Sudan University of Science and Technology

College of Engineering

School of Electrical and Nuclear Engineering

CNC Automatic Farming System

A Project Submitted In Partial Fulfillment for the Requirements of the Degree of B.Sc. (Honor) In Electrical Engineering

Prepared By:

1. **Abdalla Ibrahim Saeed Adam**
2. **Almubarak Mohammed El-fadul Abd El-karim**
3. **Al khader Abd Alwhab Al khader Mohammed**

Supervised By:

Ust. Gaffar Babiker Osman Alamath

November 2020

بسم الله الرحمن الرحيم

(خَلَقَ السَّمَاوَاتِ بِغَيْرِ عَمَدٍ تَرَوْنَهَا وَأَلْقَى فِي الْأَرْضِ رَوَاسِيَ أَنْ تَمِيدَ بِكُمْ وَبَثَّ فِيهَا مِنْ كُلِّ دَابَّةٍ وَأَنْزَلْنَا مِنَ السَّمَاءِ مَاءً فَأَنْبَتْنَا فِيهَا مِنْ كُلِّ زَوْجٍ كَرِيمٍ)

سورة لقمان – الاية (10)

DEDICATION

We would like to dedicate this important research to our families and those who stand with us and motivated us toward the accomplishment of this important and hard work, especially my mother and father who were always helping us and wishing us the best (may ALLAH bless them).

Last but means not least, dedication goes to all those teachers in faculty of Electrical Engineering (SUST) who helped us during these years.

ACKNOWLEDGEMENTS

This research is by far the most significant accomplishment in our life and it would be impossible without the help of God “ALLAH” who helped us.

First of all and always we thank our God "Allah" to make us live these moments.

We would like to extend our gratitude and our sincere thanks to our honorable, esteemed supervisor and father Ms. **Gaffar Babiker** , staff of Department of Electrical Engineering. He is a great lecturer with deep vision and also most importantly a kind person. We sincerely thank him for exemplary guidance and encouragement. His trust and support inspired us in the most important moments of making right decisions and we are glad to work under his supervision.

ABSTRACT

This research concentrate on solve the difficultness of following farm process especially for people with no farming background, to make it easier and more enjoyable alongside reducing the amount of water wasted. As it known sudan is agricultural country so by solving this problem we can increase local production and saturate the market with our agricultural products.

This research is used Computer Numerical Contol system (CNC) as reference of its work, In the way its move and process information. So for that a three simulation programs are used for three different purposes.

At the end a system with ability to plant seeds in a an equal divided areas where every seed get its required space, in addition to water each seed in a scheduled way with the minimum needed amount of water

المستخلص

يركز هذا البحث على التخلص من الصعوبات المتعلقة بمتابعة عملية الزراعة خاصة للاشخاص قليلي الخبرة بالزراعة ،لجعلها لهم اكثر سهولة ومتعة .

وكما هو معروف ان السودان بلد زراعي في المقام الاول لذلك من خلال حل هذه المشكلة وادخال تقنيات حديثة في عملية الزراعة سيسهم ذلك في زيادة الناتج المحلي وتشبع الاسواق بالمنتجات الزراعية المحلية ذات التكلفة القليلة والجودة العالية .

في هذا البحث تم اتخاذ مبدأ عمل ماكينات الحاسبات ذات التحكم الرقمي (CNC) كمرجع لنظام عمل المشروع الى جانب اقتباس طريقة الحركة و طريقة معالجة المعلومات لتلك الماكينات . لذلك تم استخدام ثلاث برامج محاكاة لثلاثة اغراض مختلفة .

وفي النهاية تم انشاء نظام له القدرة على القيام بعملية زرع البذور في منطقة مقسمة لمساحات متساوية حيث تحصل كل بذرة على مساحتها المطلوبة ، بالاضافة الى سقي كل بذرة بطريقة مجدولة بأقل كمية مياه مطلوبة .

Table of content

LIST OF FIGURE

LIST OF TABLES

LIST OF ABBREVIATIONS

CNC	COMPUTER numerical control
NC	numerical control
IOT	internet of things
MCU	machine control unit
ICT	information communication technologies
SFT	smart farming technologies
UAV	unmanned aerial vehicles
DSS	decision support system

CHAPTER ONE

INTRODUCTION

Sudan is a first class agricultural country and everyone knows the importance of this feature and taking care of it and developing it to be innovative and easy to be accepted by everyone who has a passion for agriculture.

The operational cost of agricultural projects is considered one of the most important investment constraints facing the citizen and foreign investor, in addition to the poor production efficiency due to the old agricultural methods. Most of the agriculture in Sudan depends on rain irrigation, which in turn depends on the rainfall rate which varies from year to year.

The Sudanese economy depends heavily on agriculture, either agricultural crops or products whose industry depends on crops. The focus on agricultural increases production value of crops and reach them to the degree of saturation of the local market, thus reducing the price locally, which helps citizens in general and with the possibility of exporting surplus and access to foreign currency.

1.1 General Concepts:

Agriculture is one of the most important economic fields for which the State of Sudan is known, and therefore it is necessary to pay attention to it and develop it and ensure that it is carried out efficiently and with high productivity at the lowest possible cost. Hence the idea of assembling all the machines that came as an alternative to manpower to increase efficiency with one machine that does all these machines

1.2 Research Problems:

Difficult farming process and being a tedious and tiring routine that exhausts farmers and the high cost of agricultural by tractors and the difficulty of irrigation and dependence on rain-fed irrigation and poor productivity.

1.3 Research Objectives :

- Make farming easier and more enjoyable by simplifying the tasks required, and that can be done by using internet of things (**IOT)** technology.

-The possibility of following up all what goes on the place of agriculture and remote management, using Arduino.

-Combining all agricultural mec;ghanisms into one machine reduce cost.

-The process of monitoring and disposal of weeds automatically and with high accuracy.

-Reduce the amount of water wasted by calculating the right amount of water for each crop.

-The possibility of operating the system using solar energy.

1.4 Research Methodology:

The simulation program V-rep was used to simulate the way the project works and explain its parts, and the Proteus program were used to make accurate calculation and give order to the ARDUINO to control the motors and pumps to guide the movement arm and perform the tasks.

1.5 Project layout:

The project consists of an abstract **Chapter I** contains an general Concepts, reasearch problems, objectives, metdology,project layout. **Chapter II** contains introduction, Future of Farming irrigation, the applicable methods, the four basice irrigation methods, irrigation factors, selection of irrigation methods and system, system capacity requirements, soils, physical soils characteristics. **Chapter III** Introduction, Computer Numerical Control (CNC), CNC Coordinate Systems , Open-loop and Closed-loop , Control Systems,Motion Control Systems,Advantage and,Disadvantage of CNC. **Chapter IV** contains Introduction , Processing, V-Rep, Mechanical Design, Proteus, Electrical circuit, Operation System. **Chapter V**contains a Conclusion , Recommendations, references, appendix .

CHAPTER TWO

AUTOMATIC FARMING

2.1 Introduction

Farming is an occupation which is playing the ultimate role for survive of this world. It supplies maximum needs for the human being to live in this world. But in the advancement of the technologies with invention of Internet of Things, the Automation (Smarter technologies) is replacing the traditional methodologies which in cause resulting in wide range improvement of the Fields. Now we are in the state of automation where the up gradation of smarter technologies are improving day by day in maximum sectors starting from smart homes, garbage, vehicles, industries, Farming, health, grids and so on. In the field of Farming, the improvement with the implementation of Automation is also taking place with the invention of Internet of Things. The main idea of this chapter is focused on the review of the improvement in the Smart Farming sector.

Smart Farming represents the application of modern Information and Communication Technologies (ICT) into agriculture, leading to what can be called a Third Green Revolution. Following the plant breeding and genetics revolutions, this Third Green Revolution is taking over the agricultural world based upon the combined application of ICT solutions such as precision equipment, the Internet of Things (IOT), sensors and actuators, geo-positioning systems, Big Data, Unmanned Aerial Vehicles (UAVs, drones), robotics, etc. Smart Farming has a real potential to deliver a more productive and sustainable agricultural production, based on a more precise and resource-efficient approach. However, while in the USA possibly up to 80% of farmers use some kind of **SFT**, in Europe it is no more than 24%.From the farmer's point of view, Smart

Farming should provide the farmer with added value in the form of better decision making or more efficient exploitation operations and management. In this sense, smart farming is strongly related, to three interconnected technology fields addressed by Smart AKIS Network:

Management Information Systems: Planned systems for collecting, processing, storing, and disseminating data in the form needed to carry out a farm's operations and functions. Precision Agriculture: Management of spatial and temporal variability to improve economic returns following the use of inputs and reduce environmental impact. It includes Decision Support Systems (DSS) for whole farm management with the goal of optimizing returns on inputs while preserving resources, enabled by the widespread use of GPS, GNSS, aerial images by drones and the latest generation of hyper spectral images provided by Sentinel satellites, allowing the creation of maps of the spatial variability of as many variables as can be measured (e.g. crop yield, terrain features/topography, organic matter content, moisture levels, nitrogen levels, etc.). Agricultural automation and robotics: The process of applying robotics, automatic control and artificial intelligence techniques at all levels of agricultural production, including farm-bots and farm-drones. Smart Farming applications do not target only large, conventional farming exploitations, but could also be new levers to boost other common or growing trends in agricultural exploitations, such as family farming (small or complex spaces, specific cultures and/or cattle, preservation of high quality or particular varieties,…), organic farming, and enhance a very respected and transparent farming according to European consumer, society and market consciousness.

Smart Farming can also provide great benefits in terms of environmental issues, for example, through more efficient use of water, or optimization of treatments and inputs. The farming industry will become arguably more important than ever before in the next few decades. The world will need to produce 70% more food in 2050 than it did in 2006 in order to feed the growing population of the Earth, according to the UN Food and Agriculture Organization.

To meet this demand, farmers and agricultural companies are turning to the Internet of Things for analytics and greater production capabilities. Technological innovation in farming is nothing new. Handheld tools were the standards hundreds of years ago, and then the Industrial Revolution brought about the cotton gin. The 1800s brought about grain elevators, chemical fertilizers, and the first gas powered tractor. Fast forward to the late 1900s, when farmers start using satellites to plan their work.

Figure (2.1) shows IOT device shipments:

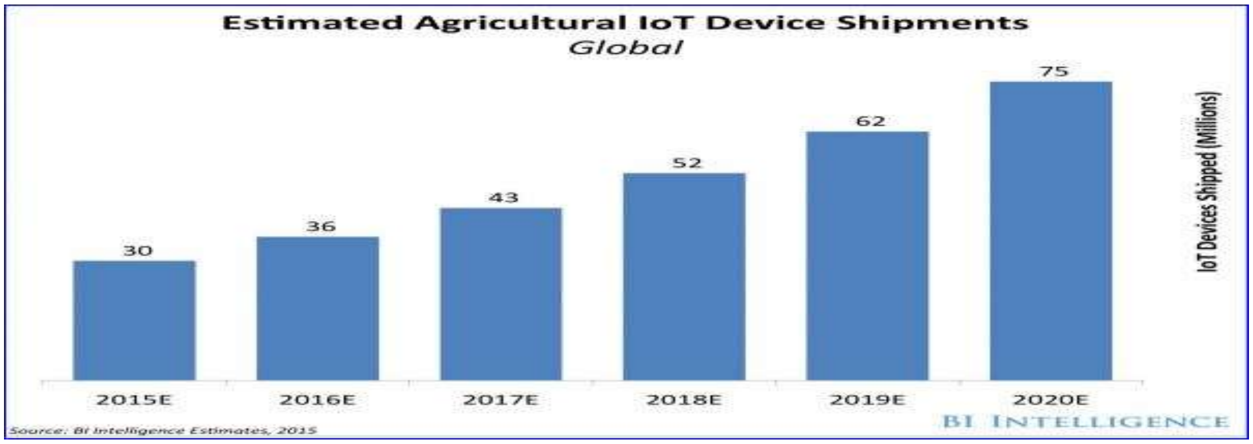


Figure (2.1) Estimated Agricultural IOT Shipments

The IOT is set to push the future of farming to the next level. Smart agriculture is already becoming more commonplace among farmers, and high tech farming is quickly becoming the standard thanks to agricultural drones and sensors. Below, we've outlined IOT applications in agriculture and how "Internet of Things farming" will help farmers meet the world's food demands in the coming years.

High Tech Farming: Precision Farming & Smart Agriculture, Farmers have already begun employing some high tech farming techniques and technologies in order to improve the efficiency of their day-to-day work. For example, sensors placed in fields allow farmers to obtain detailed maps of both the topography and resources in the area, as well as variables such as acidity and temperature of the soil. They can also access climate forecasts to predict weather patterns in the coming days and weeks. Farmers can use their smartphones to remotely monitor their equipment, crops, and livestock, as well as obtain stats on their livestock feeding and produce. They can even use this technology to run statistical predictions for their crops and livestock. And drones have become an invaluable tool for farmers to survey their lands and generate crop data.

As a concrete example, John Deere (one of the biggest names in farming equipment) has begun connecting its tractors to the Internet and has created a method to display data about farmers' crop yields. Similar to smart cars, the company is pioneering self-driving tractors, which would free up farmers to perform other tasks and further increase efficiency. All of these techniques help make up precision farming or precision agriculture, the process of using satellite imagery and other technology (such as sensors) to observe and record data with the goal of improving production output while minimizing cost and preserving resources.

2.2 Future of Farming

IOT, Agricultural Sensors, & Farming Drones. Smart agriculture and precision farming are taking off, but they could just be the precursors to even greater use of technology in the farming world. BI Intelligence, Business Insider's premium research service, predicts that IOT device installations in the agriculture world will increase up to 75 million in 2020, for a compound annual growth rate of

20%. The U.S. currently leads the world in IOT smart agriculture, as it produces 7,340kgs of cereal (e.g. wheat, rice, maize, barley, etc.) per hectare (2.5 acres) of farmland, compared to the global average of 3,851kgs of cereal per hectare. And this efficiency should only improve in the coming decades as farms become more connected. On Farm, which makes a connected farm IOT platform, expects the average farm to generate an average of 4.1 million data points per day in 2050, up from 190,000 in 2014.Furthermore, On farm ran several studies and discovered that for the average farm, yield rose by 1.75%, energy costs dropped $7 to $13 per acre, and water use for irrigation fell by 8%. Given all of the potential benefits of these IOT applications in agriculture, it's understandable that farmers are increasingly turning to agricultural drones and satellites for the future of farming.

2.3 The Future of Smart Farming

With **IOT** and Open Source Farming: Smart farming is a concept quickly catching on in the agricultural business. Offering high precision crop control, useful data collection, and automated farming techniques, there are clearly many advantages a networked farm has to offer Smart farming is a concept quickly catching on in the agricultural business. Offering high-precision crop control, useful data collection, and automated farming techniques, there are clearly many advantages a networked farm has to offer.

A recent Beecham's report entitled Towardsی Smart Farming: Agriculture Embracing the IOT Vision predicts that food production must increase by 70 percent in the year 2050 in order to meet our estimated world population of 9.6 billion people. It also describes growing concerns about farming in the future: climate change, limited arable land, and costs/availability of fossil fuels. So, what's the solution? Smart farming. Of the many advantages IOT brings to the table, its ability to innovate the landscape of current farming methods is absolutely groundbreaking. **IOT** sensors capable of providing farmers with information about crop yields, rainfall, pest infestation, and soil nutrition are invaluable to production and offer precise data which can be used to improve farming techniques over time. New hardware, like the corn-tending Robot, is making strides by pairing data-collecting software with robotics to fertilize the corn, apply seed cover-crops, and collect information in order to both maximize yields and minimize waste.

