IN-PLANT TRAINING REPORT

ON

FARMMACHINERY &POWER

AT

भा.कृ.अनु.प. - केन्द्रीय कृषिरत महिला संस्थान, भुवनेश्वर

ICAR- CIWA

ICAR- Central Institute for Women in Agriculture

Bhubaneshwar, India

ISO 9001- 2008 Certified

भाकृअनुप
ICAR

01-07-2018 TO 30-09-2018

SUBMITTED BY

DIBYA RANJAN MOHANTA	150101170006
SHUBHAM PRADHAN	150101170057
RANJAN KUMAR SAHU	150101170059

GUIDED BY

Er. Mhatre Chaitrali Shashank

Scientist (Farm Machinery and Power)

Centurion
UNIVERSITY
Shaping Lives... Empowering Communities

4TH YEAR B.TECH (AG-ENGG)

SCHOOL OF ENGINEERING & TECHNOLOGY,

CENTURION UNIVERSITY OF TECHNOLOGY AND MANAGEMENT

PARALAKHEMUNDI, ODISHA

PREFACE

SUCCESS IS WALKING FROM FAILURE TO FAILURE WITH NO LOSS OF ENTHUSIASM

Man gains knowledge from practical experiences and this signifies, how a practical is differ from the general education. It is the training which makes the trainees to gain knowledge and have a deep penetration to the various aspects while working,

We had an excellent training on Farm machinery and Power in ICAR-CIWA, Bhubaneswar from 1st July to 30th September .This is only one institute devoted to gender related research in agriculture .

During this three month we gained a lot of practical knowledge about ergonomics, some of farm machinery implements and tools, post-harvest technology on fruits, occupational health hazard .

All the scientists & the technical staffs of this research institute helps us whole heartedly. Without their cooperation, hard work & encouragement, our training would have been incomplete. All the staffs and & employees of this institute were very cordial towards us.

This three month practical training was a real opportunity for a break through practical field indeed and it will be one of the memorable parts carriers as well as in our life.

ACKNOWLEDGEMENT

We the three trainees of SOET,CUTM,Paralakhemundi articulate our deep sense of gratitude to our honourable DEAN Dr. B. P. Mishra, the training-in-charge officer Er. Subodh Tanay Panigrahi and Dy. Register (placement) Mr. S. Kameswar Rao for arranging such an excellent research centre for our in-plant training.

We are highly indebted and thankful to the Director ICAR-CIWA for permitting us to acquire practical experience at this institute. We express our cordial thanks to Er. Chaitrali S. Mhatre, Scientist (FMP) for her encouragement, guidance, love, affection and also support during training period.

We are grateful to Dr Jyoti Nayak (Principle Scientist FRM), Dr. J. Charles Jeeva (Principal Scientist, Agricultural Extension), Dr. Ananta Sarkar (Senior Scientist, Agricultural Statistics), Mrs. Ankita Sahu (Scientist, Fruit Science) for their consistent cooperation and excellent teaching during the training.

We extend our sincere thanks to Er. Pragati Kishore Rout, Technical Assistant (FMP) for his support and for being there to share our joys and sorrows.

We are thankful to Er. Gayatri Mohanty, SRF scholars of our lab for her help and support when we needed it most.

We express our sincere thanks to various staff members of ICAR-CIWA, for their continuous cooperation to make our training a very enjoyable and unforgettable one.

Last but not the least, I express my gratitude and love to my family, for their constant encouragement, care and help.

Thanks a lot to GOD, whose help is always sought before my work.

Dibya Ranjan Mohanta Ranjan Kumar Sahu Shubham Pradhan

CERTIFICATE

This is to certify that DIBYA RANJAN MOHANTA, 150101170006 , (B. Tech in Agricultural Engineering), 4th year student of Centurion University of Technology and Management, Parlakhemundi has attended three month internship programme from 01.07.2018 to 30.09.2018 under my supervision and guidance. The In-plant training with project report entitled "**Development of microprocessor based data recording system for sensing relative humidity and temperature"** has been submitted to me. He has completed assigned work successfully during said period.

Chaitrali S Mhatre
Scientist, ICAR-CIWA

CONTENTS

INTRODUCTION

ICAR- Central institute for woman in agriculture (ICAR-CIWA) is an institution first of its kind in India that is exclusively devoted to gender related research in agriculture. It is established as National Research Centre for Women in Agriculture (NRCWA) in April 1996 at Bhubaneswar (Odisha) under Indian Council of Agricultural Research, New Delhi. The directorate has been upgraded and renamed as" ICAR- Central institute for woman in agriculture (ICAR-CIWA)" in the year 2014 under XIIth plan EFC

Mission:

Generate and disseminate knowledge to promote gender sensitive decision making for enhancing efficiency and effectiveness of woman in agriculture

Vision:

Emerge as a leading centre for gender research and serve as a catalyst for gender mainstreaming and women empowerment in agriculture to realize enhanced productivity and sustainability of agriculture.

Mandate:

- Research on gender issues in agriculture and allied fields.
- Gender-equitable agricultural policies/ programmes and gender-sensitive agricultural-sector responses.
- Co-ordinate research on Home Science.

ORGANISATIONAL STRUCTURE:

<u>Research Divisions</u>

1. Agricultural Economics
2. Agricultural Entomology
3. Agricultural Extension
4. Agricultural Statistics
5. Agronomy
6. Farm Machinery and Power
7. Fish Processing Technology
8. Home Science
9. Livestock Production and Management
10. Vegetable Science /Horticulture

All India Coordinated Research Project:

The AICRP on Home Science was conceived as an instrument to develop a strong base of research and extension the State Agricultural Universities for improving the quality of life of rural families. The project was initiated during the VI Five Year Plan Period. However, it is merged with DRWA in XI Five Year Plan Period. AICRP on Home Science integrates all the five components of Home Science in it namely Foods and Nutrition, Clothing and Textiles, Family Resource Management, Human Development & Family Studies and Home Science Extension Education. Each discipline has a specific thrust area of research that has been knitted together to focus on empowerment of women in agriculture for enhancing their quality of life. At present, the AICRP on Home Science is being implemented through its **ten centres** located in different State Agricultural Universities of the country viz.

Ongoing Research Projects

- Engendering Agricultural Research and Extension through Gender Friendly Technology Hub
- Developing gender sensitive model for Doubling Farmers' income by addressing gender concerns and technological gaps
- Strengthening gender knowledge system in Agriculture
- Seed Production of Food Crops in Tribal Regions with Participation of Women
- Optimizing technological interventions with gender perspective in small scale mango orchards
- Drudgery Reduction of Women Involved in Fish Processing through Technological Interventions
- Livelihood improvement of tribal farm women through secondary agriculture
- Design and development of disc type ridges for farmwomen
- Development and evaluation of integrated floating cage aqua geoponics system for small scale Women pond holders
- Performance evaluation of selected farm tools and implements in gender perspective for operational and ergonomic parameters
- Mapping livestock and gender and studying the role of institutions in livestock development in Eastern India
- Status of Women in Peri-urban Dairy Farming : Mainstreaming their Role for Enhancing Income and Productivity
- Gender inclusive homestead aquaculture for enhancing household fish consumption and income
- Promoting gender equity through family poultry production
- Enhancing income of rural women through improved goat rearing
- Improving availability of quality pulse seed with participation of women

- Exploratory study on nutritional status of Nabarangpur district, of Odisha (Inter Institutional Project with Regional Station CTCRI, Bhubaneswar)

Extension Education developed out of multidisciplinary approach. It intends to serve farming communities providing latest technologies to increase status of livelihood of the farm women and farmers. Extension in ICAR-CIWA reaches and teaches women target group to lead quality life keeping harmony with environment in which they reside and work. It takes care of capacity building, skill improvement creating self-confidence and enabling farm women to take right position in nation building process. ICAR-CIWA sincerely organizes various extension activities like Field Days, Exhibitions, Exposure visits, Demonstrations, TOT through Mass Media, etc. to acquaint the farm women with latest technologies. The Institute also conducts various capacity building programmes for gender sensitization among various stakeholders and to bring a change in knowledge, attitude and skill of farm women.

Gender friendly extension approaches and methodologies

- Village Level Para Extension Workers (VPEW) Model
- Public Private Partnership (PPP) Model

Participatory action research in Crop Production, Crop Protection, Horticulture, Animal Sciences, Fisheries

- Technology Demonstration
- Farm Women Training

ROLE OF WOMAN IN AGRICULTURE

- Woman, who has given birth to agriculture
- Represent 43% of global agricultural labour force
- In India, 65% of economically active women are in agriculture.
- Devote 45 - 50% of their time to agricultural activities
- Women spend 354 min/day and men 36 min/day on household activities
- About 12% of rural households are women-headed-with small holdings
- Alarming number of farmer suicides
- Male migration - About 40% of the men want to quit farming
- Gender role transformation in agriculture
- Struggling to attain Sustainable Development Goals of..
- No Poverty
- Zero Hunger
- Good Health & Wellbeing
- Climate Action
- Gender Equality

Issues of women agriculture laborers

- Dawn to dusk hard labour
- Deprivation of child
- Low wage rate
- Seasonal employment
- Insecurity at work place
- Addiction to tobacco and local liquor
- Travelling to distant places for farm work
- Worst sufferers of natural calamities
- Lack of exposure/training
- Nutrition deficiency

Development of women leadership in agriculture…

- Women need the support and help of a women leader who would organize them to be bold enough in facing socio-cultural restrictions, economic backwardness, the developmental agents, risks and complex technologies

- Often they need a woman leader who can read and write and keeps the accounts of the enterprises

- The training and extension for women and ATMA model have encouraged group activities among women, train them, and leave the group to function under a women leader

Extension strategies for addressing gender issues...

- Identify gender needs and interest
- Gender balanced extension system
- Mass media support
- Women friendly technologies
- Credit and technical support
- Capacity building of women
- Women farmer groups

ICAR-CIWA…in gender mainstreaming

- Farming system research
- Innovations in rural aquaculture
- Gender sensitive extension
- Technology assessment and refinement
- Occupational health risk and drudgery
- Family nutrition
- Women entrepreneurship
- Gender sensitization
- Gender sensitive methodologies and approaches
- Consultancy and advocacy

Occupational Health Hazards

Agriculture Ranks as one of the most Hazardous Industry till today due to

- Seasonal nature of agricultural activities
- Traditional methods of work
- Mechanization
- Increasing use of pesticides and agro-chemicals
- Use of non-ergonomic tools and equipment
- Lack of education and information on the health hazards

HEALTH PROBLEMS OF WOMEN AS SEEN FROM A GENDER PERSPECTIVE

- Basically hazards posed by physical, chemical and biological agents in work place
- Similar for male and female workers
- Women on an average
  - ✓ have a smaller stature and have less physical strength
  - ✓ their vital capacity is 11% less
  - ✓ their hemoglobin is app. 20% less
  - ✓ their skin area is larger as compared to circulating volume
  - ✓ they have larger body fat content
- They have lower heat tolerance and greater cold tolerance.
- Reproductive function

Types of Occupational Hazards

- Mechanical hazards
- Psycho-social hazards
- Work organisation hazards
- Ergonomic hazards
- Others – Physical, Biological, chemical, ergonomical and psycho-social hazards

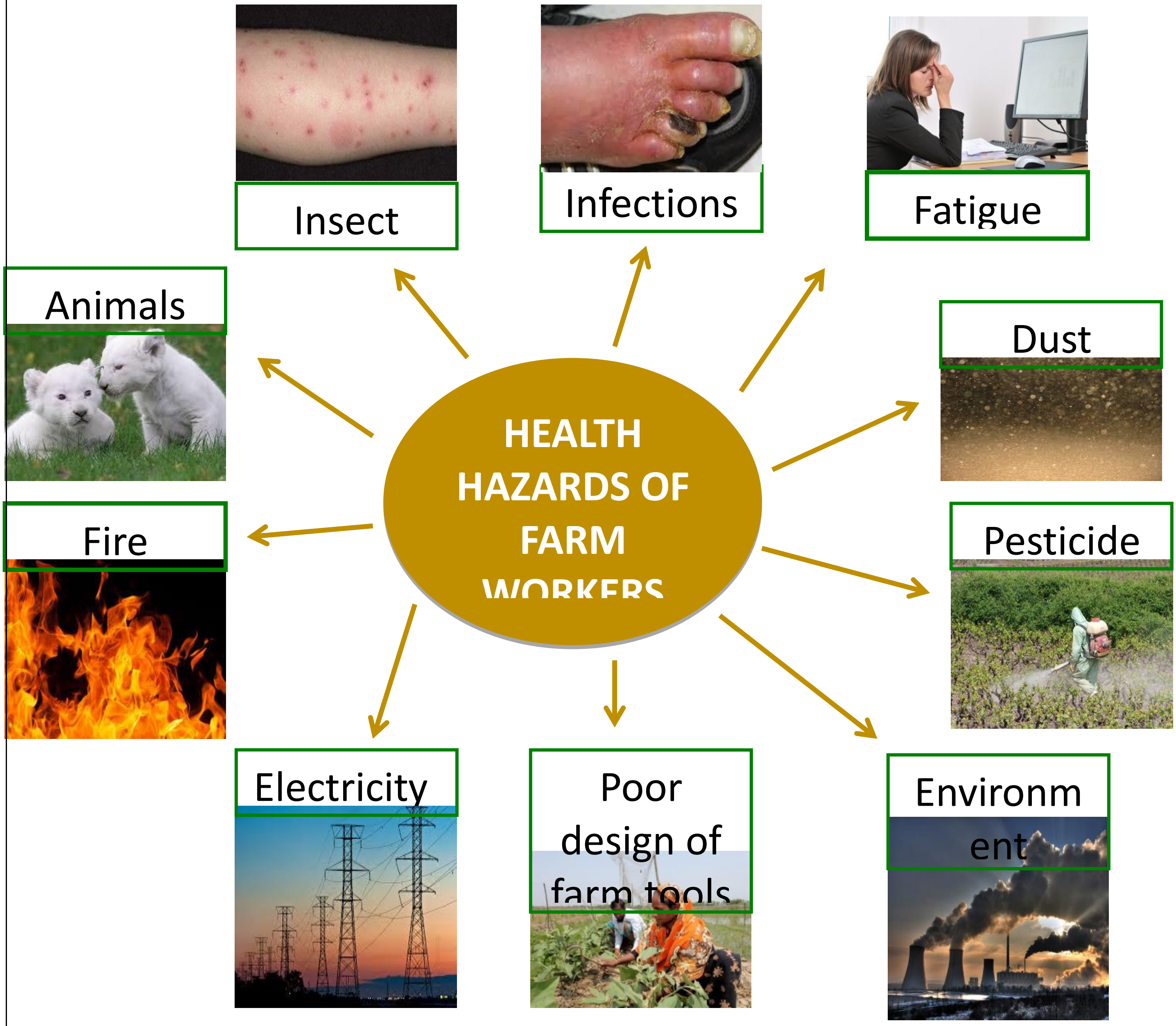


Fig.1-Health hazards of farm worker

Physical Hazards:

- Noise
- Vibration
- Extreme Temperature
- Illumination
- Radiation

Chemical Hazards:

- Renal Diseases
- Respiratory Diseases
- Skin Diseases
- Hematologic Diseases
- Cardiovascular Diseases
- Neurologic Diseases
- Carcinogenic
- Teratogenic

Other Harmful Effects of Noise:

- Hypertension
- Hyperacidity
- Palpitations
- Disturbs relaxation and sleep

Biological Hazards:

Agent / Disease	Occupation
Colds, influenza, scarlet fever, diphtheria, smallpox	**May be contracted anywhere**
Tuberculosis	**Silica workers, people exposed to heat and organic dusts, and medical personnel**
Anthrax	**Animal handlers and handlers of carcasses, skins, hides, or hair of infected animals, including wool carpet processors and handlers.**
Ringworm (in horses, cattle, deer, pigs, cats, dogs, birds)	**Pet shop salesmen, stockmen, breeders of cats and dogs, and other animal handlers**
Tetanus	**Farmers (spores in soil) or anyone in contact with manure.**

Hazard Control

- The first consideration for controlling hazards is to **eliminate** the hazard or **substitute** a less hazardous material or process.
- Engineering controls are physical changes to the work area or process that effectively minimize a worker's exposure to hazards
- If engineering controls are not feasible then consider implementing administrative controls.
- Examples of administrative controls include:

1. Limited time exposure to hazards
2. Written operating procedures

3. Safety and health rules for employees

- When Engineering/ Administrative controls are not sufficient to protect from hazard and during emergencies, PPE is applicable.
- Use of apron, goggles, mask, shoe, helmet/cap etc

An occupational health program allows you to respond effectively to workplace injuries and illnesses and to monitor potential health problems.

AGRICULTURAL STATISTICS

Statistics is a mathematical science pertaining to the collection, tabulation, analysis, interpretation or explanation, and presentation of data. It is applicable to a wide variety of academic disciplines, from the physical and social sciences to the humanities

Father of Statistics: Sir Ronald Aylmer Fisher (1890-1962)

- The elements are the entities on which data are collected.
- A variable is a characteristic of interest for the elements
- The set of measurements collected for a particular element is called an observation
- The total number of data values in a data set is the number of elements multiplied by the number of variables

Types of data:

- Qualitative Data
- Quantitative Data
- ✓ Cross-sectional data are collected at the same or approximately the same point in time.
- ✓ Time series data are collected over several time periods.

Mean

- The mean of a data set is the average of all the data values.
- The sample mean is the point estimator of the population mean *m*.

