OPEN ACCESS

Ind. Jr. of Mod. Res. and Rev *PEER-REVIEWED JOURNAL* *Volume 4 Issue 2 [Feb] Year 2026*

Indian Journal of Modern Research and Reviews

This Journal is a member of the *'Committee on Publication Ethics'*

Online ISSN:2584-184X

Research Article

Industry Process Automation Using Programmable Switching

Ajay Kumar [1*], Deepali Rana [2]

[1] Department of Electrical and Instrumentation Engineering, Sant Longowal Institute of Engineering and Technology, Longowal, Sangrur, Punjab, India

[2] Department of Computer and Science Engineering, Himachal Pradesh Technical University Hamirpur, Himachal Pradesh, India

Corresponding Author: *Ajay Kumar

DOI: https://doi.org/10.5281/zenodo.18458491

Abstract

The task operates the switching mechanism of industrial masses for repeated operations with the help of a user-programmable good judgment management device. The venture uses three switches and an Arduino microcontroller 328 IC that operates the masses in three modes: set, auto and manual mode. In set mode, the load works in step with the desired time, and in auto mode, it operates in the default time. In manual mode, the consumer can manually switch the load. In this mode user can turn on the load with the help of push buttons. The popularity of the loads and mode are displayed on a liquid crystal display. This venture is useful in industries in which the loads are operated in repetitive orders and in time c program language period.

Manuscript Information

- **ISSN No:** 2584-184X
- **Received:** 26-12-2025
- **Accepted:** 23-01-2026
- **Published:** 02-02-2026
- **MRR:**4(2); 2026: 05-12
- **©2026, All Rights Reserved**
- **Plagiarism Checked:** Yes
- **Peer Review Process:** Yes

How to Cite this Article

Kumar A. Rana D. Industry Process Automation Using Programmable Switching. Indian J Mod Res Rev. 2026;4(2):05-12.

Access this Article Online

www.multiarticlesjournal.com

KEYWORDS: Automation, Microcontroller, Time-based operation, Process, Switching.

1. INTRODUCTION

This research work demonstrates the operation of this simple operation using an Arduino 328 microcontroller IC. The development of this utility calls for the configuration of this system via input switches. In industries, there are many tasks carried out that require a few repeated operations in numerous orders and time durations. For instance, positive loads need to be switched ON/OFF in unique time intervals. In an effort to reap this, the microcontroller is programmed in this kind of way that the masses may be operated in three modes: Set mode, auto mode and manual mode. In set mode, through timers, the machinery works based totally on input time set by means of the consumer, where in automobile mode it works on default time settings, and finally in the guide mode it features whilst respective switches are pressed relying on the consumer's want and flexibility.

Industrial Automation

Automation takes a step similar to mechanisation that uses selected machinery mechanisms aided by human operators for acting an assignment. Mechanisation is the guided operation of a task using powered equipment that depends on human decision-making. However, automation replaces human involvement with logical programming instructions and powerful machinery. These automation gadgets consist of Programmable Logic Controllers (PLCs), Personal Computers (PCs), and technologies that include various industrial communication systems.

The Background & Technology

Within the absence of process automation, plant operators should manually screen, perform a cevalues and the best outputs to determine the nice settings on which to run the production system. Maintenance is completed at set periods. This typically results in operational inefficiency and hazardous running situations. Manner automation simplifies this with the assistance of sensors at heaps of spots around the plant that accumulate information on temperatures, pressures, and flows and so on. The information is saved and analysed on a laptop, and the whole plant and every piece of manufacturing equipment can be monitored on a large screen in a management room. Plant operating settings are then robotically adjusted to gain the ideal manufacturing. Plant operators can manually override the system automation structures when necessary.

Process automation using programmable switching

Process automation using programmable switching involves using software and hardware systems to automate tasks by controlling electrical switches or relays. These systems are widely used in industrial settings, smart homes, and IT infrastructure to streamline operations, reduce manual intervention, and improve efficiency.

