بسم الله الرحمن الرحيم

SUDAN UNIVERSITY OF SIENCE AND TECHNOLOGY

COLLAGE OF GRADUATE STUDIES

Monitoring and Control System of Center-Pivot Irrigation System through Internet

نظام تحكم و مراقبة لمنظومة الري المحوري عن طريق الإنترنت

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF THE DEGREE OF M.Sc. IN MECHATRONICS ENGINEERING

Prepared By: Yasir Yousif Khalid Makki

Supervisor:

Prof. Sharief Fadul Babikir

September 2020

Dedication

This thesis is dedicated to:

My great parents.

My beloved brothers and sister.

My teachers and friends.

Acknowledgements

During preparing this thesis, I was in contact with many people, researchers, Academicians and designers. They have contributed towards my understanding and thoughts. In particular, I am glad to express my sincere appreciation to my thesis supervisor, Professor: Sharief Fadul Babikir, for guidance, critics and advises.

I also appreciate the support from my postgraduate colleagues. My sincere appreciation also extends to all others who have provided assistance at various levels in this thesis. Their views and tips are helpful indeed. Unfortunately, it is not possible to list all of them in this limited space. Without their continued support and interest, this thesis would not have been the same as presented here.
I am grateful to all my family members.

المستخلص

يعتبر نظام الري المحوري من أحدث أنظمة الري المستخدمه في العصر الحديث حيث يكون توزيع المياه في هذه المنظومة عن طريق هيكل دوار يحمل انبوب مياه به العديد من الرشاشات الموزعه علي طول الانبوب، يدور هيكل المنظومه في شكل دائري لتنفذ عملية الري. يتم التحكم في هذه المنظومة عن طريق لوحة تحكم مرفقة بمركز الجسم الدوار. تهدف هذه الاطروحة الي تصميم نظام مراقبة و تحكم لمنظومة الري المحوري عن طريق الانترنت ، يتكون نظام المراقبة من حساس رطوبة و درجة حراره للوسط المحيط و كذلك حساس قياس ضغط المياه في الانبوب الرئيسي و حساس لقياس سرعة الرياح في الحقل بالاضافة الي حساس لتحديد اتجاه الجسم الدوار في منظومة الري المحوري. كذلك يتم التحكم في المنظومة عن طريق مفتاح كهربائي، للتحكم في تشغيل و ايقاف المنظومه و كذلك تشغيل و ايقاف المضخة. تتم عملية التحكم والمراقبة عن طريق صفحة انترنت مرتبط بخادم باستخدام المتحكمه (esp32) حيث ترتبط هذه المتحكمة بشبكة انترنت لاسلكية. تم عمل نموزج مبدئي للنظام و تم التحقق من فعالية استخدام صفحة الانترنت لارسال اشارات التحكم وكذلك استقبال القراءات من الحساسات علي ارض الواقع.

Abstract

The center pivot irrigation system is one of the most modern irrigation systems used in the modern world. Water distribution in this system is carried out by a rotary structure with a water pipe with many sprinklers distributed along the pipe. The structure of the system rotates in a round shape to carry out the irrigation process. This system is controlled through a control panel attached to the center of the rotor body (pivot point). The aim of this thesis is to design a monitoring and control system for the center pivot irrigation system through the Internet. The control system consists of temperature and humidity sensors of the surrounding medium, water pressure sensor in the main pipe and anemometer to measure the wind speed in the field as well as a sensor to determine the direction of the rotor body in the center pivot irrigation system. The system is also controlled by an electric switch (relay) to control the operation and shutdown of the system as well as running and stopping the water pump. The control and monitoring process is done via a web page connected to a server using the controller (ESP32) where this controller is connected to a wireless network (Wi-Fi). A prototype of the system was implemented and the effectiveness of using the web page to send control signals and receive readings from the sensors was verified.

Contents

List of Tables

List of Figures

List of Abbreviations

AC	Alternating Current
BT	Bluetooth
CP	Center Pivot
CPU	Central Processing Unit
CSS	Cascading Style Sheet
DC	Direct Current
GIS	Geographic Information System
GND	Ground
GPS	Global Positioning System
GUI	Graphic User Interface
HTML	Hypertext Markup Language
IOT	Internet of Things
LED	Light Emitting Diode
Mbps	Mega Byte per Second
MCU	Micro Controller Unit
Mpa	Mega Pascal
PIC	Programmable Integrated Circuit
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver Transmitter
VCC	Voltage at the Common Collector
Wi-Fi	Wireless Fidelity
WUSA	Wireless Underground Sensor Aide

Chapter One
Introduction

1.1 Overview

Any irrigation system design requires adjustment in the field. Designs must be tailored to the skills and willingness of the irrigation decision-maker to properly manage the system and make the adjustments [1].

Center pivot irrigation is one of the advanced systems which are used nowadays, providing a highly efficient performance in the field of irrigation as it matches many of the requirements of the proper irrigation process. Center pivot irrigation is a form of overhead (sprinkler) irrigation consisting of several segments of pipe (usually galvanized steel or aluminum) with sprinklers positioned along its length, joined together and supported by trusses, mounted on wheeled towers [2].

The machine moves in a circular pattern, and is fed with water from the pivot point at the center of the circle. The water is usually pumped from a source such as a well or a river, the pump is connected to the pivot at the pivot point [3].

Adding a remotely control and monitoring system to center pivot irrigation system will highly increase the efficiency of the system by eliminating in-field processes (control and monitoring), leading to reduce the need of workers on the field and their expected mistakes during operation.

1.2 Problem Statement

The control and monitoring functions are performed through a control panel attached to the pivot frame, which is considered a very restrictive process as it should be performed in the field.

1.3 Proposed Solution

The proposed solution is to design a remote control and monitoring system for the pivot through a website page and a server.

1.4 Objectives

- To design a control and monitoring system to be added to the center pivot irrigation system.
- To implement a website page connected to server to perform the remotely control and monitoring functions.
- To implement a hardware prototype.
- To test the system functions using the prototype.

1.5 Methodology

A control and monitoring system is designed using ESP32 to control the operation of the pivot and the water pump, furthermore the system detect the position of the pivot using rotary encoder, water pressure and wind speed in the field as well as temperature and humidity, this process is accomplished through ESP32 server. Hardware prototype is made to test the functionality of the system to provide the proper design.

A web page is designed using HTML and CSS programming languages to display the recorded data from the pivot on the server page and provide control signals to center pivot irrigation system.

1.6 Thesis outlines

This research consists of five chapters their outlines are as follows:

Chapter Two: This chapter includes a background about center pivot irrigation system and describes the literature review and related works.

Chapter Three: Methodology, this chapter describes the items chosen for the design process and picks the suitable choice considering its purpose (software programs, sensors, controllers, actuators), and gives the description of circuits of the control and monitoring system.

Chapter Four: This chapter will provide results and discussion of simulation and design of all circuits and the web page prescribed in the previous chapter.

Chapter Five: Contains the conclusion and the recommendations.

Chapter Two
Background and Literature Review

This chapter will explain the operation principle of the irrigation system and the main component, used to build the proposed system, also introduces research literature related to the topic.

2.1 The Center Pivot System Overview

The center pivot system consists of one single sprinkler pipeline of relatively large diameter and length (according to the field applied on), composed of high tensile galvanized light steel or aluminum pipes supported above ground by towers moving on wheels, long spans, steel trusses and/or cables see Figure (2.1). One end of the line is connected to a pivot mechanism at the center of the field area; the entire line rotates around the center. The applied rate of water emitters varies from lower values near the pivot to higher ones towards the outer end by the use of small and large nozzles along the line [4].

The center pivot (CP) is a low/medium pressure fully mechanized automated irrigation system of permanent assemble. It has become very popular in recent years for irrigation of most of field crops, cereals, legumes, forage and vegetables. It is also used for supplementary irrigation for rain fed grain.

The typical Center Pivot systems can be fixed permanent installations or movable/towable type with the central tower based on wheels or a skid, easy movement from field to field. The Linear Center Pivot is another common type towable system, which can irrigate rectangular or square shaped fields using a canal water resource parallel to the travel direction [4].

Figure (2.1): The pipeline, the supporting structure and the sprinkler. [5]

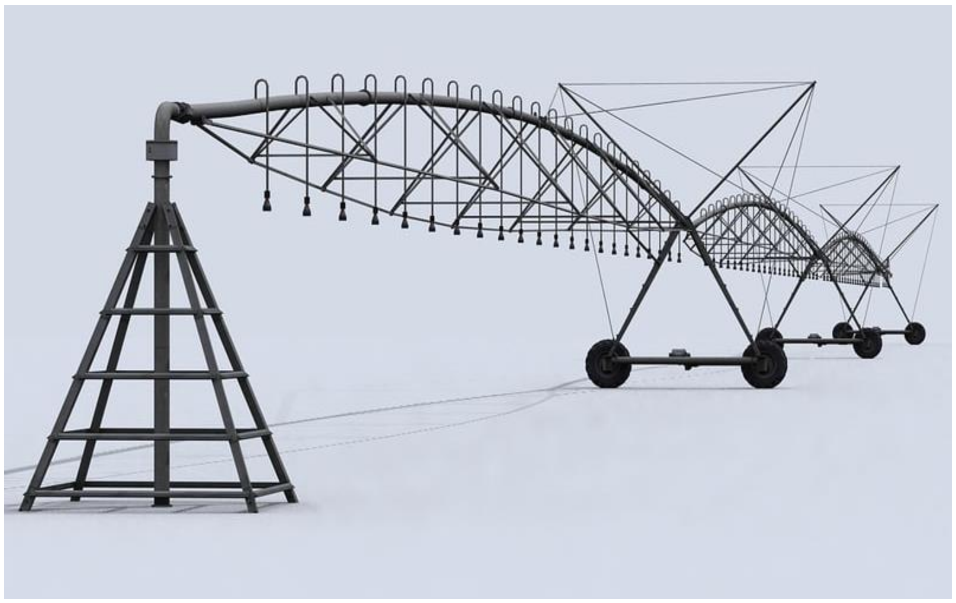


Figure (2.2): The structure of center pivot irrigation system. [6]

2.2 System Components and Operation Principle

The typical center pivot system consists of a single long irrigating pipeline attached to a central tower and moves slowly over the field in a circular pattern and irrigates the plants with sprayers, or sprinklers placed on it at frequent spacing, The central tower (pivot point) attached with a pivot mechanism and main control panel (electric) is mounted on a small concrete base at a fixed water supply point at the center of the field. The entire irrigating pipeline is supported above ground by "A-shaped" frame towers which move on wheels, long spans, steel trusses and/or cables; the end of the pipe is overhung with a sprinkler gun (It is an optional feature of the system). The whole system rotates slowly, at a typical speed (last span) of 2–3 m/min around the fixed pivot, self-propelled, providing water to the field in the form of overhead spray irrigation and covers the area in a circular pattern.

The drive system features small individual power units mounted on each wheeled tower. These units are electric drive motors, but can be hydraulic (water, oil) or mechanical drive. An automatic alignment system keeps always the irrigating pipeline straight [4].

Pivot point: the pivot point anchors the machine to a permanent location in the field see figure (2.4). Not only is it an anchor, but it also houses a system of subcomponents that contribute to the overall functionality of the pivot. These important subcomponents are the pivot legs, riser pipe, pivot swivel, control panel, J-pipe, and collector rings.

A) Pivot Legs - The pivot legs hold up the pivot point. The four pivot legs are typically bolted to a concrete pivot pad, providing support.

B) Riser Pipe - The riser pipe is connected to the first span through the pivot swivel. The purpose of the riser pipe is to supply water to the rest of the pivot.

C) Pivot Swivel - The pivot swivel is an elbow-shaped fitting that connects the riser pipe to the first span.

D) Control Panel - The control panel is mounted and accessed at the pivot point. The control panel gives the machine commands to start, stop, move in reverse, pump water, and many other functions. More on this component later.

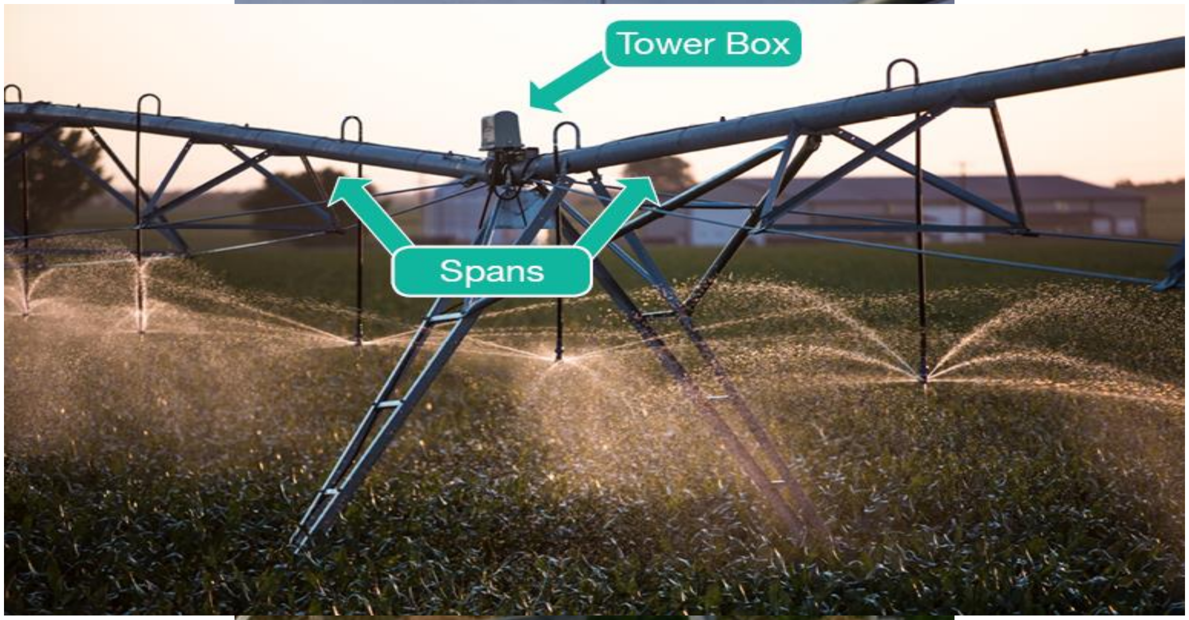
E) J-Pipe - The J-pipe houses the power and control circuit wires. These wires move through the J-pipe to the collector ring assembly.