Median

- The median of a data set is the value in the middle when the data items are arranged in ascending order
- Whenever a data set has extreme values, the median
- It is the preferred measure of central location

POST-HARVEST TECHNOLOGIES IN FRUIT CROP

Maturity of Fruits and Vegetables

- It is the particular stage in life of plant of fruit at which they attain maximum growth and size
- There are five types of indices to judge the maturity of the fruit.
- 1. Visual means
  2. Physical means
  3. Chemical analysis
  4. Computation
  5.Physiological method

Controlled Atmospheric Storage

- In controlled atmospheric storage higher CO2 and lesser O2 are maintained
- The work on controlled atmosphere storage started in England in 1927 by kid and West. Modified atmosphere does not differ in principle from controlled atmosphere.
- In this the produce is held under the atmospheric condition by package, over wrap, box liner
- **Advantages of Controlled Atmosphere Storage:**
- i) Control all types of micro-organisms.
  ii) Chilling injury and other physiological disorders.
  iii) Black heart in potato
- **Hypobaric or Sub-atmospheric System**
- **Waxing**
- **Polymeric Film**
- **Chemicals**
- **Irradiation**
- Grading helps in obtaining uniform quality with respect to size, colour etc.
- It is done by hand or with the help of grading machines.
- Mechanical graders such as screen grader, roller graders, rope or cable graders are also used.
- Screen graders (made up of copper) are most commonly used.

- Soft and berry fruits are generally graded by hand picking
- Fruits like berries, plums, cherries and olives are graded whole while peaches, apricots, pears, mangoes are graded after cutting them into halves or slices.

DESIGN METHODOLOGY OF AGRICULTRAL MACHINERY

- Farm mechanization
- Forces i.e. direction& Quantity
- Stress induced at the lower area & It should be reinforced
- Standard Components
- Properties of material
- Working principles
- Optimization of shape &size
- Ergonomics consideration
- Flow of material
- Power transmission
- Behaviour of soil & crop material
- Knowledge of environment parameters

ERGONOMICS

The term ERGONOMICS is derived from the Greek words; ergo: work; nomos: Natural It is the science dealing with MAN-MACHINE-ENVIRONMENT relationship to get the optimum output from it with less human cost.

Ergonomics in agriculture

- Agriculture has a marginal primitive image.
- Transformation of traditional agriculture to Mechanized agriculture is need of time.
- Ergonomic application is tangible, i.e, pertaining to cost-benefit ratio, intangible, human health, comfort and safety.

Anthropometry

- 'Anthropometry', the study of body dimensions and strength.
- To understand the principles of anthropometry, one must be conversant with the anatomical sites and bony prominence of the human body.
- The problems of work place and workspace are the most common anthropometric problems; and these problems fall under man-machine physical compatibility, that someone is too large or too small to fit the machine.
- By obviating these problems, often better equipment can be designed and a work place organized.

Biomechanics

- The skeleto-muscular structures determine the range, strength and speed of human movements, including response behaviour to physical forces such as acceleration and vibration.
- These information grouped under 'Biomechanics' are useful in avoiding injuries on the job, in tool design, in work place and task layout, and in the protection of personnel against mechanical forces.
- The range, strength and speed of body movements are analysed by various biomechanical techniques and the psycho-physical methodologies.
- When a muscle strength problem is identified, the information on strength characteristics of different muscles may help to assess the severity of the problem.
- Accordingly, the alternative solutions are obtained.

Energy consumption

- The type of work decides the stress on the human body.
-
- Frequent analysis is made with reference to energy delivery and the strain on cardio-vascular and respiratory system due to muscular work. physical efforts and demand of muscles, environmental factor, static muscular contraction,
-
- The endurance to work is, thus based on the adaptability of the cardio-respiratory system with concurrent development of the skeleto-muscular structures.
-
- This information is used in establishing work organization principles like work/recovery cycles, shift work or standard for an allowable load of day's work.
-
- Designing tools and jobs to conform to a permissible level of energy demand is an approach widely accepted for various applications.

POSTURAL ANALYSIS TECHNIQUES

- RULA : Rapid Upper Limb Assessment
- REBA : Rapid Entire Body Assessment
- OWAS : Ovako Working Posture Assessment system

RULA : Rapid Upper Limb Assessment

- Rapid upper limb assessment (RULA) is a survey method for use in ergonomics investigations of workplace where work-related upper limb disorders are reported.

- This tool requires no special equipment in providing a quick assessment of the posture of the neck, trunk and upper limbs along with muscle function and the external loads experienced by the body.

- RULA was developed to:

1. Provide a method of screening a working population quickly, for exposure to a likely risk of work-related upper limb disorders.

2. Identify the muscular effort which is associated with working posture, exerting force and performing static or repetitive work, and which may contribute to muscle fatigue.

REBA (Rapid Entire Body Assessment)

- Rapid Entire Body Assessment (REBA) is a postural analysis tool. It has been developed to fill a perceived need for a practitioner's field tool, specifically designed to be sensitive to the type of unpredictable working postures found in health care and other service industries.

The development of REBA aimed to:

- Develop a postural analysis system sensitive to musculoskeletal risks in a variety of tasks.

- Divide the body into segments to be coded individually, with reference to movement planes.

- Provide a scoring system for muscle activity caused by static, dynamic, rapid changing or unstable postures.

- Reflect that coupling is important in the handling of loads but may not always be via the hands.

- Give an action level with an indication of urgency.

- Require minimal equipment - pen and paper method.

OWAS : Ovako Working Posture Assessment system

- Ovako working posture assessment system (OWAS) is used method for studying awkward working postures in workplaces. It identifies the most common work postures for the back (4 postures), arm (3 postures) and legs (7 postures) and the weight of the load handed (3 categories)

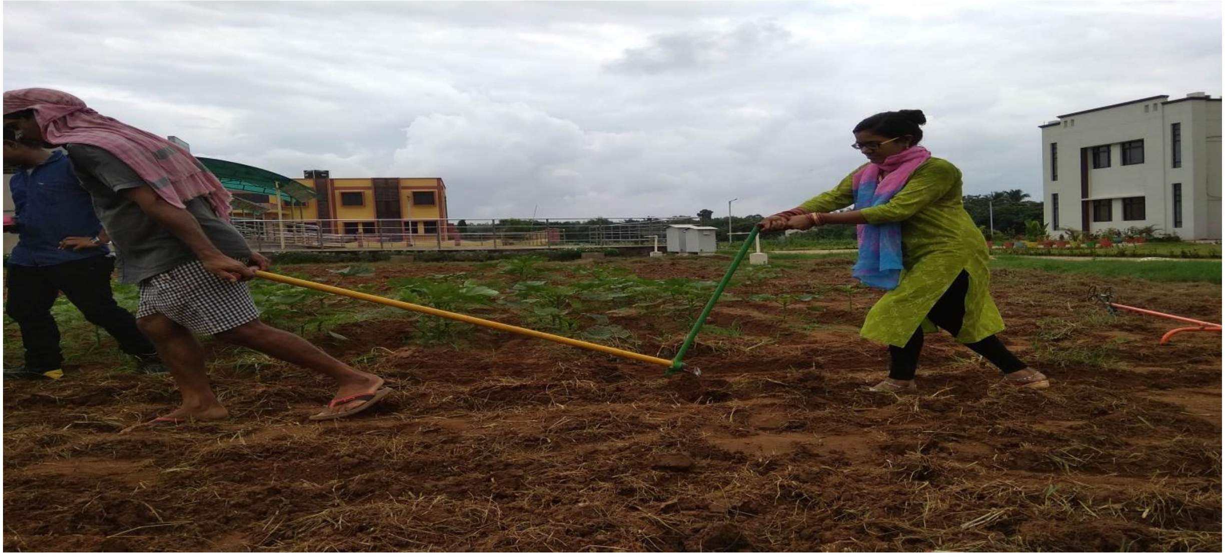


Fig.2-Ridger operation posture

RULA

Upper arm score	:	2+1=3
Lower arm score	:	1
Wrist score	:	2
Wrist twist score	:	2
Posture score (A)	:	**4**
Muscle use score	:	1
Force / load score	:	2
Wrist and arm score	*:*	***7***
Neck score	:	1
Trunk score	:	2
Leg score	:	2
Posture score (B)	:	**3**
Muscle use score	:	1
Force / load score	:	2
Neck, trunk and leg score	*:*	***6***
Final score	:	***7***
Inference	:	Investigate and implement change

REBA

Neck score	1
Trunk score	2
Leg score	2+1=3
Posture score (A)	**4**
Force / load score	
Score A	**5**
Upper arm score	2+1=3
Lower arm score	1
Wrist score	1
Posture score (B)	**3**
Coupling score	1
Score B	**4**
Table C	***5***
Activity	1
Final Score	***6***
Inference	*Medium Risk ,Further investigate &Change soon*

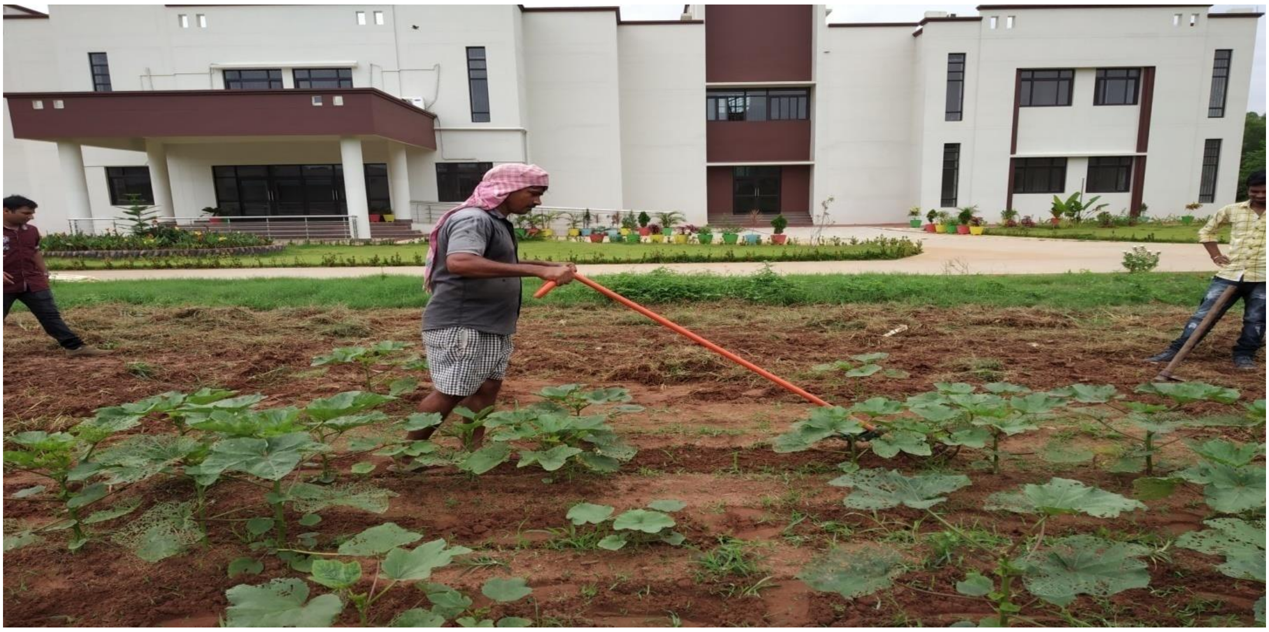


Fig.3-Wheel hoe operation posture

RULA

Upper arm score	:	1+1=2
Lower arm score	:	1
Wrist score	:	2+1=3
Wrist twist score	:	1
Posture score (A)	:	**3**
Muscle use score	:	1
Force / load score	:	2
Wrist and arm score	*:*	***6***
Neck score	:	2
Trunk score	:	2
Leg score	:	2
Posture score (B)	:	**3**
Muscle use score	:	1
Force / load score	:	2
Neck, trunk and leg score	*:*	***5***
Final score	:	***6***
Inference	:	Further Investigate and change soon

REBA

Neck score	:	1
Trunk score	:	2
Leg score	:	2+1=3
Posture score (A)	**:**	**4**
Force / load score	:	1
Score A	**:**	**5**
Upper arm score	:	1+1=2
Lower arm score	:	1
Wrist score	:	1+1=2
Posture score (B)	**:**	**2**
Coupling score	:	2
Score B	**:**	**4**
Table C	***:***	***6***
Activity	:	1
Final Score	***:***	***7***
Inference	***:***	*High Risk ,Further investigate &Change soon*

HUMAN ENERGY CONSUMPTION

- Human body is an engine.
- Food is metabolized to give energy to do mechanical work
- Energy expenditure of a person can be calculated by measuring the amount of oxygen intake.
- Calorific value of oxygen is 20.88 kJ/ l of O_2
- Maximal oxygen uptake capacity or VO_2 max is defined as the capacity of an individual to consume maximum amount of oxygen from the outer atmosphere.
- Women have 70 to 75 % of that of men.
- Indian agricultural workers :
- Female : 1.6 l/min
- Male : 2.2 l/mi
- Physiological cost of any operation is expressed in terms of heart rate and oxygen consumption rate.
- AWL : Acceptable work load, 35-40% of VO_2 max i.e. 0.6 & 0.8 l/min for female and male.
- Approximately the heart rate corresponding for this will be 110 to 120 bpm
- For most individuals HR of 120-130 bpm is 50 % of Vo_2 max. Here the person starts getting out of breath.
- Limit of continuous performance for 8-hour day (LCP) is suggested as 40-work pulse per minute
- In favourable conditions the work efficiency of the human body can be 25 – 30 %. But for field activities it ranges from 3 -25 %.
- Computerized ambulatory metabolic measurement system (K4b2)
- **Maximum heart rate = 190 - (Age in years – 25) x 0.62**

Estimation of oxygen consumption rate and energy consumption

- Male workers:
  - Y = 0.0183 HR – 1.28 *(Nag, 1981, agri workers)*
  - Y = 0.0156 Hr – 0.88 *(Tiwari et al , 2010, agri workers)*
  - Y = 0.014 HR – 0.8 *(ESA, agri workers)*
- Female workers :
  - Y = (0.159 HR -8.72) / 20.9 *(Verghese et al, 1994, college students)*
  - Y = 0.0114 HR – 0.68 *(Singh et al 2008, farm women)*
  - Y = 0.011HR – 0.59 *(ESA, agri workers)*

ANTHROPOMETRIC DATA FOR 79 BODY DIMENSION OF CENTURION TRAINEES

Dimension	Ardhendu	Dibya	Mohan	Ranjan	Shubham
Weight ,kg	72	56	70	66	68
stature	1650	1690	1630	1710	1720
Vertical reach	2030	1860	2120	2230	2220
Vertical grip reach	1922	1800	1980	2120	2110
Eye height	1500	1580	1530	1630	1620
Acromial height	1335	1430	1360	1410	1420
Elbow height	1050	1000	1030	1120	880
Olecranon height	1032	1010	980	1050	1040
Iliocrystale height	930	920	920	1000	1030
Iliospinal height	900	895	910	990	1010
Trachenteric height	785	850	800	940	950
Metacarpa height	720	750	690	740	730
Knee height	470	480	450	520	510
Medial Mallealus height	60	85	70	65	80
Lateria Mallealus height	55	90	80	80	90
Mentone to top of the head	200	200	210	220	230
Waist back	480	520	520	540	550

length					
Elbow rest height	230	270	230	240	250
Span	1620	1680	1710	1770	1850
Span akimbo	870	860	900	900	940
Arm reach from the wall	750	850	790	850	810
Thumb tip reach	670	730	710	770	750
Shoulder grip length	630	680	690	750	730
Wall to acromion distance	110	110	125	117	95
Wall to lumbosacral joint distance	45	50	35	60	50
Abdominal extension to wall	255	220	260	230	235
Chest depth	210	220	270	215	215
Biacromial breadth	280	320	300	330	310
Bideltoid breadth	440	410	430	460	440
Chest breadth	300	280	300	270	260
Inter scye breadth	310	320	340	325	310
Waist breadth	320	290	300	280	310
Hips breadth	340	300	380	310	360
Heel breadth	50	70	75	64	69
Bimalleolar breadth	70	75	70	70	72

Chest circumference	920	840	940	900	900
Waist circumference	300	290	320	280	310
Thigh circumference	500	430	520	525	450
Calf circumference	370	310	360	340	350
Wrist circumference	160	150	160	165	170
Grip diameter (inside)	40	45	40	50	45
Grip diameter (outside)	80	80	75	85	80
Middle finger-palm grip diameter	30	30	30	40	35
Vertical grip sitting	1060	1380	1340	1280	1250
Setting height	870	910	880	885	920
Sitting eye height	710	800	760	770	795
Sitting acromion height	560	630	600	620	610
Elbow rest height	230	270	230	240	250
Thigh clearance height sitting	160	130	160	130	130
Knee height sitting	520	510	520	550	550
Popliteal height sitting	440	460	440	500	480
Hand thickness at metacarpal	33	29	28	32	30

–III					
First phalanx digit III length	63	67	62	66	67
Grip length	65	60	57	65	58
Maximum grip length	110	118	117	113	125
Index finger diameter	17	18	19	21	19
Head length	180	185	180	190	195
Head breath	150	125	130	120	130
Coronoid fossa to head length	355	420	350	415	457
Fore arm head length	415	457	450	470	495
Elbow grip length	320	340	350	360	395
Hand length	180	185	175	185	190
Palm length	95	105	90	100	98
Hand breadth across Thumb	96	95	100	113	102
Hand breadth	82	87	85	90	85
Buttock knee length	565	550	530	560	550
Buttock popliteal length	445	455	470	465	460
Abdominal depth sitting	240	170	240	210	210
Hip breadth sitting	350	340	390	340	350
Elbow Elbow breadth sitting	385	330	450	360	350
Knee Knee breadth	235	190	220	180	185

Foot length	238	245	250	255	255
Instep length	193	190	200	200	200
Foot breadth	97	105	95	110	110
Functional leg length	938	960	950	1250	1300
Bicep skinfold thickness	15	4	8	5	9
Triceps skin fold Thickness	8	5	10	6	5
Sub scapular skin fold Thick ness	13	12	16	17	13
Supra Iliac skinfold Thickness	26	8	33	10	20

PROJECT

DEVELOPMENT OF MICROPROCESSOR BASED DATA RECORDING SYSTEM FOR SENSING RELATIVE HUMIDITY AND TEMPERATURE

ABBREVIATIONS

IDE	Integrated Development Environment
Ω	Ohm
VCC	Voltage common collector
GND	Ground Voltage Level.
MISO	Master In Slave Out
MOSI	Master Out Slave In
SCK	Serial clock
CS	Chip select
SCL	Clock line
SDA	Data line
°c	Celsius
$°F$	Fahrenheit,
H_0	the null hypothesis
μ_1	the mean of 1
μ_2	the mean of 2.

