IOT BASED IRRIGATION SYSTEM

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE

OF

BACHELOR IN TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by**:**

ESHANT SAGAR (2K14/EC/067)

HAMAAD ULLAH ZUBERI (2K14/EC/074)

PARSHANT KUMAR (2K14/EC/111)

PAWAN KUMAR (2K14/EC/112)

Under the supervision of**:**

Mr. DEVA NAND (ASSISTANT PROFESSOR)

DEPARTMENT OF ELECTRONICS AND COMMUNICATIONS ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

BAWANA ROAD, DELHI – 110042

(MAY 2018)

ELECTRONICS AND COMMUNICATION ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

BAWANA ROAD, DELHI-110042

CANDIDATE’S DECLARATION

We, Eshant Sagar (2K14/EC/067), Hamaad Ullah Zuberi (2K14/EC/074), Parshant Kumar (2K14/EC/111), Pawan Kumar (2K14/EC/112) students of B.Tech. (Electronics and Communications Engineering), hereby declare that the project Dissertation titled "**IoT Based Irrigation System**" which is submitted by us to the Department of Electronics and Communication Engineering, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

DATE:

ESHANT SAGAR	HAMAAD ULLAH ZUBERI
(2K14/EC/067)	(2K14/EC/074)
PARSHANT KUMAR	PAWAN KUMAR
(2K14/EC/111)	(2K14/EC/112)

ELECTRONICS AND COMMUNICATION ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

BAWANA ROAD, DELHI-110042

CERTIFICATE

We hereby certify that the Project Dissertation titled “**IoT Based Irrigation System**” which is submitted by Eshant Sagar (2K14/EC/067), Hamaad Ullah Zuberi (2K14/EC/074), Parshant Kumar (2K14/EC/111), Pawan Kumar (2K14/EC/112) [Electronics and Communications Engineering], Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology, is a record of the project work carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

DATE:

DEVA NAND
Assistant Professor
Department of Electronics and Communications Engineering
Delhi Technological University
Bawana Road, Delhi - 110042

ELECTRONICS AND COMMUNICATION ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

BAWANA ROAD, DELHI-110042

ACKNOWLEDGEMENT

The successful completion of any task would be incomplete without acknowledging the people who made it all possible and whose constant guidance and encouragement secured us the success.

First of all, we are grateful to Dr. S. Indu, Head of Department, Electronics and Communications Engineering, Delhi Technological University and all other faculty members of our department, for their astute guidance, constant encouragement and sincere support for this project work. We owe a debt of gratitude to our mentor, Mr. Deva Nand (Assistant Professor, Department of Electronics and Communications Engineering) for incorporating in us the idea of a creative Major Project, for helping us in undertaking this project and also for being there whenever we needed his assistance.

We would also like to place on record, our sense of gratitude to one and all, who directly or indirectly have lent their helping hand in this venture. We feel proud and privileged in expressing our deep sense of gratitude to all those who have helped us in presenting this project. Last but never the least, we thank our parents for always being with us, in every sense.

ESHANT SAGAR
(2K14/EC/067)

HAMAAD ULLAH ZUBERI
(2K14/EC/074)

PARSHANT KUMAR
(2K14/EC/111)

PAWAN KUMAR
(2K14/EC/112)

ABSTRACT

This project uses the concept of IoT for carrying out the task of irrigation. A Wi-Fi module ESP8266-12E ensures Wi-Fi connectivity in this IoT based project. The project model shows two different fields for irrigation each of which uses a solenoid valve for water supply to the field. The solenoid valves and the water pump are controlled through the Wi-Fi module based on the input provided by the user connected to the MQTT server using the MQTT Dash Android App. The user is provided information about the soil moisture level in the field and the temperature by respective sensors which send the data to MQTT server enabling the user to monitor the system and to take the needed action. Arduino Uno is used as an interface processor.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

India is the country with a large percentage of the population working on farms and agriculture thus plays an important role for development of country. In our country, agricultural farming depends on the monsoon which is an insufficient source of water. Due to this, irrigation is used to water fields. To provide the required amount of water, two things are very important to acquire. The information about the fertility of soil and the moisture content in the soil. Nowadays, for irrigation, different techniques are used to reduce the dependency of rain. These techniques are mostly driven by electrical power and on/off scheduling.

Due to the continuously increasing demand and a sharp decrease in supply of food necessities, there is a necessity for a rapid improvement in agriculture technology. Agriculture plays a very important role in the development of the economy like India, where the majority of people still work as farmers. Due to the decreasing levels of rains, Irrigation is the only viable solution to water farms. Irrigation can be defined as the application of controlled amounts of water to plants at needed intervals. Irrigation helps us grow agricultural crops, maintain landscapes, and revegetate disturbed soils in dry areas and during periods of less than average rainfall. It also has other uses in crop production, including frost protection, suppressing weed growth in grain fields and preventing soil consolidation.

The primary purpose of this project is to help the farmers to irrigate their fields from their houses itself, using IoT to continuously check the water content and to adequately provide water when needed so. However, this project is not usable on such a large scale yet. But it can still be used with house gardens and small-scale farms.

CHAPTER 2
INTERNET OF THINGS

2.1. INTERNET OF THINGS

The Internet of Things (IoT) is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data. Each device is uniquely identifiable through its embedded computing system but is able to inter-operate within the existing Internet infrastructure. The IoT allows objects to be sensed or controlled remotely across existing network infrastructure, creating opportunities for a direct integration of the physical world into computer-based systems, and resulting in improvement to efficiency, accuracy and economic benefits in addition to reduced human intervention.

A complete IoT system integrates four distinct components: sensors, connectivity, data processing, and a user interface.

Sensors: A device that used to detect changes in its environment and then sends the information to other devices, like a computer processor. Sensors are always used in combination with others electronics.

Connectivity: Sensors can be connected to the internet using a variety of methods. These can include satellite internet, Wi-Fi or Ethernet. Each option has its own advantages and disadvantages in power consumption, range and bandwidth. The choice about which method to use depends on the task or the application, but they essentially achieve the same purpose: getting data online.

Data Processing: Once the data has been uploaded to the cloud, there is usually a need to process it somehow. It could be as simple has a checking if the temperature is within a required range to as complicated as computer vision.

User Interface: The information has to be made useful for the user. This can be done by sending an alert to the user or making an interface which the user checks regularly. For example, video feeds to a security camera which the user checks regularly via an app. Depending the application, it may be required for the user to perform an action as well. Some actions may require active participation by the user, while others can be performed automatically. An example could be changing the temperature of the room for example. It can be predefined to turn on the cooling device if the temperature reaches a certain limit, or the user may be required to manually turn on the device.

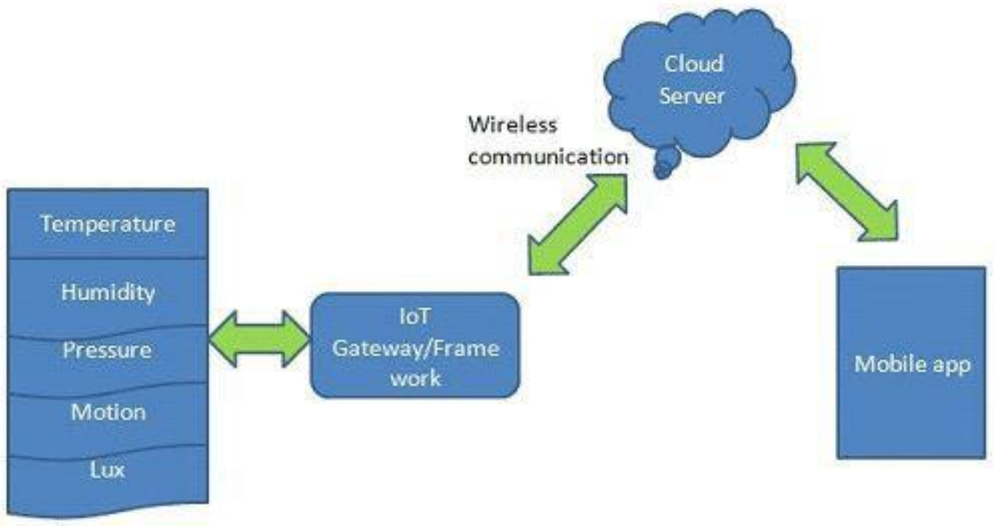


Figure 1. Internet of Things Architecture

Devices need not be connected to Internet independently. A cluster of devices can be created only the cluster can be connected to the network. This will lead to an abstract architecture with different protocols which may range from high level to low level. In addition, these devices can be independently discovered. For this, they required a unique IP address. IoT device use the IPv6 addressing scheme and all devices have either a fixed or subnet masked IP address. Unique addresses help make the IoT devices visible as independent nodes. This is one of the most important concepts in IoT.

2.2. HISTORY

The term "Internet of Things" was coined in 1999 by Kevin Ashton while he was working in Procter & Gamble. But the actual idea has be around for much longer, since at least the 70s, known as "Embedded Internet" or "Pervasive Computing". The concept started to gain popularity in 2010, when people realized that Google's StreetView service stored data about Wi-Fi networks, in addition to its primary purpose of taking 360-degree pictures. Around the same time, the Chinese Government announced it would make IoT a strategic priority in their plans. The term reached its peak and the general public became aware of the concept when Google announced its purchase of Nest for $3.2 Billion and the Consumer Electronics Show in Las Vegas was held under the theme "Internet of Things".

2.3. MQTT

MQTT stands for Message Query Telemetry Transport. It is a lightweight messaging protocol, designed for constrained devices. It uses publish/subscribe method and is mostly used with low-bandwidth, unreliable networks. It focuses on minimising network bandwidth and device resource requirement while attempting reliability. These principles make the protocol ideal for IoT, and for mobile applications where bandwidth and battery life are very important.

MQTT was invented by Dr Andy Stanford-Clark of IBM, and Arlen Nipper of Arcom (now Eurotech), in 1999.

Users can use usernames and passwords with MQTT packets in v3.1. Encryption is now handles with SLL, independently.

2.3.1. MQTT ARCHITECTURE

Messages in MQTT are delivered asynchronously ("push") through a publish subscribe architecture. The protocol works by exchanging a series of control packets in a defined way. The packet is carefully constructed to reduce data transmission and each control packet has a specific purpose. A MQTT topology has Server and a Client. They communicate through different control packets.

