MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

Department of Electronics & Communication Engineering

SMART SENSORS AND SYSTEMS LABORATORY MANUAL

III B.TECH-I SEMESTER (ECE)

R24 (MLRS) REGULATION

A.Y2025-2026

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that this manual is a Bonafide record of practical work in the Smart Sensors and Systems Laboratory in **First Semester of III-year B.Tech (ECE) Program** during the academic year **2025-2026.** This book is prepared by **Mrs. R. Babitha (Assistant Professor)** Department of Electronics and Communication Engineering.

LABI/C **Head of the Department**

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

PREFACE

This laboratory lays the foundation for the Electronics and Communication Engineering students during Third year of their course.

The IoT System Design Laboratory provides hands-on experience in designing, developing, and testing end-to-end Internet of Things solutions. This laboratory enables students to interface sensors and actuators with microcontrollers and single-board computers. Learners gain practical exposure to communication protocols, cloud platforms, and data acquisition techniques. Emphasis is placed on system integration, real-time monitoring, and control applications. The lab strengthens skills in embedded programming, networking, and IoT security basics. Overall, it bridges theoretical concepts with practical IoT applications relevant to smart systems and industry needs.

By,

Mrs. R. Babitha

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ACKNOWLEDGEMENT

It was really a good experience, working with *Smart Sensors and Systems Laboratory*. First we would like to thank Dr. N. Srinivas Assoc. Professor, HOD of Department of Electronics and Communication Engineering, Marri Laxman Reddy Institute of technology & Management for his concern and giving the technical support in preparing the document.

We are deeply indebted and gratefully acknowledge the constant support and valuable patronage of Dr. Ravi Prasad, Dean, Marri Laxman Reddy Institute of technology & Management for giving us this wonderful opportunity for preparing the *Smart Sensors and Systems Laboratory* manual.

We express our hearty thanks to Dr. R. Murali Prasad, Principal, Marri Laxman Reddy Institute of technology & Management, for timely corrections and scholarly guidance.

We express our hearty thanks to Dr. P. Sridhar, Director, Marri Laxman Reddy Institute of technology & Management, for timely corrections and scholarly guidance.

At last, but not the least I would like to thanks the entire ECE Department faculty those who had inspired and helped us to achieve our goal.

By,

Mrs. R. Babitha

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

GENERAL INSTRUCTIONS

1. Students are instructed to come to *Sensors and Systems Laboratory* on time. Late comers are not entertained in the lab.
2. Students should be punctual to the lab. If not, the conducted experiments will not be repeated.
3. Students are expected to come prepared at home with the experiments which are going to be performed.
4. Students are instructed to display their identity cards before entering into the lab.
5. Students are instructed not to bring mobile phones to the lab.
6. Any damage/loss of equipment like mouse, keyboard, CPU etc., during the lab session, it is student's responsibility and penalty or fine will be collected from the student.
7. Students should update the records and lab observation books session wise. Before leaving the lab, the student should get his lab observation book signed by the faculty.
8. Students should submit the lab records by the next lab to the concerned faculty members in the staffroom for their correction and return.
9. Students should not move around the lab during the lab session.
10. If any emergency arises, the student should take the permission from faculty member concerned in written format.
11. The faculty members may suspend any student from the lab session on disciplinary grounds.
12. Never copy the output from other students. Write down your own outputs.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SAFETY MEASURES

1. While working in the laboratory suitable precautions should be observed to prevent accidents.
2. Always follow the experimental instructions strictly.
3. Use the first aid box in case of any accident/mishap.
4. Never work in the laboratory unless administrator or teaching assistant in present.
5. When the experiment is completed, students should disconnect the setup made by them, and should return all the components/instruments taken for the purpose.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

INSTITUTION VISION AND MISSION

VISION

To be a globally recognized institution that fosters innovation, excellence, and leadership in education, research, and technology development, empowering students to create sustainable solutions for the advancement of society.

MISSION

To foster a transformative learning environment that empowers students to excel in engineering, innovation, and leadership.

To produce skilled, ethical, and socially responsible engineers who contribute to sustainable technological advancements and address global challenges.

To shape future leaders through cutting-edge research, industry collaboration, and community engagement.

Quality Policy

The management is committed in assuring quality service to all its stakeholders, students, parents, alumni, employees, employers, and the community.

Our commitment and dedication are built into our policy of continual quality improvement by establishing and implementing mechanisms and modalities ensuring accountability at all levels, transparency in procedures, and access to information and actions.

DEPARTMENT VISION, MISSION, PROGRAM OUTCOMES AND PROGRAM SPECIFIC OUTCOMES

Vision of the Department

To provide quality technical education in Electronics and Communication Engineering through research, innovation, striving for global recognition in specified domain, leadership, and sustainable societal solutions.

Mission of the Department

- **DM1:** To create a transformative learning environment that empowers students in electronics and communication engineering, fostering excellence in technical skills and leadership.
- **DM2:**To drive innovation through research, deliver a transformative education grounded in ethical principles, and nurture the development of professionals
- **DM3:**To cultivate strong industry partnerships, and engaging actively with the community For societal and technological progress.

Program Outcomes (POs):

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSOs)

1. Analyze and design analog & digital circuits or systems for a given specification and function.
2. Implement functional blocks of hardware-software co-designs for signal processing and communication applications.

Course Structure:

Level	Credits	Periods/Week	Prerequisites
UG	1	2	Embedded Systems, Microcontrollers, C language

Evaluation Scheme:

MID (Internal Lab) Semester Test	30 marks
Day to day evaluation	10 marks
End Semester Lab external Examination	60 marks

The end semester examination shall be conducted with an external examiner and internal examiner.

The external examiner shall be appointed by the principal / Chief Controller of examinations.

Course Objectives:

The students will try to learn

- Fundamentals of Arduino and Raspberry Pi for IoT applications
- Interface sensors, actuators, and communication devices with microcontrollers/microprocessors
- Design and implementation of basic automation systems using Arduino and Raspberry Pi
- Analyzing of sensor data and control real-time operations for embedded IoT applications
- Programming concepts to develop smart prototypes for health, automation, and traffic control

Course Outcomes:

After successful completion of the course, students shall be able to

- Explain the fundamentals of Arduino and Raspberry Pi platforms for IoT-based applications
- Interface sensors, actuators, and communication devices with microcontrollers/microprocessors
- Design and implement automation systems using Arduino and Raspberry Pi
- Analyze sensor data to control and monitor real-time embedded IoT applications
- Apply programming skills to develop smart IoT prototypes for domains like health, automation, and traffic control

Course Outcomes (CO's)– Program Outcomes (PO's) Mapping:

- Explain the fundamentals of Arduino and Raspberry Pi platforms for IoT-based applications
- Interface sensors, actuators, and communication devices with microcontrollers/microprocessors
- Design and implement automation systems using Arduino and Raspberry Pi
- Analyze sensor data to control and monitor real-time embedded IoT applications
- Apply programming skills to develop smart IoT prototypes for domains like health, automation, and traffic control.

MAPPING OF EACH CO WITH PO(s), PSO(s):

Course Outcomes	PROGRAM OUTCOMES													
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO1	PSO 2
CO1	✓	✓	✓		✓	-	-	-	-	✓	-	✓	✓	✓
CO2	✓	✓	✓	✓	✓	-	-	-	✓	✓	-	✓	✓	✓
CO3	✓	✓	✓	✓	✓	✓	✓	-	✓	✓	✓	✓	✓	✓
CO4	✓	✓	✓	✓	✓	-	-	-	✓	✓	-	✓	✓	✓
CO5	✓	✓	✓	✓	✓	✓	✓	-	✓	✓	✓	✓	✓	✓

JUSTIFICATIONS FOR CO – PO / PSO MAPPING - DIRECT:

Course Outcomes	**PO'S/ PSO'S**	**Justification for mapping (Students will be able to)**	**No. of Key Competencies**
CO1	PO1	1. Application of scientific principles and methodologies. 2. Application of specialized engineering knowledge in complex engineering problems 3. Integration of knowledge from various engineering disciplines.	3
	PO2	1. Recognizing and defining complex engineering problems or opportunities. 2. Structuring and abstracting the problem for systematic analysis.	2
	PO3	1. Use creativity to develop innovative engineering solutions.	1

	PO5	1. Develop engineering solutions using modern tools across various disciplines. 2. Implement simulation tools in different engineering fields.	2
	PO10	1. Communicate complex engineering concepts clearly and concisely in written reports and design documentation.	1
	PO12	1. Pursue professional, Academic, Global certifications. 2. Be adaptable to technological changes by actively pursuing new learning opportunities and challenges.	2
	PSO1	1. Analyze response of a circuit or system. 2. Design of a circuit or system for a given specification. 3. Understand and apply circuit or system specifications accurately.	3
	PSO2	1. Proficiency in the use of software tools for circuit design. 2. Hardware-software integration in analog and digital systems.	2
	PO1	1. Application of scientific principles and methodologies. 2. Application of specialized engineering knowledge in complex engineering problems. 3. Integration of knowledge from various engineering disciplines.	3
	PO2	1. Recognizing and defining complex engineering problems or opportunities. 2. Structuring and abstracting the problem for systematic analysis. 3. Applying mathematical, natural, and engineering sciences in problem-solving.	3
	PO3	1. Investigate and define a problem while identifying constraints, including environmental, sustainability, health, and safety considerations. 2. Use creativity to develop innovative engineering solutions.	2

CO2	PO5	1. Utilize IT tools in engineering analysis, design, and decision-making. 2. Implement simulation tools in different engineering fields.	2
	PO9	1. Work effectively as an individual, taking ownership of tasks and driving progress independently.	1
	PO10	1. Communicate complex engineering concepts clearly and concisely in written reports and design documentation.	1
	PO12	1. Pursue professional, Academic, Global certifications. 2. Begin and work towards advanced programs to further deepen knowledge in engineering and related areas. 3. Stay updated on industry trends and emerging technologies to remain relevant in the field. 4. Learn at least 2–3 new significant skills annually to ensure continuous growth and development.	4
	PSO1	1. Analyze response of a circuit or system 2. Design of a circuit or system for a given specification 3. Understand and apply circuit or system specifications accurately.	3
	PSO2	1. Develop Operational block diagrams 2. Proficiency in the use of software tools for circuit design. 3. Hardware-software integration in analog and digital systems	3
	PO1	1. Application of scientific principles and methodologies. 2. Application of specialized engineering knowledge in complex engineering problems. 3. Integration of knowledge from various engineering disciplines.	3

CO3	PO2	1. Recognizing and defining complex engineering problems or opportunities. 2. Structuring and abstracting the problem for systematic analysis. 3. Applying mathematical, natural, and engineering sciences in problem-solving.	3
	PO3	1. Use creativity to develop innovative engineering solutions. 2. Manage the design process and evaluate outcomes for safety and risk assessment. 3. Understand the commercial and economic context of engineering processes.	3
	PO4	1. Develop essential laboratory and workshop skills to carry out experimental investigations and gather reliable data. 2. Apply fundamental engineering principles to analyze and interpret key engineering processes and challenges.	2
	PO5	1. Develop engineering solutions using modern tools across various disciplines. 2. Identify appropriate prediction and modeling tools for diverse engineering applications. 3. Utilize IT tools in engineering analysis, design, and decision-making.	3
	PO6	1. Understand the commercial and economic context of engineering processes.	1
	PO7	1. Recognize the political implications and responsibilities of engineering solutions.	1
	PO9	1. Work effectively as an individual, taking ownership of tasks and driving progress independently. 2. Demonstrate maturity by focusing on goal achievement, requiring minimal external motivation.	2