Another direction in which smart farming is headed involves intensively controlled indoor growing methods. The Open AG Initiative at MIT Media Lab uses "personal food computers" (small indoor farming environments that monitor/administrate specific growing environments) and an open source platform to collect and share data. The collected data is termed a "climate recipe" which can be downloaded to other personal food computers and used to reproduce climate variables such as carbon dioxide, air temperature, humidity, dissolved oxygen, potential hydrogen, electrical conductivity, and root-zone temperature.

This allows users very precise control to document, share, or recreate a specific environment for growing and removes the element of poor weather conditions and human error. It could also potentially allow farmers to induce drought or other abnormal conditions producing desirable traits in specific crops that wouldn't typically occur in nature With a future of efficient, data driven, highly-precise farming methods, it is definitely safe to call this type of farming smart. We can expect **IOT** will forever change the way we grow food. [4]

2.4 Irrigation

Irrigation application method and system selection should result in optimum use of available water. The selection should be based on a full awareness of management considerations, such as water source and cost, water quantity and quality, irrigation effects on the environment, energy availability and cost, farm equipment, product marketability, and capital for irrigation system installation, operation, and maintenance.

2.4.1 The applicable methods:

- Gross irrigation water needs Energy requirements

- Effects on quantity and quality of ground water and downstream surface water

- Installation and annual operating costs

- Labor skills needed

2.4.2 The four basic irrigation methods:

Along with the many systems to apply irrigation water include:

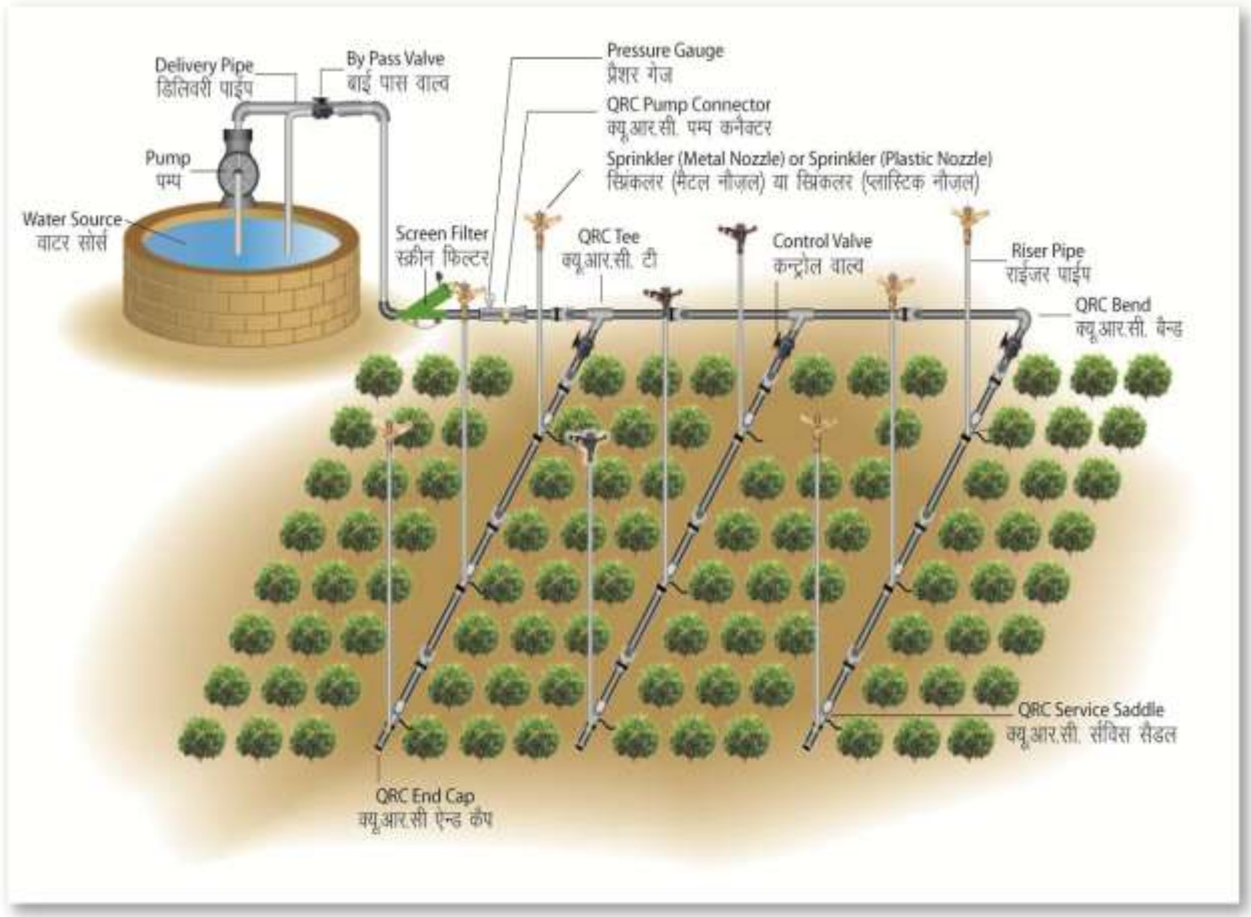
- Surface :

Water is applied by gravity across the soil surface by flooding or small channels. That is shown below in **Figure (2.2)**

Figure (2.2) Surface irrigation

- Sprinkle :

Water is applied at the point of use by a system of nozzles (impact and gear driven sprinkler or spray heads) with water delivered to the sprinkler heads by surface and buried pipelines, or by both. As shown in **Figure (2.3)**



Layout of Sprinkler Irrigation System (छिड़काव सिंचाई प्रणाली का रेखाचित्र)

Figure (2.3) Sprinkle irrigation

- Micro :

Water is applied to the point of use through low pressure, low volume discharge devices (i.e., drip emitters, line source emitters, micro spray and sprinkler heads, bubblers) supplied by small diameter surface or buried pipelines. **Figure (2.4)** shows micro irrigation.

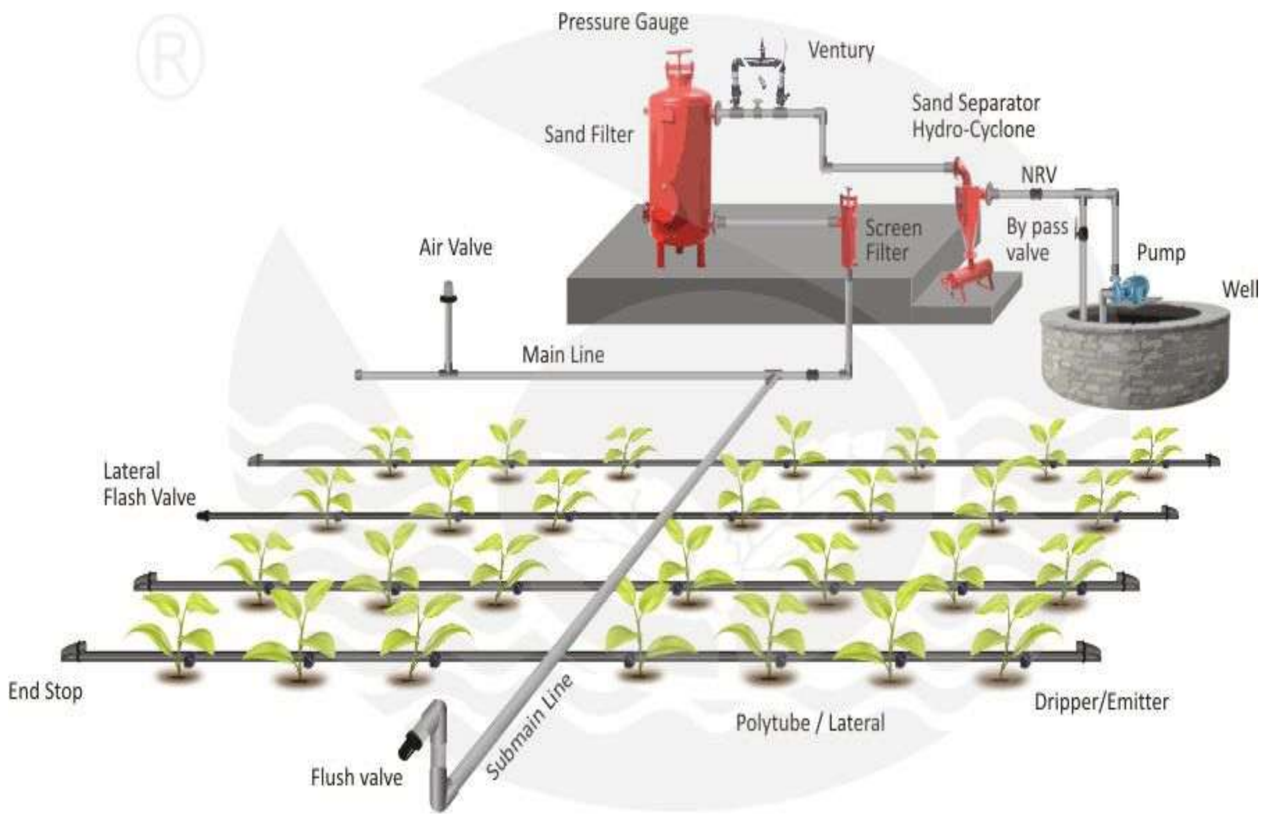


Figure (2.4) Micro irrigation

- **Subirrigation :**

Water is made available to the crop root system by upward capillary flow through the soil profile from a controlled water table.that can shown in **Figure (2.5)**

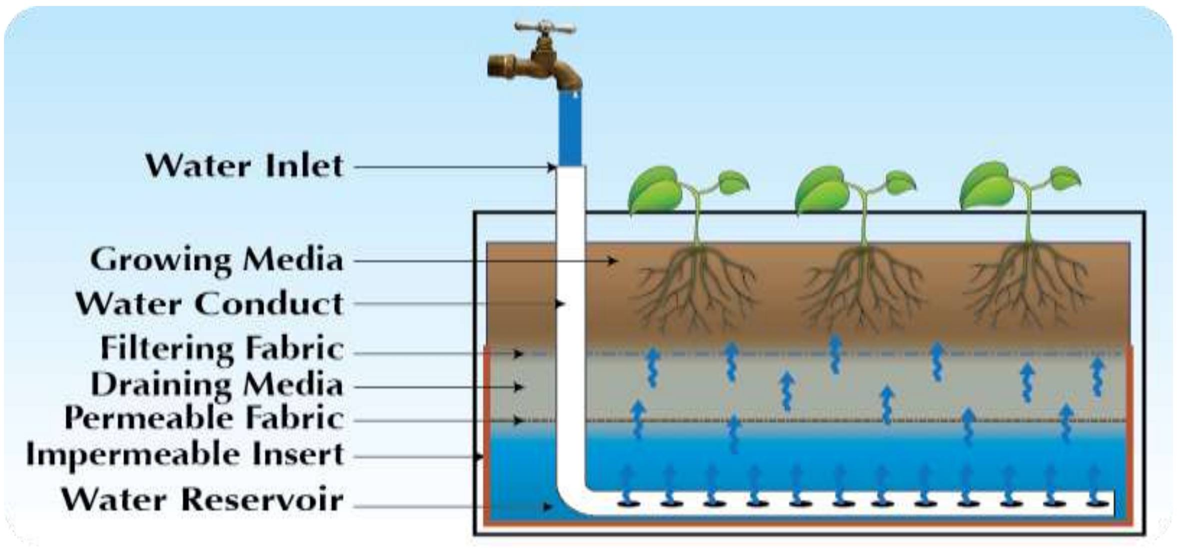


Figure (2.5) Sub irrigation

2.4.3 Irrigation Factors:

i. Crops to be grown
ii. Topography or physical site conditions
iii. Water supply
iv. Climate
v. Energy available
vi. Operation and management skills
vii. Environmental concerns
viii. Soils
ix. Farming equipment
x. Costs

2.4.6 Selection of irrigation method and system:

With the current demand for other uses of high quality water, the irrigation decision maker must provide good irrigation water management; including maximizing beneficial water use, providing good distribution uniformity, minimizing water losses, and using an appropriate irrigation scheduling method.

2.4.7 System capacity requirements:

The irrigation system must be able to deliver and apply the amount of water needed to meet the crop-water requirement.

The irrigation decision maker must decide what water supply rate(s) will be used for designing system capacity. In arid and semiarid areas with high value crops, at least 90 to 95 percent propability of peak daily plant evapotranspiration may be required. With medium value crops, 80 percent may be adequate; and with low value crops, 50 percent may be sufficient.

If the soil can only store and provide water for a few days, meeting peak daily evapotranspiration rates may be desirable.

- **Sprinkle irrigation systems**

systems is generally provided by pumping, powered by electric mot With the sprinkle irrigation method, water is applied at the point of use by a system of nozzles (impact sprinkler heads, spray nozzles, etc.) with water delivered by surface or buried pipelines. Sprinkler irrigation systems are classed by operation of the laterals. The three main types of sprinkle systems (laterals) are fixed, periodic move, and continuous/self move.

Pressure for sprinkler ors and diesel, natural gas, L P gas, or gasoline engines. Where sufficient elevation drop is available, sprinkler systems can be operated using gravity to provide the necessary operating pressure.

If the system is properly designed and operated, application efficiencies of 50 to 95 percent can be obtained. The efficiency depends on type of system, cultural practices, and management. Poor management (i.e., irrigating too soon or applying too much water) is the greatest cause of reduced water application efficiency when using sprinklers.

- **System losses are caused by:**

i. Direct evaporation in the air from the sprinkler spray, from the soil surface, and from plant leaves that intercept spray water.
ii. Wind drift (normally 5 to 10 percent depending on temperature, wind speed, and droplet size).
iii. Leaks and system drainage.
iv. Surface runoff and deep percolation resulting from, nonuniform application within the sprinkler pattern. If the system is designed to apply

water at less than the maximum soil infiltration rate, no runoff losses will occur.

With some systems where water is applied below or within the crop canopy, wind drift and most evaporation losses are reduced.

Soil surface storage is especially important where low pressure incanopy center pivot laterals are used. LEPA systems use complete soil, water, and plant management to prevent runoff.

The water infiltration process under sprinkler irrigation differs from that in surface irrigation. With surface methods, water is ponded on the surface. With sprinkle irrigation, water is applied so ponding does not occur or is only temporary.

- **Periodic move sprinkler irrigation systems :**

  A periodic move sprinkler irrigation system is set in a fixed location for a specified length of time to apply a required depth of water. The length of time in a position is called the length of set or irrigation set time. The lateral or sprinkler is then moved to the next set position.

- **Periodic move sprinkler irrigation systems:**

  i. Handmove laterals
  ii. Side (wheel) roll laterals
  iii. End-tow laterals
  iv. Hose fed (pull) laterals
  v. Gun type sprinkler
  vi. Boom sprinkler
  vii. Perforated pipe

- **Handmove laterals**

  This system is composed of portable pipelines with risers and sprinkler heads. Portable or buried mainline pipe with uniformly spaced valve outlets provides a water supply. Portable aluminum, or sometimes plastic, lateral pipe has quick couplers. Risers and sprinkler heads are either centermounted or end-mounted. Lateral sections are typically 20, 30, or 40 feet long. When the lateral has completed the last set location in the field, it must be dismantled and moved back across the field to the start position unless multiple laterals are used and the finish location is adjacent to the start location of the next set. Application efficiencies can be 60 to 75 percent with proper management. **Figure (2.6)** shows handmove laterals

Figure(2.6) Handmove laterals irrigation

- **Side (wheel) roll laterals**

  A side (wheel) roll system is similar to a handmove system except that wheels are mounted on the lateral. The lateral pipe serves as an axle to assist in moving the system sideways by rotation to the next set. The sections of the lateral pipe are semi-permanently bolted together. **Figure (2.7)** shows that

Figure(2.7) Side (wheel) roll laterals irrigation

2.5 Soils

Plant response to irrigation is influenced by the physical condition, fertility, and biological status of the soil. Soil condition, texture, structure, depth, organic matter, bulk density, salinity, sodality, acidity, drainage, topography, fertility, and chemical characteristics all determine the extent to which a plant root system grows into and uses available moisture and nutrients in the soil. Many of these factors directly influence the soils ability to store, infiltrate, or up flux water delivered by precipitation or irrigation (including water table control). The irrigation system(s) used should match all or most of these conditions. Soils to be irrigated must have adequate surface and subsurface drainage. Internal drainage within the crop root zone can be either natural or from an installed subsurface drainage system.