F) Collector Ring - The collector ring assembly is made up of stationary brass rings with contact brushes that rotate around them. This assembly makes it possible for a continuous flow of electricity to run through the machine as it rotates around the pivot point.

Spans: spans are the backbone of the whole center pivot structure. Spans consist of pipes that transfer water to the field through the pivot's sprinklers package. The span pipes are supported by trussing and the drive units.

Tower Boxes: Each drive unit has a tower box that uses various components to control pivot movement and alignment. Span cables carry 120 and 480 volts of alternating current (VAC) to each tower box. Tower boxes then send 480VAC to the drive motors when signaled to move by the 120VAC control circuit [7].

Figure (2.3): Spans and Tower Box. [5]



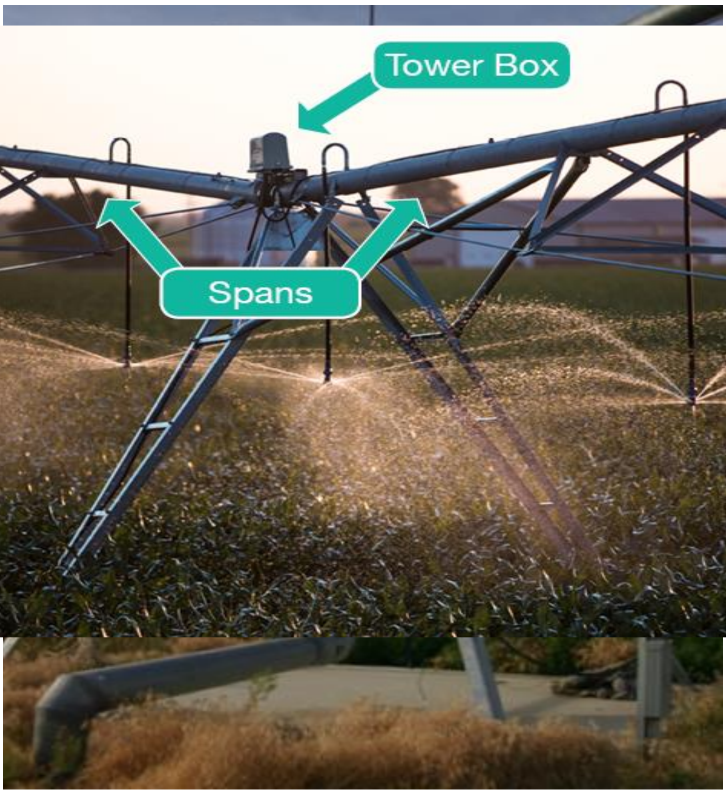


Figure (2.4): Pivot point A) Pivot Legs, B) Riser Pipe, C) Pivot Swivel, D) Control Panel, E) J-Pipe, F) Collector Ring. [5]

Drive units: the drive units and the supporting structure are also known as towers. They provide clearance above the crop for the spans and control the movement of the machine. Each drive unit consists of a drive motor, gearboxes, wheels, and an electrical control box known as a tower box.

Wheels: wheels for Irrigation use. First use High flotation wheels. Hot dip galvanized rims, with valve protection [7].

The center pivot irrigation system moves in a circular pattern, after triggering the machine to move the control circuit allows the power line to supply the last tower driving unit which allows the last tower only to move, when the last tower moves a certain distance it triggers the second from the last tower to start moving also.

Figure (2.5): The drive unit. [5]

This sequence of operation make sure that each tower control the movement of the tower after it in sequence, the system motion starts from the last tower to the first tower from the center. The system also is equipped with overwatering timer which prevent the system from overwatering specific area when the last tower stopped moving for a pre-determined time.

The system consists of cam mechanism to keep the alignment of the towers. The motion of the system is illustrated in the pictures below.

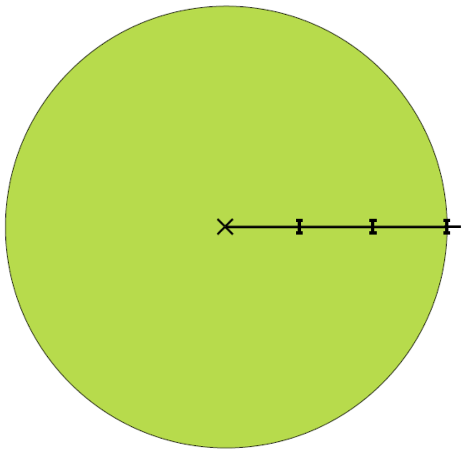


Figure (2.6): Stage one: before the motion starts.

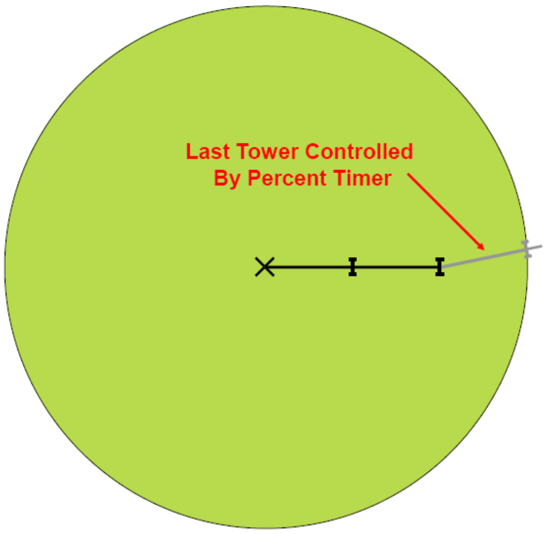


Figure (2.7): Stage two: the last tower movement.

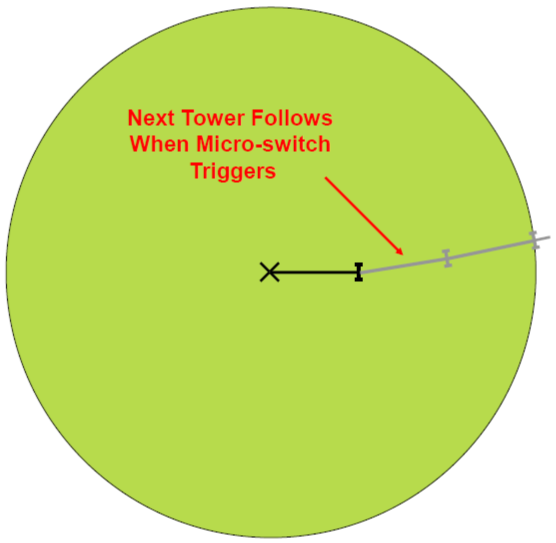


Figure (2.8): Stage three: the second from the last tower movement.

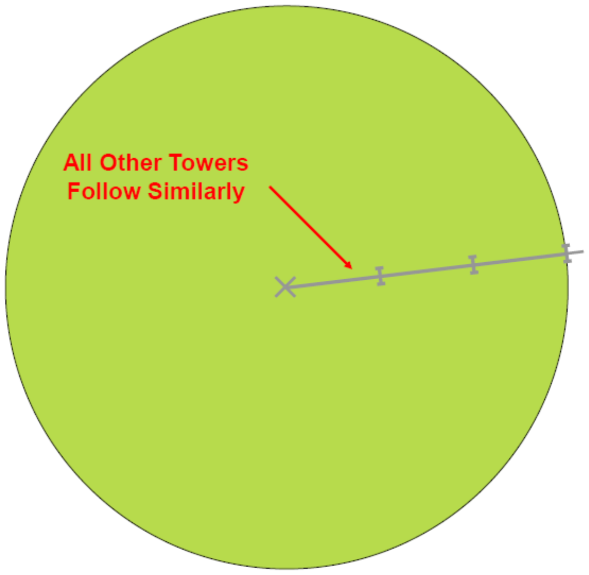


Figure (2.9): Stage four: the overall motion of the center pivot system.

2.3 Literature Review and Related Work

Kriti Taneja and Sanmeet Bhatia designed a system has soil moisture sensor inserted into the soil of the plants and a water level sensor placed in a water container from where water is pumped to plants for irrigation. An algorithm has been build out with threshold values of soil moisture sensor to control the water quantity in soil and also a water level sensor has been implemented to measure the water level in tank. Arduino board having built in ATMega328 microcontroller is required for the project. The manual irrigation is converted into an automated irrigation with the help of soil moisture sensor which detect humidity content of soil leading to turn ON/OFF of pumping motor. Human efforts is claimed to be reduced using this technique and increased the saving of water by efficiently irrigating the plants [8].

In 2016 a research called (IOT based crop-field monitoring and irrigation automation) represented a system was developed to monitor crop-field using sensors (soil moisture, temperature, humidity, Light) and automated the irrigation system. The data from sensors are sent to Web server database using wireless transmission. In server database the data are encoded in JSON format. The irrigation is automated if the moisture and temperature of the field falls below the brink. In greenhouses light intensity control is also automated in addition to irrigation [9].

The author of [10] designed a fully automated center pivot irrigation system which is consisted of water level system detecting several levels of water using sensors to each level and controlling the pump operation, and irrigation module is used to determine the time of irrigation or rest using microcontroller from PIC family. The irrigation system is connected to solar energy system.

A center pivot was thoroughly automated utilizing the temperature-time-threshold method of irrigation scheduling. An array of infrared thermometers had been mounted on the center pivot, and these were used to remotely detect the crop leaf temperature as an indicator of crop water stress. Methods which are used to automatically collect and analyze the canopy temperature data and control the moving irrigation system based on the data analysis were described. Automatic irrigation treatments were compared with manually scheduled irrigation treatments under the same center pivot for two different seasons. Manual irrigations were scheduled on a weekly basis using the neutron probe to determine the profile water content and the amount of water needed to replenish the profile to field capacity. In both seasons, there was no significant difference between manual and automatic treatments. The automatic irrigation system had the potential to simplify management, while maintaining the yields of intensely managed irrigation [11].

Yunseop Kim, Robert G. Evans and William M. Iversen, introduced a paper which described details of the design and instrumentation of variable rate irrigation, a wireless sensor network, and software for real-time in-farm sensing and control of a site-specific precision linear-move irrigation system. Farm conditions were site-specifically monitored by six in-field sensor stations distributed across the field based on a soil property map, and periodically sampled and wirelessly transmitted to a base station. An irrigation machine was transformed to be electronically controlled using a programming logic controller that updates georeferenced location of sprinklers from a differential Global Positioning System (GPS) and connected wirelessly to a computer at the base station. Communication signals from the sensor network and irrigation controller to the base station were properly interfaced utilizing low-cost Bluetooth wireless radio communication. Graphic user interface-based (GUI) software is developed and offered stable remote access to farm conditions and real-time control and monitoring of the variable-rate irrigation controller [12].

The authors in [13] proposed wireless monitoring system aims at reducing the level of the pesticides while ensuring a high quality production. A wireless sensor network was designed for the local measurement of the agro meteorological variables, which detect the disease development during the growing season of the plants. The collected information is processed in real-time by a fuzzy logic method for the assumption of the optimal pesticide dosage and the proper time to apply it. The performance of the proposed wireless system had been validated through experiment in a real test field for the optimal treatment of the grapevine downy mildew. A correct disease management is displayed from the obtained results with a reduced amount of agrochemicals up to 70% respect to the standard dosage.

A research work in 2018 proposed an automation system based on the Internet of Things (IoT), Geographic Information System (GIS) and quasi real-time in the cloud of water requirements to improve the efficiency of water use. Took into account each segment of the pivot-center moves at a different speed compared to others; thus, must be individually controlled to optimize the yield of irrigation. Moreover, it showed the necessity to integrate factors such as stage of crops' development, heterogeneity of soil, runoff, drainage, soil components, nutrients and moisture content. A complete system integrating sensors, GIS, Internet of Things and cloud computing were developed. This approach allowed to automate fine-grained the consumption of water without decreasing the yield. In addition to that, the collection of data and the soil moisture measurement was allowed to adapt coefficient of evapotranspiration to local weather without having to resort to lysimetric measures. The proposed architecture allowed to store and treat real-time,

time series data and low-priority data such as 3D images used in digital phenotyping field which were treated with batch processing [14].

In reference [15], a proof-of-concept towards an autonomous precision irrigation system is provided through the integration of a center pivot (CP) irrigation system with wireless underground sensor networks (WUSNs). This Wireless Underground Sensor-Aided Center Pivot (WUSA-CP) system had provided autonomous irrigation management capabilities by monitoring the soil conditions in real time and used wireless underground sensors. Field experiments with a hydraulic drive and continuous-move center pivot irrigation system were conducted. The results are used to evaluate empirical channel models for soil-air communications. The experiment outcomes are displayed that the concept of WUSA-CP is feasible. Through the design of an underground antenna, communication ranges are assumed to be improved by up to 400% compared to conventional antenna designs. It is highlighted through results that the wireless communication channel between soil and air was significantly affected by many spatio-temporal aspects, such as the location and burial depth of the sensors, soil texture and physical properties, soil moisture, and the vegetation canopy height. This was the first work on the development of an autonomous precision irrigation system with WUSNs.