	PO10	1. Communicate complex engineering concepts clearly and concisely in written reports and design documentation. 2. Ensure high standards of grammar and punctuation in written communication, maintaining professionalism and clarity.	2
	PO11	1. Ensure the timely delivery of project outputs, meeting the predefined objectives and quality standards.	1
	PO12	1. Pursue professional, Academic, Global certifications. 2. Begin and work towards advanced programs to further deepen knowledge in engineering and related areas.	2
	PSO1	1. Analyze response of a circuit or system 2. Design of a circuit or system for a given specification 3. Understand and apply circuit or system specifications accurately.	3
	PSO2	1. Develop Operational block diagrams 2. Proficiency in the use of software tools for circuit design. 3. Hardware-software integration in analog and digital systems	3
CO4	PO1	1. Application of scientific principles and methodologies. 2. Application of specialized engineering knowledge in complex engineering problems. 3. Integration of knowledge from various engineering disciplines.	3

	PO2	1. Recognizing and defining complex engineering problems or opportunities. 2. Structuring and abstracting the problem for systematic analysis. 3. Applying mathematical, natural, and engineering sciences in problem-solving.	3
	PO3	1. Use creativity to develop innovative engineering solutions. 2. Manage the design process and evaluate outcomes for safety and risk assessment.	2
	PO4	1. Develop essential laboratory and workshop skills to carry out experimental investigations and gather reliable data. 2. Apply fundamental engineering principles to analyze and interpret key engineering processes and challenges. 3. Address complex problems in various engineering contexts, including operations, management, and technology development.	3
	PO5	1. Develop engineering solutions using modern tools across various disciplines. 2. Identify appropriate prediction and modeling tools for diverse engineering applications. 3. Utilize IT tools in engineering analysis, design, and decision-making.	3
	PO9	1.Work effectively as an individual, taking ownership of tasks and driving progress independently. 2.Demonstrate maturity by focusing on goal achievement, requiring minimal external motivation.	2
	PO10	1.Communicate complex engineering concepts clearly and concisely in written reports and design documentation. 2.Ensure high standards of grammar and punctuation in written communication, maintaining professionalism and clarity.	2

	PO12	1. Pursue professional, Academic, Global certifications. 2. Begin and work towards advanced programs to further deepen knowledge in engineering and related areas. 3. Stay updated on industry trends and emerging technologies to remain relevant in the field.	3
	PSO1	1. Analyze response of a circuit or system. 2. Design of a circuit or system for a given specification. 3. Understand and apply circuit or system specifications accurately. 4. Knowledge of analog and digital signal processing techniques.	4
	PSO2	1. Develop Operational block diagrams 2. Proficiency in the use of software tools for circuit design. 3. Hardware-software integration in analog and digital systems	3
CO5	PO1	1. Application of scientific principles and methodologies. 2. Application of specialized engineering knowledge in complex engineering problems. 3. Applying mathematical, natural, and engineering sciences in problem-solving.	3
	PO2	1. Recognizing and defining complex engineering problems or opportunities. 2. Structuring and abstracting the problem for systematic analysis. 3. Applying mathematical, natural, and engineering sciences in problem-solving.	3
	PO3	1. Use creativity to develop innovative engineering solutions. 3. Promote sustainable development through engineering activities. 2. Be aware of legal frameworks governing engineering activities, including personnel, health, safety, and environmental risks.	3

	PO4	1. Develop essential laboratory and workshop skills to carry out experimental investigations and gather reliable data. 2. Apply fundamental engineering principles to analyze and interpret key engineering processes and challenges.	2
	PO5	1. Develop engineering solutions using modern tools across various disciplines. 2. Identify appropriate prediction and modeling tools for diverse engineering applications. 3. Utilize IT tools in engineering analysis, design, and decision-making.	3
	PO6	1. Understand the commercial and economic context of engineering processes.	1
	PO7	1. Recognize the political implications and responsibilities of engineering solutions.	1
	PO9	1.Work effectively as an individual, taking ownership of tasks and driving progress independently. 2.Demonstrate maturity by focusing on goal achievement, requiring minimal external motivation.	2
	PO10	1.Communicate complex engineering concepts clearly and concisely in written reports and design documentation. 2.Ensure high standards of grammar and punctuation in written communication, maintaining professionalism and clarity.	2
	PO11	1.Ensure the timely delivery of project outputs, meeting the predefined objectives and quality standards.	1
	PO12	1. Pursue professional, Academic, Global certifications. 2. Begin and work towards advanced programs to further deepen knowledge in engineering and related areas. 3. Stay updated on industry trends and emerging technologies to remain relevant in the field.	3

	PSO1	1. Analyze response of a circuit or system 2. Design of a circuit or system for a given specification 3. Understand and apply circuit or system specifications accurately. 4. Knowledge of analog and digital signal processing techniques.	4
	PSO2	1. Develop Operational block diagrams 2. Proficiency in the use of software tools for circuit design. 3. Hardware-software integration in analog and digital systems	3

TOTAL COUNT OF KEY COMPETENCIES FOR CO – (PO, PSO) MAPPING:

Course Outcomes	PROGRAM OUTCOMES												PSOs	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
	4	**10**	**10**	**10**	**4**	**5**	**4**	**4**	**10**	**5**	**10**	**8**	**4**	**4**
CO1	3	2	1	-	2	-	-	-	-	1	-	2	3	2
CO2	3	3	2	1	3	-	-	-	1	1	-	2	3	3
CO3	3	3	3	2	3	-	-	-	2	2	-	2	3	3
CO4	3	3	2	3	3	-	-	-	1	2	-	2	4	3
CO5	3	3	3	2	3	-	-	-	2	2	-	3	4	3

PERCENTAGE OF KEY COMPETENCIES FOR CO – (PO/ PSO):

Course Outcomes	PROGRAM OUTCOMES												PSOs	
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**
	4	**10**	**10**	**10**	**4**	**5**	**4**	**4**	**10**	**5**	**10**	**8**	**4**	**4**
CO 1	50	45	10	20	50	-	-	-	-	20	-	87.5	75	50
CO 2	50	50	20	30	75	-	-	-	10	20	-	87.5	75	75
CO 3	50	50	30	30	75	-	-	-	20	40	-	87.5	75	75
CO 4	50	50	20	30	75	-	-	-	10	40	-	87.5	100	75
CO 5	50	50	30	30	75	-	-	-	20	40	-	87.5	100	75

COURSE ARTICULATION MATRIX (PO – PSO MAPPING):

CO'S and PO'S, CO'S and PSO'S on the scale of 0 to 3, 0 being no correlation, 1 being the low correlation, 2 being medium correlation and 3 being high correlation.

0 - 0≤ C≤ 5% – No correlation,

2 - 40 % <C< 60% –Moderate

1-5 <C≤ 40% – Low/ Slight

3 - 60% ≤ C < 100% – Substantial /High

Course Outcomes	PROGRAM OUTCOMES												PSOs	
	PO 1	PO 2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO 2
CO 1	3	2	1	-	2	-	-	-	-	1	-	2	3	2
CO 2	3	3	2	1	3	-	-	-	1	1	-	2	3	3
CO 3	3	3	3	2	3	-	-	-	2	2	-	2	3	3
CO 4	3	3	2	3	3	-	-	-	1	2	-	2	4	3
CO 5	3	3	3	2	3	-	-	-	2	2	-	3	4	3
TOTAL	**15**	**14**	**11**	**8**	**14**	**-**	**-**	**-**	**6**	**8**	**-**	**11**	**15**	**14**
AVERAGE	**3**	**2.8**	**2.2**	**1.6**	**2.8**	**-**	**-**	**-**	**1.2**	**1.6**	**-**	**2.2**	**3**	**2.8**

Department of Electronics and Communication Engineering

ROOMNO:

LAB TIME TABLE

NAME OF THE LAB: Smart Sensors and Systems Lab **A.Y:**2026-2027

BRANCH: ECE **W.E.F:**

SEMESTER: I

PERIOD	1st	2nd	3rd		4th	5th	6th
TIME / DAY	09:40 - 10:35	10:35 - 11:30	11:30 - 12:25	12:25 - 01:15	01:15 - 02:10	02:10 - 03:05	03:05 - 04:00
MON				L			
TUE				U			
WED				N			
THU				C			
FRI				H			
SAT							

Time Table I/C **Time Tables C/D** **HOD–ECE**

MARRI LAXMAN REDDY INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AUTONOMOUS)

2525573: EMBEDDED REAL TIME OPERATING SYSTEM LABORATORY

III Year B.Tech. ECE I – Sem.

L	T	P	C
0	0	2	1

Couse Overview:

This course introduces Arduino and Raspberry Pi for IoT applications through hands-on experiments. Students learn to interface sensors, actuators, and communication devices, and design basic automation systems.

Course Objectives:

The students will try to learn:

The students will try to learn

- Fundamentals of Arduino and Raspberry Pi for IoT applications
- Interface sensors, actuators, and communication devices with microcontrollers/microprocessors
- Design and implementation of basic automation systems using Arduino and Raspberry Pi
- Analyzing of sensor data and control real-time operations for embedded IoT applications
- Programming concepts to develop smart prototypes for health, automation, and traffic control

Course Outcomes:

After successful completion of the course, students shall be able to

- Explain the fundamentals of Arduino and Raspberry Pi platforms for IoT-based applications
- Interface sensors, actuators, and communication devices with microcontrollers/microprocessors
- Design and implement automation systems using Arduino and Raspberry Pi
- Analyze sensor data to control and monitor real-time embedded IoT applications
- Apply programming skills to develop smart IoT prototypes for domains like health, automation, and traffic control

List of Experiments:

The following experiments are performed using any Software (Free/Open recommended like Arduino IDE / Tinkercad Circuits / PlatformIO/ Thonny / VS Code / Geany etc)

1. Blinking of LED using Arduino
2. Interfacing of LED with Push Button using Arduino
3. Interfacing DHT11- Temperature and humidity sensor using Arduino
4. Interfacing Ultrasonic sensor using Arduino
5. Interfacing PIR sensor using Arduino
6. Interfacing 16×2 LCD with Arduino
7. Design of Traffic Light Simulator using Arduino
8. Water Level Monitoring System using Arduino
9. ON/OFF Control of Electrical Appliances using Relay and Arduino
10. Buzzer Control using Push Button using Arduino
11. Raspberry Pi Programming Using Python (Simulation-Based)
12. Control of LED using Push Button on Raspberry Pi with Python
13. Interfacing DHT11-Temperature and Humidity Sensor with Raspberry Pi
14. Design of Motion Sensor Alarm using PIR Sensor

NOTE: Minimum of 12 experiments to be conducted.

