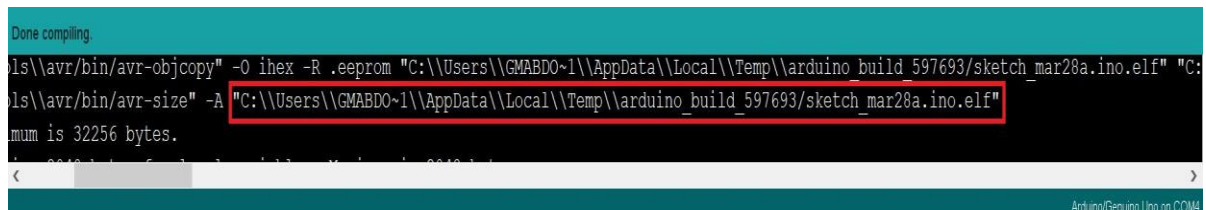
- ❖ Why are we modifying these particular orders?

The compilation errors occurred because the code generated by the Ladder Maker compiler used syntax that conflicts with the standard Arduino C++ environment. Specifically, the `inline` keyword was incorrectly applied to the **Interrupt Service Routine (ISR)**. In C++, `inline` suggests that the compiler should replace a function call with the function's actual code to save time, but an ISR is never "called" by the software; instead, it is triggered directly by hardware events like timer overflows or ADC conversions. Because the `ISR()` macro already contains specific instructions for the hardware to jump to a fixed memory address, adding `inline` creates a logical contradiction that the compiler cannot resolve.

Additionally, the variable name operator caused a conflict because it is a reserved keyword in C++ used for operator overloading. By renaming it to `(_operator)`, you provided a unique identifier that the compiler can distinguish from its internal commands. This type of manual adjustment is often necessary when moving code from specialized tools like Ladder Maker into the Arduino IDE, which requires stricter adherence to AVR libraries.

When you press 'Verify'  or 'Upload' the right-pointing arrow  the program will either begin correcting errors if you click the "verify" button, or it will transfer the program to the Arduino and delete the previous ladder logic program if you click "upload". A temporary link will appear in the "console" field, usually in the "Temp" file of the program you tested previously. This file will have the extension (.elf).

This link is what you will insert into the Arduino's Proteus to ensure the simulation works correctly.



```
Done compiling.  
ls\\avr\\bin\\avr-objcopy" -O ihex -R .eeprom "C:\\Users\\GMABDO~1\\AppData\\Local\\Temp\\arduino_build_597693\\sketch_mar28a.ino.elf" "C:  
ls\\avr\\bin\\avr-size" -A "C:\\Users\\GMABDO~1\\AppData\\Local\\Temp\\arduino_build_597693\\sketch_mar28a.ino.elf"  
imum is 32256 bytes.  
< >  
Arduino/Genuino Uno on COM4
```

Figure 2.17: The Link That Needs to be Transferred to Proteus.

### II.4.4. Proteus ISIS Part:

When we copy the link we took from the Arduino IDE console, we go to the electronic circuit that we created in order to simulate it, and we right-click twice on the Arduino. A window will open for you with the Arduino properties, and you will find a box called "Program File". We paste the link into that box.

Now let's run the emulator. The button is located at the bottom left of the screen; it looks like a black triangle pointing to the right. Pressing it will start the emulator.

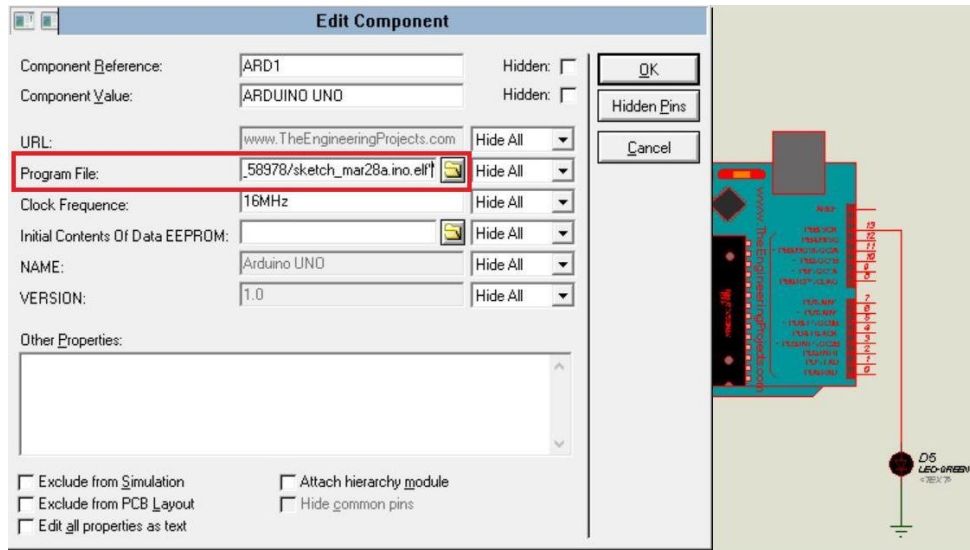


Figure 2.18: The Link in Proteus.

The simulation steps are shown in the images below in two cases: when the button is pressed and when it is released to show the "Latch" operation:

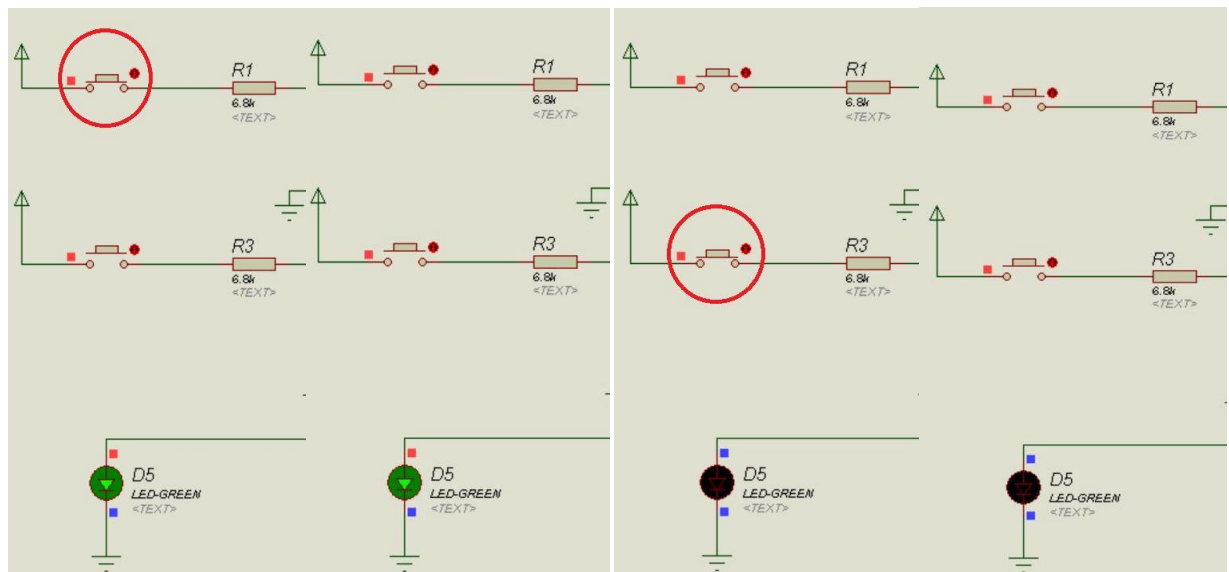
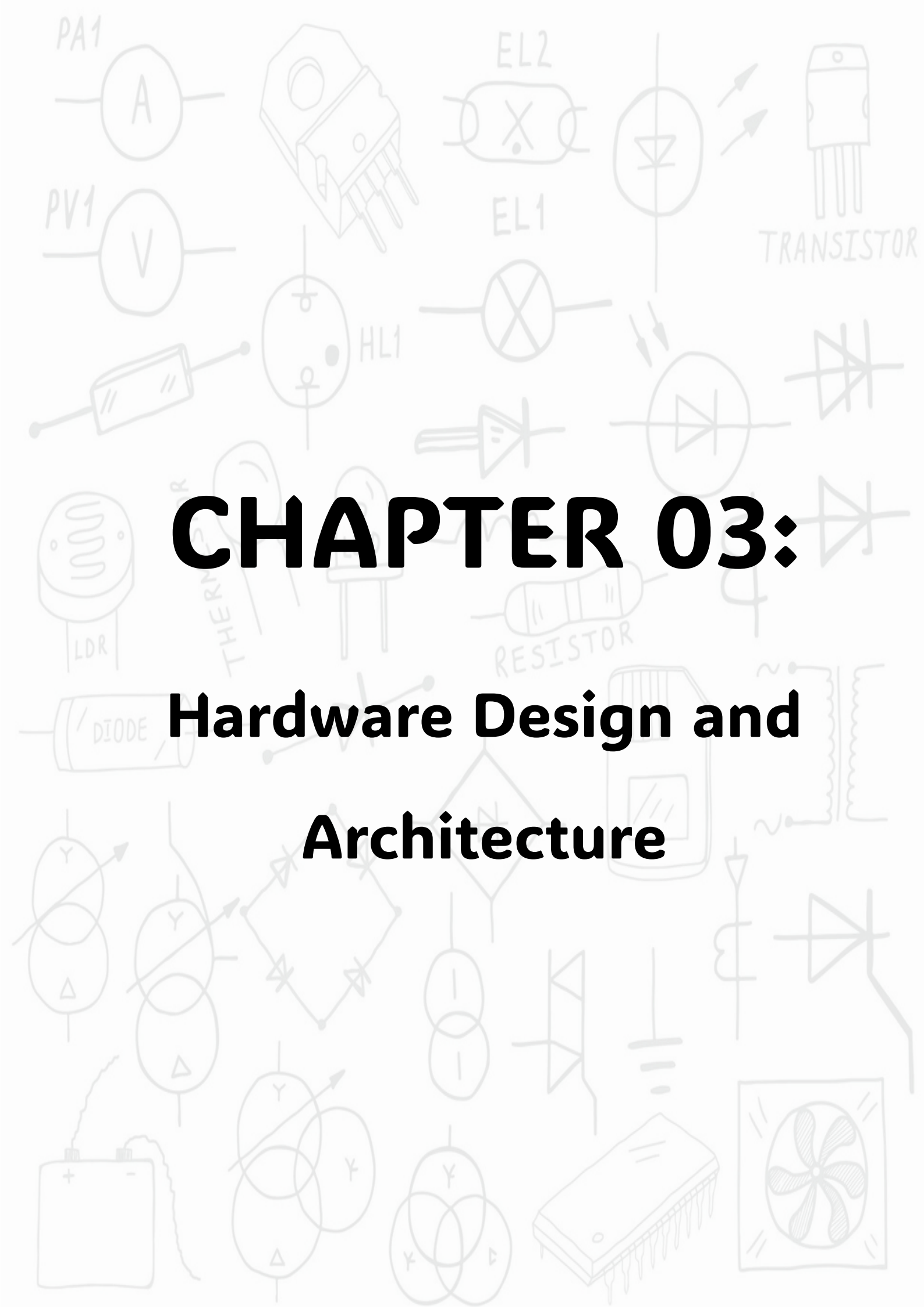


Figure 2.19: Simulation Mode (Then Press the Start and Stop Button).

## II.5. Conclusion:

By the end of this chapter, the mechanism of operation has become clear to us regarding software applications and their implementation methods using both Ladder and C++ languages, as well as simulating the project before testing it via the Proteus software.

The background is a light gray collage of various electronic symbols and components. These include: a power supply symbol (PA1), an ammeter (A), a voltmeter (V, PV1), a connector, an inductor (EL2), a capacitor (EL1), a diode, a transistor, a light bulb (HL1), a thermistor, a resistor, a diode, a light-dependent resistor (LDR), a diode, a transformer, a battery, a fan, a microcontroller, and various other electronic symbols like resistors, capacitors, and diodes. The text is centered over this collage.

# **CHAPTER 03:**

## **Hardware Design and Architecture**

## **III.1. Introduction:**

In this chapter, we will delve deeper into the design of our electronic circuit, detailing every part and its operational mechanism. We will discuss the design of the Didactic Controller and learn about its essential elements, in addition to looking into the safety security aspect.

## **III.2. Overview About the Electronic Board:**

### **III.2.1. General Explanation of The Board:**

This electronic board serves as a high-performance simulation platform for a Programmable Logic Controller (PLC), engineered to manage complex automation tasks. It integrates various input and output stages designed for industrial environments, ensuring compatibility with standard sensors and high-power actuators. The system is built to handle both digital logic and analog signal processing, providing a versatile platform for real-time control applications.

The input stage features five digital inputs configured for a 24V binary system, allowing direct interfacing with industrial switches and push buttons. To safeguard the low-voltage logic of the central controller, these inputs are isolated using CNY-17 optocouplers. This galvanic isolation prevents electrical noise and high-voltage spikes from damaging the sensitive processing unit, ensuring stable operation in electrically noisy environments.

For analog signal acquisition, the board utilizes four analog inputs routed through an LM324 operational amplifier. This IC is configured as a voltage follower (buffer) to ensure high input impedance and signal integrity. Furthermore, a resistive voltage divider is integrated to scale incoming 12V signals down to a 5V range, making them compatible with the controller's analog-to-digital converters (ADC) for precise sensor monitoring.