2.4 Main Components

It is divided into two categories hardware and software components as follow.

2.4.1 Hardware Tools and Modules

Includes the controller, sensors and actuators required for the system design.

2.4.1.1 ESP32

ESP32-WROOM-32 is a powerful, generic Wi-Fi, BT, BLE MCU module that targets a wide variety of applications, ranging from low-power sensor networks to the most demanding tasks, such as voice encoding, music streaming and MP3 decoding.

Figure (2.10): ESP32-WROOM-32 chip. [16]

At the core of this module is attached the ESP32-D0WDQ6 chip. The chip is designed to be scalable and adaptive. There are two CPU cores that can be individually controlled. The user may also power off the CPU and make use of the low-power co-processor to constantly monitor the peripherals for any changes or crossing of thresholds. ESP32 provides a rich set of peripherals, including capacitive touch sensors, Hall sensors, SD card interface, Ethernet, high-speed SPI, UART, I^2S and I^2C [17].

The integration of Bluetooth, Bluetooth LE and Wi-Fi ensures that a wide range of applications can be accomplished, and that the module is all-around: using Wi-Fi allows a large physical range and direct connection to the internet through a Wi-Fi router, while using Bluetooth allows the user to conveniently connect to the phone or broadcast low energy beacons for its detection. The sleep current of the ESP32 chip is less than 5 mA, making it suitable for battery powered and wearable electronics applications. The module supports a data rate of up to 150 Mbps, and

20 dBm output power at the antenna to ensure the widest physical range. As such the module does offer industry-leading specifications and the best performance for electronic integration, range, power consumption, and connectivity [17].

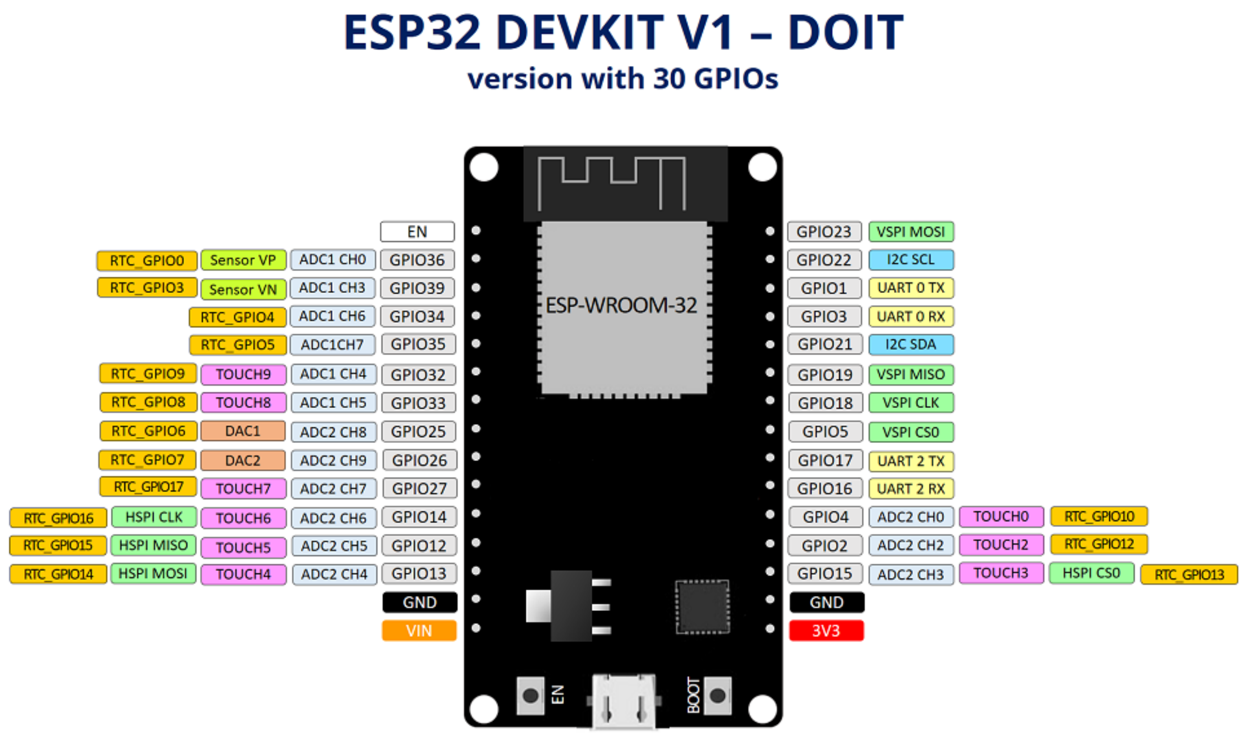

ESP32 DEVKIT V1 – DOIT
version with 30 GPIOs
ESP-WROOM-32
EN
RTC_GPIO0 Sensor VP ADC1 CH0 GPIO36
RTC_GPIO3 Sensor VN ADC1 CH3 GPIO39
RTC_GPIO4 ADC1 CH6 GPIO34
RTC_GPIO5 ADC1CH7 GPIO35
RTC_GPIO9 TOUCH9 ADC1 CH4 GPIO32
RTC_GPIO8 TOUCH8 ADC1 CH5 GPIO33
RTC_GPIO6 DAC1 ADC2 CH8 GPIO25
RTC_GPIO7 DAC2 ADC2 CH9 GPIO26
RTC_GPIO17 TOUCH7 ADC2 CH7 GPIO27
RTC_GPIO16 HSPI CLK TOUCH6 ADC2 CH6 GPIO14
RTC_GPIO15 HSPI MISO TOUCH5 ADC2 CH5 GPIO12
RTC_GPIO14 HSPI MOSI TOUCH4 ADC2 CH4 GPIO13
GND
VIN
GPIO23 VSPI MOSI
GPIO22 I2C SCL
GPIO1 UART 0 TX
GPIO3 UART 0 RX
GPIO21 I2C SDA
GPIO19 VSPI MISO
GPIO18 VSPI CLK
GPIO5 VSPI CS0
GPIO17 UART 2 TX
GPIO16 UART 2 RX
GPIO4 ADC2 CH0 TOUCH0 RTC_GPIO10
GPIO2 ADC2 CH2 TOUCH2 RTC_GPIO12
GPIO15 ADC2 CH3 TOUCH3 HSPI CS0 RTC_GPIO13
GND
3V3
EN
BOOT

Figure (2.11): ESP32 pins layout. [16]

All pins of ESP-WROOM-32 are led out to the pin headers on both sides for easy interfacing. ESP32-DevKitC features all the functions that are supported by ESP32. Users can connect these pins to peripherals as needed. The interfaces are shown in Figure (2.11) [18].

2.4.1.2 Rotary Encoder

A rotary encoder is a type of position sensor which is used for determining the angular position of a rotating shaft. The Keyes KY-040 rotary encoder is a rotary input device that provides a signal according to how much the knob has been rotated and what direction it is rotating in [19].

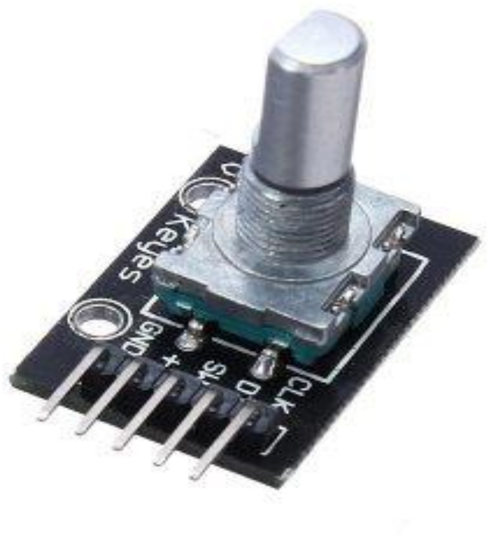


Figure (2.12): Rotary Encoder [20]

There are many different types of rotary encoders which are classified by either Output Signal or Sensing Technology. The particular rotary encoder that is used is an incremental rotary encoder and it’s the simplest position sensor to measure rotation.

When the disk will start rotating step by step, see figure (2.13) the pins A and B will start making contact with the common pin and the two square wave output signals will be generated accordingly. Any of the two outputs can be used for determining the rotated position if we just count the pulses of the signal. However, if we want to determine the rotation direction as well, we need to consider both signals at the same time. We can notice that the two output signals are displaced at 90 degrees out of phase from each other. If the encoder is rotating clockwise the output A will be ahead of output B [21].

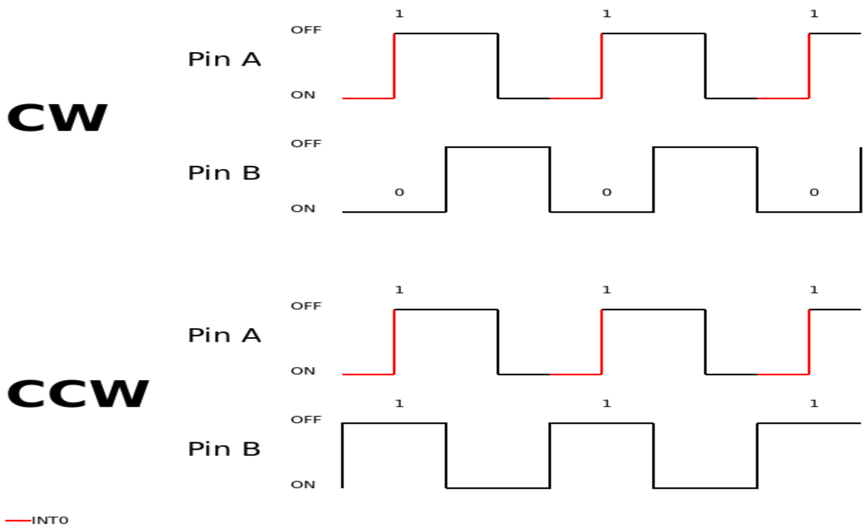


Figure (2.13): The square waves of A and B pins. [22]

The rotary encoder has 20 stages that reflect 20 different positions which can be used to detect the position of any rotary shaft.

2.4.1.3 Temperature and Humidity Sensor (DHT11)

The DHT sensors are made of two parts, a capacitive humidity sensor and a thermistor. There is also a very basic chip inside that does some analog to digital conversion and spits out a digital signal with the temperature and humidity. The digital signal is fairly easy to read using any microcontroller [23].

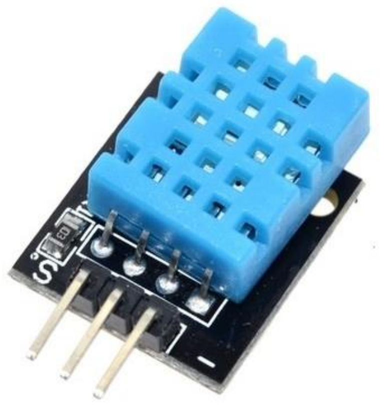


Figure (2.14): DHT sensor. [24]

It is fairly easy to connect up to the DHT sensors. They have four pins:

1. VCC- red wire Connect to 3.3 - 5V power. Sometime 3.3V power isn't enough in which case try 5V power.
2. Data out - white or yellow wire.
3. Not connected.
4. Ground - black wire.

Simply ignore pin 3, its not used. You will want to place a 10 KOhm resistor between VCC and the data pin, to act as a medium-strength pull up on the data line [23].

2.4.1.4 Water Pressure Sensor SKU: SEN0257

This is a water pressure sensor see figure (2.15) adapted with 3-pin interface. It supports standard 5V as input and 0.5~4.5V linear voltage as output. It is compatible with multiple controllers. The water pressure sensor can be plugged into controller board, wiring-free.

Figure (2.15): Water Pressure Sensor SEN0257. [25]

This water pressure sensor is a stethoscope to a water pipe. It is helpful to diagnose whether there is water, how strong the water pressure is. It can be widely applied to smart home control systems (SCS), Internet of Things (IoT) and device detection [26].

Table (2.1): Water Pressure Sensor Pinout.

Label	Name	Description
Yellow	Signal(Output:0.5~4.5V)	Analog Signal
Red	VCC(5VDC)	+
Black	GND	-

Specification:

- Medium: liquid/gas without corrosion.
- Wiring: Gravity-3Pin (Signal-VCC-GND).
- Pressure Measurement Range: 0~1.6 Mpa.

- Input Voltage: +5 VDC.
- Output Voltage: 0.5~4.5 V.
- Operating Temperature: -20~85°C.
- Quiescent Current: 2.8 mA.
- Normal Operating Pressure: ≤2.0 Mpa.
- Damaged Pressure: ≥3.0 Mpa.

2.4.1.5 Anemometer Wind Speed Sensor (Adafruit 1733)

An anemometer is a device used for measuring wind speed. This well-made anemometer is designed to be located outside and opposed directly to wind to measure wind speed with ease.it is wired through connecting the black wire to ground, the brown wire to 7-24VDC and measure the analog voltage on the blue wire. The voltage will range from 0.4V (0 m/s wind) up to 2.0V (for 32.4m/s wind speed). The sensor is rugged, and easily mounted. The cable is easy to disconnect with a few twists and has a weatherproof connector [27].