Open ended Experiments:

1. Design a home automation system using Raspberry Pi
2. Design and implement a Smart Health Monitoring System using IoT
3. Design a Smart Traffic Light Controller using Raspberry Pi and sensors.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SMART SENSORS AND SYSTEM LABORATORY LAB PLANNER

S.No	Experiment	CO	Date planned	Date conducted
1	Blinking of LED using Arduino	1		
2	Interfacing of LED with Push Button using Arduino	1		
3	Interfacing DHT11- Temperature and humidity sensor using Arduino	2		
4	Interfacing Ultrasonic sensor using Arduino	2		
5	Interfacing PIR sensor using Arduino	2		
6	Interfacing 16×2 LCD with Arduino	3		
7	Design of Traffic Light Simulator using Arduino	3		
	MID 1			
8	Design of Traffic Light Simulator using Arduino	3		
9	Water Level Monitoring System using Arduino	4		
10	ON/OFF Control of Electrical Appliances using Relay and Arduino	4		
11	Buzzer Control using Push Button using Arduino	4		
12	Control of LED using Push Button on Raspberry Pi with Python	5		
13	Interfacing DHT11-Temperature and Humidity Sensor with Raspberry Pi	5		
14	Design of Motion Sensor Alarm using PIR Sensor	5		
	MID 2			

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SMART SENSORS AND SYSTEMS LABORATORY

RUBRICS USED TO ASSESS LEARNINGS IN LABORATORIES

1. RUBRICS FOR DAYTO DAYEVALUATION

Parameter	Max Marks	Level-1 (VeryPoor)	Level-2 (Poor)	Level-3 (Average)	Level-4 (Good)	Level-5 (Excellent)
Observation Book	05	No observations orirrelevant data. (0–1)	Incomplete orincorrect data. (2)	Basicvalues with some errors. (3)	Mostly correctwith good format. (4)	Fullycorrect, clear, and well-formatted. (5)
Record Writing	05	Not submitted.(0-1)	Submitted but mostly incomplete. (2)	Submittedwith some missing/wrong parts. (3)	Submitted withminor issues. (4)	Fully complete, correct algorithm& flowchart.(5)
Result	05	No result or majorerrors. (0–1)	Result partially obtained.(2)	Acceptable result with limitederror.(3)	Near-correct result and reasonable error.(4)	Accurate result.(5)
Viva-Voce	05	Did not answer any questions.(1)	Answered very few questions. (2)	Answeredsome questions with help. (3)	Answered most questions correctly. (4)	Answeredall questions accurately. (5)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SMART SENSORS AND SYSTEMS LABORATORY

2. RUBRICS FOR INTERNAL EVALUATION

Criterion	Max Marks	Level-1 *(VeryPoor)*	Level-2 *(Poor)*	Level-3 *(Average)*	Level-4 *(Good)*	Level-5 *(Excellent)*
Design/Tool/Apparatus Selection	2 Marks	Incorrect tool/design and no reasoning. **(0)**	Tool/design selection attempted with unclear logic. **(0.5)**	Satisfactory selection with partial justification. **(1)**	Correct selection andproper analysis with few errors. **(1.5)**	Smart selection with accurate, relevant analysis. **(2)**
Execution (Code/Debug/Run) /Analysis/Method Used	4 Marks	Did not attempt or completely failed to execute.**(0)**	Attempted but unableto proceed or with major errors. **(1)**	Partial execution withsome logic/syntax errors. **(2)**	Mostly correct execution with minimal help. **(3)**	Fullycorrect and independently executed program. **(4)**
Results& Documentation	2 Marks	Incomplete or poorly presented. **(0)**	Basic structure butlacks clarityor formatting. **(0.5)**	Complete but genericor with formatting issues. **(1)**	Well-structured andmostly clear. **(1.5)**	Well-organized, professional, andengaging documentation.**(2)**
Viva-Voce (Understanding of Concepts)	2 Marks	No understanding; could not answer questions. **(0)**	Answered a few with difficulty. **(0.5)**	Answered half the questions withbasic clarity. **(1)**	Good understanding with confident answers. **(1.5)**	Answered allquestions with clarity and depth. **(2)**

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SMART SENSORS AND SYSTEMS LABORATORY

3. RUBRICS FOR SEMESTER END EXAMINATIONS

Criterion	Max Marks	Level-1 (VeryPoor *(0–2 marks)*	Level-2 (Poor)*(3–4 marks)*	Level-3 (Average)*(5–6 marks)*	Level-4 (Good)*(7–9 marks)*	Level-5 (Excellent)*(10–12marks)*
Preparedness for the Experiment	12 marks	Noclarityon objective or procedure. Unable to explain basics.	Limited idea of the objective/procedure. Needed prompting.	Has basic understanding; minor gaps in concept or preparation.	Well-prepared, with clear understanding of steps and background.	Fully preparedwith strong conceptual clarity and confident explanation.
Performance in the Laboratory	12 marks	Unable to perform experiment. Relied entirelyon examiner's help.	Performedwith multiple errors and constant support.	Performed withsome errors; required occasional help.	Performed mostly independently with minimal support.	Performed independently ,efficiently, and with precision.
Calculations & Graphs	12 marks	No or incorrect calculations. Graphs missingor irrelevant.	Multiple calculationerrors. Graphs/plots inaccurate or poorlylabelled.	Calculations partially correct. Graphspresent but with some flaws.	Correct calculations and graphs with minor errors.	Accurate calculations and well-labelled graphs with proper interpretation.
Results&Error Analysis	12 marks	No result or invalidresult. Noerror analysis attempted.	Incorrect result withvagueorno errordiscussion.	Acceptable result. Error analysis attemptedbut limited.	Correct result with sounderror discussion.	Accurate result with detailed and relevanterror analysis.
Viva-Voce (Subject Knowledge)	12 marks	Unable to answer any questions.No conceptual understanding.	Answeredfew questionswith poor logic.	Answeredhalf of the questionswith average understanding.	Answered most questions withclarity and confidence.	Answeredall questions with depth, clarity, and reasoning.

INDEX

12	Control of LED using Push Button on Raspberry Pi with Python	
13	Interfacing DHT11-Temperature and Humidity Sensor with Raspberry Pi	
14	Design of Motion Sensor Alarm using PIR Sensor	

Experiment 1

BLINKING LED USING ARDUINO

AIM: Blinking LED using Aurdino.

Components Required:

1. Arduino controller board,
2. USB connector,
3. Bread board, LED, 1.4K ohm resistor,
4. Connecting wires,
5. Arduino IDE

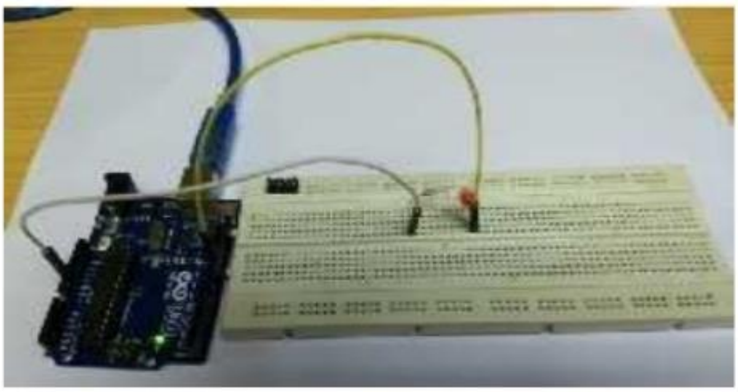
Theory:

Arduino can control digital output pins to turn devices ON or OFF. By toggling the pin state with a delay, an LED can be blinked.

Procedure:

1. Connect the LED to the Arduino using the Bread board and the connecting wires.
2. Connect the Arduino board to the PC using the USB connector;
3. Select the board type and port.
4. Write the sketch in the editor, verify and upload.
5. Connect the positive terminal of the LED to digital pin 12 and the negative terminal to the ground pin (GND) of Arduino Board.

Circuit Diagram:



Program:

```
void setup()
{
pinMode (12, OUTPUT); // set the pin mode
}
void loop ()
{
digitalWrite (12, HIGH); // Turn on the LED delay (100);
digitalWrite (12, LOW); //Turn of the LED delay (100);
}
```

Results:

LED blinks continuously with a delay of 100 nano second.

Viva Questions:

- What is Arduino?

→ Arduino is an open-source microcontroller development platform.

- Why is a resistor used with LED?

→ To limit current and protect the LED.

- What does digitalWrite() do?

→ Sets a digital pin HIGH or LOW.

- What is the function of delay()**?**

→ Pauses program execution for a specified time.

Experiment 2

Interfacing of LED with Push Button using Arduino

Aim: To Interfacing of LED with Push Button using Arduino.

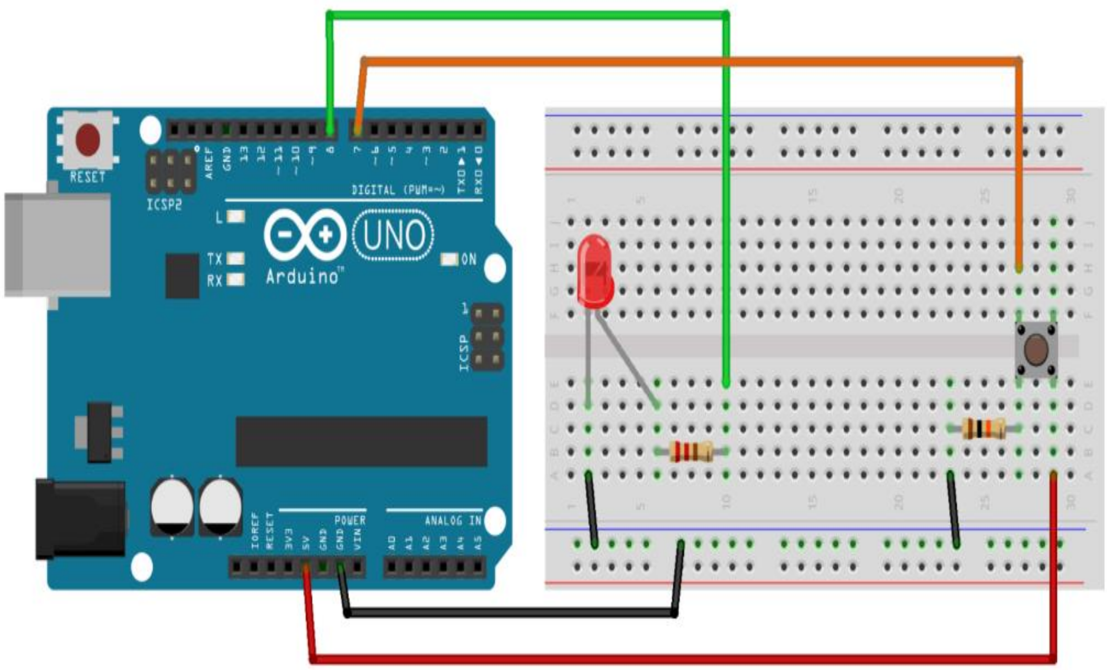
Components:

- Arduino Breadboard.
- LED – any color.
- Push button.
- 220 Ohm resistor for the LED.
- 10k Ohm resistor for the push button.
- A bunch of male to male wires.

Theory:

A push button provides digital input to Arduino. Based on input state, Arduino controls the LED.

Circuit Diagram:



Procedure:

- First, make sure to power off your Arduino – remove any USB cable.
- Plug a black wire between the blue line of the breadboard and a ground (GND) pin on the Arduino board.

- Plug the LED. You can notice that the LED has a leg shorter than the other. Plug this shorter leg to the ground (blue line here) of the circuit.
- Connect the longer leg of the LED to a digital pin (here pin no 8, you can change it). Add a 220 Ohm resistor in between to limit the current going through the LED.
- Add the push button to the breadboard, like in the picture.
- Connect one leg of the button to the ground, and put a 10k Ohm resistor in between. This resistor will act as a "pull down" resistor, which means that the default button's state will be LOW.
- Add a red wire between another leg of the button and VCC (5V).
- Finally, connect a leg of the button (same side as the pull down resistor) to a digital pin (here 7).

Program:

```
#define LED_PIN 8
#define BUTTON_PIN 7
void setup() {
pinMode(LED_PIN, OUTPUT);
pinMode(BUTTON_PIN, INPUT);
}
void loop() {
if (digitalRead(BUTTON_PIN) == HIGH) {
digitalWrite(LED_PIN, HIGH);
}
else {
digitalWrite(LED_PIN, LOW);
}
}
```

Results:

An Arduino circuit with an LED and a push button is designed successfully.

Viva Questions:

 What type of input is a push button?

→ Digital input.

- Why is a pull-down resistor used?

→ To avoid floating input condition.

- What is floating input?