The board provides two PWM (Pulse Width Modulation) outputs designed for high-speed switching and variable power control. These outputs utilize IRFZ48N MOSFETs, which are driven through CNY-17 optocouplers to maintain complete isolation between the 5V logic and the high-current load side. This setup is ideal for applications requiring fine control over DC motor speeds or the dimming of high-intensity lamps without flickering.

In addition to DC control, the system features seven high-voltage AC outputs capable of driving 220VAC loads such as motors and industrial lighting. Each output is switched by a BTA06 Triac, rated for 6A and 600V. Safety is prioritized through the use of MOC3021 Opto-

Triacs, which provide the necessary isolation between the microprocessor and the AC mains power to prevent back-EMF and electrical interference.

The system's intelligence is centered around an Arduino Nano, which manages all logic and signal processing. A critical hardware feature is the inclusion of a protection jumper between the Arduino and the computer interface. This jumper prevents power conflicts between the board's internal 5V supply and the USB power from a PC, protecting both the microcontroller and the connected computer during programming and debugging phases.

Connectivity is facilitated by four dedicated terminal blocks, for the power supply of the electronic board. 5V for the power supply of the Arduino, 12V for analog inputs, 24V for digital inputs and PWM outputs loads, and 220VAC power for Triacs outputs.

The pictures below clarify the electronic board, and the (Proteus ISIS) picture of the circuit:

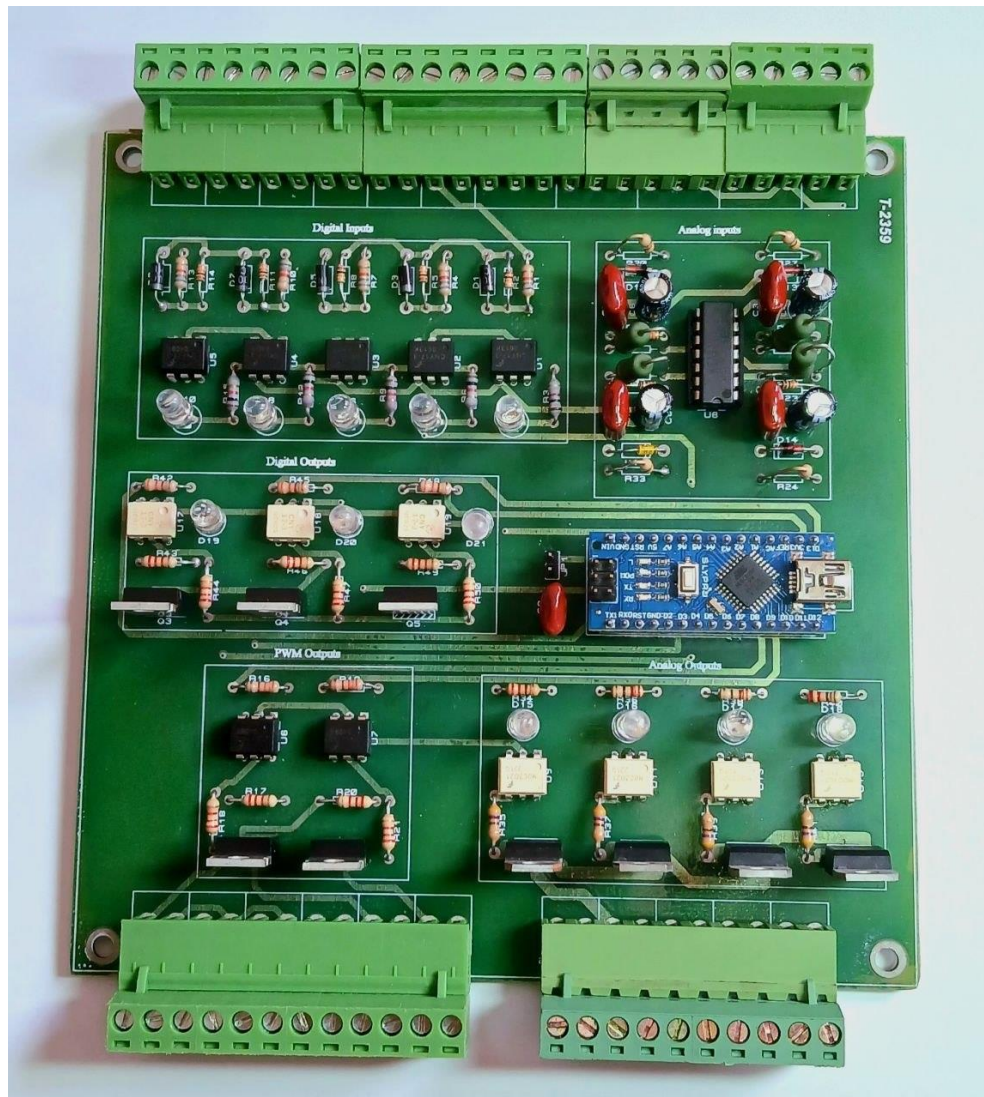


Figure 3.1: The Electronic Circuit Board.

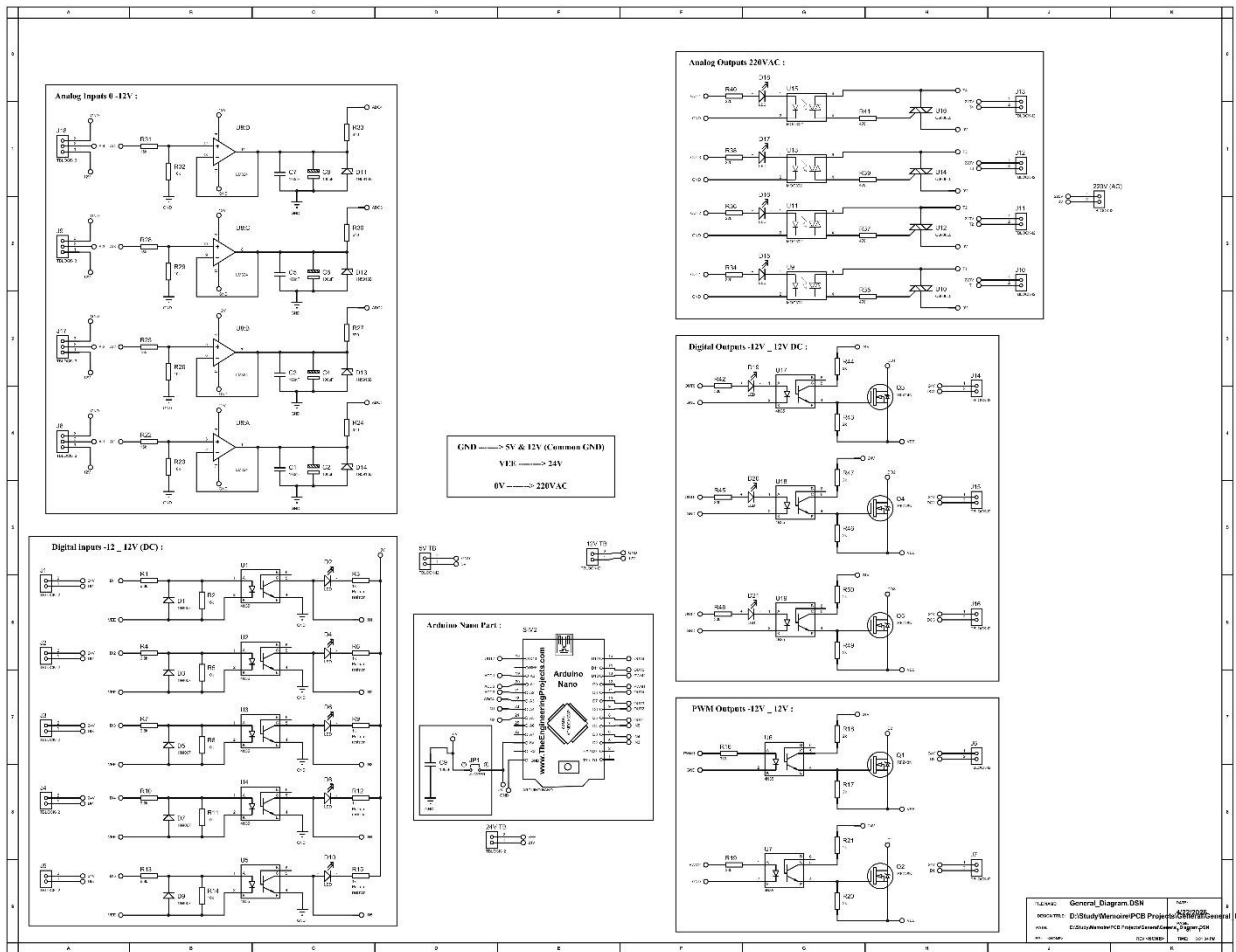


Figure 3.2: The Proteus ISIS Circuit.

## III.3. The Digital Inputs:

### III.3.1. The Installation of the Block:

The installation of this digital input circuit begins at the physical interface level, where external 24V DC field devices such as industrial sensors or push buttons are connected to the terminal pins of J1 through J5. For each channel, the positive signal wire is landed on Pin 1, while the common return is connected to Pin 2. To ensure the circuit can withstand the rigors of an industrial environment, a 1N4007 protection diode (D1) is installed in anti-parallel to the input; this serves as a critical safety measure against reverse polarity, preventing damage if the field wiring is accidentally swapped during setup.

Once the signal enters the board, it passes through a current-limiting resistor R1, which is sized to provide the correct forward current for both the status LED and the internal infrared LED of the optocoupler. The inclusion of a dedicated status LED (D2) provides an immediate visual diagnostic during the installation and testing phases; if the field device is triggered but the LED remains dark, the installer knows the issue lies in the external wiring or power

supply rather than the controller logic. Furthermore, a pull-down resistor (R2) is utilized to drain any residual voltage to ground, effectively filtering out electrical noise or "ghost" signals that could cause erratic behavior in the automation system.

The core of the installation's safety is the Optocoupler (U1–U5), which provides total galvanic isolation between the high-voltage 24V field side and the sensitive 5V microcontroller side. For the isolation to be effective, the installer must maintain a strict separation of grounds: the 24V power supply ground must remain entirely independent of the Arduino's logic ground. On the output side of the optocoupler, the phototransistor acts as a switch that pulls the microcontroller's digital input pin to a Low state when a 24V signal is present. This inverted logic ensures that the "Brain" of the system remains electrically protected from spikes, surges, and ground loops common in industrial power distribution.

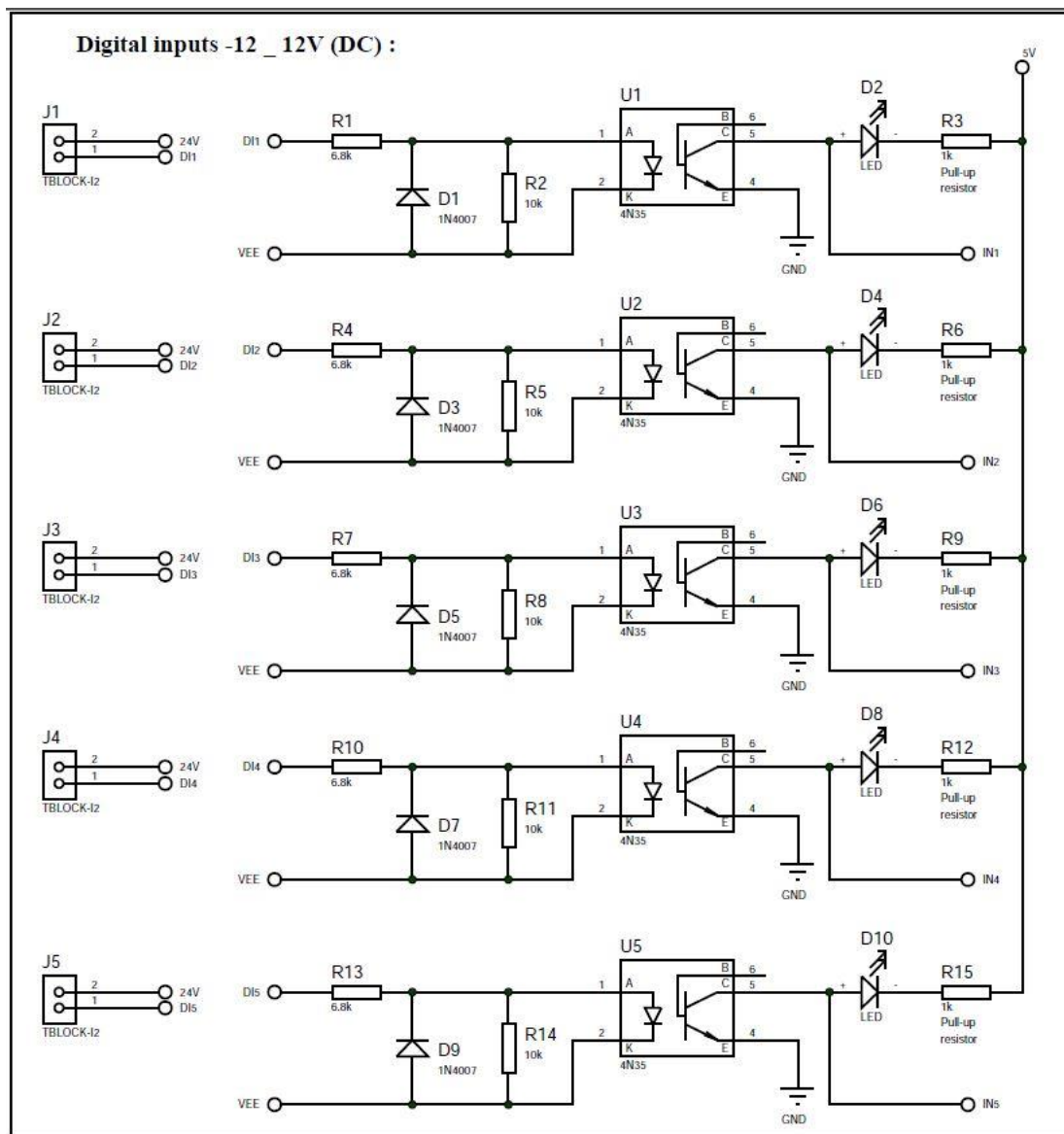


Figure 3.3: Digital Inputs Diagram (Proteus ISIS).