INTRODUCTION

- Temperature
- Humidity

TEMPERATURE:-

Temperature is a physical quantity expressing hot and cold .Ambient temperature is the air temperature of an environment or object. In computing, ambient temperature refers to the air temperature surrounding computing equipment. This measurement is crucial for equipment function and longevity. It is a proportional measure of the average kinetic energy of the random motions of the constituent particles of matter (such as atoms and molecules) in a system. Temperature is important in all fields of natural science, including physics, chemistry, Earth science, medicine, and biology, as well as most aspects of daily life.[2]

Temperature is an important factor in agriculture variations in temperature affect the growth of the crop.it also influences the flowering, fruiting, timing, which will ultimately effect the yield.

OLD METHOD :-(Measurement of temperature)

Manual Field Temperature Measurement Procedure:-

General Field temperature measurements may be made with a field thermometer, equipment thermistor, or NIST-traceable thermometer. At a minimum, the temperature measurement device should be capable of measuring in 0.1°C increments. Thermometers are the oldest of the group. [1]

Instrument:-

Field thermometers and thermistors:-

Temperature measurement devices such as field thermometers and equipment thermistors will be verified against a NIST-traceable thermometer prior to use and should agree within ± 4.0C. Corrections may be applied for measurements up to°should agree within ± 4.0 C depending on investigation objectives, but the instrument must be°± 4.0 repaired or replaced beyond that range. [1]

NIST-traceable thermometer:-

Verification of the NIST-traceable thermometers that are used to verify temperature measuring devices is accomplished by comparing temperature readings from the NIST-traceable thermometer to a thermometer that has an independent certification of accuracy traceable to the National Institute of Standards and Testing. Current certified thermometers are maintained by the SESD Analytical Support Branch and are called reference thermometers.

Each NIST-traceable thermometer is verified by comparing at least annually against a reference thermometer. If corrections need to be applied, they will be noted in the NIST-traceable thermometer. Depending on investigation objectives, project leaders may decide to apply the correction factor as necessary. [1]

Sample measurement procedures for thermometers/thermistors:-

(Make measurements in-situ when possible)

1. Clean the probe end with de-ionized water and immerse into sample.

2. If not measuring in-situ, swirl the instrument in the sample for mixing and equilibration.

3. Allow the instrument to equilibrate with the sample for at least one minute.

4. Suspend the instrument away from the sides and bottom, if not in-situ, to observe the temperature reading.

5. Record the reading in the log book. For most applications, report temperature C depending on need.°C or to the nearest 0.1°readings to the nearest 0.5

Note: Always clean the thermometer with de-ionized water or a detergent solution, if appropriate, prior to storage and/or use. [1]

Units:-

Degrees Celsius (°C) or Degrees Fahrenheit (°F)

Conversion Formulas: -

°F = (9/5 °C) + 32 or °C = 5/9 (°F - 32)

HUMIDITY:-

Humidity is a term used to describe the amount of water vapour present in air. Water vapour, the gaseous state of water, is generally invisible to the human eye. Humidity indicates the likelihood for precipitation, dew, or fog to be present. As the temperature of the air increases more water vapour can be held since the movement of molecules at higher temperatures prevents condensation from occurring.

Humidity is important to make photosynthesis possible. If the humidity is too low, plant growth is often compromised as crops take much longer to obtain the saleable size. Also lower leaves often drop off, growth is hard, and overall quality is not very good. Whether the humidity is too high or too low, the loss of quality reduces the selling price of crops and increases production costs, both of which reduce profits.

- Three primary measurements of humidity are widely employed: absolute, relative and specific. Absolute humidity describes the water content of air and is expressed in either grams per cubic meter or grams per kilogram. Relative humidity, expressed as a percentile, indicates a present state of absolute humidity relative to a maximum humidity given the same temperature. Specific humidity refers to the weight of water vapour contained in a unit weight (amount) of air (expressed as grams of water vapour per kilogram of air). Absolute and specific humidity are quite similar in concept.

Relative Humidity:-

- The relative humidity of an air-water mixture is defined as the ratio of the partial pressure of water vapour in the mixture to the equilibrium vapour pressure of water over a flat surface of pure water at a given temperature .
- Relative humidity is normally expressed as a percentage; a higher percentage means that the air-water mixture is more humid .Relative humidity is an important metric used in weather forecasts and reports, as it is an indicator of the likelihood of precipitation, dew, or fog. In hot summer weather, a rise in relative humidity increases the apparent temperature to humans (and other animals) by hindering the evaporation of perspiration from the skin. For example, according to the Heat Index, a relative humidity of 75% at air temperature of 80.0 °F (26.7 °C) would feel like 83.6 °F ±1.3 °F (28.7 °C ±0.7 °C). [3]

OLD METHOD :-(Measurement of humidity)

Psychometric method:-

The oldest method for measuring relative humidity is the psychometric method.

Psychometry is commonly known as the “wet” and “dry” bulb method. A psychometric sensor does not directly sense humidity, but rather it senses temperature to indirectly find relative humidity. The sensing elements can be thermometers, RTD’s, or thermistors. The first sensing element, the dry bulb, measures ambient temperature. The second sensing element, the wet bulb, is enclosed in a wick saturated with distilled water. Air forced across the wet bulb creates evaporation, which cools it below ambient temperature. The amount of evaporation (cooling) is dependent on the vapour pressure of the air. Using the wet and dry-bulb temperatures, the relative humidity can be looked up on a psychometric chart. Looking up the %RH on a chart for every measurement is time-consuming and cumbersome. With today’s technology, psychometric charts and dew point equations can be stored in a microprocessor, thus making this a direct sensing method for RH and dew point. [6]

Dew point method:-

In this method, the dew point is determined by cooling a highly polished surface in the gas and checking the maximum temperature at which the condensation takes place. The humidity of the gas is equal to the humidity of saturated gas at the dew point. [6]

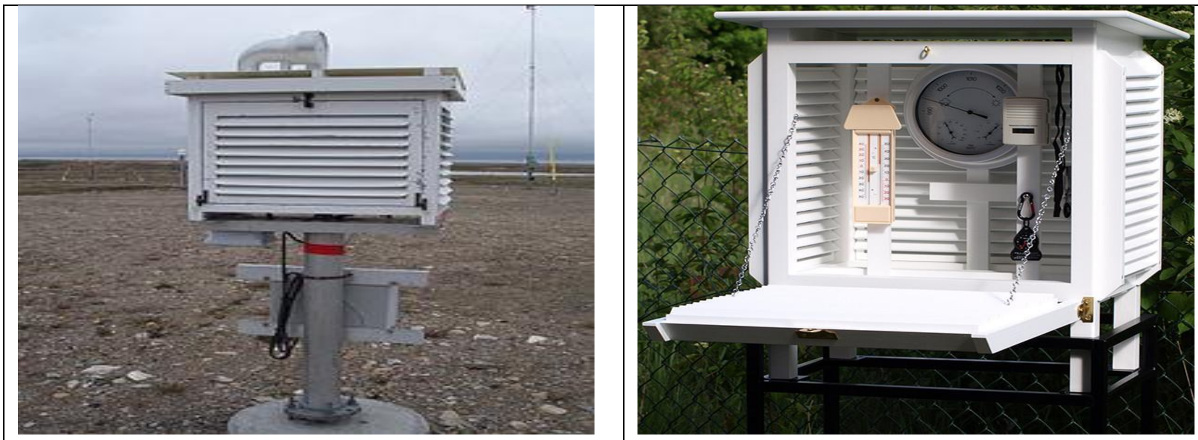
Hygrometric method:-

The hygrometric method of relative humidity sensing is the most common. The instruments are generally compact, reliable, and inexpensive .These sensors provide an output that is directly indicative of humidity. The first humidity sensing elements were mechanical in nature. Physical dimensions of various materials will change with theadsorption1 of water. Some examples of these are hair, animal membrane, and some plastics. To build a sensor from these materials the element is kept in tension with a spring. A strain-gauge monitors the displacement caused by a change in the moisture content of the air. The output of the strain gage is directly proportional to the relative humidity. A second method of hygrometry is coating an oscillating crystal (quartz) with a hygroscopic coating. When the coating adsorbs water the mass changes which then changes the crystal’s oscillating frequency. A more obscure method is an electrolytic hygrometer. This method is complicated and not used frequently enough to warrant explanation. [6]

Electrical Hygrometers:-

Electrical impedance sensors measure the changes in electrical capacitance or resistance of a hygroscopic material. The material will absorb or desorb water depending on the partial vapour pressure in the atmosphere around it, thus changing its electrical properties. These sensors measure relative humidity. While capacitive hygrometers can withstand condensation, resistive ones usually cannot. [6]

STEVENSON SCREEN:-



(A) (B)

Fig 4. (A) Outer view of Stevenson screen ,(B)inner view of Stevenson screen

A Stevenson screen or instrument shelter is a shelter or an enclosure to shield meteorological instruments against precipitation and direct heat radiation from outside sources, while still allowing air to circulate freely around them.

It forms part of a standard weather station. The Stevenson screen holds instruments that may include thermometers (ordinary, maximum/minimum), a hygrometer, a psychrometer, a dew cell, a barometer and a thermograph.

Stevenson screens may also be known as a cotton region shelter, an instrument shelter, a thermometer shelter, a thermo screen or a thermometer screen. Its purpose is to provide a standardised environment in which to measure temperature, humidity, dew point and atmospheric pressure. [3].

NOTE:-Humidity and temperature are common parameters to measure environmental conditions, but it is difficult to measure them in real time manually due to various operational limitations. Manual recording of the data may lead to erroneous results.

Hence it is proposed to develop a Microprocessor based data recording system which can sense and record the data of humidity and temperature in real time.

AIM:-

The aim of the project is to design and develop a microprocessor based temperature and humidity measuring system that will record the values with respect to time.

OBJECTIVES:-

1. To study the existing microprocessors and their coding systems.
2. To study different types of sensors and their mechanisms.
3. To design circuit and develop compatible code for sensing temperature and humidity with "DHT22" sensor.
4. To enable the sensor to read and record the data in real time using " real time clock - DS3231" and " micro SD module”

MICRO-PROCESSOR

A microprocessor is an integrated circuit (IC) which incorporates core functions of a computer’s central processing unit (CPU). It is a programmable multipurpose silicon chip, clock driven, register based, accepts binary data as input and provides output after processing it as per the instructions stored in the memory.

ADVANTAGES OF A MICRO-PROCESSOR:-

- **Low Cost**
  Microprocessors are available at low cost due to integrated circuit technology .Which will reduce the cost of a computer system.
- **High Speed**
  Microprocessor chips can work at very high speed due to the technology involved in it. It is capable of executing millions of instructions per second.
- **Small Size**
  Due to very large scale and ultra large scale integration technology, a

microprocessor is fabricated in a very less footprint. This will reduce the size of the entire computer system.

- **Versatile**
  Microprocessors are very versatile, the same chip can be used for a number of applications by simply changing the program (instructions stored in the memory).
- **Low Power Consumption**
  Microprocessors are usually manufactured using metal oxide semiconductor technology, in which MOSFETs (Metal Oxide Semiconductor Field Effect Transistors) are working in saturation and cut off modes. So the power consumption is very low compared to others.
- **Less Heat Generation**
  Compared to vacuum tube devices, semiconductor devices won't emit that much heat.
- **Reliable**
  Microprocessors are very reliable, failure rate is very less as semiconductor technology is used.
- **Portable**
  Devices or computer system made with microprocessors can be made portable due to the small size and low power consumption.

ELECTRONICS USED TO DEVELOPED THE SENSOR ARE AS FOLLOWS

SL. NO.	NAME	**QUANTITY**
01	Arduino.	1
02	Humidity & Temperature sensor (DHT22)	1
03	Real time clock (DS3231)	1
04	Resistor.	2(10K,330Ω)
05	Micro SD module	1
06	Jump Wire	As per requirement
07	Power source	(9-12)volts

In this project we have used one **Arduino** (Arduino Uno) based sensor which can give both humidity and temperature parameters value at a time and can be stored in the memory card.

ARDUINO

Arduino is a microcontroller board which functions as a tiny computer; it is a platform where creation and development of interacting objects is possible with required programming software. The Arduino software IDE (Integrated Development Environment) provides space to write codes in the language (programming languages C, C++) that Arduino board understands and responds to. Inexpensiveness, easy-to-use design and flexibility for advance modifications are some features of the microcontroller based Arduino hardware and software that are making its range of use wider. One of the most important factor that affects its increasing range of use is its freedom of use. Both the Arduino hardware and the software are open source, which means that one can easily use the ideas generated by others in their work and modify them without anyone's authorization. It can be used by anyone to do anything they want to do with it .Arduino boards are designed in such a way that one without prior knowledge of electronics or previous experience of programming can use information from other people's work and build their own interactive object that can sense the environment and control it. It comes with a cheap price which is a crucial factor that makes Arduino accessible to many students, hobbyists and teachers and ultimately a new revolution of innovation in electronics.

Arduino is the brain of circuit. It has input and output ports .Input is taken through sensors, Arduino processes inputs and gives outputs. Typical Arduino Uno is depicted in figure no.5.

- Circuit:- In the Arduino world circuit is made using an Arduino with other components and devices.
  A typical Arduino board is consisting of the following part –
- Power pins:-These pins give power to the circuit.
- Analog input:-6 Dedicated pins for taking Analog input, through Analog to digital conversation (ADC).
- Digital input Output pins:-13 digital pins, these can be used as input or output. It is needed to be declared in the code whether pin is to be used as input or output. Thus a digital input/output pin can be used as an input pin or an output pin.
- PWM Pins:-In an Arduino circuit Analog output is taken through the PWM pins.
- RX & TX Pins:- These two pins are for serial communication .Arduino is connected to an external devices through these two pins .These pins provide serial communication to the Arduino. Rx stands for receiver and Tx stands for transmitter .To connect an Arduino system to another device the Rx of Arduino should be connected to the TX of the other device and vice versa.
- Reset pins:-This pin is to reset Arduino externally.
- Reset Button:-This button is for resisting Arduino .The Arduino gets reset if this button is pressed.

- Power inputs:-It is advised to use a power supply of 9-12v with an Arduino. This input can be taken out for a circuit from pin of the power pins.
- GND:-It stands for the Ground Voltage Level.
- Breadboard:- Breadboard is a device to make prototype circuits .It has several sockets in which electronic components get inserted. It has power line sockets that provide power to the circuit. .[8]