→ An undefined voltage state at the input pin.

- Difference between INPUT and OUTPUT pins?

→ INPUT reads signals, OUTPUT sends signals.

Experiment 3

Interfacing DHT11- Temperature and humidity sensor using Arduino

Aim: Interfacing DHT11 humidity sensor with Arduino.

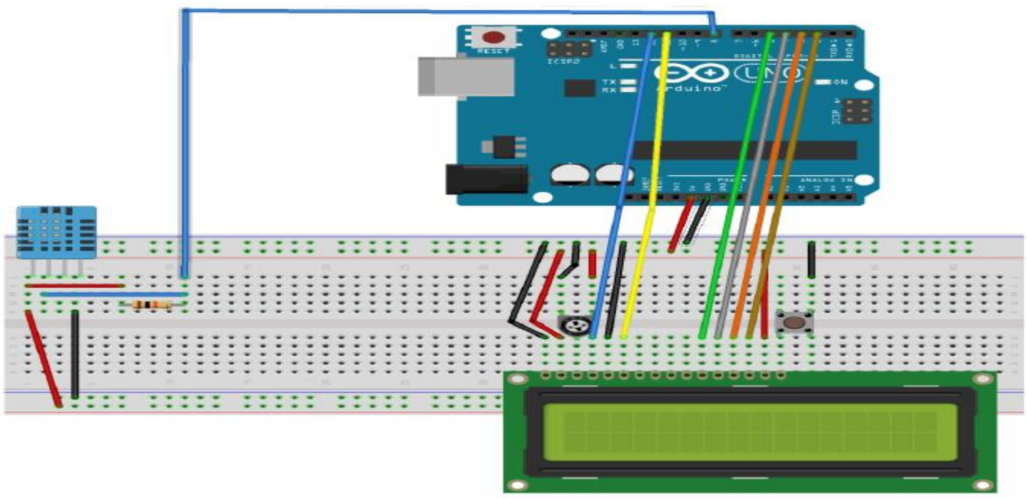
Components Required:

S.No	Component	Quantity
1	Arduino uno	1Nos
2	Bread Board	1Nos
3	16 x 2 LCD display	1Nos
4	Software Arduino IDE	1Nos
5	10 k ohm variable resistor pot	1Nos
6	DHT11 temperature and humidity sensor	1Nos
7	Usb communication cable of Arduino uno	1Nos

Theory:

DHT11 Temperature & Humidity Sensor features a temperature & humidity sensor complex with a calibrated digital signal output. By using the exclusive digital-signal-acquisition technique and temperature & humidity sensing technology, it ensures high reliability and excellent long-term stability. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness.

Circuit Diagram:



Procedure:

1. The four data Pins D4 to D7 are connected to the four pins (2 to 5) of the arduino.
2. Rs (register select) and E (Enable) pins are connected to the pin12 and pin11 of the arduino.
3. VSS pin of the LCD is connected to the ground while VDD is connected to the power supply.
4. V EE of LCD is connected to the potentiometer in order to vary the brightness of the LCD.
5. RW pin is connected to ground.
6. Download DHT11 sensor library from github and save it in the Arduino libraries and then include it in the code
7. Connect the dht11 wiring as shown in the circuit diagra

Program:

```
#include <LiquidCrystal.h> #include "DHT.h"
#define DHTPIN 8
LiquidCrystal lcd(12, 11, 5, 4, 3, 2); #define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);
void setup()
{
lcd.begin(16, 2);
dht.begin();
lcd.print("Temp: Humidity:");
}
void loop()
{
delay(500); lcd.setCursor(0, 1);
float h = dht.readHumidity();
float f = dht.readTemperature(true); if (isnan(h) || isnan(f))
{
lcd.print("ERROR"); return;
}
lcd.print(f); lcd.setCursor(7,1); lcd.print(h);
}
```

Results:

Temperature and humidity values are displayed on Serial Monitor.

Viva Questions:

- What does DHT11 measure?

→ Temperature and humidity.

- Is DHT11 analog or digital?

→ Digital sensor.

- What is the temperature range of DHT11?

→ 0°C to 50°C.

- Why delay is required in DHT11?

→ For stable sensor readings.

Experiment 4

Interfacing Ultrasonic sensor using Arduino

AIM: Measure distance using Ultrasonic sensor with an Arduino board.

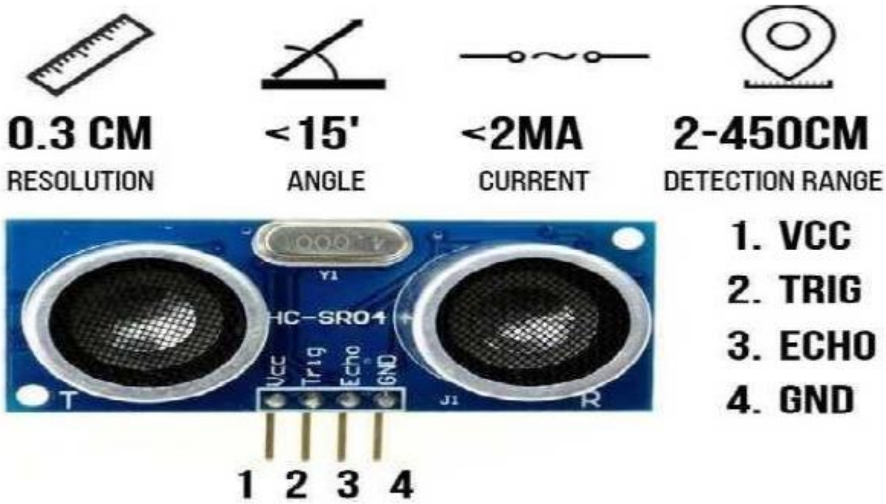
Components Required:

1. Arduino uno
2. Ultrasonic Sensor HC-SR04
3. Male to female Jumper Wires
4. breadboard

Theory:

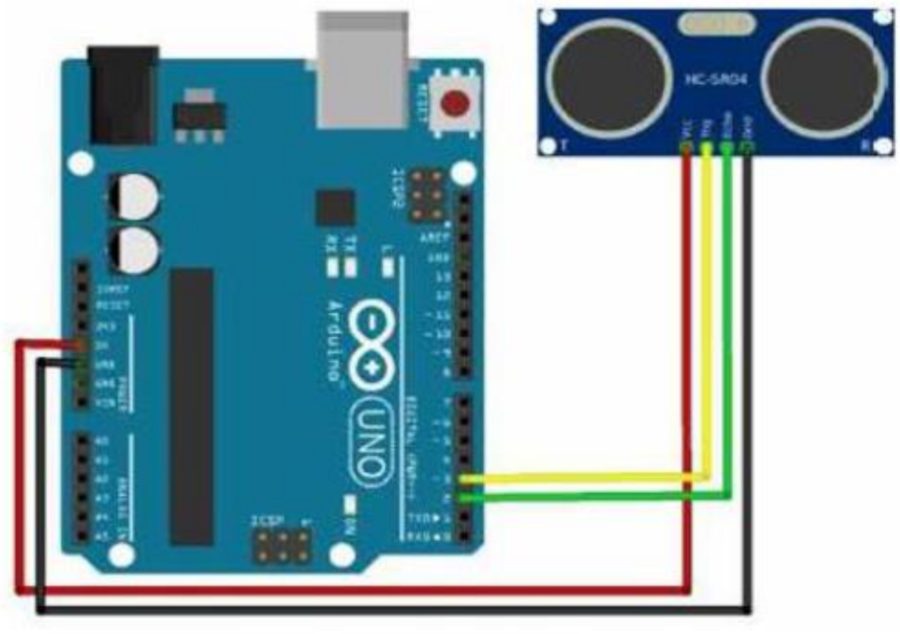
Ultrasonic Sensor HC-SR04:

Ultrasonic Sensor HC-SR04 is a sensor that can measure distance. It emits an ultrasound at 40 000 Hz (40kHz) which travels through the air and if there is an object or obstacle on its path It will bounce back to the module. Considering the travel time and the speed of the sound you can calculate the distance.



The configuration pin of HC-SR04 is VCC (1), TRIG (2), ECHO (3), and GND (4). The supply voltage of VCC is +5V and you can attach TRIG and ECHO pin to any Digital I/O in your Arduino Board.

Circuit Diagram:



Procedure:

The connection of Arduino and Ultrasonic Sensor HC-SR04

1. Connect the circuit as shown in the picture.
2. Open Arduino IDE Software and write down the code.
3. Choose Arduino board (in this case Arduino Uno), by selecting Tools > Board > Arduino/Geniuno Uno
4. Choose COM Port (usually it appears only one existing port), Tools > Port > COM. (If there are more than one ports, try it one by one
5. Sketch > Upload
6. To display the measurement data in Serial Monitor.

Program:

```
// defines pins numbers const int trigPin = 9; const int echoPin = 10;
// defines variables long duration;
int distance; void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input Serial.begin(9600); // Starts the
serial communication
```

}

void loop() {

// Clears the trigPin digitalWrite(trigPin, LOW); delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds digitalWrite(trigPin, HIGH);

delayMicroseconds(10); digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds duration =

pulseIn(echoPin, HIGH);

// Calculating the distance distance= duration*0.034/2;

// Prints the distance on the Serial Monitor Serial.print("Distance: "); Serial.println(distance);

}

Results:

Distance is displayed.

Viva Questions:

- What principle does ultrasonic sensor work on?

→ Echo and time-of-flight principle.

- What are trigger and echo pins?

→ Trigger sends pulse, echo receives reflected signal.

- Speed of sound used in calculation?

→ 343 m/s.

- Unit of distance measured?

→ Centimeters or meters.

Experiment 5
Interfacing PIR sensor using Arduino

Aim: Intruder detection using PIR sensor using Arduino

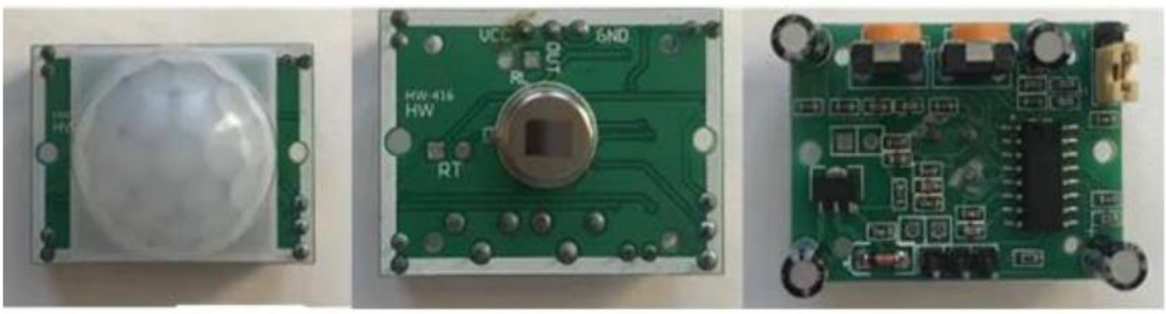
Components Required:

S.No	Component	Quantity
1.	Arduino uno	1Nos
2.	LED	1Nos
3.	Bread Board	1Nos
4.	PIR sensor	1Nos
5.	Usb communication cable of Arduino uno	1Nos

Theory:

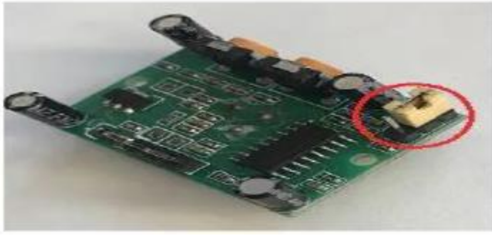
Working of PIR Sensor:

PIR sensor is a special type sensor which is usually used for security purposes. It detects the objects by reading the Infrared radiations emitted by the objects. Any object whose temperature is above absolute zero, emits radiation. This radiation is not visible to human eyes. The PIR sensor is designed to detect this Infrared radiation.