Table 1. Control Packet Types

Control Packet	Direction of Flow	Description
CONNECT	Client to Server	Client request to connect to Server.
CONNACK	Server to Client	Connect acknowledgement
PUBLISH	Client to Server or Server to Client	Publish message
PUBACK	Client to Server or Server to Client	Publish acknowledgement
PUBREC	Client to Server or Server to Client	Publish received (assured delivery part 1)
PUBREL	Client to Server or Server to Client	Publish received (assured delivery part 2)
PUBCOMP	Client to Server or Server to Client	Publish received (assured delivery part 3)
SUBSCRIBE	Client to Server	Client subscribe request
SUBACK	Server to Client	Subscribe acknowledgement
UNSUBSCRIBE	Client to Server	Unsubscribe request
UNSUBACK	Server to Client	Unsubscribe acknowledgement
PINGREQ	Client to Server	PNG request
PINGRESP	Server to Client	PING response
DISCONNECT	Client to Server	Client is disconnecting

Control Packet headers are kept as small as possible. Each MQTT Control packet consists of three parts: a fixed header, a variable header and a payload. The Fixed Header is of 2 bytes. Variable Headers and Payloads need not necessarily be a part of the Control Packet. A variable header consists of the packet identifier and the payload up to 256MB could be attached. Having a small header makes this protocol appropriate for IoT.

2.3.2. MQTT PORTS

The server listens on the following ports:

- 1883: MQTT, unencrypted
- 8883: MQTT, encrypted
- 8884: MQTT, encrypted, client certificate required
- 8080: MQTT over WebSockets, unencrypted
- 8081: MQTT over WebSockets, encrypted

This project uses 1883 an unencrypted MQTT port.

CHAPTER 3
HARDWARE

3.1. ARDUINO

Arduino is an open source platform widely used for building electronics projects. It consists of both a physical programmable circuit board, referred to as a microcontroller, and a IDE (Integrated Development Environment) that is used to write and upload the codes to the physical board.

The Arduino platform is very popular for beginners, since it does not need a separate programmer to load new code onto the board. It only requires a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it very easy to learn. Arduino also provides a standard form factor that breaks out the functions of the microcontroller into a more accessible package.

3.1.1. ARDUINO BOARDS

Arduino Boards usually consist of an Atmel 8bit AVR Microcontroller with varying amounts of memory, pins and features. The boards use single or double-row female headers that facilitate connections for programming and incorporation into other circuits. These connect with add on modules known as shields. Multiple stacked shields can be individually addressed via I2C serial bus. Boards may also include a 5V linear regulator and a 16MHz crystal oscillator or ceramic resonator.

Arduino microcontroller's pre-programmed bootloader simplifies the upload of programs to the on-chip flash memory. The default bootloader of the Arduino Uno is the Optiboot Bootloader. Program codes can be uploaded to Boards via serial connection with a Computer. Some Arduino boards may also include a level shifter to convert RS-232

Logic Levels and TTL Level signals. Current boards are programmed via USB, implemented using a USB to Serial adapter chip.

There are many Arduino compatible and Arduino derived boards in the market. Some are functionally identical to Arduino and can be used interchangeably. Some of these also enhance the Arduino Board by adding output drivers to simplify certain functions. Whereas others electrically identical but change the form factor. This may or may not make them incompatible with shields. Variants may also use different processors which may affect the compatibility as well.

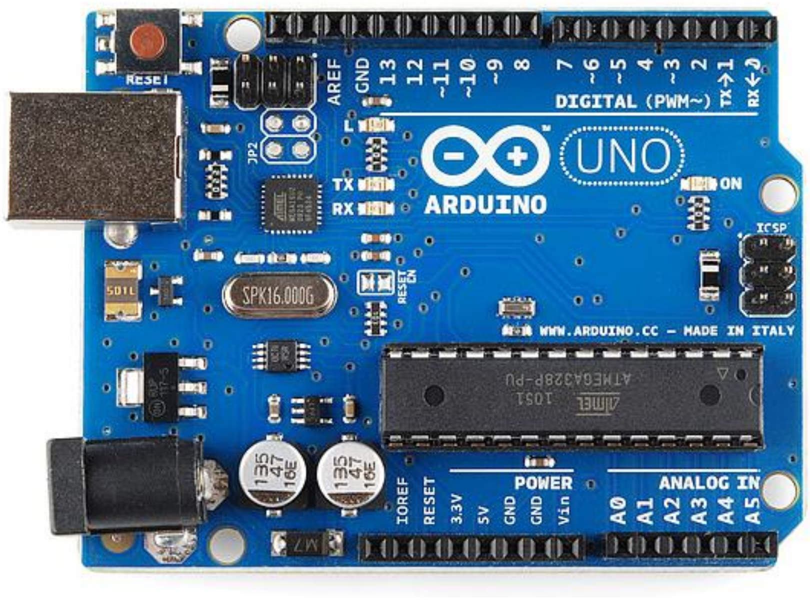


Figure 2. Arduino Uno

3.1.2. ARDUINO UNO

The Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. Each of the 14 digital pins can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default, they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference () function.

Table 2. Specifications of ATmega328P

Microcontroller	**ATmega328P**
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

3.2. ESP8266 WI-FI MODULE

ESP8266 is a complete and self-contained Wi-Fi network solutions that can carry software applications, or through another application processor uninstall all Wi-Fi networking capabilities. ESP8266 when the device is mounted and as the only application of the application processor, the flash memory can be started directly from an external Move. Built-in cache memory will help improve system performance and reduce memory requirements. Another situation is when wireless Internet access assume the task of Wi-Fi adapter, you can add it to any microcontroller-based design, and the connection is simple, just by SPI / SDIO interface or central processor AHB bridge interface. Processing and storage capacity on ESP8266 powerful piece, it can be integrated via GPIO ports sensors and other applications specific equipment to achieve the lowest early in the development and operation of at least occupy system resources. The ESP8266 highly integrated chip, including antenna switch balun, power management converter, so with minimal external circuitry, and includes front-end module, including the entire solution designed to minimize the space occupied by PCB. The system is equipped with ESP8266 manifested leading features are: energy saving VoIP quickly switch between the sleep / wake patterns, with low-power operation adaptive radio bias, front-end signal processing functions, troubleshooting and radio systems coexist characteristics eliminate cellular / Bluetooth / DDR / LVDS / LCD interference.

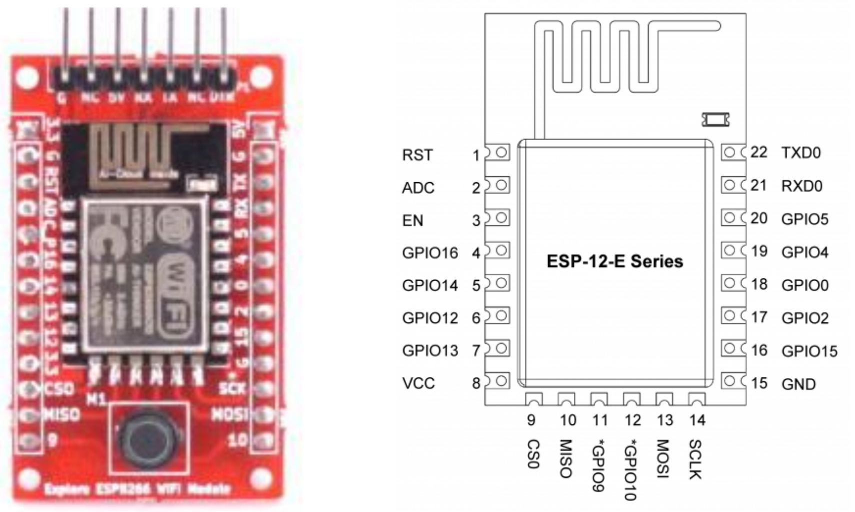


Figure 3. ESP8266-12E Module

Specifications:

- 802.11 b / g / n
- Wi-Fi Direct (P2P), soft-AP
- Built-in TCP / IP protocol stack
- Built-in TR switch, balun, LNA, power amplifier and matching network

- Built-in PLL, voltage regulator and power management components
- 802.11b mode + 19.5dBm output power
- Built-in temperature sensor
- Support antenna diversity
- off leakage current is less than 10uA
- Built-in low-power 32-bit CPU: can double as an application processor
- SDIO 2.0, SPI, UART
- STBC, 1x1 MIMO, 2x1 MIMO
- A-MPDU, A-MSDU aggregation and the 0.4 Within wake
- 2ms, connect and transfer data packets
- standby power consumption of less than 1.0mW (DTIM3)

3.2.1. ESP MODULES

The ESP8266 is a low-cost Wi-Fi chip with full TCP/IP stack and microcontroller capability produced by Shanghai-based Chinese manufacturer, Espressif. These were the first series of modules made by third-party manufacturer, AI-Thinker with the ESP8266 and remain the most widely available.

Table 3. Various Types of ESP Modules

Board ID	**Pins**	**Pitch**	**LEDs**	**Antenna**	**Dimensions (mm)**
ESP-01	8	.1”	Yes	Etched-on PCB	14.3 x 24.8
ESP-02	8	.1”	No	None	14.2 x 14.2
ESP-03	14	2mm	No	Ceramic	17.3 x 12.1
ESP-04	14	2mm	No	None	14.7 x 12.1
ESP-05	5	.1”	No	None	14.2 x 14.2
ESP-06	12+GND	misc	No	None	14.2 x 14.2
ESP-07	16	2mm	Yes	Ceramic	20.0 x 16.0
ESP-08	14	2mm	No	None	17.0 x 16.0
ESP-09	12+GND	misc	No	None	10.0 x 10.0
ESP-10	5	2mmm?	No	None	14.2 x 10.0

ESP-11	8	1.27mm	No	Ceramic	17.3 x 12.1
ESP-12	16	2mm	Yes	Etched-on PCB	24.0 x 16.0
ESP-12-E	22	2mm	Yes	Etched-on PCB	24.0 x 16.0
ESP-13	18	1.5mm	-	Etched-on PCB	24.0 x 16.0
ESP-14	22	2mm	1	Etched-on PCB	24.3 x 16.2
WROOM-02	18	1.5mm	No	Etched on PCB	20.0 x 18.0
WT8266-S1	18	1.5mm	1	Etched on PCB	15.0 x 18.6

Applications:

- Smart Power Plug
- Home Automation
- Industrial wireless control
- Baby Monitor
- Network Camera
- Wireless location-aware devices and positioning system signals

3.2.2. EXPLORE ESP8266-12E

ESP-12E Wi-Fi module is developed by Ai-thinker Team. core processor ESP8266 in smaller sizes of the module encapsulates Tensilica L106 integrates industry-leading ultra-low power 32-bit MCU micro, with the 16-bit short mode, Clock speed support 80 MHz, 160 MHz, supports the RTOS, integrated Wi-Fi MAC/BB/RF/PA/LNA, on-board antenna. The module supports standard IEEE802.11 b/g/n agreement, complete TCP/IP protocol stack. Users can use the add modules to an existing device networking or building a separate network controller. ESP8266 is high integration wireless SOCs, designed for space and power constrained mobile platform designers. It provides unsurpassed ability to embed Wi-Fi capabilities within other systems, or to function as a standalone application, with the lowest cost, and minimal space requirement.