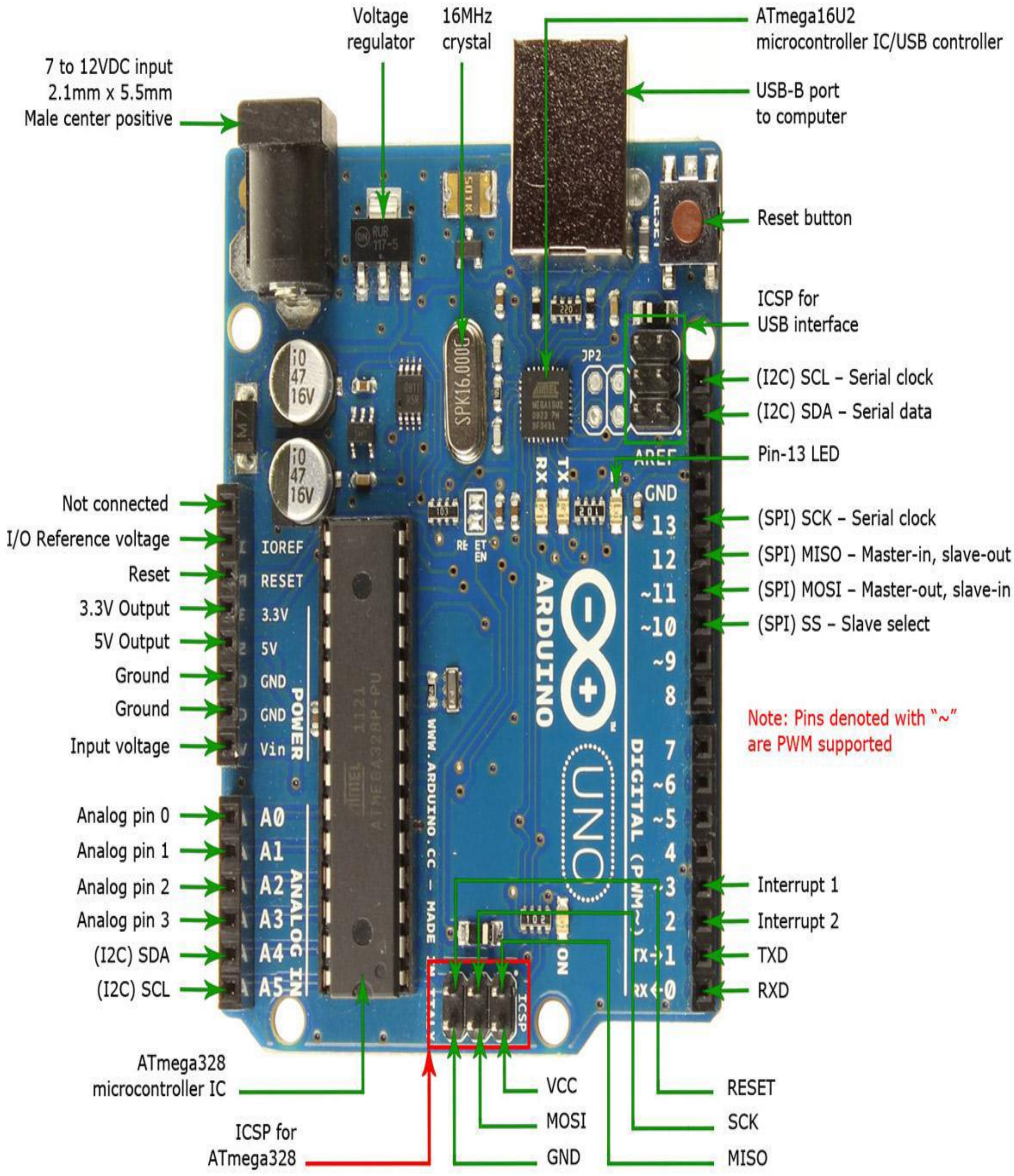


Fig.5-Pin connection of Arduino UNO

HUMIDITY AND TEMPERATURE SENSOR

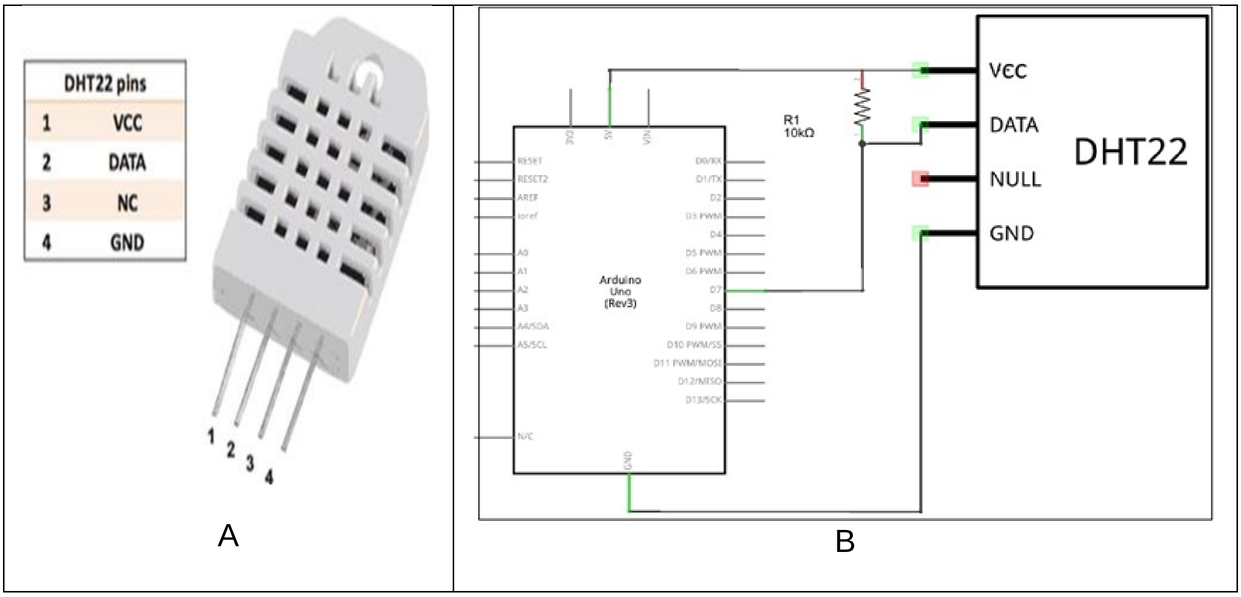


Fig 6. (A)DHT22 sensor, (B) circuit diagram of DHT22 sensor.

Table.1- Pin names of DHT22 sensor with their connection.

PIN	CONNECTION
1 VCC	5V.
2 DATA	One is Connected to 5volt through a 10kΩ resistor and another to pin 2 of Arduino.
3 NC	Null
4 GND	Ground Pin.

A sensor is an electronic device that converts a change in physical phenomenon into an electrical signal. It can send the information to computers or other electronic devices.

The sensors used in this work is temperature and humidity sensor-DHT22 as shown in figure 6.The sensor-DHT22 is an Analog sensor designed to sense the physical change in heat and moisture when exposed in air with suitable wiring and programming.

Its small size, cheap price, low power consumption, quick responses are the characteristics for being one of the best choices for many users. The sensor DHT22 is applicable in HVAC (heating, ventilation and air conditioning), it can be used in testing and inspecting equipment and consumer goods. It is also applicable to use in building a weather station or a humidity regulator. The use of DHT22 sensor has shown its usefulness measuring and controlling temperature and humidity in home appliances, medical and many other sector.

Table.2- Measurement range of DHT22 sensor

Sensors	DHT11	DHT22
Humidity:	20 to 90 % RH	0-100%
Temperature:	0 to 50°C	-40 - 125°C

Accuracy:

Temperature: ±2 %

Humidity: ±5 %

Where the operating Voltage remains between 3V to 5.5V.

We prefer DHT22 as compared to DHT11because it has high performance.

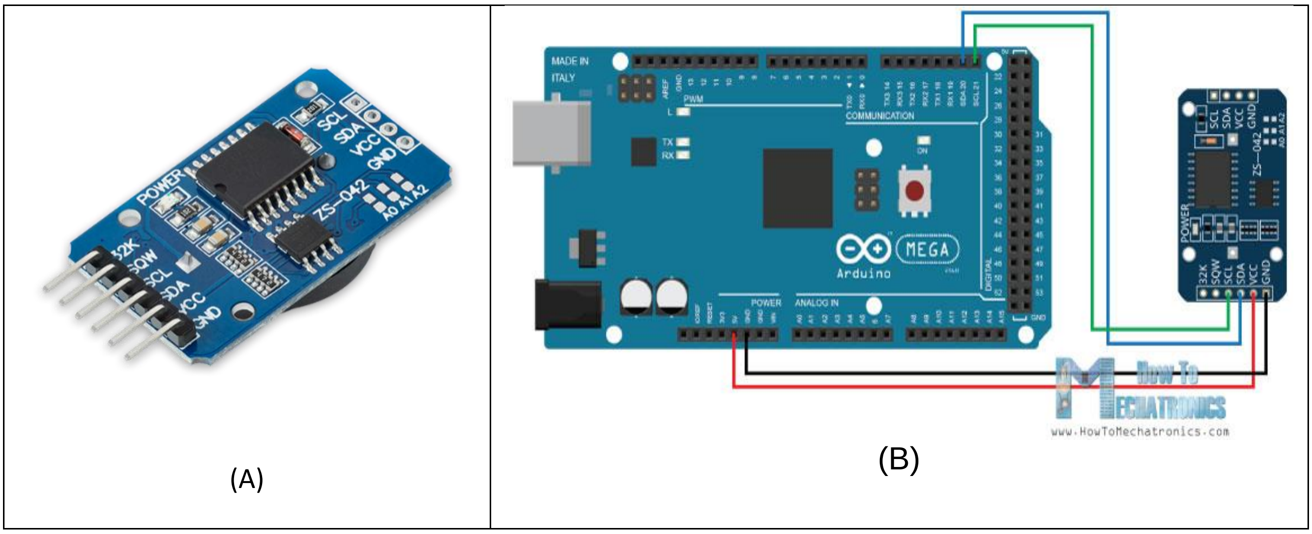
The sensor used in this project, DHT22, is designed to measure humidity in terms of relative humidity (RH). Relative humidity (RH) is the ratio of the amount of water vapour content of the air to the saturated moisture level at the same pressure or temperature.

$RH=\frac{\rho w}{\rho s} \times 100\%$

Where RH is relative humidity, ρw is the density of water vapour, and ρs is the density of water vapour at saturation.

Working:- The sensor DHT22 detects moisture in the air by measuring the electrical resistance between electrodes .It is fabricated with a moisture holding substrate. When substrate absorbs moisture, ionization takes place and results in the increase in conductivity between the electrodes .The relative humidity is proportional to the change in resistance between electrodes due to moisture absorbed.

REAL TIME CLOCK



(A)

(B)

Fig 7. (A) RTC-DS3231 (B) Circuit diagram of DS3231

Table.3- Pin names of RTC-DS3231 with their connection.

PIN	CONNECTIONS
1. 32K	none
2. SQW	none
3. SCL	A5
4. SDA	A4
5. VCC	3.3volt
6. GND	ground

The DS3231 is a low-cost, highly accurate Real Time Clock which can maintain hours, minutes and seconds, as well as, day, month and year information. Also, it has automatic compensation for leap-years and for months with fewer than 31 days.

Operating Voltage: 3.3 - 5V

Current: 15mA

Accuracy: ±2ppm at 0°C to +40°C and ±3.5ppm at -40°C to +85°C

Digital Temp Sensor Output: ±3°C Accuracy

MICRO SD MODULE

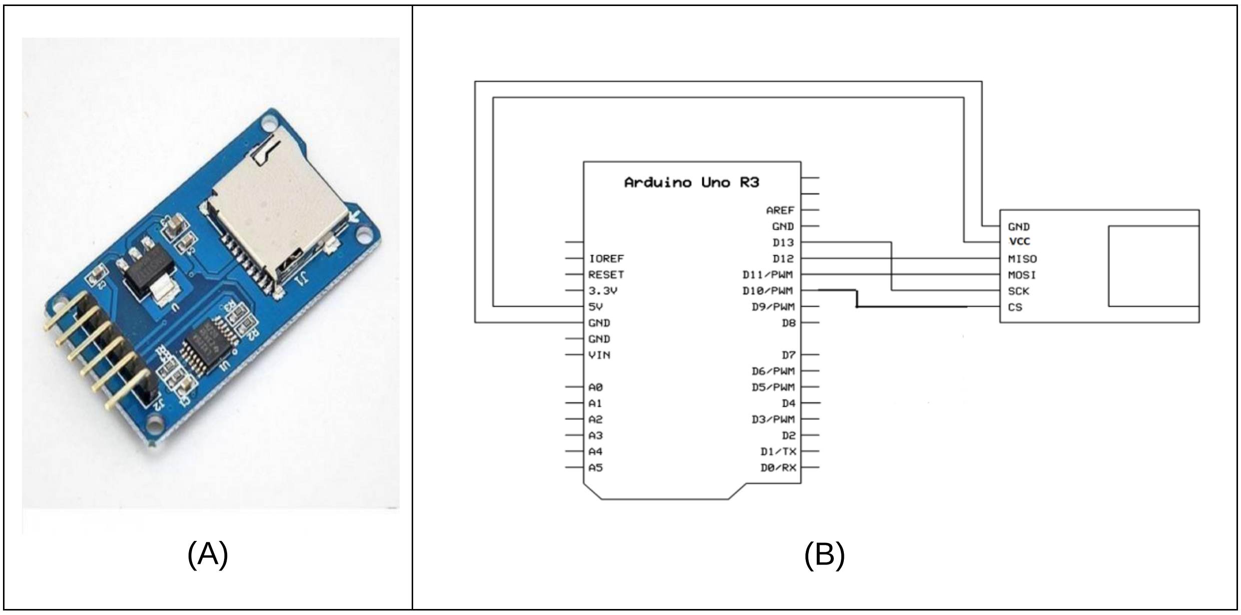


Fig 8. (A) Micro SD Module, (B) circuit diagram of micro SD module

Table.4 Pin names of micro SD module with their connection.

PIN	CONNECTION
1.GND	Ground pin
2.VCC	5volt
3.MISO	330Ω-D12
4.MOSI	D11
5.SCK	D13
6.CS	D10

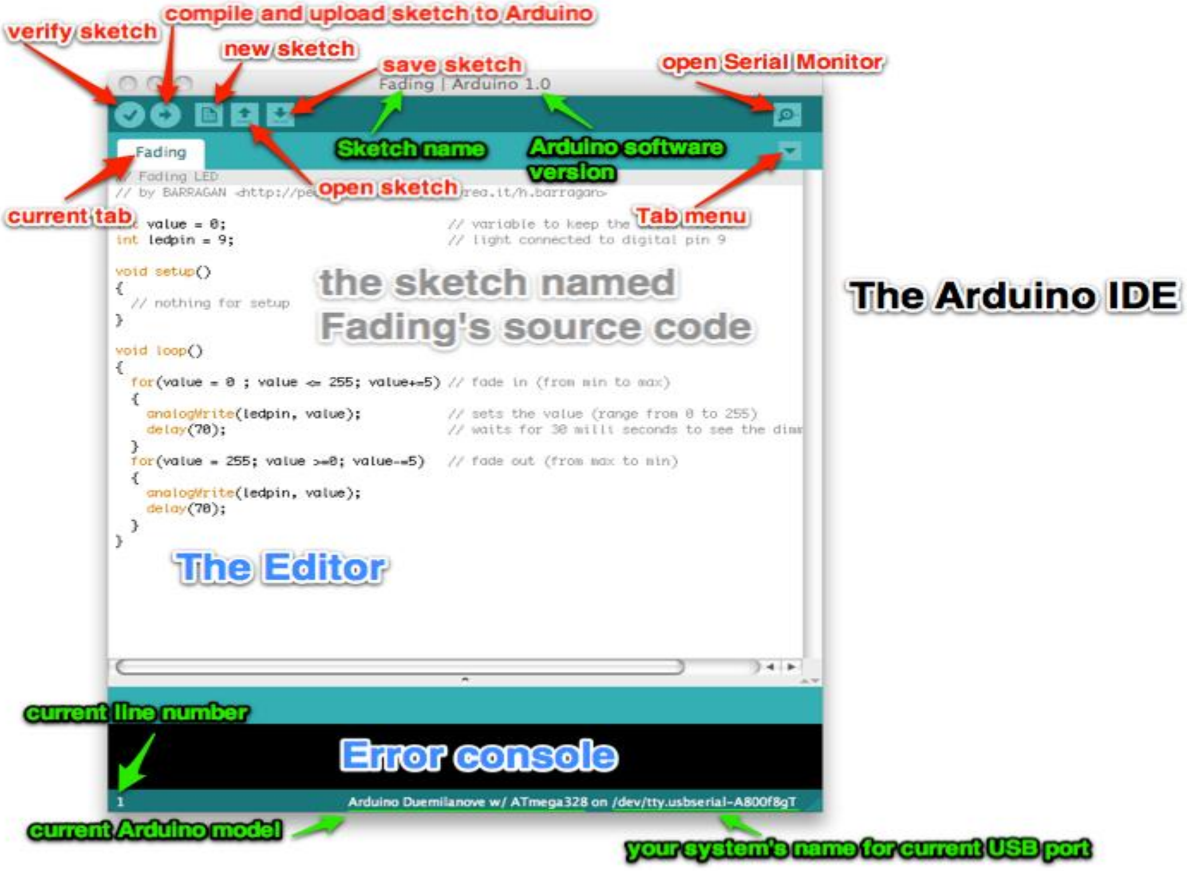
The Arduino Micro SD card Module is an SPI Communication based device. It is compatible with the TF SD cards used in mobile phones and can be used to provide some sort of external storage for micro controller and microprocessor based projects, to store different kind of data. SD cards generally are 3.3v logic level based devices, but with the aid of the Micro SD card module, the signals are converted to 5v via a logic level converter implemented on the SD card Module.

Arduino IDE

(IDE stands for Integrated Development Environment.)

The brain part of the building monitoring system, the Arduino IDE (integrated development environment), is a software development environment or software application for Arduino where users can write different kind of computer programs and test .The user can write codes in IDE in a language which an Arduino understands, i.e. C, C++. The program (codes) written in IDE, when uploaded into the Arduino microcontroller deter-mines what and how the system works. The Arduino IDE comes with a 'built-in code parser' that studies the validity of the written codes before sending it to the Arduino. The compilation and translation work is done in IDE after checking the validity of codes. After translating the code, the IDE uploads the program to the Arduino microcontroller.

The Arduino IDE looks like -



```
// Fading LED
// by BARRAGAN <http://pe                 rea.it/h.barragan>

int value = 0;                            // variable to keep the
int ledpin = 9;                           // light connected to digital pin 9

void setup()
{
  // nothing for setup
}

void loop()
{
  for(value = 0 ; value <= 255; value+=5) // fade in (from min to max)
  {
    analogWrite(ledpin, value);           // sets the value (range from 0 to 255)
    delay(70);                            // waits for 30 milli seconds to see the dimm
  }
  for(value = 255; value >=0; value-=5)   // fade out (from max to min)
  {
    analogWrite(ledpin, value);
    delay(70);
  }
}
```

Fig 9. Arduino IDE

VERIFY: - This compiles code and checks errors. If any error is found, error message is displayed in the message panel.

UPLOAD:-This transfers the compiled code to Arduino board. It first verifies, If no error is found the code is transferred to the connected Arduino board.

NEW:-This opens a new tab.

OPEN: - This opens an existing Arduino code.

SAVE:-This saves the opened code.

An Arduino code is known as sketch.

SET UP /CONNECTION AND CODE

Schematically, the connections for the memory storage and DHT11 sensor with Arduino Uno is shown below.