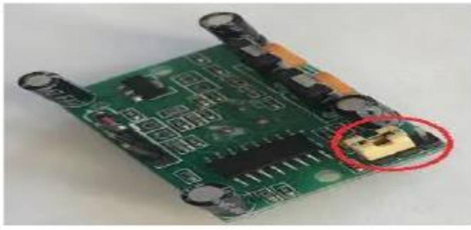


The PIR sensor has two modes. You can switch between these modes by interchanging the jumper behind the PIR sensor as shown in the images below.

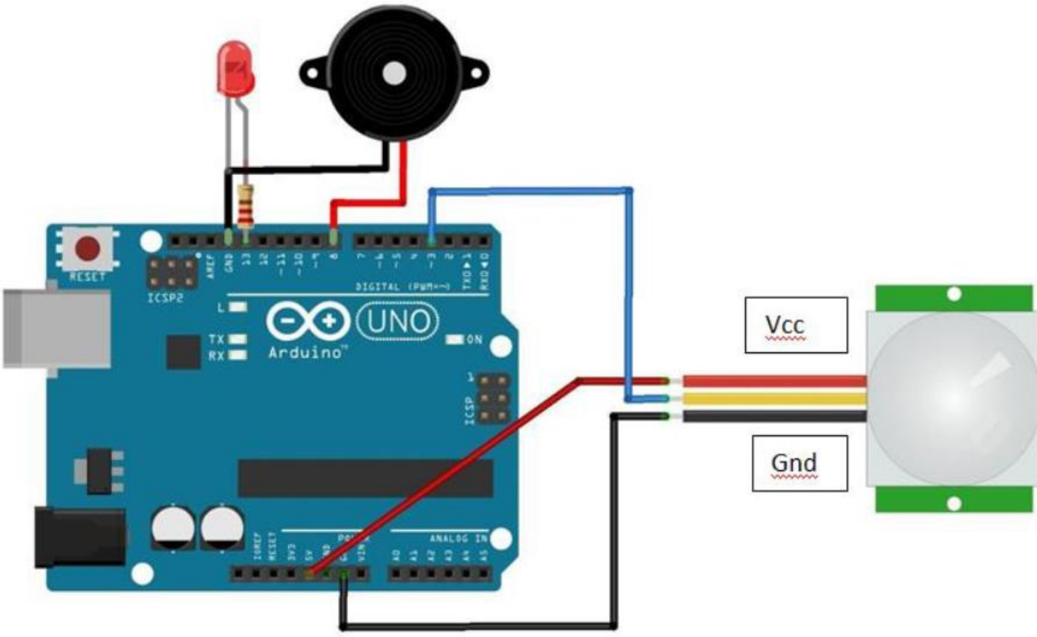
1. Single Trigger mode



2. Repeatable Trigger mode



Circuit Diagram:



Procedure:

1. Interfacing PIR sensor with Arduino.
2. 1. PIR to Arduino

- Connect the Vcc of PIR to 5V on Arduino
- Connect the GND of PIR to GND on Arduino
- Connect the OUTPUT pin of PIR to Digital pin D3 on Arduino

3. Buzzer to Arduino

- Connect one pin of buzzer to digital pin D8 on Arduino
- Connect other pin of buzzer to GND on Arduino

4. LED to Arduino

- Connect the LED positive to Digital pin D13 on Arduino through a resistor.
- Connect the LED negative to GND on Arduino.

Program:

```
int Buzz= 8; // Define Buzzer
 pin int LED= 13; // Define LED
pin int PIR= 3; // Define PIR pin
int val= 0; // Initializing the value as zero at the beginning void setup()
{
pinMode(Buzz, OUTPUT);
pinMode(LED, OUTPUT);
pinMode(PIR, INPUT);
Serial.begin(9600);
}
void loop()
{
val = digitalRead(PIR); // The value read from PIR pin if(val == HIGH)
{
digitalWrite(LED, HIGH); // Turn LED ON digitalWrite(Buzz, HIGH); // T
Serial.println("Movement Detected"); // Print this text in Serial Monitor
}
else
{
}
digitalWrite(LED, LOW); digitalWrite(Buzz, LOW);
Serial.println("Movement not Detected");
}
```

Results:

LED turns ON when motion is detected.

Viva Questions:

1. What does PIR stand for?

   → Passive Infrared Sensor.
2. What does PIR detect?

   → Motion of warm objects.
3. Is PIR active or passive sensor?

   → Passive sensor.
4. Typical applications of PIR sensor?

   → Motion alarms, automatic lights.

Experiment 6

Interfacing 16×2 LCD with Arduino

Aim: To display text on LCD.

Components Required:

1. 16 x 2 LCD.
2. Arduino
3. Arduino IDE
4. Jumper Wires

Theory:

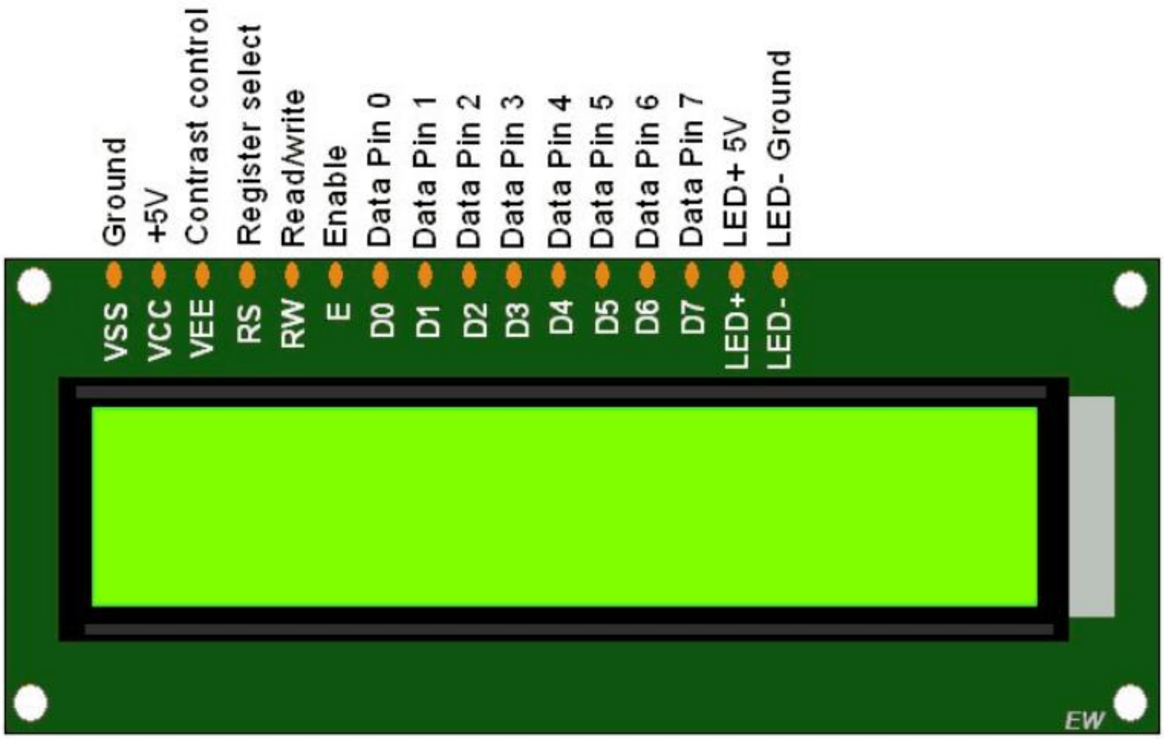
LCDs (Liquid Crystal Displays) are used in embedded system applications for displaying various parameters and status of the system.

LCD 16x2 is a 16-pin device that has 2 rows that can accommodate 16 characters each.

LCD 16x2 can be used in 4-bit mode or 8-bit mode.

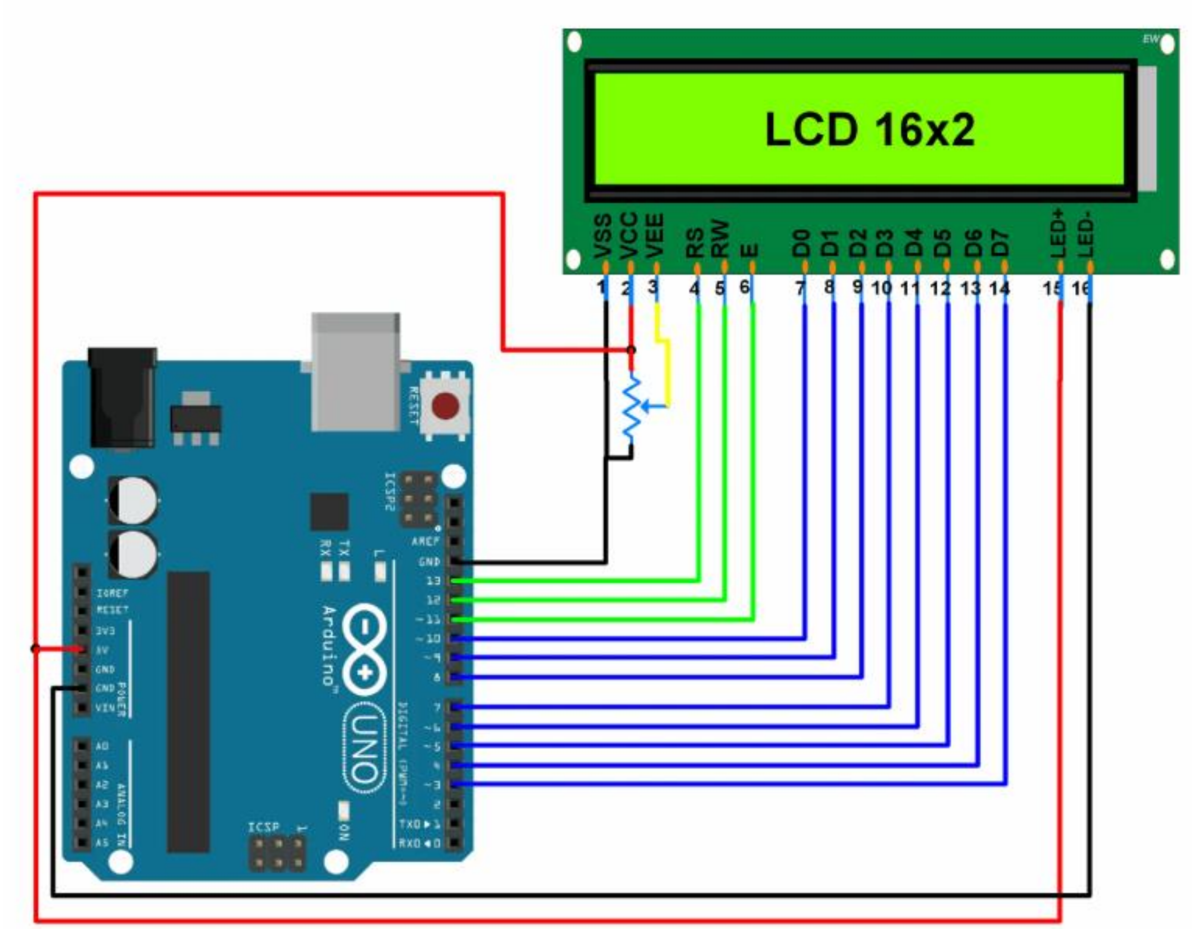
It is also possible to create custom characters.

It has 8 data lines and 3 control lines that can be used for control purposes.