### III.3.2. Explain of the Components used:

#### III.3.2.1. CNY17-3 Optocoupler (The Galvanic Isolator):

This is the most important component on the board. Inside that black 6-pin chip, there is a tiny infrared LED (Pins 1 & 2) pointing at a light-sensitive switch called a phototransistor (Pins 4 & 5).

When your 24V signal powers the internal LED, it shines light across a physical gap to turn on the phototransistor on the other side. Because the signal is transferred using *light* rather than wires, there is no electrical connection between the dangerous outside world and your delicate internal computer. If a massive 100V spike hits the input block, it will destroy the optocoupler, but your expensive microcontroller on the output side will be perfectly safe.

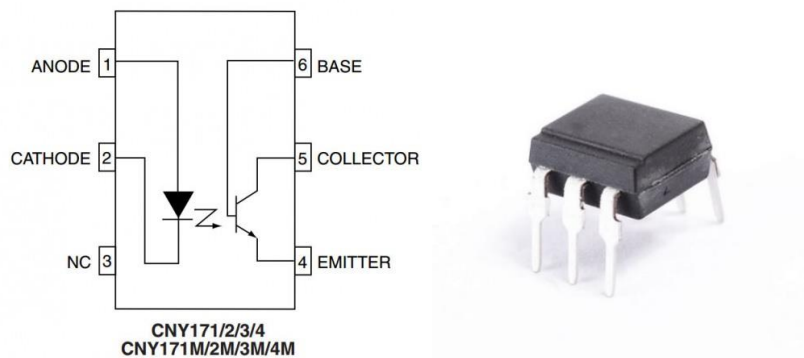


Figure 3.4: CNY17-X Optocoupler.

#### ❖ The CNY17 Optocoupler Characteristics:

Table 3.1: CNY17-3 Optocoupler Characteristics.

| Symbol    | Value       | Description                                                                                                                     |
|-----------|-------------|---------------------------------------------------------------------------------------------------------------------------------|
| $V_{ISO}$ | 5000 V      | The internal gap between the input side and the output side can withstand a massive 5000V spike without jumping across the gap. |
| CTR       | 100% – 200% | CTR is the "Gain" of the optocoupler. It is the ratio of output current ( $I_c$ ) to input current ( $I_f$ ).                   |
| $V_F$     | 1.35 V      | This is the voltage "eaten" by the internal infrared LED.                                                                       |
| $V_R$     | 6 V         | If you apply a reverse voltage higher than 6V without a protection diode, the internal LED will be destroyed.                   |
| $V_{CEO}$ | 70 V        | (Collector-Emitter Voltage), this is the maximum voltage the output "switch" can handle when it is turned off.                  |
| $I_C$     | 50 mA       | (Collector Current), this is the maximum current allowed to flow through the output side.                                       |

## III.4. Analog Inputs:

### III.4.1. The Installation of the Block:

This schematic displays a robust analog signal conditioning and protection circuit. It is specifically designed to take a higher-voltage industrial signal likely a 12V standard and scale, buffer, filter, and protect it so it can be safely read by a standard 5V logic microcontroller.

The first stage of the circuit is a passive voltage divider composed of 15k $\Omega$  and 10k $\Omega$ . Because microcontrollers operate at much lower logic levels, this divider proportionally steps down the incoming voltage. Using the voltage divider equation:

$$V_{out} = V_{in} \times \frac{R_{10k\Omega}}{R_{15k\Omega} + R_{10k\Omega}}$$

we can see how it scales. If a 24V signal is applied to the input, the voltage at the junction is:

$$V_{out} = 12V \times \frac{10k\Omega}{25k\Omega} = 4.8V$$

which equals approximately 4.8V. This maps a 0-12V industrial sensor range perfectly into a safe 0-5V window for an Analog-to-Digital Converter (ADC).

Next, the scaled signal enters an LM324 operational amplifier, which is powered directly by a 12V supply. The op-amp is configured as a voltage follower, or unity-gain buffer, meaning its output voltage exactly matches the input voltage from the divider. This stage is crucial for impedance matching. The voltage divider has a high output impedance; if connected directly to a microcontroller, the sampling mechanism of the ADC would draw current and pull the voltage down, causing inaccurate readings. The op-amp provides an exceptionally high input impedance to read the divider without altering it, and a low output impedance to drive the subsequent stages effortlessly.

Following the op-amp, the signal encounters a filtering stage consisting of two capacitors in parallel: C1 (100nF) and C2 (100 $\mu$ F). These act as a low-pass filter to ensure the microcontroller receives a clean, stable reading. The smaller 100nF ceramic or film capacitor is excellent at shunting high-frequency electromagnetic interference (EMI) to ground. The larger 100 $\mu$ F electrolytic capacitor acts as a local energy reservoir, smoothing out lower-frequency voltage ripples or sudden dips in the sensor line.

Finally, the circuit features a critical protection stage for the sensitive 5V logic hardware downstream. is a 1N5918B, which is a 5.1V Zener diode. It acts as an overvoltage clamp; if an electrical fault causes a voltage spike that exceeds the normal operating range, this diode will conduct the excess energy directly to ground, rigidly capping the output at roughly 5.1V. Paired with this is 670 $\Omega$ , a series current-limiting resistor. In the event of a fault where the Zener diode has to clamp the voltage, this resistor restricts the amount of current that can flow, preventing the diode from burning out and offering a final layer of defense for the microcontroller inputs.

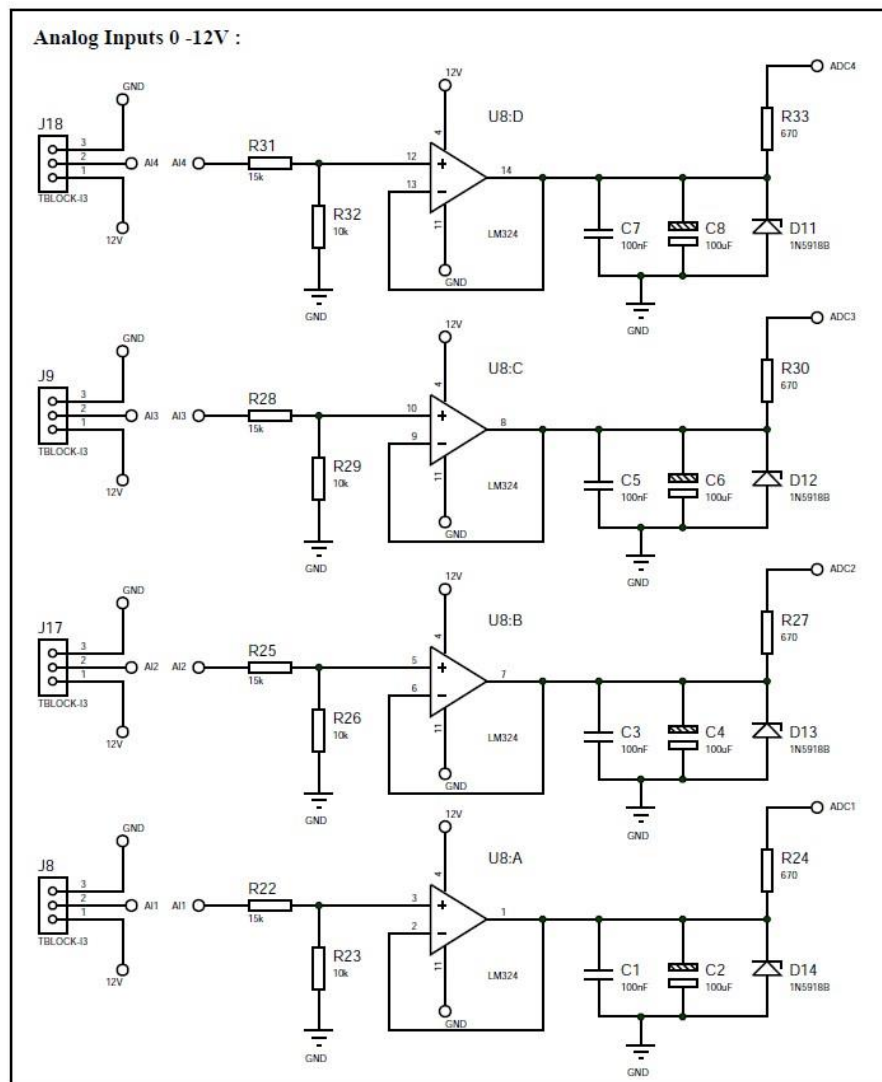


Figure 3.5: The Analog Inputs.

### III.4.2. Explain of the Components used:

#### III.4.2.1. The Operational Amplifier LM324N:

The LM324N is a quad operational amplifier IC popular for low-power analog applications. It integrates four independent op-amps in a 14-pin DIP package, offering versatility for signal processing tasks. Configured as a Voltage Follower.

Non-Inverting Input (Pin 3): Receives the scaled-down voltage from the divider.

Inverting Input (Pin 2) & Output : These are tied together in a Negative Feedback loop.

Supply Pins (4 and 11): Pin 4 is connected to +12V (VCC) and Pin 11 to Ground (GND).

This configuration is a Voltage Follower. It has a gain of 1, meaning the voltage at Pin 1 is exactly the same as Pin 3. Its primary job is Impedance Matching: it draws almost zero current from the input divider (high impedance) but can provide enough current to drive the next stage of the circuit (low impedance).

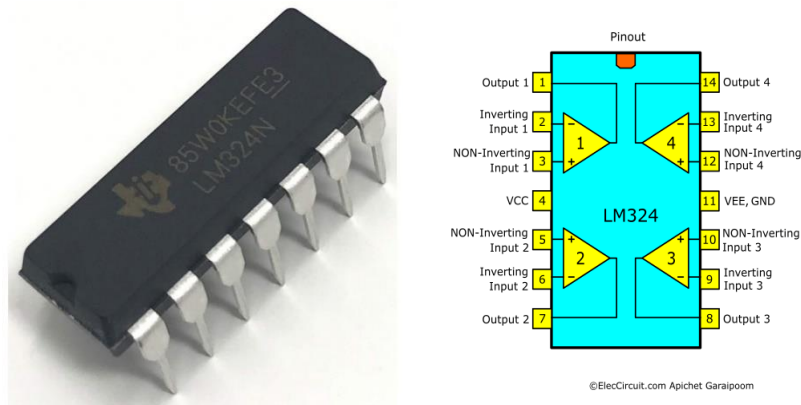


Figure 3.6: The Operational Amplifier LM324N.

❖ **The Amplifier Characteristics:**

Table 3.2: The Amplifier Characteristics.

| Specification               | Value                                              | Explanation                                                                                                                                                                                                                                |
|-----------------------------|----------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Power Supply Range</b>   | Single: 3V to 32V<br>Dual: $\pm 1.5V$ to $\pm 16V$ | The IC works with a broad array of voltages, from low-battery 3V up to high 32V without damage. Single-supply mode suits simple DC projects like sensors, while dual supplies enable precise bipolar signals in audio or measurement gear. |
| <b>Unity-Gain Bandwidth</b> | 1 MHz                                              | At 1 MHz (temperature compensated), this measures the frequency where gain drops to 1 (0 dB), indicating signal-handling speed. Temperature compensation keeps performance stable across -40°C to 85°C.                                    |
| <b>DC Voltage Gain</b>      | $\sim 100$ dB                                      | Around 100 dB (typically 100,000x amplification) refers to low-frequency open-loop gain, amplifying tiny input differences hugely.                                                                                                         |

## III.5. PWM (Pulse Width Modulation) Outputs:

### III.5.1. The Installation of the Block:

This circuit is designed to safely and efficiently control high-power loads using a low-power microcontroller. It functions through three integrated stages: Input Isolation, Voltage Level Shifting, and Power Switching.

The process begins when an Arduino Nano sends a 5V PWM (Pulse Width Modulation) signal through a  $330\Omega$  resistor. This resistor is critical because it limits the current to approximately 11mA:

$$V = R \cdot I \rightarrow I = \frac{V}{R} = \frac{5V}{330\Omega} = 0.011A = 11mA$$

which is the perfect amount to power the internal infrared LED of the CNY17-3 optocoupler without damaging the component or overworking the Arduino. The resistor value is determined by accounting for the LED's Forward Voltage  $V_{DiodeLED}$  of roughly 1.35V using the formula:

$$R = \frac{V_{Source} - V_{DiodeLED}}{I_{Desired}} = \frac{5V - 1.35V}{0.011A} = 331.81\Omega \approx 330\Omega$$

By using light to trigger a photo-transistor on the other side of a "safety gap," the optocoupler ensures there is no direct electrical connection between the input and output. This isolation is vital for protecting the controller from electrical noise and ground loops.