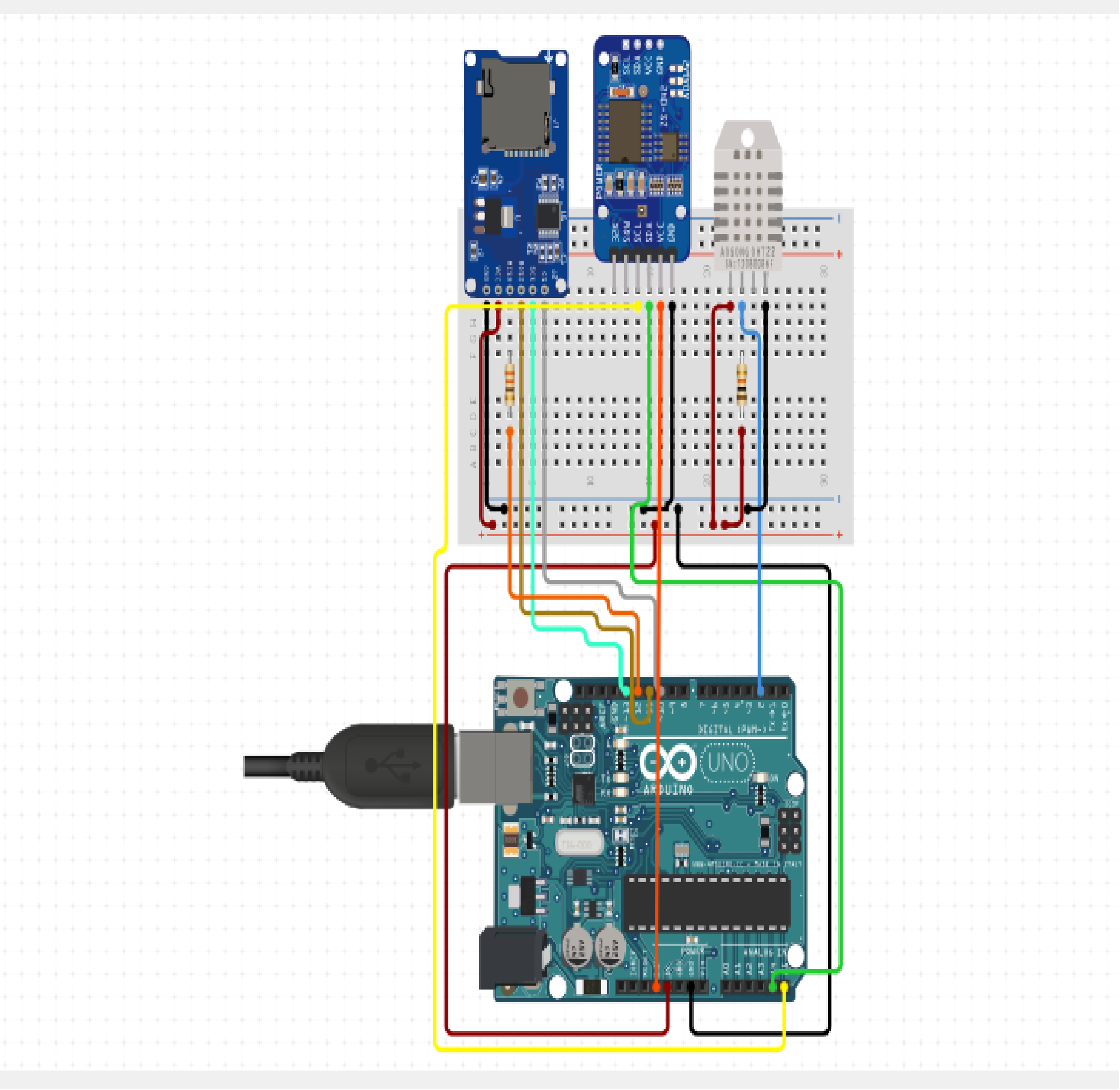
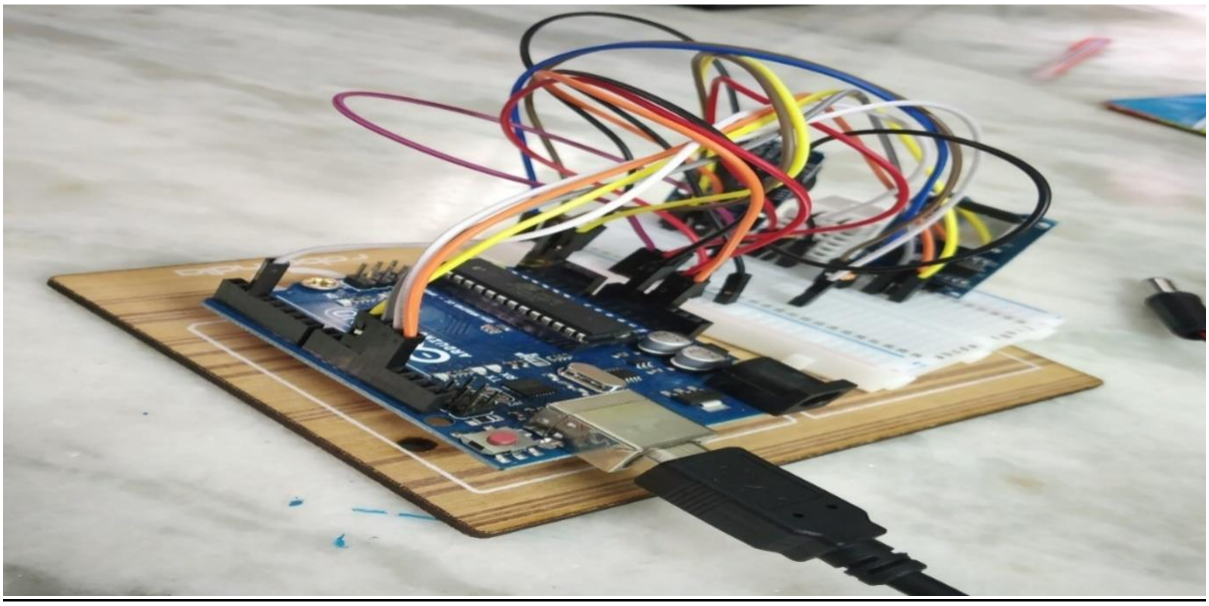
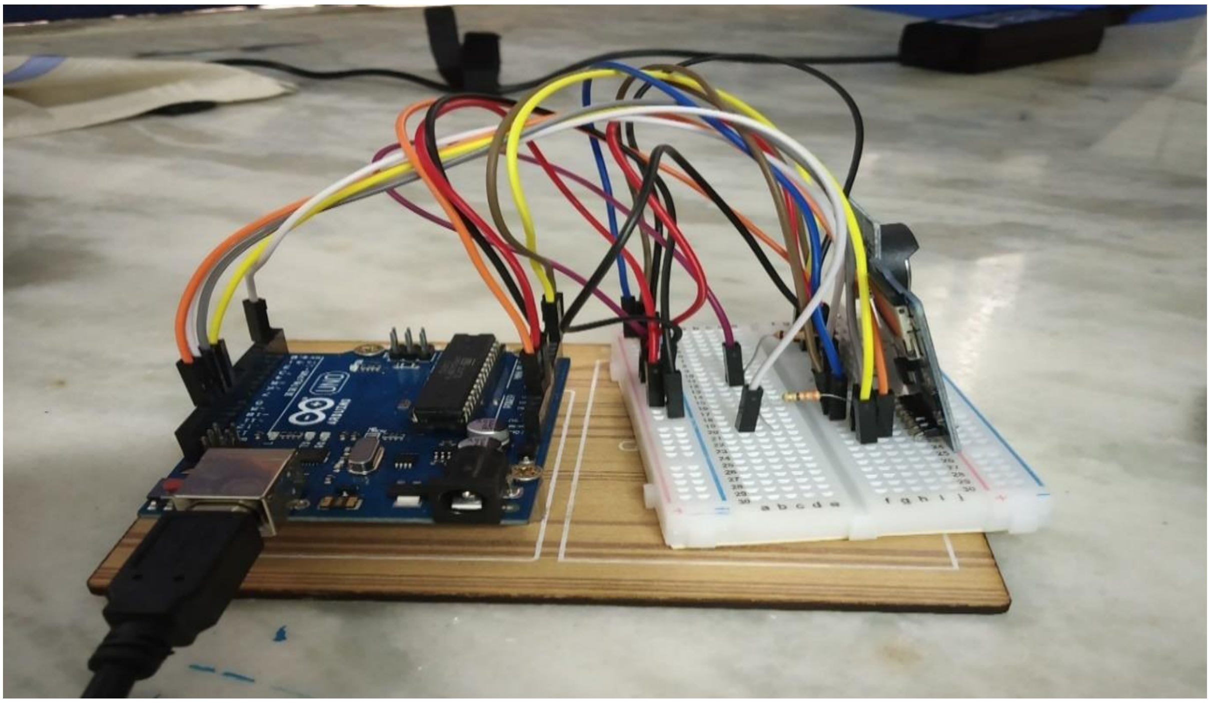


Fig 10. Set up connection of the project



(A)



(B)

Fig 11(A, B). Set up connection of the project made by us

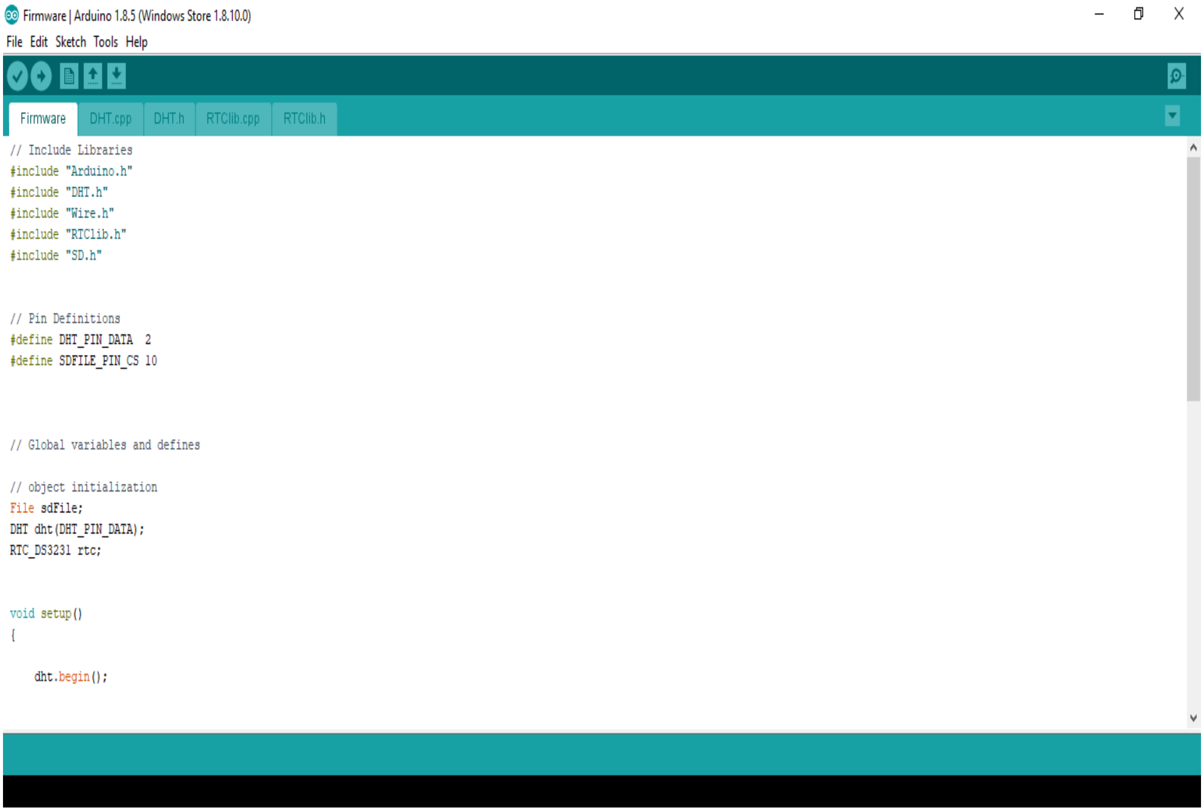
When all the connections and wiring are done, the code should be written in IDE and the codes written in IDE tells the Arduino to function so that the measurement obtained from sensor can be stored in the memory storage and can be displayed in the computer.

The programming codes that stores humidity and temperature readings with Arduino and DHT22 sensor are given below. The codes were developed with the help of an original source (circuito.io).

CODE:-

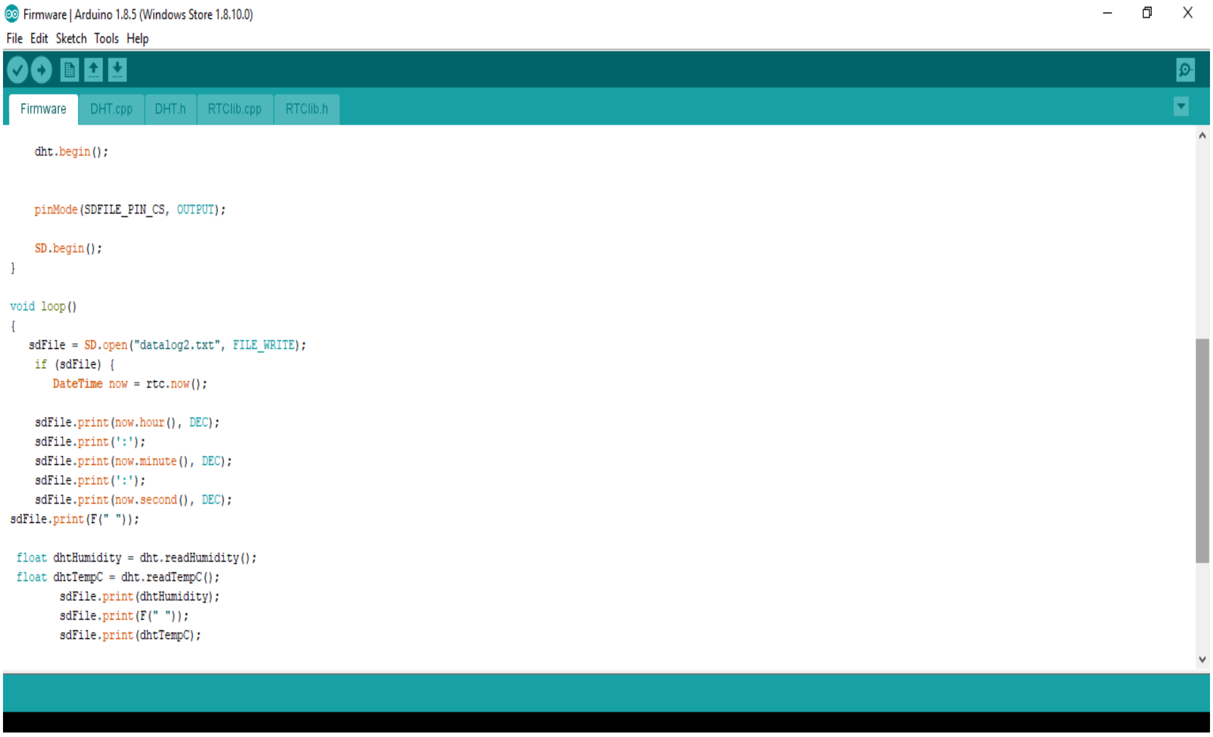
```
#include"Arduino.h"
#include"DHT.h"
#include"Wire.h"
#include"RTClib.h"
#include"SD.h"

#define DHT_PIN_DATA  2
#define SDFILE_PIN_CS 10
File sdFile;
  DHT dht(DHT_PIN_DATA);
  RTC_DS3231 rtc;
void setup()
{
     dht.begin();
     pinMode(SDFILE_PIN_CS, OUTPUT);
     SD.begin();
}
    void loop()
{
     sdFile = SD.open("file_name.txt", FILE_WRITE);
  if (sdFile)
      {
         DateTime now = rtc.now();
sdFile.print(now.hour(), DEC);
sdFile.print(':');
sdFile.print(now.minute(), DEC);
sdFile.print(':');
sdFile.print(now.second(), DEC);
sdFile.print(F(" "));
float dhtHumidity = dht.readHumidity();
float dhtTempC = dht.readTempC();
sdFile.print(dhtHumidity);
 sdFile.print(F(" "));
sdFile.print(dhtTempC);
sdFile.println();
sdFile.close();
delay(10000);
   }
}
```

After writing, the codes given above should be verified by IDE and when the verification completes the program is ready to be uploaded in Arduino.


Firmware | Arduino 1.8.5 (Windows Store 1.8.10.0)
File Edit Sketch Tools Help
Firmware
DHT.cpp
DHT.h
RTClib.cpp
RTClib.h

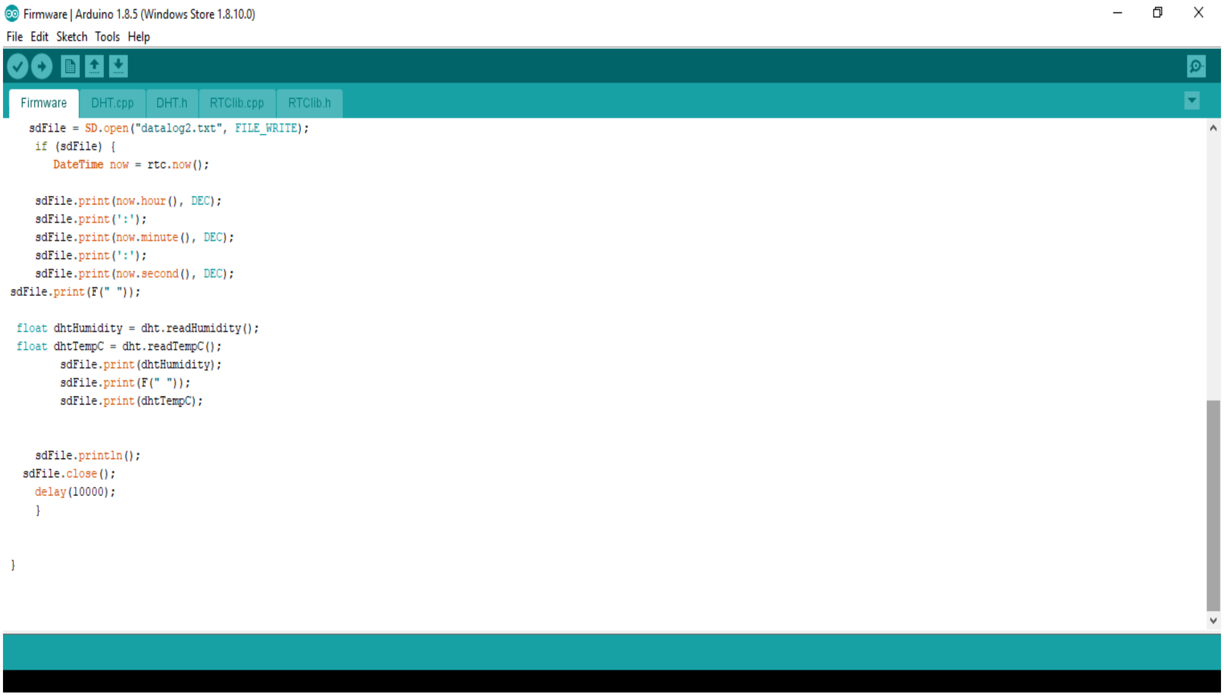
```
// Include Libraries
#include "Arduino.h"
#include "DHT.h"
#include "Wire.h"
#include "RTClib.h"
#include "SD.h"


// Pin Definitions
#define DHT_PIN_DATA  2
#define SDFILE_PIN_CS 10



// Global variables and defines

// object initialization
File sdFile;
DHT dht(DHT_PIN_DATA);
RTC_DS3231 rtc;


void setup()
{

    dht.begin();
```

(A)