LCD 16x2

Circuit Diagram:



Procedure:

- Connect the VSS and VDD pins of the 16×2 LCD to GND and +5V of the Arduino respectively.
- Connect the V0 (contrast pin) of the LCD to the middle pin of a 10 kΩ potentiometer; connect the other two ends of the potentiometer to +5V and GND.
- Connect the RS (Register Select) pin of the LCD to digital pin 12 of the Arduino.
- Connect the E (Enable) pin of the LCD to digital pin 11 of the Arduino.
- Connect the LCD data pins D4, D5, D6, and D7 to Arduino digital pins 5, 4, 3, and 2 respectively.
- Connect the RW pin of the LCD to GND to enable write mode.
- Connect the LED+ (backlight anode) of the LCD to +5V through a current-limiting resistor and LED− to GND.
- Open the Arduino IDE and include the LiquidCrystal library.
- Write the program to initialize the LCD in 16×2 mode and display the required message.
- Select the correct Board (Arduino Uno) and COM Port from the Tools menu.
- Upload the program to the Arduino board.
- Adjust the potentiometer to set proper contrast and observe the text displayed on the LCD.

Program:

```
#include <LiquidCrystal.h>
/* Create object named lcd of the class LiquidCrystal */
LiquidCrystal lcd(13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3);  /* For 8-bit mode */
//LiquidCrystal lcd(13, 12, 11, 6, 5, 4, 3);      /* For 4-bit mode */
```

```
unsigned char Character1[8] = { 0x04, 0x1F, 0x11, 0x11, 0x1F, 0x1F, 0x1F, 0x1F };        /*
Custom Character 1 */
unsigned char Character2[8] = { 0x01, 0x03, 0x07, 0x1F, 0x1F, 0x07, 0x03, 0x01 };        /*
Custom Character 2 */

void setup() {
  lcd.begin(16,2);                                    /* Initialize 16x2 LCD */
  lcd.clear();                                        /* Clear the LCD */
  lcd.createChar(0, Character1);        /* Generate custom character */
  lcd.createChar(1, Character2);
}

void loop() {
  lcd.setCursor(0,0);                         /* Set cursor to column 0 row 0 */
  lcd.print("Hello!!!!");                     /* Print data on display */
  lcd.setCursor(0,1);
  lcd.write(byte(0));                         /* Write a character to display */
  lcd.write(1);

}
```

Results:

Text displayed on LCD.

Viva Questions:

 What does 16×2 LCD mean?
→ 16 characters per line and 2 lines.

 What is RS pin?
→ Register Select pin.

 Why LCD uses contrast pin?
→ To adjust display visibility.

- Which library is used for LCD?

→ LiquidCrystal.h

Experiment 7

Design of Traffic Light Simulator using Arduino

AIM: Simulate a traffic light using an Arduino and LED’s.

Components Required:

S.No	Component	Quantity
1.	Arduino uno	1Nos
2.	LED (Red, Yellow & Green)	3Nos
3.	Bread Board	1Nos
4.	USB communication cable of Arduino uno	1Nos
5.	100ohm resistor	3Nos

Procedure:

Step 1: Supply power to the breadboard.

Step 2: Adding the LEDs.

Step 3: Completing the circuit.

Step 4: Take another jumper wire, put it on the same row that you have an LED on.

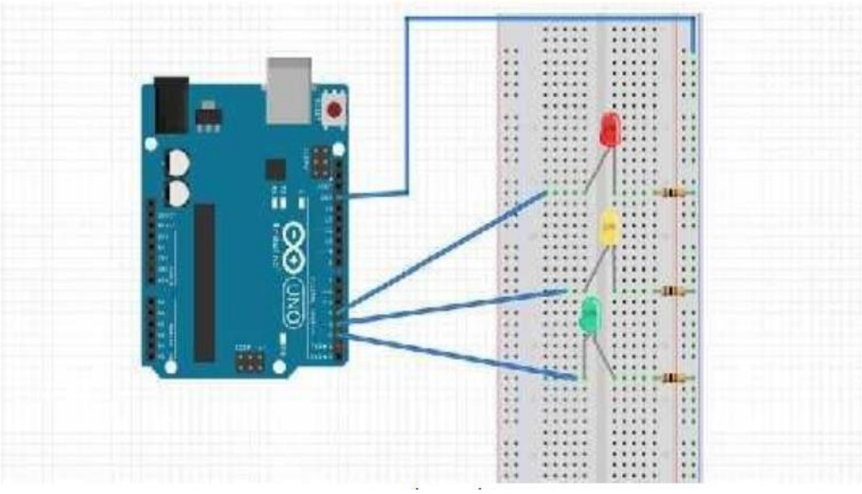
This is where the wires will go:

Green LED: Port 2, Digital PWM section

Yellow LED, Port 3, Digital PWM section

Red LED, Port 4, Digital PWM section

Circuit Diagram:



Program to Interface Traffic Lights:

```
// variables
int GREEN = 2; int YELLOW = 3; int RED = 4;
int DELAY_GREEN = 5000; int DELAY_YELLOW = 2000; int DELAY_RED = 5000;
// basic functions
void setup()
{
pinMode(GREEN, OUTPUT); pinMode(YELLOW, OUTPUT); pinMode(RED, OUTPUT);
}
void loop()
{
green_light(); delay(DELAY_GREEN); yellow_light(); delay(DELAY_YELLOW);
red_light(); delay(DELAY_RED);
}
void green_light()
{
digitalWrite(GREEN, HIGH); digitalWrite(YELLOW, LOW); digitalWrite(RED, LOW);
}
```

```
void yellow_light()
{
digitalWrite(GREEN, LOW); digitalWrite(YELLOW, HIGH); digitalWrite(RED, LOW);
}
void red_light()
{
digitalWrite(GREEN, LOW); digitalWrite(YELLOW, LOW); digitalWrite(RED, HIGH);
}
```

Results:

Traffic lights operate sequentially.

Viva Questions:

- How many LEDs are used in traffic signal?

→ Three (Red, Yellow, Green).

- What does red signal indicate?

→ Stop.

- What is the purpose of delay in traffic lights?

→ To simulate real-time signal timing.

- Can traffic light be automated using sensors?

→ Yes.

Experiment 8

Water Level Monitoring System using Arduino

AIM: use one water flow sensor with an Arduino board.

Components Required:

1. Arduino uno
2. Water flow sensor
3. breadboard cables

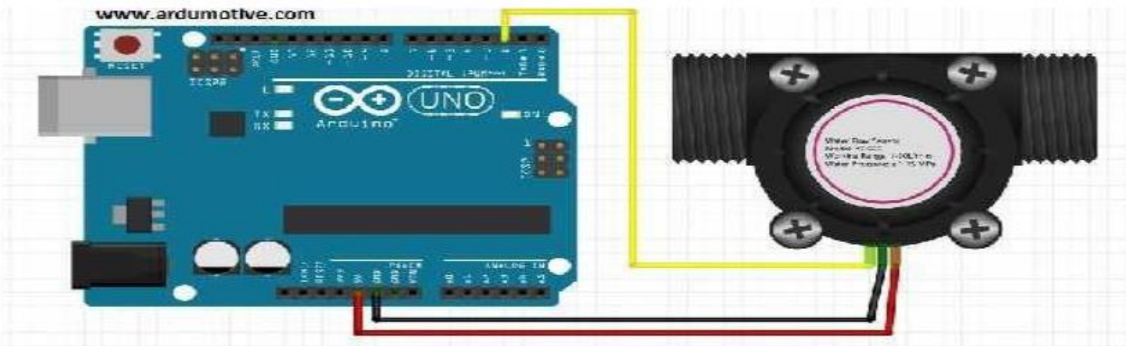
Water flow Sensor:

The water flow sensor consists of a plastic valve body, a water rotor and a hall-effect sensor. When the water flows through the rotor, rotor rolls and the speed of it changes with a different rate of flow. The hall-effect sensor outputs the corresponding pulse signal.

This type of sensor can be found on different diameters, water pressure (MPa) and flow rate (L/m) ranges. Make sure to select one that will cover your needs. The sensor that I have it has 20mm diameter, <1.75Mpa water pressure and ~30 L/m flow rate range.

In this, the serial monitor for printing the water flow rate in liters per hour and the total of liters flowed since starting. Press the connect button below to start the serial communication. Connect this sensor with your water tap, or just blow on it.

Circuit Diagram:



Program to Interface:

```
byte statusLed = 13; byte sensorInterrupt = 0; byte sensorPin= 2;
float calibrationFactor = 4.5; volatile byte pulseCount; float flowRate;
unsigned int flowMilliLitres; unsigned long totalMilliLitres; unsigned long oldTime;
void setup()
{
```

```
Serial.begin(9600);
pinMode(statusLed, OUTPUT);
digitalWrite(statusLed, HIGH); pinMode(sensorPin, INPUT); digitalWrite(sensorPin,
HIGH); pulseCount= 0;
flowRate= 0.0; flowMilliLitres= 0; totalMilliLitres= 0;
oldTime= 0; attachInterrupt(sensorInterrupt, pulseCounter, FALLING);
}
void loop()
{
if((millis() - oldTime) > 1000)
{
detachInterrupt(sensorInterrupt);
flowRate = ((1000.0 / (millis() - oldTime)) * pulseCount) / calibrationFactor; oldTime =
millis();
flowMilliLitres = (flowRate / 60) * 1000; totalMilliLitres += flowMilliLitres; unsigned int
frac;
Serial.print("Flow rate: "); Serial.print(int(flowRate)); Serial.print("L/min"); Serial.print("\t");
Serial.print("Output Liquid Quantity: "); Serial.print(totalMilliLitres); Serial.println("mL");
Serial.print("\t"); // Print tab space
Serial.print(totalMilliLitres/1000); Serial.print("L"); pulseCount = 0;
attachInterrupt(sensorInterrupt, pulseCounter, FALLING);
}
}
void pulseCounter()
{
pulseCount++;
}
```

Result:

Buzzer alerts at high water level.

Viva Questions:

- Which sensor is used for water level detection?

→ Float sensor / level sensor.

- Application of water level monitoring?

→ Overhead tanks, irrigation.

- Is water level monitoring analog or digital?

→ Can be both.

- What happens when water level is high?

→ Alarm or motor OFF.

Experiment 9

ON/OFF Control of Electrical Appliances using Relay and Arduino

Aim: To control an electrical appliance (ON/OFF) using a relay module interfaced with an Arduino.

Components Required:

- Arduino Uno
- 5V Relay Module
- Electrical load (Bulb / Fan – demonstration purpose)
- Jumper wires
- Breadboard
- External power supply (if required)

Theory

A relay is an electrically operated switch that allows low-power circuits (Arduino) to control high-power devices such as electrical appliances.
The Arduino sends a digital signal to the relay module, which energizes the relay coil and switches the load ON or OFF without direct electrical contact.

5V Relay

The SRD-05VDC-SL-C relay has three high voltage terminals (NC, C, and NO) which connect to the device you want to control. The other side has three low voltage pins (Ground, Vcc, and Signal) which connect to the Arduino.