Literature Survey

Hongchao Ji, Oliver Lenord and Dieter Schramm [1] suggest a model-driven approach for necessities engineering of commercial automation structures. It offers the ever-increasing complexity of technical systems within the context of imparting requirement specs as formal models that are accurate, complete, consistent, unambiguous and clean to examine and easy to preserve. An essential trouble in this region is the dearth of a conventional and standardised modelling language that covers the entire requirements engineering process from requirement specification, allocation, to verification. A model-driven requirements engineering process for commercial packages within the subject of automation structures is described so as to expose shortcomings in the latest modelling gear and modelling languages. Special attention is layed at the requirements definition and the automated verification of the layout against the requirements, using an executable fashion. Primarily based on the analysis, a new profile of the Unified Modelling Language (UML) called Model Driven Requirements Engineering for Bosch Rexroth (MDRE4BR) is presented, which aims to contribute to brand new investigations on this subject.

Minal Nikose, Pratibha Mishra and Avinash Agrawal [2] show the industrial process automation by Zigbee-based Wireless Remote Controller. Proper use of wireless sensor networks (WSNs) can lower the rate of catastrophic failures and increase the efficiency and productivity of industrial operations. Diversification of remote-control mode is the inevitable trend in the development of smart appliances. This paper proposes a review on remote control system of smart appliances based on the ZigBee wireless sensor network. The status of the home appliances can be queried and controlled through the remote controller. The proposed work presents the design and implementation of a novel wireless sensor network-based home security system with a modular self- re configurable remote controller.

Bhosale Kiran Uttam, Jadhav PappuShivaji, Prof.Pisal.R.S. et al. [3] mentioned that internet of Things (IoT) turned into a rapidly growing technology. IOT became the network of physical gadgets or matters embedded with digital software, sensors, and community connectivity, which enables those objects to acquire and change data. In this paper, they have been developing a gadget to be able to automatically display the commercial programs and generate alerts/Alarms or take smart choices using the concept of IoT. Protection from the leaking of raw gas and fireplaces is the most crucial requirement of domestic and industrial protection machines for humans. A traditional security machine offers the signals in terms of an alarm.

Mohd Sayeed, Syed Sayeed Ahmed, et al. [4] illustrated how to switch business masses by using a user programmable judgment manipulation tool for sequential operation. This operation is generally used for the repetitive nature of work. On this project, they have confirmed the operation of this simple operation using a microcontroller from the 8052 family. The improvement of this software requires configuration of the program through the enters switches. In industries, there have been many tasks carried out that require some repeated operation in numerous orders and time durations. As an example, positive loads need to be switched ON/OFF in precise time intervals. To obtain this, the microcontroller was programmed in this kind of way that the hundreds can be operated in three modes: Set mode, vehicle mode and manual mode. All the modes and standing of masses were displayed on an LCD. Thus, tasks accomplished by way of high-priced percent should now be achieved using a microcontroller, making the device fee powerful. Similarly, the venture will be superior by using interfacing it with a GSM modem, whereby sending an SMS to the control system, they could pick out the mode and timing remotely.

Jaikaran Singh, Mukesh Tiwari and Manish Shrivastava [5] represent that despite years of activity, truly open and intelligent control systems still seem to be a promise of the future. Agreement on common architectures and application objects is needed to raise open control systems from

exchanging raw data to the level of real interoperability of off-the- shelf components. Future control platforms and programming languages should have new built-in mechanisms that sup- port implementation of intelligent functions, such as flexible resource management and exception handling. This article argues that many of these challenges can be met by taking full advantage of emerging software engineering technologies. This also means that the modelling techniques and design practices of software engineering should be combined with the traditional ways of thinking in automation.
Niharika Thakur and Manisha Hooda [6] show the PLC and its Applications in Robotics and Automation. In a traditional industrial control system, all the control devices in the field are wired directly to each other to determine how the whole system is to be operated. The main control is in the human hands, which poses a lot of drawbacks, such as more wired connections are required, which may again lead to a large number of mechanical faults and difficulties in troubleshooting the errors. Due to these drawbacks, the PLC-based automation systems were introduced, which focused on controlling the various process control systems with the help of software and Hardware units.

Priyanka Priyadarshni and Shivam Sharma [7] represent the GUI-MATLAB Based Industrial Automation using MCU89S52. Within the ambit of wireless technology, the appearance of remote control-based devices and appliances has become the order of the day. It reduces human effort and increases efficiency. Every sector needs automation, ranging from home to industries. Automation Systems perform by allowing several devices to communicate with a central controller, which in turn communicates all information to the user or the owner of the system as per the instructions and the structure of the system. The application of such automation systems could be in areas such as heating, lighting, defence, energy management, audio and video systems, health monitoring, and entertainment. Keeping all these facts in mind, it proposes a system that is based on GUI control through a PC (personal computer) or a laptop. This paper proposes automation of appliances like fans, bulbs, motors, and fire sensors. To automate these appliances, we can use the different wireless communication media like infrared, Bluetooth, Radio frequency, RFID, GSM, DTMF and GUI- MATLAB and implemented with the help of a microcontroller-89c51 to compare the robustness and efficiency of the output to automate the industrial appliances.