3.5.2 Physical soil characteristics

I. Soil properties and qualities

Soil properties and qualities are important in design, operation, and management of irrigation systems. These properties include the falwing:

i. Water holding capacity
ii. Soil intake characteristics
iii. Permeability
iv. Soil condition
v. Organic matter
vi. Slope
vii. Water table depth
viii. Soil erodibility
ix. Chemical properties
x. Salinity
xi. Sodality
xii. Soil reaction.

II. Soil-water holding capacity

The potential for a soil to hold water is important in designing and managing an irrigation system. Total water held by a soil is called water holding capacity.

III. Soil texture

Soil texture refers to the weight proportion of the soil separates (sand, silt, and clay) for the less than 2 mm fraction, as determined from a laboratory particle size distribution analysis. It defines the fineness or coarseness of a soil.

IV. Soil structure

Soil structure is the arrangement and organization of soil particles into natural units of aggregation. These units are separated from one another by weakness planes that persist through cycles of wetting and drying and cycles of freezing and thawing. Structure influences air and water movement, root development, and nutrient supply.

V. Soil bulk density

Refers to the weight of a unit volume dry soil, which includes the volume of solids and pore space.

VI. Soil pore space

Bulk density is used to convert water measurements from a weight basis to a volume basis that can be used for irrigation related calculations. Many tools are available to measure bulk density in the field as well as in the laboratory.

CHAPTER THREE

COMPUTER NUMERICAL CONTROL CNC

3.1 Introduction

The Numerical Control (NC) is a form of programmable automation in which the mechanical actions of a machine tool or other equipment are controlled by a program containing coded alphanumerical data. Also it can be defined as any machining process in which the operations are executed automatically in sequences as specified by the program that contains the information for the tool movements.

Any NC system consists of three basic components:

i. **Program Of Instruction:** The detailed step by step commands that direct the actions of the processing equipment. In machine tool applications, the program of instructions is called a part program, and the person who prepares the program is called a part programmer. In these applications, the individual commands refer to positions of the machine's tool relative to the worktable on which the work-part (is the object being processed) is fixture. Additional instructions are usually included to the program like tool selection, feed rate, spindle speed, and other functions. The program is coded on a suitable medium for submission to the machine control unit.

ii. **Machine Control Unit (MCU):** It is controller capable of reading and interpreting a stored program and using the instructions in this to control a machine via actuation devices. Consists of a microcomputer and related control hardware that stores the program of instructions as shown in **Figure(3.1)** and executes it by converting each command into mechanical actions of the processing equipment, one command at a time. The related hardware of the MCU includes components to interface with processing equipment and feedback control elements. The MCU also includes control system software, calculation algorithms, and translation software to convert the NC part program into a usable format for the MCU.

iii. **Processing Equipment:** Performs useful work and accomplishes the processing steps to transform the starting work-piece into a completed part. Its operation is directed by the MCU, which in turn is driven by instructions contained in the part program.

In the most common example of the processing equipment consists of the worktable and spindle as well as the motors and controls to drive them.

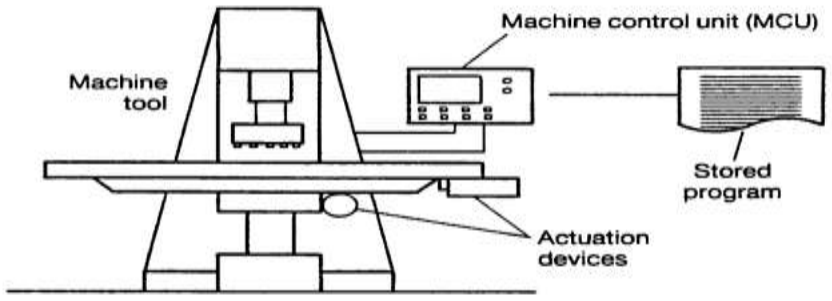


Figure (3.1) Numerical Control System Components

3.2 Computer Numerical Control (CNC)

Because the MCU is a computer, the term computer numerical control is used to distinguish this type of NC form its technological predecessors that were based entirely on a hard-wired electronics. Today, virtually all new MCUs are based on computer technology; hence, NC refers to CNC.

Computer numerical control (CNC) is defined as an NC system whose MCU is based on a dedicated microcomputer rather than on a hard-wired controller. The latest computer controllers for CNC feature high-speed processors, large memories, solid-state flash memory, improved servos, and bus architectures. Computer Numerical Control Machines are Electro Mechanical devices that can manipulate tools around a variable number of axis usually three or five axis with high precision per instruction from the computer program.

In most cases a CNC machine is programmed by a programmer using CAM software who uses the software to apply tool paths to a 3D or 2D model of the part they want to make. Once the programmer has finished his/her programming work, the program is fed into a post processor which will turn the CAM program into the G code the machine can read. The G code is uploaded into the CNC machine along with all the cutting tools needed.

CNC Software:

The computer in CNC operates by means of software. There are three types of software programs used in CNC systems:

i. **Operating System Software**: The principal function of the operating system software is to interpret the NC part programs and generate the corresponding control signals to drive the machine tool axes. It is installed by the controller manufacturer and is stored in ROM in the MCU.

ii. **Machine Interface Software:** Machine interface software is used to operate the communication link between the CPU and the machine tool to accomplish the CNC auxiliary functions.

iii. **Application Software:** Consists of the NC part programs that are written for machining or other applications in the user's plant.

Type of CNC Machines:

There are several types of it and it depends on the application it is used for but, the way the machine deal with the material makes the difference hence, there are:

i. Removal
ii. Additional

We get from the names the action that being done by the machine. In removal machine its works on eliminate or remove the material from the work-part by using different types of tools for different purposes. Most of this applications is:

Drilling, Milling, Turning, Cutting (plasma, high pressure water). The common operation feature of CNC in all of these applications is control of the Work-Head (machine tool) movement relative to the work-part. The application that uses the additional machine is 3D Printing.

3.3 CNC Coordinate Systems:

There are nine standard axis universally (straight line) movement (X,Y and Z) three primary rotary axis (A,B and C) are used to identify arc or circular movements such as a programmable turntable, lathe spindle or an articulating wrist action milling head (rotary motion). Finally we have three secondary straight-line motions called the auxiliary linear axis (U,V and W).

- **The Primary Linear Axis (X, Y, Z)** it is the basic axes used to define a three-space (three dimensional) envelope lie at 90 degrees to each other, and as such as called an orthogonal axis set (Three axes lying at mutual 90^0 angles). The limit of machine's axis travel is defined as the Work Envelope. Combining any two primary axis lines defines a flat plane. There are three planes X-Y, X-Z and Y-Z as shown in **Figure (3.2)** for example when viewing a part placed on a vertical milling machine, the table represents X-Y plane, while a lathe object is viewed in the X-Z plane usually from overhead.

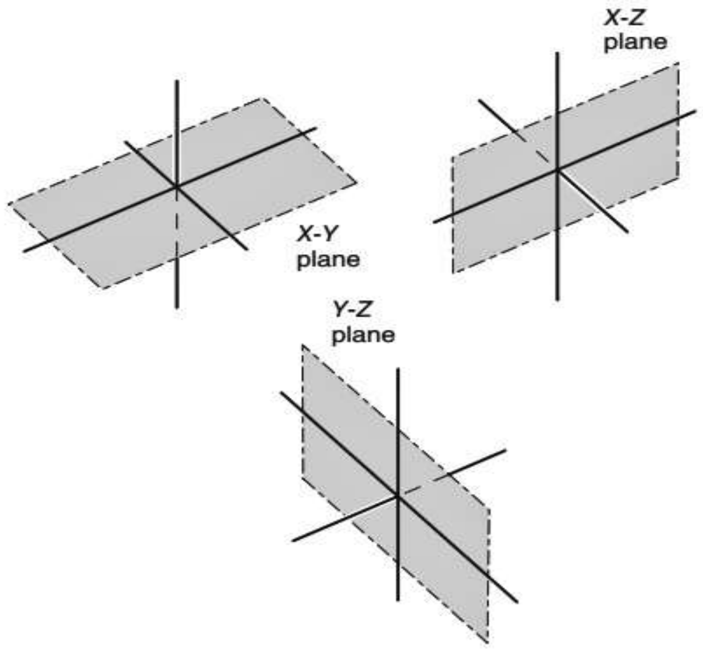


Figure (3.2) The Axis

When faced with an unfamiliar CNC machine, the *Z* axis it will be the easiest to identify. The *Z* axis carries the tool to the work, as with lathes, or the work to the spindle or vice versa, on mills. Then with the *Z* axis under control, apply the right hand rule to identify the other two axes, by pointing the thumb of the right hand along the positive *X* axis direction, first finger points out the positive *Y* axis direction. Finally, the raised middle finger points out the positive *Z* axis as **Figure (3.3).**

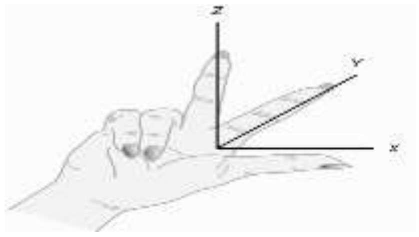


Figure (3.3) Axis Representing

- **The Primary Rotary Axis (A, B, C)** some CNC machines feature programmable axes that rotate or articulate. According to the **EIA267-B** standard, there are three primary rotary axes: A, B, and C. Each is identified by the central primary linear axis around which it pivots:-

  -A axis rotates around a line parallel to X

  -B axis rotates around a line parallel to Y

  -C axis rotates around a line parallel to Z

  To define which direction a rotary motion is to occur, clockwise or counter clockwise, it can be happened by using a plus or minus sign on the coordinate. Rule of thumb rotary axis sign value to identify whether the rotary axis direction is positive or negative (clockwise or counter clockwise). As shown in **Figure (3.4).**

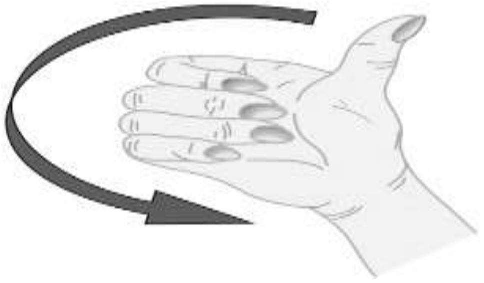


Figure (3.4) The Direction

These two primary axes define work mechanism of any CNC machine and play a big role in designing and programming the basics of the machine. The full relation between the two axes describe in **Figure (3.5).**

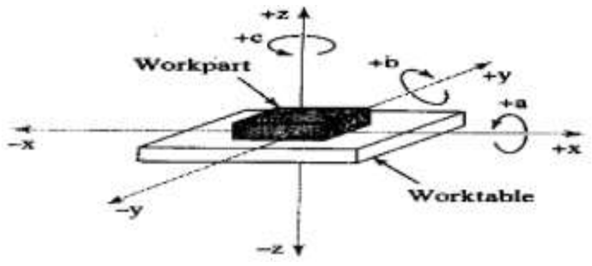


Figure (3.5) The Relation Between Axes

- **The Secondary Linear Axes (U,V,W)** CNC machines occasionally receive secondary, straight-line axes to add auxiliary tool slides or boring quills and other machining functions to their capability. To identify the secondary linear axes, determine the primary linear parallels (X, Y, or Z). If the secondary axis is parallel

  To :

  X, it is the U axis.

  Y, it is the V axis.

  Z, it is the W axis.

In machine tools the cutter may typically move in multiple directions with respect to the work-piece, or vice versa, and there the controller normally drives more than one machine axis. Examples of machine applications and numbers of axes are as follow:

- **2-axis** motion, generally in two orthogonal directions in a plane, which applied to most lathes as well as punch presses, flame and plasma-arc and cloth cutting machines, electronic component insertion and some drilling machines.
- **3-axis** motion, which is generally along the three principal directions (X,Y and Z) of the Cartesian coordinate system, and applying to milling, boring, drilling and coordinate measuring machines.
- **4-axis** motion typically involves three linear and one rotary axis, or perhaps two x-y motions, as for example for some lathes fitted with supplementary milling heads.

- **5-axis** machines normally involve three linear (x, y and z) axes, with rotation about two of these, normally x and y, and are generally milling machines.

To program the NC processing equipment, a standard axis system must be defined by which the position of the work-head relative to the work-part can be specified. There are two axis systems used in NC, one for flat and prismatic work-parts and the other for rotational parts. Both axis systems are based on the Cartesian coordinate system.

The axis system for flat and prismatic parts consists of three linear axes (x, y, z) in the Cartesian coordinate system, plus three rotational axes (a, b, c). In most machine tool applications, the x-and y-axes are used to move and position the worktable to which the part is attached, and the z-axis is used to control the vertical position of the cutting tool. The rotational axes can be used for:

- Orientation of the work-part to present different surfaces for machining.
- Orientation of the tool or work-head at some angle relative to the part.

The coordinate axes for a rotational NC system are associated with NC lathes and turning centers. Although the work rotates, this is not one of the controlled axes on most of these turning machines. Consequently, the y-axis is not used. The path of the cutting tool relative to the rotating work-piece is defined in the x-z plane, where the x-axis is the radial location of the tool, and the z-axis is parallel to the axis of rotation of the part.

- Zero point and Target point:

The part programmer must decide where the origin of the coordinate axis system should be located. This decision is usually based on programming convenience. For example the origin might be located at one of the corners of the part. If the work-part is symmetrical, the zero point might be most conveniently defined at the center of symmetry. Wherever the location, this zero point is communicated to the machine tool operator, at the beginning of the job, the operator must move the cutting tool under manual control to some target point on the worktable, where the tool can be easily accurately positioned. The target point has been previously referenced to the origin of the coordinate axis system by the part programmer. When the tool has been accurately positioned at the target point, the operator indicates to the MCU where the origin is located for the subsequent tool movements.

3.4 Open-loop and Closed-loop Control Systems:

Numerical control systems require Motors to control both position and velocity of the machine tool, and each axis must be separately driven. The control system can be implemented in two ways :

- Open loop system: **Figure (3.6)**

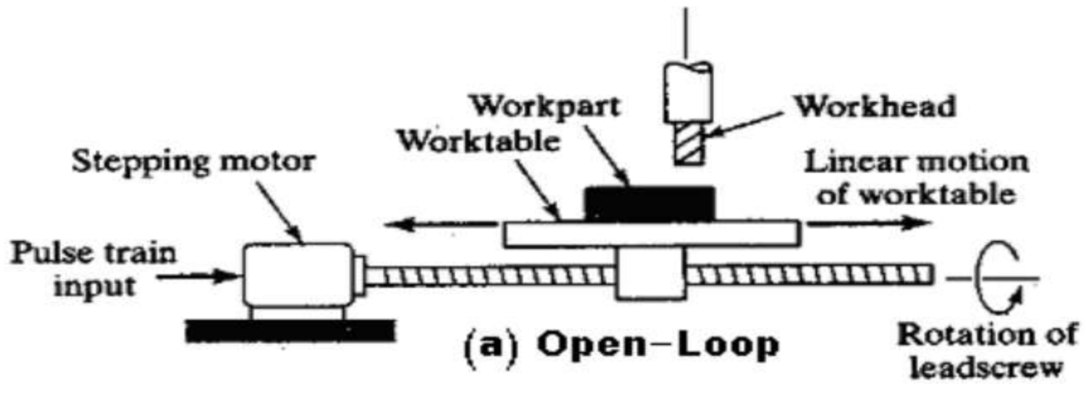


Figure (3.6) Open Loop

In this case Instructions are feed to the controller to Convert it to electrical pulses or signals Send to the Stepper motors. The number of electronic pulses determines the distance the frequency of the pulses determines the speed. Open loop control system is usually appropriate when the following conditions apply

– The actions performed by the control system are simple.