Specifications:

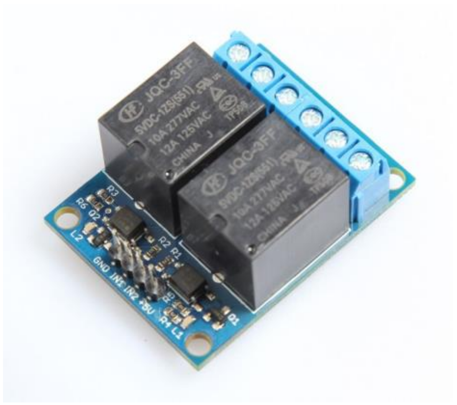
- Output: 0.4V to 2V
- Testing Range: 0.5m/s to 50m/s
- Start wind speed: 0.2 m/s
- Resolution: 0.1m/s
- Accuracy: Worst case 1 meter/s
- Max Wind Speed: 70m/s

Figure (2.16): Anemometer Wind Speed Sensor. [28]

2.4.1.6 Relay Module 2-Channel

This Relay Module 2-Channel is a module designed to provide the control of two relays in a very simple manner. Using the characteristics of the relays mounted on the module and through the use of two digital I/O pins of any controller, it is possible to control motors, inductive loads and other devices.

Figure (2.17): Relay Module 2-Channel. [29]

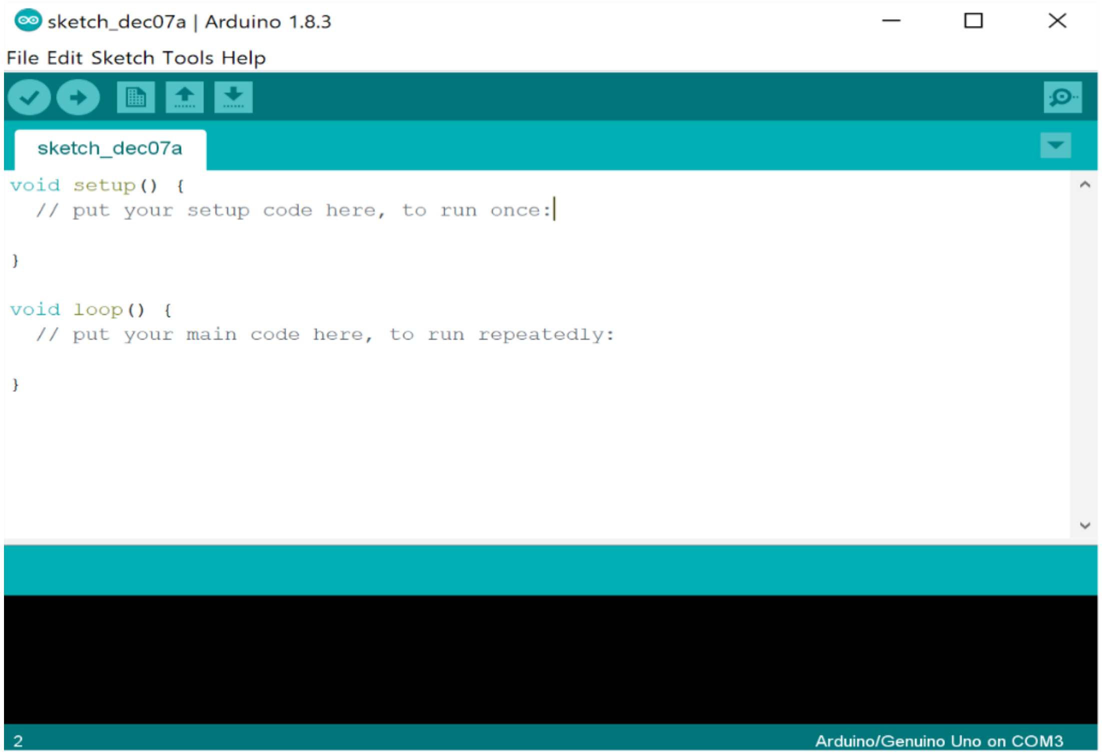


The module is equipped with opt-couplers on IN1 and IN2 lines which provide a galvanic insulation between the relay load and the control board which controls this module. Two LEDs are attached displaying the ON and OFF state of the two channels [29].

2.4.2 Software Programs

The software part is about the coding environment and every required library.

2.4.2.1 Arduino IDE


sketch_dec07a | Arduino 1.8.3
File Edit Sketch Tools Help
sketch_dec07a

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

2
Arduino/Genuino Uno on COM3

Figure (2.18): Arduino IDE Software.

The open-source Arduino Software (IDE) is very compatible software environment to write code and upload it to the board. It works on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino or ESP board. The coding languge is Arduino-C which is developed from C [30].

Chapter Three
System Design

3.1 System design

The block diagram shown in figure (3.1) is illustrating the requirements of the system to meet its overall function, also the main parts which should be included (sensors, actuators and controllers).

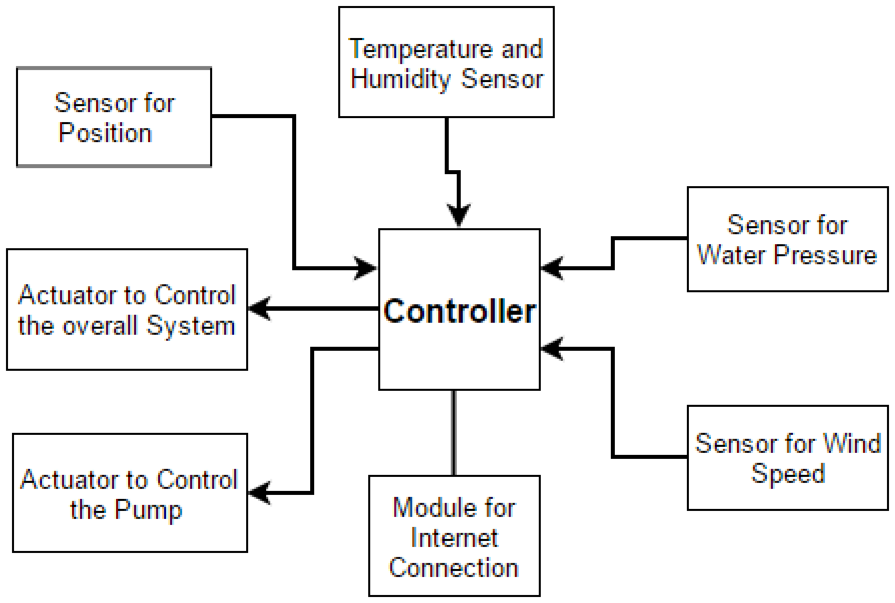

Temperature and Humidity Sensor
Sensor for Position
Sensor for Water Pressure
Actuator to Control the overall System
Controller
Actuator to Control the Pump
Sensor for Wind Speed
Module for Internet Connection

Figure (3.1): Block diagram for initial design of the system.

3.2 System Block diagram and Flow chart

Table (3.1) displays the component of the system and its functions to perform the overall system objectives properly.

Table (3.1): System component.

No	Name	Function
1.	ESP32	The controller which provides control signals received from the website page to actuate an act on the field, and acquires sensors readings' to be displayed on the web page.
2.	Rotary encoder	To detect the position of the rotary structure of center pivot irrigation system and send it to esp32 to display it on the web page.
3.	DHT11 sensor (Humidity and Temperature sensor)	To measure the ambient temperature and humidity of the field and transmit it to esp32 to be shown on the website page.
4.	Wind speed sensor (Anemometer)	To sense the wind speed in the field and represent it as voltage on the esp32 which will be converted to number that refer to the actual speed of the wind.
5.	Water pressure sensor	To measure the water pressure in main pipeline of the center pivot irrigation system which will be sent to esp32 in voltage form and to be converted mathematically to pressure in Pa.
6.	Relay Module	Close and open the control system circuit and also control the operation of the water pump. This module will be energized from the esp32 pins to perform the control function.

All lines in figure (3.2) represent a wired connection except the dashed line which represent a wireless connection between ESP32 module and client device where the website page will be displayed.

The system will be connected to the internet through Router modem providing internet service from one of the internet companies' providers, the modem should be set to port forwarding to allow access to the web server form route devices and clients.

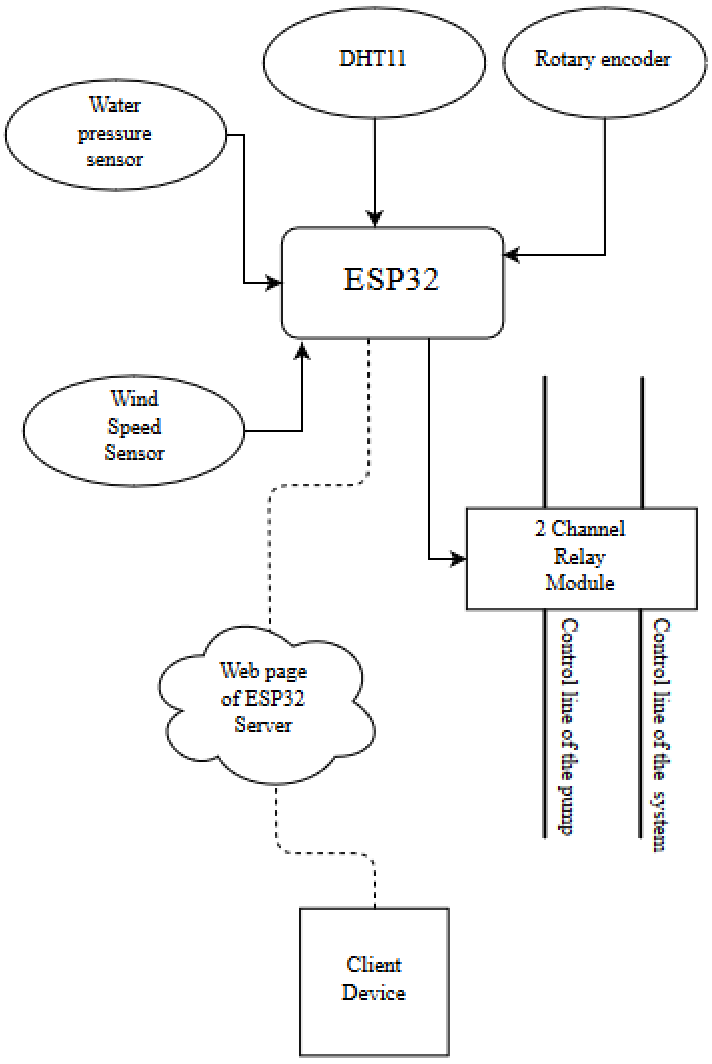


Figure (3.2): Proposed design block diagram.

In the flow chart in picture (3.3) the system starts to communicate with center pivot irrigation system, each sensor starts to acquire the field parameters and send these values to be stored in the esp32 memory, meanwhile the esp32 will initialize and search for the Wi-Fi network defined in the esp32 code, if the network is not found the esp32 will keep searching for the predefined network.

After finding and connecting successfully to the Wi-Fi network the esp32 will be a web server with IP address on the network, this web server waits for a client to connect through the IP address of the server (written in a web browser).

If the client is connected a web page will be displayed with two switches for control functions (controlling the pump and the main motor) and a table illustrating the sensors readings from the field.

The sensors on the field will continuously collect the data from the environment so the value will change. To make sure that the values displayed on the web page are fresh, you need to reload your page on the browser, which will bring and display the latest values stored in the ESP32 memory.

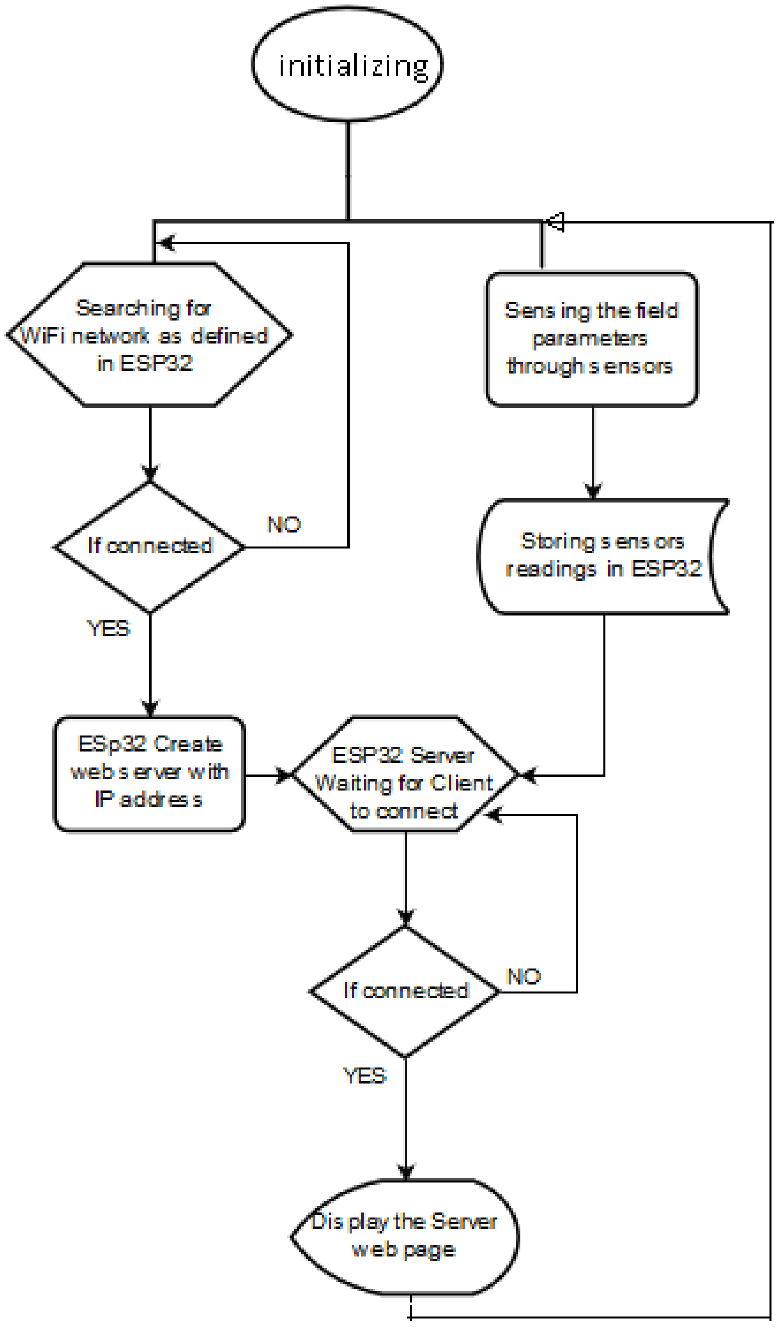

initializing
Searching for WiFi network as defined in ESP32
If connected
NO
YES
ESp32 Create web s erver with IP addres s
Sens ing the field parameters through s ens ors
Storing s ens ors readings in ESP32
ESP32 Server Waiting for Client to connect
If connected
NO
YES
Dis play the Server web page

Figure (3.3): System operation flow chart.