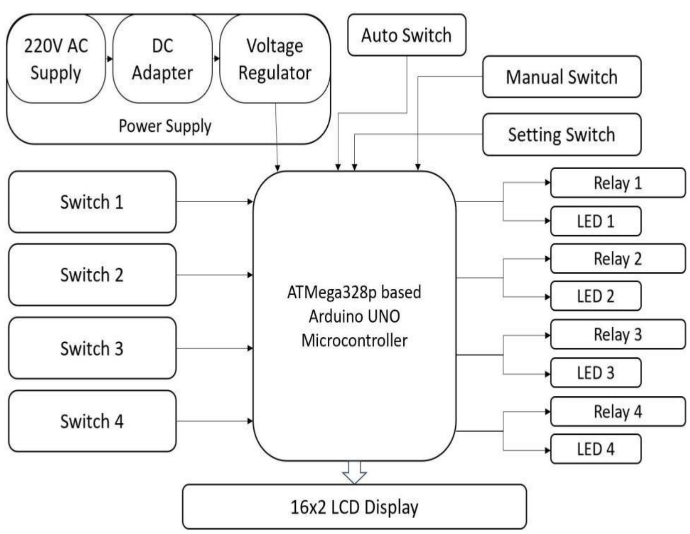
2. METHODOLOGY

This research work demonstrates an industrial process automation using programmable switches for simple industry processes. This system operates similarly to a PLC (Programmable Logic Controller) used to automate industrial functions. In this work, lamps are used as loads, which are automated via push buttons.
The system features a digital display and a start switch. There are three operational modes: Automatic, Set, and Manual. In Automatic mode, the system functions based on default time settings. When activated, one of the bulbs turns on for a specific duration, followed by the next load activating sequentially.
In Manual mode, each load is controlled manually, with the display showing the status of each load. Initially, the display indicates that all four loads are OFF, allowing users to activate the loads using the push buttons.
The Set mode allows for individual timing adjustments for each load. Users can incrementally set the duration for which each load remains ON. This approach illustrates how loads can be controlled in industrial applications through programming.

ATMEGA 328 Microcontroller IC

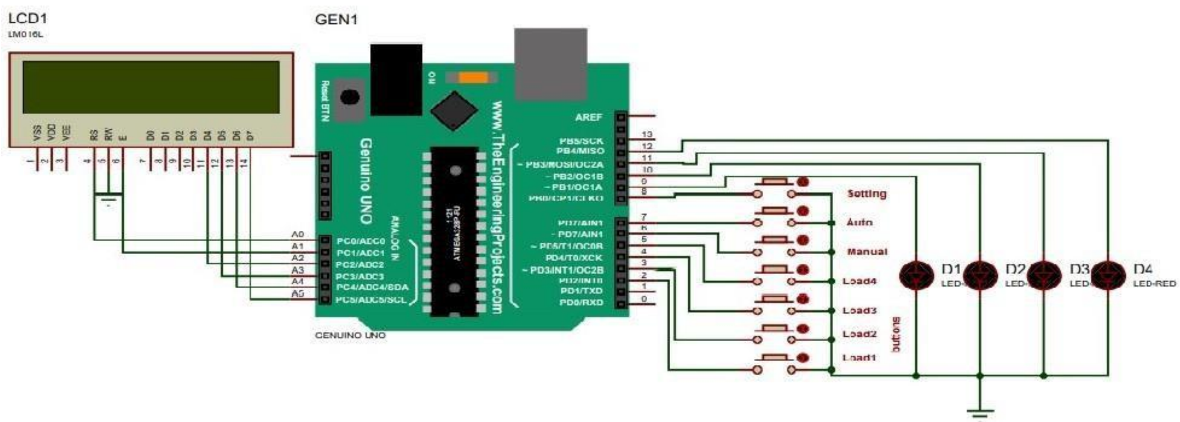
Arduino ATMEGA-328 microcontroller has been programmed for numerous packages via the use of the energy jack cable. Arduino microcontroller has been programmed so that the execution of the program may additionally takes location. Diverse varieties of Arduino boards are gift within the marketplace. Arduino software is set up within the computer and uploads the program in line with the applications. Mainly those Arduino software supports C and C++ programming languages. Numerous inputs and outputs are available within the Arduino board, and consequently, simultaneously, 8 input and output ports can be used for diverse programs. A number of the packages utilised by the usage of Arduino forums are rotating popular motor, stepper motor, control valve open, etc., the working of the Arduino microcontroller is where the proper connection is made. Checking all of the enter ports in addition to the power supply connection. The output of the pins can be connected with the external devices in keeping with their applications. This system can be executed for the applications by means of the use of the Arduino software program. From this Arduino software, we will edit in keeping with the programs. This software program can work on C and C++ programming languages. It is a fully high-level language through the use of the conditions of operating; we can create 9 software to continue for the applications.