– The actuating function is very reliable

– Reaction forces opposing the actuator are small enough to have no effect on the actuation.

Advantages

– Less expensive

– Less complicated

Disadvantages

– Accuracy

– Repeatability

– Setup

- Closed loop system: **Figure (3.7)**

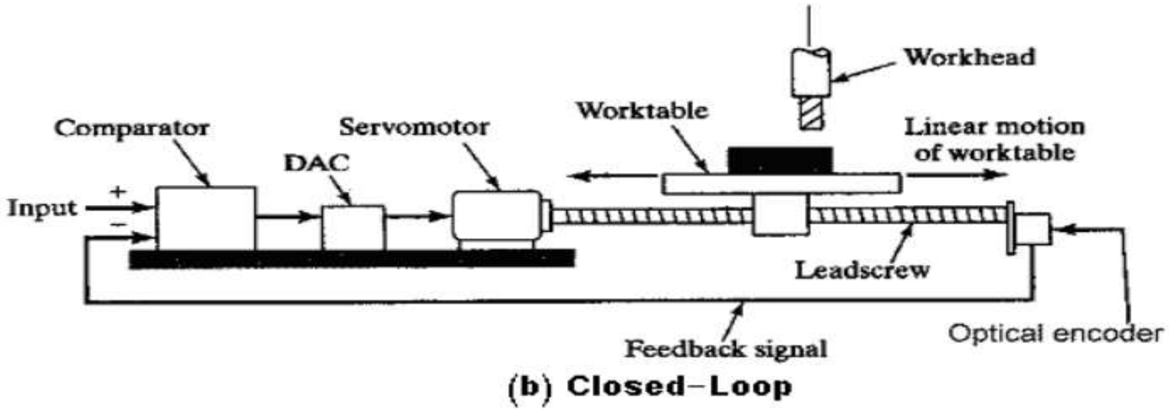


Figure (3.7) Closed Loop

Main difference from an open loop system is the inclusion of a feedback system in the controller. The feedback mechanism allows the machine to "know" where the tool is in regards to previous movements, Feedback may be analog or digital. Analog feedback Measures the variation of position and velocity in terms of voltage levels. Digital feedback Monitor output variations in the form of electrical pulses.[1]

3.5 Motion Control Systems:

Some NC processes are performed at discrete locations on the work-part (e.g., drilling and spot welding). Others are carried out while the work-head is moving (e.g turning and continuous welding). If the work-head is moving, it may be required to follow a straight line path or a circular or other curvilinear path. These different type movements are accomplished by the motion control system.

Motion control systems for NC can be divided into two types:

- Point-To-Point
- Continuous Path

- Point-to-point systems: also called positioning systems, move the worktable to a programmed location without regard for the path taken to get to that location (the path is not defined by the programmer). Once the move has been completed, some processing action is accomplished by the work-head at the location, such as drilling or punching a hole. Thus, the program consists of a series of point locations at which operations are performed **Figure (3.8).**

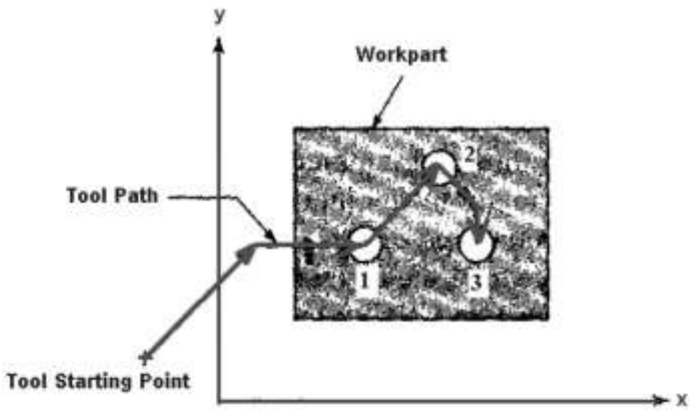


Figure (3.8) Positioning System

- Continuous path (Contouring): systems generally refer to systems that are capable of continuous simultaneous control of two or more axes

This provides control of the tool trajectory relative to the work-part. In this case, the tool performs the process while the worktable is moving, thus enabling the system to generate angular surfaces, two-dimensional curves, or three-dimensional contours in the work-part **Figure (3.9**).

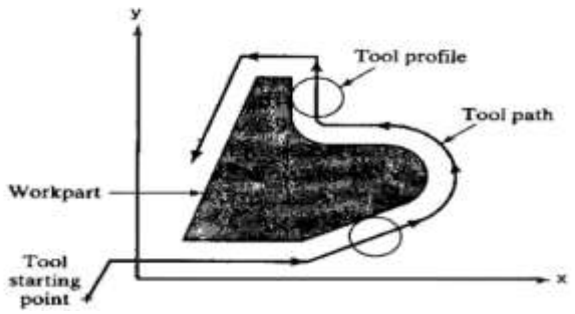


Figure (3.9) Continuous Path System

One of important aspects of the contouring is the interpolation. The paths that the contouring-type NC system is required to generate often consist of circular arcs and other smooth nonlinear shapes. Some of these shapes can be defined mathematically by relatively simple geometric formulas. Whereas other can't be mathematically defined except by approximation. A number of interpolation methods are available to deal with the various problems encountered in generating a smooth continuous path in contouring.

They include:

- **Linear interpolation:** This is the most basic method and is used when a straight line path is to be generated in continuous path NC. .The programmer specifies the beginning point and end point of the straight line and the feed rate to be used along the straight line. The interpolator computes the feed rates for each of the two (or three) axes to achieve the specified feed rate.

- **Circular interpolation:** This method permits programming of a circular arc by specifying the following parameters: the starting point, the endpoint, either the centre or radius of the arc, the direction of the cutter along the arc.
- **Helical interpolating:** This method combines the circular interpolation scheme for two axis described above with linear movement of a third axis.
- **Parabolic and cubic interpolation:** These routines provide approximations of free form curves using higher order equations.

The interpolation module in the MCU performs the calculations and directs the tool along the path. In CNC systems, the interpolator is generally accomplished by software. Linear and circular interpolators are almost always included in modern CNC systems.

Interpolation performs two functions:
– It computes individual drive axes to move the tool along a given path at a specified feed rate.
– It generates intermediate coordinates points along a program path.

- Motors:

In motion control it is very important to be sure the motor turns with a specific angle and/or speed. Thus, there are two motors are intended for motion control:

(i) Stepper Motor:

A step motor is a brushless dc electric motor that moves in precise angles called steps by converting a series of electrical pulses into rotatonal motion. They will not produce continuous motion from a continuous input voltage and it will stay at a particular position as long as the power is on. Step motors are controlled with the use of discrete electrical pulse signals. Each pulse will rotate the step motor shaft by a fixed angle called a step angle. Step motors have several different step angles to choose from (0.45, 0.9, 1.8).

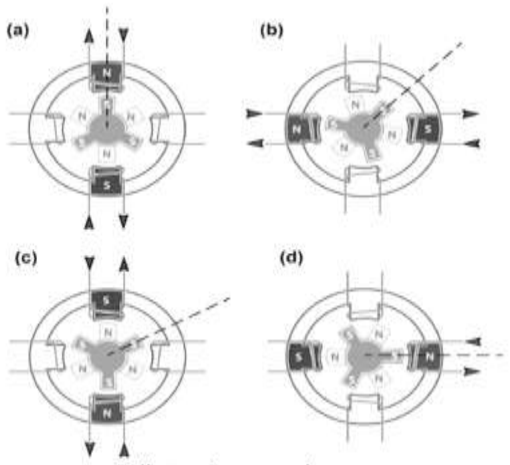
If the pulses are carried out in a specified sequence, the motor will spin continuously; the speed can be controlled by the rate at which the pulses are sent. These natural step angles allow a step motor to be accurately positioned without the accumulation of error. The step motor produces output torque from the interaction between the magnetic field in the rotor and in the stator. The magnetic field strength is proportional to the amount of current applied to the windings as well as the amount of turns in the windings.

Due to their simplicity and precision, steppers are popular in electrical devices. Analog clocks, manufacturing robots, and printers (2D and 3D) rely on steppers for motion control. An important advantage is that the controller doesn’t have to read the stepper’s position to determine its orientation. If the stepper is rated for 2.5°, each control signal will turn the rotor through an angle of 2.5°.

Modern steppers can be divided into three categories:

- **Permanent motor (PM)** High torque, poor angular resolution.
- **Variable reluctance (VR)** Excellent angular resolution, low torque.
- **Hybrid (HY)** Combines structure of PM and VR steppers, provides good torque and angular resolution.

When power is connected to the leads, a pulse of current is applied into the motor windings and turns “on” a phase, which causes the rotor to rotate until magnetic reluctance is minimized and the rotor settles into a stable magnetic position. For a bipolar motor, this pulse of current fills up the fist-phase coils with charge and creates a magnetic field for each of the 4 first-phase coils, which magnetizes the stator teeth in front of the charged coils. All four coils attract oppositely charged rotor teeth while repelling similarly charged rotor teeth on the opposite end of the rotor. The rotor teeth on the north and south ends of the rotor are offset slightly from each other, so one charged stator tooth can pull at an oppositely charged rotor tooth on one end of the rotor while repelling two similarly charged rotor teeth on the other end of the rotor until the motor settles into a detent position. This creates a “pushing and pulling” effect that increases the motor torque. The torque required to move the rotor from this stable position is called the holding torque. Turing one phase on will hold the rotor in one position, called a detent position. The charge is then turned off in the first set of coils, and turned on in the next set of coils, magnetizing the next set of stator teeth. The rotor then rotates until the next rotor tooth aligns with one of the magnetized stator, and this known as Full stepping **Figure (3.10).**



Figure(3.10) Stepping

Half stepping happens when the first pulse magnetizes the first phase, then the next pulse magnetizes the first phase and the second phase, then the next pulse magnetizes only the second phase, etc. This allows the rotor to move in smaller increments at a time, allowing for smoother and more continuous motion.

ADVANTAGES OF A STEP MOTOR:

- High reliability with little maintenance
- Open loop operation- No feedback is required for position or speed control
- Non-cumulative positional error
- inherently more fail safe than servo controlled motors

(ii) Servomotors:

This focuses on a second type of motor intended for motion control the servomotor, servo. Whereas steppers rotate through an angle and halt, many servos rotate continuously. Properly controlled, a servo can do everything a stepper can and more. In addition to setting the rotation angle, the controller can configure the servo's rotational speed and acceleration. This added control is the main advantage of servos over steppers. The main disadvantage is that designing a servo controller is a difficult process. As explained in Chapter 4, "Stepper Motors," a stepper can be controlled with a simple sequence of discrete pulses. With servos, the control signals are more involved. The goal of this chapter is to explain why this is the case and to show how this control can be accomplished. It's important to note that the term *servomotor* doesn't imply anything about the motor's structure. A servo may be brushed or brushless, AC or DC. The fundamental difference between servomotors and other electric motors is the availability of position feedback. A servo sends a signal to the controller that identifies its rotation angle and/or rotation speed. The controller uses this feedback to determine what control signals to send. Unfortunately, most servos directed toward hobbyists don't provide feedback. I'll refer to these as *hobbyist servos*, and the first part of this chapter presents them in detail.

We will also look at encoders, which are systems that can be connected to motors to provide feedback. The rest of the chapter focuses on sevos that deliver feedback to the controller.

Designing a controller for this kind of servo requires mathematical modelling a model of the servo and a model of the controller's signals The benefit of this modelling is that the motor can be made to turn with incredible precision. The drawback is that the mathematical theory has a significant learning curve.

- Pulse Width Modulation (PWM) Control: The DC Motors can be controlled by sending pulse width modulation (PWM) signals.

As a quick review, PWM delivers pulses of varying width to the motor. **Figure (3.11)** provides a simple example.

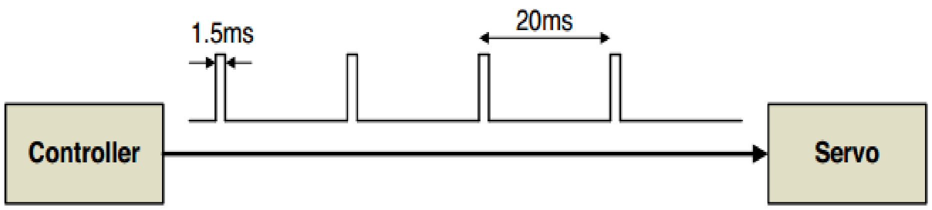


Figure (3.11) Pulse Width

According to the "pulse width range" parameter, the servo responds to pulses whose widths are between 0.7ms and 2.3ms. The neutral position is set when the pulse width is at 1.5ms, which is the pulse width illustrated in Figure 5.2. Similarly, a pulse width of 0.7ms turns the rotor to the full left position, and a pulse width of 2.3ms sets the rotor to the full right position. In theory, we'd like the servo to turn every time the pulse width changes. But in practice, the servo should ignore minor deviations in pulse width that may have been caused by noise. This is the meaning of the "dead bandwidth" parameter, which equals 0.005ms for the FS5106B. If the pulse width changes by less than 0.005ms, the servo won't move. When power is turned off, the servo's rotor stays in its position. This means that, when power is turned on again, the controller may not know the rotor's angle. For this reason, it's common to return the servo to the neutral position as soon as power becomes available.

- Analog and Digital Servos: According to its specification, the FS5106B is an analog servo instead of a digital servo. From the controller's perspective, there's no difference between the two both types are controlled by the same PWM signals. The difference involves the internal circuitry that receives control signals and delivers power to the motor.

When an analog servo receives pulses from the controller, it amplifies them and sends them to the motor to provide power. As the rotor approaches the desired position, the power diminishes to zero. If the widths of the incoming pulses are less than the servo's dead bandwidth (0.005ms in the case of the FS5106B), the motor won't receive any power . Digital servos operate in essentially the same way, but a digital servo has a microprocessor that receives pulses from the controller, processes them, and delivers pulses to the motor. The presence of the microprocessor provides three advantages:

- **Lower dead bandwidth** The processor can respond to pulses that are too small for an analog servo to notice.
- **Higher-frequency power** The pulses sent by the processor have a higher frequency than those sent by the controller. This makes digital servos more responsive than analog servos.

- **Programmability** The processor's operating characteristics can be configured by the user.

There are two significant disadvantages of using digital servos. The first is cost. Digital servos generally cost twice as much as comparable analog servos. The second disadvantage involves power. Because of its microprocessor, a digital servo requires more power than a comparable analog servo.

A minor disadvantage of digital servos relates to the dead bandwidth by default, a digital servo responds to small changes in pulse width that an analog servo would ignore.

This can be a problem in noisy environments, draining current from the power supply. However, if the servo's dead bandwidth is suitably configured with a programmer, this won't be an issue.