Firmware | Arduino 1.8.5 (Windows Store 1.8.10.0)
File Edit Sketch Tools Help
Firmware
DHT.cpp
DHT.h
RTClib.cpp
RTClib.h

```
    dht.begin();


    pinMode(SDFILE_PIN_CS, OUTPUT);

    SD.begin();
}

void loop()
{
   sdFile = SD.open("datalog2.txt", FILE_WRITE);
    if (sdFile) {
       DateTime now = rtc.now();

    sdFile.print(now.hour(), DEC);
    sdFile.print(':');
    sdFile.print(now.minute(), DEC);
    sdFile.print(':');
    sdFile.print(now.second(), DEC);
sdFile.print(F(" "));

 float dhtHumidity = dht.readHumidity();
 float dhtTempC = dht.readTempC();
        sdFile.print(dhtHumidity);
        sdFile.print(F(" "));
        sdFile.print(dhtTempC);
```

(B)


Firmware | Arduino 1.8.5 (Windows Store 1.8.10.0)
File Edit Sketch Tools Help
Firmware
DHT.cpp
DHT.h
RTClib.cpp
RTClib.h

```
   sdFile = SD.open("datalog2.txt", FILE_WRITE);
    if (sdFile) {
       DateTime now = rtc.now();

    sdFile.print(now.hour(), DEC);
    sdFile.print(':');
    sdFile.print(now.minute(), DEC);
    sdFile.print(':');
    sdFile.print(now.second(), DEC);
sdFile.print(F(" "));

 float dhtHumidity = dht.readHumidity();
 float dhtTempC = dht.readTempC();
        sdFile.print(dhtHumidity);
        sdFile.print(F(" "));
        sdFile.print(dhtTempC);


    sdFile.println();
  sdFile.close();
    delay(10000);
    }


}
```

(C)

Fig 12(A,B,C).Arduino IDE interface

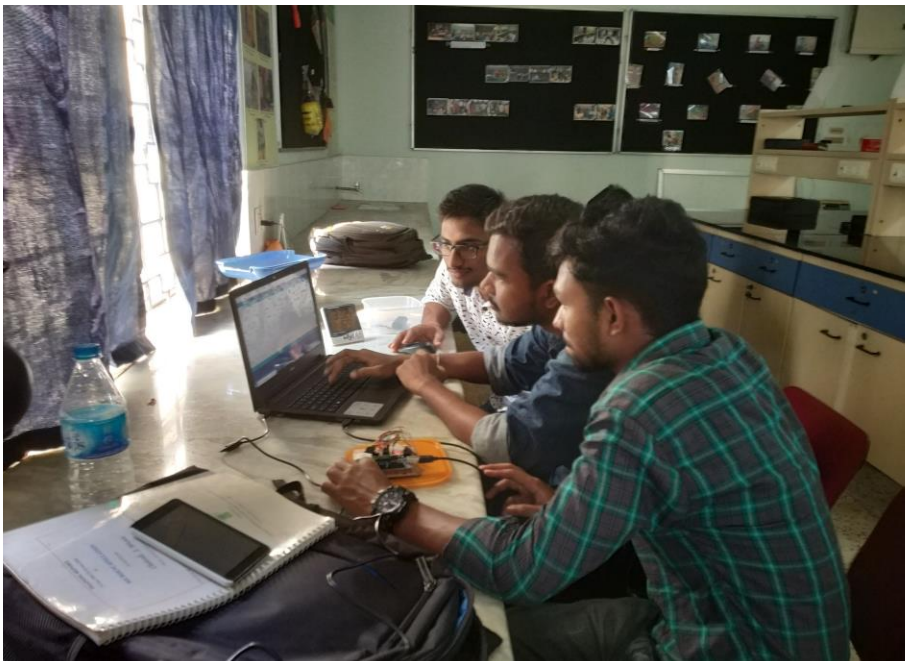
When these codes are written in IDE and uploaded to Arduino, the humidity and temperature measurement starts to store the values which is shown in the figure given below. The serial monitor gives the humidity and temperature measurement result in a frequency set as “delay ()” while writing codes.

RESULT AND ANALYSIS:-

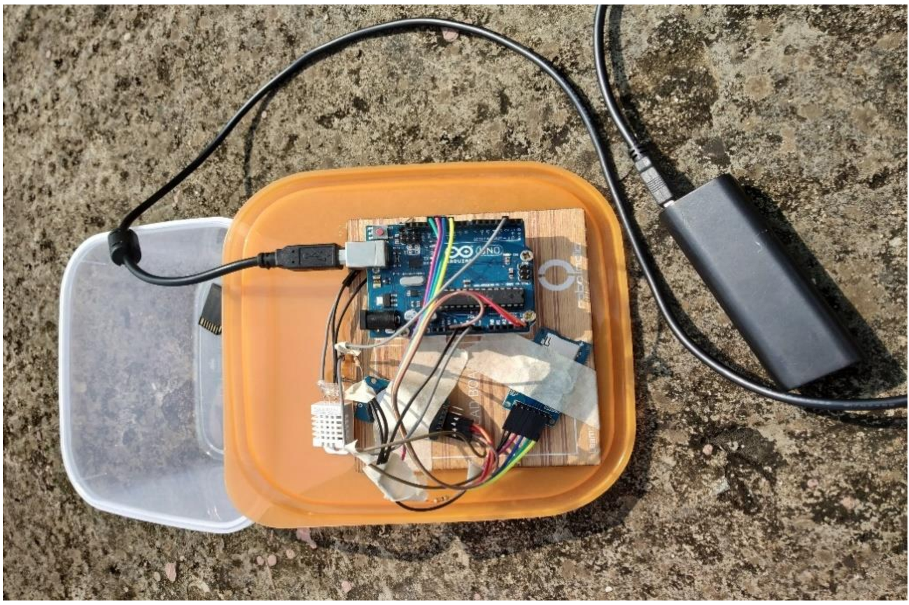
After wiring and writing the code, the program was run and the built device was successful in measuring humidity and temperature.

To study the performance characteristics of the used Arduino-based humidity and temperature sensor, the test was done in 2 different conditions. The first test was done inside an environmental laboratory room (with 5min interval) and the second test was done outside laboratory (with 15 min interval), respectively.

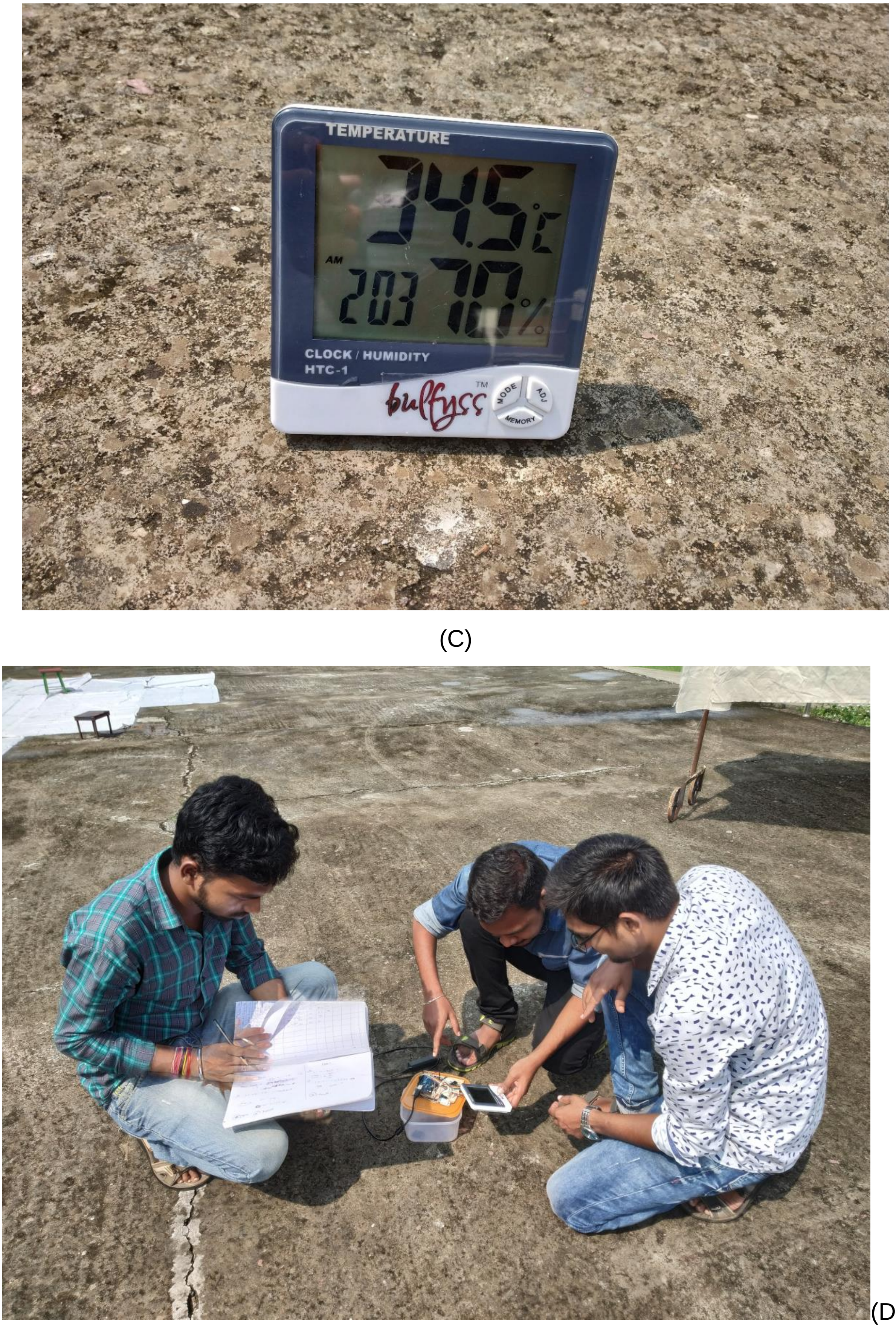
The measured data was analysed using excel. The following tables and graphs show the comparison of the commercially available temperature and humidity sensor and the developed sensor.



(A)



(B)



(C)

(D)

Fig 13(A,B,C,D). Setup and Calibration operation

Table.5- Humidity Readings of commercial and developed sensor with five mins interval

Time	Relative Humidity (%)	
	Commercial sensor	Developed Sensor
14:23:00	33.9	80.1
14:28:00	44	80.7
14:33:00	34	71.3
14:38:00	29	60.4
14:43:00	30	59.2
14:48:00	36	61.1
14:53:00	32	70.8
14:58:00	31	58.6
15:03:00	29	55.4
15:08:00	31	58.8
15:13:00	36	59

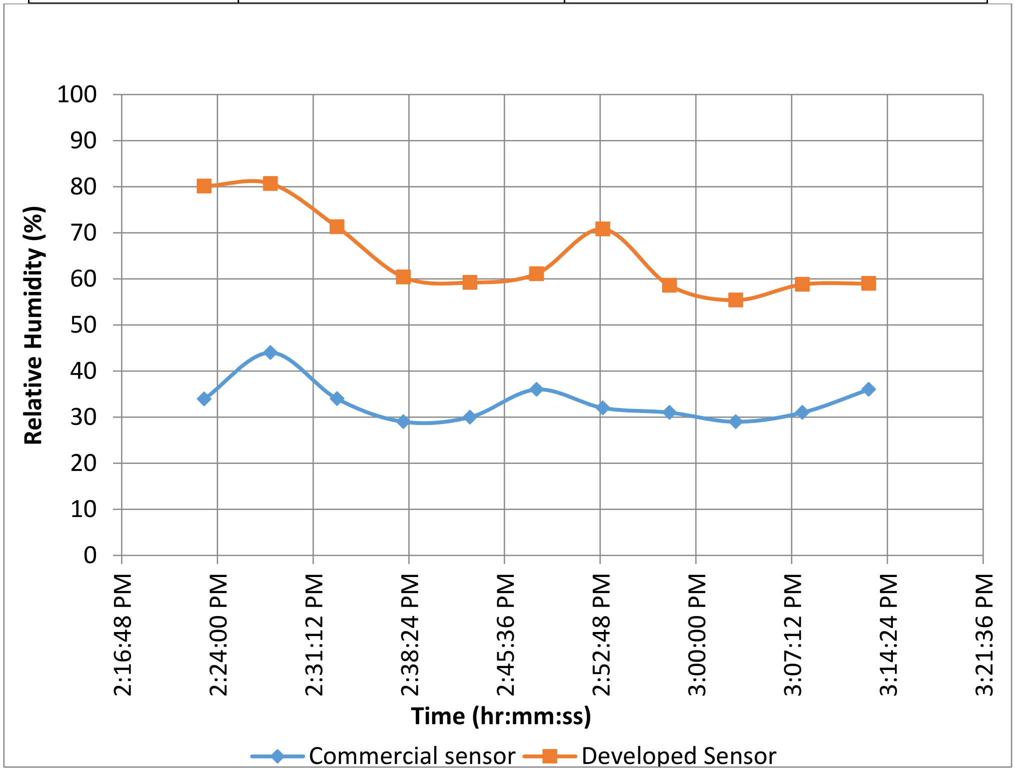


Fig 14. Comparison of commercial sensor and developed sensor for relative humidity (interval - 5 mins)

Applying paired "t-test";

H_0: $\mu_1 = \mu_2$

t $_{cal}$ = 8.589 and t $_{tab}$ = 2.086

t $_{cal}$ is greater than t $_{tab}$ value hence we reject the null hypothesis, thus the values are significantly different hence the developed sensor does not give humidity with significant accuracy as compared to commercial sensor.

Table.6- Humidity Readings of commercial and developed sensor with 15 mins interval

Time	Relative Humidity (%)	
	Commercial sensor	Developed sensor
10:53:00	53	70.9
11:08:00	51	71.8
11:23:00	52	63.9
11:38:00	54	69.1
11:53:00	52	68.1
12:08:00	54	64.9
12:23:00	53	68.2
12:38:00	54	67.3

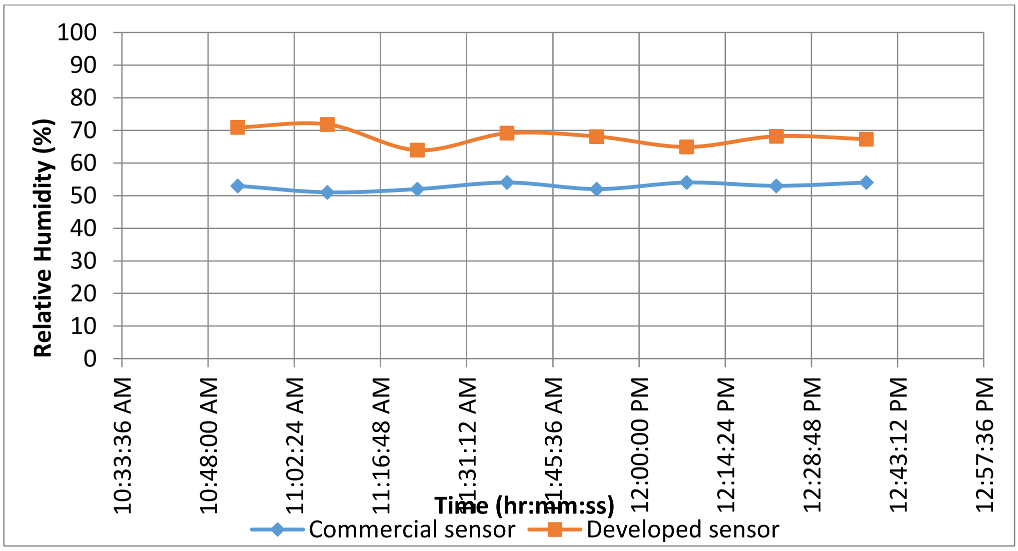


Fig 15. Comparison of commercial sensor and developed sensor for relative humidity (%) (interval - 15 mins)

Applying paired “t-test”;

H_0: $\mu_1 = \mu_2$

$t_{cal} = 7.441$ and $t_{tab} = 2.144$

t_{cal} is greater than t_{tab} value hence we reject the null hypothesis, thus the values are significantly different hence the developed sensor does not give humidity with significant accuracy as compared to commercial sensor.

Table.7- Temperature Readings of commercial and developed sensor with five mins interval

Time	Temperature	
	Commercial sensor	Developed Sensor
14:23:00	33.9	32.8
14:28:00	40.4	32.8
14:33:00	44	40.5
14:38:00	46.4	42.6
14:43:00	45.2	42.8
14:48:00	43.2	38.9
14:53:00	45.5	38.2
14:58:00	46	41.8
15:03:00	46.4	42.3
15:08:00	44.8	41.7

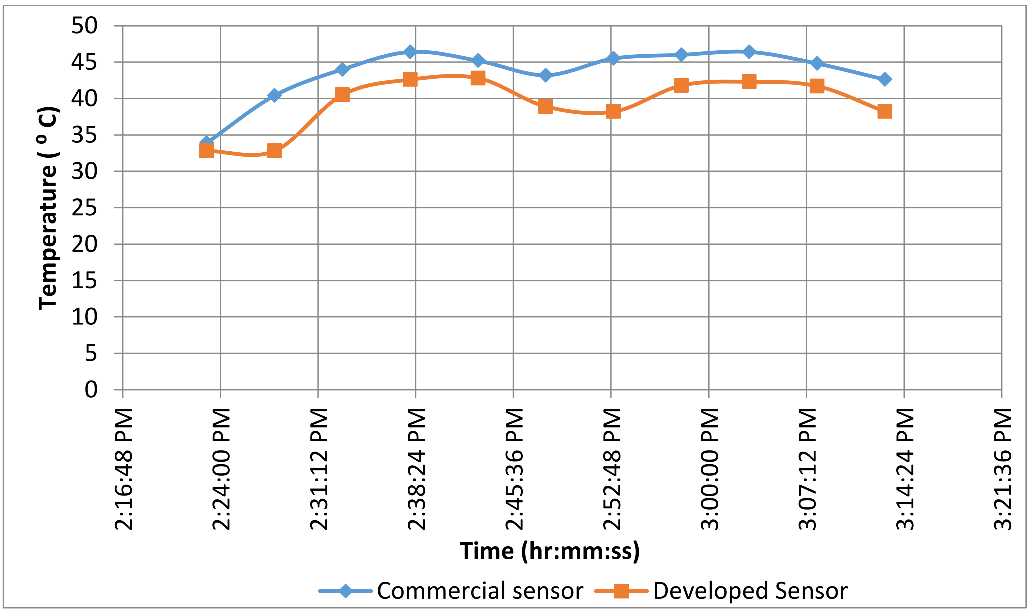


Fig 16. Comparison of commercial sensor and developed sensor for temperature(^{o}C) (interval - 5 mins)

Applying paired "t-test";

H_0: $\mu_1 = \mu_2$

t $_{cal}$ = 5.418 and t $_{tab}$ = 2.085

t $_{cal}$ is greater than t $_{tab}$ value hence we reject the null hypothesis, thus the values are significantly different hence the developed sensor does not give humidity with significant accuracy as compared to commercial sensor.