Figure 1. Block Diagram



Then, after those packages may be uploaded via the Arduino microcontroller by using the use of the electricity jack cable. The program may be uploaded to the microcontroller and equipped for similarly sys- tem. ATMEGA-328 microcontroller can save an application, and these ICs can act as a processor to do the process without any errors. After by means of giving an analogue or virtual input to the device, we are able to do the procedure according to the packages. In this research work, the manner of the software can be controlled by editing the system within the Arduino software program, and once more may be uploaded to the Arduino microcontroller via a power jack cable. There may be a choice of a reset button.

Figure 2. Circuit Diagram



The purpose of the reset button is to reset the program. This means that the previous programs are deleted, and we can use the Arduino for alternative application purposes. Likewise, these Arduino ATMEGA-328 microcontrollers can be used for a wide variety of applications. These Arduino microcontrollers are widely used in the automation industry for controlling the system and operating the device in an automation mode. Here, I have furnished a simple Arduino application to do the manner of rotating a stepper motor for one revolution. There are many examples of applications that can be found inside the Arduino software program.

Figure 3. Complete Diagram

Technical Details

The Arduino-based Electronic Load Controller Project incorporates a microcontroller to regulate the power supplied to a load. The system includes a transformer, diodes, and a capacitor for rectification and smoothing of the input power. A ceramic capacitor and a 10K resistor are employed for noise filtering and voltage stability. The 7806-voltage regulator ensures a constant output voltage for the load. A crystal oscillator provides clock signals for precise timing operations. A 16x2 LCD and potentiometer facilitate the user interface and parameter adjustment. A 6V relay acts as a switch to control the load, while push buttons enable user input for load control. Together, these components form a robust electronic load controller capable of managing power distribution effectively.