3.3 System Circuits

The block diagrams illustrate the wiring of each component with the esp32.

3.3.1 Relays Circuit

In figure (3.12) 2 Channel Relay Module is supplied with external power supply of 5v which is required for the module to function properly.

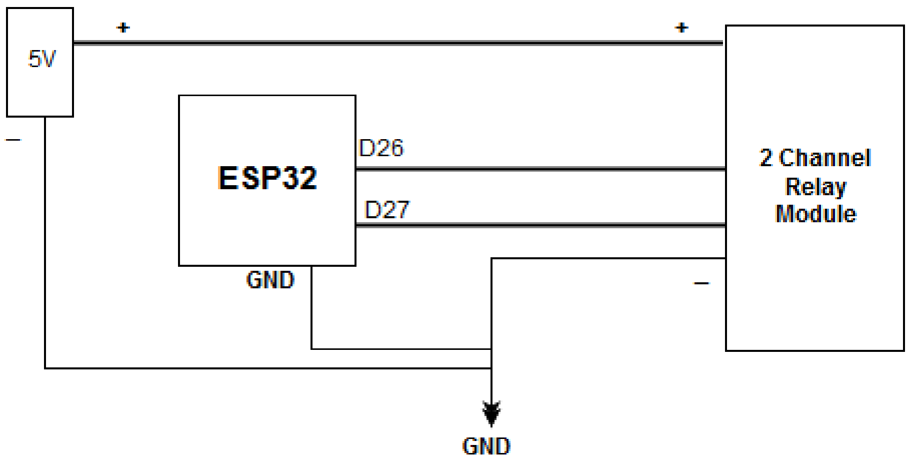


Figure (3.4): 2 Channel Relay Module Circuit with ESP32.

3.3.2 Sensors Circuits

The DHT11 sensor transmit the values of ambient temperature and humidity through data pin.

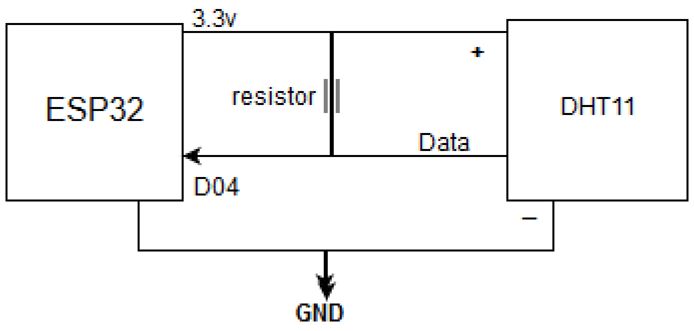


Figure (3.5): Temperature and Humidity Sensor (DHT11) Circuit with ESP32.

The esp32 software can read the measured data. It is required to place a pull-up resistor of 10KΩ between VCC and data line to keep it HIGH for proper communication between sensor and MCU.

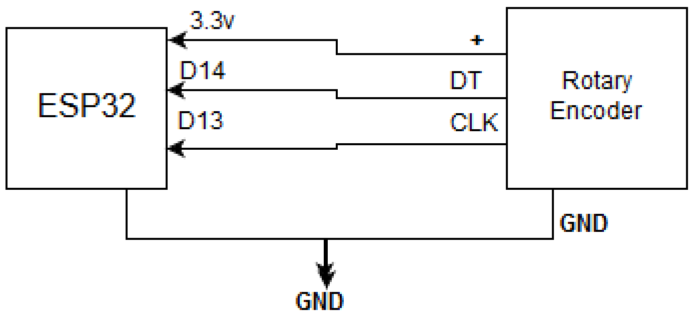


Figure (3.6): Rotary Encoder Sensor Circuit with ESP32.

Wiring the rotary encoder to ESP32:

- GND to GND
- + to 3.3V
- DT to D14
- CLK to D13

A rotary encoder has a fixed number of positions per revolution. These positions are easily felt as small “clicks” when you turn the encoder this one has 20 position.

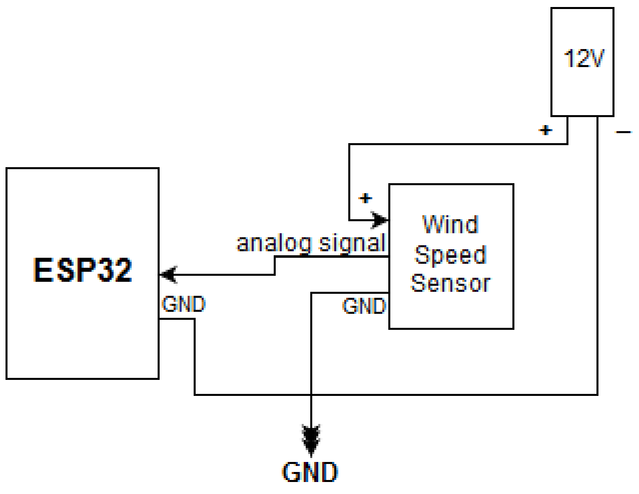


Figure (3.7): Wind Speed Sensor Circuit with ESP32.

As illustrated in figure (3.16) above the wind speed sensor (Anemometer) is wired with 12V external power supply to work properly as for its operational voltage is between (7V-24V). The ESP32, the anemometer and the external power supply should all have a common ground.

The water pressure sensor requires a 5V power supply and provides an out of range (0.5v-4.5v) and the ESP32 chip only provides and processes input voltage up to 3.3V, so it is necessary to use a voltage divider circuit as shown in figure (3.17).

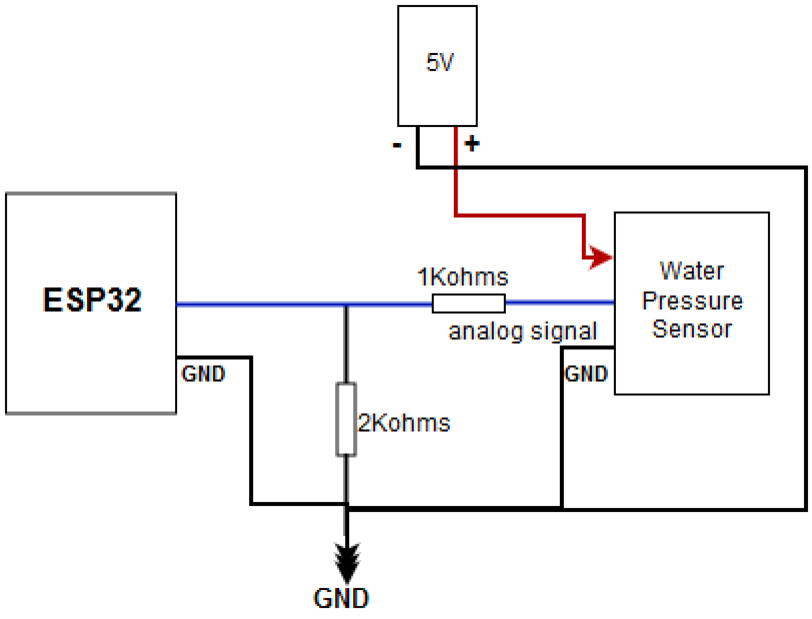


Figure (3.8): Water Pressure Sensor Circuit with ESP32

The purpose of the voltage divider is to scale down the 0-5V analog signal form the sensor to 0-3.3V analog signal which is compatible with the ESP32 input signals. The resistors used in the voltage divider are calculated through the following equations:

$$V_{out} = V_{in} * \frac{R2}{R1+R2} \qquad \text{---} \ (3.1)$$

Where:

V_{out}= Output voltage. This is the scaled down voltage.

V_{in}= Input voltage.

R_1 and R_2 = Resistor values.

The ratio $\frac{R2}{R1+R2}$ determines the scale factor.

According to the required scale down (5V to 3.3V) the resistors utilized in this voltage divider are 1Kohms for R_1 and 2Kohms for R_2.

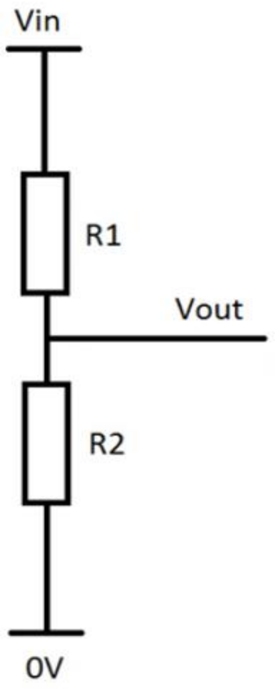


Figure (3.9): Voltage Divider Circuit.

3.3.3 The Potentiometer Circuit

The usage of the potentiometers instead of wind speed sensor and water in this place is because they are not available to be applied on the prototype. The potentiometer output is analog signal which is similar to the output of the two sensors, it should be wire with the power pin of the esp32 and the common ground of circuit, the third pin is data pin which provides analog signal (voltage) to act as sensors input. This circuit simulate the operation of the two unavailable sensors.

Figure (3.10): The Potentiometer

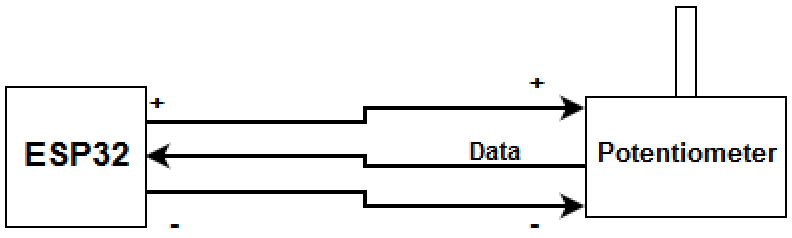


Figure (3.11): The Potentiometer Circuit.

3.4 Software of the System

The required libraries and software packages for ESP32 and the attached sensors were installed on the Arduino IDE software.

The software of the esp32 program will handle each sensor as follow:

- For the 2 channel relay module the software reads the header index of the page which changes after pressing any of the buttons, according to the header state the code controls the signals sent to the control pins of each relay to perform a control function. The buttons on the page change from ON to OFF button and vice versa according to the state of the variables which changes after pressing each button.

- The temperature and humidity sensor has a library on Arduino ide. Set of commands are predefined to acquire the sensor readings.
- The code for the rotary encoder will display the position of the center pivot irrigation system on the page. The algorism used to measure the position utilizes the output of DT and CLK pins (A and B) on the rotary encoder to detect that the rotor shaft is rotating and to which direction (positive or negative).
- The water pressure sensor and wind speed sensor work on special pins (ADC) on the ESP32 board which can process the analog input signals (36 and 39 pin), It's very important to note that, the ESP32 ADCs have 12bits of resolution, so the total range of ADCs reading go to 4,095 when a maximum of 3.3V is applied to its inputs. The commands will measure the voltage output from each sensor and save it as digital number varies for 0 to 4095, each number represents a certain value of the sensor readings according to the range of the sensor. The ADC output is converted through liner equation to display sensible number to readers on the page.
- The code for the potentiometer reads the input signal and convert the number from digital to suitable amount to be shown in the website page.

To upload the code on the esp32 memory, a USB cable connection between the esp32 and the computer is required. The boot button should be hold pressed before clicking the upload button on the Arduino ide software. After the coded is successfully uploaded, enable (EN) button on the esp32 chip should be pressed to initialize the system.

Chapter Four

Results and Discussion

This chapter discusses the results and demonstrate the system limitations.

4.1 Results and discussion

A prototype is made to test the functionality of the system. Two leds are used to represent the 2 channel relay module and two potentiometer are used instead of the wind speed sensor and the water pressure sensor see figures (4.1) and (4.2).