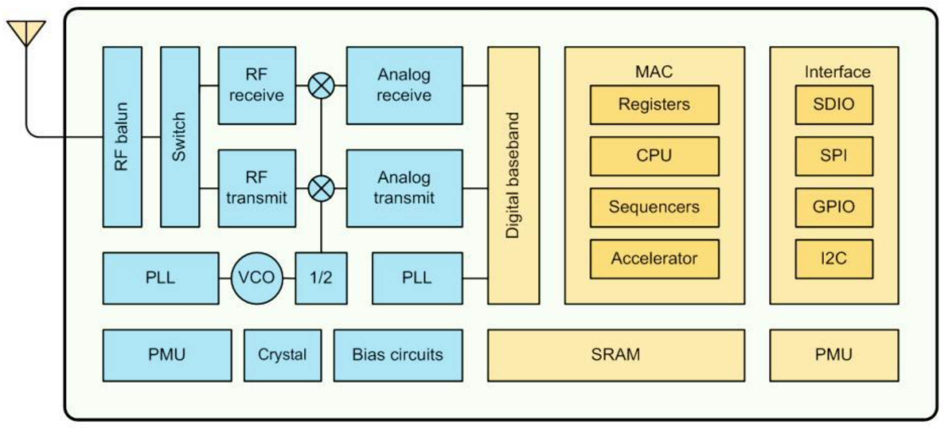


Figure 4. ESP8266EX Block Diagram

Features

- 802.11 b/g/n
- Integrated low power 32-bit MCU
- Integrated 10-bit ADC
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLL, regulators, and power management units
- Supports antenna diversity
- Wi-Fi 2.4 GHz, support WPA/WPA2
- Support STA/AP/STA+AP operation modes
- Support Smart Link Function for both Android and iOS devices
- Support Smart Link Function for both Android and iOS devices
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IRDA, PWM, GPIO
- STBC, 1x1 MIMO, 2x1 MIMO
- A-MPDU & A-MSDU aggregation and 0.4s guard interval
- Deep sleep power <10uA, Power down leakage current < 5uA
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)
- +20dBm output power in 802.11b mode
- Operating temperature range -40C ~ 125C

3.2.3. AT COMMANDS

ESP8266, in its default configuration, boots up into the serial modem mode. In this mode you can communicate with it using a set of AT commands. AT commands are based on the Hayes Command Set.

Table 4. AT Commands

Basic	Wi-Fi layer	TCP/IP Layer
AT	AT+CWMODE	AT+CIPSTATUS
AT+RST	AT+CWJAP	AT+CIPSTART
AT+GMR	AT+CWLAP	AT+CIPSEND
AT+GSLP	AT+CWQAP	AT+CIPCLOSE
ATE	AT+CWSAP	AT+CIFSR
	AT+CWLIF	AT+CIPMUX
	AT+CWDHCP	AT+CIPSERVER
	AT+CIPSTAMAC	AT+CIPMODE
	AT+CIPAPMAC	AT+CIPSTO
	AT+CIPSTA	AT+CIUPDATE
	AT+CIPAP	+IPD

3.3. YL-69 YL-38 SOIL MOISTURE SENSOR

Soil Moisture Sensor is used to measure the volumetric water content of the soil. These sensors are used to in wide range of fields for experiments such as in soil science, environmental science, agricultural science, etc.

Dry Soil has both solid material and air pockets, which are known as pore spaces. Typically, the volumetric ration is 55:45 for solid material and pore spaces. As water is added, the pore spaces get filled with water. The maximum water content that is possible is thus 45%, at which point, all the air in the soil has been displaces by water. Soil with

45% volumetric content is known as saturated soil, since after this point, the soil can hold no more water.

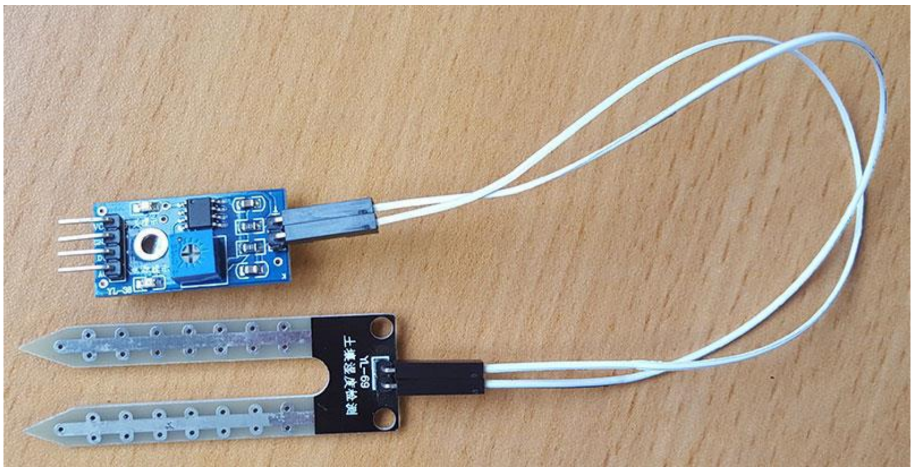


Figure 5. YL-69 YL-38 Soil Moisture Sensor

Dielectric Permittivity is the measure of resistance that is encountered while trying to form an electrical field in a dielectric material. In soil, Dielectric Permittivity is a function of the water content. The soil moisture sensor uses capacitance to measure the dielectric permittivity of the soil. The sensor creates a voltage proportional to the dielectric permittivity and therefore the water content of the soil. The reading is averaged out over the length of the sensor. There is a 2 cm zone of influence with respect to the flat surface of the sensor, but it has little or no sensitivity at the extreme edges.

The Chinese built YL-69 YL-38 sensors come with a middle man circuit which allows the user to get two types of output: an analog reading of the resistance between the probes and a digital output depending on whether the humidity is above or below the threshold, which can be adjusted using the built-in POTS. The two pins in the YL-69 sensor needs to be wired to the YL-38 Bridge. The bridge has for pins (Vcc, GND, D0 and A0).

Specifications

- Range: 0 to 45% volumetric water content in soil (capable of 0 to 100% VWC with alternate calibration)
- Accuracy: ±4% typical
- 13-bit resolution (using SensorDAQ): 0.05%

- 12-bit resolution: 0.1%
- 10-bit resolution (using CBL 2): 0.4%
- Power: 3 mA at 5V DC
- Operating temperature: –40 C to +60 C
- Dimensions: 8.9 cm × 1.8 cm × 0.7 cm (active sensor length 5 cm)
- Stored calibration: slope 108%/volt intercept –42%

3.4. 4 CHANNEL LEVEL SHIFTER

This module is used to connect two devices of different digital voltage levels. For example: connecting an Arduino (5V device) to an ESP8266 (3.3V device).

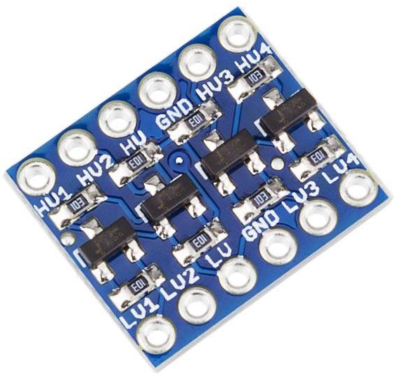


Figure 6. 4 Channel Level Shifter

A 4 Channel Level Shifter requires supply with power from both the higher voltage level and the lower voltage level. Connect the higher voltage source to pin "HV" and its ground to pin "GND" near the "HV" pin. Connect the lower voltage source to pin "LV" and its ground to pin "GND" near the "LV" pin as well. For example, if Arduino (5V signals) is connected to an ESP8266 (3.3V signals), "HV" pin on the module will be connect to the 5V supply pin on the Arduino and the "GND" pin near the "HV" pin to the "GND" pin on the Arduino. Then, the "LV" pin on the module will be connected to the 3.3V power source that is supplying power to the ESP8266 and the "GND" pin near the "LV" pin should be connected to the 3.3V power supply's ground. Then connect the signal pin from the Arduino to one of the signal pins (e.g. pin "HV1"), and then connect the desired signal pin on the ESP8266 to the corresponding pin ("LV1" if the higher voltage signal is connected to "HV1").

Specifications:

- Convert 4 pins on the high side to 4 pins on the low side or vice versa
- Power from two voltage sources (high and low); high voltage goes to the HV pin and low voltage to the LV pin
- Pinout labelled on modules
- I2C signal capable
- Breadboard compatible
- Dimensions (excluding pins): 15.2mm (0.6") length x 12.7mm (0.5") width x 2.5mm (0.1") height
- Weight: 1.2g (0.04oz)

3.5. LM2596 VOLTAGE REGULATOR

The LM2596 series of regulators are monolithic integrated circuits that provide all the active functions for a step-down (buck) switching regulator, capable of driving a 3A load with excellent line and load regulation. These devices are available in fixed output voltages of 3.3V, 5V, 12V, and an adjustable output version.

Requiring a minimum number of external components, these regulators are simple to use and include internal frequency compensation, and a fixed-frequency oscillator. LM2596 series operates at a switching frequency of 150 KHz, allowing small sized filter components with lower frequency switching regulators.