- NC: Normally closed 120-240V terminal
- NO: Normally open 120-240V terminal
- C: Common terminal
- Ground: Connects to the ground pin on the Arduino
- 5V Vcc: Connects the Arduino's 5V pin
- Signal: Carries the trigger signal from the Arduino that activates the relay

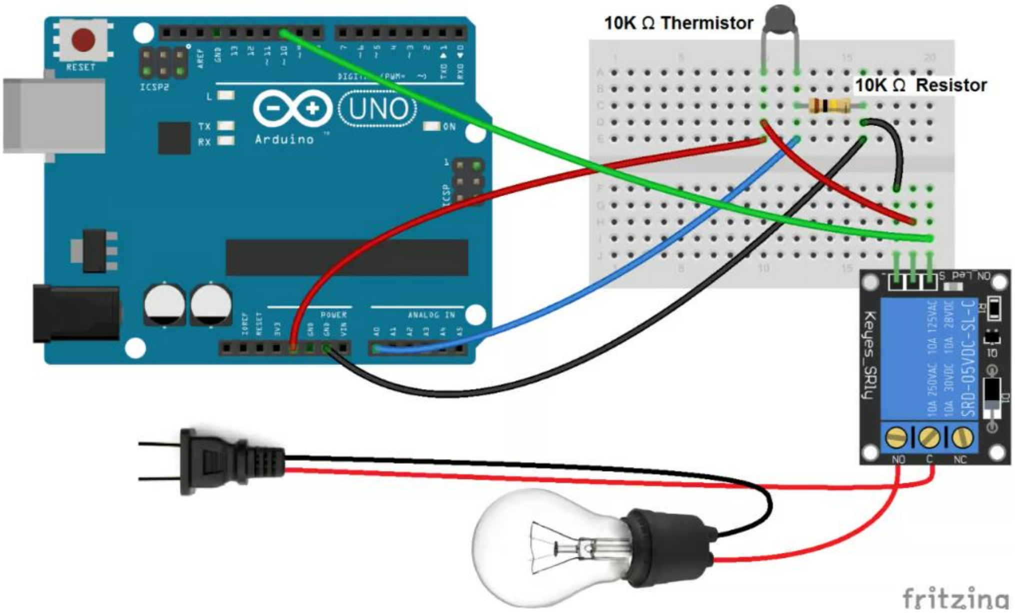
Inside the relay is a 120-240V switch that's connected to an electromagnet. When the relay receives a HIGH signal at the signal pin, the electromagnet becomes charged and moves the contacts of the switch open or closed.

Procedure

1. Connect the **VCC** pin of the relay module to 5V of Arduino.
2. Connect the **GND** pin of the relay module to GND of Arduino.
3. Connect the **IN** pin of the relay module to digital pin 8 of Arduino.

4. Connect the appliance to the COM and NO terminals of the relay.
5. Open Arduino IDE and write the program for relay control.
6. Select the correct Board and COM port.
7. Upload the program to the Arduino.
8. Observe the appliance turning ON and OFF at defined intervals.

Circuit Diagram:



Program:

```
int relayPin = 8;
void setup()
{
  pinMode(relayPin, OUTPUT);
}
void loop()
{
  digitalWrite(relayPin, HIGH); // Relay ON
  delay(2000);
  digitalWrite(relayPin, LOW);  // Relay OFF
  delay(2000);
}
```

Result

The electrical appliance is successfully turned ON and OFF using a relay controlled by Arduino.

Viva Questions

1. What is a relay?
2. Why is a relay used instead of directly connecting appliances?
3. What is the function of NO and NC terminals?
4. Can Arduino directly drive high-voltage devices?
5. What is electrical isolation?

EXPERIMENT–10

Buzzer Control Using Push Button Using Arduino

Aim: To control a buzzer using a push button interfaced with an Arduino.

Components Required

- Arduino Uno
- Push button
- Buzzer
- 10 kΩ resistor
- Jumper wires
- Breadboard

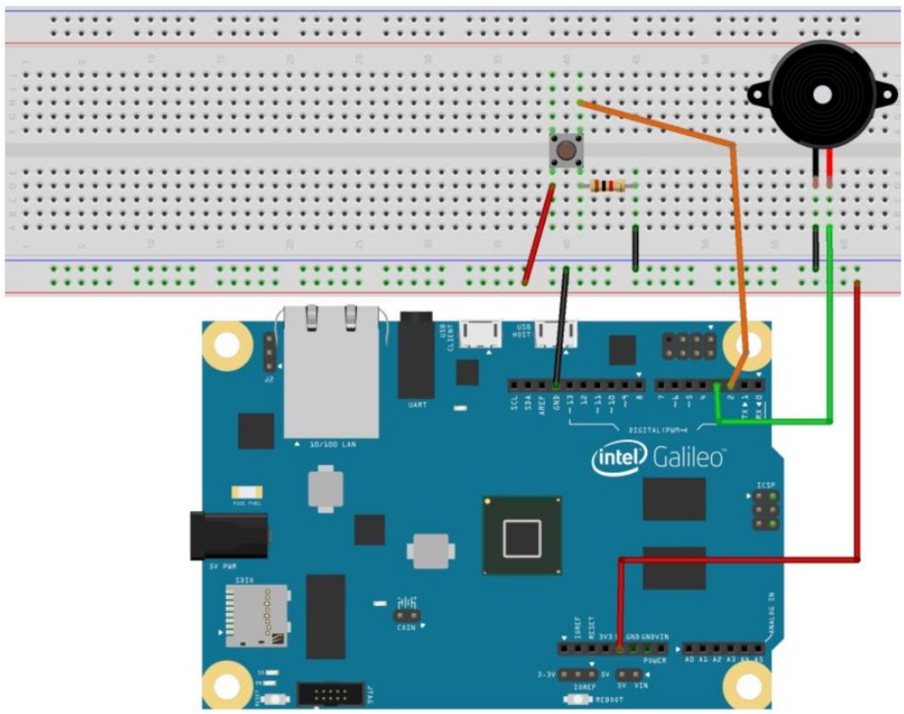
Theory:

A push button is a digital input device. When pressed, it sends a HIGH or LOW signal to Arduino.
Based on the button state, Arduino controls the buzzer, making it sound when the button is pressed.

Procedure

1. Connect one terminal of the push button to **5V**.
2. Connect the other terminal to **digital pin 7** of Arduino through a **10 kΩ resistor**.
3. Connect the buzzer positive terminal to **digital pin 9** and negative to **GND**.
4. Open Arduino IDE and write the program.
5. Select correct **Board** and **COM port**.
6. Upload the program to Arduino.
7. Press the button and observe the buzzer operation.

Circuit Diagram:



Program

```
int buttonPin = 7;
int buzzerPin = 9;

void setup()
{
  pinMode(buttonPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
}

void loop()
{
  int buttonState = digitalRead(buttonPin);

  if (buttonState == HIGH)
  {
    digitalWrite(buzzerPin, HIGH);
  }
  else
```

```
  {
    digitalWrite(buzzerPin, LOW);
  }
}
```

Result

The buzzer sounds when the push button is pressed and stops when released.

Viva Questions

1. What type of input is a push button?
2. Why is a pull-down resistor used?
3. What happens if the resistor is not used?
4. What is digitalRead()?
5. Can a buzzer be driven directly from Arduino?

Experiment 11

Raspberry Pi Programming Using Python

Aim: Develop a program to control GPIO pins of a Raspberry Pi for interfacing an LED using Python.

Apparatus / Software Required:

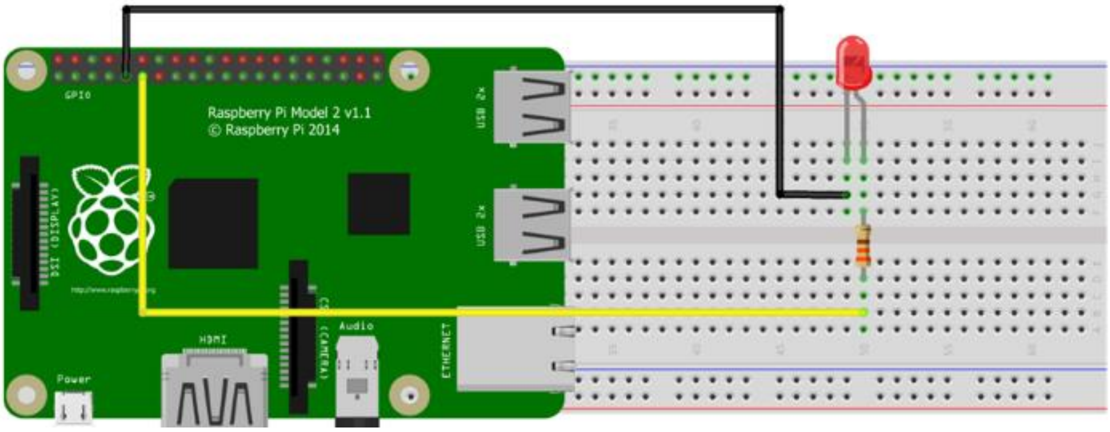
Sl. No.	Component / Software	Specification
1	Single-Board Computer	Raspberry Pi 3 / 4
2	Power Adapter	5V, 3A
3	Micro SD Card	16 GB (Raspberry Pi OS)
4	LED	5 mm
5	Resistor	330 Ω
6	Breadboard & Jumper Wires	—
7	Monitor, Keyboard, Mouse	—
8	Programming Language	Python 3
9	Operating System	Raspberry Pi OS

Theory

A Single-Board Computer (SBC) is a complete computer built on a single circuit board that includes a microprocessor, memory, I/O interfaces, and networking capabilities. The Raspberry Pi is a popular SBC widely used for embedded systems, IoT, and automation applications.

The GPIO (General Purpose Input Output) pins allow the Raspberry Pi to interact with external hardware such as LEDs, sensors, and motors. Using Python and the RPi.GPIO library, digital signals can be programmed to control these peripherals efficiently.

Circuit DIAGRAM (Description):



Procedure

1. Insert the SD card with Raspberry Pi OS into the Raspberry Pi.
2. Connect the monitor, keyboard, mouse, and power supply.
3. Boot the Raspberry Pi and open the **Terminal**.
4. Enable GPIO access (default enabled in Raspberry Pi OS).
5. Connect the LED to GPIO pin 18 through a resistor.
6. Create a Python program using a text editor.
7. Execute the program to observe LED blinking.

Program

```
import RPi.GPIO as GPIO

import time

led_pin = 17

GPIO.setmode(GPIO.BCM)

GPIO.setup(led_pin, GPIO.OUT)

while True:

GPIO.output(led_pin, GPIO.HIGH)

time.sleep(1)
```

```
GPIO.output(led_pin, GPIO.LOW)

time.sleep(1)

GPIO.cleanup( )
```

Output

The LED connected to GPIO pin 18 blinks continuously with a delay of one second.

Viva Voce Questions

1. What is a Single-Board Computer?
2. Differentiate between microcontroller and SBC.
3. What is GPIO?
4. Why is Python preferred for SBC programming?
5. Name any two applications of Raspberry Pi.