3.6 Advantage and Disadvantage of CNC:

Advantage

i. It can produce jobs with the highest accuracy and precision than any other manual machine.
ii. It can be run for 24 hours a day.
iii. The parts produced by it have the same accuracy. There is no variation in the parts manufactured by a CNC machine.
iv. A highly skilled operator is not required to operate a CNC machine. A semi-skilled operator can also operate accurately and more precisely.
v. It has the capability to produce complex designs with high accuracy in minimum possible time.

vi. The modern design software, allows the designer to simulate the manufacturer of his/her idea. And this removes the need for making a prototype or model and saves time and money.

vii. Fewer workers are required to operate a CNC machine and save running cost.

Disadvantage

Despite of having so many advantages, a CNC machine has some disadvantages too. And these are:

i. The cost of the CNC machine is very high as compared with a manually operated machine.

ii. The parts of the CNC machines are expensive.

iii. The maintenance cost in the case of CNC is quite high.

iv. It does not eliminate the need for costly tools.[2]

CHAPTER FOUR
THE APPLICATION

4.1 Introduction

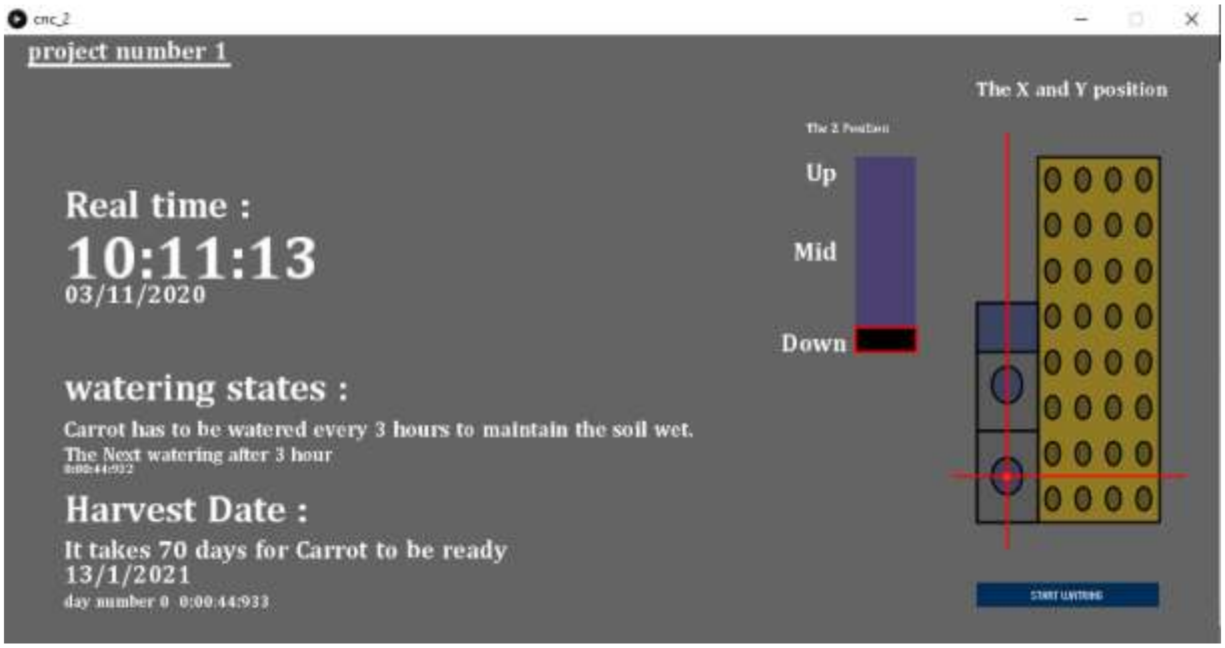
The research is based on automatic cultivation using machine without human intervention, so that there is a system that controls the process of planting and watering at the lowest costs, according to studied data of crops, methods of cultivation and harvest time. So that the system gives each crops the day daily need of water according to that data. Then the program controls the stepper motors that responsible of movement of the axes, thus change the location movement arm according to the process to be stimulated, whether it’s planting seeds to specific depth for each crop, or watering the seeds in known quantities for each crop, which reduces excess watering consumption.

4.2 Processing

Processing is an open-source graphical library and integrated development enviroment based on java language, built for the electronic arts, new media arts and visual design communities with the purpose of teaching non-programmers the fundamentals of computer programming in a visual context.

Any automatic system in the world certainly needs manager to manage the processes and give commands to other system elements to excute these commands, so in our project system we use Processing Software to play a role of the manager and to be as a user interface to show all data that the user may be need it to know what is running in the system.

Processing software has been programmed to manage all system processes whether were seeding or irrigation in addition to show the time remaining for the next irrigation operation and also the time remaining for the harvest, **Figure(4.1)** gives a clear vision of this interface.



project number 1

Real time :
10:11:13
03/11/2020

watering states :
Carrot has to be watered every 3 hours to maintain the soil wet.
The Next watering after 3 hour
0:00:44:932

Harvest Date :
It takes 70 days for Carrot to be ready
13/1/2021
day number 0 0:00:44:933

Figure (4.1) User Interface

The tasks performed by the program can be divided into:

- Points calculation.
- Send commands to the Arduino to do the processes.
- Determine the next watering time.
- Determine and display harvesting time.
- Follow the actual location of the axes.
- Alert the user with notification.

These administrative tasks require some information about the planted crop, because of that, a table was created that contains studied information about crops that can be planted using this system.

The conditions that must be provided in the crops that can be planted with this system are:

(i) The crops must be cultivable using seeds (because it’s the only way the system was designed to work).
(ii) The crop must be one time harvest type (this means that the plant is completely uprooted upon harvest). In case of multi-time harvest plants the plant will be left in the soil to increase its size until obstruct the movement of the axes.

In view of this, a **Table (4.1)** is was created containing data for some crops that can be planted using this system and which can be planted in the climate of Sudan:

Table (4.1) Data Table

CROPS	HARVEST PERIOD/ D	DEEP INCH	DISTANCE	WATER AMOUNT/DAY/PLANT
CARROT Short Half Cylindrical	70 65 75 68	1/2	1 inch	70 ml
TURNIP Amber De Milan Gold ball	50 75 35 45	1/2	1 inch	70 ml
RADISH Cherry belle Burpee Black Spanish	22-70 22 25 55	1/2	1 inch	70 ml
ONION Red White Green	90-120 110 100 110	1/2	4-6 inch	1.122 l

This information has been saved in the program in the form of variables whose value is determined when selecting the crop to be planted. [6]

Administrative Tasks:

Points calculation: it’s the most important task and has the highest priority in this program, in this task the pond area is divided evenly, given each plant the space it requires for proper growth. If the size of the pond is fixed to fixed dimensions at the level of two dimensions (2D) X,Y then the factor that affects he number of points (plants) in the ponds is the surrounding area the intent, which represents the area required for each plant.

It is represented by the diameter of this circuit (D). points are calculated using the following equations:

$\mathbf{X_p = X/D}$

$\mathbf{Y_p = Y/D}$

$\mathbf{P_{o=} X_{p\,*} Y_p}$

$\mathbf{X_p}$**:** the number of points in X axis

$\mathbf{Y_p}$**:** the number of points in Y axis

$\mathbf{P_o}$**:** the number of total points

D: diameter of plant circuit

This task has been programmed in the program so that the user selects the crop to be planted thus the variable D is determined so the points and their actual location are calculated in the form of coordinates. These coordinates are very important in directing and sending the correct instruction to move the motor to correct location.

- Send commands to the ARDUINO:

The ARDUINO is designed and programmed to control the motors and pumps in a way that is limited to electric power supply for operation and stopping only, without the ability to calculate the location and coordinates of the plant, or the irrigation and harvest time. These calculation are made by Processing program and inform the ARDUINO to the exact coordinates and follow the tasks to be performed. The method of communication used is the G-Code and is known as all CNC control system. The following **table (4.2)** shows these orders:

Table (4.2) G-code

G-Code	Meaning
X*	Move the motors in X axis
Y*	Move the motors in Y axis
XY	Move the motors in X &Y axes
M	Move the movement arm in Z axis **MU** for the first location **MW** for the second location **MD** for the third location
T1	**T1N** to carry watering tool **T1F** to put watering tool
T2	**T2N** to carry seeding tool **T2F** to put seeding tool
SH	Go to the seed holder
	Turn on the watering pump for specified time
A	**A1** turn on the air pump **A0** turned off the air pump
W	Turn on the watering pump for Specified time

These orders are send in two cases to perform a sequence of tasks, so that they can placed in one frame work namely:

Seeding Process: in this process the seed is taken and placed in its exact location, therefore the instructions are sent, for example we 4 points if the sent command will be like this:

T2N SH XY 2.5,5 MD A0 MU SH XY 7.5,5 MD A0 MU SH XY 2.5,15 MD A0 MU SH XY 7.5,15 MD A0MU T2F XY 0,0

This message is being followed up in the following post , the event is in the end of the seeding process for each points

Water Process: it's the same principle of work of seeding process with different orders, for example in the previous example in the case of 4 points, orders are following:

T1N MW XY 2.5,5 W2000 XY 7.5,5 W2000 XY 2.5,15 W2000 XY 7.5,15 W2000 MU T1F XY 0,0

Watering Time Calculation: this task mainly depends on the data that is described in **table (4.1).** And the system programming is that the watering is once per day in the case of crops that required a small amount of water and twice a day for crops that require large amount of water such as onion.

Pump opening time calculation:

$T= Q / P_f$

T: Pump opening time

P_f : Water flow rate it's constant = 0.27778 L/Sec

Q: Amount of water need per day per plant

- Determine and display harvesting time:

It's obtained from data table and set stop watch to calculate the remaining time to the harvesting and send notification when the specified time is end, also display it on main interface.

- Follow the actual location of the axes:

It is not more illustration process and limited to given illustration information about the axes movement to the user. This stage programmed the Arduino to calculate steps that moved by motors and convert that value to centimeter distance and sent it to Processing program and it shows it clearly to the axes site in the user interface.

- Alert the user with notification:

The Processing program has ability to send and display notification in the way that attract attention, and this feature was used twice for two different cases:

- Warring the user about putting water in the tank. When the water drops to a critical level impedes the work of the pumps in the tank, the ARDUINO sends a message explaining the warring. This message is shown as an alert in Processing program to the user.
- A notice that is send when the planting periods end and harvesting time is come which is done manually. This feature is programmed by adding specialized library for it in the code and adding call condition.

4.3 V-Rep:

The robot simulator CoppeliaSim, with integrated environment, is based on distributed control architecture: each object/model can be individually controlled via an embedded script, a plugin, a ROS or blue Zero node, a remote API client, or a custom solution. This makes coppeliaSim very versatile and ideal for multi-robot applications. Controllers can be written in C/ C++, Python, Java, Lua, Matlab or Octave.

CoppeliaSim is used for fast algorithm development factory automation, fast prototyping and verification robotics related education remote monitoring safety double-checking, as digital twin, and much more. You can find a feature overview here.

4.3.1 Mechanical Design:

This aspect of the project has a great impact on the functions performed by the system. This aspect is the concerned with converting ideas into actions that exist on the ground. The mechanical design shows whether this function can be done or not.

The basic mechanical functions of the project:

(i) Seeding
(ii) Watering

Knowing that the tasks are not limited to the aforementioned only, but there is possibility to add other tasks. It just needs more research such as:

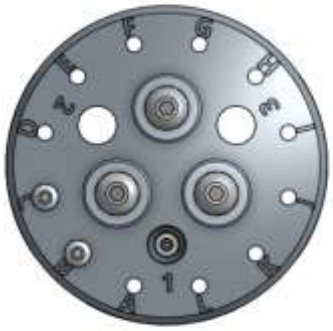
(i) Harvesting
(ii) Eliminating weeds
(iii) Spraying pesticides

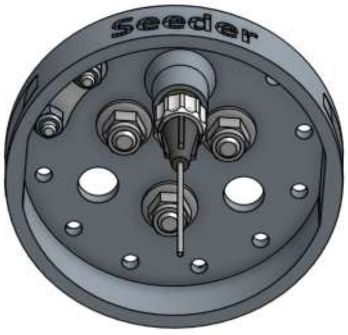
The mechanical system works in three dimension X, Y, Z, so the two dimension X and Y are specific to determining the coordinates of the points. As for Z dimension concerned with the work performance of planting seeds in the soil at a specific and studied depth depending on the type of crop, and it also changes tools and controls the spray area.

Due to the variety of functions that can be performed, it was designed to have a head that has the ability to change tools to perform different functions. This was done by adding an electromagnet to the head so that a high magnetic force passes in its circuit is sufficient to carry a lightweight tool made of plastic and has three vertical iron nails embedded in it. The tool is taken by moving the axes to fixed points in the system and that knows that they are specific points for the certain tool such as seeding process X, Y axes moves into coordinates X, Y then Z axis moves into a known high which leads to petition screws in the seeding tool of the electric magnet in the common head, then the electric circuit is connected in the common head which will activate the magnet and pick up the common. The disconnection is occur when the electrical is open.

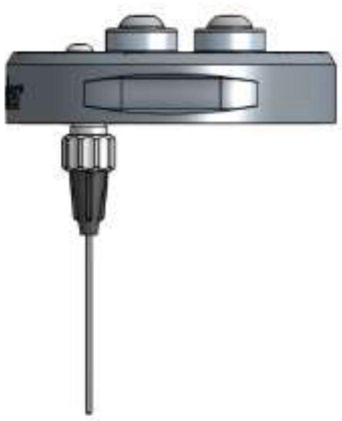
Seeding tool:

The purpose of this tool is to have the ability to carry the seeds and facilitate their transport to the site to be planted in. the design was made that the tool should be a needle with a diameter smaller than the diameter of seed. This needle is connected via pneumatic hose to an air pump. The seed is so that the seed is suctioned and is fixed at the edge of the needle , because the diameter of the needle is much smaller than the diameter of the seed and it is easy to transport it to the specific point, the needle enters the a measured depth into the soil and the seed is left by stopping the air pump. The following **Figure (4.2)** shows the design of seeding tool.





Upper Side Down Side



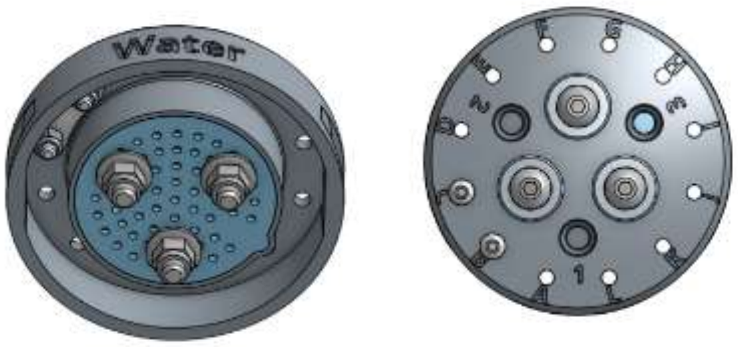
Front Side

Figure (4.2) Seeding Tool

Watering tool:

This one work in the same principle with difference in the design due to there are no needle here but the tool has a multi hole cover to add ability of spray the coming water from the pumps through the water hose connected to predetermined hole in the head that is parallel to hole number three in the tool.