Table.8- Temperature Readings of commercial and developed sensor with 15 mins interval

Time	Temperature	
	Commercial sensor	Developed sensor
10:53:00	36	35.5
11:08:00	37.5	35.5
11:23:00	36.3	36.2
11:38:00	36.1	35.1
11:53:00	36.7	34.8
12:08:00	36.5	35.7
12:23:00	36.6	35.1
12:38:00	36.6	35.5

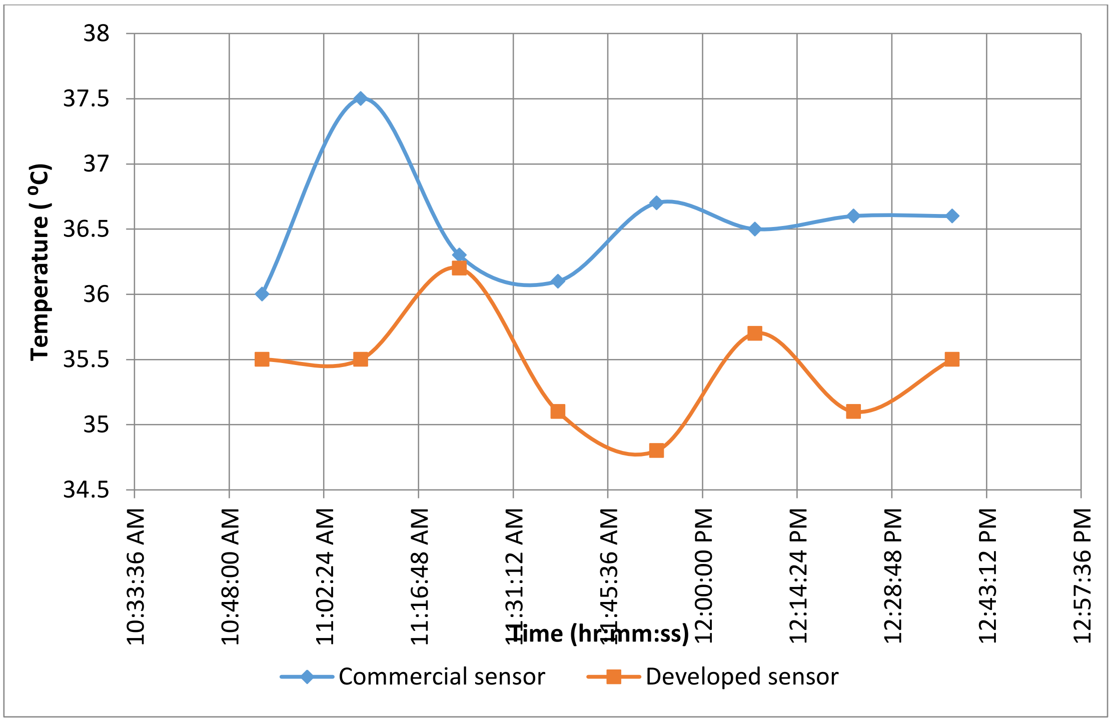


Fig 17. Comparison of commercial sensor and developed sensor for temperature (^{0}C) (interval - 15 mins)

Applying paired "t-test";

H_0: $\mu_1 = \mu_2$

t $_{cal}$ = 5.84 and t $_{tab}$ = 2.086

t $_{cal}$ is greater than t $_{tab}$ value hence we reject the null hypothesis, thus the values are significantly different hence the developed sensor does not give humidity with significant accuracy as compared to commercial sensor.

CONCLUSION:-

The work was successful in building a recording device which works as a thermohygrometer for measuring temperature and humidity; it is capable of measuring humidity and temperature both indoors and outdoors (Fields).Compared to expensive sensor, the Arduino-based recording system successfully reduces the power consumption, cost and complexity of the process. It achieved logging of the data in real time. it also facilitated remote recording of the data in a memory card without need of an laptop or a computer. The performance of the sensor with regards to recording temperature was acceptable with minor modification of code in future to compensate for the deviation of its readings as compared to commercial sensors. Same condition is applicable for the humidity readings.

The developed recording system can be used with various other sensors for measuring parameters such as wind velocity, soil moisture, distance etc with a appropriate sensor and compatible coding.

Arduino- based devices are the new possibilities for developing smart devices freely with small budget and simple work. The accelerating race of advanced technology outdates the technology used in Arduino Uno in no time; advanced software working similarly are available.

The project was interesting and was practically helpful to learn to use microcontrollers (Arduino), programming language C and basic electronics. This was a very helpful project in learning and understanding the world of microcontrollers and use of microcontrollers in day to day life.

References

1. Field Temperature Measurement (PDF) - EPA (available at- https://www.epa.gov/sites/production/files/2015-06/documents/Field-Temperature-Measurement.pdf)
2. https://en.wikipedia.org/wiki/Humidity
3. https://en.wikipedia.org/wiki/Stevenson_screen
4. https://en.wikipedia.org/wiki/Temperature
5. https://www.circuito.io/app?components=512,10167,11021,821989,1671987
6. Methods for the Determination of HumidityUploaded by Muhammad Salman (available at -https://www.scribd.com/document/152524943/Methods-for-the-Determination-of-Humidity)
7. Nagendra DangiMonitoring environmental parameters: humidity and temperature using Arduino based microcontroller and sensorsMicrocontroller based building monitoring system. (available at- http://www.theseus.fi/handle/10024/142235
8. PRO MIX , How Does Humidity Influence Crop Quality(available at- https://www.pthorticulture.com/en/training-center/how-does-humidity-influence-crop-quality/)
9. The Arduino Starter Guide, by Kunwar Imran.

APPPENDIX

LIBRARY:-

DHT.cpp

```
/* DHT library
MIT license
written by Adafruit Industries
*/
DHT::DHT(uint8_t pin, uint8_t type, uint8_t count)
{
  _pin = pin;
  _type = type;
  #ifdef __AVR
    _bit = digitalPinToBitMask(pin);
    _port = digitalPinToPort(pin);
  #endif
  _maxcycles = microsecondsToClockCycles(1000
}
void DHT::begin(void) {
  pinMode(_pin, INPUT_PULLUP);
  _lastreadtime = -MIN_INTERVAL;
  DEBUG_PRINT("Max clock cycles: "); DEBUG_PRINTLN(_maxcycles, DEC);
}
float DHT::readTemperature(bool S, bool force) {
  float f = NAN;
  if (read(force)) {
    switch (_type) {
    case DHT11:
      f = data[2];
      if(S) {
        f = convertCtoF(f);
      }
      break;
    case DHT22:
    case DHT21:
      f = data[2] & 0x7F;
      f *= 256;
      f += data[3];
      f *= 0.1;
      if (data[2] & 0x80) {
        f *= -1;
      }
      if(S) {
        f = convertCtoF(f);
      }
      break;
    }
  }
  if (isnan(f))
  {
    Serial.println("Failed to read Temperature!");
  }
  return f;
}

float DHT::convertCtoF(float c) {
  return c * 1.8 + 32;
}
```

```
float DHT::convertFtoC(float f) {
  return (f - 32) * 0.55555;
}

float DHT::readHumidity(bool force) {
  float f = NAN;
  if (read()) {
    switch (_type) {
    case DHT11:
      f = data[0];
      break;
    case DHT22:
    case DHT21:
      f = data[0];
      f *= 256;
      f += data[1];
      f *= 0.1;
      break;
    }
  }
  if (isnan(f))
  {
    Serial.println("Failed to read Humidity!");
  }
  return f;
}

float DHT::readTempC()
{
  return readTemperature();
}
float DHT::readTempF()
{
  return convertCtoF(readTemperature());
}
float DHT::computeHeatIndex(float temperature, float percentHumidity, bool isFahrenheit) {
  float hi;
  if (!isFahrenheit)
    temperature = convertCtoF(temperature);
  hi = 0.5 * (temperature + 61.0 + ((temperature - 68.0) * 1.2) + (percentHumidity * 0.094));

  if (hi > 79) {
    hi = -42.379 +
             2.04901523 * temperature +
            10.14333127 * percentHumidity +
            -0.22475541 * temperature*percentHumidity +
            -0.00683783 * pow(temperature, 2) +
            -0.05481717 * pow(percentHumidity, 2) +
             0.00122874 * pow(temperature, 2) * percentHumidity +
             0.00085282 * temperature*pow(percentHumidity, 2) +
            -0.00000199 * pow(temperature, 2) * pow(percentHumidity, 2);

    if((percentHumidity < 13) && (temperature >= 80.0) && (temperature <= 112.0))
      hi -= ((13.0 - percentHumidity) * 0.25) * sqrt((17.0 - abs(temperature - 95.0)) * 0.05882);
    else if((percentHumidity > 85.0) && (temperature >= 80.0) && (temperature <= 87.0))
      hi += ((percentHumidity - 85.0) * 0.1) * ((87.0 - temperature) * 0.2);
  }
  return isFahrenheit ? hi : convertFtoC(hi);
}
boolean DHT::read(bool force) {
```

```
uint32_t currenttime = millis();
if (!force && ((currenttime - _lastreadtime) < 2000)) {
  return _lastresult; // return last correct measurement
}
_lastreadtime = currenttime;
data[0] = data[1] = data[2] = data[3] = data[4] = 0;
digitalWrite(_pin, HIGH);
delay(250);
pinMode(_pin, OUTPUT);
digitalWrite(_pin, LOW);
delay(20);

uint32_t cycles[80];
  InterruptLock lock;
  digitalWrite(_pin, HIGH);
  delayMicroseconds(40);
  pinMode(_pin, INPUT_PULLUP);
  delayMicroseconds(10);  // Delay a bit to let sensor pull data line low.
  if (expectPulse(LOW) == 0) {
    DEBUG_PRINTLN(F("Timeout waiting for start signal low pulse."));
    _lastresult = false;
    return _lastresult;
  }
  if (expectPulse(HIGH) == 0) {
    DEBUG_PRINTLN(F("Timeout waiting for start signal high pulse."));
    _lastresult = false;
    return _lastresult;
  }
  for (int i=0; i<80; i+=2) {
    cycles[i]   = expectPulse(LOW);
    cycles[i+1] = expectPulse(HIGH);
  }
}
for (int i=0; i<40; ++i) {
  uint32_t lowCycles  = cycles[2*i];
  uint32_t highCycles = cycles[2*i+1];
  if ((lowCycles == 0) || (highCycles == 0)) {
    DEBUG_PRINTLN(F("Timeout waiting for pulse."));
    _lastresult = false;
    return _lastresult;
  }
  data[i/8] <<= 1;
  if (highCycles > lowCycles) {
    data[i/8] |= 1;
  }
}
DEBUG_PRINTLN(F("Received:"));
DEBUG_PRINT(data[0], HEX); DEBUG_PRINT(F(", "));
DEBUG_PRINT(data[1], HEX); DEBUG_PRINT(F(", "));
DEBUG_PRINT(data[2], HEX); DEBUG_PRINT(F(", "));
DEBUG_PRINT(data[3], HEX); DEBUG_PRINT(F(", "));
DEBUG_PRINT(data[4], HEX); DEBUG_PRINT(F(" =? "));
DEBUG_PRINTLN((data[0] + data[1] + data[2] + data[3]) & 0xFF, HEX);
if (data[4] == ((data[0] + data[1] + data[2] + data[3]) & 0xFF))
{
  _lastresult = true;
  return _lastresult;
}
else {
  DEBUG_PRINTLN(F("Checksum failure!"));
```

```
    _lastresult = false;
    return _lastresult;
  }
}
uint32_t DHT::expectPulse(bool level) {
  uint32_t count = 0;
  #ifdef __AVR
    uint8_t portState = level ? _bit : 0;
    while ((*portInputRegister(_port) & _bit) == portState) {
      if (count++ >= _maxcycles) {
        return 0; // Exceeded timeout, fail.
      } }
  #else
    while (digitalRead(_pin) == level) {
      if (count++ >= _maxcycles) {
        return 0; // Exceeded timeout, fail.
      } }
  #endif
  return count;
}
```

DHT.h

```
/* DHT library

MIT license
written by Adafruit Industries
*/
#ifndef DHT_H
#define DHT_H
#if ARDUINO >= 100
 #include "Arduino.h"
#else
 #include "WProgram.h"
#endif
#define DEBUG_PRINTER Serial

#define DHTTYPE DHT22

#ifdef DHT_DEBUG
  #define DEBUG_PRINT(...) { DEBUG_PRINTER.print(__VA_ARGS__); }
  #define DEBUG_PRINTLN(...) { DEBUG_PRINTER.println(__VA_ARGS__); }
#else
  #define DEBUG_PRINT(...) {}
  #define DEBUG_PRINTLN(...) {}
#endif
#define DHT11 11
#define DHT22 22
#define DHT21 21
#define AM2301 21
class DHT {
  public:
   DHT(uint8_t pin, uint8_t type = DHTTYPE, uint8_t count=6);
   void begin(void);
   float readTemperature(bool S=false, bool force=false);
   float readTempC();
   float readTempF();
   float convertCtoF(float);
   float convertFtoC(float);
   float computeHeatIndex(float temperature, float percentHumidity, bool isFahrenheit=true);
   float readHumidity(bool force=false);
boolean read(bool force=false);
```

```
 private:
  uint8_t data[5];
  uint8_t _pin, _type;
  #ifdef __AVR
    uint8_t _bit, _port;
  #endif
  uint32_t _lastreadtime, _maxcycles;
  bool _lastresult;

  uint32_t expectPulse(bool level);
}
class InterruptLock
 {
  public:
   InterruptLock() {
    noInterrupts();
   }
   ~InterruptLock() {
    interrupts();
}
};
#endif
```

