|  |
| --- |
| Image result for nuclear fuel complex logo |
| PROJECT WORK:  Automating Systems Using Arduino |
| SUBMITTED |
| By |
| Nanagiri Abhishek Varma  Garlapati Sai Kumar |
| III Yr. B. Tech  DEPT. OF COMPUTER SCIENCE AND ENGINEERING |
| Talla Kusuma Reddy |
| IV Yr. B. Tech |
| DEPT. OF ELECTRICAL AND ELECTRONICS ENGINEERING |
| **Siddhartha Institute Of Technology and Sciences –Narapally, Telangana**  **Narasaraopeta Engineering College – Narasaraopeta, Andhra Pradesh**  24th February 2025 to 23rd March 2025 |
|  |

|  |
| --- |
|  |
| Department of Atomic Energy  NUCLEAR FUEL COMPLEX |

**BONAFIDE CERTIFICATE**

This is to certify that Mr. Nanagiri Abhishek Varma, Mr. Garlapati Sai Kumar and Ms. Talla Kusuma Reddy have done their Project Work under my guidance during the period 24/02/2025 to 23/03/2025 on the topic entitled **Automating Systems Using Arduino** in Nuclear Fuel Complex.

It is ensured that the report does not contain classified or plant operational live data in any form.

|  |  |
| --- | --- |
|  | Signature: |
|  | Name: G.V. Sathya Narayana |
| Hyderabad | Designation of Guide: Deputy Manager |
| Date: | Plant: Equipment Development & Automation |

|  |
| --- |
| Approved by |
| GeneralManager, ED&A |

**ACKNOWLEDGEMENT**

First of all, I am extremely thankful to Shri Dr. Komal Kapoor, Chairman, NFC Board & Chief Executive, NFC for giving me this opportunity to carry out Internship at Nuclear Fuel Complex.

I would like to express our sincere thanks to Shri B.V ArunKiran (Deputy Manager, ED&A) for accepting to be our Guide and helping us throughout our internship.

I profoundly thank Shri N. Rajesh, GM (ED&A) allowing us to do the internship in the plant.

I would like to thank Shri A. Maruthi Ram, DCE (TP & ED&A) for facilitating our internship.

I express our sincere thanks to Shri V. V. Mahesh Kumar, DCE (Safety, HPU, TLD & HR)andShriD. Srinivas, AGM (HRD, Q. Cir., & QIS) for helping me throughout our internship at NFC and also conducting Awareness Programme on DAE/NFC activities at HRD.

I am grateful to my parents who have given me constant encouragement and inspiration to pursue my graduation.

Finally, I would like to thank everyone who has directly and indirectly helped us in the successful completion of this internship.

Table of Contents

[1. INTRODUCTION 5](#_Toc192939939)

[1.1 ABOUT NFC: 5](#_Toc192939940)

[1.2 SCOPE: 5](#_Toc192939941)

[1.3 NUCLEAR FUEL: 6](#_Toc192939942)

[1.4 MAKING OF NUCLEAR FUEL: 6](#_Toc192939943)

[1.5 ZIRCALOY PRODUCTION: 7](#_Toc192939944)

[1.6 FUEL FABRICATION: 7](#_Toc192939945)

[1.7 SELF RELIANCE: 7](#_Toc192939946)

[1.8ABOUT ED&A: 8](#_Toc192939947)

[1.9 Process Plan: 8](#_Toc192939948)

[2. LITERATURE REVIEW 10](#_Toc192939949)

[2.1. Introduction 10](#_Toc192939950)

[2.2. Purpose of the Literature Review 10](#_Toc192939952)

[2.3. Component Classification 10](#_Toc192939954)

2.4. System Block Diagram…………………………………………………………………………...29

2.5. Comparative Analysis of Components…………………………………………………………...29

2.6. Integration Challenges and Observations………………………………………………………...30

2.7. Future Scope and Upgradability……………………………………………………………….....30

[3. Design…………………………………………………………………………………………………31](#_Toc192940001)

[3.1. Introduction 31](#_Toc192940002)

[3.2. Design Considerations 31](#_Toc192940003)

[3.3. Design Calculations (Simplified) 31](#_Toc192940004)

[3.4. Iterative Design Process 31](#_Toc192940005)

[3.5. Future Enhancements 31](#_Toc192940006)

[3.6. Risk Analysis and Mitigation 31](#_Toc192940007)

[3.7. Hardware-Software Integration Strategy 32](#_Toc192940008)

[3.8. Real-World Inspiration 32](#_Toc192940009)

[4. Conclusion 33](#_Toc192940010)

[5. References 34](#_Toc192940015)

# 1. INTRODUCTION

## 1.1 ABOUT NFC:

The Nuclear Fuel Complex (NFC), established in 1971, is a key industrial unit of the Government of India's Department of Atomic Energy. The facility is in charge of supplying nuclear fuel bundles and reactor core components to all of India's nuclear power reactors. It is a one-of-a-kind factory that produces natural and enriched uranium fuel, zirconium alloy cladding, and reactor core components all under one roof, beginning with raw materials.

## 1.2 SCOPE:

The Nuclear Fuel Complex is unique in many respects. It is the only Complex of its kind where Uranium concentrates on the one hand and Zirconium mineral on the other are processed at the same location all the way to produce finished fuel assemblies and also zirconium alloy tubular components, for supplies to the Nuclear Power Industry. The complex also symbolizes the strong emphasis on self-reliance in the Indian Nuclear Power Programme. The advanced technologies for the production of nuclear grade uranium di-oxide fuel, zirconium metal and zirconium alloy tube components and the manufacture of fuel bundles conforming to reactor specifications were developed through systematic efforts during the late 50's and the 60's.