Program Design

```
#include<LiquidCrystal.h>
LiquidCrystal lcd (14,15,16,17,18,19);
int count1=10, count2=10, count3=10, count4=10; #define load1Button 2
#define load2Button 3
#define load3Button 4
#define load4Button 5
#define manualButton 6
#define autoButton 7
#define setButton 8
#define load1 9
#define load2 10
#define load3 11
#define load4 12
#define led 13 void Auto();
```

```
void timer(int, int, int ); void setup ()
{
lcd.begin(16,2); for(int i=2;i<9;i++)
pinMode(i, INPUT_PULLUP); for(int i=9;i<14;i++)
pinMode(i, OUTPUT); lcd.print("Programable");
lcd.setCursor(0,1); lcd.print("Ligic Control");

delay(2000);
}
void loop()
{
lcd.print("S=Settings       "); lcd.setCursor(0,1);
lcd.print("M=Mannual A=Auto"); button();
}
void button()
{
if(!(digitalRead(manualButton))) manual();
else if(!(digitalRead(autoButton))) Auto();
else if(!(digitalRead(setButton))) setTime();
}
void manual()
{
rly();
lcd.clear(); lcd.print("Mannual Mode"); lcd.setCursor(0,1);
lcd.print(" Selected "); delay(2000);
lcd.clear();
lcd.print("L1= OFF L2= OFF "); lcd.setCursor(0,1);
lcd.print("L3= OFF L4= OFF ");


while (1)
{
if(!(digitalRead(load1Button)))
{
rly1(); lcd.setCursor(4,0); lcd.print("ON");
lcd.print(" "); lcd.setCursor(13,0); lcd.print("OFF");
lcd.print(" "); lcd.setCursor(4,1); lcd.print("OFF");
lcd.print(" "); lcd.setCursor(13,1); lcd.print("OFF");
lcd.print(" ");
}
else if(!(digitalRead(load2Button)))
{
rly2(); lcd.setCursor(4,0); lcd.print("OFF");
lcd.print(" "); lcd.setCursor(13,0); lcd.print("ON");
lcd.print(" ");

lcd.setCursor(4,1); lcd.print("OFF");
lcd.print(" "); lcd.setCursor(13,1); lcd.print("OFF");
lcd.print(" ");
}
else if(!(digitalRead(load3Button)))
{
rly3(); lcd.setCursor(4,0); lcd.print("OFF");
lcd.print(" "); lcd.setCursor(13,0); lcd.print("OFF");
lcd.print(" "); lcd.setCursor(4,1); lcd.print("ON");
lcd.print(" "); lcd.setCursor(13,1); lcd.print("OFF");
lcd.print(" ");
}
else if(!(digitalRead(load4Button)))
{
rly4(); lcd.setCursor(4,0); lcd.print("OFF");
lcd.print(" ");

lcd.setCursor(13,0); lcd.print("OFF");
lcd.print(" "); lcd.setCursor(4,1); lcd.print("OFF");
lcd.print(" "); lcd.setCursor(13,1); lcd.print("ON");
lcd.print(" ");
}
delay(100); button();
}
}
void setTime()
{
count1=count2=count3=count4=0; rly();
lcd.clear();
lcd.print(" Set Time "); delay(2000); lcd.clear();
lcd.print("L1= 0 L2= 0 "); lcd.setCursor(0,1); lcd.print("L3= 0
L4= 0 "); delay(1000);
while(1)
{
if(!(digitalRead(load1Button)))


{
count1++; lcd.setCursor(4,0); lcd.print(count1); lcd.print(" ");
delay(500);
}
else if(!(digitalRead(load2Button)))
{
count2++; lcd.setCursor(12,0); lcd.print(count2); lcd.print(" ");
delay(500);
}
else if(!(digitalRead(load3Button)))
{
count3++; lcd.setCursor(4,1); lcd.print(count3); lcd.print(" ");
delay(500);
}
else if(!(digitalRead(load4Button)))
{
count4++; lcd.setCursor(12,1); lcd.print(count4); lcd.print(" ");

delay(500);
}
button();
}
}
void Auto()
{
rly();
lcd.clear();
lcd.print(" Auto Mode "); lcd.setCursor(0,1); lcd.print("
Selected "); delay(2000);
while(1)
{
```

```
lcd.clear(); lcd.print("L1="); lcd.print(count1);
lcd.setCursor(10,0); lcd.print("L2="); lcd.print(count2);
lcd.setCursor(0,1); lcd.print("L3="); lcd.print(count3);
lcd.setCursor(10,1); lcd.print("L4="); lcd.print(count4);
rly1();
timer (count1, 0, 3); rly2();
timer (count2, 0, 13); rly3();
timer (count3, 1, 3); rly4();
timer (count4, 1, 13);
}
}
void rly1()
{
digitalWrite(load1, HIGH); digitalWrite(load2, LOW);
digitalWrite(load3, LOW); digitalWrite(load4, LOW);
}
void rly2()
{
digitalWrite(load2, HIGH); digitalWrite(load1, LOW);
digitalWrite(load3, LOW); digitalWrite(load4, LOW);
}
void rly3()
{
digitalWrite(load3, HIGH); digitalWrite(load2, LOW);
digitalWrite(load1, LOW); digitalWrite(load4, LOW);

}
void rly4()
{
digitalWrite(load4, HIGH); digitalWrite(load2, LOW);
digitalWrite(load3, LOW); digitalWrite(load1, LOW);
}
void rly()
{
digitalWrite(load4, LOW); digitalWrite(load2, LOW);
digitalWrite(load3, LOW); digitalWrite(load1, LOW);
}
void timer(int count, int row, int col)
{
for(int i=count;i>=0;i--)
{
lcd.setCursor(col,row); lcd.print(i);
lcd.print(" "); button(); delay(1000);
}
}
```

This code is for a programmable logic control (PLC) system using an Arduino with an LCD and button inputs to manage four loads (devices). It operates in either "Manual" or "Auto" mode and allows for the configuration of timers for each load.