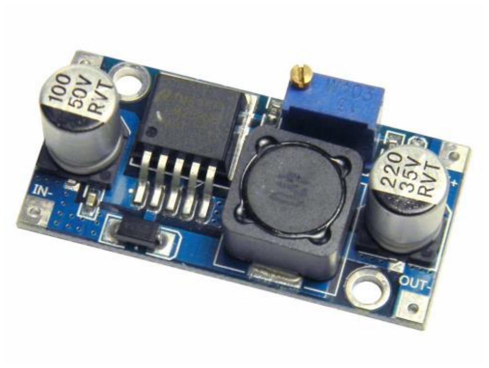


Figure 7. LM2596 Voltage Regulator

A standard series of inductors are available from several different manufacturers optimized for use with the LM2596 series. This feature greatly simplifies the design of switch-mode power supplies.

Other features include an ensured ±4% tolerance on output voltage under specified input voltage and output load conditions, and ±15% on the oscillator frequency. External shutdown is included, featuring typically 80µA standby current. Self-protection features include a two-stage frequency reducing current limit for the output switch and an over temperature shutdown for complete protection under fault conditions.

3.6. 16x2 LCD DISPLAY

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special   even custom characters (unlike in seven segments), animations and so on.

A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. 16 Characters x 2 Lines Built-in HD44780 Equivalent LCD Controller Works directly with ATMEGA, ARDUINO, PIC ARM and 8051 many other microcontroller/kits.4 or 8-bit data I/O interface Low power consumption Datasheet available on the Internet.

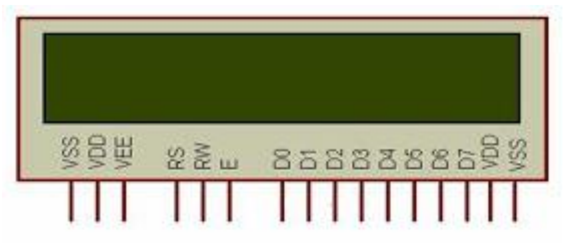


Figure 8. 16x2 LCD Display Pin layout

This LCD has two registers, namely, Command and Data. Command register is used to insert a special command into the LCD. While data register is used to insert a data into the lcd. Command is a special set of data which is used to give internal command to lcd. Like clear screen, move to line 1character, setting up the cursor etc.

3.7. SOLENOID VALVE

A solenoid valve is an electromechanically operated valve. The valve is controlled by an electric current through a solenoid: in the case of a two-port valve the flow is switched on or off; in the case of a three-port valve, the outflow is switched between the two outlet ports. Multiple solenoid valves can be placed together on a manifold.

Solenoid valves are the most frequently used control elements in fluidics. They can stop, release, distribute or mix fluids. Solenoid have safe and fast switching and good compatibility of materials used, low control power and compact design. They are highly reliable and offer a long service life.

A 2-way valve, for example, has 2 ports; if the valve is open, then the two ports are connected and fluid may flow between the ports; if the valve is closed, then ports are isolated. If the valve is open when the solenoid is not energized, then the valve is termed normally open (N.O.). Similarly, if the valve is closed when the solenoid is not energized, then the valve is termed normally closed.

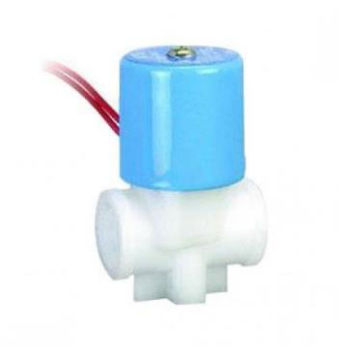


Figure 9. Solenoid Valve

The valve shown above is a normally-closed, direct-acting valve. This type of solenoid valve has the most simple and easy to understand principle of operation. Media enters the solenoid valve through the inlet port. It must flow the through orifice before reaching the outlet port. The orifice can be closed and opened by the plunger. There are two types of solenoid valves: Normally closed valves have a spring which makes the plunger close the opening of the orifice. Sealing material at the tip keeps the media from entering, until the plunger is lifted to electromagnetic field created by the coil.

3.8. SUBMERSIBLE PUMP

A submersible pump (or sub pump, electric submersible pump) is a device which has a hermetically sealed motor close-coupled to the pump body. The whole assembly is submerged in the fluid to be pumped. The main advantage of this type of pump is that it prevents pump cavitation's, a problem associated with a high elevation difference between pump and the fluid surface. Small DC Submersible water pumps push fluid to the surface as opposed to jet pumps having to pull fluids. Submersibles are more efficient than jet pumps. It is usually operated between 3V to 12V.

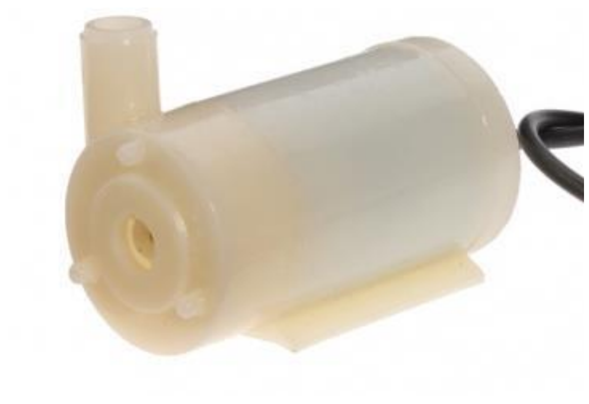


Figure 10. Submersible Pump

Specifications:

- Voltage: 2.5-10V
- Maximum lift: 40-110cm / 15.75"-43.4"
- Flow rate: 80-120L/H
- Outside diameter: 7.5mm / 0.3"
- Inside diameter: 5mm / 0.2"
- Diameter: Approx. 24mm / 0.95"
- Length: Approx. 45mm / 1.8"
- Height: Approx. 30mm / 1.2"
- Material: Engineering plastic
- Driving mode: DC design, magnetic driving
- Continuous working life for 500 hours

3.9. RELAY SWITCH

Relay is an electromagnetic device which is used to isolate two circuits electrically and connect them magnetically. They are very useful devices and allow one circuit to switch another one while they are completely separate. They are often used to interface an electronic circuit (working at a low voltage) to an electrical circuit which works at very high voltage. For example, a relay can make a 5V DC battery circuit to switch a 230V AC mains circuit. Thus, a small sensor circuit can drive, say, a fan or an electric bulb.

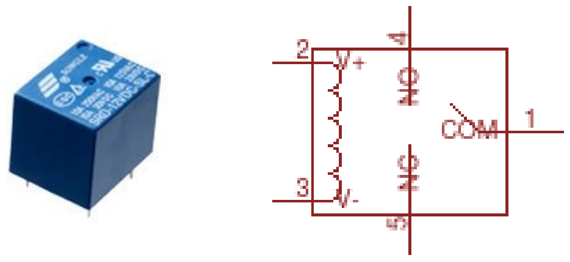


Figure 11. Relay Switch and it's pin layout

A relay switch can be divided into two parts: input and output. The input section has a coil which generates magnetic field when a small voltage from an electronic circuit is applied to it. This voltage is called the operating voltage. Commonly used relays are available in different configuration of operating voltages like 6V, 9V, 12V, 24V etc. The output section consists of contactors which connect or disconnect mechanically. In a basic relay there are three contactors: normally open (NO), normally closed (NC) and common (COM). At no input state, the COM is connected to NC. When the operating voltage is applied, the relay coil gets energized and the COM changes contact to NO.

3.10. DS18B20 DIGITAL THERMOMETER

The DS18B20 Digital Thermometer provides 9 to 12-bit (configurable) temperature readings which indicate the temperature of the device. Information is sent to/from the DS18B20 over a 1-Wire interface, so that only one wire (and ground) needs to be connected from a central microprocessor to a DS18B20. Power for reading, writing, and performing temperature conversions can be derived from the data line itself with no need for an external power source.

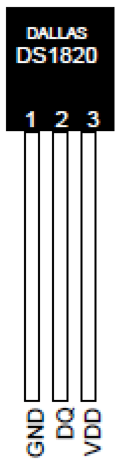


Figure 12. DSB18B20

Because each DS18B20 contains a unique silicon serial number, multiple DS18B20s can exist on the same 1-Wire bus. This allows for placing temperature sensors in many different places. Applications where this feature is useful include HVAC environmental controls, sensing temperatures inside buildings, equipment or machinery, and process monitoring and control.

Features:

- Unique 1-Wire interface requires only one port pin for communication.
- Multidrop capability simplifies distributed temperature sensing applications.
- Requires no external components.
- Can be powered from data line. Power supply range is 3V to 5.5V.
- Zero standby power required.
- Measures temperatures from -55°C to +125°C.
- ±0.5°C accuracy from -10°C to +85°C.
- Thermometer resolution is programmable from 9 to 12 bits.
- Converts 12-bit temperature to digital word in 750 ms (max.).
- User-definable, non-volatile temperature alarm settings.
- Alarm search command identifies and addresses devices whose temperature is outside of programmed limits (temperature alarm condition).
- Applications include thermostatic controls, industrial systems, consumer products, thermometers, or any thermally sensitive system.

CHAPTER 4
IMPLEMENTATION

4.1. INITIAL SETUP

1. Install Arduino IDE Software
2. Go to File > Preferences > Arduino Boards Manager URLs: http://arduino.esp8266.com/stable/package_esp8266com_index.json > OK
3. Go to Tools > Boards > Boards Manager > Download the “esp8266 by ESP8266 Community version 3.0”
4. Go to Tools > Boards > Generic ESP8266 Module
5. Go to Tools > Upload Speed > 115200; Port > choose preferred COM ports

4.2. FLASH ESP8266-12E WI-FI MODULE

1. Make the following connections from Arduino Uno to the Wi-Fi Module

Table 5. Connections between Arduino Uno and ESP8266

Arduino	ESP8266-12
5v	5v
3.3v	3.3v, CH_PD
GND	GND (both)
Tx	Tx
Rx	Rx

   Program code is directly uploaded into the ESP8266 Module. Arduino Board uses a Flash Burner for this purpose

2. Reset the ESP8266-12 Wi-Fi Module by pressing the Reset button. It can also be reset by connecting and disconnecting RESET pin to GND -> 3.3v -> GND.
3. Connect the GPIO 0 -> GND while uploading the code.