RTClib.cpp

```
#include <Wire.h>
#include "RTClib.h"
#ifdef __AVR__
 #include <avr/pgmspace.h>
#elif defined(ESP8266)
 #include <pgmspace.h>
#elif defined(ARDUINO_ARCH_SAMD)
// nothing special needed
#elif defined(ARDUINO_SAM_DUE)
 #define PROGMEM
 #define pgm_read_byte(addr) (*(const unsigned char *)(addr))
 #define Wire Wire1
#endif
#if (ARDUINO >= 100)
 #include <Arduino.h> // capital A so it is error prone on case-sensitive filesystems
 // Macro to deal with the difference in I2C write functions from old and new Arduino versions.
 #define _I2C_WRITE write
 #define _I2C_READ  read
#else
 #include <WProgram.h>
 #define _I2C_WRITE send
 #define _I2C_READ  receive
#endif
static uint8_t read_i2c_register(uint8_t addr, uint8_t reg) {
  Wire.beginTransmission(addr);
  Wire._I2C_WRITE((byte)reg);
  Wire.endTransmission();
  Wire.requestFrom(addr, (byte)1);
  return Wire._I2C_READ();
}
static void write_i2c_register(uint8_t addr, uint8_t reg, uint8_t val) {
  Wire.beginTransmission(addr);
  Wire._I2C_WRITE((byte)reg);
  Wire._I2C_WRITE((byte)val);
  Wire.endTransmission();
}
```

```
const uint8_t daysInMonth [] PROGMEM = { 31,28,31,30,31,30,31,31,30,31,30,31 };
static uint16_t date2days(uint16_t y, uint8_t m, uint8_t d) {
    if (y >= 2000)
        y -= 2000;
    uint16_t days = d;
    for (uint8_t i = 1; i < m; ++i)
        days += pgm_read_byte(daysInMonth + i - 1);
    if (m > 2 && y % 4 == 0)
        ++days;
    return days + 365 * y + (y + 3) / 4 - 1;
}

static long time2long(uint16_t days, uint8_t h, uint8_t m, uint8_t s) {
    return ((days * 24L + h) * 60 + m) * 60 + s;
}
DateTime::DateTime (uint32_t t) {
  t -= SECONDS_FROM_1970_TO_2000;    // bring to 2000 timestamp from 1970
    ss = t % 60;
    t /= 60;
    mm = t % 60;
    t /= 60;
    hh = t % 24;
    uint16_t days = t / 24;
    uint8_t leap;
    for (yOff = 0; ; ++yOff) {
        leap = yOff % 4 == 0;
        if (days < 365 + leap)
            break;
        days -= 365 + leap;
    }
    for (m = 1; ; ++m) {
        uint8_t daysPerMonth = pgm_read_byte(daysInMonth + m - 1);
        if (leap && m == 2)
            ++daysPerMonth;
        if (days < daysPerMonth)
            break;
        days -= daysPerMonth;
    }
    d = days + 1;
}

DateTime::DateTime (uint16_t year, uint8_t month, uint8_t day, uint8_t hour, uint8_t min, uint8_t sec)
{
    if (year >= 2000)
        year -= 2000;
    yOff = year;
    m = month;
    d = day;
    hh = hour;
    mm = min;
    ss = sec;
}
DateTime::DateTime (const DateTime& copy):
  yOff(copy.yOff),
  m(copy.m),
  d(copy.d),
  hh(copy.hh),
  mm(copy.mm),
  ss(copy.ss)
{}
```

```
static uint8_t conv2d(const char* p) {
   uint8_t v = 0;
   if ('0' <= *p && *p <= '9')
      v = *p - '0';
   return 10 * v + *++p - '0';
}
DateTime::DateTime (const char* date, const char* time) {
   yOff = conv2d(date + 9);
   switch (date[0]) {
      case 'J': m = (date[1] == 'a') ? 1 : ((date[2] == 'n') ? 6 : 7); break;
      case 'F': m = 2; break;
      case 'A': m = date[2] == 'r' ? 4 : 8; break;
      case 'M': m = date[2] == 'r' ? 3 : 5; break;
      case 'S': m = 9; break;
      case 'O': m = 10; break;
      case 'N': m = 11; break;
      case 'D': m = 12; break;
   }
   d = conv2d(date + 4);
   hh = conv2d(time);
   mm = conv2d(time + 3);
   ss = conv2d(time + 6);
}
DateTime::DateTime (const __FlashStringHelper* date, const __FlashStringHelper* time) {
   char buff[11];
   memcpy_P(buff, date, 11);
   yOff = conv2d(buff + 9);
   switch (buff[0]) {
      case 'J': m = (buff[1] == 'a') ? 1 : ((buff[2] == 'n') ? 6 : 7); break;
      case 'F': m = 2; break;
      case 'A': m = buff[2] == 'r' ? 4 : 8; break;
      case 'M': m = buff[2] == 'r' ? 3 : 5; break;
      case 'S': m = 9; break;
      case 'O': m = 10; break;
      case 'N': m = 11; break;
      case 'D': m = 12; break;
   }
   d = conv2d(buff + 4);
   memcpy_P(buff, time, 8);
   hh = conv2d(buff);
   mm = conv2d(buff + 3);
   ss = conv2d(buff + 6);
}
uint8_t DateTime::dayOfTheWeek() const {
   uint16_t day = date2days(yOff, m, d);
   return (day + 6) % 7; // Jan 1, 2000 is a Saturday, i.e. returns 6
}
uint32_t DateTime::unixtime(void) const {
  uint32_t t;
  uint16_t days = date2days(yOff, m, d);
  t = time2long(days, hh, mm, ss);
  t += SECONDS_FROM_1970_TO_2000;  // seconds from 1970 to 2000
  return t;
}
long DateTime::secondstime(void) const {
  long t;
  uint16_t days = date2days(yOff, m, d);
  t = time2long(days, hh, mm, ss);
  return t;
}
```

```
DateTime DateTime::operator+(const TimeSpan& span) {
  return DateTime(unixtime()+span.totalseconds());
}
DateTime DateTime::operator-(const TimeSpan& span) {
  return DateTime(unixtime()-span.totalseconds());
}
TimeSpan DateTime::operator-(const DateTime& right) {
  return TimeSpan(unixtime()-right.unixtime());
}
TimeSpan::TimeSpan (int32_t seconds):
  _seconds(seconds)
{}
TimeSpan::TimeSpan (int16_t days, int8_t hours, int8_t minutes, int8_t seconds):
  _seconds((int32_t)days*86400L + (int32_t)hours*3600 + (int32_t)minutes*60 + seconds)
{}
TimeSpan::TimeSpan (const TimeSpan& copy):
  _seconds(copy._seconds)
{}
TimeSpan TimeSpan::operator+(const TimeSpan& right) {
  return TimeSpan(_seconds+right._seconds);
}
TimeSpan TimeSpan::operator-(const TimeSpan& right) {
  return TimeSpan(_seconds-right._seconds);
}
static uint8_t bcd2bin (uint8_t val) { return val - 6 * (val >> 4); }
static uint8_t bin2bcd (uint8_t val) { return val + 6 * (val / 10);
boolean RTC_DS1307::begin(void) {
  Wire.begin();
  return true;
}
uint8_t RTC_DS1307::isrunning(void) {
  Wire.beginTransmission(DS1307_ADDRESS);
  Wire._I2C_WRITE((byte)0);
  Wire.endTransmission();
  Wire.requestFrom(DS1307_ADDRESS, 1);
  uint8_t ss = Wire._I2C_READ();
  return !(ss>>7);
}
void RTC_DS1307::adjust(const DateTime& dt) {
  Wire.beginTransmission(DS1307_ADDRESS);
  Wire._I2C_WRITE((byte)0); // start at location 0
  Wire._I2C_WRITE(bin2bcd(dt.second()));
  Wire._I2C_WRITE(bin2bcd(dt.minute()));
  Wire._I2C_WRITE(bin2bcd(dt.hour()));
  Wire._I2C_WRITE(bin2bcd(0));
  Wire._I2C_WRITE(bin2bcd(dt.day()));
  Wire._I2C_WRITE(bin2bcd(dt.month()));
  Wire._I2C_WRITE(bin2bcd(dt.year() - 2000));
  Wire.endTransmission();
}
DateTime RTC_DS1307::now() {
  Wire.beginTransmission(DS1307_ADDRESS);
  Wire._I2C_WRITE((byte)0);
  Wire.endTransmission();
  Wire.requestFrom(DS1307_ADDRESS, 7);
  uint8_t ss = bcd2bin(Wire._I2C_READ() & 0x7F);
  uint8_t mm = bcd2bin(Wire._I2C_READ());
  uint8_t hh = bcd2bin(Wire._I2C_READ());
  Wire._I2C_READ();
  uint8_t d = bcd2bin(Wire._I2C_READ());
  uint8_t m = bcd2bin(Wire._I2C_READ());
  uint16_t y = bcd2bin(Wire._I2C_READ()) + 2000
  return DateTime (y, m, d, hh, mm, ss);
}
Ds1307SqwPinMode RTC_DS1307::readSqwPinMode() {
  int mode;
  Wire.beginTransmission(DS1307_ADDRESS);
  Wire._I2C_WRITE(DS1307_CONTROL);
  Wire.endTransmission();
  Wire.requestFrom((uint8_t)DS1307_ADDRESS, (uint8_t)1);
  mode = Wire._I2C_READ();
  mode &= 0x93;
  return static_cast<Ds1307SqwPinMode>(mode);
}
void RTC_DS1307::writeSqwPinMode(Ds1307SqwPinMode mode) {
  Wire.beginTransmission(DS1307_ADDRESS);
  Wire._I2C_WRITE(DS1307_CONTROL);
  Wire._I2C_WRITE(mode);
  Wire.endTransmission();
}
void RTC_DS1307::readnvram(uint8_t* buf, uint8_t size, uint8_t address) {
  int addrByte = DS1307_NVRAM + address;
  Wire.beginTransmission(DS1307_ADDRESS);
  Wire._I2C_WRITE(addrByte);
  Wire.endTransmission()
  Wire.requestFrom((uint8_t) DS1307_ADDRESS, size);
  for (uint8_t pos = 0; pos < size; ++pos) {
    buf[pos] = Wire._I2C_READ();
  }
}
void RTC_DS1307::writenvram(uint8_t address, uint8_t* buf, uint8_t size) {
  int addrByte = DS1307_NVRAM + address;
  Wire.beginTransmission(DS1307_ADDRESS);
  Wire._I2C_WRITE(addrByte);
  for (uint8_t pos = 0; pos < size; ++pos) {
    Wire._I2C_WRITE(buf[pos]);
  }
  Wire.endTransmission();
}
uint8_t RTC_DS1307::readnvram(uint8_t address) {
  uint8_t data;
  readnvram(&data, 1, address);
  return data;
}
void RTC_DS1307::writenvram(uint8_t address, uint8_t data) {
  writenvram(address, &data, 1);
}

long RTC_Millis::offset = 0;
void RTC_Millis::adjust(const DateTime& dt) {
    offset = dt.unixtime() - millis() / 1000;
}
DateTime RTC_Millis::now() {
  return (uint32_t)(offset + millis() / 1000);
}
boolean RTC_PCF8523::begin(void) {
  Wire.begin();
  return true;
}
boolean RTC_PCF8523::initialized(void) {
  Wire.beginTransmission(PCF8523_ADDRESS);
  Wire._I2C_WRITE((byte)PCF8523_CONTROL_3);
  Wire.endTransmission();
  Wire.requestFrom(PCF8523_ADDRESS, 1);
  uint8_t ss = Wire._I2C_READ();
  return ((ss & 0xE0) != 0xE0);
}
void RTC_PCF8523::adjust(const DateTime& dt) {
  Wire.beginTransmission(PCF8523_ADDRESS);
  Wire._I2C_WRITE((byte)3); // start at location 3
  Wire._I2C_WRITE(bin2bcd(dt.second()));
  Wire._I2C_WRITE(bin2bcd(dt.minute()));
  Wire._I2C_WRITE(bin2bcd(dt.hour()));
  Wire._I2C_WRITE(bin2bcd(dt.day()));
  Wire._I2C_WRITE(bin2bcd(0)); // skip weekdays
  Wire._I2C_WRITE(bin2bcd(dt.month()));
  Wire._I2C_WRITE(bin2bcd(dt.year() - 2000));
  Wire.endTransmission();
  Wire.beginTransmission(PCF8523_ADDRESS);
  Wire._I2C_WRITE((byte)PCF8523_CONTROL_3);
  Wire._I2C_WRITE((byte)0x00);
  Wire.endTransmission();
}
DateTime RTC_PCF8523::now() {
  Wire.beginTransmission(PCF8523_ADDRESS);
  Wire._I2C_WRITE((byte)3);
  Wire.endTransmission();
  Wire.requestFrom(PCF8523_ADDRESS, 7);
  uint8_t ss = bcd2bin(Wire._I2C_READ() & 0x7F);
  uint8_t mm = bcd2bin(Wire._I2C_READ());
  uint8_t hh = bcd2bin(Wire._I2C_READ());
  uint8_t d = bcd2bin(Wire._I2C_READ());
  Wire._I2C_READ();  // skip 'weekdays'
  uint8_t m = bcd2bin(Wire._I2C_READ());
  uint16_t y = bcd2bin(Wire._I2C_READ()) + 2000;
  return DateTime (y, m, d, hh, mm, ss);
}
Pcf8523SqwPinMode RTC_PCF8523::readSqwPinMode() {
  int mode;
  Wire.beginTransmission(PCF8523_ADDRESS);
  Wire._I2C_WRITE(PCF8523_CLKOUTCONTROL);
  Wire.endTransmission();
  Wire.requestFrom((uint8_t)PCF8523_ADDRESS, (uint8_t)1);
  mode = Wire._I2C_READ();
  mode >>= 3;
  mode &= 0x7;
  return static_cast<Pcf8523SqwPinMode>(mode);
}
void RTC_PCF8523::writeSqwPinMode(Pcf8523SqwPinMode mode) {
  Wire.beginTransmission(PCF8523_ADDRESS);
  Wire._I2C_WRITE(PCF8523_CLKOUTCONTROL);
  Wire._I2C_WRITE(mode << 3);
  Wire.endTransmission();
}
boolean RTC_DS3231::begin(void) {
  Wire.begin();
  return true;
}
bool RTC_DS3231::lostPower(void) {
  return (read_i2c_register(DS3231_ADDRESS, DS3231_STATUSREG) >> 7);
```

```
}
void RTC_DS3231::adjust(const DateTime& dt) {
  Wire.beginTransmission(DS3231_ADDRESS);
  Wire._I2C_WRITE((byte)0); // start at location 0
  Wire._I2C_WRITE(bin2bcd(dt.second()));
  Wire._I2C_WRITE(bin2bcd(dt.minute()));
  Wire._I2C_WRITE(bin2bcd(dt.hour()));
  Wire._I2C_WRITE(bin2bcd(0));
  Wire._I2C_WRITE(bin2bcd(dt.day()));
  Wire._I2C_WRITE(bin2bcd(dt.month()));
  Wire._I2C_WRITE(bin2bcd(dt.year() - 2000));
  Wire.endTransmission();
  uint8_t statreg = read_i2c_register(DS3231_ADDRESS, DS3231_STATUSREG);
  statreg &= ~0x80; // flip OSF bit
  write_i2c_register(DS3231_ADDRESS, DS3231_STATUSREG, statreg);
}
DateTime RTC_DS3231::now() {
  Wire.beginTransmission(DS3231_ADDRESS);
  Wire._I2C_WRITE((byte)0);
  Wire.endTransmission();
  Wire.requestFrom(DS3231_ADDRESS, 7);
  uint8_t ss = bcd2bin(Wire._I2C_READ() & 0x7F);
  uint8_t mm = bcd2bin(Wire._I2C_READ());
  uint8_t hh = bcd2bin(Wire._I2C_READ());
  Wire._I2C_READ();
  uint8_t d = bcd2bin(Wire._I2C_READ());
  uint8_t m = bcd2bin(Wire._I2C_READ());
  uint16_t y = bcd2bin(Wire._I2C_READ()) + 2000;
  return DateTime (y, m, d, hh, mm, ss);
}
Ds3231SqwPinMode RTC_DS3231::readSqwPinMode() {
  int mode;

  Wire.beginTransmission(DS3231_ADDRESS);
  Wire._I2C_WRITE(DS3231_CONTROL);
  Wire.endTransmission();
  Wire.requestFrom((uint8_t)DS3231_ADDRESS, (uint8_t)1);
  mode = Wire._I2C_READ();
  mode &= 0x93;
  return static_cast<Ds3231SqwPinMode>(mode);
}
void RTC_DS3231::writeSqwPinMode(Ds3231SqwPinMode mode) {
  uint8_t ctrl;
  ctrl = read_i2c_register(DS3231_ADDRESS, DS3231_CONTROL);
  ctrl &= ~0x04; // turn off INTCON
  ctrl &= ~0x18; // set freq bits to 0
  if (mode == DS3231_OFF) {
    ctrl |= 0x04; // turn on INTCN
  } else {
    ctrl |= mode;
  }
  write_i2c_register(DS3231_ADDRESS, DS3231_CONTROL, ctrl);
}
```

RTClib.h

```
#ifndef _RTCLIB_H_
#define _RTCLIB_H_
#include <Arduino.h>
class TimeSpan;
#define PCF8523_ADDRESS       0x68
#define PCF8523_CLKOUTCONTROL 0x0F
```

```
#define PCF8523_CONTROL_3     0x02
#define DS1307_ADDRESS  0x68
#define DS1307_CONTROL  0x07
#define DS1307_NVRAM    0x08
#define DS3231_ADDRESS  0x68
#define DS3231_CONTROL  0x0E
#define DS3231_STATUSREG 0x0F
#define SECONDS_PER_DAY 86400L
#define SECONDS_FROM_1970_TO_2000 946684800
class DateTime {
public:
   DateTime (uint32_t t =0);
   DateTime (uint16_t year, uint8_t month, uint8_t day,
            uint8_t hour =0, uint8_t min =0, uint8_t sec =0);
   DateTime (const DateTime& copy);
   DateTime (const char* date, const char* time);
   DateTime (const __FlashStringHelper* date, const __FlashStringHelper* time);
   uint16_t year() const       { return 2000 + yOff; }
   uint8_t month() const       { return m; }
   uint8_t day() const         { return d; }
   uint8_t hour() const        { return hh; }
   uint8_t minute() const      { return mm; }
   uint8_t second() const      { return ss; }
   uint8_t dayOfTheWeek() const;
   long secondstime() const;
   uint32_t unixtime(void) const;
   DateTime operator+(const TimeSpan& span);
   DateTime operator-(const TimeSpan& span);
   TimeSpan operator-(const DateTime& right);
protected:
   uint8_t yOff, m, d, hh, mm, ss;
};
class TimeSpan {
public:
   TimeSpan (int32_t seconds = 0);
   TimeSpan (int16_t days, int8_t hours, int8_t minutes, int8_t seconds);
   TimeSpan (const TimeSpan& copy);
   int16_t days() const         { return _seconds / 86400L; }
   int8_t  hours() const        { return _seconds / 3600 % 24; }
   int8_t  minutes() const      { return _seconds / 60 % 60; }
   int8_t  seconds() const      { return _seconds % 60; }
   int32_t totalseconds() const { return _seconds; }
   TimeSpan operator+(const TimeSpan& right);
   TimeSpan operator-(const TimeSpan& right);
protected:
   int32_t _seconds;
};
enum Ds1307SqwPinMode { OFF = 0x00, ON = 0x80, SquareWave1HZ = 0x10, SquareWave4kHz =
0x11, SquareWave8kHz = 0x12, SquareWave32kHz = 0x13 };
class RTC_DS1307 {
public:
   boolean begin(void);
   static void adjust(const DateTime& dt);
   uint8_t isrunning(void);
   static DateTime now();
   static Ds1307SqwPinMode readSqwPinMode();
   static void writeSqwPinMode(Ds1307SqwPinMode mode);
   uint8_t readnvram(uint8_t address);
   void readnvram(uint8_t* buf, uint8_t size, uint8_t address);
   void writenvram(uint8_t address, uint8_t data);
   void writenvram(uint8_t address, uint8_t* buf, uint8_t size);
};
enum Ds3231SqwPinMode { DS3231_OFF = 0x01, DS3231_SquareWave1Hz = 0x00,
DS3231_SquareWave1kHz = 0x08, DS3231_SquareWave4kHz = 0x10, DS3231_SquareWave8kHz
= 0x18 };
class RTC_DS3231 {
public:
   boolean begin(void);
   static void adjust(const DateTime& dt);
   bool lostPower(void);
   static DateTime now();
   static Ds3231SqwPinMode readSqwPinMode();
   static void writeSqwPinMode(Ds3231SqwPinMode mode);
};
enum Pcf8523SqwPinMode { PCF8523_OFF = 7, PCF8523_SquareWave1HZ = 6,
PCF8523_SquareWave32HZ = 5, PCF8523_SquareWave1kHz = 4, PCF8523_SquareWave4kHz =
3, PCF8523_SquareWave8kHz = 2, PCF8523_SquareWave16kHz = 1,
PCF8523_SquareWave32kHz = 0 };
class RTC_PCF8523 {
public:
   boolean begin(void);
   void adjust(const DateTime& dt);
   boolean initialized(void);
   static DateTime now();
   Pcf8523SqwPinMode readSqwPinMode();
   void writeSqwPinMode(Pcf8523SqwPinMode mode);
};
class RTC_Millis {
public:
   static void begin(const DateTime& dt) { adjust(dt); }
   static void adjust(const DateTime& dt);
   static DateTime now();
protected:
   static long offset;
};
#endif // _RTCLIB_H_
```