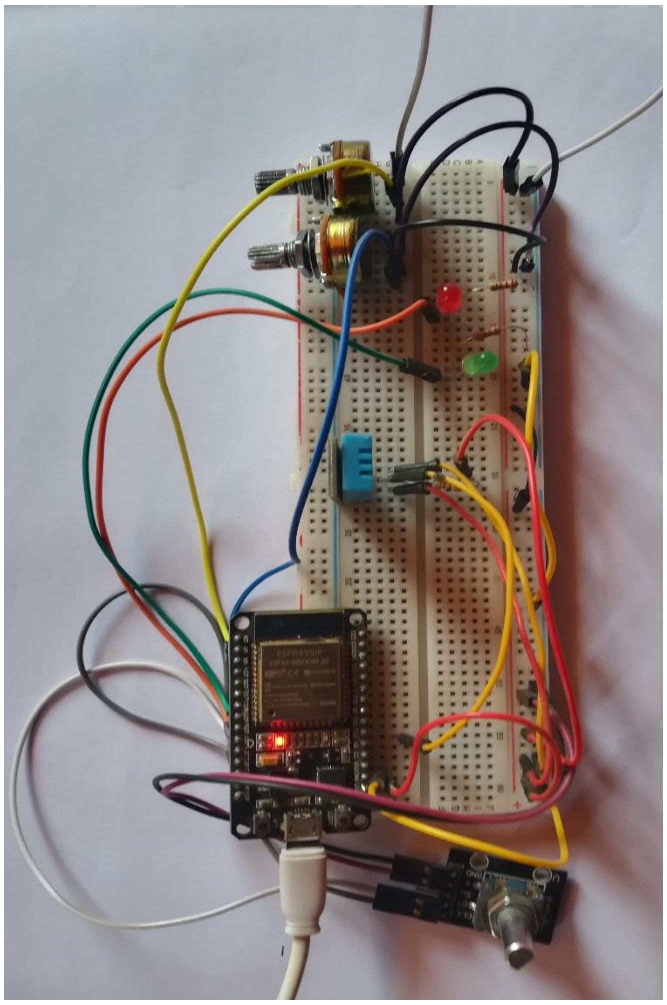


Figure (4.1): A prototype of the system.

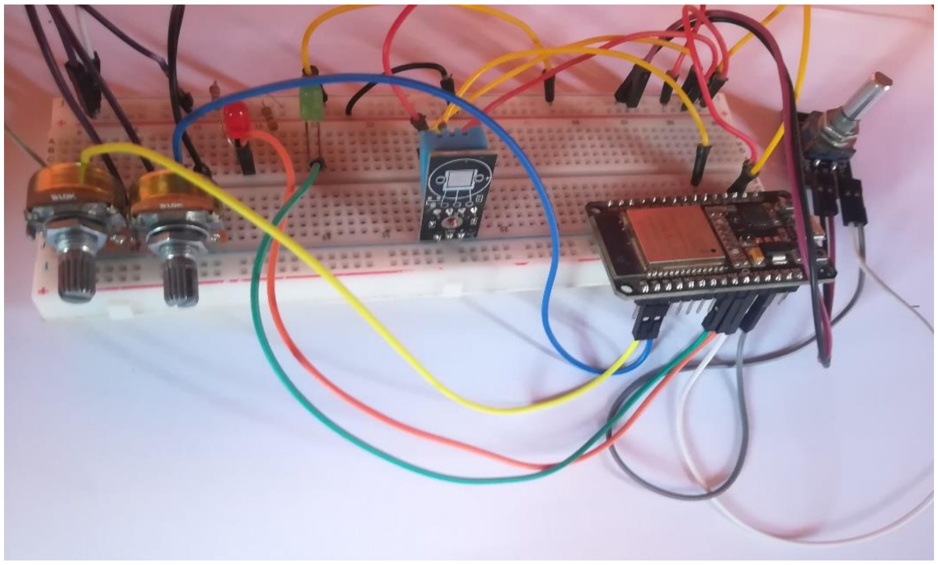


Figure (4.2): different angle of the prototype of the system.

4.2 Testing the Control Functionality of the System

The system consist of two buttons to control the operation of the whole system and running or stopping the pump separately.

4.2.1 Testing the Control Functions

As shown in figure (4.3) the state of the system and the pump is OFF and the two buttons are ON-buttons, after pressing the system ON button (the first button) the two leds will turn on see figure (4.4) and the state of the system the pump will change to ON.

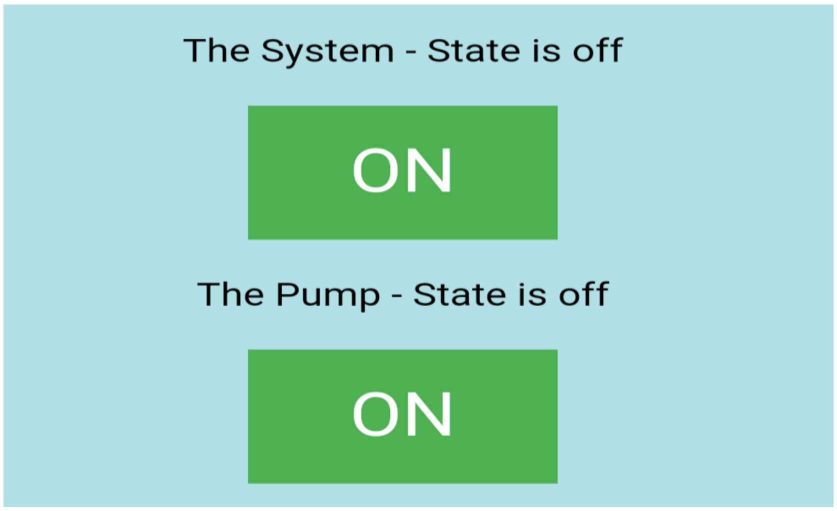


Figure (4.3): The pump and system buttons on the web page.

The two ON-buttons will be replaced with two OFF-buttons on the website page see figure (4.5).

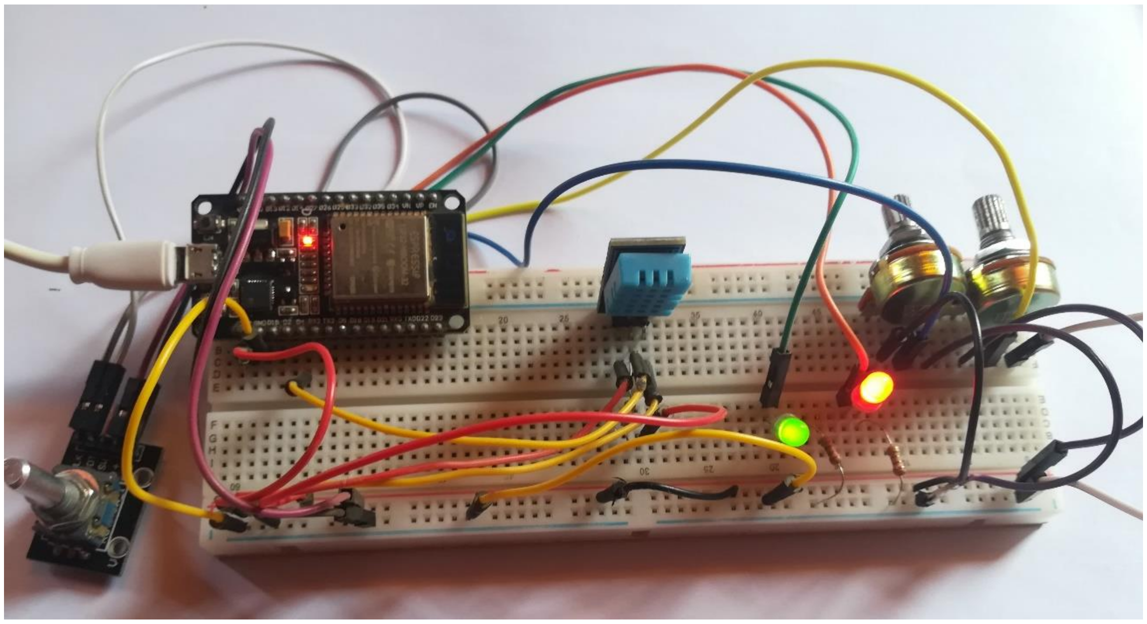
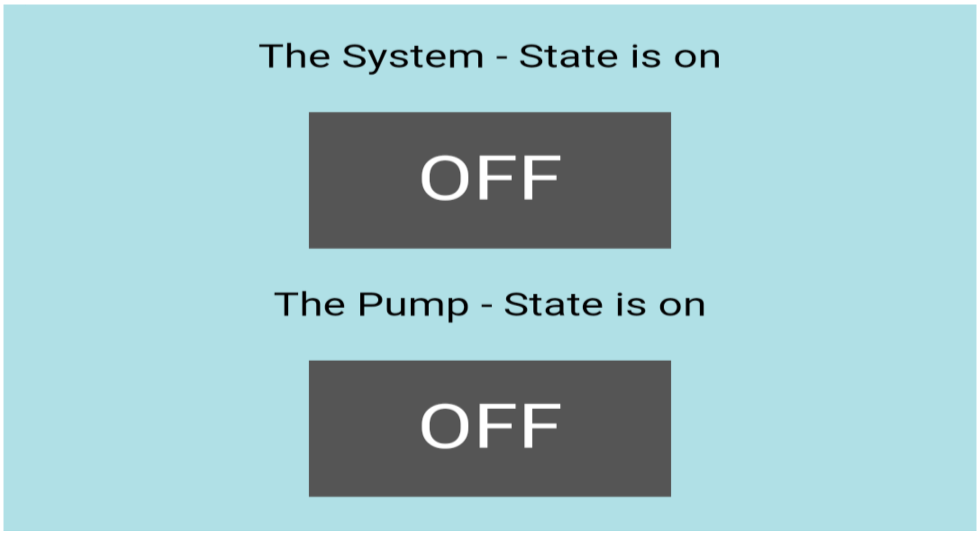


Figure (4.4): The two leds are ON after pressing the system button.



The System - State is on

The Pump - State is on

Figure (4.5): The two OFF-buttons.

When pressing the pump OFF-button the system operates in the dry mode, the red led (relay of the pump control) goes off and the green led (relay of the system control) stays ON see figure (4.6). The pump state changes to OFF, the OFF-button of the pump changes to ON-button see figure (4.7).

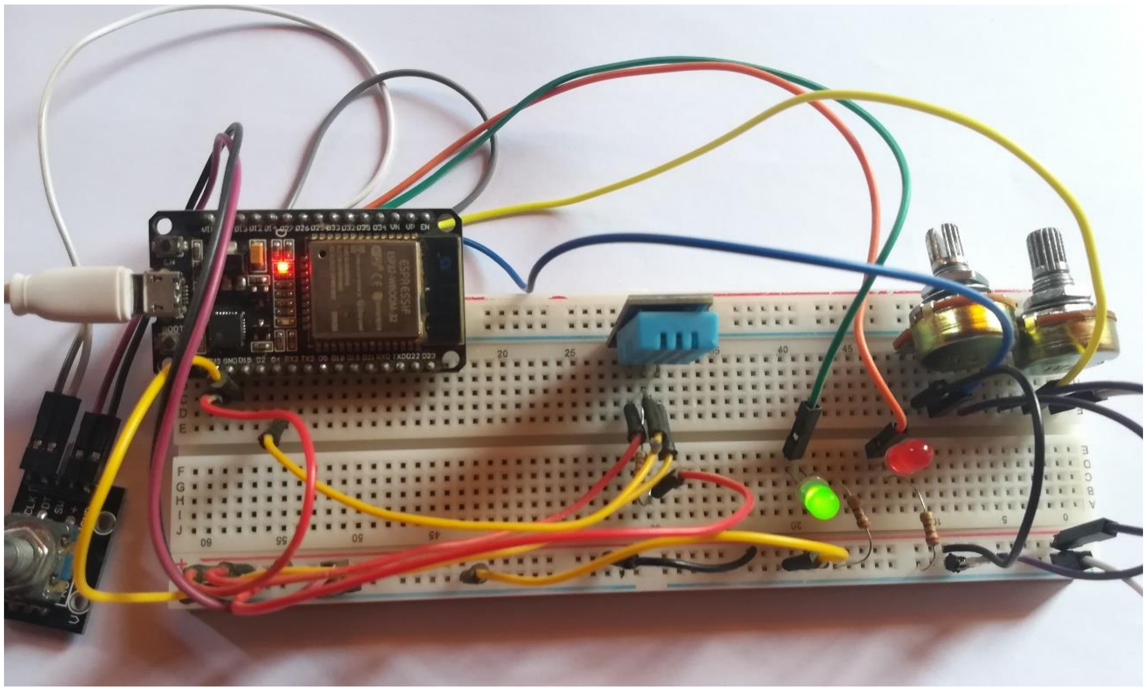
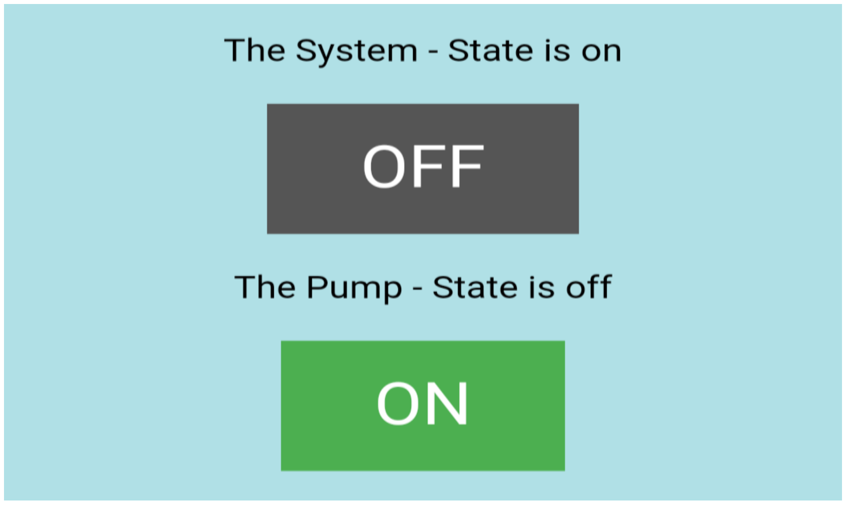


Figure (4.6): The system in dry mode.