Figure (4.3) shows the horizontal and bottom projection of the watering tool



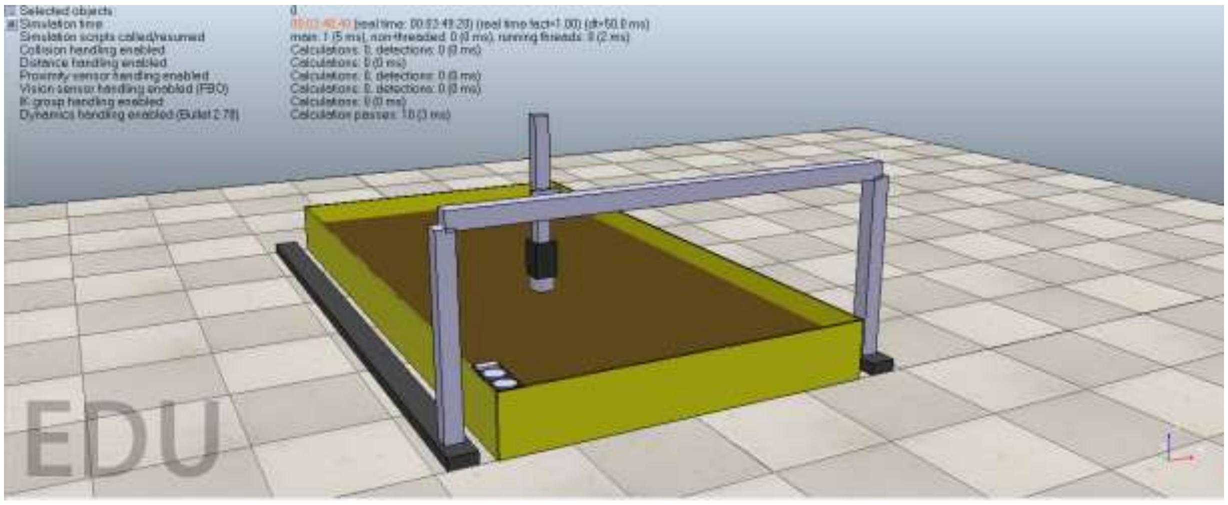
Down Side   Upper Side

Figure (4.3) Watering tool

Axes Movement:

To have a movement in a multi axes two stepper motor specialized in Y axis movement because the Y axis movement requires all vertical and horizontal columns to move so it requires more toque on stepper motor for X axis movement and another one for the Z axis movement.

Figure (4.4) shows the mechanical design that is done on V-Rep program



```
Selected objects                              0
Simulation time                               00:02:48.40 (real time: 00:53:49.20) (real time fact=1.00) (dt=50.0 ms)
Simulation scripts called/resumed             main: 1 (5 ms), non-threaded: 0 (0 ms), running threads: 0 (2 ms)
Collision handling enabled                    Calculations: 0, detections: 0 (0 ms)
Distance handling enabled                     Calculations: 0 (0 ms)
Proximity sensor handling enabled             Calculations: 0, detections: 0 (0 ms)
Vision sensor handling enabled (FBO)          Calculations: 0, detections: 0 (0 ms)
IK group handling enabled                     Calculations: 0 (0 ms)
Dynamics handling enabled (Bullet 2.78)       Calculation passes: 10 (3 ms)
```

Figure (4.4) Mechanical Design

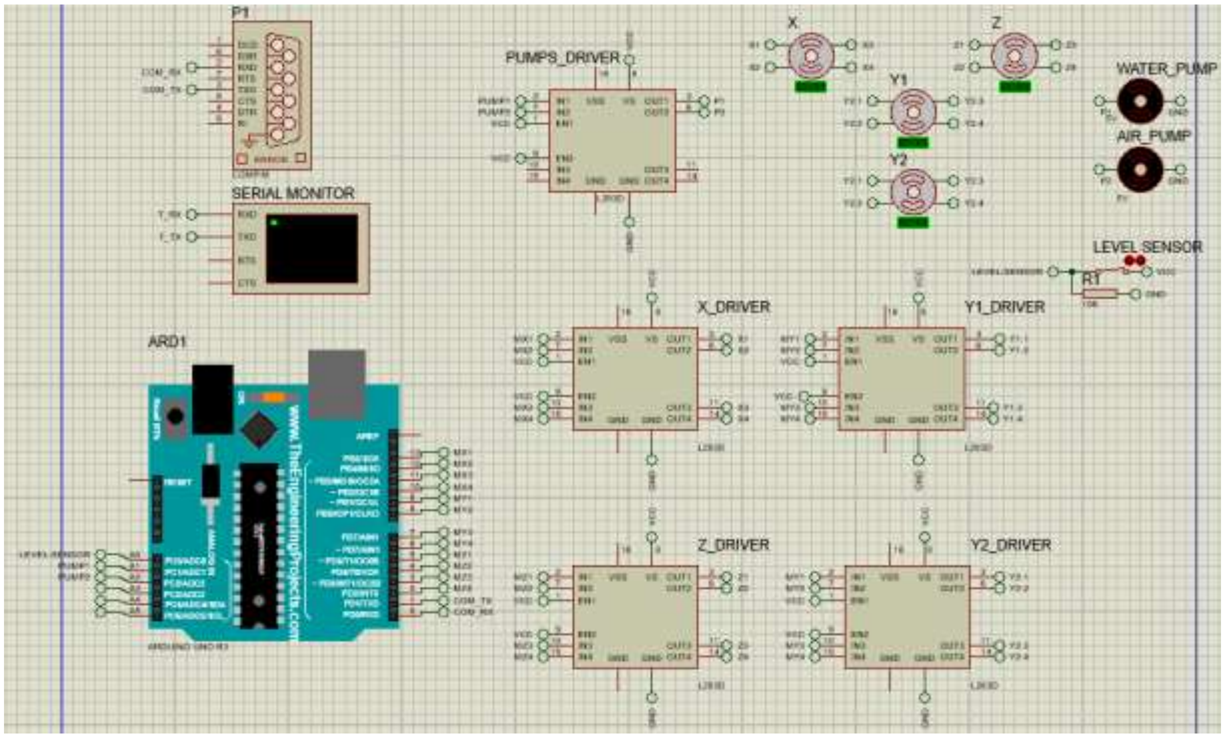
4.4 Proteus:

The Proteus design suite is the proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronic design engineers and technicians to create schematics and electronic prints for manufactory printed circuits boards.

4.4.1 Electrical circuit:

As a previously mentioned Proteus has been used to represent the electric circuit that consists of the following Equipment

Figure(4.5) below shown electrical circuit



Figure(4.5) Electrical Circuit

ARDUINO UNO :

The development of ARDUINO UNO board is considered as new compared to other ARDUINO boards. This board comes up with numerous features that helps the user to use this in their project. The ARDUINO UNO uses the Atmega16U2 microcontroller that helps to increase the transfer rate and contain large memory compared to other boards. No extra devices are needed for the ARDUINO UNO board like joystick, mouse, keyboard and many more. The ARDUINO UNO contain SCL and SDA pins and also have two additional pins fit near to RESET pin.

The board contains 14 digital input pins and output pins in which 6 pins are used as PWM, 6 pins as analog inputs, USB connection, reset button and one power jack. The ARDUINO UNO board can be attached to computer system buy USB port and also get power supply to board from computer system. The ARDUINO UNO contains flash memory of size 32 KB that is used to the data in it. The other feature of the ARDUINO UNO is compatibility with other shield and can be combined with other ARDUINO products.

The ARDUINO programmed in a form gives it the ability to communicate with processing and translate alongside understand to the communication language (G-CODE), and transform it to a series of commands which concerned with controlling different motors and pumps in terms of starting and stopping. In addition to monitor the water level in the tank by reading the value of the water sensor and notifying processing in case the water drops to a critical points.

- Stepper Motor:

A stepper motor, also known as step motor or stepping motor, is a brushless DC electric motor that divides a full rotation into a number of equal steps. The motor's position can then be commanded to move and hold at one of these steps without any position sensor for feedback (an open-loop controller), as long as the motor is carefully sized to the application in respect to torque and speed.

A two linear actuators are used for x and z axes movement with the following specific parameters:

Motor	Power Draw	Step angle	Current(Amp/phase)	Holding Torque(N-M)
	12.81 W	1.8^0	2.5	2.034

Lead Screw	Dimension(mm)	Travel Per 1.8^0 Step (mm/ step)
	8	0.1

For y axis movement two steeper motor are used, one for each leg and it has connected to work symmetrically. The parameter of this stepper motors are:

Step angle	Current(Amp/phase)	Holding torque (N-M)
1.8^0	3.15	3.06

- Drivers:

Stepper motor driver is an actuator which can transform pulse signal into angular displacement signal, stepper drivers drive stepper motor to rotate at an angle called step angle in the set direction when receiving a pulse signal.

The motor speed is up to the pulse frequency given from the controller. Stepper system consists of a stepper motor and a stepper driver. The performance of a stepper system is not only up to the motor, but also depends on the stepper driver.

- Pumps:

A pump is a device that moves fluids, or sometimes slurries, by mechanical action, typically converted from electrical energy into Hydraulic energy. Pumps can be classified into three major groups according to the method they use to move the fluid: direct lift, displacement, and gravity pumps.

There are two types of pumps are used in this system to do two different jobs. One is dedicated to pumping water for irrigation purposes, the chosen one for this goal has the following data:

24 Voltage	45 Watt	10000 Liter/Hour

The other one is an air pump for seeding purposes.

- Relay:

A relay is an electrically operated switch. It consists of a set of input terminals for a single or multiple control signals, and a set of operating contact terminals. The switch may have any number of contacts in multiple contact forms such as make contacts, break contacts, or combinations thereof.

- Level Sensor:

Level sensors detect the level of liquids and other fluids and fluidized solids, including slurries, granular materials, and powders that exhibit an upper free surface. Substances that flow become essentially horizontal in their containers (or other physical boundaries) because of gravity whereas most bulk solids pile at an angle of repose to a peak. The substance to be measured can be inside a container or can be in its natural form (e.g., a river or a lake). The level measurement can be either continuous or point values. Continuous level sensors measure level within a specified range and determine the exact amount of substance in a certain place, while point-level sensors only indicate whether the substance is above or below the sensing point. Generally the latter detect levels that are excessively high or low.

A point – level sensor has been uesd in this project and that to make sure they are water in the tank to do the irrigation process. Position of this sensor is designed to be in bottom of the tank. the readings of this sensor either 1 or 0, 1 means the water level is above the sensor and 0 for lower level.

Connection

17 ins of ARDUINO pins has been used for this circuit

- Pin 0 and 1 are communication pins for receiving and transmitting data between the ARDUINO and processing.
- Every stepper driver requires 4 pins from the ARDUINO to secure ideally operation condition, therefor 12 pins are used for motors drivers from pin2 to pin13.
- Pin A0 connected to the level sensor and designed as a digital input.
- Pin A1 and pin A2 are connected to relays to control the operation of water pump and air pump which are supplied by the power supply.
- The power supply provide the required power for the motors and pumps.

4.5 Operation System

In the beginning, the crop to be planted is selected, then the system calculates the number of the seedlings to be planted in the workspace this is done through the equation shown below.

Then the system displays the crop data such as the remaining time for harvesting and the remaining time for the next irrigation process which it will be after the first one that was after seeding process directly.

In the seeding process the stepping motors will move the movement arm at the starting point to go to the tool box to take the seeding tool and then go to the seeds holder to take seed from it every time until the seeding process is complete.

After the end of seeding process, the seeding tool is returned to the tool box and then the first irrigation process for the crop begins, where the movement arm go to the tool box to take the tool designated for the irrigation process, then the arm goes to the cultivated area and the water pump is turned on to give each planted plant the actual quantity it needs from water according to the data collected for each crop in **table(4.1),** so that the pump sprays water for a specific period of time for each single plant that was calculated according to the equation below.

The continuation of this process is done through a G-Code linking the Processing program with the arduino device in the electrical circuit that has been programmed to understand that G-Code commands from Processing program.

After the end of irrigation process, the irrigation tool is returned to the tool box and the irrigation process is repeated daily until harvest time.

When it is time to harvest the crop according to the aforementioned schedule, the system notifies the user that it is time to harvest, so that he manually performs this process.

All these processes have been represented in block diagram as shown below in **Figure (4.6)** :

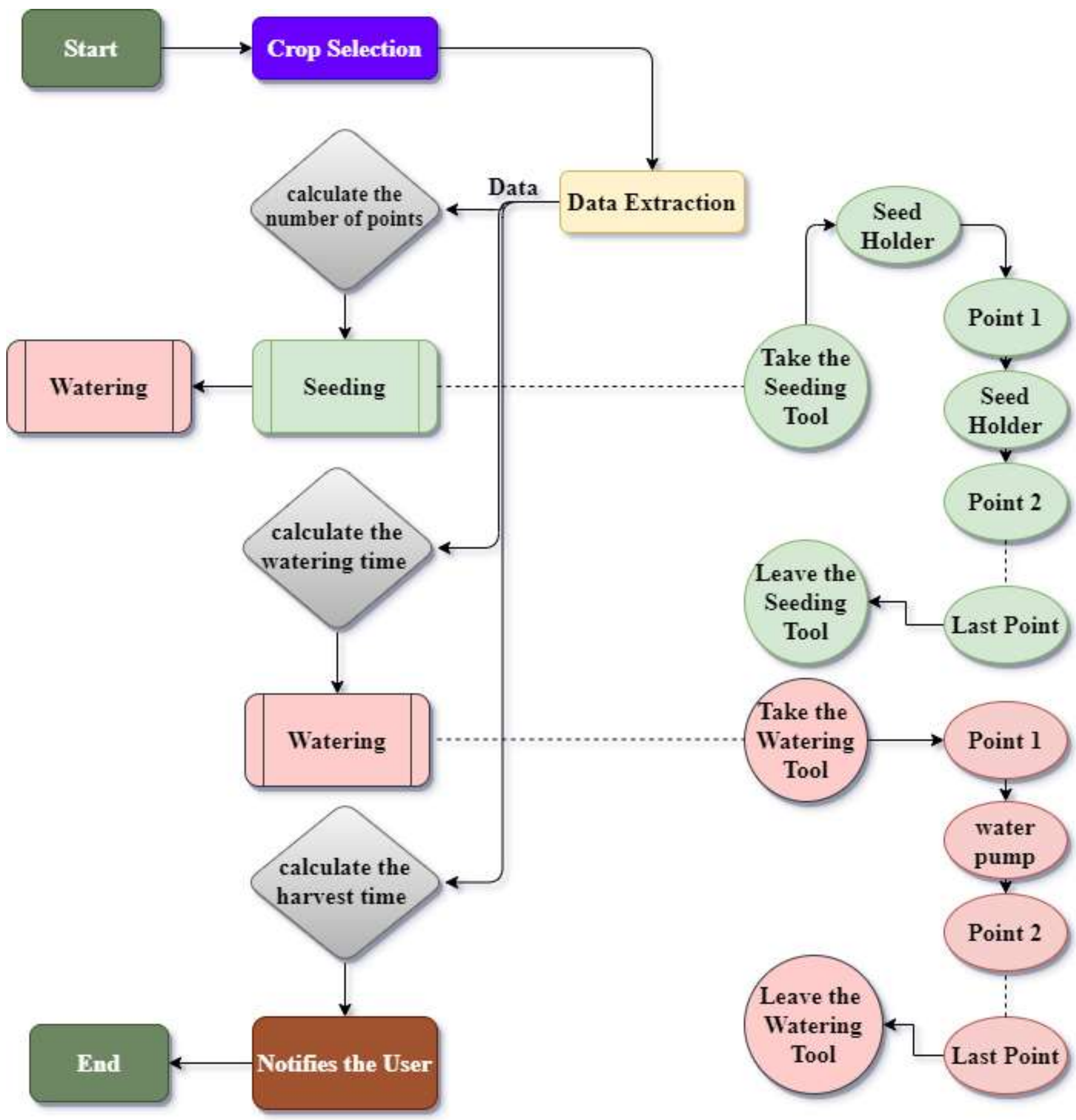


Figure (4.6) Flow Chart

CHAPTER FIVE

CONCULOTION AND RECOMMENDATIONS

5.1 Conclusion

Therefore this research helps people monitor and control their small home farm by using a fully automatic system that they can manage remotely, and the two most difficult processes in agriculture (agriculture and irrigation) were combined in one machine, which made it easier, and the amount of water needed by each crop was calculated. It reduced the wasted water and thus the cost. Also this research can be used for larger agricultural areas by increasing its dimensions and increasing the capabilities of the elements (generators and pumps).