4. Once upload is successful, disconnect GPIO 0 from GND.

4.3. CIRCUIT CONNECTIONS

1. Make the following connections on the breadboard and verify results.

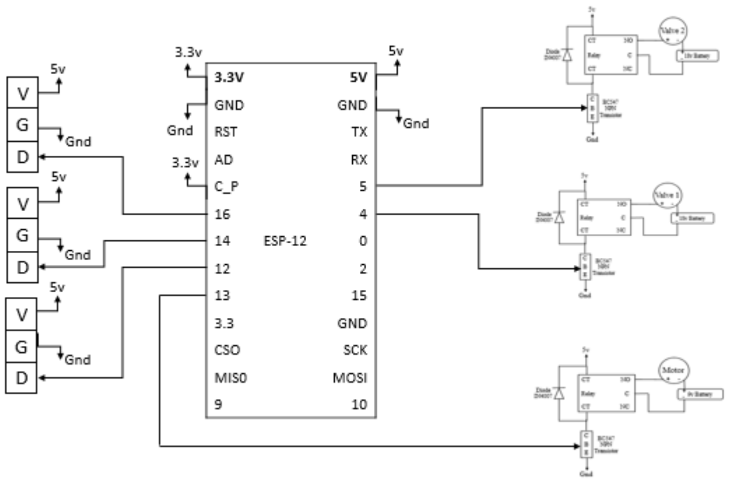


Figure 13. Circuit connections between Wi-Fi Module, Relay Switches and Sensors

2. PCB Layout is shown below

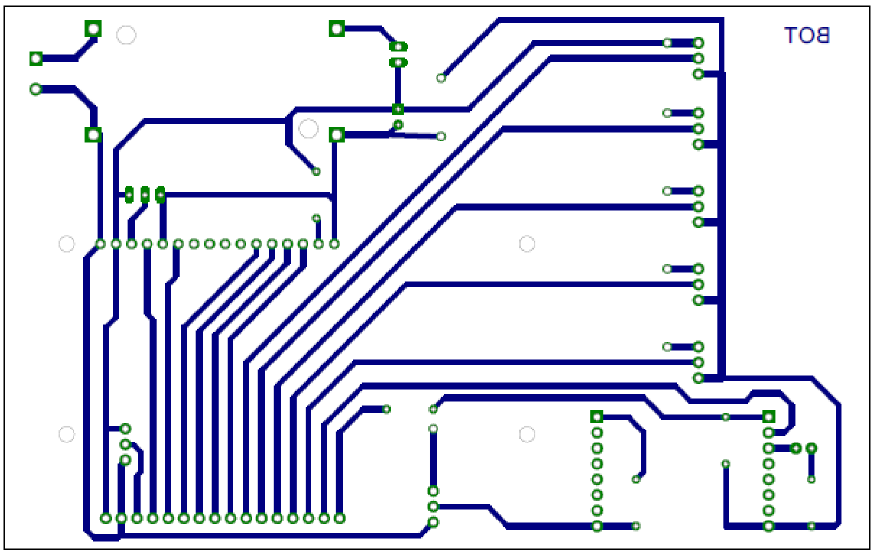


Figure 14. PCB Layout of the board

3. Print the PCB Layout onto a copper plate.
4. Immerse the copper plate in a CuSO4 solution until all the copper oxidizes, leaving only the PCB traces.
5. Carefully solder the components.
6. Complete the connections, i.e., connect the soil moisture sensors, water pump, solenoid valves, DS18B20 Temperature Sensor etc.
7. Connect the power supplies

4.4. PHYSICAL CONNECTIONS

Physical connections include the placing the sensors and actuators in a small-scale agriculture field and includes proper connections.

1. The field includes 3 regions: Two small farms and water reservoir.
2. Submerse the water pump in the water reservoir.
3. Place Valve 1 in Small Farm 1.
4. Place Valve 2 in Small Farm 2.
5. Make proper pipeline connections between pump and valves
6. Place the Soil Moisture Sensor 1 in Small Farm 1.
7. Place the Soil Moisture Sensor 2 in Small Farm 2.
8. Place the Temperature Sensor just above the Fields to measure the temperature of the area.
9. Supply the require voltage.

4.5. WORKING OF THE PROJECT

1. Wi-Fi Module connects to Internet using Mobile Hotspot
2. The status of Soil Moisture Sensor 1 is checked.
3. The status of Soil Moisture Sensor 2 is checked.

4. The status of Temperature Sensor is checked.
5. The readings are sent to the cloud server and shown on the app.
6. Depending on the readings, the user can turn on the pump and the required valves.

4.6. MQTT DASH APP SETUP

Step 1:

Initial Step to add a Server

- Name – “Iot irrigation”
- Address – “m23.cloudmqtt.com”
- Port – 14691
- Username – “zvurjfan”
- Password – “”
- ClientID – “mqttdash-0e7934c6”
- Tile Size – medium
- Metrics columns count for Vertical Position – 3
- Metrics columns count for Horizontal Position – 3

Step 2:

Insert switching operation by clicking “Add Option”

1. Motor
   - Name – “Motor”
   - Topic – “/motor”
   - Value 1 for turning motor ON
   - Value 0 for turning motor OFF

2. Valve 1
   - Name – “Valve1”
   - Topic – “/relay1”
   - Value 1 for turning valve ON

- Value 0 for turning valve OFF

3. Valve 2
    - Name – “Valve2”
    - Topic – “/relay2”
    - Value 1 for turning ON
    - Value 0 for turning OFF

Step 3:

The identity at second row and column could be temperature.

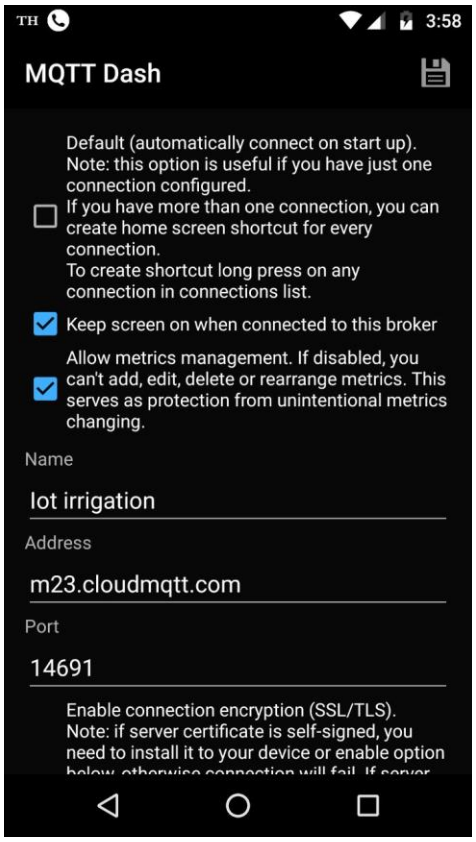
- Name – “Temp”
- Topic – “/temp”
- Minimum Value for temperature – 0
- Maximum Value for temperature – 100

For Moisture1, press “Add”.

- Name – “Moisture1”
- Topic – “/moist1”
- Minimum Value – 0
- Maximum Value – 100

For Moisture2, press “Add” once more.

- Name – “Moisture2”
- Topic – “/moist2”
- Minimum Value – 0
- Maximum Value - 100


TH 3:58

MQTT Dash

☐ Default (automatically connect on start up). Note: this option is useful if you have just one connection configured. If you have more than one connection, you can create home screen shortcut for every connection. To create shortcut long press on any connection in connections list.

☑ Keep screen on when connected to this broker

☑ Allow metrics management. If disabled, you can't add, edit, delete or rearrange metrics. This serves as protection from unintentional metrics changing.

Name

Iot irrigation

Address

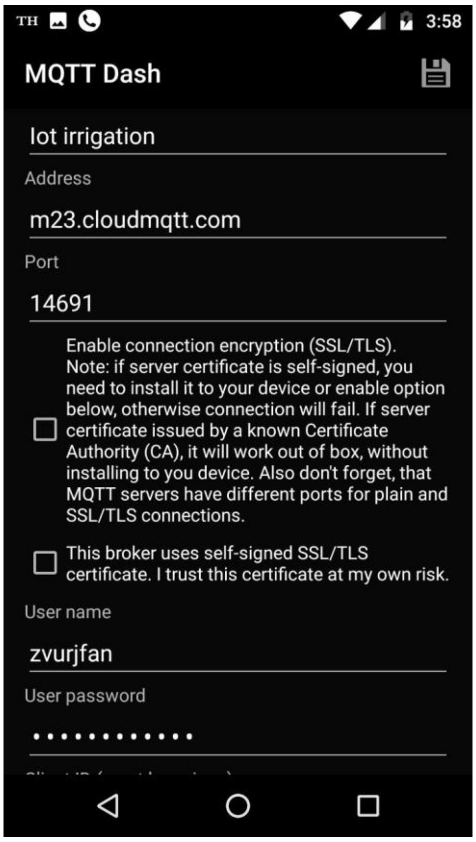
m23.cloudmqtt.com

Port

14691

Enable connection encryption (SSL/TLS). Note: if server certificate is self-signed, you need to install it to your device or enable option

Figure 15. Name, Address and Port of the MQTT Server


TH 3:58

MQTT Dash

Iot irrigation

Address

m23.cloudmqtt.com

Port

14691

☐ Enable connection encryption (SSL/TLS). Note: if server certificate is self-signed, you need to install it to your device or enable option below, otherwise connection will fail. If server certificate issued by a known Certificate Authority (CA), it will work out of box, without installing to you device. Also don't forget, that MQTT servers have different ports for plain and SSL/TLS connections.

☐ This broker uses self-signed SSL/TLS certificate. I trust this certificate at my own risk.