The System - State is on

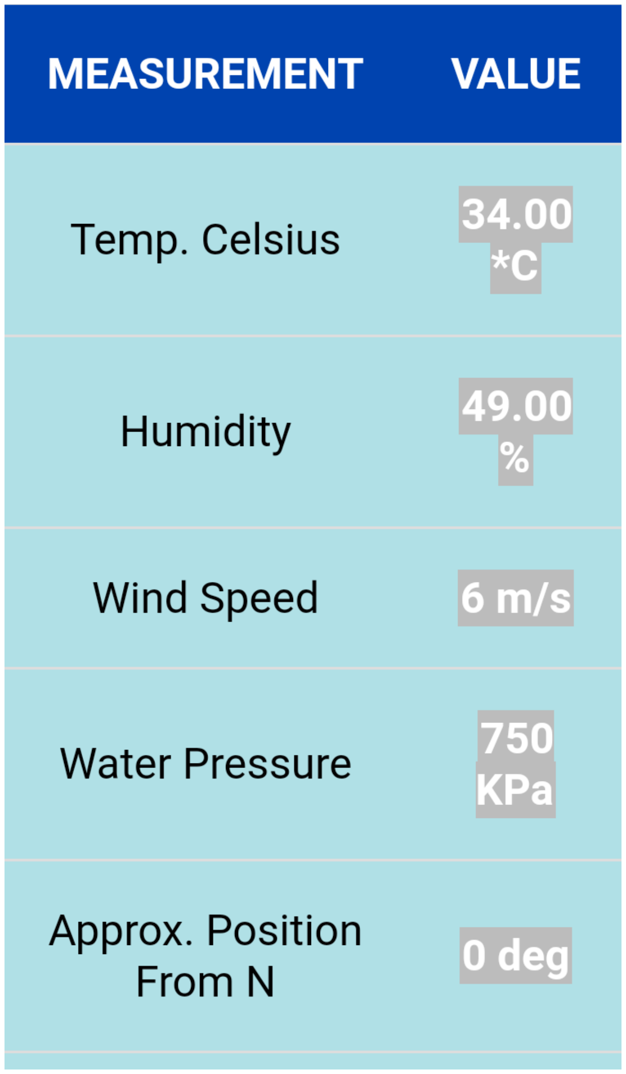
The Pump - State is off

Figure (4.7): System web page in dry mode.

4.3 The Monitoring Function of the System

The system consists of temperature and humidity sensor, wind speed sensor, water pressure sensor and angular position sensor which all preform the monitoring function.

The web page will display a table containing all sensors reading as it is detected from the field as shown in figure (4.8), the webpage should be reloaded to display the changes on the sensors readings.



MEASUREMENT	VALUE
Temp. Celsius	34.00 *C
Humidity	49.00 %
Wind Speed	6 m/s
Water Pressure	750 KPa
Approx. Position From N	0 deg

Figure (4.8): Sensors readings table.

4.3.1 The Temperature and Humidity Sensor Operation

The DHT11 sensor displays ambient temperature measured in Celsius, and ambient humidity in percentage see figure (4.8).

4.3.2 The Wind Speed and the Water Pressure Sensors Simulation

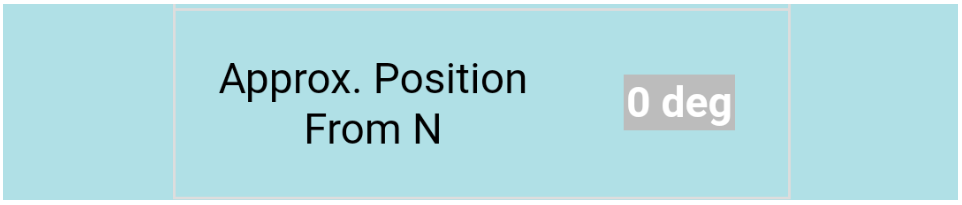
The pressure sensor and wind speed sensor are not available to be tested by this prototype, two potentiometers are used to simulate the operation of the two sensors. The output of the two sensors is analog (voltage) similar to the sensors output. The input data to the ESP32 from each potentiometer is represented on the page as sensor output, the pressure is displayed in Kilo-Pascal (KPa), and wind speed is illustrated in Meter-per-Second (m/s) as shown in figure (4.8). The circuit and the code of water pressure sensor and wind speed sensor are included in the research.

4.3.3 The Rotary Encoder Operation

The rotary encoder is showing the position of the system measured as angel taking north as reference (0 degree). When the pivot starts to rotate clockwise going to the east (90 degree) the angel is displayed as positive angel, on the other hand when the pivot rotates counter-clockwise from the north direction, the angel on will be displayed as negative angel to the west (-90 degree) direction. The angels representing the main directions will be as follow:

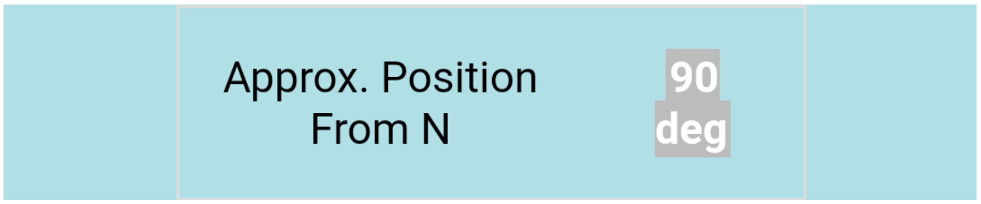
- 0 or 360 degree representing north.
- 90 or -270 degree representing east.
- 180 or -180 degree representing south.
- 270 or -90 degree representing west.

When the pivot direction changes, it is required to refresh the web page to display the new direction of the pivot on the field.


Approx. Position From N

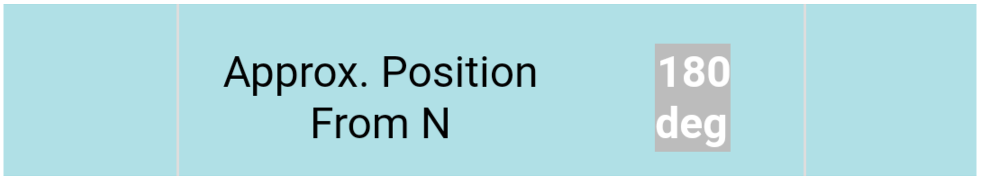
0 deg

Figure (4.9): Zero degree direction (North).


Approx. Position From N

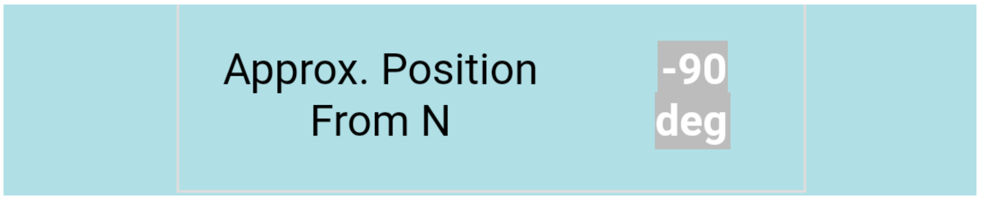
90 deg

Figure (4.10): 90 degree direction (East).


Approx. Position From N

180 deg

Figure (4.11): 180 degree direction (South).

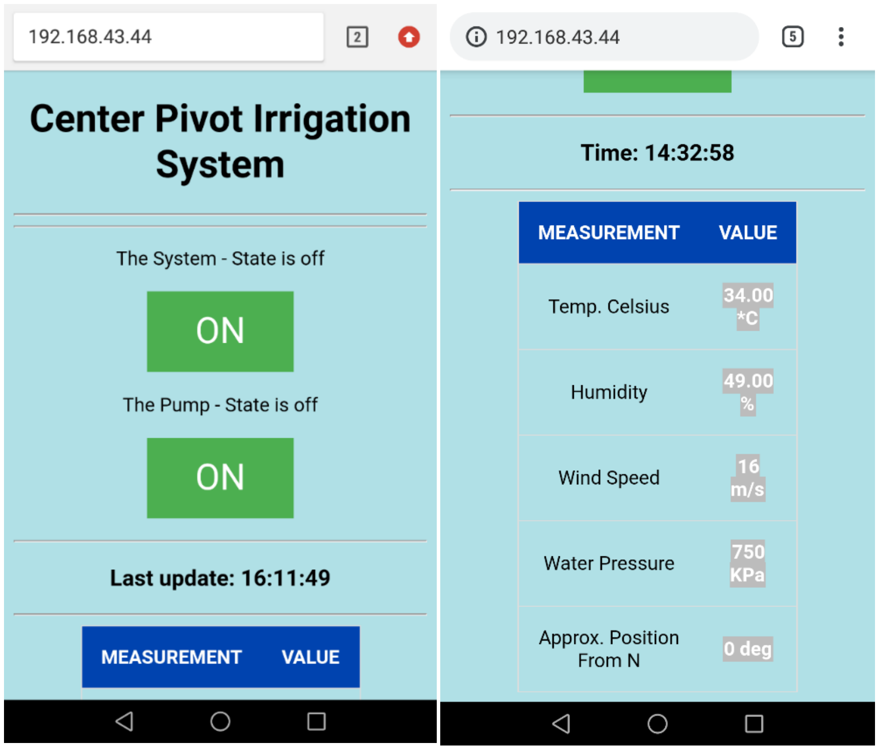

Approx. Position From N

-90 deg

Figure (4.12): -90 degree direction (West).

4.4 The Website Page of the System

The ESP32 connects to Wi-Fi and generate an IP representing its location on the network which gives an access to the web page. The web page is configured to suit different sizes of screens (mobiles or PCs).



192.168.43.44

Center Pivot Irrigation System

The System - State is off

ON

The Pump - State is off

ON

Last update: 16:11:49

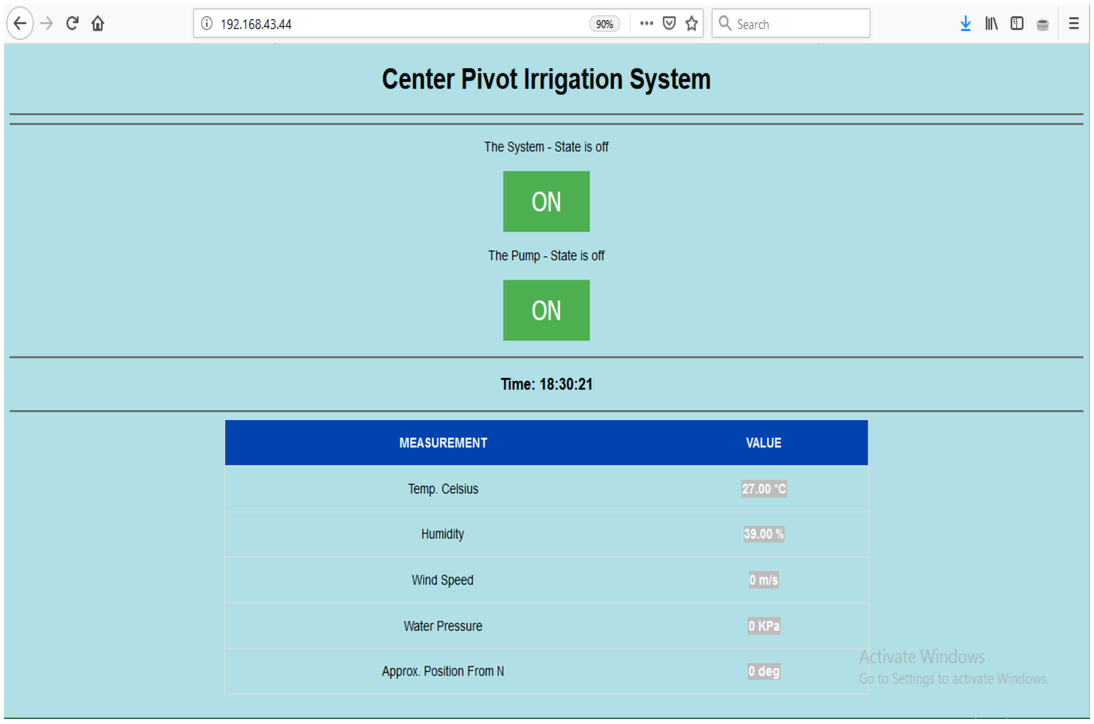
MEASUREMENT	VALUE

192.168.43.44

Time: 14:32:58

MEASUREMENT	VALUE
Temp. Celsius	34.00 *C
Humidity	49.00 %
Wind Speed	16 m/s
Water Pressure	750 KPa
Approx. Position From N	0 deg

Figure (4.13): The web page on phone screen.



192.168.43.44

Center Pivot Irrigation System

The System - State is off

ON

The Pump - State is off

ON

Time: 18:30:21

MEASUREMENT	VALUE
Temp. Celsius	27.00 *C
Humidity	39.00 %
Wind Speed	0 m/s
Water Pressure	0 KPa
Approx. Position From N	0 deg

Activate Windows
Go to Settings to activate Windows.

Figure (4.14): The web page on computer screen.