5.2 Recommendations

- The research have targeted home farming, so everyone can do it in a small yard inside the house.

- The future is moving towards smart ideas with improving technologies that replace smart applications (automation) with the invention of the Internet of Things (IOT), this thing can be done by adding a raspberry pi to the system.

- Agriculture is the main source of survival in this world, and here future agriculture is also moving towards these smarter technologies with modern improvements in order to increase productivity in a short time.

- This program can be developed to perform more tasks and it is recommended to add more tools such as a weeding tool and a harvesting tool.

- Sensors can be added to measure the distance of the watering tool from the plant in the event that the project expands to larger areas.

- It is also possible to add sensors to the path of the axes to prevent any accedient .

- Adding a solar energy source to feed the electrical circuit of the project, especially if it is used for poor power supplied areas.

REFERENCES

1. Steve krar – Arthur Gill, Computer Numerical Control Programming Basics, INDUSTRIAL PRESS INC, NEW YORK-USA.
2. John R.walker, Maching Fundamentals From Basic to Advanced Techniques, The Good-Willcox Company Inc, 2000, USA.
3. John-David warren, Josh Adams, Harald Molle, Paul Manning, 2011, USA.
4. Ravi Gorli, Future Of Smart Farming with Internet of Things, MAN TECH Publications, 2017.
5. www.linengineering.com
   23/9/2020, 8:37 pm.
6. www.harvesttotable.com
   12/10/2020, 10:13 am
7. www.hrtnews.extension.iastate.edu/2004/7-23-2004/vegguide.html
   7/11/2020, 9:50 am

APPENDIX A

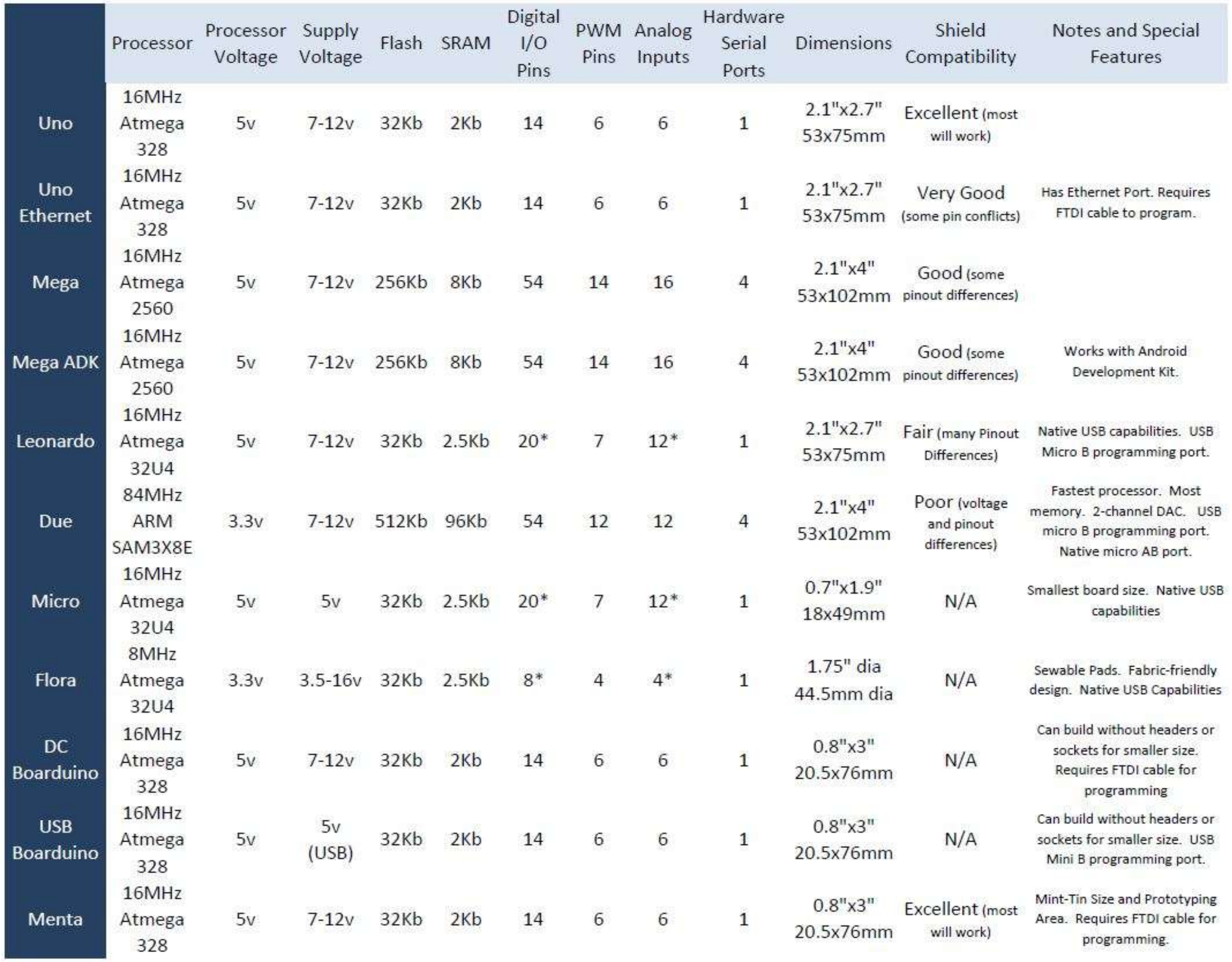
ARDUINO :

ARDUINO is an open-source platform used for building electronics projects. ARDUINO consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The ARDUINO platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the ARDUINO does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the ARDUINO IDE uses a simplified version of C++, making it easier to learn to program. Finally, ARDUINO provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

The ARDUINO board is invented for the electronics students to use this in their projects. The ARDUINO boards are provided as open source that helps the user to build their projects and instruments according to their need. This electronic platform contains microcontrollers, connections, LEDs and many more. There are various types of ARDUINO boards present in the market that includes ARDUINO UNO, ARDUINO Due, Micro, ARDUINO Mega, ARDUINO Leonardo, Flora. All these ARDUINO boards are different in specifications, features and uses and are used in different type of electronics project. **Table(5.1)** Shows the comparison between the different boards.

Table (5.1) Comparison between the different boards



	Processor	Processor Voltage	Supply Voltage	Flash	SRAM	Digital I/O Pins	PWM Pins	Analog Inputs	Hardware Serial Ports	Dimensions	Shield Compatibility	Notes and Special Features
Uno	16MHz Atmega 328	5v	7-12v	32Kb	2Kb	14	6	6	1	2.1"x2.7" 53x75mm	Excellent (most will work)	
Uno Ethernet	16MHz Atmega 328	5v	7-12v	32Kb	2Kb	14	6	6	1	2.1"x2.7" 53x75mm	Very Good (some pin conflicts)	Has Ethernet Port. Requires FTDI cable to program.
Mega	16MHz Atmega 2560	5v	7-12v	256Kb	8Kb	54	14	16	4	2.1"x4" 53x102mm	Good (some pinout differences)	
Mega ADK	16MHz Atmega 2560	5v	7-12v	256Kb	8Kb	54	14	16	4	2.1"x4" 53x102mm	Good (some pinout differences)	Works with Android Development Kit.
Leonardo	16MHz Atmega 32U4	5v	7-12v	32Kb	2.5Kb	20*	7	12*	1	2.1"x2.7" 53x75mm	Fair (many Pinout Differences)	Native USB capabilities. USB Micro B programming port.
Due	84MHz ARM SAM3X8E	3.3v	7-12v	512Kb	96Kb	54	12	12	4	2.1"x4" 53x102mm	Poor (voltage and pinout differences)	Fastest processor. Most memory. 2-channel DAC. USB micro B programming port. Native micro AB port.
Micro	16MHz Atmega 32U4	5v	5v	32Kb	2.5Kb	20*	7	12*	1	0.7"x1.9" 18x49mm	N/A	Smallest board size. Native USB capabilities
Flora	8MHz Atmega 32U4	3.3v	3.5-16v	32Kb	2.5Kb	8*	4	4*	1	1.75" dia 44.5mm dia	N/A	Sewable Pads. Fabric-friendly design. Native USB Capabilities
DC Boarduino	16MHz Atmega 328	5v	7-12v	32Kb	2Kb	14	6	6	1	0.8"x3" 20.5x76mm	N/A	Can build without headers or sockets for smaller size. Requires FTDI cable for programming
USB Boarduino	16MHz Atmega 328	5v	5v (USB)	32Kb	2Kb	14	6	6	1	0.8"x3" 20.5x76mm	N/A	Can build without headers or sockets for smaller size. USB Mini B programming port.
Menta	16MHz Atmega 328	5v	7-12v	32Kb	2Kb	14	6	6	1	0.8"x3" 20.5x76mm	Excellent (most will work)	Mint-Tin Size and Prototyping Area. Requires FTDI cable for programming.

APPENDIX B

ARDUINO code:

```
#include <Stepper.h>
#define STEPS 100;
#define BORD_BUFFER_LENGTH 512
#define LVL_SENSOR A0
#define AIR_PUMP A1 //air pump
#define WATER_PUMP A2 //water pump
#define MAGNET A3 //water pump
const int ZUp = 10;
const int ZMid = 5;
const float ZDown = 0;
const float waterTool_x = -2.5;
const float waterTool_y = 2.5;
const float seeder_x = -2.5;
const float seeder_y = 7.5;
const float seedPack_x = -2.5;
const float seedPack_y = 11;
const int STEPSp = 50;  // per rev
Stepper stepperx(STEPSp, 10, 11, 12, 13);
Stepper steppery(STEPSp, 6, 7, 8, 9);
Stepper stepperz(STEPSp, 2, 3, 4, 5);
struct point {
  float x;
  float y;
  float z;
};
struct point actuatorPos;
float StepsPercentimeterX = 50;
float StepsPercentimeterY = 50;
float StepsPercentimeterZ = 50;
float Xmin = -5;
float Xmax = 10;
float Ymin = 0;
float Ymax = 20;
float Zmin = 0;
float Zmid = 5;
float Zmax = 10;
float Xhome = 0;
float Yhome = 0;
float Xpos = Xhome;
float Ypos = Yhome;
float Zpos = Zmax;
float StepInc = 1;
int StepDelay = 7;
boolean seederReady=false;
boolean waterReady=false;
int level;
String x_info= "X";
String y_info= "Y";
String z_info= "Z";

void setup() {
 Serial.begin(115200);//Serial.println("Begin");
  Serial.setTimeout(3);
 stepperx.setSpeed(200);
  steppery.setSpeed(200);
stepperz.setSpeed(150);
pinMode(LVL_SENSOR, INPUT); // level sensor pin
  pinMode(AIR_PUMP, OUTPUT); // air pump pin
  pinMode(WATER_PUMP, OUTPUT); // water pump
  delay(200);
}

void loop() {
```

```
while (! Serial.available() > 0 ){
      level = digitalRead(LVL_SENSOR);
      if (level == 0) {
      Serial.println("L0");}
  }
  String cmd = Serial.readString();
  cmd.toUpperCase();
  cmd.replace(" ", "");
  cmd.replace("\n", "");
  cmd.replace("\r", "");
processLine(cmd);
}
void processLine( String cmd ) {
  String cmd2;
  char buf[BORD_BUFFER_LENGTH];
  cmd.toCharArray(buf, BORD_BUFFER_LENGTH);
  int currentIndex = 0;
  char buffer[ 64 ];                              // sub array to save data, it works like a RAM

  //  Needs to interpret
  //  G1 for moving
  //  MD (arm down)
  //  MU (arm up)
  //  MW (arm mid)
  //  T1N take water head
  //  T1F put water head
  //  T2N take seed head
  //  T2F put seed head

  while ( currentIndex < BORD_BUFFER_LENGTH ) {
    switch ( buf[ currentIndex++ ] ) {
      case 'A':
        buffer[0] = buf[ currentIndex++ ];
        buffer[1] = '\0';
        cmd2 = String(buffer);
        air(cmd2.toFloat());
        break;
      case 'W':
        buffer[0] = buf[ currentIndex++ ];
        buffer[1] = buf[ currentIndex++ ];
        buffer[2] = buf[ currentIndex++ ];
        buffer[3] = buf[ currentIndex++ ];
        buffer[4] = '\0';
        cmd2 = String(buffer);
        water(cmd2.toFloat());//start water pump with specific time
        break;
      case 'X':
        buffer[0] = buf[ currentIndex++ ];
        buffer[0] = buf[ currentIndex++ ];
        buffer[1] = buf[ currentIndex++ ];
        buffer[2] = buf[ currentIndex++ ];
        buffer[3] = buf[ currentIndex++ ];
        buffer[4] = buf[ currentIndex++ ];
        buffer[5] = '\0';
        cmd2 = String(buffer);
        drawLine(cmd2.toFloat(), actuatorPos.y );// go to new x / old y
        delay(2000);
        break;
      case 'Y':
        buffer[0] = buf[ currentIndex++ ];
        buffer[0] = buf[ currentIndex++ ];
        buffer[1] = buf[ currentIndex++ ];
        buffer[2] = buf[ currentIndex++ ];
        buffer[3] = buf[ currentIndex++ ];
        buffer[4] = buf[ currentIndex++ ];
        buffer[5] = '\0';
        cmd2 = String(buffer);
        drawLine( actuatorPos.x, cmd2.toFloat() );// go to old x / new y
        delay(2000);
        break;
      case 'M':
        buffer[0] = buf[ currentIndex++ ];
        buffer[1] = '\0';
        cmd2 = String(buffer);
          if(cmd2.startsWith("U"))armUp();
          if(cmd2.startsWith("D"))armDown();
          if(cmd2.startsWith("W"))armMid();
          delay(2000);

        break;
      case 'T':
        buffer[0] = buf[ currentIndex++ ];
        buffer[1] = buf[ currentIndex++ ];
        buffer[2] = '\0';
        cmd2 = String(buffer);
          if(cmd2.startsWith("1N"))
          {
            if(seederReady){/*TODO: send error warning */return;}
            drawLine(-2.5 , 2.5);//go to waterTool position
            armDown();
            digitalWrite(MAGNET,HIGH);//grap the water tool
            armUp();
            delay(3000);
            waterReady = true;
          }
         else if(cmd2.startsWith("1F"))
          {
            drawLine(-2.5 , 2.5);//go to waterTool position
            armDown();
          digitalWrite(MAGNET,LOW);//release the water tool
          armUp();
          delay(3000);
          waterReady = false;
         }
        else if(cmd2.startsWith("2N"))
        {
          if(waterReady){/*TODO: send error warning */return;}
          drawLine(-2.5 , 7.5);//go to seeder tool position
          armDown();
          digitalWrite(MAGNET,HIGH);//grap the seeder tool
          armUp();
          delay(3000);
          seederReady = true;
         }
        else if(cmd2.startsWith("2F"))
        {
          drawLine(-2.5 , 7.5);//go to seeder tool position
          armDown();
          digitalWrite(MAGNET,LOW);//release the seeder tool
          armUp();
          delay(3000);
         Serial.println("W");
          seederReady = false;
        }
       break;
       case 'S':
        buffer[0] = buf[ currentIndex++ ];        // /!\ Dirty - Only works with 3 digit commands
        buffer[1] = '\0';
        cmd2 = String(buffer);
          if(cmd2.startsWith("H"))
            {
            if(!seederReady){/*TODO: send error warning */return;}
```

```
              drawLine(-2.5, 11);//go to seedPack position
              armDown();
              air(1);//start air pump
              armUp();
              delay(3000);
              }
          break;
     }
   }
 }
void drawLine(float x1, float y1) {
 String ax;
 String ay;
  if (x1 > Xmax) x1 = Xmax;
  if (x1 < Xmin) x1 = Xmin;
  if (y1 > Ymax) y1 = Ymax;
  if (y1 < Ymin) y1 = Ymin;
  //  Convert coordinates to steps
  x1 = (int)(x1*StepsPercentimeterX);
  y1 = (int)(y1*StepsPercentimeterY);
  float x0 = Xpos;
  float y0 = Ypos;
  //  Let's find out the change for the coordinates
  long dx = abs(x1-x0);
  long dy = abs(y1-y0);
  int sx = x0<x1 ? StepInc : -StepInc;
  int sy = y0<y1 ? StepInc : -StepInc;
  long i;
    for (i=0; i<dy; ++i) {
      steppery.step(sy);
         delay(StepDelay);
    }
    for (i=0; i<dx; ++i) {
      stepperx.step(sx);
      delay(StepDelay);}
  Xpos = x1;
  Ypos = y1;
  actuatorPos.x=Xpos/StepsPercentimeterX;
  actuatorPos.y=Ypos/StepsPercentimeterY;
   ax= actuatorPos.x;
   ay= actuatorPos.y;
  Serial.println(x_info += ax);// send the new x position
  x_info = "X";
  delay(100);
  Serial.println(y_info += ay);// send the new y position
  y_info = "Y";
}
void air(boolean open) {
  if (open) {
    digitalWrite(AIR_PUMP, HIGH);
  }
  else {
    digitalWrite(AIR_PUMP, LOW);
  }
}
void water(int water_time) {
  digitalWrite(WATER_PUMP, HIGH);
  delay(water_time); // delaying watering time
  digitalWrite(WATER_PUMP, LOW);
}
void armUp(){
String az;
int z0=Zpos;
float z1=10*StepsPercentimeterZ;
int dz=abs(z1-z0);
int sz=z0<z1? StepInc:-StepInc;
```

```
long i;
for (i=0;i<dz;i++){
  stepperz.step(sz);
  delay(StepDelay);
}
Zpos=z1;
actuatorPos.z=10;
az= actuatorPos.z;
Serial.println(z_info += az);// send the new z position
z_info= "Z";
}
void armDown(){
               String az;
  int z0=Zpos;
float z1=0;
int dz=abs(z1-z0);
int sz=z0<z1? StepInc:-StepInc;
long i;
for (i=0;i<dz;i++){
  stepperz.step(sz);
  delay(StepDelay);
}
Zpos=z1;
actuatorPos.z=0;
az= actuatorPos.z;
Serial.println(z_info += az);// send the new z position
z_info= "Z";}
void armMid(){
               String az;
  int z0 = Zpos;
float z1=5*StepsPercentimeterZ;
int dz=abs(z1-z0);
int sz=z0<z1? StepInc:-StepInc;
```

```
long i;
for (i=0;i<dz;i++){
  stepperz.step(sz);
  delay(StepDelay);
}
Zpos=z1;
actuatorPos.z=5;
az= actuatorPos.z;
Serial.println(z_info += az);// send the new z position
z_info= "Z";
}
```