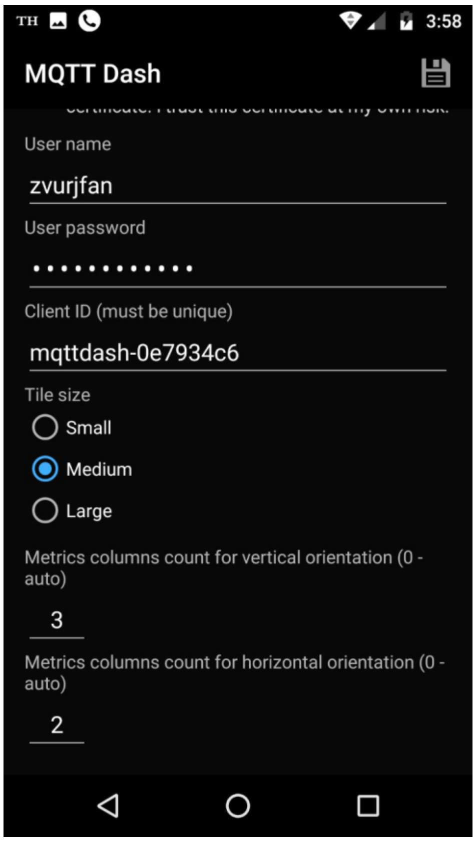
User name

zvurjfan

User password

••••••••••••

Figure 16. Username and Password for MQTT Server


TH 3:58

MQTT Dash

User name

zvurjfan

User password

•••••••••••••

Client ID (must be unique)

mqttdash-0e7934c6

Tile size

○ Small

● Medium

○ Large

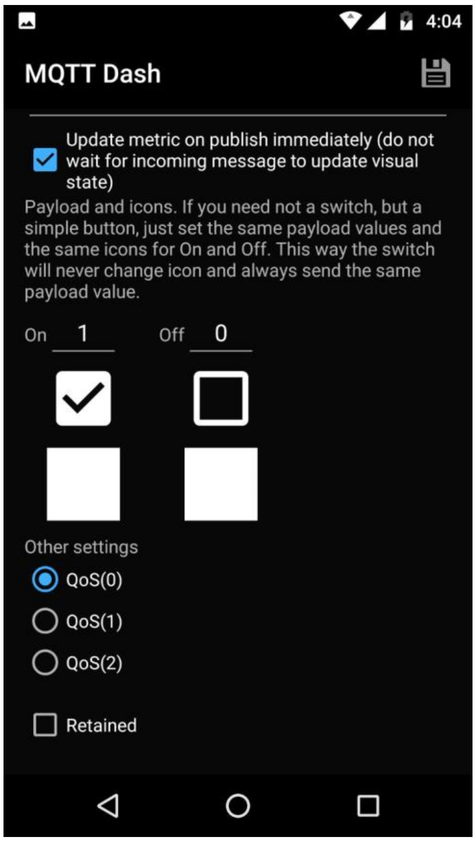
Metrics columns count for vertical orientation (0 - auto)

3

Metrics columns count for horizontal orientation (0 - auto)

2

Figure 17. App icon orientation


4:04

MQTT Dash

☑ Update metric on publish immediately (do not wait for incoming message to update visual state)

Payload and icons. If you need not a switch, but a simple button, just set the same payload values and the same icons for On and Off. This way the switch will never change icon and always send the same payload value.

On 1 Off 0

Other settings

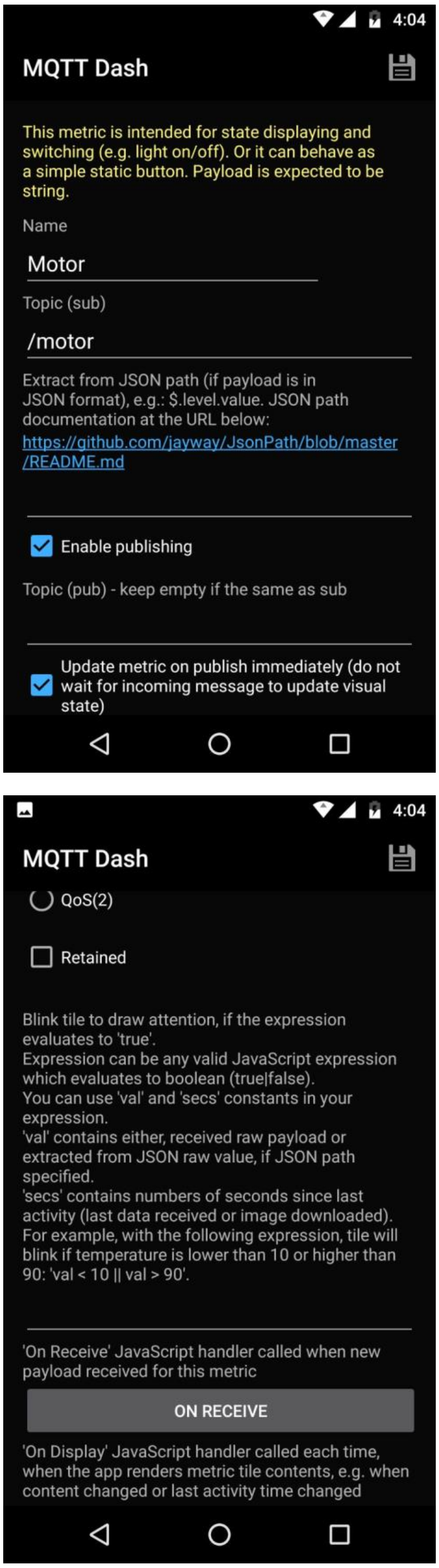
● QoS(0)

○ QoS(1)

○ QoS(2)

☐ Retained

Figure 18. ON and OFF levels


4:04

MQTT Dash

This metric is intended for state displaying and switching (e.g. light on/off). Or it can behave as a simple static button. Payload is expected to be string.

Name

Motor

Topic (sub)

/motor

Extract from JSON path (if payload is in JSON format), e.g.: $.level.value. JSON path documentation at the URL below:
https://github.com/jayway/JsonPath/blob/master/README.md

[✓] Enable publishing

Topic (pub) - keep empty if the same as sub

[✓] Update metric on publish immediately (do not wait for incoming message to update visual state)

4:04

MQTT Dash

QoS(2)

Retained

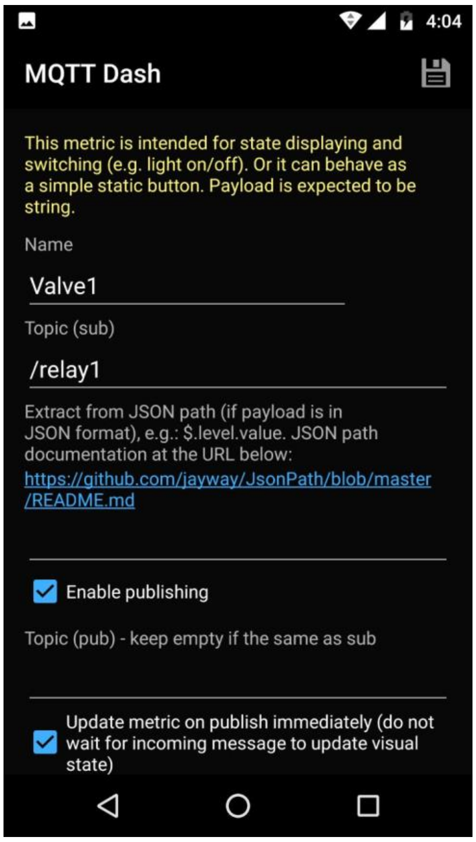
Blink tile to draw attention, if the expression evaluates to 'true'.
Expression can be any valid JavaScript expression which evaluates to boolean (true|false).
You can use 'val' and 'secs' constants in your expression.
'val' contains either, received raw payload or extracted from JSON raw value, if JSON path specified.
'secs' contains numbers of seconds since last activity (last data received or image downloaded).
For example, with the following expression, tile will blink if temperature is lower than 10 or higher than 90: 'val < 10 || val > 90'.

'On Receive' JavaScript handler called when new payload received for this metric

ON RECEIVE

'On Display' JavaScript handler called each time, when the app renders metric tile contents, e.g. when content changed or last activity time changed

Figure 19. Configuring the 'Motor' Button


4:04

MQTT Dash

This metric is intended for state displaying and switching (e.g. light on/off). Or it can behave as a simple static button. Payload is expected to be string.

Name

Valve1

Topic (sub)

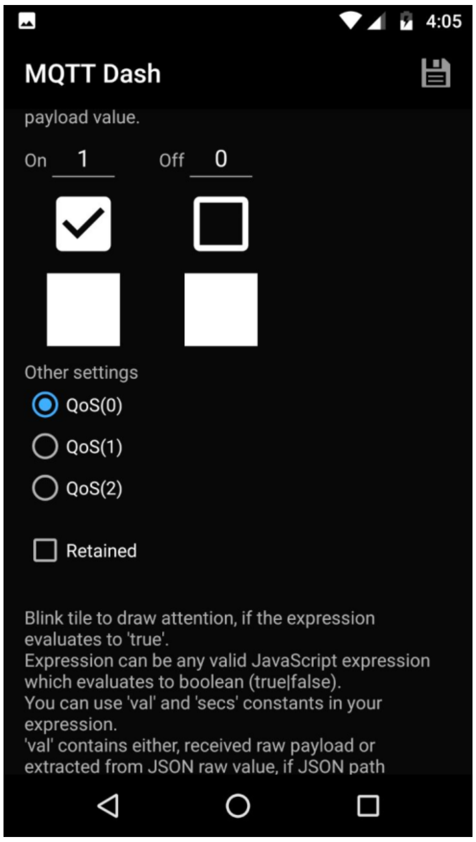
/relay1

Extract from JSON path (if payload is in JSON format), e.g.: $.level.value. JSON path documentation at the URL below:
https://github.com/jayway/JsonPath/blob/master/README.md

[✓] Enable publishing

Topic (pub) - keep empty if the same as sub

[✓] Update metric on publish immediately (do not wait for incoming message to update visual state)


4:05

MQTT Dash

payload value.