Chapter Five

Conclusion and Recommendations

5.1 Conclusion

In this thesis, a control and monitoring system for center pivot irrigation system was accomplished based on some of the internet of things technology including, 2 channel relay module to control the main control circuit and the operation of the pump, DHT11 sensor to determine the ambient temperature and humidity of the field, water pressure sensor to detect the water pressure in the main pipeline, wind speed sensor to measure the speed of the wind in the field, and rotary encoder to determine the approximate position of the pivot taking north direction as reference. A prototype and web page has been implemented and the system functions was verified through this hardware prototype and designed web page. This prototype has properly shown that these control and monitoring functions are applicable in real time field.

5.2 Recommendations

For further development of this system the following recommendation is considerable:

1. Implement control function to move the pivot to certain position.
2. Develop a dedicated website and server to improve the efficiency.
3. Add warning function to the system to warn the client about dangerous situations when sensor readings reach certain level.
4. Implement water control function to control the consumption and water appliance and distribution on the field.
5. Develop security measures for the web server page.

Reference

1. "National Engineering Handbook. U.S. Dept. of Agriculture", Natural Resources Conservation Service, 1997.
2. "Growing Rice Where it has Never Grown Before: A Missouri research program may help better feed an increasingly hungry world". College of Agriculture, Food and Natural Resources, University of Missouri. July 3, 2008. Retrieved June 6, 2012.
3. Dan Berne, "Agricultural Irrigation Initiative: Overview of Center Pivot Irrigation Systems", Northwest Energy Efficiency Alliance, 2015.
4. "Handbook on Pressurized Irrigation Techniques", Food And Agriculture Organization Of The United Nations, Rome, 2007.
5. http://www.valleyirrigation.com, December 2018.
6. https://www.cgtrader.com/3d-models/industrial/other/central-pivot, December 2018.
7. http://www.Irrigation.eduction/, December 2018.
8. Kriti Taneja, Sanmeet Bhatia, "Automatic irrigation system using Arduino UNO", IEEE, 2017.
9. P Rajalakshmi, S. Devi Mahalakshmi, "IOT based crop-field monitoring and irrigation automation", IEEE, 2016.
10. Hajir Osman, "A fully Automated Center-Pivot Irrigation System Using a Microcontroller and a Solar Energy System", 2017.
11. R. Troy Peters, Steven R. Evett, "Automation of a Center Pivot Using the Temperature-Time-Threshold Method of Irrigation Scheduling", Journal of Irrigation and Drainage Engineering, June 2008.
12. Yunseop Kim, Robert G. Evans, William M. Iversen, "Remote Sensing and Control of an Irrigation System Using a Distributed Wireless Sensor Network", IEEE, May 2008.
13. F. Viani, F. Robol, M. Bertolli, A. Polo, A. Massa, H. Ahmadi, R. Boualleague, "A wireless monitoring system for phytosanitary treatment in smart farming applications", Antennas and Propagation (APSURSI) 2016 IEEE International Symposium on, pp. 2001-2002, 2016.

14. Olivier Debauche, Meryem El Moulat, Saïd Mahmoudi, Pierre Manneback, Frédéric Lebeau, "Irrigation pivot-center connected at low cost for the reduction of crop water requirements" , IEEE, 2018.
15. Xin Dong, Mehmet C. Vuran, Suat Irmak, "Autonomous precision agriculture through integration of wireless underground sensor networks with center pivot irrigation systems", Ad Hoc Networks, Volume 11, Issue 7, September 2013, Pages 1975-1987.
16. http://katgates.com/assets/uploads/files/412_ARDUINO_WIFI_ESP32E-8266.pdf, December 2018.
17. "ESP32-WROOM-32 Datasheet", Espressif Systems, 2018.
18. "ESP32 Hardware Design Guidelines", Espressif Systems, 2017.
19. "Henry's Bench, KY040 Arduino Tutorial", Schematics and more, 2015.
20. https://blog.squix.org/2016/05/esp8266-peripherals-ky-040-rotary-encoder.html, December 2018.
21. https://howtomechatronics.com/tutorials/arduino/rotary-encoder-works-use-arduino, December 2018.
22. https://www.leniwiec.org/en/2014/04/28/rotary-shaft-encoder-how-to-connect-it-and-handle-it-with-avr-atmega8-16-32-168-328/, December 2018.
23. lady ada, "DHT11, DHT22 and AM2302 Sensors", Adafruit Industries, 2018.
24. https://robu.in/product/dht11-digital-relative-humidity-temperature-sensor-module/, December 2018.
25. https://www.gotronic.fr/art-capteur-de-pression-sen0257-27841.htm, December 2018.
26. https://www.dfrobot.com/wiki/index.php/Gravity:_Water_Pressure_Sensor_SKU:_SEN0257, December 2018.
27. https://www.adafruit.com/product/1733, December 2018.
28. https://www.amazon.es/Shengjuanfeng-Velocidad-anem%C3%B3metro-Aluminio-Sensores/dp/B08DNT34QH/ref=sr_1_8?dchild=1&keywords=arduino+sensor+viento&qid=1600795074&s=electronics&sr=1-8
29. "Relay Module 2-Channel, Datasheet", MICROBOT di Prosseda Mirko, Code: MR009-004.2.
30. https://www.arduino.cc/en/Main/Software, December 2018.

Appendix A: ESP32 code

```
#include <WiFi.h>   // Load Wi-Fi library
#include "DHT.h"     // load DHT library
#define DHTTYPE DHT11
const char* ssid = "yasir";          // network credentials
const char* password = "yasir12345";
WiFiServer server(80);         // Set web server port number to 80
uint8_t DHTPin = 4;       // DHT Sensor
DHT dht(DHTPin, DHTTYPE);             // Initialize DHT sensor.
float Temperature;           float Humidity;        // Initialize DHT varibles.
String header;          // Variable to store the HTTP request
// Auxiliar variables to store the current output state
String output26State = "off";
String output27State = "off";
// Assign output variables of relay to GPIO pins
const int output26 = 26;
const int output27 = 27;
// Rotary encoder pins and variables.
 const int outputA = 13;
 const int outputB = 14;
int counter = 0;
int aState;
int aLastState;
void setup() {
  // Rotary encoder pins setup
  pinMode(outputA, INPUT);
 pinMode(outputB, INPUT);
Serial.begin(115200);
 aLastState = digitalRead(outputA);
 // DHT pin setup
pinMode(DHTPin, INPUT);
  dht.begin();
// Initialize the output variables of relay as outputs
pinMode(output26, OUTPUT);
pinMode(output27, OUTPUT);
// Set outputs to LOW
digitalWrite(output26, LOW);
digitalWrite(output27, LOW);
// Connect to Wi-Fi network with SSID and password
Serial.print("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");    }
// Print local IP address and start web server
Serial.println("");
Serial.println("WiFi connected.");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
server.begin();         }
// the position measuring function.
int routary(){
aState = digitalRead(outputA);
 if (aState != aLastState){
  if (digitalRead(outputB) != aState){
    counter = counter-9;
    if (counter<=-360){counter=0;}
    }else{  if (counter>=360){counter=0;}
      counter = counter+9;             }
  }aLastState =aState;return counter;    }
  int windSpeed(){      // wind speed function
int sensorValue = analogRead(39);
   float outvoltage = sensorValue * (2 / 4095.0);
   int wSpeed = 16*outvoltage;
   //The level of wind speed is proportional to the output voltage.
    return wSpeed;    }
int waterPressure(){                // water pressure function
  const float  OffSet = 0.483 ;        float Vp, P;
  Vp = analogRead(36) * 3.30 / 4095;     //Sensor output voltage
  P = (Vp - OffSet) * 500;             //Calculate water pressure
P=0;    return P;       }
void loop(){
WiFiClient client = server.available(); // Listen for incoming clients
if (client) { // If a new client connects,
Serial.println("New Client."); // print a message out in the serial port
String currentLine = ""; // make a String to hold incoming data from the client
while (client.connected()) { // loop while the client's connected
  routary();              // measuring the position
  aLastState =aState;
  //measuring water pressure and wind speed
  waterPressure();     windSpeed();
if (client.available()) { // if there's bytes to read from the client,
char c = client.read(); // read a byte, then
Serial.write(c); // print it out the serial monitor
header += c;
if (c == '\n') { // if the byte is a newline character
// if the current line is blank, you got two newline characters in a row.
// that's the end of the client HTTP request, so send a response:
if (currentLine.length() == 0) {
// HTTP headers always start with a response code (e.g.HTTP/1.1 200 OK)
// and a content-type so the client knows what's coming, then a blank line:
client.println("HTTP/1.1 200 OK");
client.println("Content-type:text/html");
client.println("Connection: keep-alive");         client.println();
//DHT readinds
Temperature = dht.readTemperature(); // Gets the values of the temperature
  Humidity = dht.readHumidity(); // Gets the values of the humidity
// turns the Relay on and off
if (header.indexOf("GET /26/on") >= 0) {
Serial.println("GPIO 26 on");
output26State = "on";
digitalWrite(output26, HIGH);
Serial.println("GPIO 27 on");
output27State = "on";
digitalWrite(output27, HIGH);
} else if (header.indexOf("GET /26/off") >= 0) {
Serial.println("GPIO 26 off");
output26State = "off";
digitalWrite(output26, LOW);
Serial.println("GPIO 27 off");
output27State = "off";
digitalWrite(output27, LOW);
} else if (header.indexOf("GET /27/on") >= 0) {
Serial.println("GPIO 27 on");
output27State = "on";
digitalWrite(output27, HIGH);
} else if (header.indexOf("GET /27/off") >= 0) {
Serial.println("GPIO 27 off");
output27State = "off";
digitalWrite(output27, LOW);    }
// Display the HTML web page
```

```
client.println("<!DOCTYPE html><html>");
client.println("<head><meta name=\"viewport\"content=\"width=device-width, initial-scale=1\">");
client.println("<link rel=\"icon\" href=\"data:,\">");
// CSS to style the on/off buttons
// Feel free to change the background-color and font-size attributes to fit your preferences
client.println("<script type=\"text/javascript\">\n"
"function startTime()\n"
"{\n"
"var today=new Date();\n"
"var h=today.getHours();\n"
"var m=today.getMinutes();\n"
"var s=today.getSeconds();\n"
"// add a zero in front of numbers<10\n"
"m=checkTime(m);\n"
"s=checkTime(s);\n"
"document.getElementById('txt').innerHTML=h+\":\"+m+\":\"+s;\n"
"t=setTimeout('startTime()',500);\n"
"}\n"
"function checkTime(i)\n"
"{\n"
"if (i<10)\n"
"{\n"
"i=\"0\" + i;\n"
"}\n"
"return i;\n"
"}\n"
"</script>");
client.println("<style>html { font-family: Helvetica; display:inline-block; margin: 0px auto; text-align: center;}");
client.println("table {border-collapse: collapse;width:60%;margin-left:auto; margin-right:auto;}");
client.println("th { padding: 16px; background-color: #0043af; color: white;}");
client.println("tr { border: 1px solid #ddd; padding: 16px;}");
client.println("td { border: none; padding: 16px; }");
client.println(".sensor { color:white; font-weight: bold; background-color: #bcbcbc; padding: 1px; }");
client.println(".button { background-color: #4CAF50; border:none; color: white; padding: 16px 40px;");
client.println("text-decoration: none; font-size: 30px; margin:2px; cursor: pointer;}");
client.println(".button2 {background-color:#555555;}</style></head>");
// Web Page Heading
client.println("<body onload=\"startTime()\" style=\"background-color:powderblue;\"><h1>Center Pivot
Irrigation System</h1>");
// Display current state, and ON/OFF buttons for GPIO 26
client.println("<hr>");
client.println("<hr>");
client.println("<p>The System - State is " + output26State + "</p>");
// If the output26State is off, it displays the ON button
if (output26State=="off") {
client.println("<p><a href=\"/26/on\"><button class=\"button\">ON</button></a></p>");  } else {
client.println("<p><a href=\"/26/off\"><button class=\"button button2\">OFF</button></a></p>");  }
// Display current state, and ON/OFF buttons for GPIO 27
client.println("<p>The Pump - State is " + output27State + "</p>");
// If the output27State is off, it displays the ON button
if (output27State=="off") {
client.println("<p><a href=\"/27/on\"><button class=\"button\">ON</button></a></p>");  } else {
client.println("<p><a href=\"/27/off\"><button class=\"button button2\">OFF</button></a></p>");  }
client.println("<hr>");
client.println("<h3>Time: <span id=\"txt\"></span></h3>");
client.println("<hr>");
client.println("<table><tr><th>MEASUREMENT</th><th>VALUE</th></tr>");
client.println("<tr><td>Temp. Celsius</td><td><span class=\"sensor\">");
client.println(dht.readTemperature());
client.println(" *C</span></td></tr>");
client.println("<tr><td>Humidity</td><td><span class=\"sensor\">");
client.print(dht.readHumidity());
client.println(" %</span></td></tr>");
client.println("<tr><td>Wind Speed</td><td><span class=\"sensor\">");
client.print(windSpeed());
client.println(" m/s</span></td></tr>");
client.println("<tr><td>Water Pressure</td><td><span class=\"sensor\">");
client.print(waterPressure());
client.println(" KPa</span></td></tr>");
client.println("<tr><td>Approx. Position From N</td><td><span class=\"sensor\">");
client.print(routary());
client.println(" deg</span></td></tr>");
client.println("</body></html>");
// The HTTP response ends with another blank line
client.println();
break;    // Break out of the while loop
} else { // if you got a newline, then clear currentLine
currentLine = "";   }
} else if (c != '\r') { // if you got anything else but a carriage return character,
currentLine += c; }}}     // add it to the end of the currentLine
header = "";        // Clear the header variable
client.stop();   // Close the connection
Serial.println("Client disconnected.");
Serial.println("");  }   }
```