APPENDEX C

Processing Code:

```
import lord_of_galaxy.timing_utils.*;     // additional library to work control time
import uibooster.*;                      // additional library for UI
import processing.serial.*;
import controlP5.*;                      // additional library for UI
boolean water;
boolean harvest;
String waterstate;
String projectname;        // the project name it can be any name
String crop;               // the crops  that you wanna plant
boolean isConfirmed;     // start the project condition
float xp;
float yp;
float xp_default=0;
float yp_default=0;
int zp_default=10;
int zp;
float r;
boolean pointscal_flag = false;
float rx;
float ry;
float x ;
float y ;
int xpoints;
int ypoints;
int watering_time;
int number_of_days_to_harvest;
int adddays;
int addmonths;
Serial arduino;
Stopwatch sw;           //watering stopwatch
Stopwatch sh;            // harvest stopwtch
Stopwatch hd;            //harvest's days stopwatch
UiBooster harvnoti;      // notification window
UiBooster harvnoti2;    // system notification
UiBooster waterlevel;
UiBooster textin;
UiBooster selection;
UiBooster comf;
ControlP5 wb;           //watring button
void setup() {
  size(1000, 500);
  wb = new ControlP5(this);
  sw = new Stopwatch(this);
  sh = new Stopwatch(this);
  hd = new Stopwatch(this);
  waterlevel = new UiBooster();
  harvnoti = new UiBooster();
  harvnoti2 = new UiBooster();
  textin = new UiBooster();
  selection = new UiBooster();
  comf = new UiBooster();
  arduino = new Serial(this,"COM1",115200); //start serial communication
  projectname = textin.showTextInputDialog("The Project Name :");
  crop = selection.showSelectionDialog(
    "Choice the crop?", // Question
    "Crops", // window title
    "Carrot", "Radish","Onion");        // choice
  comf.showConfirmDialog(
    "Do you want to start the Project?", // Question
    "Are you sure?", // window title
    new Runnable() {
    public void run() {
      isConfirmed = true;
    }
  }
  ,
    new Runnable() {
    public void run() {
      isConfirmed = false;
    }
  }
  );
  if (isConfirmed) {   // start the program
    sw.start();
    sh.start();
    hd.start();
  }
  wb.addButton("start watring")
    .setValue(0)
    .setPosition(800, 450)
    .setSize(150, 20)
    ;
  PFont Font;
  Font = loadFont("Cambria-Bold-28.vlw");
  textFont(Font);
  switch(crop) {
  case "Carrot":
    watering_time =24;
    number_of_days_to_harvest =70;
    adddays=10;
    addmonths=2;
    r=2.5;
    break;
  case "Radish":
    watering_time =24;
    number_of_days_to_harvest =80;
```

```
    adddays=20;
    addmonths=2;
    r=5;
    break;
      case "Onion":
    watering_time =12;
    number_of_days_to_harvest =110;
    adddays=20;
    addmonths=3;
    r=10;
    break;
  }
      yp=yp_default;   //------------------------------------------------------------------------------
    xp=xp_default;    //------------------------------------------------------------------------------
    zp=zp_default;    //------------------------------------------------------------------------------
}


void draw() {
  if (isConfirmed) {
    float tw; // Text width
    //recieve arm position in real time
    String c;
     while ( arduino.available()>0 ) {
     c = arduino.readStringUntil('\n');
    processIncomingLine(c);
    }
    //draw layout
    //--< project name >--
    background(100);
    noFill();
    fill(255);
    stroke(255);
```

```
textSize(20);
text(projectname, 20, 20);
tw = textWidth(projectname);
fill(255);
rect(20, 24, tw, 1);
//--< real time >--
noFill();
textSize(30);
text("Real time :", 50, 150);
textSize(50);
text( nf(hour(), 2) + ":" + nf(minute(), 2) + ":" + nf(second(), 2), 50, 200);
textSize(20);
text( nf(day(), 2) + "/" + nf(month(), 2) + "/" + year(), 50, 220);
noFill();
waterstates();//update status of watering process
//draw layout
//--< harvest date >--
textSize(30);
text("Harvest Date :", 50, 400);
textSize(20);
text("It takes " + number_of_days_to_harvest + " days for " + crop + " to be ready", 50, 430);
harvestime();//update time of harvest process
//draw layout
//--<  >--
noStroke();
rect(800, 100, 150, 300);
//--<  >--
stroke(0);
strokeWeight(2);
fill(#907B24);
rect(850, 100, 100, 300);   // field
pointscal();
//draw layout
```

```
//--< draw crops position >--
x=850+rx/2;
y=100+ry/2;
for (int cx=0;cx < xpoints ; cx++){
  for (int cy=0;cy < ypoints;cy++){
   fill(#5F521C);
   ellipse(x,y,rx/2,ry/2);
   y = y+ry;
  }
  y=100+ry/2;
  x =x+rx;
}
//--< tool bar rectangle >--
noFill();
rect(800, 220, 50, 180);
line(800, 250, 850, 250);
line(800, 325, 850, 325);
fill(90);
rect(800, 250, 50, 75);
rect(800, 325, 50, 75);
fill(#3C4262);
rect(800, 220, 50, 40);
ellipse(825, 362.5, 25, 30);
ellipse(825, 287.5, 25, 30);
fill(255);
textSize(16);
text("The X and Y position", 800, 50);
xyposition();// x/y lines
zposition();//z rectangle
water=false;
harvest=false;
if (sw.hour() == watering_time) {
  sw.pause(); //pause the stopwatch
```

```
      water = sw.isPaused();
    }
    if (sh.hour() == number_of_days_to_harvest*24 ) {
      sh.pause(); //pause the stopwatch
      harvest = sh.isPaused();
    }
    if (water == true) {
      watering();
    }
    if (harvest == true) {
      harvesting();
    }
  }
}
void watering() {
  //send G-Code to the arduino
  startwatring();
  sw.restart();
  water = false;
}

void harvesting() {
  // send system notification
  harvnoti2.createNotification(
    "The plants are ready to Harvest", "Auto farming");
  // send progarm notification
  harvnoti.showWarningDialog("Harvest Time ... ", "Notification");
}

void harvestime() {
  int newd = day();
  int newm = month();
  int newy = year();
int dif=0;
int days=0;
if (hd.hour() == 24) {
  days++;
  hd.restart();
}
  switch(newm) {
  case 1:
  case 3:
  case 5:
  case 7:
  case 8:
  case 10:
  case 12:
    newd = newd +adddays;
    dif = newd - 31;
    if (dif<0) {
      newm=newm+2;
    } else if (dif>0) {
      newd=dif;
      newm=newm+3;
    }
    break;
  case 4:
  case 6:
  case 9:
  case 11:
    newd = newd + adddays;
    dif = newd - 30;
    if (dif<0) {
      newm=newm+addmonths;
    } else if (dif>0) {
      newd=dif;
```

```
        newm=newm+addmonths+1;
      }
      break;
    case 2:
      newd = newd + adddays;
      dif = newd - 28 ;
      if (dif<0) {
        newm=newm+addmonths;
      } else if (dif>0) {
        newd=dif;
        newm=newm+1+addmonths;
      }
      break;
    }
    if (newm>12) {
      switch(newm) {
      case 15:
        newm=3;
        newy = newy +1;
      case 14:
        newm=2;
        newy = newy +1;
      case 13:
        newm=1;
        newy = newy +1;
      }
    }
  noFill();
  textSize(20);
  text(newd + "/" + newm + "/" + newy, 50, 450);
  textSize(13);
  text("day number " + days + "  " + sh.hour() + ":" + nf(sh.minute(), 2) + ":" + nf(sh.second(), 2) + ":" +
nf(sh.millis(), 3), 50, 470);
```

```
}

void pointscal(){
  if(pointscal_flag)return;
 switch(crop){
   case "Carrot":
   rx=r*10;//25
   ry=r*15;//37.5
   xpoints = int(10/r);
   ypoints = int(20/r);
   break;
   case "Radish":
     rx=r*10;
   ry=r*15;
   xpoints = int(10/r);
   ypoints = int(20/r);
  break;
     case "Onion":
     rx=r*10;
   ry=r*15;
   xpoints = int(10/r);
   ypoints = int(20/r);
  break;
 }
 startseeding();
 pointscal_flag = true;
}

void processIncomingLine(String info) {
  char[] info2;
  info2 = info.toCharArray();
  switch ( info2[0] ) {
  case 'X':
    char[] sa2 = subset(info2, 1);
    xp = float(new String(sa2));
    break;
  case 'Y':
    char[] sa3 = subset(info2, 1);
    yp = float(new String(sa3));
    break;
  case 'Z':
    char[] sa4 = subset(info2, 1);
    zp = int(new String(sa4));
    break;
  case 'L':
    if (info2[1] == 1) {
    } else if (info2[1] == 0) {
      harvnoti.showWarningDialog("The tanke is empty ", "Notification");
    }
    break;
  case 'W':
    startwatring();
    break;
  default:
    break;
  }
}

void startseeding(){
 // send the G-Code to the arduino for seeding sqence
  arduino.write("T2N");   //catch tool
  x=850+rx/2;//862.5
  y=100+ry/2;//118.75
  println(xpoints);
    println(ypoints);
 for (int cx=0;cx < xpoints ; cx++){
   for (int cy=0;cy < ypoints;cy++){
      arduino.write("SH");//take seed
      arduino.write("X*"+ nf(map(x,850,950,0,10),2,2));//go to x point
       arduino.write("Y*"+ nf(map(y,100,400,20,0),2,2));//go to y point
      //arduino.write("XY"+ map(x,850,950,0,10) + ',' + map(y,100,400,20,0));//go to x,y point
      arduino.write("MD");//arm down
      arduino.write("A0");//close air pump to release the seed
      arduino.write("MU");//arm up
      y = y+ry;
   }
    y=100+ry/2;
    x =x+rx;
 }

  arduino.write("T2F");//take off tool
  //be stand by for another process
      arduino.write("Y*"+ nf(0,2,2));//go to y point
      arduino.write("X*"+ nf(0,2,2));//go to x point
}
void startwatring(){
   //send G-Code to the arduino for watring squence
     arduino.write("T1N");//grap water head
     arduino.write("MW");//hand medium
     x=850+rx/2;
     y=100+ry/2;
     for (int cx=0;cx < xpoints ; cx++){
       for (int cy=0;cy < ypoints;cy++){
        arduino.write("X*"+ nf(map(x,850,950,0,10),2,2));
        arduino.write("Y*"+ nf(map(y,100,400,20,0),2,2));
        //arduino.write("XY"+ map(x,850,950,0,10) + ',' + map(y,100,400,20,0));//go to x,y point
        arduino.write("W2000");//1=start air pump with delay 2000 ms
        y = y+ry;
        }
        y=100+ry/2;
        x =x+rx;
      }
  arduino.write("MU");
  arduino.write("T1F");
  //be stand by for another process
      arduino.write("Y*"+ nf(0,2,2));//go to y point
      arduino.write("X*"+ nf(0,2,2));//go to x point
}
void  waterstates(){
  int timelift=0;
  timelift = watering_time -sh.hour();
  if (timelift>0){
    waterstate = "The Next watering after " + timelift + " hour";
  }
  else if ( timelift ==0 ) {
    waterstate = " watring........";
  }
  textSize(30);
  text("watering states :", 50, 300);
  textSize(17);
  text(crop + " has to be watered every " + watering_time + " hours to maintain the soil wet.",50,330);
  textSize(15);
  text(waterstate, 50, 350);
  textSize(10);
  text(sw.hour() + ":" + nf(sw.minute(), 2) + ":" + nf(sw.second(), 2) + ":" + nf(sw.millis(), 3), 50, 360);
}
void  xyposition(){
  float xs;
  float ys;
  xs = map(xp,-5,10,800,950);
  ys = map(yp,0,20,400,100);
  stroke(255,0,0);
  strokeWeight(2);
  ellipse(xs,ys,7,7);
  line(780,ys,970,ys);
  line(xs,80,xs,420);
}
void zposition(){
  noStroke();
  fill(#4A4171);
 rect(700,100,50,160);
 fill(255);
 textSize(20);
 text("Up",660,120);
 text("Mid",650,185);
 text("Down",640,260);
  textSize(10);
 text("The Z Position",660,80);
  switch(zp){
   case 10:
    stroke(255,0,0);
    fill(0);
   rect(700,100,50,20);
   break;
   case 5:
   stroke(255,0,0);
   fill(0);
   rect(700,160,50,20);
   stroke(255,0,0);
   break;
   case 0:
    stroke(255,0,0);
    fill(0);
   rect(700,240,50,20);
   break;} }
```