On 1   Off 0

Other settings

[✓ QoS(0)]

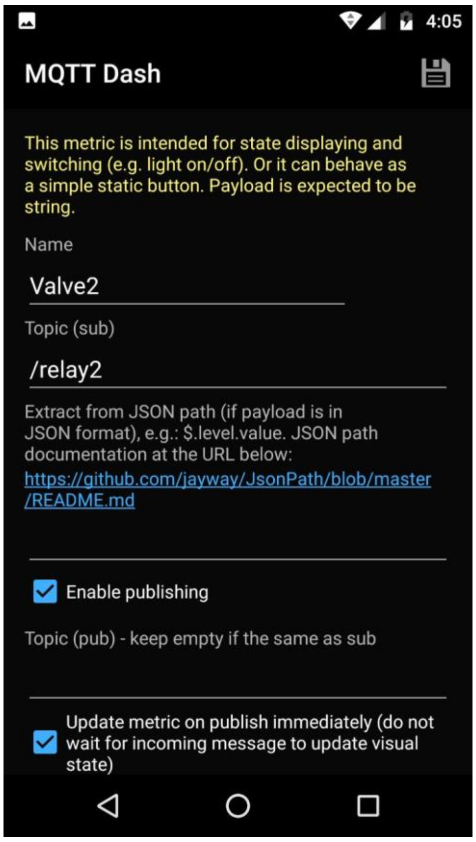
QoS(1)

QoS(2)

[ ] Retained

Blink tile to draw attention, if the expression evaluates to 'true'.
Expression can be any valid JavaScript expression which evaluates to boolean (true|false).
You can use 'val' and 'secs' constants in your expression.
'val' contains either, received raw payload or extracted from JSON raw value, if JSON path

Figure 20. Configuring the ‘Valve 1’ Button



4:05

MQTT Dash

This metric is intended for state displaying and switching (e.g. light on/off). Or it can behave as a simple static button. Payload is expected to be string.

Name

Valve2

Topic (sub)

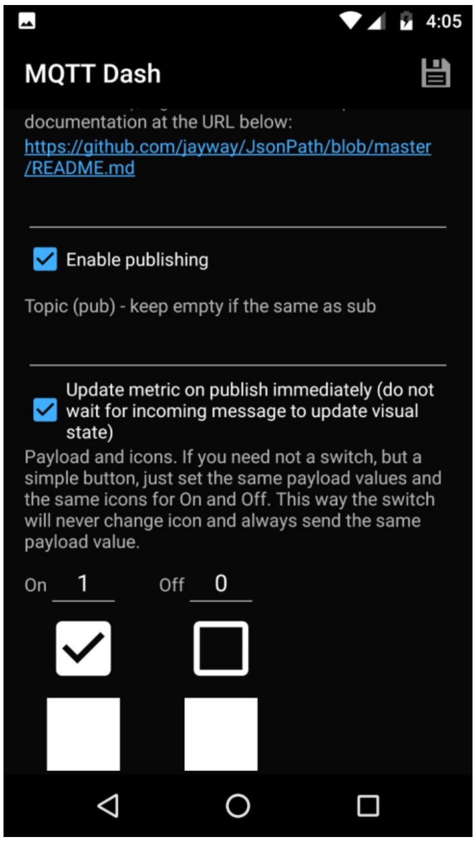
/relay2

Extract from JSON path (if payload is in JSON format), e.g.: $.level.value. JSON path documentation at the URL below:
https://github.com/jayway/JsonPath/blob/master/README.md

✓ Enable publishing

Topic (pub) - keep empty if the same as sub

✓ Update metric on publish immediately (do not wait for incoming message to update visual state)



4:05

MQTT Dash

documentation at the URL below:
https://github.com/jayway/JsonPath/blob/master/README.md

✓ Enable publishing

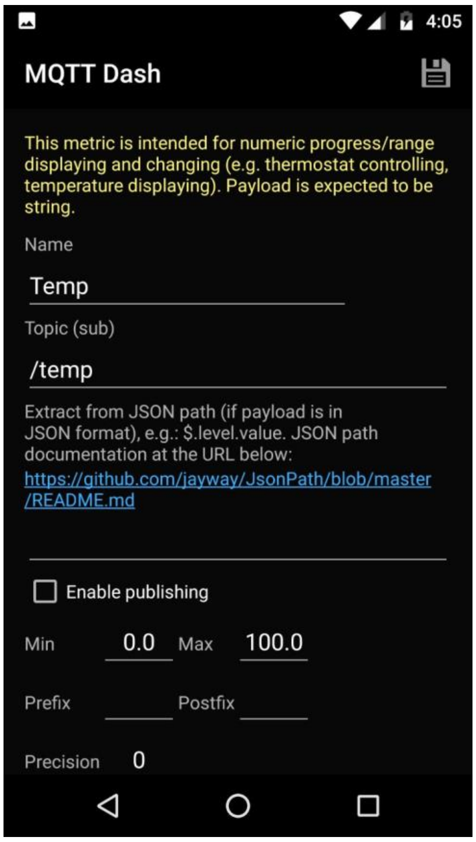
Topic (pub) - keep empty if the same as sub

✓ Update metric on publish immediately (do not wait for incoming message to update visual state)

Payload and icons. If you need not a switch, but a simple button, just set the same payload values and the same icons for On and Off. This way the switch will never change icon and always send the same payload value.

On 1 Off 0

Figure 21. Configuring the ‘Valve 2’ Button


MQTT Dash

This metric is intended for numeric progress/range displaying and changing (e.g. thermostat controlling, temperature displaying). Payload is expected to be string.

Name: Temp

Topic (sub): /temp

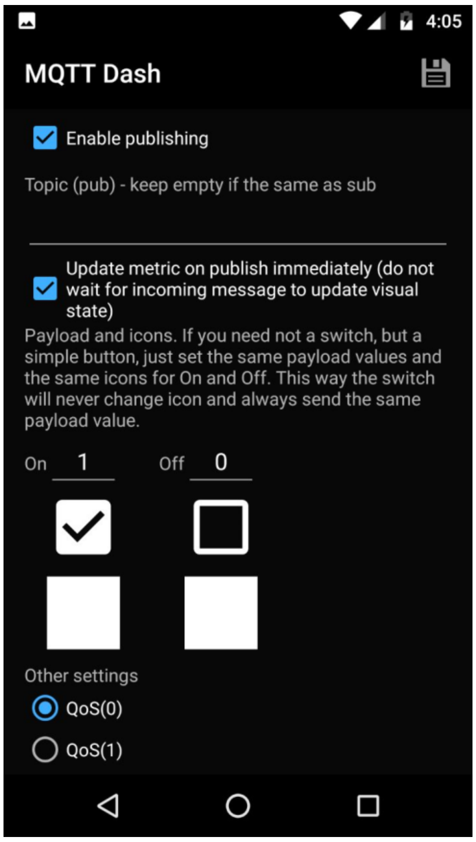
Extract from JSON path (if payload is in JSON format), e.g.: $.level.value. JSON path documentation at the URL below:
https://github.com/jayway/JsonPath/blob/master/README.md

☐ Enable publishing

Min 0.0 Max 100.0

Prefix Postfix

Precision 0


MQTT Dash

☑ Enable publishing

Topic (pub) - keep empty if the same as sub

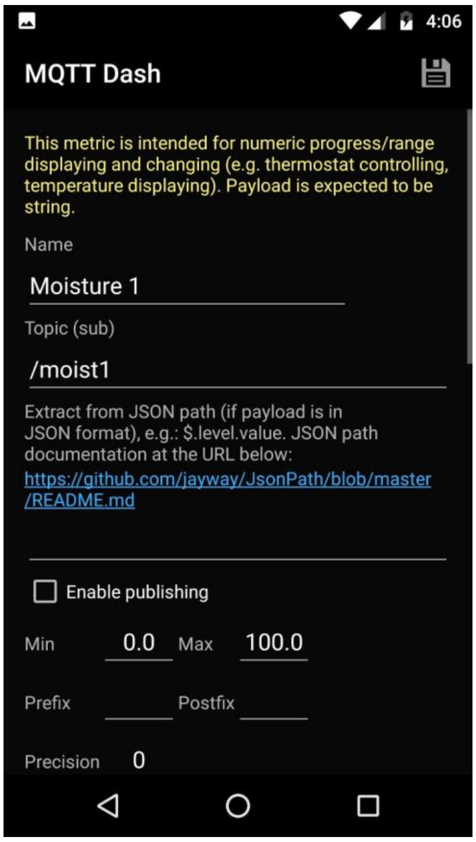
☑ Update metric on publish immediately (do not wait for incoming message to update visual state)

Payload and icons. If you need not a switch, but a simple button, just set the same payload values and the same icons for On and Off. This way the switch will never change icon and always send the same payload value.

On 1 Off 0

Other settings
● QoS(0)
○ QoS(1)

Figure 22. Configuring the Temperature Indicator



4:06

MQTT Dash

This metric is intended for numeric progress/range displaying and changing (e.g. thermostat controlling, temperature displaying). Payload is expected to be string.

Name: Moisture 1

Topic (sub): /moist1

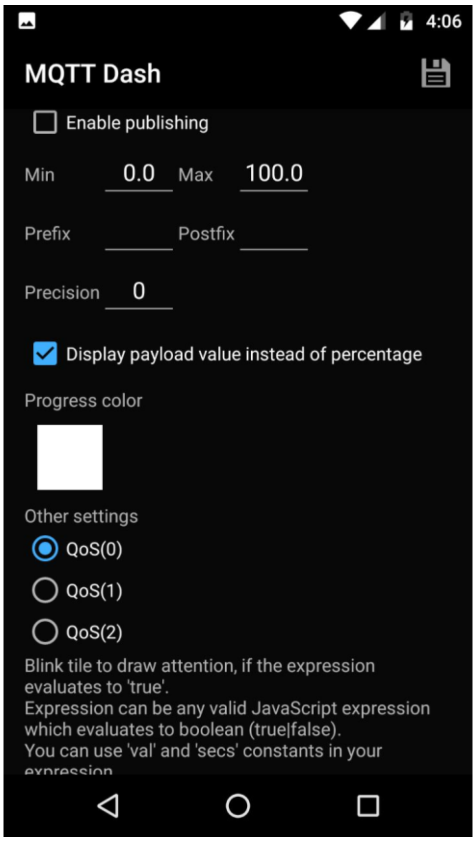
Extract from JSON path (if payload is in JSON format), e.g.: $.level.value. JSON path documentation at the URL below:
https://github.com/jayway/JsonPath/blob/master/README.md