The complex has different types of production facilities which include the Zirconium Oxide Plant for processing of Zircon to pure Zirconium oxide; the Zirconium Sponge Plant for conversion of Zirconium oxide to pure sponge metal; facilities for reclamation of zircaloy mill-scrap; the Zircaloy Fabrication Plant for producing various zirconium alloy tubing’s and also sheet, rod and wire products; the Uranium Oxide Plant for processing crude uranium concentrate to pure uranium di-oxide powder; the Ceramic Fuel Fabrication Plant for producing sintered Uranium oxide pellets and assembling of the fuel bundles for the PHWRs; the Enriched Uranium Oxide Plant for processing of imported enriched uranium hexafluoride to enriched uranium oxide powder; the Enriched Uranium Fuel Fabrication Plant for producing enriched UO2 pellets and the fuel assemblies for the BWR reactors; and a plant for fabrication of components and sub-assemblies for Fast Breeder Reactors. A Special Materials Plant for producing a number of electronic grade high purity materials for supplies to the Electronic Industry and plants producing stainless steel seamless and other special tubes have also been set up in this complex.

The common plant facilities comprising of the Quality Control Laboratory, the Central Workshop, the Compressor and Boiler House, the Civil, Electrical and Mechanical Engineering Services render strong support to the Plant operations.

While the individual plant capacities were designed to match the requirements of the Indian Nuclear Power Programme as projected in the early '70s the capacities have been under continuous review. With the experience gained in the operation of various production plants, process and equipment modifications have been incorporated to progressively improve plant performance. The stage has now been reached for substantial increase in capacities and plans have been drawn up for establishing new plants to cater to the requirements of fuel and zircaloy for the 6,000 Mwe Indian Nuclear Power Programme to be implemented in this decade.

An important feature at the Nuclear Fuel Complex, is that, apart from indigenous process development, a good portion of the plant equipment for the chemical engineering and extractive metallurgy operations has been indigenously designed and fabricated by the Indian industry. Even in the case of fabrication plants, sophisticated equipment such as Vacuum Annealing Furnace, the Pilger mill, the High Temperature Hydrogen welding units have been successfully designed and fabricated in-house.

## 1.3 NUCLEAR FUEL:

India is developing a three-stage indigenous nuclear power programme that includes closed fuel cycles of Pressurized Heavy Water Reactors (PHWRs) and Liquid Metal Cooled Fast Breeder Reactors (LMFBRs) to make the best use of the comparatively limited uranium deposits and large thorium resources. The initial step of the Power plan is PHWRs, which employ zircaloy as clad and natural uranium dioxide as fuel. Furthermore, India has been running two Boiling Water Reactors (BWRs) since 1969. NFC manufactures zircaloy clad enriched uranium oxide fuel elements and assemblies for these reactors using imported enriched uranium hexafluoride.

## 1.4 MAKING OF NUCLEAR FUEL:

Natural Uranium is mined in Jharkhand at Jaduguda. At the Nuclear Fuel Complex, it is processed into nuclear fuel assemblies. Natural uranium dioxide weighs around 15.2kg in a 220 MW PHWR fuel unit. Uranium dioxide pellets produce heat and fission products while undergoing fission. Fission products are radioactive, and they should be kept apart from cooling water. As a result, the UO2 pellets are housed in Zirconium alloy tubes with hermetically sealed ends.

A 220 MWe reactor unit comprises 3,672 of these fuel assemblies. They are usually replaced after roughly 18 months in the reactor. Careful design and meticulous quality control protect against service failures.

There is no combustion in uranium fuel, and a fuel assembly exits the reactor in the same manner that it entered. However, there is one significant difference: when a fuel assembly is withdrawn from the reactor after approximately 18 months of usage, it retains radioactive by-products from the fission process. Because of the radioactivity, the fuel assembly is transferred to a water-filled pool inside the station using a remote-controlled fuel assembly loading/unloading system.

1. Depleted Uranium (approximately 98%) is stored in fast breeder reactors for recycling.

2. Plutonium (approximately 0.4%) is created when neutrons are absorbed in non-fissile uranium atoms. This is extremely precious and can be used to power rapid reactors.

3. A small percentage of long-lived radioactive fission products is vitrified and preserved.

## 1.5 ZIRCALOY PRODUCTION:

The source mineral for zirconium metal manufacturing is zircon (zirconium silicate), which is found in beach sand deposits in Kerala, Tamil Nadu, and Orissa and is provided by the Indian Rare Earths Ltd. To obtain zirconium oxide, zircon sand is subjected to caustic fusion, dissolution, solvent extraction (to remove hafnium), precipitation, and calcination procedures. To obtain a homogenous zirconium sponge, the pure zirconium oxide is treated to high temperature chlorination, reactive metal reduction, and vacuum distillation. The sponge is then briquetted with alloying materials and vacuum arc melted numerous times to produce homogeneous zircaloy ingots, which are subsequently extruded, pilgered, and finished into seamless tubes, sheets, and bars.

## 1.6 FUEL FABRICATION:

The cylindrical UO2 pellets are stacked and enclosed in thin walled zirconium alloy tubes for PHWR fuel, with both ends sealed by resistance welding with zircaloy end plugs. A number of these fuel pins are joined together to form a fuel bundle that may be easily put into the reactor. The fuel bundles for PHWR 220 MW and PHWR 500 MW each have 19 and 37 fuel pins. There are two types of array fuel assemblies for BWRs: 6x6 and 7x7.

## 1.7 SELF RELIANCE:

The Nuclear Fuel Complex is a prime example of a successful translation of indigenously created procedures to large-scale operations. The country's high self-reliance in the critical area of nuclear fuel and core components is a major asset in not only supporting the nuclear power programme but also establishing a huge number of allied and ancillary sectors..

## 1.8ABOUT ED&A:

The EQUIPMENT DEVELOPMENT AND AUTOMATION (ED&A) branch was established in the early 1980s in response to challenges in importing special purpose welding equipment required for PHWR fuel assembly manufacturing due to the embargo imposed on India by developed countries. Beginning with special-purpose welding equipment, this sector has grown to develop a wide range of machines for specialized purposes in nuclear fuel manufacture.

This plant is critical for the original creation of special purpose equipment for a variety of applications based on user requirements. This division has successfully developed many machines during the last few decades. Certain new technologies, such as empty tube welding, were developed as part of the process of constructing full equipment.

This section's classic special purpose machines, along with their applications, are detailed below.

1. Welding machines for tiny appendages (35mm x 2.5mm x 1mm, 9mm x 2.5mm x 0.6mm) on thin-walled zircaloy tubes (thickness: 0.4mm)

2. Strength testing devices for determining the shear strength of appendage and tube weld joints

3. Graphite coating equipment for coating the inner surface of thin-walled tubes with a few microns of graphite.

4. Granule Transfer System with no leaks for transferring granules to a height of 4 metres.

5. Powder conveyance through flexible screw conveyors

In addition to the foregoing, this section is involved in the automation of intra-product transfer systems.

## 1.9 Process Plan:

The following is the typical flow chart followed for development of special purpose machines. All the projects may not necessarily have all steps.

1. Project starts with getting requirement from the user section having complete specifications

2. Field study of the engineers to understand & enrich the user requirement

3. Conducting Conceptual & feasibility studies

4. Development of concept / prototype study of the machine

5. Design of the machine / gadget with complete sequence of operations

6. Presenting the concept & design to the user for feedback & approval

7. Development of fabrication drawings & control system for the machine

8. Procurement of standard items required for assembly of the machine

9. Fabrication of components & assembly of the machine

10. Testing of the operational & safety features of the machine in automatic & manual modes

11. Installation & Commissioning of the machine at user section

12. Maintenance & performance monitoring of the machine for a period of 1 year after commissioning

13. Preparation of Operation & Maintenance manuals and handling over the machine to maintenance group of the user section.

# 2. LITERATURE REVIEW

## **2.1. Introduction**

## The automatic door system is a modern engineering solution aimed at enabling contactless, secure, and automated entry management. This system relies on a combination of sensors, actuators, and control logic implemented using microcontrollers and peripheral modules. Literature review of the components helps in understanding their role, evaluating their suitability, and establishing a technical foundation for the system.

## **2.2. Purpose of the Literature Review**

## The objective of this literature review is to evaluate the components used in the automatic door system, justify their selection over alternatives, and provide insights into their functionality and practical applications. It also supports understanding of how each component fits into the larger architecture of the system.

## **2.3. Component Classification**

## The components used in the system can be grouped based on their roles as follows:

## - Control Unit: Arduino Uno R3

## - Actuation: DC Motor with Encoder, Motor Driver, Relays

## - Sensing: PIR Sensor, Inductive Proximity Sensor, Push Buttons

## - Power and Support: SMPS, Breadboard, Wires

## **Component 1**: **DC Motor with Encoder (4 Pins - VCC, GND, A, B)**

## **Why We Use This Component:**

## A DC motor with a built-in encoder is crucial in systems where **precise movement and position tracking** are required. The encoder gives feedback on the motor’s rotation, allowing you to measure how far the motor has turned—essential in a **glass door system** where positioning is everything to avoid damage or misalignment.

## **Why This Over Alternatives:**

| Option | Pros | Cons |
| --- | --- | --- |
| DC Motor with Encoder | Real-time position feedback, cost-effective | Slightly complex to integrate |
| Stepper Motor | Precise movement control | No feedback (unless closed-loop), torque drops at high speeds |
| Servo Motor | Built-in control and feedback | Limited rotation range, costlier |
| BLDC Motor with Encoder | High torque, smooth control | Overkill for simple door systems |

## We chose the DC motor with encoder for its balance of cost, simplicity, and feedback capability.

## **Analogy:**

## Think of the encoder like a **speedometer and odometer combo** for your motor. It tells you how fast and how far you’ve gone, helping the “driver” (Arduino) decide when to slow down, stop, or reverse.

## **How It's Used in This Project:**

## Provides feedback on door position.

## Encoder pins **A and B** are used for **quadrature decoding** to determine the direction and count of movement.

## Encoder count is used to:

## **Start slow** and speed up gradually.

## **Slow down** as the door nears its open/close limits.

## **Trigger emergency stops** or actions if the door is halfway.

## **Applications:**

## Industrial automation

## CNC machines

## Robotic arms

## Smart gates and access systems

## **Use Case in Project:**

## The glass door **must stop precisely** at fully open and fully closed positions. If it overshoots, it can damage the frame or glass. Encoder count helps track this position **without needing limit switches**.

## **Extra Insight:**

## **Pins:**

## **VCC/GND** power the encoder section.

## **A and B** are quadrature channels giving a square wave based on rotation.

## Quadrature lets you **detect direction and speed**, essential for smooth acceleration/deceleration.

**Component 2: Cytron MD10C v3 Motor Driver Board**

**Why Did We Use This Component?**  
The Cytron MD10C v3 is a robust motor driver designed specifically to handle large DC motors up to 10A continuously and 30A peak. It is compatible with a wide input voltage range (up to 30V), making it ideal for our project where a 24V SMPS is used. It supports PWM and Direction (DIR) control, which is simple and effective for speed and direction control using an Arduino.

**Why This Compared to Alternatives?**

* Compared to **L298N**: L298N is a popular H-bridge motor driver but only supports up to 2A per channel, which would not be sufficient for a high-torque gear motor used in a heavy glass door application.
* Compared to **Monster Moto Shield**: Monster Moto can handle higher currents but is bulkier, pricier, and not as beginner-friendly as the Cytron MD10C.
* The MD10C v3 offers **built-in protection features** like overcurrent, overheating, and reverse polarity protection, adding safety to the system, which is crucial for glass door control.

**Application**  
Motor driver boards like the MD10C are used in robotics, home automation systems, CNC machines, conveyor belts, and any application that requires controlled motion of DC motors.

**Use Case**  
In this project, the MD10C receives PWM signals for speed control and a DIR signal to determine the direction of motor rotation. It drives the high-torque DC motor that physically moves the door forward and backward.

**Example**  
Think of the MD10C as the “brainstem” for the motor—it receives commands from the brain (Arduino) and decides how to move the muscles (motor). It interprets whether the motor should go fast or slow, forward or reverse, and ensures it doesn't hurt itself in the process.

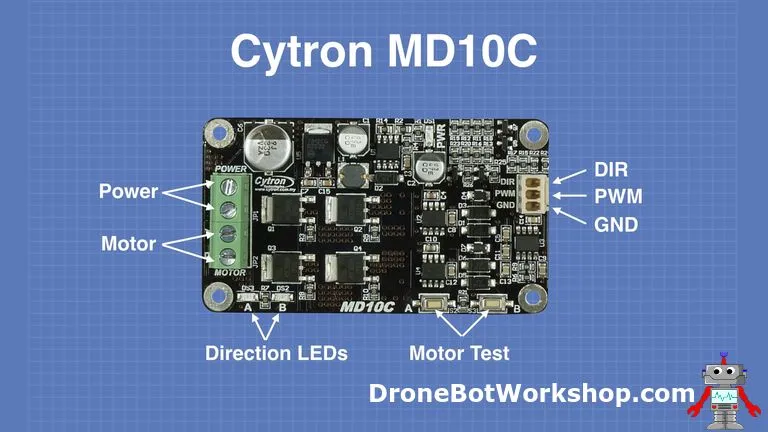
**Analogy**  
Imagine you're controlling a remote-controlled car. The Arduino is like the person holding the remote. The MD10C is the onboard electronics inside the car that understands when to move forward, backward, fast, or slow based on your input.

**How It’s Used in the Project**

* The **DIR pin** is connected to an Arduino digital output to control the motor's direction.
* The **PWM pin** is connected to a PWM-enabled digital pin for variable speed control.
* The **GND** is shared with Arduino to complete the circuit.
* The MD10C is powered directly by the **24V SMPS**, and the motor output terminals are connected to the door motor.
* We added a relay between the MD10C and motor to control emergency cut-off and safety logic.

**Additional Details**

* Version 3 of MD10C is optimized with newer MOSFETs for better efficiency and lesser heat dissipation.
* It simplifies the need for dual H-bridge circuits in bi-directional motor control.
* The clear LED indicators on board provide real-time status feedback, useful for debugging and understanding the system state during testing.



**Component 3: Inductive Proximity Sensor (Homing Sensor)**

**Why Did We Use This Component?**  
In your project, accurate homing is *critical*—you need to know exactly when the door reaches its fully open (home) position to set the encoder count to zero. An **inductive proximity sensor** is perfect for this job because:

* It can **reliably detect metal** (usually part of the door frame or an attached metal plate) without physical contact.
* It's **immune to dust, moisture, and vibrations**, making it ideal for real-world mechanical environments.
* It has **fast response time** and **long lifespan** due to its solid-state nature (no mechanical parts).



**Why This Compared to Alternatives?**

| **Sensor Type** | **Why Not?** |
| --- | --- |
| Mechanical Limit Switch | Wears out over time, physical contact can damage the glass. |
| Optical Sensor | Can be affected by ambient light, dust, or misalignment. |
| Hall Effect Sensor | Needs magnets, and can have lower precision for this task. |

The **inductive proximity sensor** is the sweet spot for durability, precision, and safety, especially when working with heavy components like a **glass door**.

**Applications**

* CNC machines for home position detection
* Industrial automation for object sensing on conveyors
* Robotic arms for motion limit feedback
* Elevators and sliding doors for limit detection

**Use Case in This Project**

* Acts as a **"home switch"** for the glass door.
* During startup, the door moves until this sensor is triggered.
* When triggered (goes **LOW** due to INPUT\_PULLUP setup), it tells the Arduino:

“Hey! Door is fully open. Set encoder to 0 now.”

This sets a **baseline reference** for all future motor movements, avoiding misalignment or crashes.

**Example**  
Think of it like the **elevator “ground floor” sensor**—every time the elevator system boots, it goes to the ground floor to recalibrate its location. Your door does the same during homing.

**Analogy**  
If the encoder is like your odometer, the inductive sensor is like the **start line** in a race. Every race begins from this line, so no matter how far you go, you can always calculate where you are from the start.

**How It’s Used in the Project**

* Connected to an Arduino digital pin with INPUT\_PULLUP.
* Active LOW: When triggered (door reaches metal point), pin reads LOW.
* As soon as triggered:
  + Door stops.
  + encoderCount is reset to 0.
  + System is marked as “homed” and ready for operation.

**Additional Details**

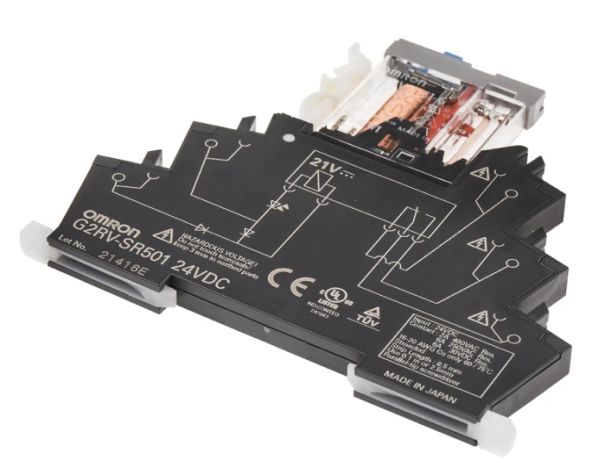
* The sensor requires **VCC, GND**, and **Output**. In your case, powered from the SMPS via a relay to ensure it only gets power when needed (saving power & adding safety).
* Some sensors have LED indicators built-in, which help during debugging—super handy!
* Detection range varies by model (commonly 2mm–8mm), so proper positioning is crucial.

**Component 4: Relay Modules (Motor Control & Sensor Power Switching)**

**Why Did We Use This Component?**

Relays act like **remote-controlled switches**—you use a small signal from the Arduino to control large power circuits (like a 24V SMPS or motor). In your project, relays serve two main purposes:

1. **Safety Shutdown & Isolation**:  
   One relay is placed between the motor and the motor driver—this allows you to **cut off power instantly** in case of emergency or for controlled startup.
2. **Selective Power Control for Homing Sensor**:  
   Another relay powers the inductive sensor **only when required** (e.g., during homing), saving energy and preventing false triggers during regular operation.



**Why This Compared to Alternatives?**

| **Option** | **Why Not?** |
| --- | --- |
| Transistors/MOSFETs | Better for fast switching, but less intuitive for full isolation and often more complex. |
| Manual Switches | Not automatic. Doesn’t suit dynamic logic from Arduino. |
| Solid-State Relays | Great but **expensive** and sometimes overkill for simple on/off tasks. |

👉 Mechanical relays offer **complete electrical isolation**, can handle **high currents**, and are **inexpensive and beginner-friendly**.

**Applications**

* Home automation (lights, fans)
* Industrial control (machine on/off control)
* Emergency stops in motors and systems
* Switching between multiple power sources or loads

**Use Case in This Project**

1. **Relay Between Motor & Motor Driver Output:**
   * Ensures the motor only gets power **after** the system has homed.
   * Can instantly **stop the motor** by cutting the circuit during emergency or shutdown.
2. **Relay Between SMPS and Homing Sensor:**
   * Activates only during **homing sequence**.
   * Keeps the sensor off otherwise, **reducing power draw** and **avoiding noise or misreads**.

**Example**

Imagine the relay as a **remote-controlled circuit breaker**.  
You (Arduino) push a button remotely, and the breaker connects or disconnects the motor or sensor power.

**Analogy**

Think of it like a **traffic police officer**—he either allows or blocks cars (current) from flowing down a road (circuit) based on higher authority orders (Arduino logic).

**How It’s Used in the Project**

* Controlled using **digital output pins** from the Arduino.
* The relay has 3 output terminals: **COM**, **NO (Normally Open)**, and **NC (Normally Closed)**.
* In this project:
  + Arduino sets the relay HIGH → closes circuit between COM and NO → power flows.
  + Arduino sets relay LOW → circuit opens → power cut.
* **Typical wiring**:
  + VCC and GND for relay board
  + Signal pin to Arduino
  + Switch side connects the SMPS or Motor Driver circuit

**Extra Insights**

* Most relay modules come with onboard transistors and flyback diodes to protect the Arduino from coil voltage spikes.
* Can hear a *click* when switching—a handy **debug cue**!
* Relays introduce **small delays** in switching, but it's negligible for your door project.
* Use **optocoupler-isolated** relay boards if working with sensitive electronics or AC loads (for better safety).

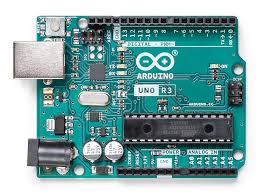
**Component 5: Arduino Uno R3 (Main Microcontroller)**

**Why Did We Use This Component?**

The Arduino Uno R3 is the **heart** of your project. It handles all logic decisions, sensor inputs, motor control, safety features, and signal processing. We chose it because it’s:

* **Beginner-friendly** with an easy-to-learn IDE
* Well-documented and widely supported
* Sufficient for the I/O and logic needs of this project
* Stable and reliable for real-world embedded applications

💡 It’s perfect for controlling your automatic door because it can easily interface with sensors, relays, motor drivers, and buttons — **all in one board**.



**Why This Compared to Alternatives?**

| **Board** | **Why Not Used?** |
| --- | --- |
| **Arduino Nano** | Compact but lacks enough I/O pins and easier debugging access |
| **Arduino Mega** | Overkill; too many pins and bulkier for this simple setup |
| **Raspberry Pi** | Powerful, but runs on an OS — not real-time, can lag or crash. Also consumes more power |
| **ESP32** | Great for Wi-Fi/Bluetooth projects, but requires more complex coding and power regulation |

The Uno R3 strikes a balance between **simplicity and power**. Plus, it supports **Input Pull-Up** logic out of the box, which is essential for your button inputs.

**Applications**

* Automation projects (like yours!)
* Robotics and motor control
* IoT (with added shields)
* Educational kits and prototypes
* Sensor data acquisition

**Use Case in This Project**

* **Takes inputs** from:
  + Encoder (via interrupt)
  + Homing sensor
  + PIR motion sensor
  + Open and Close buttons
* **Drives outputs** to:
  + Motor driver (via DIR and PWM)
  + Relay modules
  + Serial Monitor (debugging)
* **Handles logic** for:
  + Homing
  + Emergency stops
  + Auto-close delay logic
  + Motion detection (PIR)
  + Debouncing & state transitions

Basically, your Arduino is acting like a **mini real-time operating system (RTOS)** managing everything in sync.

**Example**

Just like a conductor of an orchestra, the Arduino reads each input, processes decisions instantly, and cues the “players” (motor, relays) to act.

**Analogy**

Think of the Arduino Uno as your **brain**, the sensors as your **senses**, and the motor as your **muscles**. You (Arduino) take in sensory data, decide whether to open/close the door, and command the muscles to move accordingly.

**How It’s Used in the Project**

* It powers the **logic level circuit** (buttons, sensors)
* Uses **digital pins** to control relays and read inputs
* Uses **interrupts** to track encoder pulse counts for accurate door positioning
* Calculates **auto-close timers**
* Outputs debugging info to Serial Monitor, helping with real-time feedback
* **Communicates using logic HIGH/LOW and PWM signals**

**Extra Insights**

* The Uno R3 has a **16 MHz clock**, which is more than enough for responsive real-time control in this project.
* Comes with **ATmega328P** chip — a solid 8-bit microcontroller known for reliability.
* **Can be powered via USB or external 7–12V input**.
* Supports external libraries for encoder, servo, sensor interfacing, etc.
* It’s also super easy to **clone or duplicate** this setup — making it a good prototype base.

**Component 6: Breadboard**

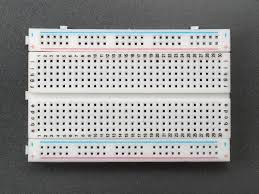
*(Used for GND pin expansion and signal distribution)*

**Why Did We Use This Component?**

The breadboard acts as a **non-permanent prototyping base**, allowing easy and flexible wiring without the need for soldering. Specifically, in your project:

* The Arduino Uno has **limited GND pins**, which isn't enough for all components.
* A breadboard allows **multiple GND lines** to be connected together via the common power rails.
* It's used to **organize wiring**, especially for sensors and buttons that share the same ground or VCC.

💡 It’s like a **power strip** for your electronic components — everyone gets a plug!



**Why This Compared to Alternatives?**

| **Alternative** | **Why Not Used?** |
| --- | --- |
| **Soldered PCB** | Not flexible, permanent — not suitable during prototyping/testing |
| **Custom PCB** | Time-consuming and expensive for a one-off academic project |
| **Jumper Wire Only** | Messy, limited GND sharing without a bus — more chances of loose connections |
| **Expansion Shields** | Adds unnecessary complexity and cost |

So, the breadboard is a **perfect match for quick, safe, and adjustable setups** — especially in academic/DIY environments.

**Applications**

* Prototyping circuits before finalizing
* Creating complex connections without soldering
* Testing multiple sensors and modules
* Debugging hardware setups quickly

**Use Case in This Project**

* Expands **GND availability** to:
  + PIR sensor
  + Proximity sensor
  + Encoder wires
  + Button circuits
* Helps you create **clean, organized wiring**
* Distributes **VCC or signal lines** to multiple inputs
* Easy to make **last-minute changes** to wiring

**Example**

Imagine the Arduino’s GND pin is a water tap and all components are thirsty plants. The breadboard becomes your **garden pipe system**, allowing everyone to get water from the same tap.

**Analogy**

It’s like using a **multi-plug adapter** at home when you have only one socket but multiple devices to connect. One input, many outputs. Simple and efficient.

**How It’s Used in the Project**

* Connected directly to **Arduino GND and VCC**
* GND rail on the breadboard powers:
  + PIR sensor
  + Homing sensor
  + Relays
  + Buttons
* Provides **central wiring hub** to keep things manageable and reduce loose connections
* Enables **quick diagnostics** (e.g., swapping sensor pins easily during testing)

**Extra Insights**

* No power limitations: the breadboard can handle the **logic-level voltage** (5V or 3.3V) without issue
* Not suitable for high-current devices like motors or SMPS outputs — so those are kept off the breadboard
* Ideal for **prototyping educational and research projects**, especially when multiple I/O lines are involved

📌 In summary, the breadboard serves as a **connection booster and layout manager**, allowing your Arduino to be the central brain without running out of “ports” to talk to other components.

**Component 7: Push Buttons (Open & Close Buttons using INPUT\_PULLUP)**

**Why Did We Use This Component?**

* To provide **manual control** over door operation.
* The user can **manually open or close** the door, regardless of automation logic (like PIR or auto-close).
* Useful for **emergency overrides**, testing, or convenience when someone doesn't trigger the PIR sensor.
* Simple and reliable hardware interface for **user interaction**.

🧠 Think of it as the “**remote control**” of your door — one for **Open**, one for **Close**.



**Why This Compared to Alternatives?**

| **Alternative** | **Why Not Used?** |
| --- | --- |
| **Touch sensors** | More expensive, less tactile feedback, harder to debug |
| **Capacitive/proximity sensors** | Can trigger falsely or be affected by humidity/dirt |
| **Rotary encoders or toggles** | Overkill for binary actions (open/close) |
| **Digital input via mobile app** | Adds complexity, requires wireless modules, less dependable for real-time control |

So, physical push buttons are: Cheap, Easy to use, Highly reliable, No false triggers

**Applications**

* Elevators (floor buttons)
* Vending machines
* Industrial machinery (start/stop)
* Gate control systems
* Door access systems

**Use Case in This Project**

* Two momentary push buttons:
  + **OPEN Button:** Starts the opening logic (if not already fully open)
  + **CLOSE Button:** Starts closing logic (if not already closed)
* Buttons use INPUT\_PULLUP mode:
  + Saves external resistors
  + Reduces noise and false triggering
  + Active LOW: Button press = LOW signal to Arduino

**Example**

Pressing the OPEN button when someone is standing by the door but not triggering the PIR sensor will open the door manually.

Also helpful when debugging — you can test movement without needing PIR trigger or sensor activation.

**Analogy**

The buttons are like the **call and close buttons in an elevator**. PIR sensor might be automatic, but buttons are there when you need **manual control**.

**How It’s Used in the Project**

* Connected between digital I/O pins and GND
* Configured with pinMode(pin, INPUT\_PULLUP);
* Inside the main loop:
  + If button reads **LOW**, it triggers either handleOpen() or handleClose()
* Supports **emergency stop logic**:
  + Pressing the opposite button while door is moving will **stop and reverse** direction
* Debounced using delay(100); to avoid false triggers

**Extra Insights**

* **Tactile feedback** improves usability over capacitive or touch systems
* Low power consumption — only active when pressed
* Input\_PULLUP reduces component count — no external resistors
* Easily replaceable, making it great for projects and academic systems

📌 In short, these push buttons are **simple yet powerful control inputs**, offering the user **instant manual override** capability, which is essential in safety-focused designs like automated glass doors.

**Component 8: PIR Sensor (4-wire Motion Detection Sensor)**

**Why Did We Use This Component?**

* To enable **hands-free automatic door operation**.
* Detects human presence/motion near the door — ideal for places where **touchless access** is preferred (e.g., hospitals, offices, labs).
* Used to **extend the open time** of the door or **trigger door opening** automatically.

🧠 Think of it as the "security guard" that watches for you and opens the door just in time.



**Why This Compared to Alternatives?**

| **Alternative** | **Why Not Chosen?** |
| --- | --- |
| **Ultrasonic sensor** | Detects objects, but may give false triggers for stationary people |
| **Laser/IR break beam** | Requires alignment and doesn’t detect slow movements |
| **Computer vision (camera)** | Overkill for simple motion detection, high power, privacy concerns |
| **Pressure mat sensor** | Only works when someone steps on it, not ideal for hands-free detection |

PIR sensors are:

* Cheap
* Passive (don’t emit waves)
* Low power
* Proven and mature tech
* Good at detecting **warm-bodied movement (humans)**

**Applications**

* Automatic lighting systems
* Alarm and security systems
* Auto-flushing washroom systems
* Retail store door openers
* Smart homes and industrial automation

**Use Case in This Project**

* PIR sensor detects if a person is **approaching or standing near the door**.
* If detected:
  + It **prevents the door from auto-closing** immediately after opening.
  + Allows the door to **stay open longer** if someone is nearby.
* Enhances **user experience** and **safety**, preventing the door from closing too quickly.

**How It Works (Basic Principle)**

* PIR = **Passive Infrared Sensor**
* Detects **infrared radiation (heat)** changes in its field of view.
* When a warm object (like a person) moves, the sensor sees this change and outputs **HIGH**.
* When no motion, output is **LOW**.

**Fun Fact:** PIR sensors actually have two halves internally — they compare infrared changes between them to detect *motion*, not just presence.

**Example**

You approach the glass door → PIR detects motion → Arduino receives HIGH → door opens or stays open longer → after a delay without motion → door closes.

**Analogy**

It’s like a **motion-sensitive lamp** in a hotel hallway — it turns on as you walk by and turns off once you leave. In our case, it’s the door that “wakes up.”

**How It’s Used in the Project**

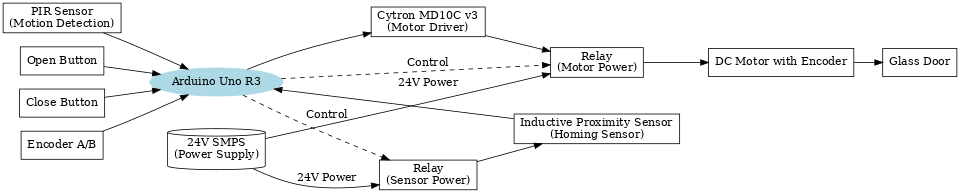
* The PIR sensor has **4 wires**:
  1. **VCC** – 5V from Arduino
  2. **GND** – Ground
  3. **OUTPUT** – Digital signal (HIGH when motion detected)
  4. **LED/Indicator** – Optional wire connected to onboard LED (some modules have this)
* It is monitored in the loop or during open logic:
  1. If PIR sensor outputs HIGH while the door is opening or open, the **auto-close timer is extended**.
  2. Prevents sudden closing while someone is still around.

**Extra Insights**

* PIR modules often have:
  + **Sensitivity adjustment** (distance detection)
  + **Time delay setting** (how long output remains HIGH)
* The **positioning** of the sensor matters:
  + It should be angled correctly to cover the entry area.
* Must avoid heat sources or moving fans nearby to **prevent false positives**.

In short, the PIR sensor is the **"smart eye"** of your door system — enabling **intelligent, human-aware interactions**, improving both **convenience** and **safety**.

**2.4. System Block Diagram**



**2.5. Comparative Analysis of Components**

| **Component** | **Alternative(s)** | **Why Selected** | **Remarks** |
| --- | --- | --- | --- |
| **DC Motor with Encoder** | Stepper Motor, Servo Motor | Encoder allows real-time feedback of speed and position for precise control. | Stepper/Servo are good but may require complex drivers or offer limited torque. |
| **Cytron MD10C v3 Driver** | L298N, L9110S | Supports high current (up to 13A), built-in protection, easy to interface. | L298N heats up quickly and has lower current capacity. |
| **Inductive Proximity Sensor** | Mechanical Limit Switch, Optical Sensor | Non-contact, durable, accurate for metallic door homing. | Works only with metal—optical or mechanical sensors are universal but less robust. |
| **Relay Module** | MOSFET switch, Solid-State Relay | Simple, inexpensive, handles high voltage switching. | Mechanical relay slower, but fine for occasional switching like this. |
| **Arduino Uno R3** | ESP32, STM32, Raspberry Pi Pico | Beginner-friendly, ample community support, sufficient I/O for this use-case. | ESP32 adds WiFi/Bluetooth but overkill here. |
| **Breadboard** | Custom PCB, Screw Terminal Board | Quick prototyping and wiring flexibility. | Not suitable for long-term deployment — fragile connections. |
| **Push Buttons (Open/Close)** | Capacitive touch, IR sensor buttons | Simple to implement with INPUT\_PULLUP, reliable tactile feedback. | Touch sensors can misfire; IR can be costly and prone to false triggers. |
| **PIR Sensor** | Ultrasonic Sensor, Laser Sensor | Low power, effective motion detection, no need for distance measurement. | Only detects movement, not proximity/distance. |
| **Wires & Jumpers** | Custom Cable Harness, Ribbon Cables | Flexible, cheap, easy to route during prototyping. | Not robust for long-term use; use harness in final version. |

**2.6. Integration Challenges and Observations**

Several challenges were encountered during the integration of components. The encoder required fine-tuning for real-time speed control, demanding precise polling and PWM logic. Isolating the sensor signals from the noise generated by the motor and power fluctuations involved careful placement of relays and power lines. Ensuring reliable homing during startup required using proximity sensors with precise calibration.

**2.7. Future Scope and Upgradability**

The system can be enhanced in future iterations by upgrading the PIR sensor to a thermal imaging sensor for multi-person detection and better sensitivity. The DC motor may be replaced with a stepper motor for finer control, and the control logic may be transferred to a more powerful microcontroller or PLC for industrial-grade applications.

# 3. Design

# 3.1. Introduction

The design of the Arduino-based automatic glass door system aims to combine automation, safety, and modularity in a real-world access control application. This chapter presents the conceptualization, planning, and execution involved in building the system. It highlights key design choices, component integration strategies, and optimization methods to ensure reliable and smooth door operation. Both hardware and software were considered from a system-level perspective to ensure interoperability and robustness.

# 3.2. Design Considerations

- Cost Efficiency: Components were chosen based on affordability without compromising core functionality.  
- Modularity: The system can be modified or scaled easily.  
- Power Constraints: Operates on 24V DC SMPS, which is safe and manageable.  
- Mechanical Constraints: The glass door requires precise control to prevent damage.  
- Safety: Emergency stop logic and PIR sensor ensure human safety.

# 3.3. Design Calculations (Simplified)

- Encoder Pulses: Used to determine maxPosition and track door travel distance.  
- Motor Power Requirement: Motor selected based on torque required to move glass door.  
- Timing: Delay logic and motion duration calculated based on encoder feedback.

# 3.4. Iterative Design Process

The project went through multiple revisions. Initially, the system was button-controlled without motion detection. Later, PIR sensors were integrated for automatic operation. Encoder feedback and speed control were implemented after initial fixed-speed trials led to abrupt starts/stops. These changes improved user safety and system efficiency.

# 3.5. Future Enhancements

- Replace Arduino Uno with ESP32 for Wi-Fi capabilities.  
- Add mobile app interface for remote door control.  
- Integrate load cell sensors for additional safety.  
- Log data such as usage frequency or maintenance alerts.

# 3.6. Risk Analysis and Mitigation

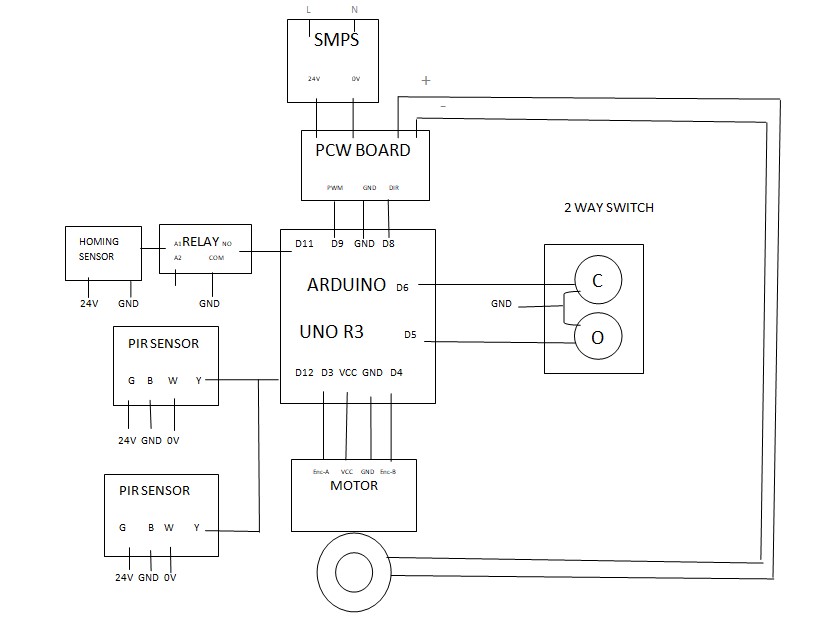
|  |  |  |
| --- | --- | --- |
| Risk | Possible Cause | Mitigation Strategy |
| Door doesn’t stop | Encoder failure | Relay override, use homing sensor fallback |
| Door doesn’t auto-close | PIR stuck or failed | Fallback to timer-based closing |
| Motor overheating | Excessive runtime | PWM control + cooling intervals |
| Unexpected opening | False PIR trigger | Add logic to verify human presence |

# 3.7. Hardware-Software Integration Strategy

The system software was modularized into control blocks for sensors, actuators, and feedback systems. Encoder interrupts were used for precise position tracking, while control logic handled safe transitions and smooth acceleration. State machines managed system behavior like homing, opening, and auto-closing.

# 3.8. Real-World Inspiration

The system design drew inspiration from commercial automatic doors used in malls and hospitals. Such systems prioritize aesthetics, seamless operation, and safety—factors that were mirrored in the current design by implementing smooth speed control, PIR-based activation, and safe hardware configurations.



# 4. Conclusion

# This project demonstrates the successful implementation of an automated door system using Arduino with real-time control logic, encoder-based position tracking, and smooth motor control using dynamic speed profiling. It also incorporates home positioning, manual override buttons, and a PIR sensor for motion detection, making the door more intelligent and context-aware.

# The use of incremental encoders to track movement allowed for precise motion control, while the S-curve-like speed control logic ensured smooth acceleration and deceleration — preventing damage due to sudden motor actions. Safety and control features such as emergency stops on conflicting commands, automatic door closing, and open signal delay extension were implemented to mimic real-world smart systems.

# The system was designed with adaptability in mind — capable of responding to human presence and interactions dynamically. While the maximum door travel (maxPosition) was estimated manually for this prototype, future iterations can improve upon this by integrating limit switches, current sensors, or calibration routines to detect the limits automatically.

# Overall, this project provided hands-on experience with embedded systems, real-time logic handling, user safety mechanisms, and debugging techniques. It serves as a strong foundational example of mechatronics in action — combining mechanical movement with electronic intelligence for smarter automation.

# References

[1] A. Banzi and M. Shiloh, *Getting Started with Arduino*, 3rd ed. Sebastopol, CA: Maker Media, 2014.  
(Reference for Arduino programming and hardware basics.)

[2] Atmel Corporation, *ATmega328P Datasheet*, San Jose, CA: Atmel, 2016. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-DataSheet.pdf  
(Used for understanding Arduino Uno’s microcontroller specifications.)

[3] S. Gupta, R. Kumar, and D. Verma, "Design and Implementation of Automatic Door Opening System Using PIR Sensor," *International Journal of Engineering Research and Technology (IJERT)*, vol. 3, no. 5, pp. 2295–2298, May 2014.  
(Reference for motion sensing with PIR in automation.)

[4] J. Norberto Pires, *Industrial Robots Programming: Building Applications for the Factories of the Future*, Springer, 2007.  
(Used for motion profiling concepts such as trapezoidal and S-curve velocity profiles.)

[5] K. Ogata, *Modern Control Engineering*, 5th ed., Prentice Hall, 2010.  
(For PID control theory and system stability concepts relevant to smooth transitions.)

[6] P. A. Tipler and G. Mosca, *Physics for Scientists and Engineers*, 6th ed., New York: W.H. Freeman, 2007.  
(For mechanical force, torque, and acceleration behavior in motors.)

[7] R. J. Goldstein, "Design and Application of Rotary Encoders in Automation Systems," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 10, pp. 3791–3798, Oct. 2008.  
(Used to understand the encoder-based positioning in automation.)

[8] S. Mishra, R. Pathak, and P. Agarwal, "PWM Based Speed Control of DC Motor Using Microcontroller," *International Journal of Engineering Science and Technology (IJEST)*, vol. 3, no. 4, pp. 3081–3087, 2011.  
(Reference for PWM-based speed control.)