Experiment 12

Control of LED using Push Button on Raspberry Pi with Python

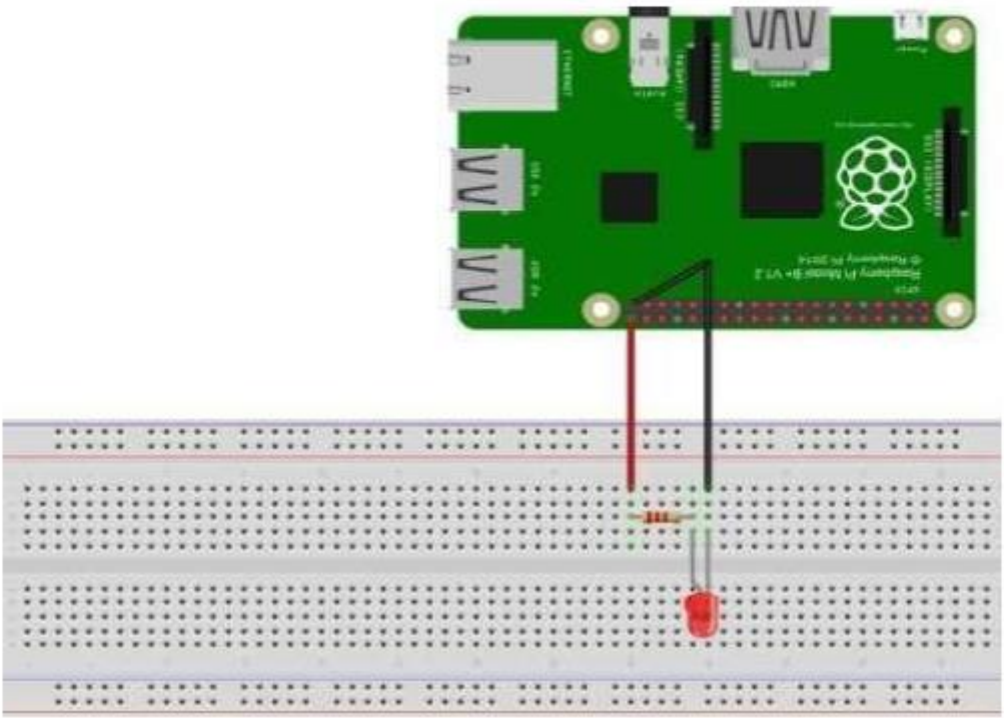
Aim: Blinking of LED Using Rasbperry Pi.

Components Required:

Here we are using Raspberry Pi 3 Model B. All the basic Hardware and Software requirements are previously discussed, you can look it up in the Raspberry Pi Introduction, other than that we need:

1. 1 - Raspberry Pi device with a 5 V power supply
2. 1 - Bread Board
3. 1 - Male to Female jumper cable
4. 1 – LED

Raspberry Pi LED Blinking with Python Programming:



As shown in the circuit diagram we are going to connect an LED between PIN40 (GPIO21) and PIN39 (GROUND). As said earlier, we cannot draw more than 15mA from any one of these pins, so to limit the current we are connecting a 220Ω or 1KΩ resistor in series with the LED.

Connection Diagram:

Connection Procedure:

1. Firstly, connect the GPIO pin ‘8’ to the bread board through jumper cable.

2. Now take LED and put it on the bread board. Connect the plus terminal of LED to the Jumper cable coming from the GPIO Pin “8”.

3. Now take another jumper cable, connect it to the GPIO pin “**Ground**” and another terminal of this wire to minus terminal of the LED.

4. Now write the python program for blinking LED every 1 second.

5. Now run the program and observe output in Blinking of LED.

Python Code:

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOAR
D)
GPIO.setup(8, GPIO.OUT)
while True:
time.sleep(1)
GPIO.output(8, False)
 time.sleep(1)
GPIO.output(8, True)
```

Result:

The LED turns ON when the push button is pressed and turns OFF when the button is released.

Viva Questions:

- What is GPIO?

- Why are resistors used with LEDs?
- What is the use of pull-up resistors?
- Difference between BOARD and BCM pin numbering?
- Why Python is preferred for Raspberry Pi?

Experiment 13

Interfacing DHT11-Temperature and Humidity Sensor with Raspberry Pi

Aim: To interface a DHT11 temperature and humidity sensor with Raspberry Pi and display the sensor readings using Python.

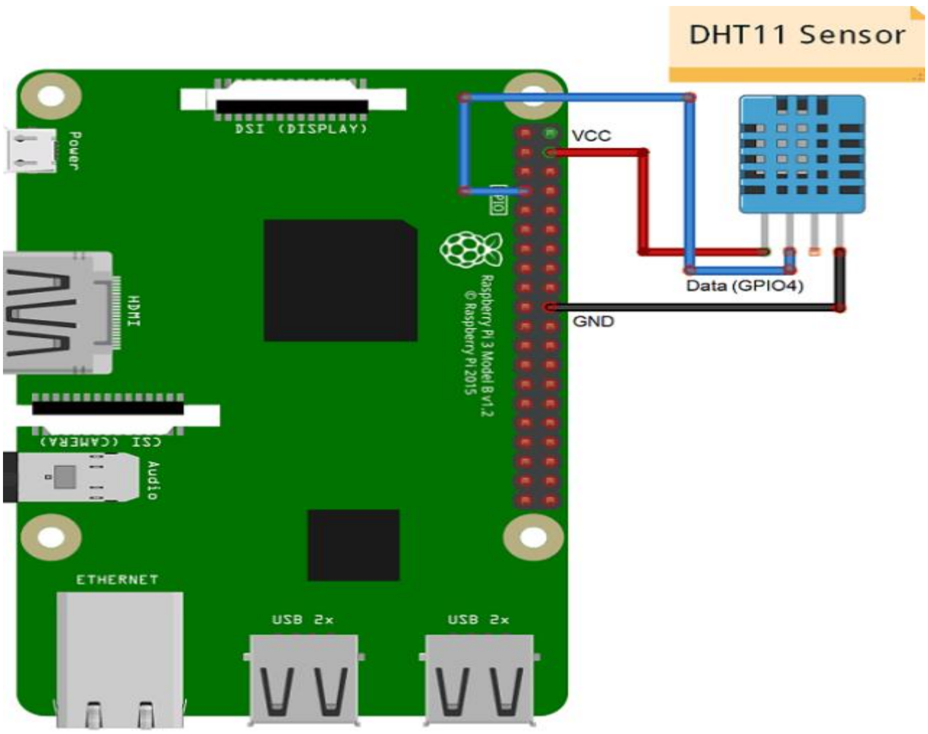
Components Required

- Raspberry Pi
- DHT11 Sensor
- 10 kΩ Resistor
- Breadboard
- Connecting Wires

Theory

DHT11 is a digital sensor used to measure temperature and humidity. It sends calibrated digital output to the Raspberry Pi through a single data pin. Python libraries help in decoding the sensor data accurately.

Circuit Diagram:



Procedure

1. Connect VCC of DHT11 to 3.3V of Raspberry Pi.
2. Connect GND to ground.
3. Connect DATA pin to GPIO 4 with a 10 kΩ pull-up resistor.
4. Install the DHT library using terminal.
5. Write Python code to read sensor values.
6. Execute the program and observe readings.

Program

```
import Adafruit_DHT

sensor = Adafruit_DHT.DHT11
pin = 4

humidity, temperature = Adafruit_DHT.read(sensor, pin)

if humidity is not None and temperature is not None:
    print(f"Temperature = {temperature}°C")
    print(f"Humidity = {humidity}%")
else:
    print("Sensor failure")
```

Result

Temperature and humidity values are successfully displayed on the terminal.

Viva Voce Questions

1. What type of sensor is DHT11?
2. Difference between DHT11 and DHT22?
3. Why is a pull-up resistor used?
4. What is digital sensing?
5. Applications of temperature sensors?

Experiment 14

Design of Motion Sensor Alarm using PIR Sensor

Aim: To design a motion detection alarm system using PIR sensor and Raspberry Pi.

Components Required:

1. Raspberry Pi 3 (any model)

2. PIR Sensor

3. Buzzer

4. Breadboard

5. Connecting wires

Working of PIR sensor:

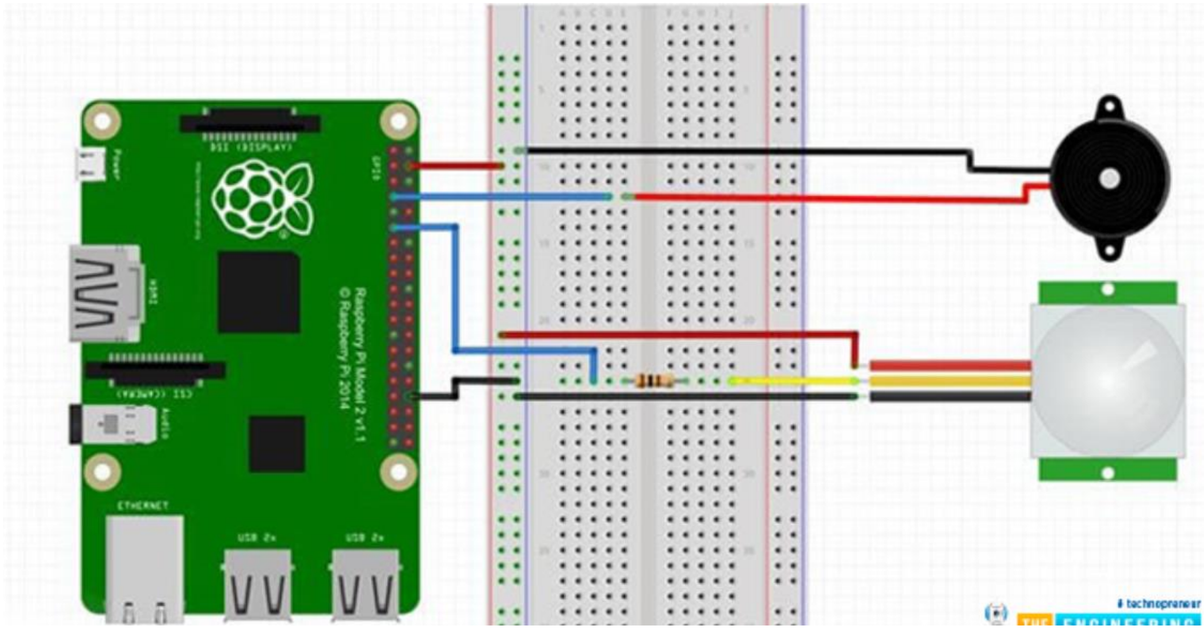
Passive Infrared (PIR) sensor is called passive because it receives infrared, not emits. Basically it detects any change in heat, and whenever it detects any change, its output PIN becomes HIGH. They are also referred as Pyroelectric or IR motion sensors. Here we should note that every object emits some amount of infrared when heated. Human also emits infrared because of body heat. **PIR sensors** can detect small amount of variation in infrared. Whenever an object passes through the sensor range, it produces infrared because of the friction between air and object, and get caught by PIR.

The main component of PIR sensor is **Pyroelectric sensor** shown in figure (rectangular crystal behind the plastic cap). Along with this, BISS0001 ("Micro Power PIR Motion Detector IC"), some resistors, capacitors and other components used to build PIR sensor. BISS0001 IC take the input from sensor and does processing to make the output pin HIGH or LOW accordingly.

Pyroelectric sensor divide in two halves, when there is no motion, both halves remain in same state, means both senses the same level of infrared. As soon as somebody enters in first half, the infrared level of one half becomes greater than other, and this causes PIRs to react and makes the output pin high.

Pyroelectric sensor is covered by a plastic cap, which has array of many Fresnel Lens inside. These lenses are curved in such a manner so that sensor can cover a wide range.

Circuit Diagram:



Procedure

1. Connect PIR sensor VCC to 5V and GND to ground.
2. Connect PIR output pin to GPIO 17.
3. Connect buzzer to GPIO 27.
4. Write Python code to monitor PIR output.
5. Run the program and observe alarm behavior.

Program

```
import RPi.GPIO as GPIO
import time


PIR = 17
BUZZER = 27
```

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(PIR, GPIO.IN)
GPIO.setup(BUZZER, GPIO.OUT)

try:
    while True:
        if GPIO.input(PIR):
            GPIO.output(BUZZER, GPIO.HIGH)
            print("Motion Detected!")
            time.sleep(1)
        else:
            GPIO.output(BUZZER, GPIO.LOW)
            time.sleep(0.5)

except KeyboardInterrupt:
    GPIO.cleanup()
```

Result

Whenever motion is detected, the buzzer alarm is activated successfully.

Viva Voce Questions

1. What is PIR sensor?
2. Why is it called passive?
3. Applications of motion sensors?
4. What is an alarm system?
5. Difference between active and passive sensors?