LCD Display:
The LCD is initialised using the Liquid Crystal library and connected to pins 14-19. It shows the current mode and load statuses.

Buttons:
Load1 Button, load2 Button, load3 Button, and load4 Button control the four loads manually or set their timer values.
Manual Button, Auto Button, and Set Button are used to switch between Manual, Auto, and Settings modes.

Relays for Loads:
load1, load2, load3, and load4 are connected to relays (pins 9-12) that control the on/off state of each load.

LED:
Pin 13 is reserved for an LED, though it isn't directly used in the provided code.

Program Overview Manual Mode (manual ()):
In this mode, the user can manually switch the loads on and off using the load X Button. Each button press updates the LCD to show which load is active (ON) and which are off (OFF).

Set Time Mode (set time ()):
In this mode, the user can set timers for each load. When the respective load X Button is pressed, the timer for that load is incremented by one second and displayed on the LCD.

Auto Mode (Auto ()):
In this mode, the loads operate automatically based on the timers set in Set Time (). Each load will activate in sequence for the set duration, as controlled by the timer () function.

Button Handling (button ()):
This function checks the state of the buttons to determine whether to enter Manual, Auto, or Settings mode.

Relay Control (rly1 (), rly2 (), rly3 (), rly4 ()):
These functions activate the respective relay for each load and ensure that only one load is active at a time.

Timer Function (timer ()):
The timer countdown is displayed on the LCD, and once a load's timer expires, it switches to the next load.

Results and Discussion

This research work features three operational modes: Set, Auto, and Manual.

Manual Mode

In Manual Mode, each switch operates a specific lamp while ensuring that the other lamps remain off, effectively isolating the control of each lamp to its corresponding switch. Here's the discussion of the results:

Switch 1 Operation:

- When Switch 1 is pressed, Lamp 1 turns **ON**, while Lamps 2, 3, and 4 remain OFF. This setup indicates that pressing the switch activates only the designated lamp, ensuring exclusive control over the lighting circuit.

Switch 2 Operation:

- Pressing Switch 2 causes Lamp 2 to turn ON, with the other lamps (Lamp 1, 3, and 4) remaining OFF. The system maintains the same isolation principle, where only the corresponding lamp reacts to the switch.
- **Switch 3 Operation:** When Switch 3 is pressed, Lamp 3 turns ON, and the remaining lamps (Lamp 1, 2, and 4) stay OFF. This further reinforces the idea of independent operation of lamps in this manual mode.

2. Switch 4 Operation:

Switch 4 activates Lamp 4, and like before, the other lamps (Lamp 1, 2, and 3) are unaffected and remain OFF. This condition finalises the pattern of manual, exclusive lamp control per switch.

General Insights:

- This mode ensures that only one lamp can be on at a time.
- It highlights mutual exclusivity in the control logic, meaning that each switch has sole control over its respective lamp without interference or overlap.
  The setup might be useful for systems that require focused, individual control (e.g., specific task lighting scenarios).

The behaviour of the system in Manual Mode is predictable and straightforward, offering reliable and clear control.

Auto Mode

In Auto Mode, the lamps operate in a sequential manner, with a 10-second interval between each lamp's activation. Here's a breakdown of how the system works and an analysis of the results:

System Behaviour:

1. **Initial State (Time = 0 seconds)**:
- All lamps start in the OFF state **(**Zero = OFF**).** None of the lamps is illuminated at the beginning of the sequence.

2. **After 10 Seconds (Time = 10 seconds)**:
- Lamp 1 turns **ON** (One = **ON**), while the other lamps remain OFF**.**
- This indicates that after the initial 10-second timer, the system activates the first lamp in the sequence.

3. **After 20 Seconds (Time = 20 seconds)**:
- Lamp 1 turns OFF, and Lamp 2 turns **ON**. The sequence ensures that only
- Lamp 2 is illuminated after the next 10-second interval.

4. **After 30 Seconds (Time = 30 seconds)**:
- Lamp 2 turns OFF, and Lamp 3 turns ON, continuing the sequential activation of lamps.
- At this point, Lamp 3 is the only one that remains ON, and the others are all OFF**.**

5. **After 40 Seconds (Time = 40 seconds)**:
- Lamp 3 turns OFF, and Lamp 4 turns **ON**.
- Now, Lamp 4 is the only ON lamp, and all others are OFF.

6. **After 50 Seconds (Time = 50 seconds)**:
- The cycle could either reset or stop, depending on how the system is designed.

If the system **resets**, the sequence will repeat with Lamp 1 turning ON again at the 50-second mark, starting a new cycle.

3. RESULT DISCUSSION

- **Sequential Activation**: The key feature of this mode is the sequential activation of lamps, where only one lamp is on at a time, and they light up in order after fixed 10-second intervals.
- **Mutual Exclusivity**: Similar to Manual Mode, Auto Mode also maintains mutual exclusivity in operation, meaning that only one lamp is active at any given time during the sequence.
- **Timer-Controlled Operation**: This mode introduces automated timing to control the lamps without any user intervention. The predictable, time-based transitions ensure the lamps light up sequentially, creating a rhythm of light changes.
- **Application**: Such a system could be useful in scenarios requiring staggered lighting in timed intervals, such as in decorative lighting sequences, signal lights, or task reminders, where the lights represent the passage of time or specific events.

Overall, Auto Mode ensures a consistent, time-regulated behaviour where lamps turn ON and OFF in a fixed sequence with no overlap. The clear transition from one lamp to the next provides a highly organised and structured operation.

Sr. No.	Mode	Lamp				Switch				Timer (Sec.)			
		1	2	3	4	1	2	3	4	1	2	3	4
1	Manual	1	0	0	0	1	0	0	0	-	-	-	-
2		0	1	0	0	0	1	0	0	-	-	-	-
3		0	0	1	0	0	0	1	0	-	-	-	-
4		0	0	0	1	0	0	0	1	-	-	-	-
7	Auto	1	0	0	0	-	-	-	-	10	-	-	-
8		0	1	0	0	-	-	-	-	-	10	-	-
9		0	0	1	0	-	-	-	-	-	-	10	-
10		0	0	0	1	-	-	-	-	-	-	-	10

4. CONCLUSION

The belief in enterprise method automation, the usage of programmable switching, highlights its capacity to streamline operations, improve performance, and decrease manual errors. By using leveraging programmable switching technologies, industries can gain more control over their methods, leading to improved productivity and cost savings. Additionally, the adaptability of programmable switching permits easier integration with existing structures and destiny scalability, making sure long-term viability and competitiveness within the marketplace. But, it is critical for groups to cautiously plan and enforce automation techniques, considering factors like safety, reliability, and personnel readiness to completely realise the benefits of programmable switching in commercial automation.

Future Scope

The maximum seen part of current automation can be stated to be industrial robotics. A few benefits are repeatability, tighter fine management, better performance, integration with commercial enterprise structures, accelerated productiveness and reduction of labour. A few negative aspects are high capital necessities, severely decreased flexibility, and increased dependence on renovation and restoration. As an example, Japan had to scrap many of its industrial robots when they were found to be incapable of adaptation to substantially changed production requirements and so not necessarily able to justify their high initial costs.

REFERENCES

1. Ji H, Lenord O, Schramm D. Empirical evaluation of SysML through the modelling of an industrial automation unit. In: *Proceedings of the 11th IEEE International Conference on Emerging Technologies and Factory Automation*; 2006.
2. Nikose M, Mishra P, Agrawal A. A TV remote controller-based programmable home appliance manager. 2013.
3. Bhosale KU, Galande AB, Jadhav PS, Pisal RS. Industrial automation using IoT. *Int Res J Eng Technol*. 2017;4(6):205–208.
4. Sayeed M, Ahmed SS, Shaik FM, Ahmed O, Uzair MAR. Programmable switching control for industrial automation in repetitive work. 2017.
5. Singh J, Tiwari M, Shrivastava M. A new intelligent remote control system for home automation and reduced energy consumption. In: *Proceedings of the Fourth Asia International Conference on Mathematical and Analytical Modelling and Computer Simulation*; 2010.
6. Thakur N, Hooda M. Different applications of programmable logic controller. 2014.
7. Priyadarshni P, Sharma S. Remote-controlled home automation systems with different approaches. Centre for Applied Internet Research; 2006.

About the corresponding author

Ajay Kumar is a Master of Technology (M.Tech.) student in Instrumentation and Control Engineering in the Department of Electrical and Instrumentation Engineering at Sant Longowal Institute of Engineering and Technology (SLIET), Punjab, India. His academic interests include control systems, industrial automation, biomedical engineering and instrumentation, thermal imaging, deep learning, and artificial intelligence.