Once the optocoupler is active, it bridges the 24V supply to a control network consisting of two  $2k\Omega$  resistors. These resistors are configured as a voltage divider, which serves to reduce the 24V supply by half to reach 12V, as calculated by:

$$V_{Out} = V_{In} \times \frac{R_1}{R_1 + R_2} = 24V \times \frac{2k\Omega}{4k\Omega} = 12V$$

This 12V signal is the ideal voltage for fully opening the gate of an IRFZ48N MOSFET, which typically operates best between 10V and 12V. The upper resistor regulates the charging current and switching speed, while the lower pull-down resistor ensures the MOSFET turns off completely by discharging gate capacitance and preventing noise-induced activations. The  $2k\Omega$  value is specifically chosen to provide a balance between fast switching and high noise immunity.

In the final stage, the IRFZ48N MOSFET acts as a "digital faucet" for the high-power side of the circuit. When the gate voltage rises, the MOSFET turns "ON," completing the circuit for any load connected to the J6 or J7 connectors and allowing current to flow to the ground. When the input signal from the Arduino ceases, the optocoupler turns off, the gate voltage drops, and the MOSFET immediately cuts off power to the load. This setup allows for stable, high-speed switching of 24V loads while keeping the sensitive control logic completely isolated and safe.

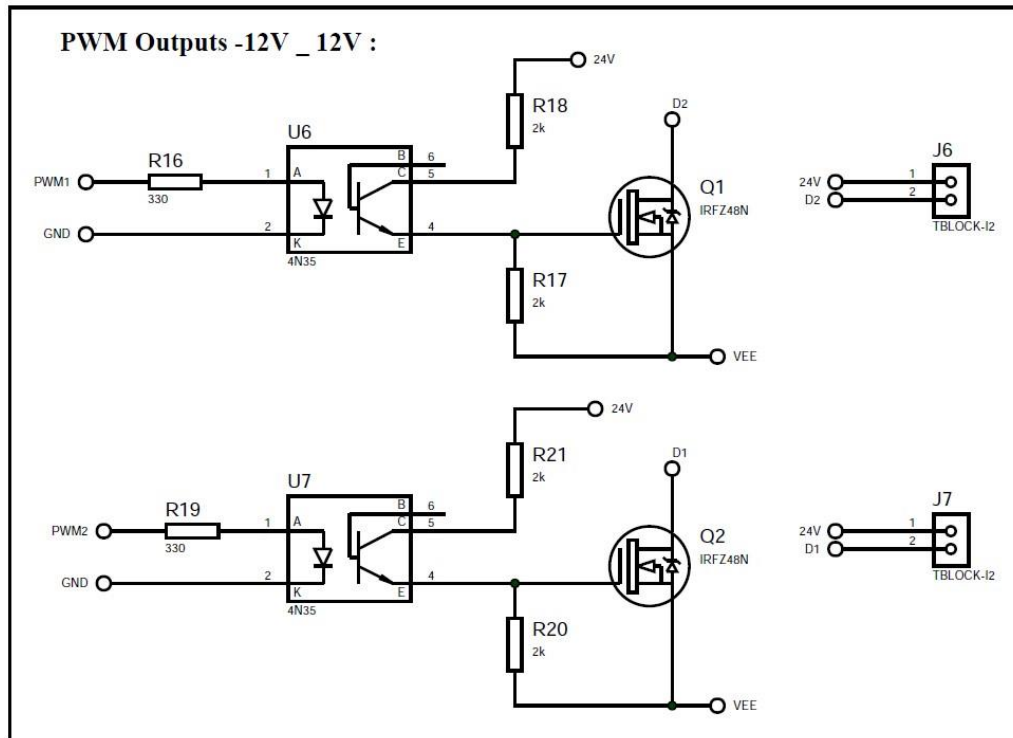


Figure 3.7: PWM Outputs.

### **III.5.2. Explain of the Components used:**

#### **III.5.2.1. The IRFZ48N MOSFET:**

The IRFZ48N is an N-channel MOSFET used in circuits as a high-current switch. in this circuit operates as a low-side power switch controlled by a PWM signal through an optocoupler. It enables efficient control of a 24V load by rapidly switching current on and off, allowing regulation of power delivered to the load while ensuring electrical isolation and protection of the control stage.

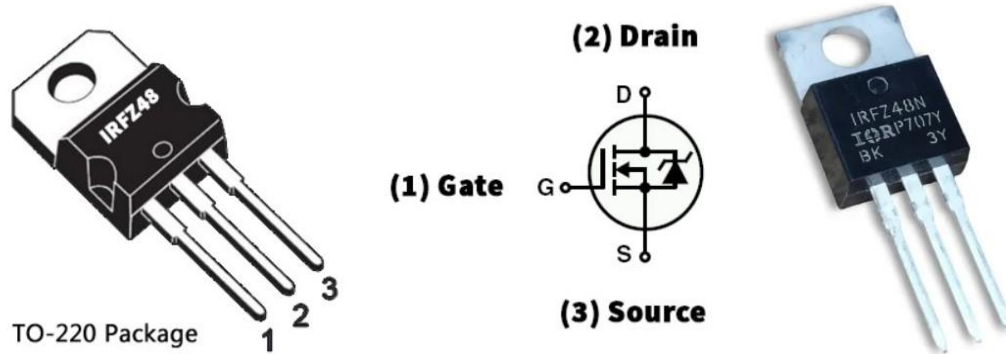


Figure 3.8: IRFZ48N MOSFET.

❖ *The IRFZ48N MOSFET Characteristics:*

Table 3.3: The IRFZ48N MOSFET Characteristics.

| Symbol       | Parameter                          | Value     |
|--------------|------------------------------------|-----------|
| $V_{ds}$     | Drain-Source Voltage.              | 55V       |
| $I_d$        | Continuous Drain Current.          | 49A - 64A |
| $R_{ds(on)}$ | Static Drain-Source On-Resistance. | 17.5 mΩ   |
| $V_{gs}$     | Gate-Source Voltage.               | ±20V      |
| $P_d$        | Power Dissipation.                 | 94W       |
| $T_j$        | Operating Junction Temperature.    | 175°C     |

## III.6. The Digital Outputs:

### III.6.1. The Installation of the Block:

The provided schematic details a robust, galvanically isolated digital output stage, specifically engineered to interface low-voltage control logic such as an Arduino Nano with high-power 24V industrial loads. The circuit features three identical channels, each functioning as a protected low-side switch capable of driving demanding industrial actuators while maintaining strict separation from the sensitive control electronics.

At the input stage, the logic signal from the microcontroller is routed through a 330Ω current-limiting resistor and a diagnostic LED, which provides immediate visual feedback of the channel's operational state. This signal then energizes the internal LED of a 4N35 optocoupler. The primary function of this optocoupler is to establish a physical galvanic barrier. By transmitting the control signal optically rather than electrically, the circuit

effectively insulates the microcontroller from the severe voltage transients, inductive spikes, and ground loops that are prevalent in harsh industrial environments.

The high-power side of the circuit incorporates a carefully designed gate drive network to safely operate the IRFZ48N N-channel MOSFET. While the MOSFET requires a sufficient gate voltage to fully saturate and conduct efficiently, applying a full 24V differential directly to the gate would exceed its absolute maximum Gate-to-Source voltage rating, leading to immediate component failure. To resolve this, the 4N35's phototransistor is paired with two 2k $\Omega$  resistors arranged as a voltage divider. When the optocoupler is activated, this network divides the 24V potential difference between the upper supply rail and the VEE rail in half, delivering a perfectly safe and optimal 12V drive directly to the MOSFET gate. Conversely, when the control signal is removed, the lower resistor securely pulls the gate down to the VEE potential, ensuring the MOSFET remains completely switched off and immune to parasitic turn-on.

In its final operational state, the MOSFET acts as a highly efficient low-side switch. Once the optimal gate voltage is applied, the IRFZ48N conducts, completing the circuit by pulling the load's return path down to the VEE rail. This architecture allows current to flow smoothly from the 24V supply, through the external industrial device connected at the terminal block, and safely to the lowest potential.

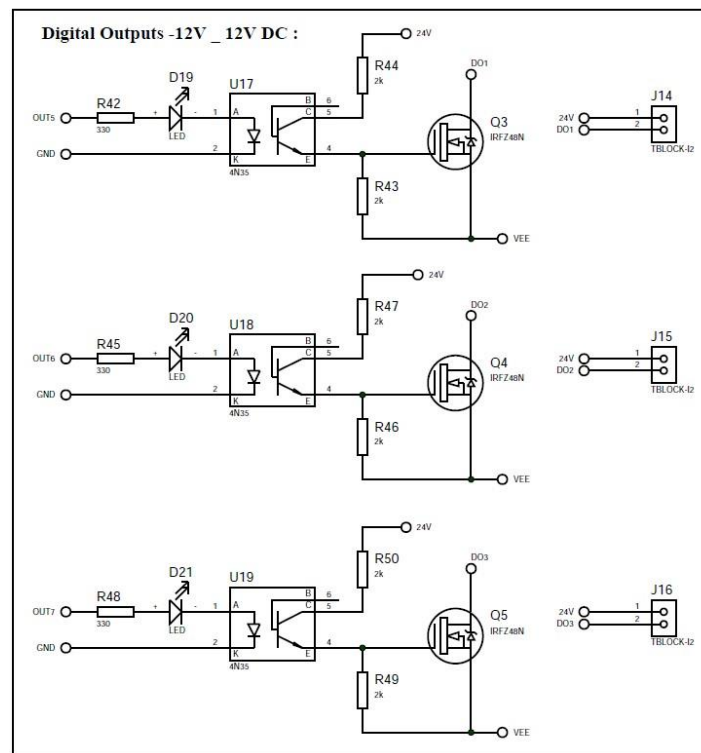


Figure 3.9: The Digital Outputs.

## III.7. The (220VAC) Analog Outputs:

### III.7.1. The Installation of the Block:

The input stage utilizes a 220Ω resistor as a current limiter for both the external indicator LED and the internal LED of the MOC3021. To ensure reliable triggering without burning out the components, the forward current ( $I_f$ ) is calculated by subtracting the voltage drops of both LEDs from the input supply. For a standard 5V logic signal, the calculation is:

$$R_{220\Omega} = \frac{V_{CC} - V_{f(LED)} - V_{f(MOC)}}{I_f} = \frac{5V - 2V - 1.2V}{0.0081A} = 222.22\Omega \approx 220\Omega$$

With typical drops of 2V and 1.2V respectively, this results in approximately 8.1mA, providing sufficient current to trigger the optoisolator (which generally requires 5mA to 15mA).

$$I_f = \frac{V_{CC} - V_{LED} - V_{MOC}}{R_{220\Omega}} = \frac{5V - 2V - 1.2V}{220\Omega} = 0.0081A = 8.1mA$$

On the output side, a 470Ω resistor ( $R_2$ ) limits the gate surge current to protect the internal triac of the MOC3021 and the gate of the BTA06. This value is critical for safety during "worst-case" scenarios, such as when the circuit is triggered at the peak of the AC sine wave. For a 220V AC system, where the peak voltage ( $V_{Peak}$ ) is approximately 311V, the maximum surge current is determined by:

$$R_{470\Omega} = \frac{V_{Peak}}{I_{Peak}} = \frac{220V \times \sqrt{2}}{0.66A} = \frac{311V}{0.66A} = 471.40\Omega \approx 470\Omega$$

Using the 470Ω resistor, the peak current is limited to roughly 0.66A. This remains well under the 1A absolute maximum rating for the MOC3021, ensuring the longevity of the components even under maximum electrical stress.

$$I_{Peak} = \frac{V_{Peak}}{R_{470\Omega}} = \frac{220V \times \sqrt{2}}{470\Omega} = \frac{311V}{470\Omega} = 0.66A \ll 1A (MOC3021)$$

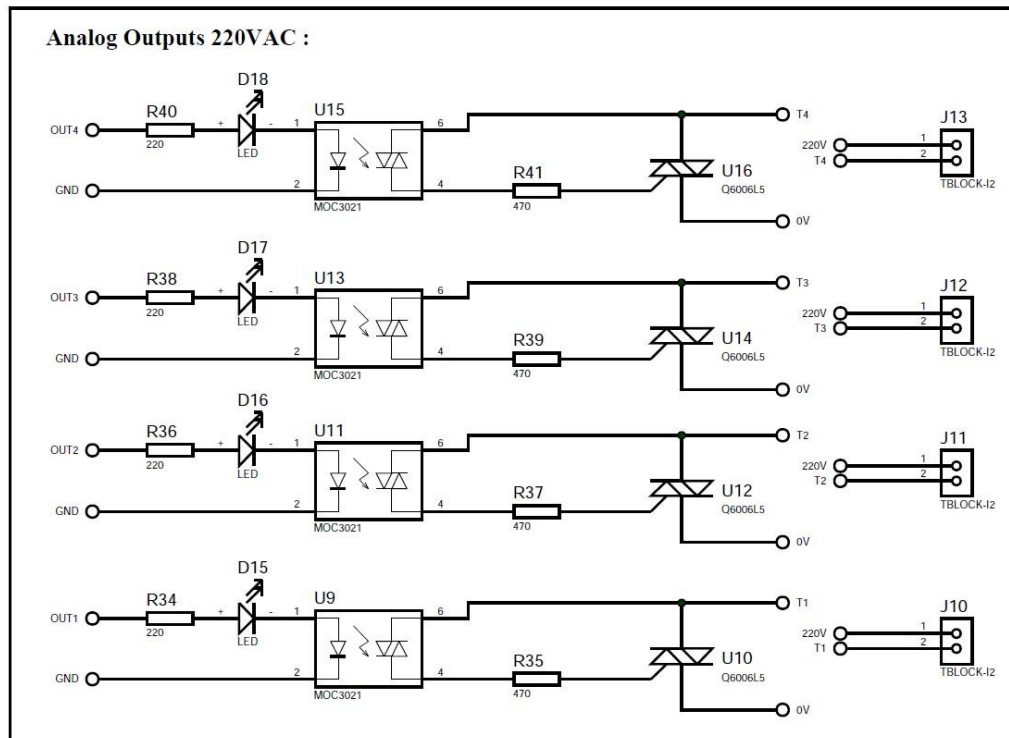


Figure 3.10: 220VAC Analog Outputs.

### III.7.2. Explain of the Components used:

#### III.7.2.1. The Opto-Triac MOC3021:

it's an optoisolator with a random-phase phototriac output. It contains an infrared LED on the input side (pins 1 and 2) and a small, light-sensitive triac on the output side (pins 4 and 6). When the internal LED is energized, its light triggers the internal phototriac to conduct. This component provides the critical safety barrier between the low-voltage DC input and the high-voltage AC output.

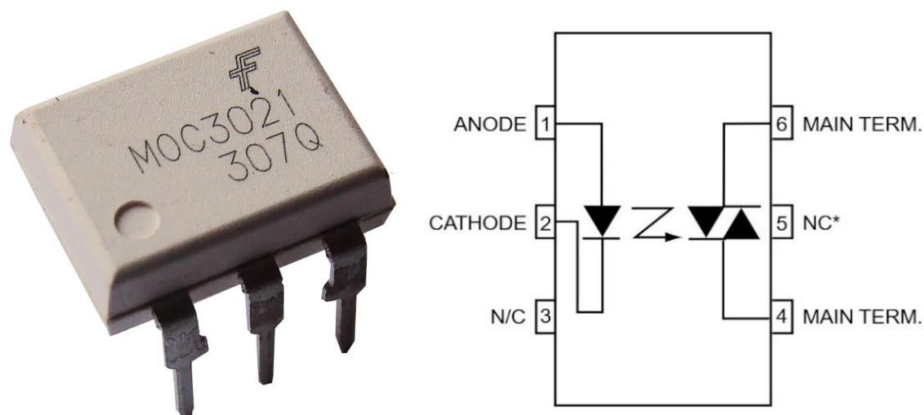


Figure 3.11: MOC3021 Opto-Triac.

#### ❖ The MOC3021 Opto-Triac Characteristics:

Table 3.4: MOC3021 Opto-Triac Characteristics.

| Characteristic        | Specification                     | Description                                                                                                                       |
|-----------------------|-----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Galvanic Isolation    | 5000V (up to 7500V peak).         | Physically separates delicate 5V logic from the mains, preventing catastrophic microcontroller damage from AC voltage spikes.     |
| Input Trigger Current | 15mA maximum.                     | Can be driven directly by standard logic pins using a simple current-limiting resistor.                                           |
| Trigger Type          | Non-Zero Crossing (Random Phase). | Conducts the instant the LED is powered. This is mandatory if the application requires phase-angle control.                       |
| Off-State Voltage     | 400V                              | Provides enough overhead to safely block voltage on standard 120V and 240V AC lines when the circuit is off.                      |
| Output Current        | 100mA peak                        | Cannot power industrial loads directly; its sole purpose is to route a tiny amount of AC current to trigger a larger power triac. |

### III.7.2.2. The Triac BTA06:

is the main power Triac, which serves as the heavy-duty AC switch. Unlike a relay, it has no moving parts. When it receives a small burst of current into its gate terminal (triggered by MOC3021 and limited by 470 $\Omega$  resistor), it turns fully on, allowing heavy AC current to flow between its main terminals and power the connected load.



Figure 3.12: BTA06 Triac Picture and Pinout.

#### ❖ The Triac BTA06 Characteristics:

Table 3.5: BTA06-600 Triac Characteristics.

| Parameter   | Maximum Rating |
|-------------|----------------|
| Voltage     | 600V           |
| Current     | 6A             |
| Temperature | 125°C          |

## III.8. Arduino Nano Part:

### III.8.1. The Installation of the Block:

This schematic represents the central processing unit of our hardware project, specifically utilizing an Arduino Nano model designed to control our electronic board.

Rather than just showing a standalone Arduino, this diagram illustrates how the Nano's standard pins are mapped to the specific inputs, outputs, and power lines of the custom circuit board surrounding it. This diagram is based on **Ladder Maker Software Pinout** to facilitate the programming of the Arduino Nano in the software (Ladder Maker).

Here is a breakdown of what this schematic is doing:

#### III.8.1.1. The I/O Mapping (Input/Output Allocation):

The labels extending from the Arduino pins indicate how the microcontroller interacts with the rest of your system. It is organized much like a custom Programmable Logic Controller (PLC) interface:

**Digital Inputs (IN1 to IN5):** Standard Arduino digital and analog pins (like D2, D3, D4, A4, and A5) are configured to read digital signals from external sources. These would typically connect to your input conditioning circuits (e.g., optocouplers) to read external push buttons, limit switches, or sensors.

**Digital Outputs (OUT1 to OUT7):** Pins like D5, D6, D7, D8, D11, D12, and D13 are dedicated to sending control signals out. These would drive external loads, likely passing through MOSFETs, Relays, Triacs, or output opto-isolators to control industrial actuators or indicators.

**Analog Inputs (ADC1 to ADC4):** Pins A0 through A3 are reserved for Analog-to-Digital Conversion. These are used to read continuous variable signals, such as temperature sensors, potentiometers, or industrial analog signals.

**PWM Outputs (PWM1, PWM2):** Pins D9 and D10 are allocated for Pulse Width Modulation. This allows for analog-like control, which is essential for adjusting motor speeds, dimming lights, or providing proportional control to valves.

### III.8.1.2. The Power Conditioning Circuit:

The smaller circuit outlined in the bottom left is dedicated to power stability and routing for the Arduino's 5V line.

**Capacitor C9 (100nF):** This is a decoupling (or bypass) capacitor. It is connected between the 5V power rail (the upward arrow) and Ground. Its job is to filter out high-frequency electrical noise and voltage spikes from the power supply. In environments dealing with relays, motors, or switching regulators, this capacitor ensures the microcontroller receives a clean, stable 5V signal so it doesn't reset or behave erratically.

**Jumper JP1:** This jumper acts as a switch between the main circuit board's 5V power supply and the Arduino Nano's 5V pin.

When closed (connected): The main board provides power to the Arduino Nano.

When open (disconnected): The Arduino is isolated from the board's power. This is highly useful when you need to plug a USB cable into the Nano to upload new code. Opening the jumper prevents the USB's 5V power from clashing with the custom board's power supply, protecting both your computer and the circuit.

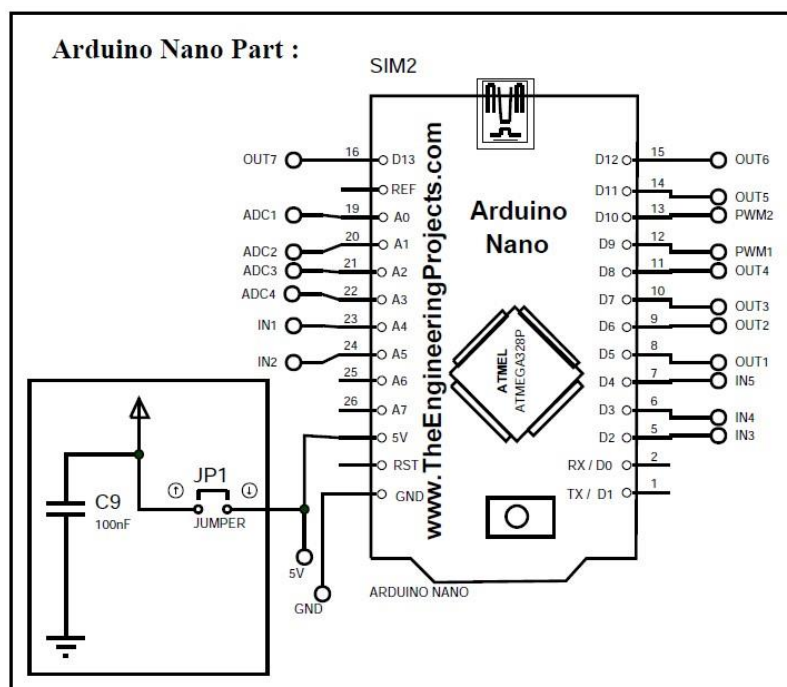


Figure 3.13: Arduino Nano Pinout and the Power Section.

### **III.8.2. Explain of the Components used:**

#### **III.8.2.1. The Arduino Nano:**

The Arduino Nano is the main controller of the circuit. It contains the ATmega328P microcontroller, which runs the program and manages all the input and output pins. In this schematic, the pins are assigned custom project labels: ADC1 to ADC4 are analog inputs, IN1 to IN5 are digital inputs, OUT1 to OUT7 are digital outputs, and PWM1 and PWM2 are PWM outputs used for variable control such as dimming or speed regulation. The RX and TX pins are used for serial communication, while 5V, GND, and RST handle power, ground, and reset functions.

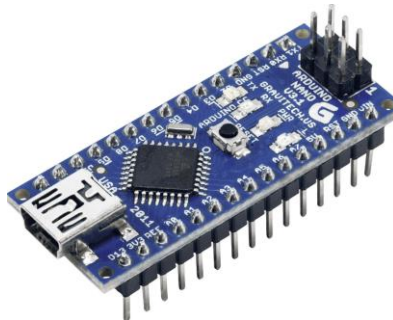


Figure 3.14: Arduino Nano.

### **III.9. The Power Terminal Blocks:**

The four terminal blocks represent the various power rails required for a multi-voltage system, likely bridging the gap between low-power logic and industrial-level actuation. Each block utilizes a standard 2-pin connector to interface different voltage levels, ranging from sensitive 5V DC microelectronics to high-voltage 220V AC mains.

The first two blocks handle the primary DC supply for the system's control logic and peripheral components. The 5V DC block serves as the VCC rail, typically used to power microcontrollers like an Arduino Nano and low-power sensors.

The 12V DC block provides a higher potential often required for cooling fans, relays, or specific analog circuits. Crucially, these two rails share a Common Ground (GND), which ensures a unified reference point across the circuit, allowing signals to pass between the 5V and 12V domains without voltage float or communication errors.

The third block is configured for 24V DC, which is the standard operating voltage for industrial automation and PLC environments. Interestingly, the schematic labels these as VEE (-12V) and VDD (+12V) to achieve a 24V differential. By using a split-supply approach or a dedicated 24V rail, this terminal provides the necessary "punch" to drive industrial-grade

solenoids, contactors, or robust input/output modules that require more power than standard 5V logic can provide.

The final terminal block is dedicated to 220V AC, representing the primary power input or the high-voltage load side of the system. Unlike the DC blocks, this interface handles alternating current, which is typically used to power the entire system's transformer/switching power supply or is switched via a relay to control heavy machinery and domestic appliances. Because this block carries lethal voltage levels, it is electrically isolated from the previous DC rails usually via optocouplers or relay coils to prevent high-voltage spikes from damaging the sensitive 5V and 12V logic components.

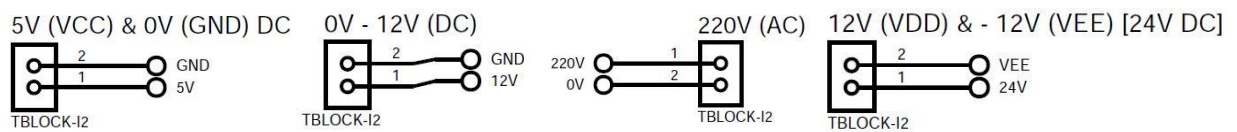


Figure 3.15: The Terminal Blocks of the Power Section.

### III.10. The PCB Part (ARES):

In Proteus ARES, the PCB layout was designed with careful component placement to ensure a logical arrangement, short signal paths, and easy wiring between related circuits. Design Rule Checks were applied throughout the design process to maintain proper track widths, clearances, and manufacturing constraints. The routing was completed using the auto-router, which optimized the connections while respecting the predefined design rules. A double-sided PCB structure was adopted, allowing copper tracks on both the top and bottom layers, reducing routing congestion, minimizing jumper wires, and improving overall board compactness and reliability.

After placing all the components in their correct places and positioning the board correctly, it becomes in the form shown in the picture, and the board printing will be done using the (Gerber extension):

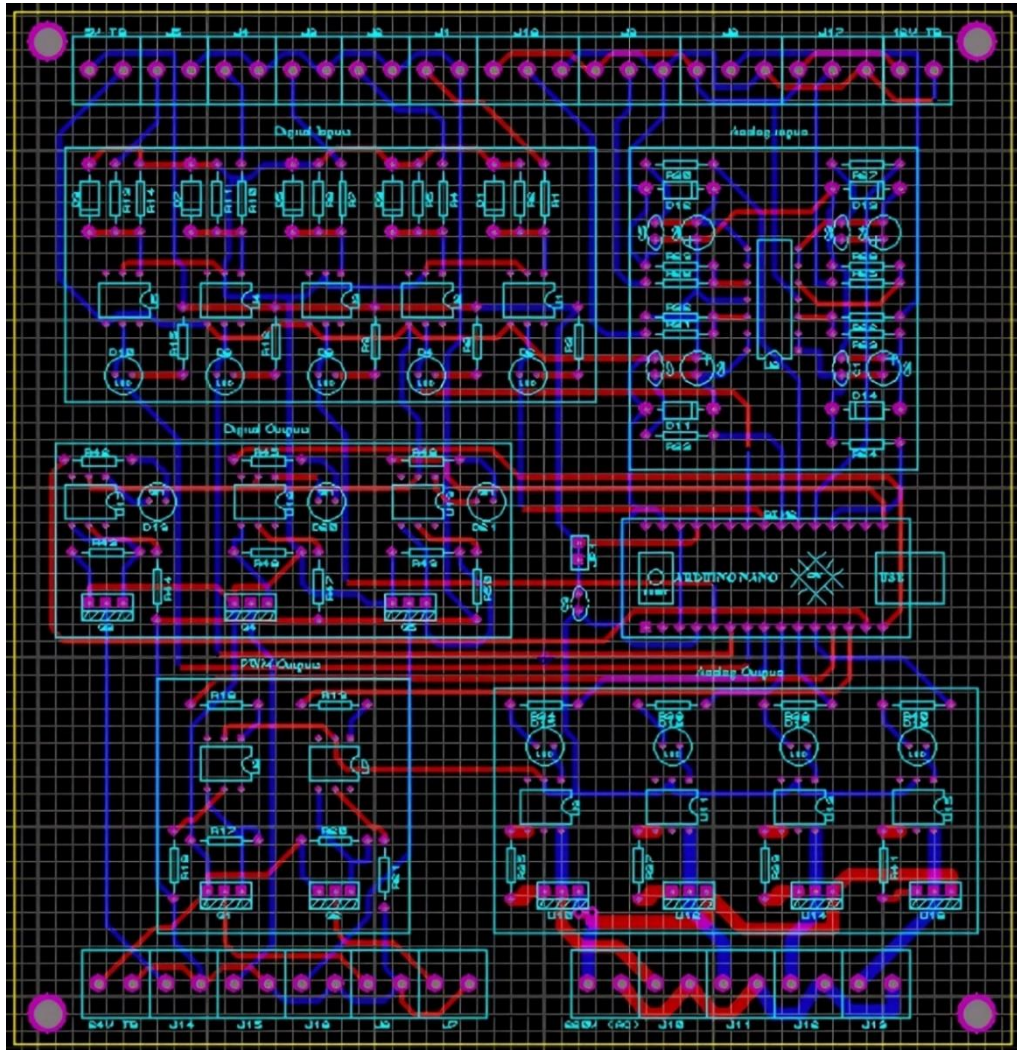


Figure 3.16: ARES Diagram.

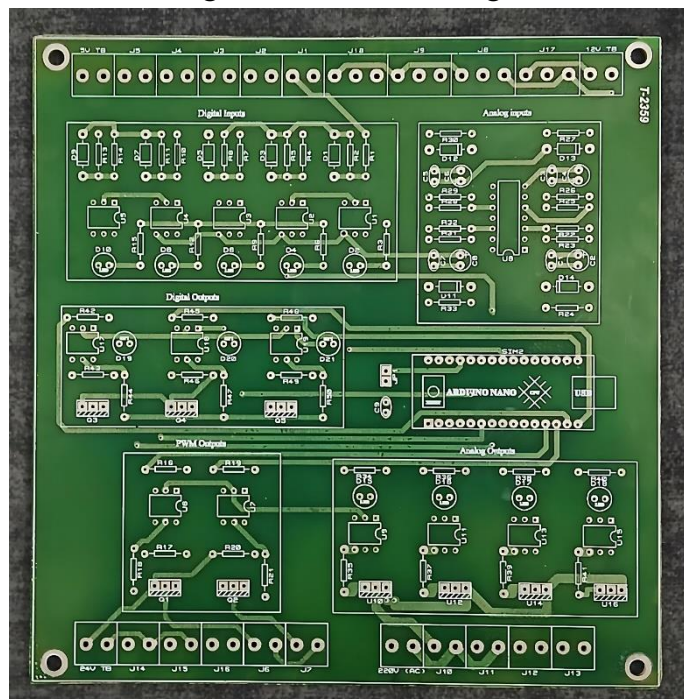


Figure 3.17: The PCB Design.

## III.11. The Didactic Controller:

### III.11.1. Explanation of the Model:

After we finish creating the board, we move on to designing the didactic Controller. This design is dedicated to education, making it simple to understand and safe to use. The material used to make the board is PMMA, which is electrically insulating and suitable for printing on.

Our design consists of our electronic board representing the PLC of the circuit, 5 lamps operating on 220VAC to perform some experiments, 3 contactors with 24VDC voltage to control the motors, two motors the first 220VAC and the second 24VDC for use with two types of outputs and to multiply examples. There are also 2 push buttons, one for starting and the other for stopping, and a potentiometer for controlling the speed of the motors or something similar.

As for the banana jacks connectors, they are a tool for safely, smoothly, and easily connecting these electronic elements.

The image below illustrates the Didactic Controller design:

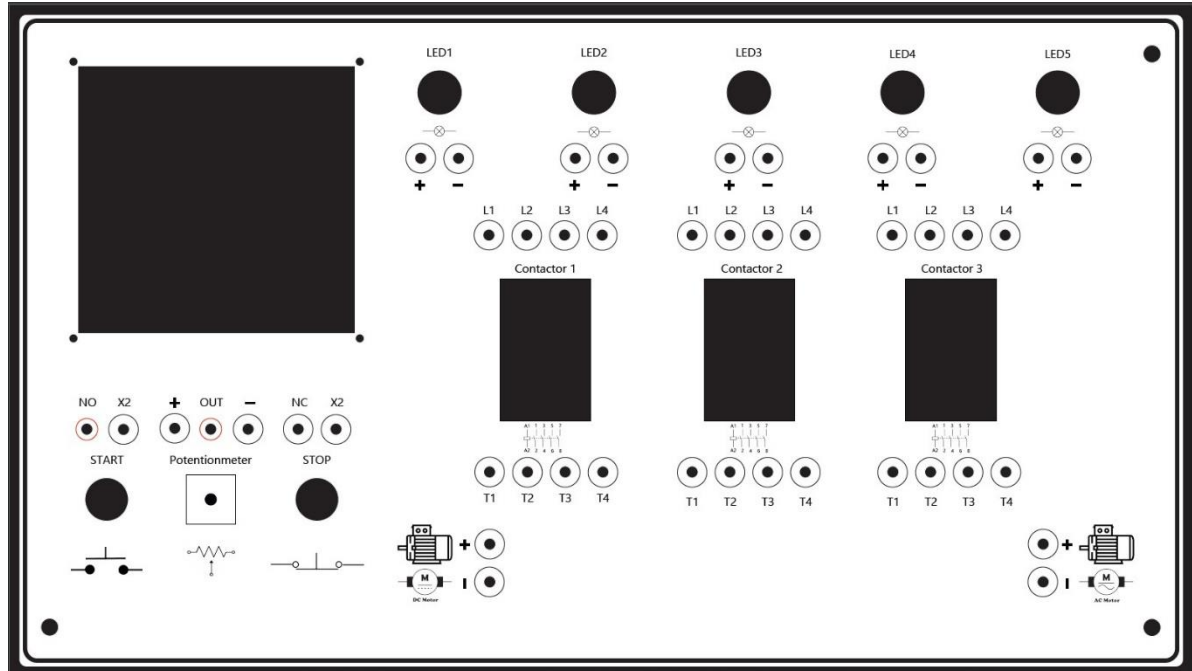


Figure 3.18: The Design of the Didactic Controller.

### III.11.2. Explanation of the Components:

#### III.11.2.1. Rotary Potentiometer (B10K):

This is a 10k $\Omega$  linear rotary potentiometer. It features three terminal pins and a rotating shaft, accompanied by a plastic knob for manual adjustment. The "B" designation typically denotes a linear taper, meaning the resistance changes at a constant rate as the shaft is turned.

It acts as a variable resistor or a voltage divider. By rotating the knob, you alter the position of an internal wiper along a resistive track, which changes the resistance between the center pin and the two outer pins. This is commonly used to manually adjust analog signals, such as setting a reference voltage, tuning the setpoint for a motor speed controller, or providing an adjustable analog input to a microcontroller.

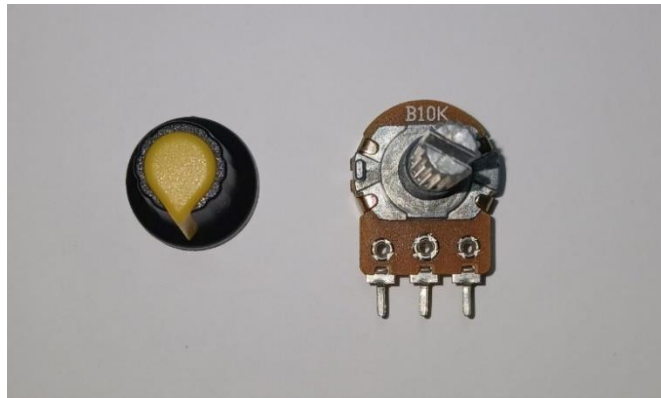


Figure 3.19: Rotary Potentiometer.

#### **III.11.2.2. ABB (24VDC) 4-Pole Contactor:**

This contactor is an ABB AF26Z model, designed for industrial applications and capable of handling AC loads with voltages up to **690 V**. For a 220 V motor, this device can carry sufficient current to operate most medium-power motors.

It features four mains Normally Open (NO) power poles and relies on a low-voltage electronic coil designed to operate on a 24V DC supply.

Contactors are used to safely switch heavy, high-power electrical loads, such as three-phase motors or high-current heating elements. The low-voltage DC coil allows a low-power logic circuit like an industrial controller interface board or a PLC to safely energize the electromagnet inside. Once energized, the electromagnet forcibly pulls the heavy-duty power contacts closed, completing the high-voltage circuit.



Figure 3.20: ABB 24VDC Contactor.

### **III.11.2.3. The Banana Jacks:**

These are screw-type terminal posts designed to be mounted onto a chassis or panel enclosure. They feature a central receptacle sized for standard 4mm banana plugs, a threaded cross-hole for inserting bare wire, and an outer plastic nut to clamp down on the wire.

They provide a robust, reusable, and insulated interface for bringing power or test signals into and out of an electronic enclosure. You will frequently find these on power supplies, test fixtures, and custom prototyping rigs, as they allow for rapid and secure connection and disconnection of external circuits without soldering.



Figure 3.21: Banana Jacks.

### **III.11.2.4. Industrial Pushbutton:**

This is a panel-mounted, spring-return pushbutton with a green actuator. The rear assembly features modular terminal blocks. The image displays a Normally Closed (NC) contact block

(terminals 1 and 2) alongside power terminals (X1 and X2), which are used to power an integrated indicator lamp inside the button.

These pushbuttons used in automated industrial systems as ‘START’ and ‘STOP’ actions.



Figure 3.22: Pushbuttons.

#### **III.11.2.5. LED Pilot Lights:**

These are standard 22mm industrial indicator lights, or pilot lights. The specifications printed on the side indicate they utilize internal LEDs, operate directly on 220V AC power, and have a very low current draw of less than 20mA.

They are mounted onto control cabinet doors or operator stations to provide immediate, visible feedback regarding the operational status of a machine or process. These lamps also work as indicators for any faults or for experiments.



Figure 3.23: LED Pilot Lights.

#### **III.11.2.6. ATX Power Supply Unit:**

This is a standard ATX (Advanced Technology eXtended) computer power supply. Key features visible on the back include an IEC receptacle for the main AC power cord, a master I/O toggle switch, a red voltage selector switch (usually for toggling between 115V and 230V

input), and a cooling fan. A bundled harness of color-coded wires exits the rear, carrying various DC voltage lines.

It takes high-voltage alternating current (AC) from your wall outlet and converts it into the stable, low-voltage direct current (DC) required by the computer's delicate internal components. It simultaneously outputs multiple specific voltages (typically +3.3V, +5V, and +12V) to safely power our electronic circuit.



Figure 3.24: ATX Power Supply.

After the installation is completely finished, this is the final shape of the didactic controller.

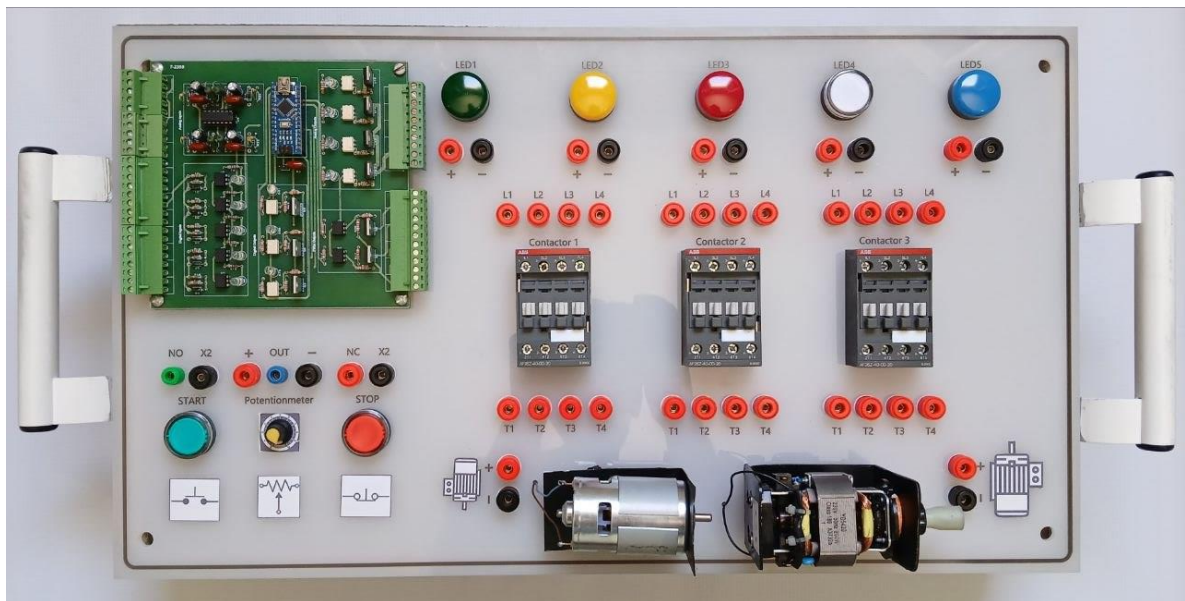


Figure 3.25: The Didactic Controller.

### III.12. The Security Steps:

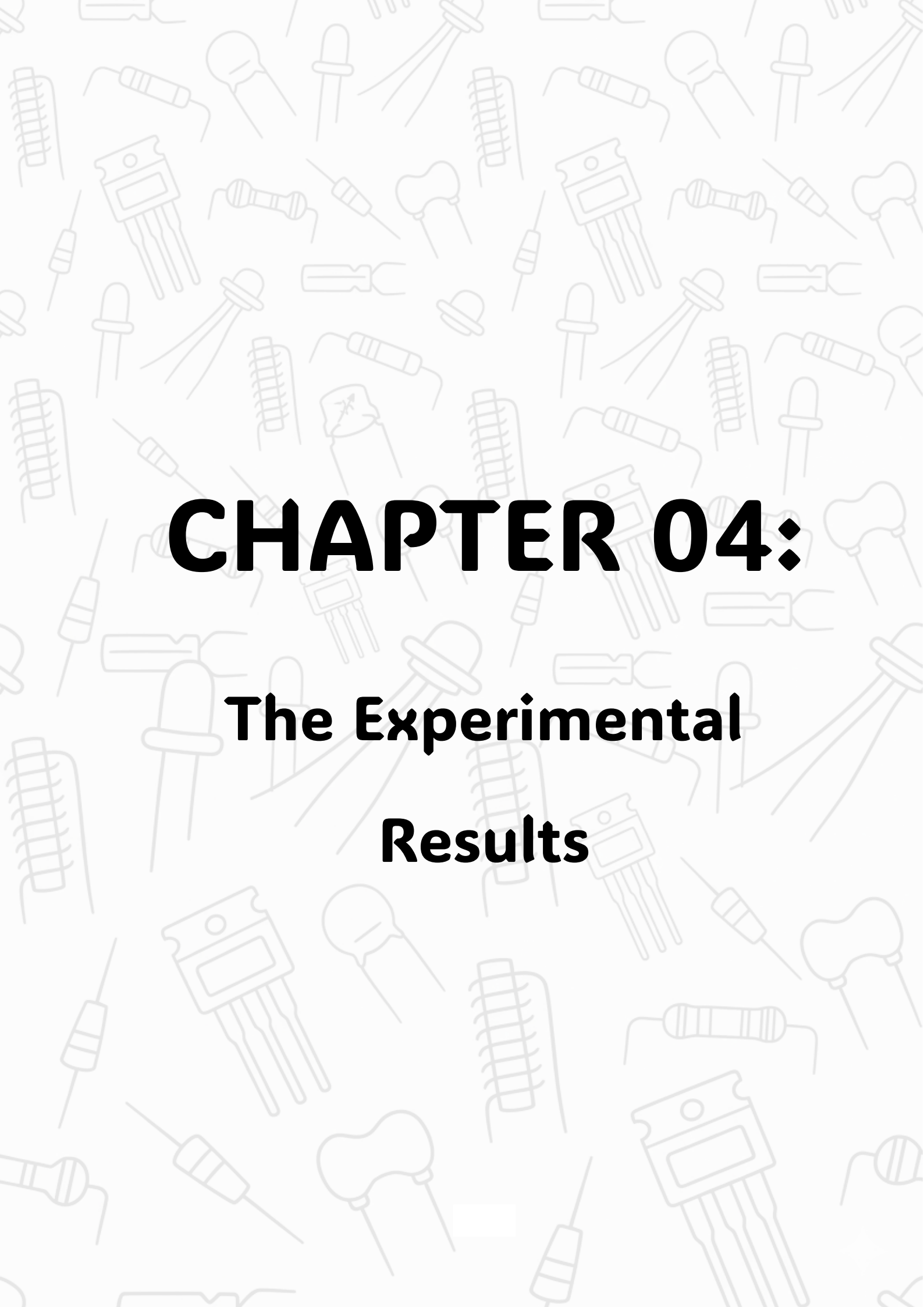
To ensure maximum operator safety and electrical reliability, the controller incorporates active high-voltage warnings and industrialized wiring standards. When the 220V AC mains power is energized, a prominent **red indicator light illuminates instantly**, providing an unambiguous visual warning of live high voltage to prevent accidental shock hazards. To streamline the layout and eliminate structural risks, the 220V power path is routed **directly from the electronic board to the primary control switch** without unnecessary intermediate junctions. Furthermore, all stranded wire connections are securely terminated using **insulated wire ferrules**. These ferrules completely eliminate bad mechanical contacts and wire fraying, successfully suppressing electrical arcing, contact noise, and signal interferences that could disrupt the system's logic.



Figure 3.26: The Security Steps.

### III.13. Conclusion:

By the end of this chapter, the practical side of our project has become entirely clear to us, including how to design an electronic circuit, how to design electrical circuits, and understanding each block, its operational mechanism, and the safety/security aspect of the circuit (**Didactic Controller**).



# **CHAPTER 04:**

## **The Experimental Results**

## IV.1. Introduction:

In this final chapter, we will test what we have accomplished in terms of programming and practical work to evaluate the performance and efficiency of the project.

## IV.2. Programming Phase and Didactic Controller Testing:

After completing the practical implementation phase and finalizing the electronic board design, we proceeded to evaluate the performance and operational efficiency of the 'Didactic Controller'. This was supported by a set of illustrative images demonstrating the use of the programming software (Ladder Maker).

### IV.2.1. Digital inputs Example:

This program demonstrates a fundamental "start/stop" or "latching" control circuit, which is the most common way to start and maintain power to a machine or motor.

Pressing the normally open "ON" button (Input 1) completes the electrical path and energizes the "LED" output. The critical part of this circuit is the parallel "LED" contact placed beneath the "ON" button. When the LED output turns on, this parallel contact closes, providing an alternative path for the power to flow. This "latches" the circuit, keeping the LED on even after you take your finger off the "ON" button.

Pressing the normally closed "OFF" button (Input 2) temporarily breaks the main power line, which drops the latch and turns the LED off.

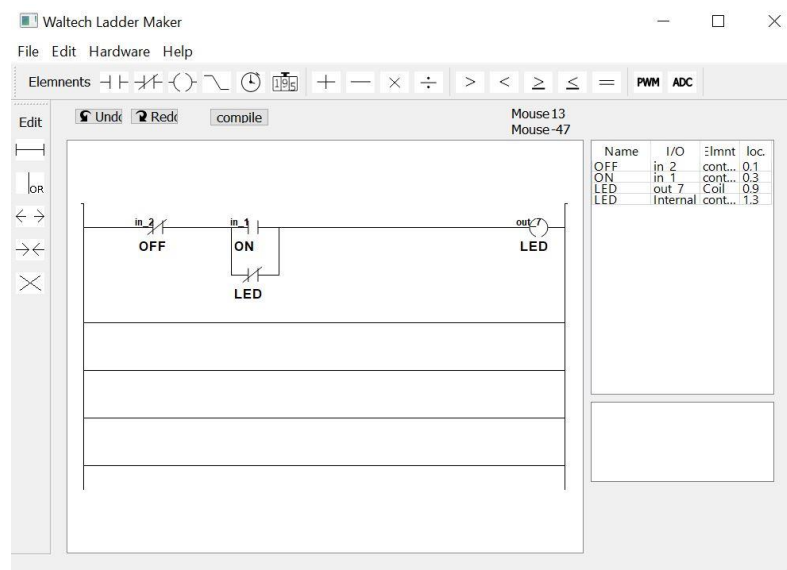


Figure 4.1: Start/Stop Latching Program.

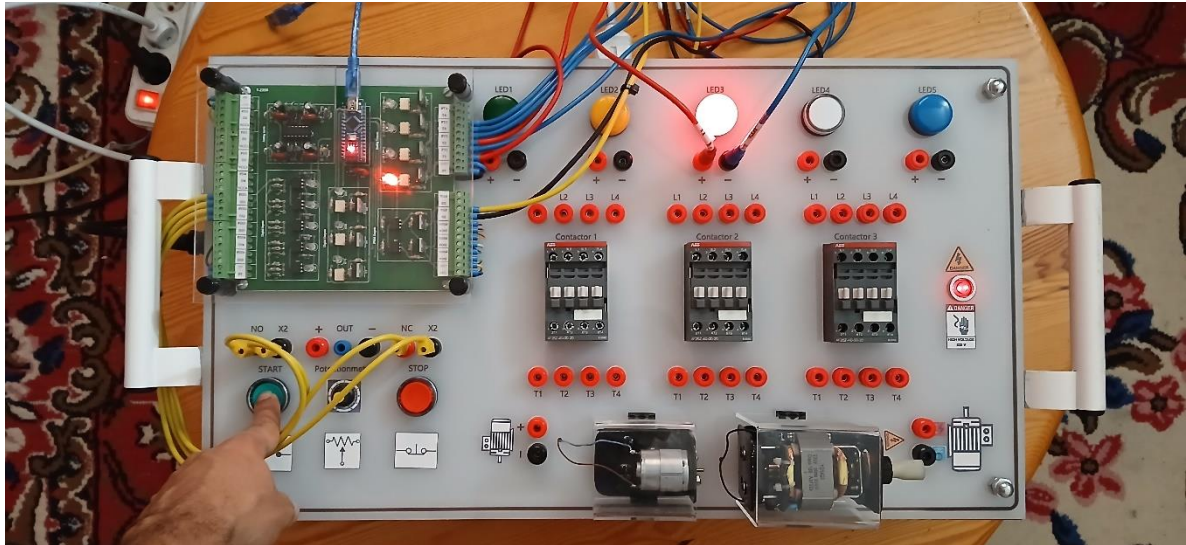


Figure 4.2: The Test of the Start/Stop Latching Program.

#### **IV.2.2. Analog to Digital Converter Program:**

This program reads an incoming analog signal and activates different outputs based on the signal's strength, acting as a multi-level indicator.

It starts with a standard latching circuit (the top rung). When the normally open "start" button is pressed, it activates the "begin" coil. The parallel "begin" contact ensures the system stays on even after you release the start button, keeping it active until the normally closed "stop" button is pressed. Once "begin" is active, the program continuously reads an Analog to Digital Converter (ADC) input (adc1). Below this, three comparator rungs monitor the adc1 value:

- If the ADC value is **128 or higher**, it turns on the **red** LED.
- If the ADC value reaches **512 or higher**, it turns on the **yellow** LED.
- If the ADC value reaches **1000 or higher**, it turns on the **green** LED.

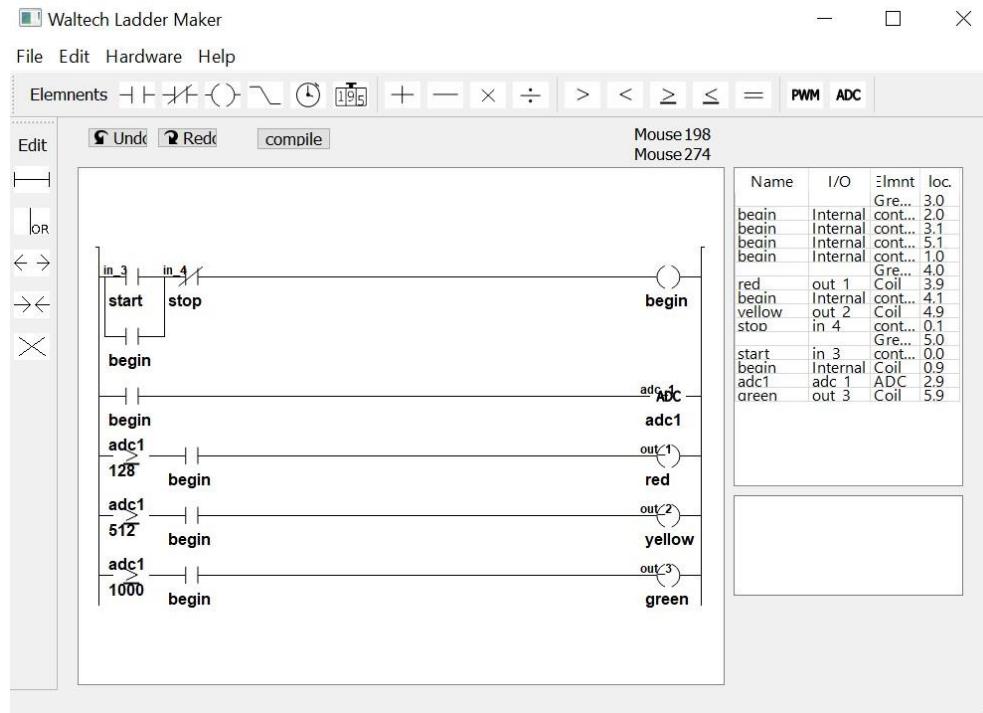


Figure 4.3: Analog to Digital Converter Program.

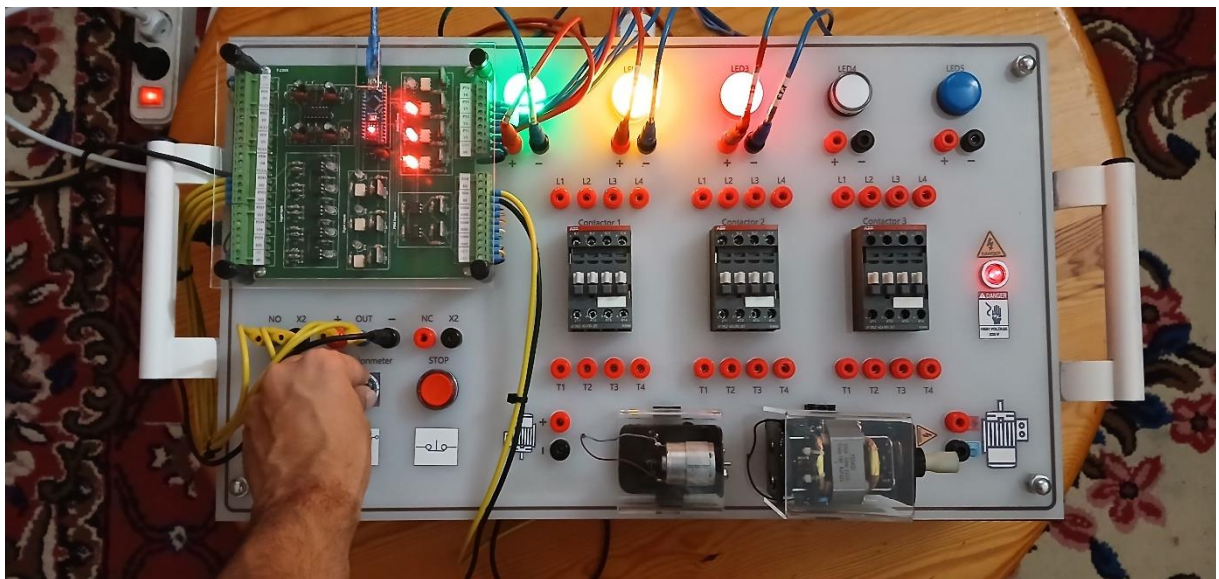


Figure 4.4: The Test of The ADC Converter Program.

## IV.3. Conclusion:

Finally, following these previously presented programs, our project has become fully complete, providing a clear method of use for anyone wishing to utilize it for educational purposes, whether theoretically or practically.

# **General Conclusion**

# General Conclusion:

In conclusion, this project was designed as an innovative and effective to high-cost PLCs. By utilizing low-cost tools and components, it successfully fulfills its purpose for educational environments, and potentially even industrial applications.

This project served as a gateway for us to acquire deep scientific and practical knowledge. It exposed us to commercial challenges and taught us how to navigate them. Furthermore, building this project enabled us to learn about several technical fields, including:

- **Electronics and Electrical Components:** Understanding their functions and the mechanisms for handling and utilizing them.
- **Circuit Design:** Learning how to design electronic circuits, starting from physical experiments to creating them digitally using "Proteus" software. We also learned how to integrate classic control elements with PLCs, using components like contactors and push buttons.
- **Software and Programming:** Gaining hands-on experience with software tools, including Proteus for simulation and PCB design, "Ladder Maker" for ladder logic programming, and the "Arduino IDE" for C++ programming.
- **PLC Architecture:** Benefiting from the theoretical aspects of PLCs, such as understanding their internal structure (inputs and outputs) and learning how to implement electrical isolation between high and low voltages a concept we successfully applied in our project.

Our future aspirations for this project are promising. It can be further developed by upgrading the electronic hardware and utilizing components with more powerful specifications at an affordable price, such as replacing the Arduino with an "ESP32". Because of its adaptability, this project has the potential to be utilized across multiple sectors, serving both the educational and industrial fields.

Furthermore, by designing a dedicated external enclosure and adding professional finishing touches, this system could become a viable competitor to standard PLCs, particularly in small to medium-scale projects with budget constraints.

# References

# References

- [1] L. A. Bryan and E. A. Bryan, *Programmable Controllers: Theory and Implementation*, 2nd ed. Industrial Text Company.
- [2] "Introduction to PLCs and Microcontrollers: Understanding the Differences," PLC Table. [Online]. Available: <http://www.plctable.com>
- [3] V. Romero Segovia and A. Theory, "History of Control: History of PLC and DCS," Jun. 2012 (minor revision Jul. 2013).
- [4] "The Main Differences Between the PLCs & DCS," PLC Technician. [Online]. Available: <http://www.plctechnician.com>
- [5] W. Bolton, *Programmable Logic Controllers*, 6th ed. Elsevier.
- [6] K. Collins, "PLC Programming for Industrial Automation," formatted and updated by S. Willis, Dec. 2014.
- [7] "Different Types of Arduino Boards," ElProCus. [Online]. Available: <https://www.elprocus.com/different-types-of-arduino-boards/>
- [8] Google, "Gemini." [Online]. Available: <https://gemini.google.com/app>
- [9] J. Mengual, "Micro PLC Arduino NANO Modelo: Miró," 2020 ed., 2020.
- [10] DevBoards.info. [Online]. Available: <https://devboards.info>
- [11] Arduino Documentation. [Online]. Available : <https://docs.arduino.cc>
- [12] *Phototransistor Optocouplers, CNY17 Datasheet*, Fairchild Semiconductor.
- [13] *1N400X General Purpose Plastic Rectifier Datasheet*, Vishay General Semiconductor.
- [14] *Quadruple Operational Amplifiers Datasheet*, Texas Instruments.
- [15] *BTA06 and BTB06 Series Datasheet*, STMicroelectronics.

# APPENDIX

## جدول مخطط الأعمال BMC :

|                                                                                                                                                                                                                                                                                                                                                                                        |                                                                                                                                                                                                     |                                                                                                                                                                                                                                                                                                                       |                                                                                                                                                                                                                                          |                                                                                                                         |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------|
| <br>شرائح العملاء<br>Customer Segments                                                                                                                                                                                                                                                                | <br>العلاقات مع العملاء<br>Customer Relationships                                                                  | <br>القيم المقترحة<br>Value Proposition                                                                                                                                                                                            | <br>الأنشطة الرئيسية<br>Key Activities                                                                                                                | <br>الشركاء الرئيسية<br>Key Partners |
| المؤسسات التعليمية والأكاديمية:<br>الجامعات ( مثل جامعة عر داية)،<br>معاهد التكوين المهني، والمدارس<br>العليا للهندسة، الثانويات والمخابر.<br><br>الطلبة والباحثون:<br><br>طلاب تخصصات الهندسة<br>الكهربائية، أتمتة الأنظمة، والتحكم<br>الآلي الراغبين في تطبيق مشاريع<br>التخرج أو البحوث التطبيقية.<br><br>أصحاب المشاريع الصغيرة:<br><br>البواه في معامل الابتكار، وكذا<br>المطورين | الدعم الفني والتقني المستمر :<br>توفير قنوات دعم مخصصة لمساعدة<br>الأساتذة والطلاب في برمجة وتشغيل<br>الجهاز وحل المشكلات التقنية.<br><br>ورش عمل تدريبية: تقديم دورات<br>تدريبية مجانية أو مدفوعة. | جهاز PLC تعليمي واقتصادي عالي<br>الكفاءة:<br><br>يدعم لغة المخطط السلمي (Ladder)<br>ولغة ++C عبر (Arduino)، ليربط<br>بين التعليم الأكاديمي والتطبيقي.<br><br>داخل ومخرج معزولة صوتياً<br>لحماية النظام، مع دعم الأحصال<br>الصناعية الخلفية.<br><br>سهل الصيانة والتطوير محلياً دون<br>الحاجة لقطع غيار مستوردة مكلفة. | الخبرة التقنية والهندسية للفريق ،<br>برمجيات ..Proteus, etc.<br><br>أدوات ومعدات التصنيع، مخبر<br>تركيب واختبار اللوحات، والمكونات<br>الإلكترونية الأساسية                                                                               | الجمعيات العلمية والوادي<br>التكنولوجية الجامعية لترويج الجهاز<br>بين مجتمعات الطلاب والمهندسين<br>المهتمين بالأتمتة.   |
| <br>مصادر الإيرادات<br>Revenue Streams                                                                                                                                                                                                                                                            |                                                                                                                                                                                                     |                                                                                                                                                                                                                                                                                                                       | <br>هيكل التكاليف<br>Cost Structure                                                                                                                 |                                                                                                                         |
| مبيعات المنتج: الإيراد الرئيسي من بيع أجهزة الـ PLC التعليمية والحقائب المخبرية المكتملة.<br><br>الصيانة وقطع الغيار: تقديم خدمات الدعم الفني وتوفير قطع الغيار للبطاقات الإلكترونية محلياً.<br><br>التدريب والاستشارات: حوالة تقديم دورات برمجية وتطبيقية متقدمة في الأتمتة الصناعية.                                                                                                 |                                                                                                                                                                                                     |                                                                                                                                                                                                                                                                                                                       | شراء المكونات الإلكترونية الأساسية وتكاليف طباعة وتجميع لوحات الـ PCB.<br><br>الصيانة والدعم الفني (مصاريف برامج التصميم، معدات الفحص المخبري)<br><br>تكاليف شحن وتوزيع الأجهزة، حملات التسويق الرقمي، المشاركة في المعارض، وأجور الفريق |                                                                                                                         |

## شهادة التوطين :



الجمهورية الجزائرية الديمقراطية الشعبية  
وزارة التعليم العالي والبحث العلمي  
جامعة غرداية  
حاضنة أعمال جامعة غرداية



رقم: 143 / ح.أ.ج.غ/2026

### شهادة توطين مشروع مبتكر وفق القرار 008 المعدل والمتمم للقرار 1275

أنا الممضي أسفله، السيد: د/ طالب أحمد نور الدين

مسير حاضنة الأعمال: جامعة غرداية

المقر الاجتماعي/ العنوان: المنطقة العلمية، ص ب 455، غرداية، 47000، الجزائر

بتاريخ: 2025/04/10

رقم علامة الحاضنة: 1004253146

طبيعة المشروع: مؤسسة ناشئة

أشهد أن الطالب(ة) / الطلبة التالية أسماؤهم:

| الإسم واللقب         | الطور الدراسي | التخصص           | الكلية                   |
|----------------------|---------------|------------------|--------------------------|
| مجد عبد اللطيف قندوز | M2            | أوتوماتيك وأنظمة | كلية العلوم والتكنولوجيا |
| زكرياء رزقي          | M2            | أوتوماتيك وأنظمة | كلية العلوم والتكنولوجيا |

تحت إشراف الأستاذ(ة)/الأستاذة التالية أسماؤهم:

| الإسم واللقب | الرتبة        | التخصص            | الكلية                   |
|--------------|---------------|-------------------|--------------------------|
| رفيق علجي    | أستاذ محاضر ب | أوتوماتيك و أنظمة | كلية العلوم والتكنولوجيا |

تم توطينه على مستوى حاضنة أعمال جامعة غرداية - بمشروع تحت اسم:

Design and implementation of arduino based PLC for an educational purpose.

خلال السنة الجامعية: 2026/2025

سلمت هذه الشهادة بطلب من المعني للإدلاء بها في حدود ما يسمح به القانون.

حرر في غرداية بتاريخ: 2026/04/26

مدير الحاضنة



## ورقة التصحيح :

|                                                                                                                                               |                                                                                     |                                                                                       |
|-----------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| Université de Ghardaïa<br>Faculté des Sciences<br>Département de la technologie                                                               | الجمهورية الجزائرية الديمقراطية الشعبية<br>وزارة التعليم العالي والبحث العلمي       | جامعة غرداية<br>كلية العلوم والتكنولوجيا<br>قسم الآلية والكهروميكانيك                 |
| غرداية في: 2026/07/05                                                                                                                         |    | شعبة : آلية<br>تخصص: آلية و أنظمة                                                     |
| شهادة ترخيص بالتصحيح والاياداع:                                                                                                               |                                                                                     |                                                                                       |
| انا الاستاذ : حامد بوخاري.<br>بصفتي المشرف المسؤول عن تصحيح مذكرة تخرج ماستر المعنونة ب:                                                      |                                                                                     |                                                                                       |
| <b>Design and Implementation of an Arduino Based PLC for an Educational Purpose</b>                                                           |                                                                                     |                                                                                       |
| من انجاز الطلبة:                                                                                                                              |                                                                                     |                                                                                       |
| فندوز محمد عبد اللطيف                                                                                                                         |                                                                                     |                                                                                       |
| رزقي زكرياء                                                                                                                                   |                                                                                     |                                                                                       |
| التي نوقشت بتاريخ 2026 /06/ 22                                                                                                                |                                                                                     |                                                                                       |
| اشهد ان الطلبة قد قاموا بالتعديلات والتصحيحات المطلوبة من طرف لجنة المناقشة وقد تم التحقق من ذلك من طرفنا<br>وقد استوفت جميع الشروط المطلوبة. |                                                                                     |                                                                                       |
| مصادقة رئيس القسم                                                                                                                             | امضاء المسؤول عن التصحيح                                                            |                                                                                       |
| <br>رئيس قسم الآلية والكهروميكانيك<br>عزواوي محمد          |  |  |