☐ Enable publishing

Min 0.0 Max 100.0

Prefix Postfix

Precision 0



4:06

MQTT Dash

☐ Enable publishing

Min 0.0 Max 100.0

Prefix Postfix

Precision 0

☑ Display payload value instead of percentage

Progress color

⬜

Other settings

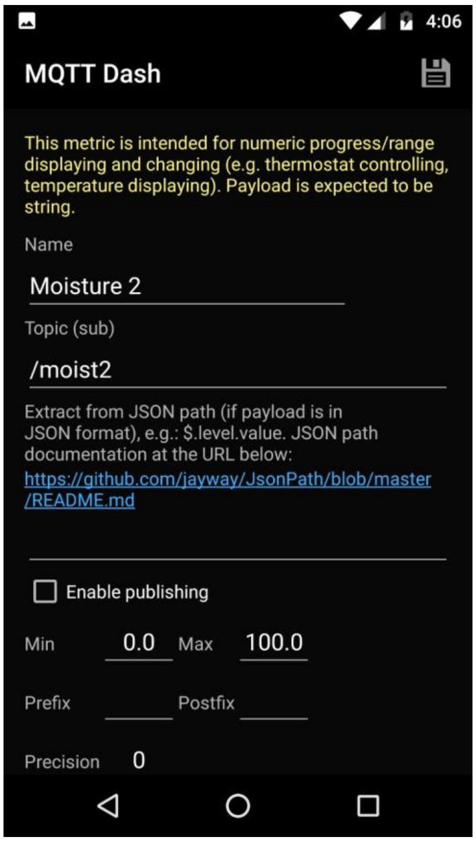
● QoS(0)

○ QoS(1)

○ QoS(2)

Blink tile to draw attention, if the expression evaluates to 'true'.
Expression can be any valid JavaScript expression which evaluates to boolean (true|false).
You can use 'val' and 'secs' constants in your expression

Figure 23. Configuring the Moisture Sensor 1


MQTT Dash

This metric is intended for numeric progress/range displaying and changing (e.g. thermostat controlling, temperature displaying). Payload is expected to be string.

Name: Moisture 2

Topic (sub): /moist2

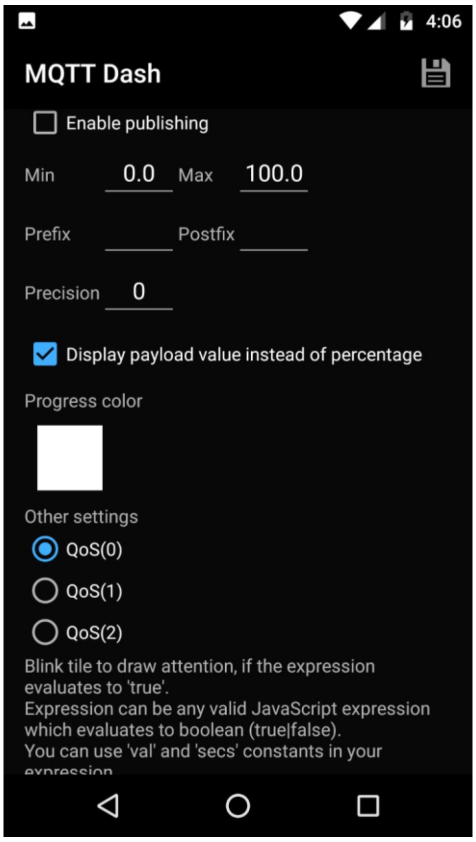
Extract from JSON path (if payload is in JSON format), e.g.: $.level.value. JSON path documentation at the URL below:
https://github.com/jayway/JsonPath/blob/master/README.md

☐ Enable publishing

Min 0.0 Max 100.0

Prefix Postfix

Precision 0


MQTT Dash

☐ Enable publishing

Min 0.0 Max 100.0

Prefix Postfix

Precision 0

☑ Display payload value instead of percentage

Progress color

Other settings

◉ QoS(0)

○ QoS(1)

○ QoS(2)

Blink tile to draw attention, if the expression evaluates to 'true'.
Expression can be any valid JavaScript expression which evaluates to boolean (true|false).
You can use 'val' and 'secs' constants in your expression

Figure 24. Configuring the Moisture Sensor 2

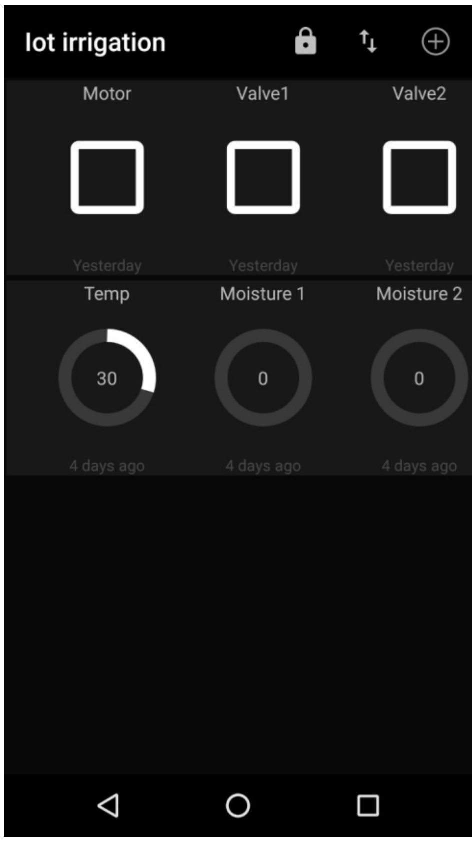


Figure 25. Final Layout of app

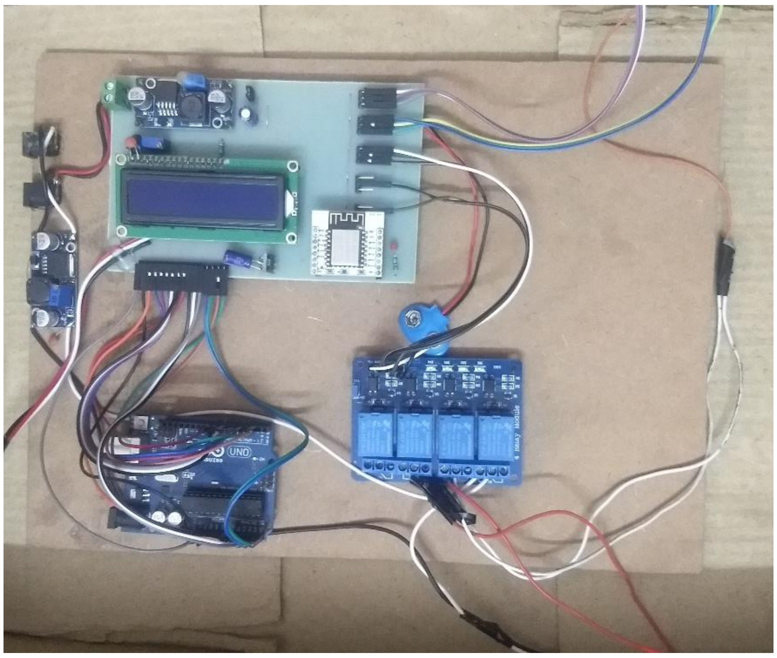


Figure 26. Final circuitry

CHAPTER 5
REFERENCES

1. P. Gagge, U. Kaware and V. Pandit, "Irrigation Monitoring and Controller System Using Internet of Things", International Journal of Electronics, Communication & Soft Computing Science and Engineering, 2017.
2. "IoT Explained — How Does an IoT System Actually Work?", Medium, 2018. [Online]. Available: https://medium.com/iotforall/iot-explained-how-does-an-iot-system-actually-work-e90e2c435fe7. [Accessed: 23- May- 2018].
3. "Why it is called Internet of Things: Definition, history, disambiguation", Iot-analytics.com, 2014. [Online]. Available: https://iot-analytics.com/internet-of-things-definition/. [Accessed: 23- May- 2018].
4. "FAQ - Frequently Asked Questions | MQTT", Mqtt.org. [Online]. Available: http://mqtt.org/faq. [Accessed: 23- May- 2018].
5. "5 Things to Know About MQTT – The Protocol for Internet of Things", Ibm.com/developerworks, 2018. [Online]. Available: https://www.ibm.com/developerworks/community/blogs/5things/entry/ 5_things_to_know_about_mqtt_the_protocol_for_internet_of_things?lang=en.
6. "Arduino - Introduction", Arduino.cc. [Online]. Available: https://www.arduino.cc/en/guide/introduction.
7. "Arduino - Products", Arduino.cc. [Online]. Available: https://www.arduino.cc/en/Main/Products.
8. "What is an Arduino?", Learn.sparkfun.com. [Online]. Available: https://learn.sparkfun.com/tutorials/what-is-an-arduino.
9. "ESP8266 Overview", Espressif.com. [Online]. Available: https://www.espressif.com/en/products/hardware/esp8266ex/overview.
10. "esp8266-module-family [ESP8266 Support WIKI]", Esp8266.com. [Online]. Available: http://www.esp8266.com/wiki/doku.php?id=esp8266-module-family.
11. D. Chen, "Wireless Communication with ESP8266 | How to Make Something that Makes (almost) Anything", Fab.cba.mit.edu, 2015. [Online]. Available:

http://fab.cba.mit.edu/classes/865.15/people/dan.chen/esp8266/. [Accessed: 23-May- 2018].

12. J.E. Arnold, "Soil Moisture". NASA. Available: http://www.ghcc.msfc.nasa.gov/landprocess/lp_home.htm
13. "Using the YL-39 + YL-69 Soil Humidity Sensor with Arduino", Arduino Project Hub. [Online]. Available: https://create.arduino.cc/projecthub/nekhbet/using-the-yl-39-yl-69-soil-humidity-sensor-with-arduino-968268.
14. "Bi-Directional Logic Level Converter (4 Channel)", United77.com. [Online]. Available: http://www.united77.com/product/bi-directional-logic-level-converter-4-channel/.
15. LM2696 Datasheet. Texas Instruments. Available: http://www.ti.com/lit/ds/symlink/lm2596.pdf
16. 16x2 Display Module – Pinout and Datasheet. Circuit Digest. Available: https://circuitdigest.com/article/16x2-lcd-display-module-pinout-datasheet
17. "How A Solenoid Valve Works", Solenoidvalvesuk.com. [Online]. Available: http://www.solenoidvalvesuk.com/how-a-solenoid-valve-works.html.
18. "Submersible pump", Ipfs.io. [Online]. Available: https://ipfs.io/ipfs/QmXoypizjW3WknFiJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Submersible_pump.html.
19. DS18B20 Programmable Resolution 1-Wire Digital Thermometer. Maxim Integrated. Available: https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf